

Progress DataDirect Connect Series for ODBC User's Guide

Release 7.1.6

Copyright

© 2022 Progress Software Corporation and/or one of its subsidiaries or affiliates. All rights reserved.

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

#1 Load Balancer in Price/Performance, 360 Central, 360 Vision, Chef, Chef (and design), Chef Habitat, Chef Infra, Code Can (and design), Compliance at Velocity, Corticon, Corticon.js, DataDirect (and design), DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Defrag This, Deliver More Than Expected, DevReach (and design), Driving Network Visibility, Flowmon, Inspec, Ipswitch, iMacros, K (stylized), Kemp, Kemp (and design), Kendo UI, Kinvey, LoadMaster, MessageWay, MOVEit, NativeChat, OpenEdge, Powered by Chef, Powered by Progress, Progress, Progress Software Developers Network, SequeLink, Sitefinity (and Design), Sitefinity, Sitefinity (and design), Sitefinity Insight, SpeedScript, Stylized Design (Arrow/3D Box logo), Stylized Design (C Chef logo), Stylized Design of Samurai, TeamPulse, Telerik, Telerik (and design), Test Studio, WebSpeed, WhatsConfigured, WhatsConnected, WhatsUp, and WS_FTP are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries.

Analytics360, AppServer, BusinessEdge, Chef Automate, Chef Compliance, Chef Desktop, Chef Workstation, Corticon Rules, Data Access, DataDirect Autonomous REST Connector, DataDirect Spy, DevCraft, Fiddler, Fiddler Classic, Fiddler Everywhere, Fiddler Jam, FiddlerCap, FiddlerCore, FiddlerScript, Hybrid Data Pipeline, iMail, InstaRelinker, JustAssembly, JustDecompile, JustMock, KendoReact, OpenAccess, PASOE, Pro2, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, Push Jobs, SafeSpaceVR, Sitefinity Cloud, Sitefinity CMS, Sitefinity Digital Experience Cloud, Sitefinity Feather, Sitefinity Thunder, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, Supermarket, SupportLink, Unite UX, and WebClient are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

Please refer to the NOTICE.txt or Release Notes – Third-Party Acknowledgements file applicable to a particular Progress product/hosted service offering release for any related required third-party acknowledgements.

Updated: 2022/05/16

Table of Contents

Welcome to the Progress DataDirect Connect Series for ODBC.....	17
What's new in this release?.....	18
Product Matrix.....	30
Product Platforms.....	31
About the Documentation Library.....	32
Contacting Technical Support.....	32
Quick Start Connect.....	33
Configuring and Connecting on Windows.....	34
Setting the Library Path Environment Variable (Salesforce Driver on Windows).....	34
Configuring a Data Source.....	34
Minimum Configuration Requirements (Windows).....	35
Testing the Connection.....	40
Configuring and Connecting on UNIX and Linux.....	41
Environment Configuration.....	41
Test Loading the Driver.....	42
Setting the Library Path Environment Variable (Salesforce Driver on UNIX/Linux).....	42
Configuring a Data Source.....	43
Minimum Configuration Requirements (UNIX/Linux).....	44
Testing the Connection.....	53
Using the Performance Wizard.....	53
Starting the Wizard.....	54
Tuning Performance Using the Wizard.....	54
General Information on Using Connect Drivers.....	57
About the Product.....	57
Support for Multiple Environments.....	58
Environment-Specific Information.....	58
For Windows Users.....	58
For UNIX and Linux Users.....	60
Using IP Addresses.....	67
Binding Parameter Markers.....	69
Driver Threading Information.....	69
Version String Information.....	70
getFileVersionString Function.....	72
Retrieving Data Type Information.....	72
Persisting a Result Set as an XML Data File.....	73
Using the Windows XML Persistence Demo Tool.....	74

Using the UNIX/Linux XML Persistence Demo Tool.....	75
Translators.....	75

Advanced Features.....77

Using Failover.....	78
Connection Failover.....	78
Extended Connection Failover.....	80
Select Connection Failover.....	81
Guidelines for Primary and Alternate Servers.....	82
Using Client Load Balancing	83
Using Connection Retry.....	84
Summary of Failover-Related Options.....	84
Using Client Information.....	87
How Databases Store Client Information.....	87
Storing Client Information.....	87
Using Security.....	89
Authentication.....	89
Data Encryption Across the Network.....	91
SSL Encryption.....	92
Using DataDirect Connection Pooling.....	97
Creating a Connection Pool.....	98
Adding Connections to a Pool.....	98
Removing Connections from a Pool.....	99
Handling Dead Connections in a Pool.....	99
Connection Pool Statistics.....	100
Summary of Pooling-Related Options.....	100
Using DataDirect Bulk Load.....	101
Bulk Export and Load Methods.....	102
Exporting Data from a Database.....	102
Bulk Loading to a Database.....	104
The Bulk Load Configuration File.....	105
Sample Applications.....	107
Character Set Conversions.....	108
External Overflow Files.....	108
Using Bulk Load for Single Inserts/Updates/Deletes (Salesforce Driver).....	108
Summary of Related options for DataDirect Bulk Load.....	109
Using Bulk Load for Batch Inserts.....	109
Determining the Bulk Load Protocol.....	110
Summary of Related Options for Bulk Load or Batch Inserts.....	110

Configuring the Product on UNIX/Linux.....111

Environment Variables.....	112
Library Search Path.....	112

ODBCINI.....	112
ODBCINST.....	113
DD_INSTALLDIR.....	113
The Test Loading Tool.....	114
Data Source Configuration.....	114
Configuration Through the Administrator.....	115
Configuration Through the System Information (odbc.ini) File.....	117
The demoodbc Application.....	132
The example Application.....	132
DSN-less Connections.....	132
Sample odbcinst.ini File.....	133
File Data Sources.....	136
UTF-16 Applications on UNIX and Linux.....	137

Drivers for 32-Bit and 64-Bit Platforms.....139

The DB2 Wire Protocol Driver.....	140
Driver Requirements.....	140
Binding.....	140
Configuring and Connecting to Data Sources.....	142
Connection Option Descriptions for DB2.....	160
Performance Considerations.....	212
IBM to IANA Code Page Values.....	213
Data Types.....	213
Unicode Support.....	215
Advanced Features.....	215
Cursor Stability Isolation Level.....	216
XQuery Expressions.....	217
Stored Procedure Support.....	217
Unexpected Characters.....	217
Support for DB2 pureScale.....	218
Persisting a Result Set as an XML Data File.....	218
Isolation and Lock Levels Supported.....	219
SQL Support.....	219
ODBC Conformance Level.....	219
Number of Connections and Statements Supported.....	219
Using Arrays of Parameters.....	219
The Informix Wire Protocol Driver.....	220
Driver Requirements.....	220
Configuring and Connecting to Data Sources.....	220
Connection Option Descriptions.....	225
Performance Considerations.....	236
Data Types.....	237
Advanced Features.....	239
MTS Support.....	239

Persisting a Result Set as an XML Data File.....	239
Isolation and Lock Levels Supported.....	239
SQL Support.....	239
ODBC Conformance Level.....	239
Number of Connections and Statements Supported.....	240
The MySQL Wire Protocol Driver.....	240
Driver Requirements.....	240
Configuring and Connecting to Data Sources.....	240
Connection Options Descriptions.....	249
Performance Considerations.....	279
Data Types.....	280
Unicode Support.....	281
Advanced Features.....	282
Persisting a Result Set as an XML Data File.....	282
Isolation and Lock Levels Supported.....	282
SQL Support.....	283
ODBC Conformance Level.....	283
Number of Connections and Statements Supported.....	283
The Oracle Wire Protocol Driver.....	283
Driver Requirements.....	284
Configuring and Connecting to Data Sources.....	284
Connection Option Descriptions for Oracle Wire Protocol	303
Performance Considerations.....	367
Data Types.....	370
Unicode Support.....	373
Advanced Features.....	374
MTS Support.....	376
OS Authentication.....	376
Support for Oracle RAC.....	376
Support of Materialized Views.....	377
Stored Procedure Results.....	377
Unexpected Characters.....	377
Persisting a Result Set as an XML Data File.....	378
Isolation and Lock Levels Supported.....	379
SQL Support.....	379
ODBC Conformance Level.....	379
Number of Connections and Statements Supported.....	379
Using Parameter Arrays.....	380
The PostgreSQL Wire Protocol Driver	380
Driver Requirements.....	380
Configuring and Connecting to Data Sources.....	380
Accessing PostgreSQL data with Power BI.....	390
Connection Option Descriptions for PostgreSQL Wire Protocol.....	391
Performance Considerations.....	431
Data Types.....	432

Unicode Support.....	435
Advanced Features.....	435
User-defined Functions' Results.....	436
Persisting a Result Set as an XML Data File.....	437
Isolation and Lock Levels Supported.....	437
SQL Support.....	437
ODBC Conformance Level.....	437
Number of Connections and Statements Supported.....	438
Using Arrays of Parameters.....	438
The Progress OpenEdge Wire Protocol Driver.....	438
Driver Requirements.....	438
Configuring and Connecting to Data Sources.....	438
Connection Option Descriptions for OpenEdge Wire Protocol.....	445
Performance Considerations.....	468
Data Types.....	469
Unicode Support.....	470
Advanced Features.....	470
Isolation and Lock Levels Supported.....	471
SQL Grammar Support.....	471
ODBC Conformance Level.....	471
Number of Connections and Statements Supported.....	471
The SQL Server Wire Protocol Driver.....	471
Driver Requirements.....	472
Configuring and Connecting to Data Sources.....	472
Connection Option Descriptions for SQL Server Wire Protocol.....	486
Performance Considerations.....	533
Data Types.....	534
Unicode Support.....	535
Advanced Features.....	536
Persisting a Result Set as an XML Data File.....	539
Isolation and Lock Levels Supported.....	539
SQL Support.....	540
ODBC Conformance Level.....	540
Number of Connections and Statements Supported.....	540
Using Arrays of Parameters.....	540
Support for Azure Synapse Analytics and Analytics Platform System.....	541
The Sybase Wire Protocol Driver.....	542
Driver Requirements.....	542
Configuring and Connecting to Data Sources.....	543
Connection Option Descriptions for Sybase Wire Protocol.....	558
Performance Considerations.....	607
Data Types.....	609
Unicode Support.....	610
Advanced Features.....	611
Performance Considerations.....	613

Unexpected Characters.....	614
MTS Support.....	615
NULL Values.....	615
Persisting a Result Set as an XML Data File.....	616
Isolation and Lock Levels Supported.....	616
SQL Grammar Support.....	616
ODBC Conformance Level.....	616
Number of Connections and Statements Supported.....	616
Using Arrays of Parameters.....	617
The Oracle Driver	617
Driver Requirements.....	617
Configuring and Connecting to Data Sources.....	618
Connection Option Descriptions.....	627
Performance Considerations.....	652
Data Types.....	653
Unicode Support.....	656
Advanced Features.....	657
Unexpected Characters.....	657
MTS Support.....	658
OS Authentication.....	658
Support for Oracle RAC.....	659
Support of Materialized Views.....	659
Stored Procedure Results.....	659
Persisting a Result Set as an XML Data File.....	660
Isolation and Lock Levels Supported.....	660
SQL Support.....	660
ODBC Conformance Level.....	660
Number of Connections and Statements Supported.....	660
Using Arrays of Parameters.....	661
The SQL Server Legacy Wire Protocol Driver.....	661
Driver Requirements.....	661
Configuring and Connecting to Data Sources.....	662
Connection Option Descriptions.....	667
Performance Considerations.....	688
Data Types.....	688
Unicode Support.....	690
Advanced Features.....	690
Isolation and Lock Levels Supported.....	690
SQL Support.....	691
ODBC Conformance Level.....	691
Number of Connections and Statements Supported.....	691
Using Arrays of Parameters.....	691
The Text Driver.....	691
Driver Requirements.....	692
Formats for Text Files.....	692

Configuring Data Sources.....692

Using a Connection String.....696

Connection Option Descriptions.....697

Defining Table Structure on Windows.....710

Defining Table Structure on UNIX and Linux712

Example of QETXT.INI.....713

Date Masks.....713

Data Types.....714

Select Statement.....714

Alter Table Statement.....714

SQL Support.....715

ODBC Conformance Level.....715

Number of Connections and Statements Supported.....715

Drivers Only Available for 32-Bit Platforms.....717

The Btrieve (Pervasive.SQL) Driver.....717

 Driver Requirements.....718

 Managing Databases.....718

 Transactions.....719

 Configuring and Connecting to Data Sources (Btrieve).....719

 Connection Option Descriptions.....723

 Defining Table Structure.....731

 Data Types.....732

 Indexes.....733

 Column Names.....734

 Select Statement.....734

 Alter Table Statement.....734

 Create and Drop Index Statements.....735

 Isolation and Lock Levels Supported.....735

 SQL Support.....736

 ODBC Conformance Level.....736

 Number of Connections and Statements Supported.....736

The dBASE Driver.....736

 Driver Requirements.....736

 Configuring and Connecting to Data Sources.....737

 Connection Option Descriptions.....744

 Defining Index Attributes on Windows.....754

 Defining Index Attributes on UNIX and Linux.....755

 Data Types.....756

 Column Names.....757

 Select Statement.....757

 Alter Table Statement.....758

 Create and Drop Index Statements.....758

 Pack Statement.....760

SQL Statements for FoxPro 3.0 Database Containers.....	760
Locking.....	761
Isolation and Lock Levels Supported.....	762
SQL Support.....	762
ODBC Conformance Level.....	762
Number of Connections and Statements Supported.....	762
The Informix Driver.....	762
Driver Requirements.....	762
Configuring and Connecting to Data Sources.....	763
Connection Option Descriptions.....	769
Performance Considerations.....	780
Data Types.....	781
MTS Support.....	783
Persisting a Result Set as an XML Data File.....	783
Isolation and Lock Levels Supported.....	783
SQL Support.....	784
ODBC Conformance Level.....	784
Number of Connections and Statements Supported.....	784
The XML Driver.....	784
Driver Requirements.....	785
Supported Tabular Formats for XML Documents.....	785
Hierarchical-Formatted XML Document Support.....	786
Defining Locations.....	788
Specifying Table Names in SQL Statements.....	788
Configuring and Connecting to Data Sources (XML).....	790
Connection Option Descriptions.....	797
Using Hints for Tabular-Formatted XML Documents.....	812
Data Types.....	814
Unicode Support.....	817
Persisting a Result Set as an XML Data File.....	817
ODBC Conformance Level.....	817
Number of Connections and Statements Supported.....	817
SQL Support.....	817

The Connect XE Drivers.....827

The Greenplum Wire Protocol Driver	827
Driver Requirements.....	828
Configuring and Connecting to Data Sources.....	828
Accessing Greenplum data with Power BI.....	837
Greenplum Connection Option Descriptions.....	838
Performance Considerations.....	879
Data Types.....	879
Unicode Support.....	881
Advanced Features.....	881

User-defined Functions' Results.....	882
Persisting a Result Set as an XML Data File.....	883
Isolation and Lock Levels Supported.....	883
SQL Support.....	883
ODBC Conformance Level.....	883
Number of Connections and Statements Supported.....	884
Using Arrays of Parameters.....	884
Limitations for Pivotal HAWQ Users.....	884
The Impala Wire Protocol Driver.....	884
Driver Requirements.....	885
Configuring and Connecting to Data Sources.....	885
Connection Option Descriptions.....	892
Performance Considerations.....	918
Data Types.....	918
Advanced Features.....	919
Materialized Views.....	920
Stored Procedures.....	920
Unicode Support.....	920
Isolation and Lock Levels Supported.....	920
SQL Support.....	920
ODBC Conformance Level.....	920
Using Arrays of Parameters.....	921
Limitations on Cloudera Impala Functionality.....	921
The Salesforce Driver.....	921
Driver Requirements.....	923
Configuring and Connecting to Data Sources.....	924
Connection Option Descriptions.....	939
Performance Considerations.....	974
Data Types.....	975
Mapping Objects to Tables.....	978
Client-Side Caches.....	979
Catalog Tables.....	981
Timeouts.....	987
Views and Remote/Local Tables.....	987
Using Identifiers.....	987
Database Configuration File.....	988
Reports.....	991
Connecting Through a Proxy Server.....	991
Configuring the SQL Engine Server.....	992
Unicode Support.....	997
Advanced Features.....	997
Parameter Metadata Support.....	997
Using DataDirect Bulk Load With the Salesforce Driver.....	999
Error Handling.....	999
Isolation and Lock Levels Supported.....	1000

SQL Support.....	1000
ODBC Conformance Level.....	1000
Number of Connections and Statements Supported.....	1001
The Sybase IQ Wire Protocol Driver.....	1001
Driver Requirements.....	1001
Configuring and Connecting to Data Sources.....	1001
Connection Option Descriptions.....	1011
Performance Considerations.....	1038
Data Types.....	1039
Unicode Support.....	1041
Advanced Features.....	1041
Unexpected Characters.....	1042
NULL Values.....	1043
Persisting a Result Set as an XML Data File.....	1043
Isolation and Lock Levels Supported.....	1044
SQL Support.....	1044
ODBC Conformance Level.....	1044
Number of Connections and Statements Supported.....	1044
Using Arrays of Parameters.....	1044
The Driver for Apache Hive.....	1045
Driver Requirements.....	1045
Configuring and Connecting to Data Sources.....	1045
Connection Option Descriptions for Apache Hive.....	1052
Performance Considerations.....	1078
Data Types.....	1078
Advanced Features.....	1079
Materialized Views.....	1080
Stored Procedures	1080
Unicode Support.....	1080
Isolation and Lock Levels Supported.....	1080
SQL Support.....	1080
ODBC Conformance Level.....	1081
Using Arrays of Parameters.....	1081
Limitations on Apache Hive Functionality.....	1081
The Driver for the Teradata Database.....	1082
Driver Requirements.....	1082
Configuring and Connecting to Data Sources.....	1082
Connection Option Descriptions.....	1089
Data Types.....	1109
Unicode Support.....	1110
Persisting a Result Set as an XML Data File.....	1111
Isolation and Lock Levels Supported.....	1111
SQL Support.....	1111
ODBC Conformance Level.....	1111
Number of Connections and Statements Supported.....	1111

Supported SQL Statements and Extensions.....1113

SQL Functionality for the Driver for Apache Hive.....	1113
Data Definition Language (DDL).....	1113
Selecting Data With the Driver.....	1114
From Clause.....	1114
Group By Clause.....	1115
Having Clause	1115
Insert.....	1115
Order By Clause.....	1116
For Update Clause.....	1116
Set Operators.....	1116
Subqueries.....	1116
SQL Expressions.....	1117
Restrictions.....	1120
SQL Statements for Flat-File Drivers.....	1121
Select Statement.....	1122
Create and Drop Table Statements.....	1134
Insert Statement.....	1135
Update Statement.....	1136
Delete Statement.....	1136
Reserved Keywords.....	1137
SQL Functionality for the Impala Wire Protocol Driver.....	1137
Data Definition Language (DDL).....	1137
Selecting Data With the Driver.....	1138
From Clause.....	1138
Group By Clause.....	1139
Having Clause.....	1139
Order By Clause.....	1139
For Update Clause.....	1139
Set Operators.....	1140
Subqueries.....	1140
SQL Expressions.....	1140
Restrictions.....	1144
SQL Statements and Extensions for the Salesforce Driver	1145
Alter Cache (EXT).....	1145
Alter Index.....	1146
Alter Sequence.....	1147
Alter Session (EXT).....	1147
Alter Table.....	1149
Checkpoint.....	1154
Create Cache (EXT).....	1154
Create Index.....	1161
Create Sequence.....	1161

Create Table.....	1162
Create View.....	1173
Delete.....	1174
Drop Cache (EXT).....	1175
Drop Index.....	1176
Drop Sequence.....	1176
Drop Table.....	1177
Drop View.....	1177
Explain Plan.....	1178
Insert.....	1178
Refresh Cache (EXT).....	1180
Refresh Schema (EXT).....	1181
Select.....	1181
Set Checkpoint Defrag.....	1191
Set Logsize.....	1191
Update.....	1192
SQL Expressions.....	1193
Subqueries.....	1205

Welcome to the Progress DataDirect Connect Series for ODBC

The Progress® DataDirect Connect® Series *for* ODBC™ provides ODBC drivers for a number of leading databases, as well as flat-file database systems. The drivers are compliant with the Open Database Connectivity (ODBC) specification and compatible with ODBC 3.8 applications. The drivers are tested and supported across [numerous platforms](#). Progress DataDirect Connect Series *for* ODBC includes the following products:

- DataDirect Connect *for* ODBC
- DataDirect Connect64 *for* ODBC
- DataDirect Connect XE (Extended Edition) *for* ODBC
- DataDirect Connect64 XE *for* ODBC

Note: This guide contains information for the 7.1.6 version of the Apache Hive Wire Protocol, Oracle Wire Protocol, and SQL Server Wire Protocol drivers. For 8.0 and higher versions of those drivers, documentation is available by data source at the Progress DataDirect Connectors Documentation page: <https://docs.progress.com/bundle/datadirect-connectors/page/DataDirect-Connectors-by-data-source.html>.

The content of this guide assumes that you are familiar with your operating system and its commands. It contains the following information:

- [Quick Start Connect](#) on page 33 explains the basics for quickly configuring and testing the drivers.
- [General Information on Using Connect Drivers](#) on page 57 explains the drivers and ODBC, and discusses environment-specific subjects.
- [Advanced Features](#) on page 77 explains at a general level advanced driver features such as failover, security, connection pooling, and bulk load.

- [Configuring the Product on UNIX/Linux](#) on page 111 discusses UNIX and Linux environment variables and configuration of the drivers. It also provides a sample system information file, as well as discussing other driver tools for UNIX and Linux.
- A chapter for each database driver. Each driver's chapter is structured in the same way. First, it lists which versions of the databases the driver supports, the operating environments in which the driver runs, and the driver requirements for your operating environment. Next, it explains how to configure a data source and how to connect to that data source. Finally, the chapter provides information about data types, ODBC conformance levels, isolation and lock levels supported, and other driver-specific information.

The documentation for DataDirect Connect Series *for* ODBC drivers also includes the *Progress DataDirect for ODBC Drivers Reference*. The reference provides general reference information for all DataDirect drivers for ODBC, including content on troubleshooting, supported SQL escapes, and DataDirect tools. For the complete documentation set, visit to the Progress DataDirect Connectors Documentation Hub:

<https://docs.progress.com/bundle/datadirect-connectors/page/DataDirect-Connectors-by-data-source.html>.

If you are writing programs to access ODBC drivers, you need to obtain a copy of the *ODBC Programmer's Reference* for the Microsoft Open Database Connectivity Software Development Kit, available from Microsoft Corporation.

For the latest information about the specific drivers available for your platform, refer to the readme file in your software package.

Database drivers are continually being added to each operating environment. For the latest information about the specific drivers available for your platform, refer to the readme file in your software package, or refer to the Progress DataDirect database support matrix Web pages at the DataDirect Support Matrices page:

<https://www.progress.com/matrices/datadirect>.

Note: This guide refers the reader to Web pages using URLs for more information about specific topics, including Web URLs not maintained by Progress DataDirect. Because it is the nature of Web content to change frequently, Progress DataDirect can guarantee only that the URLs referenced in this guide were correct at the time of publication.

For details, see the following topics:

- [What's new in this release?](#)
- [Product Matrix](#)
- [Product Platforms](#)
- [About the Documentation Library](#)
- [Contacting Technical Support](#)

What's new in this release?

Visit the following web pages for the latest support and certification information.

- Release Notes: <https://www.progress.com/odbc/release-history/>
- Supported Configurations: <https://www.progress.com/supported-configurations/datadirect>
- DataDirect Support Matrices: <https://www.progress.com/matrices/datadirect>

The highlights of this release are:

- **Support for setting the value of undocumented connection options using the setup dialog for the following drivers on Windows:**

DB2 Wire Protocol	Driver for Apache Hive
MySQL Wire Protocol	Greenplum Wire Protocol
Oracle Wire Protocol	Salesforce
PostgreSQL Wire Protocol	Sybase IQ Wire Protocol
Progress OpenEdge Wire Protocol	SQL Server Wire Protocol
Sybase Wire Protocol	Oracle
Impala Wire Protocol	

- **Driver for Apache Hive enhancements since General Availability**

- The new `AllowedOpenSSLVersions` option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 1054 or [Designating an OpenSSL Library](#) on page 94 for details.
- The driver has been enhanced to support row-level inserts when connected to Hive 0.14 or higher.
- The `Batch Mechanism` connection option has been added to the driver. By setting `Batch Mechanism` to 2 (`MultiRowInsert`), you enable the driver to use a parameterized multi-row insert statement to execute batch inserts. `MultiRowInsert` is the default setting and provides substantial performance gains when performing batch inserts. See [Batch Mechanism](#) on page 1056 for details.
- The `Authentication Method` connection option has been refreshed with a new valid value for enabling Kerberos Authentication. To use Kerberos authentication with the driver, set `AuthenticationMethod=4`.

Note: The legacy setting for enabling Kerberos Authentication (`AuthenticationMethod=1`) will continue to be valid for this release; however, it will be disabled in future versions of the product.

See [Authentication Method](#) on page 1056 for details.

- Added support for SSL encryption, including the following connection options:
 - The `CryptoLibName` connection option allows you to determine the cryptographic library used when SSL is enabled. See [CryptoLibName](#) on page 1058 for details.
 - The `Encryption Method` connection option now allows you to encrypt data sent between the driver and the database. See [Encryption Method](#) on page 1062 for details.
 - The `Host Name In Certificate` connection option now allows you to specify the host name for certificate validation when SSL encryption and validation are enabled. See [Host Name In Certificate](#) on page 1064 for details.
 - The `Key Password` connection option now allows you to specify the key password that is used to access the individual keys in the keystore file when SSL and SSL client authentication are enabled on the database server. See [Key Password](#) on page 1065 for details.
 - The `Key Store` connection option now allows you to specify the directory containing the keystore file that is to be used when SSL and SSL client authentication are enabled on the database server. See [Key Store](#) on page 1065 for details.

- The Key Store Password connection option allows you to specify the keystore password that is used to access the keystore file when SSL and SSL client authentication are enabled on the database server. See [Keystore Password](#) on page 1066 for details.
- The SSLLibName connection option allows you to determine the SSL library used when SSL is enabled. See [Key Password](#) on page 1065 for details.
- The Truststore connection option now allows you to specify the directory that contains the truststore file and the truststore file name that is to be used when SSL is enabled and the server authentication is used. See [Truststore](#) on page 1074 for details.
- The User Name connection option now allows you to specify the user ID that is used to connect to your database. See [User Name](#) on page 1076 for details.
- The Validate Server Certificate connection option now determines whether the driver validates the certificates that are sent by the database server when SSL encryption is enabled. See [Validate Server Certificate](#) on page 1077 for details.
- The Array Size configuration option has been refreshed to allow specifying the number of cells retrieved instead of rows. By determining the fetch size based on the number of cells, the driver can avoid out of memory errors when fetching from tables containing a large number of columns. See [Array Size](#) on page 1055 for details.
- Certified with Apache Hive Sentry, which enables HiveServer2 administrators to enforce role-based authorization for Apache Hadoop clusters. See [Apache Sentry](#) on page 1080 for details.
- The driver has been enhanced to support the Char data type when connected to Hive 0.13 and higher. See [Data Types](#) on page 1078 for details.
- The driver has been enhanced to support the Decimal data type when connected to Hive 0.11 and higher. See [Data Types](#) on page 1078 for details.
- The driver has been enhanced to support the Date and Varchar data types in Hive 0.12 and higher. See [Data Types](#) on page 1078 for details.
- The TCP Keep Alive connection option allows you to use TCP Keep Alive to maintain idle TCP connections. See [TCP Keep Alive](#) on page 1073 for details.
- The User Name and Password connection options are now required for a connection to HiveServer2. See [User Name](#) on page 1076 and [Password](#) on page 1068 for details.
- When connection to HiveServer2, simultaneous connections per port are supported.
- The new Wire Protocol Version connection option specifies the version of the HiveServer to which the driver will connect. See [Wire Protocol Version](#) on page 1077 for details.
- The String Describe Type connection option now allows you to describe string columns as SQL_WLONGVARCHAR or SQL_WVARCHAR. See [String Describe Type](#) on page 1072 for details.
- The Use Native Catalog Function connection option allows you to specify whether the driver uses native catalog functions to retrieve information returned by the SQLTables, SQLColumns, and SQLStatistics catalog functions. See [Use Native Catalog Functions](#) on page 1076 for details.
- Added support for Kerberos authentication, including the following connection options:
 - The GSS Client Library connection option now allows you to specify the name of the GSS client library that the driver uses to communicate with the Key Distribution center (KDC). See [GSS Client Library](#) on page 1062 for details.
 - The Authentication Method connection option now allows you to specify the method the driver uses to authenticate the user to the server when a connection is established. See [Authentication Method](#) on page 1056 for details.

- The Service Principal Name connection option allows you to specify the service principal name to be used by the driver for Kerberos authentication. See [Service Principal Name](#) on page 1070 for details.
- **DB2 Wire Protocol driver enhancements**
 - The driver has been enhanced to support Select queries with parameterized arrays.
 - The new AllowedOpenSSLVersions option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 164 or [Designating an OpenSSL Library](#) on page 94 for details.
 - The new CryptoLibName and SSLLibName connection options allow you to designate the OpenSSL libraries used when SSL is enabled. See [CryptoLibName](#) on page 178 and [SSLLibName](#) on page 204 for details.
 - The new Crypto Protocol Version connection option allows you to specify the cryptographic protocols used when SSL is enabled. See [Crypto Protocol Version](#) on page 178 for details.
 - The new Min Long Varchar Size connection option allows you to fetch SQL_LONGVARCHAR columns whose size is smaller than the minimum imposed by some third-party applications. See [Min Long Varchar Size](#) on page 196 for details.
 - The new Varchar Threshold connection option allows you to fetch columns that would otherwise exceed the upper limit of the SQL_VARCHAR type for some third-party applications. See [Varchar Threshold](#) on page 209 for details.
 - The TCP Keep Alive connection option allows you to use TCP Keep Alive to maintain idle TCP connections. See [TCP Keep Alive](#) on page 205 for details.
- **Greenplum driver enhancements**
 - The driver has been enhanced with the new Batch Mechanism (BatchMechanism) connection option, which specifies the preferred mechanism for executing batch insert operations. By setting Batch Mechanism to 2 (MultiRowInsert) or 3 (Copy), the driver can achieve substantial performance gains when performing batch inserts. The default setting is BatchMechanism=1. See [Batch Mechanism](#) on page 844 for details.
 - A Power BI connector is now included with the product package. You can use this connector to access your Greenplum data with Power BI. See [Accessing Greenplum data with Power BI](#) on page 837 for details.
 - The new AllowedOpenSSLVersions option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 841 or [Designating an OpenSSL Library](#) on page 94 for details.
 - Added support for Kerberos authentication, including the following connection options:
 - The Authentication Method connection option now allows you to specify the method the driver uses to authenticate the user to the server when a connection is established. See [Authentication Method](#) on page 843 for details.
 - The GSS Client Library connection option now allows you to specify the name of the GSS client library that the driver uses to communicate with the Key Distribution center (KDC). See [GSS Client Library](#) on page 857 for details.
 - The Service Principal Name connection option allows you to specify the service principal name to be used by the driver for Kerberos authentication. See [Service Principal Name](#) on page 871 for details.
 - The new CryptoLibName and SSLLibName connection options allow you to designate the OpenSSL libraries used when SSL is enabled. See [CryptoLibName](#) on page 848 and [SSLLibName](#) on page 872 for details.

- The new Crypto Protocol Version connection option allows you to specify the cryptographic protocols used when SSL is enabled. See [Crypto Protocol Version](#) on page 847 for details.
- The new Unbounded Numeric Precision connection option allows you to define the precision for unbounded NUMERIC columns when described within the column, parameter, result set, or table metadata. See [Unbounded Numeric Precision](#) on page 877 for details.
- The new Unbounded Numeric Scale connection option allows you to define the scale for unbounded NUMERIC columns described within the column, parameter, result set, or table metadata. See [Unbounded Numeric Scale](#) on page 877 for details.
- The TCP Keep Alive connection option allows you to use TCP Keep Alive to maintain idle TCP connections. See [TCP Keep Alive](#) on page 873 for details.
- The Max Char Size connection option specifies the maximum size of columns of type SQL_VARCHAR that the driver describes through result set descriptions and catalog functions. See [Max Char Size](#) on page 866 for details.
- The Max Long Varchar Size connection option specifies the maximum size of columns of type SQL_LONGVARCHAR that the driver describes through result set descriptions and catalog functions. See [Max Long Varchar Size](#) on page 866 for details.
- The Enable Keyset Cursors connection option enables emulated Keyset cursors to provide scrollable cursors to an ODBC application. See [Enable Keyset Cursors](#) on page 851 for details.
- The Keyset Cursor Options connection option determines which columns are used to comprise the keyset that the driver uses to create the initial keyset on which cursor operations are based. See [Keyset Cursor Options](#) on page 861 for details.
- Added support for SSL encryption with Greenplum 4.2 and higher, including the following connection options:
 - The Encryption Method connection option now allows you to encrypt data sent between the driver and the database. See [Encryption Method](#) on page 852 for details.
 - The Host Name In Certificate connection option now allows you to specify the host name for certificate validation when SSL encryption and validation are enabled. See [Host Name In Certificate](#) on page 859 for details.
 - The Key Password connection option now allows you to specify the key password that is used to access the individual keys in the keystore file when SSL and SSL client authentication are enabled on the database server. See [Key Password](#) on page 862 for details.
 - The Keystore connection option now allows you to specify the directory containing the keystore file that is to be used when SSL and SSL client authentication are enabled on the database server. See [Keystore](#) on page 862 for details.
 - The Key Store Password connection option allows you to specify the keystore password that is used to access the keystore file when SSL and SSL client authentication are enabled on the database server. See [Keystore Password](#) on page 863 for details.
 - The Truststore connection option now allows you to specify the directory that contains the truststore file and the truststore file name that is to be used when SSL is enabled and the server authentication is used. See [Truststore](#) on page 874 for details.
 - The Truststore Password connection option now allows you to specify the truststore password that is used to access the truststore file when SSL is enabled and the server authentication is used. See [Truststore Password](#) on page 875 for details.
 - The User Name connection option now allows you to specify the user ID that is used to connect to your database. See [User Name](#) on page 875 for details.

- The Validate Server Certificate connection option now determines whether the driver validates the certificates that are sent by the database server when SSL encryption is enabled. See [Validate Server Certificate](#) on page 876 for details.
- **Impala driver enhancements since General Availability**
 - The new AllowedOpenSSLVersions option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 841 or [Designating an OpenSSL Library](#) on page 94 for details.
 - The BatchMechanism connection option has been added to the driver. When BatchMechanism is set to 2 (MultiRowInsert), the driver executes a single insert for all the rows contained in a parameter array. MultiRowInsert is the default setting and provides substantial performance gains when performing batch inserts. See [Batch Mechanism](#) on page 897 for details.
 - Added support for SSL encryption, including the following connection options:
 - The CryptoLibName connection option allows you to determine the cryptographic library used when SSL is enabled. See [CryptoLibName](#) on page 898 for details.
 - Crypto Protocol Version connection allows you to specify the cryptographic protocols used when SSL is enabled. See [Crypto Protocol Version](#) on page 897 for details.
 - The Encryption Method connection option now allows you to encrypt data sent between the driver and the database. See [Encryption Method](#) on page 903 for details.
 - The Host Name In Certificate connection option now allows you to specify the host name for certificate validation when SSL encryption and validation are enabled. See [Host Name In Certificate](#) on page 905 for details.
 - The Key Password connection option now allows you to specify the key password that is used to access the individual keys in the keystore file when SSL and SSL client authentication are enabled on the database server. See [Key Password](#) on page 905 for details.
 - The Key Store connection option now allows you to specify the directory containing the keystore file that is to be used when SSL and SSL client authentication are enabled on the database server. See [Key Store](#) on page 906 for details.
 - The Keystore Password connection option allows you to specify the keystore password that is used to access the keystore file when SSL and SSL client authentication are enabled on the database server. See [Keystore Password](#) on page 907 for details.
 - The SSLibName connection option allows you to determine the SSL library used when SSL is enabled. See [SSLibName](#) on page 912 for details.
 - The Truststore connection option now allows you to specify the directory that contains the truststore file and the truststore file name that is to be used when SSL is enabled and the server authentication is used. See [Truststore](#) on page 915 for details.
 - The Trust Store Password connection option now allows you to specify the truststore password that is used to access the truststore file when SSL is enabled and the server authentication is used. See [Trust Store Password](#) on page 915 for details.
 - The User Name connection option now allows you to specify the user ID that is used to connect to your database. See [User Name](#) on page 917 for details.
 - The Validate Server Certificate connection option now determines whether the driver validates the certificates that are sent by the database server when SSL encryption is enabled. See [Validate Server Certificate](#) on page 917 for details.

- The Authentication Method connection option has been refreshed with a new valid value for enabling Kerberos Authentication. To use Kerberos authentication with the driver, set `AuthenticationMethod=4`. See [Authentication Method](#) on page 896 for details.

Note: The legacy setting for enabling Kerberos Authentication (`AuthenticationMethod=1`) will continue to be valid for this release; however, it will be disabled in future versions of the product.

- Support for Kerberos Authentication, which can be configured using the following connection options:
 - Authentication Method specifies the method the driver uses to authenticate the user to the server when a connection is established. See [Authentication Method](#) on page 896 for details.
 - GSS Client Library specifies the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC). See [GSS Client Library](#) on page 903 for details.
 - Service Principal Name specifies the service principal name to be used by driver for Kerberos authentication. See [Service Principal Name](#) on page 911 for details.
- Certified with Apache Sentry for Impala 1.1 and higher. Sentry enables administrators to control access to data and metadata stored on an Hadoop cluster by defining user roles and permissions. See [Apache Sentry](#) on page 919 for details.
- The driver has been enhanced to support the Char, Decimal, and Varchar data types when connected to Impala 2.0 and higher. See [Data Types](#) on page 918 for details.
- The Array Size configuration option has been refreshed to allow specifying the number of cells retrieved instead of rows. By determining the fetch size based on the number of cells, the driver can avoid out of memory errors when fetching from tables containing a large number of columns. See [Array Size](#) on page 895 for details.
- **MySQL driver enhancements**
 - The driver has been enhanced to support the `sha256_password` and `caching_sha2_password` authentication plugins.
 - The new `AllowedOpenSSLVersions` option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 251 or [Designating an OpenSSL Library](#) on page 94 for details.
 - The new `CryptoLibName` and `SSLLibName` connection options allow you to designate the OpenSSL libraries used when SSL is enabled. See [CryptoLibName](#) on page 257 and [SSLLibName](#) on page 274 for details.
 - The new `Crypto Protocol Version` connection option allows you to specify the cryptographic protocols used when SSL is enabled. See [Crypto Protocol Version](#) on page 256 for details.
 - The `TCP Keep Alive` connection option allows you to use TCP Keep Alive to maintain idle TCP connections. See [TCP Keep Alive](#) on page 275 for details.
- **Oracle Wire Protocol driver enhancements**
 - The new `AllowedOpenSSLVersions` option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 308 or [Designating an OpenSSL Library](#) on page 94 for details.
 - The maximum supported length of identifiers has been increased to 128 bytes when connecting to Oracle 12c R2 (12.2) databases. This change has been implemented to reflect the new maximum length supported by the server.

- The new `CryptoLibName` and `SSLLibName` connection options allow you to designate the OpenSSL libraries used when SSL is enabled. See [CryptoLibName](#) on page 323 and [SSLLibName](#) on page 360 for details.
- The new `Crypto Protocol Version` connection option allows you to specify the cryptographic protocols used when SSL is enabled. See [Crypto Protocol Version](#) on page 323 for details.
- The `TCP Keep Alive` connection option allows you to use TCP Keep Alive to maintain idle TCP connections. See [TCP Keep Alive](#) on page 362 for details.
- Oracle Advanced Security support, which can be configured using the following connection options:
 - `Data Integrity Level` sets the level of OAS data integrity used for data sent between the driver and database server. See [Data Integrity Level](#) on page 325 for details.
 - `Data Integrity Types` specifies one or multiple algorithms to protect against attacks that intercept and modify data being transmitted between the client and server when OAS data integrity is enabled using the `Data Integrity Level` option. See [Data Integrity Types](#) on page 325 for details.
 - `Encryption Level` determines whether data is encrypted and decrypted when transmitted over the network between the driver and database server using OAS encryption. See [Encryption Level](#) on page 333 for details.
 - `Encryption Types` specifies one or multiple algorithms to use if OAS encryption is enabled using the `Encryption Level` property. See [Encryption Types](#) on page 335 for details.
 - Modified to support all Oracle 11g R2 Kerberos encryption algorithms.
- **PostgreSQL driver enhancements**
 - The driver has been enhanced to support the SCRAM-SHA-256-PLUS authentication mechanism, which uses channel binding for establishing a secure connection with PostgreSQL (v11.0 and higher). The driver uses the SCRAM-SHA-256 authentication mechanism if the TLS library is unavailable in a server that is configured with both SCRAM-SHA-256 and SCRAM-SHA-256-PLUS authentication mechanisms.
 - The driver has been enhanced with the new `Batch Mechanism` (`BatchMechanism`) connection option, which specifies the preferred mechanism for executing batch insert operations. By setting `Batch Mechanism` to 2 (`MultiRowInsert`) or 3 (`Copy`), the driver can achieve substantial performance gains when performing batch inserts. The default setting is `BatchMechanism=1`. See [Batch Mechanism](#) on page 1056 for details.
 - For PostgreSQL 9.0 and later, the driver behavior has been updated to support executing multiple prepared statements in a single query that contain inserts for BYTEA values. However, for versions earlier than PostgreSQL 9.0, this functionality is not supported and the driver returns an error. See [Data Types](#) on page 879 for details.
 - A Power BI connector is now included with the product package. You can use this connector to access your PostgreSQL data with Power BI. See [Accessing PostgreSQL data with Power BI](#) on page 390 for details.
 - The new `AllowedOpenSSLVersions` option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 394 or [Designating an OpenSSL Library](#) on page 94 for details.
 - The driver has been enhanced to support Select queries with parameterized arrays.
 - The driver has been enhanced to support MD5 and SCRAM-SHA-256 authentication methods.
 - Support materialized views and foreign tables.
 - Support for Kerberos Authentication, which can be configured using the following connection options:

- Authentication Method specifies the method the driver uses to authenticate the user to the server when a connection is established. See [Authentication Method](#) on page 396 for details.
- GSS Client Library specifies the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC). See [GSS Client Library](#) on page 410 for details.
- Service Principal Name specifies the service principal name to be used by driver for Kerberos authentication. See [Service Principal Name](#) on page 424 for details.
- The new CryptoLibName and SSLLibName connection options allow you to designate the OpenSSL libraries used when SSL is enabled. See [CryptoLibName](#) on page 401 and [SSLLibName](#) on page 424 for details.
- The new Crypto Protocol Version connection option allows you to specify the cryptographic protocols used when SSL is enabled. See [Crypto Protocol Version](#) on page 400 for details.
- The new Unbounded Numeric Precision connection option allows you to define the precision for unbounded NUMERIC columns when described within the column, parameter, result set, or table metadata. See [Unbounded Numeric Precision](#) on page 428 for details.
- The new Unbounded Numeric Scale connection option allows you to define the scale for unbounded NUMERIC columns described within the column, parameter, result set, or table metadata. See [Unbounded Numeric Scale](#) on page 428 for details.
- The TCP Keep Alive connection option allows you to use TCP Keep Alive to maintain idle TCP connections. See [TCP Keep Alive](#) on page 425 for details.
- The Max Char Size connection option specifies the maximum size of columns of type SQL_CHAR that the driver describes through result set descriptions and catalog functions. See [Max Char Size](#) on page 419 for details.
- The Max Long Varchar Size connection option specifies the maximum size of columns of type SQL_LONGVARCHAR that the driver describes through result set descriptions and catalog functions. See [Max Long Varchar Size](#) on page 419 for details.
- The Enable Keyset Cursors connection option enables emulated Keyset cursors to provide scrollable cursors to an ODBC application. See [Enable Keyset Cursors](#) on page 404 for details.
- The Keyset Cursor Options connection option determines which columns are used to comprise the keyset that the driver uses to create the initial keyset on which cursor operations are based. See [Keyset Cursor Options](#) on page 416 for details.
- The Encryption Method connection option now supports Request SSL functionality. See [Encryption Method](#) on page 405 for details.
- **Progress OpenEdge driver enhancements**
 - The new AllowedOpenSSLVersions option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 447 or [Designating an OpenSSL Library](#) on page 94 for details.
 - The new CryptoLibName and SSLLibName connection options allow you to designate the OpenSSL libraries used when SSL is enabled. See [CryptoLibName](#) on page 451 and [Encryption Method](#) on page 455 for details.
 - The new Crypto Protocol Version connection option allows you to specify the cryptographic protocols used when SSL is enabled. See [Crypto Protocol Version](#) on page 450 for details.
 - The TCP Keep Alive connection option allows you to use TCP Keep Alive to maintain idle TCP connections. See [TCP Keep Alive](#) on page 465 for details.
- **Salesforce driver enhancements**

- Certified with Salesforce API Version 26, 27, 28, 29, 33, 34, 38
- The driver no longer supports 32-bit Solaris x86 and 32-bit Solaris on SPARC platforms due to changes in Salesforce security standards. As previously announced, Salesforce now requires Java SE 8 and higher versions of the JVM. The highest version currently supported on either platform is Java SE 6; therefore, the driver is no longer able to connect to Salesforce instances on these platforms.
- The Salesforce driver has been updated to require a JVM that is version Java SE 8 or higher. This change is being implemented to comply with recent revisions to Salesforce security standards.

In keeping with best security practices, Salesforce will begin deprecating support for the TLS 1.0 encryption protocol within inbound and outbound connections on June 25th, 2016. TLS 1.0 will initially be disabled in Sandbox instances, before being retired for all services in early 2017. To maintain compatibility with services after TLS 1.0 is disabled, the driver must employ a JVM that allows TLS 1.0 to be disabled independently from other encryption protocols, functionality that was first introduced in Java SE 7.

By default, the driver uses the Java SE 8 JVM that is installed with the driver. If you designate a JVM that is version Java SE 7 or earlier, the driver will return an error when attempting to establish a connection. To correct this issue, set the library path environment variable to the location of a supported JVM.

For more information on changes to the Salesforce security policy, refer to <https://help.salesforce.com/apex/HTViewSolution?id=000221207#Whatischange>.

- The Refresh Schema connection option specifies whether the driver automatically refreshes the remote object mapping and other information contained in a remote schema the first time a user connects to the specified embedded database. See [Refresh Schema](#) on page 966 for details.
- The KeywordConflictSuffix config option allows you to specify a string of up to five alphanumeric characters that the driver appends to any object or field name that conflicts with a SQL engine keyword. See [Config Options](#) on page 945 for details.
- **SQL Server Wire Protocol driver enhancements**
 - The driver has been enhanced to transparently connect to Microsoft Azure Synapse Analytics and Microsoft Analytics Platform System data sources. See [Support for Azure Synapse Analytics and Analytics Platform System](#) on page 541 for more information about supported features and functionality.
 - The new AllowedOpenSSLVersions option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 489 or [Designating an OpenSSL Library](#) on page 94 for details.
 - The driver has been enhanced to support Azure Active Directory authentication (Azure AD authentication). Azure AD authentication is an alternative to SQL Server Authentication that allows administrators to centrally manage user permissions to Azure SQL Database data stores. See [Configuring Azure Active Directory Authentication](#) on page 537 for details.
 - The driver has been enhanced to support Always On Availability Groups. Introduced in SQL Server 2012, Always On Availability Groups is a replica-database environment that provides a high-level of data availability, protection, and recovery. To support this enhancement, the following updates have been made to the driver:
 - The Host Name option has been updated to support the virtual network name (VNN) of the availability group listener as a valid value. To connect to an Always On Availability group, you must specify the VNN with this option.
 - The new Application Intent option allows you to control whether the driver requests read-only routing, thereby improving efficiency by reducing the workload on read-write nodes.
 - The new Multi-Subnet Failover option allows the driver to attempt parallel connections to all the IP addresses associated with an availability group when the primary listener is unavailable. This offers improved response time over traditional failover, which attempts connections to alternate servers one at a time.

See [Host Name](#) on page 513, [Application Intent](#) on page 492, and [Multi-Subnet Failover](#) on page 520 for details.

- The new Bulk Load Threshold option allows you to determine when the driver uses bulk load for insert, update, delete, or batch operations. See [Bulk Load Threshold](#) on page 497 for details.
- The new CryptoLibName and SSLLibName connection options allow you to designate the OpenSSL libraries used when SSL is enabled. See [CryptoLibName](#) on page 503 and [SSLLibName](#) on page 527 for details.
- The driver has been enhanced to support NTLMv2 authentication, which can be configured using the Authentication Method connection option. See [Authentication Method](#) on page 494 for details.
- The new Crypto Protocol Version connection option allows you to specify the cryptographic protocols used when SSL is enabled. See [Crypto Protocol Version](#) on page 502 for details.
- The TCP Keep Alive connection option allows you to use TCP Keep Alive to maintain idle TCP connections. See [TCP Keep Alive](#) on page 528 for details.
- **Sybase Wire Protocol driver enhancements**
 - The driver has been enhanced to support Sybase Extended Password Encryption and Sybase Extended Plus Encrypted Password, which use the asymmetrical key type. This provides stronger password encryption for the secure transmission of public key passwords over networks. See [Authentication Method](#) on page 565 for details.
 - The driver has been enhanced to support BINARY and VARBINARY data types when using the bulk load protocol.
 - The new AllowedOpenSSLVersions option allows you to determine which version of the OpenSSL library file the driver uses for data encryption. See [AllowedOpenSSLVersions](#) on page 561 or [Designating an OpenSSL Library](#) on page 94 for details.
 - The new CryptoLibName and SSLLibName connection options allow you to designate the OpenSSL libraries used when SSL is enabled. See [CryptoLibName](#) on page 572 and [Crypto Protocol Version](#) on page 571 for details.
 - The new Crypto Protocol Version connection option allows you to specify the cryptographic protocols used when SSL is enabled. See [Crypto Protocol Version](#) on page 571 for details.
 - The TCP Keep Alive connection option allows you to use TCP Keep Alive to maintain idle TCP connections. See [TCP Keep Alive](#) on page 602 for details.
- **Driver for the Teradata Database enhancements**
 - The driver has been enhanced to support the Number data type. See [Data Types](#) on page 1109 for details.
- **New driver DataDirect Connect64 for ODBC**
 - Text Driver: A 64-bit version of the driver is now generally available. This driver provides 64-bit support for the features and functionality offered by the earlier DataDirect Connect (32-bit) version of the driver. In addition, for the Connect64 driver, support for HP-UX IPF has been added. See [The Text Driver](#) on page 691 for details.

Note: The Connect64 Text driver uses a driver-specific installer, instead of the Connect Series installer. Both installers are available on the Progress website.

- **New drivers DataDirect Connect XE and DataDirect Connect64 XE for ODBC**

- Driver for Apache Hive™
 - Returns result set metadata for parameterized statements that have been prepared but not yet executed.
 - Supports parameter arrays, processing the arrays as a series of executions, one execution for each row in the array.
 - Provides a connection option that allows you to configure the driver to report that it supports transactions, although Hive does not support transactions. This provides a workaround for applications that do not operate with a driver that reports that transactions are not supported.
 - Supports the following standard SQL functionality:
 - Create Index, Create Table, and Create View
 - Insert, Update, and Delete
 - Drop Index, Drop Table, and Drop View

See [The Driver for Apache Hive](#) on page 1045 for details.

- Impala™ Wire Protocol Driver
 - Supports Cloudera Impala database servers and formally certified with the following file formats and storage handlers:
 - File Formats:
 - Parquet
 - TextFile
 - Storage Handlers:
 - HBase
 - Returns result set metadata for parameterized statements that have been prepared by not yet executed.
 - Supports parameter arrays, processing the arrays as a series of executions, one execution for each row in the array.
 - Provides a connection option that allows you to configure the driver to report that it supports transactions, although Impala does not support transactions. This provides a workaround for applications that do not operate with a driver that reports that transactions are not supported.
 - Provides a connection option that allows you to set a default limit for the number of rows returned when an ORDER BY clause is submitted. This provides a workaround for applications that are not compatible with Impala's requirement that ORDER BY clauses limit the number of rows returned.
 - Supports the following standard SQL functionality:
 - Create Index and Create Table
 - Insert, Update, and Delete
 - Drop Index and Drop Table

See [The Impala Wire Protocol Driver](#) on page 884 for details.

Product Matrix

The DataDirect Connect Series *for* ODBC products include 32- and 64-bit drivers. DataDirect Connect *for* ODBC (32-bit) and DataDirect Connect64 *for* ODBC (64-bit) are detailed in the following table.

Note: 8.0 and higher versions of Connect Series for ODBC drivers and the 7.1 version of the Connect64 Text driver use standalone installers. For information on those installers, refer to the *Progress DataDirect for ODBC Drivers Installation Guide*.

Driver	Connect <i>for</i> ODBC	Connect64 <i>for</i> ODBC
DB2 Wire Protocol	X	X
Informix Wire Protocol	X	X
MySQL Wire Protocol	X	X
Oracle Wire Protocol	X	X
PostgreSQL Wire Protocol	X	X
Progress OpenEdge® Wire Protocol	X	X
SQL Server Wire Protocol	X	X
Sybase Wire Protocol	X	X
Oracle (client)	X	X
SQL Server Legacy Wire Protocol	X	X
Text	X	X
Btrieve	X	
dBASE	X	
Informix (client)	X	
XML	X	

DataDirect Connect XE *for* ODBC (32-bit) and DataDirect Connect64 XE *for* ODBC (64-bit) products consists of the drivers detailed in the following table.

Driver	Connect XE <i>for</i> ODBC	Connect64 XE <i>for</i> ODBC
Greenplum Wire Protocol	X	X
Impala™ Wire Protocol	X	X
Salesforce	X	X

Driver	Connect XE <i>for</i> ODBC	Connect64 XE <i>for</i> ODBC
Sybase IQ	X	X
Driver for Apache Hive™	X	X
Driver for the Teradata Database	X	X

Product Platforms

DataDirect Connect Series *for* ODBC drivers allow you to connect to a variety of databases from these platforms:

Windows (32-bit)

- Windows 10
- Windows 8.1
- Windows 7
- Windows Server 2012
- Windows Server 2008

Windows (64-bit)

- Windows 10
- Windows 8.1
- Windows 7
- Windows Server 2012
- Windows Server 2008

UNIX and Linux (32-bit)

- AIX
- HP-UX aCC Enabled
- Linux
- Oracle Solaris

UNIX and Linux (64-bit)

- AIX
- HP-UX aCC Enabled
- Linux
- Oracle Solaris

About the Documentation Library

The documentation library is available on the Progress DataDirect Connectors Documentation Hub:
<https://docs.progress.com/bundle/datadirect-connectors/page/DataDirect-Connectors-by-data-source.html>.

The DataDirect Connect Series for ODBC includes the following documents:

- *DataDirect Connect Series for ODBC Installation Guide* details requirements and procedures for installing the product.
- *DataDirect Connect Series for ODBC User's Guide* provides information about configuring and using the product.
- *DataDirect Connect Series for ODBC Reference* provides detailed reference information about the product.
- *DataDirect Connect Series for ODBC Troubleshooting Guide* provides information about error messages and troubleshooting procedures for the product.

Contacting Technical Support

Progress DataDirect offers a variety of options to meet your support needs. Please visit our Web site for more details and for contact information:

<https://www.progress.com/support>

The Progress DataDirect Web site provides the latest support information through our global service network. The SupportLink program provides access to support contact details, tools, patches, and valuable information, including a list of FAQs for each product. In addition, you can search our Knowledgebase for technical bulletins and other information.

When you contact us for assistance, please provide the following information:

- Your number or the serial number that corresponds to the product for which you are seeking support, or a case number if you have been provided one for your issue. If you do not have a SupportLink contract, the SupportLink representative assisting you will connect you with our Sales team.
- Your name, phone number, email address, and organization. For a first-time call, you may be asked for full information, including location.
- The Progress DataDirect product and the version that you are using.
- The type and version of the operating system where you have installed your product.
- Any database, database version, third-party software, or other environment information required to understand the problem.
- A brief description of the problem, including, but not limited to, any error messages you have received, what steps you followed prior to the initial occurrence of the problem, any trace logs capturing the issue, and so on. Depending on the complexity of the problem, you may be asked to submit an example or reproducible application so that the issue can be re-created.
- A description of what you have attempted to resolve the issue. If you have researched your issue on Web search engines, our Knowledgebase, or have tested additional configurations, applications, or other vendor products, you will want to carefully note everything you have already attempted.
- A simple assessment of how the severity of the issue is impacting your organization.

Quick Start Connect

This chapter provides basic information about configuring your driver immediately after installation and testing your connection. To take full advantage of the features of the driver, read [About the Product](#) on page 57 and the driver-specific chapter.

Information that the driver needs to connect to a database is stored in a *data source*. The ODBC specification describes three types of data sources: user data sources, system data sources (not a valid type on UNIX/Linux), and file data sources. On Windows, user and system data sources are stored in the registry of the local computer. The difference is that only a specific user can access user data sources, whereas any user of the machine can access system data sources. On Windows, UNIX, and Linux, file data sources, which are simply text files, can be stored locally or on a network computer, and are accessible to other machines.

When you define and configure a data source, you store default connection values for the driver that are used each time you connect to a particular database. You can change these defaults by modifying the data source.

For details, see the following topics:

- [Configuring and Connecting on Windows](#)
- [Configuring and Connecting on UNIX and Linux](#)
- [Using the Performance Wizard](#)

Configuring and Connecting on Windows



The following basic information enables you to configure a data source and test connect with a driver immediately after installation. On Windows, you can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box. Default connection values are specified through the options on the tabs of the Setup dialog box and are stored either as a user or system data source in the Windows Registry, or as a file data source in a specified location.

Setting the Library Path Environment Variable (Salesforce Driver on Windows)

Before you can use the Salesforce driver, you must set the PATH environment variable to the path of the jvm.dll file of your Java™ Virtual Machine (JVM).

Configuring a Data Source

To configure a data source:

1. From the DataDirect program group, start the ODBC Administrator and click either the **User DSN**, **System DSN**, or **File DSN** tab to display a list of data sources.

- **User DSN:** If you installed default DataDirect ODBC user data sources as part of the installation, select the appropriate data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the appropriate driver and click **Finish** to display the driver Setup dialog box.

- **System DSN:** To configure a new system data source, click **Add** to display a list of installed drivers. Select the appropriate driver and click **Finish** to display the driver Setup dialog box.
- **File DSN:** To configure a new file data source, click **Add** to display a list of installed drivers. Select the driver and click **Advanced** to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise in this book.

2. The following two options appear on the General tab of all driver Setup dialog boxes:

Data Source Name: Type a string that identifies this data source configuration, such as Accounting.

Description: Type an optional long description of a data source name, such as My Accounting Database.

Provide the requested information for all other options on the General tab; then, click **Apply** to configure the data source.

Minimum Configuration Requirements (Windows)

The following section describes the minimum options required to establish a connection for drivers that support 32 and 64-bit platforms. For a complete list of supported connection options, refer to the chapter for your driver.

Apache Hive	PostgreSQL Wire Protocol
DB2 Wire Protocol	Progress OpenEdge
Greenplum Wire Protocol	Salesforce
Informix	SQL Server Wire Protocol
Informix Wire Protocol	Sybase IQ
MySQL	Sybase Wire Protocol
Oracle	Teradata
Oracle Wire Protocol	

Driver for Apache Hive

Provide the following information on the General Tab:

- **Host Name:** Type the name or the IP address of the server to which you want to connect.
- **Port Number:** Type the port number of the server listener. The default port number for the Apache Hive server is 10000. Because of reported concurrency issues, you might want to use a different port number.
- **Database Name:** Type the name of the Apache Hive database to which you want to connect by default. The database must exist, or the connection attempt will fail.

On the Security tab, provide the following information; then, click **Apply**.

- **User Name:** Type the default user ID that is used to connect to your database.

DB2 Wire Protocol

Prerequisites:

- You must have the appropriate privileges for the driver to create and bind packages with your user ID. These privileges are BINDADD for binding packages, CREATEIN on the collection specified by the Package Collection option, and GRANT EXECUTE on the PUBLIC group for executing the packages. These are typically the permissions of a Database Administrator (DBA). If you do not have these privileges, someone that has a user ID with DBA privileges needs to create packages by connecting with the driver. When connecting for the first time, the driver determines whether bind packages exist on the server. If packages do not exist, the driver creates them automatically using driver default values.

Following is a list of connection options on the General Tab:

- **IP Address:** Type the IP address of the machine where the catalog tables are stored. Specify the address using the machine's numeric address or specify its host name. If you enter a host name, the driver must find this name (with the correct address assignment) in the HOSTS file on the workstation or in a DNS server. The default is localhost.
- **Tcp Port:** Type the port number that is assigned to the DB2 DRDA listener process on the server host machine. Specify either this port's numeric address or its service name. If you specify a service name, the driver must find this name (with the correct port assignment) in the SERVICES file on the workstation. The default is 50000.

On DB2 for i only, execute `NETSTAT` from a DB2 for i command line to determine the correct port number. Select option 3 to display a list of active ports on the DB2 for i machine. Find the entry for DRDA and press F-14 to toggle and display the port number. If DRDA is not currently listening, the DB2 for i command, `CHGDDMTCPA AUTOSTART(*YES) PWDRQD(*YES)` starts the listener and ensures that it is active at IPL.

- **Location Name:** This field is valid and required only if you are connecting to a DB2 database on DB2 for i or z/OS. Type the DB2 location name. Use the name defined during the local DB2 installation.

On z/OS only, your system administrator can determine the name of your DB2 location using the `DISPLAY DDF` command.

On DB2 for i only, your system administrator can determine the name of your DB2 location using the `WRKRDBDIRE` command. The name of the database that is listed as *LOCAL is the value you should use.

Note: This field is disabled if the Database Name field is populated.

- **Collection:** This field is valid only if you are connecting to a DB2 database on DB2 for i or z/OS. By default, the user ID is used for the value of Collection. The user ID should always be used on z/OS.

Note: This field is disabled if the Database Name field is populated.

- **Database Name:** This field is valid and required only if you are connecting to a DB2 database on Linux/UNIX/Windows. Type the name of the database to which you want to connect.

Note: This field is disabled if the Location Name or Collection fields are populated.

Greenplum Wire Protocol

Provide the following information on the General Tab:

- **Host Name:** Type either the name or the IP address of the server to which you want to connect.
- **Port Number:** Type the port number of the server listener. The default is 5432.
- **Database Name:** Type the name of the database to which you want to connect by default.

Impala Wire Protocol

Provide the following information on the General Tab:

- **Host Name:** Type either the name or the IP address of the server to which you want to connect.
- **Port Number:** Type the port number of the server listener. The default is 21050.
- **Database Name:** Type the name of the database to which you want to connect by default.

Informix

Prerequisites:

- You must have all components of your database client software installed and connecting properly; otherwise, the driver will not operate correctly.

Following is a list of connection options on the General Tab:

- **Database Name:** Type the name of the database to which you want to connect by default.

You must also provide the following information on the Connection tab; then, click **Apply**.

- **Host Name:** Type the name of the machine on which the Informix server resides.
- **Service Name:** Type the name of the service as it appears on the host machine.
- **Server Name:** Type the name of the Informix server as it appears in the `sqlhosts` file.

Informix Wire Protocol

Provide the following information on the General Tab:

- **Host Name:** Type either the name or the IP address of the server to which you want to connect.
- **Port Number:** Type the port number of the server listener.
- **Server Name:** Type the name of the Informix server as it appears in the `sqlhosts` file.
- **Database Name:** Type the name of the database to which you want to connect by default.
- **User Name:** Type your user name as specified on the Informix server.

MySQL Wire Protocol

Provide the following information on the General Tab:

- **Host Name:** Type either the name or the IP address of the server to which you want to connect.
- **Port Number:** Type the port number of the server listener. The default is 3306.
- **Database Name:** Type the name of the database to which you want to connect by default.

Oracle

Prerequisites:

- You must have all components of your database client software installed and connecting properly; otherwise, the driver will not operate correctly.

Following is a list of connection options on the General Tab:

- **Server Name:** Type the client connection string designating the server and database to be accessed. The information required varies depending on the client driver you are using.

Oracle Wire Protocol

Provide the following information on the General Tab:

- **Host:** Type either the name or the IP address of the server to which you want to connect.
- **Port Number:** Type the port number of your Oracle listener. Check with your database administrator for the correct number.
- **SID:** Type the Oracle System Identifier that refers to the instance of Oracle running on the server. The default is ORCL.

This option and the Service Name option are mutually exclusive. If the Service Name option is specified, do not specify this option.
- **Service Name:** Type the Oracle service name that specifies the database used for the connection. The service name is a string that is the global database name—a name that is comprised of the database name and domain name, for example: `sales.us.acme.com`.

This option and the SID option are mutually exclusive. If the SID option is specified, do not specify this option.
- **Edition Name:** Oracle 11g R2 and higher only. Type the name of the Oracle edition that the driver is to use when establishing a connection. Oracle 11g R2 and higher allows your database administrator to create multiple editions of schema objects so that your application can still use those objects while the database is being upgraded. This option tells the driver which edition of the schema objects to use.

PostgreSQL Wire Protocol

Provide the following information on the General Tab:

- **Host Name:** Type either the name or the IP address of the server to which you want to connect.
- **Port Number:** Type the port number of the server listener. The default is 5432.
- **Database Name:** Type the name of the database to which you want to connect by default.

Progress OpenEdge Wire Protocol

Provide the following information on the General Tab:

- **Host Name:** Type either the name or the IP address of the server to which you want to connect.
- **Port Number:** Type the port number of the server listener.
- **Database Name:** Type the name of the database to which you want to connect by default.
- **User ID:** Type your user name of as specified on the Progress OpenEdge server.

Salesforce

Prerequisites:

- The driver requires Java SE 8 or higher. Before you configure a data source for the Salesforce driver, you must set the PATH library path environment variable to the path of the `jvm.dll` file of your JVM.

Provide the following information on the General Tab:

- **Host Name:** The default Salesforce instance is `login.salesforce.com`. If you are logging into a different Salesforce instance, type the root of the Salesforce URL. Otherwise, leave the field blank.

You must provide the following information in the logon dialog box:

- **User Name:** Type your logon ID for Salesforce.
- **Password:** Type your case-sensitive password for the Salesforce instance.

If your Salesforce instance requires a security token, you can append it to the password, for example, *secretXaBARTsLZReM4Px47qPLOS*, where *secret* is the password and the remainder of the value is the security token. Both the password and security token are case-sensitive.

SQL Server Wire Protocol

Provide the following information on the General Tab:

- **Host Name:** Type either the name or the IP address of the server to which you want to connect.

If your network supports named servers, you can specify an address as *server_name*. For example, you can enter *SSserver*.

You can also specify a named instance of Microsoft SQL Server. Specify this address as *server_name\instance_name*. If only a server name is specified with no instance name, the driver uses the default named instance on the server.

- **Port Number:** Type the port number of the server listener. The default is 1433.
- **Database Name:** Type the name of the database to which you want to connect by default.

Sybase IQ Wire Protocol

Provide the following information on the General Tab:

- **Network Address:** Type the IP address of the server to which you want to connect. Specify this address as *IP_address, port_number*. For example, you can enter *199.226.224.34, 2638*.

If your network supports named servers, you can specify an address as *server_name, port_number*. For example, you can enter *SybIQSserver, 2638*.

- **Database Name:** Type the name of the database to which you want to connect by default. If you do not specify a value, the default is the database defined by the system administrator for each user.
- **User Name:** The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Sybase Wire Protocol

Provide the following information on the General Tab:

- **Network Address:** Type the IP address of the server to which you want to connect. Specify this address as *IP_address, port_number*. For example, you can enter *199.226.224.34, 5000*.

If your network supports named servers, you can specify an address as *server_name, port_number*. For example, you can enter *SybSserver, 5000*.

- **Database Name:** Type the name of the database to which you want to connect by default. If you do not specify a value, the default is the database defined by the system administrator for each user.

Driver for the Teradata Database

Following is a list of connection options on the General Tab:

- **DBCName or Alias:** Type the IP address or the alias name of the Teradata Server. Using an IP address reduces the time it takes to connect, but if that address is not available at connection time, the connection fails and the driver does not attempt to fail over to another address.

Using an alias name increases the time it takes to connect because the driver must search a local hosts file to resolve the name to an IP address, but it allows the driver to try and connect to alternate IP addresses if the first address fails. If you use an alias name, you must have or create a local hosts file that contains the alias names. The alias name cannot be more than eight characters long.

- **DBCName List:** Type the IP addresses or the alias names that are to appear in the drop-down list of the logon dialog box. Separate the names with commas. The same restrictions apply as described for the DBCName or Alias option.

Using an alias name increases the time it takes to connect because the driver must search a local hosts file to resolve the name to an IP address, but it allows the driver to try and connect to alternate IP addresses if the first address fails. If you use an alias name, you must have or create a local hosts file that contains the alias names. The alias name cannot be more than eight characters long.

- **Integrated Security:** Select this check box to enable the user to connect to the database through single sign-on (SSO) using one of the authentication mechanisms that support SSO. When this check box is not selected (the default), UserID is required.

- **Security Mechanism:** Select TD2 from the drop-down list to specify the authentication mechanism used for connections to the data source. Valid values are:

- **Default**—uses TD2.
- **KRB5**— uses Kerberos as the authentication mechanism on Windows clients working with Windows servers if the server is V2R6.0.
- **KRB5C**— uses Kerberos Compatibility as the authentication mechanism on Windows clients working with Windows servers if the server is pre-V2R6.0.
- **LDAP**—uses LDAP as the authentication mechanism.
- **NTLM**— uses NTLM as the authentication mechanism on Windows clients working with Windows servers if the server is V2R6.0.: Type a string of characters that is to be regarded as a parameter to the authentication mechanism. The string is ignored by the ODBC driver and is passed on to the TeraSSO function that is called to set the authentication mechanism. The characters [] { } () , ; ? * = ! @ must be enclosed in curly braces.
- **NTLMC**— uses NTLM Compatibility as the authentication mechanism on Windows clients working with Windows servers if the server is pre-V2R6.0.
- **TD1**—uses Teradata 1 as the authentication mechanism.
- **TD2**—(default) uses Teradata 2 as the authentication mechanism.

- **Security Parameter**

- **UserID:** Type the default UserID for the Teradata database.

Testing the Connection

To test the connection:

1. After you have configured the data source, you can click **Test Connect** on the Setup dialog box to attempt to connect to the data source using the connection options specified in the dialog box. Some drivers immediately return a message indicating success or failure. For most drivers, a logon dialog box appears as described in each individual driver chapter.
2. Supply the requested information in the logon dialog box and click **OK**. Note that the information you enter in the logon dialog box during a test connect is not saved.
 - If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.
3. On the driver Setup dialog box, click **OK**. The values you have specified are saved and are the defaults used when you connect to the data source. You can change these defaults by using the previously described procedure to modify your data source. You can override these defaults by connecting to the data source using a connection string with alternate values. See individual driver chapters for information about using connection strings.

Configuring and Connecting on UNIX and Linux

UNIX[®]

The following basic information enables you to configure a data source and test connect with a driver immediately after installation. See [Configuring the Product on UNIX/Linux](#) on page 111 for detailed information about configuring the UNIX/Linux environment and data sources.

Note: In the following examples, `xx` in a driver filename represents the driver level number.

Environment Configuration

To configure the environment:

1. Check your permissions: You must log in as a user with full `r/w/x` permissions recursively on the entire product installation directory.
2. From your login shell, determine which shell you are running by executing:

```
echo $SHELL
```
3. Run one of the following product setup scripts from the installation directory to set variables: `odbc.sh` or `odbc.csh`. For Korn, Bourne, and equivalent shells, execute `odbc.sh`. For a C shell, execute `odbc.csh`. After running the setup script, execute:

```
env
```

to verify that the *installation_directory/lib* directory has been added to your shared library path.

4. Set the ODBCINI environment variable. The variable must point to the path from the root directory to the system information file where your data source resides. The system information file can have any name, but the product is installed with a default file called *odbc.ini* in the product installation directory. For example, if you use an installation directory of */opt/odbc* and the default system information file, from the Korn or Bourne shell, you would enter:

```
ODBCINI=/opt/odbc/odbc.ini export ODBCINI
```

From the C shell, you would enter:

```
setenv ODBCINI /opt/odbc/odbc.ini
```

Test Loading the Driver

The *ivtestlib* (32-bit drivers) and *ddtestlib* (64-bit drivers) test loading tools are provided to test load drivers and help diagnose configuration problems in the UNIX and Linux environments, such as environment variables not correctly set or missing database client components. This tool is installed in the */bin* subdirectory in the product installation directory. It attempts to load a specified ODBC driver and prints out all available error information if the load fails.

For example, if the drivers are installed in */opt/odbc/lib*, the following command attempts to load the 32-bit Oracle Wire Protocol driver on Solaris, where *xx* represents the version number of the driver:

```
ivtestlib /opt/odbc/lib/ivoraxx.so
```

Note: On Solaris, AIX, and Linux, the full path to the driver does not have to be specified for the tool. The HP-UX version, however, requires the full path.

If the load is successful, the tool returns a success message along with the version string of the driver. If the driver cannot be loaded, the tool returns an error message explaining why.

Setting the Library Path Environment Variable (Salesforce Driver on UNIX/Linux)

Before you can use the Salesforce driver, you must set the library path environment variable for your UNIX/Linux operating system to the directory containing your JVM's *libjvm.so* [*sl* | *a*] file, and that directory's parent directory.

NOTE FOR HP-UX: You also must set the *LD_PRELOAD* environment variable to the fully qualified path of the *libjvm.so*.

32-bit Salesforce Driver: Library Path Environment Variable

Set the library path environment variable to the directory containing your 32-bit JVM's *libjvm.so* [*sl* | *a*] file, and that directory's parent directory.

- *LD_LIBRARY_PATH* on Solaris, Linux, and HP-UX (Itanium)
- *SHLIB_PATH* on HP PA-RISC
- *LIBPATH* on AIX

64-bit Salesforce Driver: Library Path Environment Variable

Set the library path environment variable to the directory containing your 64-bit JVM's *libjvm.so* [*sl* | *a*] file, and that directory's parent directory.

- LD_LIBRARY_PATH on Solaris, HP-UX (Itanium), and Linux
- LIBPATH on AIX

Configuring a Data Source

If you have Motif 2.0.3 or higher and one of the supported Linux operating systems, you can use the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) to configure a data source. If you do not, see [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for instructions on configuring the system information file.

Note: The Linux ODBC Administrator is currently supported only on Linux for x86 and x64 processors with Motif 2.0.3 or higher. It is not supported on Linux for Itanium II or other UNIX platforms.

The Linux ODBC Administrator is located in the `/tools` directory of the product installation directory. For example:

```
/opt/odbc/tools/odbcadmin
```

To configure a data source:

1. To start the Linux ODBC Administrator, change to the `install_dir/tools` directory, where `install_dir` is the path to the product installation directory. At a command prompt, enter:

```
./odbcadmin
```

2. Click either the **User DSN** or **File DSN** tab to display a list of data sources.

- **User DSN:** Select the appropriate data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the appropriate driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** To configure a new file data source, click **Add** to display a list of installed drivers. Select the appropriate driver and click **Advanced** to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise in this book.

3. The following two options appear on the General tab of all driver Setup dialog boxes:

Data Source Name: Type a string that identifies this data source configuration, such as Accounting.

Description: Type an optional long description of a data source name, such as My Accounting Database.

Provide the requested information for all other options on the General tab; then, click **Apply** to configure the data source.

Minimum Configuration Requirements (UNIX/Linux)

The following section describes the minimum options required to establish a connection for drivers that support both 32 and 64-bit platforms. For a complete list of supported connection options, refer to the chapter for your driver.

[Apache Hive](#)

[PostgreSQL Wire Protocol](#)

[DB2 Wire Protocol](#)

[Progress OpenEdge](#)

[Greenplum Wire Protocol](#)

[Salesforce](#)

[Informix](#)

[SQL Server Wire Protocol](#)

[Informix Wire Protocol](#)

[Sybase IQ](#)

[MySQL](#)

[Sybase Wire Protocol](#)

[Oracle](#)

[Teradata](#)

[Oracle Wire Protocol](#)

Driver for Apache Hive

The following example demonstrates the minimum connection information required to establish a connection:

```
[Apache Hive Wire Protocol]
Driver=ODBCHOME/lib/xxhive.zz
...
Database=default
...
HostName=HiveServer
...
LogonID=yourid
...
PortNumber=10000
...
```

Connection option descriptions:

- **HostName:** Type either the name or the IP address of the server to which you want to connect.
- **Database:** Type the name of the Apache Hive database to which you want to connect by default. The database must exist, or the connection attempt will fail.
- **LogonID:** Type the default user ID that is used to connect to your database.
- **PortNumber:** Type the port number of the server listener. The default port number for the Apache Hive server is 10000. Because of reported concurrency issues, you might want to use a different port number.

DB2 Wire Protocol

Prerequisites:

- You must have the appropriate privileges for the driver to create and bind packages with your user ID. These privileges are BINDADD for binding packages, CREATEIN on the collection specified by the Package Collection option, and GRANT EXECUTE on the PUBLIC group for executing the packages. These are typically the permissions of a Database Administrator (DBA). If you do not have these privileges, someone that has a user ID with DBA privileges needs to create packages by connecting with the driver.

When connecting for the first time, the driver determines whether bind packages exist on the server. If packages do not exist, the driver creates them automatically using driver default values.

The following example demonstrates the minimum connection information required to establish a connection to DB2 for Linux/UNIX/Windows:

```
[DB2 Wire Protocol]
Driver=ODBCHOME/lib/xxdb2nn.zz
...
IpAddress=123.456.78.90
...
TcpPort=50000
...
Database=SAMPLE
...
```

The following example demonstrates the minimum connection information required to establish a connection to DB2 for i or z/OS:

```
[DB2 Wire Protocol]
Driver=ODBCHOME/lib/xxdb2nn.zz
...
IpAddress=123.456.78.90
TcpPort=446
...
Location=V5R2L0C
...
Collection=userid
...
```

Connection option descriptions:

- IpAddress:** The IP address of the machine where the catalog tables are stored. Specify the address using the machine's numeric address or specify its host name. If you enter a host name, the driver must find this name (with the correct address assignment) in the HOSTS file on the workstation or in a DNS server. The default is localhost.
- TcpPort:** The port number that is assigned to the DB2 DRDA listener process on the server host machine. Specify either this port's numeric address or its service name. If you specify a service name, the driver must find this name (with the correct port assignment) in the SERVICES file on the workstation. The default is 50000.

On DB2 for i only, execute NETSTAT from a DB2 for i command line to determine the correct port number. Select option 3 to display a list of active ports on the DB2 for i machine. Find the entry for DRDA and press F-14 to toggle and display the port number. If DRDA is not currently listening, the DB2 for i command, CHGDDMTCPA AUTOSTART(*YES) PWDRQD(*YES) starts the listener and ensures that it is active at IPL.

- Database:** This option is valid and required only if you are connecting to a DB2 database on Linux/UNIX/Windows. Enter the name of the database to which you want to connect.
- Location:** This option is valid and required only if you are connecting to a DB2 database on DB2 for i or z/OS. Enter the DB2 location name. Use the name defined during the local DB2 installation.

On z/OS only, your system administrator can determine the name of your DB2 location using the `DISPLAY DDF` command.

On DB2 for i only, your system administrator can determine the name of your DB2 location using the `WRKRDBDIRE` command. The name of the database that is listed as `*LOCAL` is the value you should use.

- **Collection:** This option is valid only if you are connecting to a DB2 database on DB2 for i or z/OS. By default, the user ID is used for the value of `Collection`. The user ID should always be used on z/OS.

Greenplum Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[Greenplum Wire Protocol]
...
Driver=ODBCHOME/lib/xxgplmnn.zz
...
Database=Gplumdb1
...
HostName=GreenplumServer
...
PortNumber=5432
...
```

Connection option descriptions:

- **Database:** The name of the database to which you want to connect by default.
- **Hostname:** Either the name or the IP address of the server to which you want to connect.
- **PortNumber:** The port number of the server listener. The default is 5432.

Impala Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[Impala Wire Protocol]
Driver=ODBCHOME/lib/xximpala.zz
...
Database=Impala1
...
HostName=ImpalaServer
...
PortNumber=21050
...
```

Connection option descriptions:

- **Database:** The name of the database to which you want to connect by default.
- **Hostname:** Either the name or the IP address of the server to which you want to connect.
- **PortNumber:** The port number of the server listener. The default is 21050.

Informix

Prerequisites:

- You must have all components of your database client software installed and connecting properly; otherwise, the driver will not operate correctly.

The following example demonstrates the minimum connection information required to establish a connection:

```
[Informix]
Driver=ODBCHOME/lib/ivinfxx.nn
...
Database=Informix3
...
HostName=InformixHost
...
ServerName=InformixServer
...
Service=online
...
```

Connection option descriptions:

- **Database:** The name of the database to which you want to connect by default.
- **HostName:** The name of the machine on which the Informix server resides.
- **ServerName:** The name of the Informix server as it appears in the `sqlhosts` file.
- **Service:** The name of the service as it appears on the host machine.

Informix Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[Informix Wire Protocol]
Driver=ODBCHOME/lib/xxifclnn.zz
...
Database=Informix3
...
HostName=InformixHost
...
LogonID=JohnD
...
PortNumber=1500
...
ServerName=InformixServer
...
```

Connection option descriptions:

- **Database:** The name of the database to which you want to connect by default.
- **Hostname:** Either the name or the IP address of the server to which you want to connect.
- **LogonID:** Your user name as specified on the Informix server.
- **PortNumber:** The port number of the server listener.
- **ServerNumber:** The name of the Informix server as it appears in the `sqlhosts` file.

MySQL Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[MySQL Wire Protocol]
Driver=ODBCHOME/lib/xxmysqlnn.zz
...
Database=MySQL3
...
HostName=MySQLHost
...
PortNumber=3306
...
```

Connection option descriptions:

- **Database:** The name of the database to which you want to connect by default.
- **Hostname:** Either the name or the IP address of the server to which you want to connect.
- **PortNumber:** The port number of the server listener. The default is 3306.

Oracle

Prerequisites:

- You must have all components of your database client software installed and connecting properly; otherwise, the driver will not operate correctly.

The following example demonstrates the minimum connection information required to establish a connection:

```
[Oracle]
Driver=ODBCHOME/lib/ivor8xx.nn
ServerName=OracleServer
```

Connection option descriptions:

- **ServerName:** The client connection string designating the server and database to be accessed. The information required varies depending on the client driver you are using.

Oracle Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[Oracle Wire Protocol]
Driver=ODBCHOME/lib/xxorann.zz
...
EditionName=oracle 1
...
HostName=199.226.224.34
...
PortNumber=1521
...
ServiceName=TEST
...
```

Connection option descriptions:

- **Hostname:** Either the name or the IP address of the server to which you want to connect.
- **PortNumber:** The port number of your Oracle listener. Check with your database administrator for the number.
- **ServiceName:** The Oracle service name that specifies the database used for the connection. The service name is a string that is the global database name—a name that is comprised of the database name and domain name, for example: sales.us.acme.com.
- **SID:** The Oracle System Identifier that refers to the instance of Oracle running on the server. The default is ORCL.

Note: *SID* and *ServiceName* are mutually exclusive. Only one or the other can be specified in the data source; otherwise, an error is generated.

- **EditionName:** Oracle 11g R2 and higher only. The name of the Oracle edition the driver uses when establishing a connection. Oracle 11g R2 and higher allows your database administrator to create multiple editions of schema objects so that your application can still use those objects while the database is being upgraded. This option is only valid for Oracle 11g R2 and higher databases and tells the driver which edition of the schema objects to use.

PostgreSQL Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[PostgreSQL Wire Protocol]
Driver=ODBCHOME/lib/xxpsqlnn.zz
...
Database=Pgredb1
...
HostName=PostgreSQLServer
...
PortNumber=5432
...
```

Connection option descriptions:

- **Database:** The name of the database to which you want to connect by default.
- **Hostname:** Either the name or the IP address of the server to which you want to connect.
- **PortNumber:** The port number of the server listener. The default is 5432

Progress OpenEdge Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[Progress OpenEdge Wire Protocol]
Driver=ODBCHOME/lib/xxoenn.zz
...
Database=odbl
...
HostName=OpenEdgeServer
...
PortNumber=5432
...
```

Connection option descriptions:

- **Database:** The name of the database to which you want to connect by default.
- **Hostname:** Either the name or the IP address of the server to which you want to connect.
- **PortNumber:** The port number of the server listener.

Salesforce

Prerequisites:

- The driver requires a Java Virtual Machine (JVM): Java SE 8 or higher. Before you configure a data source for the Salesforce driver, you must set the library path environment variable for your operating system to the directory containing your JVM's `libjvm.so` [`s1` | `a`] file, and that directory's parent directory. The library path environment variable is:
 - `LD_LIBRARY_PATH` on Linux, Oracle Solaris, and HP-UX Itanium
 - `SHLIB_PATH` on HP-UX PA-RISC
 - `LIBPATH` on AIX

The following example demonstrates the minimum connection information required to establish a connection:

```
[Salesforce]
Driver=ODBCHOME/lib/ivsfrc27.so
...
HostName=test.salesforce.com
...
UserName=JohnDoe
...
Password=secret
...
```

Connection option descriptions:

- **HostName:** The default Salesforce instance is `login.salesforce.com`. If you are logging into a different Salesforce instance, type the root of the Salesforce URL.
- **UserName:** Type your logon ID for Salesforce.
- **Password:** Type your case-sensitive password for the Salesforce instance.

If your Salesforce instance requires a security token, you can append it to the password, for example, `secretXaBARTsLZReM4Px47qPLOS`, where `secret` is the password and the remainder of the value is the security token. Both the password and security token are case-sensitive.

- **FetchSize:** Smaller fetch sizes can improve the initial response time of the query. Larger fetch sizes improve overall fetch times at the cost of additional memory. Smaller fetch sizes can improve the initial response time of the query. Larger fetch sizes improve overall fetch times at the cost of additional memory.
- **WSfetchSize:** If set to 0, the driver attempts to fetch up to a maximum of 2000 rows. This value typically provides the maximum throughput. Setting the value lower than 2000 can reduce the response time for returning the initial data.

SQL Server Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[SQLServer1 Wire Protocol]
Driver=ODBCHOME/lib/xsqslnn.zz
...
HostName=123.123.0.12
...
Database=master
...
PortNumber=1433
...
```

Connection option descriptions:

- **HostName:** The name or the IP address of the server to which you want to connect.
If your network supports named servers, you can specify an address as: *server_name*. For example, you can enter *SSserver*.
You can also specify a named instance of Microsoft SQL Server. Specify this address as: *server_name\instance_name*. If only a server name is specified with no instance name, the driver uses the default named instance on the server.
- **Database:** The name of the database to which you want to connect by default.
- **PortNumber:** The port number of the server listener. The default is 1433.

Sybase IQ Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[Sybase IQ Wire Protocol]
Driver=ODBCHOME/lib/xsxiqnn.zz
...
Database=master
...
NetworkAddress=123.226.224.34,2638
...
```

Connection option descriptions:

- **Database:** The name of the database to which you want to connect by default. If you do not specify a value, the default is the database defined by the system administrator for each user.
- **NetworkAddress:** The IP address of the server to which you want to connect. Specify this address as: *IP_address, port_number*. For example, you can enter *123.226.224.34, 2638*.
If your network supports named servers, you can specify an address as: *server_name, port_number*. For example, you can enter *SyIQserver, 2638*.
- **User Name:** The default user ID that is used to connect to your database. Your ODBC application may override this value.

Sybase Wire Protocol

The following example demonstrates the minimum connection information required to establish a connection:

```
[Sybase Wire Protocol]
Driver=ODBCHOME/lib/xxasenn.zz
...
Database=master
...
NetworkAddress=123.226.224.34,5000
...
```

Connection option descriptions:

- **Database:** The name of the database to which you want to connect by default. If you do not specify a value, the default is the database defined by the system administrator for each user.
- **NetworkAddress:** The IP address of the server to which you want to connect. Specify this address as: *IP_address, port_number*. For example, you can enter *123.226.224.34, 5000*.

If your network supports named servers, you can specify an address as: *server_name, port_number*. For example, you can enter *SSserver, 5000*.

Driver for the Teradata Database

Prerequisites:

- You must have all components of your database client software installed and connecting properly; otherwise, the driver will not operate correctly.

The following example demonstrates the minimum connection information required to establish a connection:

```
[Teradata]
Driver=ODBCHOME/lib/xxterann.zz
...
DBCName=123.123.12.12
...
SecurityMechanism=TD2
...
SecurityParameter=5678
...
UserID=John
...
```

Connection option descriptions:

- **DBCName:** The IP address or the alias name of the Teradata Server. Using an IP address reduces the time it takes to connect, but if that address is not available at connection time, the connection fails and the driver does not attempt to fail over to another address.

Using an alias name increases the time it takes to connect because the driver must search a local hosts file to resolve the name to the IP address information, but it allows the driver to try and connect to alternate IP addresses if the first address fails. If you use an alias name, you must have or create a local hosts file that contains the alias names. The alias name cannot be more than eight characters long.

- **SecurityMechanism:** Enter TD2.
- **SecurityParameter:** A string of characters that is to be regarded as a parameter to the authentication mechanism. The string is ignored by the ODBC driver and is passed on to the TeraSSO function that is called to set the authentication mechanism. The characters `[] { } () , ; ? * = ! @` must be enclosed in curly braces.
- **UserID:** The default UserID for the Teradata database.

Testing the Connection

To test the connection (GUI environment):

1. After you have configured the data source, you can click **Test Connect** on the Setup dialog box to attempt to connect to the data source using the connection options specified in the dialog box. Some drivers immediately return a message indicating success or failure. For most drivers, a logon dialog box appears as described in each individual driver chapter.
2. Supply the requested information in the logon dialog box and click **OK**. Note that the information you enter in the logon dialog box during a test connect is not saved.
 - If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.
3. On the driver Setup dialog box, click **OK**. The values you have specified are saved and are the defaults used when you connect to the data source. You can change these defaults by using the previously described procedure to modify your data source. You can override these defaults by connecting to the data source using a connection string with alternate values. See individual driver chapters for information about using connection strings.

Using the Performance Wizard

The Performance Wizard leads you step-by-step through a series of questions about your application. Based on your answers, the Wizard provides the optimal settings for performance-related connection string options. The Wizard applies to the following drivers:

- DB2 Wire Protocol
- Informix Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol
- SQL Server Wire Protocol
- Sybase Wire Protocol
- Oracle

The Wizard runs as an applet within a browser window. The browser must be configured to run applets. Refer to your browser's documentation for instructions on configuring your browser.

Note: Security features set in your browser can prevent the Performance Wizard from launching. If this is the case, a security warning message is displayed. Often, the warning message provides instructions for unblocking the Performance Wizard for the current session. To allow the Performance Wizard to launch without encountering a security warning message, the security settings in your browser can be modified. Check with your system administrator before disabling any security features.

Starting the Wizard

You can start the Wizard in the following ways:

- On Windows, you can start the Wizard by selecting it from the product program group.
- On all platforms, you can start the Wizard by launching the following file from your browser window, where *install_dir* is your product installation directory:

```
install_dir/wizards/index.html
```

Tuning Performance Using the Wizard

After you start the Wizard, a Welcome window appears. Click **Start** to start the process and select a driver.

The following is an example of one of the questions you may be asked to answer for the DB2 Wire Protocol driver.

Progress | DataDirect Connect THE WORLD LEADER IN DATA CONNECTIVITY PROGRESS SOFTWARE

PERFORMANCE WIZARD
DataDirect Connect® for ODBC
DataDirect Connect64® for ODBC

DB2 Wire Protocol

Choose Driver

Stored Procedures

Connection Pooling

Multi-Threaded Application

Failover

Encryption

Result

Do you need to access database objects (such as tables or stored procedures) that are grouped in different schemas (as opposed to accessing objects that are contained in a single schema)?

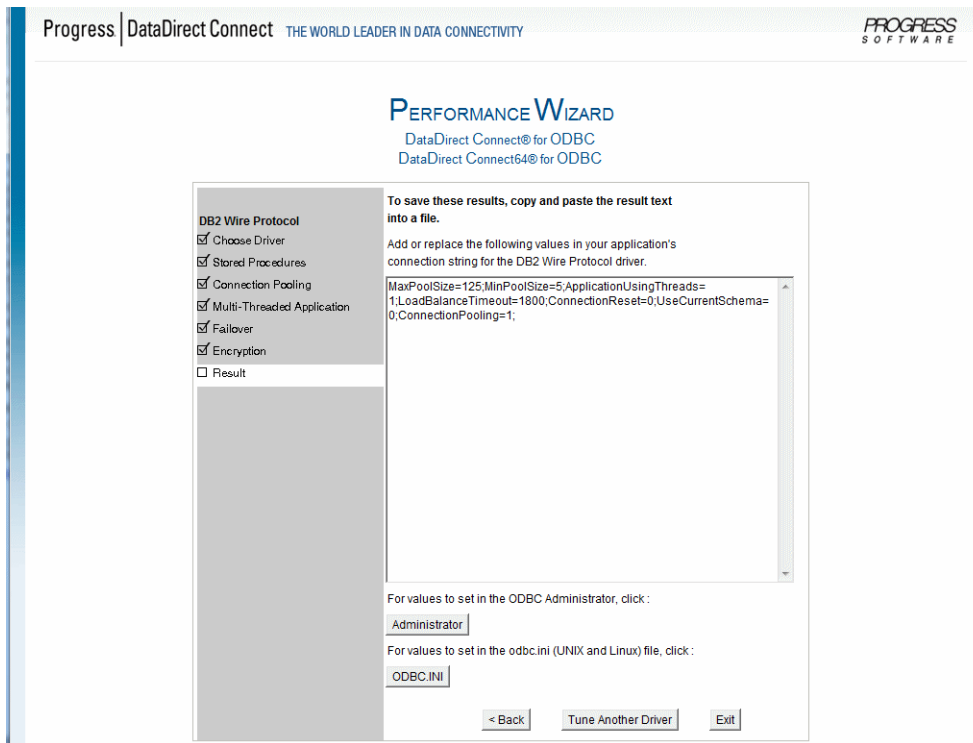
Yes

No

Detail:

Applicable connection string attribute: UseCurrentSchema. If your application needs to access database objects owned only by the current user, performance of your application can be improved. In this case, the UseCurrentSchema attribute should be enabled (set to 1). When this attribute is enabled, the driver returns only database objects owned by the current user when executing catalog functions. Calls to catalog functions are optimized by grouping queries. Enabling this attribute is equivalent to passing the Logon ID used on the connection as the SchemaName argument to the catalog functions.

When you have answered all questions for a driver, the results appear in the form of a connection string, as shown in the following example:



You can copy these results to an existing connection string for immediate use or to a text file for later reference.

You can also either click **Administrator**, if you are using the Windows or Linux ODBC administrator, or **ODBC.INI**, if you are editing the `odbc.ini` file. Clicking either of these buttons displays a window that provides the values to use for configuring a data source.

UNIX® See [Data Source Configuration](#) on page 114 for details about configuring data sources through the `odbc.ini` file.

General Information on Using Connect Drivers

For details, see the following topics:

- [About the Product](#)
- [Environment-Specific Information](#)
- [Using IP Addresses](#)
- [Binding Parameter Markers](#)
- [Driver Threading Information](#)
- [Version String Information](#)
- [Retrieving Data Type Information](#)
- [Persisting a Result Set as an XML Data File](#)
- [Translators](#)

About the Product

The DataDirect Connect Series *for* ODBC drivers are compliant with the Open Database Connectivity (ODBC) specification and compatible with ODBC 3.8 applications.

Progress DataDirect provides ODBC drivers for both relational and flat-file database systems. The flat-file drivers provide full SQL support; see [SQL Statements for Flat-File Drivers](#) for details.

Support for Multiple Environments

Progress DataDirect provides ODBC-compliant database drivers for Windows, UNIX, and Linux operating systems. See [Environment-Specific Information](#) for an explanation of the environment-specific differences when using the database drivers in your operating environment.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

Environment-Specific Information

The sections [For Windows Users](#) on page 58 and [For UNIX and Linux Users](#) on page 60 contain information specific to your operating environment.

The following sections refer to threading models.

Refer to "Threading" in the *Progress DataDirect for ODBC Drivers Reference* for more information.

For Windows Users

The following are requirements for the 32- and 64-bit drivers on Windows operating systems.

32-Bit Drivers

- All required network software that is supplied by your database system vendors must be 32-bit compliant.
- If your application was built with 32-bit system libraries, you must use 32-bit drivers. If your application was built with 64-bit system libraries, you must use 64-bit drivers (see [64-Bit Drivers](#) on page 59). The database to which you are connecting can be either 32-bit or 64-bit enabled.
- The following processors are supported:
 - x86: Intel
 - x64: Intel and AMD
- The following operating systems are supported for DataDirect Connect *for* ODBC. All editions are supported unless otherwise noted.
 - Windows 10
 - Windows 8.1
 - Windows Server 2012
 - Windows 7
 - Windows Server 2008

- The following operating systems are supported for DataDirect Connect XE *for* ODBC. All editions are supported unless otherwise noted.
 - Windows 10
 - Windows 8.1
 - Windows Server 2012
 - Windows 7
 - Windows Server 2008 Enterprise, Datacenter, Web and Small Business Editions
- For the Salesforce driver: A 32-bit Java Virtual Machine (JVM), Java SE 8 or higher, is required. Also, you must set the PATH environment variable to the directory containing your 32-bit JVM's `jvm.dll` file, and that directory's parent directory.
- An application that is compatible with components that were built using Microsoft Visual Studio 2010 compiler and the standard Win32 threading model.
- You must have ODBC header files to compile your application. For example, Microsoft Visual Studio includes these files.

64-Bit Drivers

- All required network software that is supplied by your database system vendors must be 64-bit compliant.
- The following processors are supported:
 - Intel
 - AMD
- The following operating systems are supported for DataDirect Connect64 *for* ODBC. All editions are supported unless otherwise noted.
 - Windows 10
 - Windows 8.1
 - Windows Server 2012
 - Windows 7
 - Windows Server 2008
- The following operating systems are supported for DataDirect Connect64 XE *for* ODBC. All editions are supported unless otherwise noted.
 - Windows 10
 - Windows 8.1
 - Windows Server 2012
 - Windows 7
 - Windows Server 2008 Enterprise, Standard, or Datacenter Editions

- An application that is compatible with components that were built using Microsoft C/C++ Optimizing Compiler Version 14.00.40310.41 and the standard Windows 64 threading model.
- For the Salesforce driver: A 64-bit JVM, Java SE 8 or higher, is required. Also, you must set the PATH environment variable to the directory containing your 32-bit JVM's jvm.dll file, and that directory's parent directory.
- You must have ODBC header files to compile your application. For example, Microsoft Visual Studio includes these files.

Setup of the Drivers

The driver must be configured before it can be used. See [Quick Start Connect](#) on page 33 for information about using the Windows ODBC Administrator. See the individual driver chapters for details about driver configuration.

Driver Names

The prefix for all 32-bit driver file names is IV. The prefix for all 64-bit driver file names is DD. The file extension is .DLL, which indicates dynamic link libraries. For example, the 32-bit DB2 Wire Protocol driver file name is IVDB2 nn .DLL, where nn is the revision number of the driver.

Refer to the readme file shipped with the product for the file name of each driver.

For UNIX and Linux Users

UNIX[®] The following are requirements for the 32- and 64-bit drivers on UNIX/Linux operating systems.

32-Bit Drivers

- All required network software that is supplied by your database system vendors must be 32-bit compliant.
- If your application was built with 32-bit system libraries, you must use 32-bit drivers. If your application was built with 64-bit system libraries, you must use 64-bit drivers (see [64-Bit Drivers](#) on page 63). The database to which you are connecting can be either 32-bit or 64-bit enabled.
- For the Salesforce driver: A 32-bit Java Virtual Machine (JVM), Java SE 8 or higher, is required. Also, you must set the library path environment variable of your operating system to the directory containing your JVM's libjvm.so [sl | a] file and that directory's parent directory.

The library path environment variable is:

- LD_LIBRARY_PATH on Linux, HP-UX Itanium, and Oracle Solaris
- SHLIB_PATH on HP-UX PA-RISC
- LIBPATH on AIX

AIX

- IBM POWER processor
- AIX 5L operating system, version 5.3 fixpack 5 and higher, 6.1, and 7.1
- An application compatible with components that were built using Visual Age C++ 6.0.0.0 and the AIX native threading model

Note: SALESFORCE USERS: When compiling an application on AIX for use with the driver for Salesforce, you must **not** use the `-brtl` option.

Note: TERADATA USERS: When compiling an application on AIX for use with the driver for the Teradata database, you must use the `-brtl` option. For example, use `cc -o pgm pgm.o -brtl -lodbc` or `ld -o pgm -brtl pgm.o -lodbc`

HP-UX

- The following processors are supported:
 - PA-RISC
 - Intel Itanium II (IPF)
- The following operating systems are supported:
 - For PA-RISC: HP-UX 11i Versions 2 and 3 (B.11.23 and B.11.3x), 11i (B.11.11), and 11
 - For IPF: HP-UX IPF 11i Versions 2 and 3 (B.11.23 and B.11.3x)
- For PA-RISC: An application compatible with components that were built using HP aC++ 3.30 and the HP-UX 11 native (kernel) threading model (posix draft 10 threads).
All of the standard 32-bit UNIX drivers are supported on HP PA-RISC.
- For IPF: An application compatible with components that were built using HP aC++ 5.36 and the HP-UX 11 native (kernel) threading model (posix draft 10 threads)

Note: All of the standard 32-bit UNIX drivers are supported on HP PA-RISC.

For IPF, the following DataDirect Connect *for* ODBC are supported:

- DB2 Wire Protocol
- Informix Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol
- PostgreSQL Wire Protocol
- Progress OpenEdge Wire Protocol
- SQL Server Wire Protocol
- Sybase Wire Protocol
- Oracle
- SQL Server Legacy Wire Protocol

The following DataDirect Connect XE *for* ODBC drivers are supported:

- Driver for Apache Hive
- Greenplum
- Impala Wire Protocol
- Salesforce

- Sybase IQ Wire Protocol

Considerations for Salesforce users:

- PA-RISC: Set the `LD_PRELOAD` environment variable to the `libjvm.sl` from your JVM installation.
- Itanium:
 - Do not link with the `-lc` linker option.
 - Set the `LD_PRELOAD` environment variable to the `libjvm.so` from your JVM installation.

Linux

- The following processors are supported:
 - x86: Intel
 - x64: Intel and AMD
- The following operating systems are supported:
 - CentOS Linux 4.x, 5.x, 6.x, 7.x, and 8.x
 - Debian 7.11 and 8.5
 - Oracle Linux 4.x, 5.x, 6.x, 7.x, and 8.x
 - Red Hat Enterprise Linux AS, ES, and WS version 4.x, 5.x, 6.x, 7.x, and 8.x
 - SUSE Linux Enterprise Server 10.x, 11
 - Ubuntu 14.04, 16.04, and 18.04
- An application compatible with components that were built using g++ GNU project C++ Compiler version 3.4.6 and the Linux native pthread threading model (Linuxthreads).

Note: All drivers are supported on Linux except for the Informix driver.

Oracle Solaris

- The following processors are supported:
 - Oracle SPARC
 - x86: Intel
 - x64: Intel and AMD
- The following operating systems are supported:
 - For Oracle SPARC: Oracle Solaris 8, 9, 10, 11.x
 - For x86/x64: Oracle Solaris 10, Oracle Solaris 11.x
- For Oracle SPARC: An application compatible with components that were built using Oracle Workshop v. 6 update 2 and the Solaris native (kernel) threading model.
- For x86/x64: An application compatible with components that were built using Oracle C++ 5.8 and the Solaris native (kernel) threading model

Note: All of the standard 32-bit UNIX drivers are supported on Solaris SPARC.

For x86, the following DataDirect Connect *for* ODBC drivers are supported:

- DB2 Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol
- PostgreSQL Wire Protocol
- SQL Server Wire Protocol
- Sybase Wire Protocol
- SQL Server Legacy Wire Protocol

The following DataDirect Connect XE *for* ODBC drivers are supported:

- Driver for Apache Hive
- Impala Wire Protocol
- Salesforce (64-bit platforms only)
- Sybase IQ Wire Protocol

64-Bit Drivers

All required network software that is supplied by your database system vendors must be 64-bit compliant.

- For the Salesforce driver: A 64-bit Java Virtual Machine (JVM), Java SE 8 or higher, is required. Also, you must set the library path environment variable of your operating system to the directory containing your JVM's `libjvm.so` [`sl` | `a`] file and that directory's parent directory.
- The library path environment variable is:
 - `LD_LIBRARY_PATH` on Linux, HP-UX Itanium, and Oracle Solaris
 - `LIBPATH` on AIX

AIX

- IBM POWER Processor
- AIX 5L operating system, version 5.3 fixpack 5 and higher, 6.1, and 7.1
- An application compatible with components that were built using Visual Age C++ version 6.0.0.0 and the AIX native threading model

Note: SALESFORCE USERS: When compiling an application on AIX for use with the driver for Salesforce, you must not use the `-brtl` option.

HP-UX

- Intel Itanium II (IPF) processor
- HP-UX IPF 11i operating system, Versions 2 and 3 (B.11.23 and B.11.31)
- HP aC++ v. 5.36 and the HP-UX 11 native (kernel) threading model (posix draft 10 threads)

The following drivers are supported on IPF:

<p>DataDirect Connect64 <i>for</i> ODBC drivers</p>	<ul style="list-style-type: none"> • DB2 Wire Protocol • Informix Wire Protocol • MySQL Wire Protocol • Oracle Wire Protocol • PostgreSQL Wire Protocol • Progress OpenEdge Wire Protocol • SQL Server Wire Protocol • Sybase Wire Protocol • Oracle • SQL Server Legacy Wire Protocol • Text
<p>DataDirect Connect64 XE <i>for</i> ODBC drivers</p>	<ul style="list-style-type: none"> • Driver for Apache Hive • Greenplum Wire Protocol • Impala Wire Protocol • Salesforce • Sybase IQ Wire Protocol • Teradata

NOTES FOR SALESFORCE:

- Do not link with the `-lc` linker option.
- Set the `LD_PRELOAD` environment variable to the `libjvm.so` of your JVM installation.

Linux

- The following processors are supported:
 - Intel Itanium II (IPF)
 - x64: Intel and AMD
- The following operating systems are supported:
 - For Itanium II:
 - Red Hat Enterprise Linux AS, ES, and WS versions 4.x, 5.x, 6.x, 7.x, and 8.x
 - For x64:
 - CentOS Linux 4.x, 5.x, 6.x, 7.x, and 8.x
 - Debian 7.11 and 8.5
 - Oracle Linux 4.x, 5.x, 6.x, 7.x, and 8.x

- Red Hat Enterprise Linux AS, ES, and WS version 4.x, 5.x, 6.x, 7.x, and 8.x
- SUSE Linux Enterprise Server 10.x, 11
- Ubuntu 14.04, 16.04, and 18.04

Note: The Oracle (client) driver is not supported on the Red Hat x64 operating system.

- For Itanium II: an application compatible with components that were built using g++ GNU project C++ Compiler version 3.3.2 and the Linux native pthread threading model (Linuxthreads)
- For x64: an application compatible with components that were built using g++ GNU project C++ Compiler version 3.4 and the Linux native pthread threading model (Linuxthreads)

Note: The Greenplum Wire Protocol driver is the only Connect XE driver supported on Linux Itanium. II.

Oracle Solaris

- The following processors are supported:
 - Oracle SPARC
 - x64: Intel and AMD
- The following operating systems are supported:
 - For Oracle SPARC: OracleSolaris 8, 9, 10, and 11.x
 - For x64: OracleSolaris 10 and Oracle Solaris 11.x Express
- For Oracle SPARC: An application compatible with components that were built using Oracle Workshop v. 6 update 2 and the Solaris native (kernel) threading model
- For x64: An application compatible with components that were built using Oracle C++ Compiler version 5.8 and the Solaris native (kernel) threading model

All of the standard 32-bit UNIX drivers are supported on Solaris SPARC. For x64, The following drivers are supported for Oracle Solaris:

DataDirect Connect <i>for</i> ODBC drivers	<ul style="list-style-type: none"> • DB2 Wire Protocol • MySQL Wire Protocol • Oracle Wire Protocol • PostgreSQL Wire Protocol • SQL Server Wire Protocol • Sybase Wire Protocol • SQL Server Legacy Wire Protocol
DataDirect Connect XE <i>for</i> ODBC drivers	<ul style="list-style-type: none"> • Driver for Apache Hive • Greenplum Wire Protocol • Impala Wire Protocol • Salesforce (64-bit platforms only) • Sybase IQ Wire Protocol

AIX

If you are building 64-bit binaries, you must pass the define ODBC64. Demoodbc provides an example of this. See the installed file demoodbc.txt and [The demoodbc Application](#) for details.

You must also include the correct compiler switches if you are building 64-bit binaries. For example, to build demoodbc, you would use:

```
xlC_r -DODBC64 -q64 -qlonglong -qlongdouble -qvftable -o demoodbc
-I../include demoodbc.c -L../lib -lc_r -lC_r -lodbc
```

HP-UX 11 aCC

The ODBC drivers require certain runtime library patches. The patch numbers are listed in the readme file for your product. HP-UX patches are publicly available from the HP Web site www.hp.com.

HP updates the patch database regularly; therefore, the patch numbers in the readme file may be superseded by newer versions. If you search for the specified patch on an HP site and receive a message that the patch has been superseded, download and install the replacement patch.

If you are building 64-bit binaries, you must pass the define ODBC64. Demoodbc provides an example of this. See the installed file demoodbc.txt and [The demoodbc Application](#) for details. You must also include the +DD64 compiler switch if you are building 64-bit binaries. For example, to build demoodbc, you would use:

```
aCC -Wl,+s +DD64 -DODBC64 -o demoodbc -I../include demoodbc.c -L../lib -lodbc
```

Linux

If you are building 64-bit binaries, you must pass the define ODBC64. Demoodbc provides an example of this. See the installed file demoodbc.txt and [The demoodbc Application](#) for details.

You must also include the correct compiler switches if you are building 64-bit binaries. For example, to build demoodbc, you would use:

```
g++ -o demoodbc -DODBC64 -I../include demoodbc.c -L../lib -lodbc -lodbcinst -lc
```

Oracle Solaris

If you are building 64-bit binaries, you must pass the define ODBC64. Demoodbc provides an example of this. See the installed file demoodbc.txt and [The demoodbc Application](#) for details.

You must also include the -xarch=v9 compiler switch if you are building 64-bit binaries. For example, to build demoodbc, you would use:

```
CC -mt -DODBC64 -xarch=v9 -o demoodbc -I../include demoodbc.c -L../lib -lodbc -lCrun
```

Setup of the Environment and the Drivers

On UNIX and Linux, several environment variables and the system information file must be configured before the drivers can be used. See [Configuring and Connecting on UNIX and Linux](#) for a brief description of these variables and information about using the DataDirect ODBC Data Source Administrator for Linux. See the individual driver chapters for details about driver configuration. See [Configuring the Product on UNIX/Linux](#) for complete information about using the drivers on UNIX and Linux.

Driver Names

The drivers are ODBC API-compliant dynamic link libraries, referred to in UNIX and Linux as shared objects. The prefix for all 32-bit driver file names is iv. The prefix for all 64-bit driver file names is dd. The driver file names are lowercase and the extension is .so, the standard form for a shared object. For example, the 32-bit DB2 Wire Protocol driver file name is ivdb2nn.so, where nn is the revision number of the driver. For drivers on HP-UX PA-RISC only, the extension is .sl, for example, ivdb2nn.sl.

Refer to the readme file shipped with your DataDirect product for the file name of each driver.

Using IP Addresses

The drivers support Internet Protocol (IP) addresses in IPv4 and IPv6 format as shown in the following tables.

Table 1: IP Address Formats Supported by 32- and 64-bit DataDirect Connect for ODBC Drivers

Driver	IPv4	IPv6
DB2 Wire Protocol	All supported versions	All supported versions
Informix Wire Protocol	All supported versions	Informix 10 and higher
MySQL Wire Protocol	All supported versions	Not supported

Driver	IPv4	IPv6
Oracle Wire Protocol	All supported versions	Oracle 11gR2
Microsoft SQL Server Legacy Wire Protocol (UNIX/Linux only)	All supported versions	All supported versions
Microsoft SQL Server Wire Protocol	All supported versions	All supported versions
PostgreSQL Wire Protocol	All supported versions	All supported versions
Progress OpenEdge Wire Protocol	All supported versions	All supported versions
Sybase Wire Protocol	All supported versions	Sybase 12.5.2 and higher

Table 2: IP Address Formats Supported by 32- and 64-bit DataDirect Connect XE for ODBC Drivers

Driver	IPv4	IPv6
Driver for Apache Hive	All supported versions	Not supported
Greenplum Wire Protocol	All supported versions	All supported versions
Impala Wire Protocol	All supported versions	Not supported
Salesforce	All supported versions	All supported versions
Sybase IQ Wire Protocol	All supported versions	All supported versions

If your network supports named servers, the server name specified in the data source can resolve to an IPv4 or IPv6 address.

In the following connection string example, the IP address for the DB2 server is specified in IPv4 format:

```
DRIVER=DataDirect DB2 Wire Protocol;
IpAddress=123.456.78.90;PORT=5179;
DB=DB2ACCT;UID=JOHN;PWD=XYZZYYou
```

In the following connection string example, the IP address for the DB2 server is specified in IPv6 format:

```
DRIVER=DataDirect DB2 Wire Protocol;
IpAddress=2001:DB8:0000:0000:8:800:200C:417A;PORT=5179;
DB=DB2ACCT;UID=JOHN;PWD=XYZZYYou
```

In addition to the normal IPv6 format, the drivers in the preceding tables support IPv6 alternative formats for compressed addresses. For example, the following connection string specifies the server using IPv6 format, but uses the compressed syntax for strings of zero bits:

```
DRIVER=DataDirect DB2 Wire Protocol;
IpAddress=2001:DB8:0:0:8:800:200C:417A;PORT=5179;
DB=DB2ACCT;UID=JOHN;PWD=XYZZYYou
```

For complete information about IPv6 formats, go to the following URL:

<http://tools.ietf.org/html/rfc4291#section-2.2>

Binding Parameter Markers

An ODBC application can prepare a query that contains dynamic parameters. Each parameter in a SQL statement must be associated, or bound, to a variable in the application before the statement is executed. When the application binds a variable to a parameter, it describes that variable and that parameter to the driver. Therefore, the application must supply the following information:

- The data type of the variable that the application maps to the dynamic parameter
- The SQL data type of the dynamic parameter (the data type that the database system assigned to the parameter marker)

The two data types are identified separately using the SQLBindParameter function. You can also use descriptor APIs as described in the Descriptor section of the ODBC specification (version 3.0 and higher).

The driver relies on the binding of parameters to know how to send information to the database system in its native format. If an application furnishes incorrect parameter binding information to the ODBC driver, the results will be unpredictable. For example, the statement might not be executed correctly.

To ensure interoperability, the DataDirect Connect *for* ODBC driver uses only the parameter binding information that is provided by the application. Some DBMSs cannot publish dynamic parameter information back to an ODBC driver. For example, both the Microsoft SQL Server and Oracle databases can determine that a parameter is an integer; however, the Oracle query process cannot publish this information back to the driver.

Driver Threading Information

The following tables summarize the threading information available at this time for the drivers. Always consult the readme file for the most up-to-date information as threading information is subject to change with new database transport and server revisions. Currently, the XML driver is the only thread-impaired driver.

Refer to "Threading" in the *Progress DataDirect for ODBC Drivers Reference* for more information.

Table 3: Threading Information for DataDirect Connect *for* ODBC

Driver	Fully Threaded	Thread Per Connect
Btrieve	X	
dBASE	X	
DB2 Wire Protocol		X
Informix Wire Protocol		X
Informix		X
MySQL Wire Protocol		X
Oracle Wire Protocol		X

Driver	Fully Threaded	Thread Per Connect
Oracle		X
PostgreSQL Wire Protocol		X
Progress OpenEdge		X
SQL Server Legacy Wire Protocol		X
SQL Server Wire Protocol		X
Sybase Wire Protocol		X
Text	X	

Table 4: Threading Information for DataDirect Connect XE for ODBC

Driver	Fully Threaded	Thread Per Connect
Driver for Apache Hive		X ¹
Greenplum Wire Protocol		X
Impala Wire Protocol		X
Salesforce		X
Sybase IQ Wire Protocol		X
Teradata	X	

Version String Information

All drivers, except the flat-file drivers and the Salesforce driver, have a version string of the format:

```
XX.YY.ZZZZ(bAAAA, uBBBB)
```

or

```
XX.YY.ZZZZbAAAA, uBBBB)
```

All flat-file drivers have a version string of the format:

```
XX.YY.ZZZZ(bAAAA, uBBBB, FCCCC)
```

The Salesforce driver has a version string of the format:

```
XX.YY.ZZZZ(bAAAA, UBBBB, SDDDDDD)
```

¹ Because Apache HiveServer1 servers do not currently handle multiple connections well, consider using a single Apache HiveServer1 for each connection.

The Driver Manager on UNIX and Linux has a version string of the format:

```
XX.YY.ZZZZ(UBBBB)
```

The component for the Unicode conversion tables (ICU) has a version string of the format:

```
XX.YY.ZZZZ
```

where:

XX is the major version of the product.

YY is the minor version of the product.

ZZZZ is the build number of the driver or ICU component.

AAAA is the build number of the driver's bas component.

BBBB is the build number of the driver's utl component.

CCCC is the build number of a flat-file driver's flt component.

DDDDDD is the version of the Java components used by the Salesforce driver.

For example:

```
07.16.0002 (b0001, u0002, F0001)
  |__|  |__|  |__|  |__|
  Driver Bas  Utl  Flt
```



On Windows, you can check the version string through the properties of the driver DLL. Right-click the driver DLL and select **Properties**. The Properties dialog box appears. On the Version tab, click **File Version** in the Other version information list box.

You can always check the version string of a driver on Windows by looking at the About tab of the driver's Setup dialog.

UNIX[®] On UNIX and Linux, you can check the version string by using the test loading tool shipped with the product. This tool, `ivtestlib` for 32-bit drives and `ddtestlib` for 64-bit drivers, is located in `install_directory/bin`.

The syntax for the tool is:

```
ivtestlib shared_object
```

or

```
ddtestlib shared_object
```

For example, for the 32-bit Oracle Wire Protocol driver on Oracle Solaris:

```
ivtestlib ivora27.so
```

returns:

```
07.16.0001 (B0002, U0001)
```

Note that the Oracle Wire Protocol driver is not a flat-file driver; therefore, there is no flt component listed in the example.

For example, for the Driver Manager on Solaris:

```
ivtestlib libodbc.so
```

returns:

```
07.16.0001 (U0001)
```

For example, for the 64-bit Driver Manager on Solaris:

```
ddtestlib libodbc.so
```

returns:

```
07.16.0001 (U0001)
```

For example, for 32-bit ICU component on Solaris:

```
ivtestlib libivicu27.so
07.16.0001
```

Note: On AIX, Linux, and Solaris, the full path to the driver does not have to be specified for the test loading tool. The HP-UX version of the tool, however, requires the full path.

getFileVersionString Function

Version string information can also be obtained programmatically through the function `getFileVersionString`. This function can be used when the application is not directly calling ODBC functions.

This function is defined as follows and is located in each driver's shared object:

```
const unsigned char* getFileVersionString();
```

This function is prototyped in the `qesqlx.h` file shipped with the product.

Retrieving Data Type Information

At times, you might need to get information about the data types that are supported by the data source, for example, precision and scale. You can use the ODBC function `SQLGetTypeInfo` to do this.

On Windows, you can use ODBC Test to call `SQLGetTypeInfo` against the ODBC data source to return the data type information.

Refer to "Diagnostic tools" in the *Progress DataDirect for ODBC Drivers Reference* for details about ODBC Test.

On UNIX, Linux, or Windows, an application can call `SQLGetTypeInfo`. Here is an example of a C function that calls `SQLGetTypeInfo` and retrieves the information in the form of a SQL result set.

```
void ODBC_GetTypeInfo(SQLHANDLE hstmt, SQLSMALLINT dataType)
{
    RETCODE rc;

    // There are 19 columns returned by SQLGetTypeInfo.
    // This example displays the first 3.
    // Check the ODBC 3.x specification for more information.
    // Variables to hold the data from each column
    char          typeName[30];
    short         sqlDataType;
    unsigned long columnSize;

    SQLINTEGER    strlenTypeName,
                 strlenSqlDataType,
```



```

        strlenColumnName;

    rc = SQLGetTypeInfo(hstmt, dataType);
    if (rc == SQL_SUCCESS) {

// Bind the columns returned by the SQLGetTypeInfo result set.
    rc = SQLBindCol(hstmt, 1, SQL_C_CHAR, &typeName,
        (SDWORD)sizeof(typeName), &strlenTypeName);
    rc = SQLBindCol(hstmt, 2, SQL_C_SHORT, &sqlDataType,
        (SDWORD)sizeof(sqlDataType), &strlenSqlDataType);
    rc = SQLBindCol(hstmt, 3, SQL_C_LONG, &columnName,
        (SDWORD)sizeof(columnName), &strlenColumnName);

// Print column headings
    printf ("TypeName          DataType          ColumnSize\n");
    printf ("-----\n");

    do {

// Fetch the results from executing SQLGetTypeInfo
        rc = SQLFetch(hstmt);
        if (rc == SQL_ERROR) {
// Procedure to retrieve errors from the SQLGetTypeInfo function
            ODBC_GetDiagRec(SQL_HANDLE_STMT, hstmt);
            break;
        }

// Print the results
        if ((rc == SQL_SUCCESS) || (rc == SQL_SUCCESS_WITH_INFO)) {
printf ("%30s %10i %10u\n", typeName, sqlDataType, columnName);
        }

    } while (rc != SQL_NO_DATA);
}
}

```

For information about how a database's data types map to the standard ODBC data types, see the appropriate driver chapter in this book.

Persisting a Result Set as an XML Data File

The DataDirect Connect Series *for* ODBC drivers allow you to persist a result set as an XML data file with embedded schema. To implement XML persistence, a client application must do the following:

1. Turn on STATIC cursors. For example:

```
SQLSetStmtAttr (hstmt, SQL_ATTR_CURSOR_TYPE, SQL_CURSOR_STATIC, SQL_IS_INTEGER)
```

Note: A result set can be persisted as an XML data file only if the result set is generated using STATIC cursors. Otherwise, the following error is returned:

Driver only supports XML persistence when using driver's static cursors.

2. Execute a SQL statement. For example:

```
SQLExecDirect (hstmt, "SELECT * FROM GTABLE", SQL_NTS)
```

3. Persist the result set as an XML data file. For example:

```
SQLSetStmtAttr (hstmt, SQL_PERSIST_AS_XML, "C:\temp\GTABLE.XML", SQL_NTS)
```

Note: A statement attribute is available to support XML persistence, `SQL_PERSIST_AS_XML`. A client application must call `SQLSetStmtAttr` with this attribute as an argument. See the following table for the definition of valid arguments for `SQLSetStmtAttr`.

Argument	Definition
<i>StatementHandle</i>	The handle of the statement that contains the result set to persist as XML.
<i>Attribute</i>	<code>SQL_PERSIST_AS_XML</code> . This statement attribute can be found in the file <code>qesqlxt.h</code> , which is installed with the driver.
<i>ValuePtr</i>	Pointer to a URL that specifies the full path name of the XML data file to be generated. The directory specified in the path name must exist, and if the specified file name exists, the file will be overwritten.
<i>StringLength</i>	The length of the string pointed to by <i>ValuePtr</i> or <code>SQL_NTS</code> if <i>ValuePtr</i> points to a NULL-terminated string.

A client application can choose to persist the data at any time that the statement is in an executed or cursor-positioned state. At any other time, the driver returns the following message:

Function Sequence Error

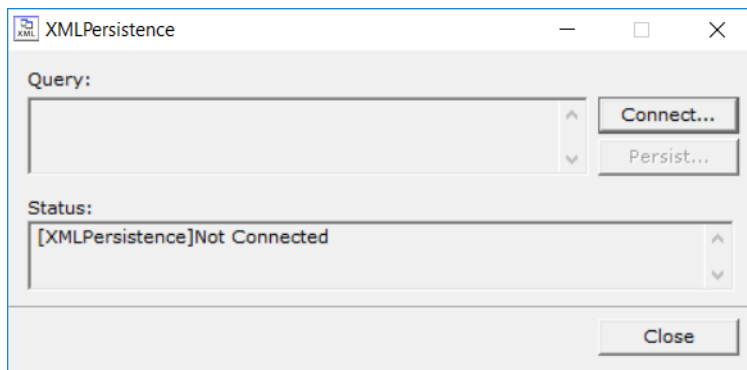
Using the Windows XML Persistence Demo Tool

The 32-bit drivers for Windows are shipped with an XML persistence demo tool. This tool is installed in the product installation directory.

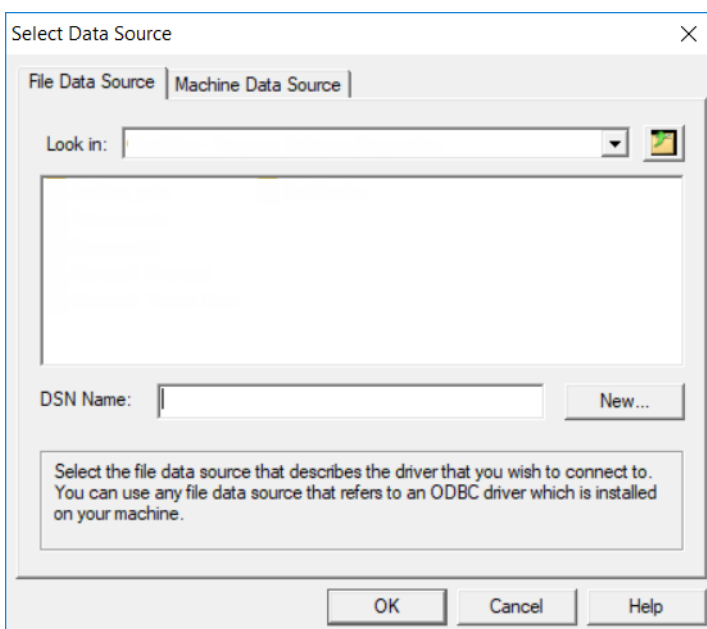
The tool has a graphical user interface and allows you to persist data as an XML data file.

To use the Windows XML Persistence Demo tool:

1. From the product program group, select **XML Persistence Demo**. The XMLPersistence dialog box appears.



2. First, you must connect to the database. Click **Connect**. The Select Data Source dialog box appears.



3. You must either select an existing data source or create a new one. Take one of the following actions:
 - Select an existing data source and click **OK**.
 - Create a new file data source by clicking **New**. The Create New Data Source dialog box appears. Follow the instructions in the dialog box.
 - Create a new machine data source by clicking the **Machine Data Source** tab and clicking **New**. The Create New Data Source dialog box appears. Follow the instructions in the dialog box.
4. After you have connected to a database, type a SQL Select statement in the Query text box of the XML Persistence dialog box. Then, click **Persist**. The Save As dialog box appears.
5. Specify a name and location for the XML data file that will be created. Then, click **OK**.
Note that the Status box in the XML Persistence dialog box displays whether the action failed or succeeded.
6. Click **Disconnect** to disconnect from the database.
7. Click **Close** to exit the tool.

Using the UNIX/Linux XML Persistence Demo Tool

UNIX[®] On UNIX and Linux, the drivers are shipped with an XML persistence demo tool named demoodbc. This tool is installed in the installation directory, in the /samples/demo subdirectory. For information about how to use this tool, refer to the demoodbc.txt file installed in the demo subdirectory.

Translators

Progress DataDirect provides a sample translator named "OEM to ANSI" that provides a framework for coding a translation library.



On Windows, refer to the readme.trn file in the \TRANSLAT subdirectory in the product installation directory.

UNIX[®] On UNIX and Linux, refer to the readme.trn file in the product installation directory, in the /samples/trn subdirectory.

Advanced Features

The drivers support many advanced features, including:

- [Using Failover](#) on page 78
- [Using Client Information](#) on page 87
- [Using DataDirect Connection Pooling](#) on page 97
- Industry-standard security features such as Secure Socket Layer (SSL) data encryption and Kerberos authentication provide secure transmission of data. See [Using Security](#) on page 89 for more information.
- [Using DataDirect Bulk Load](#) on page 101

For details, see the following topics:

- [Using Failover](#)
- [Using Client Information](#)
- [Using Security](#)
- [Using DataDirect Connection Pooling](#)
- [Using DataDirect Bulk Load](#)
- [Using Bulk Load for Batch Inserts](#)

Using Failover

To ensure continuous, uninterrupted access to data, the DataDirect Connect Series *for* ODBC drivers provide the following levels of failover protection, listed from basic to more comprehensive:

- *Connection failover* provides failover protection for new connections only. The driver fails over new connections to an alternate, or backup, database server if the primary database server is unavailable, for example, because of a hardware failure or traffic overload. If a connection to the database is lost, or dropped, the driver does not fail over the connection. This failover method is the default.
- *Extended connection failover* provides failover protection for new connections and lost database connections. If a connection to the database is lost, the driver fails over the connection to an alternate server, preserving the state of the connection at the time it was lost, but not any work in progress.
- *Select Connection failover* provides failover protection for new connections and lost database connections. In addition, it provides protection for Select statements that have work in progress. If a connection to the database is lost, the driver fails over the connection to an alternate server, preserving the state of the connection at the time it was lost and preserving the state of any work being performed by Select statements.

The method you choose depends on how failure tolerant your application is. For example, if a communication failure occurs while processing, can your application handle the recovery of transactions and restart them? Your application needs the ability to recover and restart transactions when using either extended connection failover mode or select connection failover mode. The advantage of select mode is that it preserves the state of any work that was being performed by the Select statement at the time of connection loss. If your application had been iterating through results at the time of the failure, when the connection is reestablished the driver can reposition on the same row where it stopped so that the application does not have to undo all of its previous result processing. For example, if your application were paging through a list of items on a Web page when a failover occurred, the next page operation would be seamless instead of starting from the beginning. Performance, however, is a factor in selecting a failover mode. Select mode incurs additional overhead when tracking what rows the application has already processed.

You can specify which failover method you want to use by setting the [Failover Mode](#) connection option. Read the following sections for details on each failover method:

- [Connection Failover](#) on page 78
- [Extended Connection Failover](#) on page 80
- [Select Connection Failover](#) on page 81

Regardless of the failover method you choose, you must configure one or multiple alternate servers using the [Alternate Servers](#) connection option. See [Guidelines for Primary and Alternate Servers](#) on page 82 for information about primary and alternate servers.

Connection Failover

Connection failover is available in the following DataDirect Connect Series *for* ODBC drivers:

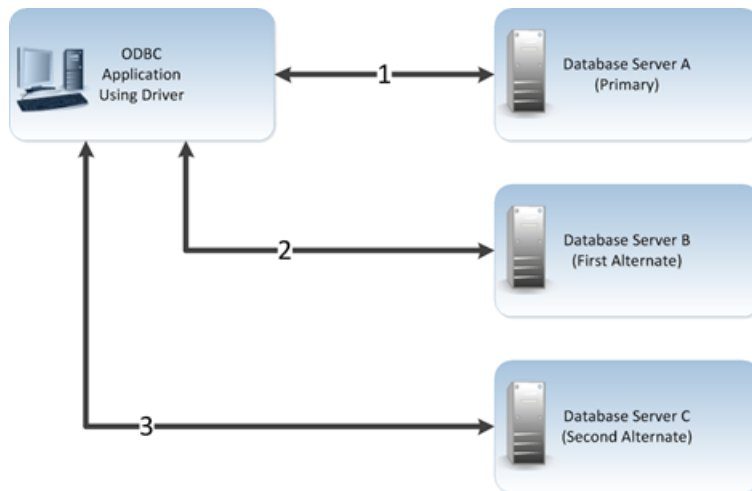
- DB2 Wire Protocol
- Greenplum Wire Protocol
- Informix Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol

- Oracle
- PostgreSQL Wire Protocol
- Progress OpenEdge Wire Protocol
- SQL Server Wire Protocol
- SQL Server Legacy Wire Protocol (UNIX only)
- Sybase Wire Protocol
- Sybase IQ Wire Protocol

Connection failover allows an application to connect to an alternate, or backup, database server if the primary database server is unavailable, for example, because of a hardware failure or traffic overload. Connection failover provides failover protection for new connections only and does not provide protection for lost connections to the database, nor does it preserve states for transactions or queries.

You can customize the drivers for connection failover by configuring a list of alternate database servers that are tried if the primary server is not accepting connections. Connection attempts continue until a connection is successfully established or until all the alternate database servers have been tried the specified number of times.

For example, suppose you have the environment shown in the following illustration with multiple database servers: Database Server A, B, and C. Database Server A is designated as the primary database server, Database Server B is the first alternate server, and Database Server C is the second alternate server.



First, the application attempts to connect to the primary database server, Database Server A (1). If connection failover is enabled and Database Server A fails to accept the connection, the application attempts to connect to Database Server B (2). If that connection attempt also fails, the application attempts to connect to Database Server C (3).

In this scenario, it is probable that at least one connection attempt would succeed, but if no connection attempt succeeds, the driver can retry each alternate database server (primary and alternate) for a specified number of attempts. You can specify the number of attempts that are made through the *connection retry* feature. You can also specify the number of seconds of delay, if any, between attempts through the *connection delay* feature. See [Using Connection Retry](#) on page 84 for more information about connection retry.

A driver fails over to the next alternate database server only if a successful connection cannot be established with the current alternate server. If the driver successfully establishes communication with a database server and the connection request is rejected by the database server because, for example, the login information is invalid, then the driver generates an error and does not try to connect to the next database server in the list. It is assumed that each alternate server is a mirror of the primary and that all authentication parameters and other related information are the same.

For details on configuring connection failover for your driver, see the appropriate driver chapter in this book.

Extended Connection Failover

Extended connection failover is available in the following DataDirect Connect Series *for* ODBC drivers:

- DB2 Wire Protocol
- Greenplum Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol
- PostgreSQL Wire Protocol
- Progress OpenEdge Wire Protocol
- SQL Server Wire Protocol
- Sybase Wire Protocol
- Sybase IQ Wire Protocol

Extended connection failover provides failover protection for the following types of connections:

- New connections, in the same way as described in [Connection Failover](#) on page 78
- Lost connections

When a connection to the database is lost, the driver fails over the connection to an alternate server, restoring the same state of the connection at the time it was lost. For example, when reestablishing a lost connection on the alternate database server, the driver performs the following actions:

- Restores the connection using the same connection options specified by the lost connection
- Reallocates statement handles and attributes
- Logs in the user to the database with the same user credentials
- Restores any prepared statements associated with the connection and repopulates the statement pool
- Restores manual commit mode if the connection was in manual commit mode at the time of the failover

The driver does not preserve work in progress. For example, if the database server experienced a hardware failure while processing a query, partial rows processed by the database and returned to the client would be lost. If the driver was in manual commit mode and one or more Inserts or Updates were performed in the current transaction before the failover occurred, then the transaction on the primary server is rolled back. The Inserts or Updates done before the failover are not committed to the primary server. Your application needs to rerun the transaction after the failover because the Inserts or Updates done before the failover are not repeated by the driver on the failover connection.

When a failover occurs, if a statement is in allocated or prepared state, the next operation on the statement returns a SQL state of 01000 and a vendor code of 0. If a statement is in an executed or prepared state, the next operation returns a SQL state of 40001 and a vendor code of 0. Either condition returns an error message similar to:

```
Your connection has been terminated. However, you have been successfully connected to
the next available AlternateServer: 'HOSTNAME=Server4:PORTNUMBER= 1521:SERVICENAME=test'.
All active transactions have been rolled back.
```

The driver retains all connection settings made through ODBC API calls when a failover connection is made. It does not, however, retain any session settings established through SQL statements. This can be done through the Initialization String connection option, described in the individual driver chapters.

The driver retains the contents of parameter buffers, which can be important when failing over after a fetch. All Select statements are re-prepared at the time the failover connection is made. All other statements are placed in an allocated state.

If an error occurs while the driver is reestablishing a lost connection, the driver can fail the entire failover process or proceed with the process as far as it can. For example, suppose an error occurred while reestablishing the connection because a table for which the driver had a prepared statement did not exist on the alternate connection. In this case, you may want the driver to notify your application of the error and proceed with the failover process. You can choose how you want the driver to behave if errors occur during failover by setting the [Failover Granularity](#) connection option.

During the failover process, your application may experience a short pause while the driver establishes a connection on an alternate server. If your application is time-sensitive (a real-time customer order application, for example) and cannot absorb this wait, you can set the [Failover Preconnect](#) connection option to true. Setting the Failover Preconnect option to true instructs the driver to establish connections to the primary server and an alternate server at the same time. Your application uses the first connection that is successfully established. If this connection to the database is lost at a later time, the driver saves time in reestablishing the connection on the server to which it fails over because it can use the spare connection in its failover process.

This pre-established failover connection is not used by the driver until the driver determines that it needs to fail over. If the server to which the driver is connected or the network equipment through which the connection is routed is configured with a timeout, the pre-configured failover connection could time out. The pre-configured failover connection can also be lost if the failover server is brought down and back up again. The driver tries to establish the connection to the failover server again if the connection is lost.

Select Connection Failover

Select connection failover is available in the following DataDirect Connect Series *for* ODBC drivers:

- DB2 Wire Protocol
- Greenplum Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol
- PostgreSQL Wire Protocol
- Progress OpenEdge Wire Protocol
- SQL Server Wire Protocol
- Sybase Wire Protocol
- Sybase IQ Wire Protocol

Select connection failover provides failover protection for the following types of connections:

- New connections, in the same way as described in [Connection Failover](#) on page 78
- Lost connections, in the same way as described in [Extended Connection Failover](#) on page 80

In addition, the driver can recover work in progress because it keeps track of the last Select statement the application executed on each Statement handle, including how many rows were fetched to the client. For example, if the database had only processed 500 of 1,000 rows requested by a Select statement when the connection was lost, the driver would reestablish the connection to an alternate server, re-execute the Select statement, and position the cursor on the next row so that the driver can continue fetching the balance of rows as if nothing had happened.

Performance, however, is a factor when considering whether to use Select mode. Select mode incurs additional overhead when tracking what rows the application has already processed.

The driver only recovers work requested by Select statements. You must explicitly restart the following types of statements after a failover occurs:

- Insert, Update, or Delete statements
- Statements that modify the connection state, for example, SET or ALTER SESSION statements
- Objects stored in a temporary tablespace or global temporary table
- Partially executed stored procedures and batch statements

When in manual transaction mode, no statements are rerun if any of the operations in the transaction were Insert, Update, or Delete. This is true even if the statement in process at the time of failover was a Select statement.

By default, the driver verifies that the rows that are restored match the rows that were originally fetched and, if they do not match, generates an error warning your application that the Select statement must be reissued. By setting the Failover Granularity connection option, you can customize the driver to ignore this check altogether or fail the entire failover process if the rows do not match.

When the row comparison does not agree, the default behavior of Failover Granularity returns a SQL state of 40003 and an error message similar to:

```
Unable to position to the correct row after a successful failover attempt to
AlternateServer: 'HOSTNAME=Server4:PORTNUMBER= 1521:SERVICENAME=test'. You must reissue
the select statement.
```

If you have configured Failover Granularity to fail the entire failover process, the driver returns a SQL state of 08S01 and an error message similar to:

```
Your connection has been terminated and attempts to complete the failover process to the
following Alternate Servers have failed: AlternateServer: 'HOSTNAME=Server4:PORTNUMBER=
1521:SERVICENAME=test'. All active transactions have been rolled back.
```

Guidelines for Primary and Alternate Servers

Many databases provide advanced database replication technologies such as DB2 High Availability Disaster Recovery (HADR) and Oracle Real Application Clusters (RAC), and Microsoft Cluster Server (MSCS). The failover functionality provided by the drivers does not require any of these technologies, but can work with them to provide comprehensive failover protection. Use the following guidelines for primary and alternate servers to ensure that failover works correctly in your environment:

- Alternate servers should mirror data on the primary server or be part of a configuration where multiple database nodes share the same physical data.

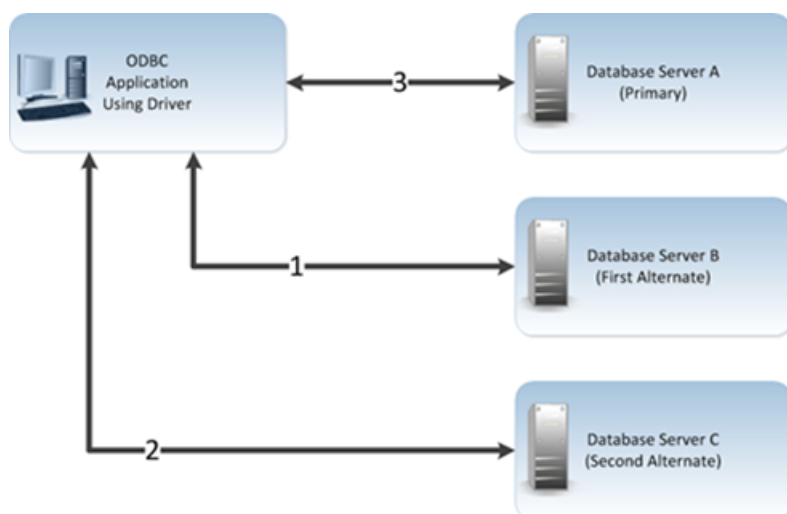
- If using failover with DB2 HADR, the primary server must be the primary server configured in your HADR system and any alternate server must be a standby server configured in your HADR system.

Using Client Load Balancing

Client load balancing is available in the following DataDirect Connect Series *for* ODBC drivers:

- DB2 Wire Protocol
- Greenplum Wire Protocol
- Informix Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol
- Oracle
- PostgreSQL Wire Protocol
- Progress OpenEdge Wire Protocol
- SQL Server Wire Protocol
- SQL Server Legacy Wire Protocol (UNIX only)
- Sybase Wire Protocol
- Sybase IQ Wire Protocol

Client load balancing helps distribute new connections in your environment so that no one server is overwhelmed with connection requests. When client load balancing is enabled, the order in which primary and alternate database servers are tried is random. For example, suppose that client load balancing is enabled as shown in the following illustration:



First, Database Server B is tried (1). Then, Database Server C may be tried (2), followed by a connection attempt to Database Server A (3). In contrast, if client load balancing were not enabled in this scenario, each database server would be tried in sequential order, primary server first, then each alternate server based on its entry order in the alternate servers list.

Client load balancing is controlled by the [Load Balancing](#) connection option. For details on configuring client load balancing, see the appropriate driver chapter in this book.

Using Connection Retry

Connection retry is available in the following DataDirect Connect Series *for* ODBC drivers:

- DB2 Wire Protocol
- Greenplum Wire Protocol
- Informix Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol
- Oracle
- PostgreSQL Wire Protocol
- Progress OpenEdge Wire Protocol
- SQL Server Wire Protocol
- SQL Server Legacy Wire Protocol (UNIX only)
- Sybase Wire Protocol
- Sybase IQ Wire Protocol

Connection retry defines the number of times the driver attempts to connect to the primary server and, if configured, alternate database servers after the initial unsuccessful connection attempt. It can be used with connection failover, extended connection failover, and select failover. Connection retry can be an important strategy for system recovery. For example, suppose you have a power failure in which both the client and the server fails. When the power is restored and all computers are restarted, the client may be ready to attempt a connection before the server has completed its startup routines. If connection retry is enabled, the client application can continue to retry the connection until a connection is successfully accepted by the server.

Connection retry can be used in environments that have only one server or can be used as a complementary feature with connection failover in environments with multiple servers.

Using the connection options [Connection Retry Count](#) and [Connection Retry Delay](#), you can specify the number of times the driver attempts to connect and the time in seconds between connection attempts. For details on configuring connection retry, see the appropriate driver chapter in this book.

Summary of Failover-Related Options

The following table summarizes how failover-related connection options work with the drivers. See "Connection Option Descriptions" in each driver chapter for details about configuring the options. Not all options are available in every failover-enabled driver. The step numbers in the table refer the procedure that follows the table

Table 5: Summary: Failover and Related Connection Options

Option	Characteristic
Alternate Servers (See step 1 on page 85)	One or multiple alternate database servers. An IP address or server name identifying each server is required.
Connection Retry Count (See step 5 on page 86)	Number of times the driver retries the primary database server, and if specified, alternate servers until a successful connection is established.
Connection Retry Delay (See step 6 on page 86)	Wait interval, in seconds, between connection retry attempts when the Connection Retry Count option is set to a positive integer.
Failover Granularity (See step 3 on page 85)	The type of behavior that the driver exhibits when errors are detected during the failover process.
Failover Mode (See step 2 on page 85)	The type of failover that the driver attempts.
Failover Preconnect (See step 4 on page 86)	Determines whether the driver makes a connection attempt to the next server in the Alternate Servers list at the time of the initial connection.
Load Balancing (See step 7 on page 86)	Determines whether the driver uses client load balancing in its attempts to connect to primary and alternate database servers. If enabled, the driver attempts to connect to the database servers in random order.

1. To configure connection failover, you **must** specify one or more alternate database servers that are tried at connection time if the primary server is not accepting connections. To do this, use the Alternate Servers connection option. Connection attempts continue until a connection is successfully established or until all the database servers in the list have been tried once (the default).
2. Choose a failover method by setting the Failover Mode connection option. The default method is Connection (FailoverMode=0).
3. If Failover Mode is Extended Connection (FailoverMode=1) or Select (FailoverMode=2), set the Failover Granularity connection option to specify how you want the driver to behave if errors occur while trying to reestablish a lost connection. The default behavior of the driver is Non-Atomic (FailoverGranularity=0), which continues with the failover process and posts any errors on the statement on which they occur. Other values are:

Atomic (FailoverGranularity=1): the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

Atomic including Repositioning (FailoverGranularity=2): the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

Disable Integrity Check (FailoverGranularity=3: the driver does not verify that the rows restored during the failover process match the original rows. This value applies only when Failover Mode is set to Select (FailoverMode=2).

4. Optionally, enable the Failover Preconnect connection option (`FailoverPreconnect=1`) if you want the driver to establish a connection with the primary and an alternate server at the same time. This value applies only when Failover Mode is set to Extended Connection (`FailoverMode=1`) or Select (`FailoverMode=2`). The default behavior is to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection (`FailoverPreconnect=0`).
5. Optionally, specify the number of times the driver attempts to connect to the primary and alternate database servers after the initial unsuccessful connection attempt. By default, the driver does not retry. To set this feature, use the Connection Retry Count connection option.
6. Optionally, specify the wait interval, in seconds, between attempts to connect to the primary and alternate database servers. The default interval is 3 seconds. To set this feature, use the Connection Retry Delay connection option.
7. Optionally, specify whether the driver will use client load balancing in its attempts to connect to primary and alternate database servers. If load balancing is enabled, the driver uses a random pattern instead of a sequential pattern in its attempts to connect. The default value is not to use load balancing. To set this feature, use the Load Balancing connection option.

A Connection String Example

The following connection string configures the Oracle Wire Protocol driver to use connection failover in conjunction with some of its optional features.

```
DSN=AcctOracleServer;AlternateServers=(HostName=AccountingOracleServer:PortNumber=1521:SID=Accounting, HostName=255.201.11.24:PortNumber=1522:ServiceName=ABackup.NA.MyCompany);ConnectionRetryCount=4;ConnectionRetryDelay=5;LoadBalancing=1;FailoverMode=0
```

Specifically, this connection string configures the driver to use two alternate servers as connection failover servers, to attempt to connect four additional times if the initial attempt fails, to wait five seconds between attempts, to try the primary and alternate servers in a random order, and to attempt reconnecting on new connections only. The additional connection information required for the alternate servers is specified in the data source `AcctOracleServer`.

An odbc.ini File Example

To configure the 32-bit Oracle Wire Protocol driver to use connection failover in conjunction with some of its optional features in your `odbc.ini` file, you could set the following connection string attributes:

```
Driver=ODBCHOME/lib/ivoraxx.so
Description=DataDirect Oracle Wire Protocol driver
...
AlternateServers=(HostName=AccountingOracleServer:PortNumber=1521:SID=Accounting,
HostName=255.201.11.24:PortNumber=1522:ServiceName=ABackup.NA.MyCompany)
...
ConnectionRetryCount=4
ConnectionRetryDelay=5
...
LoadBalancing=0
...
FailoverMode=1
...
FailoverPreconnect=1
...
```

Specifically, this `odbc.ini` configuration tells the driver to use two alternate servers as connection failover servers, to attempt to connect four additional times if the initial attempt fails, to wait five seconds between attempts, to try the primary and alternate servers in sequential order (do not use load balancing), to attempt reconnecting on new and lost connections, and to establish a connection with the primary and alternate servers at the same time.

Using Client Information

Many databases allow applications to store client information associated with a connection. For example, the following types of information can be useful for database administration and monitoring purposes:

- Name of the application currently using the connection.
- User ID for whom the application using the connection is performing work. The user ID may be different than the user ID that was used to establish the connection.
- Host name of the client on which the application using the connection is running.
- Product name and version of the driver on the client.
- Additional information that may be used for accounting or troubleshooting purposes, such as an accounting ID.

Client information is available in the following DataDirect Connect Series *for* ODBC drivers:

- DB2 Wire Protocol
- Oracle Wire Protocol

For DB2 V9.5 and higher for Linux/UNIX/Windows and DB2 for z/OS, this information can feed directly into the Workload Manager (WLM) for workload management and monitoring purposes.

For Oracle 11g R2 and higher, this information is managed through the client information feature.

See [Storing Client Information](#) on page 87 for more information about how DB2 and Oracle store client information.

How Databases Store Client Information

Typically, databases that support storing client information do so by providing a register, a variable, or a column in a system table in which the information is stored. If an application attempts to store information and the database does not provide a mechanism for storing that information, the driver caches the information locally. Similarly, if an application returns client information and the database does not provide a mechanism for storing that information, the driver returns the locally cached value.

Storing Client Information

Your application can store client information associated with a connection. The following table shows the driver connection options that your application can use to store client information and where that client information is stored for each database. See the specific driver chapters for a description of each option.

Table 6: Database Locations for Storing Client Information

Option	Description	Database	Location
Accounting Info	Additional information that may be used for accounting or troubleshooting purposes, such as an accounting ID	DB2	CURRENT CLIENT_ACCTNG register (DB2 for Linux/UNIX/Windows) or CLIENT ACCTNG register (DB2 for z/OS).
		Oracle	CLIENT_INFO value in the V\$SESSION table.
Action	The current action within the current module.	Oracle	ACTION value in the V\$SESSION table.
Application Name	Name of the application currently using the connection	DB2	CURRENT CLIENT_APPLNAME register (DB2 for Linux/UNIX/Windows) or CLIENT APPLNAME register (DB2 for z/OS). For DB2 V9.1 and higher for Linux/UNIX/Windows, this value is also stored in the APPL_NAME value in the SYSIBMADM.APPLICATIONS table.
		Oracle	CLIENT_IDENTIFIER attribute. In addition, this value is also stored in the PROGRAM value in the V\$SESSION table.
Client Host Name	Host name of the client on which the application using the connection is running	DB2	CURRENT CLIENT_WRKSTNNAME register (DB2 for Linux/UNIX/Windows) or CLIENT WRKSTNNAME register (DB2 for z/OS).
		Oracle	MACHINE value in the V\$SESSION table.
Client ID	Additional information about the client	Oracle	CLIENT_IDENTIFIER value in the V\$SESSION table.
Client User	User ID for whom the application using the connection is performing work	DB2	CURRENT CLIENT_USERID register (DB2 for Linux/UNIX/Windows) or CLIENT USERID register (DB2 for z/OS).
		Oracle	OSUSER value in the V\$SESSION table.
Module	The name of a stored procedure or the name of the application	Oracle	MODULE value in the V\$SESSION table.
Program ID	Product name and version of the driver on the client	DB2	CLIENT_PRDID value. For DB2 V9.1 and higher for Linux/UNIX/Windows, the CLIENT_PRDID value is located in the SYSIBMADM.APPLICATIONS table.
		Oracle	PROCESS value in the V\$SESSION table.

Using Security

The drivers support the following security features:

- *Authentication* is the process of identifying a user.
- *Data encryption* is the conversion of data into a form that cannot be easily understood by unauthorized users.

Authentication

On most computer systems, a password is used to prove a user's identity. This password often is transmitted over the network and can possibly be intercepted by malicious hackers. Because this password is the one secret piece of information that identifies a user, anyone knowing a user's password can effectively be that user. Authentication methods protect the identity of the user.

The drivers support the following authentication methods:

- *User ID/password authentication* authenticates the user to the database using a database user name and password.
- *Client authentication* uses the user ID and password of the user logged onto the system on which the driver is running to authenticate the user to the database. The database server relies on the client to authenticate the user and does not provide additional authentication.
- *Kerberos authentication* is a trusted third-party authentication service that verifies user identities. DataDirect Connect Series *for* ODBC supports both Windows Active Directory Kerberos and MIT Kerberos implementations.
- *NTLM authentication* authenticates clients to the database through a challenge-response authentication mechanism that enables clients to prove their identities without sending a database password to the server.

Kerberos Authentication

Kerberos authentication is available in the following DataDirect Connect Series *for* ODBC drivers:

- Driver for Apache Hive
- DB2 Wire Protocol
- Greenplum Wire Protocol
- Impala Wire Protocol
- Oracle Wire Protocol
- PostgreSQL Wire Protocol
- SQL Server Wire Protocol
- Sybase Wire Protocol
- Driver for the Teradata Database

Kerberos authentication can take advantage of the user name and password maintained by the operating system to authenticate users to the database or use another set of user credentials specified by the application.

The Kerberos method requires knowledge of how to configure your Kerberos environment. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

To use Kerberos authentication, the application user first must obtain a Kerberos Ticket Granting Ticket (TGT) from the Kerberos server. The Kerberos server verifies the identity of the user and controls access to services using the credentials contained in the TGT.



If the application uses Kerberos authentication from a Windows client, the application user does not explicitly need to obtain a TGT. Windows Active Directory automatically obtains a TGT for the user.

UNIX® If the application uses Kerberos authentication from a UNIX or Linux client, the user must explicitly obtain a TGT. To obtain a TGT explicitly, the user must log onto the Kerberos server using the `kinit` command. For example, the following command requests a TGT from the server with a lifetime of 10 hours, which is renewable for 5 days:

```
kinit -l 10h -r 5d user
```

where `user` is the application user.

Refer to your Kerberos documentation for more information about using the `kinit` command and obtaining TGTs for users.

NTLM Authentication

NTLM authentication is available in the following the DataDirect Connect Series *for* ODBC drivers:

Support for NTLMv2 and NTLMv1:

- SQL Server Wire Protocol

Support for NTLMv1:

- Oracle Wire Protocol
- Driver for the Teradata database

The following table provides the platform support information for the drivers.

Table 7: Driver Support for NTLM Authentication

Driver	Windows	Linux/UNIX
Oracle Wire Protocol	X	
SQL Server Wire Protocol	X	X ²
Driver for the Teradata database	X	

Summary of Authentication-Related Options

The following table summarizes how authentication-related connection options work with the drivers. Note that some authentication-enabled drivers support only a subset of the listed options, as determined by natively supported features. See "Connection Option Descriptions" in each driver chapter for details about configuring the options.

² NTLM single sign on is not supported. To connect to SQL Server, users must use connection attributes to supply the Windows User Id, Password, and Domain to the driver.

Table 8: Summary: Security Connection Options

Option	Characteristic
Authentication Method	The method the driver uses to authenticate the user to the server when a connection is established.
GSS Client Library	The name of the Generic Security Service (GSS) client library that the driver uses to communicate with the Key Distribution Center (KDC).
Proxy User	Specifies the UserID used for impersonation.
Service Principal Name	The service principal name to be used by driver for Kerberos authentication.
User Name	The default user ID used to connect to your database.

Connection String Examples for Configuring Authentication

The following connection string configures the Oracle Wire Protocol driver to use authentication, specifically Kerberos authentication. The examples contains the connection options necessary to configure Kerberos authentication as well as the minimum options required to establish a connection.

```
DSN=AcctOracleServer;HostName=AccountingOracleServer;AuthenticationMethod=4;
GSSClient=native;PortNumber=1521;SID=Accounting;UID=JohnSmith
```

odbc.ini File Examples for Configuring Authentication

The following example `odbc.ini` file configures the 32-bit Oracle Wire Protocol driver to use authentication, specifically Kerberos authentication. The examples contains the connection options necessary to configure Kerberos authentication as well as the minimum options required to establish a connection.

```
Driver=ODBCHOME/lib/ivoraxx.so
Description=DataDirect Oracle Wire Protocol driver
...
AuthenticationMethod=4
...
GSSClient=native
...
HostName=AccountingOracleServer
...
PortNumber=1521
...
SID=Accounting
...
UID=JohnSmith
...
```

Data Encryption Across the Network

If your database connection is not configured to use data encryption, data is sent across the network in a format that is designed for fast transmission and can be decoded by interceptors, given some time and effort. For example, text data is often sent across the wire as clear text. Because this format does not provide complete protection from interceptors, you may want to use data encryption to provide a more secure transmission of data.

For example, you may want to use data encryption in the following scenarios:

- You have offices that share confidential information over an intranet.

- You send sensitive data, such as credit card numbers, over a database connection.
- You need to comply with government or industry privacy and security requirements.

Certain DataDirect Connect Series *for* ODBC drivers support Secure Sockets Layer (SSL). SSL is an industry-standard protocol for sending encrypted data over database connections. SSL secures the integrity of your data by encrypting information and providing client/server authentication. In addition, the DataDirect Connect Series *for* ODBC DB2 Wire Protocol driver supports DB2 database-specific encryption.

Note: Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) required to encrypt and decrypt data.

SSL Encryption

SSL encryption is available in the following DataDirect Connect Series *for* ODBC drivers:

- Apache Hive Wire Protocol
- DB2 Wire Protocol
- Greenplum Wire Protocol
- Impala Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol
- PostgreSQL Wire Protocol
- Progress OpenEdge Wire Protocol
- Salesforce
- SQL Server Wire Protocol
- Sybase Wire Protocol

Note: Communication between the Salesforce driver and Salesforce.com and Force.com is always SSL encrypted.

SSL works by allowing the client and server to send each other encrypted data that only they can decrypt. SSL negotiates the terms of the encryption in a sequence of events known as the *SSL handshake*. During the handshake, the driver negotiates the highest SSL/TLS protocol available. The result of this negotiation determines the encryption cipher suite to be used for the SSL session. The drivers support the following protocols using OpenSSL cipher suites:

- TLSv v1.0, v1.1, v1.2
- SSL v2, v3

The encryption cipher suite defines the type of encryption that is used for any data exchanged through an SSL connection. Some cipher suites are very secure and, therefore, require more time and resources to encrypt and decrypt data, while others provide less security, but are also less resource intensive.

Refer to "SSL encryption cipher suites" in the *Progress DataDirect for ODBC Drivers Reference* for a list of the encryption cipher suites supported by the drivers.

The handshake involves the following types of authentication:

- *SSL server authentication* requires the server to authenticate itself to the client.
- *SSL client authentication* is optional and requires the client to authenticate itself to the server after the server has authenticated itself to the client. Not all databases support SSL client authentication.

Certificates

SSL requires the use of a digitally-signed document, an x.509 standard certificate, for authentication and the secure exchange of data. The purpose of this certificate is to tie the public key contained in the certificate securely to the person/company that holds the corresponding private key. The DataDirect Connect Series *for* ODBC drivers support many popular formats. Supported formats include:

- DER Encoded Binary X.509
- Base64 Encoded X.509
- PKCS #12 / Personal Information Exchange

SSL Server Authentication

When the client makes a connection request, the server presents its public certificate for the client to accept or deny. The client checks the issuer of the certificate against a list of trusted Certificate Authorities (CAs) that resides in an encrypted file on the client known as a *truststore*. If the certificate matches a trusted CA in the truststore, an encrypted connection is established between the client and server. If the certificate does not match, the connection fails and the driver generates an error.

Most truststores are password-protected. The driver must be able to locate the truststore and unlock the truststore with the appropriate password. Two connection string attributes are available to the driver to provide this information: `TrustStore` and `TrustStorePassword`. The value of `TrustStore` is a pathname that specifies the location of the truststore file. The value of `TrustStorePassword` is the password required to access the contents of the truststore.

Alternatively, you can configure the driver to trust any certificate sent by the server, even if the issuer is not a trusted CA. Allowing a driver to trust any certificate sent from the server is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

`ValidateServerCertificate`, another connection string attribute, allows the driver to accept any certificate returned from the server regardless of whether the issuer of the certificate is a trusted CA.

Finally, the connection string attribute, `HostNameInCertificate`, allows an additional method of server verification. When a value is specified for `HostNameInCertificate`, it must match the host name of the server, which has been established by the SSL administrator. This prevents malicious intervention between the client and the server and ensures that the driver is connecting to the server that was requested.

SSL Client Authentication

If the server is configured for SSL client authentication, the server asks the client to verify its identity after the server identity has been proven. Similar to server authentication, the client sends a public certificate to the server to accept or deny. The client stores its public certificate in an encrypted file known as a *keystore*. Public certificates are paired with a private key in the keystore. To send the public certificate, the driver must access the private key.

Like the truststore, most keystores are password-protected. The driver must be able to locate the keystore and unlock the keystore with the appropriate password. Two connection string attributes are available to the driver to provide this information: `KeyStore` and `KeyStorePassword`. The value of `KeyStore` is a pathname that specifies the location of the keystore file. The value of `KeyStorePassword` is the password required to access the keystore.

The private keys stored in a keystore can be individually password-protected. In many cases, the same password is used for access to both the keystore and to the individual keys in the keystore. It is possible, however, that the individual keys are protected by passwords different from the keystore password. The driver needs to know the password for an individual key to be able to retrieve it from the keystore. An additional connection string attribute, `KeyPassword`, allows you to specify a password for an individual key.

Not all databases support SSL client authentication. The individual driver chapters indicate whether client authentication is supported.

Designating an OpenSSL Library

The driver uses OpenSSL library files (TLS/SSL Support Files) to implement cryptographic functions for data sources or connections when encrypting data. By default, the driver is configured to use the most secure version of the library installed with the product; however, you can designate a different version to address security vulnerabilities or incompatibility issues with your current library. Although the driver is only certified against libraries provided by Progress, you can also designate libraries that you supply. The methods described in this section can be used to designate an OpenSSL library file.

Note: For the default library setting, current information, and a complete list of installed OpenSSL libraries, refer to the readme file installed with your product.

File replacement

In the default configuration, the drivers use the OpenSSL library file located in the `\drivers` subdirectory for Windows installations and the `/lib` subdirectory for UNIX/Linux. You can replace this file with a different library to change the version used by the drivers. When using this method, the replacement file must contain both the cryptographic and SSL libraries and use the same file name as the default library. For example, the latest version of the library files use the following naming conventions:

Windows:

- Latest version: `xxtls27.dll`
- 1.0.2 and earlier versions: `xxssl27.dll`

UNIX/Linux:

- Latest version: `libxxtls27.so [.sl]`
- 1.0.2 and earlier versions: `libxxssl27.so [.sl]`

Designating a library in the default directory

If you are using the default directory structure for the product, you can use the `AllowedOpenSSLVersions` option to designate a library. To use the `AllowedOpenSSLVersions` option, specify the version number of the library you want to load. For example, `AllowedOpenSSLVersions=1.0.2` loads the 1.0.2 version of OpenSSL library using the following naming convention and format:

- Windows: `install_dir\drivers\xxssl27.so [.sl]`
- UNIX/Linux: `install_dir/lib/libxxtls27.so [.sl]`

Note that this method works only with OpenSSL library files that match Progress's naming convention and relative installation location.

If you are using the GUI, this option is not exposed on the setup dialog. Instead, use the `Extended Options` field on the `Advanced` tab to configure this option. For more information, see "AllowedOpenSSLVersions" in the chapter for your driver.

Designating the absolute path to a library

For libraries that do not use the default directory structure or file names, you must specify the absolute path to your cryptographic library for the `CryptoLibName` (`CryptoLibName`) option and the absolute path to your SSL library for the `SSLLibName` (`SSLLibName`) option. If you are using OpenSSL library files provided by Progress, these libraries are combined into a single file; therefore, the value specified for these options should be the same. For non-Progress library files, the libraries may use separate files, which would require specifying the unique paths to the `libeay32.dll` (cryptographic library) and `ssleay32.dll` (SSL library) files.

If you are using a GUI, these options are not exposed on the setup dialog. Instead, use the Extended Options field on the Advanced tab to configure these options. For details, see "CryptoLibName" and "SSLLibName" in the chapter for your driver.

Summary of Data Encryption Related Options

The following table summarizes how security-related connection options work with the drivers. See "Connection Option Descriptions" in each driver chapter for details about configuring the options.

Table 9: Summary: Security Connection Options

Option	Characteristic
AllowedOpenSSLVersions	Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.
Crypto Protocol Version	The cryptographic protocols the driver uses when SSL is enabled.
CryptoLibName	The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection.
Encryption Method	The method the driver uses to encrypt data sent between the driver and the database server.
Host Name In Certificate	The host name established by the SSL administrator for the driver to validate the host name contained in the certificate.
Key Password	The password required to access an individual key in the keystore.
Keystore	The path that specifies the location of the keystore file.
Keystore Password	The password required to access the keystore.
PRNGSeedFile (UNIX/Linux only)	The absolute path for the entropy-source file or device used as a seed for SSL key generation.
PRNGSeedSource (UNIX/Linux only)	The source of the seed the driver uses for SSL key generation.
SSLLibName	The absolute path for the OpenSSL library file containing the SSL library to be used by the data source or connection.

Option	Characteristic
Truststore	The path that specifies the location of the truststore file.
Truststore Password	The password required to access the truststore.
Validate Server Certificate	Validates the security certificate of the server as part of the SSL authentication handshake.

Connection String Examples for Configuring Data Encryption

The following connection strings configure the Oracle Wire Protocol driver to use data encryption via the SSL server authentication and SSL client authentication methods. These examples contain the connection options necessary to configure data encryption as well as the minimum options required to establish a connection.

SSL Server Authentication

This connection string configures the driver to use the SSL Server Authentication method. In this configuration, since `ValidateServerCertificate=1`, the driver validates the certificate sent by the server and the host name specified by the `HostNameInCertificate` option.

```
DSN=AcctOracleServer;EncryptionMethod=1;HostName=AccountingOracleServer;
HostNameInCertificate=MySubjectAltName;PortNumber=1521;
Truststore=TrustStoreName;TruststorePassword=TSXYZZY;SID=Accounting;
ValidateServerCertificate=1
```

SSL Client Authentication

This connection string configures the driver to use the SSL Server Authentication method. In this configuration, since `ValidateServerCertificate=1`, the driver validates the certificate sent by the server and the host name specified by `HostNameInCertificate`.

```
DSN=AcctOracleServer;EncryptionMethod=1;HostName=AccountingOracleServer;
HostNameInCertificate=MySubjectAltName;KeyPassword=YourKeyPassword;
Keystore=KeyStoreName;KeystorePassword=YourKSPassword;PortNumber=1521;
SID=Accounting;Truststore=TrustStoreName;TruststorePassword=YourTSPassword;
ValidateServerCertificate=1
```

odbc.ini File Examples for Configuring Data Encryption

The following example `odbc.ini` files demonstrate how to configure the 32-bit Oracle Wire Protocol driver to use data encryption via the SSL Server Authentication and SSL Client Authentication methods. These examples include the necessary options to configure data encryption as well as the minimum options required to establish a connection.

SSL Server Authentication

This `odbc.ini` file configures the driver to use the SSL Server Authentication method. In this configuration, since `ValidateServerCertificate=1`, the driver validates the certificate sent by the server and the host name specified by the `HostNameInCertificate` option.

```
Driver=ODBCHOME/lib/ivoraxx.so
Description=DataDirect Oracle Wire Protocol driver
...
EncryptionMethod=1
...
HostName=AccountingOracleServer
HostNameInCertificate=MySubjectAltName
...
```



```

PortNumber=1521
...
SID=Accounting
...
Truststore=TrustStoreName
TruststorePassword=TSXYZZY
...
ValidateServerCertificate=1
...

```

SSL Client Authentication

This `odbc.ini` file configures the driver to use the SSL Client Authentication method. In this configuration, since `ValidateServerCertificate=1`, the driver validates the certificate sent by the server and the host name specified by the `HostNameInCertificate` option.

```

Driver=ODBCHOME/lib/ivoraxx.so
Description=DataDirect Oracle Wire Protocol driver
...
EncryptionMethod=1
...
HostName=AccountingOracleServer
HostNameInCertificate=MySubjectAltName
...
KeyPassword=YourKeyPassword
Keystore=KeyStoreName
KeystorePassword=YourKSPassword
...
PortNumber=1521
...
SID=Accounting
...
Truststore=TrustStoreName
TruststorePassword=TSXYZZY
...
ValidateServerCertificate=1
...

```

Using DataDirect Connection Pooling

DataDirect connection pooling is available in the following DataDirect Connect Series *for* ODBC drivers:

- DB2 Wire Protocol
- Greenplum Wire Protocol
- MySQL Wire Protocol
- Oracle Wire Protocol
- Oracle
- PostgreSQL Wire Protocol
- SQL Server Wire Protocol
- Sybase Wire Protocol
- Sybase IQ Wire Protocol

Connection pooling allows you to *reuse* connections rather than creating a new one every time the driver needs to establish a connection to the underlying database. The DataDirect Connect Series *for* ODBC drivers enable connection pooling without requiring changes to your client application.

Note: Connection pooling works only with connections that are established using `SQLConnect` or `SQLDriverConnect` with the `SQL_DRIVER_NO_PROMPT` argument and only with applications that are thread-enabled.

DataDirect connection pooling that is implemented by the DataDirect driver is different than connection pooling implemented by the Windows Driver Manager. The Windows Driver Manager opens connections dynamically, up to the limits of memory and server resources. DataDirect connection pooling, however, allows you to control the number of connections in a pool through the Min Pool Size (minimum number of connections in a pool) and Max Pool Size (maximum number of connections in a pool) connection options. In addition, DataDirect connection pooling is cross-platform, allowing it to operate on UNIX and Linux. See the "Connection Option Descriptions" section in each driver's chapter for details about how the connection options manage DataDirect connection pooling.

Important: On a Windows system, do not use both Windows Driver Manager connection pooling and DataDirect connection pooling at the same time.

Creating a Connection Pool

Each connection pool is associated with a specific connection string. By default, the connection pool is created when the first connection with a unique connection string connects to the data source. The pool is populated with connections up to the minimum pool size before the first connection is returned. Additional connections can be added until the pool reaches the maximum pool size. If the Max Pool Size option is set to 10 and all connections are active, a request for an eleventh connection has to wait in queue for one of the 10 pool connections to become idle. The pool remains active until the process ends or the driver is unloaded.

If a new connection is opened and the connection string does not exactly match an existing pool, a new pool must be created. By using the same connection string, you can enhance the performance and scalability of your application.

Adding Connections to a Pool

A connection pool is created in the process of creating each unique connection string that an application uses. When a pool is created, it is populated with enough connections to satisfy the minimum pool size requirement, set by the Min Pool Size connection option. The maximum pool size is set by the Max Pool Size connection option. If an application needs more connections than the number set by Min Pool Size, The driver allocates additional connections to the pool until the number of connections reaches the value set by Max Pool Size.

Once the maximum pool size has been reached and no usable connection is available to satisfy a connection request, the request is queued in the driver. The driver waits for the length of time specified in the Login Timeout connection option for a usable connection to return to the application. If this time period expires and a connection has not become available, the driver returns an error to the application.

A connection is returned to the pool when the application calls `SQLDisconnect`. Your application is still responsible for freeing the handle, but this does not result in the database session ending.

Removing Connections from a Pool

A connection is removed from a connection pool when it exceeds its lifetime as determined by the Load Balance Timeout connection option. In addition, DataDirect has created connection attributes described in the following table to give your application the ability to reset connection pools. If connections are in use at the time of these calls, they are marked appropriately. When SQLDisconnect is called, the connections are discarded instead of being returned to the pool.

Table 10: Pool Reset Connection Attributes

Connection Attribute	Description
SQL_ATTR_CLEAR_POOLS Value: SQL_CLEAR_ALL_CONN_POOL	Calling SQLSetConnectAttr (SQL_ATTR_CLEAR_POOLS, SQL_CLEAR_ALL_CONN_POOL) clears all the connection pools associated with the driver that created the connection. This is a write-only connection attribute. The driver returns an error if SQLGetConnectAttr (SQL_ATTR_CLEAR_POOLS) is called.
SQL_ATTR_CLEAR_POOLS Value: SQL_CLEAR_CURRENT_CONN_POOL	Calling SQLSetConnectAttr (SQL_ATTR_CLEAR_POOLS, SQL_CLEAR_CURRENT_CONN_POOL) clears the connection pool that is associated with the current connection. This is a write-only connection attribute. The driver returns an error if SQLGetConnectAttr (SQL_ATTR_CLEAR_POOLS) is called.

Note: By default, if removing a connection causes the number of connections to drop below the number specified in the Min Pool Size option, a new connection is not created until an application needs one.

Handling Dead Connections in a Pool

What happens when an idle connection loses its physical connection to the database? For example, suppose the database server is rebooted or the network experiences a temporary interruption. An application that attempts to connect could receive errors because the physical connection to the database has been lost.

DataDirect Connect Series *for* ODBC drivers handle this situation transparently to the user. The application does not receive any errors on the connection attempt because the driver simply returns a connection from a connection pool. The first time the connection handle is used to execute a SQL statement, the driver detects that the physical connection to the server has been lost and attempts to reconnect to the server *before* executing the SQL statement. If the driver can reconnect to the server, the result of the SQL execution is returned to the application; no errors are returned to the application.

The driver uses connection failover option values, if they are enabled, when attempting this seamless reconnection; however, it attempts to reconnect even if these options are not enabled. See [Connection Failover](#) on page 78 for information about configuring the driver to connect to a backup server when the primary server is not available.

Note: If the driver cannot reconnect to the server (for example, because the server is still down), an error is returned indicating that the reconnect attempt failed, along with specifics about the reason the connection failed.

The technique that Progress DataDirect uses for handling dead connections in connection pools allows for maximum performance of the connection pooling mechanism. Some drivers periodically test the server with a dummy SQL statement while the connections sit idle. Other drivers test the server when the application requests the use of the connection from the connection pool. Both of these approaches add round trips to the database server and ultimately slow down the application during normal operation.

Connection Pool Statistics

Progress DataDirect has created a connection attribute to monitor the status of the DataDirect Connect Series for ODBC connection pools. This attribute, which is described in the following table, allows your application to fetch statistics for the pool to which a connection belongs.

Table 11: Pool Statistics Connection Attribute

Connection Attribute	Description
SQL_ATTR_POOL_INFO Value: SQL_GET_POOL_INFO	Calling SQLGetConnectAttr (SQL_ATTR_POOL_INFO, SQL_GET_POOL_INFO) returns a PoolInfoStruct that contains the statistics for the connection pool to which this connection belongs. This PoolInfoStruct is defined in qesqlx.h. For example: <code>SQLGetConnectAttr(hdbc, SQL_ATTR_POOL_INFO, PoolInfoStruct *, SQL_LEN_BINARY_ATTR(PoolInfoStruct), &len);</code> This is a read-only connection attribute. The driver returns an error if SQLSetConnectAttr (SQL_ATTR_POOL_INFO) is called.

Summary of Pooling-Related Options

The following table summarizes how connection pooling-related connection options work with the drivers. See "Connection Option Descriptions" in each driver chapter for details about configuring the options.

Table 12: Summary: Connection Pooling Connection Options

Option	Characteristic
Connection Pooling	Enables connection pooling.
Connection Reset	Resets a connection that is removed from the connection pool to the initial configuration settings of the connection.
Load Balance Timeout	An integer value to specify the amount of time, in seconds, to keep connections open in a connection pool.
Max Pool Size	An integer value to specify the maximum number of connections within a single pool.
Min Pool Size	An integer value to specify the minimum number of connections that are opened and placed in a connection pool when it is created.

Using DataDirect Bulk Load

The drivers support DataDirect Bulk Load, a feature that allows your application to send large numbers of rows of data to a database or Salesforce instance.

The following tables describe the bulk load behavior for the drivers.

Table 13: Bulk Load Behavior for DataDirect Connect for ODBC

Driver	Bulk Load Behavior
Oracle ³ Microsoft SQL Server ⁴ Sybase	The driver sends the data to the database in a continuous stream instead of numerous smaller database packets. Similar to batch operations, using bulk load improves performance because far fewer network round trips are required. Bulk load bypasses the data parsing usually done by the database, providing an additional performance gain over batch operations.
DB2	Because DB2 does not have native bulk load support, the driver supports bulk through the native parameter array mechanism.

Table 14: Bulk Load Behavior for DataDirect Connect XE for ODBC

Driver	Bulk Load Behavior
Salesforce	The driver sends data to a Salesforce instance using the Salesforce Bulk API instead of the Web Service API. Using the Bulk API significantly reduces the number of Web service calls the driver uses to transfer data and may improve performance.

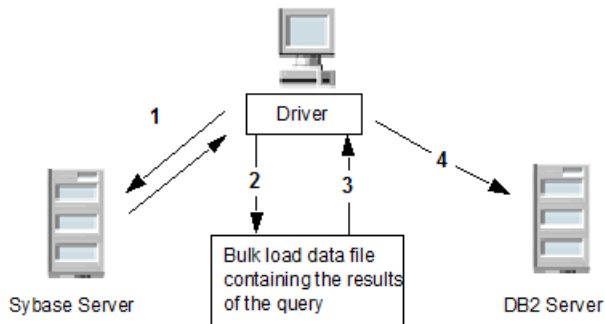
- DataDirect Bulk Load requires a licensed installation of the drivers. If the drivers are installed with an evaluation license, the bulk load feature is available for prototyping with your applications, but with limited scope. Contact your sales representative or Progress DataDirect SupportLink for further information.
- Because a bulk load operation may bypass data integrity checks, your application must ensure that the data it is transferring does not violate integrity constraints in the database. For example, suppose you are bulk loading data into a database table and some of that data duplicates data stored as a primary key, which must be unique. The driver will not throw an exception to alert you to the error; your application must provide its own data integrity checks.

Bulk load operations are accomplished by exporting the results of a query from a database into a comma-separated value (CSV) file, a bulk load data file. The driver then loads the data from bulk load data file into a different database. The file can be used by any DataDirect Connect Series *for* ODBC drivers. In addition, the bulk load data file is supported by other DataDirect Connect product lines that feature bulk loading, for example, a DataDirect Connect for ADO.NET data provider that supports bulk load.

Suppose that you had customer data on a Sybase server and need to export it to a DB2 server. The driver would perform the following steps:

³ Supports bulk load for Oracle9i R2 and higher.

⁴ Supports bulk load for Microsoft SQL Server 2000 and higher.



1. Application using Sybase Wire Protocol driver sends query to and receives results from Sybase server.
2. Driver exports results to bulk load data file.
3. Driver retrieves results from bulk load data file.
4. Driver bulk loads results on DB2 server.

Refer to "DataDirect Bulk Load" in the *Progress DataDirect for ODBC Drivers Reference* for supported functions and statement attributes.

Bulk Export and Load Methods

You can take advantage of DataDirect Bulk Load either through the Driver setup dialog or programmatically.

Applications that are already coded to use parameter array batch functionality can leverage DataDirect Bulk Load features through the Enable Bulk Load connection option on the Bulk tab of the Driver setup dialog. Enabling this option automatically converts the parameter array batch operation to use the database bulk load protocol without any code changes to your application.

If you are not using parameter array batch functionality, the bulk operation buttons **Export Table** and **Load Table** on the Bulk tab of the driver Setup dialog also allow you to use bulk load functionality without any code changes. See the individual driver chapters for a description of the Bulk tab.

If you want to integrate bulk load functionality seamlessly into your application, you can include code to use the bulk load functions exposed by the driver.

For your applications to use DataDirect Bulk Load functionality, they must obtain driver connection handles and function pointers, as follows:

1. Use `SQLGetInfo` with the parameter `SQL_DRIVER_HDBC` to obtain the driver's connection handle from the Driver Manager.
2. Use `SQLGetInfo` with the parameter `SQL_DRIVER_HLIB` to obtain the driver's shared library or DLL handle from the Driver Manager.
3. Obtain function pointers to the bulk load functions using the function name resolution method specific to your operating system. The `bulk.c` example program shipped with the drivers contains the function `resolveName` that illustrates how to obtain function pointers to the bulk load functions.

Exporting Data from a Database

You can export data from a database in one of three ways:

- From a table by using the driver Setup dialog
- From a table by using DataDirect functions

- From a result set by using DataDirect statement attributes

From the DataDirect driver Setup dialog, select the **Bulk** tab and click **Export Table**. See the individual driver chapters for a description of this procedure.

Your application can export a table using the DataDirect functions `ExportTableToFile` (ANSI application) or `ExportTableToFileW` (Unicode application). The application must first obtain driver connection handles and function pointers, as shown in the following example:

```
HDBC      hdbc;
HENV      henv;
void      *driverHandle;
HMODULE   hmod;
PEXPORTTABLETOFILE exportTableToFile;

char      tableName[128];
char      fileName[512];
char      logFile[512];
int       errorTolerance;
int       warningTolerance;
int       codePage;

/* Get the driver's connection handle from the DM.
   This handle must be used when calling directly into the driver. */
rc = SQLGetInfo (hdbc, SQL_DRIVER_HDBC, &driverHandle, 0, NULL);
if (rc != SQL_SUCCESS) {
    ODBC_error (henv, hdbc, SQL_NULL_HSTMT);
    EnvClose (henv, hdbc);
    exit (255);
}

/* Get the DM's shared library or DLL handle to the driver. */
rc = SQLGetInfo (hdbc, SQL_DRIVER_HLIB, &hmod, 0, NULL);
if (rc != SQL_SUCCESS) {
    ODBC_error (henv, hdbc, SQL_NULL_HSTMT);
    EnvClose (henv, hdbc);
    exit (255);
}
exportTableToFile = (PEXPORTTABLETOFILE)
    resolveName (hmod, "ExportTableToFile");
if (! exportTableToFile) {
    printf ("Cannot find ExportTableToFile!\n");
    exit (255);
}

rc = (*exportTableToFile) (
    driverHandle,
    (const SQLCHAR *) tableName,
    (const SQLCHAR *) fileName,
    codePage,
    errorTolerance, warningTolerance,
    (const SQLCHAR *) logFile);
if (rc == SQL_SUCCESS) {
    printf ("Export succeeded.\n");
}
else {
    driverError (driverHandle, hmod);
}
}
```

Your application can export a result set using the DataDirect statement attributes `SQL_BULK_EXPORT` and `SQL_BULK_EXPORT_PARAMS`.

The export operation creates a bulk load data file with a .csv extension in which the exported data is stored. For example, assume that an Oracle source table named GBMAXTABLE contains four columns. The resulting bulk load data file GBMAXTABLE.csv containing the results of a query would be similar to the following:

```
1,0x6263,"bc","bc"
2,0x636465,"cde","cde"
3,0x64656667,"defg","defg"
4,0x6566676869,"efghi","efghi"
5,0x666768696a6b,"fghijk","fghijk"
6,0x6768696a6b6c6d,"ghijklm","ghijklm"
7,0x68696a6b6c6d6e6f,"hijklmno","hijklmno"
8,0x696a6b6c6d6e6f7071,"ijklmnopq","ijklmnopq"
9,0x6a6b6c6d6e6f70717273,"jklmnopqrs","jklmnopqrs"
10,0x6b,"k","k"
```

A bulk load configuration file with an .xml extension is also created when either a table or a result set is exported to a bulk load data file. See [The Bulk Load Configuration File](#) on page 105 for an example of a bulk load configuration file.

In addition, a log file of events as well as external overflow files can be created during a bulk export operation. The log file is configured through either the driver Setup dialog Bulk tab, the ExportTableToFile function, or the SQL_BULK_EXPORT statement attribute. The external overflow files are configured through connection options; see [External Overflow Files](#) on page 108 for details.

Bulk Loading to a Database

The Enable Bulk Load connection option specifies the method by which bulk data is loaded to a database. When the option is enabled, the driver uses database bulk load protocols. When not enabled, the driver uses standard parameter arrays.

You can load data from the bulk load data file into the target database through the DataDirect driver Setup dialog by selecting the Bulk tab and clicking **Load Table**. See the individual driver chapters of the drivers that support bulk load for a description of this procedure.

Your application can also load data from the bulk load data file into the target database using the using the DataDirect functions LoadTableFromFile (ANSI application) or LoadTableFromFileW (Unicode application). The application must first obtain driver connection handles and function pointers, as shown in the following example:

```
HDBC      hdbc;
HENV      henv;
void      *driverHandle;
HMODULE   hmod;
PLoadTableFromFile loadTableFromFile;
char      tableName[128];
char      fileName[512];
char      configFile[512];
char      logFile[512];
char      discardFile[512];
int       errorTolerance;
int       warningTolerance;
int       loadStart;
int       loadCount;
int       readBufferSize;

/* Get the driver's connection handle from the DM.
   This handle must be used when calling directly into the driver.*/

rc = SQLGetInfo (hdbc, SQL_DRIVER_HDBC, &driverHandle, 0, NULL);
if (rc != SQL_SUCCESS) {
    ODBC_error (henv, hdbc, SQL_NULL_HSTMT);
    EnvClose (henv, hdbc);
    exit (255);
}
```



```

}
/* Get the DM's shared library or DLL handle to the driver. */

rc = SQLGetInfo (hdbc, SQL_DRIVER_HLIB, &hmod, 0, NULL);
if (rc != SQL_SUCCESS) {
    ODBC_error (henv, hdbc, SQL_NULL_HSTMT);
    EnvClose (henv, hdbc);
    exit (255);
}

loadTableFromFile = (PLoadTableFromFile)
    resolveName (hmod, "LoadTableFromFile");
if (! loadTableFromFile) {
    printf ("Cannot find LoadTableFromFile!\n");
    exit (255);
}

rc = (*loadTableFromFile) (
    driverHandle,
    (const SQLCHAR *) tableName,
    (const SQLCHAR *) fileName,
    errorTolerance, warningTolerance,
    (const SQLCHAR *) configFile,
    (const SQLCHAR *) logFile,
    (const SQLCHAR *) discardFile,
    loadStart, loadCount,
    readBufferSize);
if (rc == SQL_SUCCESS) {
    printf ("Load succeeded.\n");
}
else {
    driverError (driverHandle, hmod);
}

```

Use the `BulkLoadBatchSize` connection attribute to specify the number of rows the driver loads to the data source at a time when bulk loading data. Performance can be improved by increasing the number of rows the driver loads at a time because fewer network round trips are required. Be aware that increasing the number of rows that are loaded also causes the driver to consume more memory on the client.

A log file of events as well as a discard file that contains rows rejected during the load can be created during a bulk load operation. These files are configured through either the driver Setup dialog Bulk tab or the `LoadTableFromFile` function.

The discard file is in the same format as the bulk load data file. After fixing reported issues in the discard file, the bulk load can be reissued using the discard file as the bulk load data file.

Note: FOR SYBASE USERS: Additional database configuration is required for destination tables that do not have an index. See the "Persisting a Result Set as an XML Data File" section in your driver chapter for more information.

Note: FOR SALESFORCE USERS: In addition to bulk Insert, the Salesforce driver also supports bulk Delete, Update, and Upsert. This functionality is enabled with the `SetBulkOperation` function which is implemented in the driver.

The Bulk Load Configuration File

A bulk load configuration file is created when either a table or a result set is exported to a bulk load data file. This file has the same name as the bulk load data file, but with an `.xml` extension.

The bulk load configuration file defines in its metadata the names and data types of the columns in the bulk load data file. The file defines these names and data types based on the table or result set created by the query that exported the data.

It also defines other data properties, such as length for character and binary data types, the character encoding code page for character types, precision and scale for numeric types, and nullability for all types.

When a bulk load data file cannot read its configuration file, the following defaults are assumed:

- All data is read in as character data. Each value between commas is read as character data.
- The default character set is defined, on Windows, by the current Windows code page. On UNIX/Linux, it is the IANAAppCodePage value, which defaults to 4.

For example, the format of the bulk load data file GBMAXTABLE.csv (discussed in [Exporting Data from a Database](#) on page 102) is defined by the bulk load configuration file, GBMAXTABLE.xml, as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<table codepage="UTF-16LE" xsi:noNamespaceSchemaLocation=
"http://media.datadirect.com/download/docs/ns/bulk/BulkData.xsd" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance">
  <row>
    <column datatype="DECIMAL" precision="38" scale="0" nullable=
      "false">INTEGERCOL</column>
    <column datatype="VARBINARY" length="10" nullable=
      "true">VARBINCOL</column>
    <column datatype="VARCHAR" length="10" sourcecodepage="Windows-1252"
      externalfilecodepage="Windows-1252" nullable="true">VCHARCOL</column>
    <column datatype="VARCHAR" length="10" sourcecodepage="Windows-1252"
      externalfilecodepage="Windows-1252" nullable="true">UNIVCHARCOL</column>
  </row>
</table>
```

Bulk Load Configuration File Schema

The bulk load configuration file is supported by an underlying XML Schema defined at:

<http://media.datadirect.com/download/docs/ns/bulk/BulkData.xsd>

The bulk load configuration file must conform to the bulk load configuration XML schema. Each bulk export operation generates a bulk load configuration file in UTF-8 format. If the bulk load data file cannot be created or does not comply with the XML Schema described in the bulk load configuration file, an error is generated.

Verification of the Bulk Load Configuration File

You can verify the metadata in the configuration file against the data structure of the target database table. This insures that the data in the bulk load data file is compatible with the target database table structure.

The verification does not check the actual data in the bulk load data file, so it is possible that the load can fail even though the verification succeeds. For example, if you were to update the bulk load data file manually such that it has values that exceed the maximum column length of a character column in the target table, the load would fail.

Not all of the error messages or warnings that are generated by verification necessarily mean that the load will fail. Many of the messages simply notify you about possible incompatibilities between the source and target tables. For example, if the bulk load data file has a column that is defined as an integer and the column in the target table is defined as smallint, the load may still succeed if the values in the source column are small enough that they fit in a smallint column.

To verify the metadata in the bulk load configuration file through the DataDirect driver Setup dialog, select the Bulk tab and click **Verify**. See the individual driver chapters of the drivers that support bulk load for a description of this procedure.

Your application can also verify the metadata of the bulk load configuration file using the DataDirect functions `ValidateTableFromFile` (ANSI application) or `ValidateTableFromFileW` (Unicode application). The application must first obtain driver connection handles and function pointers, as shown in the following example:

```

HDBC      hdbc;
HENV      henv;
void      *driverHandle;
HMODULE   hmod;
PValidateTableFromFile validateTableFromFile;
char      tableName[128];
char      configFile[512];
char      messageList[10240];
SQLLEN    numMessages;
/* Get the driver's connection handle from the DM.
   This handle must be used when calling directly into the driver. */
rc = SQLGetInfo (hdbc, SQL_DRIVER_HDBC, &driverHandle, 0, NULL);
if (rc != SQL_SUCCESS) {
    ODBC_error (henv, hdbc, SQL_NULL_HSTMT);
    EnvClose (henv, hdbc);
    exit (255);
}/* Get the DM's shared library or DLL handle to the driver. */
rc = SQLGetInfo (hdbc, SQL_DRIVER_HLIB, &hmod, 0, NULL);
if (rc != SQL_SUCCESS) {
    ODBC_error (henv, hdbc, SQL_NULL_HSTMT);
    EnvClose (henv, hdbc);
    exit (255);
}validateTableFromFile = (PValidateTableFromFile)
    resolveName (hmod, "ValidateTableFromFile");
if (!validateTableFromFile) {
    printf ("Cannot find ValidateTableFromFile!\n");
    exit (255);
}messageList[0] = 0;
numMessages = 0;
rc = (*validateTableFromFile) (
    driverHandle,
    (const SQLCHAR *) tableName,
    (const SQLCHAR *) configFile,
    (SQLCHAR *) messageList,
    sizeof (messageList),
    &numMessages);
printf ("%d message%s%s\n", numMessages,
    (numMessages == 0) ? "s" :
    ((numMessages == 1) ? " : " : "s : "),
    (numMessages > 0) ? messageList : "");
if (rc == SQL_SUCCESS) {
    printf ("Validate succeeded.\n");
}else {
    driverError (driverHandle, hmod);
}

```

Sample Applications

Progress DataDirect provides a sample application that demonstrates the bulk export, verification, and bulk load operations. This application is located in the `\samples\bulk` subdirectory of the product installation directory along with a text file named `bulk.txt`. Please consult `bulk.txt` for instructions on using the sample bulk load application.

A bulk streaming application is also provided in the `\samples\bulkstrm` subdirectory along with a text file named `bulkstrm.txt`. Please consult `bulkstrm.txt` for instructions on using the bulk streaming application.

Character Set Conversions

It is most performance-efficient to transfer data between databases that use the same character sets. At times, however, you might need to bulk load data between databases that use different character sets. You can do this by choosing a character set for the bulk load data file that will accommodate all data. If the source table contains character data that uses different character sets, then one of the Unicode character sets, UTF-8, UTF-16BE, or UTF-16LE must be specified for the bulk load data file. A Unicode character set should also be specified in the case of a target table uses a different character set than the source table to minimize conversion errors. If the source and target tables use the same character set, that set should be specified for the bulk load data file.

A character set is defined by a code page. The code page for the bulk load data file is defined in the configuration file and is specified through either the Code Page option of the Export Table driver Setup dialog or through the IANAAppCodePage parameter of the ExportTableToFile function.

Any code page listed in "Code page values" in the *Progress DataDirect for ODBC Drivers Reference* is supported for the bulk load data file.

Any character conversion errors are handled based on the value of the Report CodePage ConversionErrors connection option. See the individual driver chapters for a description of this option.

The configuration file may optionally define a second code page value for each character column (`externalfilecodepage`). If character data is stored in an external overflow file (see [External Overflow Files](#) on page 108), this second code page value is used for the external file.

External Overflow Files

In addition to the bulk load data file, DataDirect Bulk Load can store bulk data in external overflow files. These overflow files are located in the same directory as the bulk load data file. Different files are used for binary data and character data. Whether or not to use external overflow files is a performance consideration. For example, binary data is stored as hexadecimal-encoded character strings in the main bulk load data file, which increases the size of the file per unit of data stored. External files do not store binary data as hex character strings, and, therefore, require less space. On the other hand, more overhead is required to access external files than to access a single bulk load data file, so each bulk load situation must be considered individually.

The value of the Bulk Binary Threshold connection option determines the threshold, in KB, over which binary data is stored in external files instead of in the bulk load data file. Likewise, the Bulk Character Threshold connection option determines the threshold for character data.

In the case of an external character data file, the character set of the file is governed by the bulk load configuration file. If the bulk load data file is Unicode and the maximum character size of the source data is 1, then the data is stored in its source code page. See [Character Set Conversions](#) on page 108.

The file name of the external file contains the bulk load data file name, a six-digit number, and a ".lob" extension in the following format: `CSVfilename_nnnnnn.lob`. Increments start at 000001.lob.

Using Bulk Load for Single Inserts/Updates/Deletes (Salesforce Driver)

When the Enable Bulk Load connection option is set to 1, the driver uses the Salesforce Bulk API for single Insert, Update, and Delete statements if the number of rows affected by the operation exceeds the threshold set by the Bulk Load Threshold connection property.

For example, if you set the Enable Bulk Load connection option to 1 and the Bulk Load Threshold connection option to 2000, executing the following statement would use the Bulk API if the number of rows returned by `SELECT rowid, sys_name FROM account` is more than 2000 rows.

```
INSERT INTO tmpAccounts(accountId, accountName)
SELECT rowid, sys_name FROM account
```

Summary of Related options for DataDirect Bulk Load

The following table summarizes how DataDirect Bulk Load related connection options work with the drivers. See "Connection Option Descriptions" in each driver chapter for details about configuring the options.

Table 15: Summary: Bulk Load Connection Options

Option	Characteristic
Batch Size	An integer value that specifies the number of rows at a time that the driver sends to the database during bulk operations.
Bulk Binary Threshold	An integer value that specifies the maximum size, in KB, of binary data exported to the bulk data file. Any data exceeding this size is exported to an external file.
Bulk Character Threshold	An integer value that specifies the maximum size, in KB, of character data exported to the bulk data file. Any data exceeding this size is exported to an external file.
Bulk Options	Toggles options for the bulk load process.
Field Delimiter	Specifies the character that the driver will use to delimit the field entries in a bulk load data file.
Record Delimiter	Specifies the character that the driver will use to delimit the record entries in a bulk load data file.

Using Bulk Load for Batch Inserts

For all drivers that support bulk operations, except the Salesforce driver, the driver uses the native bulk load protocol for database connections when the Enable Bulk Load connection option is set to `true`. For example, if you set the Enable Bulk Load connection option to `true`, the driver would use bulk load for the native parameter array insert request.

In some cases, the driver may not be able to use bulk load because of restrictions enforced by the bulk load protocol and will downgrade to a batch mechanism. For example, if the data being loaded has a data type that is not supported by the bulk load protocol, the driver cannot use bulk load, but will use the native parameter array insert mechanism instead.

For the Salesforce driver, when the Enable Bulk Load connection option is set to `true` and the number of rows to be inserted in the batch is larger than Bulk Load Threshold, the driver uses the Salesforce Bulk API instead of the Web service API.

For all drivers that support bulk operations, use the Bulk Load Batch Size connection option to specify the number of rows the driver loads at a time when bulk loading data. Performance can be improved by increasing the number of rows the driver loads at a time because fewer network round trips are required. Be aware that increasing the number of rows that are loaded also causes the driver to consume more memory on the client.

Determining the Bulk Load Protocol

Bulk operations can be performed using a dedicated bulk load protocol, that is, the protocol of the underlying database system, or by using parameter array batch operations. Dedicated protocols are generally more performance-efficient than parameter arrays. In some cases, however, you must use parameter arrays, for example, when the data to be loaded is in a data type not supported by the dedicated bulk protocol.

The Enable Bulk Load connection option determines bulk load behavior. When the option is enabled, the driver uses database bulk load protocols unless it encounters a problem, in which case it returns an error. In this situation, you must disable Enable Bulk Load so that the driver uses standard parameter arrays.

Summary of Related Options for Bulk Load or Batch Inserts

The following table summarizes how connection options related to bulk load for batch inserts work with the drivers. See "Connection Option Descriptions" in each driver chapter for details about configuring the options.

Table 16: Summary: Bulk Load Connection Options

Option	Characteristic
Batch Size	An integer value that specifies the number of rows at a time that the driver sends to the database during bulk operations.
Enable Bulk Load	When enabled, the driver uses database bulk load protocols. When not enabled, the driver uses standard parameter arrays.

Configuring the Product on UNIX/Linux

UNIX[®]

This chapter contains specific information about using the DataDirect Connect Series *for* ODBC drivers in the UNIX and Linux environments.

See [Environment-Specific Information](#) on page 58 for additional platform information.

For details, see the following topics:

- [Environment Variables](#)
- [The Test Loading Tool](#)
- [Data Source Configuration](#)
- [The demoodbc Application](#)
- [The example Application](#)
- [DSN-less Connections](#)
- [File Data Sources](#)
- [UTF-16 Applications on UNIX and Linux](#)

Environment Variables

The first step in setting up and configuring the drivers for use is to set several environment variables. The following procedures require that you have the appropriate permissions to modify your environment and to read, write, and execute various files. You must log in as a user with full r/w/x permissions recursively on the entire DataDirect Connect Series *for* ODBC installation directory.

Library Search Path

The library search path variable can be set by executing the appropriate shell script located in the ODBC home directory. From your login shell, determine which shell you are running by executing:

```
echo $SHELL
```

C shell login (and related shell) users must execute the following command before attempting to use ODBC-enabled applications:

```
source ./odbc.csh
```

Bourne shell login (and related shell) users must initialize their environment as follows:

```
. ./odbc.sh
```

Executing these scripts sets the appropriate library search path environment variable:

- LD_LIBRARY_PATH on HP-UX IPF, Linux, and Oracle Solaris
- LIBPATH on AIX
- SHLIB_PATH on HP-UX PA-RISC

The library search path environment variable must be set so that the ODBC core components and drivers can be located at the time of execution. After running the setup script, execute:

```
env
```

to verify that the *installation_directory/lib* directory has been added to your shared library path.

Some of the client-based drivers must have additional environment variables set. Consult the driver requirements in each of the individual driver chapters for additional environment variable information.

ODBCINI

Setup installs in the product installation directory a default system information file, named `odbc.ini`, that contains data sources. See [Data Source Configuration](#) on page 114 for an explanation of the `odbc.ini` file. The system administrator can choose to rename the file and/or move it to another location. In either case, the environment variable `ODBCINI` must be set to point to the fully qualified path name of the `odbc.ini` file.

For example, to point to the location of the file for an installation on `/opt/odbc` in the C shell, you would set this variable as follows:

```
setenv ODBCINI /opt/odbc/odbc.ini
```


In the Bourne or Korn shell, you would set it as:

```
ODBCINI=/opt/odbc/odbc.ini;export ODBCINI
```

As an alternative, you can choose to make the `odbc.ini` file a hidden file and not set the `ODBCINI` variable. In this case, you would need to rename the file to `.odbc.ini` (to make it a hidden file) and move it to the user's `$HOME` directory.

The driver searches for the location of the `odbc.ini` file as follows:

1. The driver checks the `ODBCINI` variable
2. The driver checks `$HOME` for `.odbc.ini`

If the driver does not locate the system information file, it returns an error.

ODBCINST

Setup installs in the product installation directory a default file, named `odbcinst.ini`, for use with DSN-less connections. See [DSN-less Connections](#) on page 132 for an explanation of the `odbcinst.ini` file. The system administrator can choose to rename the file or move it to another location. In either case, the environment variable `ODBCINST` must be set to point to the fully qualified path name of the `odbcinst.ini` file.

For example, to point to the location of the file for an installation on `/opt/odbc` in the C shell, you would set this variable as follows:

```
setenv ODBCINST /opt/odbc/odbcinst.ini
```

In the Bourne or Korn shell, you would set it as:

```
ODBCINST=/opt/odbc/odbcinst.ini;export ODBCINST
```

As an alternative, you can choose to make the `odbcinst.ini` file a hidden file and not set the `ODBCINST` variable. In this case, you would need to rename the file to `.odbcinst.ini` (to make it a hidden file) and move it to the user's `$HOME` directory.

The driver searches for the location of the `odbcinst.ini` file as follows:

1. The driver checks the `ODBCINST` variable
2. The driver checks `$HOME` for `.odbcinst.ini`

If the driver does not locate the `odbcinst.ini` file, it returns an error.

DD_INSTALLDIR

This variable provides the driver with the location of the product installation directory so that it can access support files. `DD_INSTALLDIR` must be set to point to the fully qualified path name of the installation directory.

For example, to point to the location of the directory for an installation on `/opt/odbc` in the C shell, you would set this variable as follows:

```
setenv DD_INSTALLDIR /opt/odbc
```

In the Bourne or Korn shell, you would set it as:

```
DD_INSTALLDIR=/opt/odbc;export DD_INSTALLDIR
```

The driver searches for the location of the installation directory as follows:

1. The driver checks the DD_INSTALLDIR variable
2. The driver checks the odbc.ini or the odbcinst.ini files for the InstallDir keyword (see [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for a description of the InstallDir keyword)

If the driver does not locate the installation directory, it returns an error.

The next step is to test load the driver.

The Test Loading Tool

The second step in preparing to use a driver is to test load it.

The ivtestlib (32-bit drivers) and dctestlib (64-bit drivers) test loading tools are provided to test load drivers and help diagnose configuration problems in the UNIX and Linux environments, such as environment variables not correctly set or missing database client components. This tool is installed in the /bin subdirectory in the product installation directory. It attempts to load a specified ODBC driver and prints out all available error information if the load fails.

The ivtestlib test loading tool is provided to test load drivers and help diagnose configuration problems in the UNIX and Linux environments, such as environment variables not correctly set or missing database client components. This tool is installed in the bin subdirectory in the product installation directory. It attempts to load a specified ODBC driver and prints out all available error information if the load fails.

For example, if the drivers are installed in /opt/odbc/lib, the following command attempts to load the 32-bit Oracle Wire Protocol driver on Solaris, where *xx* represents the version number of the driver:

```
ivtestlib /opt/odbc/lib/ivoraxx.so
```

Note: On Solaris, AIX, and Linux, the full path to the driver does not have to be specified for the tool. The HP-UX version, however, requires the full path.

If the load is successful, the tool returns a success message along with the version string of the driver. If the driver cannot be loaded, the tool returns an error message explaining why.

See [Version String Information](#) on page 70 for details about version strings.

The next step is to configure a data source through the system information file.

Data Source Configuration

In the UNIX and Linux environments, a system information file is used to store data source information. Setup installs a default version of this file, called odbc.ini, in the product installation directory (see [ODBCINI](#) on page 112 for details about relocating and renaming this file). This is a plain text file that contains data source definitions. If you have a Motif graphical user interface (GUI) in your Linux environment, you can use the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) to create or modify data source definitions in this file (see [Configuration Through the Administrator](#) on page 115 for details). If you do not, see [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 and follow the instructions for configuring the file.

Note: The Linux ODBC Administrator is currently supported only on Linux for x86 and x64 processors with Motif 2.0.3 and higher. It is not supported on Linux for Itanium II or other UNIX platforms.

On Linux, you can determine if you are using Motif through the following command:

```
rpm -qa |grep motif
```

The rpm command returns output similar to:

```
nc-linuxqa3[/home2/users/mike] rpm -qa |grep motif
openmotif-2.2.2-124
openmotif-devel-2.2.2-124
```

If you are not using a GUI, you can use any text editor to create or modify data source definitions directly in the odbc.ini file. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for details.

Configuration Through the Administrator

The Linux ODBC Administrator is located in the /tools directory of the product installation directory. For example,

```
/opt/odbc/tools/odbcadmin
```

The following drivers can be configured with the Linux ODBC Administrator:

- DB2 Wire Protocol
- dBASE
- Greenplum Wire Protocol
- Informix Wire Protocol
- Oracle
- Oracle Wire Protocol
- PostgreSQL Wire Protocol
- SQL Server Legacy Wire Protocol
- Sybase Wire Protocol
- The Driver for Teradata
- Text

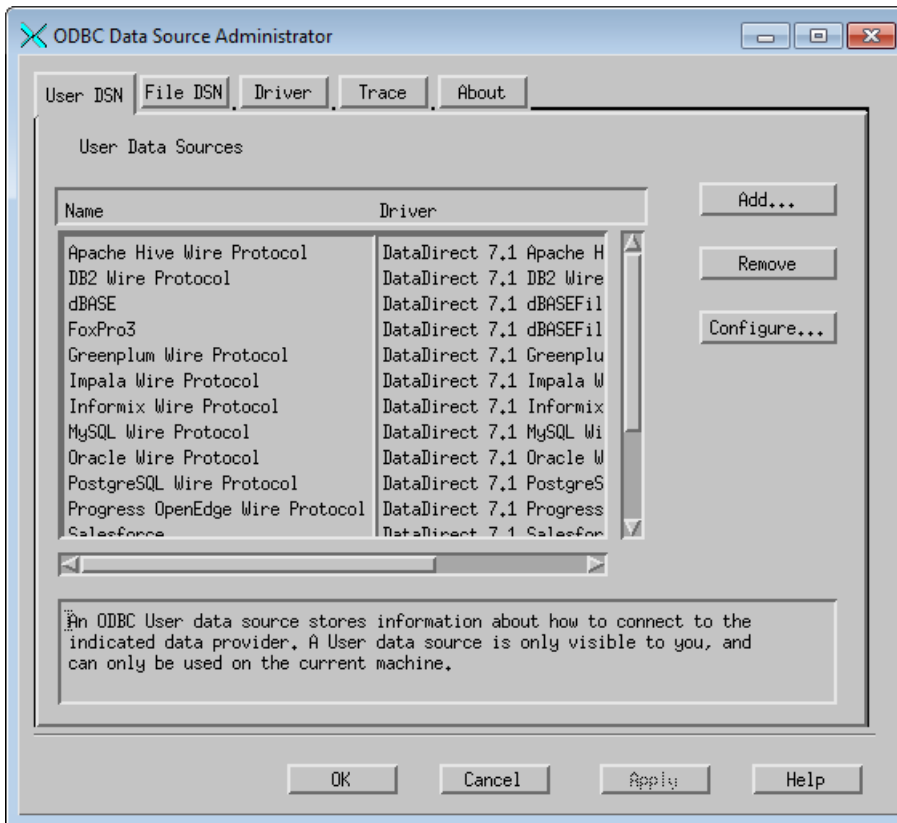
Note that all of the drivers included in your odbc.ini file are shown in the User Data Sources list box, even though some of them cannot be used with the Linux ODBC Administrator.

To configure a data source:

1. To start the Linux ODBC Administrator, change to the *install_dir/tools* directory, where *install_dir* is the path to the product installation directory. At a command prompt, enter:

```
./odbcadmin
```

The Administrator dialog box appears.



2. Click either the **User DSN** or **File DSN** tab to display a list of data sources.

- **User DSN:** If you are configuring an existing user data source, select the appropriate data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the appropriate driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the appropriate data source file and click **Configure** to display the driver Setup dialog box.

To configure a new file data source, click **Add** to display a list of installed drivers. Select the appropriate driver and click **Advanced** to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

Note: If you want to set a default directory for File DSNs, select the directory from the Directories list; then, click **Set Directory**. The next time that you open the Administrator, it displays data source files from this directory.

The General tab of the driver Setup dialog box appears by default.

See the individual driver chapters for specific information about the driver Setup dialog. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources.

Drivers

The Drivers tab shows a list of all installed ODBC drivers.

Tracing

The Tracing tab allows you to trace calls to ODBC drivers and create a log of the traces for troubleshooting purposes.

To specify the path and name of the trace log file, type the path and name in the Trace File field or click **Browse** to select a log file. If no location is specified, the trace log resides in the working directory of the application you are using.

DataDirect ships an enhanced library to perform tracing. This library appears by default in the Trace Library field. If you want to use a custom library instead, type the path and name of the library in the Trace Library field or click **Browse** to select a library.

To enable tracing, select the **Enable Tracing** check box on the Trace tab of the Administrator. Clear the check box to disable tracing. Tracing continues until you disable it. Be sure to turn off tracing when you are finished reproducing the issue because tracing decreases the performance of your ODBC application.

After making changes on the Trace tab, click **Apply** for them to take effect.

When you are finished with the Linux ODBC Administrator, click **OK** or **Cancel**. If you click **OK**, any changes you have made to the Trace tab are accepted and the Administrator closes. If you click **Cancel**, the Administrator closes without saving any changes.

For a complete discussion of tracing, refer to "ODBC trace" in the *Progress DataDirect for ODBC Drivers Reference*.

Configuration Through the System Information (odbc.ini) File

To configure a data source manually, you edit the odbc.ini file with a text editor. The content of this file is divided into three sections.

At the beginning of the file is a section named [ODBC Data Sources] containing *data_source_name=installed-driver* pairs, for example:

```
Oracle Wire Protocol=DataDirect Oracle Wire Protocol.
```

The driver uses this section to match a data source to the appropriate installed driver.

The [ODBC Data Sources] section also includes data source definitions. The default odbc.ini contains a data source definition for each driver. Each data source definition begins with a data source name in square brackets, for example, [Oracle Wire Protocol 2]. The data source definitions contain connection string *attribute=value* pairs with default values. You can modify these values as appropriate for your system. Descriptions of these attributes are in each individual driver chapter. See [Sample Default odbc.ini File](#) on page 118 for sample data sources.

The second section of the file is named [ODBC File DSN] and includes one keyword:

```
[ODBC File DSN]
DefaultDSNDir=
```

This keyword defines the path of the default location for file data sources (see [File Data Sources](#) on page 136).

Note: This section is not included in the default odbc.ini file that is installed by the product installer. You can add this section manually or, if you are using the Linux ODBC Administrator, it will be added automatically when you click **Set Directory** on the File DSN tab (see [Configuration Through the Administrator](#) on page 115 under [Configuration Through the Administrator](#) on page 115).

The third section of the file is named [ODBC] and includes several keywords, for example:

```
[ODBC]
IANAAppCodePage=4
InstallDir=/opt/odbc
Trace=0
TraceFile=odbttrace.out
TraceDll=/opt/odbc/lib/ivtrc27.so
ODBCTraceMaxFileSize=102400
ODBCTraceMaxNumFiles=10
```

The IANAAppCodePage keyword defines the default value that all UNIX/Linux drivers use if individual data sources have not specified a different value. The default value is 4.

See the individual driver chapters, and refer to "Code page values" in the *Progress DataDirect for ODBC Drivers Reference* for details.

The InstallDir keyword must be included in this section. The value of this keyword is the path to the installation directory under which the /lib and /locale directories are contained. The installation process automatically writes your installation directory to the default odbc.ini file.

For example, if you choose an installation location of /opt/odbc, then the following line is written to the [ODBC] section of the default odbc.ini:

```
InstallDir=/opt/odbc
```

Note: If you are using only DSN-less connections through an odbccinst.ini file and do not have an odbc.ini file, then you must provide [ODBC] section information in the [ODBC] section of the odbccinst.ini file. The drivers and Driver Manager always check first in the [ODBC] section of an odbc.ini file. If no odbc.ini file exists or if the odbc.ini file does not contain an [ODBC] section, they check for an [ODBC] section in the odbccinst.ini file. See [DSN-less Connections](#) on page 132 for details.

ODBC tracing allows you to trace calls to ODBC drivers and create a log of the traces for troubleshooting purposes. The following keywords all control tracing: Trace, TraceFile, TraceDLL, ODBCTraceMaxFileSize, and ODBCTraceMaxNumFiles.

For a complete discussion of tracing, refer to "ODBC trace" in the *Progress DataDirect for ODBC Drivers Reference*.

Sample Default odbc.ini File

The following is a sample odbc.ini file that Setup installs in the installation directory. All occurrences of ODBC_HOME are replaced with your installation directory path during installation of the file. Values that you must supply are enclosed by angle brackets (< >). If you are using the installed odbc.ini file, you must supply the values and remove the angle brackets before that data source section will operate properly. Commented lines are denoted by the # symbol. This sample shows 32-bit drivers with file names beginning with iv. A 64-bit driver file would be identical except that driver names would begin with dd and the list of data sources would include only the 64-bit drivers.

```
[ODBC Data Sources]
Apache Hive Wire Protocol=DataDirect 7.1 Apache Hive WP Driver
DB2 Wire Protocol=DataDirect 7.1 DB2 Wire Protocol
dBASE=DataDirect 7.1 dBASEFile (*.dbf)
FoxPro3=DataDirect 7.1 dBASEFile (*.dbf)
Greenplum Wire Protocol=DataDirect 7.1 Greenplum Wire Protocol
Impala Wire Protocol=DataDirect 7.1 Impala Wire Protocol
Informix Wire Protocol=DataDirect 7.1 Informix Wire Protocol
MySQL Wire Protocol=DataDirect 7.1 MySQL Wire Protocol
Oracle Wire Protocol=DataDirect 7.1 Oracle Wire Protocol
PostgreSQL Wire Protocol=DataDirect 7.1 PostgreSQL Wire Protocol
```

```

Progress OpenEdge Wire Protocol=DataDirect 7.1 Progress OpenEdge Wire Protocol
Salesforce=DataDirect 7.1 Salesforce
SQLServer Wire Protocol=DataDirect 7.1 SQL Server Wire Protocol
Sybase Wire Protocol=DataDirect 7.1 Sybase Wire Protocol
Sybase IQ Wire Protocol=DataDirect 7.1 Sybase IQ Wire Protocol
Teradata=DataDirect 7.1 Teradata
Text=DataDirect 7.1 TextFile(*.*)
Informix=DataDirect 7.1 Informix
Oracle=DataDirect 7.1 Oracle
SQLServ Legacy Wire Protocol=DataDirect 7.1 SQL Server Legacy Wire Protocol

```

```

[Apache Hive Wire Protocol]
Driver=ODBCHOME/lib/ivhive27.so
Description=DataDirect 7.1 Apache Hive WP Driver
AllowedOpenSSLVersions=1.1.1,1.0.2
ArraySize=50000
AuthenticationMethod=0
BatchMechanism=2
CryptoLibName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1,SSLv3
Database=<database_name>
EnableDescribeParam=0
EncryptionMethod=0
GSSClient=native
HostName=
HostNameInCertificate=
KeepAlive=0
KeyPassword=
KeystorePassword=
LoginTimeout=30
LogonID=
MaxVarcharSize=2147483647
PortNumber=10000
ProxyUser=
RemoveColumnQualifiers=0
ServicePrincipalName=
SSLLibName=
StringDescribeType=12
TransactionMode=0
Truststore=
TruststorePassword=
UseNativeCatalogFunctions=0
UseCurrentSchema=0
ValidateServerCertificate=1
WireProtocolVersion=0

```

```

[DB2 Wire Protocol]
Driver=ODBCHOME/lib/ivdb227.so
Description=DataDirect 7.1 DB2 Wire Protocol
AccountingInfo=
AddStringToCreateTable=
AllowedOpenSSLVersions=1.1.1,1.0.2
AlternateID=
AlternateServers=
ApplicationName=
ApplicationUsingThreads=1
AuthenticationMethod=0
BulkBinaryThreshold=32
BulkCharacterThreshold=-1
BulkLoadBatchSize=1024
BulkLoadFieldDelimiter=
BulkLoadRecordDelimiter=
CatalogSchema=
CharsetFor65535=0
ClientHostName=
ClientUser=
#Collection applies to DB2 for z/OS and DB2 for i series only
Collection=
ConcurrentAccessResolution=0

```

```
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
CryptoLibName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1,SSLv3
CurrentFunctionPath=
#Database applies to DB2 UDB only
Database=<database_name>
DefaultIsolationLevel=1
DynamicSections=1000
EnableBulkLoad=0
EncryptionMethod=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchTSWTZasTimestamp=0
GrantAuthid=PUBLIC
GrantExecute=1
GSSClient=native
HostNameInCertificate=
IpAddress=<DB2_server_host>
KeepAlive=0
KeyPassword=
KeyStore=
KeyStorePassword=
LoadBalanceTimeout=0
LoadBalancing=0
#Location applies to DB2 for z/OS and DB2 for i series only
Location=<location_name>
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinLongVarcharSize=
MinPoolSize=0
Password=
PackageCollection=NULLID
PackageNamePrefix=DD
PackageOwner=
Pooling=0
ProgramID=
QueryTimeout=0
ReportCodePageConversionErrors=0
SSLLibName=
TcpPort=50000
TrustStore=
TrustStorePassword=
UseCurrentSchema=1
ValidateServerCertificate=1
VarcharThreshold=
WithHold=1
XMLDescribeType=-10

[dBASE]
Driver=ODBCHOME/lib/ivdbf27.so
Description=DataDirect 7.1 dBASEFile (*.dbf)
ApplicationUsingThreads=1
CacheSize=4
CreateType=dBASE5
Database=ODBCHOME/demo
DataFileExtension=DBF
ExtensionCase=UPPER
FileOpenCache=0
IntlSort=0
LockCompatibility=dBASE
Locking=RECORD
UseLongNames=0
UseLongQualifiers=0

[FoxPro3]
```



```

Driver=ODBCHOME/lib/ivdbf27.so
Description=DataDirect 7.1 dBASEFile (*.dbf)
ApplicationUsingThreads=1
CacheSize=4
CreateType=FoxPro30
Database=ODBCHOME/demo
DataFileExtension=DBF
ExtensionCase=UPPER
FileOpenCache=0
IntlSort=0
LockCompatibility=Fox
Locking=RECORD
UseLongNames=0
UseLongQualifiers=0

[Greenplum Wire Protocol]
Driver=ODBCHOME/lib/ivgplm27.so
Description=DataDirect 7.1 Greenplum Wire Protocol
AllowedOpenSSLVersions=1.1.1,1.0.2
AlternateServers=
ApplicationUsingThreads=1
AuthenticationMethod=0
BatchMechanism=1
ConnectionRetryCount=0
ConnectionRetryDelay=3
CryptoLibName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1,SSLv3
Database=<database_name>
EnableDescribeParam=1
EnableKeysetCursors=0
EncryptionMethod=0
ExtendedColumnMetaData=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchRefCursors=1
FetchTSWTZasTimestamp=0
FetchTWFSasTime=0
GSSClient=native
HostName=<Greenplum_host>
HostNameInCertificate=
IANAAppCodePage=4
InitializationString=
KeepAlive=0
KeyPassword=
KeysetCursorOptions=0
KeyStore=
KeyStorePassword=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxCharSize=
MaxLongVarcharSize=
MaxPoolSize=100
MaxVarcharSize=
MinPoolSize=0
Password=
Pooling=0
PortNumber=<Greenplum_server_port>
QueryTimeout=0
ReportCodepageConversionErrors=0
ServicePrincipalName=
SSLLibName=
TransactionErrorBehavior=1
TrustStore=
TrustStorePassword=
UnboundedNumericPrecision=1000
UnboundedNumericScale=6

```

```
ValidateServerCertificate=1
XMLDescribeType=-10

[Impala Wire Protocol]
Driver=ODBCHOME/lib/ivimpala27.so
Description=DataDirect Impala Wire Protocol Driver
AllowedOpenSSLVersions=1.1.1,1.0.2
ArraySize=50000
AuthenticationMethod=0
BatchMechanism=2
CryptoProtocolVersion=TLSv1.2,TLSv1.1,TLSv1,SSLv3
CryptoLibName=
Database=<database_name>
DefaultLongDataBuffLen=1024
DefaultOrderByLimit=-1
EnableDescribeParam=0
EncryptionMethod=0
GSSClient=native
HostName=<host_name>
HostNameInCertificate=
KeyPassword=
Keystore=
KeystorePassword=
LoginTimeout=30
LogonID=
MaxVarcharSize=
PortNumber=<Impala_server_port>
ProxyUser=
RemoveColumnQualifiers=0
ServicePrincipalName=
SSLLibName=
StringDescribeType=-9
TransactionMode=0
Truststore=
TruststorePassword=
UseCurrentSchema=0
ValidateServerCertificate=1

[Informix Wire Protocol]
Driver=ODBCHOME/lib/ivifcl27.so
Description=DataDirect 7.1 Informix Wire Protocol
AlternateServers=
ApplicationUsingThreads=1
CancelDetectInterval=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=<database_name>
HostName=<Informix_host>
LoadBalancing=0
LogonID=
Password=
PortNumber=<Informix_server_port>
ServerName=<Informix_server>
TrimBlankFromIndexName=1
UseDelimitedIdentifiers=0

[MySQL Wire Protocol]
Driver=ODBCHOME/lib/ivmysql27.so
Description=DataDirect 7.1 MySQL Wire Protocol
AllowedOpenSSLVersions=1.1.1,1.0.2
AlternateServers=
ApplicationUsingThreads=1
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
CryptoLibName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1,SSLv3
Database=<database_name>
DefaultLongDataBuffLen=1024
```

```

EnableDescribeParam=0
EncryptionMethod=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
HostName=<MySQL_host>
HostNameInCertificate=
InteractiveClient=0
KeepAlive=0
KeyPassword=
Keystore=
KeystorePassword=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
Password=
MaxPoolSize=100
MinPoolSize=0
Pooling=0
PortNumber=<MySQL_server_port>
QueryTimeout=0
ReportCodepageConversionErrors=0
SSLLibName=
TreatBinaryAsChar=0
TrustStore=
TrustStorePassword=
ValidateServerCertificate=1

[Oracle Wire Protocol]
Driver=ODBCHOME/lib/ivora27.so
Description=DataDirect 7.1 Oracle Wire Protocol
AccountingInfo=
Action=
AllowedOpenSSLVersions=1.1.1,1.0.2
AlternateServers=
ApplicationName=
ApplicationUsingThreads=1
ArraySize=60000
AuthenticationMethod=1
BulkLoadBatchSize=1024
BulkBinaryThreshold=32
BulkCharacterThreshold=-1
BulkLoadFieldDelimiter=
BulkLoadOptions=0
BulkLoadRecordDelimiter=
CachedCursorLimit=32
CachedDescLimit=0
CatalogIncludesSynonyms=1
CatalogOptions=0
ClientHostName=
ClientID=
ClientUser=
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
CryptoLibName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1,SSLv3
DataIntegrityLevel=0
DataIntegrityTypes=
DefaultLongDataBufLen=1024
DescribeAtPrepare=0
EditionName=
EnableBulkLoad=0
EnableDescribeParam=0
EnableNcharSupport=0
EnableScrollableCursors=1
EnableServerResultCache=0
EnableStaticCursorsForLongData=0

```

```
EnableTimestampWithTimeZone=0
EncryptionLevel=0
EncryptionMethod=0
EncryptionTypes=
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchTSWTZasTimestamp=0
GSSClient=native
HostName=<Oracle_server>
HostNameInCertificate=
InitializationString=
KeepAlive=0
KeyPassword=
KeyStore=
KeyStorePassword
LoadBalanceTimeout=0
LoadBalancing=0
LocalTimeZoneOffset=
LockTimeOut=-1
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
Module=0
Password=
Pooling=0
PortNumber=<Oracle_server_port>
ProcedureRetResults=0
ProgramID=
PRNGSeedFile=/dev/random
PRNGSeedSource=0
QueryTimeout=0
ReportCodePageConversionErrors=0
ReportRecycleBin=0
ServerName=<server_name_in_tnsnames.ora>
ServerType=0
ServiceName=
SID=<Oracle_System_Identifier>
SSLLibName=
TimestampEscapeMapping=0
TNSNamesFile=<tnsnames.ora_filename>
TrustStore=
TrustStorePassword=
UseCurrentSchema=1
ValidateServerCertificate=1
WireProtocolMode=1

[PostgreSQL Wire Protocol]
Driver=ODBCHOME/lib/ivpsql27.so
Description=DataDirect 7.1 PostgreSQL Wire Protocol
AlternateServers=
AllowedOpenSSLVersions=1.1.1,1.0.2
ApplicationUsingThreads=1
AuthenticationMethod=0
BatchMechanism=2
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
CryptoLibName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1,SSLv3
Database=<database_name>
EnableDescribeParam=1
EnableKeysetCursors=0
EncryptionMethod=0
ExtendedColumnMetaData=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
```

```

FetchTSWTZasTimestamp=0
FetchTWFSasTime=0
GSSClient=native
HostName=<PostgreSQL_host>
HostNameInCertificate=
InitializationString=
KeepAlive=0
KeyPassword=
KeysetCursorOptions=0
Keysetstore=
KeystorePassword=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxLongVarcharSize=
MaxPoolSize=100
MaxVarcharSize=
MinPoolSize=0
Password=
Pooling=0
PortNumber=<PostgreSQL_server_port>
QueryTimeout=0
ReportCodepageConversionErrors=0
ServicePrincipalName=
SSLLibName=
TransactionErrorBehavior=1
TrustStore=
TrustStorePassword=
UnboundedNumericPrecision=1000
UnboundedNumericScale=6
ValidateServerCertificate=1
XMLDescribeType=-10

[Progress OpenEdge Wire Protocol]
Driver=ODBCHOME/lib/ivoe27.so
Description=DataDirect 7.1 Progress OpenEdge Wire Protocol
AllowedOpenSSLVersions=1.1.1,1.0.2
AlternateServers=
ArraySize=
ConnectionRetryCount=0
ConnectionRetryDelay=3
CryptoLibName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1,SSLv3
Database=<database_name>
DefaultIsolationLevel=1
EnableTimestampWithTimezone=1
Encryption Method=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
HostName=<Progress_server>
HostNameInCertificate=
KeepAlive=0
LoadBalancing=0
LoginTimeout=15
PortNumber=<Progress_server_port>
QueryTimeout=0
SSLLibName=
TrustStore=
TrustStorePassword=
UseWideCharacterTypes=0
ValidateServerCertificate=1

[Salesforce]
Driver=ODBCHOME/lib/ivsfr27.so
Description=DataDirect 7.1 Salesforce
ApplicationUsingThreads=1
BulkLoadAsync=0

```

```
BulkLoadBatchSize=1024
BulkLoadConcurrencyMode=1
BulkLoadPollInterval=10
BulkLoadProtocol=0
BulkLoadThreshold=4000
BulkLoadTimeout=0
ConnectionReset=0
ConfigOptions=
CreateDB=2
EnableBulkLoad=False
Database=
ExtendedOptions=
FetchSize=100
HostName=
InitializationString=
IANAAppCodePage=
JVMArgs=-Xmx256m
JVMClasspath=
LoadBalanceTimeout=0
LogConfigFile=
LoginTimeout=15
LogonDomain=
LogonID=
MaxPoolSize=100
MinPoolSize=0
Password=
Pooling=0
ProxyHost=
ProxyPassword=
ProxyPort=
ProxyUser=
QueryTimeout=0
ReportCodepageConversionErrors=0
ReadOnly=0
RefreshCurrentSchema=0
RefreshDirtyCache=0
SecurityToken=
StmtCallLimit=20
StmtCallLimitBehavior=2
TransactionMode=0
WSFetchSize=0
WSRetryCount=0
WSTimeout=120

[SQLServer Wire Protocol]
Driver=ODBCHOME/lib/ivsqs27.so
Description=DataDirect 7.1 SQL Server Wire Protocol
AllowedOpenSSLVersions=1.1.1,1.0.2
AlternateServers=
AlwaysReportTriggerResults=0
AnsiNPW=1
ApplicationIntent=0
ApplicationName=
ApplicationUsingThreads=1
AuthenticationMethod=1
BulkBinaryThreshold=32
BulkCharacterThreshold=-1
BulkLoadBatchSize=1024
BulkLoadFieldDelimiter=
BulkLoadOptions=2
BulkLoadRecordDelimiter=
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
CryptoLibName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1,SSLv3
Database=<database_name>
DefaultLongDataBuffLen=1024
EnableBulkLoad=0
```

```
EnableQuotedIdentifiers=0
EncryptionMethod=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchTSWTZasTimestamp=0
FetchTWFSasTime=1
GSSClient=native
HostName=<SQL_Server_host>
HostNameInCertificate=
InitializationString=
KeepAlive=0
Language=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
MultiSubnetFailover=0
PacketSize=-1
Password=
Pooling=0
PortNumber=<SQL_Server_server_port>
PRNGSeedFile=/dev/random
PRNGSeedSource=0
QueryTimeout=0
ReportCodePageConversionErrors=0
SnapshotSerializable=0
SSLLibName=
TrustStore=
TrustStorePassword=
ValidateServerCertificate=1
WorkStationID=
XMLDescribeType=-10

[Sybase Wire Protocol]
Driver=ODBCHOME/lib/ivase27.so
Description=DataDirect 7.1 Sybase Wire Protocol
AlternateServers=
ApplicationName=
ApplicationUsingThreads=1
ArraySize=50
AuthenticationMethod=0
BulkBinaryThreshold=32
BulkCharacterThreshold=-1
BulkLoadBatchSize=1024
BulkLoadFieldDelimiter=
BulkLoadRecordDelimiter=
Charset=
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
CryptoLibName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1,SSLv3
CursorCacheSize=1
Database=<database_name>
DefaultLongDataBuffLen=1024
DistributedTransactionModel=0
EnableBulkLoad=0
EnableDescribeParam=0
EnableQuotedIdentifiers=0
EncryptionMethod=0
FailoverGranularity=0
FailoverMode=0
FailoverNetworkAddress=
FailoverPreconnect=0
FetchTWFSasTime=1
GSSClient=native
```

```
HostNameInCertificate=
InitializationString=
InterfacesFileName=
Language=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
KeepAlive=0
MaxPoolSize=100
MinPoolSize=0
NetworkAddress=<Sybase_host,Sybase_server_port>
OptimizePrepare=1
PacketSize=0
Password=
Pooling=0
PRNGSeedFile=/dev/random
PRNGSeedSource=0
QueryTimeout=0
RaiseErrorPositionBehavior=0
ReportCodePageConversionErrors=0
SelectMethod=0
ServicePrincipalName=
SSLLibName=
TightlyCoupledDistributedTransactions=
TruncateTimeTypeFractions=0
TrustStore=
TrustStorePassword=
ValidateServerCertificate=1
WorkStationID=
XAOpenStringParameters=

[Sybase IQ Wire Protocol]
Driver=ODBCHOME/lib/ivsyiq27.so
Description=DataDirect 7.1 Sybase IQ Wire Protocol
AllowedOpenSSLVersions=1.1.1,1.0.2
AlternateServers=
ApplicationName=
ApplicationUsingThreads=1
ArraySize=50
BulkBinaryThreshold=32
BulkCharacterThreshold=-1
BulkLoadBatchSize=1024
Charset=
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
CursorCacheSize=1
Database=<database_name>
DefaultLongDataBufLen=1024
DistributedTransactionModel=0
EnableBulkLoad=0
FailoverGranularity=0
FailoverMode=0
FailoverNetworkAddress=
FailoverPreconnect=0
FetchTWFSasTime=1
InitializationString=
InterfacesFile=
InterfacesFileName=
Language=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
NetworkAddress=<SybaseIQ_host,SybaseIQ_server_port>
OptimizePrepare=1
```



```
PacketSize=0
Password=
Pooling=0
QueryTimeout=0
RaiseErrorPositionBehavior=0
ReportCodePageConversionErrors=0
SelectMethod=0
ServicePrincipalName=
TightlyCoupledDistributedTransactions=
TruncateTimeTypeFractions=0
ValidateServerCertificate=1
WorkStationID=
XAOpenStringParameters=

[Teradata]
Driver=ODBCHOME/lib/ivtera27.so
Description=DataDirect 7.1 Teradata
AccountString=
AuthenticationDomain=
AuthenticationPassword=
AuthenticationUserid=
CharacterSet=ASCII
Database=
DBCName=<Teradata_server>
EnableDataEncryption=0
EnableExtendedStmtInfo=0
EnableLOBs=1
EnableReconnect=0
IntegratedSecurity=0
LoginTimeout=20
LogonID=
MapCallEscapeToExec=0
MaxRespSize=8192
Password=
PortNumber=1025
PrintOption=N
ProcedureWithSplSource=Y
ReportCodePageConversionErrors=0
SecurityMechanism=
SecurityParameter=
ShowSelectableTables=1
TDProfile=
TDRole=
TDUserName=

[Text]
Driver=ODBCHOME/lib/ivtxt27.so
Description=DataDirect 7.1 TextFile(*.*)
AllowUpdateAndDelete=0
ApplicationUsingThreads=1
CacheSize=4
CenturyBoundary=20
Database=ODBCHOME/demo
DataFileExtension=TXT
DecimalSymbol=.
Delimiter=,
FileOpenCache=0
FirstLineNames=0
IntlSort=0
ScanRows=25
TableType=Comma
UndefinedTable=GUESS

[Informix]
Driver=ODBCHOME/lib/ivinf27.so
Description=DataDirect 7.1 Informix
ApplicationUsingThreads=1
CancelDetectInterval=0
CursorBehavior=0
```

```
Database=<database_name>
EnableInsertCursors=0
GetDBListFromInformix=1
HostName=<Informix_host>
LogonID=
Password=
Protocol=onsoctcp
ServerName=<Informix_server>
Service=<Informix_service_name>
TrimBlankFromIndexName=1

[Oracle]
Driver=ODBCHOME/lib/ivor827.so
Description=DataDirect 7.1 Oracle
AlternateServers=
ApplicationUsingThreads=1
ArraySize=60000
CatalogIncludesSynonyms=1
CatalogOptions=0
ClientVersion=9iR2
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
DefaultLongDataBuffLen=1024
DescribeAtPrepare=0
EnableDescribeParam=0
EnableNcharSupport=0
EnableScrollableCursors=1
EnableStaticCursorsForLongData=0
EnableTimestampWithTimeZone=0
LoadBalanceTimeout=0
LoadBalancing=0
LocalTimeZoneOffset=
LockTimeOut=-1
LogonID=
MaxPoolSize=100
MinPoolSize=0
OptimizeLongPerformance=0
Password=
Pooling=0
ProcedureRetResults=0
ReportCodePageConversionErrors=0
ReportRecycleBin=0
ServerName=<Oracle_server>
TimestampEscapeMapping=0
UseCurrentSchema=1

[SQLServ Legacy Wire Protocol]
Driver=ODBCHOME/lib/ivmsss27.so
Description=DataDirect 7.1 SQL Server Legacy Wire Protocol
Address=<SQLServer_host, SQLServer_server_port>
AlternateServers=
AnsiNPW=Yes
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=<database_name>
FetchTSWTZasTimestamp=0
FetchTWFSasTime=0
LoadBalancing=0
LogonID=
Password=
QuotedId=No
ReportCodepageConversionErrors=0
SnapshotSerializable=0

[ODBC]
IANAAppCodePage=4
InstallDir=ODBCHOME
Trace=0
```

```
TraceFile=odbctrace.out
TraceDll=ODBCHOME/lib/ivtrc27.so
ODBCTraceMaxFileSize=102400
ODBCTraceMaxNumFiles=10
```

```
[ODBC File DSN]
DefaultDSNDir=
UseCursorLib=0
```

To modify or create data sources in the `odbc.ini` file, use the following procedures.

- **To modify a data source:**

- Using a text editor, open the `odbc.ini` file.
- Modify the default attributes in the data source definitions as necessary based on your system specifics, for example, enter the host name and port number of your system in the appropriate location.
Consult the "Connection String Attributes" table of each driver chapter for other specific attribute values.
- After making all modifications, save the `odbc.ini` file and close the text editor.

Important: The "Connection Option Descriptions" section of each driver chapter lists both the long and short names of the attribute. When entering attribute names into `odbc.ini`, you must use the long name of the attribute. The short name is not valid in the `odbc.ini` file.

- **To create a new data source:**

- Using a text editor, open the `odbc.ini` file.
- Copy an appropriate existing default data source definition and paste it to another location in the file.
- Change the data source name in the copied data source definition to a new name. The data source name is between square brackets at the beginning of the definition, for example, `[Oracle Wire Protocol]`.
- Modify the attributes in the new definition as necessary based on your system specifics, for example, enter the host name and port number of your system in the appropriate location.
Consult the "Connection String Attributes" table of each driver chapter for other specific attribute values.
- In the `[ODBC]` section at the beginning of the file, add a new `data_source_name=installed-driver` pair containing the new data source name and the appropriate installed driver name.
- After making all modifications, save the `odbc.ini` file and close the text editor.

Important: The "Connection String Attributes" table of each driver chapter lists both the long and short name of the attribute. When entering attribute names into `odbc.ini`, you must use the long name of the attribute. The short name is not valid in the `odbc.ini` file.

Translators

Progress DataDirect provides a sample translator named "OEM to ANSI" that provides a framework for coding a translation library. Refer to the `readme.trn` file in the `/samples/src/trn` subdirectory in the product installation directory for details.

To perform a translation with a particular driver, you must include the TranslationSharedLibrary keyword in that driver's data source definition in the `odbc.ini` file. The TranslationSharedLibrary keyword represents the full path to the translation library.

For example, the 32-bit DB2 driver would be:

```
[DB2]
Driver=ODBCHOME/lib/ivdb227.so
Description=DataDirect 7.1 DB2 Wire Protocol
TranslationSharedLibrary=ODBCHOME/lib/ivtrn27.so
```

The TranslationOption keyword is the ASCII representation of the 32-bit integer translation option. Use of the TranslationOption keyword is optional.

The demoodbc Application

Progress DataDirect ships an application, named `demoodbc`, that is installed in the `/samples/demo` subdirectory of the product installation directory. Once you have set up your environment and data source, use the `demoodbc` application to test your connection. The syntax to run the application is:

```
demoodbc -uid user_name -pwd passworddata_source_name
```

For example:

```
demoodbc -uid johndoe -pwd secret DataSource3
```

The `demoodbc` application is coded to execute a Select statement from a table named `emp`. If you have an `emp` table in your database, the results are returned. If you do not have an `emp` table, you receive the message: `Invalid object name 'EMP'`. This message confirms a successful connection to the database.

Refer to the `demoodbc.txt` file in the `demo` subdirectory for an explanation of how to build and use this application.

The example Application

Progress DataDirect ships an application, named `example`, that is installed in the `/samples/example` subdirectory of the product installation directory. Once you have configured your environment and data source, use the `example` application to test passing SQL statements. To run the application, enter `example` and follow the prompts to enter your data source name, user name, and password.

If successful, a `SQL>` prompt appears and you can type in SQL statements, such as `SELECT * FROM table_name`. If `example` is unable to connect to the database, an appropriate error message appears.

Refer to the `example.txt` file in the `example` subdirectory for an explanation of how to build and use this application.

DSN-less Connections

Connections to a data source can be made via a connection string without referring to a data source name (DSN-less connections). This is done by specifying the "DRIVER=" keyword instead of the "DSN=" keyword in a connection string, as outlined in the ODBC specification. A file named `odbcinst.ini` must exist when the driver encounters `DRIVER=` in a connection string.

Setup installs a default version of this file in the product installation directory (see [ODBCINST](#) on page 113 for details about relocating and renaming this file). This is a plain text file that contains default DSN-less connection information. You should not normally need to edit this file. The content of this file is divided into several sections.

At the beginning of the file is a section named `[ODBC Drivers]` that lists installed drivers, for example,

```
DataDirect Oracle Wire Protocol=Installed.
```

This section also includes additional information for each driver.

The next section of the file is named `[Administrator]`. The keyword in this section, `AdminHelpRootDirectory`, is required for the Linux ODBC Administrator to locate its help system. The installation process automatically provides the correct value for this keyword.

The final section of the file is named `[ODBC]`. The `[ODBC]` section in the `odbcinst.ini` file fulfills the same purpose in DSN-less connections as the `[ODBC]` section in the `odbc.ini` file does for data source connections. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for a description of the other keywords this section.

Note: The `odbcinst.ini` file and the `odbc.ini` file include an `[ODBC]` section. If the information in these two sections is not the same, the values in the `odbc.ini` `[ODBC]` section override those of the `odbcinst.ini` `[ODBC]` section.

Sample odbcinst.ini File

The following is a sample `odbcinst.ini`. All occurrences of `ODBCHOME` are replaced with your installation directory path during installation of the file. Commented lines are denoted by the `#` symbol. This sample shows 32-bit drivers with file names beginning with `iv`; a 64-bit driver file would be identical except that driver names would begin with `dd`.

```
[ODBC Drivers]
DataDirect 7.1 Apache Hive WP Driver=Installed
DataDirect 7.1 DB2 Wire Protocol=Installed
DataDirect 7.1 dBASEFile (*.dbf)=Installed
DataDirect 7.1 Greenplum Wire Protocol=Installed
DataDirect 7.1 Impala Wire Protocol=Installed
DataDirect 7.1 Informix Wire Protocol=Installed
DataDirect 7.1 MySQL Wire Protocol=Installed
DataDirect 7.1 Oracle Wire Protocol=Installed
DataDirect 7.1 PostgreSQL Wire Protocol=Installed
DataDirect 7.1 Progress OpenEdge Wire Protocol=Installed
DataDirect 7.1 SQL Server Wire Protocol=Installed
DataDirect 7.1 Sybase Wire Protocol=Installed
DataDirect 7.1 Salesforce=Installed
DataDirect 7.1 Teradata=Installed
DataDirect 7.1 TextFile (*.*)=Installed
DataDirect 7.1 Informix=Installed
DataDirect 7.1 Oracle=Installed
DataDirect 7.1 SQL Server Legacy Wire Protocol=Installed

[Apache Hive Wire Protocol]
Driver=ODBCHOME/lib/ivhive27.so
APILevel=1
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivhive27.so
SQLLevel=0

[DataDirect 7.1 DB2 Wire Protocol]
Driver=ODBCHOME/lib/ivdb227.so
```

```
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivdb27.so
SQLLevel=0

[DataDirect 7.1 dBASEFile (*.dbf)]
Driver=ODBCHOME/lib/ivdbf27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileExtns=*.dbf
FileUsage=1
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivdbf27.so
SQLLevel=0

[DataDirect 7.1 Greenplum Wire Protocol]
Driver=ODBCHOME/lib/ivgplm27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivgplm27.so
SQLLevel=0

[Impala Wire Protocol]
Driver=ODBCHOME/lib/ivimpala27.so
APILevel=1
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivimpala27.so
SQLLevel=0

[DataDirect 7.1 Informix Wire Protocol]
Driver=ODBCHOME/lib/ivifcl27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivifcl27.so
SQLLevel=0

[DataDirect 7.1 MySQL Wire Protocol]
Driver=ODBCHOME/lib/ivmysql27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivmysql27.so
SQLLevel=0

[DataDirect 7.1 Oracle Wire Protocol]
Driver=ODBCHOME/lib/ivora27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivora27.so
SQLLevel=0
```

```
[DataDirect 7.1 PostgreSQL Wire Protocol]
Driver=ODBCHOME/lib/ivpsql27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivpsql27.so
SQLLevel=0

[DataDirect 7.1 Progress OpenEdge Wire Protocol]
Driver=ODBCHOME/lib/ivoe27.so
APILevel=1
ConnectFunctions=YYN
DriverODBCVer=3.52
SQLLevel=0

[DataDirect 7.1 SQL Server Wire Protocol]
Driver=ODBCHOME/lib/ivsqs27.so
APILevel=1
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
SQLLevel=0

[DataDirect 7.1 Sybase Wire Protocol]
Driver=ODBCHOME/lib/ivase27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivase27.so
SQLLevel=0

[DataDirect 7.1 Salesforce]
Driver=ODBCHOME/lib/ivsfr27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
SQLLevel=0

[DataDirect 7.1 Sybase IQ Wire Protocol]
Driver=ODBCHOME/lib/ivsqliq27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
SQLLevel=0

[DataDirect 7.1 Teradata]
Driver=ODBCHOME/lib/ivtera27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivtera27s.so
SQLLevel=0

[DataDirect 7.1 TextFile (*.*)]
Driver=ODBCHOME/lib/ivtxt27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
```

```
FileExtns=*. *
FileUsage=1
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivtxt27.so
SQLLevel=0

[DataDirect 7.1 Informix]
Driver=ODBCHOME/lib/ivinf27.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
SQLLevel=0

[DataDirect 7.1 Oracle]
Driver=ODBCHOME/lib/ivor827.so
APILevel=0
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivor827s.so
SQLLevel=0

[DataDirect 7.1 SQL Server Legacy Wire Protocol]
Driver=ODBCHOME/lib/ivmsss27.so
APILevel=2
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=0
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ivmsss27s.so
SQLLevel=0

[Administrator]
HelpRootDirectory=ODBCHOME/adminhelp

[ODBC]
#This section must contain values for DSN-less connections
#if no odbc.ini file exists. If an odbc.ini file exists,
#the values from that [ODBC] section are used.

IANAAppCodePage=4
InstallDir=ODBCHOME
Trace=0
TraceFile=odbctrace.out
TraceDll=ODBCHOME/lib/odbctrc27.so
ODBCTraceMaxFileSize=102400
ODBCTraceMaxNumFiles=10
```

File Data Sources

The Driver Manager on UNIX and Linux supports file data sources. The advantage of a file data source is that it can be stored on a server and accessed by other machines, either Windows, UNIX, or Linux. See [Quick Start Connect](#) on page 33 for a general description of ODBC data sources on both Windows and UNIX.

A file data source is simply a text file that contains connection information. It can be created through the Linux ODBC Administrator (see [Configuration Through the Administrator](#) on page 115) or it can be created with a text editor. The file normally has an extension of .dsn.

For example, a file data source for the Oracle Wire Protocol driver would be similar to the following:

```
[ODBC]
Driver=DataDirect Oracle Wire Protocol
Port=1522
HostName=ORA2
LogonID=JOHN
Servicename=SALES.US.ACME.COM
CatalogOptions=1
```

It must contain all basic connection information plus any optional attributes. Because it uses the "DRIVER=" keyword, an `odbcinst.ini` file containing the driver location must exist (see [DSN-less Connections](#) on page 132).

The file data source is accessed by specifying the "FILEDSN=" instead of the "DSN=" keyword in a connection string, as outlined in the ODBC specification. The complete path to the file data source can be specified in the syntax that is normal for the machine on which the file is located. For example, on Windows:

```
FILEDSN=C:\Program Files\Common Files\ODBC\DataSources\Oraclewp.dsn
```

or, on UNIX and Linux:

```
FILEDSN=/home/users/john/filedsn/Oraclewp2.dsn
```

If no path is specified for the file data source, the Driver Manager uses the `DefaultDSNDir` property, which is defined in the `[ODBC File DSN]` setting in the `odbc.ini` file to locate file data sources (see [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for details). If the `[ODBC File DSN]` setting is not defined, the Driver Manager uses the `InstallDir` setting in the `[ODBC]` section of the `odbc.ini` file. The Driver Manager does not support the `SQLReadFileDSN` and `SQLWriteFileDSN` functions.

As with any connection string, you can specify attributes to override the default values in the data source:

```
FILEDSN=/home/users/john/filedsn/Oraclewp2.dsn;UID=james;PWD=test01
```

UTF-16 Applications on UNIX and Linux

Because the DataDirect Driver Manager allows applications to use either UTF-8 or UTF-16 Unicode encoding, applications written in UTF-16 for Windows platforms can also be used on UNIX and Linux platforms.

The Driver Manager assumes a default of UTF-8 applications; therefore, two things must occur for it to determine that the application is UTF-16:

- The definition of `SQLWCHAR` in the ODBC header files must be switched from "char *" to "short *." To do this, the application uses `#define SQLWCHARSHORT`.
- The application must set the encoding for the environment or connection using one of the following attributes. If your application passes UTF-8 encoded strings to some connections and UTF-16 encoded strings to other connections in the same environment, encoding should be set for the connection only; otherwise, either method can be used.

- To configure the encoding for the environment, set the ODBC environment attribute `SQL_ATTR_APP_UNICODE_TYPE` to a value of `SQL_DD_CP_UTF16`, for example:

```
rc = SQLSetEnvAttr(*henv,
SQL_ATTR_APP_UNICODE_TYPE, (SQLPOINTER)SQL_DD_CP_UTF16, SQL_IS_INTEGER);
```

- To configure the encoding for the connection only, set the ODBC connection attribute `SQL_ATTR_APP_UNICODE_TYPE` to a value of `SQL_DD_CP_UTF16`. For example:

```
rc = SQLSetConnectAttr(hdbc, SQL_ATTR_APP_UNICODE_TYPE, SQL_DD_CP_UTF16,  
SQL_IS_INTEGER);
```

Drivers for 32-Bit and 64-Bit Platforms

The following sections describe the drivers that are available in both 32- and 64-bit versions. See [Drivers Only Available for 32-Bit Platforms](#) on page 717 and [The Connect XE Drivers](#) on page 827 for information on additional Connect Series drivers.

For details, see the following topics:

- [The DB2 Wire Protocol Driver](#)
- [The Informix Wire Protocol Driver](#)
- [The MySQL Wire Protocol Driver](#)
- [The Oracle Wire Protocol Driver](#)
- [The PostgreSQL Wire Protocol Driver](#)
- [The Progress OpenEdge Wire Protocol Driver](#)
- [The SQL Server Wire Protocol Driver](#)
- [The Sybase Wire Protocol Driver](#)
- [The Oracle Driver](#)
- [The SQL Server Legacy Wire Protocol Driver](#)
- [The Text Driver](#)

The DB2 Wire Protocol Driver

The DataDirect Connect *for* ODBC and DataDirect Connect64 *for* ODBC DB2 Wire Protocol driver (the DB2 Wire Protocol driver) each support the following DB2 database servers:

- Db2 for i
- Db2 for Linux, UNIX, and Windows
- Db2 for z/OS
- Db2 Hosted
- Db2 Warehouse on Cloud (formerly dashDB)

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The DB2 Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect product for the file name of the DB2 Wire Protocol driver.

Driver Requirements

The server requirement for all platforms is the same. The DB2 database can be installed as the Server Version or the Personal Edition.

The driver has no client requirements.

Binding

The driver does not work properly unless bind packages exist on every server to which you intend to connect.

IMPORTANT: You must have the appropriate privileges for the driver to create and bind packages with your user ID. These privileges are:

- BINDADD for binding packages
- CREATEIN on the collection specified by the Package Collection option
- GRANT EXECUTE on the PUBLIC group for executing the packages

These are typically the permissions of a Database Administrator (DBA). If you do not have these privileges, someone who has a user ID with DBA privileges needs to create packages by connecting with the driver.

When connecting for the first time, the driver determines whether bind packages exist on the server. If packages do not exist, the driver creates them automatically using driver data source default values.

Note: The initial driver connection to a particular server may take a few minutes because of the number and size of the packages that must be created on the server. Subsequent connections do not incur this delay.

If you change default values in a data source before connecting with the driver for the first time, the new defaults are used when creating the packages. If you want to change these values after the packages have been created, you can create or modify packages from the Modify Bindings tab of the Setup dialog. See [Modify Bindings tab](#) under [Configuring and Connecting to Data Sources](#) on page 142 for details.

On UNIX and Linux, you can also create or modify packages through a special bind utility. Depending on the platform of the DB2 server, the attribute values that must be set in the data source to bind packages are:

Linux/UNIX/Windows DB2 Servers: IpAddress, Database, TcpPort

DB2 for z/OS and DB2 for i Servers: IpAddress, Location, TcpPort

Other attribute values also affect binding. See the note for [Modify Bindings tab](#) under [Configuring and Connecting to Data Sources](#) on page 142 for details. See [Connection Option Descriptions for DB2](#) on page 160 for a description of these connection string attributes and their values. You must use the default values or specify new ones for these attributes in the DB2 data source section of the `odbc.ini` file before binding. See [Configuring the Product on UNIX/Linux](#) on page 111 for details on creating the DB2 data source.

The bind utility is located in `installation_directory/bin`. After specifying the appropriate connection string attribute values in the `odbc.ini` file, create or modify packages by entering the command:

```
bindxxdsn
```

where `xx` is the driver level number in the driver file name and `dsn` is the ODBC data source name in the `odbc.ini` file. For example:

```
bind27 DB2 Wire Protocol
```

You are prompted for a user ID and password if they are not stored in the data source. If packages are created and bound successfully, a message indicating success appears. If there are problems connecting or creating the packages, an appropriate error message appears.

Creating DB2 Packages Using List Files

You can bind the following list files on your database server to create DB2 packages:

- `DDODBC_LUW.lst` (DB2 for Linux/UNIX/Windows)
- `DDODBC_MVS.lst` (DB2 for z/OS)
- `DDODBC_400.lst` (DB2 for i)

The list files are located in the `\bind\LUW`, `\bind\zOS`, and `\bind\iSeries` directories, respectively, in your DataDirect Connect Series *for* ODBC installation directory. When you bind the list files, if any DataDirect DB2 packages exist, they will be replaced by the new packages. The list files create DB2 packages that, by default, contain 200 dynamic sections and are created in the NULLID collection. You can override the default number of dynamic sections and the collection in which the packages are created by changing the appropriate connection option in the data source, as described previously.

To create DB2 packages by binding list files:

1. Copy the appropriate list (`*.lst`) file and bind (`*.bnd`) files located in the `/bind` directory to a directory on the database server.
2. From the database server directory where you placed the list and bind files, start the DB2 command-line utility. Use the utility to connect to the database where you want to bind the packages. Connect using the following command:

```
connect to database_name user authorization_name using password
```

where:

`database_name` is the name of the database to which you are connecting.

`authorization_name` is the name of the user you are authenticating to the server.

`password` is the user's password.

3. Execute the DB2 bind command:

```
bind @ list_file grant public
```

where *list_file* is the name of the list file you want to bind.

Creating DB2 Packages Manually

On DB2 for z/OS and DB2 for i servers, you can bind files manually to create DB2 packages. Refer to one of the following instruction files, as appropriate:

- CFODBC ZOS MANUAL PACKAGE CREATION INSTRUCTIONS.TXT (DB2 for z/OS)
- CFODBC AS400 MANUAL PACKAGE CREATION INSTRUCTIONS.TXT (DB2 for i)

These instruction files are located in the `bind\ZOS` and `\bind\ISERIES` directories, respectively, in your DataDirect Connect Series *for* ODBC installation directory.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 158 and [Connection Option Descriptions for DB2](#) on page 160 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux `odbc.ini` File

On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, `odbc.ini`). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the `odbc.ini` file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions for DB2](#) on page 160 lists driver connection string attributes that must be used in the `odbc.ini` file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (DB2)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX® On UNIX and Linux, data sources are stored in the `odbc.ini` file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a DB2 data source:

1. Start the ODBC Administrator:


-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- UNIX**® On Linux, change to the `install_dir/tools` directory and, at a command prompt, enter:


```
odbcadmin
```

 where `install_dir` is the path to the product installation directory.

2. Select a tab:

- User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

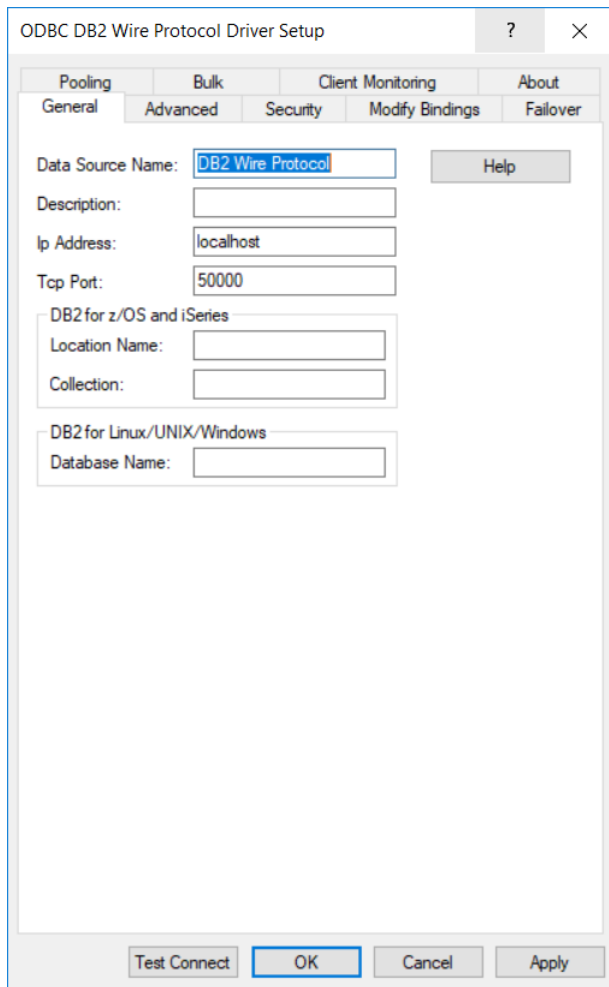
If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
- File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 1: General tab



Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

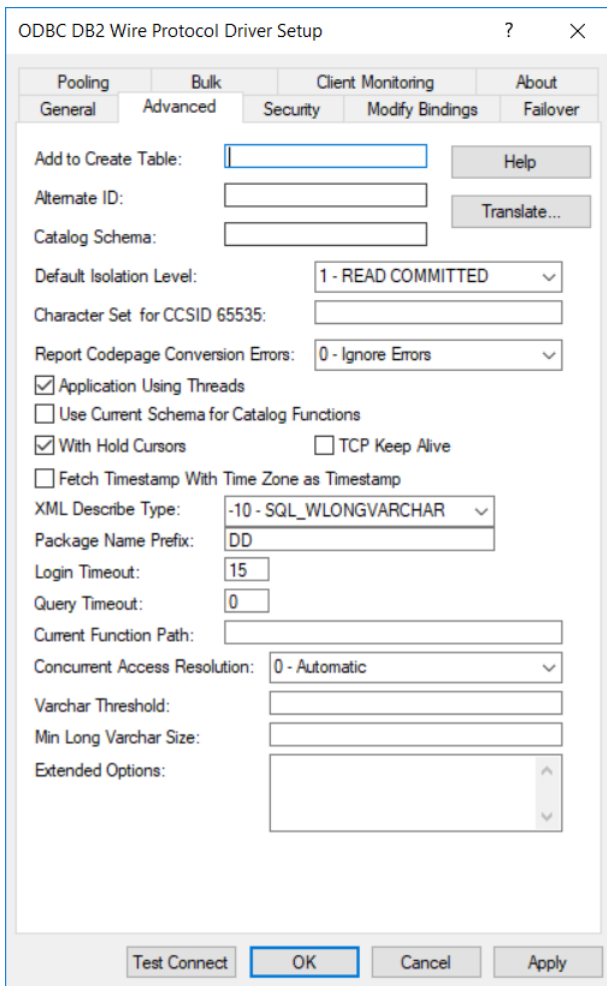
3. On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 180	None
Description on page 183	None
Ip Address on page 191	localhost
Tcp Port on page 206	50000
Location Name on page 195	None

Connection Options: General	Default
Collection on page 173	None
Database Name on page 181	None

4. Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 2: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Add to Create Table on page 163	None
Alternate ID on page 165	None
Catalog Schema on page 170	None
Default Isolation Level on page 181	1 - READ COMMITTED

Connection Options: Advanced	Default
Character Set for CCSID 65535 on page 171	None
Report Codepage Conversion Errors on page 203	0 - Ignore Errors
Application Using Threads on page 167	Enabled
Use Current Schema for Catalog Functions on page 208	Disabled
With Hold Cursors on page 210	Enabled
Fetch Time Stamp With Time Zone as Timestamp on page 187	Disabled
TCP Keep Alive on page 205	Disabled
XML Describe Type on page 211	-10 - SQL_WLONGVARCHAR
Package Name Prefix on page 199	DD
Login Timeout on page 195	15
Query Timeout on page 202	0
Current Function Path on page 180	None
Concurrent Access Resolution on page 174	0 - Automatic
Varchar Threshold on page 209	None
Min Long Varchar Size on page 196	None
IANAAppCodePage on page 190 UNIX ONLY	4 (ISO 8559-1 Latin 1)



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
DefaultIsolationLevel=1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.

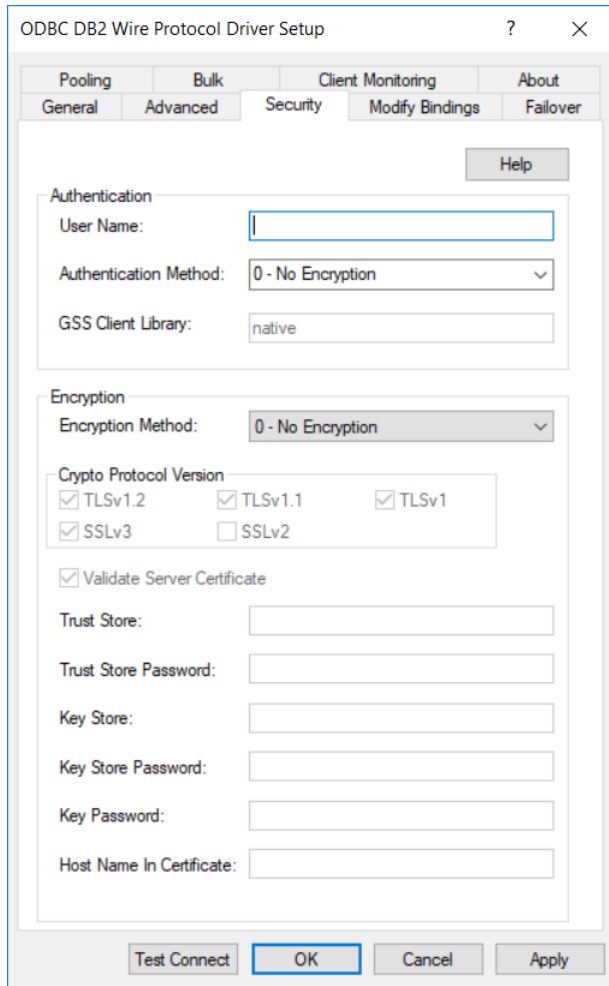


Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- Optionally, click the **Security** tab to specify security data source settings.

Figure 3: Security tab



See [Using Security](#) on page 89 for a general description of authentication and encryption and their configuration requirements.

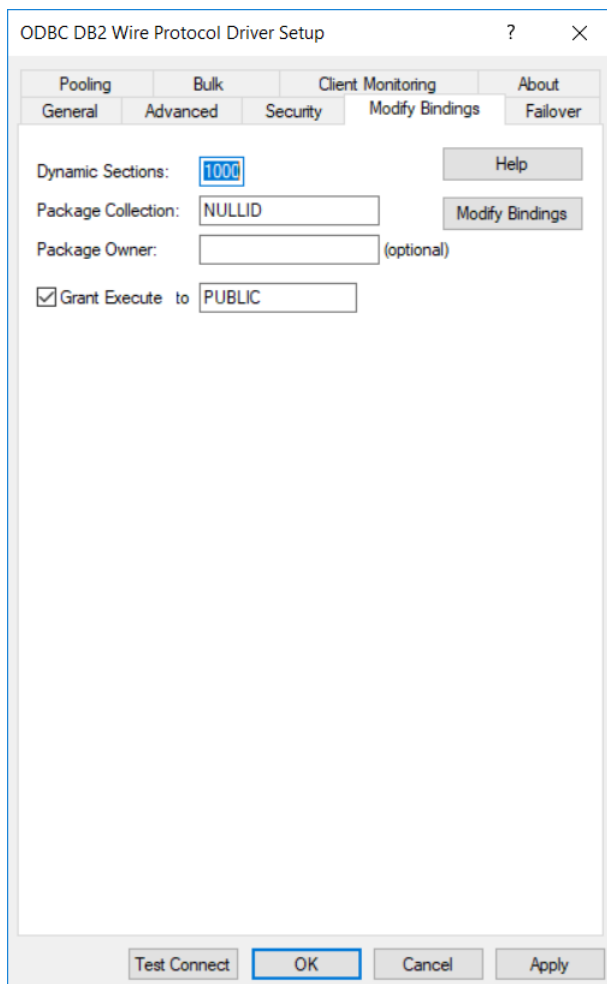
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Security	Default
User Name on page 208	None
Authentication Method on page 168	0 (No Encryption)
GSS Client Library on page 189	native
Encryption Method on page 184	0 (No Encryption)

Connection Options: Security	Default
Crypto Protocol Version on page 178	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Validate Server Certificate on page 209	Enabled
Truststore on page 206	None
Truststore Password on page 207	None
Keystore on page 192	None
Keystore Password on page 193	None
Key Password on page 192	None
Host Name In Certificate on page 189	None

6. Optionally, click the **Modify Bindings** tab to configure options for creating or modifying bind packages.

Figure 4: Modify Bindings tab



The Modify Bindings tab allows you to create or modify bind packages on the server accessed by the driver. If you connect with the driver before explicitly creating bind packages, the driver creates packages on the server automatically using the current values from the Setup dialog box. Alternatively, you can create a bind package before testing the connection. You can also modify bind packages after their creation from the Modify Bindings tab. You must also provide appropriate values for the options on the General tab.

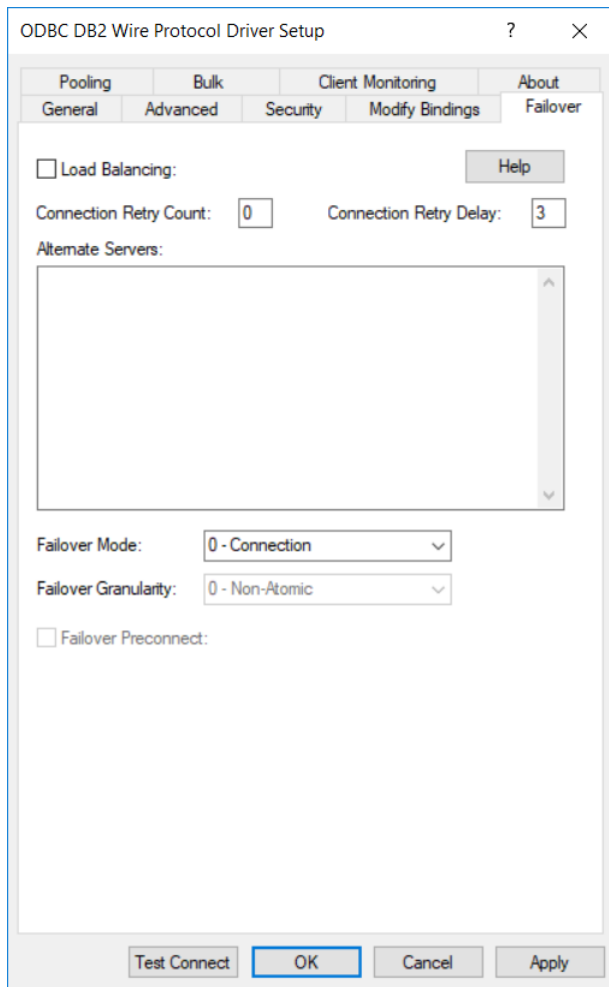
Note: If you change any of the values on this tab after having initially created bind packages, you must rebind the packages. The changes are reflected only on new connections after rebinding.

On this tab, provide values for any of the options in the following table; then, click **Apply**. Optionally, click **Modify Bindings** to create a bind package before connection or modify an existing one. If you do not click **Modify Bindings**, the driver uses the provided settings the next time it establishes a connection. The table provides links to descriptions of the connection options.

Connection Options: Modify Bindings	Default
Dynamic Sections on page 183	1000
Package Name Prefix on page 199	NULLID
Package Collection on page 198	None
Grant Execute to [check box] on page 188	Enabled
Grant Execute to [field] on page 188	PUBLIC

- Optionally, click the **Failover** tab to specify failover data source settings.

Figure 5: Failover tab



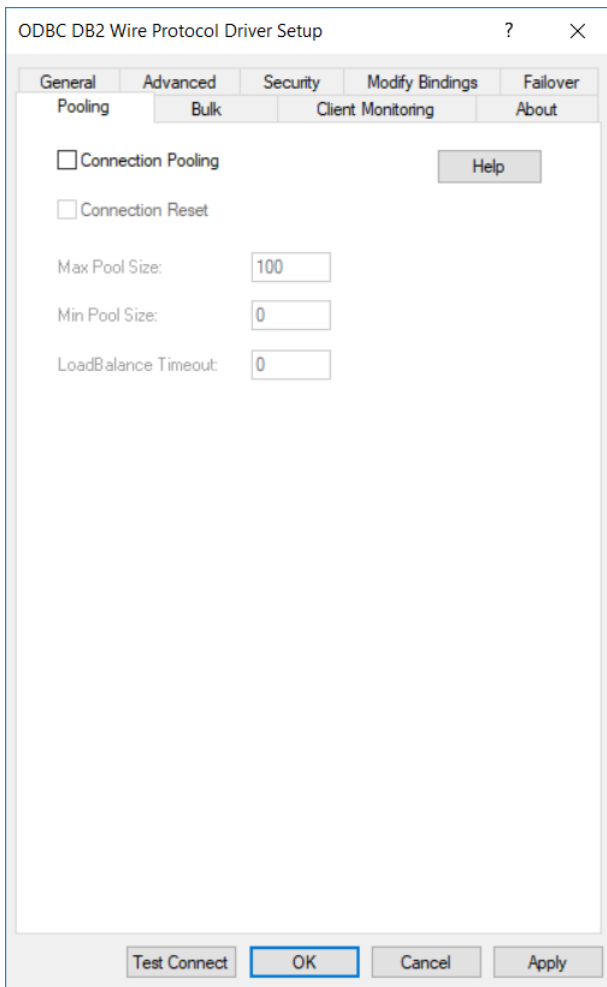
See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 194	Disabled
Connection Retry Count on page 176	0
Connection Retry Delay on page 177	3
Alternate Servers on page 166	None
Failover Mode on page 185	0 (Connection)
Failover Granularity on page 184	0 (Non-Atomic)
Failover Preconnect on page 186	Disabled

8. Optionally, click the **Pooling** tab to specify connection pooling data source settings.

Figure 6: Pooling tab



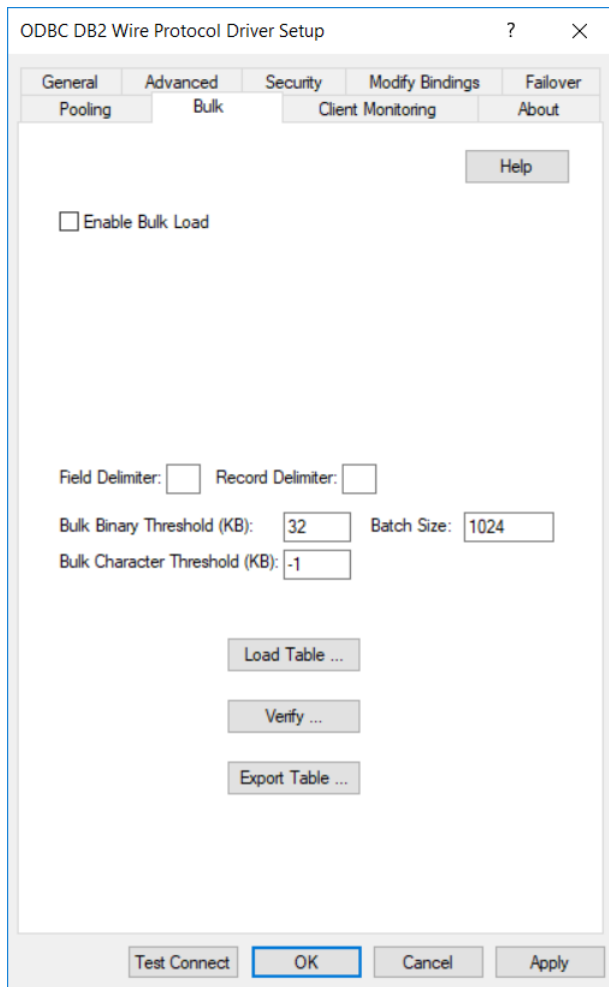
See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 175	Disabled
Connection Reset on page 176	Disabled
Max Pool Size on page 197	100
Min Pool Size on page 198	0
Load Balance Timeout on page 193	0

9. Optionally, click the **Bulk** tab to specify DataDirect Bulk Load data source settings.

Figure 7: Bulk tab



See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect Bulk Load.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Bulk	Default
Enable Bulk Load on page 211	Disabled
Field Delimiter on page 187	None
Record Delimiter on page 203	None
Bulk Binary Threshold on page 169	32
Bulk Character Threshold on page 170	-1
Batch Size on page 169	1024

If your application is already coded to use parameter array batch functionality, you can leverage DataDirect Bulk Load features through the Enable Bulk Load connection option. Enabling this option automatically converts the parameter array batch operation to use the database bulk load protocol.

If you are not using parameter array batch functionality, you can export data to a bulk load data file, verify the metadata of the bulk load configuration file against the structure of the target table, and bulk load data to a table. Use the following steps to accomplish these tasks.

- a) To export data from a table to a bulk load data file, click **Export Table** from the Bulk tab. The Export Table dialog box appears.

Figure 8: Export Table dialog box

Both a bulk data file and a bulk configuration file are produced by exporting a table. The configuration file has the same name as the data file, but with an XML extension. See [Using DataDirect Bulk Load](#) on page 101 for details about these files.

The bulk export operation can create a log file and can also export to external files. See [External Overflow Files](#) on page 108 for more information. The export operation can be configured such that if any errors or warnings occur:

- The operation always completes.
- The operation always terminates.
- The operation terminates after a certain threshold of warnings or errors is exceeded.

Table Name: A string that specifies the name of the source database table containing the data to be exported.

Export Filename: A string that specifies the path (relative or absolute) and file of the bulk load data file to which the data is to be exported. It also specifies the file name of the bulk configuration file. The file name must be the fully qualified path to the bulk data file. These files must not already exist; if one of both of them already exists, an error is returned.

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. The log file is created if it does not exist. The file name must be the fully qualified path to the log file. Events logged to this file are:

- Total number of rows fetched
- A message for each row that failed to export
- Total number of rows that failed to export

- Total number of rows successfully exported

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not supply a value for Log Filename, no log file is created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

Code Page: A value that specifies the code page value to which the driver must convert all data for storage in the bulk data file. See [Character Set Conversions](#) on page 108 for more information.

The default value on Windows is the current code page of the machine. On UNIX/Linux, the default value is 4 (ISO 8559-1 Latin-1).

Click **Export Table** to connect to the database and export data to the bulk data file or click **Cancel**.

To verify the metadata of the bulk load configuration file against the structure of the target database table, click **Verify** from the Bulk tab. See [Verification of the Bulk Load Configuration File](#) on page 106 for details. The Verify dialog box appears.

Figure 9: Verify dialog box

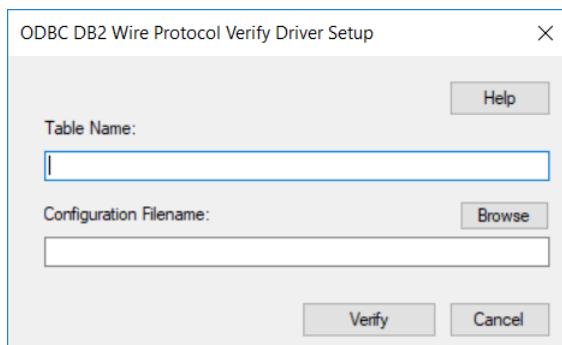


Table Name: A string that specifies the name of the target database table into which the data is to be loaded.

Configuration Filename: A string that specifies the path (relative or absolute) and file name of the bulk configuration file. The file name must be the fully qualified path to the configuration file.

Click **Verify** to verify table structure or click **Cancel**.

- b) To bulk load data from the bulk data file to a database table, click **Load Table** from the Bulk tab. The Load File dialog box appears.

The load operation can create a log file and can also create a discard file that contains rows rejected during the load. The discard file is in the same format as the bulk load data file. After fixing reported issues in the discard file, the bulk load can be reissued using the discard file as the bulk load data file.

Figure 10: Load table dialog box

The export operation can be configured such that if any errors or warnings occur:

- The operation always completes.
- The operation always terminates.
- The operation terminates after a certain threshold of warnings or errors is exceeded.

If a load fails, the Load Start and Load Count options can be used to control which rows are loaded when a load is restarted after a failure.

Table Name: A string that specifies the name of the target database table into which the data is loaded.

Load Data Filename: A string that specifies the path (relative or absolute) and file name of the bulk data file from which the data is loaded. The file name must be the fully qualified path to the bulk data file.

Configuration Filename: A string that specifies the path (relative or absolute) and file name of the bulk configuration file. The file name must be the fully qualified path to the configuration file.

If you do not specify a value for Configuration Filename, the bulk configuration file name is assumed to be `bulk_data_file_name.xml`.

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. The file name must be the fully qualified path to the log file. Specifying a value for Log Filename creates the file if it does not already exist. Events logged to this file are:

- Total number of rows read
- Message for each row that failed to load
- Total number of rows that failed to load

- Total number of rows successfully loaded

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Log Filename, no log file is created.

Discard Filename: A string that specifies the path (relative or absolute) and file name of the bulk discard file. The file name must be the fully qualified path to the discard file. Any row that cannot be inserted into database as result of bulk load is added to this file, with the last row rejected added to the end of the file.

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Discard Filename, a discard file is not created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Load Start: A value that specifies the first row to be loaded from the data file. Rows are numbered starting with 1. For example, when Load Start is 10, the first 9 rows of the file are skipped and the first row loaded is row 10. This option can be used to restart a load after a failure.

The default value is 1.

Read Buffer Size (KB): A value that specifies the size, in KB, of the buffer that is used to read the bulk data file for a bulk load operation.

The default value is 2048.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

Load Count: A value that specifies the number of rows to be loaded from the data file. The bulk load operation loads rows up to the value of Load Count from the file to the database. It is valid for Load Count to specify more rows than exist in the data file. The bulk load operation completes successfully when either the number of rows specified by the Load Count value has been loaded or the end of the data file is reached. This option can be used in conjunction with Load Start to restart a load after a failure.

The default value is the maximum value for SQLULEN. If set to 0, no rows are loaded.

Click **Load Table** to connect to the database and load the table or click **Cancel**.

10. Optionally, click the **Client Monitoring** tab to specify additional data source settings.

Figure 11: Client Monitoring tab

ODBC DB2 Wire Protocol Driver Setup

General Advanced Security Modify Bindings Failover
Pooling Bulk Client Monitoring About

Help

Accounting Info:

Application Name:

Client User:

Client Host Name:

Program ID:

Test Connect OK Cancel Apply

See [Using Client Information](#) on page 87 for additional information about client monitoring.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Client Monitoring	Default
Accounting Info on page 163	None
Application Name on page 166	None
Client User on page 173	None
Client Host Name on page 172	None
Program ID on page 201	None

11. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(DB2\)](#) on page 159 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.

- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
- If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Important: If you have not already created bind packages by clicking the `Modify Bindings` button on the `Modify Bindings` tab, the initial connection through the `Test Connect` button may take a few minutes because of the number and size of the packages that must be created on the server. Subsequent connections occur without this delay.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the `Test Connect` button tests only the primary server, not the alternate servers.

12. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or `FILEDSN` connection string tells the driver where to find the default connection information. Optionally, you may specify `attribute=value` pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]. . .]
```

The `FILEDSN` connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]. . .]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{driver_name}] [;attribute=value[;attribute=value]. . .]
```

[Connection Option Descriptions for DB2](#) on page 160 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for DB2 for Linux/UNIX/Windows is:

```
DSN=DB2ACCOUNT;DB=DB2DATA;UID=JOHN;PWD=XYZZY
```

A `FILEDSN` connection string is similar except for the initial keyword:

```
FILEDSN=DB2.dsn;DB=DB2DATA;UID=JOHN;PWD=XYZZY
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 DB2 Wire Protocol};IpAddress=123.456.78.90;
PORT=5179;DB=DB2DATA;UID=JOHN;PWD=XYZZY
```

Using a Logon Dialog Box (DB2)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

In this dialog box, provide the following information:

1. In the Ip Address field, type the IP (Internet Protocol) address of the machine where the catalog tables are stored. Specify the address using the machine's numeric address (for example, 123.456.78.90) or specify its host name. If you enter a host name, the driver must find this name (with the correct address assignment) in the HOSTS file on the workstation or in a DNS server. The default is localhost.

The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details concerning these formats.

2. In the Tcp Port field, type the port number that is assigned to the DB2 server on the machine where the catalog tables are stored. Specify either this port's numeric address or its service name (50000 is the default port address). If you specify a service name, the driver must find this name (with the correct port assignment) in the SERVICES file on the workstation.
3. If you are running DB2 for z/OS or DB2 for i, perform Steps 3a and 3b. Otherwise, skip to Step 4.
 - a) In the Location field, type the DB2 location name. Use the name defined during the local DB2 installation.
 - b) By default, the User ID is used for the value of Collection. The User ID must always be used on DB2 for z/OS.

On DB2 for i, you can type the name of the schema that is to be the default qualifier for unqualified object names. If you want to access a table outside of this schema, you need to specify the appropriate two-part name, for example, `SELECT * FROM Schema.Tablename`. On DB2 for i only, Collection is also the current library.

Skip to Step 5.

4. If you are running DB2 for Linux/UNIX/Windows, type the name of the database to which you want to connect in the Database field.
5. If required, type your logon ID in the User Name field.

6. If required, type your password in the Password field.
7. Click **OK** to complete the logon and to update the values in the Registry.

Connection Option Descriptions for DB2

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the DB2 Wire Protocol driver.

Table 17: DB2 Wire Protocol Attribute Names

Attribute (Short Name)	Default
AccountingInfo (AI)	None
AddStringToCreateTable (ASCT)	None
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
AlternateID (AID)	None
AlternateServers (ASRV)	None
ApplicationName (AN)	None
ApplicationUsingThreads (AUT)	1 (Enabled)
AuthenticationMethod (AM)	0 (No Encryption)
BulkBinaryThreshold (BBT)	32
BulkCharacterThreshold (BCT)	-1
BulkLoadBatchSize (BLBS)	1024
BulkLoadFieldDelimiter (BLFD)	None
BulkLoadRecordDelimiter (BLRD)	None
CatalogSchema (CS)	None
CharsetFor65535 (CF6)	0
ClientHostName (CHN)	None

Attribute (Short Name)	Default
ClientUser (CU)	None
Collection (COL)	None
ConcurrentAccessResolution (CAR)	0 (Automatic)
ConnectionReset (CR)	0 (Disabled)
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1, SSLv3
CurrentFunctionPath (CFP)	None
Database (DB)	None
DataSourceName (DSN)	None
DefaultIsolationLevel (DIL)	1 (READ_COMMITTED)
Description (n/a)	None
DynamicSections (DS)	1000
EnableBulkLoad (EBL)	0 (Disabled)
EncryptionMethod (EM)	0 (No Encryption)
FailoverGranularity (FG)	0 (Non-Atomic)
FailoverMode (FM)	0 (Connection)
FailoverPreconnect (FP)	0 (Disabled)
FetchTSWTZasTimestamp	0 (Disabled)
GrantAuthid (GA)	PUBLIC
GrantExecute (GE)	1 (Enabled)
GSSClient (GSSC)	native
HostNameInCertificate (HNIC)	None
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
IpAddress (IP)	localhost

Attribute (Short Name)	Default
KeepAlive (KA)	Disabled
KeyPassword (KP)	None
Keystore (KS)	None
KeystorePassword (KSP)	None
LoadBalanceTimeout (LBT)	0
LoadBalancing (LB)	0 (Disabled)
Location (LOC)	None
LoginTimeout (LT)	15
LogonID (UID)	None
MaxPoolSize (MXPS)	100
MinLongVarcharSize (MINLVS)	None
MinPoolSize (MNPS)	0
PackageCollection (PC)	NULLID
PackageNamePrefix (PNP)	DD
Password (PWD)	None
Pooling (POOL)	0 (disabled)
ProgramID (PID)	None
QueryTimeout (QT)	0
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
SSLLibName (SLN)	Empty string
TcpPort (PORT)	50000
Truststore (TS)	None
TruststorePassword (TSP)	None
UseCurrentSchema (UCS)	0 (disabled)
ValidateServerCertificate (VSC)	1 (enabled)
VarcharThreshold (VT)	None

Attribute (Short Name)	Default
WithHold (WH)	1 (enabled)
XMLDescribeType (XDT)	-10

Accounting Info

Attribute

AccountingInfo (AI)

Purpose

Specifies accounting information to be stored in the database. This value sets the CURRENT CLIENT_ACCTNG register (DB2 for Linux/UNIX/Windows) or the CLIENT ACCTNG register (DB2 for z/OS) in the database. This value is used by the DB2 Workload Manager.

Valid Values

string

where:

string

is the accounting information.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 212

Add to Create Table

Attribute

AddStringToCreateTable (ASCT)

Purpose

Specifies a string that is automatically added to all Create Table statements. This option is for users who need to add an In Database clause to Create Table statements.

Valid Values

string

where:

string

is valid syntax for the In Database clause of a Create Table statement.

Default

None

GUI Tab

[Advanced tab](#)

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

`latest | openssl_version_number[[,openssl_version_number]...]`

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to `openssl_version_number`, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the `CryptoLibName` and `SSLLibName` options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

1.1.1,1.0.2

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

[Advanced tab](#)

Alternate ID

Attribute

AlternateID (AID)

Purpose

Specifies the name of the default schema that is used to qualify unqualified database objects in dynamically prepared SQL statements. If the attempt to change the current schema fails, the connection fails and you receive the message `Invalid value for Alternate ID`. Refer to IBM for i documentation for permission requirements imposed by the database.

Valid Values

string

where:

string

is a valid DB2 schema name.

Default

None

GUI Tab

[Advanced tab](#)

Alternate Servers

Attribute

AlternateServers (ASRV)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
(IPAddress=ipvalue:TcpPort=portvalue:{Database | Location}=  
databasevalue [, . . .])
```

You must specify the IP address, port number, and database name (Linux/UNIX/Windows) or location (DB2 for z/OS and DB2 for i) of each alternate server.

Example

The following Alternate Servers values define two alternate database servers for connection failover:

```
AlternateServers=(IpAddress=123.456.78.90:TcpPort=5177:Database=DB2DAT,  
IpAddress=223.456.78.90:TcpPort=5178:Database=DB2DAT3)
```

or

```
AlternateServers=(IpAddress=123.456.78.90:  
TcpPort=5177:Location=DB2DAT, IpAddress=223.456.78.90:TcpPort=5178:  
Location=DB2DAT3)
```

Notes

- An alternate server address in IPv6 format must be enclosed in double quotation marks.

Default

None

GUI Tab

[Failover tab](#)

Application Name

Attribute

ApplicationName (AN)

Purpose

Specifies the name of the application to be stored in the database. This value sets the CURRENT CLIENT_APPLNAME register (DB2 for Linux/UNIX/Windows) or CLIENT APPLNAME register (DB2 for z/OS) in the database. For DB2 V9.1 and higher for Linux/UNIX/Windows, this value also sets the APPL_NAME value of the SYSIBMADM.APPLICATIONS table. This value is used by the DB2 Workload Manager.

Valid Values

string

where:

string

is the name of the application.

Notes

- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 212

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 212

Authentication Method

Attribute

AuthenticationMethod (AM)

Purpose

Specifies the method the driver uses to authenticate the user to the server when a connection is established. If the specified authentication method is not supported by the database server, the connection fails and the driver generates an error.

Valid Values

0 | 1 | 2 | 3 | 4 | 7 | 8

Behavior

If set to 0 (No Encryption), the driver sends the user ID and password in clear text to the server for authentication.

If set to 1 (Encrypt Password), the driver sends the user ID in clear text and an encrypted password to the server for authentication.

If set to 2 (Encrypt UID and Password), the driver sends an encrypted user ID and password to the server for authentication.

If set to 3 (Client Authentication), the driver uses client authentication when establishing a connection. The database server relies on the client to authenticate the user and does not provide additional authentication.

If set to 4 (Kerberos Authentication), the driver uses Kerberos authentication. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

If set to 7 (Encrypted Password AES), the driver encrypts the password with 256-bit AES encryption in the connection request. (DB2 V9.7 and higher only.)

If set to 8 (Encrypted UID and Password AES), the driver encrypts the user id and password with 256-bit AES encryption in the connection request. (DB2 V9.7 and higher only.)

Notes

- The use of AES encryption (values 7 and 8) requires that the DataDirect OpenSSL library be installed.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

Batch Size

Attribute

BulkLoadBatchSize (BLBS)

Purpose

The number of rows that the driver sends to the database at a time during bulk operations. This value applies to all methods of bulk loading.

Valid Values

0 | x

where:

x

is a positive integer that specifies the number of rows to be sent.

Default

1024

GUI Tab

[Bulk tab](#)

Bulk Binary Threshold

Attribute

BulkBinaryThreshold (BBT)

Purpose

The maximum size, in KB, of binary data that is exported to the bulk data file.

Valid Values

-1 | 0 | x

where:

x

is an integer that specifies the number of KB.

Behavior

If set to -1, all binary data, regardless of size, is written to the bulk data file, not to an external file.

If set to 0, all binary data, regardless of size, is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

If set to x , any binary data exceeding this specified number of KB is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

Default

32

GUI Tab

[Bulk tab](#)

Bulk Character Threshold

Attribute

BulkCharacterThreshold (BCT)

Purpose

The maximum size, in KB, of character data that is exported to the bulk data file.

Valid Values

-1 | 0 | x

where:

x

is an integer that specifies the number of KB.

Behavior

If set to -1, all character data, regardless of size, is written to the bulk data file, not to an external file.

If set to 0, all character data regardless of size, is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

If set to x, any character data exceeding this specified number of KB is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

Default

-1

GUI Tab

[Bulk tab](#)

Catalog Schema

Attribute

CatalogSchema (CS)

Purpose

Specifies the DB2 schema to use for Catalog functions. Specifying a schema allows you to use copies or views of the system catalog tables for catalog functions.

Valid Values

schema_name

where:

schema_name

is the name of a valid DB2 schema. If you do not specify a value for this attribute, the driver uses SYSIBM when connected to DB2 for z/OS, QSYS2 when connected to DB for ii, and SYSCAT when connected to Linux/UNIX/Windows.

Example

Create a view DB2ADMIN.TABLES from SYSCAT.TABLES.

```
CREATE VIEW DB2ADMIN.TABLES AS SELECT * FROM SYSCAT.TABLES WHERE OWNER LIKE 'ODBC%'
```

Set CatalogSchema=DB2ADMIN, and do the SQLTables thing.

```
"TABLE_CAT", "TABLE_SCHEM", "TABLE_NAME", "TABLE_TYPE", "REMARKS"
```

The results come from the DB2ADMIN.TABLES view. Three rows are fetched from five columns.

```
<Null>, "DB2ADMIN", "BUG", "TABLE", <Null>
<Null>, "DB2ADMIN", "DATETEST", "TABLE", <Null>
<Null>, "DB2ADMIN", "TESTCP", "TABLE", <Null>
```

Default

None

GUI Tab

[Advanced tab](#)

Character Set for CCSID 65535

Attribute

CharsetFor65535 (CF6)

Purpose

Specifies the IANA code page to be used by the driver to convert character data stored as bit data in character columns (Char, Varchar, Longvarchar, Clob, Char for Bit Data, Varchar for Bit Data, Longvarchar for Bit Data) defined with CCSID 65535.

Valid Values

0 | IANA_code_page

where:

IANA_code_page

is a valid IANA code page. Refer to "IBM to IANA code page values" in the *Progress DataDirect for ODBC Drivers Reference* for a list of the most commonly used IBM code pages and their IANA code page equivalents.

Behavior

If unspecified or set to 0, the driver returns these columns as binary columns (SQL_BINARY, SQL_VARBINARY, SQL_LONGVARBINARY) and does no conversion of the data.

If an IANA code page is specified, the driver returns these columns as character columns in the character set specified. The driver does no conversion of data supplied in bound parameters.

Default

0

GUI Tab

[Advanced tab](#)

Client Host Name

Attribute

ClientHostName (CHN)

Purpose

Specifies the host name of the client machine to be stored in the database. This value sets the CURRENT CLIENT_WRKSTNNAME register (DB2 for Linux/UNIX/Windows) or CLIENT WRKSTNNAME register (DB2 for z/OS) in the database. This value is used by the DB2 Workload Manager.

Valid Values

string

where:

string

is the host name of the client machine.

Notes

- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 212

Client User

Attribute

ClientUser (CU)

Purpose

The user ID to be stored in the database. This option sets the CURRENT CLIENT_USERID register (DB2 for Linux/UNIX/Windows) and CLIENT USERID register (DB2 for z/OS) in the database. This value is used by the DB2 Workload Manager.

Valid Values

-1 | client_userid

where:

client_userid

is a valid user ID.

Behavior

When set to -1, the driver uses the userid of the user that is currently logged onto the client.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 212

Notes

- This connection option can affect performance.

Collection

Attribute

Collection (COL)

Purpose

The current collection or library. Valid only on DB2 for z/OS and DB2 for i.

For DB2 for z/OS, this value is the user ID. If an attempt to change the current schema fails, the connection fails and you receive the message `Invalid value for Alternate ID.`

For DB2 for i, this value is the name of the schema to be used as the default qualifier for unqualified object names. If you want to access a table outside of this schema, you must specify the schema name and the table name. For example:

```
SELECT * FROM Schema.Tablename
```

Also, if the Alternate ID option is set, it overrides this option.

Valid Values

user_ID (DB2 for z/OS) | schema_name (DB2 for i)

where:

user_ID

is a valid user ID. DB2 permissions on the user ID must be set to SYSADM.

schema_name

is the default schema to use for unqualified object names.

Notes

- This option is mutually exclusive with the Database Name option.

Default

None

GUI Tab

[General tab](#)

Concurrent Access Resolution

Attribute

ConcurrentAccessResolution (CAR)

Purpose

Specifies whether a read-only query can access the currently committed value of rows that are locked by a transaction that is updating the rows. The driver must be connected to DB2 V9.7 for LUW or higher and the application isolation level must be either read committed or repeatable read.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Automatic), the driver persists the server behavior, as specified by the cur_commit server parameter. If cur_commit is set to "Available" or "Disable," then the current behavior, pending until the row lock is released, is used. When cur_commit is set to "On," the driver returns the last committed value of the row, regardless of whether the row is locked.

If set to 1 (Wait For Outcome), the driver always waits for the transaction to be completed before returning a row of data that has been locked by another transaction, regardless of how the `cur_commit` parameter is configured on the server.

If set to 2 (Use Currently Committed), the driver returns the value that was committed during the last transaction if the `cur_commit` parameter is configured to "On" or "Available," even though the row is locked.

Notes

- This option is ignored when connecting to a DB2 server version earlier than DB2 V9.7 for LUW.
- DB2 V10 for z/OS and DB2 for i 7.1 using DRDA do not currently support reading committed data while performing UPDATE statements. In this scenario, transactions always wait for a commit or rollback operation if they encounter data that is being updated or deleted.

Default

0 (Automatic)

GUI Tab

[Advanced tab](#)

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- The application must be thread-enabled to use connection pooling.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 212

Connection Reset

Attribute

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 212

Connection Retry Count

Attribute

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x , the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay (CRD)

Purpose

The number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to x , the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (EncryptionMethod=1). When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, the connection fails and the driver returns an error.

Valid Values

```
cryptographic_protocol [ [ , cryptographic_protocol ] ... ]
```

where:

```
cryptographic_protocol
```

is one of the following cryptographic protocols:

```
TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2
```

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Notes

- This option is ignored if Encryption Method is set to 0 (No Encryption) or 2 (Database Encryption).
- Consult your database administrator concerning the data encryption settings of your server.

Default

```
TLSv1.2, TLSv1.1, TLSv1
```

GUI Tab

[Security tab](#)

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\
Drivers\OpenSSL\1.0.0r\ddssl27.dll; (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the CryptoLibName and SSLibName options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

For UNIX/Linux:

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\
Connect64_for_ODBC_71\drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLibName](#) on page 204

Current Function Path

Attribute

CurrentFunctionPath (CFP)

Purpose

Specifies a comma-separated list of DB2 schema names used to resolve unqualified function names and data type references in dynamically prepared SQL statements. This value also is used to resolve unqualified stored procedure names specified in CALL statements.

Valid Values

```
schema_name[ , ... ]
```

where:

```
schema_name
```

is the name of a valid DB2 schema.

Default

None

GUI Tab

[Advanced tab](#)

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

The name of a data source in your Windows Registry or odbc.ini file.

Valid Values

```
string
```

where:

```
string
```

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database Name

Attribute

Database (DB)

Purpose

The name of the database to which you want to connect.

Valid only for DB2 for Linux/UNIX/Windows.

This option is mutually exclusive with the Location Name and Collection options.

Valid Values

ext

where:

ext

is the name of the one- to three-character file name extension.

This value is used for all Create Table statements. Sending a Create Table using an extension other than the value specified for this option causes an error.

In other SQL statements, such as Select or Insert, users can specify an extension other than the one specified for this connection option. The Data File Extension value is used when no extension is specified.

Default

None

GUI Tab

[General tab](#)

Default Isolation Level

Attribute

DefaultIsolationLevel (DIL)

Purpose

Specifies the method by which locks on data in the database are acquired and released.

The following table shows how ODBC isolation levels map to DB2 isolation levels.

ODBC	DB2
Read Uncommitted	Uncommitted Read
Read Committed	Cursor Stability

ODBC	DB2
Repeatable Read	Read Stability
Serializable	Repeatable Read

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Valid Values

0 | 1 | 2 | 3 | 4

Behavior

If set to 0 (READ_UNCOMMITTED), other processes can be read from the database. Only modified data is locked and is not released until the transaction ends.

If set to 1 (READ_COMMITTED) other processes can change a row that your application has read if the cursor is not on the row you want to change. This level prevents other processes from changing records that your application has changed until your application commits them or ends the transaction.

It also prevents your application from reading a modified record that has not been committed by another process, unless the Concurrent Access Resolution connection option is set to:

- Automatic (0) and the cur_commit server parameter is set to On
- Use Currently Committed (2) and the cur_commit server parameter is set to On or Available

In either of these cases, the application can read the last committed value. See the connection option [Concurrent Access Resolution](#) on page 174 for further details.

See [Cursor Stability Isolation Level](#) on page 216 for information about enhancements to the Read Committed (Cursor Stability) isolation level.

If set to 2 (REPEATABLE_READ), other processes are prevented from accessing data that your application has read or modified. All read or modified data is locked until transaction ends.

If set to 3 (SERIALIZABLE), other processes are prevented from changing records that are read or changed by your application (including phantom records) until your program commits them or ends the transaction. This level prevents the application from reading modified records that have not been committed by another process. If your application opens the same query during a single unit of work under this isolation level, the results table will be identical to the previous table; however, it can contain updates made by your application.

If set to 4 (NONE), your application can read modified records even if they have not been committed by another application. This level can only be set in the data source, not from the application. (This level is valid only on DB2 for i, and is the only isolation level that works for collections that have journaling disabled.)

Default

1 (READ_COMMITTED)

GUI Tab

[Advanced tab](#)

Description

Attribute

Description (n/a)

Purpose

An optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Dynamic Sections

Attribute

DynamicSections (DS)

Purpose

Specifies the maximum number of prepared statements that the driver can have open at any time.

A dynamic section is associated with a prepared statement. The driver only keeps open the number of prepared statements specified by the dynamic sections value. If the driver detects that the number of dynamic sections available in the bound DB2 packages is less than the number of dynamic sections requested in the connection string or data source, it generates the following message:

```
The current number of dynamic sections available for use is different than the number
of dynamic sections currently specified in the connection string or data source.
```

Valid Values

x

where:

x

Is a positive integer that represents a number of prepared statements.

Default

1000

GUI Tab

[Modify Bindings tab](#)

Encryption Method

Attribute

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server. If the specified encryption method is not supported by the database server, the connection fails and the driver returns an error.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using the SSL protocols specified in the Crypto Protocol Version connection option.

If set to 2 (Database Encryption), data is encrypted using the DB2 encryption protocol (supported only on DB2 for Linux/UNIX/Windows and DB2 for z/OS).

This option can only be set to 1 or 2 when Authentication Method is set to 0, 1, or 2.

Notes

- This connection option can affect performance.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

See also

[Crypto Protocol Version](#) on page 178

[Performance Considerations](#) on page 212

Failover Granularity

Attribute

FailoverGranularity (FG)

Purpose

Determines whether the driver fails the entire failover process or continues with the process if errors occur while trying to reestablish a lost connection.

This option applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select).

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 (Non-Atomic), the driver continues with the failover process and posts any errors on the statement on which they occur.

If set to 1 (Atomic) the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

If set to 2 (Atomic Including Repositioning), the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

If set to 3 (Disable Integrity Check), the driver does not verify that the rows that were restored during the failover process match the original rows. This value applies only when Failover Mode is set to 2 (Select).

Default

0 (Non-Atomic)

GUI Tab

[Failover tab](#)

Failover Mode

Attribute

FailoverMode (FM)

Purpose

Specifies the type of failover method the driver uses.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Connection), the driver provides failover protection for new connections only.

If set to 1 (Extended Connection), the driver provides failover protection for new and lost connections, but not any work in progress.

If set to 2 (Select), the driver provides failover protection for new and lost connections. In addition, it preserves the state of work performed by the last Select statement executed.

Notes

- This connection option can affect performance.

Default

0 (Connection)

GUI Tab

[Failover tab](#)

See also

[Performance Considerations](#) on page 212

Failover Preconnect

Attribute

FailoverPreconnect (FP)

Purpose

Specifies whether the driver tries to connect to the primary and an alternate server at the same time.

This attribute applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select) and at least one alternate server is specified.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver tries to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection. This value provides the best performance, but your application typically experiences a short wait while the failover connection is attempted.

If set to 1 (Enabled), the driver tries to connect to the primary and an alternate server at the same time. This can be useful if your application is time-sensitive and cannot absorb the wait for the failover connection to succeed.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Fetch Time Stamp With Time Zone as Timestamp

Attribute

FetchTSWTZasTimestamp

Purpose

Determines whether the driver returns Timestamp with Time Zone columns as an ODBC SQL_TYPE_TIMESTAMP or as a SQL_VARCHAR data type.

Valid only for DB2 for z/OS, version 10 or higher.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), Timestamp with Time Zone columns are mapped to SQL_VARCHAR. Use this setting if your application needs to retrieve the information as a string.

If set to 1 (Enabled), the driver maps Timestamp with Time Zone columns to the ODBC SQL_TYPE_TIMESTAMP data type. The time zone information is truncated from the results.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Field Delimiter

Attribute

BulkLoadFieldDelimiter (BLFD)

Purpose

Specifies the character that the driver will use to delimit the field entries in a bulk load data file.

Valid Values

x

where:

x

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash(-), the colon(:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Field Delimiter character must be different from the Bulk Load Record Delimiter.

Default

None

GUI Tab

[Bulk tab](#)

Grant Execute to [check box]

Attribute

GrantExecute (GE)

Purpose

Determines how EXECUTE privileges are granted on DB2 packages.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), EXECUTE privileges are granted on DB2 packages that you are creating. By default, the schema to which privileges are granted is PUBLIC.

If set to 0 (Disabled), EXECUTE privileges are granted to the schema that created the DB2 packages.

Default

1 (Enabled)

GUI Tab

[Modify Bindings tab](#)

Grant Execute to [field]

Attribute

GrantAuthid (GA)

Purpose

Determines which DB2 schema is granted EXECUTE privileges for DB2 packages.

Valid Values

schema_name

where:

schema_name

is the name of a valid DB2 schema.

Default

PUBLIC

GUI Tab

[Modify Bindings tab](#)

GSS Client Library

Attribute

GSSClient (GSSC)

Purpose

The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC). The driver uses the path defined by the PATH environment variable for loading the specified client library.

Valid Values

native | *client_library*

where:

client_library

is a GSS client library installed on the client.

Behavior

If set to *client_library*, the driver uses the specified GSS client library.

If set to native, the driver uses the GSS client shipped with the operating system.

Default

native

GUI Tab

[Security tab](#)

Host Name In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

`host_name` | `#SERVERNAME#`

where:

`host_name`

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If `host_name` is specified, the driver examines the `subjectAltName` values included in the certificate. If a `dnsName` value is present in the `subjectAltName` values, then the driver compares the value specified for Host Name In Certificate with the `dnsName` value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the `dnsName` value.

If no `subjectAltName` values exist or a `dnsName` value is not in the list of `subjectAltName` values, then the driver compares the value specified for Host Name In Certificate with the `commonName` part of the Subject name in the certificate. The `commonName` typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the `commonName`. If multiple `commonName` parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the `commonName` parts.

If `#SERVERNAME#` is specified, the driver compares the host server name specified as part of a data source or connection string to the `dnsName` or the `commonName` value.

Default

None

GUI Tab

[Security tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)

- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Ip Address

Attribute

IpAddress (IP)

Purpose

Identifies the machine where catalog tables are stored.

Valid Values

host_name | *IP_address*

where:

host_name

is the host name of the machine where catalog tables are stored. The driver must be able to find this name (with the correct address assignment) in the HOSTS file on the workstation or in a DNS server.

IP_address

is the IP address of the machine where catalog tables are stored. The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details about these formats.

Default

localhost

GUI Tab

[General tab](#)

Key Password

Attribute

KeyPassword (KP)

Purpose

Specifies the password used to access the individual keys in the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. Keys stored in a keystore can be individually password-protected. To extract the key from the keystore, the driver must have the password of the key.

Valid Values

key_password

where:

key_password

is the password of a key in the keystore.

Default

None

GUI Tab

[Security tab](#)

Keystore

Attribute

Keystore (KS)

Purpose

The name of the directory containing the keystore file to be used when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request. If you do not specify a directory, the current directory is used.

Valid Values

keystore_directory

where:

keystore_directory

is the location of the keystore file.

Notes

- The keystore and truststore files can be the same file.

Default

None

GUI Tab

[Security tab](#)

Keystore Password

Attribute

KeystorePassword (KSP)

Purpose

The password used to access the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request.

Valid Values

keystore_password

where:

keystore_password

is the password of the keystore file.

Notes

- The keystore and truststore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Load Balance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

Specifies the number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Valid Values

0 | x

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to x, inactive connections are closed after the specified number of seconds passes.

Notes

- The Min Pool Size option may cause some connections to ignore this value.

Default

0

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 212

Load Balancing

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Location Name**Attribute**

Location (LOC)

Purpose

Specifies the name of the DB2 location that you want to access.

For DB2 for z/OS, your system administrator can determine the name of your DB2 location using the following command:

```
DISPLAY DDF
```

For DB2 for i, your system administrator can determine the name of your DB2 location using the following command. The name of the database that is listed as *LOCAL" is the value that you should use for this attribute.

```
WRKRDBDIRE
```

This option is mutually exclusive with the Database Name option.

Valid Values

location_name

where:

location_name

is the name of a valid DB2 location.

Notes

- Valid only for DB2 for z/OS and i.

Default

None

GUI Tab

[General tab](#)

Login Timeout**Attribute**

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x, the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

15

GUI Tab

[Advanced tab](#)

Min Long Varchar Size

Attribute

MinLongVarcharSize (MINLVS)

Purpose

Specifies the minimum count of characters the driver reports for columns mapped as SQL_LONGVARCHAR. If the size of a SQL_LONGVARCHAR column is less than the value specified, the driver will increase the reported size of the column to this value when calling SQLDescribeCol and SQLColumns. This allows you to fetch SQL_LONGVARCHAR columns whose size is smaller than the minimum imposed by some third-party applications.

Valid Values

x

where:

x

is the minimum size in characters the driver will report for columns mapped to the SQL_LONGVARCHAR type.

Notes

- Configuring the VarcharThreshold and MinLongVarcharSize options allows you to fetch SQL_VARCHAR and SQL_LONGVARCHAR columns with sizes that fall between the data-type ranges used by some applications.

Default

None. If no value is specified, the driver will not change the column size reported for SQL_LONGVARCHAR columns.

GUI Tab

[Advanced tab](#)

See also

[Varchar Threshold](#) on page 209

Max Pool Size

Attribute

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

Notes

- This connection option can affect performance.

Example

If set to 20, the maximum number of connections allowed in the pool is 20.

Default

100

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 212

Min Pool Size

Attribute

MinPoolSize (MNPS)

Purpose

The minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Valid Values

0 | x

where:

x

is an integer from 1 to 65535.

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

Example

If set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Default

0

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 212

Package Collection

Attribute

PackageCollection (PC)

Purpose

Specifies the name of the DB2 collection or location where the driver creates bind packages and, when required, searches for them.

Valid Values

`collection_name`

where:

collection_name

is a valid DB2 collection or location name.

Default

NULLID

GUI Tab

[Modify Bindings tab](#)

Package Name Prefix

Attribute

PackageNamePrefix (PNP)

Purpose

Specifies a two-character prefix used for package names when the driver executes dynamic SQL. The default package name uses the following syntax:

`DDiVRMx`

where:

DD

is the two-character prefix.

i

is one of the following characters:

- S—Serializable : DB2 RR
- R—Repeatable Read : DB2 RS
- C—Committed Read : DB2 CS
- U—Uncommitted Read : DB2 UC
- N—Not committed: DB2 NC

VRM

is the Version Release Modification, for example, you can specify 520 to represent version 5.2.0.

x

is a one-character suffix that specifies:

- A—Cursor queries/updates
- B—Cursor queries/updates with hold
- C—Stored procedures (section 1 is for stored procedures that do not have parameters; section 2 is for procedures that do have parameters)

For example, the package name DDOC520A would represent a package using the Committed Read isolation level, at version 5.20, and using cursor queries/updates.

Valid Values

xx

where:

xx

is a two-character prefix.

Default

DD

GUI Tab

[Advanced tab](#)

Package Owner

Attribute

PackageOwner (PO)

Purpose

Specifies the AuthID assigned to the package.

Valid Values

authid

where:

authid

is a valid DB2 AuthID that has permissions to execute all the SQL in the package.

Default

None

GUI Tab

[Modify Bindings Tab](#)

Password

Attribute

Password (PWD)

Purpose

Specifies the password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Program ID

Attribute

ProgramID (PID)

Purpose

Specifies the product and version information of the driver on the client to be stored in the database. This value sets the CLIENT_PRDID value in the database. For DB2 V9.1 and higher for Linux/UNIX/Windows, this value is located in the SYSIBMADM.APPLICATIONS table. This value is used by the DB2 Workload Manager.

Valid Values

DDTVVRRM

where:

DDT

identifies a DataDirect Connect driver.

VV

identifies a 2-digit version number (with high-order 0 in the case of a 1-digit version).

RR

identifies a 2-digit release number (with high-order 0 in the case of a 1-digit release).

M

identifies a 1-character modification level (0-9 or A-Z).

Notes

- This connection option can affect performance.

Example

DDT06010

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 212

Query Timeout

Attribute

QueryTimeout (QT)

Purpose

Specifies the number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the SQL_ATTR_QUERY_TIMEOUT statement attribute on the SQLSetStmtAttr() function.

Query timeout is supported on DB2 for Linux/UNIX/Windows 8.1 and higher and on DB2 for z/OS 8.1 and higher

Valid Values

-1 | 0 | x

where:

x

is a number of seconds.

Behavior

If set to -1, the query does not time out. The driver silently ignores the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to 0, the query does not time out, but the driver responds to the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to x, all queries time out after the specified number of seconds unless the application overrides this value by setting the SQL_ATTR_QUERY_TIMEOUT attribute.

Default

0

GUI Tab

[Advanced tab](#)

Record Delimiter

Attribute

BulkLoadRecordDelimiter (BLRD)

Purpose

Specifies the character that the driver will use to delimit the record entries in a bulk load data file.

Valid Values

x

where:

x

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash(-), the colon(:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Record Delimiter character must be different from the Bulk Load Field Delimiter.

Default

None

GUI Tab

[Bulk tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where x is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Advanced tab](#)

SSLibName

Attribute

SSLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\
```

```
Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
SSLLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 178

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Tcp Port

Attribute

TcpPort (PORT)

Purpose

Specifies the port number that is assigned to the DB2 DRDA listener process on the server host machine.

On DB2 for i only, execute `NETSTAT` from a command line to determine the correct port number. Select option 3 to display a list of active ports on the DB2 for i machine. Find the entry for DRDA, and press F14 to toggle and display the port number. If DRDA is not currently listening, the command `CHGDDMTCPA AUTOSTART(*YES) PWDRQD(*YES)` starts the listener and ensures that it is active at IPL.

Valid Values

`IP_address | service_name`

where:

IP_address

is the port's IP address.

service_name

is the port's service name. The driver must be able to find this name (with the correct port assignment) in the SERVICES file on the workstation.

Default

50000

GUI Tab

[General tab](#)

Truststore

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used.

Valid Values

`truststore_directory\filename`

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Notes

- The truststore and keystore files may be the same file.

Default

None

GUI Tab

[Security tab](#)

Truststore Password

Attribute

TruststorePassword (TSP)

Purpose

Specifies the password that is used to access the truststore file when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

- The truststore and keystore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Use Current Schema for Catalog Functions

Attribute

UseCurrentSchema (UCS)

Purpose

Specifies whether results are restricted to the tables and views in the current schema if a catalog function call is made without specifying a schema or if the schema is specified as the wildcard character %. Restricting results to the tables and views in the current schema improves performance of catalog calls that do not specify a schema.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), results of catalog function calls are restricted to the tables and views in the current schema.

If set to 0 (Disabled), results of catalog function calls are not restricted.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Security tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Notes

- Truststore information is specified using the TrustStore and TrustStorePassword options.

Default

1 (Enabled)

GUI Tab

[Security tab](#)

Varchar Threshold

Attribute

VarcharThreshold (VT)

Purpose

Specifies the threshold at which the driver describes columns of the data type SQL_VARCHAR as SQL_LONGVARCHAR. If the size of the SQL_VARCHAR column exceeds the value specified, the driver will describe the column as SQL_LONGVARCHAR when calling SQLDescribeCol and SQLColumns. This option

allows you to fetch columns that would otherwise exceed the upper limit of the SQL_VARCHAR type for some third-party applications.

Valid Values

x

where:

x

is the maximum size in characters of columns the driver will describe as SQL_VARCHAR.

Notes

- Configuring the VarcharThreshold and MinLongVarcharSize options allows you to fetch SQL_VARCHAR and SQL_LONGVARCHAR columns with sizes that fall between the data-type ranges used by some applications.

Default

None. If no value is specified, the driver will not change the described type for SQL_VARCHAR columns.

GUI Tab

[Advanced tab](#)

See also

[Min Long Varchar Size](#) on page 196

With Hold Cursors

Attribute

WithHold (WH)

Purpose

Determines whether the cursor stays open on a commit.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), cursor behavior is Preserve, which keeps cursors open after a commit or rollback (SQLGetInfo() returns SQL_CB_PRESERVE for SQL_COMMIT_CURSOR_BEHAVIOR).

If set to 0 (Disabled), cursor behavior is Delete, which closes all cursors open after a commit or rollback (SQLGetInfo() returns SQL_CB_DELETE).

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

XML Describe Type

Attribute

XMLDescribeType (XDT)

Purpose

The SQL data type that is returned by SQLGetTypeInfo for the XML data type.

See [Using the XML Data Type](#) on page 214 for further information about the XML data type.

Valid Values

-4 | -10

Behavior

If set to -4 (SQL_LONGVARBINARY), the driver uses the description SQL_LONGVARBINARY for columns that are defined as the XML data type.

If set to -10 (SQL_WLONGVARCHAR), the driver uses the description SQL_WLONGVARCHAR for columns that are defined as the XML data type.

Default

-10

GUI Tab

[Advanced tab](#)

Enable Bulk Load

Attribute

EnableBulkLoad (EBL)

Purpose

Specifies the bulk load method.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses the database bulk load protocol when an application executes an INSERT with multiple rows of parameter data. If the protocol cannot be used, the driver returns a warning.

If set to 0 (Disabled), the driver uses standard parameter arrays.

Default

0 (Disabled)

GUI Tab[Bulk tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Connection Pooling (Pooling): If you enable the driver to use connection pooling, you can set additional options that affect performance:

- **Load Balance Timeout (LoadBalanceTimeout):** You can define how long to keep connections in the pool. The time that a connection was last used is compared to the current time and, if the timespan exceeds the value of the Load Balance Timeout option, the connection is destroyed. The Min Pool Size option can cause some connections to ignore this value.
- **Connection Reset (ConnectionReset):** Resetting a re-used connection to the initial configuration settings impacts performance negatively because the connection must issue additional commands to the server.
- **Max Pool Size (MaxPoolSize):** Setting the maximum number of connections that the pool can contain too low might cause delays while waiting for a connection to become available. Setting the number too high wastes resources.
- **Min Pool Size (MinPoolSize):** A connection pool is created when the first connection with a unique connection string connects to the database. The pool is populated with connections up to the minimum pool size, if one has been specified. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Encryption Method (EncryptionMethod): Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) that is required to encrypt and decrypt data.

Failover Mode (FailoverMode): Although high availability that replays queries after a failure provides increased levels of protection, it can adversely affect performance because of increased overhead.

Use Current Schema for Catalog Functions (UseCurrentSchema): If your application needs to access database objects owned only by the current user, then performance can be improved. In this case, the Use Current Schema for Catalog Functions option must be enabled. When this option is enabled, the driver returns only database objects owned by the current user when executing catalog functions. Calls to catalog functions are optimized by grouping queries. Enabling this option is equivalent to passing the Logon ID used on the connection as the SchemaName argument to the catalog functions.

Workload Manager: The Workload Manager (WLM) automatically adjusts server resources, such as CPU and memory, based on the service class associated with a DB2 workload. Therefore, an application's performance is tied to the DB2 workload to which it is assigned and, ultimately, to the service class associated with that workload. The DB2 Wire Protocol driver allows your application to set client information in the DB2 database that can be used by the WLM to classify work. If you know that your database environment uses WLM, coordinate with your database administrator to determine how setting the following options affects performance.

- **Accounting Info (AccountingInfo):** Sets the CURRENT CLIENT_ACCTNG register (DB2 for Linux/UNIX/Windows) or the CLIENT ACCTNG register (DB2 for z/OS) on the server.
- **Application Name (ApplicationName):** Sets the CURRENT CLIENT_APPLNAME register (DB2 for Linux/UNIX/Windows) or CLIENT APPLNAME register (DB2 for z/OS) on the server.
- **Client Host Name (ClientHostName):** Sets the CURRENT CLIENT_WRKSTNNAME register (DB2 for Linux/UNIX/Windows) or CLIENT WRKSTNNAME register (DB2 for z/OS) on the server.
- **Client User (ClientUser):** Sets the CURRENT CLIENT_USERID register (DB2 for Linux/UNIX/Windows) and CLIENT USERID register (DB2 for z/OS) on the server.
- **Program ID (ProgramID):** Sets the CLIENT_PRDID value on the server.

IBM to IANA Code Page Values

Refer to "IBM to IANA code page values" in the *Progress DataDirect for ODBC Drivers Reference* for a list of the most commonly used IBM code pages and their IANA code page equivalents.

The IANA values are valid for the CharSetFor65535 connection string attribute and the Character Set for the CCSID 65535 option.

Data Types

The following table shows how the DB2 data types map to the standard ODBC data types. [Unicode Support](#) on page 215 lists DB2 to Unicode data type mappings.

Table 18: DB2 Data Types

DB2	ODBC
Bigint ⁵	SQL_BIGINT
Blob ⁶	SQL_LONGVARBINARY
Char	SQL_CHAR
Char() for Bit Data	SQL_BINARY
Clob ⁷	SQL_LONGVARCHAR
Date	SQL_TYPE_DATE

⁵ Supported on DB2 V8.x and higher for Linux/UNIX/Windows, DB2 V9 and higher for DB2 for z/OS, and DB2 V5R3 and higher for DB2 for i.

⁶ Supported on DB2 V8.x and higher for Linux/UNIX/Windows; DB2 for z/OS; and DB2 V5R3 and higher for DB2 for i.

⁷ On DB2 for Linux/UNIX/Windows versions previous to V8.1 and DB2 V5R2 for DB2 for i, only the first 32 KB of the Clob data are returned when fetching, and only 32 KB can be inserted and updated.

DB2	ODBC
Decfloat ⁸	SQL_DOUBLE
Decimal	SQL_DECIMAL
Double	SQL_DOUBLE
Float	SQL_DOUBLE
Integer	SQL_INTEGER
Long Varchar	SQL_LONGVARCHAR
Long Varchar for Bit Data	SQL_LONGVARBINARY
Numeric	SQL_NUMERIC
Real	SQL_REAL
Smallint	SQL_SMALLINT
Time	SQL_TYPE_TIME
Timestamp ⁹	SQL_TYPE_TIMESTAMP
Timestamp With Time Zone ^{6, 10}	SQL_VARCHAR
Varchar	SQL_VARCHAR
Varchar() for Bit Data	SQL_VARBINARY
XML ¹¹	SQL_LONGVARCHAR

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

Using the XML Data Type

By default, DB2 returns XML data to the driver encoded as UTF-8. To avoid data loss, an application must bind XML data as SQL_C_WCHAR. The driver then returns the data as either UTF-8 or UTF-16, depending on platform and application settings. If the application binds XML data as SQL_C_CHAR, the driver converts it to the client character encoding, possibly causing data loss or corruption. To prevent any conversion of XML data, the application must set the attribute [XMLDescribeType](#) (XDT) to SQL_LONGVARBINARY (-10) and bind the data as SQL_C_BINARY.

⁸ Supported only on DB2 V9 and higher for Linux/UNIX/Windows, DB2 V9 and higher for DB2 for z/OS, and DB2 i V6R1 for DB2 for i.

⁹ Timestamp values with a fractional seconds precision greater than 9 are described as the ODBC SQL_VARCHAR data type.

¹⁰ Timestamp with Time Zone mapping changes based on the setting of the FetchTSWTZasTimestamp option only on DB2 V10 and higher for DB2 for z/OS.

¹¹ Supported only on DB2 V10 for z/OS.

Unicode Support

The DB2 Wire Protocol driver supports Unicode data types if the database was created with a multi-byte character set.

The DB2 Wire Protocol driver maps the DB2 data types to Unicode data types as shown in the following table:

DB2 Data Type	Mapped to . . .
Dbclob ¹²	SQL_WLONGVARCHAR
Graphic	SQL_WCHAR
Long Vargraphic	SQL_WLONGVARCHAR
Vargraphic	SQL_WVARCHAR

Advanced Features

The driver supports the following advanced features:

- Failover
- Client Information
- Security
- Connection Pooling
- DataDirect Bulk Load

Failover

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Security

The driver supports authentication and encryption. Security connection options are located on the [Security tab](#) of the driver Setup dialog box. See [Using Security](#) on page 89 for a general description of security and its implementation. The following security information is specific to the DB2 Wire Protocol Driver.

Authentication

If you are using Kerberos, verify that your environment meets the requirements listed in the following table before you configure the driver for Kerberos authentication.

¹² Supported on DB2 V8.x and higher for Linux/UNIX/Windows, DB2 V9 and higher for DB2 for z/OS, and DB2 V5R3 and higher for DB2 for i.

Table 19: Kerberos Authentication Requirements for the DB2 Wire Protocol Driver

Component	Requirements
Database server	The database server must be running one of the following database versions: <ul style="list-style-type: none"> • DB2 V8.1 or higher for Linux/UNIX/Windows • DB2 V8.x or higher for z/OS
Kerberos server	The Kerberos server is the machine where the user IDs for authentication are administered. The Kerberos server is also the location of the Kerberos KDC. Network authentication must be provided by one of the following methods: <ul style="list-style-type: none"> • Windows Active Directory • MIT Kerberos 1.4.2 or higher

Connection Pooling

The driver supports connection pooling and its related connection options. Connection pooling connection options are located on the [Pooling tab](#) of the driver Setup dialog box. See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling and its implementation.

DataDirect Bulk Load

The driver supports DataDirect bulk load and its related connection options. Bulk load connection options are located on the [Bulk tab](#) of the driver Setup dialog box. See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect bulk load and its implementation.

Cursor Stability Isolation Level

The DB2 Cursor Stability (CS) isolation level has been enhanced in DB2 V9.7 to reduce significantly instances of lock wait and deadlock. In previous DB2 versions, CS prevented any row that was changed by other applications from being read until the change was committed.

In this enhanced implementation, CS, where possible, avoids a read operation waiting for a row to commit before returning a value. CS now returns the currently committed result, ignoring what might happen to an uncommitted operation. Some exceptions, such as updatable cursors, exist; currently committed results cannot be returned immediately when the row might be updated based upon its previous contents. CS behavior is determined through the [Concurrent Access Resolution \(CAR\)](#) connection option.

Consider the following example, in which deadlocks are avoided under the currently committed semantics. In this scenario, two applications update two separate tables, but do not yet commit. Each application then attempts to read (with a read-only cursor) from the table that the other application has updated.

Table 20: Cursor Stability Examples

Step	Application A	Application B
1	UPDATE T1 SET col1 = ? WHERE col2 = ?	UPDATE T2 SET col1 = ? WHERE col2 = ?

Step	Application A	Application B
2	SELECT col1, col3, col4 FROM T2 WHERE col2 >= ?	SELECT col1, col5, FROM T1 WHERE col5 = ? AND col2 = ?
3	Commit	Commit

Without currently committed semantics, these applications running under the cursor stability isolation level might create a deadlock, causing one of the applications to fail. This happens when each application needs to read data that is being updated by the other application.

Under currently committed semantics, if the query in step 2 (for either application) happens to require the data currently being updated by the other application, that application does not wait for the lock to be released, making a deadlock impossible. The previously committed version of the data is located and used instead.

XQuery Expressions

The DB2 Wire Protocol driver supports execution of XQuery expressions in DB2 V9.1 and higher for Linux/UNIX/Windows. IBM provides a tutorial on this topic at the following URL:

http://www-01.ibm.com/support/knowledgecenter/SSEPGG_9.1.0/com.ibm.db2.xquery.doc/doc/xqbtutorial.htm

Stored Procedure Support

The DB2 Wire Protocol driver supports DB2 Remote Procedure Calls (RPCs) as follows:

- Multiple result sets are returned.
- RPCs must take an argument list. The driver does not support RPCs that use a SQL descriptor area (SQLDA) data structure to specify the arguments.
- Literals are supported as stored procedure parameters.

Unexpected Characters

Users are sometimes surprised when they insert a character into a database, only to have a different character displayed when they fetch it from the database. There are many reasons this can happen, but it most often involves code page issues, not driver errors.

Client and server machines in a database system each use code pages, which can be identified by a name or a number, such as Shift_JIS (Japanese) or cp1252 (Windows English). A code page is a mapping that associates a sequence of bits, called a code point, with a specific character. Code pages include the characters and symbols of one or more languages. Regardless of geographical location, a machine can be configured to use a specific code page. Most of the time, a client and database server would use similar, if not identical, code pages. For example, a client and server might use two different Japanese code pages, such as Shift_JIS and EUC_JP, but they would still share many Japanese characters in common. These characters might, however, be represented by different code points in each code page. This introduces the need to convert between code pages to maintain data integrity. In some cases, no one-to-one character correspondence exists between the two code points. This causes a substitution character to be used, which can result in displaying an unexpected character on a fetch.

When the driver on the client machine opens a connection with the database server, the driver determines the code pages being used on the client and the server. This is determined from the Active Code Page on a Windows-based machine. If the client machine is UNIX-based, the driver checks the IANAAppCodePage attribute (see [IANAAppCodePage](#) on page 190). If it does not find a specific setting for IACP, it defaults to a value of ISO_8859_1 Latin_1.

If the client and server code pages are compatible, the driver transmits data in the code page of the client. Even though the pages are compatible, a one-to-one correspondence for every character may not exist. If the client and server code pages are completely dissimilar, for example, Russian and Japanese, then many substitutions occur because very few, if any, of the characters are mapped between the two code pages.

The following is a specific example of an unexpected character:

- The client machine is running the Japanese code page EUC_JP.
- The DB2 server is running the Japanese code page Shift_JIS.
- When you insert the EUC_JP code point 0xA1BD and then fetch it back, you do not see the character you expected. In fact, what displays on the client may not be a recognizable character.

This substitution occurs because the code points do not correspond in the two code pages. EUC_JP code point 0xA1BD is converted to UTF-16 code point 0x2014. Code point 0x2014 does not map to anything in Shift_JIS, resulting in the Shift_JIS substitution code point, 0x3F, being sent to, and stored in, the database. When this character is retrieved, depending on the client display, it may not display as a recognizable character.

This is not a driver error. It occurs because the code points map differently and because some characters do not exist in a code page. The best way to avoid these problems is to use the same code page on both the client and server machines.

Support for DB2 pureScale

IBM introduced DB2 pureScale to provide scaleout active and active services for IBM DB2 running on AIX on Power Systems servers. It is designed to deliver distributed availability and scalability in a clustered database system. DB2 pureScale allows a single physical DB2 database to be accessed by concurrent instances of DB2 running across several different cluster members.

A DB2 pureScale shared disk cluster is composed of a group of independent servers, or members, that cooperate as a single system. A cluster architecture such as this provides applications access to more computing power when needed, while allowing computing resources to be used for other applications when database resources are not as heavily required. For example, in the event of a sudden increase in network traffic, a DB2 pureScale cluster can distribute the load over many nodes, a feature referred to as *transaction level workload balancing*. DB2 pureScale features are available to you simply by connecting to a DB2 pureScale system with the DB2 driver. No additional configuration is required.

Connection failover and *client load balancing* can be used in conjunction with a DB2 pureScale shared disk cluster, but they are not specifically part of DB2 pureScale. See [Using Failover](#) on page 78 for details about how these features work in DataDirect ConnectSeries for ODBC drivers.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

DB2 supports isolation level 0 (read uncommitted), isolation level 1 (read committed), isolation level 2 (repeatable read), isolation level 3 (serializable), and, on DB2 for i only, isolation level 4 (none).

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the minimum SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

In addition, the following functions are supported:

- SQLProcedures
- SQLProcedureColumns

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The DB2 database system supports multiple connections and multiple statements per connection.

Using Arrays of Parameters

DB2 for Linux/UNIX/Windows natively supports parameter arrays, and the DB2 Wire Protocol driver, in turn, supports them when connected to these DB2 databases. When designing an application for performance, using native parameter arrays for bulk inserts or updates, for example, can improve performance.

Refer to "Designing ODBC applications for performance optimization" in the *Progress DataDirect for ODBC Drivers Reference* for details.

The DB2 Wire Protocol driver accepts a proprietary statement attribute called `SQL_ATTR_PARAM_ARRAY_ATOMIC`. It has two values: `SQL_PA_ATOMIC_YES` (1) and `SQL_PA_ATOMIC_NO` (0).

When set to `SQL_PA_ATOMIC_YES`, the default, parameter array operations are atomic, meaning that if one row in the parameter array fails, then the entire array must fail.

When set to `SQL_PA_ATOMIC_NO`, parameter array operations are not atomic, meaning that the parameter array continues to be processed, even if one of the rows fails.

The Informix Wire Protocol Driver

The DataDirect Connect *for* ODBC and DataDirect Connect64 *for* ODBC Informix Wire Protocol driver (the Informix Wire Protocol driver) each support multiple connections to the Informix Dynamic Server.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The Informix Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect product for the file name of the Informix Wire Protocol driver.

Note: The Informix Wire Protocol driver does not require any Informix client software. Progress DataDirect also provides an Informix client-based driver that can access earlier versions of Informix databases. See [The Informix Driver](#) on page 762 for details.

Driver Requirements

The driver has no client requirements.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 224 and [Connection Option Descriptions](#) on page 225 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

UNIX[®] On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions](#) on page 225 lists driver connection string attributes that must be used in the `odbc.ini` file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Informix)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX® On UNIX and Linux, data sources are stored in the `odbc.ini` file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure an Informix data source:

1. Start the ODBC Administrator:

-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- **UNIX**® On Linux, change to the `install_dir/tools` directory and, at a command prompt, enter:



```
odbcadmin
```

 where `install_dir` is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

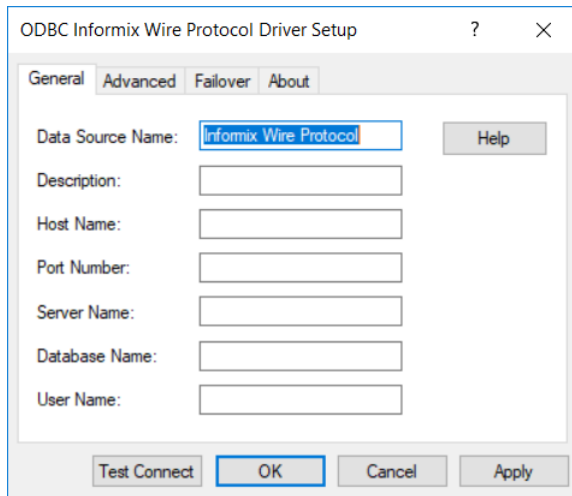
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 12: General tab



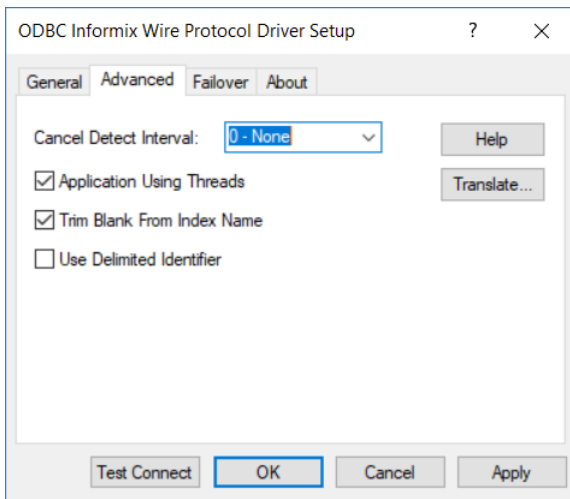
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

3. On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 230	None
Description on page 231	None
Host Name on page 231	None
Port Number on page 234	None
Server Name on page 234	None
Database Name on page 230	None
User Name on page 236	None

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 13: Advanced tab

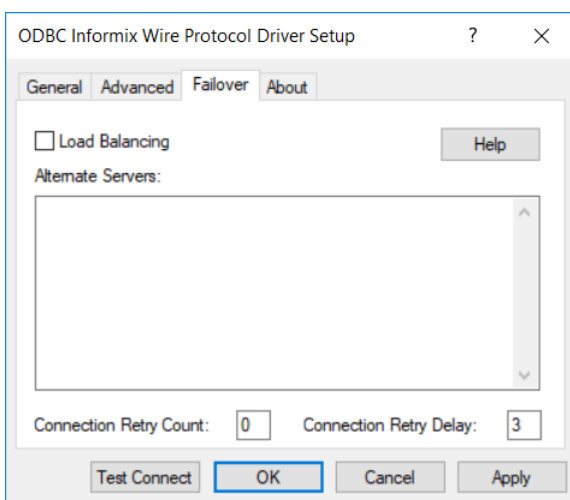


On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Cancel Detect Interval on page 228	0 - None
Application Using Threads on page 227	Enabled
Trim Blank From Index on page 235	Enabled
Use Delimited Identifier on page 235	Disabled
IANAAppCodePage on page 232 UNIX ONLY	4 (ISO 8559-1 Latin-1)

- Optionally, click the **Failover** tab to specify failover data source settings.

Figure 14: Failover tab



See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 233	Disabled
Alternate Servers on page 226	None
Connection Retry Count on page 229	0
Connection Retry Delay on page 229	3

6. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Informix\)](#) on page 225 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

7. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{ }driver_name{ }][;attribute=value[;attribute=value]...]
```


[Connection Option Descriptions](#) on page 225 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Informix is:

```
DSN=INFORMIX TABLES;DB=PAYROLL
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Informix.dsn;DB=DBPAYROLL
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Informix Wire Protocol};HOST=INF2;PORT=4321;  
SRVR=ACCT;DB=PAYROLL;UID=JOHN;PWD=XYZZY
```

Using a Logon Dialog Box (Informix)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

In this dialog box, provide the following information:

1. In the Host Name field, type the name or IP address of the host machine on which the Informix server resides.

The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details concerning these formats.

2. In the Port Number field, type the port number of the server listener.
3. In the Server Name field, type the name of the Informix server.
4. In the Database Name field, type the name of the database to which you want to connect.
5. If required, type your user name as specified on the Informix server.
6. If required, type your password.
7. Click **OK** to complete the logon and to update these values in the Registry.

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Informix Wire Protocol driver.

Table 21: Informix Wire Protocol Attribute Names

Attribute (Short Name)	Default
AlternateServers (ASRV)	None
ApplicationUsingThreads (AUT)	1 (Enabled)
CancelDetectInterval (CDI)	0 (None)
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
Database (DB)	None
DataSourceName (DSN)	None
Description (n/a)	None
HostName (HOST)	None
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
LoadBalancing (LB)	0 (Disabled)
LogonID (UID)	None
Password (PWD)	None
PortNumber (PORT)	None
ServerName (SRVR)	None
TrimBlankFromIndexName (TBFIN)	1 (Enabled)
UseDelimitedIdentifier (UDI)	0 (Disabled)

Alternate Servers

Attribute

AlternateServers (ASRV)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
(Database=dbname:HostName=hostvalue:  
PortNumber=portvalue:ServerName=servervalue[, . . .])
```

You must specify the database, host name, port number, and the server name.

Example

The following Alternate Servers value defines two alternate database servers for connection failover:

```
(Database=Infdb1:HostName=Informixhost1:PortNumber=5177:ServerName=accounting1,  
Database=Infdb2:HostName=Informixhost2:PortNumber=5178:ServerName=accounting2)
```

Notes

- An alternate server address in IPv6 format must be enclosed in double quotation marks.

Default

None

GUI Tab

[Failover tab](#)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

This connection option can affect performance. See [Performance Considerations](#) on page 236 for details.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

See also

- [Performance Considerations](#) on page 236

Cancel Detect Interval

Attribute

CancelDetectInterval (CDI)

Purpose

Determines whether long-running queries in threaded applications can be cancelled if the application issues a SQLCancel.

This connection option can affect performance. See [Performance Considerations](#) on page 236 for details.

Valid Values

0 | x

where:

x

is the number of seconds the driver waits before checking for SQLCancel calls.

Behavior

If set to 0 (None), the driver does not allow long-running queries in threaded applications to be canceled, even if the application issues a SQLCancel.

If set to x (seconds), for every pending query, the driver checks for SQLCancel calls at the specified interval. If the driver determines that a SQLCancel has been issued, the driver cancels the query.

Example

If you specify 5, for every pending query, the driver checks every five seconds to see whether the application has issued a SQLCancel call. If it detects a SQLCancel call, the driver cancels the query.

Default

0 (None)

GUI Tab

[Advanced tab](#)

See also

- [Performance Considerations](#) on page 236

Connection Retry Count

Attribute

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x , the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

See [Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay (CRD)

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to *x*, the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

See [Failover tab](#)

Database Name

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where:

database_name

is the name of a valid database.

Default

None

GUI Tab

[General tab](#)

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or `odbc.ini` file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Host Name

Attribute

HostName (HOST)

Purpose

The name or the IP address of the server to which you want to connect.

Valid Values

server_name | *IP_address*

where:

server_name

is the name of the server to which you want to connect.

IP_address

is the IP address of the server to which you want to connect.

The IP address can be specified in either IPv4 or IPv6 format. See [Using IP Addresses](#) on page 67 for details about these formats.

Default

None

GUI Tab

[General tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The Driver Manager checks for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Load Balancing**Attribute**

LoadBalancing (LB)

Description

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Password**Attribute**

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Purpose

The port number of the server listener.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your database administrator for the correct number.

Default

None

GUI Tab

[General tab](#)

Server Name

Attribute

ServerName (SRVR)

Purpose

The name of the Informix server.

Valid Values

server_name

where:

server_name

is a name that uniquely identifies the Informix server.

Default

None

GUI Tab

[General tab](#)

Trim Blank From Index

Attribute

TrimBlankFromIndexName (TBFIN)

Purpose

Determines whether the driver trims leading spaces from system-generated index names. Some applications cannot process a leading space in index names.

Valid Values

If set to 1 (Enabled), the driver trims leading spaces from system-generated index names.

If set to 0 (Disabled), the driver does not trim leading spaces from system-generated index names.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Use Delimited Identifier

Attribute

UseDelimitedIdentifier (UDI)

Purpose

Determines whether the driver sets the Informix DELIMIDENT environment variable. The DELIMIDENT environment variable specifies that strings enclosed between double quotation marks (") are delimited database identifiers.

Valid Values

0 | 1

Behavior

If set to 1 (enabled), the Informix server interprets strings enclosed in double quotation marks as identifiers, not as string literals.

If set to 0 (disabled), the Informix server interprets strings enclosed in double quotation marks as string literals, not as identifiers.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[General tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Cancel Detect Interval (CancelDetectInterval): If your application uses threads, it may allow canceling of long running queries (may issue synchronous SQLCancel calls). If your application does not issue synchronous SQLCancel calls, the driver can improve performance if the CancelDetectInterval attribute is disabled (set to 0). In this case, the driver does not incur the overhead of periodically checking for SQLCancel. In the case where your application does issue synchronous SQLCancel calls, this attribute should be set to a value that specifies how often the driver checks to see if a long running query has been canceled.

Data Types

The following table shows how the Informix data types map to the standard ODBC data types.

Table 22: Informix Data Types

Informix	ODBC
BLOB	SQL_LONGVARBINARY
BOOLEAN	SQL_BIT
BYTE	SQL_LONGVARBINARY
CHAR	SQL_CHAR
CLOB	SQL_LONGVARCHAR
DATE	SQL_TYPE_DATE
DATETIME YEAR TO FRACTION(f) ¹³	SQL_TYPE_TIMESTAMP
DATETIME YEAR TO SECOND	SQL_TYPE_TIMESTAMP
DATETIME YEAR TO DAY	SQL_TYPE_DATE
DATETIME HOUR TO SECOND	SQL_TYPE_TIME
DATETIME HOUR TO FRACTION(f) ¹³	SQL_TYPE_TIME
DECIMAL	SQL_DECIMAL
FLOAT	SQL_DOUBLE
INT8	SQL_BIGINT
INTEGER	SQL_INTEGER
INTERVAL YEAR(p) TO YEAR	SQL_INTERVAL_YEAR
INTERVAL YEAR(p) TO MONTH	SQL_INTERVAL_YEAR_TO_MONTH
INTERVAL MONTH(p) TO MONTH	SQL_INTERVAL_MONTH

¹³ (f) can have a value of 1, 2, 3, 4, or 5. The precision is type-dependent and the scale is 5.

Informix	ODBC
INTERVAL DAY(p) TO DAY	SQL_INTERVAL_DAY
INTERVAL DAY(p) TO HOUR	SQL_INTERVAL_DAY_TO_HOUR
INTERVAL DAY(p) TO MINUTE	SQL_INTERVAL_DAY_TO_MINUTE
INTERVAL DAY(p) TO SECOND	SQL_INTERVAL_DAY_TO_SECOND
INTERVAL DAY(p) TO FRACTION(f) ¹³	SQL_INTERVAL_DAY_TO_SECOND
INTERVAL HOUR(p) TO HOUR	SQL_INTERVAL_HOUR
INTERVAL HOUR(p) TO MINUTE	SQL_INTERVAL_HOUR_TO_MINUTE
INTERVAL HOUR(p) TO SECOND	SQL_INTERVAL_HOUR_TO_SECOND
INTERVAL HOUR(p) TO FRACTION(f) ¹³	SQL_INTERVAL_HOUR_TO_SECOND
INTERVAL MINUTE(p) TO MINUTE	SQL_INTERVAL_MINUTE
INTERVAL MINUTE(p) TO SECOND	SQL_INTERVAL_MINUTE_TO_SECOND
INTERVAL MINUTE(p) TO FRACTION(f) ¹³	SQL_INTERVAL_MINUTE_TO_SECOND
INTERVAL SECOND(p) TO SECOND	SQL_INTERVAL_SECOND
INTERVAL SECOND(p) TO FRACTION(f) ¹³	SQL_INTERVAL_SECOND
LVARCHAR(p) ¹⁴	SQL_VARCHAR
MONEY	SQL_DECIMAL
NCHAR	SQL_CHAR
NVARCHAR	SQL_VARCHAR
SERIAL	SQL_INTEGER
SERIAL8	SQL_BIGINT
SMALLFLOAT	SQL_REAL
SMALLINT	SQL_SMALLINT
TEXT	SQL_LONGVARCHAR
VARCHAR	SQL_VARCHAR

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

¹⁴ Supported only on Informix 9.4 and higher servers.

Advanced Features

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

MTS Support



On Windows, the driver can take advantage of Microsoft Transaction Server (MTS) capabilities, specifically, the Distributed Transaction Coordinator (DTC) using the XA Protocol. For a general discussion of MTS and DTC, refer to the help file of the Microsoft Transaction Server SDK.

Note: The DataDirect Connect *for* ODBC 32-bit drivers can operate in a 64-bit Windows environment; however, they do not support DTC in this environment. Only the DataDirect Connect64 *for* ODBC 64-bit drivers support DTC in a 64-bit Windows environment.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

Informix supports isolation levels 0 (read uncommitted), 1 (read committed), and 3 (serializable). The default is 1. Informix supports record-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the minimum SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

In addition, the following functions are supported:

- SQLColumnPrivileges
- SQLForeignKeys
- SQLTablePrivileges

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The Informix Wire Protocol driver supports multiple connections and multiple statements per connection to the Informix database system.

The MySQL Wire Protocol Driver

The DataDirect Connect *for* ODBC and DataDirect Connect64 *for* ODBC MySQL Wire Protocol driver (the MySQL Wire Protocol driver) each support multiple connections to MySQL Enterprise Edition servers and the following storage engines:

- Storage engines
 - InnoDB – Transactional
 - MyISAM – Non-Transactional
 - Memory (formerly HEAP) – Non-Transactional

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

Note: The DataDirect Connect Series *for* ODBC drivers for MySQL Enterprise were developed using the MySQL Protocol Documentation whose copyright is owned by, and licensed by DataDirect from, MySQL AB. If any of the DataDirect Connect Series *for* ODBC is licensed for the MySQL database the following shall apply: You must purchase commercially licensed MySQL database software or a MySQL Enterprise subscription in order to use the DataDirect Connect Series *for* ODBC drivers for MySQL Enterprise with MySQL software.

Note: The MySQL Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

See the readme file shipped with your DataDirect Connect product for the file name of the MySQL Wire Protocol driver.

Driver Requirements

The driver has no client requirements.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 248 and [Connection Options Descriptions](#) on page 249 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

UNIX[®] On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Options Descriptions](#) on page 249 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (MySQL)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX[®] On UNIX and Linux, data sources are stored in the odbc.ini file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a MySQL data source on Windows:

1. Start the ODBC Administrator:


-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- **UNIX**[®] On Linux, change to the `install_dir/tools` directory and, at a command prompt, enter:
`odbcadmin`

where *install_dir* is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

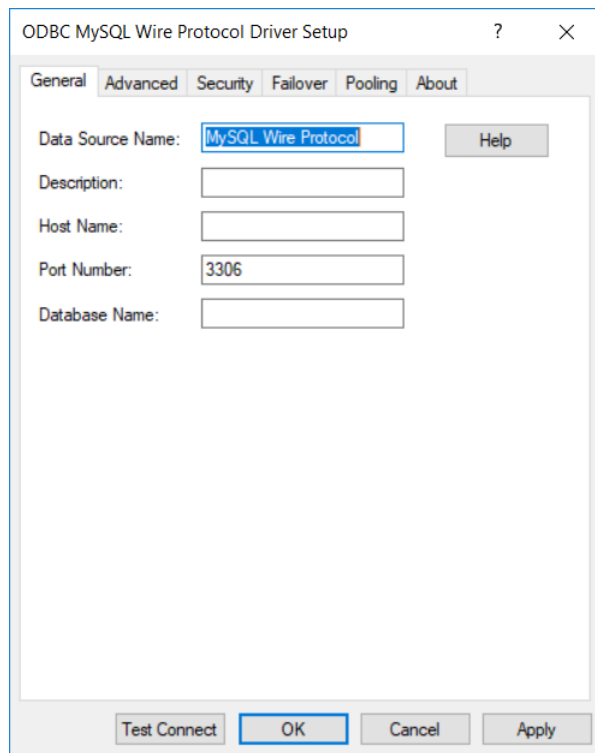
-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

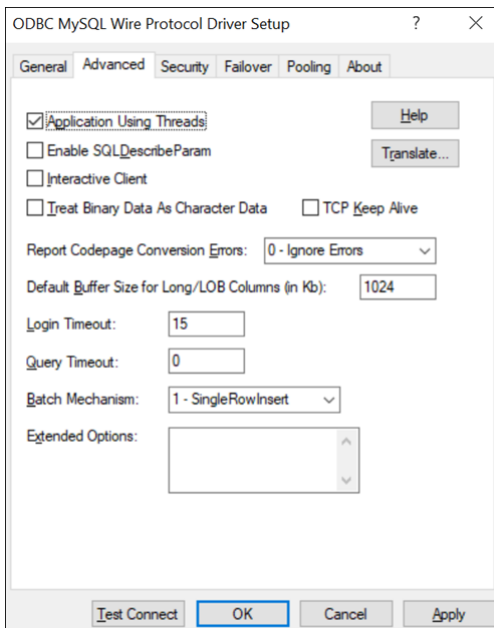


Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

3. On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source name on page 258	None
Description on page 260	None
Host Name on page 263	None
Port Number on page 272	3306
Database Name on page 259	None

4. Optionally, click the **Advanced** tab to specify data source settings.



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Application Using Threads on page 253	Enabled
Enable SQLDescribeParam on page 260	Disabled
Interactive Client on page 266	Disabled
Treat Binary Data as Character Data on page 276	Disabled
TCP Keep Alive on page 275	Disabled
Report Codepage Conversion Errors on page 273	0 - Ignore Errors
Default Buffer Size for Long/LOB Columns (in Kb) on page 259	1024

Connection Options: Advanced	Default
Login Timeout on page 269	15
Query Timeout on page 273	0
IANAAppCodePage on page 265 UNIX ONLY	4 (ISO 8559-1 Latin-1)



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.

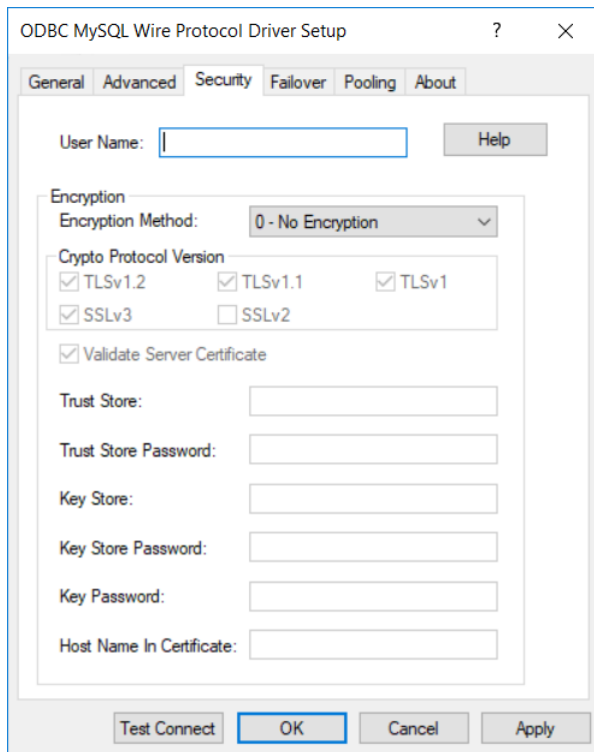


Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- Optionally, click the **Security** tab to specify security data source settings.

Figure 15: Security tab



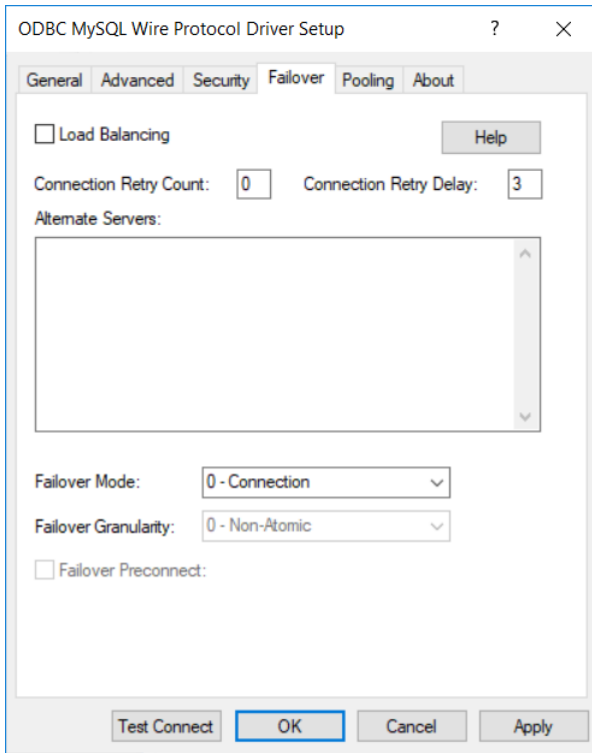
See [Using Security](#) on page 89 for a general description of authentication and encryption and their configuration requirements.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Security	Default
User Name on page 278	None
Encryption Method on page 261	0 - No Encryption
Crypto Protocol Version on page 256	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Validate Server Certificate on page 278	Enabled
Truststore on page 276	None
Truststore Password on page 277	None
Keystore on page 267	None
Keystore Password on page 267	None
Key Password on page 266	None
Host Name In Certificate on page 264	None

6. Optionally, click the **Failover** tab to specify failover data source settings.

Figure 16: Failover tab



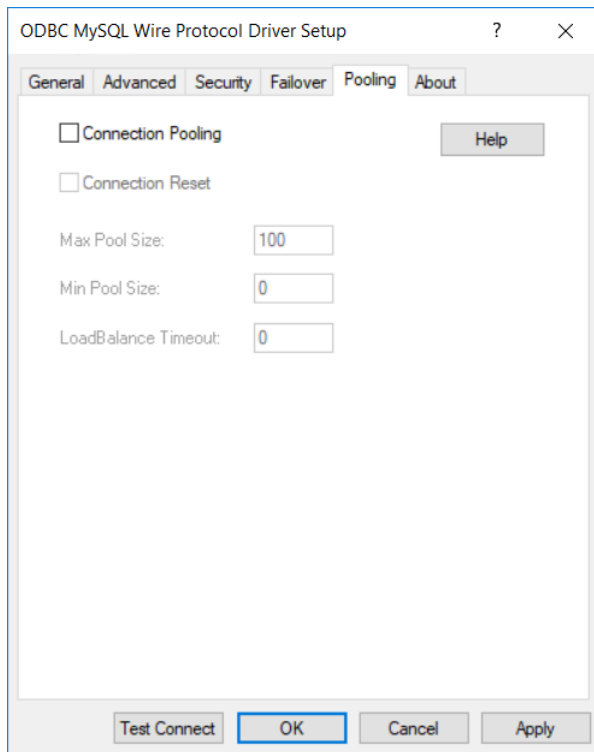
See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 269	Disabled
Connection Retry Count on page 255	0
Connection Retry Delay on page 255	3
Alternate Servers on page 252	None
Failover Mode on page 262	0 - Connection
Failover Granularity on page 261	0 - Non-Atomic
Failover Preconnect on page 263	Disabled

7. Optionally, click the **Pooling** tab to specify connection pooling data source settings.

Figure 17: Pooling tab



See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 253	Disabled
Connection Reset on page 254	Disabled
Max Pool Size on page 270	100
Min Pool Size on page 271	0
Load Balance Timeout on page 268	0

8. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection properties specified in the driver Setup dialog box. A logon dialog box appears; see [Using a Logon Dialog Box \(MySQL\)](#) on page 248 for details. Note that the information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an improper environment or incorrect connection value, it displays an appropriate error message.

Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

9. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={ }driver_name[ ] [ ;attribute=value[ ;attribute=value] ... ]
```

[Connection Options Descriptions](#) on page 249 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for MySQL is:

```
DSN=MySQL TABLES;DB=PAYROLL
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=MySQL.dsn;DB=DBPAYROLL
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 MySQL Wire Protocol};HOST=MySQL2;PORT=3306;  
DB=PAYROLL;UID=JOHN;PWD=XYZZY
```

Using a Logon Dialog Box (MySQL)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

In this dialog box, perform the following steps:

1. In the Host Name field, type either the name or the IP address of the server to which you want to connect. The IP address must be in IPv4 format.
2. In the Port Number field, type the port number of the server listener. The default is 3306.
3. In the Database Name field, type the name of the database to which you want to connect.
4. If required, type your user name as specified on the MySQL server.
5. If required, type your password.
6. Click **OK** to complete the logon and to update these values in the Registry.

Connection Options Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the MySQL Wire Protocol driver.

Table 23: MySQL Wire Protocol Attribute Names

Attribute (Short Name)	Default
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
AlternateServers (ASRV)	None
ApplicationUsingThreads (AUT)	1 (Enabled)
ConnectionReset (CR)	0 (Disabled)
ConnectionRetryCount	0

Attribute (Short Name)	Default
ConnectionRetryDelay	3
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Database (DB)	None
DataSourceName (DSN)	None
DefaultLongDataBuffLen (DLDBL)	1024
Description (n/a)	None
EnableDescribeParam (EDP)	0 (Disabled)
EncryptionMethod (EM)	0 (Disabled)
FailoverGranularity (FG)	0 (Non-Atomic)
FailoverMode (FM)	0 (Connection)
FailoverPreconnect (FP)	0 (Disabled)
HostName (HOST)	None
HostNameInCertificate (HNIC)	None
IANAAppCodePage (IACP)	4 (ISO 8559-1 Latin-1)
InteractiveClient (IC)	0 (Disabled)
KeepAlive (KA)	0 (Disabled)
KeyPassword (KP)	None
Keystore (KS)	None
KeystorePassword (KSP)	None
LoadBalanceTimeout (LBT)	0 (Disabled)
LoadBalancing (LB)	0 (Disabled)
LoginTimeout (LT)	15
LogonID (UID)	None
MaxPoolSize (MXPS)	100
MinPoolSize (MNPS)	0

Attribute (Short Name)	Default
Password (PWD)	None
Pooling (POOL)	0 (Disabled)
PortNumber (PORT)	3306
QueryTimeout (QT)	0
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
SSLlibName (SLN)	Empty string
TreatBinaryAsChar (TBAC)	0 (Disabled)
Truststore (TS)	None
TruststorePassword (TSP)	None
ValidateServerCertificate (VSC)	1 (Enabled)

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

`latest | openssl_version_number[[,openssl_version_number]...]`

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to `openssl_version_number`, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the `CryptoLibName` and `SSLibName` options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

1.1.1,1.0.2

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

[Advanced tab](#)

Alternate Servers

Attribute

AlternateServers (ASRV)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
(Database=dbname:HostName=hostvalue:PortNumber=portvalue[, . . .])
```

You must specify the database name, host name, and port number. The string has the format:

Example

The following Alternate Servers value defines two alternate database servers for connection failover:

```
(Database=MySQLdb1:HostName=MySQLhost1:PortNumber=5177,  
Database=MySQLdb2:HostName=MySQLhost2:PortNumber=5178)
```

Default

None

GUI Tab

[Failover tab](#)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI tab

[Advanced tab](#)

See also

See [Performance Considerations](#)

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI tab

[Pooling tab](#)

See also

- [Performance Considerations](#) on page 279

Connection Reset

Attribute

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI tab

[Pooling tab](#)

See also

- [Performance Considerations](#) on page 279

Connection Retry Count

Attribute

ConnectionRetryCount

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x , the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to x , the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when TLS/SSL is enabled using the Encryption Method connection option. When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, the connection fails and the driver returns an error.

Valid Values

cryptographic_protocol [, *cryptographic_protocol*]...

where:

cryptographic_protocol

is one of the following cryptographic protocols:

TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Default

```
TLSv1.2, TLSv1.1, TLSv1
```

GUI Tab

[Security tab](#)

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

```
absolute_path\openssl_filename
```

where:

```
absolute_path
```

is the absolute path to where the OpenSSL file is located

```
openssl_filename
```

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll; (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLibName](#) on page 274

Data Source name

Attribute

`DataSourceName` (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or `odbc.ini` file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database Name

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where:

database_name

is the name of a valid database.

Default

None

GUI Tab

[General tab](#)

Default Buffer Size for Long/LOB Columns (in Kb)

Attribute

DefaultLongDataBuffLen (DLDBL)

Purpose

The maximum length of data (in KB) the driver can fetch from long columns in a single round trip and the maximum length of data that the driver can send using the SQL_DATA_AT_EXEC parameter.

Valid Values

An integer in multiples of 1024

The value must be in multiples of 1024 (for example, 1024, 2048). You need to increase the default value if the total size of any Long data exceeds 1 MB. This value is multiplied by 1024 to determine the total maximum length of fetched data. For example, if you enter a value of 2048, the maximum length of data would be 1024 x 2048, or 2097152 (2 MB).

Notes

- This connection option can affect performance.

Default

1024

GUI tab

[Advanced tab](#)

See also

- [Performance Considerations](#) on page 279

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable SQLDescribeParam

Attribute

EnableDescribeParam (EDP)

Purpose

Determines whether the driver uses the SQLDescribeParam function, which describes parameters as a data type of SQL_VARCHAR with a length of 255 for statements.

Valid Values

0 | 1

Behavior

If set to 1 (enabled), the SQLDescribeParam function describes parameters as a data type of SQL_VARCHAR with a length of 255 for statements.

If set to 0 (disabled), the SQLDescribeParam function returns the standard ODBC error IM001.

Default

0 (Disabled)

GUI tab

[Advanced tab](#)

Encryption Method**Attribute**

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server. If the specified encryption method is not supported by the database server, the connection fails and the driver returns an error.

Valid Values

0 | 1

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using the SSL protocols specified in the Crypto Protocol Version connection option.

Notes

- This connection option can affect performance.

Default

0 (No Encryption)

GUI tab

[Security tab](#)

See also

- [Crypto Protocol Version](#) on page 256
- [Performance Considerations](#) on page 279

Failover Granularity**Attribute**

FailoverGranularity (FG)

Purpose

Determines whether the driver fails the entire failover process or continues with the process if errors occur while trying to reestablish a lost connection.

This option applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select).

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 (Non-Atomic), the driver continues with the failover process and posts any errors on the statement on which they occur.

If set to 1 (Atomic) the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

If set to 2 (Atomic Including Repositioning), the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

If set to 3 (Disable Integrity Check), the driver does not verify that the rows that were restored during the failover process match the original rows. This value applies only when Failover Mode is set to 2 (Select).

Default

0 (Non-Atomic)

GUI Tab

[Failover tab](#)

Failover Mode

Attribute

FailoverMode (FM)

Purpose

Specifies the type of failover method the driver uses.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Connection), the driver provides failover protection for new connections only.

If set to 1 (Extended Connection), the driver provides failover protection for new and lost connections, but not any work in progress.

If set to 2 (Select), the driver provides failover protection for new and lost connections. In addition, it preserves the state of work performed by the last Select statement executed.

Default

0 (Connection)

GUI Tab

[Failover tab](#)

Failover Preconnect

Attribute

FailoverPreconnect (FP)

Purpose

Specifies whether the driver tries to connect to the primary and an alternate server at the same time.

This attribute applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select) and at least one alternate server is specified.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver tries to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection. This value provides the best performance, but your application typically experiences a short wait while the failover connection is attempted.

If set to 1 (Enabled), the driver tries to connect to the primary and an alternate server at the same time. This can be useful if your application is time-sensitive and cannot absorb the wait for the failover connection to succeed.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Host Name

Attribute

HostName (HOST)

Purpose

The name or the IP address of the server to which you want to connect.

Valid Values

server_name | *IP_address*

where:

server_name

is the name of the server to which you want to connect.

IP_address

is the IP address of the server to which you want to connect.

Notes

- The IP address must be in IPv4 format.

Default

None

GUI Tab

[General tab](#)

Host Name In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

host_name | *#SERVERNAME#*

where:

host_name

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If set to a host name, the driver examines the subjectAltName values included in the certificate. If a dnsName value is present in the subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the dnsName value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the dnsName value.

If no `subjectAltName` values exist or a `dnsName` value is not in the list of `subjectAltName` values, then the driver compares the value specified for Host Name In Certificate with the `commonName` part of the Subject name in the certificate. The `commonName` typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the `commonName`. If multiple `commonName` parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the `commonName` parts.

If set to `#SERVERNAME#`, the driver compares the host server name specified as part of a data source or connection string to the `dnsName` or the `commonName` value.

Default

None

GUI tab

[Security tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI tab

[Advanced tab](#)

Interactive Client

Attribute

InteractiveClient (IC)

Purpose

Determines how long a connection can be idle before the server disconnects it.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver initializes the `wait_time` session variable for the connection with the value of the global `interactive_timeout` variable.

If set to 0 (Disabled), the driver initializes the `wait_timeout` session variable with the value of the global `wait_timeout` variable.

Notes

- The `wait_timeout` variable controlled by the Interactive Client option is a session variable that can be modified by the application after the connection has been established. The Interactive Client option controls only the initial value of the `wait_timeout` session variable.

Default

0 (Disabled)

GUI tab

[Advanced tab](#)

Key Password

Attribute

KeyPassword (KP)

Purpose

The password used to access the individual keys in the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. Keys stored in a keystore can be individually password-protected. To extract the key from the keystore, the driver must have the password of the key.

Valid Values

key_password

where:

key_password

is the password of a key in the keystore.

Default

None

GUI tab

[Security tab](#)

Keystore

Attribute

Keystore (KS)

Purpose

The name of the directory containing the keystore file to be used when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request. If you do not specify a directory, the current directory is used.

Valid Values

keystore_directory

where:

keystore_directory

is the location of the keystore file.

Notes

- The keystore and truststore files may be the same file.

Default

None

GUI tab

[Security tab](#)

Keystore Password

Attribute

KeystorePassword (KSP)

Purpose

The password used to access the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request.

Valid Values

`keystore_password`

where:

keystore_password

is the password of the keystore file.

Notes

- The keystore and truststore files may be the same file; therefore, they may have the same password.

Default

None

GUI tab

[Security tab](#)

Load Balance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

Specifies the number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Valid Values

0 | *x*

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to *x*, inactive connections are closed after the specified number of seconds passes.

Notes

- The Min Pool Size option may cause some connections to ignore this value.

Default

0 (Disabled)

GUI tab

[Pooling tab](#)

See also

- [Performance Considerations](#) on page 279

Load Balancing

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x, the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

15

GUI tab

[Advanced tab](#)

Max Pool Size

Attribute

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

For example, if set to 20, the maximum number of connections allowed in the pool is 20.

Default

100

GUI tab

[Pooling tab](#)

See also

- [Performance Considerations](#) on page 279

Min Pool Size

Attribute

MinPoolSize (MNPS)

Purpose

The minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Valid Values

0 | x

where:

x

is an integer from 1 to 65535.

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

Notes

- This connection option can affect performance.

Example

If set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Default

0

GUI tab

[Pooling tab](#)

See also

- [Performance Considerations](#) on page 279

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Purpose

The port number of the server listener.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your database administrator for the correct number.

Notes

- This option is mutually exclusive with the Server Name and TNSNames File options.

Default

3306

GUI Tab

[General tab](#)

Query Timeout

Attribute

QueryTimeout (QT)

Purpose

The number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the SQL_ATTR_QUERY_TIMEOUT statement attribute on the SQLSetStmtAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to -1, the query does not time out. The driver silently ignores the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to 0, the query does not time out, but the driver responds to the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to x , all queries time out after the specified number of seconds unless the application overrides this value by setting the SQL_ATTR_QUERY_TIMEOUT attribute.

Default

0

GUI tab

[Advanced tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x` , where x is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI tab

[Advanced tab](#)

SSLibName

Attribute

SSLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\
```

```
Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
SSLLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 257

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Treat Binary Data as Character Data

Attribute

TreatBinaryAsChar (TBAC)

Purpose

Allows data that MySQL stores as BINARY or VARBINARY to be described and returned as CHAR or VARCHAR values, respectively.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver describes and returns data that MySQL stores as BINARY or VARBINARY as CHAR or VARCHAR values, respectively.

If set to 0 (Disabled), the driver describes and returns data that MySQL describes as BINARY or VARBINARY as BINARY or VARBINARY values, respectively.

Example

Create the following MySQL table:

```
CREATE TABLE binTable (col1 binary(3))
```

Then, execute the following Insert statement:

```
INSERT INTO binTable values('abc')
```

Then, execute the following query:

```
SELECT col1 FROM binTable
```

Using this example, the driver would return the value of col1 as a CHAR value, "abc", instead of a BINARY value "616263".

Default

0 (Disabled)

GUI tab

[Advanced tab](#)

Truststore

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used.

Valid Values

truststore_directory\filename

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Notes

- The truststore and keystore files may be the same file.

Default

None

GUI tab

[Security tab](#)

Truststore Password

Attribute

TruststorePassword (TSP)

Purpose

The password that is used to access the truststore file when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

- The truststore and keystore files may be the same file; therefore, they may have the same password.

Default

None

GUI tab

[Security tab](#)

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI tab

[Security tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Truststore information is specified using the Trust Store and Trust Store Password options.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Default

1 (Enabled)

GUI tab

[Security tab](#)

Performance Considerations

The following connection options can enhance driver performance.

The option names found on the tabs of the driver Setup dialog box are the same as the connection string attribute names unless otherwise noted in parentheses. The connection string attribute name does not have spaces between the words. For example, the option name Application Using Threads is equivalent to the connection string attribute name ApplicationUsingThreads.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Default Buffer Size for Long/LOB Columns (DefaultLongDataBuffLen): To improve performance when your application fetches images, pictures, or long text or binary data, a buffer size can be set to accommodate the maximum size of the data. The buffer size should only be large enough to accommodate the maximum amount of data retrieved; otherwise, performance is reduced by transferring large amounts of data into an oversized buffer. If your application retrieves more than 1 MB of data, the buffer size should be increased accordingly.

Connection Pooling (Pooling): If you enable the driver to use connection pooling, you can set additional options that affect performance:

- **Load Balance Timeout (LoadBalanceTimeout):** You can define how long to keep connections in the pool. The time that a connection was last used is compared to the current time and, if the timespan exceeds the value of the Load Balance Timeout option, the connection is destroyed. The Min Pool Size option can cause some connections to ignore this value.
- **Connection Reset (ConnectionReset):** Resetting a re-used connection to the initial configuration settings impacts performance negatively because the connection must issue additional commands to the server.
- **Max Pool Size (MaxPoolSize):** Setting the maximum number of connections that the pool can contain too low might cause delays while waiting for a connection to become available. Setting the number too high wastes resources.

- **Min Pool Size (MinPoolSize):** A connection pool is created when the first connection with a unique connection string connects to the database. The pool is populated with connections up to the minimum pool size, if one has been specified. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Encryption Method (EncryptionMethod): Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) that is required to encrypt and decrypt data.

Failover Mode (FailoverMode): Although high availability that replays queries after a failure provides increased levels of protection, it can adversely affect performance because of increased overhead.

Data Types

The following table shows how the MySQL data types map to the standard ODBC data types.

Table 24: MySQL Data Types

MySQL	ODBC
BIGINT	SQL_BIGINT
BIGINT UNSIGNED	SQL_BIGINT
BINARY	SQL_BINARY
BIT	SQL_BINARY
BLOB	SQL_LONGVARBINARY
CHAR	SQL_CHAR
DATE	SQL_TYPE_DATE
DATETIME	SQL_TYPE_TIMESTAMP
DECIMAL	SQL_DECIMAL
DECIMAL UNSIGNED	SQL_DECIMAL
DOUBLE	SQL_DOUBLE
DOUBLE UNSIGNED	SQL_DOUBLE
FLOAT	SQL_REAL
FLOAT UNSIGNED	SQL_REAL
INTEGER	SQL_INTEGER
INTEGER UNSIGNED	SQL_INTEGER
LONGBLOB	SQL_LONGVARBINARY
LONGTEXT	SQL_LONGVARCHAR

MySQL	ODBC
MEDIUMBLOB	SQL_LONGVARBINARY
MEDIUMINT	SQL_INTEGER
MEDIUMINT UNSIGNED	SQL_INTEGER
MEDIUMTEXT	SQL_LONGVARCHAR
SMALLINT	SQL_SMALLINT
SMALLINT UNSIGNED	SQL_SMALLINT
TEXT	SQL_LONGVARCHAR
TIME	SQL_TYPE_TIME
TIMESTAMP	SQL_TYPE_TIMESTAMP
TINYBLOB	SQL_LONGVARBINARY
TINYINT	SQL_TINYINT
TINYINT UNSIGNED	SQL_TINYINT
TINYTEXT	SQL_LONGVARCHAR
VARBINARY	SQL_VARBINARY
VARCHAR	SQL_VARCHAR
YEAR	SQL_SMALLINT

See [Retrieving Data Type Information](#) on page 72 for more information about data types.

Note: The Treat Binary Data as Character Data connection option affects how certain ODBC data types are reported. See [Treat Binary Data as Character Data](#) on page 276 for details.

Unicode Support

When the character set of a character column is Unicode, then the MySQL Wire Protocol driver maps the MySQL data type to Unicode data type as follows:

MySQL Data Type	Mapped to . . .
CHAR	SQL_WCHAR
LONGTEXT	SQL_WLONGVARCHAR
MEDIUMTEXT	SQL_WLONGVARCHAR

MySQL Data Type	Mapped to . . .
TEXT	SQL_WLONGVARCHAR
TINYTEXT	SQL_WLONGVARCHAR
VARCHAR	SQL_WVARCHAR

Advanced Features

The driver supports the following advanced features:

- Failover
- Security
- Connection Pooling

Failover

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Security

The driver supports authentication and encryption. Security connection options are located on the [Security tab](#) of the driver Setup dialog box. See [Using Security](#) on page 89 for a general description of security and its implementation.

Connection Pooling

The driver supports connection pooling and its related connection options. Connection pooling connection options are located on the [Pooling tab](#) of the driver Setup dialog box. See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling and its implementation.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

MySQL supports isolation levels 0 (read uncommitted), 1 (read committed), 2 (repeatable read), and 3 (serializable). The default is 1.

MySQL supports record-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the minimum SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

In addition, the following functions are supported:

- SQLColumnPrivileges
- SQLForeignKeys
- SQLTablePrivileges

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The MySQL Wire Protocol driver supports multiple connections and multiple statements per connection to the MySQL database system.

The Oracle Wire Protocol Driver

Note: This section documents the features and functionality of the 7.1 version of the driver. For the current version of the driver, visit Progress DataDirect Connectors Documentation page:

<https://docs.progress.com/datadirect-connectors/page/DataDirect-Connectors-by-data-source.html>.

The DataDirect Connect *for* ODBC and DataDirect Connect64 *for* ODBC Oracle Wire Protocol driver (the Oracle Wire Protocol driver) each support Oracle database servers.

For the latest support information, visit the Progress DataDirect Supported Configurations page:

<https://www.progress.com/supported-configurations/datadirect>.

The Oracle Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the Windows, UNIX, and Linux environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect product for the file name of the Oracle Wire Protocol driver.

Note: The Oracle Wire Protocol driver does not require any Oracle client software. Progress DataDirect also provides an Oracle client-based driver; see [The Oracle Driver](#) on page 617 for details.

Driver Requirements

The driver has no client requirements.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 301 and [Connection Option Descriptions for Oracle Wire Protocol](#) on page 303 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

UNIX[®] On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions for Oracle Wire Protocol](#) on page 303 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Oracle)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX[®] On UNIX and Linux, data sources are stored in the odbc.ini file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure an Oracle data source:

1. Start the ODBC Administrator:

-  On Windows, start the ODBC Administrator by selecting its icon from the DataDirect Connect program group.
- **UNIX**[®] On Linux, change to the *install_dir/tools* directory and, at a command prompt, enter:


```
odbcadmin
```

where *install_dir* is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

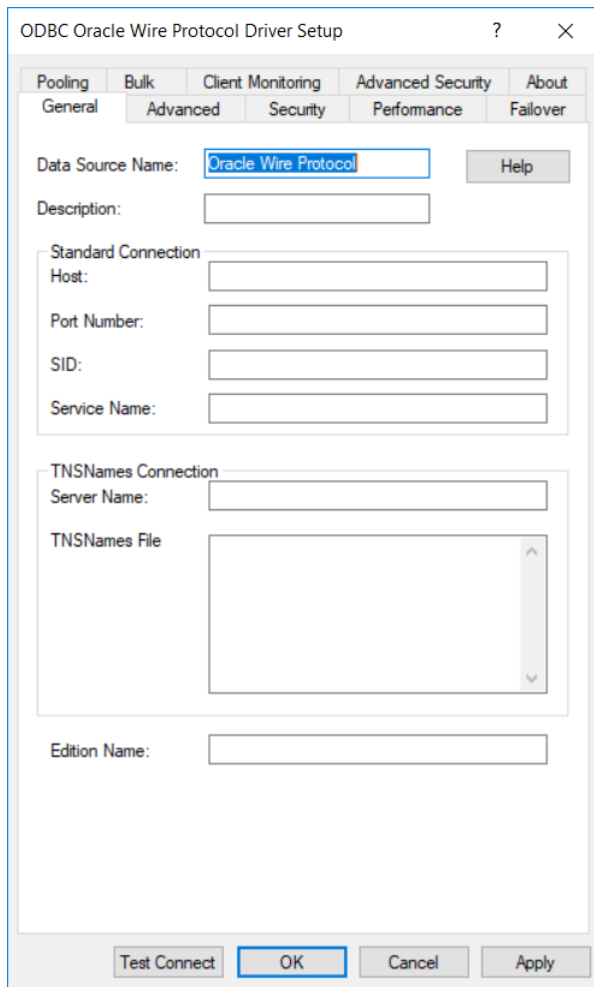
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 18: General tab



Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

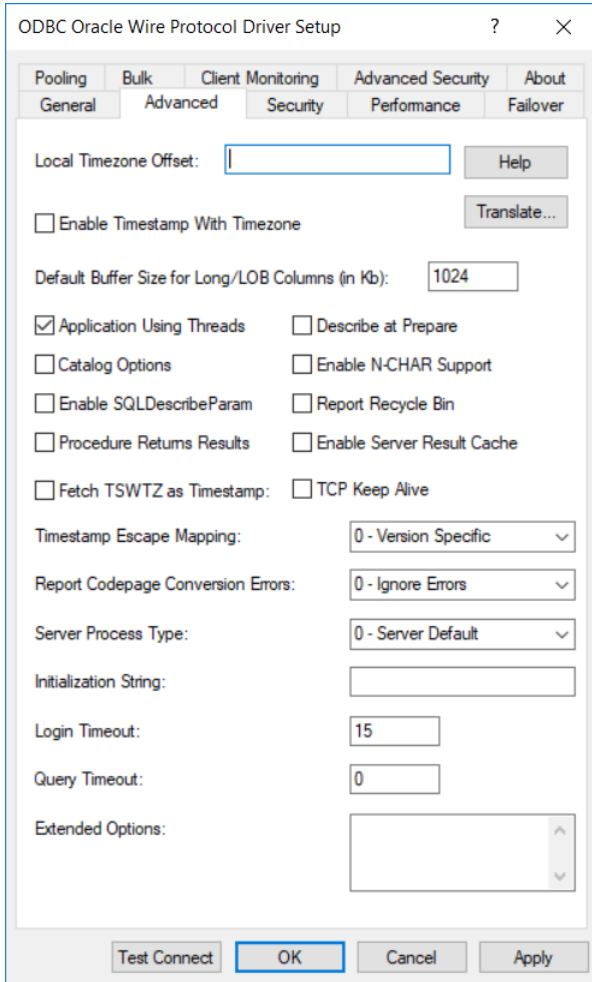
3. On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 326	None
Description on page 328	None
Host on page 340	None
Port Number on page 351	None
SID on page 360	None
Service Name on page 359	None

Connection Options: General	Default
Server Name on page 358	None
TNSNames File on page 363	None
Edition Name on page 328	None

4. Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 19: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Local Timezone Offset on page 346	None
Enable Timestamp with Timezone on page 333	Disabled
Default Buffer Size for Long/LOB Columns (in Kb) on page 327	1024

Connection Options: Advanced	Default
Application Using Threads on page 311	Enabled
Describe at Prepare on page 327	Disabled
Catalog Options on page 317	Disabled
Enable N-CHAR Support on page 330	Disabled
Enable SQLDescribeParam on page 332	Disabled
Report Recycle Bin on page 357	Disabled
Procedure Returns Results on page 354	Disabled
Enable Server Result Cache on page 331	Disabled
Fetch TSWTZ as Timestamp on page 338	Disabled
TCP Keep Alive on page 362	Disabled
Timestamp Escape Mapping on page 362	0 - Version Specific
Report Codepage Conversion Errors on page 356	0 - Ignore Errors
Server Process Type on page 358	0 - Server Default
Initialization String on page 342	None
Login Timeout on page 347	15
Query Timeout on page 355	0
IANAAppCodePage on page 342 UNIX ONLY	4 (ISO 8559-1 Latin-1)



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.



Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- Optionally, click the **Security** tab to specify security data source settings.

Figure 20: Security tab

See [Using Security](#) on page 89 for a general description of authentication and encryption and their configuration requirements.

See [OS Authentication](#) on page 376 for a discussion of Oracle and SSL encryption.

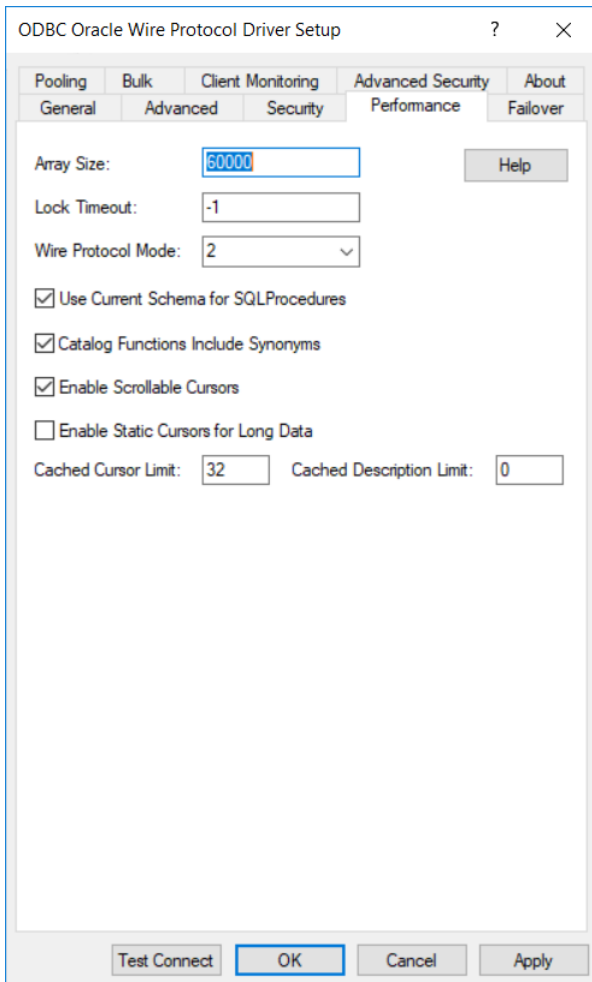
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Security	Default
User Name on page 366	None
Authentication Method on page 312	1 - Encrypt Password
GSS Client Library on page 340	native

Connection Options: Security	Default
Encryption Method on page 334	0 - No Encryption
Crypto Protocol Version on page 323	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Validate Server Certificate on page 366	Enabled
Truststore on page 364	None
Truststore Password on page 364	None
Key Store on page 344	None
Key Store Password on page 344	None
Key Password on page 343	None
Host Name In Certificate on page 341	None

- Optionally, click the **Performance** tab to specify performance data source settings.

Figure 21: Performance tab

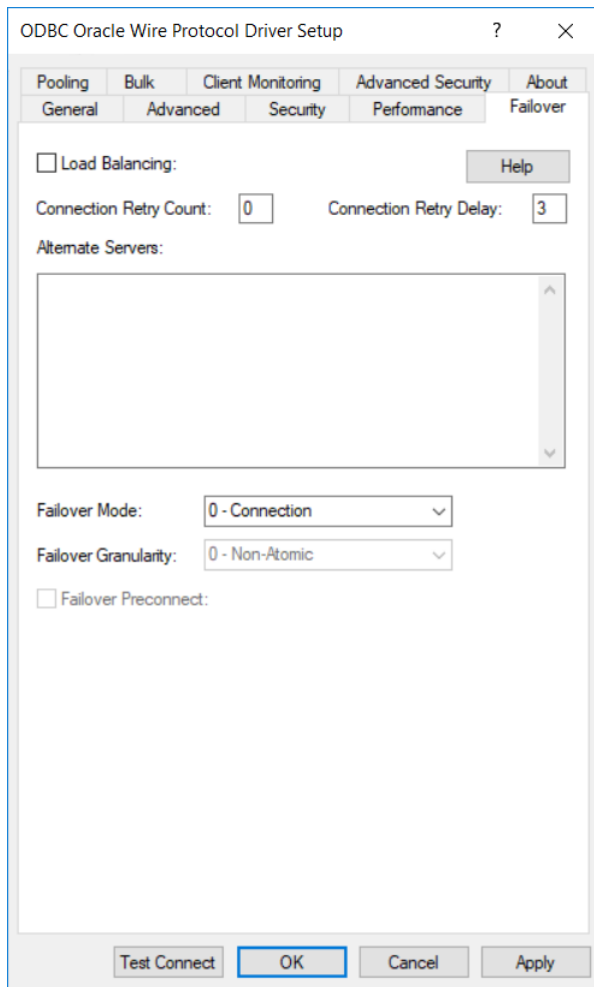


On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Performance	Default
Array Size on page 311	60000
Lock Timeout on page 347	-1
Wire Protocol Mode on page 367	2
Use Current Schema for SQLProcedures on page 365	Enabled
Catalog Functions Include Synonyms on page 317	Enabled
Enable Scrollable Cursors on page 330	Enabled
Enable Static Cursors for Long Data on page 332	Disabled
Cached Cursor Limit on page 315	32
Cached Description Limit on page 316	0

7. Optionally, click the **Failover** tab to specify failover data source settings.

Figure 22: Failover tab



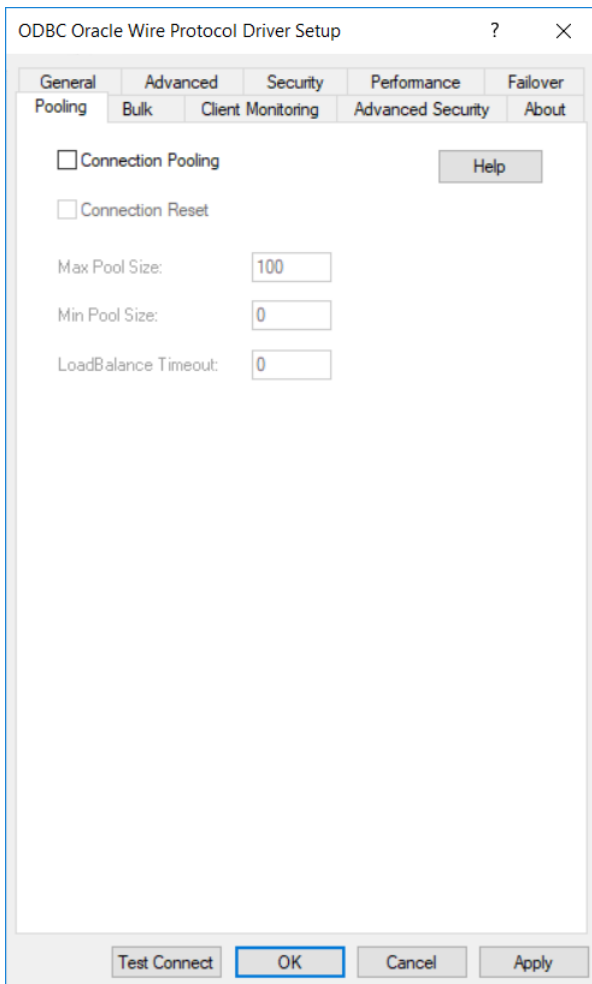
See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 346	Disabled
Connection Retry Count on page 321	0
Connection Retry Delay on page 322	3
Alternate Servers on page 309	None
Failover Mode on page 337	0 - Connection
Failover Granularity on page 336	0 - Non-Atomic
Failover Preconnect on page 338	Disabled

8. Optionally, click the **Pooling** tab to specify connection pooling data source settings.

Figure 23: Pooling tab



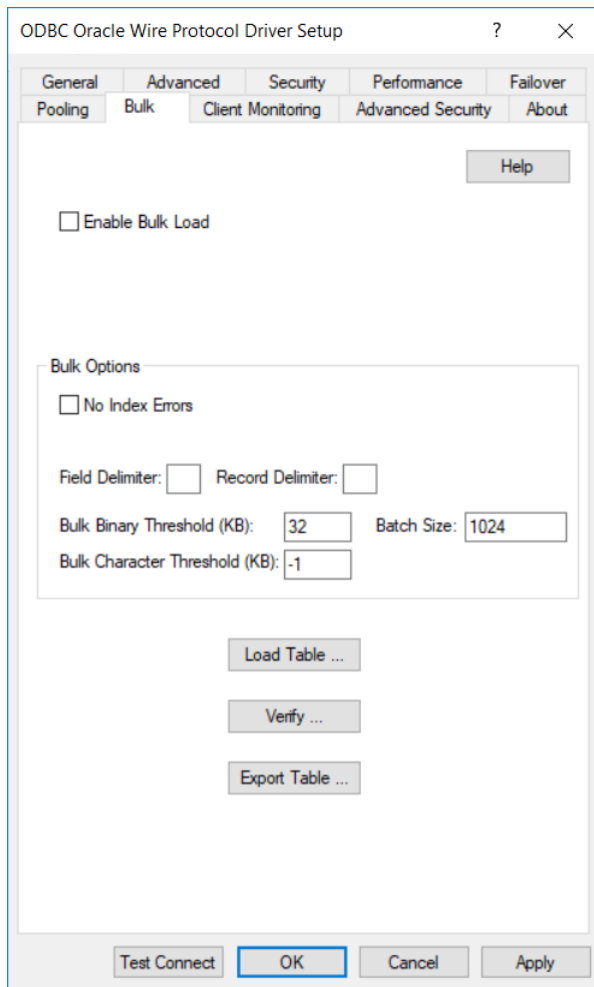
See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 320	Disabled
Connection Reset on page 321	Disabled
Max Pool Size on page 348	100
Min Pool Size on page 349	0
Load Balance Timeout on page 345	0

9. Optionally, click the **Bulk** tab to specify DataDirect Bulk Load data source settings.

Figure 24: Bulk tab



See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect Bulk Load.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Bulk	Default
Enable Bulk Load on page 329	Disabled
Bulk Options on page 315 (No Index Errors is described under the Bulk Options description.)	No Index Errors: disabled
Field Delimiter on page 339	None
Record Delimiter on page 356	None
Bulk Binary Threshold on page 313	32
Batch Size on page 313	1024
Bulk Character Threshold on page 314	-1

If your application is already coded to use parameter array batch functionality, you can leverage DataDirect Bulk Load features through the Enable Bulk Load connection option. Enabling this option automatically converts the parameter array batch operation to use the database bulk load protocol.

If you are not using parameter array batch functionality, you can export data to a bulk load data file, verify the metadata of the bulk load configuration file against the structure of the target table, and bulk load data to a table. Use the following steps to accomplish these tasks.

- a) To export data from a table to a bulk load data file, click **Export Table** from the Bulk tab. The Export Table dialog box appears.

Figure 25: Export Table dialog box

Both a bulk data file and a bulk configuration file are produced by exporting a table. The configuration file has the same name as the data file, but with an XML extension. See [Using DataDirect Bulk Load](#) on page 101 for details about these files.

The bulk export operation can create a log file and can also export to external files. See [External Overflow Files](#) on page 108 for more information. The export operation can be configured such that if any errors or warnings occur:

- The operation always completes.
- The operation always terminates.
- The operation terminates after a certain threshold of warnings or errors is exceeded.

Table Name: A string that specifies the name of the source database table containing the data to be exported.

Export Filename: A string that specifies the path (relative or absolute) and file of the bulk load data file to which the data is to be exported. It also specifies the file name of the bulk configuration file. The file name must be the fully qualified path to the bulk data file. These files must not already exist; if one of both of them already exists, an error is returned.

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. The log file is created if it does not exist. The file name must be the fully qualified path to the log file. Events logged to this file are:

- Total number of rows fetched
- A message for each row that failed to export
- Total number of rows that failed to export

- Total number of rows successfully exported

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not supply a value for Log Filename, no log file is created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

Code Page: A value that specifies the code page value to which the driver must convert all data for storage in the bulk data file. See [Character Set Conversions](#) on page 108 for more information.

The default value on Windows is the current code page of the machine. On UNIX/Linux, the default value is 4 (ISO 8559-1 Latin-1).

Click **Export Table** to connect to the database and export data to the bulk data file or click **Cancel**.

Click **Export Table** to connect to the database and export data to the bulk data file or click **Cancel**.

- b) To verify the metadata of the bulk load configuration file against the structure of the target database table, click **Verify** from the Bulk tab. See [Verification of the Bulk Load Configuration File](#) on page 106 for details. The ODBC Oracle Wire Protocol Verify Driver Setup dialog box appears.

Figure 26: ODBC Oracle Wire Protocol Verify Driver Setup dialog box

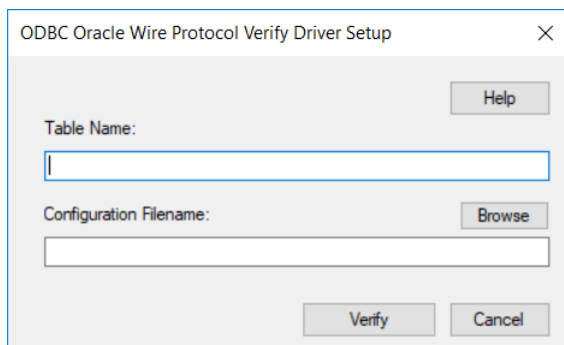


Table Name: A string that specifies the name of the target database table into which the data is to be loaded.

Configuration Filename: A string that specifies the path (relative or absolute) and file name of the bulk configuration file. The file name must be the fully qualified path to the configuration file.

Click **Verify** to verify table structure or click **Cancel**.

- c) To bulk load data from the bulk data file to a database table, click **Load Table** from the Bulk tab. The Load File dialog box appears.

Figure 27: Load File dialog box

The load operation can create a log file and can also create a discard file that contains rows rejected during the load. The discard file is in the same format as the bulk load data file. After fixing reported issues in the discard file, the bulk load can be reissued using the discard file as the bulk load data file.

The export operation can be configured such that if any errors or warnings occur:

- The operation always completes.
- The operation always terminates.
- The operation terminates after a certain threshold of warnings or errors is exceeded.

If a load fails, the Load Start and Load Count options can be used to control which rows are loaded when a load is restarted after a failure.

Table Name: A string that specifies the name of the target database table into which the data is loaded.

Load Data Filename: A string that specifies the path (relative or absolute) and file name of the bulk data file from which the data is loaded. The file name must be the fully qualified path to the bulk data file.

Configuration Filename: A string that specifies the path (relative or absolute) and file name of the bulk configuration file. The file name must be the fully qualified path to the configuration file.

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. The file name must be the fully qualified path to the log file. Specifying a value for Log Filename creates the file if it does not already exist. Events logged to this file are:

- Total number of rows read
- Message for each row that failed to load

- Total number of rows that failed to load
- Total number of rows successfully loaded

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Log Filename, no log file is created.

Discard Filename: A string that specifies the path (relative or absolute) and file name of the bulk discard file. The file name must be the fully qualified path to the discard file. Any row that cannot be inserted into database as result of bulk load is added to this file, with the last row rejected added to the end of the file.

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Discard Filename, a discard file is not created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Load Start: A value that specifies the first row to be loaded from the data file. Rows are numbered starting with 1. For example, when Load Start is 10, the first 9 rows of the file are skipped and the first row loaded is row 10. This option can be used to restart a load after a failure.

The default value is 1.

Read Buffer Size (KB): A value that specifies the size, in KB, of the buffer that is used to read the bulk data file for a bulk load operation.

The default value is 2048.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

Load Count: A value that specifies the number of rows to be loaded from the data file. The bulk load operation loads rows up to the value of Load Count from the file to the database. It is valid for Load Count to specify more rows than exist in the data file. The bulk load operation completes successfully when either the number of rows specified by the Load Count value has been loaded or the end of the data file is reached. This option can be used in conjunction with Load Start to restart a load after a failure.

The default value is the maximum value for SQLULEN. If set to 0, no rows are loaded.

Click **Load Table** to connect to the database and load the table or click **Cancel**.

10. Optionally, click the **Client Monitoring** tab to specify additional data source settings.

Figure 28: Client Monitoring tab

ODBC Oracle Wire Protocol Driver Setup

General Advanced Security Performance Failover
Pooling Bulk Client Monitoring Advanced Security About

Accounting Info: Help

Action:

Application Name:

Client Host Name:

Client ID:

Client User:

Module:

Program ID:

Test Connect OK Cancel Apply

See [Using Client Information](#) on page 87 for additional information about client monitoring.

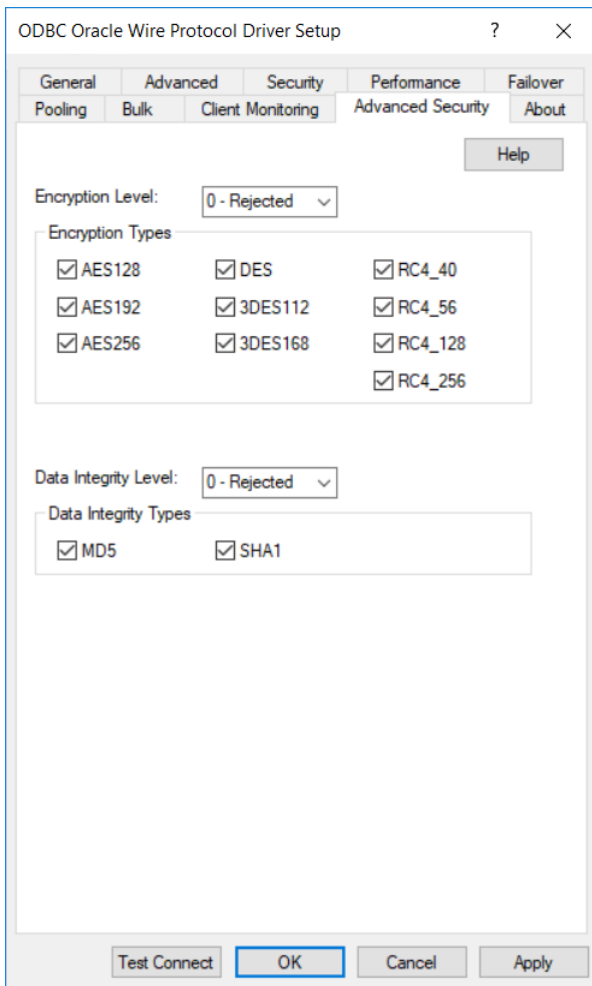
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Client Monitoring	Default
Accounting Info on page 307	None
Action on page 307	None
Application Name on page 310	None
Client Host Name on page 318	None
Client ID on page 319	None
Client User on page 319	None

Connection Options: Client Monitoring	Default
Module on page 350	None
Program ID on page 354	None

11. Optionally, click the **Advanced Security** tab to specify settings for Oracle Advanced Security (OAS).

Figure 29: Advanced Security tab



See [Oracle Advanced Security](#) on page 375 for a general description of encryption configuration.

Refer to your Oracle documentation for a discussion of Oracle Advanced Security.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced Security	Default
Encryption Level on page 333	0 - Rejected
Encryption Types on page 335	All listed encryption algorithms are selected

Connection Options: Advanced Security	Default
Data Integrity Level on page 325	0 - Rejected
Data Integrity Types on page 325	SHA1 and MD5 are selected

12. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Oracle\)](#) on page 302 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.

- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
- If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

13. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify `attribute=value` pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{ }driver_name[ }][;attribute=value[;attribute=value]...]
```

[Connection Option Descriptions for Oracle Wire Protocol](#) on page 303 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Oracle Wire Protocol is:

```
DSN=Accounting;ID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=OracleWP.dsn;ID=JOHN;PWD=XYZZY
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Oracle Wire Protocol};HOST=server1;PORT=1522;  
UID=JOHN;PWD=XYZZY;SERVICENAME=SALES.US.ACME.COM
```

Using a Logon Dialog Box (Oracle)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Figure 30: Logon to Oracle Wire Protocol dialog box

In this dialog box, provide the following information:

Note: To configure a standard connection, complete the first four fields and skip to Step 6 on page 303

1. In the Host field, type either the name or the IP address of the server to which you want to connect.
The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details concerning these formats.
If you enter a value for this field, the Server Name field is not available.
This field is not available if you enter a value for the Server Name field.
2. In the Port Number field, type the number of your Oracle listener. Check with your database administrator for the correct number.
If you enter a value for this field, the Server Name field is not available.
This field is not available if you enter a value for the Server Name field.
3. In the SID field, type the Oracle System Identifier that refers to the instance of Oracle running on the server.
If you enter a value for this field, the Server Name and Service Name fields are not available.
This field is not available if you enter a value for the Service Name or Server Name fields.
4. In the Service Name field, type the Oracle service name that specifies the database used for the connection.
See Service Name under Step 3 on page 286 in [Data Source Configuration through a GUI \(Oracle\)](#) on page 284 for details.

If you enter a value for this field, the Server Name and SID fields are not available.

This field is not available if you enter a value for the SID or Server Name field.

Note: If you want to configure a TNSNames connection, complete only the following two fields.

5. In the Server Name field, type a net service name that exists in the TNSNAMES.ORA file. The corresponding entry in the TNSNAMES.ORA file is used to obtain Host, Port Number, and SID information.

If you enter a value for this field, the Host, Port Number, SID, and Service Name fields are not available.

If you enter a value for either the Host, Port Number, SID, or Service Name fields, this field is not available.

6. If required, type your Oracle user name.
7. If required, type your Oracle password.
8. Click **OK** to log on to the Oracle database installed on the server you specified and to update the values in the Registry.

Note: You can also use OS Authentication to connect to an Oracle database. See [OS Authentication](#) on page 376 for details.

Connection Option Descriptions for Oracle Wire Protocol

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Oracle Wire Protocol driver.

Table 25: Oracle Wire Protocol Attribute Names

Attribute (Short Name)	Default
AccountingInfo (AI)	None
Action (ACT)	None
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
AlternateServers (ASRV)	None
ApplicationName (AN)	None
ApplicationUsingThreads (AUT)	1 (Enabled)
ArraySize (AS)	60000

Attribute (Short Name)	Default
AuthenticationMethod (AM)	1 (Encrypt Password)
BulkBinaryThreshold (BBT)	32
BulkCharacterThreshold (BCT)	-1
BulkLoadBatchSize (BLBS)	1024
BulkLoadFieldDelimiter (BLFD)	None
BulkLoadOptions (BLO)	0
BulkLoadRecordDelimiter (BLRD)	None
CachedCursorLimit (CCL)	32
CachedDescriptionLimit (CDL)	0
CatalogIncludesSynonyms (CIS)	1 (Enabled)
CatalogOptions (CO)	0 (Disabled)
ClientHostName (CHN)	None
ClientID (CID)	None
ClientUser (CU)	None
ConnectionReset (CR)	0 (Disabled)
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1, SSLv3
DataIntegrityLevel (DIL)	0 (Disabled)
DataIntegrityTypes (DIT)	SHA1,MD5
DataSourceName (DSN)	None
DefaultLongDataBuffLen (DLDBL)	1024
DescribeAtPrepare (DAP)	0 (Disabled)
Description (n/a)	None
EditionName (EN)	None

Attribute (Short Name)	Default
EnableBulkLoad (EBL)	0 (Disabled)
EnableDescribeParam (EDP)	0 (Disabled)
EnableNcharSupport (ENS)	0 (Disabled)
EnableScrollableCursors (ESC)	1 (Enabled)
EnableServerResultCache (ESRC)	0 (Disabled)
EnableStaticCursorsForLongData (ESCLD)	0 (Disabled)
EnableTimestampwithTimezone (ETWT)	0 (Disabled)
EncryptionLevel (EL)	0 (Disabled)
EncryptionMethod (EM)	0 (No Encryption)
EncryptionTypes (ET)	No encryption methods are specified. The driver sends a list of all of the encryption methods to the Oracle server.
FailoverGranularity (FG)	0 (Non-Atomic)
FailoverMode (FM)	0 (Connection)
FailoverPreconnect (FP)	0 (Disabled)
FetchTSWTZasTimestamp (FTSWTZAT)	0 (Disabled)
GSSClient (GSSC)	native
HostName (HOST)	None
HostNameInCertificate (HNIC)	None
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
InitializationString (IS)	None
KeepAlive (KA)	0 (Disabled)
KeyPassword (KP)	None
Keystore (KS)	None
KeystorePassword (KSP)	None
LoadBalanceTimeout (LBT)	0

Attribute (Short Name)	Default
LoadBalancing (LB)	0 (Disabled)
LocalTimezoneOffset (LTZO)	"" (Empty String)
LockTimeout (LTO)	-1
LoginTimeout (LT)	15
LogonID (UID)	None
MaxPoolSize (MXPS)	100
MinPoolSize (MNPS)	0
Module (MOD)	None
Password (PWD)	None
Pooling (POOL)	0 (Disabled)
PortNumber (PORT)	None
PRNGSeedFile (PSF) UNIX/Linux only	/dev/random
PRNGSeedSource (PSS) UNIX/Linux only	0 (File)
ProcedureRetResults (PRR)	0 (Disabled)
ProgramID (PID)	None
QueryTimeout (QT)	0
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
ReportRecycleBin (RRB)	0 (Disabled)
ServerName (SRVR)	None
ServerType (ST)	0 (Server Default)
ServiceName (SN)	None
SID (SID)	None
SSLLibName (SLN)	Empty string
TimestampEscapeMapping (TEM)	0 (Oracle Version Specific)
TNSNamesFile (TNF)	None
Truststore (TS)	None

Attribute (Short Name)	Default
TruststorePassword (TSP)	None
UseCurrentSchema (UCS)	1 (Enabled)
ValidateServerCertificate (VSC)	1 (Enabled)
WireProtocolMode (WPM)	2

Accounting Info

Attribute

AccountingInfo (AI)

Purpose

Accounting information to be stored in the database. This value sets the CLIENT_INFO value of the V\$SESSION table on the server. This value is used by the client information feature.

Valid Values

string

where:

string

is the accounting information.

Notes

- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 367

Action

Attribute

Action (ACT)

Purpose

The current action (Select, Insert, Update, or Delete, for example) within the current module. This value sets the ACTION column of the V\$SESSION table on the server. This value is used by the client information feature.

This option only applies to connections to Oracle 10g R2 and higher database servers.

Valid Values

string

where:

string

is the current action.

Notes

- You can also specify this information using the Oracle DBMS_APPLICATION_INFO.SET_ACTION procedure or the DBMS_APPLICATION_INFO.SET_MODULE procedure.
- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 367

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

latest | *openssl_version_number*[[,*openssl_version_number*] ...]

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to *openssl_version_number*, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the `CryptoLibName` and `SSLibName` options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

1.1.1,1.0.2

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

[Advanced tab](#)

Alternate Servers

Attribute

AlternateServers (ASRV)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
(HostName=hostvalue:PortNumber=portvalue:{SID=sidvalue | ServiceName=servicevalue}[,  
. . .])
```

You must specify the host name, port number, and either the SID or service name of each alternate server.

Example

The following Alternate Servers value defines two alternate database servers for connection failover:

```
(HostName=AccountingOracleServer:PortNumber=1521:  
SID=Accounting,HostName=255.201.11.24:PortNumber=1522:  
ServiceName=ABackup.NA.MyCompany)
```

Default

None

GUI tab

[Failover tab](#)

Application Name

Attribute

ApplicationName (AN)

Purpose

The name of the application to be stored in the database. This value sets the `dbms_session` value in the database and the `PROGRAM` value of the `V$SESSION` table on the server. This value is used by the client information feature.

Valid Values

string

where:

string

is the name of the application.

Notes

- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 367

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 367

Array Size

Attribute

ArraySize (AS)

Purpose

The number of bytes the driver can fetch in a single network round trip. Larger values increase throughput by reducing the number of times the driver fetches data across the network. Smaller values increase response time, as there is less of a delay waiting for the server to transmit data.

Valid Values

An integer from 1 to 4,294,967,296 (4 GB)

The value 1 does not define the number of bytes but, instead, causes the driver to allocate space for exactly one row of data.

Notes

- This connection option can affect performance.

Default

60000

GUI Tab

[Performance tab](#)

See also

[Performance Considerations](#) on page 367

Authentication Method

Attribute

AuthenticationMethod (AM)

Purpose

Specifies the method the driver uses to authenticate the user to the server when a connection is established. If the specified authentication method is not supported by the database server, the connection fails and the driver generates an error.

Valid Values

1 | 3 | 4 | 5 | 6

Behavior

If set to 1 (Encrypt Password), the driver sends the user ID in clear text and an encrypted password to the server for authentication.

If set to 3 (Client Authentication), the driver uses client authentication when establishing a connection. The database server relies on the client to authenticate the user and does not provide additional authentication.

If set to 4 (Kerberos Authentication), the driver uses Kerberos authentication. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

When set to 5 (Kerberos with UID & PWD), the driver uses both Kerberos authentication and user ID and password authentication. The driver first authenticates the user using Kerberos. If a user ID and password are specified, the driver reauthenticates using the user name and password supplied. An error is generated if a user ID and password are not specified.

If set to 6 (NTLM), the driver uses NTLMv1 authentication for Windows clients.

Default

1 (Encrypt Password)

GUI tab

[Security tab](#)

Batch Size

Attribute

BulkLoadBatchSize (BLBS)

Purpose

The number of rows that the driver sends to the database at a time during bulk operations. This value applies to all methods of bulk loading.

Valid Values

0 | x

where

x

is a positive integer that specifies the number of rows to be sent.

Default

1024

GUI Tab

[Bulk tab](#)

Bulk Binary Threshold

Attribute

BulkBinaryThreshold (BBT)

Purpose

The maximum size, in KB, of binary data that is exported to the bulk data file.

Valid Values

-1 | 0 | x

where

x

is an integer that specifies the number of KB.

Behavior

If set to -1, all binary data, regardless of size, is written to the bulk data file, not to an external file.

If set to 0, all binary data, regardless of size, is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

If set to x , any binary data exceeding this specified number of KB is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

Default

32

GUI Tab

[Bulk tab](#)

Bulk Character Threshold

Attribute

BulkCharacterThreshold (BCT)

Purpose

The maximum size, in KB, of character data that is exported to the bulk data file.

Valid Values

-1 | 0 | x

where

x

is an integer that specifies the number of KB.

Behavior

If set to -1, all character data, regardless of size, is written to the bulk data file, not to an external file.

If set to 0, all character data regardless of size, is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

If set to x , any character data exceeding this specified number of KB is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

Default

-1

GUI Tab

[Bulk tab](#)

Bulk Options

Attribute

BulkLoadOptions (BLO)

Purpose

Toggles options for the bulk load process.

This option only applies to connections to Oracle 11g R2 and higher database servers.

Valid Values

0 | x

where:

x

is a positive integer representing the cumulative total of the Bulk Options values.

Behavior

If set to 0, none of the options for bulk load are enabled.

If set to x, the values represented by x are enabled.

Currently, the only bulk load option available is:

No Index Errors - The driver stops a bulk load operation when a value that would cause an index to be invalidated is loaded. For example, if a value is loaded that violates a unique or non-null constraint, the driver stops the bulk load operation and discards all data being loaded, including any data that was loaded prior to the problem value. If not enabled, the bulk load operation continues even if a value that would cause an index to be invalidated is loaded. Value=128.

Notes

- The cumulative value of the options is only used in a connection string with the connection string attribute, BulkLoadOptions. On the Bulk tab of the driver Setup dialog, the individual options are enabled by selecting the appropriate check box.

Default

0

GUI Tab

[Bulk tab](#)

Cached Cursor Limit

Attribute

CachedCursorLimit (CCL)

Purpose

Specifies the number of Oracle Cursor Identifiers that the driver stores in cache. A Cursor Identifier is needed for each concurrent open Select statement. When a Select statement is closed, the driver stores the identifier in its cache, up to the limit specified, rather than closing the Cursor Identifier. When a new Cursor Identifier is needed, the driver takes one from its cache, if one is available. Cached Cursor Identifiers are closed when the connection is closed.

Valid Values

An integer from 0 to 65535

Default

32

GUI Tab

[Performance tab](#)

See also

[Performance Considerations](#) on page 367

Cached Description Limit

Attribute

CachedDescriptionLimit (CDL)

Purpose

Specifies the number of descriptions that the driver saves for Select statements. These descriptions include the number of columns, data type, length, and scale for each column. The matching is done by an exact-text match through the FROM clause.

Valid Values

An integer from 0 to 65535

Notes

- If the Select statement contains a Union or a nested Select, the description is not cached.

Default

0

GUI Tab

[Performance tab](#)

See also

[Performance Considerations](#) on page 367

Catalog Functions Include Synonyms

Attribute

CatalogIncludesSynonyms (CIS)

Purpose

Determines whether synonyms are included in calls to SQLProcedures, SQLStatistics, and SQLProcedureColumns.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), synonyms are included in calls to SQLProcedures, SQLStatistics, and SQLProcedureColumns.

If set to 0 (Disabled), synonyms are excluded (a non-standard behavior) and performance is thereby improved.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Performance tab](#)

See also

[Performance Considerations](#) on page 367

Catalog Options

Attribute

CatalogOptions (CO)

Purpose

Determines whether SQL_NULL_DATA is returned for the result columns REMARKS and COLUMN_DEF.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the result column REMARKS (for the catalog functions SQLTables and SQLColumns) and the result column COLUMN_DEF (for the catalog function SQLColumns) return actual values. Enabling this option reduces the performance of your catalog (SQLColumns and SQLTables) queries.

If set to 0 (Disabled), SQL_NULL_DATA is returned for the result columns REMARKS and COLUMN_DEF.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 367

Client Host Name

Attribute

ClientHostName (CHN)

Purpose

The host name of the client machine to be stored in the database. This value sets the MACHINE value in the V\$SESSION table on the server. This value is used by the client information feature.

Valid Values

string

where:

string

is the host name of the client machine.

If a value for this option is not specified, the driver uses the current machine name and IP address in the following format:

machine_name/IP_address

Notes

- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 367

Client ID

Attribute

ClientID (CID)

Purpose

Additional information about the client to be stored in the database. This value sets the CLIENT_IDENTIFIER value in the V\$SESSION table on the server. This value is used by the client information feature.

This option only applies to connections to Oracle 10g R2 and higher database servers.

Valid Values

string

where:

string

is additional information about the client.

Notes

- You can also specify this information using the Oracle DBMS_SESSION.SETIDENTIFIER procedure or the DBMS_APPLICATION_INFO.SET_CLIENT_INFO procedure.
- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 367

Client User

Attribute

ClientUser (CU)

Purpose

The user ID to be stored in the database. This value sets the OSUSER value in the V\$SESSION table on the server. This value is used by the client information feature.

Valid Values

-1 | *string*

where:

string

is a valid user ID.

Behavior

When set to -1, the driver uses the userid of the user that is currently logged onto the client.

If a value for this option is not specified, the driver uses name of the user that is currently logged into the OS.

Notes

- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 367

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- The application must be thread-enabled to use connection pooling.
- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 367

Connection Reset**Attribute**

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 367

Connection Retry Count**Attribute**

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x , the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay (CRD)

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to x , the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (EncryptionMethod=1). When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, the connection fails and the driver returns an error.

Valid Values

```
cryptographic_protocol [, cryptographic_protocol ]...
```

where:

```
cryptographic_protocol
```

is one of the following cryptographic protocols:

```
TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2
```

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Default

```
TLSv1.2, TLSv1.1, TLSv1
```

GUI Tab

[Security tab](#)

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\  
Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the CryptoLibName and SSLibName options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLibName](#) on page 360

Data Integrity Level

Attribute

DataIntegrityLevel (DIL)

Purpose

Specifies a preference for the data integrity to be used on data sent between the driver and the database server. The connection fails if the database server does not have a compatible integrity algorithm. See [Oracle Advanced Security](#) on page 375 for more information.

Valid Values

Rejected | Accepted | Requested | Required

Behavior

If set to 0 - Rejected, a data integrity check on data sent between the driver and the database server is refused. The connection fails if the database server specifies REQUIRED.

If set to 1 - Accepted, a data integrity check can be made on data sent between the driver and the database server. Data integrity is used if the database server requests or requires it.

If set to 2 - Requested, the driver enables a data integrity check on data sent between the driver and the database server if the database server permits it.

If set to 3 - Required, a data integrity check must be performed on data sent between the driver and the database server. The connection fails if the database server specifies REJECTED.

Notes

- Consult your database administrator concerning the data integrity settings of your Oracle server.
- This connection option can affect performance.

Default

0 - Rejected

GUI Tab

[Advanced Security tab](#)

See also

[Performance Considerations](#) on page 367

Data Integrity Types

Attribute

DataIntegrityTypes (DIT)

Purpose

Determines the method the driver uses to protect against attacks that intercept and modify data being transmitted between the client and server. You can enable data integrity protection without enabling encryption. See [Oracle Advanced Security](#) on page 375 for more information.

Valid Values

MD5 | SHA1 | SHA1,MD5

If set to MD5, the driver uses the Message Digest 5 (MD5) algorithm.

If set to SHA1, the driver uses the Secure Hash Algorithm (SHA-1).

If multiple values are specified and Oracle Advanced Security data integrity is enabled using the Data Integrity Level option, the database server determines which algorithm is used based on how it is configured.

Notes

- This option has no effect if [Data Integrity Level](#) on page 325 is set to 0 - Rejected.
- Consult your database administrator concerning the data integrity settings of your Oracle server.
- This connection option can affect performance.

Default

SHA1,MD5

GUI Tab

[Advanced Security tab](#)

See also

[Performance Considerations](#) on page 367

Data Source Name

Attribute

DataSourceName (DSN)

Description

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Default Buffer Size for Long/LOB Columns (in Kb)

Attribute

DefaultLongDataBuffLen (DLDBL)

Purpose

The maximum length of data (in KB) the driver can fetch from long columns in a single round trip and the maximum length of data that the driver can send using the SQL_DATA_AT_EXEC parameter.

Valid Values

An integer in multiples of 1024

The value must be in multiples of 1024 (for example, 1024, 2048). You need to increase the default value if the total size of any Long data exceeds 1 MB. This value is multiplied by 1024 to determine the total maximum length of fetched data. For example, if you enter a value of 2048, the maximum length of data would be 1024 x 2048, or 2097152 (2 MB).

Notes

- This connection option can affect performance.

Default

1024

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 367

Describe at Prepare

Attribute

DescribeAtPrepare (DAP)

Purpose

Determines whether the driver describes the SQL statement at prepare time.

This connection option can affect performance.

Valid Values

0 | 1

If set to 1 (Enabled), the driver describes the SQL statement at prepare time.

If set to 0 (Disabled), the driver does not describe the SQL statement at prepare time.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 367

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Edition Name

Attribute

EditionName (EN)

Purpose

The name of the Oracle edition the driver uses when establishing a connection. Oracle 11g R2 and higher allows your database administrator to create multiple editions of schema objects so that your application can still use those objects while the database is being upgraded. This option is only valid for Oracle 11g R2 and higher databases and tells the driver which edition of the schema objects to use.

The driver uses the default edition in the following cases:

- When the specified edition is not a valid edition. The driver generates a warning indicating that it was unable to set the current edition to the specified edition.
- When the value for this option is not specified or is set to an empty string.

If failover is enabled using the Failover Mode connection option and a connection fails over to another database server, the driver connects to the alternate server using the same edition that was used for the failed connection. The driver does not track changes to the current edition made using the ALTER SESSION SQL statement.

Valid Values

string

where:

string

is the name of a valid Oracle edition.

Default

None

GUI Tab

[General tab](#)

Enable Bulk Load

Attribute

EnableBulkLoad (EBL)

Purpose

Specifies the bulk load method.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses the database bulk load protocol when an application executes an INSERT with multiple rows of parameter data. If the protocol cannot be used, the driver returns a warning.

If set to 0 (Disabled), the driver uses standard parameter arrays.

Default

0 (Disabled)

GUI Tab

[Bulk Tab](#)

See Also

[DataDirect Bulk Load](#) on page 376

Enable N-CHAR Support

Attribute

EnableNcharSupport (ENS)

Purpose

Determines whether the driver provides support for the N-types NCHAR, NVARCHAR2, and NCLOB. These types are described as SQL_WCHAR, SQL_WVARCHAR, and SQL_WLONGVARCHAR, and are returned as supported by SQLGetTypeInfo. In addition, the "normal" char types (char, varchar2, long, clob) are described as SQL_CHAR, SQL_VARCHAR, and SQL_LONGVARCHAR regardless of the character set on the Oracle server.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver provides support for the N-types NCHAR, NVARCHAR2, and NCLOB.

If set to 0 (Disabled), the driver does not provide support for the N-types NCHAR, NVARCHAR2, and NCLOB.

Notes

- Valid only on Oracle 9i and higher.

Default

0 (Disabled)

See also

See [Unicode Support](#) on page 373 for details.

GUI Tab

[Advanced tab](#)

Enable Scrollable Cursors

Attribute

EnableScrollableCursors (ESC)

Purpose

Determines whether scrollable cursors, both Keyset and Static, are enabled for the data source.

This connection option can affect performance.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), scrollable cursors are enabled for the data source.

If set to 0 (Disabled), scrollable cursors are not enabled.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Performance tab](#)

See also

[Performance Considerations](#) on page 367

Enable Server Result Cache

Attribute

EnableServerResultCache (ESRC)

Purpose

Determines whether the driver sets the RESULT_CACHE_MODE session parameter to FORCE.

This option only applies to connections to Oracle 11g or higher database servers that support server-side result set caching.

This connection option can affect performance.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver sets the RESULT_CACHE_MODE session parameter to FORCE.

If set to 0 (Disabled), the driver does not sets the RESULT_CACHE_MODE session parameter.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 367

Enable SQLDescribeParam

Attribute

EnableDescribeParam (EDP)

Purpose

Determines whether the driver supports the SQLDescribeParam function, which allows an application to describe parameters in SQL statements and in stored procedure calls.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver supports SQLDescribeParam. If using Microsoft Remote Data Objects (RDO) to access data, you must use this value.

If set to 0 (Disabled), the driver does not support SQLDescribeParam and returns the error: `unimplemented function`.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Enable Static Cursors for Long Data

Attribute

EnableStaticCursorsForLongData (ESCLD)

Purpose

Determines whether the driver supports Long columns when using a static cursor. Enabling this option causes a performance penalty at the time of execution when reading Long data.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver supports Long columns when using a static cursor.

If set to 0 (Disabled), the driver does not support Long columns when using a static cursor.

Notes

- You must enable this option if you want to persist a result set that contains Long data into an XML data file.
- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Performance tab](#)

See also

[Performance Considerations](#) on page 367

Enable Timestamp with Timezone

Attribute

EnableTimestampwithTimezone (ETWT)

Purpose

Determines whether the driver exposes timestamps with timezones to the application.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver exposes timestamps with timezones to the application. The driver issues an ALTER SESSION at connection time to modify NLS_TIMESTAMP_TZ_FORMAT. NLS_TIMESTAMP_TZ_FORMAT is changed to the ODBC definition of a timestamp literal with the addition of the timezone literal: `'YYYY-MM-DD HH24:MI:SSXFF TZR'`.

If set to 0 (Disabled), timestamps with timezones are not exposed to the application.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Encryption Level

Attribute

EncryptionLevel (EL)

Purpose

Specifies a preference on whether to use encryption on data being sent between the driver and the database server.

Valid Values

Rejected | Accepted | Requested | Required

Behavior

If set to 0 - Rejected, or if no match is found between the driver and server encryption types, data sent between the driver and the database server is not encrypted or decrypted. The connection fails if the database server specifies REQUIRED.

If set to 1 - Accepted, encryption is used on data sent between the driver and the database server if the database server requests or requires it.

If set to 2 - Requested, data sent between the driver and the database server is encrypted and decrypted if the database server permits it.

If set to 3 - Required, data sent between the driver and the database server must be encrypted and decrypted. The connection fails if the database server specifies REJECTED.

Notes

- Consult your database administrator concerning the data encryption settings of your Oracle server.
- This connection option can affect performance.

Default

0 - Rejected

GUI Tab

[Advanced Security tab](#)

See also

[Performance Considerations](#) on page 367

Encryption Method

Attribute

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server. If the specified encryption method is not supported by the database server, the connection fails and the driver returns an error.

Valid Values

0 | 1 | 3 | 4 | 5

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using the SSL protocols specified in the Crypto Protocol Version connection option.

If set to 3 (SSL3), the driver uses SSL3 data encryption.

If set to 4 (SSL2), the driver uses SSL2 data encryption.

If set to 5 (TLS1), the driver uses TLS1 data encryption.

Notes

- Consult your database administrator concerning the SSL settings of your Oracle server.
- This connection option can affect performance.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

See also

[Crypto Protocol Version](#) on page 323

[Performance Considerations](#) on page 367

Encryption Types

Attribute

EncryptionTypes (ET)

Purpose

Specifies a comma-separated list of the encryption algorithms to use if Oracle Advanced Security encryption is enabled using the Encryption Level connection property.

Valid Values

encryption_algorithm [, *encryption_algorithm*]...

where:

encryption_algorithm

is a encryption algorithm specifying an algorithm in the following table:

AES256 | RC4_256 | AES192 | 3DES168 | AES128 | RC4_128 | 3DES112 | RC4_56 | DES | RC4_40

Encryption Algorithm	Description
3DES112	Two-key Triple-DES (with an effective key size of 112-bit).
AES128	AES with a 128-bit key size.
AES192	AES with a 192-bit key size.

Encryption Algorithm	Description
AES256	AES with a 256-bit key size.
DES	DES (with an effective key size of 56-bit).
DES168	Three-key Triple-DES (with an effective key size of 168-bit).
RC4_128	RC4-128 with a 128-bit key size.
RC4_256	RC4 with a 256-bit key size.
RC4_40	RSA RC4 with a 40-bit key size.
RC4_56	RSA RC4 with a 56-bit key size.

Example

Your security environments specifies that you can use RC4 with a 256-bit key size, AES with a 192-bit key size, or two-key Triple-DES with an effective key size of 112-bit. Use the following values:

```
EncryptionTypes=RC4_256,AES192,3DES112
```

Notes

- This option is ignored if Encryption Level is set to 0 - Rejected.
- Consult your database administrator concerning the data encryption settings of your Oracle server.
- This connection option can affect performance.

Default

On the GUI tab: all check boxes are selected.

In the connection string: no encryption methods are specified. The driver sends a list of all of the encryption methods to the Oracle server.

GUI Tab

[Advanced Security tab](#)

See also

[Performance Considerations](#) on page 367

Failover Granularity

Attribute

FailoverGranularity (FG)

Purpose

Determines whether the driver fails the entire failover process or continues with the process if errors occur while trying to reestablish a lost connection.

This option applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select).

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 (Non-Atomic), the driver continues with the failover process and posts any errors on the statement on which they occur.

If set to 1 (Atomic) the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

If set to 2 (Atomic Including Repositioning), the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

If set to 3 (Disable Integrity Check), the driver does not verify that the rows that were restored during the failover process match the original rows. This value applies only when Failover Mode is set to 2 (Select).

Default

0 (Non-Atomic)

GUI Tab

[Failover tab](#)

Failover Mode

Attribute

FailoverMode (FM)

Purpose

Specifies the type of failover method the driver uses.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Connection), the driver provides failover protection for new connections only.

If set to 1 (Extended Connection), the driver provides failover protection for new and lost connections, but not any work in progress.

If set to 2 (Select), the driver provides failover protection for new and lost connections. In addition, it preserves the state of work performed by the last Select statement executed.

Default

0 (Connection)

GUI Tab

[Failover tab](#)

Failover Preconnect

Attribute

FailoverPreconnect (FP)

Description

Specifies whether the driver tries to connect to the primary and an alternate server at the same time.

This attribute applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select) and at least one alternate server is specified.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver tries to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection. This value provides the best performance, but your application typically experiences a short wait while the failover connection is attempted.

If set to 1 (Enabled), the driver tries to connect to the primary and an alternate server at the same time. This can be useful if your application is time-sensitive and cannot absorb the wait for the failover connection to succeed.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Fetch TSWTZ as Timestamp

Attribute

FetchTSWTZasTimestamp (FTSWTZAT)

Purpose

Determines whether the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_TYPE_TIMESTAMP or SQL_VARCHAR.

Valid on Oracle 10g R2 or higher.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the timestamp with time zone data type as the ODBC type SQL_TYPE_TIMESTAMP. The time zone information in the fetched value is truncated. Use this value if your application needs to process values the same way as TIMESTAMP columns.

If set to 0 (Disabled), the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_VARCHAR. Use this value if your application requires the time zone information in the fetched value.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Field Delimiter

Attribute

BulkLoadFieldDelimiter (BLFD)

Purpose

Specifies the character that the driver will use to delimit the field entries in a bulk load data file.

Valid Values

x

where:

x

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash(-), the colon(:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Field Delimiter character must be different from the Bulk Load Record Delimiter.

Default

None

GUI Tab

[Bulk tab](#)

GSS Client Library

Attribute

GSSClient (GSSC)

Purpose

The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC).
The driver uses the path defined by the PATH environment variable for loading the specified client library.

Valid Values

native | *client_library*

where:

client_library is a GSS client library installed on the client.

Behavior

If set to *client_library*, the driver uses the specified GSS client library.

If set to native, the driver uses the GSS client shipped with the operating system.

Default

native

GUI Tab

[Security tab](#)

Host

Attribute

HostName (HOST)

Purpose

The name or the IP address of the server to which you want to connect.

Valid Values

server_name | *IP_address*

where:

server_name

is the name of the server to which you want to connect.

IP_address

is the IP address of the server to which you want to connect.

The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details about these formats.

Notes

- This option is mutually exclusive with the Server Name and TNSNames File options.

Default

None

GUI Tab

[General tab](#)

Host Name In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

host_name | *#SERVERNAME#*

where

host_name

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If set to a host name, the driver examines the subjectAltName values included in the certificate. If a dnsName value is present in the subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the dnsName value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the dnsName value.

If no subjectAltName values exist or a dnsName value is not in the list of subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the commonName part of the Subject name in the certificate. The commonName typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the commonName. If multiple commonName parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the commonName parts.

If set to *#SERVERNAME#*, the driver compares the host server name specified as part of a data source or connection string to the dnsName or the commonName value.

Default

None

GUI Tab

[Security tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Initialization String

Attribute

InitializationString (IS)

Purpose

A SQL command that is issued immediately after connecting to the database to manage session settings.

Valid Values

SQL_command

where:

SQL_command

is a valid SQL command that is supported by the database.

Example

To set the date format on every connection, specify:

```
Initialization String=ALTER SESSION SET DATE_FORMAT = 'DD/MM/YYYY'
```

Notes

- If the statement fails to execute, the connection fails and the driver reports the error returned from the server.

Default

None

GUI Tab

[Advanced tab](#)

Key Password

Attribute

KeyPassword (KP)

Purpose

The password used to access the individual keys in the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. Keys stored in a keystore can be individually password-protected. To extract the key from the keystore, the driver must have the password of the key.

Valid Values

key_password

where:

key_password

is the password of a key in the keystore.

Default

None

GUI Tab

[Security tab](#)

Key Store

Attribute

Keystore (KS)

Purpose

The name of the directory containing the keystore file to be used when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request. If you do not specify a directory, the current directory is used.

Valid Values

`keystore_directory`

where:

`keystore_directory`

is the location of the keystore file.

Notes

- The keystore and truststore files can be the same file.

Default

None

GUI Tab

[Security tab](#)

Key Store Password

Attribute

KeystorePassword (KSP)

Purpose

The password used to access the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request.

Valid Values

`keystore_password`

where:

keystore_password

is the password of the keystore file.

Notes

- The keystore and truststore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Load Balance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

Specifies the number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Valid Values

0 | x

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to x , inactive connections are closed after the specified number of seconds passes.

Notes

- The Min Pool Size option may cause some connections to ignore this value.

Default

0

GUI Tab

[Pooling tab](#)

Load Balancing

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Local Timezone Offset

Attribute

LocalTimezoneOffset (LTZO)

Purpose

A value to alter local time zone information. The default is "" (empty string), which means that the driver determines local time zone information from the operating system. If it is not available from the operating system, the driver defaults to using the setting on the Oracle server.

Valid Values

Valid values are specified as offsets from GMT as follows: $(-)HH:MM$. For example, $-08:00$ equals GMT minus 8 hours.

The driver uses the value of this option to issue an ALTER SESSION for local time zone at connection time.

Default

"" (empty string)

GUI Tab

[Advanced tab](#)

Lock Timeout

Attribute

LockTimeout (LTO)

Purpose

Specifies the amount of time, in seconds, the Oracle server waits for a lock to be released before generating an error when processing a Select...For Update statement on an Oracle 9i or higher server.

This connection option can affect performance.

Valid Values

-1 | 0 | *x*

where:

x

is an integer that specifies a number of seconds.

Behavior

If set to -1, the server waits indefinitely for the lock to be released.

If set to 0, the server generates an error immediately and does not wait for the lock to time out.

If set to *x*, the server waits for the specified number of seconds for the lock to be released.

Notes

- If you are connected to an Oracle 8i server, any value greater than 0 is equivalent to the value -1.
- This connection option can affect performance.

Default

-1

GUI Tab

[Performance tab](#)

See also

[Performance Considerations](#) on page 367

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x, the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

15

GUI Tab

[Advanced tab](#)

Max Pool Size

Attribute

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

Notes

- This connection option can affect performance.

Example

If set to 20, the maximum number of connections allowed in the pool is 20.

Default

100

GUI Tab[Pooling tab](#)**See also**[Performance Considerations](#) on page 367**Min Pool Size****Attribute**

MinPoolSize (MNPS)

Purpose

The minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Valid Values0 | x

where:

 x

is an integer from 1 to 65535.

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

If set to x , the start-up number of connections in the pool is 5 in addition to the current existing connection.

Notes

- This connection option can affect performance.

Example

If set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Default

0

GUI Tab[Pooling tab](#)**See also**[Performance Considerations](#) on page 367

Module

Attribute

Module (MOD)

Purpose

Provides additional information about the client to be stored in the database. This value sets the CLIENT_IDENTIFIER value in the V\$SESSION table on the server. This value is used by the client information feature.

This option only applies to connections to Oracle 10g R2 and higher database servers.

Valid Values

string

where:

string

is a the name of a stored procedure or the name of the application.

Notes

- If a value is not specified for this option, the driver uses the PROGRAM value in the V\$SESSION table.
- You can also specify this information using the Oracle DBMS_SESSION.SETIDENTIFIER procedure or the DBMS_APPLICATION_INFO.SET_CLIENT_INFO procedure.
- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 367

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Description

The port number of the server listener.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your database administrator for the correct number.

Notes

- This option is mutually exclusive with the Server Name and TNSNames File options.

Default

None

GUI Tab

[General tab](#)

PRNGSeedFile

Attribute

PRNGSeedFile (PSF)

Purpose

UNIX[®] Specifies the absolute path for the entropy-source file or device used as a seed for SSL key generation.

Valid Values

string | RANDFILE

where:

string

is the absolute path for the entropy-source file or device that seeds the random number generator used for TLS/SSL key generation.

Behavior

If set to *string*, the specified entropy-source file or device seeds the random number generator used for TLS/SSL key generation. Entropy levels and behavior may vary for different files and devices. See the following section for a list of commonly used entropy sources and their behavior.

If set to RANDFILE, the `RAND_file_name()` function in your application generates a default path for the random seed file. The seed file is `$RANDFILE` if that environment variable is set; otherwise, it is `$HOME/.rnd`. If `$HOME` is not set either, an error occurs.

Common Valid Values

Although other entropy-source files may be specified, the following valid values are for files and devices that are commonly used for seeding:

`/dev/random`

is a pseudorandom number generator (blocking) that creates a seed from random bits of environmental noise it collects in an entropy pool. When there is insufficient noise in the pool, the file blocks calls until enough noise is collected. This provides more secure SSL key generation, but at the expense of blocked calls.

`/dev/urandom`

is a pseudorandom number generator (non-blocking) that creates seeds from random bits from environmental noise it collects in an entropy pool. When there is insufficient noise in the pool, the file reuses bits from the pool instead of blocking calls. This eliminates potential delays associated with blocked calls, but may result in less secure TLS/SSL key generation.

`/dev/hwrng`

is a hardware random number generator. The behavior is dependent on the device used in your environment.

Notes

- This option is ignored when TLS/SSL is disabled (`EncryptionMethod=0`) or the seed source is set to Poll Only (`PRNGSeedSource=1`).
- For processes that employ multiple TLS/SSL-enabled drivers, the behavior of this option for all drivers is determined by the values specified for the driver that first connects to the process and loads the OpenSSL library. Since the OpenSSL library loads only once per process, the values specified for drivers that subsequently connect are ignored. To ensure that the correct security settings are used, we recommend configuring this option identically for all drivers used in a process.

Default

`/dev/random`

GUI tab

NA

See also

[PRNGSeedSource](#) on page 353

PRNGSeedSource

Attribute

PRNGSeedSource (PSS)

Purpose

UNIX[®] Specifies the source of the seed the driver uses for TLS/SSL key generation. Seeds are a pseudorandom or random value used to set the initial state of the random number generator used to generate TLS/SSL keys. Using seeds with a higher level of entropy, or randomness, provides a more secure transmission of data encrypted using TLS/SSL.

Valid Values

0 | 1

Behavior

If set to 0 (File), the driver uses entropy-source file or device specified in the `PRNGSeedFile` connection option as the seed used for TLS/SSL key generation.

If set to 1 (Poll Only) , the driver uses the `RAND_poll` function in TLS/SSL to create the seed used for TLS/SSL key generation.

Notes

- For processes that employ multiple TLS/SSL-enabled drivers, the behavior of this option for all drivers is determined by the values specified for the driver that first connects to the process and loads the OpenSSL library. Since the OpenSSL library loads only once per process, the values specified for drivers that subsequently connect are ignored. To ensure that the correct security settings are used, we recommend configuring this option identically for all drivers used in a process.
- This option is ignored when TLS/SSL is disabled (`EncryptionMethod=0`)

Default

0 (File)

GUI Tab

NA

See also

[PRNGSeedFile](#) on page 351

Procedure Returns Results

Attribute

ProcedureRetResults (PRR)

Purpose

Determines whether the driver returns result sets from stored procedures/functions.

See [Support of Materialized Views](#) on page 377 for details.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns result sets from stored procedures/functions. When set to 1 and you execute a stored procedure that does not return result sets, you will incur a small performance penalty.

If set to 0 (Disabled), the driver does not return result sets from stored procedures.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 367

Program ID

Attribute

ProgramID (PID)

Purpose

The product and version information of the driver on the client to be stored in the database. This value sets the PROCESS value in the V\$SESSION table on the server. This value is used by the client information feature.

Valid Values

string

where:

string

is a value that identifies the product and version of the driver on the client.

If a value for this option is not specified, the driver uses the process ID of the session.

Notes

- This connection option can affect performance.

Default

None

GUI Tab

[Client Monitoring tab](#)

See also

[Performance Considerations](#) on page 367

Query Timeout

Attribute

QueryTimeout (QT)

Description

The number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the SQL_ATTR_QUERY_TIMEOUT statement attribute on the SQLSetStmtAttr() function.

Valid Values

where:

x

is a number of seconds.

Behavior

If set to -1, the query does not time out. The driver silently ignores the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to 0, the query does not time out, but the driver responds to the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to `x`, all queries time out after the specified number of seconds unless the application overrides this value by setting the `SQL_ATTR_QUERY_TIMEOUT` attribute.

Default

0

GUI Tab

[Advanced tab](#)

Record Delimiter

Attribute

BulkLoadRecordDelimiter (BLRD)

Purpose

Specifies the character that the driver will use to delimit the record entries in a bulk load data file.

Valid Values

`x`

where:

`x`

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash(-), the colon(:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Record Delimiter character must be different from the Bulk Load Field Delimiter.

None

GUI Tab

[Bulk tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where `x` is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Advanced tab](#)

Report Recycle Bin

Attribute

ReportRecycleBin (RRB)

Purpose

Determines whether support is provided for reporting objects that are in the Oracle Recycle Bin.

On Oracle 10g R1 and higher, when a table is dropped, it is not actually removed from the database, but placed in the recycle bin instead.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), support is provided for reporting objects that are in the Oracle Recycle Bin.

If set to 0 (Disabled), the driver does not return tables contained in the recycle bin in the result sets returned from `SQLTables` and `SQLColumns`. Functionally, this means that the driver filters out any results whose Table name begins with `BIN$`.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Server Name

Attribute

ServerName (SRVR)

Purpose

Specifies a net service name that exists in the TNSNAMES.ORA file. The corresponding net service name entry in the TNSNAMES.ORA file is used to obtain Host, Port Number, and Service Name or SID information.

Valid Values

server_name

where:

server_name

is a net service name in the TNSNAMES.ORA file.

Notes

- This option is mutually exclusive with the Host, Port Number, SID, and Service Name options.

Default

None

GUI Tab

[General tab](#)

Server Process Type

Attribute

ServerType (ST)

Purpose

Determines whether the connection is established using a shared or dedicated server process (dedicated thread on Windows).

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Server Default), the driver uses the default server process set on the server.

If set to 1 (Shared), the server process used is retrieved from a pool. The socket connection between the application and server is made to a dispatcher process on the server. This setting allows there to be fewer processes than the number of connections, reducing the need for server resources. Use this value when a server must handle a large number of connections.

If set to 2 (Dedicated), a server process is created to service only that connection. When that connection ends, so does the process (UNIX and Linux) or thread (Windows). The socket connection is made directly between the application and the dedicated server process or thread. When connecting to UNIX and Linux servers, a dedicated server process can provide significant performance improvement, but uses more resources on the server. When connecting to Windows servers, the server resource penalty is insignificant. Use this value if you have a batch environment with a low number of connections.

Notes

- The server must be configured for shared connections (the SHARED_SERVERS initialization parameter on the server has a value greater than 0) for the driver to be able to specify the shared server process type.
- This connection option can affect performance.

Default

0 (Server Default)

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 367

Service Name

Attribute

ServiceName (SN)

Purpose

The Oracle service name that specifies the database used for the connection. The service name is a string that is the global database name—a name that is comprised of the database name and domain name, for example:

```
sales.us.acme.com
```

The service name is included as part of the Oracle connect descriptor, which is a description of the destination for a network connection. The service name is specified in the CONNECT_DATA parameter of the connect descriptor, for example:

```
(CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com))
```

In this example, you would specify `sales.us.acme.com` as the value for the Service Name connection option.

Valid Values

service_name

where:

service_name

is the description of the destination for a network connection.

Notes

- This option is mutually exclusive with the SID, Server Name, and TNSNames File options.

Default

None

GUI Tab

[General tab](#)

SID

Attribute

SID (SID)

Purpose

The Oracle System Identifier that refers to the instance of Oracle running on the server.

Valid Values

sid

where:

sid

is the name of the Oracle System Identifier.

Notes

- This option is mutually exclusive with the Service Name, Server Name, and TNSNames File options.

Default

None

GUI Tab

[General tab](#)

SSLibName

Attribute

SSLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\
Drivers\OpenSSL\1.0.0r\ddssl27.dll; (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
SSLibName=C:\Program Files\Progress\DataDirect\
Connect64_for_ODBC_71\drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 323

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Timestamp Escape Mapping

Attribute

TimestampEscapeMapping (TEM)

Purpose

Determines how the driver maps Date, Time, and Timestamp literals.

Valid Values

0 | 1

Behavior

If set to 0 (Oracle Version Specific), the driver determines whether to use the TO_DATE or TO_TIMESTAMP function based on the version of the Oracle server to which it is connected. If the driver is connected to an 8.x server, it maps the Date, Time, and Timestamp literals to the TO_DATE function. If the driver is connected to a 9.x or higher server, it maps these escapes to the TO_TIMESTAMP function.

If set to 1 (Oracle 8x Compatible), the driver always uses the Oracle 8.x TO_DATE function as if connected to an Oracle 8.x server.

Default

0 (Oracle Version Specific)

GUI Tab

[Advanced tab](#)

TNSNames File

Attribute

TNSNamesFile (TNF)

Purpose

Specifies the name of the TNSNAMES.ORA file. In a TNSNAMES.ORA file, connection information for Oracle services is associated with an Oracle net service name. The entry in the TNSNAMES.ORA file specifies Host, Port Number, and Service Name or SID.

TNSNames File is ignored if no value is specified in the Server Name option. If the Server Name option is specified but the TNSNames File option is left blank, the TNS_ADMIN environment setting is used for the TNSNAMES.ORA file path. If there is no TNS_ADMIN setting, the ORACLE_HOME environment setting is used. On Windows, if ORACLE_HOME is not set, the path is taken from the Oracle section of the Registry.

Using an Oracle TNSNAMES.ORA file to centralize connection information in your Oracle environment simplifies maintenance when changes occur. If, however, the TNSNAMES.ORA file is unavailable, then it is useful to be able to open a backup version of the TNSNAMES.ORA file (TNSNames file failover). You can specify one or more backup, or alternate, TNSNAMES.ORA files.

Valid Values

path_filename

where:

path_filename

is the entire path, including the file name, to the TNSNAMES.ORA file.

Behavior

To specify multiple TNSNAMES.ORA file locations, separate the names with a comma and enclose the locations in parentheses (you do not need parentheses for a single entry). For example:

```
(F:\server2\oracle\tnsnames.ora, C:\oracle\product\10.1\db_1\network\admin\tnsnames.ora)
```

The driver tries to open the first file in the list. If that file is not available, then it tries to open the second file in the list, and so on.

[Connection Retry Count](#) on page 321 and [Connection Retry Delay](#) on page 322 are also valid with TNSNames failover. The driver makes at least one attempt to open the files, and, if Connection Retry Count is enabled, more than one. If Connection Retry Delay is enabled, the driver waits the specified number of seconds between attempts. Load Balancing is not available for TNSNames failover.

Notes

- This option is mutually exclusive with the Host, Port Number, SID, and Service Name options.

Default

None

GUI Tab

[General tab](#)

Truststore

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used.

Valid Values

truststore_directory \ filename

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Notes

- The truststore and keystore files may be the same file.

Default

None

GUI Tab

[Security tab](#)

Truststore Password

TruststorePassword (TSP)

Purpose

Specifies the password that is used to access the truststore file when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

- The truststore and keystore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Use Current Schema for SQLProcedures

Attribute

UseCurrentSchema (UCS)

Description

Determines whether the driver returns only procedures owned by the current user when executing SQLProcedures.

Valid Values

0 | 1

Behavior

When set to 1 (Enabled), the call for SQLProcedures is optimized, but only procedures owned by the user are returned.

When set to 0 (Disabled), the driver does not specify only the current user.

Default

1 (Enabled)

GUI Tab

[Performance tab](#)

See also

[Performance Considerations](#) on page 367

User Name

Attribute

LogonID (UID)

Description

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

You can also use OS Authentication to connect to your Oracle database. See [OS Authentication](#) on page 376 for details.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Security tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Notes

- Truststore information is specified using the TrustStore and TrustStorePassword options.

Default

1 (Enabled)

GUI Tab

[Security tab](#)

Wire Protocol Mode**Attribute**

WireProtocolMode (WPM)

Description

Specifies whether the driver optimizes network traffic to the Oracle server.

Valid Values

1 | 2

Behavior

If set to 1, the driver does not optimize network traffic to the Oracle server.

If set to 2, the driver optimizes network traffic to the Oracle server for result sets that contain repeating data in some or all of the columns, and the repeating data is in consecutive rows. It also optimizes network traffic if the application is updating or inserting images, pictures, or long text or binary data.

Notes

- This connection option can affect performance.

Default

2

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 367 for details.

Performance Considerations

The following connection options can enhance driver performance.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Array Size (ArraySize): If this connection string attribute is set appropriately, the driver can improve performance of your application by reducing the number of round trips on the network. For example, if your application normally retrieves 200 rows, it is more efficient for the driver to retrieve 200 rows at one time over the network than to retrieve 50 rows at a time during four round trips over the network.

Cached Cursor Limit (CachedCursorLimit): To improve performance when your application executes concurrent Select statements, Cursor Identifiers can be cached. In this case, the Cursor Identifier is retrieved from a cache rather than being created for each connection. When an Identifier is needed, the driver takes one from its cache, if one is available, rather than creating a new one. Cached Cursor Identifiers are closed when the connection is closed. To cache Cursor Identifiers, the CachedCursorLimit attribute must be set to the appropriate number of concurrent open Select statements.

Cached Description Limit (CachedDescLimit): The driver can cache descriptions of Select statements and improve the performance of your ODBC application; therefore, if your application issues a fixed set of SQL queries throughout the life of the application, the description of the query should be cached. If a description is not cached, the description must be retrieved from the server, which reduces performance. The descriptions include the number of columns and the data type, length, and scale for each column. The matching is done by an exact-text match through the From clause. If the statement contains a Union or a subquery, the driver cannot cache the description.

Catalog Functions Include Synonyms (CatalogIncludesSynonyms): Standard ODBC behavior is to include synonyms in the result set of calls to the following catalog functions: SQLProcedures, SQLStatistics and SQLProcedureColumns. Retrieving this synonym information degrades performance. If your ODBC application does not need to return synonyms when using these catalog functions, the driver can improve performance if the CatalogIncludesSynonyms attribute is disabled (set to 0).

Catalog Options (CatalogOptions): If your application does not need to access the comments/remarks for database tables, performance of your application can be improved. In this case, the CatalogOptions attribute should be disabled (set to 0) because retrieving comments/remarks degrades performance. If this attribute is enabled (set to 1), result column REMARKS (for the catalog functions SQLTables and SQLColumns) and the result column COLUMN_DEF (for the catalog function SQLColumns) return actual values.

Client Information: The client information feature automatically adjusts server resources, such as CPU and memory, based on the service class associated with a workload. Therefore, an application's performance is tied to the workload to which it is assigned and, ultimately, to the service class associated with that workload. The Oracle Wire Protocol driver allows your application to set client information in the Oracle database that can be used by the client information feature to classify work. If you know that your database environment can use client information, coordinate with your database administrator to determine how setting the following options affects performance.

- **Accounting Info (AccountingInfo):** Sets the CLIENT_INFO value of the V\$SESSION table on the server.
- **Action (Action):** Sets ACTION column of the V\$SESSION table on the server.
- **Application Name (ApplicationName):** Sets the dbms_session value in the database and the PROGRAM value of the V\$SESSION table on the server.
- **Client Host Name (ClientHostName):** Sets the MACHINE value in the V\$SESSION table on the server.
- **Client ID (ClientID):** Sets the CLIENT_IDENTIFIER value in the V\$SESSION table on the server.
- **Client User (ClientUser):** Sets the OSUSER value in the V\$SESSION table on the server.

- **Module (Module):** Sets the CLIENT_IDENTIFIER value in the V\$SESSION table on the server.
- **Program ID (ProgramID):** Sets the PROCESS value in the V\$SESSION table on the server.

Connection Pooling (Pooling): If you enable the driver to use connection pooling, you can set additional options that affect performance:

- **Load Balance Timeout (LoadBalanceTimeout):** You can define how long to keep connections in the pool. The time that a connection was last used is compared to the current time and, if the timespan exceeds the value of the Load Balance Timeout option, the connection is destroyed. The Min Pool Size option can cause some connections to ignore this value.
- **Connection Reset (ConnectionReset):** Resetting a re-used connection to the initial configuration settings impacts performance negatively because the connection must issue additional commands to the server.
- **Max Pool Size (MaxPoolSize):** Setting the maximum number of connections that the pool can contain too low might cause delays while waiting for a connection to become available. Setting the number too high wastes resources.
- **Min Pool Size (MinPoolSize):** A connection pool is created when the first connection with a unique connection string connects to the database. The pool is populated with connections up to the minimum pool size, if one has been specified. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Data Integrity Level (DataIntegrityLevel) and Data Integrity Types (DataIntegrityTypes): Checking data integrity may adversely reduce performance because of the additional overhead (mainly CPU usage) that is required to perform the check.

Default Buffer Size for Long/LOB Columns (DefaultLongDataBuffLen): To improve performance when your application fetches images, pictures, or long text or binary data, a buffer size can be set to accommodate the maximum size of the data. The buffer size should only be large enough to accommodate the maximum amount of data retrieved; otherwise, performance is reduced by transferring large amounts of data into an oversized buffer. If your application retrieves more than 1 MB of data, the buffer size should be increased accordingly.

Describe At Prepare (DescribeAtPrepare): When enabled, this option requires extra network traffic. If your application does not require result set information at prepare time (for instance, you request information about the result set using SQLColAttribute(s), SQLDescribeCol, SQLNumResultCols, and so forth, before calling SQLExecute on a prepared statement), you can increase performance by disabling this option.

Enable Bulk Load (EnableBulkLoad): If your application performs bulk loading of data, you can improve performance by configuring the driver to use the database system's bulk load functionality instead of database array binding. The trade-off to consider for improved performance is that using the bulk load functionality can bypass data integrity constraints.

EnableServerResultCache (EnableServerResultCache): If your application connects to Oracle 11g and executes the same query multiple times, you can improve performance by using the Oracle feature server-side resultset caching. When enabled, Oracle stores the result set in database memory. On subsequent executions of the same query, the result set is returned from database memory if the underlying tables have not been modified. Without result set caching, the server would process the query and formulate a new result set.

Enable Scrollable Cursors (EnableScrollableCursors) and Enable Static Cursors for Long Data (EnableStaticCursorsForLongData): When your application uses Static or Keyset (Scrollable) cursors, the EnableScrollableCursors attribute must be enabled (set to 1). Also, if your application retrieves images, pictures, long text or binary data while using Static cursors, the EnableStaticCursorsForLongData attribute must be enabled (set to 1). However, this can degrade performance when retrieving long data with Static cursors as the entire result set is stored on the client. To improve performance, you might consider designing your application to retrieve long data through forward-only cursors.

Encryption Method (EncryptionMethod), Encryption Level (EncryptionLevel), and Encryption Types (EncryptionTypes): Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) required to encrypt and decrypt data. Using data encryption can degrade performance more than performing data integrity checks.

Failover Mode (FailoverMode): Although high availability that replays queries after a failure provides increased levels of protection, it can adversely affect performance because of increased overhead.

Lock Timeout (LockTimeOut): Sometimes users attempt to select data that is locked by another user. Oracle provides three options when accessing locked data with SELECT ... FOR UPDATE statements:

- Wait indefinitely for the lock to be released (-1)
- Return an error immediately (0)
- Return an error if the lock has not been released within a specific number of seconds (*n* seconds)

Note: This option is not available with Oracle 8.

Some applications may benefit by not waiting indefinitely and continuing execution; this keeps the application from hanging. The application, however, needs to handle lock timeouts properly with an appropriate timeout value; otherwise, processing time could be wasted handling lock timeouts, and deadlocks could go undetected.

To improve performance, either enter a number of seconds or enter 0 as the value for this option.

Procedure Returns Results (ProcedureRetResults): The driver can be tuned for improved performance if your application's stored procedures do not return results. In this case, the ProcedureRetResults attribute should be disabled (set to 0).

Server Process Type (ServerType): When using a dedicated server connection, a server process on UNIX (a thread on Windows) is created to serve only your application connection. When you disconnect, the process goes away. The socket connection is made directly between your application and this dedicated server process. This can provide tremendous performance improvements, but will use significantly more resources on UNIX servers. Because this is a thread on Oracle servers running on Windows platforms, the additional resource usage on the server is significantly less. This option should be set to 2 (dedicated) when you have a batch environment with lower numbers of connections, your Oracle server has excess processing capacity and memory available when at maximum load, or if you have a performance-sensitive application that would be degraded by sharing Oracle resources with other applications.

Use Current Schema for Catalog Functions (UseCurrentSchema): If your application needs to access database objects owned only by the current user, then performance can be improved. In this case, the Use Current Schema for Catalog Functions option must be enabled. When this option is enabled, the driver returns only database objects owned by the current user when executing catalog functions. Calls to catalog functions are optimized by grouping queries. Enabling this option is equivalent to passing the Logon ID used on the connection as the SchemaName argument to the catalog functions.

Data Types

The following table shows how the Oracle data types are mapped to the standard ODBC data types. [Unicode Support](#) on page 373 lists Oracle to Unicode data type mappings.

Table 26: Oracle Data Types

Oracle	ODBC
BFILE ¹⁵	SQL_LONGVARBINARY
BINARY DOUBLE ¹⁶	SQL_REAL
BINARY FLOAT ¹⁶	SQL_DOUBLE
BLOB ^{16, 17}	SQL_LONGVARBINARY
CHAR ¹⁸	SQL_CHAR
CLOB ^{16, 17, 19}	SQL_LONGVARCHAR
DATE	SQL_TYPE_TIMESTAMP
LONG ²⁰	SQL_LONGVARCHAR
LONG RAW	SQL_LONGVARBINARY
NCHAR ²¹	SQL_WVARCHAR
NCLOB ²¹	SQL_WLONGVARCHAR
NVARCHAR2 ²¹	SQL_WVARCHAR
NUMBER	SQL_DOUBLE
NUMBER (p,s)	SQL_DECIMAL
RAW	SQL_VARBINARY
TIMESTAMP ²²	SQL_TIMESTAMP
TIMESTAMP WITH LOCAL TIMEZONE ^{22, 23}	SQL_TIMESTAMP
TIMESTAMP WITH TIMEZONE ^{22, 23}	SQL_VARCHAR

¹⁵ Read-Only

¹⁶ Supported only on Oracle 10g and higher.

¹⁷ Supported in basic file and SecureFiles storage.

¹⁸ If the database character set is set to UTF-8, the Oracle driver maps the CHAR data type to SQL_WCHAR.

¹⁹ If the database character set is set to UTF-8, the Oracle driver maps the CLOB data type to SQL_WLONGVARCHAR.

²⁰ If the database character set is set to UTF-8, the Oracle driver maps the LONG data type to SQL_WLONGVARCHAR.

²¹ Supported only when the EnableNcharSupport connection option is enabled.

²² Supported only on Oracle 9i and higher.

²³ Timestamp with timezone mapping changes based on the setting of the Fetch TSWTZ as Timestamp option only on Oracle 10g R2 and higher.

Oracle	ODBC
VARCHAR2 ²⁴	SQL_VARCHAR
XMLType ²⁵	SQL_LONGVARCHAR

The Oracle Wire Protocol driver does not support any object types (also known as abstract data types). When the driver encounters an object type during data retrieval, it returns an Unknown Data Type error (SQL State HY000).

See [Retrieving Data Type Information](#) on page 72 for more information about data types.

XMLType

The driver supports tables containing columns whose data type is specified as XMLType, except those with binary or object relational storage.

When inserting or updating XMLType columns, the data to be inserted or updated must be in the form of an XMLType data type. The database provides functions to construct XMLType data. The `xmlData` argument to `xmltype()` may be specified as a string literal.

Examples

If the XMLType column is created with the CLOB storage type, then the driver returns it without use of the special `getClobVal` function, that is, you can use:

```
SELECT XML_col FROM table_name...
```

instead of

```
SELECT XML_col.getClobVal()...
```

The following example illustrates using the CLOB storage type:

```
CREATE TABLE po_xml_tab(
  poId NUMBER(10),
  poDoc XMLTYPE
)
XMLType COLUMN poDoc
  STORE AS CLOB (
    TABLESPACE lob_seg_ts
    STORAGE (INITIAL 4096 NEXT 4096)
    CHUNK 4096 NOCACHE LOGGING
  );
```

The next example illustrates how to create a table, insert data, and retrieve data when not using the CLOB storage type:

```
CREATE TABLE PURCHASEORDER (PODOCUMENT sys.XMLTYPE);
```

²⁴ If the database character set is set to UTF-8, the Oracle driver maps the VARCHAR2 data type to SQL_WVARCHAR.

²⁵ XMLType columns with binary or object relational storage are not supported.

The PURCHASEORDER table contains one column—PODOCUMENT—with a data type of XMLType (`sys.XMLTYPE`). The next step is to insert one purchase order, created by the static function `sys.XMLTYPE.createXML`:

```
INSERT INTO PURCHASEORDER (PODOCUMENT) values (
sys.XMLTYPE.createXML(
'<PurchaseOrder>
  <Reference>BLAKE-2001062514034298PDT</Reference>
  <Actions>
    <Action>
      <User>KING</User>
      <Date/>
    </Action>
  </Actions>
  <Reject/>
  <Requester>David E. Blake</Requester>
  <User>BLAKE</User>
  <CostCenter>S30</CostCenter>
  <ShippingInstructions>
    <name>David E. Blake</name>
    <address>400 Oracle Parkway Redwood Shores, CA, 94065 USA</address>
    <telephone>650 999 9999</telephone>
  </ShippingInstructions>
  <SpecialInstructions>Air Mail</SpecialInstructions>
  <LineItems>
    <LineItem ItemNumber="1">
      <Description>The Birth of a Nation</Description>
      <Part Id="EE888" UnitPrice="65.39" Quantity="31"/>
    </LineItem>
  </LineItems>
</PurchaseOrder>
');
```

Use the `getClobVal` function to retrieve the data:

```
SELECT p.podocument.getClobVal() FROM PURCHASEORDER p;
```

Unicode Support

The Oracle Wire Protocol driver automatically determines whether the Oracle database is a Unicode database.

If the database character set is set to UTF-8, the Oracle driver maps the Oracle data types to Unicode data types as shown in the following table:

Oracle Data Type	Mapped to . . .
CHAR	SQL_WCHAR
CLOB	SQL_WLONGVARCHAR
VARCHAR2	SQL_WVARCHAR
LONG	SQL_WLONGVARCHAR

The driver also continues to map these Oracle data types to the normal character data types. See [Data Types](#) on page 370 for these mappings. The only exception to this is that when the Enable N-CHAR Support option is enabled, the N-CHAR types are mapped to the Unicode types SQL_WCHAR, SQL_WVARCHAR, and SQL_WLONGVARCHAR, and the normal character types are mapped to the data types SQL_CHAR, SQL_LONGVARCHAR, and SQL_VARCHAR, regardless of the character set on the Oracle server.

Advanced Features

The driver supports the following advanced features:

- Failover
- Client Information
- Security
- Connection Pooling
- DataDirect Bulk Load

Failover

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Client Information

Oracle provides a client information feature that allows an administrator to define different work load classifications and store client information associated with a connection. These workload classifications can be assigned different priorities and resource allocations. To enable applications to leverage these work load classifications fully, the Oracle Wire Protocol driver provides connection options for setting the session properties that are used in identifying a work load. These options are located on the [Client Monitoring tab](#) of the driver Setup dialog box. See [Using Client Information](#) on page 87 for a general description of client information and its implementation.

Security

The driver supports authentication in addition to encryption and data integrity checks. Security connection options are located on the [Security tab](#) and [Advanced Security tab](#) of the driver Setup dialog box. See [Using Security](#) on page 89 for a general description of security and its implementation. The following security information is specific to the Oracle Wire Protocol Driver.

Authentication

If you are using Kerberos, verify that your environment meets the requirements listed in the following table before you configure the driver for Kerberos authentication.

Table 27: Kerberos Authentication Requirements for the Oracle Wire Protocol Driver

Component	Requirements
Database server	<p>The database server must be administered by the same domain controller that administers the client and must be running one of the following databases²⁶ :</p> <ul style="list-style-type: none"> • Oracle 11g (R1 and R2) • Oracle 10g (R1 and R2) • Oracle 9i (R2) <p>In addition, Oracle Advanced Security is required.</p>
Kerberos server	<p>The Kerberos server is the machine where the user IDs for authentication are administered. The Kerberos server is also the location of the Kerberos KDC. Network authentication must be provided by one of the following methods:</p> <ul style="list-style-type: none"> • Windows Active Directory • MIT Kerberos 1.4.2 or higher
Client	<p>The client must be administered by the same domain controller that administers the database server.</p>

Oracle Advanced Security

To enable support for TLS/SSL connections to Oracle, the Oracle database must be configured with the Oracle Advanced Security bundle. This is an option available from Oracle as an add-on to Oracle Enterprise Edition Servers.

The driver also supports encryption and data integrity checks through Oracle Advanced Security. Oracle Advanced Security provides the Advanced Encryption Standard (AES), DES, 3DES, and RC4 symmetric cryptosystems for protecting the confidentiality of network traffic.

Encrypting network data provides data privacy so that unauthorized parties cannot view and alter clear text data as it passes over the network. Attacks on intercepted data include data modification and replay attacks.

- In a data modification attack, an unauthorized party intercepts transmitted data, alters it, and retransmits it. For example, suppose a customer order for 5 widgets for delivery to an office in San Francisco is intercepted. A data modification attack might change the quantity to 500 and the delivery address to a warehouse in Los Angeles, and then retransmit the order.
- In a replay attack, a set of valid data is retransmitted a number of times. For example, an order for 100 widgets is intercepted and then retransmitted ten times so the final order quantity equals 1,000 widgets.

Because data integrity protection operates independently from the encryption process, you can enable data integrity with or without enabling encryption.

Connection Pooling

The driver supports connection pooling and its related connection options. Connection pooling connection options are located on the [Pooling tab](#) of the driver Setup dialog box. See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling and its implementation.

²⁶ See the readme for the latest information, including support for Kerberos on Oracle 12c.

DataDirect Bulk Load

The driver supports DataDirect bulk load and its related connection options. Bulk load connection options are located on the [Bulk](#) tab of the driver Setup dialog box. See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect bulk load and its implementation.

The Oracle Wire Protocol driver uses array binding instead of DataDirect Bulk Load when the Oracle server version is older than Oracle 9i R1 (9.0.1)

Limitations

- A bulk operation is not allowed in a manual transaction if it is not the first event.
- Once a bulk operation is started, any non-bulk operation is disallowed until the transaction is committed.
- Because of Oracle limitations, issuing a SELECT statement to determine a row count may return different results before and after a bulk load operation.

MTS Support

On Windows, the driver can take advantage of Microsoft Transaction Server (MTS) capabilities, specifically, the Distributed Transaction Coordinator (DTC) using the XA Protocol. For a general discussion of MTS and DTC, refer to the help file of the Microsoft Transaction Server SDK.

Note: The DataDirect Connect *for* ODBC 32-bit drivers can operate in a 64-bit Windows environment; however, they do not support DTC in this environment. Only the DataDirect Connect64 *for* ODBC 64-bit drivers support DTC in a 64-bit Windows environment.

To enable DTC support, you must be connected to an Oracle 8.1.7 or higher server.

OS Authentication

On Windows, UNIX, and Linux, Oracle has a feature called OS Authentication that allows you to connect to an Oracle database via the operating system user name and password. To connect, use a forward slash (/) for the user name and leave the password blank. To configure the Oracle server, refer to the Oracle server documentation. This feature is valid when connecting from a data source, a connection string, or a logon dialog box.

Support for Oracle RAC

Oracle introduced Real Application Clusters (RAC) with Oracle 9i, and RAC is also a key feature of Oracle 10g. Oracle RAC allows a single physical Oracle database to be accessed by concurrent instances of Oracle running across several different CPUs.

An Oracle RAC is composed of a group of independent servers, or nodes, that cooperate as a single system. A cluster architecture such as this provides applications access to more computing power when needed, while allowing computing resources to be used for other applications when database resources are not as heavily required. For example, in the event of a sudden increase in network traffic, an Oracle RAC can distribute the load over many nodes, a feature referred to as *server load balancing*. Oracle RAC features are available to you simply by connecting to an Oracle RAC system with a DataDirect Connect Series *for* ODBC driver. There is no additional configuration required.

Connection failover and *client load balancing* can be used in conjunction with an Oracle RAC system, but they are not specifically part of Oracle RAC. The drivers can also use these two features on DB2, Informix, SQL Server, and Sybase database systems. See [Using Failover](#) on page 78 for details about how these features work in DataDirect Connect Series *for* ODBC drivers.

Support of Materialized Views

When connected to an Oracle 9i or higher server, the Oracle Wire Protocol driver supports the creation of materialized views. Materialized views are like any other database view with the following additions: the results are stored as a database object and the results can be updated on a schedule determined by the Create View statement.

Materialized views improve performance for data warehousing and replication. Refer to the Oracle documentation for more information about materialized views.

Stored Procedure Results

When you enable the Procedure Returns Results connection option, the driver returns result sets from stored procedures/functions. In addition, SQLGetInfo(SQL_MULT_RESULTS_SETS) returns Y and SQLGetInfo(SQL_BATCH_SUPPORT) returns SQL_BS_SELECT_PROC. If this option is enabled and you execute a stored procedure that does not return result sets, you incur a small performance penalty.

This feature requires that stored procedures be in a certain format. First, a package must be created to define all of the cursors used in the procedure; then, the procedure can be created using the new cursor. For example:

```
Create or replace package GEN_PACKAGE as
CURSOR G1 is select CHARCOL from GTABLE2;
type GTABLE2CHARCOL is ref cursor return G1%rowtype;
end GEN_PACKAGE;
Create or replace procedure GEN_PROCEDURE1 (
  rset IN OUT GEN_PACKAGE.GTABLE2CHARCOL, icol INTEGER) as
begin
  open rset for select CHARCOL from GTABLE2
  where INTEGERCOL <= icol order by INTEGERCOL;
end;
```

When executing the stored procedures with result sets, do not include the result set arguments (Oracle ref cursors) in the list of procedure parameters. The result set returned through the ref cursor is returned as a normal ODBC result set.

```
{call GEN_PROCEDURE1 (?)}
```

where ? is the parameter for the icol argument.

For more information, refer to your Oracle SQL documentation.

Unexpected Characters

Users are sometimes surprised when they insert a character into a database, only to have a different character displayed when they fetch it from the database. There are many reasons this can happen, but it most often involves code page issues, not driver errors.

Client and server machines in a database system each use code pages, which can be identified by a name or a number, such as Shift_JIS (Japanese) or cp1252 (Windows English). A code page is a mapping that associates a sequence of bits, called a code point, with a specific character. Code pages include the characters and symbols of one or more languages. Regardless of geographical location, a machine can be configured to use a specific code page. Most of the time, a client and database server would use similar, if not identical, code pages. For example, a client and server might use two different Japanese code pages, such as Shift_JIS and EUC_JP, but they would still share many Japanese characters in common. These characters might, however, be represented by different code points in each code page. This introduces the need to convert between code pages to maintain data integrity. In some cases, no one-to-one character correspondence exists between the two code points. This causes a substitution character to be used, which can result in displaying an unexpected character on a fetch.

When the driver on the client machine opens a connection with the database server, the driver determines the code pages being used on the client and the server. This is determined from the Active Code Page on a Windows-based machine. If the client machine is UNIX-based, the driver checks the IANAAppCodePage option. If it does not find a specific setting for IACP, it defaults to a value of ISO_8859_1.

If the client and server code pages are compatible, the driver transmits data in the code page of the server. Even though the pages are compatible, a one-to-one correspondence for every character may not exist. If the client and server code pages are completely dissimilar, for example, Russian and Japanese, then many substitutions occur because very few, if any, of the characters are mapped between the two code pages.

The following is a specific example of an unexpected character:

- The Windows client machine is running code page cp1252.
- The Oracle server is running code page ISO-8859-P1.
- When you insert a Euro character (€) from the Windows client and then fetch it back, an upside down question mark (¿) is displayed on the client instead of the Euro symbol.

This substitution occurs because the Euro character does not exist within the characters defined by the ISO-8859-P1 character set on the Oracle server. The Oracle server records the code point for its substitution character in the table instead of the code point for the Euro. This code point is an upside down question mark in the Windows cp1252 code page.

This is not a driver error. The code page of the Oracle database could not recognize the Euro code point and used its substitution character in the table. The best way to avoid these problems is to use the same code page on both the client and server machines.

You can check the native code point stored in the Oracle database using SQL*Plus with a SQL statement similar to the following:

```
SELECT dump(columnname, 1016) FROM yourtable;
```

Check the returned hexadecimal values to verify whether the data you intended to reside in the table is there. If it appears that Oracle substituted a different code point, then check the Oracle database code page to see if your intended character exists. If your character does not exist in the code page, then no error is involved; Oracle simply does not recognize the original character, and uses its substitution character instead.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Note: If you are persisting a result set that contains Long data, you must enable the EnableStaticCursorsforLongData connection string attribute.

Isolation and Lock Levels Supported

Oracle supports isolation level 1 (read committed) and isolation level 3 (serializable). Oracle supports record-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the core SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

In addition, the following functions are supported:

- SQLColumnPrivileges
- SQLDescribeParam (if EnableDescribeParam=1)
- SQLForeignKeys
- SQLPrimaryKeys
- SQLProcedures
- SQLProcedureColumns
- SQLSetPos
- SQLTablePrivileges

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The Oracle Wire Protocol driver supports multiple connections and multiple statements per connection.

Using Parameter Arrays

The driver supports native parameter arrays in Oracle 9i and higher databases with the following limitations:

- A bulk operation is not allowed in a manual transaction if it is not the first event.
- Bulk inserts into views are not allowed.
- Once a bulk operation is started, any non-bulk operation is disallowed until the transaction is committed.
- The Oracle Wire Protocol driver currently does not support the use of BLOB, CLOB, LONG, LONG RAW, and XMLType data types when using bulk load for parameter array batch.
- Because of Oracle limitations, issuing a SELECT statement to determine a row count may return different results before and after a bulk load operation.
- Oracle does not support literal values in a bulk load operation. You must use parameter markers for all columns being loaded.
- INSERT INTO SELECT statements are not supported.

When designing an application for performance, using native parameter arrays for bulk inserts or updates, for example, can improve performance.

Refer to "Designing ODBC applications for performance optimization" in the *Progress DataDirect for ODBC Drivers Reference* for details.

The PostgreSQL Wire Protocol Driver

The DataDirect Connect *for* ODBC and DataDirect Connect64 *for* ODBC PostgreSQL Wire Protocol driver (the PostgreSQL Wire Protocol driver) each support PostgreSQL database servers.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The PostgreSQL Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the Windows, UNIX, and Linux environments supported by this driver.

Refer to the readme file shipped with your DataDirect product for the file name of the PostgreSQL Wire Protocol driver.

Driver Requirements

The driver has no client requirements.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 389 and [Connection Option Descriptions for PostgreSQL Wire Protocol](#) on page 391 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

UNIX® On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions for PostgreSQL Wire Protocol](#) on page 391 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (PostgreSQL)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX® On UNIX and Linux, data sources are stored in the odbc.ini file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a PostgreSQL data source:

1. Start the ODBC Administrator:


-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- **UNIX**® On Linux, change to the `install_dir/tools` directory and, at a command prompt, enter:
`odbcadmin`

where *install_dir* is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

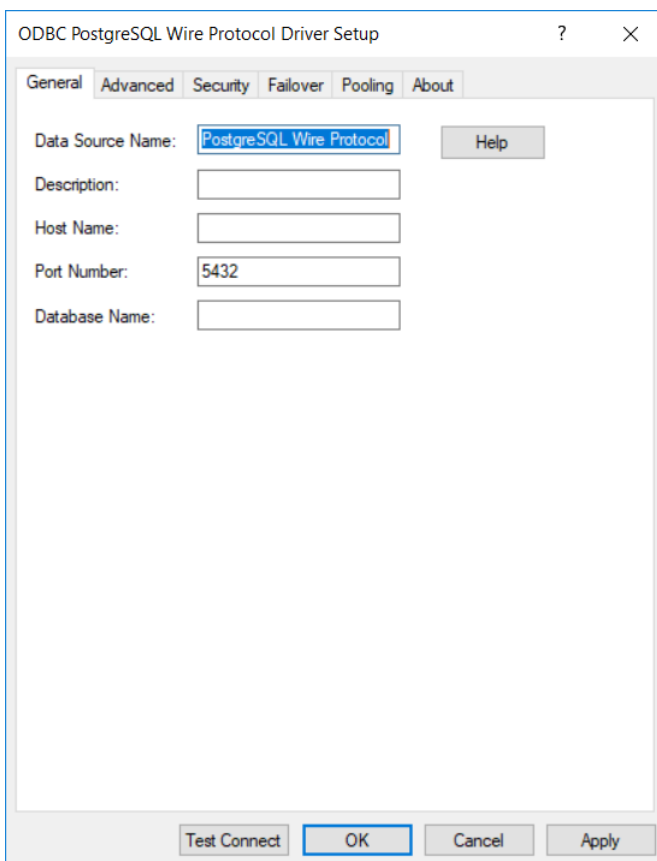
-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

Figure 31: General tab



The General tab of the Setup dialog box appears.

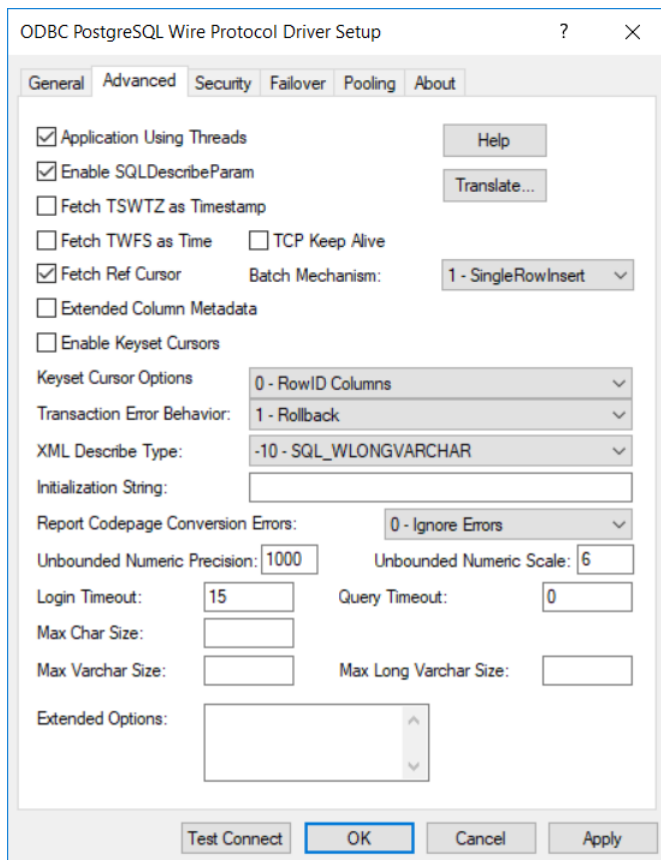
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 402	None
Description on page 403	None
Host Name on page 411	None
Port Number on page 422	5432
Database Name on page 403	None

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 32: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Application Using Threads on page 395	Enabled
Enable SQLDescribeParam on page 404	Enabled
Fetch TSWTZ as Timestamp on page 338	Disabled
Fetch TWFS as Time on page 410	Disabled
TCP Keep Alive on page 425	Disabled
Fetch Ref Cursors on page 409	Enabled
Batch Mechanism on page 397	1 -SingleRowInsert
Extended Column MetaData on page 406	Disabled
Enable Keyset Cursors on page 404	Disabled
Keyset Cursor Options on page 416	0 - RowID Columns
Transaction Error Behavior on page 426	1 - Rollback
XML Describe Type on page 430	-10 - SQL_WLONGVARCHAR
Initialization String on page 413	None
Report Codepage Conversion Errors on page 423	0 - Ignore Errors
Unbounded Numeric Precision on page 428	1000
Unbounded Numeric Scale on page 428	6
Login Timeout on page 418	15
Query Timeout on page 422	0
Max Char Size on page 419	None
Max Varchar Size on page 420	None
Max Long Varchar Size on page 419	None
IANAAppCodePage on page 413UNIX ONLY	4 (ISO 8559-1 Latin-1)



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```


If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.

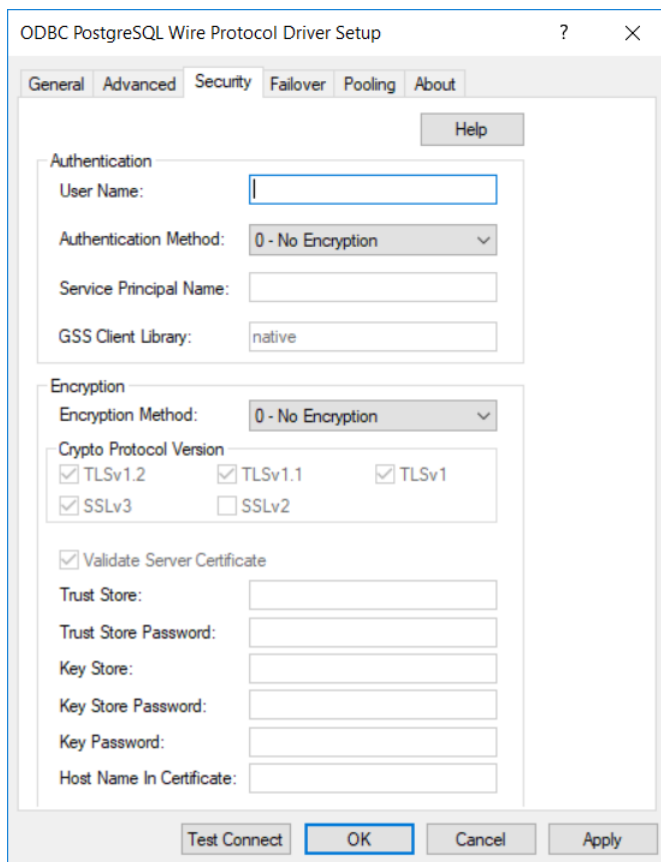


Translate : Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- Optionally, click the **Security** tab to specify security data source settings.

Figure 33: Security tab



See [Using Security](#) on page 89 for a general description of encryption and its configuration requirements.

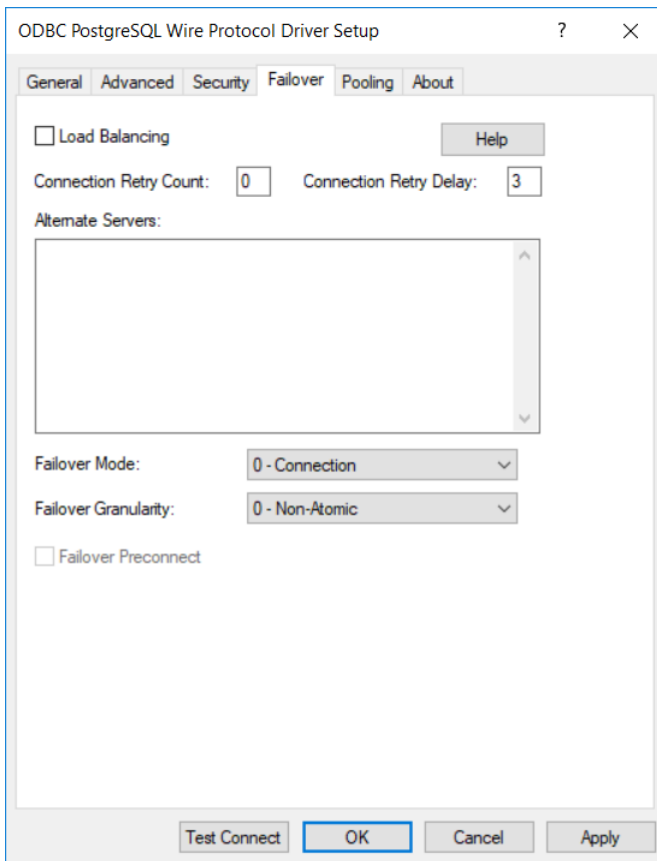
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Security	Default
User Name on page 429	None

Connection Options: Security	Default
Authentication Method on page 396	0 - No Encryption
Service Principal Name on page 424	None
GSS Client Library on page 410	native
Encryption Method on page 405	0 - No Encryption
Crypto Protocol Version on page 400	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Validate Server Certificate on page 430	Enabled
Trust Store on page 427	None
Trust Store Password on page 427	None
Key Store on page 415	None
Key Store Password on page 415	None
Key Password on page 414	None
Host Name In Certificate on page 412	None

6. Optionally, click the **Failover** tab to specify failover data source settings.

Figure 34: Failover tab



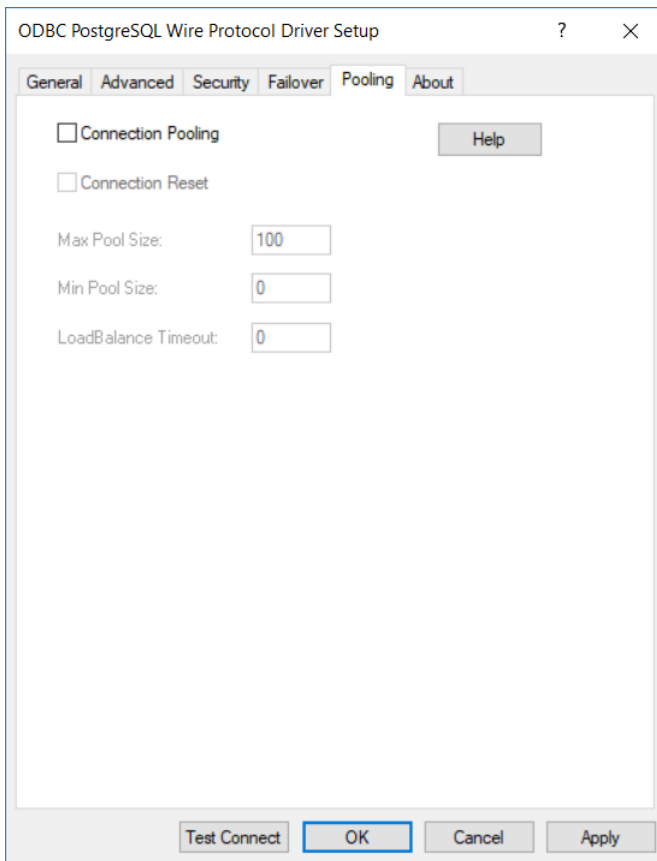
See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 417	Disabled
Connection Retry Count on page 399	0
Connection Retry Delay on page 400	3
Alternate Servers on page 395	None
Failover Mode on page 407	0 - Connection
Failover Granularity on page 407	0 - Non-Atomic
Failover Preconnect on page 408	Disabled

7. Optionally, click the **Pooling** tab to specify pooling data source settings.

Figure 35: Pooling tab



See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 398	Disabled
Connection Reset on page 398	Disabled
Max Pool Size on page 420	100
Min Pool Size on page 421	0
Load Balance Timeout on page 417	0

8. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(PostgreSQL\)](#) on page 390 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.

- If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

9. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={ }driver_name[;attribute=value[;attribute=value]...]
```

[Connection Option Descriptions for PostgreSQL Wire Protocol](#) on page 391 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for PostgreSQL Wire Protocol is:

```
DSN=Accounting;UID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=PostgreSQLWP.dsn;UID=JOHN;PWD=XYZZY
```

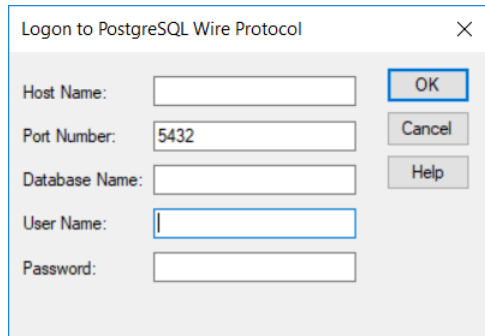
A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 PostgreSQL Wire Protocol};
HOST=PostgreSQLServer;PORT=5432;UID=JOHN;PWD=XYZZY;DB=Pgredbl
```

Using a Logon Dialog Box (PostgreSQL)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Figure 36: Logon to PostgreSQL Wire Protocol dialog box



In this dialog box, provide the following information:

1. In the Host Name field, type either the name or the IP address of the server to which you want to connect. The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details concerning these formats.
2. In the Port Number field, type the number of your PostgreSQL listener. Check with your database administrator for the correct number.
3. In the Database Name field, type the name of the database to which you want to connect.
4. If required, type your PostgreSQL user name.
5. If required, type your PostgreSQL password.
6. Click **OK** to log on to the PostgreSQL database installed on the server you specified and to update the values in the Registry.

Accessing PostgreSQL data with Power BI

After you have configured your data source, you can use the driver to access your PostgreSQL data with Power BI. Power BI is a business intelligence software program that allows you to easily create reports and visualized representations of your data. By using the driver with Power BI, you can improve performance when retrieving data while leveraging the driver's relational mapping tools.

To use the driver to access data with Power BI:

1. Navigate to the `\tools\Power BI` subdirectory of the Progress DataDirect installation directory; then, locate the Power BI connector file `DataDirectPostgreSQL.pqx`, and the installation batch file `install.bat`.
2. Run the `install.bat` file. The following operations are executed by running the `install.bat` file.
 - The `DataDirectPostgreSQL.pqx` file is copied to the following directory.
`%USERPROFILE%\Documents\Power BI Desktop\Custom Connectors`
 - The following Windows registry entry is updated.
`HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Power BI Desktop\TrustedCertificateThumbprints`

3. Open the Power BI desktop application.
4. From the **Get Data** window, navigate to **Other > DataDirect PostgreSQL Connector**.
5. Click **Connect**. Then, from the **DataDirect PostgreSQL Connector** pop-up, provide the following information. Then, click **OK**.
 - **Data Source:** Enter a name for the data source. For example, `PostgreSQL ODBC DSN`.
 - **SQL Statement:** If desired, provide a SQL command.
 - **Data Connectivity mode:**
 - Select **Import** to import data to Power BI.
 - Select **DirectQuery** to query live data. (For details, including limitations, refer to the Microsoft Power BI article [Use DirectQuery in Power BI Desktop](#).)
6. Enter authentication information when prompted. Once connected, the **Navigator** window displays schema and table information.
7. Select and load tables. Then, prepare your Power BI dashboard as desired.

Results:

You have successfully accessed your data and are now ready to create reports with Power BI. For more information, refer to the Power BI product documentation at [Power BI documentation](#).

Connection Option Descriptions for PostgreSQL Wire Protocol

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the PostgreSQL Wire Protocol driver.

Table 28: PostgreSQL Wire Protocol Attribute Names

Attribute (Short Name)	Default
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
AlternateServers (ASRV)	None
ApplicationUsingThreads (AUT)	1 (Enabled)
AuthenticationMethod (AM)	0 (No Encryption)
BatchMechanism (BM)	1 (SingleRowInsert)
ConnectionReset (CR)	0 (Disabled)

Attribute (Short Name)	Default
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Database (DB)	None
DataSourceName (DSN)	None
Description (n/a)	None
EnableDescribeParam (EDP)	1 (Enabled)
EnableKeysetCursors (EKC)	0 (Disabled)
EncryptionMethod (EM)	0 (No Encryption)
ExtendedColumnMetaData (ECMD)	0 (Disabled)
FailoverGranularity (FG)	0 (Non-Atomic)
FailoverMode (FM)	0 (Connection)
FailoverPreconnect (FP)	0 (Disabled)
FetchRefCursors (FRC)	1 (Enabled)
FetchTSWTZasTimestamp (FTSWTZAT)	0 (Disabled)
FetchTWFSasTime (FTWFSAT)	0 (Disabled)
GSSClient (GSSC)	native
HostName (HOST)	None
HostNameInCertificate (HNIC)	None
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
InitializationString (IS)	None
KeepAlive (KA)	Disabled
KeyPassword (KP)	None
Keyset Cursor Options	0 - RowID Columns
Keystore (KS)	None

Attribute (Short Name)	Default
KeystorePassword (KSP)	None
LoadBalanceTimeout (LBT)	0
LoadBalancing (LB)	0 (Disabled)
LoginTimeout (LT)	15
LogonID (UID)	None
Max Char Size	None
MaxLongVarcharSize (MLVS)	None
MaxPoolSize (MXPS)	100
MaxVarcharSize (MVS)	None
MinPoolSize (MNPS)	0
Password (PWD)	None
Pooling (POOL)	0 (Disabled)
PortNumber (PORT)	5432
QueryTimeout (QT)	0
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
ServicePrincipalName (SPN)	None
SSLlibName (SLN)	Empty string
TransactionErrorBehavior (TEB)	1 (Rollback Transaction)
Truststore (TS)	None
TruststorePassword (TSP)	None
UnboundedNumericPrecision (UNP)	1000
UnboundedNumericScale (UNS)	6
ValidateServerCertificate (VSC)	1 (Enabled)
XMLDescribeType (XDT)	-10

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

`latest | openssl_version_number[[, openssl_version_number] ...]`

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to `openssl_version_number`, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the `CryptoLibName` and `SSLlibName` options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

`1.1.1,1.0.2`

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

[Advanced tab](#)

Alternate Servers

Attribute

AlternateServers (ASRV)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
(HostName=hostvalue:PortNumber=portvalue:Database=databasevalue[, . . .])
```

You must specify the host name, port number, and database name of each alternate server.

Example

The following Alternate Servers value defines two alternate database servers for connection failover:

```
AlternateServers=(HostName=PostgreSQLServer:  
PortNumber=5431:Database=Pgredb1,  
HostName=255.201.11.24:PortNumber=5432:Database=Pgredb2)
```

Notes

- An alternate server address in IPv6 format must be enclosed in double quotation marks.

Default

None

GUI Tab

[Failover tab](#)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 431

Authentication Method

Attribute

AuthenticationMethod (AM)

Purpose

Specifies the method the driver uses to authenticate the user to the server when a connection is established. If the specified authentication method is not supported by the database server, the connection fails and the driver generates an error.

Important: When Kerberos is enabled, if the database user name differs from the domain user name, you are required to pass the database user name via the User Name (LogonID) option in the datasource or connection string.

Valid Values

0 | 4

Behavior

If set to 0 (No Encryption), the driver sends the user ID and password in clear text to the server for authentication.

If set to 4 (Kerberos Authentication), the driver uses Kerberos authentication. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

Batch Mechanism

Attribute

BatchMechanism (BM)

Purpose

Determines the mechanism that is used to execute batch operations.

Valid Values

1 | 2 | 3

Behavior

If set to 1 (SingleRowInsert), the driver executes an insert statement for each row contained in a parameter array. Specify this value if you are experiencing out-of-memory errors when performing batch inserts.

If set to 2 (MultiRowInsert), the driver attempts to execute a single insert statement for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the driver, the driver executes multiple statements. Specify this value for substantial performance gains over 1 (SingleRowInsert) when performing batch inserts.

If set to 3 (Copy), the driver uses the PostgreSQL COPY command to insert rows into the target table. Specify this value for substantial performance gains over 1 (SingleRowInsert) when performing batch inserts.

Default

1 (SingleRowInsert)

Notes

- Batch Mechanism determines the mechanism used to perform batch inserts only. For update and delete batch operations, the driver uses the native batch mechanism to handle the request.
- When BatchMechanism=3, substantial performance gains can be made. However, the following limitations apply:
 - Individual update counts are not returned. However, the total number of inserted rows are returned the execution of a batch operation.
 - The entire batch insert is ATOMIC. If any issues are encountered, the entire operation fails and no rows are inserted.

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 431 for details.

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- The application must be thread-enabled to use connection pooling.
- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 431

Connection Reset

Attribute

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 431

Connection Retry Count

Attribute

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x , the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay (CRD)

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to x , the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (EncryptionMethod=1 | 6). When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, behavior is determined by the setting of the EncryptionMethod connection option.

Valid Values

cryptographic_protocol [, *cryptographic_protocol*]...

where:

cryptographic_protocol

is one of the following cryptographic protocols:

TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Notes

- This option is ignored if Encryption Method is set to 0 - No Encryption.
- Consult your database administrator concerning the data encryption settings of your server.

Default

TLSv1.2, TLSv1.1, TLSv1

GUI Tab

[Security tab](#)

See also

[Encryption Method](#) on page 405

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\  
Drivers\OpenSSL\1.0.0r\ddssl27.dll; (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLibName](#) on page 424

Data Source Name

Attribute

`DataSourceName` (DSN)

Description

Specifies the name of a data source in your Windows Registry or `odbc.ini` file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database Name

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where

database_name

is the name of a valid database.

Default

None

GUI Tab

[General tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable Keyset Cursors

Attribute

EnableKeysetCursors (EKC)

Purpose

Determines whether the driver emulates keyset cursors to provide scrollable keyset cursors to an ODBC application.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver emulates keyset cursors.

If set to 0 (Disabled), the driver does not emulate keyset cursors. If an application requests a keyset cursor and this option is set to 0, the driver uses a static cursor and returns a message that a different value was used.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Enable SQLDescribeParam

Attribute

EnableDescribeParam (EDP)

Purpose

Determines whether SQLDescribeParam returns the Datatype, ParameterSize, DecimalDigits, and Nullable information for parameters in a prepared statement.

Valid Values

0 | 1

Behavior

If set to 1 (enabled), SQLDescribeParam returns the Datatype, ParameterSize, DecimalDigits, and Nullable information for parameters in a prepared statement.

If set to 0 (disabled), the driver does not support SQLDescribeParam and returns the message: `Driver does not support this function.`

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Encryption Method

Attribute

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server. It supports the RequestSSL functionality. When RequestSSL is enabled, login requests and data are encrypted if the server is configured for SSL. If the server is not configured for SSL, an unencrypted connection is established.

Valid Values

0 | 1 | 6

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using the SSL protocols specified in the Crypto Protocol Version connection option. If the specified encryption method is not supported by the database server, the connection fails and the driver returns an error.

If set to 6 (RequestSSL), the login request and data are encrypted using SSL if the server is configured for SSL. If the server is not configured for SSL, an unencrypted connection is established. The SSL protocol used is determined by the setting of the Crypto Protocol Version connection option.

Notes

- For values 1 and 6, the SSL protocol used is determined by the setting of the Crypto Protocol Version connection option.
- This connection option can affect performance.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

See also

[Crypto Protocol Version](#) on page 400

[Performance Considerations](#) on page 431

Extended Column MetaData

Attribute

ExtendedColumnMetaData (ECMD)

Purpose

Determines how the driver returns column metadata when using SQLDescribeCol and SQLColAttribute.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), SQLDescribeCol returns the actual values for Data Type, Column Size, Decimal Digits, and Nullable. SQLColAttribute returns the actual values for:

- SQL_DESC_CATALOG_NAME: *catalog_name*
- SQL_DESC_TABLE_NAME: *table_name*
- SQL_DESC_BASE_COLUMN_NAME: *base_column_name*
- SQL_DESC_LOCAL_TYPE_NAME: *local_type_name*
- SQL_DESC_NULLABLE: *nullable*
- SQL_DESC_AUTO_UNIQUE_VALUE: *auto_unique_value*

If set to 0 (Disabled), SQLDescribeCol returns the Data Type, Column Size, and Decimal Digits for the column. The value SQL_NULLABLE_UNKNOWN is returned for Nullable. SQLColAttribute returns the following attribute values:

- SQL_DESC_CATALOG_NAME: empty string
- SQL_DESC_TABLE_NAME: empty string
- SQL_DESC_BASE_COLUMN_NAME: empty string
- SQL_DESC_LOCAL_TYPE_NAME: empty string
- SQL_DESC_NULLABLE: SQL_NULLABLE_UNKNOWN
- SQL_DESC_AUTO_UNIQUE_VALUE: SQL_FALSE

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Failover Granularity

Attribute

FailoverGranularity (FG)

Purpose

Determines whether the driver fails the entire failover process or continues with the process if errors occur while trying to reestablish a lost connection.

This option applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select).

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 (Non-Atomic), the driver continues with the failover process and posts any errors on the statement on which they occur.

If set to 1 (Atomic) the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

If set to 2 (Atomic Including Repositioning), the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

If set to 3 (Disable Integrity Check), the driver does not verify that the rows that were restored during the failover process match the original rows. This value applies only when Failover Mode is set to 2 (Select).

Default

0 (Non-Atomic)

GUI Tab

[Failover tab](#)

Failover Mode

Attribute

FailoverMode (FM)

Purpose

Specifies the type of failover method the driver uses.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Connection), the driver provides failover protection for new connections only.

If set to 1 (Extended Connection), the driver provides failover protection for new and lost connections, but not any work in progress.

If set to 2 (Select), the driver provides failover protection for new and lost connections. In addition, it preserves the state of work performed by the last Select statement executed.

Default

0 (Connection)

GUI Tab

[Failover tab](#)

Failover Preconnect

Attribute

FailoverPreconnect (FP)

Purpose

Specifies whether the driver tries to connect to the primary and an alternate server at the same time.

This attribute applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select) and at least one alternate server is specified.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver tries to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection. This value provides the best performance, but your application typically experiences a short wait while the failover connection is attempted.

If set to 1 (Enabled), the driver tries to connect to the primary and an alternate server at the same time. This can be useful if your application is time-sensitive and cannot absorb the wait for the failover connection to succeed.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Fetch Ref Cursors

Attribute

FetchRefCursors (FRC)

Purpose

Determines whether the driver returns refcursors from stored procedures as results sets.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns refcursors from stored procedures as result sets. The driver fetches all the data from the refcursor and then closes the refcursor. If a stored procedure returns multiple refcursors, the driver generates multiple result sets, one for each refcursor returned.

If set to 0 (Disabled), the driver returns the cursor name for refcursors. The application must fetch the actual data from the refcursor using the cursor name and must close the cursor before additional processing can be done on the statement. The application must close the cursor regardless of whether it actually fetches data from the cursor.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Fetch TSWTZ as Timestamp

Attribute

FetchTSWTZasTimestamp (FTSWTZAT)

Purpose

Determines whether the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_TYPE_TIMESTAMP or SQL_VARCHAR.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the timestamp with time zone data type as the ODBC type SQL_TYPE_TIMESTAMP. The time zone information in the fetched value is truncated. Use this value if your application needs to process values the same way as TIMESTAMP columns.

If set to 0 (Disabled), the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_VARCHAR. Use this value if your application requires the time zone information in the fetched value.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Fetch TWFS as Time

Attribute

FetchTWFSasTime (FTWFSAT)

Purpose

Determines whether the driver returns column values with the time data type as the ODBC data type SQL_TYPE_TIME or SQL_TYPE_TIMESTAMP.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the time data type as the ODBC data type SQL_TYPE_TIME. The fractional seconds portion of the value is truncated.

If set to 0 (Disabled), the driver returns column values with the time data type as the ODBC data type SQL_TYPE_TIMESTAMP. The fractional seconds portion of the value is preserved. Time columns are not searchable when they are described and fetched as timestamp.

Notes

- When returning time with fractional seconds data as SQL_TYPE_TIMESTAMP, the Year, Month and Day parts of the timestamp must be set to zero.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

GSS Client Library

Attribute

GSSClient (GSSC)

Purpose

The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC).

The driver uses the path defined by the PATH environment variable for loading the specified client library.

Valid Values

`native` | `client_library`

where:

`client_library`

is a GSS client library installed on the client.

Behavior

If set to `client_library`, the driver uses the specified GSS client library.

Note: For MIT Kerberos distributions, you must provide a full path to the MIT Library. For example, the 64-bit version for Windows would use the following value: `C:\Program Files\MIT\Kerberos\bin\gssapi64.dll`.

If set to `native`, the driver uses the GSS client for Windows Kerberos. All other users must provide the full path to the library name.

Default

`native`

GUI Tab

[Security tab](#)

Host Name

Attribute

HostName (HOST)

Purpose

The name or the IP address of the server to which you want to connect.

Valid Values

`server_name` | `IP_address`

where:

`server_name`

is the name of the server to which you want to connect.

`IP_address`

is the IP address of the server to which you want to connect.

The IP address can be specified in either IPv4 or IPv6 format. See [Using IP Addresses](#) on page 67 for details about these formats.

Example

If your network supports named servers, you can specify a server name such as `MainServer`. Or, you can specify an IP address such as `199.226.224.34..`

Default

None

GUI Tab

[General tab](#)

Host Name In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

host_name | *#SERVERNAME#*

where:

host_name

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If set to a host name, the driver examines the subjectAltName values included in the certificate. If a dnsName value is present in the subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the dnsName value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the dnsName value.

If no subjectAltName values exist or a dnsName value is not in the list of subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the commonName part of the Subject name in the certificate. The commonName typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the commonName. If multiple commonName parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the commonName parts.

If set to *#SERVERNAME#*, the driver compares the host server name specified as part of a data source or connection string to the dnsName or the commonName value.

Default

None

GUI Tab

[Security tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Initialization String

Attribute

InitializationString (IS)

Purpose

A SQL command that is issued immediately after connecting to the database to manage session settings.

Valid Values

SQL_command

where:

SQL_command

is a valid SQL command that is supported by the database.

Example

To set the date format on every connection, specify:

```
Set DateStyle='ISO, MDY'
```

Notes

- If the statement fails to execute, the connection fails and the driver reports the error returned from the server.

Default

None

GUI Tab

[Advanced tab](#)

Key Password

Attribute

KeyPassword (KP)

Purpose

The password used to access the individual keys in the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. Keys stored in a keystore can be individually password-protected. To extract the key from the keystore, the driver must have the password of the key.

Valid Values

key_password

where:

key_password

is the password of a key in the keystore.

Default

None

GUI Tab

[Security tab](#)

Key Store

Attribute

Keystore (KS)

Purpose

The name of the directory containing the keystore file to be used when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request. If you do not specify a directory, the current directory is used.

Valid Values

`keystore_directory`

where:

`keystore_directory`

is the location of the keystore file.

Notes

- The keystore and truststore files can be the same file.

Default

None

GUI Tab

[Security tab](#)

Key Store Password

Attribute

KeystorePassword (KSP)

Purpose

The password used to access the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request.

Valid Values

`keystore_password`

where:

keystore_password

is the password of the keystore file.

Notes

- The keystore and truststore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Keyset Cursor Options

Attribute

KeysetCursorOptions (KCO)

Purpose

Determines which columns are used to comprise the keyset that the driver uses to create the initial keyset on which cursor operations are based. PostgreSQL does not offer a true row identifier column; the driver instead uses a hidden system column provided by the PostgreSQL database, ctid. Because the database might reassign the ID following a Vacuum operation, the driver can be configured to also include other columns to help ensure that data integrity is maintained.

Valid Values

0 | 1

Behavior

If set to 1 (RowID and Searchable Columns), the driver uses a combination of every non-LOB column in the Select list and the ctid ahidden column to build the keyset. By adding other Select list fields to the keyset, the driver is able to indicate the row cannot be found if the IDs change following a Vacuum operation.

If set to 0 (RowID Columns), the driver uses the ctid hidden system column.

Notes

- This option has no effect unless the EnableKeysetCursors (EKC) connection option is enabled.

Default

0 (RowID) Columns

GUI Tab

[Advanced tab](#)

Load Balance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

Specifies the number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Valid Values

0 | x

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to x, inactive connections are closed after the specified number of seconds passes.

Notes

- The Min Pool Size option may cause some connections to ignore this value.
- This connection option can affect performance.

Default

0

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 431

Load Balancing

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | *x*

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to *x*, the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

15

GUI Tab

[Advanced tab](#)

Max Char Size**Attribute**

MaxCharSize (MCS)

Purpose

Specifies the maximum size of columns of type SQL_CHAR that the driver describes through result set descriptions and catalog functions.

Valid Values

A positive integer from 1 to 10485760

Behavior

When not specified, the actual size of the columns from the database is persisted to the application.

If you specify a value that is not in the specified range, the driver uses the maximum value of the SQL_CHAR data type, 10485760.

Default

None. The actual size of the columns from the database is persisted to the application.

GUI Tab

[Advanced tab](#)

Max Long Varchar Size**Attribute**

MaxLongVarcharSize (MLVS)

Purpose

Specifies the maximum size of columns of type SQL_LONGVARCHAR that the driver describes through result set descriptions and catalog functions.

Valid Values

A positive integer from 1 to x

where:

x

is maximum size of the SQL_LONGVARCHAR data type.

Default

None. The actual size of the columns from the database is persisted to the application.

GUI Tab

[Advanced tab](#)

Max Pool Size

Attribute

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

For example, if set to 20, the maximum number of connections allowed in the pool is 20.

Notes

- This connection option can affect performance.

Default

100

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 431

Max Varchar Size

Attribute

MaxVarcharSize (MVS)

Purpose

Specifies the maximum size of columns of type SQL_VARCHAR that the driver describes through result set descriptions and catalog functions.

Valid Values

A positive integer from 1 to x

where:

x

is maximum size of the SQL_VARCHAR data type.

Default

None. The actual size of the columns from the database is persisted to the application.

GUI Tab

[Advanced tab](#)

Min Pool Size

Attribute

MinPoolSize (MNPS)

Purpose

The minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Valid Values

0 | x

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

Notes

- This connection option can affect performance.

Example

If set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Default

0

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 431

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Purpose

The port number of the server listener.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your database administrator for the correct number.

Default

5432

GUI Tab

[General tab](#)

Query Timeout

Attribute

QueryTimeout (QT)

Purpose

The number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the `SQL_ATTR_QUERY_TIMEOUT` statement attribute on the `SQLSetStmtAttr()` function.

Valid Values

-1 | 0 | *x*

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to -1, the query does not time out. The driver silently ignores the `SQL_ATTR_QUERY_TIMEOUT` attribute.

If set to 0, the query does not time out, but the driver responds to the `SQL_ATTR_QUERY_TIMEOUT` attribute.

If set to *x*, all queries time out after the specified number of seconds unless the application overrides this value by setting the `SQL_ATTR_QUERY_TIMEOUT` attribute.

Default

0

GUI Tab

[Advanced tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where *x* is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Advanced tab](#)

Service Principal Name

Attribute

ServicePrincipalName (SPN)

Purpose

The service principal name to be used by driver for Kerberos authentication.

Valid Values

servicePrincipalName

where:

servicePrincipalName

is a valid service principal name.

If unspecified, the value of the Network Address option is used as the service principal name.

Notes

- If Authentication Method is set to 0, the value of the Service Principal Name option is ignored.

Default

None

GUI Tab

[Security tab](#)

SSLibName

Attribute

SSLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\
Drivers\OpenSSL\1.0.0r\ddssl27.dll; (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the CryptoLibName and SSLibName options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\
Connect64_for_ODBC_71\drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 401

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Transaction Error Behavior

Attribute

TransactionErrorBehavior (TEB)

Purpose

Determines how the driver handles errors that occur within a transaction. When an error occurs in a transaction, the PostgreSQL server does not allow any operations on the connection except for rolling back the transaction.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (None), the driver does not roll back the transaction when an error occurs. The application must handle the error and roll back the transaction. Any operation on the statement other than a rollback results in an error.

If set to 1 (Rollback Transaction), the driver rolls back the transaction when an error occurs. In addition to the original error message, the driver posts an error message indicating that the transaction has been rolled back.

If set to 2 (Rollback Savepoint), the driver rolls back the transaction to the last savepoint when an error is detected. In manual commit mode, the driver automatically sets a savepoint after each statement issued. This value makes transaction behavior resemble that of most other database system types, but uses more resources on the database server and may incur a slight performance penalty.

Default

1 (Rollback Transaction)

GUI Tab

[Advanced tab](#)

Trust Store

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used.

Valid Values

truststore_directory\filename

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Notes

- The keystore and truststore files may be the same file.

Default

None

GUI Tab

[Security tab](#)

Trust Store Password

Attribute

TruststorePassword (TSP)

Purpose

The password that is used to access the truststore file when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

- The truststore and keystore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Unbounded Numeric Precision

Attribute

UnboundedNumericPrecision (UNP)

Purpose

Specifies the precision for unbounded NUMERIC columns when they are described within the column, parameter, result set, or table metadata. Executing aggregation operations (for example, sum or avg) on bounded NUMERIC columns often results in the server returning the aggregate column as an unbounded NUMERIC column. When this occurs, this option defines the precision for the aggregate column.

Valid Values

A positive integer from 1 to 1000

Default

1000

GUI Tab

[Advanced tab](#)

Unbounded Numeric Scale

Attribute

UnboundedNumericScale (UNS)

Purpose

Specifies the scale for unbounded NUMERIC columns when they are described within the column, parameter, result set, or table metadata. Executing aggregation operations (for example, sum or avg) on bounded NUMERIC columns often results in the server returning the aggregate column as an unbounded NUMERIC column. When this occurs, this option defines the scale for the aggregate column.

Valid Values

A positive integer from 1 to 998

Notes

- The driver returns the scale specified in this option for the affected columns regardless of the number of decimal digits in a value. If a value contains fewer digits to the right of the decimal than the specified scale, the remaining digits are automatically returned as padded 0s. For example, if your scale is set to 6 and your value is 22.22, the value returned is 22.220000.

Default

6

GUI Tab

[Advanced tab](#)

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Important: When Kerberos is enabled, if the database user name differs from the domain user name, you are required to pass the database user name via the User Name (LogonID) option in the datasource or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Security tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Truststore information is specified using the Trust Store and Trust Store Password options.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Default

1 (Enabled)

GUI Tab

[Security tab](#)

XML Describe Type

Attribute

XMLDescribeType (XDT)

Purpose

The SQL data type that is returned by SQLGetTypeInfo for the XML data type.

See [Using the XML Data Type](#) on page 435 for further information about the XML data type.

Valid Values

-4 | -10

Behavior

If set to -4 (SQL_LONGVARIABLE), the driver uses the description SQL_LONGVARIABLE for columns that are defined as the XML data type.

If set to -10 (SQL_WLONGVARIABLE), the driver uses the description SQL_WLONGVARIABLE for columns that are defined as the XML data type.

Default

-10

GUI Tab

[Advanced tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Batch Mechanism (BatchMechanism): Setting BatchMechanism to 2 (MultiRowInsert) or 3 (Copy) provides significant performance gains over 1 (SingleRowInsert) when executing batch inserts:

- When `BatchMechanism=2`, the driver executes a single insert statement for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the server, the driver executes multiple statements.
- When `BatchMechanism=3`, the driver uses the PostgreSQL COPY command to insert rows into the target table.

Connection Pooling (Pooling): If you enable the driver to use connection pooling, you can set additional options that affect performance:

- **Load Balance Timeout (LoadBalancing):** You can define how long to keep connections in the pool. The time that a connection was last used is compared to the current time and, if the timespan exceeds the value of the Load Balance Timeout option, the connection is destroyed. The Min Pool Size option can cause some connections to ignore this value.
- **Connection Reset (ConnectionReset):** Resetting a re-used connection to the initial configuration settings impacts performance negatively because the connection must issue additional commands to the server.
- **Max Pool Size (MaxPoolSize):** Setting the maximum number of connections that the pool can contain too low might cause delays while waiting for a connection to become available. Setting the number too high wastes resources.
- **Min Pool Size (MinPoolSize):** A connection pool is created when the first connection with a unique connection string connects to the database. The pool is populated with connections up to the minimum pool size, if one has been specified. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Encryption Method (EncryptionMethod): Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) that is required to encrypt and decrypt data.

Failover Mode (FailoverMode): Although high availability that replays queries after a failure provides increased levels of protection, it can adversely affect performance because of increased overhead.

Data Types

The following table shows how the PostgreSQL data types are mapped to the standard ODBC data types. [Using the XML Data Type](#) on page 435 describes PostgreSQL to Unicode data type mappings.

Table 29: PostgreSQL Data Types

PostgreSQL	ODBC
Bigint	SQL_BIGINT
Bigserial	SQL_BIGINT
Bit ²⁷	SQL_BIT
Bit varying	SQL_VARBINARY
Boolean	SQL_BIT
Bytea	SQL_VARBINARY
Character	SQL_CHAR
Character varying	SQL_VARCHAR
Citext ^{28,29}	SQL_LONGVARCHAR
Date	SQL_TYPE_DATE
Decimal	SQL_DECIMAL
Double Precision	SQL_DOUBLE
Float	SQL_REAL
Integer	SQL_INTEGER
Money	SQL_VARCHAR
Name	SQL_VARCHAR

²⁷ Bit maps to SQL_BIT when the length for the bit is 1. If the length is greater than 1, the driver maps the column to SQL_BINARY.

²⁸ The Citext data type behaves the same as the Text data type, except that it is case-insensitive. The select operations performed on Citext columns return case-insensitive results.

²⁹ Supported for PostgreSQL versions 9.4 and higher.

PostgreSQL	ODBC
Numeric ³⁰	SQL_NUMERIC
Real	SQL_REAL
Serial	SQL_INTEGER
Smallint	SQL_SMALLINT
Text	SQL_LONGVARCHAR
Time ³¹	SQL_TYPE_TIME
Timestamp	SQL_TYPE_TIMESTAMP
Timestamp with timezone ³²	SQL_VARCHAR
Tinyint	SQL_SMALLINT
Wchar	SQL_CHAR
Wvarchar	SQL_VARCHAR
XML	SQL_WLONGVARCHAR

See [Retrieving Data Type Information](#) on page 72 for more information about data types.

Partially and Unsupported Data Types

The following section describes how the driver handles partially and unsupported data types.

Partially supported data types

The following table contains the PostgreSQL data types that are partially supported by the driver. These data types map the SQL_VARCHAR ODBC type with a scale of 0.

Table 30: Partially Supported Data Types

PostgreSQL Data Type	Precision
Box	255
Cidr	50
Inet	50
Int4	255

³⁰ Numeric maps to SQL_NUMERIC if the precision of the Numeric is less than or equal to 38. If the precision is greater than 38, the driver maps the column to SQL_VARCHAR.

³¹ Time mapping changes based on the setting of the Fetch TWFS as Time option

³² Timestamp with timezone mapping changes based on the setting of the Fetch TSWTZ as Timestamp option.

PostgreSQL Data Type	Precision
Interval	255
Lseg	255
Macaddr	17
Money	255
Point	255
Point	255
Polygon	255
Timetz	21

Unsupported data types

The following data types are not supported by the driver. When encountered by the driver, unsupported types map to the SQL_VARCHAR type with a precision of 10485760 and scale 0.

- UUID
- Record

Interval Types

The following Interval Types are not supported and map to SQL_VARCHAR with an incorrect precision:

- INTERVAL DAY
- INTERVAL DAY TO HOUR
- INTERVAL DAY TO MINUTE
- INTERVAL DAY TO SECOND
- INTERVAL HOUR
- INTERVAL HOUR TO SECOND
- INTERVAL MINUTE
- INTERVAL MINUTE TO SECOND
- INTERVAL MONTH
- INTERVAL SECOND
- INTERVAL YEAR
- INTERVAL YEAR TO MONTH

Information Schema Types

The columns of the information schema views use special data types that are defined in the information schema. These data types are not used while creating the table and are not supported by the driver. By default, the driver maps the following information schema types to SQL_VARCHAR:

- cardinal_number

- `character_data`
- `sql_identifier`
- `time_stamp`
- `yes_or_no`

Using the XML Data Type

By default, PostgreSQL returns XML data to the driver encoded as UTF-8. To avoid data loss, an application must bind XML data as `SQL_C_WCHAR`. The driver then returns the data as either UTF-8 or UTF-16, depending on platform and application settings. If the application binds XML data as `SQL_C_CHAR`, the driver converts it to the client character encoding, possibly causing data loss or corruption. To prevent any conversion of XML data, the application must set the option [XMLDescribeType \(XDT\)](#) to `SQL_LONGVARBINARY (-4)` and bind the data as `SQL_C_BINARY`.

Unicode Support

The PostgreSQL Wire Protocol driver automatically determines whether the PostgreSQL database is a Unicode database.

Advanced Features

The driver supports the following advanced features:

- Failover
- Security
- Connection Pooling

Failover

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Security

The driver supports authentication and encryption. Security connection options are located on the [Security tab](#) of the driver Setup dialog box. See [Using Security](#) on page 89 for a general description of security and its implementation.

Connection Pooling

The driver supports connection pooling and its related connection options. Connection pooling connection options are located on the [Pooling tab](#) of the driver Setup dialog box. See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling and its implementation.

User-defined Functions' Results

PostgreSQL provides functionality to create user-defined functions. PostgreSQL does not define a call mechanism for invoking a user-defined function. User-defined functions must be invoked via a SQL statement. For example, given a function defined as:

```
CREATE table foo (intcol int, varcharcol varchar(123))
CREATE or REPLACE FUNCTION insertFoo
  (IN idVal int, IN nameVal varchar) RETURNS void
  AS $$
    insert into foo values ($1, $2);
  $$
LANGUAGE SQL;
```

must be invoked natively as:

```
SELECT * FROM insertFoo(100, 'Mark')
```

even though the function does not return a value or results. The Select SQL statement returns a result set that has one column named insertFoo and no row data.

The PostgreSQL Wire Protocol driver supports invoking user-defined functions using the ODBC call Escape. The previously described function can be invoked using:

```
{call insertFoo(100, 'Mark')}
```

PostgreSQL functions return data from functions as a result set. If multiple output parameters are specified, the values for the output parameters are returned as columns in the result set. For example, the function defined as:

```
CREATE or REPLACE FUNCTION addValues(in v1 int, in v2 int)
  RETURNS int
  AS $$
    SELECT $1 + $2;
  $$
LANGUAGE SQL;
```

returns a result set with a single column of type SQL_INTEGER, whereas the function defined as:

```
CREATE or REPLACE FUNCTION selectFooRow2
  (IN idVal int, OUT id int, OUT name varchar)
  AS $$
    select intcol, varcharcol from foo where intcol = $1;
  $$
LANGUAGE SQL
```

returns a result set that contains two columns, a SQL_INTEGER id column and a SQL_VARCHAR name column.

In addition, when calling PostgreSQL functions that contain output parameters, the native syntax requires that the output parameter values be omitted from the function call. This, in addition to output parameter values being returned as a result set, makes the PostgreSQL behavior of calling functions different from most other databases.

The PostgreSQL Wire Protocol driver provides a mechanism that makes the invoking of functions more consistent with how other databases behave. In particular, the PostgreSQL Wire Protocol driver allows parameter markers for output parameters to be specified in the function argument list when the Escape call is used. The driver allows buffers to be bound to these output parameters. When the function is executed, the output parameters are removed from the argument list sent to the server. The driver extracts the output parameter values from the result set returned by the server and updates the bound output parameter buffers with those values. For example, the function `selectFooRow2` described previously can be invoked as:

```
sql = L"{call selectFooRow2(?, ?, ?)}";
retVal = SQLPrepare(hPrepStmt, sql, SQL_NTS);
retVal = SQLBindParameter(
    hPrepStmt, 1, SQL_PARAM_INPUT, SQL_C_LONG,
    SQL_INTEGER, 0, 0, &idBuf, 0, &idInd);
retVal = SQLBindParameter(
    hPrepStmt, 2, SQL_PARAM_OUTPUT, SQL_C_LONG,
    SQL_INTEGER, 0, 0, &idBuf2, 4, &idInd2);
retVal = SQLBindParameter(
    hPrepStmt, 3, SQL_PARAM_OUTPUT, SQL_C_WCHAR,
    SQL_VARCHAR, 30, 0, &nameBuf, 123, &nameInd);
retVal = SQLExecute(hPrepStmt);
```

The values of the `id` and `name` output parameters are returned in the `idBuf2` and `nameBuf` buffers.

If output parameters are bound to a function call, the driver returns the output parameters in the bound buffers. An error is returned if the number of output parameters bound when the function is executed is less than the number of output parameters defined in the function. If no output parameters are bound to a function call, the driver returns the output parameters as a result set.

PostgreSQL can also return results from a function as a refcursor. There can be, at most, one refcursor per result; however, a function can return multiple results where each result is a refcursor. A connection option defines how the driver handles refcursors. See [Fetch Ref Cursors](#) on page 409 for details about this option.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

PostgreSQL supports isolation level 0 (read uncommitted), level 1 (read committed), 2 (Repeatable read), and level 3 (serializable). PostgreSQL supports record-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the core SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

In addition, the following functions are supported:

- `SQLColumnPrivileges`

- SQLDescribeParam (if EnableDescribeParam=1)
- SQLForeignKeys
- SQLTablePrivileges

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The PostgreSQL Wire Protocol driver supports multiple connections and multiple statements per connection.

Using Arrays of Parameters

PostgreSQL supports returning a set of output parameters or return values, but no ODBC standard method exists for returning arrays of output parameters or return values. If the call Escape is used to invoke a function that returns a set of output parameters and buffers are bound for those output parameters, the PostgreSQL Wire Protocol driver places the first set of output parameters in the bound buffers. If no output parameters are bound for functions that return a set of results or output parameters, the driver returns a result set with a row for each set of output parameters.

The Progress OpenEdge Wire Protocol Driver

The DataDirect Connect *for* ODBC Progress OpenEdge® Wire Protocol driver (the Progress OpenEdge Wire Protocol driver) supports Progress OpenEdge database systems and services.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The Progress OpenEdge Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the Windows, UNIX, and Linux environments supported by this driver.

See the README file shipped with your DataDirect product for the file name of the Progress OpenEdge Wire Protocol driver.

Driver Requirements

There are no client requirements for the Progress OpenEdge Wire Protocol driver.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 444 and [Connection Option Descriptions for OpenEdge Wire Protocol](#) on page 445 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX odbc.ini File

UNIX[®]

On UNIX and Linux, data sources are configured and modified by editing the system information file (by default, odbc.ini) and storing default connection values there. See [Configuring the Product on UNIX/Linux](#) on page 111 for detailed information about the specific steps needed to set up the UNIX and Linux environments and to configure a data source.

[Connection Option Descriptions for OpenEdge Wire Protocol](#) on page 445 lists driver connection string attributes that must be used in the odbc.ini file to set the value of connection options. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (OpenEdge)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Progress OpenEdge data source:

1. Start the ODBC Administrator. On Windows, start the ODBC Administrator by selecting its icon from the DataDirect Connect program group.
2. Select a tab:
 - **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.



- **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

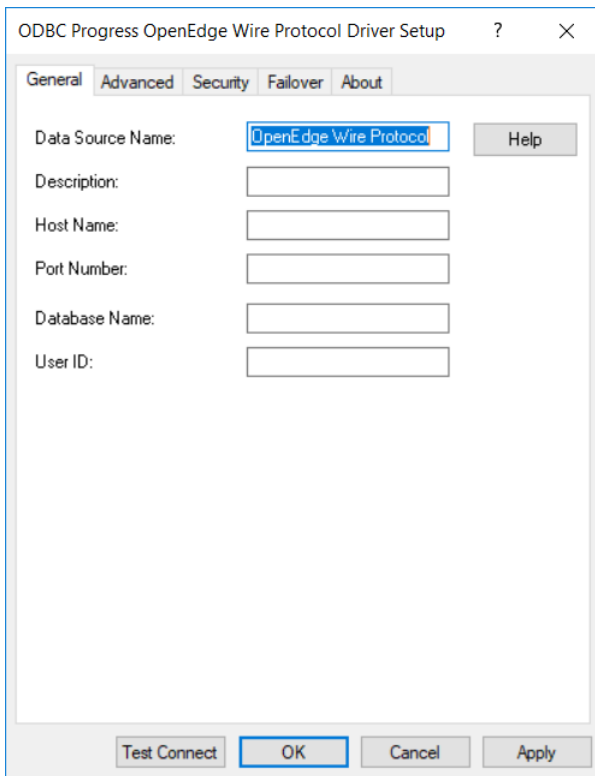
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 37: General tab



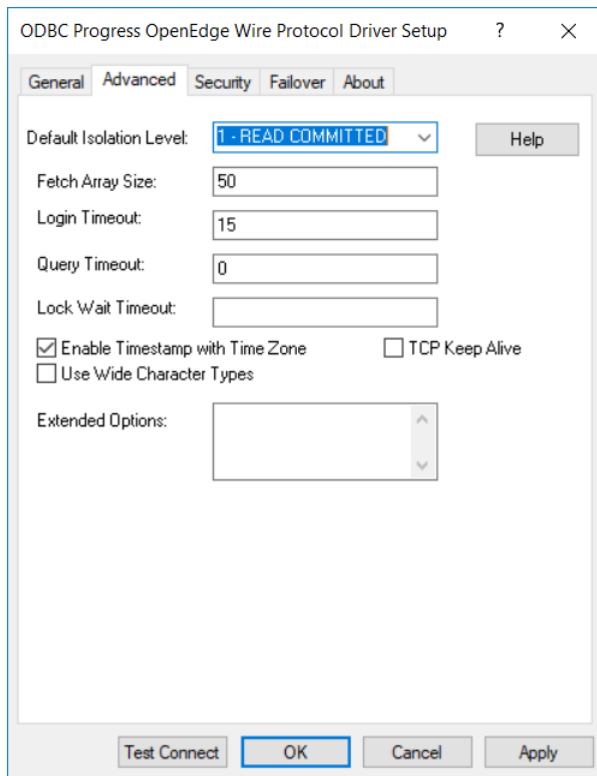
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

3. Provide values for the options on this tab in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 452	None
Description on page 454	None
Host Name on page 458	None
Port Number on page 463	None
Database Name on page 453	None
User ID on page 467	None

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 38: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Default Isolation Level on page 453	1- READ COMMITTED
Fetch Array Size on page 458	50
Login Timeout on page 461	15
Query Timeout on page 463	0
Enable Timestamp with Timezone on page 454	Enabled
TCP Keep Alive on page 465	Disabled
Use Wide Character Types on page 467	Disabled
IANAAppCodePage on page 460 UNIX ONLY	4 (ISO 8559-1 Latin 1)



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.

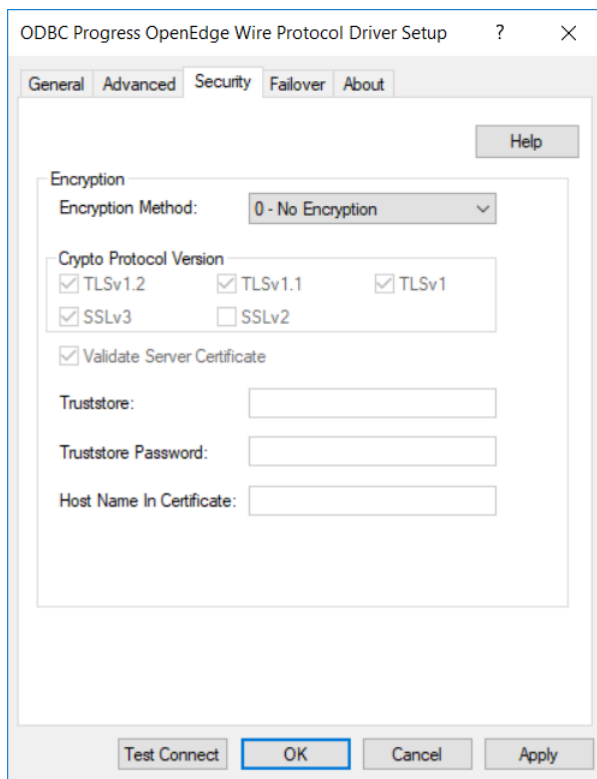


Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- Optionally, click the **Security** tab to specify security data source settings.

Figure 39: Security tab



See [Using Security](#) on page 89 for a general description of authentication and encryption and their configuration requirements.

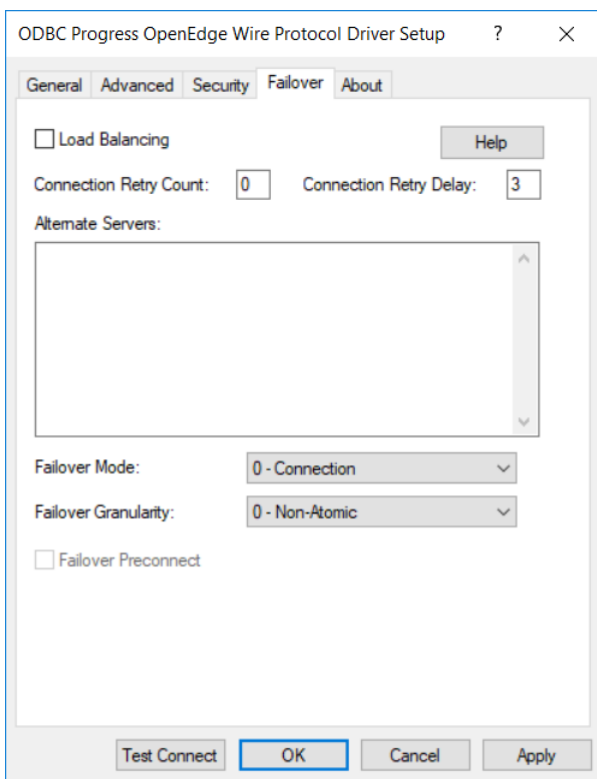
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Security	Default
Encryption Method on page 455	0 (No Encryption)

Connection Options: Security	Default
Crypto Protocol Version on page 450	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Validate Server Certificate on page 468	Enabled
Truststore on page 465	None
Truststore Password on page 466	None
HostName In Certificate on page 459	None

6. Optionally, click the **Failover** tab to specify failover data source settings.

Figure 40: Failover tab



See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 461	Disabled
Connection Retry Count on page 449	0
Connection Retry Delay on page 449	3
Alternate Servers on page 448	None

Connection Options: Failover	Default
Failover Mode on page 456	0 (Connection)
Failover Granularity on page 456	0 (Non-Atomic)
Failover Preconnect on page 457	Disabled

7. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(OpenEdge\)](#) on page 445 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

8. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify `attribute=value` pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={[driver_name]}[;attribute=value[;attribute=value]...]
```

[Connection Option Descriptions for OpenEdge Wire Protocol](#) on page 445 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Progress OpenEdge is:

```
DSN=PROGRESS;DB=PAYROLL;UID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=ProgOpen.dsn;DB=PAYROLL;UID=JOHN;PWD=XYZZY
```

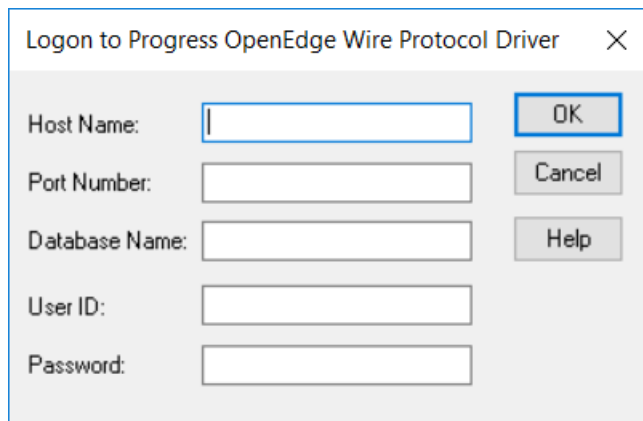
A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Progress OpenEdge Wire Protocol};DB=PAYROLL;UID=JOHN;  
PWD=XYZZY;HOST=LOCALHOST;PORT=2055
```

Using a Logon Dialog Box (OpenEdge)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Figure 41: Logon to Progress OpenEdge Wire Protocol Driver dialog box



In this dialog box, provide the following information:

1. In the Host Name field, type the name of the system where the database is stored.
2. Type the Port Number setup for the database listener process.
3. Type the name of the database to which you want to connect.
4. Type your user name.
5. Type your password.
6. Click **OK** to complete the logon.

Connection Option Descriptions for OpenEdge Wire Protocol

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Progress OpenEdge Wire Protocol driver.

Table 31: Progress OpenEdge Wire Protocol Attribute Names

Attribute (Short Name)	Default
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
AlternateServers (ASVR)	None
ArraySize (AS)	None
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1, SSLv3
CryptoLibName (CLN)	Empty string
Database (DB)	None
DataSourceName (DSN)	None
DefaultIsolationLevel (DIL)	1- READ COMMITTED
Description (n/a)	None
EnableTimestampwithTimezone (ETWT)	1 (enabled)
EncryptionMethod (EM)	0 (No Encryption)
FailoverGranularity (FG)	0 (Non-Atomic)
FailoverMode (FM)	0 (Connection)
FailoverPreconnect (FP)	0 (Disabled)
Fetch Array Size	50
HostName (HOST)	None
HostNameInCertificate (HNIC)	None
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
KeepAlive (KA)	Disabled
LoadBalancing (LB)	0 (Disabled)
LoginTimeout (LT)	15
LogonID (UID)	None

Attribute (Short Name)	Default
Password (PWD)	None
PortNumber (PORT)	None
QueryTimeout (QT)	0 (Disabled)
SSLibName (SLN)	Empty string
Truststore (TS)	None
TruststorePassword (TSP)	None
UseWideCharacterTypes (UWCT)	0 (disabled)
ValidateServerCertificate (VSC)	1 (enabled)

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

`latest | openssl_version_number[[, openssl_version_number] ...]`

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to `openssl_version_number`, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the `CryptoLibName` and `SSLLibName` options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

1.1.1,1.0.2

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

[Advanced tab](#)

Alternate Servers

Attribute

AlternateServers (ASVR)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
(HostName=hostvalue:PortNumber=portvalue:Database=databasevalue[, . . .])
```

You must specify the host name, port number, and database name of each alternate server.

Notes

- An alternate server address in IPv6 format must be enclosed in double quotation marks.

Example

The following Alternate Servers values define two alternate database servers for connection failover:

```
AlternateServers=(HostName=123.456.78.90:PortNumber=5177:Database=PAYROLL1,  
HostName=223.456.78.90:PortNumber=5178:Database=PAYROLL2)
```


Default

None

GUI Tab

[Failover tab](#)

Connection Retry Count**Attribute**

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x, the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay**Attribute**

ConnectionRetryDelay (CRD)

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | *x*

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to *x*, the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (EncryptionMethod=1). When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, the connection fails and the driver returns an error.

Valid Values

cryptographic_protocol [, *cryptographic_protocol*]...

where:

cryptographic_protocol

is one of the following cryptographic protocols:

TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Default

```
TLSv1.2, TLSv1.1, TLSv1
```

GUI Tab

[Security tab](#)

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Progress\DataDirect\Connect64_for_ODBC_71\  
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLLibName](#) on page 464

Data Source Name

Attribute

`DataSourceName` (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or `odbc.ini` file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database Name

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where:

database_name

is the name of a valid database.

Default

None

GUI Tab

[General tab](#)

Default Isolation Level

Attribute

DefaultIsolationLevel (DIL)

Purpose

The method by which locks on data in the database are acquired and released.

Valid Values

READ COMMITTED | READ UNCOMMITTED | REPEATABLE READ | SERIALIZABLE

0 | 1 | 2 | 3

Behavior

If set to 0 (READ_UNCOMMITTED), other processes can be read from the database. Only modified data is locked and is not released until the transaction ends.

If set to 1 (READ_COMMITTED) other processes can change a row that your application has read if the cursor is not on the row you want to change. This level prevents other processes from changing records that your application has changed until your application commits them or ends the transaction.

If set to 2 (REPEATABLE_READ), other processes are prevented from accessing data that your application has read or modified. All read or modified data is locked until transaction ends.

If set to 3 (SERIALIZABLE), other processes are prevented from changing records that are read or changed by your application (including phantom records) until your program commits them or ends the transaction. This level prevents the application from reading modified records that have not been committed by another process. If your application opens the same query during a single unit of work under this isolation level, the results table will be identical to the previous table; however, it can contain updates made by your application.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Valid Values

Default

1 - READ COMMITTED

GUI Tab

[Advanced tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the `odbc.ini` file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable Timestamp with Timezone

Attribute

EnableTimestampwithTimezone (ETWT)

Purpose

Determines whether the driver exposes timestamps with timezones to the application.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver exposes timestamps with timezones to the application.

If set to 0 (Disabled), timestamps with timezones are not exposed to the application.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Encryption Method

Attribute

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server. If the specified encryption method is not supported by the database server, the connection fails and the driver returns an error.

Valid Values

0 | 1

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using SSL. If the server supports protocol negotiation, the driver and server negotiate the use of TLS v1, SSL v3, or SSL v2 in that order.

Notes

- This connection option can affect performance.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

See Also

See [Performance Considerations](#) on page 468 for details.

Failover Granularity

Attribute

FailoverGranularity (FG)

Purpose

Determines whether the driver fails the entire failover process or continues with the process if errors occur while trying to reestablish a lost connection.

This option applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select).

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 (Non-Atomic), the driver continues with the failover process and posts any errors on the statement on which they occur.

If set to 1 (Atomic) the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

If set to 2 (Atomic Including Repositioning), the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

If set to 3 (Disable Integrity Check), the driver does not verify that the rows that were restored during the failover process match the original rows. This value applies only when Failover Mode is set to 2 (Select).

Default

0 (Non-Atomic)

GUI Tab

[Failover tab](#)

Failover Mode

Attribute

FailoverMode (FM)

Purpose

Specifies the type of failover method the driver uses.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Connection), the driver provides failover protection for new connections only.

If set to 1 (Extended Connection), the driver provides failover protection for new and lost connections, but not any work in progress.

If set to 2 (Select), the driver provides failover protection for new and lost connections. In addition, it preserves the state of work performed by the last Select statement executed.

Notes

- This connection option can affect performance.

Default

0 (Connection)

GUI Tab

[Failover tab](#)

See Also

See [Performance Considerations](#) on page 212 for details.

Failover Preconnect

Attribute

FailoverPreconnect (FP)

Purpose

Specifies whether the driver tries to connect to the primary and an alternate server at the same time.

This attribute applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select) and at least one alternate server is specified.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver tries to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection. This value provides the best performance, but your application typically experiences a short wait while the failover connection is attempted.

If set to 1 (Enabled), the driver tries to connect to the primary and an alternate server at the same time. This can be useful if your application is time-sensitive and cannot absorb the wait for the failover connection to succeed.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Fetch Array Size

Attribute

ArraySize (AS)

Purpose

The number of rows the driver retrieves from the server for a fetch. This is not the number of rows given to the user.

This connection option can affect performance. See [Performance Considerations](#) on page 212 for details.

Valid Values

x

where:

x

is a positive integer specifying the number of bytes.

Notes

- This connection option can affect performance.

Default

50

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 212 for details.

Host Name

Attribute

HostName (HOST)

Purpose

The name or the IP address of the server to which you want to connect.

Valid Values

server_name | *IP_address*

where:

server_name

is the name of the server to which you want to connect.

IP_address

is the IP address of the server to which you want to connect.

The IP address can be specified in either IPv4 or IPv6 format. See [Using IP Addresses](#) on page 67 for details about these formats.

Default

None

GUI Tab

[General tab](#)

HostName In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

host_name | *#SERVERNAME#*

where:

host_name

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If set to a host name, the driver examines the subjectAltName values included in the certificate. If a dnsName value is present in the subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the dnsName value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the dnsName value.

If no subjectAltName values exist or a dnsName value is not in the list of subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the commonName part of the Subject name in the certificate. The commonName typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the commonName. If multiple commonName parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the commonName parts.

If set to #SERVERNAME#, the driver compares the host server name specified as part of a data source or connection string to the dnsName or the commonName value.

Default

None

GUI Tab

[Security tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

N/A

Load Balancing**Attribute**

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Login Timeout**Attribute**

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x, the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

15

GUI Tab

[Advanced tab](#)

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Purpose

The port number of the server listener.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your database administrator for the correct number.

Default

None

GUI Tab

[General tab](#)

Query Timeout

Attribute

QueryTimeout (QT)

Purpose

The number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the SQL_ATTR_QUERY_TIMEOUT statement attribute on the SQLSetStmtAttr() function.

Valid Values

-1 | 0 | *x*

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to -1, the query does not time out. The driver silently ignores the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to 0, the query does not time out, but the driver responds to the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to *x*, all queries time out after the specified number of seconds unless the application overrides this value by setting the SQL_ATTR_QUERY_TIMEOUT attribute.

Default

0

GUI Tab

[Advanced tab](#)

SSLLibName

Attribute

SSLLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\ODBC_71\
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the CryptoLibName and SSLLibName options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
SSLLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 451

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Truststore

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used.

Valid Values

truststore_directory\filename

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Notes

- The truststore and keystore files may be the same file.

Default

None

GUI Tab

[Security tab](#)

Truststore Password

Attribute

TruststorePassword (TSP)

Purpose

The password that is used to access the truststore file when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

- The truststore and keystore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Use Wide Character Types**Attribute**

UseWideCharacterTypes (UWCT)

Purpose

A value that determines whether character data types are described to the application as SQL_CHAR or SQL_WCHAR when connected to a Unicode database.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), character data types are described to the application as SQL_CHAR.

If set to 1 (Enabled), character data types are described to the application as SQL_WCHAR.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

User ID**Attribute**

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[General tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Truststore information is specified using the Trust Store and Trust Store Password options.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Default

1 (Enabled)

GUI Tab

[Security tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Encryption Method (EncryptionMethod): Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) that is required to encrypt and decrypt data.

Failover Mode (FailoverMode): Although high availability that replays queries after a failure provides increased levels of protection, it can adversely affect performance because of increased overhead.

Fetch Array Size (ArraySize): Reducing the number of round trips on the network to the approximate number of rows being fetched increases performance. For example, if your application normally fetches 200 rows, it is more efficient for the driver to fetch 200 rows at one time over the network than to fetch 50 rows at a time during four round trips over the network.

Notes

- The ideal setting for your application will vary. To calculate the ideal setting for this option, you must know the size in bytes of the rows that you are fetching and the size in bytes of your Network Packet. Then, you must calculate the number of rows that will fit in your Network Packet, leaving space for packet overhead. For example, suppose your Network Packet size is 1024 bytes and the row size is 8 bytes. Dividing 1024 by 8 equals 128; however, the ideal setting for Fetch Array Size is 127, not 128, because the number of rows times the row size must be slightly smaller than the Network Packet size.

Data Types

The following table shows how the Progress OpenEdge data types are mapped to the standard ODBC data types.

Table 32: Progress OpenEdge Data Types

OpenEdge	ODBC
Bigint	SQL_BIGINT
Binary	SQL_BINARY
Bit	SQL_BIT
Blob	SQL_LONGVARBINARY
Char	SQL_CHAR
Clob	SQL_LONGVARCHAR
Date	SQL_TYPE_DATE
Decimal	SQL_DECIMAL
Double precision	SQL_DOUBLE
Float	SQL_FLOAT
Integer	SQL_INTEGER
Lvarbinary	SQL_LONGVARBINARY
Lvarchar	SQL_LONGVARCHAR
Numeric	SQL_NUMERIC
Real	SQL_FLOAT

OpenEdge	ODBC
Smallint	SQL_SMALLINT
Time	SQL_TYPE_TIME
Timestamp	SQL_TYPE_TIMESTAMP
Timestamp with Time Zone	SQL_CHAR
Tinyint	SQL_TINYINT
Varbinary	SQL_VARBINARY
Varchar	SQL_VARCHAR

See [Retrieving Data Type Information](#) on page 72 for more information about data types.

Unicode Support

When connected to a Unicode database, the Progress OpenEdge Wire Protocol driver supports the Unicode data types listed in the following table, in addition to standard ODBC data types listed in [Data Types](#) on page 213. The Use Wide Character Types connection string option must be enabled.

Progress OpenEdge Data Type	Mapped to . . .
Char	SQL_WCHAR
Varchar	SQL_WVARCHAR

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Advanced Features

The driver supports the following advanced features:

- Failover
- Security

Failover

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Security

The driver supports data security by providing both user authentication and SSL data encryption, including Kerberos authentication and SSL data encryption. See [Using Security](#) on page 89 for a general description of authentication and encryption and its configuration requirements.

You configure the driver for data security on the [Security tab](#) of the driver Setup dialog box. See the description of the Security tab under [Configuring and Connecting to Data Sources](#) on page 142 for specific implementations.

Isolation and Lock Levels Supported

Progress OpenEdge supports isolation level 0 (read uncommitted), isolation level 1 (read committed), isolation level 2 (repeatable read), and isolation level 3 (serializable).

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Grammar Support

The driver supports the core SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

The driver also supports the function `SQLSetPos`.

The driver supports the function `SQLCancel` for `SELECT` statements with OpenEdge V12.4 and higher. If a `SELECT` statement is cancelled during the first or a subsequent fetch operation, the driver returns a "query aborted" exception. This function can be used by a thread to cancel a statement that is being executed by another thread. One or more statements may be cancelled if the method is called on a statement object that is executing multiple statements simultaneously and the driver may not return expected results. The driver supports this function for ODBC 3.x applications only.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The Progress OpenEdge database system supports multiple connections and multiple statements per connection.

The SQL Server Wire Protocol Driver

Note: This section documents the features and functionality of the 7.1 version of the driver. For the current version of the driver, visit Progress DataDirect Connectors Documentation page: <https://docs.progress.com/datadirect-connectors/page/DataDirect-Connectors-by-data-source.html>.

The DataDirect Connect *for* ODBC and DataDirect Connect64 *for* ODBC SQL Server Wire Protocol driver (the SQL Server Wire Protocol driver) each support the following databases and services:

Cloud:

- Microsoft Azure Synapse Analytics
- Microsoft Windows Azure SQL Database

On premise:

- Microsoft Analytics Platform System
- Microsoft SQL Server

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The SQL Server Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect product for the file name of the SQL Server Wire Protocol driver.

Driver Requirements

The driver has no client requirements.

The SQL Server Wire Protocol driver connects via TCP/IP. TCP/IP connections must be configured on the Windows server on which the Microsoft SQL Server database resides.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 485 and [Connection Option Descriptions for SQL Server Wire Protocol](#) on page 486 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). You can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions for SQL Server Wire Protocol](#) on page 486 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (SQL Server)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.

UNIX® On UNIX and Linux, data sources are stored in the `odbc.ini` file.


When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Microsoft SQL Server data source:

1. Start the ODBC Administrator by selecting its icon from the DataDirect Connect program group.
2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

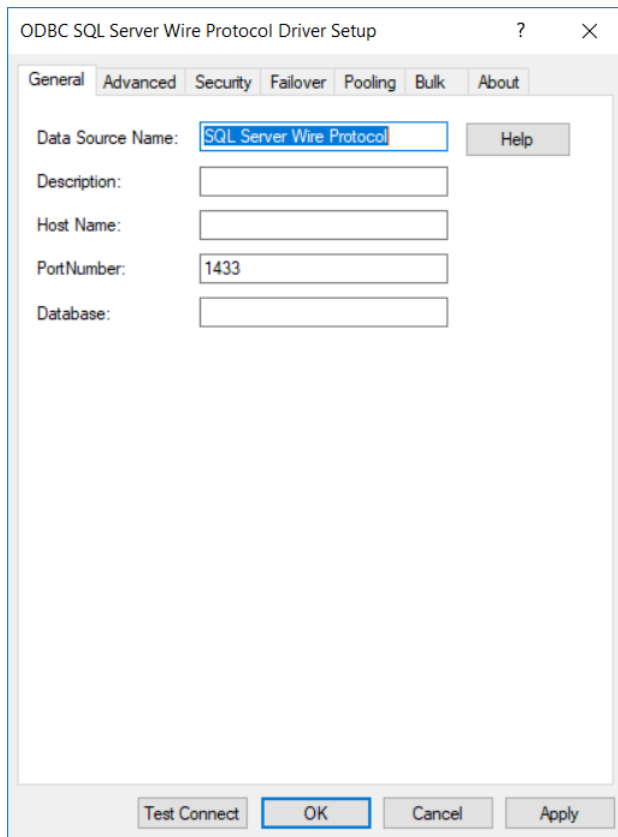
If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.
 - If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 42: General tab



Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 504	None
Description on page 505	None
Host Name on page 513	None
Port Number on page 522	1433
Database on page 504	None

3. Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 43: Advanced tab

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Application Name on page 493	None
Initialization String on page 515	None
Language on page 516	None
Packet Size on page 521	-1
Workstation ID on page 532	None
Login Timeout on page 518	15
Query Timeout on page 525	0
Report Codepage Conversion Errors on page 526	0 (Ignore Errors)
XML Describe Type on page 532	-10
Application Intent on page 492	0 (READWRITE)

Connection Options: Advanced	Default
AnsiNPW on page 492	Enabled
Application Using Threads on page 494	Enabled
Always Report Trigger Results on page 491	0 (Disabled)
Enable Quoted Identifiers on page 506	Disabled
Fetch TWFS as Time on page 511	Disabled
Fetch TSWTZ as Timestamp on page 510	Disabled
Use Snapshot Transactions on page 530	Disabled
TCP Keep Alive on page 528	Disabled
IANAAppCodePage on page 514 UNIX ONLY	4 (ISO 8559-1 Latin-1)



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.

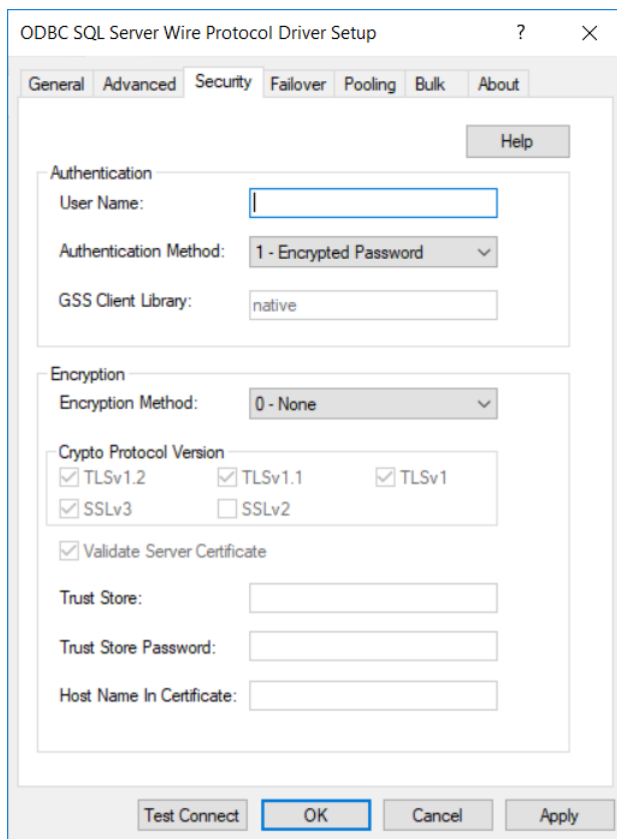


Translate : Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

4. Optionally, click the **Security** tab to specify additional data source settings.

Figure 44: Security tab



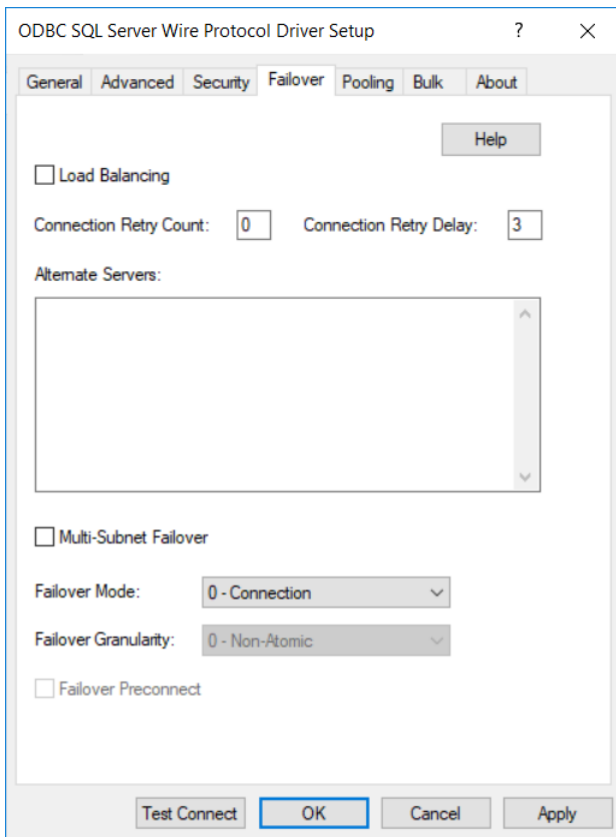
See [Using Security](#) on page 89 for a general description of authentication and encryption and their configuration requirements.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Security	Default
User Name on page 531	None
Authentication Method on page 494	1 (Encrypt Password)
GSS Client Library on page 512	native
Encryption Method on page 507	0 (No Encryption)
Crypto Protocol Version on page 502	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Validate Server Certificate on page 531	Enabled
Trust Store on page 529	None
Trust Store Password on page 529	None
Host Name In Certificate on page 514	None

5. Optionally, click the **Failover** tab to specify additional data source settings.

Figure 45: Failover tab



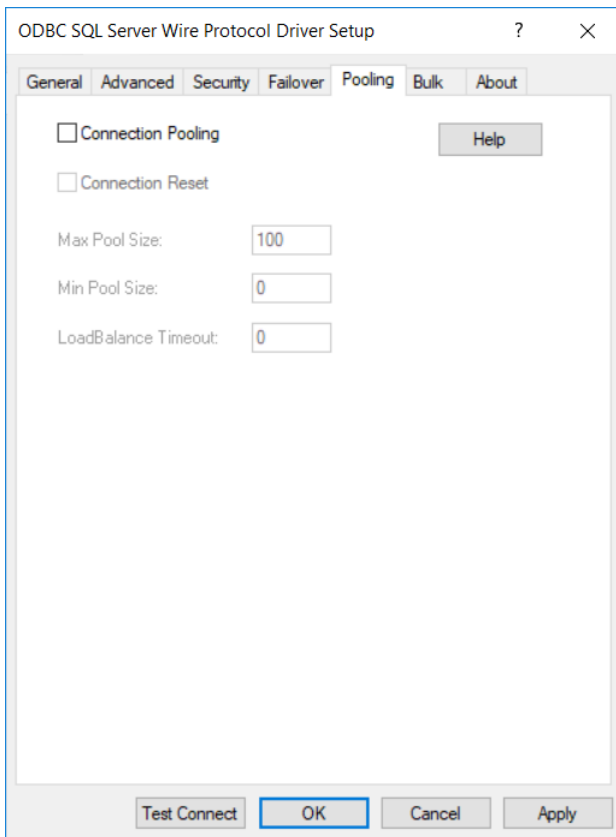
See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 517	Disabled
Connection Retry Count on page 500	0
Connection Retry Delay on page 501	3
Alternate Servers on page 490	None
Multi-Subnet Failover on page 520	0 (Disabled)
Failover Mode on page 509	0 (Connection)
Failover Granularity on page 508	0 (Non-Atomic)
Failover Preconnect on page 509	Disabled

6. Optionally, click the **Pooling** tab to specify additional data source settings.

Figure 46: Pooling tab



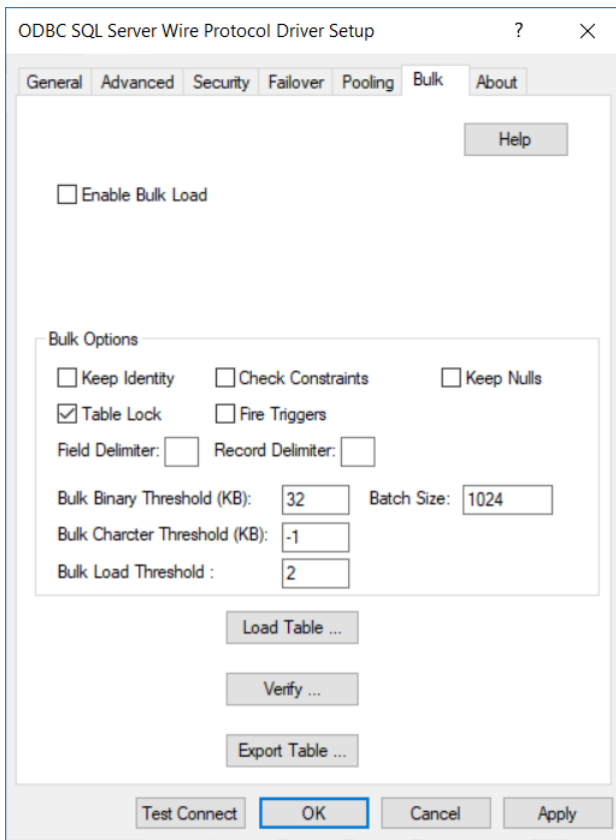
See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 499	Disabled
Connection Reset on page 500	Disabled
Max Pool Size on page 519	100
Min Pool Size on page 519	0
Load Balance Timeout on page 516	0

- Optionally, click the **Bulk** tab to specify additional data source settings.

Figure 47: Bulk tab



See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect Bulk Load.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Bulk	Default
Enable Bulk Load on page 506	Disabled
Bulk Options on page 498 Individual item descriptions are grouped under the Bulk Option description.	Only Table Lock enabled
Field Delimiter on page 511	None
Record Delimiter on page 526	None
Bulk Binary Threshold on page 496	32
Bulk Character Threshold on page 496	-1
Batch Size on page 495	1024
Bulk Load Threshold on page 497	2

If your application is already coded to use parameter array batch functionality, you can leverage DataDirect Bulk Load features through the Enable Bulk Load connection option. Enabling this option automatically converts the parameter array batch operation to use the database bulk load protocol.

If you are not using parameter array batch functionality, you can export data to a bulk load data file, verify the metadata of the bulk load configuration file against the structure of the target table, and bulk load data to a table. Use the following steps to accomplish these tasks.

- a) To export data from a table to a bulk load data file, click **Export Table** from the Bulk tab. The Export Table dialog box appears.

Figure 48: ODBC SQL Server Wire Protocol Export Table Driver Setup dialog box

Both a bulk data file and a bulk configuration file are produced by exporting a table. The configuration file has the same name as the data file, but with an XML extension. See [Using DataDirect Bulk Load](#) on page 101 for details about these files.

The bulk export operation can create a log file and can also export to external files. See [External Overflow Files](#) on page 108 for more information. The export operation can be configured such that if any errors or warnings occur:

- The operation always completes.
- The operation always terminates.
- The operation terminates after a certain threshold of warnings or errors is exceeded.

Table Name: A string that specifies the name of the source database table containing the data to be exported.

Export Filename: A string that specifies the path (relative or absolute) and file of the bulk load data file to which the data is to be exported. It also specifies the file name of the bulk configuration file. The file name must be the fully qualified path to the bulk data file. These files must not already exist; if one of both of them already exists, an error is returned.

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. The log file is created if it does not exist. The file name must be the fully qualified path to the log file. Events logged to this file are:

- Total number of rows fetched
- A message for each row that failed to export
- Total number of rows that failed to export
- Total number of rows successfully exported

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not supply a value for Log Filename, no log file is created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

Code Page: A value that specifies the code page value to which the driver must convert all data for storage in the bulk data file. See [Character Set Conversions](#) on page 108 for more information.

The default value on Windows is the current code page of the machine. On UNIX/Linux, the default value is 4 (ISO 8559-1 Latin-1).

Click **Export Table** to connect to the database and export data to the bulk data file or click **Cancel**.

To verify the metadata of the bulk load configuration file against the structure of the target database table, click **Verify** from the Bulk tab. See [Verification of the Bulk Load Configuration File](#) on page 106 for details. The Verify dialog box appears.

Figure 49: ODBC SQL Server Wire Protocol Verify Driver Setup dialog box

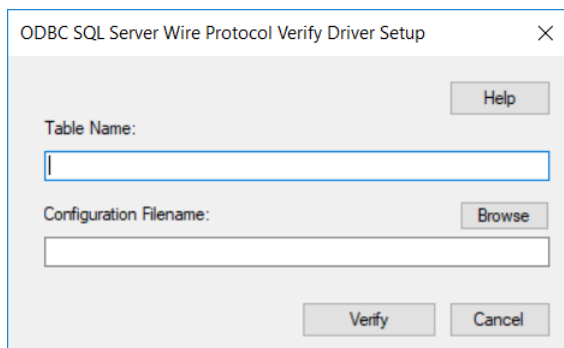


Table Name: A string that specifies the name of the target database table into which the data is to be loaded.

Configuration Filename: A string that specifies the path (relative or absolute) and file name of the bulk configuration file. The file name must be the fully qualified path to the configuration file.

Click **Verify** to verify table structure or click **Cancel**.

- b) To load data from the bulk data file to a database table, click **Load Table** from the Bulk tab. The Load File dialog box appears.

Figure 50: ODBC SQL Server Wire Protocol Load File Driver Setup dialog box

The load operation can create a log file and can also create a discard file that contains rows rejected during the load. The discard file is in the same format as the bulk load data file. After fixing reported issues in the discard file, the bulk load can be reissued using the discard file as the bulk load data file.

The export operation can be configured such that if any errors or warnings occur:

- The operation always completes.
- The operation always terminates.
- The operation terminates after a certain threshold of warnings or errors is exceeded.

If a load fails, the Load Start and Load Count options can be used to control which rows are loaded when a load is restarted after a failure.

Table Name: A string that specifies the name of the target database table into which the data is loaded.

Load Data Filename: A string that specifies the path (relative or absolute) and file name of the bulk data file from which the data is loaded. The file name must be the fully qualified path to the bulk data file.

Configuration Filename: A string that specifies the path (relative or absolute) and file name of the bulk configuration file.

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. The file name must be the fully qualified path to the log file. Specifying a value for Log Filename creates the file if it does not already exist. Events logged to this file are:

- Total number of rows read
- Message for each row that failed to load

- Total number of rows that failed to load
- Total number of rows successfully loaded

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Log Filename, no log file is created.

Discard Filename: A string that specifies the path (relative or absolute) and file name of the bulk discard file. The file name must be the fully qualified path to the discard file. Any row that cannot be inserted into database as result of bulk load is added to this file, with the last row rejected added to the end of the file.

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Discard Filename, a discard file is not created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Load Start: A value that specifies the first row to be loaded from the data file. Rows are numbered starting with 1. For example, when Load Start is 10, the first 9 rows of the file are skipped and the first row loaded is row 10. This option can be used to restart a load after a failure.

The default value is 1.

Read Buffer Size (KB): A value that specifies the size, in KB, of the buffer that is used to read the bulk data file for a bulk load operation.

The default value is 2048.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

Load Count: A value that specifies the number of rows to be loaded from the data file. The bulk load operation loads rows up to the value of Load Count from the file to the database. It is valid for Load Count to specify more rows than exist in the data file. The bulk load operation completes successfully when either the number of rows specified by the Load Count value has been loaded or the end of the data file is reached. This option can be used in conjunction with Load Start to restart a load after a failure.

The default value is the maximum value for SQLULEN. If set to 0, no rows are loaded.

Click **Load Table** to connect to the database and load the table or click **Cancel**.

At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A Logon dialog box appears; see [Using a Connection String](#) on page 485 for details. Note that the information you enter in the Logon dialog box during a test connect is not saved.

- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
- If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

8. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={ }driver_name[ ] [ ;attribute=value[ ;attribute=value] ... ]
```

[Connection Option Descriptions for SQL Server Wire Protocol](#) on page 486 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Microsoft SQL Server is:

```
DSN=ACCOUNTING;DATABASE=ACCT
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=SQLServer.dsn;DATABASE=ACCT
```

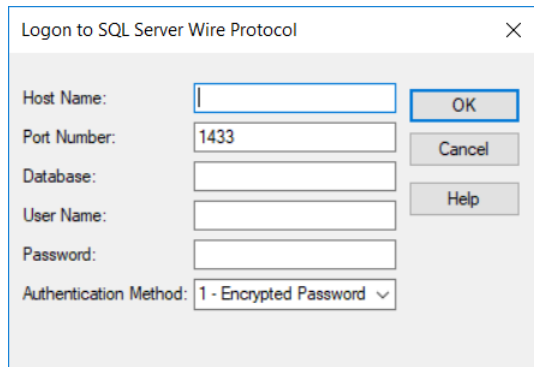
A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 SQL Server Wire Protocol};HOST=SQLServer1;PORT=1433;  
UID=JOHN;PWD=XYZZY;DB=SQLSdb1
```

Using a Logon Dialog Box (SQL Server)

Some ODBC applications display a Logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Note: The Logon Dialog is not displayed if Authentication Mode has previously been set to Kerberos and the Host Name is specified in the data source.



In the Logon dialog box, provide the following information:

1. Type an IP address in Host Name in following format: *IP_address*. For example, you can enter 199.226.224.34.

The IP address can be specified in IPv4 on Windows, and in either IPv4 or IPv6 format, or a combination of the two, on UNIX. See [Using IP Addresses](#) on page 67 for details about these formats.

If your network supports named servers, you can specify an address as: *server_name*. For example, you can enter *SSserver*.

To specify a named instance of Microsoft SQL Server, use the format: *server_name\instance_name*. If only a server name is specified with no instance name, the driver uses the default instance on the server.

2. Type the Port Number of the server listener.
3. Type the name of the database to which you want to connect. If you do not specify a value, the default database that is defined by Microsoft SQL Server is used.
4. Type your Microsoft SQL Server login ID.
5. Type your password.
6. Select an Authentication Method:

If set to 1 - Encrypt Password, the driver sends the user ID in clear text and an encrypted password to the server for authentication.

If set to 4 - Kerberos, the driver uses Kerberos authentication. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

7. Click **OK** to complete the logon and to update the values in the Registry.

Connection Option Descriptions for SQL Server Wire Protocol

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the SQL Server Wire Protocol driver.

Table 33: SQL Server Wire Protocol Attribute Names

Attribute (Short Name)	Default
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
AlternateServers (ASRV)	None
AlwaysReportTriggerResults (ARTR)	0 (Disabled)
AnsiNPW (ANPW)	1 (Enabled)
ApplicationIntent (AI)	0 (READWRITE)
ApplicationName (APP)	None
ApplicationUsingThreads (AUT)	1 (Enabled)
Authentication Method	1 (Encrypt Password)
BulkBinaryThreshold (BBT)	32
BulkCharacterThreshold (BCT)	-1
BulkLoadBatchSize (BLBS)	1024
BulkLoadOptions (BLO)	2
BulkLoadFieldDelimiter (BLFD)	None
BulkLoadRecordDelimiter (BLRD)	None
BulkLoadThreshold (BLTH)	2
ConnectionReset (CR)	0 (Disabled)
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Database (DB)	None
DataSourceName (DSN)	None
Description (n/a)	None

Attribute (Short Name)	Default
Domain (DOM)	None
EnableBulkLoad (EBL)	0 (Disabled)
EnableQuotedIdentifiers (EQI)	0 (Disabled)
EncryptionMethod (EM)	0 (No Encryption)
FailoverGranularity (FG)	0 (Non-Atomic)
FailoverMode (FM)	0 (Connection)
FailoverPreconnect (FP)	0 (Disabled)
FetchTSWTZasTimestamp (FTSWTZAT)	0 (Disabled)
FetchTWFSasTime (FTWFSAT)	1 (Enabled)
GSSClient (GSSC)	native
HostName (HOST)	None
HostNameInCertificate (HNIC)	None
IANAAppCodePage (IACP) (UNIX ONLY)	4 (ISO 8559-1 Latin-1)
InitializationString (IS)	None
KeepAlive (KA)	Disabled
Language (LANG)	None
LoadBalanceTimeout (LBT)	0
LoadBalancing (LB)	0 (Disabled)
LoginTimeout (LT)	None
LogonID (UID)	None
MaxPoolSize (MXPS)	100
MinPoolSize (MNPS)	0
MultiSubnetFailover (MSF)	0 (Disabled)
PacketSize (PS)	-1
Password (PWD)	None
Pooling (POOL)	0 (disabled)

Attribute (Short Name)	Default
PortNumber (PORT)	1433
PRNGSeedFile (PSF) UNIX/Linux only	/dev/random
PRNGSeedSource (PSS) UNIX/Linux only	0 (File)
QueryTimeout (QT)	0
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
SSLibName (SLN)	Empty string
SnapshotSerializable (SS)	0 (Disabled)
Truststore (TS)	None
TruststorePassword (TSP)	None
User Name	None
ValidateServerCertificate (VSC)	1 (enabled)
WorkstationID (WSID)	None
XMLDescribeType (XDT)	-10

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

`latest | openssl_version_number[[,openssl_version_number] ...]`

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to *openssl_version_number*, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the `CryptoLibName` and `SSLlibName` options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

`1.1.1,1.0.2`

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

[Advanced tab](#)

Alternate Servers

Attribute

`AlternateServers (ASRV)`

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
(HostName=hostvalue:PortNumber=portvalue:Database=databasevalue[, . . .])
```

You must specify the host name, port number, and database name of each alternate server.

Notes

- An alternate server address in IPv6 format must be enclosed in double quotation marks.

Example

The following Alternate Servers value defines two alternate database servers for connection failover:

```
AlternateServers=(HostName=SqlsServer:PortNumber=1433:Database=Sqlsdb1,  
HostName=255.201.11.24:PortNumber=1434:Database=Sqlsdb2)
```

Default

None

GUI Tab

[Failover tab](#)

Always Report Trigger Results

Attribute

AlwaysReportTriggerResults (ARTR)

Purpose

Determines how the driver reports results that are generated by database triggers (procedures that are stored in the database and executed, or fired, when a table is modified). For Microsoft SQL Server 2005 and higher and Windows Azure SQL Database, this includes triggers that are fired by Data Definition Language (DDL) events.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns all results, including results that are generated by triggers. Multiple trigger results are returned one at a time. You can use the `SQLMoreResults` function to return individual trigger results. Warnings and errors are reported in the results as they are encountered.

If set to 0 (Disabled):

- For Microsoft SQL Server 2005 and higher and Windows Azure SQL Database, the driver does not report trigger results if the statement is a single INSERT, UPDATE, DELETE, CREATE, ALTER, DROP, GRANT, REVOKE, or DENY statement.
- For other Microsoft SQL Server databases, the driver does not report trigger results if the statement is a single INSERT, UPDATE, or DELETE statement.

When set to 0, the only result that is returned is the update count that is generated by the statement that was executed (if no errors occurred). Although trigger results are ignored, any errors and warnings that are generated by the trigger are reported. If errors are reported, the update count is not reported.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

AnsiNPW

Attribute

AnsiNPW (ANPW)

Purpose

Determines whether ANSI-defined behaviors are exposed. Setting this option has no effect on NULL concatenation for Windows Azure SQL Database or SQL Server versions higher than SQL Server 2012.

Valid Values

0 | 1

Behavior

When set to 1 (Enabled), the driver sets four ANSI-defined behaviors for handling NULL comparisons: NULLS, character data padding, warnings, and NULL concatenation.

When set to 0 (Disabled), ANSI-defined behaviors are not exposed. If the driver appears to be truncating trailing blank spaces, set this attribute to 0 (Disabled).

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Application Intent

Attribute

ApplicationIntent (AI)

Purpose

Specifies whether the driver connects to read-write databases or requests read-only routing to connect to read-only database replicas. Read-only routing only applies to connections in Microsoft SQL Server 2012 where Always On Availability Groups have been deployed.

Valid Values

0 | 1

Behavior

If set to 0 (READWRITE), the driver connects to a read-write node in the Always On environment.

If set to 1 (READONLY), the driver requests read-only routing and connects to the read-only database replicas specified by the server.

Notes

- By setting ApplicationIntent to 1 (ReadOnly) and querying read-only database replicas when possible, you can improve efficiency of your environment by reducing the work load on read-write nodes.
- When ApplicationIntent is enabled, the virtual network name (VNN) of the availability group listener must be specified in the Host Name connection option.

Default

0 (READWRITE)

GUI Tab

[Advanced tab](#)

Application Name

Attribute

ApplicationName (APP)

Purpose

The name the database uses to identify your application.

Valid Values

string

where:

string

is your application name.

Default

None

GUI Tab

[Advanced tab](#)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 533 for details.

Authentication Method

Attribute

AuthenticationMethod (AM)

Purpose

Specifies the method the driver uses to authenticate the user to the server when a connection is established. If the specified authentication method is not supported by the database server, the connection fails and the driver generates an error.

Valid Values

1 | 4 | 9 | 10 | 13

Behavior

If set to 1 (Encrypt Password), the driver sends the user ID in clear text and an encrypted password to the server for authentication.

If set to 4 (Kerberos Authentication), the driver uses Kerberos authentication. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

Setting this value to 4 also enables NTLMv2 and NTLMv1 authentication on Windows platforms. The protocol used for a connection is determined by the local security policy settings for the client.

(UNIX and Linux only) If set to 9 on Linux and UNIX platforms, the driver uses NTLMv1 or NTLMv2 authentication. The driver determines which protocol to use based on the size of the password provided. For passwords 14 bytes or less, the driver uses NTLMv1; otherwise, the driver uses NTLMv2. To connect to the database, users must supply the Windows User Id, Password, and, in some cases, Domain to the driver.

(UNIX and Linux only) If set to 10, the driver uses NTLMv2 authentication. To connect to the database, users must supply the Windows User Id, Password, and, in some cases, Domain to the driver.

If set to 13 (Active Directory Password), the driver uses Azure Active Directory (Azure AD) authentication when establishing a connection to an Azure SQL Database data store. All communications to the service are encrypted using SSL.

Important: Before enabling Azure AD authentication, see "Configuring Azure Active Directory Authentication" for requirements and additional information.

Notes

- NTLM single sign on is supported only on Windows.

Default

1 (Encrypt Password)

GUI Tab

[Security tab](#)

See Also

[Configuring Azure Active Directory Authentication](#) on page 537

Batch Size

Attribute

BulkLoadBatchSize (BLBS)

Purpose

The number of rows that the driver sends to the database at a time during bulk operations. This value applies to all methods of bulk loading.

Valid Values

0 | x

where:

x

is a positive integer that specifies the number of rows to be sent.

Default

1024

GUI Tab

[Bulk tab](#)

Bulk Binary Threshold

Attribute

BulkBinaryThreshold (BBT)

Purpose

The maximum size, in KB, of binary data that is exported to the bulk data file.

Valid Values

-1 | 0 | *x*

where:

x

is an integer that specifies the number of KB.

Behavior

If set to -1, all binary data, regardless of size, is written to the bulk data file, not to an external file.

If set to 0, all binary data, regardless of size, is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

If set to *x*, any binary data exceeding this specified number of KB is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

Default

32

GUI Tab

[Bulk tab](#)

Bulk Character Threshold

Attribute

BulkCharacterThreshold (BCT)

Purpose

The maximum size, in KB, of character data that is exported to the bulk data file.

Valid Values

-1 | 0 | x

where:

x

is an integer that specifies the number of KB.

Behavior

If set to -1, all character data, regardless of size, is written to the bulk data file, not to an external file.

If set to 0, all character data regardless of size, is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

If set to x , any character data exceeding this specified number of KB is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

Default

-1

GUI Tab

[Bulk tab](#)

Bulk Load Threshold

Attribute

BulkLoadThreshold (BLTH)

Purpose

Determines when the driver uses bulk load for insert, update, delete, or batch operations. If the Enable Bulk Load option is set to `True` and the number of rows affected by an insert, update, delete, or batch operation exceeds the threshold specified by this option, the driver uses SQL Server bulk load protocol to perform the operation.

Valid Values

0 | x

where:

x

is a positive integer that represents a threshold (number of rows).

Behavior

If set to 0, the driver always uses bulk load to execute insert, update, delete, or batch operations.

If set to `x`, the driver only uses bulk load if the Enable Bulk Load option is enabled and the number of rows to be updated by an insert, update, delete, or batch operation exceeds the threshold. If the operation times out, the driver returns an error.

Notes

- If the Enable Bulk Load option is set to `false`, this option is ignored.

Default

2

GUI Tab

Bulk tab

Bulk Options

Attribute

BulkLoadOptions (BLO)

Purpose

Toggles options for the bulk load process.

Valid Values

0 | `x`

where:

`x`

is a positive integer representing the cumulative total of the Bulk Options values.

Behavior

If set to 0, none of the options for bulk load are enabled.

If set to `x`, the values represented by `x` are enabled.

Note: The cumulative value of the options is only used in a connection string with the connection string attribute, BulkLoadOptions. On the Bulk tab of the driver Setup dialog, the individual options are enabled by selecting the appropriate check box.

The following bulk load options are available:

- Check Constraints - Checks constraints while data is being inserted. Value=16.
- Fire Triggers - Causes the server to fire the insert triggers for rows being inserted into the database. Value=32.
- Keep Identity - Preserves source identity values. When not enabled, identity values are assigned by the destination. Value=1.
- Keep Nulls - Preserves null values in the destination table regardless of the settings for default values. When not enabled, null values are replaced by column default values, where applicable. Value=64.

- Table Lock - Assigns a table lock for the duration of the bulk copy operation. Other applications are not permitted to update the table during the copy operation. When not enabled, the default bulk locking mechanism (row or table) specified by the table lock on bulk load server option is used. Value=2.

Example

If you wanted to enable Check Constraints (16), Fire Triggers (32), and Keep Identity (1) in a connection string, you would add the values together:

```
BulkLoadOptions=49
```

To enable these options on the Bulk tab of the driver Setup dialog, you would simply select the check box for each one.

Default

2 (Table Lock enabled)

GUI Tab

[Bulk tab](#)

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- The application must be thread-enabled to use connection pooling.
- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 533 for details.

Connection Reset

Attribute

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 533 for details.

Connection Retry Count

Attribute

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x , the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay (CRD)

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to x , the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (EncryptionMethod=1 | 6 | 7). When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, driver behavior is determined by the Encryption Method connection option.

Valid Values

```
cryptographic_protocol [ [ , cryptographic_protocol ] ... ]
```

where:

```
cryptographic_protocol
```

is one of the following cryptographic protocols:

```
TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2
```

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Notes

- This option is ignored if Encryption Method is set to 0 - No Encryption.
- Consult your database administrator concerning the data encryption settings of your server.

Default

```
TLSv1.2, TLSv1.1, TLSv1
```

GUI Tab

[Security tab](#)

See also

[Encryption Method](#) on page 507

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\Drivers  
\OpenSSL\1.0.0r\ddssl27.dll; (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the CryptoLibName and SSLLibName options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLLibName](#) on page 527

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or `odbc.ini` file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where:

database_name

is the name of a valid database.

Default

None

GUI Tab

[General tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Domain

Attribute

UNIX[®]

Domain (DOM)

Purpose

Specifies the Windows domain that the driver uses when connecting to a SQL Server Instance.

To connect to the database, users must supply the Windows User Id, Password, and, in some cases, domain to the driver. NTLM single sign on is not supported.

Valid Values

string

where:

string

is a valid Windows domain for the user specified by LoginId. This attribute applies only when Authentication Mode is set to 9.

Default

None

GUI Tab

n/a

Enable Bulk Load

Attribute

EnableBulkLoad (EBL)

Purpose

Specifies the bulk load method.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses the database bulk load protocol when an application executes an INSERT with multiple rows of parameter data. If the protocol cannot be used, the driver returns a warning.

If set to 0 (Disabled), the driver uses standard parameter arrays.

Default

0 (Disabled)

GUI Tab

[Bulk tab](#)

Enable Quoted Identifiers

Attribute

EnableQuotedIdentifiers (EQI)

Purpose

Determines whether the driver allows the use of quoted identifiers.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the database enforces ANSI rules regarding quotation marks. Double quotation marks can only be used for identifiers, such as column and table names. Character strings must be enclosed in single quotation marks, for example:

```
SELECT "au_id"  
FROM "authors"  
WHERE "au_lname" = 'O''Brien'
```

If set to no (Disabled), applications that use quoted identifiers encounter errors when they generate SQL statements with quoted identifiers.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Encryption Method

Attribute

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server.

Valid Values

0 | 1 | 6 | 7

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using the SSL protocols specified in the Crypto Protocol Version connection option.

If set to 6 (RequestSSL), the login request and data are encrypted using SSL if the server is configured for SSL. If the server is not configured for SSL, an unencrypted connection is established. The SSL protocol used is determined by the setting of the Crypto Protocol Version connection option.

If set to 7 (LoginSSL), the login request is encrypted using SSL regardless of whether the server is configured for SSL. The data is encrypted using SSL if the server is configured for SSL, and the data is unencrypted if the server is not configured for SSL. The SSL protocol used is determined by the setting of the Crypto Protocol Version connection option.

This option can only be set to 1 when Authentication Method is set to 1.

Notes

- For values 1 through 7, the SSL protocol used is determined by the setting of the Crypto Protocol Version connection option.
- The driver must use the server-specified packet size when using SSL encryption. If SSL is used, any value set for the Packet Size connection option is ignored.

- This connection option can affect performance.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

See Also

[Crypto Protocol Version](#) on page 502

[Performance Considerations](#) on page 533

Failover Granularity

Attribute

FailoverGranularity (FG)

Purpose

Determines whether the driver fails the entire failover process or continues with the process if errors occur while trying to reestablish a lost connection.

This option applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select).

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 (Non-Atomic), the driver continues with the failover process and posts any errors on the statement on which they occur.

If set to 1 (Atomic) the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

If set to 2 (Atomic Including Repositioning), the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

If set to 3 (Disable Integrity Check), the driver does not verify that the rows that were restored during the failover process match the original rows. This value applies only when Failover Mode is set to 2 (Select).

Default

0 (Non-Atomic)

GUI Tab

[Failover tab](#)

Failover Mode

Attribute

FailoverMode (FM)

Purpose

Specifies the type of failover method the driver uses.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Connection), the driver provides failover protection for new connections only.

If set to 1 (Extended Connection), the driver provides failover protection for new and lost connections, but not any work in progress.

If set to 2 (Select), the driver provides failover protection for new and lost connections. In addition, it preserves the state of work performed by the last Select statement executed.

Notes

- This connection option can affect performance.

Default

0 (Connection)

GUI Tab

[Failover tab](#)

See Also

See [Performance Considerations](#) on page 533 for details.

Failover Preconnect

Attribute

FailoverPreconnect (FP)

Purpose

Specifies whether the driver tries to connect to the primary and an alternate server at the same time.

This attribute applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select) and at least one alternate server is specified.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver tries to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection. This value provides the best performance, but your application typically experiences a short wait while the failover connection is attempted.

If set to 1 (Enabled), the driver tries to connect to the primary and an alternate server at the same time. This can be useful if your application is time-sensitive and cannot absorb the wait for the failover connection to succeed.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Fetch TSWTZ as Timestamp

Attribute

FetchTSWTZasTimestamp (FTSWTZAT)

Purpose

Determines whether the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_TYPE_TIMESTAMP or SQL_VARCHAR.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the timestamp with time zone data type as the ODBC type SQL_TYPE_TIMESTAMP. The time zone information in the fetched value is truncated. Use this value if your application needs to process values the same way as TIMESTAMP columns.

If set to 0 (Disabled), the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_VARCHAR. Use this value if your application requires the time zone information in the fetched value.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Fetch TWFS as Time

Attribute

FetchTWFSasTime (FTWFSAT)

Purpose

Determines whether the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIME` or `SQL_TYPE_TIMESTAMP`.

Supported only for Microsoft SQL Server 2008.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIME`. The fractional seconds portion of the value is truncated.

If set to 0 (Disabled), the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIMESTAMP`. The fractional seconds portion of the value is preserved. Time columns are not searchable when they are described and fetched as timestamp.

Notes

- When returning time with fractional seconds data as `SQL_TYPE_TIMESTAMP`, the Year, Month and Day parts of the timestamp must be set to zero.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Field Delimiter

Attribute

BulkLoadFieldDelimiter (BLFD)

Purpose

Specifies the character that the driver will use to delimit the field entries in a bulk load data file.

Valid Values

x

where:

x

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash(-), the colon(:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Field Delimiter character must be different from the Bulk Load Record Delimiter.

Default

None

GUI Tab

[Bulk tab](#)

GSS Client Library

Attribute

GSSClient (GSSC)

Purpose

The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC). The driver uses the path defined by the PATH environment variable for loading the specified client library.

Valid Values

native | *client_library*

where:

client_library

is a GSS client library installed on the client.

Behavior

If set to *client_library*, the driver uses the specified GSS client library.

If set to native, the driver uses the GSS client shipped with the operating system.

Default

native

GUI Tab

[Security tab](#)

Host Name

Attribute

HostName (HOST)

Purpose

The name or the IP address of the server to which you want to connect.

Valid Values

IP_address | *named_server* | *named_instance* | *server_name* | *virtual_network_name*

where:

IP_address

is the IP address of the server to which you want to connect. Specify this address as: *IP_address*. For example, you can enter 199.226.224.34.

The IP address can be specified in either IPv4 or IPv6 format. See [Using IP Addresses](#) on page 67 for details about these formats.

named_server

is the named server address of the server to which you want to connect. Specify this address as: *named_server*. For example, you can enter SSserver.

named_instance

is a named instance of Microsoft SQL Server or Windows Azure SQL Database. Specify this address as: *server_name\instance_name*.

virtual_network_name

is the virtual network name (VNN) of the availability group listener when using an Always On Availability Group.

Notes

- If only a server name is specified with no instance name, the driver uses the default instance on the server.
- If only a server name is specified with a backward slash \ or * at the end with no instance name, the driver uses the first instance on the server with a TCP port.

Default

None

GUI Tab

[General tab](#)

Host Name In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

host_name | #SERVERNAME#

where:

host_name

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If set to a host name, the driver examines the subjectAltName values included in the certificate. If a dnsName value is present in the subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the dnsName value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the dnsName value.

If no subjectAltName values exist or a dnsName value is not in the list of subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the commonName part of the Subject name in the certificate. The commonName typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the commonName. If multiple commonName parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the commonName parts.

If set to #SERVERNAME#, the driver compares the host server name specified as part of a data source or connection string to the dnsName or the commonName value.

Default

None

GUI Tab

[Security tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Initialization String

Attribute

InitializationString (IS)

Purpose

A SQL command that is issued immediately after connecting to the database to manage session settings.

Valid Values

SQL_command

where:

SQL_command

is a valid SQL command that is supported by the database.

Example

To set the date format on every connection, specify:

```
Set DateStyle='ISO, MDY'
```

Notes

- If the statement fails to execute, the connection fails and the driver reports the error returned from the server.

Default

None

GUI Tab

[Advanced tab](#)

Language

Attribute

Language (LANG)

Purpose

The national language to use for Microsoft SQL Server system messages.

Valid Values

lang

where:

lang

is the language to use for Microsoft SQL Server system messages. This overrides the default language specified for the login on the server. If no language is specified, the connection uses the default language specified for the login on the server.

Default

None

GUI Tab

[Advanced tab](#)

Load Balance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

The number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Valid Values

0 | x

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to x , inactive connections are closed after the specified number of seconds passes.

Notes

- The Min Pool Size option may cause some connections to ignore this value.
- This connection option can affect performance.

Default

0

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 533

Load Balancing

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x, the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

15

GUI Tab

[Advanced tab](#)

Max Pool Size

Attribute

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

For example, if set to 20, the maximum number of connections allowed in the pool is 20.

Notes

- This connection option can affect performance.

Default

100

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 533

Min Pool Size

Attribute

MinPoolSize (MNPS)

Purpose

The minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Valid Values

0 | x

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

If set to x , the start-up number of connections in the pool is x in addition to the current existing connection.

Notes

- This connection option can affect performance.

Example

If set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Default

0

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 533

Multi-Subnet Failover

Attribute

MultiSubnetFailover (MSF)

Purpose

Determines whether the driver attempts parallel connections to the failover IP addresses of an Availability Group during a multi-subnet failover. When Multi-Subnet Failover is enabled, the driver simultaneously attempts to connect to all IP addresses associated with the Availability Group listener when the connection is broken or the listener IP address becomes unavailable. The first IP address to successfully respond to the request is used for the connection. Using parallel-connection attempts offers improved response time over traditional failover, which attempts to connect to alternate servers one at a time.

Valid Values

1 | 0

Behavior

If set to 1 (Enabled), the driver attempts parallel connections to all failover IP addresses in an Availability Group when the connection is broken or the listener IP address is unavailable. The first IP address to successfully respond to the request is used for the connection. This setting is only supported when your environment is configured for Always On Availability Groups.

If set to 0 (Disabled), the driver uses the failover method specified by the Failover Mode connection option when the primary server is unavailable. Use this setting if your environment is not configured for Always On Availability Groups.

Notes

- When MultiSubnetFailover is enabled, the virtual network name (VNN) of the availability group listener must be specified by the Host Name connection option.
- When MultiSubnetFailover is enabled, the Alternate Servers, Load Balancing, and Failover Preconnect connection options are disabled.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Packet Size

Attribute

PacketSize (PS)

Purpose

Determines the number of bytes for each database protocol packet that is transferred from the database server to the client machine. Adjusting the packet size can improve performance. The optimal value depends on the typical size of data that is inserted, updated, or returned by the application and the environment in which it is running. Typically, larger packet sizes work better for large amounts of data. For example, if an application regularly returns character values that are 10,000 characters in length, using a value of 32 (16 KB) typically results in improved performance.

Valid Values

-1 | 0 | x

Behavior

If set to -1, the driver uses the maximum packet size that is set by the database server.

If set to 0, the driver uses the default packet size that is used by the database server.

If set to x, an integer from 1 to 127, the driver uses a packet size that is a multiple of 512 bytes. For example, PacketSize=8 means to set the packet size to 8 * 512 bytes (4096 bytes).

Notes

- If SSL encryption is used, the driver must use the packet size that is specified by the server. Any value set for this option or the SQL_PACKET_SIZE connect option is ignored if SSL encryption is used.
- The ODBC connection option SQL_PACKET_SIZE provides the same functionality as the Packet Size option; however SQL_PACKET_SIZE and the Packet Size option are mutually exclusive. If Packet Size is specified, the driver returns the message Driver Not Capable if an application attempts to call SQL_PACKET_SIZE. If you do not set the Packet Size option, application calls to SQL_PACKET_SIZE are accepted by the driver.

Default

-1

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 533

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Purpose

The port number of the server listener.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your database administrator for the correct number.

Default

1433

GUI Tab

General tab

PRNGSeedFile

Attribute

PRNGSeedFile (PSF)

Purpose

UNIX[®] Specifies the absolute path for the entropy-source file or device used as a seed for SSL key generation.

Valid Values

string | RANDFILE

where:

string

is the absolute path for the entropy-source file or device that seeds the random number generator used for TLS/SSL key generation.

Behavior

If set to *string*, the specified entropy-source file or device seeds the random number generator used for TLS/SSL key generation. Entropy levels and behavior may vary for different files and devices. See the following section for a list of commonly used entropy sources and their behavior.

If set to RANDFILE, the `RAND_file_name()` function in your application generates a default path for the random seed file. The seed file is `$RANDFILE` if that environment variable is set; otherwise, it is `$HOME/.rnd`. If `$HOME` is not set either, an error occurs.

Common Valid Values

Although other entropy-source files may be specified, the following valid values are for files and devices that are commonly used for seeding:

`/dev/random`

is a pseudorandom number generator (blocking) that creates a seed from random bits of environmental noise it collects in an entropy pool. When there is insufficient noise in the pool, the file blocks calls until enough noise is collected. This provides more secure SSL key generation, but at the expense of blocked calls.

`/dev/urandom`

is a pseudorandom number generator (non-blocking) that creates seeds from random bits from environmental noise it collects in an entropy pool. When there is insufficient noise in the pool, the file reuses bits from the pool instead of blocking calls. This eliminates potential delays associated with blocked calls, but may result in less secure TLS/SSL key generation.

`/dev/hwrng`

is a hardware random number generator. The behavior is dependent on the device used in your environment.

Notes

- This option is ignored when TLS/SSL is disabled (`EncryptionMethod=0`) or the seed source is set to Poll Only (`PRNGSeedSource=1`).
- For processes that employ multiple TLS/SSL-enabled drivers, the behavior of this option for all drivers is determined by the values specified for the driver that first connects to the process and loads the OpenSSL library. Since the OpenSSL library loads only once per process, the values specified for drivers that subsequently connect are ignored. To ensure that the correct security settings are used, we recommend configuring this option identically for all drivers used in a process.

Default

`/dev/random`

GUI tab

NA

See also

[PRNGSeedSource](#) on page 524

PRNGSeedSource

Attribute

PRNGSeedSource (PSS)

Purpose

UNIX[®] Specifies the source of the seed the driver uses for TLS/SSL key generation. Seeds are a pseudorandom or random value used to set the initial state of the random number generator used to generate TLS/SSL keys. Using seeds with a higher level of entropy, or randomness, provides a more secure transmission of data encrypted using TLS/SSL.

Valid Values

0 | 1

Behavior

If set to 0 (File), the driver uses entropy-source file or device specified in the PRNGSeedFile connection option as the seed used for TLS/SSL key generation.

If set to 1 (Poll Only), the driver uses the RAND_poll function in TLS/SSL to create the seed used for TLS/SSL key generation.

Notes

- For processes that employ multiple TLS/SSL-enabled drivers, the behavior of this option for all drivers is determined by the values specified for the driver that first connects to the process and loads the OpenSSL library. Since the OpenSSL library loads only once per process, the values specified for drivers that subsequently connect are ignored. To ensure that the correct security settings are used, we recommend configuring this option identically for all drivers used in a process.
- This option is ignored when TLS/SSL is disabled (`EncryptionMethod=0`)

Default

0 (File)

GUI Tab

NA

See also

[PRNGSeedFile](#) on page 523

Query Timeout

Attribute

QueryTimeout (QT)

Purpose

The number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the `SQL_ATTR_QUERY_TIMEOUT` statement attribute on the `SQLSetStmtAttr()` function.

Valid Values

-1 | 0 | *x*

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to -1, the query does not time out. The driver silently ignores the `SQL_ATTR_QUERY_TIMEOUT` attribute.

If set to 0, the query does not time out, but the driver responds to the `SQL_ATTR_QUERY_TIMEOUT` attribute.

If set to *x*, all queries time out after the specified number of seconds unless the application overrides this value by setting the `SQL_ATTR_QUERY_TIMEOUT` attribute.

Default

0

GUI Tab

[Advanced tab](#)

Record Delimiter

Attribute

BulkLoadRecordDelimiter (BLRD)

Purpose

Specifies the character that the driver will use to delimit the record entries in a bulk load data file.

Valid Values

x

where:

x

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash(-), the colon(:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Record Delimiter character must be different from the Bulk Load Field Delimiter.

Default

None

GUI Tab

[Bulk tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where *x* is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Advanced tab](#)

SSLLibName

Attribute

SSLLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\Connect64_for_ODBC_71\
Drivers\OpenSSL\1.0.0r\ddssl27.dll; (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
SSLLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 503

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Trust Store

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used

Valid Values.

truststore_directory\filename

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Default

None

GUI Tab

[Security tab](#)

Trust Store Password

Attribute

TruststorePassword (TSP)

Purpose

The password that is used to access the truststore file when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

- The truststore and keystore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Use Snapshot Transactions

Attribute

SnapshotSerializable (SS)

Purpose

Allows your application to use the snapshot isolation level if your Microsoft SQL Server database is configured for Snapshot isolation. Supported only for Microsoft SQL Server 2005 and higher.

See [Using The Snapshot Isolation Level](#) on page 540 for details about using the snapshot isolation level.

Valid Values

0 | 1

Behavior

When set to 1 (Enabled) and your application has the transaction isolation level set to serializable, the application uses the snapshot isolation level.

When set to 0 (Disabled) and your application has the transaction isolation level set to serializable, the application uses the serializable isolation level.

This option is useful for existing applications that set the isolation level to serializable. Using Snapshot Transactions in this case allows you to change to the snapshot isolation level with no or minimum code changes. If developing a new application, you can code it to set the connection attribute `SQL_COPT_SS_TXN_ISOLATION` to the value `SQL_TXN_SS_SNAPSHOT`.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 533 for details.

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Security tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Truststore information is specified using the Trust Store and Trust Store Password options.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Default

1 (Enabled)

GUI Tab

[Security tab](#)

Workstation ID

Attribute

WorkstationID (WSID)

Purpose

The workstation ID that is used by the client.

Valid Values

string

where:

string

is the workstation ID.

Default

None

GUI Tab

[Advanced tab](#)

XML Describe Type

Attribute

XMLDescribeType (XDT)

Purpose

The SQL data type that is returned by SQLGetTypeInfo for the XML data type.

See [Using the XML Data Type](#) on page 536 for further information about the XML data type.

Valid Values

-4 | -10

Behavior

If set to -4 (SQL_LONGVARBINARY), the driver uses the description SQL_LONGVARBINARY for columns that are defined as the XML data type.

If set to -10 (SQL_WLONGVARCHAR), the driver uses the description SQL_WLONGVARCHAR for columns that are defined as the XML data type.

Default

-10

GUI Tab

[Advanced tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Connection Pooling (Pooling): If you enable the driver to use connection pooling, you can set additional options that affect performance:

- **Load Balance Timeout (LoadBalanceTimeout):** You can define how long to keep connections in the pool. The time that a connection was last used is compared to the current time and, if the timespan exceeds the value of the Load Balance Timeout option, the connection is destroyed. The Min Pool Size option can cause some connections to ignore this value.
- **Connection Reset (ConnectionReset):** Resetting a re-used connection to the initial configuration settings impacts performance negatively because the connection must issue additional commands to the server.
- **Max Pool Size (MaxPoolSize):** Setting the maximum number of connections that the pool can contain too low might cause delays while waiting for a connection to become available. Setting the number too high wastes resources.
- **Min Pool Size (MinPoolSize):** A connection pool is created when the first connection with a unique connection string connects to the database. The pool is populated with connections up to the minimum pool size, if one has been specified. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Enable Bulk Load (EnableBulkLoad): If your application performs bulk loading of data, you can improve performance by configuring the driver to use the database system's bulk load functionality instead of database array binding. The trade-off to consider for improved performance is that using the bulk load functionality can bypass data integrity constraints.

Encryption Method (EncryptionMethod): Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) that is required to encrypt and decrypt data.

Failover Mode (FailoverMode): Although high availability that replays queries after a failure provides increased levels of protection, it can adversely affect performance because of increased overhead.

Packet Size (PacketSize): Typically, it is optimal for the client to use the maximum packet size that the database server allows. This reduces the total number of round trips required to return data to the client, thus improving performance. Therefore, performance can be improved if the PacketSize attribute is set to the maximum packet size of the server.

Use Snapshot Transactions (SnapshotSerializable): You must have your Microsoft SQL Server 2005 and higher database configured for snapshot isolation for this connection option to work. Snapshot Isolation provides transaction-level read consistency and an optimistic approach to data modifications by not acquiring locks on data until data is to be modified. This Microsoft SQL Server 2005 and higher feature can be useful if you want to consistently return the same result set even if another transaction has changed the data and 1) your application executes many read operations or 2) your application has long running transactions that could potentially block users from reading data. This feature has the potential to eliminate data contention between read operations and update operations. When this connection option is enabled, performance is improved due to increased concurrency.

See [Using The Snapshot Isolation Level](#) on page 540 for details.

Data Types

The following table shows how the Microsoft SQL Server and Windows Azure SQL Database data types are mapped to the standard ODBC data types. [Unicode Support](#) on page 535 lists Microsoft SQL Server to Unicode data type mappings.

Table 34: Microsoft SQL Server Data Types

SQL Server	ODBC
binary	SQL_BINARY
bigint	SQL_BIGINT
bigint identity	SQL_BIGINT
bit	SQL_BIT
char	SQL_CHAR
date ³³	SQL_TYPE_DATE
datetime	SQL_TYPE_TIMESTAMP
datetime2 ³³	SQL_TYPE_TIMESTAMP
decimal	SQL_DECIMAL
decimal() identity	SQL_DECIMAL
float	SQL_FLOAT
image	SQL_LONGVARBINARY
int	SQL_INTEGER
int identity	SQL_INTEGER
money	SQL_DECIMAL

³³ Supported only on Microsoft SQL Server 2008 and higher.

SQL Server	ODBC
numeric	SQL_NUMERIC
numeric() identity	SQL_NUMERIC
real	SQL_REAL
smalldatetime	SQL_TYPE_TIMESTAMP
smallint	SQL_SMALLINT
smallint identity	SQL_SMALLINT
smallmoney	SQL_DECIMAL
text	SQL_LONGVARCHAR
time ^{33, 34}	SQL_TYPE_TIMESTAMP
timestamp	SQL_BINARY
tinyint	SQL_TINYINT
tinyint identity	SQL_TINYINT
uniqueidentifier	SQL_GUID
varbinary	SQL_VARBINARY
varbinary(max) ³⁵	SQL_LONGVARBINARY
varchar	SQL_VARCHAR
varchar(max) ³⁵	SQL_LONGVARCHAR

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

Unicode Support

The SQL Server Wire Protocol driver maps the Microsoft SQL Server and Windows Azure SQL Database data types to Unicode data types as shown in the following table:

³⁴ Time mapping changes based on the setting of the Fetch TWFS as Time option.

³⁵ Supported only on Microsoft SQL Server 2005 and higher.

Table 35: Mapping Microsoft SQL Server and Windows Azure SQL Database Data Types to Unicode Data Types

SQL Server Data Type	Mapped to . . .
datetimeoffset ^{36, 37}	SQL_WVARCHAR
nchar	SQL_WCHAR
ntext	SQL_WLONGVARCHAR
nvarchar	SQL_WVARCHAR
nvarchar(max) ³⁸	SQL_WLONGVARCHAR
sysname	SQL_WVARCHAR
xml ³⁸	SQL_WLONGVARCHAR

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Using the XML Data Type

By default, Microsoft SQL Server returns XML data to the driver encoded as UTF-8. To avoid data loss, an application must bind XML data as SQL_C_WCHAR. The driver then returns the data as either UTF-8 or UTF-16, depending on platform and application settings. If the application binds XML data as SQL_C_CHAR, the driver converts it to the client character encoding, possibly causing data loss or corruption. To prevent any conversion of XML data, the application must set the attribute [XML Describe Type](#) on page 532 to SQL_LONGVARBINARY (-10) and bind the data as SQL_C_BINARY.

Advanced Features

The driver supports the following advanced features:

- Failover
- Security
- Connection Pooling
- DataDirect Bulk Load

³⁶ Supported only for Microsoft SQL Server 2008 and higher, and Windows Azure SQL Database.

³⁷ Datetimeoffset mapping changes based on the setting of the Fetch TSWTZ as Timestamp option.

³⁸ nvarchar(max) and xml are supported for Microsoft SQL Server 2005 and higher.

Failover

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Security

The driver supports authentication and encryption. Security connection options are located on the [Security tab](#) of the driver Setup dialog box. See [Using Security](#) on page 89 for a general description of security and its implementation.

Authentication

If you are using Kerberos, verify that your environment meets the requirements listed in the following table before you configure the driver for Kerberos authentication.³⁹

Table 36: Kerberos Authentication Requirements for the SQL Server Wire Protocol Driver

Component	Requirements
Microsoft SQL Server database server	The database server must be administered by the same domain controller that administers the client and must be running one of the following databases: <ul style="list-style-type: none"> • Microsoft SQL Server 2014 • Microsoft SQL Server 2012 • Microsoft SQL Server 2008 • Microsoft SQL Server 2005 • Microsoft SQL Server 2000 • Microsoft SQL Server 2000 Enterprise Edition (64-bit) Service Pack 2 or higher
Kerberos server	The Kerberos server is the machine where the user IDs for authentication are administered. The Kerberos server is also the location of the Kerberos KDC. Network authentication must be provided by Windows Active Directory on Windows Server 2008 or higher.
Client	The client must be administered by the same domain controller that administers the database server.

Configuring Azure Active Directory Authentication

The driver supports Azure Active Directory authentication (Azure AD). Azure AD authentication is an alternative to SQL Server Authentication that allows administrators to centrally manage user permissions to Azure SQL Database data stores. When Azure AD authentication is enabled, all communications to the service are encrypted.

³⁹ Not supported for Microsoft Windows Azure for SQL Database. You must provide credentials every time when you connect to SQL Database.

To configure the driver to use Azure AD authentication:

- Set the Authentication Method option to 13 (Active Directory Password).
- Set the Trust Store connection option to specify the absolute path of the digital certificate file for the root CA certificates. The driver requires these certificates to maintain a secure connection.

Note: For testing purposes, you can disable the truststore requirement by setting the Validate Server Certificate to 0 (disabled). Disabling the Validate Server Certificate option leaves your connection vulnerable to man-in-the-middle attacks; therefore, it is not recommended for extended use.

- Set the Host Name In Certificate option to specify the host name for SSL certificate validation. For example, `*.database.windows.net`.
- Set the User Name option to specify your Active Directory username using the `userid@domain.com` format.
- Set the Password option to specify your Active Directory password.
- Specify values for minimum required options for establishing a connection:
 - Set the Host Name option to specify either the IP address in IPv4 or IPv6 format, or the server name for your Azure server. For example, `your_server.database.windows.net`.
 - Set the Port Number option to specify the TCP port of the primary database server that is listening for connections to the database.
 - Set Database option to specify the name of the database to which you want to connect.
 - If using data sources, set the Data Source Name to specify the name of your data source.

For example, the following is a DSN-less connection string with only the required options for making a connection using Azure AD authentication:

```
DRIVER={DataDirect 7.1 SQL Server Wire Protocol};AM=13;DB=SQLSdb1;  
HOST=myserver.database.windows.net;HNIC=*.database.windows;PORT=1433;  
TS=\<truststore_path>\ca-bundle.crt;VSC=1;UID=test@domain.com;PWD=secret;
```

The following example demonstrates a data source definition in the `odbc.ini` file with only the required options for making a connection using Azure AD authentication:

```
[SQLServer Wire Protocol]  
Driver=ODBCHOME/lib/ivsqs27.so  
Description=DataDirect 7.1 SQL Server Wire Protocol  
AuthenticationMethod=13  
Database=SQLSdb1  
HostName=myserver.database.windows.net  
HostNameInCertificate=*.database.windows  
LogonID=test@domain.com  
Password=secret  
PortNumber=1433  
TrustStore=/<truststore_path>/ca-bundle.crt  
ValidateServerCertificate=1
```

See also

[Authentication Method](#) on page 494

[Trust Store](#) on page 529

[Host Name In Certificate](#) on page 514

[User Name](#) on page 531

[Password](#) on page 522
[Host Name](#) on page 513
[Port Number](#) on page 522
[Database](#) on page 504
[Data Source Name](#) on page 504

Connection Pooling

The driver supports connection pooling and its related connection options. Connection pooling connection options are located on the [Pooling tab](#) of the driver Setup dialog box. See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling and its implementation.

DataDirect Bulk Load

The driver supports DataDirect bulk load and its related connection options. Bulk load connection options are located on the [Bulk tab](#) of the driver Setup dialog box. See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect bulk load and its implementation.

For optimal performance, you must enable minimal logging and Table Locking. Please refer to the following Web site for more information on minimal logging:

<https://msdn.microsoft.com/en-us/library/ms190422.aspx>

<https://msdn.microsoft.com/en-us/library/ms190203.aspx>

Table Locking, one of the Bulk Options, is enabled by default. This prevents other transactions from accessing the table during bulk load. See [Bulk Options](#) on page 498 for details about this option.

Limitations

- A bulk operation is not allowed in a manual transaction if it is not the first event.
- Once a bulk operation is started, any non-bulk operation is disallowed until the transaction is committed.
- Because of Oracle limitations, issuing a SELECT statement to determine a row count may return different results before and after a bulk load operation.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

Microsoft SQL Server supports isolation levels 0 (Read Uncommitted), 1 (Read Committed), 2 (Repeatable Read), and 3 (Serializable). Microsoft SQL Server supports row-level and table-level locking.

Microsoft SQL Server 2005 and higher supports the following additional isolation levels:

- Snapshot
- Read Committed with Snapshots
- Read Committed with Locks (equivalent to Read Committed in previous Microsoft SQL Server versions)

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Using The Snapshot Isolation Level

The Snapshot isolation level is available only with Microsoft SQL Server 2005 and higher. Setting the SnapshotSerializable connection string attribute changes the behavior of the Serializable isolation level to use the Snapshot Isolation level. This allows an application to use the Snapshot Isolation level with minimal or no code changes.

If you are writing a new application, you may want to code it to set the connection attribute SQL_COPT_SS_TXN_ISOLATION to the value SQL_TXN_SS_SNAPSHOT. The application then uses the snapshot isolation level without requiring the Use Snapshot Transactions connection option.

See [Use Snapshot Transactions](#) on page 530 for additional information.

SQL Support

The driver supports the core SQL grammar.

ODBC Conformance Level

The driver supports ODBC conformance level 1.

In addition, the following functions are supported:

- SQLForeignKeys
- SQLTablePrivileges
- SQLDescribeParam
- SQLColumnPrivileges

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The SQL Server Wire Protocol driver supports multiple connections and multiple statements per connection.

Using Arrays of Parameters

Microsoft SQL Server databases natively support parameter arrays, and the SQL Server Wire Protocol driver, in turn, supports them. When designing an application for performance, using native parameter arrays for bulk inserts or updates, for example, can improve performance.

Refer to "Designing ODBC applications for performance optimization" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Support for Azure Synapse Analytics and Analytics Platform System

The driver transparently connects to Microsoft Azure Synapse Analytics and Microsoft Analytics Platform System (APS); however, the following limitations to features and functionality apply:

- No support for unquoted identifiers. The driver always enforces ANSI rules regarding quotation marks for all ADW and APS connections (`EnabledQuotedIdentifiers=1`); therefore, the Enable Quoted Identifiers option is disabled.
- No support for connection pooling reauthentication.
- No support for Data Definition Language (DDL) queries within transactions.
- No support for closing holdable cursors when a transaction is committed.
- No support for server side cursors; therefore:
 - Scroll-sensitive result sets are not supported.
 - The driver always disables server side cursors.
- No support for XA connections.
- Support for isolation levels is limited to only the read uncommitted level.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

- Support for the `varchar(max)`, `nvarchar(max)`, `varbinary(max)` data types is limited to Heap and Clustered Index Tables.
- No support for the following SQL Server data types:

<code>decimal()</code> identity	timestamp
image	<code>tinyint()</code> identity
<code>numeric()</code> identity	ntext
<code>smallint</code> identity	xml
text	

- Support for scalar string functions is limited to the following functions:

ASCII	LEFT	RTRIM
CHAR	LTRIM	SOUNDEX
CONCAT	REPLACE	SPACE
DIFFERENCE	RIGHT	SUBSTRING

- Support for scalar numeric functions is limited to the following functions:

ABS	EXP	ROUND
ACOS	FLOOR	SIGN
ASIN	LOG	SIN
ATAN	LOG10	SQRT

CEILING	PI	TAN
COS	POWER	TRUNCATE
COT (ADW only)	RADIANS	
DEGREES	RAND	

- Support for scalar date and time functions is limited to the following functions:

CURDATE	DAYOFWEEK	QUARTER
CURRENT_DATE	DAYOFYEAR	SECOND
CURRENT_TIME	HOUR	WEEK
CURTIME	MINUTE	YEAR
DAYNAME	MONTH	
DAYOFMONTH	MONTHNAME	

Refer to "Scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for more information.

See also

[Enable Quoted Identifiers](#) on page 506

[Isolation and Lock Levels Supported](#) on page 539

The Sybase Wire Protocol Driver

The DataDirect Connect *for* ODBC and DataDirect Connect64 *for* ODBC Sybase Wire Protocol driver (the Sybase Wire Protocol driver) each support the following databases and services:

- SAP Adaptive Server Enterprise
- Sybase Adaptive Server Enterprise

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The Sybase Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect product for the file name of the Sybase Wire Protocol driver.

Driver Requirements

The driver has no client requirements.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 557 and [Connection Option Descriptions for Sybase Wire Protocol](#) on page 558 for an alphabetical list of driver connection string attributes and their initial default values

Data Source Configuration in the UNIX/Linux odbc.ini File

UNIX[®] On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions for Sybase Wire Protocol](#) on page 558 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Sybase)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.



UNIX[®] On UNIX and Linux, data sources are stored in the odbc.ini file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Sybase data source:

1. Start the ODBC Administrator:

-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
-  On Linux, change to the *install_dir/tools* directory and, at a command prompt, enter:


```
odbcadmin
```

where *install_dir* is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 51: General tab

ODBC Sybase Wire Protocol Driver Setup

Performance Failover Pooling Bulk About
General Advanced Security Connection

Data Source Name: Sybase Wire Protocol Help

Description:

Network Address:

Database Name:

Use Interfaces File for Connection Information (Optional)

Interfaces File:

Server Name:

Test Connect OK Cancel Apply

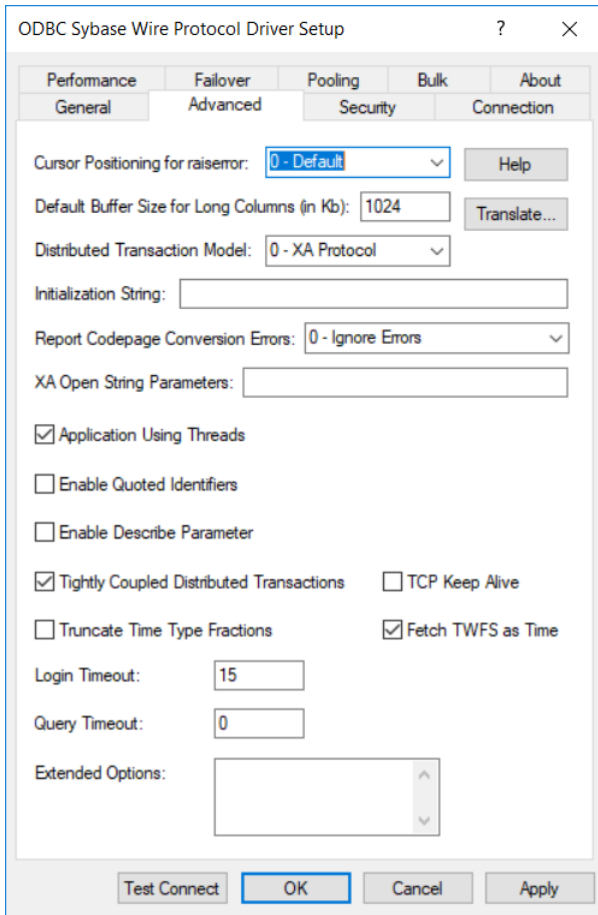
Note: The General tab displays the only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 574	None
Description on page 575	None
Network Address on page 592	None
Database Name on page 575	None
Interfaces File on page 588	None
Server Name on page 600	None

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 52: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Cursor Positioning for Raiserror on page 573	0 - Default
Default Buffer Size for Long/LOB Columns (in Kb) on page 576	1024
Distributed Transaction Model on page 577	0 - XA Protocol
Initialization String on page 587	None
Report Codepage Conversion Errors on page 599	0 - Ignore Errors
XA Open String Parameters on page 607	None
Application Using Threads on page 564	Enabled
Enable Quoted Identifiers on page 578	Disabled
Enable Describe Parameter on page 578	Disabled

Connection Options: Advanced	Default
Tightly Coupled Distributed Transactions on page 603	Enabled
TCP Keep Alive on page 602	Disabled
Truncate Time Type Fractions on page 603	Disabled
Fetch TWFS as Time on page 582	Enabled
Login Timeout on page 590	15
Query Timeout on page 597	0
IANAAppCodePage on page 586 UNIX ONLY	4 (ISO 8559-1 Latin-1)



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.

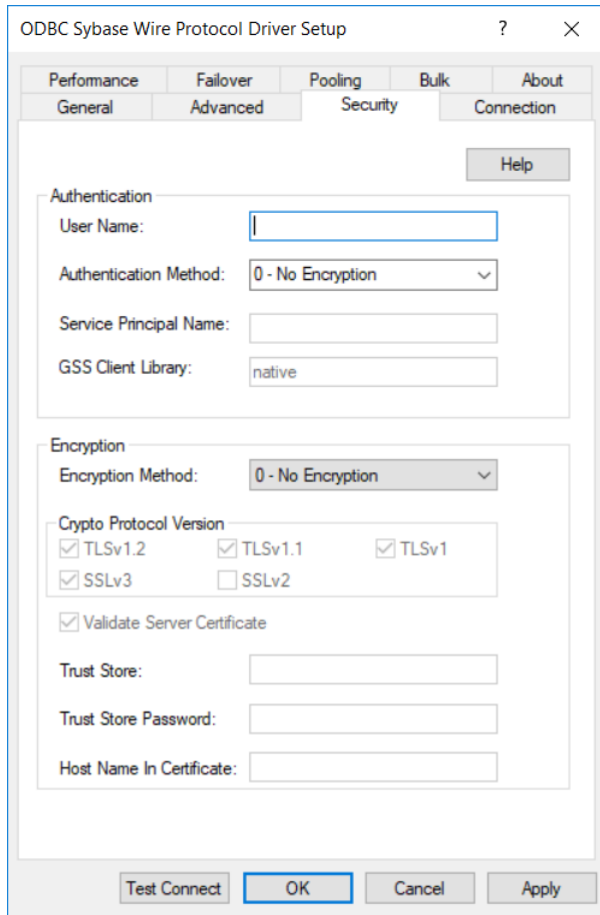


Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- Optionally, click the **Security** tab to specify security data source settings.

Figure 53: Security tab



See [Using Security](#) on page 89 for a general description of authentication and encryption and their configuration requirements.

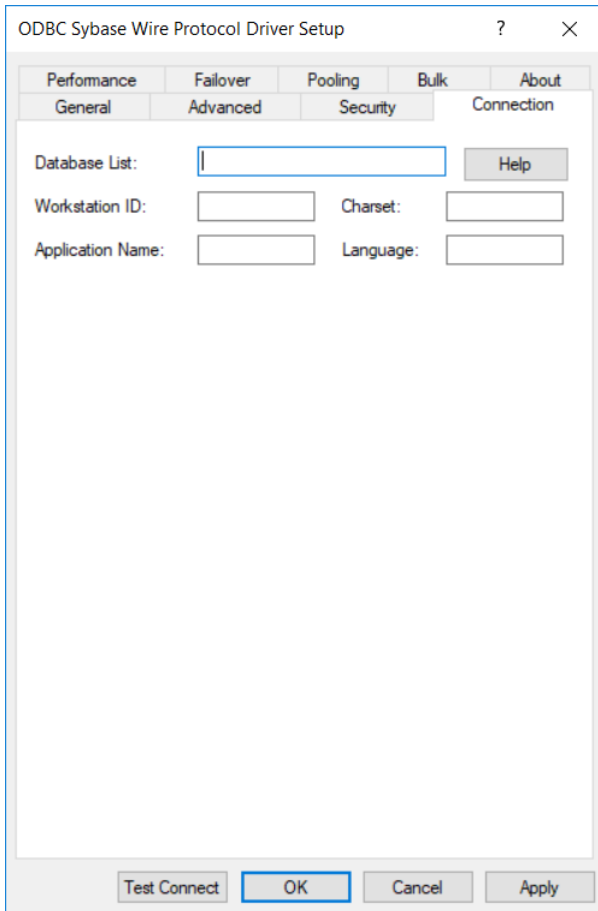
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Security	Default
User Name on page 605	None
Authentication Method on page 565	0 - No Encryption
Service Principal Name on page 601	None
GSS Client Library on page 584	native
Encryption Method on page 579	0 - No Encryption
Crypto Protocol Version on page 571	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Validate Server Certificate on page 606	Enabled

Connection Options: Security	Default
Truststore on page 604	None
Truststore Password on page 605	None
Host Name In Certificate on page 585	None

6. Optionally, click the **Connection** tab to specify data source settings.

Figure 54: Connection tab



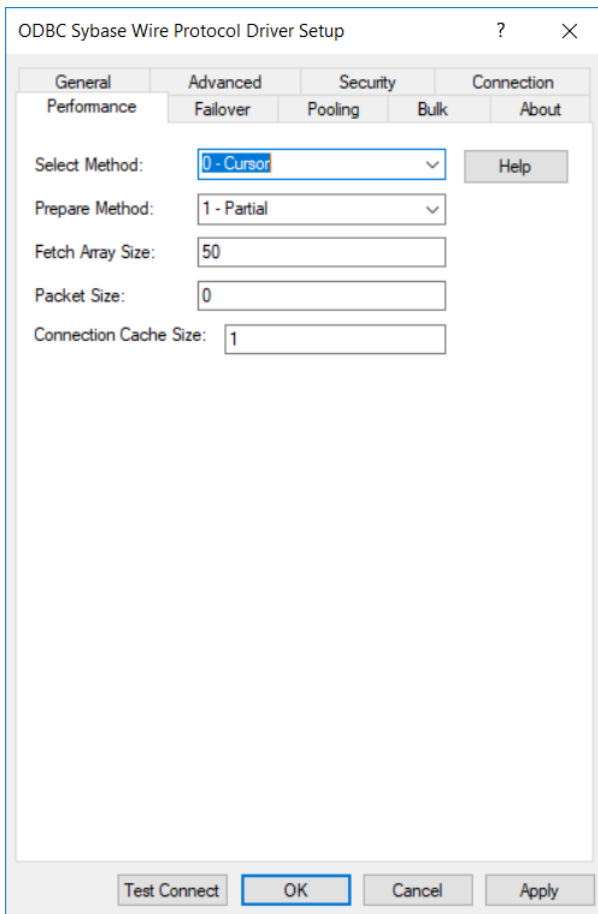
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Connection	Default
Database List on page 574	None
Workstation ID on page 606	None
Charset on page 567	None

Connection Options: Connection	Default
Application Name on page 563	None
Language on page 588	None

7. Optionally, click the **Performance** tab to specify performance data source settings.

Figure 55: Performance tab

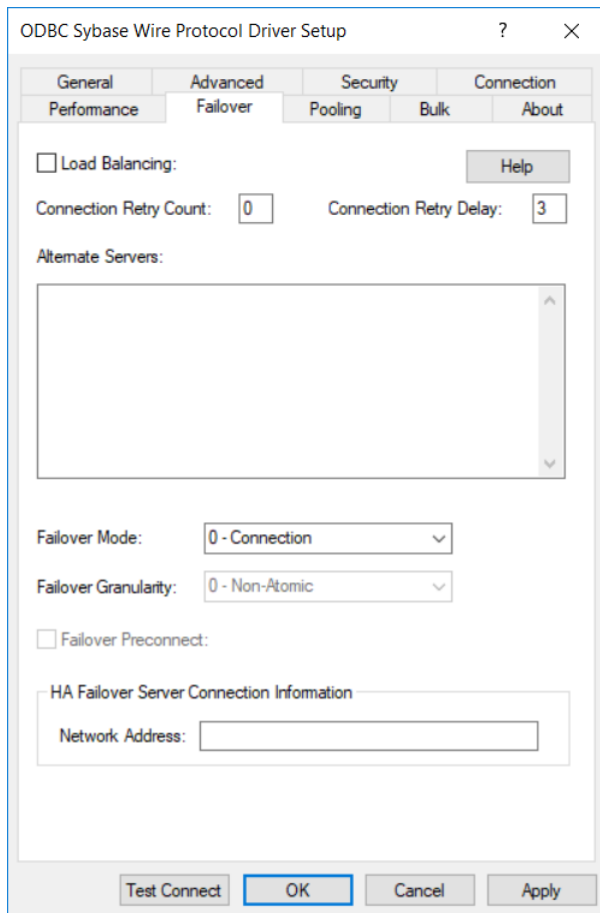


On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Performance	Default
Select Method on page 599	0 - Cursor
Prepare Method on page 594	1 - Partial
Fetch Array Size on page 582	50
Packet Size on page 593	0
Connection Cache Size on page 568	1

8. Optionally, click the **Failover** tab to specify failover data source settings.

Figure 56: Failover tab



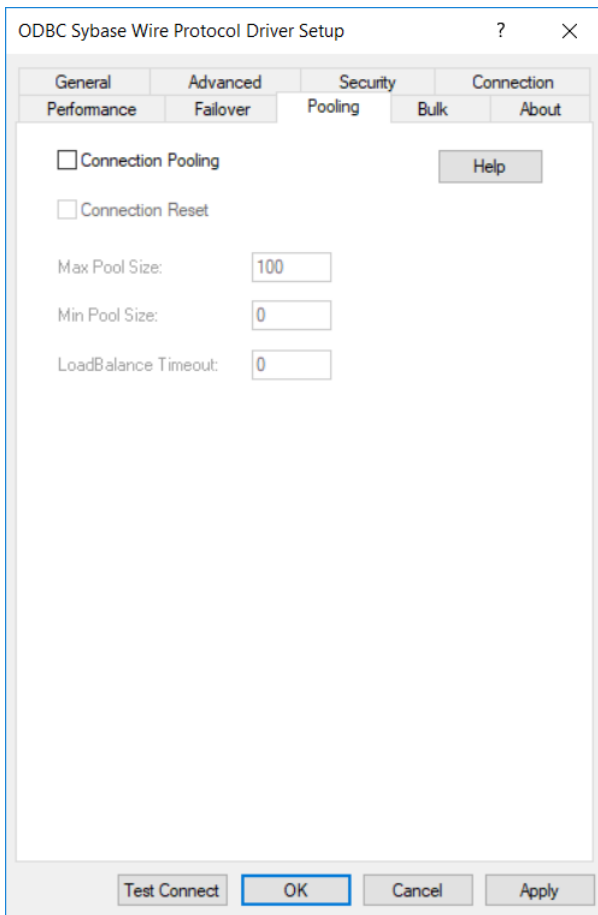
See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 590	Disabled
Connection Retry Count on page 570	0
Connection Retry Delay on page 571	3
Alternate Servers on page 563	None
Failover Mode on page 580	0 - Connection
Failover Granularity on page 579	0 - Non-Atomic
Failover Preconnect on page 581	Disabled
HA Failover Server Connection Information/Network Address on page 584	None

9. Optionally, click the **Pooling** tab to specify connection pooling data source settings.

Figure 57: Pooling tab



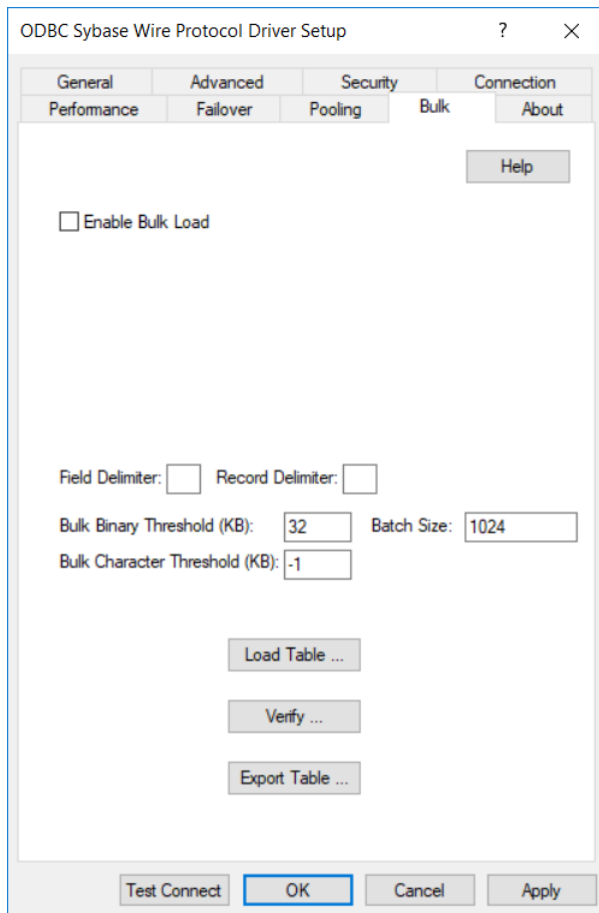
See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 569	Disabled
Connection Reset on page 569	Disabled
Max Pool Size on page 591	100
Min Pool Size on page 592	0
Load Balance Timeout on page 589	0

10. Optionally, click the **Bulk** tab to specify DataDirect Bulk Load data source settings.

Figure 58: Bulk tab



See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect Bulk Load.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Bulk	Default
Enable Bulk Load on page 577	Disabled
Field Delimiter on page 583	None
Record Delimiter on page 598	None
Bulk Binary Threshold on page 566	32
Bulk Character Threshold on page 567	-1
Batch Size on page 565	1024

If your application is already coded to use parameter array batch functionality, you can leverage DataDirect Bulk Load features through the Enable Bulk Load connection option. Enabling this option automatically converts the parameter array batch operation to use the database bulk load protocol.

If you are not using parameter array batch functionality, you can export data to a bulk load data file, verify the metadata of the bulk load configuration file against the structure of the target table, and bulk load data to a table. Use the following steps to accomplish these tasks.

- a) To export data from a table to a bulk load data file, click **Export Table** from the Bulk tab. The Export Table dialog box appears.

Figure 59: ODBC Sybase Wire Protocol Export Table Driver Setup dialog box

Table Name: A string that specifies the name of the source database table containing the data to be exported.

Export Filename: A string that specifies the path (relative or absolute) and file of the bulk load data file to which the data is to be exported. It also specifies the file name of the bulk configuration file. The file name must be the fully qualified path to the bulk data file. These files must not already exist; if one of both of them already exists, an error is returned.

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. The log file is created if it does not exist. Events logged to this file are:

- Total number of rows fetched
- A message for each row that failed to export
- Total number of rows that failed to export
- Total number of rows successfully exported

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not supply a value for Log Filename, no log file is created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

Code Page: A value that specifies the code page value to which the driver must convert all data for storage in the bulk data file. See [Character Set Conversions](#) on page 108 for more information.

The default value on Windows is the current code page of the machine. On UNIX/Linux, the default value is 4 (ISO 8559-1 Latin-1).

Click **Export Table** to connect to the database and export data to the bulk data file or click **Cancel**.

- b) To verify the metadata of the bulk load configuration file against the structure of the target database table, click **Verify** from the Bulk tab. See [Verification of the Bulk Load Configuration File](#) on page 106 for details. The Verify dialog box appears.

Figure 60: ODBC Sybase Wire Protocol Verify Driver Setup dialog box

Table Name: A string that specifies the name of the target database table into which the data is to be loaded.

Configuration Filename: A string that specifies the path (relative or absolute) and file name of the bulk configuration file.

Click **Verify** to verify table structure or click **Cancel**.

- c) To bulk load data from the bulk data file to a database table, click **Load Table** from the Bulk tab. The Load File dialog box appears.

Figure 61: ODBC Sybase Wire Protocol Load File Driver Setup dialog box

Table Name: A string that specifies the name of the target database table into which the data is loaded.

Load Data Filename: A string that specifies the path (relative or absolute) and file name of the bulk data file from which the data is loaded.

Configuration Filename: A string that specifies the path (relative or absolute) and file name of the bulk configuration file..

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. Specifying a value for Log Filename creates the file if it does not already exist. Events logged to this file are:

- Total number of rows read
- Message for each row that failed to load
- Total number of rows that failed to load
- Total number of rows successfully loaded

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Log Filename, no log file is created.

Discard Filename: A string that specifies the path (relative or absolute) and file name of the bulk discard file. The file name must be the fully qualified path to the discard file. Any row that cannot be inserted into database as result of bulk load is added to this file, with the last row rejected added to the end of the file.

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Discard Filename, a discard file is not created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Load Start: A value that specifies the first row to be loaded from the data file. Rows are numbered starting with 1. For example, when Load Start is 10, the first 9 rows of the file are skipped and the first row loaded is row 10. This option can be used to restart a load after a failure.

The default value is 1.

Read Buffer Size (KB): A value that specifies the size, in KB, of the buffer that is used to read the bulk data file for a bulk load operation.

The default value is 2048.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

Load Count: A value that specifies the number of rows to be loaded from the data file. The bulk load operation loads rows up to the value of Load Count from the file to the database. It is valid for Load Count to specify more rows than exist in the data file. The bulk load operation completes successfully when either the number of rows specified by the Load Count value has been loaded or the end of the data file is reached. This option can be used in conjunction with Load Start to restart a load after a failure.

The default value is the maximum value for SQLULEN. If set to 0, no rows are loaded.

Click **Load Table** to connect to the database and load the table or click **Cancel**.

11. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Sybase\)](#) on page 558 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

12. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={}{driver_name[]}[:attribute=value[:attribute=value]...]
```

[Connection Option Descriptions for Sybase Wire Protocol](#) on page 558 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Sybase is:

```
DSN=SYB TABLES;DB=PAYROLL;UID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=SYB.dsn;DB=PAYROLL;UID=JOHN;PWD=XYZZY
```

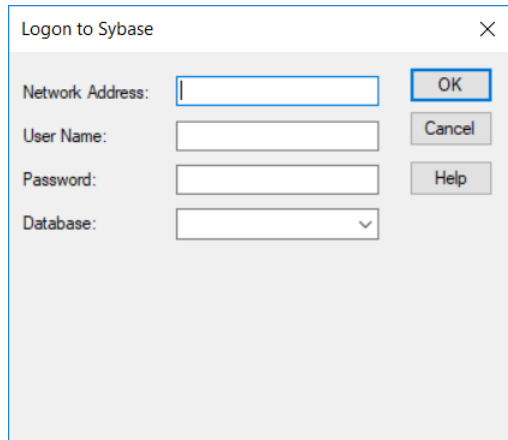
A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Sybase Wire Protocol};NA=123.456.78.90,5000;DB=SYBACCT;UID=JOHN;PWD=XYZZY
```

Using a Logon Dialog Box (Sybase)

Some ODBC applications display a Logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Figure 62: Logon to Sybase Dialog Box



In the Logon dialog box, provide the following information:

1. In the Network Address field, specify an IP address for the Sybase server as follows: *IP address,port_number*. For example, you might enter *199.226.224.34,5000*. If your network supports named servers, you can specify an address as: *servername,port_number*. For example, you might enter *Sybaseserver,5000*.

The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details concerning these formats.

2. If required, type your case-sensitive login ID.
3. If required, type your case-sensitive password for the system.
4. In the Database field, type the name of the database you want to access (case-sensitive). Or, select the name from the Database drop-down list, which displays the names that you specified on the Connection tab of the ODBC Sybase Wire Protocol driver Setup dialog box.

Note: If you are connecting through the **Test Connect** button of the Setup dialog box, only the default database specified on the General tab of the Setup dialog box is available in the Database drop-down list. The database names specified on the Connection tab are not available.

5. Click **OK** to complete the logon and to update the values in the Registry.

Connection Option Descriptions for Sybase Wire Protocol

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Sybase Wire Protocol driver.

Table 37: Sybase Wire Protocol Attribute Names

Attribute (Short Name)	Default
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
AlternateServers (ASRV)	None
ApplicationName (APP)	None
ApplicationUsingThreads (AUT)	1 (Enabled)
ArraySize (AS)	50
AuthenticationMethod (AM)	0 (No Encryption)
BulkBinaryThreshold (BBT)	32
BulkCharacterThreshold (BCT)	-1
BulkLoadBatchSize (BLBS)	1024
BulkLoadFieldDelimiter (BLFD)	None
BulkLoadRecordDelimiter (BLRD)	None
Charset (CS)	None
ConnectionReset (CR)	0 (Disabled)
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
CursorCacheSize (CCS)	1
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Database (DB)	None
Database List	None
DataSourceName (DSN)	None
DefaultLongDataBuffLen (DLDBL)	1024
Description (n/a)	None

Attribute (Short Name)	Default
DistributedTransactionModel (DTM)	0 (XA Protocol)
EnableBulkLoad (EBL)	0 (Disabled)
EnableDescribeParam (EDP)	0 (Disabled)
EnableQuotedIdentifiers (EQI)	0 (Disabled)
EncryptionMethod (EM)	0 (No Encryption)
FailoverGranularity (FG)	0 (Non-Atomic)
FailoverMode (FM)	0 (Connection)
FailoverNetworkAddress (FNA)	None
FailoverPreconnect (FP)	0 (Disabled)
FetchTWFSasTime (FTWFSAT)	1
GSSClient (GSSC)	native
HostNameInCertificate (HNIC)	None
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
InitializationString (IS)	None
InterfacesFile (IF)	None
InterfacesFileServerName (IFSN)	None
KeepAlive (KA)	Disabled
Language	None
LoadBalanceTimeout (LBT)	None
LoadBalancing (LB)	0
LoginTimeout (LT)	15
LogonID (UID)	None
MaxPoolSize (MXPS)	100
MinPoolSize (MNPS)	0
NetworkAddress (NA)	None
OptimizePrepare (OP)	1 (Partial)

Attribute (Short Name)	Default
PacketSize (PS)	0
Password (PWD)	None
Pooling (POOL)	0 (Disabled)
PRNGSeedFile (PSF) UNIX/Linux only	/dev/random
PRNGSeedSource (PSS) UNIX/Linux only	0 (File)
QueryTimeout (QT)	0
RaiseErrorPositionBehavior (REPB)	0 (Default)
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
SelectMethod (SM)	0 (Cursor)
ServicePrincipalName (SPN)	None
SSLlibName (SLN)	Empty string
TightlyCoupledDistributedTransactions (TCDT)	1 (Enabled)
TruncateTimeTypeFractions (TTTF)	0 (Disabled)
Truststore (TS)	None
TruststorePassword (TSP)	Password
ValidateServerCertificate (VSC)	1 (Enabled)
WorkstationID (WKID)	None
XAOpenStringParameters (XAOSP)	None

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

`latest | openssl_version_number[[, openssl_version_number] ...]`

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to `openssl_version_number`, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the `CryptoLibName` and `SSLLibName` options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

`1.1.1,1.0.2`

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

[Advanced tab](#)

Alternate Servers

Attribute

AlternateServers (ASRV)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
( {NetworkAddress=addressvalue | InterfacesFileName=sectionvalue}[, ...])
```

NetworkAddress and InterfacesFileName can be used in the same string.

You must specify the network address of each alternate database server or the section in the Interfaces file that contains the network connection information for the Sybase database server you want to access (InterfacesFileName).

Notes

- An alternate server address in IPv6 format must be enclosed in double quotation marks.
- The Alternate Servers option and the HA Failover Server Connection Information option are mutually exclusive.

Example

The following example Alternate Servers values define three alternate database servers for connection failover:

```
( InterfacesFileName=Accounting, NetworkAddress="255.125.1.11, 4200",  
  NetworkAddress="SybaseASE2, 4200" )
```

In this example, the network address of the last two alternates contain commas. In this case, enclose the network address with double quotation marks as shown.

Default

None

GUI Tab

[Failover tab](#)

Application Name

Attribute

ApplicationName (APP)

Purpose

The name used by Sybase to identify your application.

Valid Values

string

where:

`string`

is a valid application name.

Default

None

GUI Tab

[Connection tab](#)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Authentication Method

Attribute

AuthenticationMethod (AM)

Purpose

Specifies the method the driver uses to authenticate the user to the server when a connection is established. If the specified authentication method is not supported by the database server, the connection fails and the driver generates an error.

Valid Values

0 | 1 | 4

Behavior

If set to 0 (No Encryption), the driver sends the user ID and password in clear text to the server for authentication.

If set to 1 (Encrypt Password), the driver negotiates a secure login with the database server by sending an encrypted password. The password is encrypted using Sybase Extended Password Encryption, Sybase Extended Plus Encrypted Password, or Sybase proprietary encryption, depending on the response by the server. Note that the proprietary Sybase password must contain characters from the 7-bit ASCII set.

If set to 4 (Kerberos Authentication), the driver uses Kerberos authentication. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

If the specified authentication method is not supported by the database server, the connection fails and the driver generates an error.

Notes

- Sybase Extended Password Encryption and Sybase Extended Plus Encrypted Password, which use an asymmetrical key type, provide stronger password encryption for the secure transmission of public key passwords over networks than the proprietary encryption.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

Batch Size

Attribute

BulkLoadBatchSize (BLBS)

Purpose

The number of rows that the driver sends to the database at a time during bulk operations. This value applies to all methods of bulk loading.

Valid Values

0 | x

where

x

is the number of rows to send during a bulk operation.

Notes

- This connection option can affect performance.

Default

1024

GUI Tab

[Bulk tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Bulk Binary Threshold

Attribute

BulkBinaryThreshold (BBT)

Purpose

The maximum size, in KB, of binary data that is exported to the bulk data file.

Valid Values

-1 | 0 | x

where

x

is an integer that specifies the number of KB.

Behavior

If set to -1, all binary data, regardless of size, is written to the bulk data file, not to an external file.

If set to 0, all binary data, regardless of size, is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

If set to x, any binary data exceeding this specified number of KB is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

Notes

- This connection option can affect performance.

Default

32

GUI Tab

[Bulk tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Bulk Character Threshold**Attribute**

BulkCharacterThreshold (BCT)

Purpose

The maximum size, in KB, of character data that is exported to the bulk data file.

Valid Values

-1 | 0 | *x*

where:

x

is an integer that specifies the number of KB.

Behavior

If set to -1, all character data, regardless of size, is written to the bulk data file, not to an external file.

If set to 0, all character data regardless of size, is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

If set to *x*, any character data exceeding this specified number of KB is written to an external file, not the bulk data file. A reference to the external file is written to the bulk data file.

Default

-1

GUI Tab

[Bulk tab](#)

Charset**Attribute**

Charset (CS)

Purpose

The name of a character set installed on the Sybase server to be used by the driver.

This option is not a substitute for the IANAAppCodePage option. See IANAAppCodePage for details.

Valid Values

charset

where:

charset

is the name of a character set installed on the Sybase server.

If unspecified, the character set setting on the Sybase server is used.

For the driver to return Unicode SQL types for connections to Sybase 12.5 and higher, use a value of UTF-8. Refer to the Sybase server documentation for a list of valid character sets.

Example

If your client needs to receive data in iso-8859-1 from a non-Unicode Sybase server, you would specify a value of iso_1.

Default

None

GUI Tab

[Connection tab](#)

Connection Cache Size

Attribute

CursorCacheSize (CCS)

Purpose

The number of connections that the connection cache can hold.

Valid Values

x

where:

x

is a positive integer representing the number of connections that the connection cache can hold.

To enable the connection cache, you must set the Select Method option to 1 (enabled). Increasing the connection cache may increase performance of some applications but requires additional database resources.

Default

1

GUI Tab

[Performance tab](#)

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Connection Reset

Attribute

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Connection Retry Count

Attribute

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x , the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay (CRD)

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to x , the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (EncryptionMethod=1). When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, driver behavior is determined by the Encryption Method connection option.

Valid Values

cryptographic_protocol [, *cryptographic_protocol*]...

where:

cryptographic_protocol

is one of the following cryptographic protocols:

TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Notes

- This option is ignored if Encryption Method is set to 0 - No Encryption.
- Consult your database administrator concerning the data encryption settings of your server.

Default

```
TLSv1.2, TLSv1.1, TLSv1
```

GUI Tab

[Security tab](#)

See also

[Encryption Method](#) on page 579

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Progress\DataDirect\Connect64_for_ODBC_71\  
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLibName](#) on page 601

Cursor Positioning for Raiserror

Attribute

`RaiseErrorPositionBehavior` (REPB)

Purpose

Determines whether the driver returns raiserrors when the next statement is executed or handles them separately.

Valid Values

0 | 1

Behavior

If set to 0 (Default), raiserrors are handled separately from surrounding statements. The error is returned when a raiserror is processed (for example, resulting from SQLExecute, SQLExecDirect, or SQLMoreResults). The result set is empty.

If set to 1 (Microsoft compatible), raiserrors are returned when the next statement is processed, and the cursor is positioned on the first row of the subsequent result set. This could result in multiple raiserrors being returned on a single execute.

Default

0 (Default)

GUI Tab

[Advanced tab](#)

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database List

Attribute

n/a

Purpose

A list of database names that will appear in the drop-down list of the logon dialog box (see [Using a Logon Dialog Box \(Sybase\)](#) on page 558 for a description).

Valid Values

database_list

where:

database_list

is a comma-separated list of database names that will appear in the drop-down list of the logon dialog box.

Default

None

GUI Tab

[Connection tab](#)

Database Name

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where:

database_name

is the name of a valid database.

Default

None

GUI Tab

[General tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Default Buffer Size for Long/LOB Columns (in Kb)

Attribute

DefaultLongDataBuffLen (DLDBL)

Purpose

The maximum length of data (in KB) the driver can fetch from long columns in a single round trip and the maximum length of data that the driver can send using the SQL_DATA_AT_EXEC parameter.

This option also applies to binding long parameters in chunks. The driver truncates any data passed in a Long/LOB SQL_DATA_AT_EXEC parameter to the size specified.

Valid Values

An integer in multiples of 1024

The value must be in multiples of 1024 (for example, 1024, 2048). You need to increase the default value if the total size of any Long data exceeds 1 MB. This value is multiplied by 1024 to determine the total maximum length of fetched data. For example, if you enter a value of 2048, the maximum length of data would be 1024 x 2048, or 2097152 (2 MB).

Notes

- This connection option can affect performance.

Default

1024

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Distributed Transaction Model

Attribute

DistributedTransactionModel (DTM)

Purpose

The model to use for distributed transaction support. The driver supports two different models: XA Protocol and Native OLE.

Valid Values

0 | 1

Specify the appropriate distributed transaction protocol, either 0 (XA Protocol) or 1 (Native OLE)

Default

0 (XA Protocol)

GUI Tab

[Advanced tab](#)

Enable Bulk Load

Attribute

EnableBulkLoad (EBL)

Purpose

Specifies the bulk load method.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses the database bulk load protocol when an application executes an INSERT with multiple rows of parameter data. If the protocol cannot be used, the driver returns a warning.

If set to 0 (Disabled), the driver uses standard parameter arrays.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Bulk tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Enable Describe Parameter

Attribute

EnableDescribeParam (EDP)

Purpose

Determines whether the driver supports the SQLDescribeParam function, which allows an application to describe parameters in SQL statements and in stored procedure calls.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver supports SQLDescribeParam. The Prepare Method option must be set to 0 or 1, and the SQL statement must not include long parameters. If using Microsoft Remote Data Objects (RDO) to access data, you must use this value.

If set to 0 (Disabled), the driver does not support SQLDescribeParam.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Enable Quoted Identifiers

Attribute

EnableQuotedIdentifiers (EQI)

Purpose

Determines whether the driver supports the use of quoted identifiers.

Valid Values

0 | 1

If set to 1 (Enabled), the driver supports the use of quoted identifiers. Double quotation marks (") must be used to enclose identifiers, such as column and table names. Character strings must be enclosed in single quotation marks, for example:

```
SELECT "au_id"  
FROM "authors"  
WHERE "au_lname" = 'O'Brien'
```

If set to 0 (Disabled), the driver does not support the use of quoted identifiers and generates an error when quoted identifiers are encountered.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Encryption Method**Attribute**

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server. If the specified encryption method is not supported by the database server, the connection fails and the driver returns an error.

Valid Values

0 | 1

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using the SSL protocols specified in the Crypto Protocol Version connection option.

This option can only be set to 1 when Authentication Method is set to 0 or 1.

Notes

- This connection option can affect performance.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

See Also

[Crypto Protocol Version](#) on page 571

[Performance Considerations](#) on page 607

Failover Granularity**Attribute**

FailoverGranularity (FG)

Purpose

Determines whether the driver fails the entire failover process or continues with the process if errors occur while trying to reestablish a lost connection.

This option applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select).

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 (Non-Atomic), the driver continues with the failover process and posts any errors on the statement on which they occur.

If set to 1 (Atomic) the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

If set to 2 (Atomic Including Repositioning), the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

If set to 3 (Disable Integrity Check), the driver does not verify that the rows that were restored during the failover process match the original rows. This value applies only when Failover Mode is set to 2 (Select).

Default

0 (Non-Atomic)

GUI Tab

[Failover tab](#)

Failover Mode

Attribute

FailoverMode (FM)

Purpose

Specifies the type of failover method the driver uses.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Connection), the driver provides failover protection for new connections only.

If set to 1 (Extended Connection), the driver provides failover protection for new and lost connections, but not any work in progress.

If set to 2 (Select), the driver provides failover protection for new and lost connections. In addition, it preserves the state of work performed by the last Select statement executed.

Notes

- This connection option can affect performance.

Default

0 (Connection)

GUI Tab

[Failover tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Failover Preconnect

Attribute

FailoverPreconnect (FP)

Purpose

Specifies whether the driver tries to connect to the primary and an alternate server at the same time.

This attribute applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select) and at least one alternate server is specified.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver tries to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection. This value provides the best performance, but your application typically experiences a short wait while the failover connection is attempted.

If set to 1 (Enabled), the driver tries to connect to the primary and an alternate server at the same time. This can be useful if your application is time-sensitive and cannot absorb the wait for the failover connection to succeed.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Fetch Array Size

Attribute

ArraySize (AS)

Purpose

The number of rows the driver retrieves from the server for a fetch. This is not the number of rows given to the user. This connection option can affect performance.

Valid Values

x

where:

x

is a positive integer specifying the number of rows.

Notes

- This connection option can affect performance.

Default

50

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Fetch TWFS as Time

Attribute

FetchTWFSasTime (FTWFSAT)

Purpose

Determines which ODBC data type the driver uses to return column values with the BIGTIME data type.

Valid Values

0 | 1

Behavior

If set to 1, the driver returns column values for the BIGTIME data type as the ODBC data type SQL_TYPE_TIME. The fractional seconds portion of the value is truncated.

If set to 0, the driver returns column values for the BIGTIME data type as the ODBC data type SQL_TYPE_TIMESTAMP. When a timestamp is returned for BIGTIME, the Year, Month and Day parts of the timestamp must be set to zero.

Notes

- The BIGTIME data type is supported in Sybase 15.5 and higher.

Default

1

GUI Tab

[Advanced tab](#)

Field Delimiter

Attribute

BulkLoadFieldDelimiter (BLFD)

Purpose

Specifies the character that the driver will use to delimit the field entries in a bulk load data file.

Valid Values

x

where:

x

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash(-), the colon(:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Field Delimiter character must be different from the Bulk Load Record Delimiter.

Default

None

GUI Tab

[Bulk tab](#)

GSS Client Library

Attribute

GSSClient (GSSC)

Purpose

The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC). The driver uses the path defined by the PATH environment variable for loading the specified client library.

Valid Values

native | *client_library*

where:

client_library

is a GSS client library installed on the client.

Behavior

If set to *client_library*, the driver uses the specified GSS client library.

If set to native, the driver uses the GSS client shipped with the operating system.

Default

native

GUI Tab

[Security tab](#)

HA Failover Server Connection Information/Network Address

Attribute

FailoverNetworkAddress (FNA)

Purpose

The network address of the High Availability (HA) Failover server to be used in the event of a connection loss. The driver detects the dropped connection and automatically reconnects to the specified HA Failover server. This option is valid only for Sybase 12 and higher servers that have the High Availability Failover feature enabled.

Valid Values

IP_address , *port_number* | *pipe_address* , *port_number* | *server_name* , *port_number*

where:

IP_address

is the IP address that uniquely identifies the HA Failover server.

port_number

is the port number assigned to the listener process on the HA Failover server.

server_name

is a name that uniquely identifies the HA Failover server. You can use this format if your environment supports named servers.

pipe_address

is the pipe address of the HA Failover server. This format is required if using NamedPipes as the network protocol.

Notes

- The HA Failover Server Connection Information option and the Alternate Servers option are mutually exclusive.

Example

199.226.224.34, 5000

or

\\machine1\sybase\pipe\query, 5000

or

Sybaseserver, 5000

Default

None

GUI Tab

[Failover tab](#)

Host Name In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

host_name | #SERVERNAME#

where:

host_name

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If set to a host name, the driver examines the subjectAltName values included in the certificate. If a dnsName value is present in the subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the dnsName value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the dnsName value.

If no subjectAltName values exist or a dnsName value is not in the list of subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the commonName part of the Subject name in the certificate. The commonName typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the commonName. If multiple commonName parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the commonName parts.

If set to *#SERVERNAME#*, the driver compares the host server name specified as part of a data source or connection string to the dnsName or the commonName value.

Default

None

GUI Tab

[Security tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Initialization String

Attribute

InitializationString (IS)

Purpose

A SQL command that is issued immediately after connecting to the database to manage session settings.

Valid Values

SQL_command

where:

SQL_command

is a valid SQL command that is supported by the database.

Notes

- If the statement fails to execute, the connection fails and the driver reports the error returned from the server.

Example

To allow delimited identifiers, specify:

```
InitializationString=set QUOTED_IDENTIFIER on
```

Default

None

GUI Tab

[Advanced tab](#)

Interfaces File

Attribute

InterfacesFile (IF)

Purpose

The directory to the Interfaces file.

Valid Values

file_dir

where:

file_dir

is the directory to the Interfaces file.

Behavior

If unspecified and a value is specified for the Server Name option, the driver looks for the path name of the Interfaces file in the Registry under HKEY_LOCAL_MACHINE\SOFTWARE\DataDirect\InterfacesFile. If this Registry value is empty, the driver will try to open the SQL.INI file found in the same directory where the driver is located and use it as the Interfaces file.

Notes

- This option and the Network Address option are mutually exclusive.

Default

None

GUI Tab

[General tab](#)

Language

Attribute

Language (LANG)

Purpose

The national character set installed on the Sybase server.

Valid Values

charset

where:

charset

is the national character set installed on the Sybase server.

Default

None (English)

GUI Tab

[Connection tab](#)

Load Balance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

The number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | *x*

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to *x*, inactive connections are closed after the specified number of seconds passes.

Notes

- The Min Pool Size option may cause some connections to ignore this value.
- This connection option can affect performance.

Default

0 (Disabled)

See also

See [Performance Considerations](#) on page 607 for details.

GUI Tab

[Pooling tab](#)

Load Balancing

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x, the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

15

GUI Tab

[Advanced tab](#)

Max Pool Size

Attribute

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

For example, if set to 20, the maximum number of connections allowed in the pool is 20.

Notes

- This connection option can affect performance.

Default

100

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Min Pool Size

Attribute

MinPoolSize (MNPS)

Purpose

The minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

This connection option can affect performance. See Performance Considerations for details.

Valid Values

0 | *x*

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

Notes

- This connection option can affect performance.

Example

If set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Default

0

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Network Address

Attribute

NetworkAddress (NA)

Purpose

A unique identifier assigned to the Sybase server machine.

Valid Values

server_name | *IP_address*

where:

server_name

is the Sybase server name specified as: *named_server, port_number*. For example, you can enter *SSserver, 5000*.

IP_address

is the Sybase server address specified as: *IP_address, port_number*. For example, you can enter *199.226.224.34, 5000*. The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details about these formats.

Notes

This option is mutually exclusive with the Interfaces File and the Server Name option.

Default

None

GUI Tab

[General tab](#)

Packet Size

Attribute

PacketSize (PS)

Purpose

Determines the number of bytes for each database protocol packet that is transferred from the database server to the client machine. Adjusting the packet size can improve performance. The optimal value depends on the typical size of data that is inserted, updated, or returned by the application and the environment in which it is running. Typically, larger packet sizes work better for large amounts of data. For example, if an application regularly returns character values that are 10,000 characters in length, using a value of 32 (16 KB) typically results in improved performance.

Valid Values

-1 | 0 | *x*

Behavior

If set to -1, the driver uses the maximum packet size that is set by the database server.

If set to 0, the driver uses the default packet size that is used by the database server.

If set to *x*, an integer from 1 to 127, the driver uses a packet size that is a multiple of 512 bytes. For example, PacketSize=8 means to set the packet size to 8 * 512 bytes (4096 bytes).

Notes

- If SSL encryption is used, the driver must use the packet size that is specified by the server. Any value set for this option or the SQL_PACKET_SIZE connect option is ignored if SSL encryption is used.
- The ODBC connection option SQL_PACKET_SIZE provides the same functionality as the Packet Size option; however SQL_PACKET_SIZE and the Packet Size option are mutually exclusive. If Packet Size is

specified, the driver returns the message Driver Not Capable if an application attempts to call SQL_PACKET_SIZE. If you do not set the Packet Size option, application calls to SQL_PACKET_SIZE are accepted by the driver.

- This connection option can affect performance.

Default

0

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Prepare Method

Attribute

OptimizePrepare (OP)

Purpose

Determines whether stored procedures are created on the server for calls to SQLPrepare.

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 - None, stored procedures are created for every call to SQLPrepare. This setting can result in decreased performance when processing statements that do not contain parameters.

If set to 1 - Partial, the driver creates stored procedures only if the statement contains parameters. Otherwise, the statement is cached and run directly at the time of SQLExecute.

If set to 2 - Full, stored procedures are never created. The driver caches the statement, executes it directly at the time of SQLExecute, and reports any syntax or similar errors at the time of SQLExecute.

If set to 3 - Full at Prepare, stored procedures are never created. This is identical to value 2 except that any syntax or similar errors are returned at the time of SQLPrepare instead of SQLExecute. Use this setting only if you must have syntax errors reported at the time of SQLPrepare.

Notes

- This connection option can affect performance.

Default

1 (Partial)

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

PRNGSeedFile

Attribute

PRNGSeedFile (PSF)

Purpose

UNIX[®] Specifies the absolute path for the entropy-source file or device used as a seed for SSL key generation.

Valid Values

string | RANDFILE

where:

string

is the absolute path for the entropy-source file or device that seeds the random number generator used for TLS/SSL key generation.

Behavior

If set to *string*, the specified entropy-source file or device seeds the random number generator used for TLS/SSL key generation. Entropy levels and behavior may vary for different files and devices. See the following section for a list of commonly used entropy sources and their behavior.

If set to `RANDFILE`, the `RAND_file_name()` function in your application generates a default path for the random seed file. The seed file is `$RANDFILE` if that environment variable is set; otherwise, it is `$HOME/.rnd`. If `$HOME` is not set either, an error occurs.

Common Valid Values

Although other entropy-source files may be specified, the following valid values are for files and devices that are commonly used for seeding:

`/dev/random`

is a pseudorandom number generator (blocking) that creates a seed from random bits of environmental noise it collects in an entropy pool. When there is insufficient noise in the pool, the file blocks calls until enough noise is collected. This provides more secure SSL key generation, but at the expense of blocked calls.

`/dev/urandom`

is a pseudorandom number generator (non-blocking) that creates seeds from random bits from environmental noise it collects in an entropy pool. When there is insufficient noise in the pool, the file reuses bits from the pool instead of blocking calls. This eliminates potential delays associated with blocked calls, but may result in less secure TLS/SSL key generation.

`/dev/hwrng`

is a hardware random number generator. The behavior is dependent on the device used in your environment.

Notes

- This option is ignored when TLS/SSL is disabled (`EncryptionMethod=0`) or the seed source is set to Poll Only (`PRNGSeedSource=1`).
- For processes that employ multiple TLS/SSL-enabled drivers, the behavior of this option for all drivers is determined by the values specified for the driver that first connects to the process and loads the OpenSSL library. Since the OpenSSL library loads only once per process, the values specified for drivers that subsequently connect are ignored. To ensure that the correct security settings are used, we recommend configuring this option identically for all drivers used in a process.

Default

`/dev/random`

GUI tab

NA

See also

[PRNGSeedSource](#) on page 597

PRNGSeedSource

Attribute

PRNGSeedSource (PSS)

Purpose

UNIX[®] Specifies the source of the seed the driver uses for TLS/SSL key generation. Seeds are a pseudorandom or random value used to set the initial state of the random number generator used to generate TLS/SSL keys. Using seeds with a higher level of entropy, or randomness, provides a more secure transmission of data encrypted using TLS/SSL.

Valid Values

0 | 1

Behavior

If set to 0 (File), the driver uses entropy-source file or device specified in the PRNGSeedFile connection option as the seed used for TLS/SSL key generation.

If set to 1 (Poll Only) , the driver uses the RAND_poll function in TLS/SSL to create the seed used for TLS/SSL key generation.

Notes

- For processes that employ multiple TLS/SSL-enabled drivers, the behavior of this option for all drivers is determined by the values specified for the driver that first connects to the process and loads the OpenSSL library. Since the OpenSSL library loads only once per process, the values specified for drivers that subsequently connect are ignored. To ensure that the correct security settings are used, we recommend configuring this option identically for all drivers used in a process.
- This option is ignored when TLS/SSL is disabled (`EncryptionMethod=0`)

Default

0 (File)

GUI Tab

NA

See also

[PRNGSeedFile](#) on page 595

Query Timeout

Attribute

QueryTimeout (QT)

Purpose

The number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the `SQL_ATTR_QUERY_TIMEOUT` statement attribute on the `SQLSetStmtAttr()` function.

Valid Values

-1 | 0 | *x*

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to -1, the query does not time out. The driver silently ignores the `SQL_ATTR_QUERY_TIMEOUT` attribute.

If set to 0, the query does not time out, but the driver responds to the `SQL_ATTR_QUERY_TIMEOUT` attribute.

If set to *x*, all queries time out after the specified number of seconds unless the application overrides this value by setting the `SQL_ATTR_QUERY_TIMEOUT` attribute.

Default

0

GUI Tab

[Advanced tab](#)

Record Delimiter

Attribute

`BulkLoadRecordDelimiter` (BLRD)

Purpose

Specifies the character that the driver will use to delimit the record entries in a bulk load data file.

Valid Values

x

where:

x

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash (-), the colon (:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Record Delimiter character must be different from the Bulk Load Field Delimiter.

Default

None

GUI Tab

[Bulk tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where `x` is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Advanced tab](#)

Select Method

Attribute

SelectMethod (SM)

Purpose

Determines whether database cursors are used for Select statements.

Valid Values

0 | 1

If set to 0 (Cursor), database cursors are used. In some cases performance degradation can occur when performing large numbers of sequential Select statements because of the amount of overhead associated with creating database cursors.

If set to 1 (Direct), Select statements are run directly without using database cursors, and the data source is limited to one active statement.

Notes

- This connection option can affect performance.

Default

0 (Cursor)

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 607 for details.

Server Name

Attribute

InterfacesFileName (IFSN)

Purpose

The name of the section in the Interfaces file containing the network connection information for the Sybase server. Typically, the section name is the host name of the Sybase server.

Valid Values

section_name

where:

section_name

is a section in the Interfaces file containing the network connection information for the Sybase server.

Notes

The Network Address option and the Server Name option are mutually exclusive.

Default

None

GUI Tab

General tab

Service Principal Name

Attribute

ServicePrincipalName (SPN)

Purpose

The service principal name to be used by driver for Kerberos authentication.

Valid Values

servicePrincipalName

where:

servicePrincipalName

is a valid service principal name.

If unspecified, the value of the Network Address option is used as the service principal name. If Authentication Method is set to 0 or 1, the value of the Service Principal Name option is ignored.

Default

None

GUI Tab

Security tab

SSLibName

Attribute

SSLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\ODBC_71\  
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
SSLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 572

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Tightly Coupled Distributed Transactions

Attribute

TightlyCoupledDistributedTransactions (TCDT)

Purpose

Sybase 12 or higher server only. Determines whether the driver ensures that multiple connections within the same distributed transaction obey other's locks.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses tightly coupled distributed transactions. Multiple connections within the same distributed transaction obey other's locks.

If set to 0 (Disabled), the driver does not use tightly coupled distributed transactions. Multiple connections within the same distributed transaction may hang each other because the connections do not obey other's locks. This value can provide better performance if concurrency of data is not needed.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Truncate Time Type Fractions

Attribute

TruncateTimeTypeFractions (TTTF)

Purpose

Sybase 12.5.1 and higher only. Determines whether the driver sets fractional seconds to zero (0) when converting data from the TIME data type to TIMESTAMP, CHAR, or WCHAR data types.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver converts fractional seconds to zero when converting the TIME data type.

If set to 0 (Disabled), the driver does not set fractional seconds to zero when converting the TIME data type.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Truststore

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used.

Valid Values

truststore_directory\filename

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Notes

- The truststore and keystore files may be the same file.

Default

None

GUI Tab

[Security tab](#)

Truststore Password

Attribute

TruststorePassword (TSP)

Purpose

The password that is used to access the truststore file when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

The truststore and keystore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Security tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Truststore information is specified using the Trust Store and Trust Store Password options.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Default

1 (Enabled)

GUI Tab

[Security tab](#)

Workstation ID

Attribute

WorkstationID (WKID)

Purpose

An identifier for the client machine.

Valid Values

ID

where:

ID

is workstation ID use by the client machine.

Default

None

GUI Tab

[Connection tab](#)

XA Open String Parameters

Attribute

XAOpenStringParameters (XAOSP)

Purpose

Determines the name of trace files generated for XA open string parameters.

Valid Values

-Ltrace_filename

where:

trace_filename

is a string that identifies trace files generated for XA open string parameters. If specified, two trace files are created. The first trace file traces all XA call activities and is named exactly as specified. The second trace file traces any enlistment and unenlistment procedures and is named as specified with a "driver" extension.

Example

If you specify `-LXAtrace`, the driver creates two trace files: XAtrace and XAtrace.driver.

Default

None

GUI Tab

[Advanced tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Connection Pooling (Pooling): If you enable the driver to use connection pooling, you can set additional options that affect performance:

- **Load Balance Timeout (LoadBalanceTimeout):** You can define how long to keep connections in the pool. The time that a connection was last used is compared to the current time and, if the timespan exceeds the value of the Load Balance Timeout option, the connection is destroyed. The Min Pool Size option can cause some connections to ignore this value.
- **Connection Reset (ConnectionReset):** Resetting a re-used connection to the initial configuration settings impacts performance negatively because the connection must issue additional commands to the server.
- **Max Pool Size (MaxPoolSize):** Setting the maximum number of connections that the pool can contain too low might cause delays while waiting for a connection to become available. Setting the number too high wastes resources.
- **Min Pool Size (MinPoolSize):** A connection pool is created when the first connection with a unique connection string connects to the database. The pool is populated with connections up to the minimum pool size, if one has been specified. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Default Buffer Size for Long/LOB Columns (DefaultLongDataBuffLen): To improve performance when your application fetches images, pictures, or long text or binary data, a buffer size can be set to accommodate the maximum size of the data. The buffer size should only be large enough to accommodate the maximum amount of data retrieved; otherwise, performance is reduced by transferring large amounts of data into an oversized buffer. If your application retrieves more than 1 MB of data, the buffer size should be increased accordingly.

Enable Bulk Load (EnableBulkLoad): If your application performs bulk loading of data, you can improve performance by configuring the driver to use the database system's bulk load functionality instead of database array binding. The trade-off to consider for improved performance is that using the bulk load functionality can bypass data integrity constraints.

Encryption Method (EncryptionMethod): Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) that is required to encrypt and decrypt data.

Failover Mode (FailoverMode): Although high availability that replays queries after a failure provides increased levels of protection, it can adversely affect performance because of increased overhead.

Fetch Array Size (ArraySize): If the Select Method connection option is set to 0 and your application fetches more than 50 rows at a time, you should set Fetch Array Size to the approximate number of rows being fetched. This reduces the number of round trips on the network, thereby increasing performance. For example, if your application normally fetches 200 rows, it is more efficient for the driver to fetch 200 rows at one time over the network than to fetch 50 rows at a time during four round trips over the network. You should use Fetch Array Size in conjunction with Select Method.

NOTE: The ideal setting for your application will vary. To calculate the ideal setting for this option, you must know the size in bytes of the rows that you are fetching and the size in bytes of your Network Packet. Then, you must calculate the number of rows that will fit in your Network Packet, leaving space for packet overhead. For example, suppose your Network Packet size is 1024 bytes and the row size is 8 bytes. Dividing 1024 by 8 equals 128; however, the ideal setting for Fetch Array Size is 127, not 128, because the number of rows times the row size must be slightly smaller than the Network Packet size.

Packet Size (PacketSize): Typically, it is optimal for the client to use the maximum packet size that the database server allows. This reduces the total number of round trips required to return data to the client, thus improving performance. Therefore, performance can be improved if the PacketSize attribute is set to the maximum packet size of the Sybase ASE server.

Prepare Method (OptimizePrepare): If your application executes the same SQL statements multiple times, performance can be improved by creating a stored procedure on the server at prepare time. If your application executes one of these prepared statements multiple times, performance will improve because the driver created a stored procedure and executing a stored procedure is faster than executing a single SQL statement; however, if a prepared statement is only executed once or is never executed, performance can decrease. If your application executes the same SQL statements multiple times, the Prepare Method option should be set to 1.

Select Method (SelectMethod): If your application often executes a SQL statement before processing or closing the previous result set, then it uses multiple active statements per connection. The default setting (0) of this option causes the driver to use database cursors for Select statements and allows an application to process multiple active statements per connection. An active statement is defined as a statement where all the result rows or result sets have not been fetched. This can cause high overhead on the server. If your application does not use multiple active statements, however, setting Select Method to 1 will increase performance of Select statements by allowing the server to return results without using a database cursor. If this option is set to 0, it should be used in conjunction with Fetch Array Size (ArraySize). If this option is set to 1, Fetch Array Size (ArraySize) has no effect.

Data Types

The following table shows how the Sybase data types are mapped to the standard ODBC data types. [Unicode Support](#) on page 610 lists Sybase to Unicode data type mappings.

Table 38: Sybase Data Type Mapping

Sybase Data Type...	Maps to ODBC Data Type
BIGDATETIME ⁴⁰	SQL_DATETIME
BIGINT ⁴¹	SQL_BIGINT
BIGTIME ⁴⁰	SQL_DATETIME
BINARY	SQL_BINARY
BIT	SQL_BIT
CHAR	SQL_CHAR
DATE ⁴²	SQL_TYPE_DATE
DATETIME	SQL_TYPE_TIMESTAMP
DECIMAL	SQL_DECIMAL
FLOAT	SQL_FLOAT

⁴⁰ Sybase 15.5 and higher only.

⁴¹ Sybase 15 and higher only.

⁴² Sybase 12.5.1 and higher only.

Sybase Data Type...	Maps to ODBC Data Type
IMAGE	SQL_LONGVARBINARY
INT	SQL_INTEGER
MONEY	SQL_DECIMAL
NUMERIC	SQL_NUMERIC
REAL	SQL_REAL
SMALLDATETIME	SQL_TYPE_TIMESTAMP
SMALLINT	SQL_SMALLINT
SMALLMONEY	SQL_DECIMAL
SYSNAME	SQL_VARCHAR
TEXT	SQL_LONGVARCHAR
TIME ⁴²	SQL_TYPE_TIME
TIMESTAMP	SQL_VARBINARY
TINYINT	SQL_TINYINT
UNSIGNED BIGINT2	SQL_BIGINT
UNSIGNED INT2	SQL_INTEGER
UNSIGNED SMALLINT ⁴¹	SQL_SMALLINT
VARBINARY	SQL_VARBINARY
VARCHAR	SQL_VARCHAR

Note: FOR USERS OF SYBASE 12.5 and higher: The Sybase Wire Protocol driver supports extended new limits (XNL) for character and binary columns—columns with lengths greater than 255.

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

Unicode Support

When connected to a Unicode database, the Sybase Wire Protocol driver supports Unicode data types listed in the following table, in addition to standard ODBC data types listed in [Data Types](#) on page 609.

Sybase Data Type	Mapped to . . .
CHAR ⁴³	SQL_WCHAR
SYSNAME ⁴³	SQL_VARCHAR
TEXT ⁴³	SQL_WLONGVARCHAR
UNICHAR ⁴⁴	SQL_WCHAR
UNITEXT ⁴⁵	SQL_WLONGVARCHAR
UNIVARCHAR ⁴⁴	SQL_WVARCHAR
VARCHAR ⁴³	SQL_WVARCHAR

For data types that require the UTF-8 character set, set the Charset connection string attribute. See [Charset](#) on page 567 for information about using this connection string attribute.

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Advanced Features

The driver supports the following advanced features:

- Failover
- Security
- Connection Pooling
- DataDirect Bulk Load

Failover

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

⁴³ This data type is available only if the data source is configured to use the UTF-8 character set.

⁴⁴ On Sybase 12.5 servers, this data type is available only if the data source is configured to use the UTF-8 character set. On Sybase 12.5.1 and higher servers, this data type is always available, even if the data source is not configured to use the UTF-8 character set.

⁴⁵ This data type is available on Sybase 15 and higher servers only.

Security

The driver supports authentication and encryption. Security connection options are located on the [Security tab](#) of the driver Setup dialog box. See [Using Security](#) on page 89 for a general description of security and its implementation. The following security information is specific to the Sybase Wire Protocol Driver.

Authentication

If you are using Kerberos, verify that your environment meets the requirements listed in the following table before you configure the driver for Kerberos authentication.

Table 39: Kerberos Authentication Requirements for the Sybase Wire Protocol Driver

Component	Requirements
Database server	The database server must be administered by the same domain controller that administers the client and must be running Sybase 12.0 or higher. In addition, the Sybase Security and directory services package, ASE_SECDIR, is required.
Kerberos server	The Kerberos server is the machine where the user IDs for authentication are administered. The Kerberos server is also the location of the Kerberos KDC. Network authentication must be provided by one of the following methods: <ul style="list-style-type: none"> • Windows Active Directory • MIT Kerberos 1.4.2 or higher
Client	The client must be administered by the same domain controller that administers the database server.

Connection Pooling

The driver supports connection pooling and its related connection options. Connection pooling connection options are located on the [Pooling tab](#) of the driver Setup dialog box. See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling and its implementation.

DataDirect Bulk Load

The driver supports DataDirect bulk load and its related connection options. Bulk load connection options are located on the [Bulk tab](#) of the driver Setup dialog box. See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect bulk load and its implementation.

For Sybase, some additional database configuration is required when the destination table for a bulk load operation does not have an index defined. If you are using a destination table that does not have an index defined, you can ask the database operator to execute the following commands:

```
use master
sp_dboption test, "select into/bulkcopy/pllsort", true
```

This option is required to perform operations that do not keep a complete record of the transaction in the log. For more information, refer to the Sybase documentation.

Alternatively, you can define an index on the destination table.

Failure to properly configure the database results in errors such as the following:

```
"You cannot run the non-logged version of bulk copy in this database. Please check with the DBO."
```

Bulk Copy Operations and Transactions

Sybase does not support a bulk insert within a transaction, and returns an error if a bulk copy operation is attempted in the scope of an existing transaction.

The Sybase server treats each batch of the bulk copy operation as a single transaction. If any rows in the batch are rejected, the entire transaction is rolled back.

Limitations

- The Sybase server ensures the accuracy of Real data type values only up to 6 digits.
- For the Money data type, if an application submits a value with a scale larger than 4, the driver changes the scale to 4 by truncating the value. Ideally, it should set the scale to 4 and round off the value. For example, if the value is 100.141592, the driver truncates it to 100.1415. Instead, it should round it off to 100.1416.
- The driver does not support inserting data containing LOB columns. In such cases, the driver throws a warning and falls back to the native protocol to continue executing the inserts.
- When executing an insert statement, the operation will fail with an error message if non-identity columns are omitted from a statement.

Performance Considerations

Sybase defines two bulk copy modes, described in the following table. Sybase automatically selects the appropriate mode at run time. For more information, refer to your Sybase documentation.

Table 40: Summary of Fast and Slow Bulk Copy Mode Characteristics

Characteristic	Fast Bulk Copy Mode	Slow Bulk Copy Mode
Destination Table Characteristics	No indexes or triggers on destination table	One or more indexes or triggers
Database Configuration Required	The into/bulkcopy/pllsort dboption must be set to true.	None
Logging Performed	Page allocations are logged, but row inserts are not	Row inserts are logged
Transaction Log Handling	You must dump the database before backing up (dumping) the transaction log.	The transaction log can become very large. After the bulk copy completes, back up your database with dump database, then truncate the log with dump transaction.

Unexpected Characters

Users are sometimes surprised when they insert a character into a database, only to have a different character displayed when they fetch it from the database. There are many reasons this can happen, but it most often involves code page issues, not driver errors.

Client and server machines in a database system each use code pages, which can be identified by a name or a number, such as Shift_JIS (Japanese) or cp1252 (Windows English). A code page is a mapping that associates a sequence of bits, called a code point, with a specific character. Code pages include the characters and symbols of one or more languages. Regardless of geographical location, a machine can be configured to use a specific code page. Most of the time, a client and database server would use similar, if not identical, code pages. For example, a client and server might use two different Japanese code pages, such as Shift_JIS and EUC_JP, but they would still share many Japanese characters in common. These characters might, however, be represented by different code points in each code page. This introduces the need to convert between code pages to maintain data integrity. In some cases, no one-to-one character correspondence exists between the two code points. This causes a substitution character to be used, which can result in displaying an unexpected character on a fetch.

When the driver on the client machine opens a connection with the database server, the driver determines the code pages being used on the client and the server. This is determined from the Active Code Page on a Windows-based machine. If the client machine is UNIX-based, the driver checks the [IANAAppCodePage](#) attribute (see [IANAAppCodePage](#) on page 586). If it does not find a specific setting for IACP, it defaults to a value of ISO_8859_1.

If the client and server code pages are compatible, the driver transmits data in the code page of the server. Even though the pages are compatible, a one-to-one correspondence for every character may not exist. If the client and server code pages are completely dissimilar, for example, Russian and Japanese, then many substitutions occur because very few, if any, of the characters are mapped between the two code pages.

The following is a specific example of an unexpected character:

- The Windows client machine is running code page cp1252.
- The Sybase server is running code page cp850.
- You insert decimal literals for character data. You think you are inserting LATIN SMALL LETTER I WITH ACUTE (i) and BOX DRAWINGS DOUBLE VERTICAL (||) in the database. When you fetch the data, you see INVERTED EXCLAMATION MARK (¡) and MASCULINE ORDINAL INDICATOR (º) displayed on the client instead.

This occurs because the code points do not correspond in the two code pages. An example of syntax you would use to insert the decimal literals is:

```
CREATE table cp850chars(val text )
INSERT INTO cp850chars values( CHAR(161)+CHAR(186))
```

This effectively inserts the hexadecimal bytes for the numbers 161 (0xA1) and 186 (0xBA) into the text column. Each of these hexadecimal bytes is treated as the single byte code point for the character it represents. The problem is that the character representation for these two particular hexadecimal values is different from code page cp850 to code page cp1252. On cp850, these hexadecimal values represent í (0xA1) and || (0xBA), which is what you thought you were inserting by using the previously described syntax. When you fetch these hexadecimal values, however, the characters displayed on your client machine are ¡ (0xA1) and º (0xBA), because that is what the hexadecimal values represent in code page cp1252. This is not a matter of data corruption or substitution; these hexadecimal values simply represent different values in the two different code pages.

This is not a driver error. It occurs because the code points map differently and because some characters do not exist in a code page. The best way to avoid these problems is to use the same code page on both the client and server machines.

MTS Support



On Windows, the driver can take advantage of Microsoft Transaction Server (MTS) capabilities, specifically, the Distributed Transaction Coordinator (DTC) using the XA Protocol. For a general discussion of MTS and DTC, refer to the help file of the Microsoft Transaction Server SDK.

Note: The DataDirect Connect for ODBC 32-bit drivers can operate in a 64-bit Windows environment; however, they do not support DTC in this environment. Only the DataDirect Connect64 for ODBC 64-bit drivers support DTC in a 64-bit Windows environment.

To enable DTC support, you must be accessing Sybase Adaptive Server Enterprise 12.0 or higher. You can choose either Native OLE and XA protocol distributed transactions. See the Distributed Transaction Model option documented in [Configuring and Connecting to Data Sources](#) on page 543 for details.

To enable distributed transaction in the Sybase server:

1. Assign the `dtm_tm_role` to each user who will participate in distributed transactions (who will log in to Adaptive Server). You can do this using the `sp_role` command. For example:

```
sp_role "grant", dtm_tm_role, user_name
```

In the open string for resource managers, the specified username must have the `dtm_tm_role`.

2. Specify a default database other than the master for each user. Sybase cannot start distributed transactions in a master database.

NULL Values

When the Sybase Wire Protocol driver establishes a connection, the driver sets the Sybase database option `ansinull` to `on`. Setting `ansinull` to `on` ensures that the driver is compliant with the ANSI SQL standard, which makes developing cross-database applications easier.

By default, Sybase does not evaluate NULL values in SQL equality (`=`), inequity (`<>`), or aggregate function comparisons in an ANSI SQL-compliant manner. For example, the ANSI SQL specification defines that `col1=NULL` always evaluates to `false`:

```
SELECT * FROM table WHERE col1 = NULL
```

Using the default database setting (`ansinull=off`), the same comparison evaluates to `true` instead of `false`.

Setting `ansinull` to `on` changes the default database behavior so that SQL statements must use `IS NULL` instead of `=NULL`. For example, using the Sybase Wire Protocol driver, if the value of `col1` in the following statement is `NULL`, the comparison evaluates to `true`:

```
SELECT * FROM table WHERE col1 IS NULL
```

In your application, you can restore the default Sybase behavior for a connection in the following ways:

- Use the Initialization String option to specify the SQL command `set ANSINULL off`. For example, the following connection string ensures that the handling of NULL values is restored to the Sybase default for the current connection:

```
DSN=SYB TABLES;DB=PAYROLL;IS=set ANSINULL off
```

- Explicitly execute the following statement after the connection is established:

```
SET ANSINULL OFF
```

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

The Sybase database system supports isolation levels 0 (read uncommitted), 1 (read committed, the default), 2 (repeatable read), and 3 (serializable). It supports page-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Grammar Support

The driver supports the minimum SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions. In addition, the following functions are supported:

- SQLColumnPrivileges
- SQLDescribeParam (if EnableDescribeParam=1)
- SQLForeignKeys
- SQLPrimaryKeys
- SQLProcedureColumns
- SQLProcedures
- SQLTablePrivileges

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The Sybase database system supports multiple connections and multiple statements per connection. If the Select Method option on the Performance tab or the connection string attribute SelectMethod is set to 1 (Direct), Sybase data sources are limited to one active statement in manual commit mode.

Using Arrays of Parameters

When designing an application, using parameter arrays for bulk inserts or updates, for example, can improve performance.

Refer to "Designing ODBC applications for performance optimization" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Because Sybase databases do not support parameter arrays natively, the Sybase Wire Protocol driver emulates them by sending T-SQL batches of Insert or Update statements to the database, which will improve performance.

The Oracle Driver

The DataDirect Connect for ODBC and DataDirect Connect64 for ODBC Oracle driver (the Oracle driver) each support Oracle database servers when using the appropriate client software.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The Oracle driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect product for the file name of the Oracle driver.

Note: The Oracle driver requires Oracle client software. Progress DataDirect also provides an Oracle driver that does not require any client software to access Oracle databases.

See [The Oracle Wire Protocol Driver](#) on page 283 for details.

Driver Requirements

This section provides the system requirements for using the Oracle driver on Windows, UNIX, and Linux.

Important: You must have all components of the Oracle client software installed; otherwise, the driver will not operate properly. You must have the appropriate DLLs or shared libraries and objects on your path.

Although an earlier version of a client can access a later version of a database, for example, client 9i to server 10g, to ensure that you have access to all of the features of a particular database, you should use the client that matches the database version, for example, client 10g to server 10g.

Note: The Oracle driver supports Oracle 10g clients; however, the clients are not available for all operating systems supported by the driver. Consult the Oracle Web site for current client availability.

Windows



For 32-bit drivers, Oracle Net8 Client 9.2 or higher is required.

For 64-bit drivers, Oracle client software 10.1 or higher is required on x64.

UNIX and Linux

UNIX[®] For 32-bit drivers, Oracle Net8 Client 9.2 or higher is required.

For 64-bit drivers, Oracle client software 9i R2 or higher is required on Linux for Itanium II and UNIX. Oracle client software 10.1 or higher is required for Linux on x64.

Before you can use the Oracle driver, you must have a supported Oracle client installed on your workstation in the \$ORACLE_HOME source tree. ORACLE_HOME is an environment variable created by the Oracle installation process that identifies the location of your Oracle client components.

Set the environment variable ORACLE_HOME to the directory where you installed the Oracle client. For example, for C-shell users, the following syntax is valid:

```
setenv ORACLE_HOME /databases/oracle
```

For Bourne- or Korn-shell users, the following syntax is valid:

```
ORACLE_HOME=/databases/oracle;export ORACLE_HOME
```

32-bit drivers—Building the Required Oracle Net8 Shared Library on HP-UX 11

You must build a replacement shared library for Oracle Net8 Client 9.2 on HP-UX 11. This shared library, libclntsh.sl, contains your unique Oracle Net8 configuration, which is used by the Oracle driver to access local and remote Oracle databases.

The shared library libclntsh.sl is built by the Oracle script genclntsh. The genclntsh script provided by Oracle causes errors resulting from undefined symbols. Run the genclntsh92 script provided by Progress DataDirect to build a replacement libclntsh.sl. This script, in the src/oracle directory, places the new libclntsh.sl in ../../lib, which is your \$ODBC_HOME/lib directory; it does not overwrite the original libclntsh.sl in the \$ORACLE_HOME/lib directory.

Before you build the Oracle Net8 shared library, install Oracle and set the environment variable ORACLE_HOME to the directory where you installed Oracle.

For Oracle Net8 Client 9.2 on HP-UX 11, the following commands build the Oracle Net8 shared library:

```
cd ${ODBC_HOME}/src/oracle
genclntsh92
```

warning: The \$ODBC_HOME/lib directory, containing the correct libclntsh library, *must* be on the SHLIB_PATH *before* \$ORACLE_HOME/lib. Otherwise, the original Oracle library will be loaded, resulting in the unresolved symbol error.

Connecting to Oracle 9.2 from HP-UX

To connect to Oracle 9.2 from HP-UX, you must have the HP patch PHSS_22514 installed on the operating system, and you must set the LD_PRELOAD system variable to the absolute path of the libjava.sl library.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 626 and [Connection Option Descriptions](#) on page 627 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

UNIX® On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions](#) on page 627 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Oracle Client)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX® On UNIX and Linux, data sources are stored in the odbc.ini file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure an Oracle data source:

1. Start the ODBC Administrator:

-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
-

UNIX®

On Linux, change to the `install_dir/tools` directory and, at a command prompt, enter:


```
odbcadmin
```

where *install_dir* is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

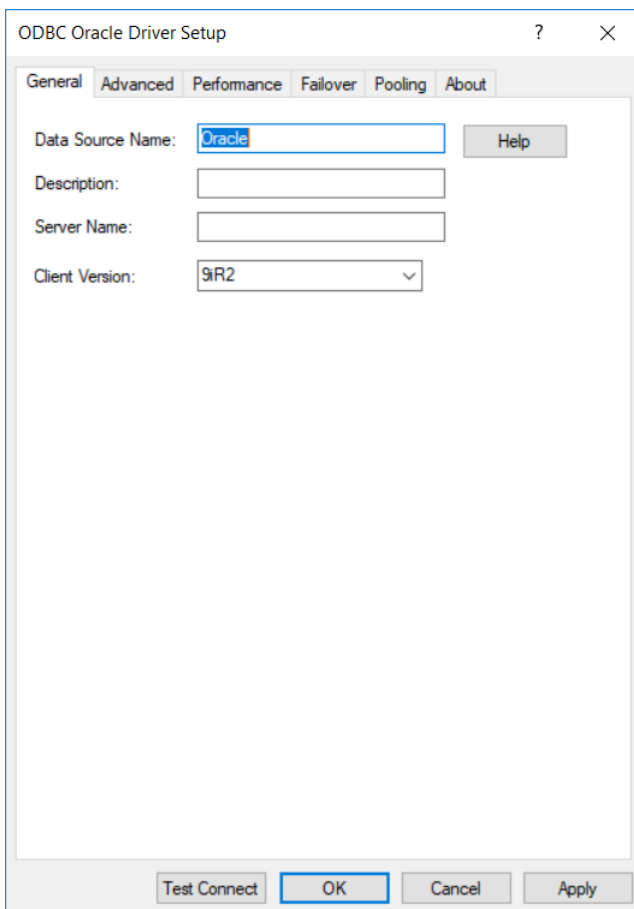
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 63: General tab



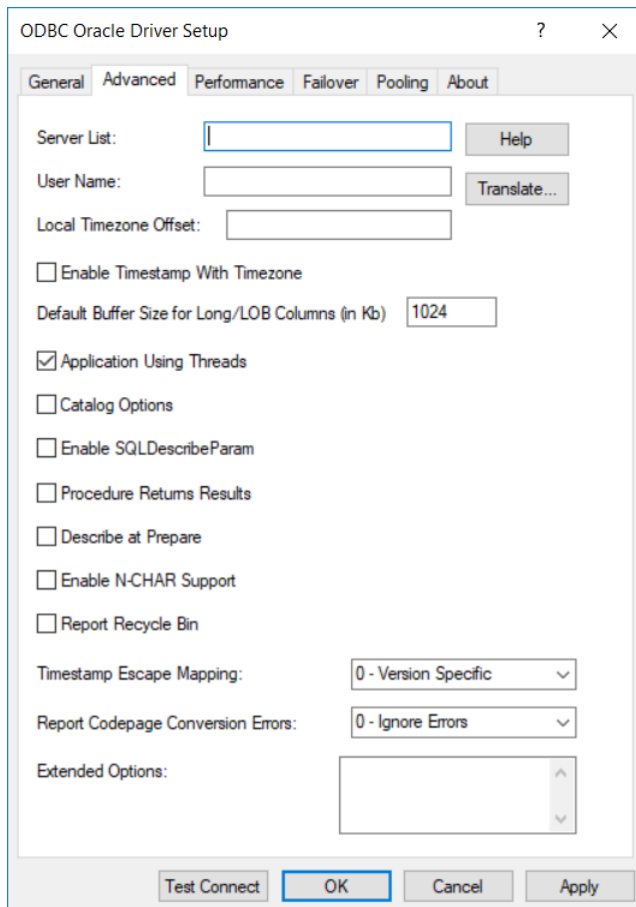
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 636	None
Description on page 637	None
Server Name on page 650	None
Client Version on page 632	9iR2

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 64: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Server List on page 649	None
User Name on page 651	None
Local Timezone Offset on page 643	None
Enable Timestamp with Timezone on page 640	Disabled
Default Buffer Size for Long/LOB Columns (in Kb) on page 636	1024
Application Using Threads on page 630	Enabled
Catalog Options on page 632	Disabled
Enable SQLDescribeParam on page 639	Disabled
Procedure Returns Results on page 647	Disabled
Describe At Prepare on page 637	Disabled
Enable N-CHAR Support on page 638	Disabled
Report Recycle Bin on page 648	Disabled
Timestamp Escape Mapping on page 650	0 - Version Specific
Report Codepage Conversion Errors on page 648	0 - Ignore Errors
IANAAppCodePage on page 641 UNIX ONLY	4 (ISO 8559-1 Latin-1)



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.

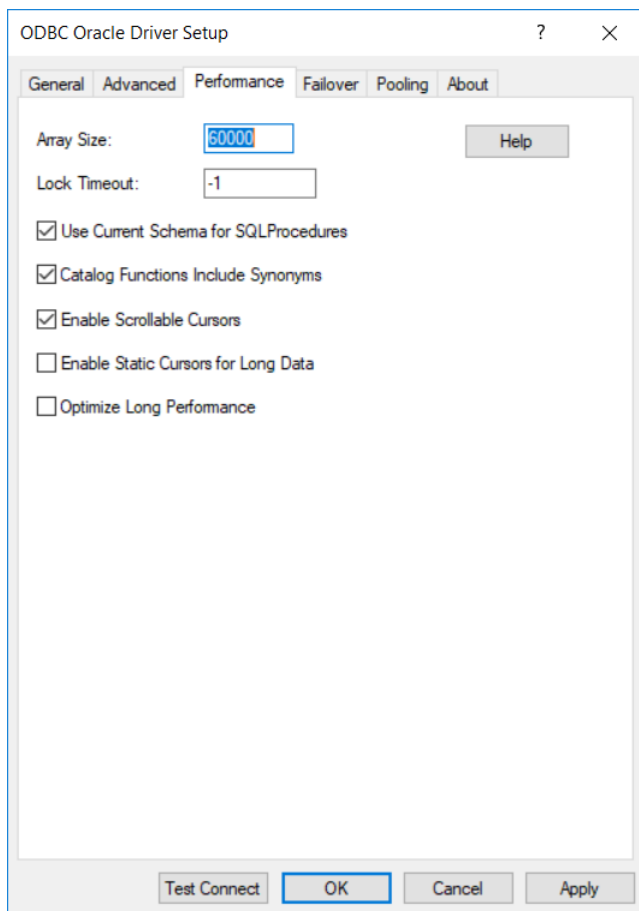


Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- Optionally, click the **Performance** tab to specify performance data source settings.

Figure 65: Performance tab

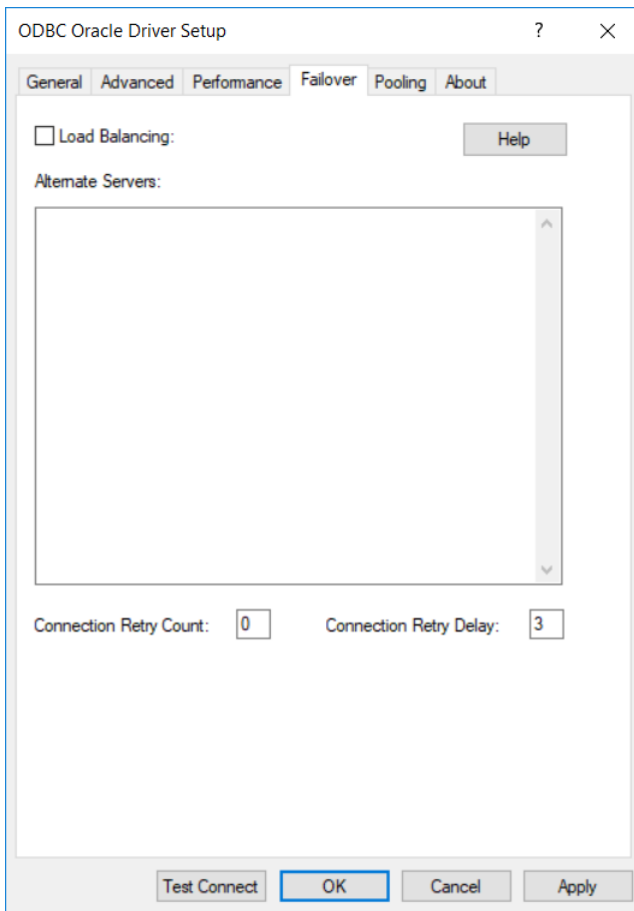


On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Performance	Default
Array Size on page 630	60000
Lock Timeout on page 644	-1
Use Current Schema for SQLProcedures on page 651	Enabled
Catalog Functions Includes Synonyms on page 631	Enabled
Enable Scrollable Cursors on page 639	Enabled
Enable Static Cursors For LongData on page 640	Disabled
Optimize Long Performance on page 646	Disabled

6. Optionally, click the **Failover** tab to specify failover data source settings.

Figure 66: Failover tab



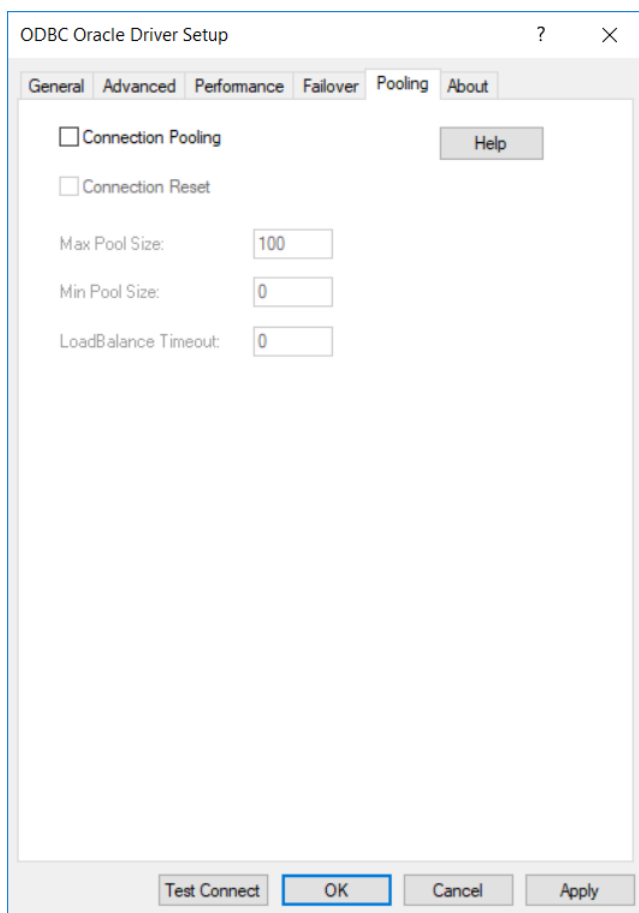
See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 643	Disabled
Alternate Servers on page 629	None
Connection Pooling on page 633	0
Connection Retry Delay on page 635	3

7. Optionally, click the **Pooling** tab to specify connection pooling data source settings.

Figure 67: Pooling tab



See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 633	Disabled
Connection Reset on page 634	Disabled
Max Pool Size on page 645	100
Min Pool Size on page 645	0
Load Balance Timeout on page 642	0

8. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Oracle Client\)](#) on page 627 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.

- If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

9. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={ }driver_name[ ] [ ;attribute=value[ ;attribute=value]... ]
```

The following table lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Oracle Wire Protocol is:

```
DSN=Accounting;SRVR=QESRVR;UID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Oracle.dsn;SRVR=QESRVR;UID=JOHN;PWD=XYZZY
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Oracle};SRVR=QESRVR;CV=10GR1;UID=JOHN;PWD=XYZZY
```

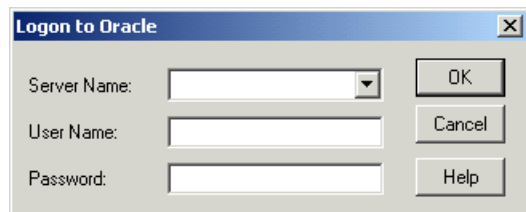
If the server name contains a semicolon, enclose it in quotation marks:

```
DSN=Accounting;SRVR="QE;SRVR";UID=JOHN;PWD=XYZZY
```

Using a Logon Dialog Box (Oracle Client)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Figure 68: Logon to Oracle dialog box



In this dialog box, provide the following information:

1. In the Server Name field, type the client connection string of the computer containing the Oracle database tables you want to access. Or, select the string from the Server Name drop-down list, which displays the names you specified in the ODBC Oracle driver Setup dialog box.

For local servers, use the SQL*Net connection string. If the SQL*Net connection string contains semicolons, enclose it in quotation marks. Refer to your SQL*Net documentation for more information.

For remote servers, the Oracle TNS Client connection string is the alias name of the Oracle Listener on your network.

2. If required, type your Oracle user name.
3. If required, type your Oracle password.
4. Click **OK** to log on to the Oracle database installed on the server you specified and to update the values in the Registry.

Note: You can also use OS Authentication to connect to an Oracle database. See [OS Authentication](#) on page 658 for details.

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Oracle driver.

Table 41: Oracle Attribute Names

Attribute (Short Name)	Default
AlternateServers (ASRV)	None

Attribute (Short Name)	Default
ApplicationUsingThreads (AUT)	1 (Enabled)
ArraySize (AS)	60000
CatalogIncludesSynonyms (CIS)	1 (Enabled)
CatalogOptions (CO)	0 (Disabled)
ClientVersion (CV)	9iR2
ConnectionReset (CR)	0 (Disabled)
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
DataSourceName (DSN)	None
DefaultLongDataBuffLen (DLDBL)	1024
DescribeAtPrepare (DAP)	0 (Disabled)
Description (n/a)	Empty
EnableDescribeParam (EDP)	0 (Disabled)
EnableNcharSupport (ENS)	0 (Disabled)
EnableScrollableCursors (ESC)	1 (Enabled)
EnableStaticCursorsForLongData (ESCLD)	0 (Disabled)
EnableTimestampwithTimezone (ETWT)	0 (Disabled)
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
LoadBalanceTimeout (LBT)	0 (Disabled)
LoadBalancing (LB)	0 (Disabled)
LocalTimezoneOffset (LTZO)	None
LockTimeout (LTO)	-1
LogonID (UID)	None
MaxPoolSize (MXPS)	100
MinPoolSize (MNPS)	0
OptimizeLongPerformance (OLP)	0 (Disabled)

Attribute (Short Name)	Default
Password (PWD)	Empty
Pooling (POOL)	0 (Disabled)
ProcedureRetResults (PRR)	0 (Disabled)
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
ReportRecycleBin (RRB)	0 (Disabled)
ServerList	None
ServerName (SRVR)	None
TimestampEscapeMapping (TEM)	0 (Version Specific)
UseCurrentSchema (UCS)	1 (Enabled)

Alternate Servers

Attribute

AlternateServers (ASRV)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
(ServerName=servervalue[, . . .])
```

You must specify the server name of each alternate server.

Example

The following Alternate Servers value defines two alternate database servers for connection failover:

```
(ServerName=AcctBackup1, ServerName=AcctBackup2)
```

Default

None

GUI tab

[Failover tab](#)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Array Size

Attribute

ArraySize (AS)

Purpose

The number of bytes the driver can fetch in a single network round trip. Larger values increase throughput by reducing the number of times the driver fetches data across the network. Smaller values increase response time, as there is less of a delay waiting for the server to transmit data.

Valid Values

An integer from 1 to 4,294,967,296 (4 GB)

The value 1 does not define the number of bytes but, instead, causes the driver to allocate space for exactly one row of data.

Notes

- This connection option can affect performance.

Default

60000

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Catalog Functions Includes Synonyms

Attribute

CatalogIncludesSynonyms (CIS)

Purpose

Determines whether synonyms are included in calls to SQLProcedures, SQLStatistics, and SQLProcedureColumns.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), synonyms are included in calls to SQLProcedures, SQLStatistics, and SQLProcedureColumns.

If set to 0 (Disabled), synonyms are excluded (a non-standard behavior) and performance is thereby improved.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Catalog Options

Attribute

CatalogOptions (CO)

Purpose

Determines whether SQL_NULL_DATA is returned for the result columns REMARKS and COLUMN_DEF.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the result column REMARKS (for the catalog functions SQLTables and SQLColumns) and the result column COLUMN_DEF (for the catalog function SQLColumns) return actual values. Enabling this option reduces the performance of your catalog (SQLColumns and SQLTables) queries.

If set to 0 (Disabled), SQL_NULL_DATA is returned for the result columns REMARKS and COLUMN_DEF.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Client Version

Attribute

ClientVersion (CV)

Purpose

A value to specify the Oracle client software version. The driver assumes that it is using the version of Oracle client software specified by this option to connect to an Oracle server.

Valid Values

8i | 9iR1 | 9iR2 | 10gR1

Behavior

When set to 10gR1 and later, the driver binds all non-integer numerics as BINARY FLOAT and BINARY DOUBLE. When set to any Oracle version previous to Oracle10g R1, the driver binds non-integer numerics as if connected to an Oracle 9i R2 or earlier version of the server (regardless of the actual version of the server to which it is connected). When connecting to an Oracle 10g server with a pre-10g client, this attribute must be set to the same version as the actual Oracle client software in use; otherwise, numeric parameter bindings may fail. Versions of the Oracle client software prior to 10g R1 do not fully support the new features of the Oracle 10g database server.

Default

9iR2

GUI Tab

[General tab](#)

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- This connection option can affect performance.
- The application must be thread-enabled to use connection pooling.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Connection Reset

Attribute

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Connection Retry Count

Attribute

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x , the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay (CRD)

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to x , the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Default Buffer Size for Long/LOB Columns (in Kb)

Attribute

DefaultLongDataBuffLen (DLDBL)

Purpose

The maximum length of data (in KB) the driver can fetch from long columns in a single round trip and the maximum length of data that the driver can send using the SQL_DATA_AT_EXEC parameter.

Valid Values

An integer in multiples of 1024

The value must be in multiples of 1024 (for example, 1024, 2048). You need to increase the default value if the total size of any Long data exceeds 1 MB. This value is multiplied by 1024 to determine the total maximum length of fetched data. For example, if you enter a value of 2048, the maximum length of data would be 1024 x 2048, or 2097152 (2 MB).

Notes

- If this option is enabled, the Optimize Long Performance option is ignored.
- This connection option can affect performance.

Default

1024

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Describe At Prepare

Attribute

DescribeAtPrepare (DAP)

Purpose

Determines whether the driver describes the SQL statement at prepare time.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver describes the SQL statement at prepare time.

If set to 0 (Disabled), the driver does not describe the SQL statement at prepare time.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable N-CHAR Support

Attribute

EnableNcharSupport (ENS)

Purpose

Determines whether the driver provides support for the N-types NCHAR, NVARCHAR2, and NCLOB. These types are described as SQL_WCHAR, SQL_WVARCHAR, and SQL_WLONGVARCHAR, and are returned as supported by SQLGetTypeInfo. In addition, the "normal" char types (char, varchar2, long, clob) are described as SQL_CHAR, SQL_VARCHAR, and SQL_LONGVARCHAR regardless of the character set on the Oracle server.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver provides support for the N-types NCHAR, NVARCHAR2, and NCLOB.

If set to 0 (Disabled), the driver does not provide support for the N-types NCHAR, NVARCHAR2, and NCLOB.

Notes

- Valid only on Oracle 9i and higher.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See Also

See [Unicode Support](#) on page 656 for details.

Enable Scrollable Cursors

Attribute

EnableScrollableCursors (ESC)

Purpose

Determines whether scrollable cursors, both Keyset and Static, are enabled for the data source.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), scrollable cursors are enabled for the data source.

If set to 0 (Disabled), scrollable cursors are not enabled.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Enable SQLDescribeParam

Attribute

EnableDescribeParam (EDP)

Purpose

Determines whether the driver supports the SQLDescribeParam function, which allows an application to describe parameters in SQL statements and in stored procedure calls.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver supports SQLDescribeParam. If using Microsoft Remote Data Objects (RDO) to access data, you must use this value.

If set to 0 (Disabled), the driver does not support `SQLDescribeParam` and returns the error: `unimplemented function`.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Enable Static Cursors For LongData

Attribute

`EnableStaticCursorsForLongData` (ESCLD)

Purpose

Determines whether the driver supports Long columns when using a static cursor. Enabling this option causes a performance penalty at the time of execution when reading Long data.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver supports Long columns when using a static cursor.

If set to 0 (Disabled), the driver does not support Long columns when using a static cursor.

Notes

- You must enable this option if you want to persist a result set that contains Long data into an XML data file.
- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Enable Timestamp with Timezone

Attribute

`EnableTimestampwithTimezone` (ETWT)

Purpose

Determines whether the driver exposes timestamps with timezones to the application.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver exposes timestamps with timezones to the application. The driver issues an ALTER SESSION at connection time to modify NLS_TIMESTAMP_TZ_FORMAT. NLS_TIMESTAMP_TZ_FORMAT is changed to the ODBC definition of a timestamp literal with the addition of the timezone literal: 'YYYY-MM-DD HH24:MI:SSXFF TZR'.

If set to 0 (Disabled), timestamps with timezones are not exposed to the application.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Load Balance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

The number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Valid Values

0 | x

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to x , inactive connections are closed after the specified number of seconds passes.

Notes

- The Min Pool Size option may cause some connections to ignore this value.
- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Load Balancing

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Local Timezone Offset

Attribute

LocalTimezoneOffset (LTZO)

Purpose

A value to alter local time zone information. The default is "" (empty string), which means that the driver determines local time zone information from the operating system. If it is not available from the operating system, the driver defaults to using the setting on the Oracle server.

Valid Values

Valid values are specified as offsets from GMT as follows: $(-)HH:MM$. For example, $-08:00$ equals GMT minus 8 hours.

The driver uses the value of this option to issue an ALTER SESSION for local time zone at connection time.

Default

"" (Empty String)

GUI Tab

[Advanced tab](#)

Lock Timeout

Attribute

LockTimeout (LTO)

Purpose

Specifies the amount of time, in seconds, the Oracle server waits for a lock to be released before generating an error when processing a Select...For Update statement on an Oracle 9i or higher server.

Valid Values

-1 | 0 | x

where:

x

is an integer that specifies a number of seconds.

Behavior

If set to -1, the server waits indefinitely for the lock to be released.

If set to 0, the server generates an error immediately and does not wait for the lock to time out.

If set to x , the server waits for the specified number of seconds for the lock to be released.

Notes

- If you are connected to an Oracle 8i server, any value greater than 0 is equivalent to the value -1.
- This connection option can affect performance.

Default

-1

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Max Pool Size

Attribute

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

For example, if set to 20, the maximum number of connections allowed in the pool is 20.

Notes

- This connection option can affect performance.

Default

100

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Min Pool Size

Attribute

MinPoolSize (MNPS)

Purpose

The minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Valid Values

0 | x

where:

x is an integer from 1 to 65535.

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

Notes

- This connection option can affect performance.

Example

If set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Default

0

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Optimize Long Performance

Attribute

OptimizeLongPerformance (OLP)

Purpose

Allows the driver to fetch Long data directly into the application's buffers rather than allocating buffers and making a copy. This option decreases fetch times on Long data; however, it can cause the application to be limited to one active statement per connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver fetches Long data directly into the application's buffers rather than allocating buffers and making a copy.

If set to 0 (Disabled), the driver does not fetch Long data directly into the application's buffers.

Notes

- If this option is enabled, the Default Buffer Size for Long/LOB Columns option is ignored.
- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Procedure Returns Results

Attribute

ProcedureRetResults (PRR)

Purpose

Determines whether the driver returns result sets from stored procedures/functions.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns result sets from stored procedures/functions. When set to 1 and you execute a stored procedure that does not return result sets, you will incur a small performance penalty.

If set to 0 (Disabled), the driver does not return result sets from stored procedures.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See Also

See [MTS Support](#) on page 658 for details.

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where `x` is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Advanced tab](#)

Report Recycle Bin

Attribute

ReportRecycleBin (RRB)

Purpose

Determines whether support is provided for reporting objects that are in the Oracle Recycle Bin.

On Oracle 10g R1 and higher, when a table is dropped, it is not actually removed from the database, but placed in the recycle bin instead.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), support is provided for reporting objects that are in the Oracle Recycle Bin.

If set to 0 (Disabled), the driver does not return tables contained in the recycle bin in the result sets returned from SQLTables and SQLColumns. Functionally, this means that the driver filters out any results whose Table name begins with BIN\$.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Server List

Attribute

ServerList

Purpose

A list of client connection strings that appear in the logon dialog box. This option applies to GUIs only and is not a runtime connection string attribute.

Valid Values

string

where:

string

is a list of valid client connection strings. Separate the strings with commas. If the client connection string contains a comma, enclose it in quotation marks, for example, "Serv,1", "Serv,2", "Serv,3".

Default

None

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 652 for details about the logon dialog box.

Server Name

Attribute

ServerName (SRVR)

Purpose

The client connection string of the computer containing the Oracle database tables you want to access.

Valid Values

string

where:

string

is a valid client connection string.

For local servers, use the SQL*Net connection string. If the SQL*Net connection string contains semicolons, enclose it in quotation marks. Refer to your SQL*Net documentation for more information.

For remote servers, the Oracle TNS Client connection string is the alias name of the Oracle Listener on your network.

Default

None

GUI Tab

[General tab](#)

Timestamp Escape Mapping

Attribute

TimestampEscapeMapping (TEM)

Purpose

Determines how the driver maps Date, Time, and Timestamp literals.

Valid Values

0 | 1

Behavior

If set to 0 (Oracle Version Specific), the driver determines whether to use the TO_DATE or TO_TIMESTAMP function based on the version of the Oracle server to which it is connected. If the driver is connected to an 8.x server, it maps the Date, Time, and Timestamp literals to the TO_DATE function. If the driver is connected to a 9.x or higher server, it maps these escapes to the TO_TIMESTAMP function.

If set to 1 (Oracle 8x Compatible), the driver always uses the Oracle 8.x TO_DATE function as if connected to an Oracle 8.x server.

Default

0 (Oracle Version Specific)

GUI Tab

[Advanced tab](#)

Use Current Schema for SQLProcedures**Attribute**

UseCurrentSchema (UCS)

Purpose

Determines whether the driver returns only procedures owned by the current user when executing SQLProcedures.

Valid Values

0 | 1

Behavior

When set to 1 (Enabled), the call for SQLProcedures is optimized, but only procedures owned by the user are returned.

When set to 0 (Disabled), the driver does not specify only the current user.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 652 for details.

User Name**Attribute**

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

You can also use OS Authentication to connect to your Oracle database.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Advanced tab](#)

See Also

See [OS Authentication](#) on page 658 for details.

Performance Considerations

The following connection options can enhance driver performance.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Array Size (ArraySize): If this connection string attribute is set appropriately, the driver can improve performance of your application by reducing the number of round trips on the network. For example, if your application normally retrieves 200 rows, it is more efficient for the driver to retrieve 200 rows at one time over the network than to retrieve 50 rows at a time during four round trips over the network.

Catalog Functions Include Synonyms (CatalogIncludesSynonyms): Standard ODBC behavior is to include synonyms in the result set of calls to the following catalog functions: SQLProcedures, SQLStatistics and SQLProcedureColumns. Retrieving this synonym information degrades performance. If your ODBC application does not need to return synonyms when using these catalog functions, the driver can improve performance if the CatalogIncludesSynonyms attribute is disabled (set to 0).

Catalog Options (CatalogOptions): If your application does not need to access the comments/remarks for database tables, performance of your application can be improved. In this case, the CatalogOptions attribute should be disabled (set to 0) because retrieving comments/remarks degrades performance. If this attribute is enabled (set to 1), result column REMARKS (for the catalog functions SQLTables and SQLColumns) and the result column COLUMN_DEF (for the catalog function SQLColumns) return actual values.

Default Buffer Size for Long/LOB Columns (DefaultLongDataBuffLen): To improve performance when your application fetches images, pictures, or long text or binary data, a buffer size can be set to accommodate the maximum size of the data. The buffer size should only be large enough to accommodate the maximum amount of data retrieved; otherwise, performance is reduced by transferring large amounts of data into an oversized buffer. If your application retrieves more than 1 MB of data, the buffer size should be increased accordingly.

Describe At Prepare (DescribeAtPrepare): When enabled, this option requires extra network traffic. If your application does not require result set information at prepare time (for instance, you request information about the result set using SQLColAttribute(s), SQLDescribeCol, SQLNumResultCols, and so forth, before calling SQLExecute on a prepared statement), you can increase performance by disabling this option.

Enable Scrollable Cursors (EnableScrollableCursors) and Enable Static Cursors for Long Data (EnableStaticCursorsForLongData): When your application uses Static or Keyset (Scrollable) cursors, the EnableScrollableCursors attribute must be enabled (set to 1). Also, if your application retrieves images, pictures, long text or binary data while using Static cursors, the EnableStaticCursorsForLongData attribute must be enabled (set to 1). However, this can degrade performance when retrieving long data with Static cursors as the entire result set is stored on the client. To improve performance, you might consider designing your application to retrieve long data through forward-only cursors.

Lock Timeout (LockTimeOut): Sometimes users attempt to select data that is locked by another user. Oracle provides three options when accessing locked data with SELECT ... FOR UPDATE statements:

- Wait indefinitely for the lock to be released (-1)
- Return an error immediately (0)
- Return an error if the lock has not been released within a specific number of seconds (*n* seconds)

Note: This option is not available with Oracle 8.

Some applications may benefit by not waiting indefinitely and continuing execution; this keeps the application from hanging. The application, however, needs to handle lock timeouts properly with an appropriate timeout value; otherwise, processing time could be wasted handling lock timeouts, and deadlocks could go undetected.

To improve performance, either enter a number of seconds or enter 0 as the value for this option.

Optimize Long Performance (OptimizeLongPerformance): When enabled, this option fetches Long data directly into the application's buffers rather than allocating buffers and making a copy. Also, when enabled, this option decreases fetch times on Long data; however, it can cause the application to be limited to one active statement per connection.

Procedure Returns Results (ProcedureRetResults): The driver can be tuned for improved performance if your application's stored procedures do not return results. In this case, the ProcedureRetResults attribute should be disabled (set to 0).

Use Current Schema for SQLProcedures (UseCurrentSchema): If your application needs to access database objects owned only by the current user, performance of your application can be improved. In this case, the UseCurrentSchema attribute should be enabled (set to 1). When this attribute is enabled, the driver returns only database objects owned by the current user when executing catalog functions. Calls to catalog functions are optimized by grouping queries. Enabling this attribute is equivalent to passing the Logon ID used on the connection as the SchemaName argument to the catalog functions.

Data Types

The following table shows how the Oracle data types are mapped to the standard ODBC data types. [Unicode Support](#) on page 656 lists Oracle to Unicode data type mappings.

Table 42: Oracle Data Types

Oracle	ODBC
BFILE ⁴⁶	SQL_LONGVARBINARY

⁴⁶ Read-Only

Oracle	ODBC
BINARY DOUBLE ⁴⁷	SQL_REAL
BINARY FLOAT ⁴⁷	SQL_DOUBLE
BLOB ⁴⁸	SQL_LONGVARBINARY
CHAR ⁴⁹	SQL_CHAR
CLOB ^{48, 50}	SQL_LONGVARCHAR
DATE	SQL_TYPE_TIMESTAMP
LONG ⁵¹	SQL_LONGVARCHAR
LONG RAW	SQL_LONGVARBINARY
NCHAR ⁵²	SQL_WVARCHAR
NCLOB ⁵²	SQL_WLONGVARCHAR
NVARCHAR2 ⁵²	SQL_WVARCHAR
NUMBER	SQL_DOUBLE
NUMBER (p,s)	SQL_DECIMAL
RAW	SQL_VARBINARY
Oracle	ODBC
TIMESTAMP ⁵³	SQL_TIMESTAMP
TIMESTAMP WITH LOCAL TIMEZONE ⁵³	SQL_TIMESTAMP
TIMESTAMP WITH TIMEZONE ⁵³	SQL_VARCHAR
VARCHAR2 ⁵⁵	SQL_VARCHAR
XMLType ⁵⁶	SQL_LONGVARCHAR

⁴⁷ Supported only on Oracle 10g and higher.

⁴⁸ Valid when connecting to Oracle 8 servers; these data types support output parameters to stored procedures

⁴⁹ If the database character set is set to UTF-8, the Oracle driver maps the CHAR data type to SQL_WCHAR.

⁵⁰ If the database character set is set to UTF-8, the Oracle driver maps the CLOB data type to SQL_WLONGVARCHAR.

⁵¹ If the database character set is set to UTF-8, the Oracle driver maps the LONG data type to SQL_WLONGVARCHAR.

⁵² Supported only when the EnableNcharSupport connection option is enabled.

⁵³ Supported only on Oracle 9i and higher.

⁵⁴ If the database character set is set to UTF-8, the Oracle driver maps the VARCHAR2 data type to SQL_WVARCHAR.

⁵⁵ If the database character set is set to UTF-8, the Oracle driver maps the VARCHAR2 data type to SQL_WVARCHAR.⁵⁴

⁵⁶ XMLType columns with binary or object relational storage are not supported.

The Oracle driver does not support any object types (also known as abstract data types). When the driver encounters an object type during data retrieval, it will return an Unknown Data Type error (SQL State HY000).

See [Retrieving Data Type Information](#) on page 72 for more information about data types.

XMLType

The driver supports tables containing columns whose data type is specified as XMLType, except those with binary or object relational storage. The driver supports tables containing columns whose data type is specified as XMLType.

When inserting or updating XMLType columns, the data to be inserted or updated must be in the form of an XMLType data type. The database provides functions to construct XMLType data. The `xmlData` argument to `xmltype()` may be specified as a string literal.

Examples

If the XMLType column is created with the CLOB storage type, then the driver returns it without use of the special `getClobVal` function, that is, you can use:

```
SELECT XML_col FROM table_name...
```

instead of

```
SELECT XML_col.getClobVal()...
```

The following example illustrates using the CLOB storage type:

```
CREATE TABLE po_xml_tab(
  poid NUMBER(10),
  poDoc XMLTYPE
)
XMLType COLUMN poDoc
STORE AS CLOB (
  TABLESPACE lob_seg_ts
  STORAGE (INITIAL 4096 NEXT 4096)
  CHUNK 4096 NOCACHE LOGGING
);
```

The next example illustrates how to create a table, insert data, and retrieve data when not using the CLOB storage type:

```
CREATE TABLE PURCHASEORDER (PODOCUMENT sys.XMLTYPE);
```

The PURCHASEORDER table contains one column—PODOCUMENT—with a data type of XMLType (`sys.XMLTYPE`). The next step is to insert one purchase order, created by the static function `sys.XMLTYPE.createXML`:

```
INSERT INTO PURCHASEORDER (PODOCUMENT) values (
  sys.XMLTYPE.createXML(
    '<PurchaseOrder>
      <Reference>BLAKE-2001062514034298PDT</Reference>
      <Actions>
        <Action>
          <User>KING</User>
          <Date/>
        </Action>
      </Actions>
      <Reject/>
      <Requester>David E. Blake</Requester>
      <User>BLAKE</User>
    ')
```

```

<CostCenter>S30</CostCenter>
<ShippingInstructions>
  <name>David E. Blake</name>
  <address>400 Oracle Parkway Redwood Shores, CA, 94065 USA</address>
  <telephone>650 999 9999</telephone>
</ShippingInstructions>
<SpecialInstructions>Air Mail</SpecialInstructions>
<LineItems>
  <LineItem ItemNumber="1">
    <Description>The Birth of a Nation</Description>
    <Part Id="EE888" UnitPrice="65.39" Quantity="31"/>
  </LineItem>
</LineItems>
</PurchaseOrder>
');

```

Use the `getClobVal` function to retrieve the data:

```
SELECT p.podocument.getClobVal() FROM PURCHASEORDER p;
```

Unicode Support

The Oracle driver uses the `NLS_LANG` environment variable setting of the Oracle client to determine how to transmit data to the client.

On Windows, UNIX, and Linux, a Unicode setting is determined if the `NLS_LANG` environment variable is set to:

```
LANGUAGE_TERRITORY.CHARSET
```

where *CHARSET* is either UTF8, AL24UTFFSS, or AL32UTF8. For example:

```
AMERICAN_AMERICA.UTF8
```

Alternatively, on Windows, instead of the `NLS_LANG` environment variable, the value of the `HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\oracle_home_key` registry key can be set to:

```
LANGUAGE_TERRITORY.CHARSET
```

where *oracle_home_key* is `HOME0` for Oracle 9i R2 and earlier, and is the Oracle home name used at the time of client installation for Oracle 10g.

If the *CHARSET* is a Unicode setting and a Unicode application is accessing the driver, then no data conversion is necessary. If an ANSI application is accessing the driver, then the driver must convert the data from the application from ANSI to Unicode (UTF-8) for the client.

If the *CHARSET* is ANSI and an ANSI application is accessing the driver, then no data conversion is necessary. If a Unicode application is accessing the driver, then the driver must convert the data from the application from Unicode to ANSI for the client.

If `NLS_LANG` is set to UTF-8, the Oracle driver maps the Oracle data types to Unicode data types as shown in the following table:

Oracle Data Type	Mapped to . . .
CHAR	SQL_WCHAR
CLOB	SQL_WLONGVARCHAR

Oracle Data Type	Mapped to . . .
VARCHAR2	SQL_WVARCHAR
LONG	SQL_WLONGVARCHAR

The driver also continues to map these Oracle data types to the normal character data types. See [Data Types](#) on page 653 for these mappings.

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Advanced Features

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Unexpected Characters

Users are sometimes surprised when they insert a character into a database, only to have a different character displayed when they fetch it from the database. There are many reasons this can happen, but it most often involves code page issues, not driver errors.

Client and server machines in a database system each use code pages, which can be identified by a name or a number, such as Shift_JIS (Japanese) or cp1252 (Windows English). A code page is a mapping that associates a sequence of bits, called a code point, with a specific character. Code pages include the characters and symbols of one or more languages. Regardless of geographical location, a machine can be configured to use a specific code page. Most of the time, a client and database server would use similar, if not identical, code pages. For example, a client and server might use two different Japanese code pages, such as Shift_JIS and EUC_JP, but they would still share many Japanese characters in common. These characters might, however, be represented by different code points in each code page. This introduces the need to convert between code pages to maintain data integrity. In some cases, no one-to-one character correspondence exists between the two code points. This causes a substitution character to be used, which can result in displaying an unexpected character on a fetch.

When the driver on the client machine opens a connection with the database server, the driver determines the code pages being used on the client and the server. This is determined from the Active Code Page on a Windows-based machine. If the client machine is UNIX-based, the driver checks the IANAAppCodePage option. If it does not find a specific setting for IACP, it defaults to a value of ISO_8859_1.

If the client and server code pages are compatible, the driver transmits data in the code page of the server. Even though the pages are compatible, a one-to-one correspondence for every character may not exist. If the client and server code pages are completely dissimilar, for example, Russian and Japanese, then many substitutions occur because very few, if any, of the characters are mapped between the two code pages.

The following is a specific example of an unexpected character:

- The Windows client machine is running code page cp1252.

- The Oracle server is running code page ISO-8859-P1.
- When you insert a Euro character (€) from the Windows client and then fetch it back, an upside down question mark (¿) is displayed on the client instead of the Euro symbol.

This substitution occurs because the Euro character does not exist within the characters defined by the ISO-8859-P1 character set on the Oracle server. The Oracle server records the code point for its substitution character in the table instead of the code point for the Euro. This code point is an upside down question mark in the Windows cp1252 code page.

This is not a driver error. The code page of the Oracle database could not recognize the Euro code point and used its substitution character in the table. The best way to avoid these problems is to use the same code page on both the client and server machines.

You can check the native code point stored in the Oracle database using SQL*Plus with a SQL statement similar to the following:

```
SELECT dump(columnname, 1016) FROM yourtable;
```

Check the returned hexadecimal values to verify whether the data you intended to reside in the table is there. If it appears that Oracle substituted a different code point, then check the Oracle database code page to see if your intended character exists. If your character does not exist in the code page, then no error is involved; Oracle simply does not recognize the original character, and uses its substitution character instead.

MTS Support



On Windows, the driver can take advantage of Microsoft Transaction Server (MTS) capabilities, specifically, the Distributed Transaction Coordinator (DTC) using the XA Protocol. For a general discussion of MTS and DTC, refer to the help file of the Microsoft Transaction Server SDK.

Note: The DataDirect Connect for ODBC 32-bit drivers can operate in a 64-bit Windows environment; however, they do not support DTC in this environment. Only the DataDirect Connect64 for ODBC 64-bit drivers support DTC in a 64-bit Windows environment.

To enable DTC support, you must be accessing Oracle 8.0.5 or higher servers using Oracle Net8 Client 8.1.6 or higher.

OS Authentication

On Windows, UNIX, and Linux, Oracle has a feature called OS Authentication that allows you to connect to an Oracle database via the operating system user name and password. To connect, use a forward slash (/) for the user name and leave the password blank. To configure the Oracle server, refer to the Oracle server documentation. This feature is valid when connecting from a data source, a connection string, or a logon dialog box.

Support for Oracle RAC

Oracle introduced Real Application Clusters (RAC) with Oracle 9i, and RAC is also a key feature of Oracle 10g. Oracle RAC allows a single physical Oracle database to be accessed by concurrent instances of Oracle running across several different CPUs.

An Oracle RAC is composed of a group of independent servers, or nodes, that cooperate as a single system. A cluster architecture such as this provides applications access to more computing power when needed, while allowing computing resources to be used for other applications when database resources are not as heavily required. For example, in the event of a sudden increase in network traffic, an Oracle RAC can distribute the load over many nodes, a feature referred to as *server load balancing*. Oracle RAC features are available to you simply by connecting to an Oracle RAC system with a DataDirect Connect Series for ODBC driver. There is no additional configuration required.

Connection failover and *client load balancing* can be used in conjunction with an Oracle RAC system, but they are not specifically part of Oracle RAC. The drivers can also use these two features on DB2, Informix, SQL Server, and Sybase database systems. See [Using Failover](#) on page 78 for details about how these features work in DataDirect Connect Series for ODBC drivers.

Support of Materialized Views

When connected to an Oracle 9i or higher server, the Oracle driver supports the creation of materialized views. Materialized views are like any other database view with the following additions: the results are stored as a database object and the results can be updated on a schedule determined by the Create View statement.

Materialized views improve performance for data warehousing and replication. Refer to the Oracle documentation for more information about materialized views.

Stored Procedure Results

When you enable the Procedure Returns Results connection option, the driver returns result sets from stored procedures/functions. In addition, SQLGetInfo(SQL_MULT_RESULTS_SETS) returns Y and SQLGetInfo(SQL_BATCH_SUPPORT) returns SQL_BS_SELECT_PROC. If this option is enabled and you execute a stored procedure that does not return result sets, you incur a small performance penalty.

This feature requires that stored procedures be in a certain format. First, a package must be created to define all of the cursors used in the procedure; then, the procedure can be created using the new cursor. For example:

```

Create or replace package GEN_PACKAGE as
  CURSOR G1 is select CHARCOL from GTABLE2;
  type GTABLE2CHARCOL is ref cursor return G1%rowtype;
end GEN_PACKAGE;
Create or replace procedure GEN_PROCEDURE1 (
  rset IN OUT GEN_PACKAGE.GTABLE2CHARCOL, icol INTEGER) as
begin
  open rset for select CHARCOL from GTABLE2
    where INTEGERCOL <= icol order by INTEGERCOL;
end;
```

When executing the stored procedures with result sets, do not include the result set arguments (Oracle ref cursors) in the list of procedure parameters. The result set returned through the ref cursor is returned as a normal ODBC result set.

```
{call GEN_PROCEDURE1 (?)}
```

where ? is the parameter for the icol argument.

For more information, refer to your Oracle SQL documentation.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Note: If you are persisting a result set that contains Long data, you must enable the `EnableStaticCursorsforLongData` connection string attribute.

Isolation and Lock Levels Supported

Oracle supports isolation level 1 (read committed) and isolation level 3 (serializable). Oracle supports record-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the core SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions. In addition, the following functions are supported:

- `SQLColumnPrivileges`
- `SQLDescribeParam` (if `EnableDescribeParam=1`)
- `SQLForeignKeys`
- `SQLPrimaryKeys`
- `SQLProcedures`
- `SQLProcedureColumns`
- `SQLSetPos`
- `SQLTablePrivileges`

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The Oracle driver supports multiple connections and multiple statements per connection.

Using Arrays of Parameters

Oracle 8i and higher databases natively support parameter arrays, and the Oracle driver, in turn, supports them when connected to these versions of Oracle databases. When designing an application for performance, using native parameter arrays for bulk inserts or updates, for example, can improve performance.

Refer to "Designing ODBC applications for performance optimization" in the *Progress DataDirect for ODBC Drivers Reference* for details.

If the database does not support parameter arrays, the Oracle driver emulates them so that you can design your applications to use arrays of parameters and take advantage of the performance improvements where applicable. The driver emulates parameter arrays by sending individual rows to the database.

The SQL Server Legacy Wire Protocol Driver

The DataDirect Connect for ODBC and DataDirect Connect64 for ODBC SQL Server Legacy Wire Protocol driver (the SQL Server Legacy Wire Protocol driver) each support the following Microsoft SQL Server database servers.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The SQL Server Legacy Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect product for the file name of the SQL Server Legacy Wire Protocol driver.

Driver Requirements

The driver has no client requirements.

Windows



For support of Microsoft SQL Server 7.0, 2000, and 2005, the driver requires the SQL Server 7.0 versions of Net-Library DLL files, which are installed when you install the SQL Server Legacy Wire Protocol driver. The driver communicates with network software through the SQL Server Net-Library interface.

UNIX and Linux

UNIX[®] To use the SQL Server Legacy Wire Protocol driver on UNIX and Linux, you must have TCP/IP configured on both the UNIX and Linux clients and the Windows server on which the Microsoft SQL Server database resides. The UNIX and Linux SQL Server TCP/IP network client library is built into the SQL Server Legacy Wire Protocol driver on UNIX and Linux.

The Microsoft SQL Server Client configuration has been merged with the ODBC driver configuration and is set in the system information file.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 666 and [Connection Option Descriptions](#) on page 667 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions](#) on page 667 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (SQL Server Legacy)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX® On UNIX and Linux, data sources are stored in the odbc.ini file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.


When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Microsoft SQL Server data source:

1. Start the ODBC Administrator:

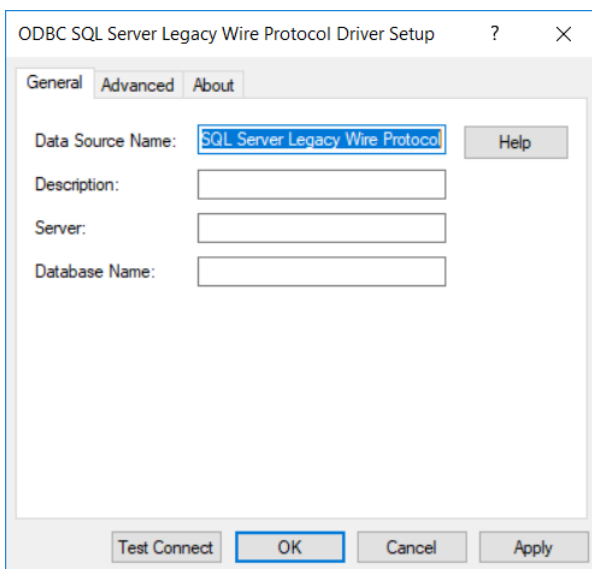
-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- **UNIX**[®] On Linux, change to the *install_dir/tools* directory and, at a command prompt, enter:
odbcadmin
where *install_dir* is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.
If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 69: General tab



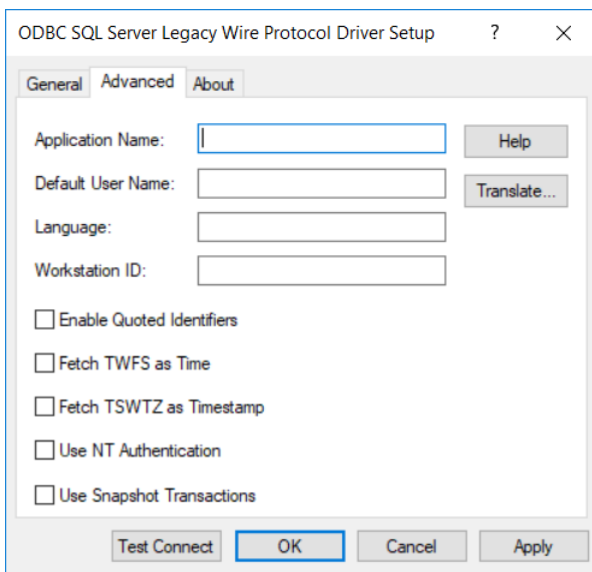
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 674	None
Description on page 676	None
Server on page 684	None
Database Name on page 675	None

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 70: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Application Name on page 671	None
Default User Name on page 675	None
Language on page 679	None
Workstation ID on page 687	None
Enable Quoted Identifiers on page 676	Disabled
Fetch TWFS as Time on page 678	Disabled
Fetch TSWTZ as Timestamp on page 677	Disabled

Connection Options: Advanced	Default
Use NT Authentication on page 686 WINDOWS ONLY	Disabled
Use Snapshot Transactions on page 686	Disabled
IANAAppCodePage on page 678UNIX ONLY	4 (ISO 8559-1 Latin-1)



Translate : Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

5. **UNIX**[®] Optionally, click the **Failover** tab to specify failover data source settings. This tab is available only on UNIX and Linux.

See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 680	Disabled
Alternate Servers on page 670	None
Connection Retry Count on page 673	0
Connection Retry Delay on page 674	3

6. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A Login dialog box appears; see [Using a Connection String](#) on page 666 for details. Note that the information you enter in the Login dialog box during a test connect is not saved.
 - If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.
7. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={}{driver_name[]}[:attribute=value[:attribute=value]...]
```

[Connection Option Descriptions](#) on page 667 provides the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Microsoft SQL Server is:

```
DSN=ACCOUNTING;DATABASE=ACCT
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=SQLServer.dsn;DATABASE=ACCT
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 SQL Server Legacy Wire Protocol};  
DATABASE=ACCT;SERVER=SQL2;UID=JOHN;PWD=XYZZY
```

The connection string attribute names are case-sensitive.

Using a Login Dialog Box (SQL Server Legacy)

Some ODBC applications display a Login dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

In the Login dialog box, provide the following information:

1. Type an IP address in Server in the following format: *IP_address, port_number*. For example, you can enter *199.226.224.34, 5000*. If your network supports named servers, you can specify an address as: *server_name, port_number*. For example, you can enter *SSserver, 5000*.

The IP address can be specified in IPv4 on Windows, and in either IPv4 or IPv6 format, or a combination of the two, on UNIX. See [Using IP Addresses](#) on page 67 for details about these formats.

To specify a named instance of Microsoft SQL Server, use the format: *server_name\instance_name*. If only a server name is specified with no instance name, the driver uses the default named instance on the server.



Type the name of a server on your network. It must be an entry on the Alias tab of the SQL Server Network Client Utility or the network name of a server running Microsoft SQL Server.

You can enter *(local)* when the driver is on the same computer as the Microsoft SQL Server database. You can connect to a local copy of Microsoft SQL Server, even when it is a non-networked version. Microsoft SQL Server 2000 and higher support multiple instances of Microsoft SQL Server running on the same computer.



2. Select the **Use Trusted Connection** check box to specify that the SQL Server Legacy Wire Protocol driver request a secure (or trusted) connection to Microsoft SQL Server. SQL Server uses integrated login security to establish connections using this data source, regardless of the current login security mode at the server. Any login ID or password supplied is ignored. The Microsoft SQL Server system administrator must have associated your Windows network ID with a Microsoft SQL Server login ID.

Clear this box to specify that Microsoft SQL Server use standard login security to establish connections using this data source. You must specify a login ID and password for all connection requests.

3. Type the Microsoft SQL Server login ID to use for the connection if Use Trusted Connection is not selected. If Use Trusted Connection is selected, the Login ID field is disabled.
4. Type the password to use for the connection if Use Trusted Connection is not selected. If Use Trusted Connection is selected, the Password field is disabled.
5. Type the name of the database to which you want to connect. If you do not specify a value, the default database defined by Microsoft SQL Server is used.
6. Click **OK** to log on to the Microsoft SQL Server database installed on the server you specified and to update the values in the Registry.

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog. The connection string attribute name is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

Note: SQL Server driver connection string attributes do not use short name equivalents.

The following lists the connection string attributes supported by the SQL Server Legacy Wire Protocol driver on Windows.

Windows

Table 43: SQL Server Legacy Wire Protocol Attribute Names on Windows

Attribute	Default
AnsiNPW	yes (Enabled)
APP	None
AttachDBFileName	None
AutoTranslate	yes (Enabled)
DATABASE	None
DataSourceName (DSN)	None
Description (n/a)	None
FetchTSWTZasTimestamp (FTSWTZAT)	0 (Disabled)
FetchTWFSasTime (FTWFSAT)	0 (Disabled)
LANGUAGE	None
Network	None
PWD (use Password for odbc.ini file)	None
QueryLog_On	no (Disabled)
QueryLogFile	None
QueryLogTime	None
QuotedID	no (Disabled)
Regional	yes (Enabled)
SAVEFILE	None
SERVER	None
SnapshotSerializable	0 (Disabled)
StatsLog_On	no (Disabled)
StatsLogFile	None
Trusted_Connection	no (Disabled)

Attribute	Default
UID (use LogonID for odbc.ini file)	None
WSID	None

UNIX and Linux

The following table lists the connection string attributes supported by the SQL Server Legacy Wire Protocol driver on UNIX/Linux.

Note: SQL Server driver connection string attributes do not use short name equivalents.

Table 44: SQL Server Legacy Wire Protocol Attribute Names on UNIX/Linux

Attribute	Default
Address	None
AlternateServers	None
AnsiNPW	yes (Enabled)
APP	None
ConnectionRetryCount	0
ConnectionRetryDelay	3
DataSourceName	None
DATABASE	None
Description (n/a)	None
FetchTSWTZasTimestamp (FTSWTZAT)	0 (Disabled)
FetchTWFSasTime (FTWFSAT)	0 (Disabled)
IANAAppCodePage	4 (ISO 8559-1 Latin-1)
LANGUAGE	None
LoadBalancing (LB)	0 (Disabled)
PWD (use Password for odbc.ini file)	None
QuotedID	no (Disabled)
SnapshotSerializable	0 (Disabled)

Attribute	Default
UID (use LogonID for odbc.ini file)	None
WSID	None

Alternate Servers

UNIX[®]

Attribute

AlternateServers

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

(Address=addressvalue[, . . .])

You must specify the network address of each alternate server.

Example

The following two Alternate Servers values define two alternate database servers for connection failover:

```
AlternateServers=(Address=MySQLServer\Instance1,
Address="255.125.1.11, 5002")
```

In this example, the network address of the last alternate contains commas. In this case, enclose the network address with double quotation marks as shown.

Notes

- An alternate server address in IPv6 format must be enclosed in double quotation marks.

Default

None

GUI Tab

Failover tab

AnsiNPW

Attribute

AnsiNPW

Purpose

Determines whether ANSI-defined behaviors are exposed.

Valid Values

yes | no

Behavior

When set to yes (Enabled), the driver uses ANSI-defined behaviors for handling NULL comparisons, character data padding, warnings, and NULL concatenation. If the driver appears to be truncating trailing blank spaces, this attribute should be set to no.

When set to no (Disabled), ANSI-defined behaviors are not exposed.

Default

yes (Enabled)

GUI Tab

n/a

Application Name**Attribute**

APP

Purpose

The name Microsoft SQL Server uses to identify your application.

Valid Values

string

where:

string

is your application name.

Default

None

GUI Tab

[Advanced tab](#)

AttachDBFileName**Attribute**

AttachDBFileName

Purpose

The name of the primary file of an attachable database.

Valid Values

string

where:

string

is name of the primary file of an attachable database.

Include the full path and escape any slash (\) characters if using a C character string variable:

```
AttachDBFileName=C:\\MyFolder\\MyDB.mdf
```

This database is attached and becomes the default database for the connection. To use AttachDBFileName, you must also specify the database name in either the SQLDriverConnect DATABASE parameter or the SQL_COPT_CURRENT_CATALOG connection attribute. If the database was previously attached, Microsoft SQL Server will not reattach it; it will use the attached database as the default for the connection.

Default

None

GUI Tab

n/a

AutoTranslate

Attribute

AutoTranslate

Purpose

Determines how ANSI character strings are translated.

Valid Values

yes | no

Behavior

If set to yes (Enabled), ANSI character strings sent between the client and server are translated by converting through Unicode to minimize problems in matching extended characters between the code pages on the client and the server.

These conversions are performed on the client by the SQL Server Legacy Wire Protocol driver. This requires that the same ANSI code page (ACP) used on the server be available on the client.

These settings have no effect on the conversions that occur for the following transfers:

- Unicode SQL_C_WCHAR client data sent to char, varchar, or text on the server.
- Char, varchar, or text server data sent to a Unicode SQL_C_WCHAR variable on the client.
- ANSI SQL_C_CHAR client data sent to Unicode nchar, nvarchar, or ntext on the server.

- Unicode char, varchar, or text server data sent to an ANSI SQL_C_CHAR variable on the client.

If set to no (Disabled), character translation is not performed.

The SQL Server Legacy Wire Protocol driver does not translate client ANSI character SQL_C_CHAR data sent to char, varchar, or text variables, parameters, or columns on the server. No translation is performed on char, varchar, or text data sent from the server to SQL_C_CHAR variables on the client. If the client and Microsoft SQL Server are using different ACPs, then extended characters can be misinterpreted.

Default

yes (Enabled)

GUI Tab

n/a

Connection Retry Count

UNIX[®]

Attribute

ConnectionRetryCount

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

The Connection Retry Delay option specifies the wait interval, in seconds, to occur between retry attempts.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x , the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

Failover tab

Connection Retry Delay

UNIX[®]

Attribute

ConnectionRetryDelay

Purpose

The number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

Valid Values

0 | *x*

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to *x*, the driver waits between connection retry attempts the specified number of seconds.

Default

3

GUI Tab

Failover tab

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database Name**Attribute**

DATABASE

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where:

database_name

is the name of a valid database.

Default

None

GUI Tab

[General tab](#)

Default User Name**Attribute**

UID (use LogonID for odbc.ini file)

Purpose

The default user ID used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Advanced tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable Quoted Identifiers

Attribute

QuotedID

Purpose

Determines whether the driver allows the use of quoted identifiers.

Valid Values

yes | no

Behavior

If set to yes (Enabled), Microsoft SQL Server enforces ANSI rules regarding quotation marks. Double quotation marks can only be used for identifiers, such as column and table names. Character strings must be enclosed in single quotation marks, for example:

```
SELECT "au_id"  
FROM "authors"  
WHERE "au_lname" = 'O'Brien'
```

If set to no (Disabled), applications that use quoted identifiers encounter errors when they generate SQL statements with quoted identifiers.

Default

no (Disabled)

GUI Tab

[Advanced tab](#)

Fetch TSWTZ as Timestamp

Attribute

FetchTSWTZasTimestamp (FTSWTZAT)

Purpose

Determines whether the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_TYPE_TIMESTAMP or SQL_VARCHAR.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the timestamp with time zone data type as the ODBC type SQL_TYPE_TIMESTAMP. The time zone information in the fetched value is truncated. Use this value if your application needs to process values the same way as TIMESTAMP columns.

If set to 0 (Disabled), the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_VARCHAR. Use this value if your application requires the time zone information in the fetched value.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Fetch TWFS as Time

Attribute

FetchTWFSasTime (FTWFSAT)

Purpose

Determines whether the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIME` or `SQL_TYPE_TIMESTAMP`.

Supported only for Microsoft SQL Server 2008.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIME`. The fractional seconds portion of the value is truncated.

If set to 0 (Disabled), the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIMESTAMP`. The fractional seconds portion of the value is preserved. Time columns are not searchable when they are described and fetched as timestamp.

Notes

- When returning time with fractional seconds data as `SQL_TYPE_TIMESTAMP`, the Year, Month and Day parts of the timestamp must be set to zero.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)

- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Language

Attribute

LANGUAGE

Purpose

The national language to use for Microsoft SQL Server system messages.

Valid Values

lang

where:

lang

is the language to use for Microsoft SQL Server system messages. This overrides the default language specified for the login on the server. If no language is specified, the connection uses the default language specified for the login on the server.

Default

None

GUI Tab

[Advanced tab](#)

Load Balancing

UNIX[®]

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Default

0 (Disabled)

GUI Tab

Failover tab

Network

Attribute

Network

Purpose

The name of a network library dynamic-link library.

Valid Values

string

where:

string

is the name of a network library dynamic-link library. The name need not include the path and must not include the .DLL file name extension, for example, `Network=dbnmpntw`.

Default

None

GUI Tab

n/a

PWD**Attribute**

PWD (use Password for odbc.ini file)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

PWD need not be specified if the login has a NULL password.

Default

None

GUI Tab

n/a

QueryLogFile**Attribute**

QueryLogFile

Purpose

The full path and file name of a file to be used for logging data about long-running queries. The QueryLog_On option must be set to yes.

Valid Values

string

where:

string

is the full path and file name of the file to be used for logging data.

Default

None

GUI Tab

n/a

QueryLog_On

Attribute

QueryLog_on

Purpose

Determines whether data about long-running queries data is logged.

Valid Values

yes | no

Behavior

When set to yes (Enabled), logging data about long-running queries data is enabled on the connection.

When set to no (Disabled), long-running query data is not logged.

Default

no (Disabled)

GUI Tab

n/a

QueryLogTime

Attribute

QueryLogTime

Purpose

A digit character string specifying the threshold for logging data about long-running queries.

Valid Values

string

where *string* is a digit character string specifying the threshold in milliseconds, for logging data.

Any query that does not receive a response in the time specified is written to the long-running query log file.

Default

None

GUI Tab

n/a

Regional**Attribute**

Regional

Purpose

Determines how currency, date, and time data are converted.

Valid Values

yes | no

Behavior

When set to yes (Enabled), the SQL Server Legacy Wire Protocol driver uses client settings when converting currency, date, and time data to character data. The conversion is one way only; the driver does not recognize non-ODBC standard formats for date strings or currency values.

When set to no (Disabled), the driver uses ODBC standard strings to represent currency, date, and time data that is converted to string data.

Default

yes (Enabled)

GUI Tab

n/a

SAVEFILE**Attribute**

SAVEFILE

Purpose

The name of an ODBC data source file into which the attributes of the current connection are saved.

Valid Values

string

where

string

is the name of an ODBC data source file into which the attributes of the current connection are saved if the connection is successful.

Default

None

GUI Tab

n/a

Server

Attribute

SERVER

Attribute

Address

Purpose

The location of the server.

Valid Values

IP_address | *named_server* | *named_instance* | *server_name*

where:

IP_address

is the IP address of the server to which you want to connect. Specify this address as: *IP_address*, *port_number*. For example, you can enter 199.226.224.34, 5000.

The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details about these formats.

named_server

is the named server address of the server to which you want to connect. Specify this address as: *named_server*, *port_number*. For example, you can enter SSserver, 5000.

named_instance

is a named instance of Microsoft SQL Server. Specify this address as: *server_name**instance_name*. If only a server name is specified with no instance name, the driver uses the default named instance on the server.

server_name

is the name of a server on your network. It must be an entry on the Alias tab of the SQL Server Network Client Utility or the network name of a server running Microsoft SQL Server. You can enter (local) when the driver is on the same computer as the Microsoft SQL Server database. You can connect to a local copy of Microsoft SQL Server, even when it is a non-networked version. Microsoft

SQL Server 2000 and higher support multiple instances of Microsoft SQL Server running on the same computer.

Default

None

GUI Tab

General tab

StatsLogFile**Attribute**

StatsLogFile

Purpose

The full path and file name of a file to be used for recording SQL Server Legacy Wire Protocol driver performance data. The StatsLog_On option must be set to yes.

Valid Values

string

where:

string

is the full path and file name of the file to be used for recording data.

Default

None

GUI Tab

n/a

**StatsLog_On****Attribute**

StatsLog_On

Purpose

Determines whether SQL Server Legacy Wire Protocol driver performance data is made available.

Valid Values

yes | no

Behavior

When set to yes (Enabled), SQL Server Legacy Wire Protocol driver performance data is captured.

When set to no (Disabled), SQL Server Legacy Wire Protocol driver performance data is not available on the connection.

Default

no (Disabled)

GUI Tab

n/a

Use NT Authentication



Attribute

Trusted_Connection

Purpose

Specifies that the SQL Server Legacy Wire Protocol driver request a secure (or trusted) connection to Microsoft SQL Server.

Valid Values

0 | 1

Behavior

When set to 1 (Enabled), Microsoft SQL Server uses integrated login security to establish connections using this data source, regardless of the current login security mode at the server. Any login ID or password supplied is ignored. The Microsoft SQL Server system administrator must have associated your Windows network ID with a Microsoft SQL Server login ID.

When set to 0 (Disabled), Microsoft SQL Server uses standard login security to establish connections using this data source. In this case, you must specify a login ID and password for all connection requests.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Use Snapshot Transactions

Attribute

SnapshotSerializable

Purpose

Allows your application to use the snapshot isolation level if your Microsoft SQL Server database is configured for Snapshot isolation. Supported only for Microsoft SQL Server 2005 and higher.

Valid Values

0 | 1

Behavior

When set to 1 (Enabled) and your application has the transaction isolation level set to serializable, the application uses the snapshot isolation level.

When set to 0 (Disabled) and your application has the transaction isolation level set to serializable, the application uses the serializable isolation level.

This option is useful for existing applications that set the isolation level to serializable. Using Snapshot Transactions in this case allows you to change to the snapshot isolation level with no or minimum code changes. If developing a new application, you can code it to set the connection attribute `SQL_COPT_SS_TXN_ISOLATION` to the value `SQL_TXN_SS_SNAPSHOT`.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See Also

See [Isolation and Lock Levels Supported](#) on page 690 for details about using the snapshot isolation level.

See [Performance Considerations](#) on page 688 for details

Workstation ID

Attribute

WSID

Purpose

The workstation ID that is used by the client.

Valid Values

string

where:

string

is the workstation ID.

Default

None

GUI Tab[Advanced tab](#)

Performance Considerations

Use Snapshot Transactions (SnapshotSerializable): You must have your Microsoft SQL Server 2005 and higher database configured for snapshot isolation for this connection option to work. Snapshot Isolation provides transaction-level read consistency and an optimistic approach to data modifications by not acquiring locks on data until data is to be modified. This Microsoft SQL Server 2005 and higher feature can be useful if you want to consistently return the same result set even if another transaction has changed the data and 1) your application executes many read operations or 2) your application has long running transactions that could potentially block users from reading data. This feature has the potential to eliminate data contention between read operations and update operations. When this connection option is enabled, performance is improved due to increased concurrency.

See [Using The Snapshot Isolation Level](#) on page 691 for details.

Data Types

The following table shows how the Microsoft SQL Server data types are mapped to the standard ODBC data types. [Unicode Support](#) on page 690 lists Microsoft SQL Server to Unicode data type mappings.

Table 45: Microsoft SQL Server Data Types

SQL Server	ODBC
binary	SQL_BINARY
bigint ⁵⁷	SQL_BIGINT
bit	SQL_BIT
char	SQL_CHAR
date ⁵⁸	SQL_TYPE_DATE
datetime	SQL_TYPE_TIMESTAMP
datetime2 ⁵⁸	SQL_TYPE_TIMESTAMP
datetimeoffset ^{58, 59}	SQL_WVARCHAR
decimal	SQL_DECIMAL
decimal() identity	SQL_DECIMAL

⁵⁷ Bigint supported on Windows driver only.

⁵⁸ Supported only on Microsoft SQL Server 2008 and higher.

⁵⁹ Datetimeoffset mapping changes based on the setting of the Fetch TSWTZ as Timestamp option.

SQL Server	ODBC
float	SQL_FLOAT
image	SQL_LONGVARBINARY
int	SQL_INTEGER
int identity	SQL_INTEGER
money	SQL_DECIMAL
numeric	SQL_NUMERIC
numeric() identity	SQL_NUMERIC
real	SQL_REAL
smalldatetime	SQL_TYPE_TIMESTAMP
smallint	SQL_SMALLINT
smallint identity	SQL_SMALLINT
smallmoney	SQL_DECIMAL
text	SQL_LONGVARCHAR
time ^{58, 60}	SQL_TYPE_TIMESTAMP
timestamp	SQL_VARBINARY
tinyint	SQL_TINYINT
tinyint identity	SQL_TINYINT
uniqueidentifier	SQL_GUID
varbinary	SQL_VARBINARY
varbinary(max) ⁶¹	SQL_LONGVARBINARY
varchar	SQL_VARCHAR
varchar(max) ⁶¹	SQL_LONGVARCHAR

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

⁶⁰ Time mapping changes based on the setting of the Fetch TWFS as Time option.

⁶¹ Supported only on Microsoft SQL Server 2005 and higher.

Unicode Support

The SQL Server Legacy Wire Protocol driver maps the Microsoft SQL Server data types to Unicode data types as shown in the following table:

SQL Server Data Type	Mapped to . . .
nchar	SQL_WCHAR
ntext	SQL_WLONGVARCHAR
nvarchar	SQL_WVARCHAR
nvarchar(max) ⁶²	SQL_WLONGVARCHAR
sysname	SQL_WVARCHAR
xml ⁶²	SQL_WLONGVARCHAR

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Advanced Features

The driver supports failover and its related connection options. Failover connection options are located on the Failover of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Note: **UNIX**[®] Only the UNIX/Linux driver supports this feature.

Isolation and Lock Levels Supported

Microsoft SQL Server supports isolation levels 0 (Read Uncommitted), 1 (Read Committed), 2 (Repeatable Read), and 3 (Serializable). Microsoft SQL Server supports row-level and table-level locking.

Microsoft SQL Server 2005 and higher supports the following additional isolation levels:

- Snapshot
- Read Committed with Snapshots
- Read Committed with Locks (equivalent to Read Committed in previous Microsoft SQL Server versions)

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

⁶² Supported only for Microsoft SQL Server 2005 and higher.

Using The Snapshot Isolation Level

The Snapshot isolation level is available only with Microsoft SQL Server 2005 and higher. Setting the SnapshotSerializable connection string attribute changes the behavior of the Serializable isolation level to use the Snapshot Isolation level. This allows an application to use the Snapshot Isolation level with minimal or no code changes.

If you are writing a new application, you may want to code it to set the connection attribute SQL_COPT_SS_TXN_ISOLATION to the value SQL_TXN_SS_SNAPSHOT. The application then uses the snapshot isolation level without requiring the Use Snapshot Transactions connection option.

See [Use Snapshot Transactions](#) on page 686 for additional information.

SQL Support

The driver supports the core SQL grammar.

ODBC Conformance Level

The driver supports ODBC conformance level 1.

In addition, the following function is supported: SQLDescribeParam.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The SQL Server Legacy Wire Protocol driver supports multiple connections and a single statement per connection.

Using Arrays of Parameters

Microsoft SQL Server databases natively support parameter arrays, and the SQL Server Legacy Wire Protocol driver, in turn, supports them. When designing an application for performance, using native parameter arrays for bulk inserts or updates, for example, can improve performance.

Refer to "Designing ODBC applications for performance optimization" in the *Progress DataDirect for ODBC Drivers Reference* for details.

The Text Driver

The DataDirect Connect for ODBC and DataDirect Connect64 for ODBC Text driver (the Text driver) supports ASCII text files.

These files can be printed directly or edited with text editors or word processors, because none of the data is stored in a binary format.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The 32-bit and 64-bit Text driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

The Text driver executes SQL statements directly on the text files. The driver supports Insert statements and inserts the record at the end of the file. You can execute Update and Delete statements conditionally.

The Text driver can access files up to 15 GB in size.

Refer to the readme file shipped with your DataDirect Connect product for the file names for the Text driver.

Driver Requirements

There are no client requirements for the Text driver.

Formats for Text Files

Some common formats for text files are listed in the following table.

Table 46: Text File Formats

Format	Description
Comma-separated values	Commas separate column values, and each line is a separate record. Column values can vary in length. These files often have the .CSV extension.
Tab-separated values	Tabs separate column values, and each line is a separate record. Column values can vary in length.
Character-separated values	Any printable character except single and double quotes can separate column values, and each line is a separate record. Column values can vary in length.
Fixed	No character separates column values. Instead, values start at the same position and have the same length in each line. The values appear in fixed columns if you display the file. Each line is a separate record.
Stream	No character separates column values nor records. The table is one long stream of bytes.

Comma-, tab-, and character-separated files are called character-delimited files because values are separated by a special character.

Configuring Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 696 and [Connection Option Descriptions](#) on page 697 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

UNIX® On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions](#) on page 697 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Text)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX® On UNIX and Linux, data sources are stored in the odbc.ini file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Text data source:

1. Start the ODBC Administrator:


-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- **UNIX**® On Linux, change to the `install_dir/tools` directory and, at a command prompt, enter:
`odbcadmin`

where `install_dir` is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

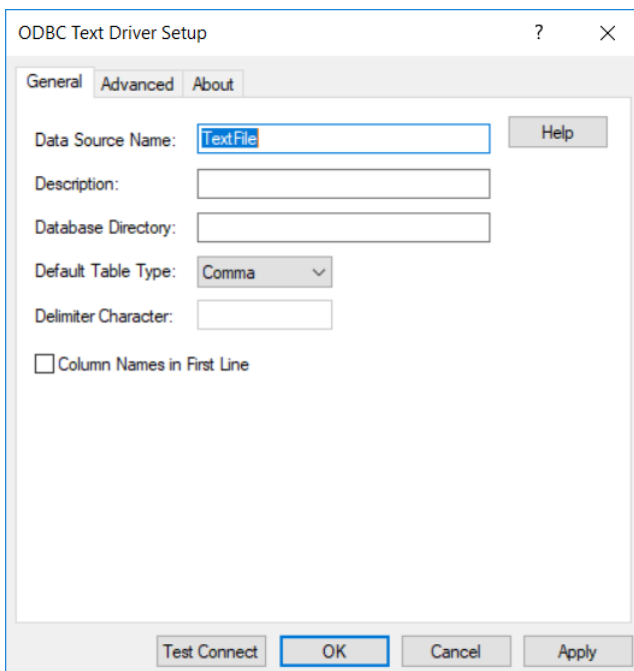
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 71: General tab



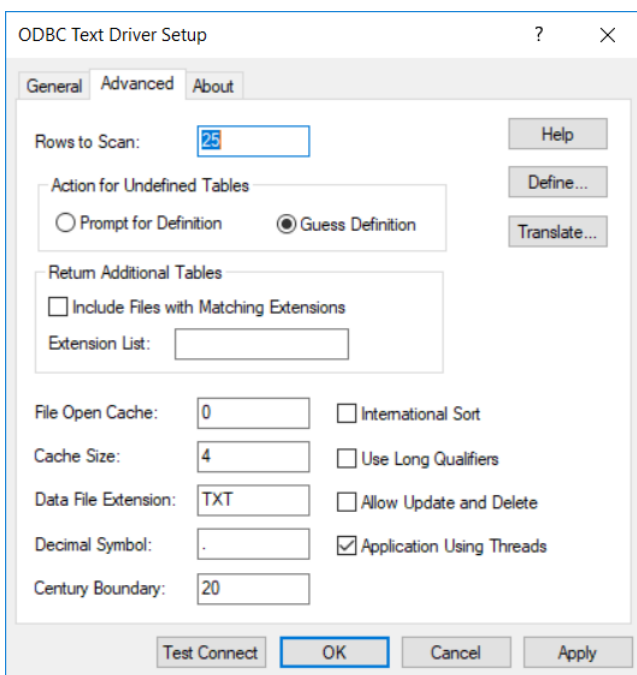
NOTE: The General tab displays the only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

3. On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 702	None
Description on page 705	None
Database Directory on page 703	None
Default Table Type on page 704	Comma
Delimiter Character on page 704	None
Column Names in First Line on page 701	Disabled

4. Optionally, click the **Advanced** tab to specify data source settings.

Figure 72: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Rows to Scan on page 709	25
Action for Undefined Tables on page 698	GUESS
Include Files with Matching Extensions on page 708	Disabled
Extension List on page 705	None
File Open Cache on page 706	0

Connection Options: Advanced	Default
International Sort on page 708	Disabled
Cache Size on page 700	4
Use Long Qualifiers on page 709 WINDOWS ONLY	Disabled
Data File Extension on page 702	TXT
Allow Update And Delete on page 699	Disabled
Decimal Symbol on page 703	. (Period)
Application Using Threads on page 699	Enabled
Century Boundary on page 700	20
IANAAppCodePage on page 707 UNIX ONLY	4 (ISO 8559-1 Latin-1)

Define: Click **Define** to define the structure of your text files as described in [Defining Table Structure on Windows](#) on page 710.



Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection properties specified in the driver Setup dialog box.
 - If the driver can connect, it releases the connection and displays a `connection established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.
- Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[:attribute=value[:attribute=value]. . .]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[:attribute=value[:attribute=value]. . .]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={ [driver_name] }[:attribute=value[:attribute=value]. . .]
```

[Connection Option Descriptions](#) on page 697 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Text is:

```
DSN=Text1;DB=C:\TEXTDATA;TT=Comma
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Text1.dsn;DB=C:\TEXTDATA;TT=Comma
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 TextFile (*.*)};DB=C:\TEXTDATA;TT=Comma
```

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Text driver.

Table 47: Text Attribute Names

Attribute (Short Name)	Default
AllowUpdateAndDelete (AUD)	0 (Disabled)
ApplicationUsingThreads (AUT)	1 (Enabled)
CacheSize (CSZ)	4
CenturyBoundary (CB)	20

Attribute (Short Name)	Default
Database (DB)	None
DataFileExtension (DFE)	TXT
DataSourceName (DSN)	None
DecimalSymbol (CS)	. (Period)
Delimiter (DC)	, (Comma)
Description (n/a)	None
ExtraExtensions (EE) WINDOWS ONLY	None
FileOpenCache (FOC)	0
FirstLineNames (FLN)	0 (Disabled)
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
IntlSort (IS)	0 (Disabled)
ScanRows (SR)	25
TableType (TT)	Comma
UndefinedTable (UT))	GUESS
UseLongQualifiers (ULQ) WINDOWS ONLY	0 (Disabled)

Action for Undefined Tables

Attribute

UndefinedTable (UT)

Purpose

Determines whether the driver prompts the user when it encounters a table for which it has no structure information.

Valid Values

PROMPT | GUESS

Behavior

Specify PROMPT to prompt the user.

Specify GUESS for the driver to guess the format of the file.

Default

GUESS

GUI Tab[Advanced tab](#)**Allow Update And Delete****Attribute**

AllowUpdateAndDelete (AUD)

Purpose

Allows Update and Delete statements. Because Update and Delete statements cause immediate changes to a text file, only one connection at a time can operate on a file. Each update and delete on a text file can cause significant changes to the file, and performance may be degraded. Consider a more appropriate database form if performance is a significant factor.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), text files are opened exclusively by the current connection.

If set to 0 (Disabled), Update and Delete statements are not allowed.

Default

0 (Disabled)

GUI Tab[Advanced tab](#)**Application Using Threads****Attribute**

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Cache Size

Attribute

CacheSize (CSZ)

Purpose

The number of 64 KB blocks the driver uses to cache database records. The larger the number of blocks, the better the performance.

Valid Values

0 | x

where:

x

is a positive integer that specifies the number of 64 KB blocks for caching.

Behavior

If set to 0, no records are cached.

If set to x , the specified number of 64 KB blocks are set aside for caching. The maximum number of blocks you can set depends on the system memory available. If the cache size is greater than 0, when browsing backwards, you are not able to see updates made by other users until you run the Select statement again.

Default

4

GUI Tab

[Advanced tab](#)

Century Boundary

Attribute

CenturyBoundary (CB)

Purpose

The cutoff year for century inference when converting two-digit dates to four-digit dates.

Valid Values

xx

where:

xx

is a two-digit number.

Behavior

Two-digit dates that are less than the specified year number are converted to 20xx. Two-digit dates greater than or equal to the number are converted to 19xx. For example, using the default value of 20, a date of 19 will be interpreted as 2019 and a date of 21 is interpreted as 1921.

Default

20

GUI Tab

[Advanced tab](#)

Column Names in First Line

Attribute

FirstLineNames (FLN)

Purpose

Determines whether the driver looks for column names in the first line of the file.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver looks for column names in the first line of the file.

If set to 0 (Disabled), the driver does not look for column names in the first line of the file.

Notes

- The Column Names in First Line setting applies only to tables not previously defined. It also determines the attributes of new tables created with the Create Table statement.

Default

0 (Disabled)

GUI Tab

[General tab](#)

Data File Extension

Attribute

DataFileExtension (DFE)

Purpose

A one- to three-character file name extension to use for data files.

Valid Values

ext

where:

ext

is the name of the one- to three-character file name extension.

Behavior

This value is used for all Create Table statements. Sending a Create Table using an extension other than the value specified for this option causes an error.

In other SQL statements, such as Select or Insert, users can specify an extension other than the one specified for this connection option. The Data File Extension value is used when no extension is specified.

Default

TXT

GUI Tab

[Advanced tab](#)

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database Directory**Attribute**

Database (DB)

Purpose

The directory that contains the data files.

Valid Values

database_directory

where:

database_directory

is the full path name of the directory in which the data files are stored. If no directory is specified, the current working directory is used.

Default

None

GUI Tab

[General tab](#)

Decimal Symbol**Attribute**

DecimalSymbol (CS)

Purpose

The decimal separator used when data is stored.

Valid Values

, | .

Behavior

If set to Comma (,), the driver uses a comma as the decimal separator.

If set to Period (.), the driver uses a period as the decimal separator.

The international decimal symbol (.) must be used in DML statements and parameter buffers.

Default

. (Period)

GUI Tab

[Advanced tab](#)

Default Table Type

Attribute

TableType (TT)

Purpose

The type of text file (table) that is used when creating a new table and opening an undefined table.

Valid Values

Comma | Tab | Character | Fixed | Stream

The value chosen determines the type of text used for a table: comma-separated, tab-separated, character-separated, fixed length, or stream.

Notes

- The Default Table Type setting applies only to tables not previously defined. It also determines the attributes of new tables created with the Create Table statement.

Default

Comma

GUI Tab

[General tab](#)

Delimiter Character

Attribute

Delimiter (DC)

Purpose

The character used as a delimiter for character-separated files.

Valid Values

x

where:

x

is any printable character except single quotes, double quotes, or semicolons.



Note that it is possible to specify a semicolon if you configure the data source using the Windows ODBC Administrator.

Notes

- The Delimiter Character setting applies only to tables not previously defined. It also determines the attributes of new tables created with the Create Table statement.

Default

, (Comma)

GUI Tab

[General tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Extension List

Attribute



ExtraExtensions (EE)

Purpose

A comma-separated list of file name extensions for the files that you want returned in addition to the extension specified in the Data File Extension field.

Valid Values

ext | NONE

where:

ext

is a file name extension.

To have files with no extensions returned, specify NONE. For example, if some of your files have the extensions TXT and CSV and others have no extension, specify *TXT, CSV, NONE*.

By default, when an application requests a list of tables, only files that have been defined are returned.

Notes

- You must have also enabled the Files with Matching Extensions option.

Default

None

GUI Tab

[Advanced tab](#)

File Open Cache

Attribute

FileOpenCache (FOC)

Purpose

The maximum number of used file handles to cache.

Valid Values

0 | *x*

where:

x

is a positive integer.

Behavior

If set to 0, no file open caching is performed.

If set to x , when a user opens and closes x tables, the tables are not actually closed. The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive. The advantage of file open caching is improved performance. The disadvantage is that a user who tries to open the file exclusively may get a file locking conflict even though no one appears to have the file open.

Default

0 (No File Open Caching)

GUI Tab

[Advanced tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The Driver Manager checks for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Include Files with Matching Extensions

Attribute

n/a

Purpose

On Windows, enables the driver to return files with a given file name extension in addition to the extension specified through the Data File Extension option. After enabling this option, specify the file name extensions through the Extension List option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns files with the file name extensions specified through the Extension List and Data File Extension options.

If set to 0 (Disabled), the driver returns only files with the file name extension specified through the Data File Extension option.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

International Sort

Attribute

IntlSort (IS)

Purpose

Uses international sort order as defined by your operating system when you issue a Select statement with an Order By clause.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), this order is always alphabetic, regardless of case; the letters are sorted as "A, b, C." Refer to your operating system documentation concerning the sorting of accented characters.

If set to 0 (Disabled), ASCII sort order is used. This order sorts items alphabetically with uppercase letters preceding lowercase letters. For example, "A, b, C" is sorted as "A, C, b."

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Rows to Scan**Attribute**

ScanRows (SR)

Purpose

The number of rows in a text file that the driver scans to determine the data types in the file.

Valid Values

0 | x

where:

x

is a positive integer.

Behavior

If set to 0, all rows in the file are scanned.

If set to x , x rows are scanned to determine the data types in a file.

Notes

- The Rows to Scan setting applies only to tables not previously defined. It also determines the attributes of new tables created with the Create Table statement.

Default

25

GUI Tab

[Advanced tab](#)

Use Long Qualifiers**Attribute**

UseLongQualifiers (ULQ)

Purpose

Determines whether the driver uses long path names.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), path names can be a maximum of 255 characters.

If set to 0 (Disabled), path names can be a maximum of 128 characters.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Defining Table Structure on Windows

Because text files do not all have the same structure, the driver provides the option of defining the structure of an existing file. Although defining the structure is not mandatory (the driver can attempt to guess the names and types of the columns), this feature is extremely useful.

To define the structure of a file:

1. Display the ODBC Text Driver Setup dialog box through the ODBC Administrator. Click the **Advanced** tab; then, click **Define** to display the Define File dialog box.
2. Select the correct file and click **Open** to display the Define Table dialog box.

Database Name: This field displays the name of the database directory that you selected in the Define File dialog box.

File: This field displays the name of the file that you selected in the Define File dialog box.

Table: Type a table name in the Table field. This name specifies the table name associated with the text file you selected earlier. The name can be a maximum of 32 characters and must be unique. This name is returned by SQLTables. By default, it is the file name without its extension (for example, Trc_read).

Column Names in First Line: Select this check box if the first line of the file contains column names; otherwise, do not select this box.

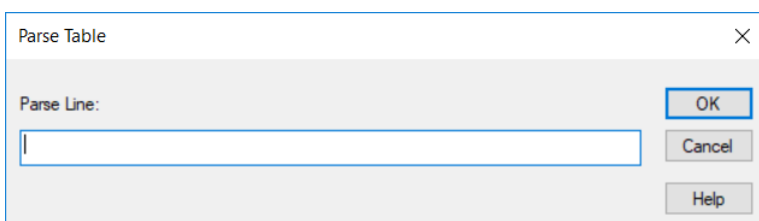
Table Type: Select the type of text file, either comma, tab, fixed, character, or stream.

Delimiter Character: If the table type is Character, type the delimiter used in character-separated files. The value can be any printable character except single and double quotes.

Decimal Symbol: Type the decimal separator used when data is stored. Valid values are a comma or a period. The international decimal symbol (.) must be used in DML statements and parameter buffers.

3. If you specified a comma-separated, tab-separated, or character-separated type in the Table Type field, the Guess/Parse button displays Guess. Click **Guess** to have the driver guess at the column names and display them in the list box of the Column Information pane.

If you specified a fixed-length or stream type in the Table Type field, the Guess/Parse button displays Parse. Click **Parse** to have the driver display the Parse Table dialog box and define the table columns.



This dialog box displays the first line of the file. You must mark where each field begins and ends by enclosing it in square brackets []. These brackets indicate the position and length of each field value in the record. Click **OK** to close the Parse Table dialog box. The driver will suggest column names in the list box of the Column Information pane.

4. If you do not want the driver to guess or parse, enter values in the following fields to define each column. Click **Add** to add the column name to the Column Information box.

Name: Type the name of the column.

Type: Select the data type of the column. If the field type is Date, the Mask field is enabled and you must select a date mask or type one in. See [Date Masks](#) on page 713 for more information.

Mask: Select a date mask. If you selected Date for the Type field, you must select a date mask for the field or type one in. See [Date Masks](#) on page 713 for more information.

Precision: Type the precision of the column. The precision of numeric data types is defined as the maximum number of digits used by the data type of the column. For character types, this is the length in characters of the data. Note that the precision and scale values determine how numeric data is to be returned.

Scale: Type the scale of the column. The scale of numeric data types is defined as the maximum number of digits to the right of the decimal point. Note that the precision and scale values determine how numeric data is to be returned.

Length: If you specified a fixed-length table type, type the length, which is the number of bytes the data takes up in storage.

Offset: If you specified a fixed-length table type, type the offset, which is the number of bytes from the start of the table to the start of the field.

5. To modify an existing column definition, select the column name in the Column Information box. Modify the values for that column name; then, click **Modify**.
6. To delete an existing column definition, select a column name in the Column Information box and click **Remove**.
7. Click **OK** to define the table.

Defining Table Structure on UNIX and Linux

UNIX[®] Because text files do not all have the same structure, the driver provides the option to define the structure of an existing file. Although defining the structure is not mandatory, because the driver can attempt to guess the names and types of the columns, this feature is extremely useful.

To define the structure of a text file, you create a QETXT.INI file using any plain text editor, such as vi. The file name must be in uppercase. All of the tables you want to define are specified in the QETXT.INI file. When you specify table attributes in QETXT.INI, you override the attributes specified in the system information file (odbc.ini) or in the connection string.

To define the QETXT.INI file:

1. Create a [Defined Tables] section and list all of the tables you are defining. Specify the text file name (in either upper or lowercase, depending on the file) followed by the name you want to give the table, for example:

```
emptext.txt=EMP
```

Table names can be up to 32 characters in length and cannot be the same as another defined table in the database. This name is returned by SQLTables. By default, it is the file name without its extension.

2. For each table listed in the [Defined Tables] section, you must specify the text file (FILE=), the table type (TT=), whether the first line of the file contains column names (FLN=), and the delimiter character (DC=).

- Specify the text file name. For example:

```
FILE=emptext.txt
```

- To define the table type, specify how the fields are separated (comma, tab, fixed, or character). For example:

```
TT=COMMA
```

- If the table type is CHARACTER, specify the delimiter character. The value can be any printable character except single and double quotes. For example, if the fields are separated by comma:

```
DC=,
```

- Specify whether the first line of the file contains column names, using 1 for yes and 0 for no. For example:

```
FLN=0
```

3. Define the fields in the table, beginning with FIELD1. For each field, specify the field name, field type, precision, scale, length, offset (for fixed tables), and date/time mask. See [Date Masks](#) on page 713 for information about masks.

Separate the values with commas. For example, to define two fields:

```
FIELD1=EMP_ID,VARCHAR,6,0,6,0,
```

```
FIELD2=HIRE_DATE,DATE,10,0,10,0,m/d/yy
```

4. Save the file as QETXT.INI. The driver looks for this file in the directory specified by the Database attribute in odbc.ini, or in the current directory.

Example of QETXT.INI

The following is an example of a QETXT.INI file. This file defines the structure of the emptext.txt file, which is a sample data file shipped with the DataDirect ODBC Text file.

```
[Defined Tables]
emptext.txt=EMP
[EMP]
FILE=emptext.txt
FLN=1
TT=Comma
FIELD1=FIRST_NAME, VARCHAR, 10, 0, 10, 0,
FIELD2=LAST_NAME, VARCHAR, 9, 0, 9, 0,
FIELD3=EMP_ID, VARCHAR, 6, 0, 6, 0,
FIELD4=HIRE_DATE, DATE, 10, 0, 10, 0, m/d/yy
FIELD5=SALARY, NUMERIC, 8, 2, 8, 0,
FIELD6=DEPT, VARCHAR, 4, 0, 4, 0,
FIELD7=EXEMPT, VARCHAR, 6, 0, 6, 0,
FIELD8=INTERESTS, VARCHAR, 136, 0, 136, 0,
```

Date Masks

Date masks tell the driver how a date is stored in a text file. When a value is inserted into a text file, the date is formatted so that it matches the mask. When reading a text file, the driver converts the formatted date into a date data type.

The following table lists the symbols to use when specifying the date mask.

Table 48: Date Masks for Text Driver

Symbol	Description
m	Output the month's number (1–12).
mm	Output a leading zero if the month number is less than 10.
mmm, Mmm, MMM	Output the three-letter abbreviation for the month depending on the case of the Ms (for example, jan, Jan, JAN).
mmmm, Mmmm, MMMM	Output the name of the full month depending on the case of the Ms (for example, january, January, JANUARY).
d	Output the day number (1–31).
dd	Output a leading zero if the day number is less than 10.
ddd, Ddd, DDD	Output the three-letter day abbreviation depending on the case of the Ds (for example, mon, Mon, MON).
dddd, Dddd, DDDD	Output the name of the full day depending on the case of the Ds (for example, monday, Monday, MONDAY).
yy	Output the last two digits of the year.
yyyy	Output the full four digits of the year.

Symbol	Description
J	Output the Julian value for the date. The Julian value is the number of days since 4712 BC.
\ - . : , (space)	Special characters used to separate the parts of a date.
\	Output the next character. For example, if the mask is mm/dd/yyyy \AD, the value appears as 10/01/2003 AD in the text file.
"string", 'string'	Output the string in the text file.

The following table shows some example date values, masks, and how the date appears in the text file.

Table 49: Date Mask Examples

Date	Mask	Value
2003-10-01	yyyy-mm-dd	2003-10-01
	m/d/yy	10/1/03
	Ddd, Mmm dd, yyyy	Fri, Oct 01, 2003

Data Types

The following table shows how the text file data types are mapped to the standard ODBC data types.

Table 50: Text Data Types

Text	ODBC
Numeric	SQL_NUMERIC
Date	SQL_TYPE_DATE
Varchar	SQL_VARCHAR

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

Select Statement

You use a SQL Select statement to specify the columns and records to be read. All of the Select statement clauses described in [SQL Statements for Flat-File Drivers](#) are supported by the Text driver.

Alter Table Statement

The Text driver supports the Alter Table statement to add one or more columns to a table or to delete (drop) a single column.

The Alter Table statement has the form:

```
ALTER TABLE table_name {ADD column_namedata_type |  
ADD(column_namedata_type [, column_namedata_type]... ) |  
DROP[COLUMN] column_name}
```

table_name is the name of the table to which you are adding or dropping columns.

column_name assigns a name to the column you are adding or specifies the column you are dropping.

data_type specifies the native data type of each column you add.

For example, to add two columns to the emp table:

```
ALTER TABLE emp (ADD startdate date, dept varchar(10))
```

You cannot add columns and drop columns in a single statement, and you can drop only one column at a time. For example, to drop a column:

```
ALTER TABLE emp DROP startdate
```

The Alter Table statement fails when you attempt to drop a column upon which other objects, such as indexes or views, are dependent.

SQL Support

The driver supports the minimum SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions. In addition, the SQLSetPos function is supported.

The driver supports backward and random fetching in SQLExtendedFetch and SQLFetchScroll.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

Text files support multiple connections and multiple statements per connection.

Drivers Only Available for 32-Bit Platforms

The following sections describe the drivers that are available only in 32-bit versions. See [Drivers for 32-Bit and 64-Bit Platforms](#) on page 139 and [The Connect XE Drivers](#) on page 827 for information on additional Connect Series drivers.

For details, see the following topics:

- [The Btrieve \(Pervasive.SQL\) Driver](#)
- [The dBASE Driver](#)
- [The Informix Driver](#)
- [The XML Driver](#)

The Btrieve (Pervasive.SQL) Driver

The DataDirect Connect for ODBC Btrieve driver (the Btrieve driver) supports the following versions of Btrieve files:

- Pervasive.SQL
- Btrieve

The driver executes SQL statements directly on Btrieve files.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The Btrieve driver is 32-bit only and is supported in the Windows environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect product for the file name of the Btrieve driver.

Driver Requirements

To access a Btrieve database, you must be using the appropriate client software for the version of the Btrieve database to which you are connecting:

Database Versions	Client Names
Pervasive.SQL 8.5	Pervasive.SQL 8.5 client software
Pervasive.SQL 2000	Pervasive.SQL 2000 client software
Pervasive.SQL 7.0	Pervasive.SQL 7.0 client software
Btrieve 6.15 for Windows 9x	Btrieve Developer's Kit or Btrieve WorkStation Client Engine
Btrieve 6.15 for Windows NT	Btrieve Developer's Kit, Btrieve WorkStation Client Engine, or Btrieve Client/Server Database Engine

Note: The Btrieve driver may experience problems if the Btrieve Microkernel Engine's communication buffer size is smaller than that of the Btrieve driver's Array Size option. You can increase the communication buffer size with the Pervasive Software Setup Utility, or you can decrease the value of Array Size option through the ODBC Btrieve Driver setup dialog box or through the ArraySize connection string attribute.

Before you attempt to access Btrieve files, you must incorporate existing Btrieve files into a Scalable SQL database. See [Managing Databases](#) on page 718 for information about Scalable SQL databases.

Managing Databases

If you already use Scalable SQL, the Btrieve driver can access your Scalable SQL databases directly. If not, your Btrieve files must be incorporated into a Scalable SQL database.

A Scalable SQL database is composed of data files that contain your records and data dictionary files that describe the database. The data files are Btrieve files. The data dictionary files are special Btrieve files that contain descriptions of the data files, views, fields, and indexes in your database.

All Btrieve files in a Scalable SQL database must reside in the same directory. In addition to the Btrieve data files, the three data dictionary files (FILE.DDF, FIELD.DDF, and INDEX.DDF) also must be in the directory.

Incorporating a Btrieve file into a Scalable SQL database does not change the Btrieve file in any way. You can continue to access the file directly with any existing Btrieve application.

Transactions

The Btrieve driver supports *transactions*. A transaction is a series of database changes that is treated as a single unit. In applications that do not use transactions, the Btrieve driver immediately executes Insert, Update, and Delete statements on the database files and the changes are automatically committed when the SQL statement is executed. You cannot undo these changes. In applications that use transactions, the Btrieve driver holds inserts, updates, and deletes until you issue a Commit or Rollback. A Commit saves the changes to the database file; a Rollback undoes the changes.

Transactions affect the removal of record locking. All locks are removed when SQLTransact is called with the Commit or Rollback option to end the active transaction.

To use the Btrieve driver's transaction processing capabilities, consult the Pervasive documentation.

Configuring and Connecting to Data Sources (Btrieve)

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box. See [Data Source Configuration through a GUI \(Btrieve\)](#) on page 719 for details.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 722 and [Connection Option Descriptions](#) on page 723 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration through a GUI (Btrieve)

On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Btrieve data source:

1. Start the ODBC Administrator by selecting its icon from the DataDirect Connect program group; then, select a tab:
 - **User DSN:** If you are configuring an existing user data source, select the data source name on the User DSN tab and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** on the User DSN tab to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

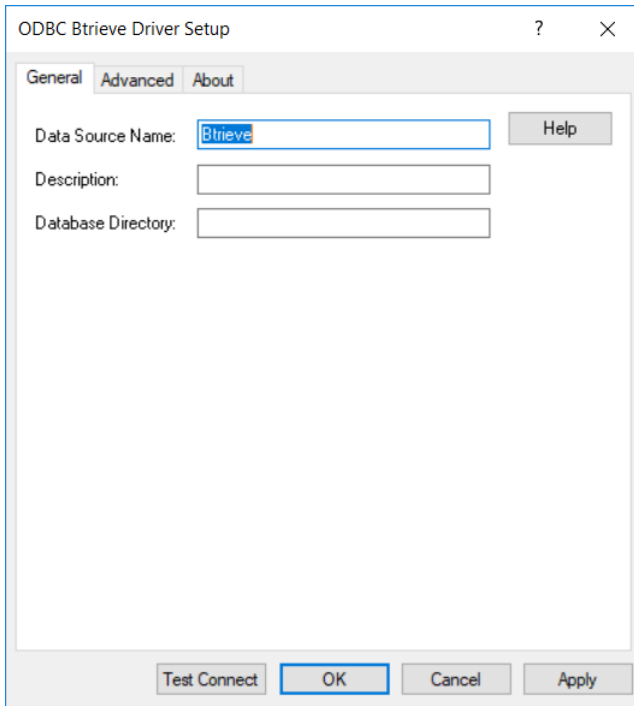
- **System DSN:** To configure a new system data source, click **Add** on the System DSN tab to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source name on the File DSN tab and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** on the File DSN tab to display a list of installed drivers. Select the driver and click **Next**. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 73: General tab



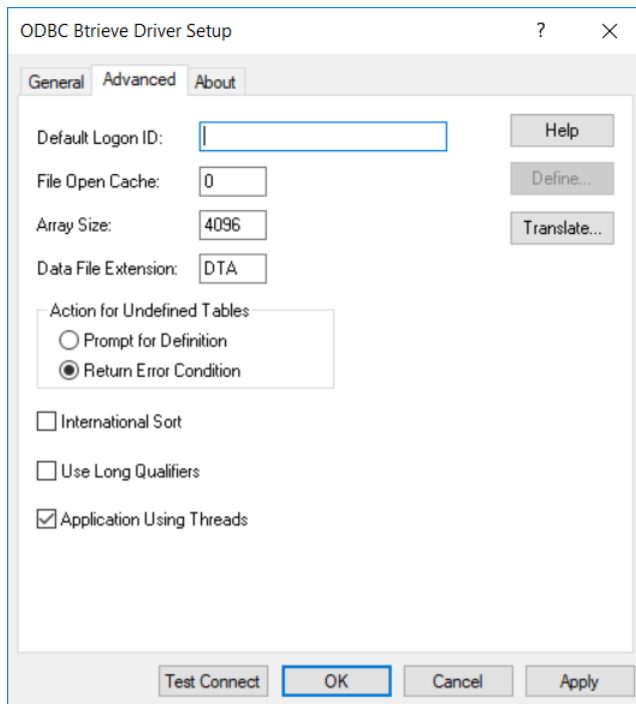
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

2. On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 726	None
Description on page 728	None
Database Directory on page 727	None

3. Optionally, click the **Advanced** tab to specify data source settings.

Figure 74: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Default Logon ID on page 727	None
File Open Cache on page 728	0
Array Size on page 725	4096
Data File Extension on page 726	DTA
Action for Undefined Tables on page 724	Return Error Condition
International Sort on page 729	Disabled
Use Long Qualifiers on page 730	Disabled
Application Using Threads on page 725	Enabled

Define: Click **Define** to define table structure as described in [Defining Table Structure](#) on page 731.

Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

4. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection properties specified in the driver Setup dialog box.
 - If the driver can connect, it releases the connection and displays a `Connection established!` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message.

Verify that all required client software is properly installed. If it is not, you will see the message:

```
Specified driver could not be loaded due to system error [xxx].
```

Click **OK**.

5. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={}{driver_name[]}[:attribute=value[;attribute=value]...]
```

[Connection Option Descriptions](#) on page 723 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Btrieve is:

```
DSN=BTRIEVE FILES;DB=J:\Btrvdata
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Btrieve.dsn;DB=J:\Btrvdata
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Btrieve};DB=J:\Btrvdata;UID=JOHN;PWD=XYZZY
```

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{ }driver_name[ ]][;attribute=value[;attribute=value]...]
```

[Connection Option Descriptions](#) on page 723 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Btrieve is:

```
DSN=BTRIEVE FILES;DB=J:\Btrvdata
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Btrieve.dsn;DB=J:\Btrvdata
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Btrieve};DB=J:\Btrvdata;UID=JOHN;PWD=XYZZY
```

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name. For example:

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Btrieve driver.

Table 51: Btrieve Attribute Names

Attribute (Short Name)	Default
ApplicationUsingThreads (AUT)	1 (Enabled)
ArraySize (AS)	4096
Database (DB)	None
DataFileExtension (DFE)	DTA
DataSourceName (DSN)	None
Description (n/a)	None
FileOpenCache (FOC)	0 (No File Open Caching)
IntlSort (IS)	0 (Disabled)
LogonID (UID)	None
Password (PWD)	None
UndefinedTable (UT)	Error
UseLongQualifiers (ULQ)	0 (Disabled)

Action for Undefined Tables

Attribute

UndefinedTable (UT)

Purpose

Determines whether the driver prompts the user when it encounters a table for which it has no structure information.

Valid Values

PROMPT | ERROR

Specify PROMPT to prompt the user.

Specify ERROR to return an error.

Default

ERROR (driver returns an error)

GUI Tab

[Advanced tab](#)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Array Size

Attribute

ArraySize (AS)

Purpose

The number of bytes in the array. This connection option enables the driver to retrieve an array of records from the Btrieve database and, in most cases, results in improved performance for the application.

Valid Values

A positive integer from 1 to 65535

Default

4096

GUI Tab

[Advanced tab](#)

Data File Extension

Attribute

DataFileExtension (DFE)

Purpose

A one- to three-character file name extension to use for data files.

Valid Values

ext

where:

ext

is the name of the one- to three-character file name extension.

Behavior

This value is used for all Create Table statements. Sending a Create Table using an extension other than the value specified for this option causes an error.

In other SQL statements, such as Select or Insert, users can specify an extension other than the one specified for this connection option. The Data File Extension value is used when no extension is specified.

Default

DTA

GUI Tab

[Advanced tab](#)

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database Directory**Attribute**

Database (DB)

Purpose

The directory that contains the data files.

Valid Values

database_directory

where:

database_directory

is the full path name of the directory in which the data files are stored. If no directory is specified, the current working directory is used.

This includes both Btrieve files and the data dictionary files (.DDF). Data dictionary files describe the structure of Btrieve data.

Default

None

GUI Tab

[General tab](#)

Default Logon ID**Attribute**

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Advanced tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

File Open Cache

Attribute

FileOpenCache (FOC)

Purpose

The maximum number of used file handles to cache.

Valid Values

0 | *x*

where:

x

is a positive integer.

Behavior

If set to 0, no file open caching is performed.

If set to x , when a user opens and closes x tables, the tables are not actually closed. The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive. The advantage of file open caching is improved performance. The disadvantage is that a user who tries to open the file exclusively may get a file locking conflict even though no one appears to have the file open.

Default

0 (No File Open Caching)

GUI Tab

[Advanced tab](#)

International Sort

Attribute

IntlSort (IS)

Purpose

Uses international sort order as defined by your operating system when you issue a Select statement with an Order By clause.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), this order is always alphabetic, regardless of case; the letters are sorted as "A, b, C." Refer to your operating system documentation concerning the sorting of accented characters.

If set to 0 (Disabled), ASCII sort order is used. This order sorts items alphabetically with uppercase letters preceding lowercase letters. For example, "A, b, C" is sorted as "A, C, b."

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Password

Attribute

Password (PWD)

Purpose

The password that you must enter if your Scalable SQL data dictionary files have security restrictions set. The Password option cannot be specified through the Administrator GUI.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Use Long Qualifiers

Attribute

UseLongQualifiers (ULQ)

Purpose

Determines whether the driver uses long path names.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), path names can be a maximum of 255 characters.

If set to 0 (Disabled), path names can be a maximum of 128 characters.

Default

0 (Disabled)

GUI Tab

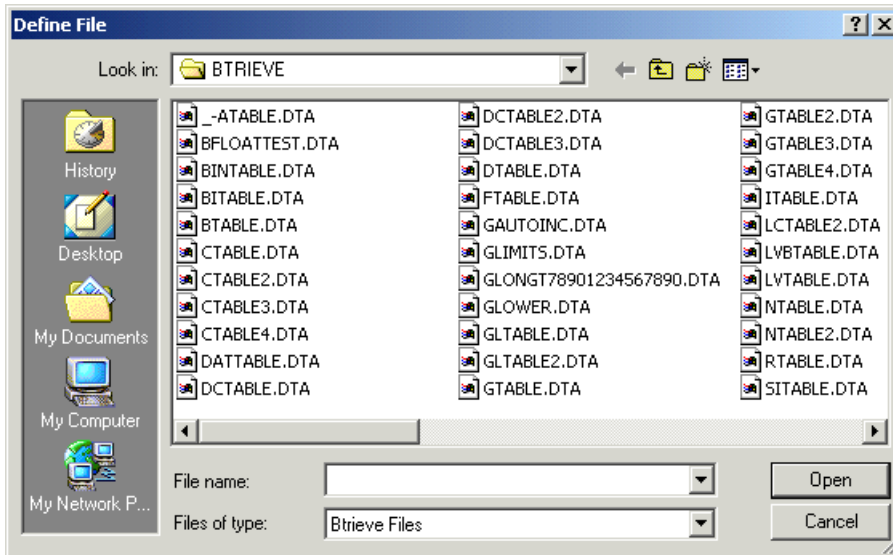
[Advanced tab](#)

Defining Table Structure

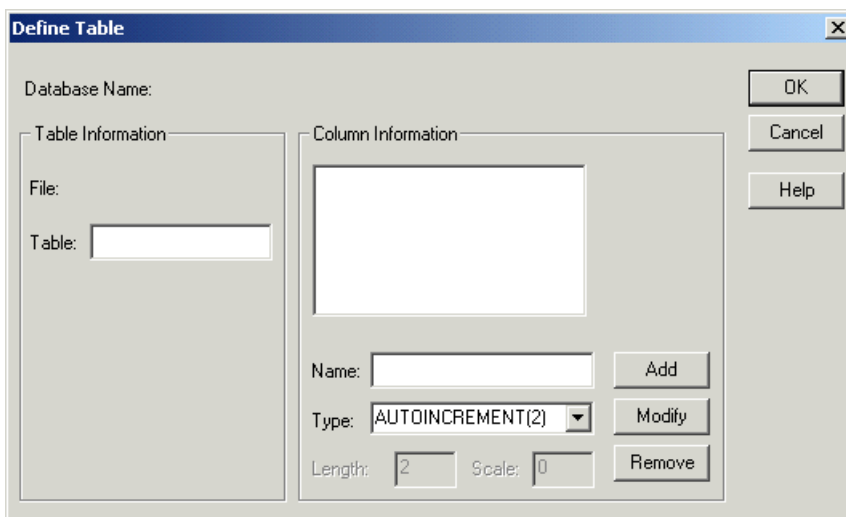
Because Btrieve does not store any column information in the data file, you may need to define its structure. Tables created by the Btrieve driver or by Scalable SQL will not require this. Utilities are also available from Pervasive Software that will perform this operation.

To define the structure of a file:

1. Display the ODBC Btrieve Driver Setup dialog box through the ODBC Administrator. Click the **Advanced** tab; then, click **Define** to display the Define File dialog box.



2. In the Define File dialog box, select the file you want to define and click **Open** to display the Define Table dialog box.



Database Name: This field displays the directory in which the file you selected in the Define File dialog box is located.

File: This field displays the name of the file that you selected in the Define File dialog box.

Table: Type the name of the table to be returned by SQLTables. The name can be a maximum of 20 characters and cannot be the same as another defined table in the database. This field is required.

- Specify values in the following fields to define each column. Click **Add** to add the column name to the list box.

Name: Type the name of the column.

Type: Select the data type of the column.

Length: Type the length of the column, if applicable.

Scale: Type the scale of the column, if applicable.

- To modify an existing column definition, select the column name in the list box. Modify the values for that column name; then, click **Modify**.
- To delete an existing column definition, select a column name in the list box and click **Remove**.
- Click **OK** to define the table.

Data Types

The following table shows how the Btrieve data types map to the standard ODBC data types. The Btrieve data types are used when you incorporate Btrieve files into a Scalable SQL database.

Table 52: Btrieve Data Types

Btrieve	ODBC
Autoincrement(2)	SQL_SMALLINT
Autoincrement(4)	SQL_INTEGER
Bfloat(4)	SQL_REAL
Bfloat(8)	SQL_DOUBLE
Bit	SQL_BIT
Blob	SQL_LONGVARGINARY
Char	SQL_CHAR
Currency	SQL_DECIMAL
Date	SQL_TYPE_DATE
Decimal	SQL_DECIMAL
Float(4)	SQL_REAL
Float(8)	SQL_DOUBLE
Integer(1)	SQL_TINYINT

Btrieve	ODBC
Integer(2)	SQL_SMALLINT
Integer(4)	SQL_INTEGER
Integer(8)	SQL_BIGINT
Logical(1)	SQL_BIT
Logical(2)	SQL_BIT
Lstring	SQL_VARCHAR
Money	SQL_DECIMAL
Note	SQL_LONGVARCHAR
Numeric	SQL_NUMERIC
Numericsts	SQL_NUMERIC
Time	SQL_TYPE_TIME
Timestamp	SQL_TYPE_TIMESTAMP
Unsigned(1)	SQL_TINYINT
Unsigned(8)	SQL_BIGINT
Zstring	SQL_VARCHAR

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

Indexes

Note: If you define an index using the Btrieve driver, the index will not have the restrictions discussed here.

For query optimization, the Btrieve driver does not use the following:

- Indexes containing all-segment-null keys or any-segment-null keys.
- Any index key that is marked case-insensitive.
- Any index keys where the data type of the index key does not match the data type of the field. The one exception is if the index key is declared as an unsigned integer and the field in the file is declared as signed integer, or vice versa, then the driver assumes the field contains only unsigned quantities and uses the index. Note that this can lead to incorrect results if the field in fact does contain signed quantities.

The Btrieve driver only uses an alternate-collating-sequence (ASC) index key for equality lookups. Additionally, if an ASC key is part of a segmented index, the other index segments are not used for query optimization unless the Where clause contains an equality condition for the ASC key.

Column Names

Column names in SQL statements (such as Select and Insert) can be up to 20 characters long. If column names are in all lowercase, a combination of upper and lowercase, contain blank spaces, or are reserved words, they must be surrounded by the grave character (`) (ASCII 96). For example:

```
SELECT `name` FROM emp
```

Select Statement

You use the SQL Select statement to specify the columns and records to be read. Btrieve Select statements support all the Select statement clauses described in [SQL Statements for Flat-File Drivers](#). This section describes the information that is specific to Btrieve.

Rowid Pseudo-Column

Each Btrieve record contains a special column named Rowid. This field contains a unique number that indicates the record's sequence in the database. You can use Rowid in Where and Select clauses.

Rowid is particularly useful when you are updating records. You can retrieve the Rowid of the records in the database along with the other field values. For example:

```
SELECT last_name, first_name, salary, rowid FROM emp
```

Then, you can use the Rowid of the record that you want to update to ensure that you are updating the correct record and no other. For example:

```
UPDATE emp set salary = 40000 FROM emp WHERE rowid=21
```

The fastest way of updating a single row is to use a Where clause with the Rowid. You cannot update the Rowid column.

Select statements that use the Rowid pseudo-column in the Where clause achieve maximum performance only for exact equality matches. If you use range scans instead of exact equality matches, a full table scan is performed. For example:

```
SELECT * FROM emp WHERE rowid=21 //fast search
SELECT * FROM emp WHERE rowid <=25 //full table scan
```

Alter Table Statement

The Btrieve driver supports the Alter Table statement to add one or more columns to a table or to delete (drop) a single column.

The Alter Table statement has the form:

```
ALTER TABLE table_name {ADD column_namedata_type | ADD (column_namedata_type [,
column_namedata_type]...)
| DROP [COLUMN] column_name}
```

table_name is the name of the table to which you are adding or dropping columns.

column_name assigns a name to the column you are adding or specifies the column you are dropping.

data_type specifies the native data type of each column you add.

For example, to add two columns to the emp table:

```
ALTER TABLE emp (ADD startdate date, dept char 10)
```

You cannot add columns and drop columns in a single statement, and you can drop only one column at a time. For example, to drop a column:

```
ALTER TABLE emp DROP startdate
```

The Alter Table statement fails when you attempt to drop a column upon which other objects, such as indexes or views, are dependent.

Create and Drop Index Statements

The Btrieve driver supports SQL statements to create and delete indexes.

Create Index

The Create Index statement for Btrieve files has the form:

```
CREATE [UNIQUE] INDEX index_name ON table_name ([field_name [ASC | DESC] [,field_name
[ASC | DESC]]...)
```

Unique means that Btrieve does not let you insert two records with the same index values.

index_name is the name of the index.

table_name is the name of the table on which the index is to be created.

ASC tells Btrieve to create the index in ascending order. DESC tells Btrieve to create the index in descending order. By default, indexes are created in ascending order. For example:

```
CREATE INDEX lname ON emp (last_name)
```

Drop Index

The form of the Drop Index statement is:

```
DROP INDEX table_name.index_name
```

table_name is the name of the table from which the index is to be dropped.

index_name is the name of the index.

For example:

```
DROP INDEX emp.lname
```

Isolation and Lock Levels Supported

Btrieve supports isolation level 1 (read committed) only. Btrieve supports record-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the minimum SQL grammar with several core extensions.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

In addition, the following function is supported: SQLSetPos.

The driver also supports backward and random fetching in SQLExtendedFetch and SQLFetchScroll.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

Btrieve files support a single connection and multiple statements per connection.

The dBASE Driver

The DataDirect Connect for ODBC dBASE driver (the dBASE driver) supports the following file types:

- dBASE
- Clipper
- FoxPro
- FoxPro database container (DBC)

For the latest support information, visit the Progress DataDirect Supported Configurations page:
<https://www.progress.com/supported-configurations/datadirect>.

The dBASE driver runs the SQL statements directly on dBASE- and FoxPro-compatible files. You do not need to own dBASE or FoxPro products to access these files. The dBASE driver cannot access files that are larger than 2 GB.

The dBASE driver is 32-bit only and is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect product for the file name of the dBASE driver.

Driver Requirements

There are no client requirements for the dBASE driver.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 743 and [Connection Option Descriptions](#) on page 744 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See Environment Configuration for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

Connection Option Descriptions lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (dBase)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX® On UNIX and Linux, data sources are stored in the odbc.ini file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a dBASE data source:

1. Start the ODBC Administrator:

-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- **UNIX**® On Linux, change to the *install_dir/tools* directory and, at a command prompt, enter:


```
odbcadmin
```

where *install_dir* is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

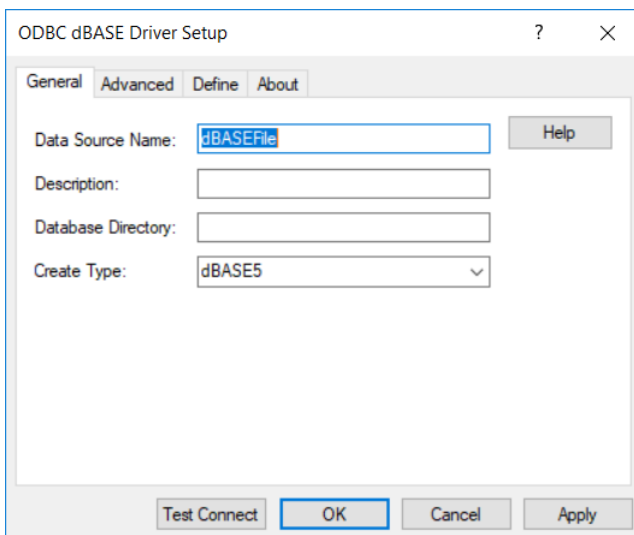
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 75: General tab



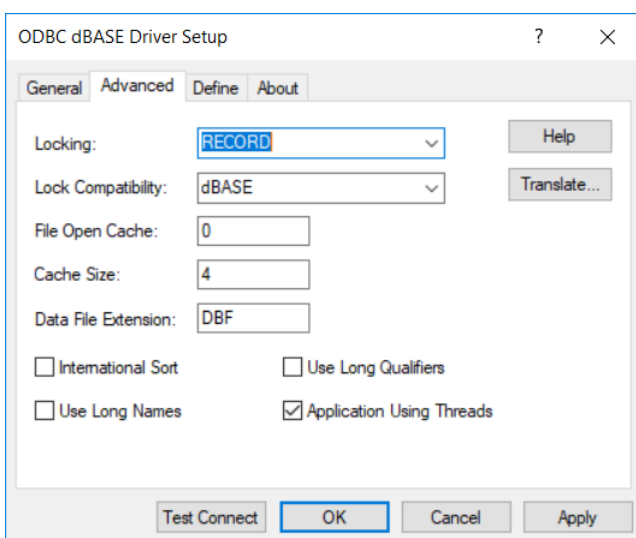
Note: The General tab displays the only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

3. On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 747	None
Description on page 749	None
Database Directory on page 748	None
Create Type [dBASE] on page 746	dBASE5

4. Optionally, click the **Advanced** tab to specify data source settings.

Figure 76: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Locking on page 753	RECORD
Lock Compatibility on page 752	dBASE
File Open Cache on page 750	0
Cache Size on page 745	4
Data File Extension on page 747	DBF
International Sort on page 751	Disabled
Use Long Names on page 753	Disabled
Use Long Qualifiers on page 754	Disabled

Connection Options: Advanced	Default
Application Using Threads on page 745	Enabled
IANAAppCodePage on page 751 UNIX ONLY	4 (ISO 8559-1 Latin-1)



Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

5. If you use index files that have different names than their corresponding data files and you have not defined this association, click the **Define** tab. See [Defining Index Attributes on Windows](#) on page 754 for step-by-step instructions.
6. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection properties specified in the driver Setup dialog box.
 - If the driver can connect, it releases the connection and displays a `connection established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.
7. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Data Source Configuration through a GUI (FoxPro)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.



UNIX[®] On UNIX and Linux, data sources are stored in the `odbc.ini` file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a FoxPro 3.0 database container data source :

1. Start the ODBC Administrator:

-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
-  On Linux, change to the *install_dir/tools* directory and, at a command prompt, enter:



```
odbcadmin
```

 where *install_dir* is the path to the product installation directory.

2. Select a tab:

- User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

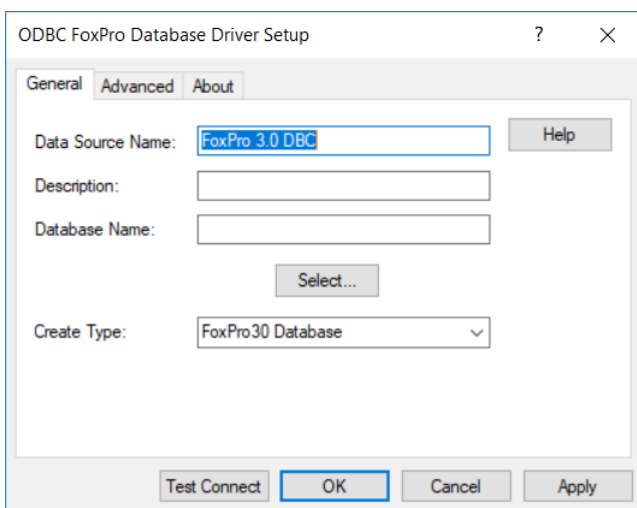
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 77: General tab



Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

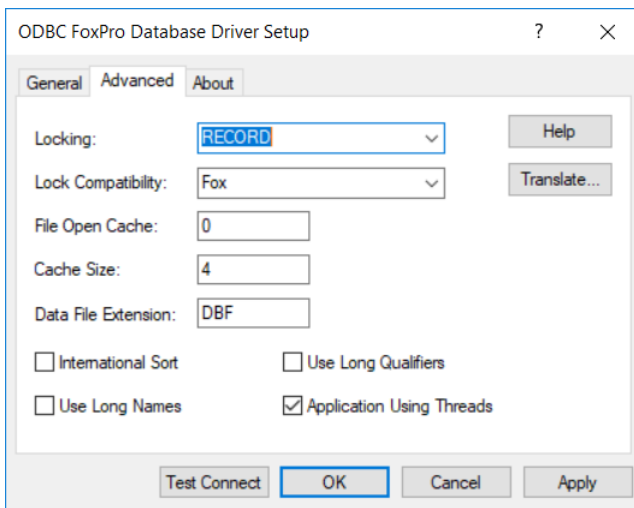
- On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 747	None
Description on page 749	None
Database Name on page 748	None
Create Type [FoxPro] on page 746	FoxPro30 Database

Click **Select** to choose the directory and .DBC file that you want to use.

- Optionally, click the **Advanced** tab to specify data source settings.

Figure 78: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Locking on page 753	RECORD
Lock Compatibility on page 752	Fox
File Open Cache on page 750	0
Cache Size on page 745	4
Data File Extension on page 747	DBF
International Sort on page 751	Disabled
Use Long Names on page 753	Disabled
Use Long Qualifiers on page 754	Disabled

Connection Options: Advanced	Default
Application Using Threads on page 745	Enabled
IANAAppCodePage on page 751 UNIX ONLY	4 (ISO 8559-1 Latin-1)



Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection properties specified in the driver Setup dialog box.
 - If the driver can connect, it releases the connection and displays a `connection established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.
- Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify `attribute=value` pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{driver_name}][;attribute=value[;attribute=value]...]
```

The following table lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for dBASE is:

```
DSN=DBASE FILES;LCK=NONE;IS=0
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=DBASE.dsn;LCK=NONE;IS=0
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 dBASEFile (*.dbf)};DB=C:\DBASE;CT=dBASE5
```

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the dBASE driver.

Table 53: dBASE Attribute Names

Attribute (Short Name)	Default
ApplicationUsingThreads (AUT)	1 (Enabled)
CacheSize (CSZ)	4
CreateType (CT) [dBASE]	dBASE5
CreateType (CT)) [FoxPro]	FoxPro30 Database
Database (DB) [dBASE]	None
Database (DB) [FoxPro]	None
DataFileExtension (DFE)	DBF
DataSourceName (DSN)	None
Description (n/a)	None
ExtensionCase (EC)	UPPER
FileOpenCache (FOC)	0 (no file open caching)
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
IntlSort (IS)	0 (Disabled)
LockCompatibility (LCOMP)	dBASE

Attribute (Short Name)	Default
Locking (LCK)	RECORD
UseLongNames (ULN)	0 (Disabled)
UseLongQualifiers (ULQ)	0 (Disabled)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Default

1 (Enabled)

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

Cache Size

Attribute

CacheSize (CSZ)

Purpose

The number of 64 KB blocks the driver uses to cache database records. The larger the number of blocks, the better the performance.

Valid Values

0 | x

where:

x

is a positive integer that specifies the number of 64 KB blocks for caching.

Behavior

If set to 0, no records are cached.

If set to *x*, the specified number of 64 KB blocks are set aside for caching. The maximum number of blocks you can set depends on the system memory available. If the cache size is greater than 0, when browsing backwards, you are not able to see updates made by other users until you run the Select statement again.

Default

4

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

Create Type [dBASE]

Attribute

CreateType (CTS) [dBASE]

Purpose

The type of table or index to be created on a Create Table or Create Index statement.

Valid Values

dBASE4 | dBASE5 | Clipper | FoxPro25 | FoxPro30

Default

dBASE5

GUI tab

[General tab \[dBASE\]](#)

Create Type [FoxPro]

Attribute

CreateType (CT) [FoxPro]

Purpose

The type of table or index to be created on a Create Table or Create Index statement.

Valid Value

FoxPro30 Database

Default

FoxPro30 Database

GUI tab

[General tab \[FoxPro\]](#)

Data File Extension**Attribute**

DataFileExtension (DFE)

Purpose

A one- to three-character file name extension to use for data files.

Valid Values

ext

where:

ext

is the name of the one- to three-character file name extension.

Behavior

This value is used for all Create Table statements. Sending a Create Table using an extension other than the value specified for this option causes an error.

In other SQL statements, such as Select or Insert, users can specify an extension other than the one specified for this connection option. The Data File Extension value is used when no extension is specified.

The file extension cannot be one the driver already uses, such as MDX or CDX.

Default

DBF

GUI tab

[General tab \[dBASE\]](#)

[General tab \[FoxPro\]](#)

Data Source Name**Attribute**

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab \[dBASE\]](#)

[General tab \[FoxPro\]](#)

Database Directory

Attribute

Database (DB) [dBASE]

Purpose

The directory that contains the data files.

Valid Values

database_directory

where:

database_directory

is the full path name of the directory in which the data files are stored. If no directory is specified, the current working directory is used.

Default

None

GUI Tab

[General tab \[dBASE\]](#)

Database Name

Attribute

Database (DB) [FoxPro]

Purpose

The directory that contains the database container (.DBC) files.

Valid Values

database_directory

where:

database_directory

is the full path name of the directory and .DBC file that you want to use.

Default

None

GUI Tab

[General tab \[FoxPro\]](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab \[dBASE\]](#)

[General tab \[FoxPro\]](#)

Extension Case

Attribute

ExtensionCase (EC)

Purpose

This option determines whether uppercase or lowercase file extensions are accepted.

Valid Values

LOWER | UPPER

Behavior

When set to UPPER, uppercase extensions are accepted.

When set to LOWER, lowercase extensions are accepted.

Default

UPPER

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

File Open Cache

Attribute

FileOpenCache (FOC)

Purpose

The maximum number of used file handles to cache.

Valid Values

0 | x

where:

x

is a positive integer.

Behavior

If set to 0, no file open caching is performed.

If set to x , when a user opens and closes x tables, the tables are not actually closed. The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive. The advantage of file open caching is improved performance. The disadvantage is that a user who tries to open the file exclusively may get a file locking conflict even though no one appears to have the file open.

Default

0 (No File Open Caching)

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The Driver Manager checks for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

International Sort

Attribute

IntlSort (IS)

Purpose

Uses international sort order as defined by your operating system when you issue a Select statement with an Order By clause.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), this order is always alphabetic, regardless of case; the letters are sorted as "A, b, C." Refer to your operating system documentation concerning the sorting of accented characters.

If set to 0 (Disabled), ASCII sort order is used. This order sorts items alphabetically with uppercase letters preceding lowercase letters. For example, "A, b, C" is sorted as "A, C, b."

Default

0 (Disabled)

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

Lock Compatibility

Attribute

LockCompatibility (LCOMP)

Purpose

The locking scheme the driver uses when locking records.

Valid Values

Clipper | dBASE | Fox | Q+E | Q+EVirtual

- Clipper specifies Clipper-compatible locking.
- dBASE specifies Borland-compatible locking.
- Fox specifies FoxPro-compatible locking.
- Q+E specifies that locks be placed on the actual bytes occupied by the record. Only applications that use the dBASE driver can read and write to the database. Other applications are locked out of the table completely (they cannot even read other records). This locking is compatible with earlier versions of Q+E products.
- Q+EVirtual specifies that locks be placed on bytes beyond the physical end-of-file. Q+EVirtual is the same as Q+E except that other applications can open the table and read the data.

The advantage of using a Q+E locking scheme over dBASE locking is that, on Inserts and Updates, Q+E locks only individual index tags, while dBASE locks the entire index. The following values determine locking support as described:

If you are accessing a table with an application that uses the dBASE driver, your locking scheme does not have to match the Create Type. If you access a table with two applications, however, and only one uses the dBASE driver, set your locking scheme to match the other application. For example, you do not have to set this value to Fox to work with a FoxPro table. But if you are using a FoxPro application simultaneously with an application using the dBASE driver on the same set of tables, set this value to Fox to ensure that your data does not become corrupted.

Default

dBASE

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

Locking

Attribute

Locking (LCK)

Purpose

The level of locking for the database file.

Valid Values

NONE | RECORD | FILE

- NONE offers the best performance, but is intended only for single-user environments. See [Locking](#) on page 761 for details.
- RECORD locks only the records affected by the statement.
- FILE locks all of the records in the table.

Default

RECORD

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

Use Long Names

Attribute

UseLongNames (ULN)

Purpose

Specifies whether to use long file names as table names.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses long file names as table names. The maximum table name length is specific to the environment in which you are running.

If set to 0 (Disabled), the driver does not long file names as table names.

Default

0 (Disabled)

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

Use Long Qualifiers

Attribute

UseLongQualifiers (ULQ)

Purpose

Determines whether the driver uses long path names.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), path names can be a maximum of 255 characters.

If set to 0 (Disabled), path names can be a maximum of 128 characters.

Default

0 (Disabled)

GUI tab

[Advanced tab \[dBASE\]](#)

[Advanced tab \[FoxPro\]](#)

Defining Index Attributes on Windows

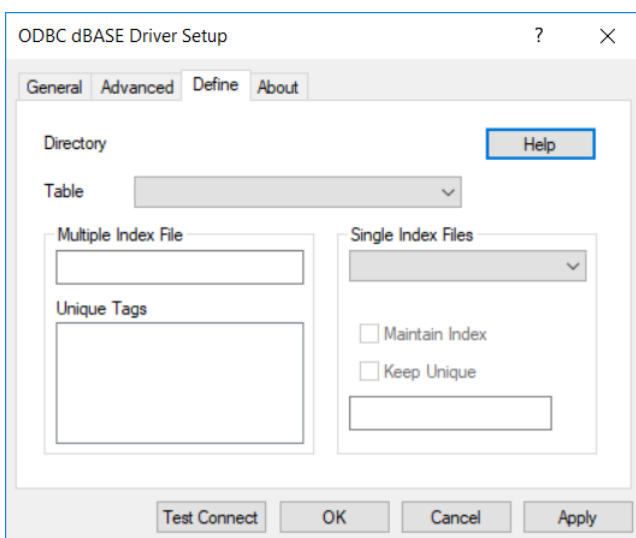


The Define tab of the ODBC dBASE Driver Setup dialog box allows you to define the attributes of index files. With the exception of Clipper, the family of databases that includes dBASE and FoxPro uses a multiple index file associated with a particular table (database file). This index file has a .MDX or .CDX extension and is automatically maintained by the driver. Tags within this index can be marked as unique.

Clipper uses single index files that are not automatically associated with a particular table. You can choose to have the driver maintain an index and choose whether or not the index is unique.

To define index file attributes:

1. Display the ODBC dBASE Driver Setup dialog box.
2. Click the **Define** tab.



On this tab, provide the following information; then, click **Apply**.

Table: Type or select the name of the table that contains the database information.

Multiple Index File: This field displays the name of any multiple index file (with a .CDX extension or .MDX extension) associated with the table you selected. This index file cannot be marked as unique, but tags within it can be.

Unique Tags: This field displays tags associated with the multiple index file. To mark tags as unique, click each one; each one remains selected until you click it again.

Single Index Files: The Single Index Files group is active only if you have selected a Clipper table.

Select the file from the drop-down list to define the attributes of a single index file.

Maintain Index: Select this check box to associate the specified single index file with the selected table.

Keep Unique: Select this check box to specify that the single index file is unique.

3. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Defining Index Attributes on UNIX and Linux

UNIX[®] Index files for dBASE contain index tags for each index that exists for a database file. These index tags can be marked as unique, that is, the driver will ensure that no duplicate values exist for the columns that define the index tag. The unique attribute is not natively supported by the dBASE or FoxPro products. The enforcement and recognition of the unique attribute is an extension of the dBASE driver. The driver must be notified that index tags are unique. No configuration is needed for unique indexes that were created using the DataDirect Connect for ODBC dBASE driver. When using files that were not created with the dBASE driver, you must define unique index tags as outlined in the following procedure.

In the directory where the database and index files are located, use any text editor, such as vi, to define or edit the QEDBF.INI as follows:

1. Create a [*filename*] section where *filename* is the name of the database file. This entry is case-sensitive and the file extension must be included, for example, [*accts.dbf*].
2. In the [*filename*] section, specify the number of unique indexes on the file (NUMUNIQUE=) and the index specifications (UNIQUE#=*index_filename,index_tag*). The *index_tag* can be determined by calling the ODBC function SQLStatistics and examining the INDEX_NAME result column.

For example, to define two unique indexes on the accts.dbf database file, the QEDBF.INI would be defined as:

```
[accts.dbf]
NUMUNIQUE=2
UNIQUE0=accts.mdx,ACCT_NAME
UNIQUE1=accts.mdx,ACCT_ID
```

Data Types

The following table shows how dBASE data types map to the standard ODBC data types. These dBASE data types can be used in a [Create Table](#) statement.

The following table shows how the additional FoxPro 3.0 data types map to the ODBC data types.

NOTE: A few products can create dBASE files with numbers that do not conform to the precision and scale of the Number column. For example, these products can store 100000 in a column declared as NUMBER(5,2). When this occurs, the dBASE driver displays error 1244, *Unsupported decimal format*. To remedy this situation, multiply the nonconforming column by 1, which converts it to the Float data type. For example:

```
SELECT BADCOL * 1 FROM BADFILE
```

BADCOL * 1 is evaluated as an expression and is returned as a float value.

Table 54: dBASE Data Types

dBASE	ODBC
Binary ⁶³	SQL_LONGVARBINARY
Char ⁶⁴	SQL_CHAR
Date	SQL_TYPE_DATE
Float ⁶⁵	SQL_DECIMAL
General ⁶⁶	SQL_LONGVARBINARY
Logical	SQL_BIT
Memo	SQL_LONGVARCHAR
Numeric	SQL_DECIMAL

⁶³ dBASE V only.

⁶⁴ 254 characters maximum (1024 for Clipper).

⁶⁵ dBASE IV and V only.

⁶⁶ FoxPro and dBASE V only.

Table 55: Additional FoxPro 3.0 Data Types

FoxPro 3.0	ODBC
Character (binary)	SQL_CHAR
Currency	SQL_DOUBLE
Datetime	SQL_TYPE_TIMESTAMP
Double	SQL_DOUBLE
Integer	SQL_INTEGER
Memo (binary)	SQL_LONGVARBINARY

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

Column Names

Column names in SQL statements (such as Select and Insert, for example) can be up to ten characters long. A column name can contain alphanumeric characters and the hyphen character (-). The first character must be a letter (a through z).

Select Statement

You use a SQL Select statement to specify the columns and records to be read. All of the Select statement clauses described in [SQL Statements for Flat-File Drivers](#) are supported by dBASE Select statements. This section describes the information that is specific to dBASE, which is Rowid.

Rowid Pseudo-Column

Each dBASE record contains a special column named Rowid. This field contains a unique number that indicates the record's sequence in the database. For example, a table that contains 50 records has Rowid values from 1 to 50 (if no records are marked deleted). You can use Rowid in Where and Select clauses.

Rowid is particularly useful when you are updating records. You can retrieve the Rowid of the records in the database along with the other field values. For example:

```
SELECT last_name, first_name, salary, rowid FROM emp
```

Then, you can use the Rowid of the record that you want to update to ensure that you are updating the correct record and no other. For example:

```
UPDATE emp set salary = 40000 FROM emp WHERE rowid=21
```

The fastest way of updating a single row is to use a Where clause with the Rowid. You cannot update the Rowid column.

Select statements that use the Rowid pseudo-column in the Where clause achieve maximum performance only for exact equality matches. If you use range scans instead of exact equality matches, a full table scan is performed. For example:

```
SELECT * FROM emp WHERE rowid=21    //fast search
SELECT * FROM emp WHERE rowid <=25  //full table scan
```

Alter Table Statement

The dBASE driver supports the Alter Table statement to add one or more columns to a table or to delete (drop) a single column.

The Alter Table statement has the form:

```
ALTER TABLE table_name {ADD column_namedata_type |
ADD(column_namedata_type [, column_namedata_type]... ) |
DROP[COLUMN] column_name}
```

table_name is the name of the table to which you are adding or dropping columns.

column_name assigns a name to the column you are adding or specifies the column you are dropping.

data_type specifies the native data type of each column you add.

For example, to add two columns to the emp table:

```
ALTER TABLE emp (ADD startdate date, dept char (10))
```

You cannot add columns and drop columns in a single statement, and you can drop only one column at a time. For example, to drop a column:

```
ALTER TABLE emp DROP startdate
```

The Alter Table statement fails if you attempt to drop a column upon which other objects, such as indexes or views, are dependent.

Create and Drop Index Statements

The dBASE driver supports SQL statements to create and delete indexes.

Create Index

The type of index you create is determined by the value of the CreateType attribute, which you set in the driver Setup dialog box (for UNIX and Linux, edit the system information file) or as a connection string attribute. The index can be:

- dBASE IV or V (.MDX)
- Clipper (.NTX)
- FoxPro (.CDX)

The syntax for creating an index is:

```
CREATE [UNIQUE] INDEX index_name ON base_table_name
(field_name [ASC | DESC] [,field_name [ASC | DESC]]...)
```

Unique means that the driver creates an ANSI-style unique index over the column and ensures uniqueness of the keys. Use of unique indexes improves performance. ANSI-style unique indexes are different from dBASE-style unique indexes. With ANSI-style unique indexes, you receive an error message when you try to insert a duplicate value into an indexed field. With dBASE-style unique indexes, you do not see an error message when you insert a duplicate value into an indexed field. This is because only one key is inserted in the index file.

index_name is the name of the index file. For FoxPro and dBASE IV or V, this is a tag, which is required to identify the indexes in an index file. Each index for a table must have a unique name.

base_table_name is the name of the database file whose index is to be created. The .DBF extension is not required; the driver automatically adds it if it is not present. By default, dBASE IV or V index files are named *base_table_name*.MDX and FoxPro indexes are named *base_table_name*.CDX.

field_name is a name of a column in the dBASE table. You can substitute a valid dBASE-style index expression for the list of field names.

ASC tells dBASE to create the index in ascending order. DESC tells dBASE to create the index in descending order. By default, indexes are created in ascending order. You cannot specify both ASC and DESC orders within a single Create Index statement. For example, the following statement is invalid:

```
CREATE INDEX emp_i ON emp (last_name ASC, emp_id DESC)
```

The following table shows the attributes of the different index files supported by the dBASE driver. For each type supported, it provides the following details:

- Whether dBASE-style unique indexes are supported
- Whether descending order is supported
- The maximum size supported for key columns
- The maximum size supported for the column specification in the Create Index statement
- Whether production/structural indexes are supported

Table 56: dBASE-Compatible Index Summary

Create Type.Extension	dBASE UNIQUE	DESC	Max Size of Key Column	Max Size of Column Specification	Production/Structural Indexes	Supports FOR Expressions
dBASE IV, V .MDX	Yes	Yes	100	220	Yes	Yes
Clipper .NTX	Yes	Yes	250	255	No	Yes
FoxPro .IDX ⁶⁷	Yes	Yes	240	255	No	Yes
FoxPro .CDX	Yes	Yes	240	255	Yes	Yes

Drop Index

The syntax for dropping an index is as follows:

```
DROP INDEX table_name.index_name
```

table_name is the name of the dBASE file without the extension.

⁶⁷ Compact IDX indexes have the same internal structure as a tag in a CDX file. These indexes can be created if the IDX extension is included with the index name in the Create Index statement.

For FoxPro and dBASE IV or V, *index_name* is the tag. Otherwise, *index_name* is the name of the index file without the extension.

To drop the index EMPHIRE.MDX, issue the following statement:

```
DROP INDEX emp.emphire
```

Pack Statement

When records are deleted from a dBASE file, they are not removed from the file. Instead, they are marked as having been deleted. Also, when memo fields are updated, space may be wasted in the files. To remove the deleted records and free the unused space from updated memo fields, you must use the Pack statement. It has the following form:

```
PACK filename
```

filename is the name of the dBASE file to be packed. The .DBF extension is not required; the driver automatically adds the extension if it is not present. For example:

```
PACK emp
```

You cannot pack a file that is opened by another user, and you cannot use the Pack statement in manual commit mode.

For the specified file, the Pack statement performs the following actions:

- Removes all deleted records from the file
- Compresses unused space in the memo file (.DBT or .FPT)
- Removes the entries for all deleted records from .CDX and .MDX files having the same name as the file

SQL Statements for FoxPro 3.0 Database Containers

The FoxPro DBC driver supports four additional SQL statements:

- Create Database
- Add Table
- Remove Table
- Use

To create a new FoxPro 3.0 database container, use:

```
CREATE DATABASE database_name
```

To add an existing table to the database container, use:

```
ADD TABLE table_name
```

To remove a table from the database container (not delete the table, but unlink it from the database container), use:

```
REMOVE TABLE table_name
```


To set the current database container to an existing database container, use:

```
USE database_name
```

To add or delete columns from a table in a database container, use the Alter Table statement (see [Alter Table Statement](#) on page 758).

Locking

With the dBASE driver, you can build and run applications that share dBASE database files on a network. Whenever more than one user is running an application that accesses a shared database file, the applications should lock the records that are being changed. Locking a record prevents other users from locking, updating, or deleting the record.

Levels of Database Locking

The dBASE driver supports three levels of database locking: NONE, RECORD, and FILE. You can set these levels in:

- The connection string (LCK=)
- The Setup dialog box

No locking offers the best performance, but is intended only for single-user environments.

With record or file locking, the system locks the database files during Insert, Update, Delete, or Select...For Update statements. The locks are released when the user commits the transaction. The locks prevent other users from modifying the locked objects, but they do not lock out readers.

With record locking, only records affected by the statement are locked. Record locking provides better concurrency with other users who also want to modify the database file.

With file locking, all the records in the database file are locked. File locking has lower overhead and may work better if records are modified infrequently, if records are modified primarily by one user, or if a large number of records are modified.

Limit on Number of Locks

There is a limit on the number of locks that can be placed on a file. If you are accessing a dBASE file from a server, the limit depends on the server (refer to your server documentation).

If you are accessing a dBASE file locally, the limit depends on the buffer space allocated when SHARE.EXE was loaded (refer to your DOS documentation). If you are exceeding the number of locks available, you may want to switch to file locking.

How Transactions Affect Record Locks

When an Update or Delete statement is run, the driver locks the records affected by that statement. The locks are released after the driver commits the changes. Under manual commit mode, the locks are held until the application commits the transaction. Under autocommit mode, the locks are held until the statement is run.

When a Select...For Update statement is run, the driver locks a record only when the record is fetched. If the record is updated, the driver holds the lock until the changes are committed. Otherwise, the lock is released when the next record is fetched.

Isolation and Lock Levels Supported

dBASE supports isolation level 1 (read committed). It supports both file-level and record-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the minimum SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions. In addition, the SQLSetPos function is supported.

The driver also supports backward and random fetching in SQLExtendedFetch and SQLFetchScroll.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

dBASE supports multiple connections and multiple statements per connection.

The Informix Driver

The DataDirect Connect for ODBC Informix driver (the Informix driver) supports multiple connections to the Informix Dynamic Server when using the appropriate client software.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The Informix driver is 32-bit only and is supported in the Windows and UNIX environments, but not in the Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect product for the file name of the Informix driver.

Note: The Informix driver requires Informix client software. Progress DataDirect also provides an Informix driver that does not require any client software to access Informix databases. See [The Informix Wire Protocol Driver](#) on page 220 for details.

Driver Requirements

This section provides the system requirements for using the Informix driver on all supported platforms.

Windows

To access supported remote Informix databases through the Informix driver, you need one of the following:

- Informix Connect for Windows platforms, version 2.x
- Informix Client Software Development Kit for Windows platforms, version 2.x

Use the Setnet32 utility supplied by Informix to define servers and the location of the INFORMIX directory. Use Ilogin to test your connection to the Informix server. The path to the ISQLT09A.DLL must be in your PATH environment variable.

UNIX (AIX, HP-UX PA-RISC, and Solaris)

The environment variable INFORMIXDIR must be set to the directory where you have installed the Informix client.

For example, the following syntax is valid for C-shell users:

```
setenv INFORMIXDIR /databases/informix
```

For Bourne- or Korn-shell users, the following syntax is valid:

```
INFORMIXDIR=/databases/informix;export INFORMIXDIR
```

In addition, the INFORMIXSERVER variable must be set to the name of the Informix server (as defined in your \$INFORMIXDIR/etc/sqlhosts file). For further details, refer to the Informix documentation.

To access supported remote Informix databases through the Informix driver, you need one of the following:

- On AIX: Informix Client Software Development Kit version 2.2 or higher; or Informix Connect version 2.2 or higher
- On HP-UX and Solaris: Informix Connect version 2.x
- On HP-UX and Solaris: Informix Client Software Development Kit version 2.x

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 768 and [Connection Option Descriptions](#) on page 769 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

UNIX® On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, `odbc.ini`). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the `odbc.ini` file and storing default connection values there. See [Configuration Through the Administrator](#) on page 115 and [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions](#) on page 769 lists driver connection string attributes that must be used in the `odbc.ini` file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Informix Client)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX® On UNIX and Linux, data sources are stored in the `odbc.ini` file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure an Informix data source:

1. Start the ODBC Administrator:

-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- **UNIX**® On Linux, change to the `install_dir/tools` directory and, at a command prompt, enter:


```
odbcadmin
```

where `install_dir` is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- 
System DSN: If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

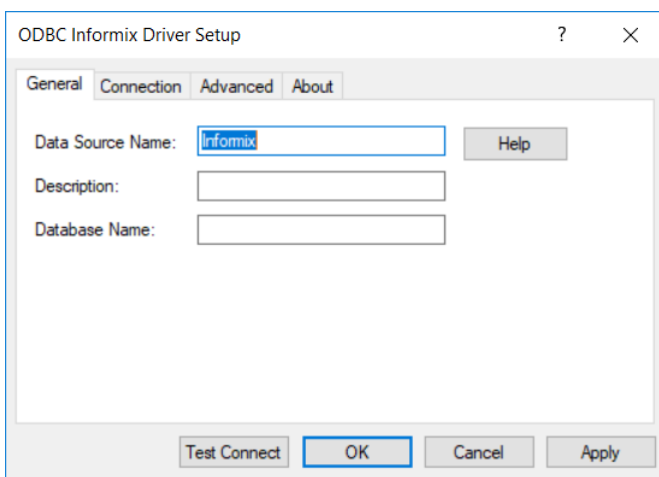
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the ODBC Informix Driver Setup dialog box appears by default.

Figure 79: General tab



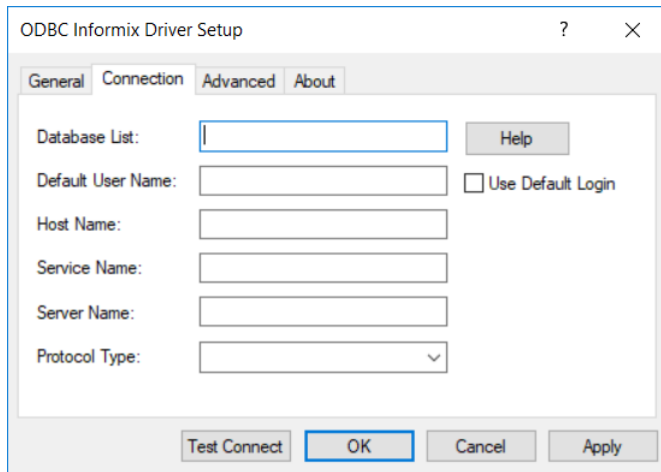
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 773	None
Description on page 774	None
Database Name on page 773	None

- Optionally, click the **Connection** tab to specify connection information. If you want to configure the data source so that the logon dialog box does not appear during connection, you must specify the connection information on this tab.

Figure 80: Connection tab

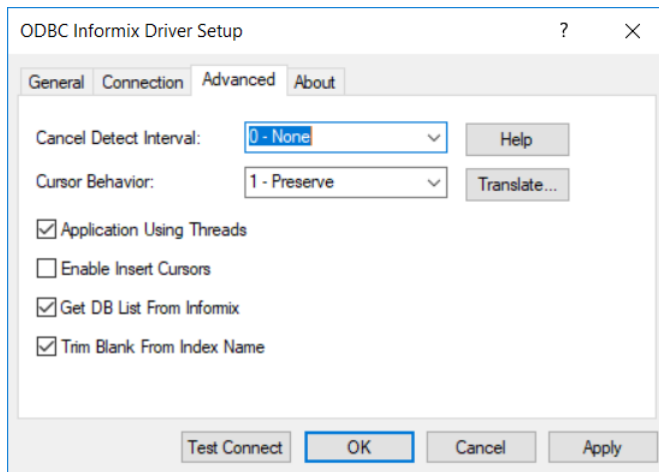


On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Connection	Default
Database List on page 772	None
Default User Name on page 774	None
Use Default Login on page 780 WINDOWS ONLY	Disabled
Host Name on page 776	None
Service Name on page 779 WINDOWS ONLY	None
Server Name on page 778	None
Protocol Type on page 778 WINDOWS ONLY	None

5. Optionally, click the **Advanced** tab to specify data source settings.

Figure 81: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Cancel Detect Interval on page 771	0 - None
Cursor Behavior on page 772	0 - Close
Application Using Threads on page 770	Enabled
Enable Insert Cursors on page 775	Disabled
Get DB List From Informix (GDBLFI) on page 775	Enabled
Trim Blank From Index Name on page 779	Enabled
IANAAppCodePage on page 776 UNIX ONLY	4 (ISO 8559-1 Latin 1)



Translate : Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

6. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection properties specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Informix Client\)](#) on page 768 for details). The information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a "connection established" message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message.

Verify that all required client software is properly installed. If it is not, you will see the message:

```
Specified driver could not be loaded due to system error [xxx]
```

Click **OK**.

7. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={}{driver_name[]}[:attribute=value[:attribute=value]...]
```

[Connection Option Descriptions](#) on page 769 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Informix is:

```
DSN=INFORMIX TABLES;DB=PAYROLL
```

A FILEDSN connection string is similar except for the initial keyword:

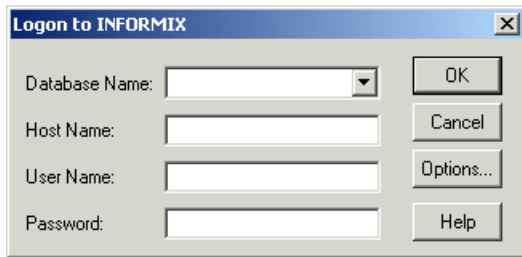
```
FILEDSN=Informix.dsn;DB=DBPAYROLL
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Informix};DB=PAYROLL;UID=JOHN;PWD=XYZZY
```

Using a Logon Dialog Box (Informix Client)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.




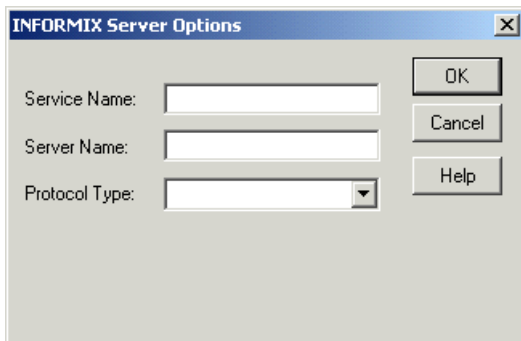
In this dialog box, provide the following information:

1. Type the name of the database you want to access, or, on Windows, select the name from the Database Name drop-down list.



On Windows, the names on the list are determined by the status of the **Get DB List From Informix** check box on the Advanced tab of the ODBC Informix driver Setup dialog box. If the check box is selected, the names displayed are returned from the Informix server. If cleared, the names displayed are returned from the user-entered list, which you specify in the Database List field on the Connection tab of the driver Setup dialog box.

2. Type the name of the host machine on which the Informix server resides.
3. If required, type your user name as specified on the Informix server.
4. If required, type your password.
5.  On Windows, click **Options** to display the Informix Server Options dialog box, where you can change the Service Name, Server Name, and Protocol Type that you specified in the ODBC Informix Driver Setup dialog box. Click **OK** to save your changes.



6. Click **OK** to complete the logon and to update these values in the Registry.

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Informix driver.

Table 57: Informix Attribute Names

Attribute (Short Name)	Default
ApplicationUsingThreads (AUT)	1 (Enabled)
CancelDetectInterval (CDI)	0 (None)
CursorBehavior (CB)	0 - Close
Database (DB)	None
Databases (DL)	None
DataSourceName (DSN)	None
Description (n/a)	None
EnableInsertCursors (EIC)	Disabled
HostName (HOST)	None
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin 1)
LogonID (UID)	None
Password (PWD)	None
Protocol (PRO)	None
ServerName (SRVR)	None
Service (SERV)	None
TrimBlankFromIndexName (TBFIN)	1 (Enabled)
UseDefaultLogin (UDL)	0 (Disabled)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 780 for details.

Cancel Detect Interval

Attribute

CancelDetectInterval (CDI)

Purpose

Determines whether long-running queries in threaded applications can be cancelled if the application issues a SQLCancel.

Valid Values

0 | x

where:

x

is the number of seconds the driver waits before checking for SQLCancel calls.

Behavior

If set to 0 (None), the driver does not allow long-running queries in threaded applications to be canceled, even if the application issues a SQLCancel.

If set to x (seconds), for every pending query, the driver checks for SQLCancel calls at the specified interval. If the driver determines that a SQLCancel has been issued, the driver cancels the query.

Notes

- This connection option can affect performance.

Example

If you specify 5, for every pending query, the driver checks every five seconds to see whether the application has issued a SQLCancel call. If it detects a SQLCancel call, the driver cancels the query.

Default

0 (None)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 780 for details.

Cursor Behavior

Attribute

CursorBehavior (CB)

Purpose

Determines whether cursors will be preserved or closed at the end of transactions.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), cursors are held at their current position when transactions end. This value may slow the performance of your database operations.

If set to set to 0 (Disabled), cursors are closed at the end of transactions.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Database List

Attribute

Databases (DL)

Purpose

A list of database names that will be displayed in the Logon dialog box if **Get DB List From Informix** on the Advanced tab is *not* selected.

Valid Values

```
database_name [ , database_name ] [ ... ]
```

where:

database_name

is a database name you want to appear in the Logon dialog box. Separate multiple values with commas.

Example

db1, db2, db3

Default

None

GUI Tab

[Connection tab](#)

Database Name

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where:

database_name

is the name of a valid database.

Default

None

GUI Tab

[General tab](#)

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Default User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Connection tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable Insert Cursors

Attribute

EnableInsertCursors (EIC)

Purpose

Determines whether the driver can use Insert cursors during inserts governed by parameters.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses Insert cursors.

If set to 0 (Disabled), the driver does not use Insert cursors.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Get DB List From Informix (GDBLFI)

Attribute

GetDBListFromInformix (GDBLFI)

Purpose

Determines whether the driver requests the database list to be returned from the Informix server or from the database list that the user entered at driver setup.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver requests the database list from the Informix server.

If set to 0 (Disabled), the driver uses the list that was entered by the user at driver setup.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Host Name

Attribute

HostName (HOST)

Purpose

The name of the server to which you want to connect.

Valid Values

server_name

where:

server_name

is the name of the server to which you want to connect.

Default

None

GUI Tab

[General tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The Driver Manager checks for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Password

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Protocol Type

Attribute

Protocol (PRO)

Purpose

Determines the protocol used by the driver to communicate with the server.

Valid Values

olsocspix | olsoctcp | onsocspix | onsoctcp | seipcpx | sesocspix | sesoctcp

Specify the appropriate Informix protocol.

Default

None

GUI Tab

[Connection tab](#)

Server Name

Attribute

ServerName (SRVR)

Purpose

The name of the Informix server.

Valid Values

server_name

where:

server_name

is a name that uniquely identifies the Informix server.

Default

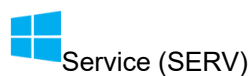
None

GUI Tab

[Connection tab](#)

Service Name

Attribute



Purpose

The name of the Informix service. The service name is assigned by the system administrator.

Valid Values

service_name

where:

service_name

is the a name that uniquely identifies the Informix service. This name must be specified as it appears in the services file on the server machine.

Default

None

GUI Tab

[Connection tab](#)

Trim Blank From Index Name

Attribute

TrimBlankFromIndexName (TBFIN)

Purpose

Determines whether the driver trims leading spaces from system-generated index names. Some applications cannot process a leading space in index names.

Valid Values

0|1

Behavior

If set to 1 (Enabled), the driver trims leading spaces from system-generated index names.

If set to 0 (Disabled), the driver does not trim leading spaces from system-generated index names.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Use Default Login

Attribute

UseDefaultLogin (UDL)

Purpose

Determines where the driver reads login credentials.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), login credentials are read from the Windows Registry, the connection string, or the Logon to Informix dialog box.

If set to 1 (Enabled), login credentials are read directly from the Informix registry.

Default

0 (Disabled)

GUI Tab

[Connection tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Cancel Detect Interval (CancelDetectInterval): If your application uses threads, it may allow canceling of long running queries (may issue synchronous SQLCancel calls). If your application does not issue synchronous SQLCancel calls, the driver can improve performance if Cancel Detect Interval is disabled (set to 0). In this case, the driver does not incur the overhead of periodically checking for SQLCancel. In the case where your application does issue synchronous SQLCancel calls, this option should be set to a value that specifies how often the driver checks to see if a long running query has been canceled.

Data Types

The following table shows how the Informix data types map to the standard ODBC data types.

Table 58: Informix Data Types

Informix	ODBC
BLOB	SQL_LONGVARBINARY
BOOLEAN	SQL_BIT
BYTE ⁶⁸	SQL_LONGVARBINARY
CHAR	SQL_CHAR
CLOB	SQL_LONGVARCHAR
DATE	SQL_TYPE_DATE
DATETIME YEAR TO FRACTION(f) ⁶⁹	SQL_TYPE_TIMESTAMP
DATETIME YEAR TO SECOND	SQL_TYPE_TIMESTAMP
DATETIME YEAR TO DAY	SQL_TYPE_DATE
DATETIME HOUR TO SECOND	SQL_TYPE_TIME
DATETIME HOUR TO FRACTION(f) ⁶⁹	SQL_TYPE_TIME
DECIMAL	SQL_DECIMAL
FLOAT	SQL_DOUBLE
INT8	SQL_BIGINT
INTEGER	SQL_INTEGER
INTERVAL YEAR(p) TO YEAR	SQL_INTERVAL_YEAR
INTERVAL YEAR(p) TO MONTH	SQL_INTERVAL_YEAR_TO_MONTH
INTERVAL MONTH(p) TO MONTH	SQL_INTERVAL_MONTH
INTERVAL DAY(p) TO DAY	SQL_INTERVAL_DAY
INTERVAL DAY(p) TO HOUR	SQL_INTERVAL_DAY_TO_HOUR
INTERVAL DAY(p) TO MINUTE	SQL_INTERVAL_DAY_TO_MINUTE

⁶⁸ Not supported for Standard Engine databases.

⁶⁹ (f) can have a value of 1, 2, 3, 4, or 5. The precision is type-dependent and the scale is 5.

Informix	ODBC
INTERVAL DAY(p) TO SECOND	SQL_INTERVAL_DAY_TO_SECOND
INTERVAL DAY(p) TO FRACTION(f) ⁶⁹	SQL_INTERVAL_DAY_TO_SECOND
INTERVAL HOUR(p) TO HOUR	SQL_INTERVAL_HOUR
INTERVAL HOUR(p) TO MINUTE	SQL_INTERVAL_HOUR_TO_MINUTE
INTERVAL HOUR(p) TO SECOND	SQL_INTERVAL_HOUR_TO_SECOND
INTERVAL HOUR(p) TO FRACTION(f) ⁶⁹	SQL_INTERVAL_HOUR_TO_SECOND
INTERVAL MINUTE(p) TO MINUTE	SQL_INTERVAL_MINUTE
INTERVAL MINUTE(p) TO SECOND	SQL_INTERVAL_MINUTE_TO_SECOND
INTERVAL MINUTE(p) TO FRACTION(f) ⁶⁹	SQL_INTERVAL_MINUTE_TO_SECOND
INTERVAL SECOND(p) TO SECOND	SQL_INTERVAL_SECOND
INTERVAL SECOND(p) TO FRACTION(f) ⁶⁹	SQL_INTERVAL_SECOND
INTERVAL FRACTION TO FRACTION(f)2	SQL_INTERVAL_SECOND
LVARCHAR(p) ⁶⁹	SQL_VARCHAR
MONEY	SQL_DECIMAL
NCHAR	SQL_CHAR
NVARCHAR	SQL_VARCHAR
SERIAL	SQL_INTEGER
SERIAL8	SQL_BIGINT
SMALLFLOAT	SQL_REAL
SMALLINT	SQL_SMALLINT
TEXT ⁶⁸	SQL_LONGVARCHAR
VARCHAR ⁶⁸	SQL_VARCHAR

The Informix driver does not support any complex data types (for example, set, multiset, list, and named/unnamed abstract types). When the driver encounters a complex type it will return an Unknown Data Type error (SQL State HY000).

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

MTS Support



On Windows, the driver can take advantage of Microsoft Transaction Server (MTS) capabilities, specifically, the Distributed Transaction Coordinator (DTC) using the XA Protocol. For a general discussion of MTS and DTC, refer to the help file of the Microsoft Transaction Server SDK.

Note: The DataDirect Connect for ODBC 32-bit drivers can operate in a 64-bit Windows environment; however, they do not support DTC in this environment. Only the DataDirect Connect64 for ODBC 64-bit drivers support DTC in a 64-bit Windows environment.

To enable DTC support, you must be using Informix Connect version 2.20 or higher clients.

To enable support for the DTC:

1. Use the **Setnet32** utility supplied by Informix to define:
 - The INFORMIXDIR environment variable, which identifies the location of the client programs, library files, message files, header files, and other Informix software components
 - The INFORMIXSERVER environment variable, which identifies the default database server
 - An Informix server, which identifies either an existing Informix database server or a new one
 - A host name, which identifies the host computer with the database server you want to use
 - A user name, which identifies a user name for an account on the currently selected host computer
 - A password for the specified user name, if required

When enlisting in a distributed transaction, the Informix clients only use the defaults specified in **Setnet32**.

2. Run the **regcopy** utility provided with INFORMIX-Connect to copy the registry entries created by **Setnet32** to an area in the registry that is accessible by the DTC. The DTC is a service, and services do not search for configuration information in the Windows registry where **Setnet32** stores client products environment variables. Therefore, if you do not run **regcopy** after setting the defaults in **Setnet32**, enlistment in a distributed transaction will fail.

For information on using the **Setnet32** and **regcopy** utilities, see the Informix documentation.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

If connected to an Online Server, Informix supports isolation levels 0 (read uncommitted), 1 (read committed), and 3 (serializable). The default is 1. The Standard Engine supports isolation level 0 (read uncommitted) only.

Informix also supports an alternative isolation level 1, called "cursor stability." Your ODBC application can use this isolation level by calling `SQLSetConnectAttr (1040,1)`.

Additionally, if transaction logging has not been enabled for your database, then transactions are not supported by the driver (the driver is always in auto-commit mode).

Informix supports page-level and row-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the core SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

In addition, the following functions are supported:

- SQLProcedures
- SQLColumnPrivileges
- SQLTablePrivileges
- SQLPrimaryKeys
- SQLForeignKeys
- SQLProcedureColumns

The driver also supports scrollable cursors with SQLFetchScroll or SQLExtendedFetch.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The Informix driver supports multiple connections and multiple statements per connection to the Informix database system.

The XML Driver

The DataDirect Connect for ODBC XML driver (the XML driver) supports:

Tabular- and hierarchical-formatted XML documents that can be accessed from either a local file system, a web server, or a web service. The three main types of tabular-formatted files that the driver supports are Microsoft Data Islands, ADO 2.5 persisted files, and DataDirect Format.

See [Supported Tabular Formats for XML Documents](#) on page 785 for more details.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The XML driver is 32-bit only and is supported in the Windows environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

The XML driver includes a SQL Engine that provides ANSI SQL-92 support. The following table lists the SQL statements that the driver supports for the different types of file formats.

File Format	Select	Create/Drop	Insert	Update	Delete
Tabular, Microsoft Data Islands	X	X	X	X	X
Tabular, ADO 2.5 Persisted	X	X	X	X	X
Tabular, DataDirect	X	X	X	X	X
Tabular, other formats	X		X	X	X
Hierarchical	X				

See [SQL Support](#) on page 817 for more information.

Refer to the readme file shipped with your DataDirect Connect product for the file name of the XML driver.

Driver Requirements

You must have Internet Explorer 5 or higher installed. You must also have the Microsoft XML parser, msxml4.dll, not a higher version, installed. If you need to download the file, go to the site:

<http://www.microsoft.com>

On the Microsoft site, search on "msxml4.dll". Select the link for downloading the parser.

Supported Tabular Formats for XML Documents

The three main XML tabular-formats that the XML driver can access are described in the following table. In some instances, you may need to define hints to help the XML driver read the tabular-format of an XML document correctly. See [Configure Location Dialog Box Descriptions](#) on page 804.

Table 59: Common Tabular Formats for XML Documents

Format	Description
ADO 2.5 persisted files	<p>These files are identified by a unique schema namespace URL. Although ADO uses the same data types defined by XML-Data, the data types use extensions, such as adding a maximum column width for string columns. ADO 2.5 persisted files are identified by the following unique XML element:</p> <pre><xml xmlns:s="uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882" xmlns:dt="uuid:C2F41010-65B3-11d1-A29F- 00AA00C14882" xmlns:rs="urn:schemas-microsoft-com:rowset" xmlns:z="#RowsetSchema"></pre>

Format	Description
DataDirect Format	<p>This XML format conforms to the W3C recommendation for XML schema, Working Draft April 07, 2000. These files are identified by the following unique XML element (schema namespace URL):</p> <pre data-bbox="446 359 1421 636"><table targetNamespace= "http://www.merant.com/namespaces/datadirect/xmlrecordset" xsi:schemaLocation= "<http://www.merant.com/namespaces/datadirect/xmlrecordset/EMP.xml"> xmlns="http://www.w3.org/1999/XMLSchema" xmlns:xsi= "http://www.w3.org/1999/XMLSchema-instance" xmlns:rs= "http://www.merant.com/namespaces/datadirect/xmlrecordset"><table xmlns="http://www.w3.org/1999/XMLSchema" xmlns:xsi= "http://www.w3.org/1999/XMLSchema-instance" xmlns:rs= "http://www.datadirect-technologies.com/namespaces /datadirect/xmlrecordset"></pre>
Microsoft Data Islands	<p>These islands are identified by the <XML> tag in an HTML document. The Data Island can be embedded in the HTML document. Data Islands can include the following Schema definition and namespace:</p> <pre data-bbox="446 816 1161 863"><Schema xmlns="urn:schemas-microsoft-com:xml-data" xmlns:dt="urn:schemas-microsoft-com:datatypes"></pre>

Hierarchical-Formatted XML Document Support

The XML driver can be configured so that it supports hierarchical-formatted documents. In this case, the driver assumes that the document that it is accessing can contain more than one table. The driver scans the document to locate all tables; the available tables are visible through a SQLTables operation. Then, the driver does a second scan to gather each table's column information and to determine a data type for each column.

The following is an example of a hierarchical document:

```
<?xml version="1.0"?>
<purchaseOrder orderDate="1999-10-20">
  <shipTo country="US">
    <name>Alice Smith</name>
    <street>123 Maple Street</street>
    <city>Mill Valley</city>
    <state>CA</state>
    <zip>90952</zip>
  </shipTo>
  <billTo country="US">
    <name>Robert Smith</name>
    <street>8 Oak Avenue</street>
    <city>Old Town</city>
    <state>PA</state>
    <zip>95819</zip>
  </billTo>
  <comment>Hurry, my lawn is going wild!</comment>
  <items>
    <item partNum="872-AA">
      <productName>Lawnmower</productName>
      <quantity>1</quantity>
      <USPrice>148.95</USPrice>
      <comment>Confirm this is electric</comment>
    </item>
    <item partNum="926-AA">
      <productName>Baby Monitor</productName>
```

```

    <quantity>1</quantity>
    <USPrice>39.98</USPrice>
    <shipDate>1999-05-21</shipDate>
  </item>
</items>
</purchaseOrder>

```

First, the XML driver returns two tables: "purchaseOrder" and "items." Two tables are returned because two items are found for a single purchase order. The XML driver found commonality of child elements.

Second, the XML driver determines which columns are in a specific table. An `_ID` column, which is essentially a primary key, is automatically generated for each table. If a table is determined to be a child of another table, then it is given a second generated column. The name of this column is prefixed with the parent table's name and ends with `_ID`, for example, `_purchaseOrder_ID`.

Consider the previous example document. The items table will receive two generated columns, `_ID` and `_purchaseOrder_ID`, which are assigned an integer data type. The purchaseOrder table receives only the `_ID` column, because it does not have a parent table.

The tables returned from the example file include the following columns:

Table	Columns	
items	<code>_ID</code> <code>_purchaseOrder_ID</code> <code>partNum</code> <code>productName</code>	<code>quantity</code> <code>USPrice</code> <code>comment</code> <code>Date</code>
purchaseOrder	<code>_ID</code> <code>orderDate</code> <code>shipTo_country</code> <code>shipTo_name</code> <code>shipTo_street</code> <code>shipTo_city</code> <code>shipTo_state</code> <code>shipTo_zip</code>	<code>billTo_country</code> <code>billTo_name</code> <code>billTo_street</code> <code>billTo_city</code> <code>billTo_state</code> <code>billTo_zip</code> <code>comment</code>

Column Data Types

The XML driver determines the column data types by inspecting the column values. The data type determination limits its data types to a subset of the DataDirect Format data types, as listed in the following table. For a complete list of DataDirect Format data types, see [Supported Tabular Formats for XML Documents](#) on page 785.

Data Type	Sample Values
wvchar	"Foo", "best320"
varbinary	"27AB2F9C"
int	"34", "-7000"

unsignedint	"0", "123456789"
long	"-12345678012345"
unsignedlong	"123456789012345"
boolean	"true", "false"
date	1963-12-19
time	10:09:58
timeinstant	1963-12-19T10:09:58
decimal	1245.678

Defining Locations

When configuring an XML data source, you must define the location of the XML or HTML documents that the driver will access. The locations can be either from a local file system or from a Web server.

The following table describes the types of locations:

Folder	Implies that each XML file is a single table. When defining a Folder location, you specify only a directory as the location (not a directory and a file name), for example, C:\xmlsample.
XML Document	Implies that the full path to the XML document, including the XML file name, is the location. Using this type of location, each document can have one or more tables and can be a hierarchical-formatted XML document. When defining an XML Document location, you specify a path and an XML file name as the location, for example: C:\xmlsample\file.xml You can also specify a web service through a URL, for example: http://xxx.company.com/search=XML&mode=books
HTML Document	Implies the use of an HTML document with embedded XML Data Islands. Using this type of location, each document can have one or more tables. When defining an HTML Document location, you specify a path and an HTML file name, for example, C:\htmlsample\file.html, as the location.

Specifying Table Names in SQL Statements

When defining locations, you specify a name for the location along with a directory, or path and file name. For example, suppose you define two locations for a data source, a Folder location and an XML Document location. The Folder location is on a local filing system and the XML Document location is on a web server with a URL prefix of <http://www.acme.com/xmldata>.

For example:

The Folder location:

c:\xmldata\xmlsample as LOC1

The XML Document location: http://www.acme.com/xmldata/doc.xml as LOC2

For complete information about how to configure locations in an XML data source, see [Data Source Configuration through a GUI \(XML\)](#) on page 790.

If you are connected to this data source and the data source had the "Show Manufactured Schemas" option set as the Schema Mode (see the Schema Mode option under [Data Source Configuration through a GUI \(XML\)](#) on page 790) and then you performed an unqualified SQLTables operation, you would get the following results.

Schema name	Table name
LOC1#	FILE1
LOC1#	FILE2
LOC2#	TABLE1
LOC2#	TABLE2

Location names are fabricated into the schema name by adding a # symbol to the end of the location name.

Note: If you had the "Show Virtual Schemas" option set, the above table would have "XML" listed in the Schema name column.

To fully qualify a table name in a SQL statement, you could use the following:

LOC1#.FILE1

or

XML.FILE1

LOC2#.TABLE2

or

XML.TABLE2

This design gives you a simpler table name qualifier. This is an important advantage given the complexity of URL names, and the requirement to double quote them in SQL statements. For example, the following query uses a fully qualified table name for an XML Document location:

```
SELECT * FROM "http://www.acme.com/xmldata/doc.xml#TABLE2" WHERE productName='lawnmower'
```

Compare that to the same query using a location name:

```
SELECT * FROM LOC2#.TABLE2 WHERE productName='lawnmower'
```

Another example demonstrating the Folder location is as follows:

```
SELECT * FROM "c:\xmldata\xmlsample\FILE1.XML" WHERE productName='lawnmower'
```

Compare that to the same query using a location name:

```
SELECT * FROM LOC1#.FILE1 WHERE productName='lawnmower'
```

Configuring and Connecting to Data Sources (XML)

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box. See [Data Source Configuration through a GUI \(XML\)](#) on page 790 for details.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 796 and [Connection Option Descriptions](#) on page 797 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration through a GUI (XML)

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure an XML data source:

1. Start the ODBC Administrator by selecting its icon from the DataDirect Connect program group; then, select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name on the User DSN tab and click **Configure** to display the driver Setup dialog box.

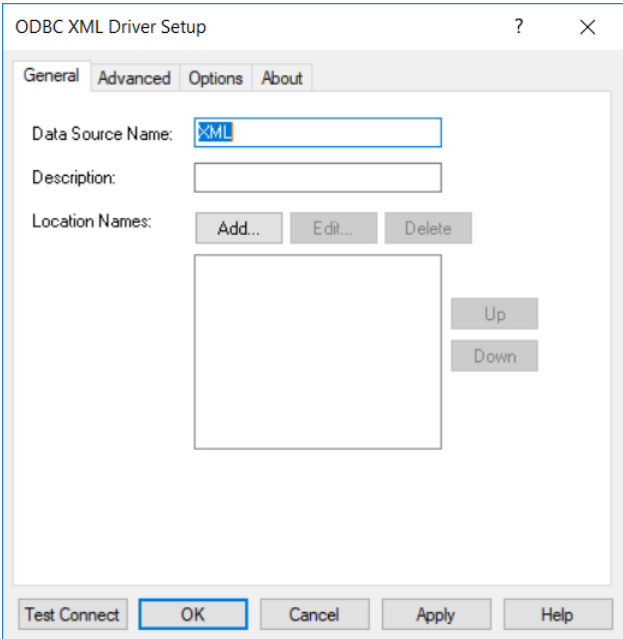
If you are configuring a new user data source, click **Add** on the User DSN tab to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **System DSN:** To configure a new system data source, click **Add** on the System DSN tab to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
- **File DSN:** If you are configuring an existing file data source, select the data source name on the File DSN tab and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** on the File DSN tab to display a list of installed drivers. Select the driver and click **Next**. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 82: Driver Setup: General tab



Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

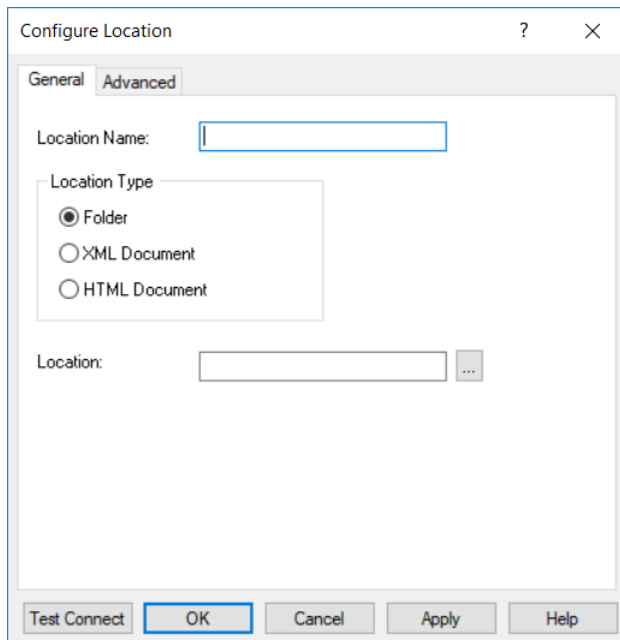
- 2. On the General tab, provide the following information; then, click **Apply**.

Connection Options: General	Default
Data Source Name on page 797	None
Description on page 798	None
Location Names on page 799	None

- 3. If you want to edit or delete a location name, or change its position in the list, select it; then, click **Edit**, **Delete**, **Up**, or **Down** as appropriate.


- If you want to define a location, click **Add**. The Configure Location dialog box appears.

Figure 83: Configure Location: General tab



- On the General tab of the Configure Location dialog box, provide the following required information; then, click **Apply**.

Configure Location Options: General	Default
Location Names on page 799	None
Location Type on page 807	None
Location on page 806	localhost

Location: Either type the full path to the location you are defining or click the select button:  to select a path.

6. Optionally, click the **Advanced** tab of the Configure Location dialog box to specify additional information about the location you are defining.

Figure 84: Configure Location: Advanced tab

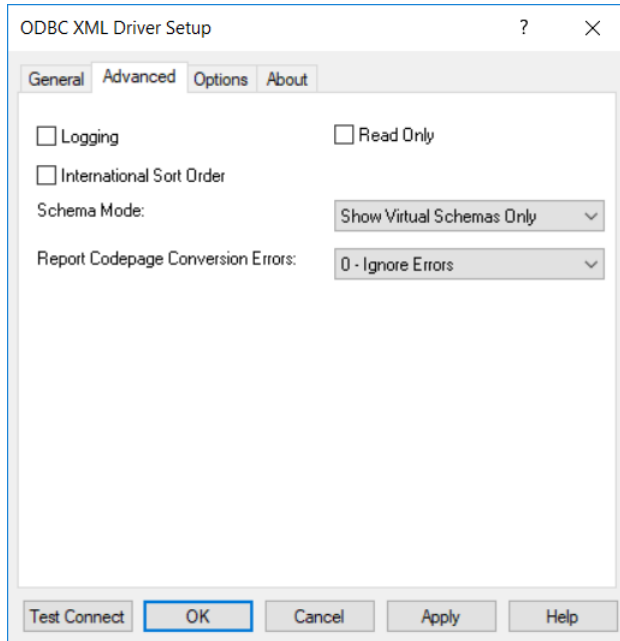
On this tab, provide any of the following optional information; then, click **Apply**.

Configure Location Options: Advanced	Default
Table Hint on page 811	None
Row Hint on page 809	None
Validate Schema on page 812	Disabled
Resolve External References on page 809	Disabled
Flush Every Change on page 805	Enabled
Require User ID/Password on page 808	Disabled
User ID on page 811	None
Password on page 808	None
Table Creation on page 810	DataDirect Format
Delete Linked Schema on page 804	Disabled
Max Rows to Scan on page 807	0

7. You can click **Test Connect** to attempt to connect to the location.
- If the driver can connect, it releases the connection and displays a `connection established` message. Click **OK**.

- If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.
8. Click **OK** to return to the ODBC XML driver Setup dialog box or **Cancel**. If you click **OK**, the values you have specified become the defaults for this location.
 9. Optionally, click the **Advanced** tab of the ODBC XML driver Setup dialog box to specify data source settings.

Figure 85: Driver Setup: Advanced tab

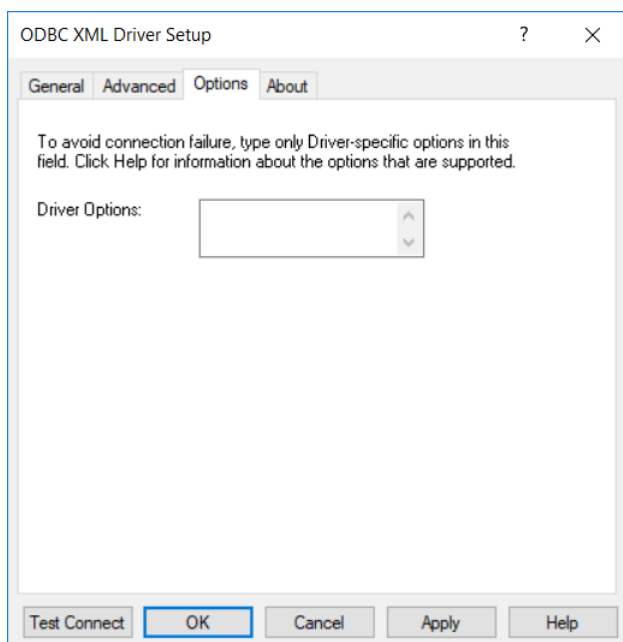


On this tab, provide any of the following optional information; then, click **Apply**.

Connection Options: Advanced	Default
Logging on page 800	Disabled
International Sort Order on page 799	Disabled
Schema Mode on page 802	Show Virtual Schemas Only
Report Codepage Conversion Errors on page 801	0 - Ignore Errors
Read Only on page 800	Disabled

10. Optionally, click the **Options** tab to specify data source connection values.

Figure 86: Driver Setup: Options tab



Driver Options: Type configuration options specific to the XML driver.

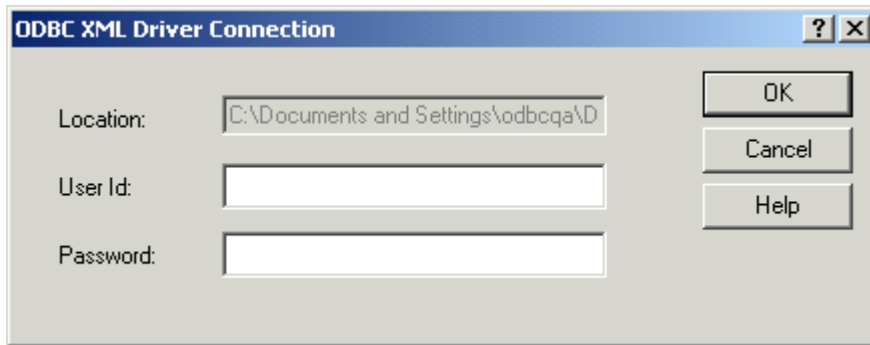
Connection Options: Options	Default
Driver Options on page 798	Disabled

Warning: The properties you set in the Options tab override other properties for this session only and can adversely affect the operation of the XML driver. Use only authorized entries. For information about authorized entries for the Options tab, contact Progress DataDirect customer support.

11. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(XML\)](#) on page 795 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a `connection established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.
12. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Logon Dialog Box (XML)

Some ODBC applications display a Logon dialog box when you are connecting to a data source. For XML, the dialog box is as follows:



This dialog box appears for each password-protected location that you have defined for the data source.

In this dialog box, provide the following information:

1. Type your user ID and password in the appropriate fields for the Location that appears in the Location field.
2. Click **OK** to connect to the data source.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={ }driver_name[ ] [;attribute=value[;attribute=value]...]
```

[Connection Option Descriptions](#) on page 797 and [Configure Location Dialog Box Descriptions](#) on page 804 give the names and descriptions of the attributes, as well as the initial default value when the driver is first installed.

An example of a DSN connection string with overriding attribute values for XML is:

```
DSN=XML FILES;LOC1.Create Type=ADO25;Logging=1
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=XML.dsn;LOC1.Create Type=ADO25;Logging=1
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 XML};LOC1={DataDirect Closed XML ADO Provider}
```

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog. The connection string attribute name is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

Note: XML driver connection string attributes do not use short name equivalents.

The following table lists the connection string attributes associated with General, Advanced, and Options tabs of the XML driver Setup dialog box. The descriptions themselves are listed below the table.

Table 60: Attributes on the XML Driver Setup Dialog Box

Attribute	Default
DataSourceName	None
Description	None
InternationalSort	Disabled
Location	None
Logging	Disabled
ReadOnly	Disabled
ReportCodepageConversionErrors	0 (Ignore Errors)
ShowManufacturedSchemas	0 (Disabled)
ShowVirtualSchemas	1 (Enabled)

Data Source Name

Attribute

DataSourceName

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[Driver Setup: General tab](#)

Description

Attribute

Description

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[Driver Setup: General tab](#)

Driver Options

Attribute

n/a

Purpose

Type configuration options specific to the XML driver.

Valid Values

WARNING: The properties you set in the Options tab override other properties for this session only and can adversely affect the operation of the XML driver. Use only authorized entries. For information about authorized entries for the Options tab, contact Progress DataDirect customer support.

Default

None

GUI Tab

[Driver Setup: Options tab](#)

International Sort Order

Attribute

InternationalSort

Purpose

Uses international sort order as defined by your operating system when you issue a Select statement with an Order By clause.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), this order is always alphabetic, regardless of case; the letters are sorted as "A, b, C." Refer to your operating system documentation concerning the sorting of accented characters.

If set to 0 (Disabled), ASCII sort order is used. This order sorts items alphabetically with uppercase letters preceding lowercase letters. For example, "A, b, C" is sorted as "A, C, b."

Default

0 (Disabled)

GUI Tab

[Driver Setup: Advanced tab](#)

Location Names

Attribute

Location

Purpose

A display of all existing location names defined for the data source you are configuring.

Valid Values

string

where:

```
string
```

is the name of a location.

The location names listed in the text box are used for connections according to the order that they are displayed. If you want to change the order or precedence, use the Up and Down buttons.

Default

None

GUI Tab

[Driver Setup: General tab](#)

Logging

Attribute

Logging

Purpose

Creates a log file that logs the SQL execution plan. A value of 0 means no logging is performed.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), a log file is created in the current directory. The default log file name is \Integrator.txt.

If set to 0 (Disabled), no logging is performed.

Default

0 (Disabled)

GUI Tab

[Driver Setup: Advanced tab](#)

Read Only

Attribute

ReadOnly

Purpose

Controls whether the driver opens files for Read-Write access or Read-Only access

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver opens XML files for Read-Only access. In this case, the XML driver opens XML files with a Shared Read lock. This allows other connections and applications to read the same XML file that the XML driver has open; however, they cannot write to the XML file.

If set to 0 (Disabled), the XML driver opens XML files for Read-Write access. Opening an XML file for Read-Write access places an exclusive lock on the file. No other connections or applications can open the XML file while the driver has the file open.

Default

0 (Disabled)

GUI Tab

[Driver Setup: Advanced tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where `x` is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Driver Setup: Advanced tab](#)

Schema Mode

Attribute

n/a

Purpose

Specifies whether to show virtual schemas, manufactured schemas, or both.

Valid Values

Choose one of the following options:

- Show Virtual Schemas Only. This option returns "XML" in the Schema Name column when a SQLTables or SQLColumns operation is performed when connected to a data source. See [Show Virtual Schemas](#) on page 803.
- Show Manufactured Schemas Only. This option returns the manufactured schema names in the Schema Name column when a SQLTables or SQLColumns operation is performed when connected to a data source. See [Show Manufactured Schemas](#) on page 802.
- Show Both Virtual and Manufactured Schemas. This option returns both virtual and manufactured schema names when a SQLTables or SQLColumns operation is performed when connected to a data source.

Default

Show Virtual Schemas Only

GUI Tab

[Driver Setup: Advanced tab](#)

Show Manufactured Schemas

Attribute

ShowManufacturedSchemas

Purpose

Returns the manufactured schema names in the Schema Name column when a SQLTables or SQLColumns operation is performed when connected to a data source. The Location names you define for a data source are manufactured into a schema name by adding a # symbol after the Location names. For example:

Schema Name	Table Name
LOC1#	TAB1A
LOC1#	TAB1B
LOC2#	TAB2A
LOC2#	TAB2B

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), manufactured schema names are returned.

If set to 0 (Disabled), manufactured schema names are not returned.

To return both manufactured and virtual schema names, set this option to 1 (Enabled) and the Show Virtual Schemas option to 1 (Enabled).

Default

0 (Disabled)

GUI Tab

[Driver Setup: Advanced tab](#)

Show Virtual Schemas**Attribute**

ShowVirtualSchemas

Purpose

Returns "XML" in the Schema Name column when a SQLTables or SQLColumns operation is performed when connected to a data source. For example:

Schema Name	Table Name
XML	TAB1A
XML	TAB1B
XML	TAB2A
XML	TAB2B

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), virtual schema names are returned.

If set to 0 (Disabled), virtual schema names are not returned.

To return both virtual and manufactured schema names, set this option to 1 (Enabled) and the Show Manufactured Schemas option to 1 (Enabled).

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Configure Location Dialog Box Descriptions

The following table lists the connection string attributes associated with General and Advanced tabs of the XML driver Configure Location dialog box. The descriptions themselves are listed below the table. See [Defining Locations](#) on page 788 for an explanation of locations.

Note: XML driver connection string attributes do not use short name equivalents.

The names of all connection options in this section are preceded by *location_name*, where *location_name* represents the name of a specific location that you have defined, for example, LOC1. See the description of the [Location Names](#) on page 799 option for details.

Table 61: XML Configure Location Attribute Names

Attribute	Default
location_name	None
location_name.Catalog Type Hint	Folder
location_name.Create Type	DataDirect
location_name.Delete Schema	0 (Disabled)
location_name.Flush Every Change	1 (Enabled)
location_name.Initial Catalog	None
location_name.Password	None
location_name.Require Passwd	0 (Disabled)
location_name.Resolve External	0 (Disabled)
location_name.Row Hint	None
location_name.Scan Rows	0
location_name.Table Hint	None
location_name.User ID	None
location_name.Validate Schema	0 (Disabled)

Delete Linked Schema

Attribute

location_name.Delete Schema

Purpose

Specifies whether an externally-linked schema file is deleted when a table is deleted. This option is valid only for Folder location types. The XML document for the table contains a link to this external schema file. By default, this check box is not selected.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the externally-linked schema file is deleted when the table is deleted. If multiple XML documents are linked to the same schema file, the schema file is not deleted when a table is deleted.

If set to 0 (Disabled), the externally-linked schema file is not deleted when the table is deleted.

Default

0 (Disabled)

GUI Tab

[Configure Location: Advanced tab](#)

Flush Every Change

Attribute

location_name.Flush Every Change

Purpose

Writes the data document to disk after every insert, update, or delete operation. This option is valid only for Folder location types.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver writes the data document to disk after every change.

If set to 0 (Disabled), the driver does not write the data document to disk after every change. Disabling this option can improve performance.

Default

1 (Enabled)

GUI Tab

[Configure Location: Advanced tab](#)

Location

Attribute

location_name.Initial Catalog

Purpose

The full path name to the location you are defining.

Valid Values

location_directory

where *location_directory* is the full path name of the directory in which the data files are stored. For example:

```
LOC1.Initial Catalog=C:\Documents\filesml
```

Default

None

GUI Tab

[Configure Location: General tab](#)

Location Name

Attribute

location_name

Purpose

A unique name for the location you are defining, for example, LOC1.

Valid Values

location_name={DataDirect Closed XML ADO Provider}

where *location_name* is the unique name of the location you are defining. For example, if you choose the location name LOC1, then:

```
LOC1={DataDirect Closed XML ADO Provider}
```

Default

None

GUI Tab

[Configure Location: General tab](#)

Location Type

Attribute

location_name.Catalog Type Hint

Purpose

Specifies the type of location you are defining for the connection.

Valid Values

Folder | XML Document | HTML Document

For example:

```
LOC1.Catalog Type Hint=XML Document
```

Default

Folder

GUI Tab

[Configure Location: General tab](#)

See also

See [Defining Locations](#) on page 788 for the definition of each type.

Max Rows to Scan

Attribute

location_name.Scan Rows

Purpose

An integer that represents the maximum number of rows to scan when the XML driver is determining the data type of each column. This option is valid only for XML Document location types.

Valid Values

0 | x

where:

x

is the number of rows to scan.

Behavior

If set to x, the driver scans a maximum of x rows in the table. During the scan, the driver inspects each column value in the row of a table and adjusts the data type determination for each column based on the corresponding value. The more sample column values it encounters, the more accurate the determination.

If set to 0, the driver scans all rows in the table. Disabling this option can improve performance because limiting the number of rows can reduce the amount of time it takes to determine the column information on very large documents. Because less information is available, however, the determination of the data types can be incorrect.

Default

0

GUI Tab

[Configure Location: Advanced tab](#)

Password

Attribute

location_name.Password

Purpose

The password used to establish a connection to the location specified by *location_name*. A password is required only if the location to which you are connecting is password-protected.

This option is not available unless the Require User ID/Password option is enabled.

Valid Values

pwd

where:

pwd

is a valid password.

Warning: The encrypted password is stored in the Windows Registry.

Default

None

GUI Tab

[Configure Location: Advanced tab](#)

Require User ID/Password

Attribute

location_name.Require Passwd

Purpose

Specifies whether a User ID and password are required to establish a connection to the location you are defining.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), a User ID and password are required to establish a connection to the location. You must enable this option if the location you are defining is password-protected; otherwise, the connection will fail. Enabling this option causes a Logon dialog box to appear when connecting with the driver.

If set to 0 (Disabled), no user ID and password are required to establish a connection to the location.

Default

0 (Disabled)

GUI Tab

[Configure Location: Advanced tab](#)

Resolve External References

Attribute

location_name.Resolve External

Purpose

Determines whether external references such as DTDs, Schemas, Entities, and Notations are resolved for the XML documents contained within the location specified by *location_name*.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the documents are not processed if the XML parser cannot locate the external references.

If set to 0 (Disabled), the document is processed, even if the XML parser cannot locate the external references.

Default

0 (Disabled)

GUI Tab

[Configure Location: Advanced tab](#)

Row Hint

Attribute

location_name.Row Hint

Purpose

A string that specifies an Extensible Stylesheet Language (XSL) pattern to identify the nodes that make up the rows in the rowset of a tabular-formatted XML document contained within the location specified by *location_name*. See [Using Hints for Tabular-Formatted XML Documents](#) on page 812 for details.

This option is valid only for Folder and HTML Document location types.

Valid Values

row_hint

where:

row_hint

is an XSL pattern.

Default

None

GUI Tab

[Configure Location: Advanced tab](#)

Table Creation

Attribute

location_name.Create Type

Purpose

Determines the style of XML that is generated when a new table is created. This option is valid only for Folder location types.

Valid Values

IE5DataIsland | ADO25 | DataDirect

- Data Island Format (IE5DataIsland): New tables are created with the Internet Explorer 5 Data Island XML style.
- ADO Format (ADO25): New tables are created with the ADO 2.5 XML style.
- DataDirect Format (DataDirect): New tables are created with the DataDirect format. This format conforms to the W3C recommendation for XML schema, working draft April 07, 2000.

Default

DataDirect

GUI Tab

[Configure Location: Advanced tab](#)

See also

See [Supported Tabular Formats for XML Documents](#) on page 785 for a description of each of these formats.

Table Hint

Attribute

location_name.Table Hint

Purpose

A string that specifies an Extensible Stylesheet Language (XSL) pattern to identify the table or rowset nodes in a tabular-formatted XML document contained within the location specified by *location_name*. See [Using Hints for Tabular-Formatted XML Documents](#) on page 812 for details.

This option is valid only for Folder and HTML Document location types.

Valid Values

table_hint

where:

table_hint

is an XSL pattern.

Default

None

GUI Tab

[Configure Location: Advanced tab](#)

User ID

Attribute

location_name.User ID

Purpose

The User ID (user name) used to establish a connection to the location specified by *location_name*. A password is required only if the location to which you are connecting is password-protected.

This option is not available unless the Require User ID/Password option is enabled.

Valid Values

userid

where:

userid

is a valid user name.

Default

None

GUI Tab

[Configure Location: Advanced tab](#)

Validate Schema

Attribute

location_name.Validate Schema

Purpose

Determines whether the XML documents contained within the location specified by *location_name* are validated against their schema.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the XML documents are validated against their schema. This allows a well-formed XML document to be processed, even if the document is not valid.

If set to 0 (Disabled), the XML documents are not validated against their schema.

Default

0 (Disabled)

GUI Tab

[Configure Location: Advanced tab](#)

Using Hints for Tabular-Formatted XML Documents

The XML driver supports table and row hints. You can specify a table hint, a row hint, or both, when configuring an XML data source or using a connection string.

Table hints should be specified so that they resolve to a single node. If a table hint resolves to a set of nodes, the first node in the set is used as the table node. The context of the table hint is always the root node of the XML document.

Row hints define the "row" element and specify whether the rowset is element-based or attribute-based. If a table hint is supplied, the context of the row node is the node to which the table hint resolves; otherwise, the context is the root node of the XML document. The column mode identifier specifies whether the columns of a row are child nodes or attributes of the row node.

When working with hints, keep in mind that the XML driver assumes that the row nodes are the immediate children of the table node.

- If only a table hint is specified, the row nodes are the children of the node to which the hint resolves. It is assumed that all of the child nodes have the same name.
- If only a row hint is specified, the table node is the parent of the node to which the hint resolves. If the row hint resolves to a set of nodes, the nodes in that set must all have the same parent.

- If both a table hint and a row hint are specified, the row hint is taken to be relative to the node to which the table hint resolves.

The column mode identifier has the format:

```
\column mode
```

where mode can be one of the following options:

- child: The columns are child nodes of the row node.
- attr: The columns are attributes of the row node.

In the following examples, the columns are the children of the row nodes.

Example 1

Table Hint:

Row Hint: //Item

The row nodes are the nodes named Item. The table node is the parent of the row nodes. Use this form only when all of the Item nodes reside under one parent.

If some Item nodes have different parents, use a table hint or a more specific row hint to select the set of Item nodes.

Example 2

Table Hint:

Row Hint: /Bookstore/Books/Item

The row nodes are the nodes named Item. The table node is Books, which is a child of the Bookstore node.

Example 3

Table Hint: /Bookstore/Books

Row Hint:

The table node is Books, which is a child of the Bookstore node. The row nodes are the children of the Books node. It is assumed that all of the child nodes under the Books nodes have the same name. If the child nodes do not all have the same name, the name of the first child node encountered is used as the row node name. In that case, it would be better to specify both a table and row hint.

Example 4

Table Hint: /Bookstore [@location = "Raleigh"]/Books

Row Hint: ./Item

The table node is Books, which is a child of the Bookstore node. Bookstore has a "location" attribute with the value Raleigh. The row nodes are the Item nodes that are children of the Books node.

Column Mode Identifier

The following examples illustrate the use of the optional column mode identifier.

Example 5

Table Hint:

Row Hint: //Item \column attr

The row nodes are named Item. The table node is the parent of the row nodes. The columns are attributes of the row node.

Example 6

Table Hint:

Row Hint: //Item \column child

The row nodes are the nodes named Item. The table node is the parent of the row nodes. The columns are attributes of the row node.

Data Types

This section provides three tables that show how the data types for each supported tabular-formatted XML document map to the standard ODBC data types.

Table 62: Data Islands Data Types

Data Islands	Internal XML Name	ODBC
binhex	bin.hex	SQL_LONGVARBINARY
boolean	boolean	SQL_BIT
currency	fixed.14.4	SQL_DECIMAL
date	date	SQL_TYPE_DATE
dateTime	dateTime	SQL_TYPE_TIMESTAMP
float	float	SQL_DOUBLE
i1	i1	SQL_TINYINT SIGNED
i2	i2	SQL_SMALLINT SIGNED
i4	i4	SQL_INTEGER SIGNED
int	int	SQL_INTEGER SIGNED
number	number	SQL_DOUBLE
r4	r4	SQL_REAL
r8	r8	SQL_DOUBLE
singleChar	singleChar	SQL_SMALLINT
string	string	SQL_WLONGVARCHAR
time	time	SQL_TYPE_TIME

Data Islands	Internal XML Name	ODBC
ui1	ui1	SQL_TINYINT UNSIGNED
ui2	ui2	SQL_SMALLINT UNSIGNED
ui4	ui4	SQL_INTEGER UNSIGNED

Table 63: ADO 2.5 Persisted Files Data Types

ADO 2.5 Persisted Files	Internal XML Name	ODBC
binhex	bin.hex	SQL_LONGVARBINARY
boolean	boolean	SQL_BIT
currency	fixed.14.4	SQL_DECIMAL
date	date	SQL_TYPE_DATE
dateTime	dateTime	SQL_TYPE_TIMESTAMP
float	float	SQL_DOUBLE
i1	i1	SQL_TINYINT SIGNED
i2	i2	SQL_SMALLINT SIGNED
i4	i4	SQL_INTEGER SIGNED
i8	i8	SQL_BIGINT SIGNED
int	int	SQL_INTEGER UNSIGNED
number	number	SQL_DOUBLE
r4	r4	SQL_REAL
r8	r8	SQL_DOUBLE
singleChar	singleChar	SQL_SMALLINT SIGNED
time	time	SQL_TYPE_TIME
ui1	ui1	SQL_TINYINT UNSIGNED
ui2	ui2	SQL_SMALLINT UNSIGNED
ui4	ui4	SQL_INTEGER UNSIGNED
ui8	ui8	SQL_BIGINT UNSIGNED
wchar	string	SQL_CHAR

ADO 2.5 Persisted Files	Internal XML Name	ODBC
wchar	string	SQL_WCHAR
wlvarchar	string	SQL_WLONGVARBINARY
wvarchar	string	SQL_WVARCHAR

Table 64: DataDirect Format Data Types

DataDirect	Internal XML Name	ODBC
binary	binary	SQL_BINARY
boolean	boolean	SQL_BIT
byte	byte	SQL_TINYINT SIGNED
date	date	SQL_TYPE_DATE
decimal	decimal	SQL_NUMERIC
double	double	SQL_DOUBLE
float	float	SQL_REAL
int	int	SQL_INTEGER UNSIGNED
long	long	SQL_BIGINT SIGNED
lvarbinary	binary	SQL_LONGVARBINARY
short	short	SQL_SMALLINT SIGNED
time	time	SQL_TYPE_TIME
timeInstant	timeInstant	SQL_TYPE_TIMESTAMP
unsignedByte	unsignedByte	SQL_TINYINT UNSIGNED
unsignedInt	unsignedInt	SQL_INTEGER UNSIGNED
unsignedLong	unsignedLong	SQL_BIGINT UNSIGNED
unsignedShort	unsignedShort	SQL_SMALLINT UNSIGNED
varbinary	binary	SQL_VARBINARY
wchar	string	SQL_CHAR
wchar	string	SQL_WCHAR

DataDirect	Internal XML Name	ODBC
wlvarchar	string	SQL_WLONGVARBINARY
wvarchar	string	SQL_WVARCHAR

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

Unicode Support

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

In addition, the driver supports SQLSetPos.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

There is no limit to the number of connections and statements supported.

SQL Support

This section provides information about the SQL statements that the XML driver processes, and about SQL standards and conventions that the driver supports:

- [SQL Statements](#) on page 818
- [Extensions to SQL Standards](#) on page 818
- [Grammar Token Definitions](#) on page 818

SQL Statements

The SQL Engine included with the XML driver supports the following SQL statements:

- Select
- Create and Drop Table
- Insert
- Update
- Delete

Note: See the table at the beginning of this chapter for the SQL statements that the XML driver supports for the different types of supported file formats.

Extensions to SQL Standards

The XML driver uses SQL grammar that is compliant with entry level ANSI SQL-92. The following table summarizes significant extensions to the grammar.

Table 65: SQL Extensions

Entry Level ANSI SQL-92 Extension	Relevant Standard or Convention
Aliasing table references	Intermediate level ANSI SQL-92
ANSI date, time, and timestamp literals	Intermediate level ANSI SQL-92
Dynamic parameter specification	Full level ANSI SQL-92
GUID literals	COM
Hex string literals	Full level ANSI SQL-92
Left Outer Joins	Intermediate level ANSI SQL-92
ODBC escape support	ODBC 3.0
Scalar functions	ODBC 3.0

Grammar Token Definitions

The tokens used in the XML driver SQL grammar are defined in the following sections:

- [Regular Identifiers](#) on page 819
- [Delimited Identifiers](#) on page 819
- [Integer Numbers](#) on page 819
- [Real Numbers](#) on page 819
- [Character String Literals](#) on page 820

- [GUID Literals](#) on page 820
- [Hex Literals](#) on page 820
- [Time and Date Literals](#) on page 820
- [SQL Operators and Symbols](#) on page 821
- [Keywords for the XML Driver](#) on page 821
- [SQL Comments](#) on page 826

Regular Identifiers

A regular identifier must begin with a letter and may not exceed 128 characters. In addition, all ASCII characters are converted to uppercase.

The following are examples of regular identifiers:

- FOO
- COLUMN_NAME
- SCHEMA#NAME
- Col3 (legal, but converted to COL3)

Delimited Identifiers

Delimited identifiers may not exceed 128 characters. A double quotation character can be embedded within the string by specifying two consecutive double quotation mark characters. A delimited identifier can span multiple lines. The body of a delimited identifier can contain any character except the newline character.

The following examples show delimited identifiers:

- "\$ % ^ (\$"
- "This is a delimited variable name"

Integer Numbers

Examples of integer numbers are:

- 5
- 1004

Real Numbers

Examples of real numbers are:

- .10
- 12.01
- 10.
- .01e-10
- 12E+10

- 12.01e2
- 12.01e-10
- 12.e-10

Character String Literals

Character string literals are delimited with single quotation mark characters. A single quotation mark character can be embedded within the string by specifying two consecutive single quotation mark characters. A character string literal can span multiple lines.

Examples are:

- '\$%^('\$)
- 'This is a character string literal'

GUID Literals

A GUID uses the following format, where x is a hexadecimal digit:

```
xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

Hex Literals

Hex literal values are introduced with an uppercase *x* followed by a single quoted string of hexadecimal characters.

Examples are:

- X'39FA'
- X'B0F00D'

Time and Date Literals

Date, time, and timestamp literals are date, time, and timestamp values surrounded by a standard prefix and suffix. Date literals are specified in a *YYYY-MM-DD* format. Time literals are specified in an *HH:MM:SS* format with an optional fraction component. Timestamp literals are a concatenation of date and time values.

Examples for ODBC and SQL syntax are shown in the following table.

Table 66: Time and Date Literals

Literal Type	ODBC Syntax	ANSI SQL-92 Syntax
Date Literal	{d '1999-09-19'}	date '1999-09-19'
Time Literal	{t '11:11:11.225'}	time '11:11:11.225'
Timestamp Literal	{ts '1999-09-19 11:11:11.225'}	timestamp '1999-09-19 11:11:11.225'
Timestamp Literal	{ts '1999-09-19'}	timestamp '1999-09-19'

Note: ODBC 1.x style ODBC escape sequences such as the following are not supported:

```
--(*VENDOR(Microsoft), PRODUCT(ODBC) ...*)--
```

SQL Operators and Symbols

Table 67: SQL Operators and Symbols

Symbol	Description	Symbol	Description
':'	Colon	'<'	Less than operator
','	Semicolon)'	Right parenthesis
'.'	Period	'='	Equal operator
','	Comma	'+'	Plus operator
'<>'	Not equal operator	'-'	Minus operator
'<='	Less than or equal operator	'*'	Multiply operator
'>='	Greater than or equal operator	'/'	Divide operator
'>'	Greater than operator	'?'	Dynamic parameter
'('	Left parenthesis		

Keywords for the XML Driver

A keyword may not be used as a regular identifier. For example, the following statement would generate a syntax error because INDICATOR is a keyword:

```
SELECT INDICATOR FROM T1
```

You can, however, enclose a keyword in double quotation marks to form a delimited identifier. For example, the following statement is valid:

```
SELECT "INDICATOR" FROM T1
```

The following table lists all of the keywords that are reserved for use in SQL statements or designated as potential future reserved words.

Table 68: Reserved Keywords

ABSOLUTE	ACTION	ADD
AFTER	ALIAS	ALL
ALLOCATE	ALTER	AND
ANY	ARE	AS
ASC	ASSERTION	ASYNC
AT	AUTHORIZATION	AVG
BEFORE	BEGIN	BETWEEN
BIT	BIT_LENGTH	BOOLEAN
BOTH	BREADTH	BY
CALL	CASCADE	CASCADED
CASE	CAST	CATALOG
CHAR	CHAR_LENGTH	CHARACTER
CHARACTER_LENGTH	CHECK	CLOSE
COALESCE	COLLATE	COLLATION
COLUMN	COLUMNS	COMMIT
COMPLETION	CONCAT	CONNECT
CONNECTION	CONSTRAINT	CONSTRAINTS
CONTINUE	CONVERT	CORRESPONDING
COUNT	CREATE	CROSS
CURDATE	CURRENT	CURRENT_DATE
CURRENT_TIME	CURRENT_TIMESTAMP	CURRENT_USER
CURSOR	CURTIME	CYCLE

DATA	DATE	DAY
DAYOFMONTH	DAYOFWEEK	DEALLOCATE
DEC	DECIMAL	DECLARE
DEFAULT	DEFERRABLE	DEFERRED
DELETE	DEPTH	DESC
DESCRIBE	DESCRIPTOR	DIAGNOSTICS
DICTIONARY	DISCONNECT	DISTINCT
DOMAIN	DOUBLE	DROP
EACH	ELSE	ELSEIF
END	END_EXEC	EQUALS
ESCAPE	EXCEPT	EXCEPTION
EXEC	EXECUTE	EXISTS
EXTERNAL	EXTRACT	FALSE
FETCH	FIRST	FLOAT
FLOOR	FOR	FOREIGN
FOUND	FROM	FULL
GENERAL	GET	GLOBAL
GO	GOTO	GRANT
GROUP	HAVING	HOUR
IDENTIFY	IF	IFNULL
IGNORE	IMMEDIATE	IN
INDEX	INFO	INDICATOR
INITIALLY	INNER	INPUT
INSENSITIVE	INSERT	INT
INTEGER	INTERSECT	INTERVAL
INTO	IS	ISOLATION
JOIN	KEY	LANGUAGE

LAST	LCASE	LEADING
LEAVE	LEFT	LENGTH
LESS	LEVEL	LIKE
LIMIT	LOCAL	LOOP
LOWER	LTRIM	MATCH
MAX	MIN	MINUTE
MOD	MODIFY	MODULE
MONTH	NAMES	NATIONAL
NATURAL	NCHAR	NEW
NEXT	NO	NONE
NOT	NOW	NULL
NULLIF	NUMERIC	OBJECT
OCTET_LENGTH	OF	OFF
OID	OLD	ON
ONLY	OPEN	OPERATION
OPERATORS	OPTION	OR
ORDER	OTHERS	OUTER
OUTPUT	OVERLAPS	PAD
PARAMETERS	PARTIAL	PENDANT
POSITION	POWER	PRECISION
PREORDER	PREPARE	PRESERVE
PRIMARY	PRIOR	PRIVATE
PRIVILEGES	PROCEDURE	PROTECTED
PUBLIC	RCASE	READ
REAL	RECURSIVE	REF
REFERENCES	REFERENCING	RELATIVE
REMOVE	REPLACE	RESIGNAL

RESTRICT	RETURN	RETURNS
REVOKE	RIGHT	ROLE
ROLLBACK	ROUND	ROUTINE
ROW	ROWS	RTRIM
SAVEPOINT	SCHEMA	SCROLL
SEARCH	SECOND	SECTION
SELECT	SENSITIVE	SEQUENCE
SESSION	SESSION_USER	SET
SIGNAL	SIMILAR	SIZE
SMALLINT	SOME	SPACE
SQL	SQLCODE	SQLERROR
SQLException	SQLSTATE	SQLWARNING
STRUCTURE	SUBSTRING	SUM
SYSTEM_USER	TABLE	TEMPORARY
TEST	THEN	THERE
TIME	TIMESTAMP	TIMEZONE_HOUR
TIMEZONE_MINUTE	TO	TRAILING
TRANSACTION	TRANSLATE	TRANSLATION
TRIGGER	TRIM	TRUE
TYPE	UCASE	UNDER
UNION	UNIQUE	UNKNOWN
UPDATE	UPPER	USAGE
USER	USING	VALUE
VALUES	VARCHAR	VARIABLE
VARYING	VIEW	VIRTUAL
VISIBLE	WAIT	WHEN
WHENEVER	WHERE	WHILE

WITH	WITHOUT	WORK
WRITE	YEAR	ZONE

SQL Comments

ANSI SQL-92 standard comments (--) and C++ standard comments (/*...*/ , //) are supported. Comments can be nested.

For example, in the following query columns col2, col3, and col4 are ignored:

```
SELECT col1 /* col1 comment */
/*
  col2,-- col2 comment
  col3,// col3 comment
  col4,/* col4 comment */
*/
FROM t1
```

The Connect XE Drivers

This part describes the Progress DataDirect Connect XE drivers. See [Drivers Only Available for 32-Bit Platforms](#) on page 717 and [Drivers for 32-Bit and 64-Bit Platforms](#) on page 139 for information on additional Connect Series drivers.

For details, see the following topics:

- [The Greenplum Wire Protocol Driver](#)
- [The Impala Wire Protocol Driver](#)
- [The Salesforce Driver](#)
- [The Sybase IQ Wire Protocol Driver](#)
- [The Driver for Apache Hive](#)
- [The Driver for the Teradata Database](#)

The Greenplum Wire Protocol Driver

The DataDirect Connect XE for ODBC and DataDirect Connect64 XE for ODBC Greenplum Wire Protocol driver (the Greenplum Wire Protocol driver) each support the following database servers:

- Greenplum
- Pivotal HAWQ

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The Greenplum Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the Windows, UNIX, and Linux environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect XE product for the file name of the Greenplum Wire Protocol driver.

Driver Requirements

The driver has no client requirements.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 836 and [Greenplum Connection Option Descriptions](#) on page 838 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Greenplum Connection Option Descriptions](#) on page 838 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Greenplum)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX® On UNIX and Linux, data sources are stored in the odbc.ini file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Greenplum data source:

1. Start the ODBC Administrator:

-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- **UNIX**[®] On Linux, change to the *install_dir/tools* directory and, at a command prompt, enter:


```
odbcadmin
```

where *install_dir* is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears.

Figure 87: General tab

ODBC Greenplum Wire Protocol Driver Setup

General | Advanced | Security | Failover | Pooling | About

Data Source Name: Help

Description:

Host Name:

Port Number:

Database Name:

Test Connect | OK | Cancel | Apply

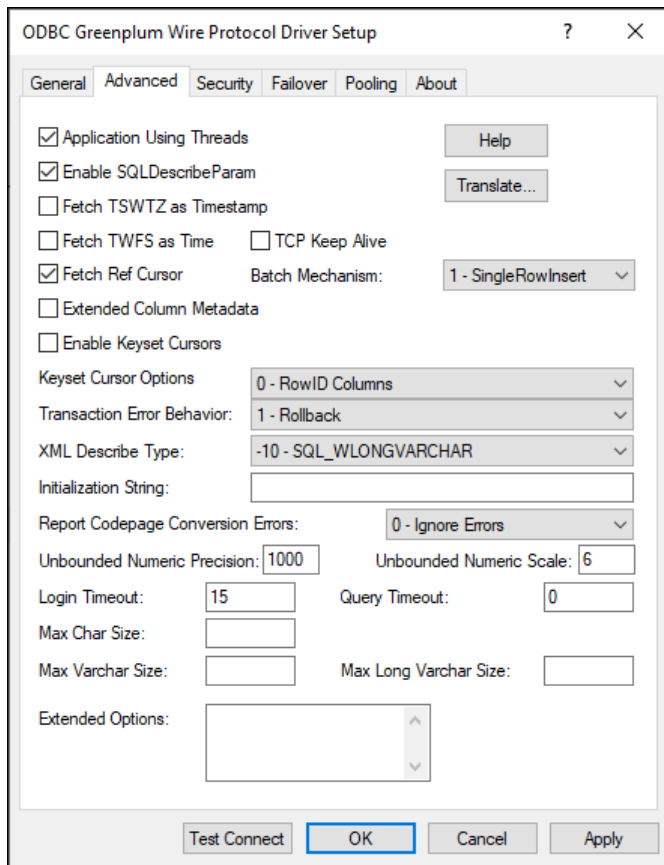
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 849	None
Description on page 850	None
Host Name on page 858	None
Port Number on page 869	5432
Database Name on page 850	None

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 88: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Application Using Threads on page 842	Enabled
Enable SQLDescribeParam on page 851	Enabled
Fetch TSWTZ as Timestamp on page 856	Disabled
Fetch TWFS as Time on page 857	Disabled
TCP Keep Alive on page 873	Disabled
Fetch RefCursors on page 855	Enabled
Batch Mechanism on page 844	1 - SingleRowInsert
Extended Column MetaData on page 853	Disabled
Enable Keyset Cursors on page 851	Disabled
Keyset Cursor Options on page 861	0 - RowID Columns

Connection Options: Advanced	Default
Transaction Error Behavior on page 873	1 - Rollback
XML Describe Type on page 878	-10 - SQL_WLONGVARCHAR
Initialization String on page 860	None
Report Codepage Conversion Errors on page 870	0 - Ignore Errors
Unbounded Numeric Precision on page 877	1000
Unbounded Numeric Scale on page 877	6
Login Timeout on page 865	15
Query Timeout on page 870	0
Max Char Size on page 866	None
Max Long Varchar Size on page 866	None
Max Varchar Size on page 868	None
IANAAppCodePage on page 860 UNIX ONLY	4 (ISO 8559-1 Latin-1)



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

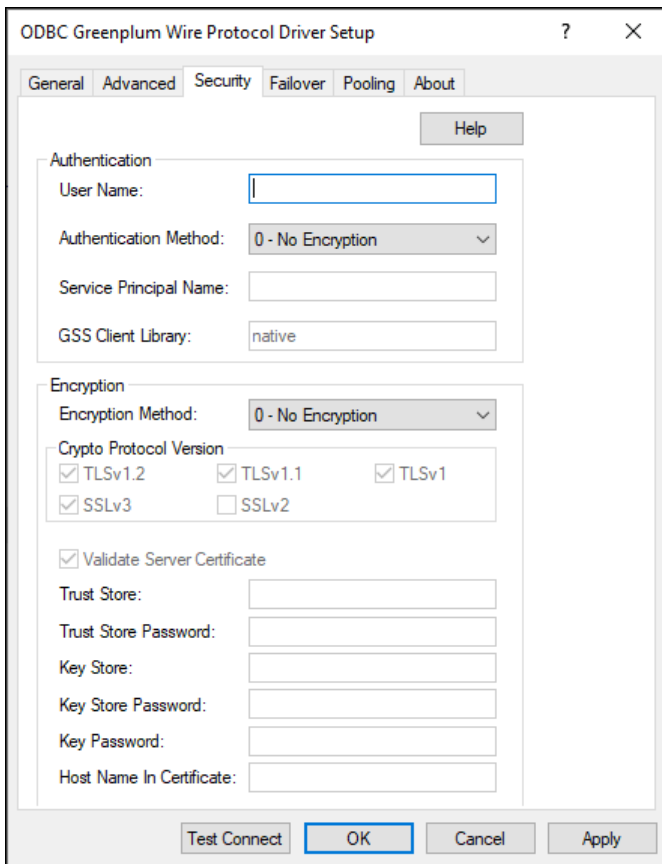
Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.



Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

- Optionally, click the **Security** tab to specify security data source settings.



See [Using Security](#) on page 89 for a general description of authentication and encryption and their configuration requirements.

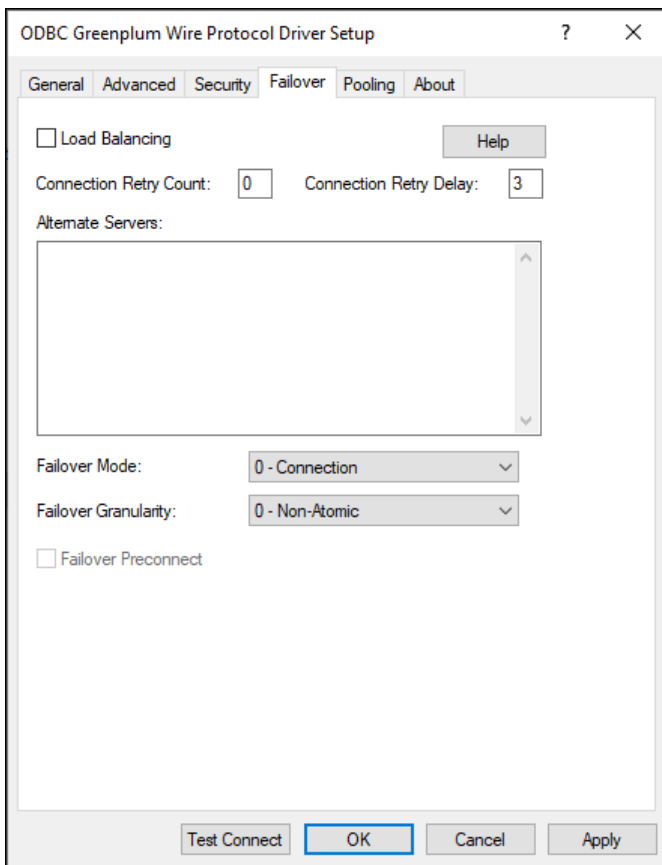
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Security	Default
User Name on page 878	None
Authentication Method on page 843	0 - No Encryption
Service Principal Name on page 871	None
GSS Client Library on page 857	native
Encryption Method on page 852	0 - No Encryption
Crypto Protocol Version on page 847	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Validate Server Certificate on page 876	1 (Enabled)
Truststore on page 874	None
Truststore Password on page 875	None
Keystore on page 862	None

Connection Options: Security	Default
Keystore Password on page 863	None
Key Password on page 862	None
Keystore on page 862	None
Host Name In Certificate on page 859	None

6. Optionally, click the **Failover** tab to specify failover data source settings.

Figure 89: Failover tab



See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

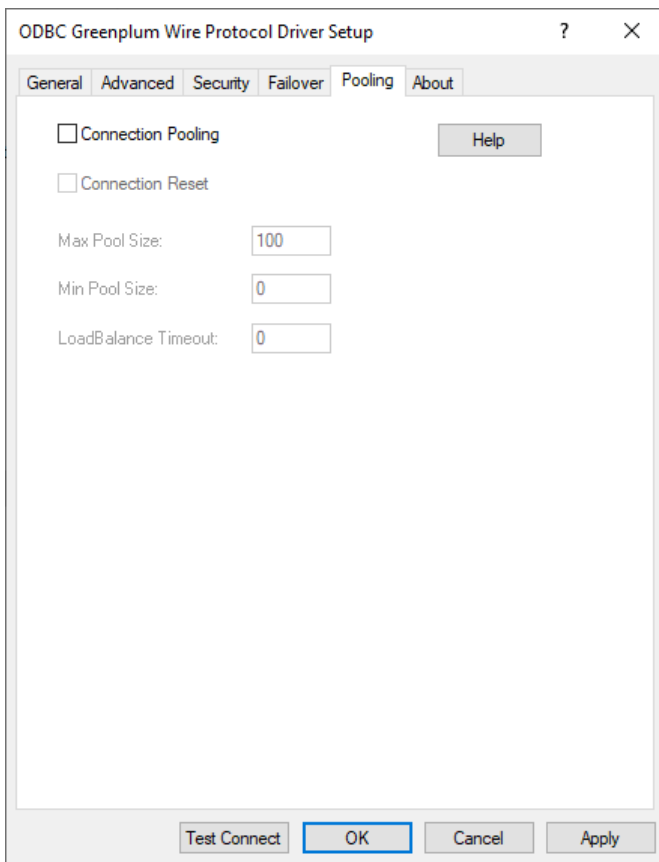
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 864	Disabled
Connection Retry Count on page 846	0
Connection Retry Delay on page 847	3

Connection Options: Failover	Default
Alternate Servers on page 842	None
Failover Mode on page 854	0 - Connection
Failover Granularity on page 853	0 - Non-Atomic
Failover Preconnect on page 855	Disabled

7. Optionally, click the **Pooling** tab to specify pooling data source settings.

Figure 90: Pooling tab



See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 845	Disabled
Connection Reset on page 845	Disabled
Max Pool Size on page 867	100

Connection Options: Pooling	Default
Min Pool Size on page 867	0
Load Balance Timeout on page 864	0

8. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Greenplum\)](#) on page 837 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
 - If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

9. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify `attribute=value` pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[:attribute=value[:attribute=value]. . .]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[:attribute=value[:attribute=value]. . .]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{driver_name}][:attribute=value[:attribute=value]. . .]
```

[Greenplum Connection Option Descriptions](#) on page 838 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Greenplum Wire Protocol is:

```
DSN=Accounting;UID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

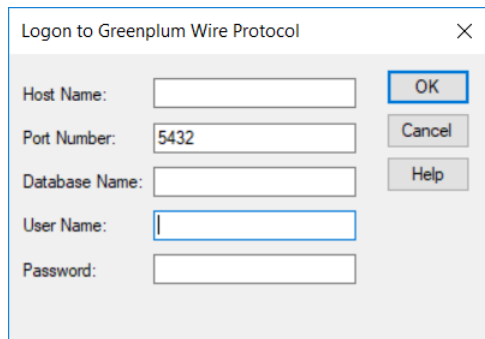
```
FILEDSN=GreenplumWP.dsn;UID=JOHN;PWD=XYZZY
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Greenplum Wire Protocol};
HOST=GreenplumServer;PORT=5432;UID=JOHN;PWD=XYZZY;DB=Gplumdb1
```

Using a Logon Dialog Box (Greenplum)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.



In this dialog box, provide the following information:

1. In the Host Name field, type either the name or the IP address of the server to which you want to connect. The IP address must be in IPv4 format.
2. In the Port Number field, type the number of your Greenplum listener. Check with your database administrator for the correct number.
3. In the Database Name field, type the name of the database to which you want to connect.
4. If required, type your Greenplum user name.
5. If required, type your Greenplum password.
6. Click **OK** to log on to the Greenplum database installed on the server you specified and to update the values in the Registry.

Accessing Greenplum data with Power BI

After you have configured your data source, you can use the driver to access your Greenplum data with Power BI. Power BI is a business intelligence software program that allows you to easily create reports and visualized representations of your data. By using the driver with Power BI, you can improve performance when retrieving data while leveraging the driver's relational mapping tools.

To use the driver to access data with Power BI:

1. Navigate to the `\tools\Power BI` subdirectory of the Progress DataDirect installation directory; then, locate the Power BI connector file `DataDirectGreenPlum.pqx`, and the installation batch file `install.bat`.
2. Run the `install.bat` file. The following operations are executed by running the `install.bat` file.
 - The `DataDirectGreenPlum.pqx` file is copied to the following directory.

```
%USERPROFILE%\Documents\Power BI Desktop\Custom Connectors
```

- The following Windows registry entry is updated.

```
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Power BI
Desktop\TrustedCertificateThumbprints
```

3. Open the Power BI desktop application.
4. From the **Get Data** window, navigate to **Other > DataDirect Greenplum Connector**.
5. Click **Connect**. Then, from the **DataDirect Greenplum Connector** pop-up, provide the following information. Then, click **OK**.
 - **Data Source:** Enter a name for the data source. For example, `Greenplum ODBC DSN`.
 - **SQL Statement:** If desired, provide a SQL command.
 - **Data Connectivity mode:**
 - Select **Import** to import data to Power BI.
 - Select **DirectQuery** to query live data. (For details, including limitations, refer to the Microsoft Power BI article [Use DirectQuery in Power BI Desktop](#).)
6. Enter authentication information when prompted. Once connected, the **Navigator** window displays schema and table information.
7. Select and load tables. Then, prepare your Power BI dashboard as desired.

Results:

You have successfully accessed your data and are now ready to create reports with Power BI. For more information, refer to the Power BI product documentation at [Power BI documentation](#).

Greenplum Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Greenplum Wire Protocol driver.

Table 69: Greenplum Wire Protocol Attribute Names

Attribute (Short Name)	Default
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
AlternateServers (ASRV)	None
ApplicationUsingThreads (AUT)	1 (Enabled)

Attribute (Short Name)	Default
AuthenticationMethod (AM)	0 (No Encryption)
ConnectionReset (CR)	0 (Disabled)
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Database (DB)	None
DataSourceName (DSN)	None
Description (n/a)	None
Enable SQLDescribeParam	1 (Enabled)
EnableKeysetCursors (EKC)	0
EncryptionMethod (EM)	0 (No Encryption)
ExtendedColumnMetaData (ECMD)	0 (Disabled)
FailoverGranularity (FG)	0 (Non-Atomic)
FailoverMode (FM)	0 (Connection)
FailoverPreconnect (FP)	0 (Disabled)
FetchRefCursors (FRC)	1 (Enabled)
FetchTSWTZasTimestamp (FTSWTZAT)	0 (Disabled)
FetchTWFSasTime (FTWFSAT)	0 (Disabled)
GSSClient (GSSC)	native
HostName (HOST)	None
HostNameInCertificate (HNIC)	None
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
InitializationString (IS)	None
KeepAlive (KA)	Disabled

Attribute (Short Name)	Default
KeysetCursorOptions (KCO)	0 - RowID Columns
KeyPassword (KP)	None
KeystorePassword (KSP)	None
Keystore (KS)	None
LoadBalanceTimeout (LBT)	0
LoadBalancing (LB)	0 (Disabled)
LoginTimeout (LT)	15
LogonID (UID)	None
MaxCharSize (MCS)	None
MaxLongVarcharSize (MLVS)	None
MaxPoolSize (MXPS)	100
MaxVarcharSize (MVS)	None
MinPoolSize (MNPS)	0
Password (PWD)	None
Pooling (POOL)	0 (Disabled)
PortNumber (PORT)	5432
QueryTimeout (QT)	0
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
ServicePrincipalName (SPN)	None
SSLLibName (SLN)	Empty string
TransactionErrorBehavior (TEB)	1 (Rollback Transaction)
Truststore (TS)	None
TruststorePassword (TSP)	None
UnboundedNumericPrecision (UNP)	1000
UnboundedNumericScale (UNS)	6

Attribute (Short Name)	Default
ValidateServerCertificate (VSC)	1 (Enabled)
XMLDescribeType (XDT)	-10

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

`latest | openssl_version_number[[, openssl_version_number] ...]`

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to `openssl_version_number`, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the `CryptoLibName` and `SSLlibName` options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

1.1.1,1.0.2

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

[Advanced tab](#)

Alternate Servers

Attribute

AlternateServers (ASRV)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
(HostName=hostvalue:PortNumber=portvalue:Database=databasevalue[, . . .])
```

You must specify the host name, port number, and database name of each alternate server.

Example

The following Alternate Servers value defines two alternate database servers for connection failover:

```
AlternateServers=(HostName=GreenplumServer:PortNumber=5431:  
Database=Pgredb1, HostName=255.201.11.24:PortNumber=5432:Database=Pgredb2)
```

Default

None

GUI Tab

[Failover tab](#)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#)

Authentication Method

Attribute

AuthenticationMethod (AM)

Purpose

Specifies the method the driver uses to authenticate the user to the server when a connection is established. If the specified authentication method is not supported by the database server, the connection fails and the driver generates an error.

Important: When Kerberos is enabled, if the database user name differs from the domain user name, you are required to pass the database user name via the User Name (LogonID) option in the datasource or connection string.

Valid Values

0 | 4

Behavior

If set to 0 (No Encryption), the driver sends the user ID and password in clear text to the server for authentication.

If set to 4 (Kerberos Authentication), the driver uses Kerberos authentication. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

Batch Mechanism

Attribute

BatchMechanism (BM)

Purpose

Determines the mechanism that is used to execute batch operations.

Valid Values

1 | 2 | 3

Behavior

If set to 1 (SingleRowInsert), the driver executes an insert statement for each row contained in a parameter array. Specify this value if you are experiencing out-of-memory errors when performing batch inserts.

If set to 2 (MultiRowInsert), the driver attempts to execute a single insert statement for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the driver, the driver executes multiple statements. Specify this value for substantial performance gains over 1 (SingleRowInsert) when performing batch inserts.

If set to 3 (Copy), the driver uses the Greenplum COPY command to insert rows into the target table. Specify this value for substantial performance gains over 1 (SingleRowInsert) when performing batch inserts.

Default

1 (SingleRowInsert)

Notes

- Batch Mechanism determines the mechanism used to perform batch inserts only. For update and delete batch operations, the driver uses the native batch mechanism to handle the request.
- When `BatchMechanism=3`, substantial performance gains can be made. However, the following limitations apply:
 - Individual update counts are not returned. However, the total number of inserted rows is returned upon the execution of a batch operation.
 - The entire batch insert is ATOMIC. If any issues are encountered, the entire operation fails and no rows are inserted.

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 879 for details.

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 879

Connection Reset

Attribute

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 879

Connection Retry Count

Attribute

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x, the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay (CRD)

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to x, the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (EncryptionMethod=1 | 6). When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, behavior is determined by the setting of the EncryptionMethod connection option.

Valid Values

cryptographic_protocol [[, *cryptographic_protocol*]...]

where:

cryptographic_protocol

is one of the following cryptographic protocols:

TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Notes

- This option is ignored if Encryption Method is set to 0 - No Encryption.
- Consult your database administrator concerning the data encryption settings of your server.

Default

TLSv1.2, TLSv1.1, TLSv1

GUI Tab

[Security tab](#)

See also

[Encryption Method](#) on page 852

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Progress\DataDirect\Connect64_for_ODBC_71\  
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLibName](#) on page 872

Data Source Name

Attribute

`DataSourceName` (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or `odbc.ini` file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database Name

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where:

database_name

is the name of a valid database.

Default

None

GUI Tab

[General tab](#)

Description

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable Keyset Cursors**Attribute**

EnableKeysetCursors (EKC)

Purpose

Determines whether the driver emulates keyset cursors to provide scrollable keyset cursors to an ODBC application.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver emulates keyset cursors.

If set to 0 (Disabled), the driver does not emulate keyset cursors. If an application requests a keyset cursor and this option is set to 0, the driver uses a static cursor and returns a message that a different value was used.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Enable SQLDescribeParam**Attribute**

EnableDescribeParam (EDP)

Purpose

Determines whether SQLDescribeParam returns the Datatype, ParameterSize, DecimalDigits, and Nullable information for parameters in a prepared statement.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), SQLDescribeParam returns the Datatype, ParameterSize, DecimalDigits, and Nullable information for parameters in a prepared statement.

If set to 0 (Disabled), the driver does not support SQLDescribeParam and returns the message: Driver does not support this function.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Encryption Method

Attribute

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server. If the specified encryption method is not supported by the database server, the connection fails and the driver returns an error.

Valid Values

0 | 1 | 6

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using the SSL protocols specified in the Crypto Protocol Version connection option.

If set to 6 (RequestSSL), the login request and data are encrypted using SSL if the server is configured for SSL. If the server is not configured for SSL, an unencrypted connection is established. The SSL protocol used is determined by the setting of the Crypto Protocol Version connection option.

Notes

- This connection option can affect performance.
- Supported by Greenplum 4.2 and higher.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

See also

[Performance Considerations](#) on page 1038

Extended Column MetaData

Attribute

ExtendedColumnMetaData (ECMD)

Purpose

Determines how the driver returns column metadata when using SQLDescribeCol and SQLColAttribute.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), SQLDescribeCol returns the actual values for Data Type, Column Size, Decimal Digits, and Nullable. SQLColAttribute returns the actual values for:

- SQL_DESC_CATALOG_NAME: *catalog_name*
- SQL_DESC_TABLE_NAME: *table_name*
- SQL_DESC_BASE_COLUMN_NAME: *base_column_name*
- SQL_DESC_LOCAL_TYPE_NAME: *local_type_name*
- SQL_DESC_NULLABLE: *nullable*
- SQL_DESC_AUTO_UNIQUE_VALUE: *auto_unique_value*

If set to 0 (Disabled), SQLDescribeCol returns the Data Type, Column Size, and Decimal Digits for the column. The value SQL_NULLABLE_UNKNOWN is returned for Nullable. SQLColAttribute returns the following attribute values:

- SQL_DESC_CATALOG_NAME: empty string
- SQL_DESC_TABLE_NAME: empty string
- SQL_DESC_BASE_COLUMN_NAME: empty string
- SQL_DESC_LOCAL_TYPE_NAME: empty string
- SQL_DESC_NULLABLE: SQL_NULLABLE_UNKNOWN
- SQL_DESC_AUTO_UNIQUE_VALUE: SQL_FALSE

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Failover Granularity

Attribute

FailoverGranularity (FG)

Purpose

Determines whether the driver fails the entire failover process or continues with the process if errors occur while trying to reestablish a lost connection.

This option applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select).

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 (Non-Atomic), the driver continues with the failover process and posts any errors on the statement on which they occur.

If set to 1 (Atomic) the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

If set to 2 (Atomic Including Repositioning), the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

If set to 3 (Disable Integrity Check), the driver does not verify that the rows that were restored during the failover process match the original rows. This value applies only when Failover Mode is set to 2 (Select).

Default

0 (Non-Atomic)

GUI Tab

[Failover tab](#)

Failover Mode

Attribute

FailoverMode (FM)

Purpose

Specifies the type of failover method the driver uses.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Connection), the driver provides failover protection for new connections only.

If set to 1 (Extended Connection), the driver provides failover protection for new and lost connections, but not any work in progress.

If set to 2 (Select), the driver provides failover protection for new and lost connections. In addition, it preserves the state of work performed by the last Select statement executed.

Default

0 (Connection)

GUI Tab

[Failover tab](#)

Failover Preconnect**Attribute**

FailoverPreconnect (FP)

Purpose

Specifies whether the driver tries to connect to the primary and an alternate server at the same time.

This attribute applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select) and at least one alternate server is specified.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver tries to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection. This value provides the best performance, but your application typically experiences a short wait while the failover connection is attempted.

If set to 1 (Enabled), the driver tries to connect to the primary and an alternate server at the same time. This can be useful if your application is time-sensitive and cannot absorb the wait for the failover connection to succeed.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Fetch RefCursors**Attribute**

FetchRefCursors (FRC)

Purpose

Determines whether the driver returns refcursors from stored procedures as results sets.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns refcursors from stored procedures as result sets. The driver fetches all the data from the refcursor and then closes the refcursor. If a stored procedure returns multiple refcursors, the driver generates multiple result sets, one for each refcursor returned.

If set to 0 (Disabled), the driver returns the cursor name for refcursors. The application must fetch the actual data from the refcursor using the cursor name and must close the cursor before additional processing can be done on the statement. The application must close the cursor regardless of whether it actually fetches data from the cursor.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

Fetch TSWTZ as Timestamp

Attribute

FetchTSWTZasTimestamp (FTSWTZAT)

Purpose

Determines whether the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_TYPE_TIMESTAMP or SQL_VARCHAR.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the timestamp with time zone data type as the ODBC type SQL_TYPE_TIMESTAMP. The time zone information in the fetched value is truncated. Use this value if your application needs to process values the same way as TIMESTAMP columns.

If set to 0 (Disabled), the driver returns column values with the timestamp with time zone data type as the ODBC data type SQL_VARCHAR. Use this value if your application requires the time zone information in the fetched value.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Fetch TWFS as Time

Attribute

FetchTWFSasTime (FTWFSAT)

Purpose

Determines whether the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIME` or `SQL_TYPE_TIMESTAMP`.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIME`. The fractional seconds portion of the value is truncated.

If set to 0 (Disabled), the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIMESTAMP`. The fractional seconds portion of the value is preserved. Time columns are not searchable when they are described and fetched as timestamp.

Notes

- When returning time with fractional seconds data as `SQL_TYPE_TIMESTAMP`, the Year, Month and Day parts of the timestamp must be set to zero.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

GSS Client Library

Attribute

GSSClient (GSSC)

Purpose

The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC).

The driver uses the path defined by the `PATH` environment variable for loading the specified client library.

Valid Values

`native` | `client_library`

where:

client_library

is a GSS client library installed on the client.

Behavior

If set to *client_library*, the driver uses the specified GSS client library.

Note: For MIT Kerberos distributions, you must provide a full path to the MIT Library. For example, the 64-bit version for Windows would use the following value: C:\Program Files\MIT\Kerberos\bin\gssapi64.dll.

If set to *native*, the driver uses the GSS client for Windows Kerberos. All other users must provide the full path to the library name.

Default

native

GUI Tab

[Security tab](#)

Host Name

Attribute

HostName (HOST)

Purpose

The name or the IP address of the server to which you want to connect.

Valid Values

server_name | *IP_address*

where:

server_name

is the name of the server to which you want to connect.

IP_address

is the IP address of the server to which you want to connect.

The IP address must be in IPv4 format.

Example

If your network supports named servers, you can specify a server name such as `MainServer`. Or, you can specify an IP address such as `199.226.224.34..`

Default

None

GUI Tab

[General tab](#)

Host Name In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1 or 6) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

`host_name` | `#SERVERNAME#`

where:

`host_name`

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If `host_name` is specified, the driver examines the `subjectAltName` values included in the certificate. If a `dnsName` value is present in the `subjectAltName` values, then the driver compares the value specified for Host Name In Certificate with the `dnsName` value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the `dnsName` value.

If no `subjectAltName` values exist or a `dnsName` value is not in the list of `subjectAltName` values, then the driver compares the value specified for Host Name In Certificate with the `commonName` part of the Subject name in the certificate. The `commonName` typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the `commonName`. If multiple `commonName` parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the `commonName` parts.

If `#SERVERNAME#` is specified, the driver compares the host server name specified as part of a data source or connection string to the `dnsName` or the `commonName` value.

Default

None

Notes

- Supported by Greenplum 4.2 and higher.

GUI Tab

[Security tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Initialization String

Attribute

InitializationString (IS)

Purpose

A SQL command that is issued immediately after connecting to the database to manage session settings.

Valid Values

SQL_command

where:

SQL_command

is a valid SQL command that is supported by the database.

Example

To set the date format on every connection, specify:

```
Set DateStyle='ISO, MDY'
```

Notes

- If the statement fails to execute, the connection fails and the driver reports the error returned from the server.

Default

None

GUI Tab

[Advanced tab](#)

Keyset Cursor Options

Attribute

KeysetCursorOptions (KCO)

Purpose

Determines which columns are used to comprise the keyset that the driver uses to create the initial keyset on which cursor operations are based. Greenplum does not offer a true row identifier column; the driver instead uses two hidden system columns provided by the Greenplum database, `ctid` and `gp_segment_id`. Because the database might reassign these IDs following a Vacuum operation, the driver can be configured to also include other columns to help ensure that data integrity is maintained.

Valid Values

0 | 1

Behavior

If set to 1 - RowID and Searchable Columns (Enabled), the driver uses a combination of every non-LOB column in the Select list and the `ctid` and `gp_segment_id` hidden columns to build the keyset. By adding other Select list fields to the keyset, the driver is able to indicate the row cannot be found if the IDs change following a Vacuum operation.

If set to 0 - RowID Columns (Disabled), the driver uses the `ctid` and `gp_segment_id` hidden system columns.

Notes

- This option has no effect unless the `EnableKeysetCursors` (EKC) connection option is enabled.

Default

0

GUI Tab

[Advanced tab](#)

Key Password

Attribute

KeyPassword (KP)

Purpose

Specifies the password used to access the individual keys in the keystore file when SSL is enabled (Encryption Method=1 or 6) and SSL client authentication is enabled on the database server. Keys stored in a keystore can be individually password-protected. To extract the key from the keystore, the driver must have the password of the key.

Valid Values

key_password

where:

key_password

is the password of a key in the keystore.

Default

None

Notes

- Supported by Greenplum 4.2 and higher.

GUI Tab

[Security tab](#)

Keystore

Attribute

Keystore (KS)

Purpose

The name of the directory containing the keystore file to be used when SSL is enabled (Encryption Method=1 or 6) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request. If you do not specify a directory, the current directory is used.

Valid Values

keystore_directory

where:

keystore_directory

is the location of the keystore file.

Notes

- The keystore and truststore files can be the same file.
- Supported by Greenplum 4.2 and higher.

Default

None

GUI Tab

[Security tab](#)

Keystore Password

Attribute

KeystorePassword (KSP)

Purpose

The password used to access the keystore file when SSL is enabled (Encryption Method=1 or 6) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request.

Valid Values

keystore_password

where:

keystore_password

is the password of the keystore file.

Notes

- The keystore and truststore files may be the same file; therefore, they may have the same password.
- Supported by Greenplum 4.2 and higher.

Default

None

GUI Tab

[Security tab](#)

Load Balance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

Specifies the number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Valid Values

0 | x

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to x , inactive connections are closed after the specified number of seconds passes.

Notes

- The Min Pool Size option may cause some connections to ignore this value.

Default

0

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 879

Load Balancing

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x , the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

15

GUI Tab

[Advanced tab](#)

Max Char Size

Attribute

MaxCharSize (MCS)

Purpose

Specifies the maximum size of columns of type SQL_CHAR that the driver describes through result set descriptions and catalog functions.

Valid Values

A positive integer from 1 to 10485760

When not specified, the actual size of the columns from the database is persisted to the application.

If you specify a value that is not in the specified range, the driver uses the maximum value of the SQL_CHAR data type.

Default

None. The actual size of the columns from the database is persisted to the application.

GUI Tab

[Advanced tab](#)

Max Long Varchar Size

Attribute

MaxLongVarcharSize (MLVS)

Purpose

Specifies the maximum size of columns of type SQL_LONGVARCHAR that the driver describes through result set descriptions and catalog functions.

Valid Values

A positive integer from 1 to x

where:

x

is maximum size of the SQL_LONGVARCHAR data type.

Default

None. The actual size of the columns from the database is persisted to the application.

GUI Tab

[Advanced tab](#)

Max Pool Size

Attribute

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

For example, if set to 20, the maximum number of connections allowed in the pool is 20.

Notes

- This connection option can affect performance.

Default

100

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 879

Min Pool Size

Attribute

MinPoolSize (MNPS)

Purpose

The minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Valid Values

0 | x

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

If set to x , the start-up number of connections in the pool is 5 in addition to the current existing connection.

Notes

- This connection option can affect performance.

Example

If set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Default

0

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 879

Max Varchar Size

Attribute

MaxVarcharSize (MVS)

Purpose

Specifies the maximum size of columns of type SQL_VARCHAR that the driver describes through result set descriptions and catalog functions.

Valid Values

A positive integer from 1 to x

where:

x

is maximum size of the SQL_VARCHAR data type.

Default

None. The actual size of the columns from the database is persisted to the application.

GUI Tab

[Advanced tab](#)

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Purpose

The port number of the server listener.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your database administrator for the correct number.

Default

5432

GUI Tab

[General tab](#)

Query Timeout

Attribute

QueryTimeout (QT)

Purpose

The number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the SQL_ATTR_QUERY_TIMEOUT statement attribute on the SQLSetStmtAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to -1, the query does not time out. The driver silently ignores the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to 0, the query does not time out, but the driver responds to the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to x , all queries time out after the specified number of seconds unless the application overrides this value by setting the SQL_ATTR_QUERY_TIMEOUT attribute.

Default

0

GUI Tab

[Advanced tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where `x` is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Advanced tab](#)

Service Principal Name

Attribute

ServicePrincipalName (SPN)

Purpose

The service principal name to be used by driver for Kerberos authentication.

Valid Values

servicePrincipalName

where:

servicePrincipalName

is a valid service principal name.

If unspecified, the value of the Network Address option is used as the service principal name.

Notes

- If Authentication Method is set to 0, the value of the Service Principal Name option is ignored.

Default

None

GUI Tab

Security tab

SSLLibName

Attribute

SSLLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\ODBC_71\
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the CryptoLibName and SSLLibName options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:


```
SSLLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 848

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Transaction Error Behavior

Attribute

TransactionErrorBehavior (TEB)

Purpose

Determines how the driver handles errors that occur within a transaction. When an error occurs in a transaction, the Greenplum server does not allow any operations on the connection except for rolling back the transaction.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (None), the driver does not roll back the transaction when an error occurs. The application must handle the error and roll back the transaction. Any operation on the statement other than a rollback results in an error.

If set to 1 (Rollback Transaction), the driver rolls back the transaction when an error occurs. In addition to the original error message, the driver posts an error message indicating that the transaction has been rolled back.

Default

1 (Rollback Transaction)

GUI Tab

[Advanced tab](#)

Truststore

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used.

Valid Values

truststore_directory\filename

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Notes

- The truststore and keystore files may be the same file.
- Supported by Greenplum 4.2 and higher.

Default

None

GUI Tab

[Security tab](#)

Truststore Password

Attribute

TruststorePassword (TSP)

Purpose

Specifies the password that is used to access the truststore file when SSL is enabled (Encryption Method=1 or 6) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

- The truststore and keystore files may be the same file; therefore, they may have the same password.
- Supported by Greenplum 4.2 and higher.

Default

None

GUI Tab

[Security tab](#)

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Advanced tab](#)

[Security tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Notes

- Truststore information is specified using the TrustStore and TrustStorePassword options.
- Supported by Greenplum 4.2 and higher.

Default

1 (Enabled)

GUI Tab

[Security tab](#)

Unbounded Numeric Precision

Attribute

UnboundedNumericPrecision (UNP)

Purpose

Specifies the precision for unbounded NUMERIC columns when they are described within the column, parameter, result set, or table metadata. Executing aggregation operations (for example, sum or avg) on bounded NUMERIC columns often results in the server returning the aggregate column as an unbounded NUMERIC column. When this occurs, this option defines the precision for the aggregate column.

Valid Values

A positive integer from 1 to 1000

Default

1000

GUI Tab

[Advanced tab](#)

Unbounded Numeric Scale

Attribute

UnboundedNumericScale (UNS)

Purpose

Specifies the scale for unbounded NUMERIC columns when they are described within the column, parameter, result set, or table metadata. Executing aggregation operations (for example, sum or avg) on bounded NUMERIC columns often results in the server returning the aggregate column as an unbounded NUMERIC column. When this occurs, this option defines the scale for the aggregate column.

Valid Values

A positive integer from 1 to 998

Notes

- The driver returns the scale specified in this option for the affected columns regardless of the number of decimal digits in a value. If a value contains fewer digits to the right of the decimal than the specified scale, the remaining digits are automatically returned as padded 0s. For example, if your scale is set to 6 and your value is 22.22, the value returned is 22.220000.

Default

6

GUI Tab

[Advanced tab](#)

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Important: When Kerberos is enabled, if the database user name differs from the domain user name, you are required to pass the database user name via the User Name (LogonID) option in the datasource or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Security tab](#)

XML Describe Type

Attribute

XMLDescribeType (XDT)

Purpose

The SQL data type that is returned by SQLGetTypeInfo for the XML data type.

See [Using the XML Data Type](#) on page 881 for further information about the XML data type.

Valid Values

-4 | -10

Behavior

If set to -4 (SQL_LONGVARIABLE), the driver uses the description SQL_LONGVARIABLE for columns that are defined as the XML data type.

If set to -10 (SQL_WLONGVARIABLE), the driver uses the description SQL_WLONGVARIABLE for columns that are defined as the XML data type.

Default

-10

GUI Tab

[Advanced tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Batch Mechanism (BatchMechanism): Setting BatchMechanism to 2 (MultiRowInsert) or 3 (Copy) provides significant performance gains over 1 (SingleRowInsert) when executing batch inserts:

- When `BatchMechanism=2`, the driver executes a single insert statement for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the server, the driver executes multiple statements.
- When `BatchMechanism=3`, the driver uses the Greenplum COPY command to insert rows into the target table.

Connection Pooling (Pooling): If you enable the driver to use connection pooling, you can set additional options that affect performance:

- **Load Balance Timeout (LoadBalanceTimeout):** You can define how long to keep connections in the pool. The time that a connection was last used is compared to the current time and, if the timespan exceeds the value of the Load Balance Timeout option, the connection is destroyed. The Min Pool Size option can cause some connections to ignore this value.
- **Connection Reset (ConnectionReset):** Resetting a re-used connection to the initial configuration settings impacts performance negatively because the connection must issue additional commands to the server.
- **Max Pool Size (MaxPoolSize):** Setting the maximum number of connections that the pool can contain too low might cause delays while waiting for a connection to become available. Setting the number too high wastes resources.
- **Min Pool Size (MinPoolSize):** A connection pool is created when the first connection with a unique connection string connects to the database. The pool is populated with connections up to the minimum pool size, if one has been specified. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Failover Mode (FailoverMode): Although high availability that replays queries after a failure provides increased levels of protection, it can adversely affect performance because of increased overhead.

Data Types

The following table shows how the Greenplum data types are mapped to the standard ODBC data types.

Table 70: Greenplum Data Types

Greenplum	ODBC
Bigint	SQL_BIGINT
Bigserial	SQL_BIGINT
Bit ⁷⁰	SQL_BIT
Bit varying	SQL_VARBINARY
Boolean	SQL_BIT
Bytea	SQL_VARBINARY
Character	SQL_CHAR
Character varying	SQL_VARCHAR
Citext ^{71,72}	SQL_LONGVARCHAR
Date	SQL_TYPE_DATE
Double Precision	SQL_DOUBLE
Float	SQL_REAL
Integer	SQL_INTEGER
Money	SQL_DOUBLE
Name	SQL_VARCHAR
Numeric ⁷³	SQL_NUMERIC
Real	SQL_REAL
Serial	SQL_INTEGER
Smallint	SQL_SMALLINT
Text	SQL_LONGVARCHAR
Time ⁷⁴	SQL_TYPE_TIME
Timestamp	SQL_TYPE_TIMESTAMP

⁷⁰ Bit maps to SQL_BIT when the length for the bit is 1. If the length is greater than 1, the driver maps the column to SQL_BINARY.

⁷¹ The Citext data type behaves the same as the Text data type, except that it is case-insensitive. The select operations performed on Citext columns return case-insensitive results.

⁷² Supported for Greenplum versions 5.3 and higher.

⁷³ Numeric maps to SQL_NUMERIC if the precision of the Numeric is less than or equal to 38. If the precision is greater than 38, the driver maps the column to SQL_VARCHAR.

⁷⁴ Time mapping changes based on the setting of the Fetch TWFS as Time option.

Greenplum	ODBC
Timestamp with timezone ⁷⁵	SQL_VARCHAR
Tinyint	SQL_SMALLINT
Wchar	SQL_CHAR
Wvarchar	SQL_VARCHAR

See [Retrieving Data Type Information](#) on page 72 for more information about data types.

Using the XML Data Type

By default, Greenplum returns XML data to the driver encoded as UTF-8. To avoid data loss, an application must bind XML data as SQL_C_WCHAR. The driver then returns the data as either UTF-8 or UTF-16, depending on platform and application settings. If the application binds XML data as SQL_C_CHAR, the driver converts it to the client character encoding, possibly causing data loss or corruption. To prevent any conversion of XML data, the application must set the attribute [XMLDescribeType](#) (XDT) to SQL_LONGVARIABLE (-10) and bind the data as SQL_C_BINARY.

Unicode Support

The Greenplum Wire Protocol driver automatically determines whether the Greenplum database is a Unicode database.

Advanced Features

The driver supports the following advanced features:

- Failover
- Connection pooling
- Security

Failover

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Connection Pooling

The driver supports connection pooling and its related connection options. Connection pooling connection options are located on the [Pooling tab](#) of the driver Setup dialog box. See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling and its implementation.

⁷⁵ Timestamp with timezone mapping changes based on the setting of the Fetch TSWTZ as Timestamp option.

Security

The driver supports authentication and encryption. Security connection options are located on the [Security tab](#) of the driver Setup dialog box. See [Using Security](#) on page 89 for a general description of security and its implementation.

User-defined Functions' Results

Greenplum provides functionality to create user-defined functions. Greenplum does not define a call mechanism for invoking a user-defined function. User-defined functions must be invoked via a SQL statement.

For example, a function defined as:

```
CREATE table foo (intcol int, varcharcol varchar(123))
CREATE or REPLACE FUNCTION insertFoo
(IN idVal int, IN nameVal varchar) RETURNS void
AS $$
    insert into foo values ($1, $2);
$$
LANGUAGE SQL;
```

must be invoked natively as:

```
SELECT * FROM insertFoo(100, 'Mark')
```

even though the function does not return a value or results. The Select SQL statement returns a result set that has one column named insertFoo and no row data.

The Greenplum Wire Protocol driver supports invoking user-defined functions using the ODBC call Escape. The previously described function can be invoked using:

```
{call insertFoo(100, 'Mark')}
```

Greenplum functions return data from functions as a result set. If multiple output parameters are specified, the values for the output parameters are returned as columns in the result set. For example, the function defined as:

```
CREATE or REPLACE FUNCTION addValues(in v1 int, in v2 int)
RETURNS int
AS $$
    SELECT $1 + $2;
$$
LANGUAGE SQL;
```

returns a result set with a single column of type SQL_INTEGER, whereas the function defined as:

```
CREATE or REPLACE FUNCTION selectFooRow2
(IN idVal int, OUT id int, OUT name varchar)
AS $$
    select intcol, varcharcol from foo where intcol = $1;
$$
LANGUAGE SQL
```

returns a result set that contains two columns, a SQL_INTEGER id column and a SQL_VARCHAR name column.

In addition, when calling Greenplum functions that contain output parameters, the native syntax requires that the output parameter values be omitted from the function call. This, in addition to output parameter values being returned as a result set, makes the Greenplum behavior of calling functions different from most other databases.

The Greenplum Wire Protocol driver provides a mechanism that makes the invoking of functions more consistent with how other databases behave. In particular, the Greenplum Wire Protocol driver allows parameter markers for output parameters to be specified in the function argument list when the Escape call is used. The driver allows buffers to be bound to these output parameters. When the function is executed, the output parameters are removed from the argument list sent to the server. The driver extracts the output parameter values from the result set returned by the server and updates the bound output parameter buffers with those values. For example, the function `selectFooRow2` described previously can be invoked as:

```
sql = L"{call selectFooRow2(?, ?, ?)}";
retVal = SQLPrepare(hPrepStmt, sql, SQL_NTS);
retVal = SQLBindParameter(
    hPrepStmt, 1, SQL_PARAM_INPUT, SQL_C_LONG,
    SQL_INTEGER, 0, 0, &idBuf, 0, &idInd);
retVal = SQLBindParameter(
    hPrepStmt, 2, SQL_PARAM_OUTPUT, SQL_C_LONG,
    SQL_INTEGER, 0, 0, &idBuf2, 4, &idInd2);
retVal = SQLBindParameter(
    hPrepStmt, 3, SQL_PARAM_INPUT, SQL_C_WCHAR,
    SQL_VARCHAR, 30, 0, &nameBuf, 123, &nameInd);
retVal = SQLExecute(hPrepStmt);
```

The values of the `id` and `name` output parameters are returned in the `idBuf2` and `nameBuf` buffers.

If output parameters are bound to a function call, the driver returns the output parameters in the bound buffers. An error is returned if the number of output parameters bound when the function is executed is less than the number of output parameters defined in the function. If no output parameters are bound to a function call, the driver returns the output parameters as a result set.

Greenplum can also return results from a function as a refcursor. There can be, at most, one refcursor per result; however, a function can return multiple results where each result is a refcursor. A connection option defines how the driver handles refcursors. See [Fetch RefCursors](#) on page 855 for details about this option.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

Greenplum supports isolation level 0 (read uncommitted), level 1 (read committed), 2 (Repeatable read), and level 3 (serializable). Greenplum supports record-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the core SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

In addition, the following functions are supported:

- `SQLColumnPrivileges`

- SQLDescribeParam (if EnableDescribeParam=1)
- SQLForeignKeys
- SQLTablePrivileges

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The Greenplum Wire Protocol driver supports multiple connections and multiple statements per connection.

Using Arrays of Parameters

Greenplum supports returning a set of output parameters or return values, but no ODBC standard method exists for returning arrays of output parameters or return values. If the call Escape is used to invoke a function that returns a set of output parameters and buffers are bound for those output parameters, the Greenplum Wire Protocol driver places the first set of output parameters in the bound buffers. If no output parameters are bound for functions that return a set of results or output parameters, the driver returns a result set with a row for each set of output parameters.

Limitations for Pivotal HAWQ Users

The following are known limitations for using Pivotal HAWQ:

- No support for updates
- No support for deletes
- No support for stored procedures

For a more complete listing of known issues and limitations for your version of Pivotal HAWQ, refer to the Greenplum Pivotal HAWQ user documentation.

The Impala Wire Protocol Driver

The DataDirect Connect XE *for* ODBC and DataDirect Connect64 XE *for* ODBC for Impala™ Wire Protocol driver support Cloudera Impala database servers.

The Impala server is compatible with data stored in a variety of file formats. In addition, Impala can work with data stored in other systems through the use of storage handlers. The file formats and storage handlers in use should work seamlessly with the Impala driver. The Impala Wire Protocol driver is formally certified with the most prevalent file formats and storage handlers:

- Certified File Formats:
 - Parquet
 - TextFile
- Certified Storage Handlers:
 - HBase

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect driver for the file name of the driver.

Driver Requirements

The driver has no client requirements.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 1051 and [Connection Option Descriptions for Apache Hive](#) on page 1052 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX odbc.ini File

UNIX[®] On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). You can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions for Apache Hive](#) on page 1052 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Impala)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section. **UNIX**® On UNIX and Linux, data sources are stored in the `odbc.ini` file.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.



To configure a data source for Impala:

1. Start the ODBC Administrator:
 - On Windows, start the ODBC Administrator by selecting its icon from the Datadirect Connect program group.
2. Select a tab:
 - **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
 - **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
 - **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.
If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 91: General tab

ODBC Impala Wire Protocol Driver Setup

General | Advanced | Security | About

Data Source Name: Help

Description:

Host Name:

Port Number:

Database Name:

Test Connect | OK | Cancel | Apply

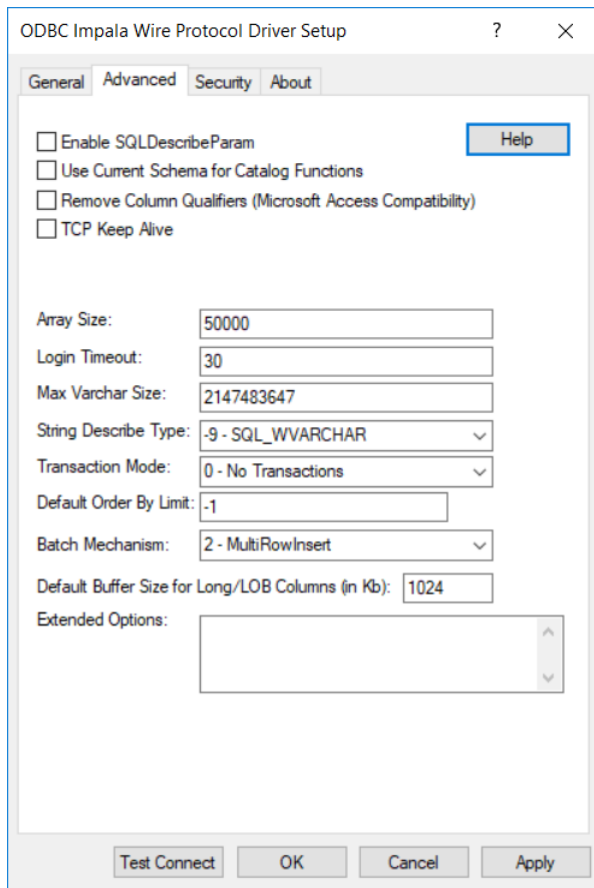
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 899	None
Description on page 902	None
Host Name on page 904	None
Port Number on page 909	None
Database on page 900	default

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 92: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Enable SQLDescribeParam on page 902	Disabled
Use Current Schema for Catalog Functions on page 916	Disabled
Remove Column Qualifiers on page 910	Disabled
TCP Keep Alive on page 914	Disabled
Array Size on page 895	50000
Login Timeout on page 907	30
Max Varchar Size on page 908	2147483647
String Describe Type on page 913	-9 - SQL_WVARCHAR
Transaction Mode on page 914	0 - No Transactions

Connection Options: Advanced	Default
Default Order By Limit on page 901	-1 (Disabled)
Batch Mechanism on page 897	2 - MultiRowInsert
Default Buffer Size for Long/LOB Columns (in Kb) on page 900	1024

Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1; UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.

- Optionally, click the **Security** tab to specify security settings.

Figure 93: Security tab

ODBC Impala Wire Protocol Driver Setup

General Advanced **Security** About

Help

Authentication

Authentication Method: 0 - User ID/Password

User Name:

Proxy User:

Service Principal Name:

GSS Client Library: native

Encryption

Encryption Method: 0 - No Encryption

Crypto Protocol Version

TLSv1.2 TLSv1.1 TLSv1

SSLv3 SSLv2

Validate Server Certificate

Trust Store:

Trust Store Password:

Key Store:

Key Store Password:

Key Password:

Host Name In Certificate:

Test Connect OK Cancel Apply

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options:Security	Default
Authentication Method on page 896	0 - User ID/Password
User Name on page 917	None
Proxy User on page 910	None
Service Principal Name on page 911	None
GSS Client Library on page 903	native
Encryption Method on page 903	0 - No Encryption
Crypto Protocol Version on page 897	TLSv1.2,TLSv1.1,TLSv1,SSLv3
Validate Server Certificate on page 917	1 - Enabled
Truststore on page 915	None
Trust Store Password on page 915	None
Key Store on page 906	None
Keystore Password on page 907	None
Key Password on page 905	None
Host Name In Certificate on page 905	None

6. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Hive\)](#) on page 1051) for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

7. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{driver_name}][:attribute=value[;attribute=value]...]
```

[Connection Option Descriptions](#) on page 892 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Hive is:

```
DSN=Accounting;UID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Impala.dsn;UID=JOHN;PWD=XYZZY
```

A DSN-less connection string must provide all necessary connection information:

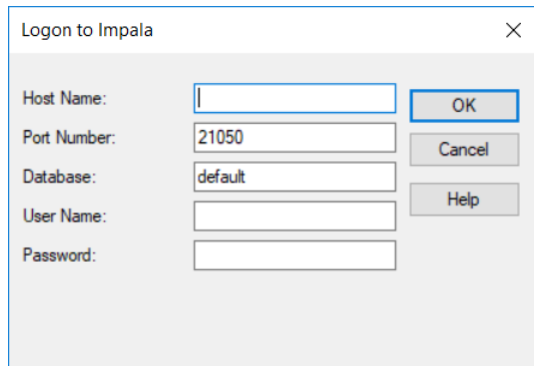
```
DRIVER={DataDirect 7.1 Impala};HOST=server1;PORT=21050;UID=JOHN;PWD=XYZZY;
```

Using a Logon Dialog Box (Impala)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Note: A user name and password are not required to connect to Impala.

Figure 94: Logon to Impala dialog box



Note: To configure a standard connection, complete the first two fields and skip to Step 4.

In this dialog box, provide the following information:

1. In the Host field, type either the name or the IP address of the server to which you want to connect.
2. In the Port Number field, type the port number that your Impala server is listening on. Check with your Impala administrator for the correct number.
3. In the Database field, type the name of the Impala database. The database must exist, or the connection attempt will fail.
4. If required, type your logon ID In the User Name field.
5. If required, type your password In the Password field.
6. Click **OK** to log on to the Impala server you specified and to update the values in the Registry.

Note: The User Name and Password fields are not used at this time to connect to Impala Server.

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name. For example:

Array Size

Attribute

ArraySize (AS)

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

The following table lists the connection string attributes supported by the driver.

Table 71: Attribute Names for the Driver for Impala

Attribute (Short Name)	Default
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
ArraySize (AS)	50000
AuthenticationMethod (AM)	0 (User ID/Password)
BatchMechanism (BM)	2 (MultiRowInsert)
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Database (DB)	default
DataSourceName (DSN)	None
DefaultLongDataBuffLen (DLDBL)	1024
DefaultOrderByLimit (DOBL)	-1 (Disabled)
Description (n/a)	None
EnableDescribeParam (EDP)	0 (Disabled)
EncryptionMethod (EM)	0 (No Encryption)
GSSClient (GSSC)	native
HostName (HOST)	None
HostNameInCertificate (HNIC)	None
KeepAlive (KA)	0 (Disabled)
KeyPassword (KP)	None
Keystore (KS)	None
KeystorePassword (KSP)	None
LoginTimeout (LT)	30
LogonID (UID)	None
MaxVarcharSize (MVS)	2147483647
Password (PWD)	None
PortNumber (PORT)	None

Attribute (Short Name)	Default
ProxyUser (PU)	None
RemoveColumnQualifiers (RCQ)	0 (Disabled)
ServicePrincipalName (SPN)	None
SSLibName (SLN)	Empty String
StringDescribeType (SDT)	-9 - SQL_WVARCHAR
TransactionMode (TM)	0 (No Transactions)
Truststore (TS)	None
TruststorePassword (TSP)	None
UseCurrentSchema (UCS)	0 (Disabled)
ValidateServerCertificate (VSC)	1 (Enabled)

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

`latest | openssl_version_number[[,openssl_version_number]...]`

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to `openssl_version_number`, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the `CryptoLibName` and `SSLLibName` options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

`1.1.1,1.0.2`

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

[Advanced tab](#)

Array Size

Attribute

`ArraySize (AS)`

Purpose

The number of cells the driver retrieves from a server for a fetch. When executing a fetch, the driver divides the value specified by the number columns in a particular table to determine the number of rows to retrieve. By determining the fetch size based on the number of cells, the driver can avoid out of memory errors when fetching from tables containing a large number of columns while continuing to provide improved performance when fetching from tables containing a small number of columns.

Valid Values

`x`

where:

`x`

is a positive integer specifying the number of cells the driver retrieves for a fetch.

Notes

- You can improve performance by increasing the value specified for this option; however, if the number of cells specified exceeds the available buffer memory for the server, an out of memory error will be returned. If you receive this error, decrease the value specified until fetches are successfully executed.
- This connection option can affect performance.

Default

50000

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 918 for details.

Authentication Method

Attribute

AuthenticationMethod (AM)

Purpose

Specifies the method the driver uses to authenticate the user to the server when a connection is established. If the specified authentication method is not supported by the database server, the connection fails and the driver generates an error.

Valid Values

0 | 4 | -1

Behavior

If set to 0 (User ID/Password), the driver sends the user ID in clear text and an encrypted password to the server for authentication.

If set to 4 (Kerberos Authentication), the driver uses Kerberos authentication. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

If set to -1 (No Authentication), the driver sends the user ID and password in clear text to the server for authentication.

Default

0 (User ID/Password)

GUI Tab

[Security tab](#)

Batch Mechanism

Attribute

BatchMechanism (BM)

Purpose

Determines the mechanism that is used to execute batch operations.

Valid Values

1 | 2

Behavior

If set to 1 (SingleInsert), the driver executes an insert statement for each row contained in a parameter array. Select this setting if you are experiencing out-of-memory errors when performing batch inserts.

If set to 2 (MultiRowInsert), the driver attempts to execute a single insert statement for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the driver, the driver executes multiple statements. Select this setting for substantial performance gains when performing batch inserts.

Default

2 (MultiRowInsert)

Notes

- This connection option can affect performance.

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 918 for details.

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (EncryptionMethod=1). When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, behavior is determined by the setting of the EncryptionMethod connection option.

Valid Values

cryptographic_protocol [[, *cryptographic_protocol*]...]

where:

cryptographic_protocol

is one of the following cryptographic protocols:

TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Notes

- This option is ignored if Encryption Method is set to 0 - No Encryption.
- Consult your database administrator concerning the data encryption settings of your server.

Default

```
TLSv1.2, TLSv1.1, TLSv1
```

GUI Tab

[Security tab](#)

See also

[Encryption Method](#) on page 903

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Progress\DataDirect\Connect64_for_ODBC_71\
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLibName](#) on page 912

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or `odbc.ini` file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database

Attribute

Database (DB)

Purpose

Specifies the name of the Impala database. The database must exist, or the connection attempt will fail.

Valid Values

database_name

where:

database_name

is the name of the Impala database.

Default

default

GUI Tab

[General tab](#)

Default Buffer Size for Long/LOB Columns (in Kb)

Attribute

DefaultLongDataBuffLen (DLDBL)

Purpose

The maximum length of data (in KB) the driver can fetch from long columns in a single round trip and the maximum length of data that the driver can send using the SQL_DATA_AT_EXEC parameter.

Valid Values

An integer in multiples of 1024

The value must be in multiples of 1024 (for example, 1024, 2048). You need to increase the default value if the total size of any Long data exceeds 1 MB. This value is multiplied by 1024 to determine the total maximum length of fetched data. For example, if you enter a value of 2048, the maximum length of data would be 1024 x 2048, or 2097152 (2 MB).

Notes

- This connection option can affect performance.

Default

1024

GUI tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 918 for details.

Default Order By Limit

Attribute

DefaultOrderByLimit (DOBL)

Purpose

Specifies the maximum number of rows returned when a SQL statement containing an ORDER BY clause is executed. Cloudera Impala requires statements containing the ORDER BY clause to limit the number of rows returned. This option allows these statements to return a result set without specifying a limit in the application..

Valid Values

-1 | x

where:

x

is a positive integer that represents maximum number of rows returned.

Behavior

If set to -1 (disabled), there is no default limit the number of rows returned by a statement containing an ORDER BY clause. The application must limit the number of rows returned by SQL statements that contain an ORDER BY clause, or Impala will return an error.

If set to x , the number of rows returned by a SQL statement containing an ORDER BY clause are limited to the specified number of rows for the session. To override this value, specify a new value in a LIMIT clause in the statement that is being executed.

Default

-1 (Disabled)

GUI Tab

[Advanced tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable SQLDescribeParam

Attribute

EnableDescribeParam (EDP)

Purpose

Determines whether the driver uses the SQLDescribeParam function, which describes parameters as a data type of SQL_VARCHAR with a length of 255 for statements.

Valid Values

0 | 1

Behavior

If set to 1 (enabled), the SQLDescribeParam function describes parameters as a data type of SQL_VARCHAR with a length of 255 for statements.

If set to 0 (disabled), the SQLDescribeParam function returns the standard ODBC error IM001.

Default

0 (Disabled)

GUI tab

[Advanced tab](#)

Encryption Method

Attribute

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server. If the specified encryption method is not supported by the database server, the connection fails and the driver returns an error.

Valid Values

0 | 1

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using the SSL protocols specified in the Crypto Protocol Version connection option.

Notes

- This connection option can affect performance.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

See also

[Performance Considerations](#) on page 918

GSS Client Library

Attribute

GSSClient (GSSC)

Purpose

The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC).

The driver uses the path defined by the PATH environment variable for loading the specified client library.

Valid Values

native | *client_library*

where:

client_library

is a GSS client library installed on the client.

Behavior

If set to *client_library*, the driver uses the specified GSS client library.

Note: For MIT Kerberos distributions, you must provide a full path to the MIT Library. For example, the 64-bit version for Windows would use the following value: C:\Program Files\MIT\Kerberos\bin\gssapi64.dll.

If set to *native*, the driver uses the GSS client for Windows Kerberos. All other users must provide the full path to the library name.

Default

native

GUI Tab

[Security tab](#)

Host Name

Attribute

HostName (HOST)

Purpose

The name or the IP address of the server to which you want to connect.

Valid Values

host_name | *IP_address*

where:

hostname

is the name of the Impala server to which you want to connect

IP_address

is the IP address of the server to which you want to connect.

Default

None

GUI Tab

[General tab](#)

Host Name In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

`host_name` | `#SERVERNAME#`

where:

`host_name`

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If `host_name` is specified, the driver examines the `subjectAltName` values included in the certificate. If a `dnsName` value is present in the `subjectAltName` values, then the driver compares the value specified for Host Name In Certificate with the `dnsName` value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the `dnsName` value.

If no `subjectAltName` values exist or a `dnsName` value is not in the list of `subjectAltName` values, then the driver compares the value specified for Host Name In Certificate with the `commonName` part of the Subject name in the certificate. The `commonName` typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the `commonName`. If multiple `commonName` parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the `commonName` parts.

If `#SERVERNAME#` is specified, the driver compares the host server name specified as part of a data source or connection string to the `dnsName` or the `commonName` value.

Default

None

GUI Tab

[Security tab](#)

Key Password

Attribute

KeyPassword (KP)

Purpose

Specifies the password used to access the individual keys in the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. Keys stored in a keystore can be individually password-protected. To extract the key from the keystore, the driver must have the password of the key.

Valid Values

key_password

where:

key_password

is the password of a key in the keystore.

Default

None

GUI Tab

[Security tab](#)

Key Store

Attribute

Keystore (KS)

Purpose

The name of the directory containing the keystore file to be used when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request. If you do not specify a directory, the current directory is used.

Valid Values

keystore_directory

where:

keystore_directory

is the location of the keystore file.

Notes

- The keystore and truststore files can be the same file.

Default

None

GUI Tab

[Security tab](#)

Keystore Password

Attribute

KeystorePassword (KSP)

Purpose

The password used to access the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request.

Valid Values

keystore_password

where:

keystore_password

is the password of the keystore file.

Notes

- The keystore and truststore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

Specifies the number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x , the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

30

GUI Tab

[Advanced tab](#)

Max Varchar Size

Attribute

MaxVarcharSize (MVS)

Purpose

Specifies the maximum size of columns of type SQL_VARCHAR that the driver describes through result set descriptions and catalog functions.

Valid Values

A positive integer from 255 to x

where:

x

is maximum size of the SQL_VARCHAR data type.

Default

2147483647

GUI Tab

[Advanced tab](#)

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Note: Not used to log on to Impala at this time.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Purpose

Specifies the port number of the server listener.

Valid Values

port_number

where:

port_number

is the port number of the server listener. Check with your database administrator for the correct number.

Default

None

GUI Tab

[General tab](#)

Proxy User

Attribute

ProxyUser (PU)

Purpose

Specifies the UserID used for impersonation. When impersonation is enabled on the server, this value determines your identity and access rights to files when executing queries. If no value is provided for this option or if impersonation is disabled, you will execute queries as the user who initiated the process.

Impersonation provides a method for administrators to control access to data. Administrators set access rights to files by using HDFS and directory permissions on the server.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

Notes

- Supported in Impala 1.2 and higher.

GUI Tab

[Security tab](#)

Remove Column Qualifiers

Attribute

RemoveColumnQualifiers (RCQ)

Purpose

Specifies whether the driver removes 3-part column qualifiers and replaces them with alias.column qualifiers. Microsoft Access executes a Select statement using this syntax when an index is specified on a linked table.

Valid Values

0 | 1

Behavior

If set to 1 (enabled) the driver removes 3-part column qualifiers and replaces them with alias.column qualifiers. Column qualifiers are Microsoft Access compatible in this setting.

If set to 0, the driver does not modify the request.

Notes

- When using the driver with Microsoft Access in creating a linked table, it is highly recommended that you do not specify an index. Specifying an index causes Access to execute a Select statement for each row, which results in very slow performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 918 for details.

Service Principal Name

Attribute

ServicePrincipalName (SPN)

Purpose

The service principal name to be used by driver for Kerberos authentication.

Valid Values

ServicePrincipalName

where:

ServicePrincipalName

is the three-part service principal name registered with the key distribution center (KDC).

You must specify the service principal name using the following format:

Service_Name/Fully_Qualified_Domain_Name@REALM.COM

where:

Service_Name

is the name of the service hosting the instance. For example, `impala`.

The server automatically generates the service name. Refer to Cloudera Impala documentation for additional information.

Fully_Qualified_Domain_Name

is the fully qualified domain name of the host machine. For example, `yourserver.example.com`.

REALM.COM

is the domain name of the host machine. This part of the value must be specified in upper-case characters. For example, *EXAMPLE.COM*.

Example

The following is an example of a valid service principal name:

```
impala/yourserver.example.com@EXAMPLE.COM
```

Notes

- If unspecified, the value of the Network Address option is used as the service principal name.
- If Authentication Method is set to 0 or -1, the value of the Service Principal Name option is ignored.

Default

None

GUI Tab

[Security tab](#)

SSLibName

Attribute

SSLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\ODBC_71\  
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
SSLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 898

String Describe Type

Attribute

StringDescribeType (SDT)

Purpose

Specifies whether all string columns are described as `SQL_VARCHAR`. This connection option affects `SQL_Columns`, `SQLDescribeCol`, `SQLColAttributes`, etc. It does not affect `SQLGetTypeInfo`.

Valid Values

-10 | -9

Behavior

If set to -10 (`SQL_WLONGVARCHAR`), all string columns are described as `SQL_WLONGVARCHAR`.

If set to -9 (`SQL_WVARCHAR`), all string columns are described as `SQL_WVARCHAR`.

Default

-9 - `SQL_WVARCHAR`

GUI Tab

[Advanced tab](#)

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Transaction Mode

Attribute

TransactionMode (TM)

Purpose

Specifies how the driver handles manual transactions.

Valid Values

0 | 1

Behavior

If set to 1 (Ignore), the data source does not support transactions and the driver always operates in auto-commit mode. Calls to set the driver to manual commit mode and to commit transactions are ignored. Calls to rollback a transaction cause the driver to return an error indicating that no transaction is started. Metadata indicates that the driver supports transactions and the ReadUncommitted transaction isolation level.

If set to 0 (No Transactions), the data source and the driver do not support transactions. Metadata indicates that the driver does not support transactions.

Default

0 (No Transactions)

GUI Tab

[Advanced tab](#)

Trust Store Password

Attribute

TruststorePassword (TSP)

Purpose

Specifies the password that is used to access the truststore file when SSL is enabled (`Encryption Method=1`) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

- The truststore and keystore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Truststore

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (`Encryption Method=1`) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used.

Valid Values

truststore_directory\filename

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Notes

- The truststore and keystore files may be the same file.

Default

None

GUI Tab

[Security tab](#)

Use Current Schema for Catalog Functions

Attribute

UseCurrentSchema (UCS)

Purpose

Specifies whether results are restricted to the tables and views in the current schema if a catalog function call is made without specifying a schema or if the schema is specified as the wildcard character %. Restricting results to the tables and views in the current schema improves performance of catalog calls that do not specify a schema.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), results of catalog function calls are restricted to the tables and views in the current schema.

If set to 0 (Disabled), results of catalog function calls are not restricted.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 918 for details.

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database.

Valid Values

N/A

GUI Tab

[Security tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Notes

- Truststore information is specified using the TrustStore and TrustStorePassword options.

Default

1 (Enabled)

GUI Tab[Security tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Array Size (ArraySize): To improve throughput, consider increasing the value of Array Size. By increasing the value of Array Size, you increase the number of rows the driver will retrieve from the server for a fetch. In turn, increasing the number of rows that the driver can retrieve reduces the number, and expense, of network round trips. For example, if an application attempts to fetch 100,000 rows, it is more efficient for the driver to retrieve 2000 rows over the course of 50 round trips than to retrieve 500 rows over the course of 200 round trips. Note that improved throughput does come at the expense of increased demands on memory and slower response time. Furthermore, if the fetch size exceeds the available buffer memory of the server, an out of memory error is returned when attempting to execute a fetch. If you receive this error, decrease the value specified until fetches are successfully executed.

Batch Mechanism (BatchMechanism): If your application does not require individual update counts for each statement or parameter set in the batch, then BatchMechanism should be set to 2 (MultiRowInsert). Unlike the native batch mechanism, the multi-row insert mechanism only returns the total number of update counts for batch inserts. Therefore, setting BatchMechanism to MultiRowInsert offers substantial performance gains when performing batch inserts.

Default Buffer Size for Long/LOB Columns (DefaultLongDataBuffLen): To improve performance when your application fetches images, pictures, or long text or binary data, a buffer size can be set to accommodate the maximum size of the data. The buffer size should only be large enough to accommodate the maximum amount of data retrieved; otherwise, performance is reduced by transferring large amounts of data into an oversized buffer. If your application retrieves more than 1 MB of data, the buffer size should be increased accordingly.

Encryption Method (EncryptionMethod): Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) that is required to encrypt and decrypt data.

Use Current Schema for Catalog Functions (UseCurrentSchema): If your application needs to access database objects owned only by the current user, then performance can be improved. In this case, the Use Current Schema for Catalog Functions option must be enabled. When this option is enabled, the driver returns only database objects owned by the current user when executing catalog functions. Calls to catalog functions are optimized by grouping queries. Enabling this option is equivalent to passing the Logon ID used on the connect as the Schema Name argument to catalog functions.

Data Types

The following table shows how the Impala data types are mapped to the standard ODBC data types.

Table 72: Impala Data Types

Impala	ODBC
Bigint	SQL_BIGINT
Boolean	SQL_BIT

Impala	ODBC
Char ⁷⁶	SQL_CHAR
Decimal ⁷⁶	SQL_DECIMAL
Double	SQL_DOUBLE
Float	SQL_REAL
Int	SQL_INTEGER
Smallint	SQL_SMALLINT
String ⁷⁷	SQL_WVARCHAR, SQL_WLONGVARCHAR ⁷⁸
Timestamp	SQL_TYPE_TIMESTAMP
Tinyint	SQL_TINYINT
Varchar ⁷⁶	SQL_VARCHAR

Advanced Features

The driver supports the following advanced features:

- Security

Security

The driver supports authentication and data encryption. Security connection options are located on the [Security tab](#) of the driver Setup dialog box. See [Using Security](#) on page 89 for a general description of security and its implementation. The following security-related information is specific to the Impala Wire Protocol driver.

Apache Sentry

Apache Sentry is a modular security system that enables administrators to control access to data and metadata stored on an Apache Hadoop cluster by defining user roles and permissions. The driver works transparently with Sentry and does not require further configuration. To use Sentry, Kerberos authentication must be enabled, and a Kerberos logon must be provided at connection.

Note: When establishing a connection, the driver attempts to set the user's default database to be used for the session. In environments using Sentry, the user must be granted access to this database; otherwise, the connection will fail.

⁷⁶ Supported with Impala 2.0 and higher.

⁷⁸ If the StringDescribeType property is set to *wvarchar* (the default), this data type maps to WVARCHAR. If set to *wlongvarchar*, this data type maps to WLONGVARCHAR.

⁷⁷ Maximum of 2 GB

For more information, refer to the Cloudera Impala documentation at <http://cloudera.com/content/cloudera/en/documentation.html>.

Materialized Views

Impala supports views but purely as logical objects with no associated storage. As such, there is no support for materialized views in Impala; therefore, the driver does not support materialized views.

Stored Procedures

Impala has no concept of stored procedures. Therefore, they are not supported in the driver.

Unicode Support

The driver is fully Unicode enabled. On UNIX and Linux platforms, the driver supports both UTF-8 and UTF-16. On Windows platforms, the Impala driver supports UCS-2/UTF-16 only.

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Isolation and Lock Levels Supported

Impala supports isolation level 0 (read uncommitted).

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the core SQL grammar and the Cloudera Impala query language.

Refer to the [Impala SQL Language Reference](#) for information about using Impala SQL.

Also, see [SQL Functionality for the Impala Wire Protocol driver](#).

ODBC Conformance Level

The driver supports ODBC API Conformance Level 1.

Note: SQLCancel and SQLTransact execute successfully but perform no functions.

Note: SQLStatistics always returns an empty result set.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Using Arrays of Parameters

By default, the driver supports multi-row inserts for parameterized arrays. For a multi-row insert, the driver attempts to execute a single insert for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the driver, the driver executes multiple statements. This behavior provides substantial performance gains for batch inserts.

The driver modifies the native statement to perform a multi-row insert. Therefore, the default multi-row insert behavior may not be desirable in all scenarios. You can disable this behavior by setting the Batch Mechanism connection option to 1 (SingleInsert). When `BatchMechanism=1`, Impala's single row insert is used to execute batch operations, and an insert statement is executed for each row contained in a parameter array.

Refer to "Designing ODBC applications for performance optimization" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Limitations on Cloudera Impala Functionality

The following restrictions are based on using Cloudera Impala version 1.0:

- No support for transactions
- No support for canceling a running query
- No support for row-level inserts, updates, or deletes
- No support for cursors or scrollable cursors
- No support for prepared statements

For a more complete listing of known issues and limitations for your version of Impala, refer to the Cloudera Impala user documentation:

<http://www.cloudera.com/content/support/en/documentation.html>

Note: Note that Cloudera Impala is not designed for OLTP workloads and does not offer real-time queries or row-level updates. Instead, Impala is designed for batch type jobs over large data sets with high latency. This means that queries such as "SELECT * FROM mytable" return quickly. However, other SELECT statements are much slower.

The Salesforce Driver

Note: This section documents the features and functionality of the 7.1 version of the driver. For the current version of the driver, visit Progress DataDirect Connectors Documentation page:

<https://docs.progress.com/datadirect-connectors/page/DataDirect-Connectors-by-data-source.html>

The DataDirect Connect XE for ODBC and DataDirect Connect64 XE for ODBC Salesforce driver (the Salesforce driver) supports the standard SQL query language to fetch, insert, update, and delete data from Salesforce.com and Force.com. The Salesforce Web Service API is fully supported, including any Salesforce App Exchange application.

Note: You can query the SYSTEM_REMOTE_SESSIONS system table to get the version of the Web Service API the driver supports.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The driver translates the SQL statements provided by the application to Salesforce queries (SOQL) and Web service calls. See [SQL Statements and Extensions for the Salesforce Driver](#) for the SQL statements that the driver supports.

The driver maps the Salesforce data model into a set of related relational tables. The mapping representation is stored in XML files external to the driver. This allows the sharing of map files among different client machines.

The driver uses a client-side data cache for improved performance. You can define rules that specify which data to cache on the client as well as when the cached data becomes invalid and needs to be refreshed (see [Client-Side Caches](#) on page 979 for details).

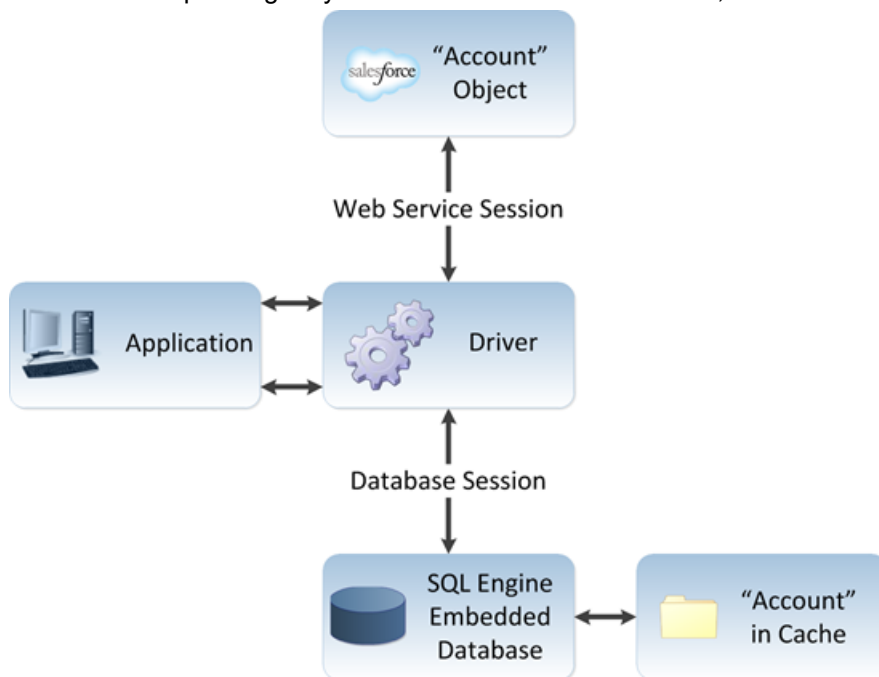
The Salesforce driver can be used with industry standard tools, which means that developers can leverage their existing SQL knowledge instead of having to learn the Salesforce query language and APIs. Examples include the following tools:

SAP Crystal Reports	Cognos
Microsoft Access	Microsoft Excel
Oracle Gateway	Oracle Business Intelligence (OBIEE)
SAS/Access for ODBC	SQL Server Linked Server
Tableau	

The driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect XE product for the file name of the driver.

The following figure shows the different components of an environment that uses the Salesforce driver to access Salesforce. Depending on your license from Salesforce.com, the driver could instead connect to Force.com.



When an application connects to Salesforce through the driver, connectivity to Salesforce is real-time, out of the box. In the background, the driver establishes a Web service session with the Salesforce instance and a database session that opens an embedded database. The application can establish multiple sessions with Salesforce; however, additional Web service and database sessions are always opened in the ratio of one database session per Web service session. The database session maintains the object-to-relational table mapping. In addition, it maintains cached tables and local tables, as well as maintaining views. See [Database Configuration File](#) on page 988 for more details.

On Windows, the SQL Engine can be run within the same process space as the ODBC application, or it can be run as a separate process. Some applications may experience problems loading the JVM required for the SQL Engine because the process exceeds the available heap space. If your application experiences problems loading the JVM, you can configure the Salesforce driver to run in a separate 32-bit process within its own JVM. See [Configuring the SQL Engine Server](#) on page 992 for more information.

Salesforce has certain standard objects that always exist, even if they do not contain anything. Salesforce administrators can also create custom objects using the Salesforce browser interface. The relationships among these objects are tabular, like those among the tables in a database. The Salesforce driver recognizes the relationships among both standard Salesforce objects and custom objects and can access, create, and update both. The relationships among objects can be reported through the ODBC `SQLForeignKeys` and `SQLPrimaryKeys` functions. The driver leverages Salesforce mechanisms for joining data, minimizing the amount of data that needs to be fetched over the network.

Driver Requirements

The driver requires a Java Virtual Machine (JVM): Java SE 8 or higher.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 938 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux `odbc.ini` File

UNIX® On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detailed information about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, `odbc.ini`).

You can configure and modify data sources directly by editing the `odbc.ini` file and storing default connection values there. See [Configuration Through the System Information \(`odbc.ini`\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions](#) on page 939 lists driver connection string attributes that must be used in the `odbc.ini` file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Salesforce)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Salesforce data source:

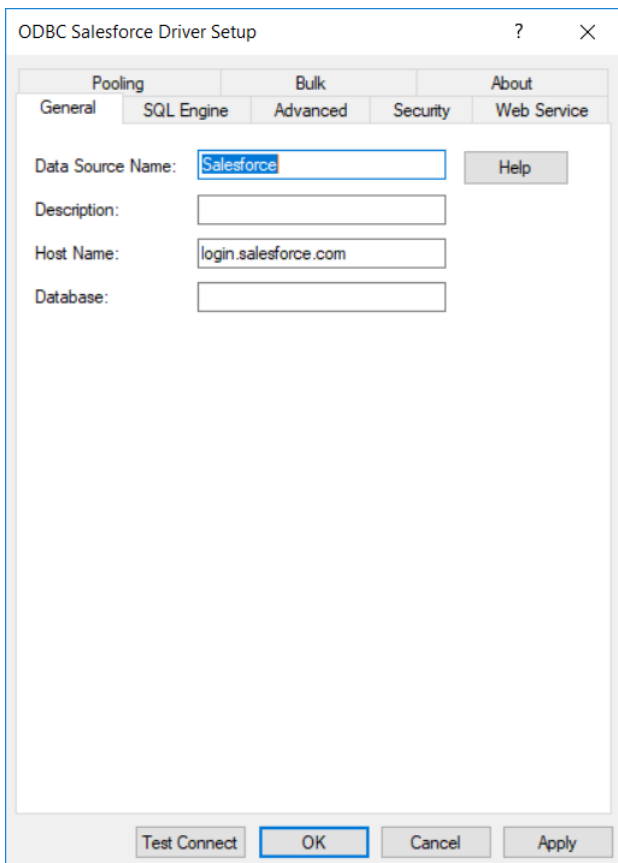
1. Start the ODBC Administrator by selecting its icon from the DataDirect Connect program group.
2. Select a tab:
 - **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
 - **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 95: General tab



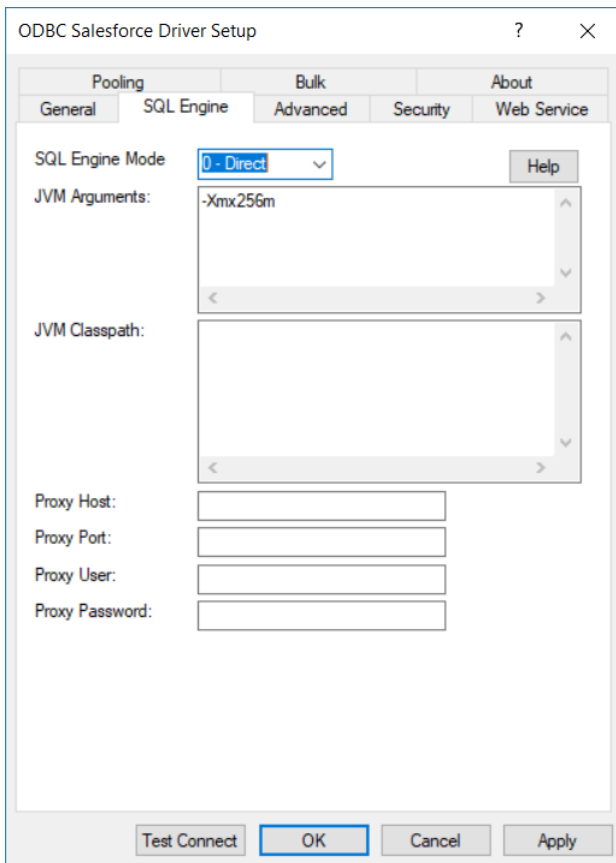
Note: : The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 951	None
Description on page 952	None
Host Name on page 955	login.salesforce.com
Database on page 951	None

3. Optionally, click the **SQL Engine** tab to specify additional data source settings.

Figure 96: SQL Engine tab



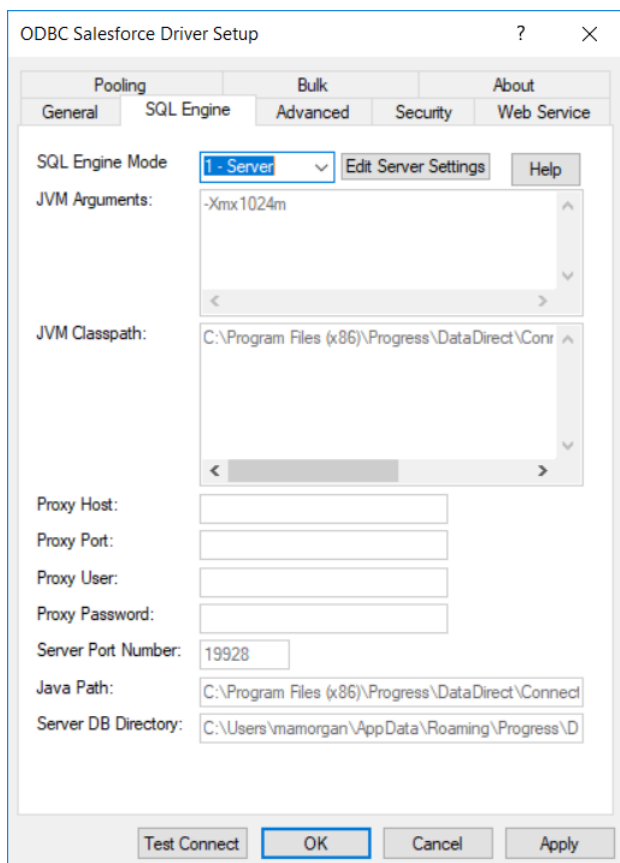
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: SQL Engine	Default (Direct Mode)
JVM Arguments on page 956	-Xmx256m
JVM Classpath on page 957	None
Proxy Host on page 962	None
Proxy Port on page 964	None
Proxy User on page 964	None
Proxy Password on page 963	None

By default, the Salesforce driver operates in direct mode, with both the driver and its SQL engine running in the ODBC application's address space. Some applications may experience problems loading the JVM because the process exceeds the available heap space. You can configure the Salesforce driver to operate in server mode. Server mode allows the driver to connect to a 32-bit SQL engine JVM running as a separate service.

To run the driver in server mode, set **SQL Engine Mode** to 1-Server. Additional configuration settings appear on the SQL Engine tab. All fields except SQL Engine Mode are read only.

Figure 97: SQL Engine tab: SQL Engine Mode



To define the settings for server mode, click `Edit Server Settings` from the SQL Engine tab. The SQL Engine Service Setup dialog box appears.

Note: You must be an administrator to modify the server mode settings. Otherwise, the `Edit Server Settings` button does not appear on the SQL Engine tab.

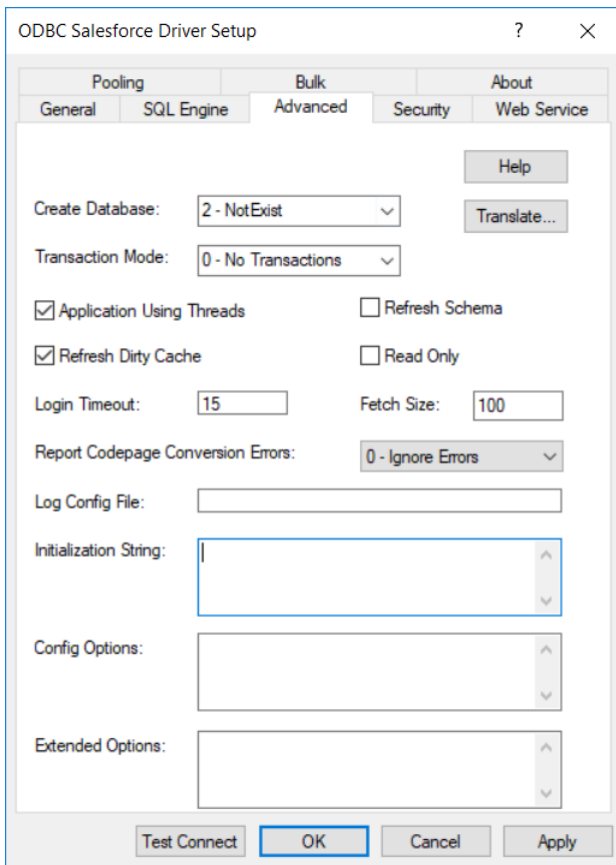
You use the SQL Engine Service Setup dialog box to configure server mode and to start or stop the service. See [Configuring Server Mode](#) on page 992 for detailed information.

Configuration Options: SQL Engine Service	Default
JVM Arguments on page 956	None
JVM Classpath on page 957	<code>install_dir\javallib\sforce.jar</code>
Proxy Host on page 962	None
Proxy Port on page 964	None
Proxy User on page 964	None
Proxy Password on page 963	None

Configuration Options: SQL Engine Service	Default
Server Port Number on page 969	19928
Java Path	Fully qualified path to the JVM executable (java.exe)
Server DB Directory	Path of the working directory for the SQL Engine service
Services	Progress DataDirect Salesforce SQL Engine

4. Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 98: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Create Database on page 950	2 -NotExist
Transaction Mode on page 971	0 - No Transactions
Application Using Threads on page 941	Enabled
Refresh Dirty Cache on page 966	Enabled

Connection Options: Advanced	Default
Refresh Schema on page 966	Disabled
Read Only on page 964	Disabled
Login Timeout on page 960	15
Fetch Size on page 953	100
Report Codepage Conversion Errors on page 967	0 - Ignore Errors
Log Config File on page 959	None
Initialization String on page 955	None
Config Options on page 945	None

Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
CreateDB=2;UndocumentedOption1=value [;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

5. Optionally, click the **Security** tab to specify security data source settings.

Figure 99: Security tab

The screenshot shows the 'ODBC Salesforce Driver Setup' dialog box with the 'Security' tab selected. The dialog contains the following elements:

- Buttons at the top: Pooling, Bulk, About, General, SQL Engine, Advanced, Security (selected), Web Service.
- A 'Help' button located above the input fields.
- Input fields: 'User Name:', 'Logon Domain:', and 'Security Token:'.
- Buttons at the bottom: 'Test Connect', 'OK', 'Cancel', and 'Apply'.

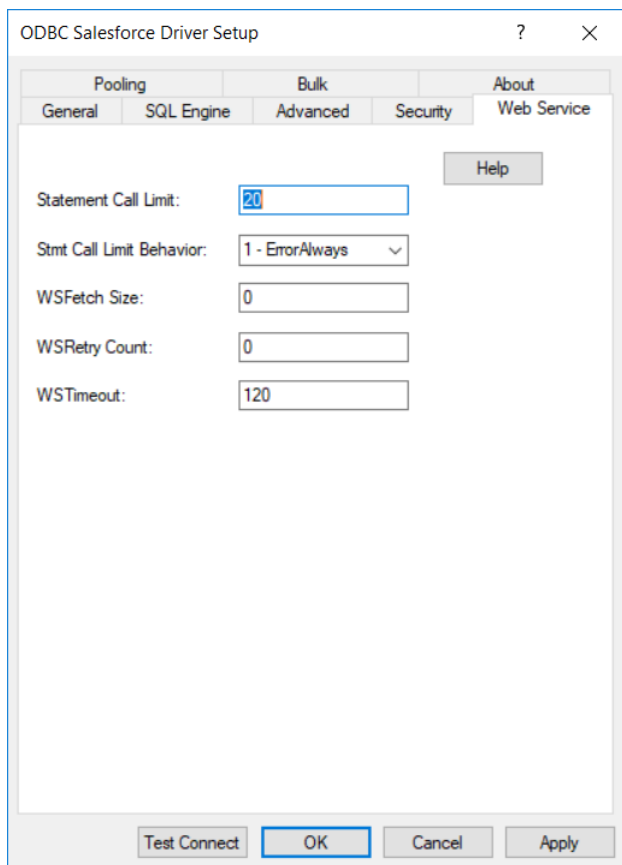
See [Using Security](#) on page 89 for a general description of authentication and encryption and their configuration requirements.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Security	Default
User on page 989	None
Logon Domain on page 959	None
Security Token on page 968	None

6. Optionally, click the **Web Service** tab to specify additional data source settings.

Figure 100: Web Service tab

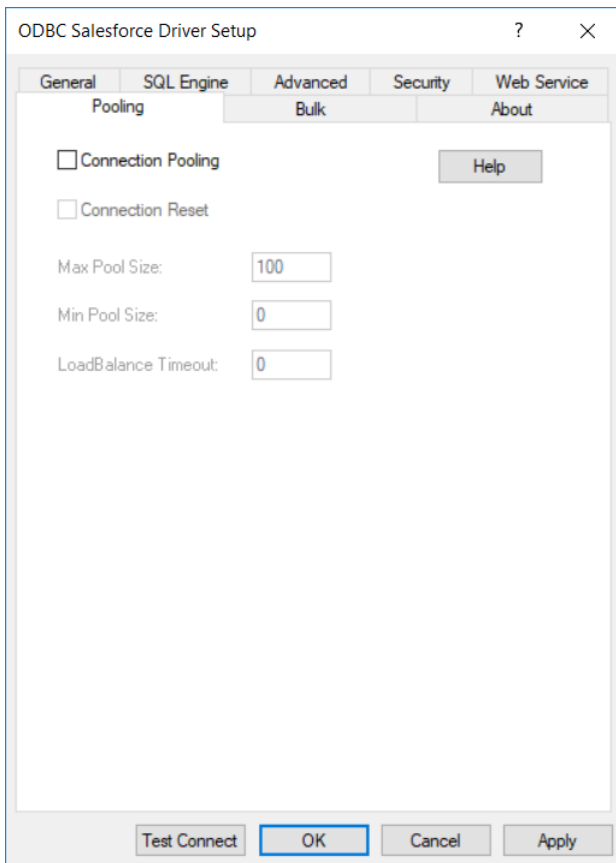


On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Web Service	Default
Statement Call Limit on page 970	20
Statement Call Limit Behavior on page 971	1- ErrorAlways
WSFetchSize on page 972	0
WSRetry Count on page 973	0
WSTimeout on page 974	120

7. Optionally, click the **Pooling** tab to specify connection pooling data source settings.

Figure 101: Pooling tab



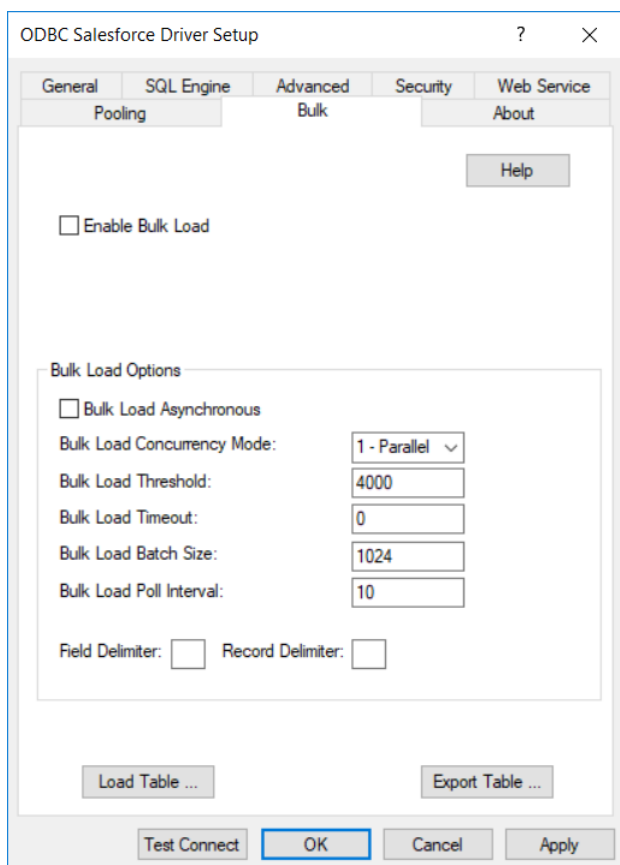
See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 949	Disabled
Connection Reset on page 949	Disabled
Max Pool Size on page 960	100
Min Pool Size on page 961	0
LoadBalance Timeout on page 958	0

8. Optionally, click the **Bulk** tab to specify DataDirect Bulk Load data source settings.

Figure 102: Bulk tab



See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect Bulk Load.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Bulk	Default
Enable Bulk Load on page 953	Disabled
Bulk Load Asynchronous on page 942	Disabled
Bulk Load Concurrency Mode on page 943	1 (Parallel)
Bulk Load Threshold on page 944	4000
Bulk Load Timeout on page 945	0
Bulk Load Batch Size on page 942	1024
Bulk Load Poll Interval on page 943	10
Field Delimiter on page 954	None
Record Delimiter on page 965	None

If your application is already coded to use parameter array batch functionality, you can leverage DataDirect Bulk Load features through the Enable Bulk Load connection option. Enabling this option automatically converts the parameter array batch operation to use the database bulk load protocol.

If you are not using parameter array batch functionality, you can export data to a bulk load data file, verify the metadata of the bulk load configuration file against the structure of the target table, and bulk load data to a table. Use the following steps to accomplish these tasks.

- a) To export data from a table to a bulk load data file, click **Export Table** from the Bulk tab. The Export Table dialog box appears.

Figure 103: ODBC Salesforce Export Table Driver Setup dialog box

Both a bulk data file and a bulk configuration file are produced by exporting a table. The configuration file has the same name as the data file, but with an XML extension. See [Using DataDirect Bulk Load](#) on page 101 for details about these files.

The bulk export operation can create a log file and can also export to external files. See [External Overflow Files](#) on page 108 for more information. The export operation can be configured such that if any errors or warnings occur:

- The operation always completes.
- The operation always terminates.
- The operation terminates after a certain threshold of warnings or errors is exceeded.

Table Name: A string that specifies the name of the source database table and, optionally, the columns containing the data to be exported. The driver uses the table name in the FROM clause of a `SELECT * FROM tablename` SQL statement. If you want to only export certain columns from your Salesforce table, then you can enter a SELECT statement in this field using the format:

```
(SELECT column1, column2, ... FROM tablename)
```

For example, to export data from the Salesforce ACCOUNT table excluding some of the audit columns, enter the following SQL in the Table Name field:

```
(SELECT SYS_NAME, TYPE, BILLINGSTREET, BILLINGCITY, BILLINGSTATE, BILLINGPOSTALCODE,
BILLINGCOUNTRY, SHIPPINGSTREET, SHIPPINGCITY, SHIPPINGSTATE, SHIPPINGPOSTALCODE,
SHIPPINGCOUNTRY, PHONE, FAX, WEBSITE, INDUSTRY, ANNUALREVENUE, NUMBEROFEMPLOYEES,
DESCRIPTION FROM ACCOUNT)
```

Export Filename: A string that specifies the path (relative or absolute) and file of the bulk load data file to which the data is to be exported. It also specifies the file name of the bulk configuration file. The file name must be the fully qualified path to the bulk data file. These files must not already exist; if one of both of them already exists, an error is returned.

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. The log file is created if it does not exist. The file name must be the fully qualified path to the log file. Events logged to this file are:

- Total number of rows fetched
- A message for each row that failed to export
- Total number of rows that failed to export
- Total number of rows successfully exported

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not supply a value for Log Filename, no log file is created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

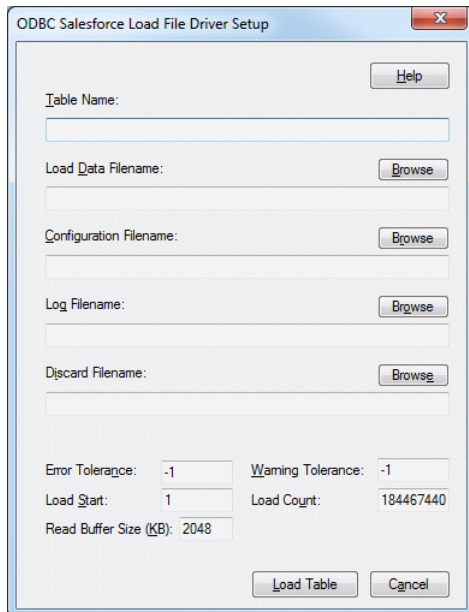
Code Page: A value that specifies the code page value to which the driver must convert all data for storage in the bulk data file. See [Character Set Conversions](#) on page 108 for more information.

The default value on Windows is the current code page of the machine.

Click **Export Table** to connect to the database and export data to the bulk data file or click **Cancel**.

- b) To bulk load data from the bulk data file to a database table, click **Load Table** from the Bulk tab. The Load File dialog box appears.

Figure 104: ODBC Salesforce Load File Driver Setup dialog box



The load operation can create a log file and can also create a discard file that contains rows rejected during the load. The discard file is in the same format as the bulk load data file. After fixing reported issues in the discard file, the bulk load can be reissued using the discard file as the bulk load data file.

The export operation can be configured such that if any errors or warnings occur:

- The operation always completes.
- The operation always terminates.
- The operation terminates after a certain threshold of warnings or errors is exceeded.

If a load fails, the Load Start and Load Count options can be used to control which rows are loaded when a load is restarted after a failure.

Table Name: A string that specifies the name of the target database table and, optionally, the columns into which the data is loaded.

The fields defined in the load data file must have the same ordering of the fields defined in the Salesforce destination table. Because Salesforce defines additional audit columns that are managed by the database, your load data file may not contain data to load into these fields.

In this case, you can specify the exact columns that you want for the data to be inserted into using a Table Name string of the format:

```
table(column1, column2, ...)
```

For example, if your load data file contains only five fields of billing data that you wanted to load into the Salesforce ACCOUNT table, then the Table Name field would contain:

```
ACCOUNT(BILLINGSTREET, BILLINGCITY, BILLINGSTATE, BILLINGPOSTALCODE,  
        BILLINGCOUNTRY)
```

Load Data Filename: A string that specifies the path (relative or absolute) and file name of the bulk data file from which the data is loaded.

Configuration Filename: A string that specifies the path (relative or absolute) and file name of the bulk configuration file.

Log Filename: A string that specifies the path (relative or absolute) and file name of the bulk log file. The file name must be the fully qualified path to the log file. Specifying a value for Log Filename creates the file if it does not already exist. Events logged to this file are:

- Total number of rows read
- Message for each row that failed to load
- Total number of rows that failed to load
- Total number of rows successfully loaded

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Log Filename, no log file is created.

Discard Filename: A string that specifies the path (relative or absolute) and file name of the bulk discard file. Any row that cannot be inserted into the database as result of bulk load is added to this file, with the last row rejected added to the end of the file.

Information about the load is written to this file, preceded by a header. Information about the next load is appended to the end of the file.

If you do not specify a value for Discard Filename, a discard file is not created.

Error Tolerance: A value that specifies the number of errors to tolerate before an operation terminates. A value of 0 indicates that no errors are tolerated; the operation fails when the first error is encountered.

The default of -1 means that an infinite number of errors is tolerated.

Load Start: A value that specifies the first row to be loaded from the data file. Rows are numbered starting with 1. For example, when Load Start is 10, the first 9 rows of the file are skipped and the first row loaded is row 10. This option can be used to restart a load after a failure.

The default value is 1.

Read Buffer Size (KB): A value that specifies the size, in KB, of the buffer that is used to read the bulk data file for a bulk load operation.

The default value is 2048.

Warning Tolerance: A value that specifies the number of warnings to tolerate before an operation terminates. A value of 0 indicates that no warnings are tolerated; the operation fails when the first warning is encountered.

The default of -1 means that an infinite number of warnings is tolerated.

Load Count: A value that specifies the number of rows to be loaded from the data file. The bulk load operation loads rows up to the value of Load Count from the file to the database. It is valid for Load Count to specify more rows than exist in the data file. The bulk load operation completes successfully when either the number of rows specified by the Load Count value has been loaded or the end of the data file is reached. This option can be used in conjunction with Load Start to restart a load after a failure.

The default value is the maximum value for SQLULEN. If set to 0, no rows are loaded.

Click **Load Table** to connect to the database and load the table or click **Cancel**.

At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Salesforce\)](#) on page 939 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.

- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.
9. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER={ {driver_name} }[;attribute=value[;attribute=value]...]
```

[Connection Option Descriptions](#) on page 939 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Salesforce for Linux/UNIX/Windows is:

```
DSN=Salesforce;UID=JOHN@MYCOMPANY.COM;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Salesforce;UID=JOHN@MYCOMPANY.COM;PWD=XYZZY
```

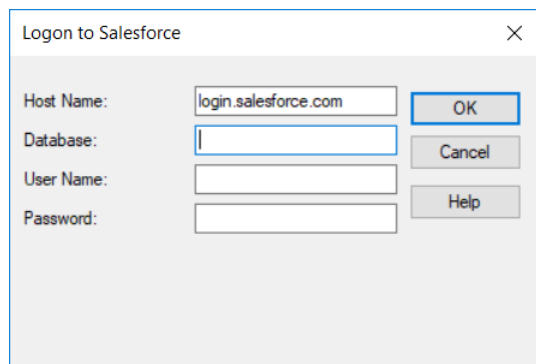
A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Salesforce};UID=JOHN@MYCOMPANY.COM;PWD=XYZZY
```

Using a Logon Dialog Box (Salesforce)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the host name has already been specified.

Figure 105: Logon to Salesforce dialog box



In this dialog box, provide the following information:

1. In the Host Name field, type the root of the Salesforce URL to which you want to connect. The default is login.salesforce.com.
2. Type the file name prefix the driver uses to create or locate the set of files that define the embedded database per connection. See [Mapping Objects to Tables](#) on page 978 for an explanation of embedded database.
3. Type your logon ID in the User Name field.
4. Type your password in the Password field.
5. Click **OK** to complete the logon.

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Salesforce driver.

Table 73: Salesforce Attribute Names

Attribute (Short Name)	Default
ApplicationUsingThreads (AUT)	1 (Enabled)
BulkLoadAsync (BLA)	0 (unselected)
BulkLoadBatchSize (BLBS)	1024

Attribute (Short Name)	Default
BulkLoadConcurrencyMode (BLCM)	1 (Parallel)
BulkLoadFieldDelimiter (BLFD)	None
BulkLoadPollInterval (BLPI)	10
BulkLoadRecordDelimiter (BLRD)	None
BulkLoadThreshold (BLTH)	4000
BulkLoadTimeout (BLTO)	0
ConfigOptions (CFGO)	Empty string
ConnectionReset (CR)	0 (Disabled)
CreateDB (CDB)	2 (NotExist)
DataSourceName (DSN)	None
Database	The file name prefix the driver uses to create or locate the set of files that define the embedded database per connection.
EnableBulkLoad (EBL)	0 (Disabled)
FetchSize (FS)	100
HostName (HOST)	Empty string (logs into the default Salesforce instance, login.salesforce.com).
InitializationString (IS)	Empty string
JVMArgs (JVMA)	-Xmx256m
JVMClasspath (JVMC)	<i>install_dir</i> \javallib\sforce.jar
LoadBalanceTimeout (LBT)	0
LogConfigFile (LCF)	Empty string
LoginTimeout (LT)	15
LogonDomain (LD)	Empty string
LogonID (UID)	None
MaxPoolSize (MXPS)	100
MinPoolSize (MNPS)	0

Attribute (Short Name)	Default
Pooling (POOL)	0 (Disabled)
ProxyHost (PXHN)	Empty string
ProxyPassword (PXPW)	Empty string
ProxyPort (PXPT)	Empty string
ProxyUser (PXUN)	Empty string
ReadOnly (RO)	0
RefreshDirtyCache (RDC)	1 (Enabled)
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
SecurityToken (STK)	Empty string
ServerPortNumber (SPN)	19928
SQLEngineMode (SEM)	0 (Direct)
StmtCallLimit (SCL)	20
StmtCallLimitBehavior (SCLB)	1 (ErrorAlways)
TransactionMode (TM)	0 (No Transactions)
WSFetchSize (WSFS)	0
WSRetryCount (WSRC)	0
WSTimeout (WST)	120

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

See also

[Performance Considerations](#) on page 974

Bulk Load Asynchronous

Attribute

BulkLoadAsync (BLA)

Purpose

Determines whether the driver treats bulk load operations as synchronous or asynchronous.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), bulk load operations are synchronous. The driver does not return from the function that invoked an operation until the operation is complete or the BulkLoadTimeout period has expired. If the operation times out, the driver returns an error.

If set to 1 (Enabled), bulk load operations are asynchronous. The driver returns from the function that invoked an operation after the operation is submitted to the server. The driver does not verify the completion status of the bulk load operation.

Default

0 (Disabled)

GUI Tab

[Bulk tab](#)

Bulk Load Batch Size

Attribute

BulkLoadBatchSize (BLBS)

Purpose

The number of rows that the driver sends to the database at a time during bulk operations. This value applies to all methods of bulk loading.

Valid Values

x

where:

x

is a positive integer that specifies the number of rows to be sent.

Default

1024

GUI Tab

[Bulk tab](#)

Bulk Load Concurrency Mode

Attribute

BulkLoadConcurrencyMode (BLCM)

Purpose

Determines whether multiple batches associated with a bulk load operation are processed by Salesforce in parallel or one at a time. See [Using DataDirect Connection Pooling](#) on page 97 for more information.

Valid Values

0 | 1

Behavior

If set to 0 (Serial), multiple batches associated with a bulk load operation are processed one at a time.

If set to 1 (Parallel), multiple batches associated with a bulk load operation are processed in parallel. The order in which the batches are processed can vary.

Default

1 (Parallel)

GUI Tab

[Bulk tab](#)

Bulk Load Poll Interval

Attribute

BulkLoadPollInterval (BLPI)

Purpose

Specifies the number of seconds the driver waits to request bulk operation status. This interval is used by the driver the first time it requests status and for all subsequent status requests. See [Using DataDirect Bulk Load](#) on page 101 for more information.

Valid Values

x

where:

x

is a positive integer that represents the number of seconds the driver waits before requesting bulk operation status.

Default

10

GUI Tab

[Bulk tab](#)

Bulk Load Threshold

Attribute

BulkLoadThreshold (BLTH)

Purpose

Determines when the driver uses bulk load for insert, update, delete, or batch operations. If the Enable Bulk Load option is set to `True` and the number of rows affected by an insert, update, delete, or batch operation exceeds the threshold specified by this option, the driver uses the Salesforce Bulk API to perform the operation.

Valid Values

0 | x

where:

x

is a positive integer that represents a threshold (number of rows).

Behavior

If set to 0, the driver always uses bulk load to execute insert, update, delete, or batch operations.

If set to x , the driver only uses bulk load if the Enable Bulk Load option is set to a value of `True` and the number of rows to be updated by an insert, update, delete, or batch operation exceeds the threshold. If the operation times out, the driver returns an error.

Notes

- If the Enable Bulk Load option is set to `false`, this option is ignored.

Default

4000

GUI Tab

[Bulk tab](#)

Bulk Load Timeout

Attribute

BulkLoadTimeout (BLTO)

Purpose

The time, in seconds, that the driver waits for a Salesforce bulk job to complete. A value of zero means there is no timeout.

Valid Values

x

where:

x

is a positive integer that represents a number of seconds the driver waits before requesting bulk operation status.

Default

0

GUI Tab

[Bulk tab](#)

Config Options

Attribute

ConfigOptions (CFGO)

Purpose

Determines how the embedded database and the mapping of the remote data model to the relational data model is configured, customized, and updated.

Notes

- This option is primarily used for initial configuration of the driver for a particular user. It is not intended for use with every connection. By default, the driver configures itself and this option is normally not needed. If Config Options is specified on a connection after the initial configuration, the values specified for Config Options must match the values specified for the initial configuration. The preferred method for setting the configuration options for a particular user is through the database configuration file. See [Database Configuration File](#) on page 988 for details.

Valid Values

```
{ key = value [ ; key = value ] }
```

where:

key

is one of the following values: `AuditColumns`, `CustomSuffix`, `MapSystemColumnNames`, `NumberFieldMapping`, or `UppercaseIdentifiers`.

The value is a set of key value pairs separated by a semicolon (;). The value must be enclosed in curly brackets. For example:

```
{AuditColumns=AuditOnly;UppercaseIdentifiers=false}
```

AuditColumns: Determines whether the driver includes audit fields, which Salesforce adds to all objects defined in a Salesforce instance, as table columns when it defines the remote data model to relational table mapping.

The audit columns added by Salesforce are:

`IsDeleted`

`CreatedById`

`CreatedDate`

`LastModifiedById`

`LastModifiedDate`

`SystemModstamp`

Valid values for `AuditColumns` are:

Value	Description
All	The driver includes the all of the audit columns and the master record id column in its table definitions.
AuditOnly	The driver adds only the audit columns in its table definitions.
MasterOnly	The driver adds only the MasterRecordId column in its table definitions.
None	The driver does not add the audit columns or the MasterRecordId column in its table definitions.

The default value for `AuditColumns` is `None`.

In a typical Salesforce instance, not all users are granted access to the Audit or MasterRecordId columns. If `AuditColumns` is set to a value other than `None` and the driver cannot include the columns requested, the connection fails and the driver generates a `SQLException` with a `SQLState` of 08001.

CustomSuffix (Custom objects and fields only): Determines whether the driver includes or strips the "`__c`" suffix from the table and column names when mapping the remote data model to the relational data model. Salesforce adds the suffix to all custom objects and fields.

Valid values for `CustomSuffix` are:

Value	Description
Include	The driver includes the "__c" suffix.
Strip	The driver strips the "__c" suffix.

The default value for CustomSuffix is `Strip`.

KeywordConflictSuffix: Specifies a string of up to five alphanumeric characters that the driver appends to any object or field name that conflicts with a SQL engine keyword. For example, if you specify `KeywordConflictSuffix=TAB`, then the driver maps the Case object in Salesforce to `CASETAB`.

Do not use a string that matches the suffix of a custom table, for example, `CASEOFICE`. If you specify `KeywordConflictSuffix=OFICE`, a name collision occurs with the Standard object `CASE` and the custom table `CASEOFICE`, or a table with a column called `CASEOFICE`. In this situation, the standard object `CASE` is returned. The custom object is ignored.

Valid values for `KeywordConflictSuffix` are:

Value	Description
string	One to five alphanumeric characters.

The default value for `KeywordConflictSuffix` is an empty string.

MapSystemColumnNames: Determines how the driver maps Salesforce system columns. Valid values for `MapSystemColumnNames` are:

Value	Description																						
0	The driver does not change the names of the Salesforce system columns.																						
1	<p>The driver changes the names of the Salesforce system columns as described in the following table:</p> <table border="1"> <thead> <tr> <th>Field Name</th> <th>Mapped Name</th> </tr> </thead> <tbody> <tr> <td>Id</td> <td>ROWID</td> </tr> <tr> <td>Name</td> <td>SYS_NAME</td> </tr> <tr> <td>IsDeleted</td> <td>SYS_ISDELETED</td> </tr> <tr> <td>CreatedDate</td> <td>SYS_CREATEDDATE</td> </tr> <tr> <td>CreatedById</td> <td>SYS_CREATEDBYID</td> </tr> <tr> <td>LastModifiedDate</td> <td>SYS_LASTMODIFIEDDATE</td> </tr> <tr> <td>LastModifiedId</td> <td>SYS_LASTMODIFIEDID</td> </tr> <tr> <td>SystemModstamp</td> <td>SYS_SYSTEMMODSTAMP</td> </tr> <tr> <td>LastActivityDate</td> <td>SYS_LASTACTIVITYDATE</td> </tr> <tr> <td>OwnerId</td> <td>SYS_OWNERID</td> </tr> </tbody> </table>	Field Name	Mapped Name	Id	ROWID	Name	SYS_NAME	IsDeleted	SYS_ISDELETED	CreatedDate	SYS_CREATEDDATE	CreatedById	SYS_CREATEDBYID	LastModifiedDate	SYS_LASTMODIFIEDDATE	LastModifiedId	SYS_LASTMODIFIEDID	SystemModstamp	SYS_SYSTEMMODSTAMP	LastActivityDate	SYS_LASTACTIVITYDATE	OwnerId	SYS_OWNERID
Field Name	Mapped Name																						
Id	ROWID																						
Name	SYS_NAME																						
IsDeleted	SYS_ISDELETED																						
CreatedDate	SYS_CREATEDDATE																						
CreatedById	SYS_CREATEDBYID																						
LastModifiedDate	SYS_LASTMODIFIEDDATE																						
LastModifiedId	SYS_LASTMODIFIEDID																						
SystemModstamp	SYS_SYSTEMMODSTAMP																						
LastActivityDate	SYS_LASTACTIVITYDATE																						
OwnerId	SYS_OWNERID																						

The default value for `MapSystemColumnNames` is 1.

NumberFieldMapping: Determines how the driver maps fields defined as NUMBER in Salesforce. The Salesforce API uses DOUBLE values to transfer data to and from NUMBER fields, which can cause problems when the precision of the NUMBER field is greater than the precision of a DOUBLE value. Rounding can occur when converting large values to and from DOUBLE. The NumberFieldMapping option allows you to map the NUMBER fields to the required SQL types based on their precision.

Valid values for NumberFieldMapping are:

Value	Description
1	The driver maps NUMBER fields with a precision of 9 or less and a scale of 0 to the INTEGER SQL type and maps all other NUMBER fields to the DOUBLE SQL type.
2	The driver maps all NUMBER fields to the DOUBLE SQL type regardless of their precision.
3	The driver maps all NUMBER fields to the DECIMAL SQL type. It can be used when the precision of the NUMBER field is greater than the precision of a DOUBLE value.

The default value for NumberFieldMapping is 1.

UppercaseIdentifiers: Defines how the driver maps identifiers. By default, the driver maps all identifier names to uppercase.

Note: Do not change the value of UppercaseIdentifiers unless the data source you are connecting to has objects with names that differ only by case.

Valid values for UppercaseIdentifiers are:

Value	Description
true	The driver maps identifiers to uppercase.
false	The driver maps identifiers to the mixed case name of the object being mapped. If mixed case identifiers are used, SQL statements must enclose those identifiers in double quotes, and the case of the identifier, must exactly match the case of the identifier name. For example, if UppercaseIdentifiers=false, to query the Account table you specify: <code>SELECT "Phone", "Website" FROM "Account"</code>

The default value for UppercaseIdentifiers is true.

Default

```
AuditColumns=None;CustomSuffix=strip;KeywordConflictSuffix=;
MapSystemColumnNames=1;NumberFieldMapping=1;UppercaseIdentifiers=true;
```

GUI Tab

[Advanced tab](#)

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- The application must be thread-enabled to use connection pooling.
- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 974

Connection Reset

Attribute

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See also

This connection option can affect performance. See [Performance Considerations](#) on page 974 for details.

Create Database

Attribute

CreateDB (CDB)

Purpose

Determines whether the driver creates a new embedded database when establishing the connection.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (No), the driver uses the current embedded database specified by DatabaseName. If one does not exist, the connection fails.

If set to 1 (ForceNew), the driver deletes the current embedded database specified by Database and creates a new one at the same location.

Warning: This causes all views, data caches, and map customizations defined in the current database to be lost.

If set to 2 (NotExist), the driver uses the current embedded database specified by DatabaseName. If one does not exist, the driver creates one.

Default

2 (NotExist)

GUI Tab

[Advanced tab](#)

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database

Attribute

Database (DBN)

Purpose

Specifies the file name prefix the driver uses to create or locate the set of files that define the Object mapping and the embedded database used by the connection. See [Mapping Objects to Tables](#) on page 978 for an explanation of embedded database.

Valid Values

prefix | *path+prefix*

where:

prefix

is the file name prefix for the embedded database. For example, if Database is set to a value of JohnQPublic, the embedded database files that are created or loaded have the form johnqpublic.xxx.

path+prefix

is a relative or absolute path appended to the file name prefix. The path defines the directory the driver uses to store the newly created database files or locate the existing database files. For example, if Database is set to a value of `C:\data\db\johnqpublic`, the driver either creates or looks for the database `johnqpublic.xxx` in the directory `C:\data\db`. If you do not specify a path, the current working directory is used.

Notes

- The driver parses the User ID value and removes all non-alphanumeric characters. For example, if User ID is specified as `John.Q.Public`, the value used for Database is `JohnQPublic`.
- When SQL Engine Mode is set to Server, the *path+prefix* value overrides the value specified by the Server DB Directory configuration setting (see [Configuring Server Mode](#) on page 992).

Default

The user ID specified for the connection.

GUI Tab

[General tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the `odbc.ini` file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable Bulk Load

Attribute

EnableBulkLoad (EBL)

Purpose

Specifies whether the driver can use the bulk load protocol for insert, update, delete, and batch operations. Bulk load can reduce the number of Web service calls used to execute a statement when compared to statements that are executed individually and may improve performance. Whether the driver actually uses bulk load is determined by the BulkLoadThreshold (BLTH) connection option.

Valid Values

True | False

Behavior

If set to `True`, the driver can use the bulk load protocol for insert, update, delete, and batch operations.

If set to `False`, the driver cannot use the bulk load protocol for insert, update, delete, and batch operations.

Default

False

GUI Tab

[Bulk tab](#)

Fetch Size

Attribute

FetchSize (FS)

Purpose

Specifies the number of rows that the driver processes before returning data to the application. Smaller fetch sizes can improve the initial response time of the query. Larger fetch sizes improve overall fetch times at the cost of additional memory.

FetchSize is related to, but different than, `WSFetchSize`. `WSFetchSize` specifies the number of rows of raw data that the driver fetches from the remote data source, while `FetchSize` specifies how many of these raw data rows the driver processes before returning data to the application. Processing the data includes converting from the remote data source data type to the driver SQL data type used by the application. If `FetchSize` is greater than `WSFetchSize`, the driver makes multiple round trips to the data source to get the requested number of rows before returning control to the application.

Valid Values

0 | x

where:

x

is a positive integer.

Behavior

If set to 0, the driver fetches and processes all of the rows of the result before returning control to the application.

If set to x, the driver fetches and processes the specified number of rows before returning data to the application.

Default

100

GUI Tab

[Advanced tab](#)

Field Delimiter

Attribute

BulkLoadFieldDelimiter (BLFD)

Purpose

Specifies the character that the driver will use to delimit the field entries in a bulk load data file.

Valid Values

x

where:

x

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash(-), the colon(:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Field Delimiter character must be different from the Bulk Load Record Delimiter.

Default

None

GUI Tab

[Bulk tab](#)

Host Name

Attribute

HostName (HOST)

Purpose

The base Salesforce URL to use for logging in. If you are logging into a Salesforce instance other than the default, you must provide the root of the Salesforce URL.

Valid Values

url

where:

url

is the is the root of the Salesforce URL to which you want to connect.

Example

Suppose you have a Salesforce instance that is configured with a production instance and a sandbox instance. You can specify `login.salesforce.com` as the value for the HostName attribute to connect to the production instance or `test.salesforce.com` to connect to the sandbox instance:

Salesforce Instance	URL
Production	login.salesforce.com
Sandbox	test.salesforce.com

Default

login.salesforce.com

GUI Tab

[General tab](#)

Initialization String

Attribute

InitializationString (IS)

Purpose

One or multiple SQL commands to be executed by the driver after it has established the connection to the database and has performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and the driver returns an error indicating which SQL command or commands failed.

Valid Values

string

where:

string

is one or multiple SQL commands.

Multiple commands must be separated by semicolons. In addition, if this option is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified.

Example

Because fetching metadata and generating mapping files can significantly increase the time it takes to connect to Salesforce, the driver caches this information on the client the first time the driver connects on behalf of each user. The cached metadata is used in subsequent connections made by the user instead of re-fetching the metadata from Salesforce. To force the driver to re-fetch the metadata information for a connection, use the InitializationString property to pass the REFRESH SCHEMA SFORCE command in the connection URL. For example:

```
DSN=Salesforce;UID={test@abccorp.com};PWD=secret;InitializationString=(REFRESH SCHEMA SFORCE)
```

Default

None

GUI Tab

[Advanced tab](#)

JVM Arguments

Attribute

JVMArgs (JVMA)

Purpose

A string that contains the arguments that are passed to the JVM that the driver is starting. The location of the JVM must be specified on the driver library path. For information on setting the location of the JVM in your environment, see:

- [Setting the Library Path Environment Variable \(Salesforce Driver on Windows\)](#) on page 34
- [Setting the Library Path Environment Variable \(Salesforce Driver on UNIX/Linux\)](#) on page 42.

When specifying the heap size for the JVM, note that the JVM tries to allocate the heap memory as a single contiguous range of addresses in the application's memory address space. If the application's address space is fragmented so that there is no contiguous range of addresses big enough for the amount of memory specified for the JVM, the driver fails to load, because the JVM cannot allocate its heap. This situation is typically encountered only with 32-bit applications, which have a much smaller application address space. If you encounter problems with loading the driver in an application, try reducing the amount of memory requested for the JVM heap. If possible, switch to a 64-bit version of the application.

Valid Values

string

where:

string

contains arguments that are defined by the JVM. Values that include special characters or spaces must be enclosed in curly braces { } when used in a connection string.

Example

To set the heap size used by the JVM to 256 MB and the http proxy information, specify:

```
{-Xmx256m -Dhttp.proxyHost=johndoe -Dhttp.proxyPort=808}
```

To set the heap size to 256 MB and configure the JVM for remote debugging, specify:

```
{-Xmx256m  
-Xrunjdp:transport=dt_socket, address=9003,server=y,suspend=n -Xdebug}
```

Default

`-Xmx256m`

GUI Tab

[SQL Engine tab](#)

JVM Classpath

Attribute

JVMClasspath (JVMC)

Purpose

Specifies the CLASSPATH for the Java Virtual Machine (JVM) used by the driver. The CLASSPATH is the search string the JVM uses to locate the Java jar files the driver needs.

Valid Values

string

where:

string

specifies the CLASSPATH. Separate multiple jar files by a semi-colon on Windows platforms and by a colon on Linux and UNIX platforms. CLASSPATH values with multiple jar files must be enclosed in curly braces { } when used in a connection string.

Example

On Windows:

```
{.;c:\install_dir\java\lib\  
}
```

On UNIX:

```
{./:/home/user1/install_dir/java/lib/sforce.jar}
```

Default

install_dir\java\lib\sforce.jar

GUI Tab

[SQL Engine tab](#)

LoadBalance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

Specifies the number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Valid Values

0 | *x*

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to *x*, inactive connections are closed after the specified number of seconds passes.

Notes

- The Min Pool Size option may cause some connections to ignore this value.
- This connection option can affect performance.

Default

0

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 974

Log Config File

Attribute

LogConfigFile (LCF)

Purpose

Specifies the filename of the configuration file used to initialize the driver logging mechanism.

If the driver cannot locate the specified file when establishing the connection, the connection fails and the driver returns an error.

Valid Values

string

where:

string

is the relative or fully qualified path of the configuration file used to initialize the driver logging mechanism. If the specified file does not exist, the driver continues searching for an appropriate configuration file.

Default

Empty string

GUI Tab

[Advanced tab](#)

Logon Domain

Attribute

LogonDomain (LD)

Purpose

Specifies the domain part of the Salesforce user id. If Logon Domain is not an empty string, the driver first appends the @ character to the end of the User Name value and then appends the value of Logon Domain.

Valid Values

string

where:

string

is a valid user ID domain.

Default

Empty string

GUI Tab

[Security tab](#)

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the `SQL_ATTR_LOGIN_TIMEOUT` connection attribute using the `SQLSetConnectAttr()` function.

Valid Values

0 | *x*

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, the connection request does not time out, but the driver responds to the `SQL_ATTR_LOGIN_TIMEOUT` attribute.

If set to *x*, the connection request times out after the specified number of seconds unless the application overrides this setting with the `SQL_ATTR_LOGIN_TIMEOUT` attribute.

Default

15

GUI Tab

[Advanced tab](#)

Max Pool Size

Attribute

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

For example, if set to 20, the maximum number of connections allowed in the pool is 20.

Notes

This connection option can affect performance.

Default

100

GUI Tab

[Pooling tab](#)

See also

[Performance Considerations](#) on page 974

Min Pool Size

Attribute

MinPoolSize (MNPS)

Purpose

Specifies the minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Valid Values

0 | x

where:

x

is an integer from 1 to 65535.

For example, if set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

Notes

- This connection option can affect performance.

Default

0

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 974 for details.

Password

Attribute

Password (PWD)

Purpose

Specifies the password to use to connect to your Salesforce instance. A password is required. Contact your system administrator to obtain your password.

Important: Setting the password using a data source is not recommended. The data source persists all options, including the Password option, in clear text.

Valid Values

password | *password+securitytoken*

where:

password

is a valid password. The password is case-sensitive.

password+securitytoken

is a valid password appended by the security token required to connect to the Salesforce instance, for example, `secretXaBARTsLZReM4Px47qPLOS`, where `secret` is the password and the remainder of the value is the security token. Both the password and security token are case-sensitive.

Optionally, you can specify the security token in the Security Token option. Do not specify the security token in both options.

Default

None

GUI Tab

[Pooling tab](#)

See also

[Security Token](#) on page 968

Proxy Host

Attribute

ProxyHost (PXHN)

Purpose

Specifies the Hostname and possibly the Domain of the Proxy Server. The value specified can be a host name, a fully qualified domain name, or an IPv4 or IPv6 address.

Valid Values

server_name | *IP_address*

where:

server_name

is the name of the server or a fully qualified domain name to which you want to connect.

The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details about these formats.

Default

Empty string

GUI Tab

[SQL Engine tab](#)

Proxy Password

Attribute

ProxyPassword (PXPW)

Purpose

Specifies the password needed to connect to the Proxy Server.

Valid Values

String

where:

String

specifies the password to use to connect to the Proxy Server. Contact your system administrator to obtain your password.

Default

Empty string

GUI Tab

[SQL Engine tab](#)

Proxy Port

Attribute

ProxyPort (PXPT)

Purpose

Specifies the port number where the Proxy Server is listening for HTTP and/or HTTPS requests.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your system administrator for the correct number.

Default

0

GUI Tab

[SQL Engine tab](#)

Proxy User

Attribute

ProxyUser

Purpose

Specifies the user name needed to connect to the Proxy Server.

Valid Values

The default user ID that is used to connect to the Proxy Server.

Default

Empty string

GUI Tab

[SQL Engine tab](#)

Read Only

Attribute

ReadOnly (RO)

Purpose

Specifies whether the connection has read-only access to the data source.

Valid Values

0 | 1

Behavior

If set to 1, the connection has read-only access. The following commands are the only commands that you can use when a connection is read-only:

- Call (if the procedure does not update data)
- Explain Plan
- Select (except Select Into)
- Set Schema

The driver returns an error if any other command is executed.

If set to 0, the connection is opened for read/write access, and you can use all commands supported by the product.

Default

0

GUI Tab

[Advanced tab](#)

Record Delimiter

Attribute

BulkLoadRecordDelimiter (BLRD)

Purpose

Specifies the character that the driver will use to delimit the record entries in a bulk load data file.

Valid Values

x

where:

x

is any printable character.

For simplicity, avoid using a value that can be in the data, including all alphanumeric characters, the dash(-), the colon(:), the period (.), the forward slash (/), the space character, the single quote (') and the double quote ("). You can use some of these characters as delimiters if all of the data in the file is contained within double quotes.

Notes

- The Bulk Load Record Delimiter character must be different from the Bulk Load Field Delimiter.

Default

None

GUI Tab

[Bulk tab](#)

Refresh Dirty Cache

Attribute

RefreshDirtyCache (RDC)

Purpose

Specifies whether the driver refreshes a dirty cache on the next fetch operation from the cache. A cache is marked as dirty when a row is inserted into or deleted from a cached table or a row in the cached table is updated.

Valid Values

1 | 0

Behavior

If set to 1 (Enabled), a dirty cache is refreshed when the cache is referenced in a fetch operation. The cache state is set to initialized if the refresh succeeds.

If set to 0 (Disabled), a dirty cache is not refreshed when the cache is referenced in a fetch operation.

Default

1 (Enabled)

See also

[Refreshing Cache Data](#) on page 980

GUI Tab

[Advanced tab](#)

Refresh Schema

Attribute

RefreshSchema (RS)

Purpose

Determines whether the driver automatically refreshes the information in a remote schema (rebuilds the database map for the schema) the first time a user connects to the specified embedded database. The database is opened when the user first makes a connection with the application. When all connections associated with that user are closed, then the driver closes the database. The database must be reopened before it can be used again.

Valid Values

1 | 0

Behavior

If set to 1 (Enabled), the driver automatically refreshes the schema the first time a user connects to the specified database. Any schema objects that have changed since the last time the database map was rebuilt are reflected in the metadata. See Database Configuration File for information about embedded databases and map files.

If set to 0 (Disabled), the driver does not automatically refresh the schema the first time a user connects to the specified database.

Notes

This connection option is functionally equivalent to executing the Refresh Schema statement (see [Refresh Schema \(EXT\)](#)). You can refresh a schema manually at any time by using the Refresh Schema statement.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where `x` is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Advanced tab](#)

Security Token

Attribute

SecurityToken (STK)

Purpose

Specifies the security token required to make a connection to a Salesforce instance that is configured for a security token. If a security token is required and you do not supply one, the driver returns an error indicating that an invalid user or password was supplied. Contact your Salesforce administrator to find out if a security token is required.

Valid Values

string

where:

string

is the value of the security token assigned to the user.

Optionally, you can specify the security token in the Password option by appending the security token to the password, for example, `secretXaBARTsLZReM4Px47qPLOS`, where `secret` is the password and the remainder of the value is the security token. Do not specify the security token in both options.

Notes

- When setting the security token using a data source on Windows, the Security Token option is encrypted.

Default

Empty string

GUI Tab

[Security tab](#)

Server Port Number

Attribute

ServerPortNumber (SPN)

Purpose

Specifies a valid port on which the SQL engine listens for requests from the driver.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your system administrator for the correct number.

Notes

- This option is ignored unless SQL Engine Mode is set to 1 (Server).

Default

19928

GUI Tab

[SQL Engine tab](#)

SQL Engine Mode

Attribute

SQLEngineMode (SEM)

Purpose

Specifies whether the driver's SQL engine runs in the same 32-bit process as the driver (direct mode) or runs in a process that is separate from the driver (server mode). You must be an administrator to modify the server mode configuration values, and to start or stop the SQL engine service.

Valid Values

0 | 1

Behavior

If set to 0 (Direct), the SQL engine runs in direct mode. The driver and its SQL engine run in a single process within the same JVM.

If set to 1 (Server), the SQL engine runs in server mode. The SQL engine operates in a separate process from the driver within its own JVM. You must start the SQL Engine service before using the driver (see [Starting the SQL Engine Server](#) on page 994 for more information). Multiple drivers on different clients can use the same service.

Important: Changes you make to the server mode configuration affect all DSNs sharing the service.

Default

0 - Direct

GUI Tab

[SQL Engine tab](#)

Statement Call Limit

Attribute

StmtCallLimit (SCL)

Purpose

Specifies the maximum number of Web service calls the driver can make when executing any single SQL statement or metadata query.

Valid Values

0 | *x*

where:

x

is a positive integer that defines the maximum number of Web service calls the driver can make when executing any single SQL statement or metadata query.

Behavior

If set to 0, there is no limit.

If set to *x*, the driver uses this value to set the maximum number of Web service calls on a single connection that can be made when executing a SQL statement. This limit can be overridden by changing the STMT_CALL_LIMIT session attribute using the ALTER SESSION statement. For example, the following statement sets the statement call limit to 10 Web service calls:

```
ALTER SESSION SET STMT_CALL_LIMIT=10
```

If the Web service call limit is exceeded, the behavior of the driver depends on the value specified for the Stmt Call Limit Behavior option.

Default

20

GUI Tab

[Web Service tab](#)

Statement Call Limit Behavior

Attribute

StmtCallLimitBehavior (SCLB)

Purpose

Specifies the behavior of the driver when the maximum Web service call limit specified by the Statement Call Limit option is exceeded.

Valid Values

1 | 2

Behavior

If set to 1 (`ErrorAlways`), the driver returns an error if the maximum Web service call limit is exceeded.

If set to 2 (`ReturnResults`), the driver returns any partial results it received prior to the call limit being exceeded. The driver generates a warning that not all of the results were fetched.

Default

1 (`ErrorAlways`)

GUI Tab

[Web Service tab](#)

Transaction Mode

Attribute

TransactionMode (TM)

Purpose

Specifies how the driver handles manual transactions.

Valid Values

0 | 1

Behavior

If set to 1 - `Ignore`, the data source does not support transactions and the driver always operates in auto-commit mode. Calls to set the driver to manual commit mode and to commit transactions are ignored. Calls to rollback a transaction cause the driver to return an error indicating that no transaction is started. Metadata indicates that the driver supports transactions and the `ReadUncommitted` transaction isolation level.

If set to 0 - `No Transactions`, the data source and the driver do not support transactions. Metadata indicates that the driver does not support transactions.

Default

0 (`No Transactions`)

GUI Tab

[Advanced tab](#)

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[Security tab](#)

See also

[Logon Domain](#) on page 959

WSFetchSize

Attribute

WSFetchSize (WSFS)

Purpose

Specifies the number of rows of data the driver attempts to fetch for each ODBC call.

Valid Values

0 | *x*

where:

x

is a positive integer from 1 to 2000 that defines a number of rows.

Behavior

If set to 0, the driver attempts to fetch up to a maximum of 2000 rows. This value typically provides the maximum throughput.

If set to x , the driver attempts to fetch up to a maximum of the specified number of rows. Setting the value lower than 2000 can reduce the response time for returning the initial data. Consider using a smaller WSFetch Size for interactive applications only.

Default

0 (up to a maximum of 2000 rows)

GUI Tab

[Web Service tab](#)

See also

[Fetch Size](#) on page 953

[WSTimeout](#) on page 974

WSRetry Count

Attribute

WSRetryCount (WSRC)

Purpose

The number of times the driver retries a timed-out Select request. Insert, Update, and Delete requests are never retried. The timeout period is specified by the WSTimeout (WST) connection option.

Valid Values

0 | x

where:

x

is a positive integer.

Behavior

If set to 0, the driver does not retry timed-out requests after the initial unsuccessful attempt.

If set to x , the driver retries the timed-out request the specified number of times.

Default

3

GUI Tab

[Web Service tab](#)

See also

[WSTimeout](#) on page 974

WSTimeout

Attribute

WSTimeout (WST)

Purpose

Specifies the time, in seconds, that the driver waits for a response to a Web service request.

Valid Values

0 | x

where:

x

is a positive integer that defines the number of seconds the driver waits for a response to a Web service request.

Behavior

If set to 0, the driver waits indefinitely for a response; there is no timeout.

If set to x , the driver uses the value as the default timeout for any statement created by the connection.

If a Select request times out and `WSRetryCount` (WSRC) is set to retry timed-out requests, the driver retries the request the specified number of times.

Default

120 (seconds)

GUI Tab

[Web Service tab](#)

See also

[WSRetry Count](#) on page 973

Performance Considerations

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the `ApplicationUsingThreads` attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Connection Pooling (Pooling): If you enable the driver to use connection pooling, you can set additional options that affect performance:

- **Load Balance Timeout (LoadBalanceTimeout):** You can define how long to keep connections in the pool. The time that a connection was last used is compared to the current time and, if the timespan exceeds the value of the Load Balance Timeout option, the connection is destroyed. The Min Pool Size option can cause some connections to ignore this value.
- **Connection Reset (ConnectionReset):** Resetting a re-used connection to the initial configuration settings impacts performance negatively because the connection must issue additional commands to the server.
- **Max Pool Size (MaxPoolSize):** Setting the maximum number of connections that the pool can contain too low might cause delays while waiting for a connection to become available. Setting the number too high wastes resources.
- **Min Pool Size (MinPoolSize):** A connection pool is created when the first connection with a unique connection string connects to the database. The pool is populated with connections up to the minimum pool size, if one has been specified. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Enable Bulk Load (EnableBulkLoad): For batch inserts and individual inserts, updates, and deletes, the driver can use the Salesforce Bulk API instead of the Web service API. Using the Bulk API significantly reduces the number of Web service calls the driver uses to transfer data to Salesforce and may improve performance.

Fetch Size/WS Fetch Size (FetchSize/WSFetchSize): The connection options Fetch Size and WSFetch Size can be used to adjust the trade-off between throughput and response time. In general, setting larger values for WSFetch Size and Fetch Size will improve throughput, but can reduce response time.

For example, if an application attempts to fetch 100,000 rows from the remote data source and WSFetch Size is set to 500, the driver must make 200 Web service calls to get the 100,000 rows. If, however, WSFetch Size is set to 2000 (the maximum), the driver only needs to make 50 Web service calls to retrieve 100,000 rows. Web service calls are expensive, so generally, minimizing Web service calls increases throughput. In addition, many Cloud data sources impose limits on the number of Web service calls that can be made in a given period of time. Minimizing the number of Web service calls used to fetch data also can help prevent exceeding the data source call limits.

For many applications, throughput is the primary performance measure, but for interactive applications, such as Web applications, response time (how fast the first set of data is returned) is more important than throughput. For example, suppose that you have a Web application that displays data 50 rows to a page and that, on average, you view three or four pages. Response time can be improved by setting Fetch Size to 50 (the number of rows displayed on a page) and WSFetch Size to 200. With these settings, the driver fetches all of the rows from the remote data source that you would typically view in a single Web service call and only processes the rows needed to display the first page.

Data Types

The following table lists the data types supported by the Salesforce driver for local tables, how the Salesforce data types exposed by the Salesforce Web Service API map to them, and how the Salesforce Web Service API data types map to the ODBC data types.

Table 74: Salesforce Data Types for Local Tables

Salesforce Data Type	Web Service API Data Type	ODBC Data Type
ANYTYPE	anytype	SQL_WVARCHAR
AUTONUMBER	string	SQL_WVARCHAR

Salesforce Data Type	Web Service API Data Type	ODBC Data Type
BINARY	binary	SQL_LONGVARBINARY
CHECKBOX	boolean	SQL_BIT
COMBOBOX	combobox	SQL_WVARCHAR
CURRENCY	currency	SQL_DOUBLE
DATE	date	SQL_TYPE_DATE
DATETIME	datetime	SQL_TYPE_TIMESTAMP
EMAIL	email	SQL_WVARCHAR
ENCRYPTEDTEXT	encryptedtext	SQL_WVARCHAR
HTML	html	SQL_WLONGVARCHAR
ID	id	SQL_WVARCHAR
INT	double	SQL_INTEGER
LONGTEXTAREA	longtextarea	SQL_WLONGVARCHAR
MULTISELECTPICKLIST	multipicklist	SQL_WVARCHAR
NUMBER	double	SQL_DOUBLE if scale does not = 0 or precision > 9 or the NumberFieldMapping key of the ConfigOptions connection option is set to 2. SQL_INTEGER if scale = 0 and precision <= 9 and the NumberFieldMapping key of the ConfigOptions connection option is set to 1.
PERCENT	percent	SQL_DOUBLE
PHONE	phone	SQL_WVARCHAR
PICKLIST	picklist	SQL_WVARCHAR
REFERENCE	reference	SQL_WVARCHAR
TEXT	string	SQL_WVARCHAR
TEXTAREA	textarea	SQL_WVARCHAR
TIME	time	SQL_TYPE_TIME
URL	url	SQL_WVARCHAR

The following table lists the data types supported by the Salesforce driver for remote tables, how the Salesforce data types exposed by the Salesforce Web Service API map to them, and how the Salesforce Web Service API data types map to the ODBC data types.

Table 75: Salesforce Data Types for Remote Tables

Salesforce Data Type	Web Service API Data Type	ODBC Data Type
ANYTYPE	anytype	SQL_WVARCHAR
AUTONUMBER	string	SQL_WVARCHAR
BINARY	binary	SQL_LONGVARBINARY
CHECKBOX	boolean	SQL_BIT
COMBOBOX	combobox	SQL_WVARCHAR
CURRENCY	currency	SQL_DOUBLE
DATACATEGORYGROUPREFERENCE	DataCategoryGroupReference	SQL_WVARCHAR
DATE	date	SQL_TYPE_DATE
DATETIME	datetime	SQL_TYPE_TIMESTAMP
EMAIL	email	SQL_WVARCHAR
HTML	html	SQL_WLONGVARCHAR
ID	id	SQL_WVARCHAR
INT	double	SQL_INTEGER
LONGTEXTAREA	longtextarea	SQL_WLONGVARCHAR
MULTISELECTPICKLIST	multipicklist	SQL_WVARCHAR
NUMBER	double	SQL_DOUBLE if scale does not = 0 or precision > 9 or the NumberFieldMapping key of the ConfigOptions connection option is set to 2. SQL_INTEGER if scale = 0 and precision <= 9 and the NumberFieldMapping key of the ConfigOptions connection option is set to 1.
PERCENT	percent	SQL_DOUBLE
PHONE	phone	SQL_WVARCHAR

Salesforce Data Type	Web Service API Data Type	ODBC Data Type
PICKLIST	picklist	SQL_WVARCHAR
REFERENCE	reference	SQL_WVARCHAR
TEXT	string	SQL_WVARCHAR
TEXTAREA	textarea	SQL_WVARCHAR
TIME	time	SQL_TYPE_TIME
URL	url	SQL_WVARCHAR

Mapping Objects to Tables

The driver automatically maps Salesforce objects and fields to tables and columns the first time it connects to a Salesforce instance. The driver maps both standard and custom objects and includes any relationships defined between objects.

The driver uses a local embedded database to instantiate the mapping of the remote data source objects to tables and the metadata associated with those tables. The driver creates a database per user. The embedded database is created in the directory from which the application is run and uses the user ID specified for the connection as the name of the database. If the user ID contains punctuation or other non-alphanumeric characters, the driver strips those characters from the user ID to form the name of the database. The driver provides connection options that you can use to override the default setting for the name and location of the database (Database).

By default, the Salesforce driver does not include audit columns in table definitions when mapping Salesforce objects to tables. The Config Options connection option can be used to include audit columns. The following columns can be included or excluded:

- CreatedById
- CreatedByDate
- LastModifiedId
- LastModifiedDate
- SystemModestamp
- MasterRecordId

When mapping custom objects and fields, the Salesforce driver strips the standard "__c" suffix from the names of the custom objects and fields by default. You can set the CustomSuffix key of the [Config Options](#) on page 945 connection option to prevent the driver from stripping the "__c" suffix. When mapping Salesforce system fields to columns in a table, the driver changes the column name to make it evident that the column is a system column. If you do not want the driver to change the names of system columns, set the MapSystemColumnNames key of the Config Options connection option to 0.

The [Create Database](#) on page 950 (CDB) connection option allows you to update or re-create the embedded database that defines and handles the object-to-table mapping.

Client-Side Caches

The Salesforce driver can implement a client-side data cache for improved performance. Data is cached from the remote data source to the local machine on which the driver is located.

The driver caches data on a per-table basis, as opposed to caching the result of a particular query. Caching data on a table level allows the caches to be queried, filtered, and sorted in other queries. Once a cache is created, its use is transparent to the application. For example, if a cache is created on the Account table, then all subsequent queries that reference Account access the Account cache. Disabling or dropping the cache allows references to the Account table to access the remote data again. Because the use of the cache is transparent, no changes to the application are required to take advantage of the cache.

You must specifically create a cache before it can be populated; caches are not created automatically. After you have created a cache on a table, the cache will be populated as a result of the next operation on the table. For example, after creating a cache on Account, data is returned from the Salesforce data source and stored locally in the cache when you first execute the following statement:

```
SELECT ROWID, SYS_NAME FROM Account
```

Any subsequent queries against the Account table return data from the cache, which reduces response time. SQL queries can access both cached data and remote data (data stored in Salesforce that has not been assigned to a cache) in the same statement.

The caches maintained by the Salesforce driver are write-through caches. This means that, for any operation that modifies data in a table that is cached, the driver performs the operation on the remote data first and then updates the cache as much as possible.

To create, modify, refresh, or delete client-side data caches, use the following SQL statement extensions:

- Create Cache
- Alter Cache
- Refresh Cache
- Drop Cache

See the following sections for overviews of each extension. See [SQL Statements and Extensions for the Salesforce Driver](#) for descriptions of the syntax of these extensions.

Creating a Cache

You create a cache using the Create Cache statement (see [Create Cache \(EXT\)](#)). A cache can be created on a single table or on a set of related tables. When creating a cache on a single table, you specify the name of the table to cache and can optionally specify a filter for the table. The filter determines whether the cache holds all of the data in the remote table or a subset of the data that matches the filter. You can also specify attributes for the Create Cache statement that determine:

- Whether the cache data is held on disk or in memory
- How often the cache data is refreshed
- Whether the cache is initially enabled
- Whether the driver checks to see if a refresh is needed at connect time

Creating a cache for a set of related tables is similar to creating a cache on a single table except that a primary table and one or more referencing tables are specified. This is useful if you want to cache a subset of data for a table and also cache data related to that subset of data. For example, you might have three tables, Account, Contact, and Opportunity, where both a contact and an opportunity belong to a particular account. Using a relational cache, you could specify that accounts that have had activity in the past year be cached, as well as caching the opportunities and contacts for only those cached accounts.

Modifying a Cache Definition

Once a cache has been created, you can modify the definition of the cache or set of related caches with the Alter Cache statement (see [Alter Cache \(EXT\)](#)). Only the attributes of the cache can be modified through the Alter Cache statement; the table or related set of tables cannot be changed and a single table cache cannot be changed to a relational cache.

Warning: Changing the attributes of a cache may cause the current data in the cache to be discarded and refetched from the remote data source.

Disabling and Enabling a Cache

When a cache is defined on a table, all fetch operations performed on that table access the cache, essentially hiding the remote table from the application. At times, you may want an application to query the remote data instead of the cached data. For example, assume that a cache was created on Account with a filter set to cache accounts that have had activity in the past year. You may want to run a query to get information about an account that has not been active for two years. One alternative would be to drop the Account cache, run the query, and then recreate the cache on Account, but this can be problematic. First, you must recreate the cache and make sure it had the same attributes as before. Second, the data in the cache is discarded and needs to be refetched when the cache is recreated. Depending on the amount of cached data, this could take a significant amount of time. To address this type of issue, the Salesforce driver can temporarily disable a cache. When a cache is disabled, its definition and data are maintained. Any queries that reference a table with a disabled cache access the remote table. When you want to access cached data again, the cache can be enabled.

Refreshing Cache Data

To prevent the data in a cache from becoming out of date, the driver must periodically refresh the cache data with data from the remote data source. To minimize the amount of data that needs to be moved when a cache is refreshed, and therefore the time required to refresh it, the driver checks to see which records in the remote table have been added, modified, or deleted since the last time the cache was refreshed. The driver retrieves only data for added or modified records and removes only deleted records from the cache. You or the application can refresh the cache manually or the driver can refresh the cache automatically.

You can refresh a cache manually at any time by using the Refresh Cache statement (see [Refresh Schema \(EXT\)](#)). The Refresh Cache statement can also be used to perform a Clean (complete) refresh in addition to the standard optimized refresh. A Clean refresh discards all of the data from the cache and repopulates it with data from the remote data source.

The driver can refresh a cache automatically in one of two ways. When you create a cache, one of the attributes that you set is the refresh interval for the cache. During each cache query, the driver checks to see whether the time elapsed since the last refresh exceeds the refresh interval for the cache. If it has, the driver refreshes the cache before satisfying the query.

Update operations to a table that is cached can trigger the driver to refresh the cache automatically. The caches maintained by the Salesforce driver are write-through caches. For any operation that modifies data in a table that is cached, the driver performs the operation on the remote data first and then updates the cache as much as possible. The driver may not be able to update the cache with all of the modifications because some of the modified data may have been generated by the remote data source. For example, if a row is inserted but a value for all columns in the row is not required, any default values generated by the remote data source for columns not specified in the Insert statement would not be set in the cache. Because the driver cannot reflect all of the changes made when a cached table is modified, it sets the cache state to dirty. When a cache state is dirty, the next query that attempts to fetch data from that cache causes the driver to refresh the cache before the fetch operation is performed. This allows the fetch to see the values populated by the remote data source.

Automatically refreshing a dirty cache is not always desirable. For example, if an application alternates fetches and inserts on a table, and the insert does not depend on any remote data source generated values, then the refresh between fetches is unnecessary. The RefreshDirtyCache (RDC) connection option (see [Connection Option Descriptions](#) on page 939) controls whether the driver automatically refreshes a cache with a dirty state. The state of a cache can be viewed by selecting the STATUS column of the SYSTEM_CACHES catalog table. See [SYSTEM_CACHES Catalog Table](#) on page 982 for more information.

Dropping a Cache

You can drop an existing cache using the Drop Cache statement (see [Drop Cache \(EXT\)](#)). If a cache is a relational cache, the Drop Cache statement drops the cache for the primary table as well as the caches for the related tables.

Note: When a cache is dropped, all of the data in that cache is discarded.

Cache MetaData

The Salesforce driver maintains information about the caches that have been created. The driver provides two system tables to expose the cache information, the SYSTEM_CACHES table and the SYSTEM_CACHE_REFERENCES table.

The SYSTEM_CACHES and SYSTEM_CACHE_REFERENCES system tables exist in the INFORMATION_SCHEMA schema. See [Catalog Tables](#) on page 981 for a complete description of the contents of these system tables.

Catalog Tables

The Salesforce driver provides a standard set of catalog tables that maintain the information returned by various ODBC catalog functions such as SQLTables, SQLColumns, SQLDescribeParam and SQLDescribeCol. If possible use the ODBC catalog functions to obtain this information instead of querying the catalog tables directly.

The driver also provides additional catalog tables that maintain metadata specific to the Salesforce driver. This section defines the catalog tables that provide Salesforce driver-specific information. The catalog tables are defined in the INFORMATION_SCHEMA schema.

SYSTEM_CACHES Catalog Table

The SYSTEM_CACHES catalog table stores the definitions of the caches created on remote tables. The data in the SYSTEM_CACHES table provides the name, type (single table or relational), status, and other information for each defined cache. The table name returned for a remote relational cache is the name of the primary table of the relational cache; however, its type is REMOTE RELATIONAL. You can query SYSTEM_CACHES to determine the caches currently defined by the driver. The values in the SYSTEM_CACHES table are read-only. The referenced tables of a relational cache can be determined by querying the SYSTEM_CACHE_REFERENCES catalog table (see [SYSTEM_CACHE_REFERENCES Catalog Table](#) on page 983).

The following table describes the columns of the SYSTEM_CACHES table, which is sorted on the following columns: CACHE_TYPE, TABLE_SCHEMA, and TABLE_NAME.

Table 76: SYSTEM_CACHES Catalog Table

Column Name	Data Type	Description
TABLE_CAT	VARCHAR(128),NULLABLE	The catalog that contains the remote table on which the cache is defined. It is NULL for the Salesforce driver.
TABLE_SCHEM	VARCHAR(128),NULLABLE	The schema that contains the remote table on which the cache is defined.
TABLE_NAME	VARCHAR(128),NOT NULL	The name of the remote table on which the cache is defined.
CACHE_TYPE	VARCHAR(20),NOT NULL	The type cache, which can be either REMOTE TABLE or REMOTE RELATIONAL.
REFRESH_INTERVAL	INTEGER,NOT NULL	The refresh interval (in minutes).
INITIAL_CHECK	VARCHAR(20),NOT NULL	The value that defines when the initial refresh check is performed: ONFIRSTCONNECT or FIRSTUSE.
PERSIST	VARCHAR(20),NOT NULL	The value that defines whether the data in the cache is persisted past the lifetime of the connection: TEMPORARY, MEMORY, or DISK.
ENABLED	BOOLEAN,NOT NULL	The value that defines whether the cache is enabled for use with SQL statements: TRUE or FALSE.
CALL_LIMIT	INTEGER,NOT NULL	The maximum number of Web service calls that can be made when refreshing the cache. The value 0 indicates no call limit.
REFRESH_MODE	INTEGER,NOT NULL	For internal use only.
FILTER	VARCHAR(128),NULLABLE	The Where clause used to filter the rows that are cached.

Column Name	Data Type	Description
LAST_REFRESH	DATETIME, NULLABLE	The time, in Coordinated Universal Time (UTC), the cache was last refreshed.
STATUS	VARCHAR(30)	The Cache status. Valid values are: New: The cache has been created, but the data has not been populated. Initialized: The cache has been created and the data has been populated. Load aborted: The cache has been created, but the last attempt to populate the data failed. The cache is still valid. The next access attempts to populate the data again. Invalid: The cache is invalid. The second attempt to populate the data failed. Dirty: An insert or update operation has been performed on the cache and the cache has not been refreshed.

SYSTEM_CACHE_REFERENCES Catalog Table

The referenced tables in a relational cache can be determined by querying the SYSTEM_CACHE_REFERENCES system table. This table contains the names of the referenced tables as well as the name of the primary table with which they are associated.

The following table defines the columns of the SYSTEM_CACHES table, which is sorted on the following columns: TABLE_SCHEMA, TABLE_NAME, and REF_TABLE_NAME.

Table 77: SYSTEM_CACHE_REFERENCES

Column	Data Type	Description
PRIMARY_TABLE_CAT	VARCHAR (128), NULLABLE	The catalog that contains the primary table of the relational cache. It is NULL for the Salesforce driver.
PRIMARY_TABLE_SCHEM	VARCHAR (128), NULLABLE	The schema that contains the primary table of the relational cache.
PRIMARY_TABLE_NAME	VARCHAR (128), NOT NULL	The primary table of the relational cache.
REF_TABLE_NAME	VARCHAR (128), NOT NULL	The name of the referenced table.
RELATIONSHIP_NAME	VARCHAR(128), NOT NULL	The name of the foreign key relationship used to relate this table to the primary table or one of the other tables in the relational cache.

SYSTEM_REMOTE_SESSIONS Catalog Table

The system table named SYSTEM_REMOTE_SESSIONS stores information about the each of the remote sessions that are active for a given database. The values in the SYSTEM_REMOTE_SESSION table are read-only.

The following table defines the columns of the SYSTEM_REMOTE_SESSIONS table, which is sorted on the following columns: SESSION_ID and SCHEMA.

Table 78: SYSTEM_REMOTE_SESSIONS Catalog Table

Column Name	Data Type	Description
SESSION_ID	INTEGER, NOT NULL	The connection (session) id with which the remote session is associated.
SCHEMA	VARCHAR(128), NOT NULL	The schema name that is mapped to the remote session.
TYPE	VARCHAR(30), NOT NULL	The remote session type. The current valid type is Salesforce.
INSTANCE	VARCHAR(128)	The remote session instance name or null if the remote data source does not have multiple instances. The Salesforce value for INSTANCE has the following form: <i>Organization_Name</i> [Sandbox] where <i>Organization_Name</i> is the organization name of the Salesforce instance to which the connection is established. If the connection is established to a sandbox of the organization, then the word Sandbox is added to the end of the name.
VERSION	VARCHAR(30), NOT NULL	The version of the remote data source to which the session is connected. For Salesforce, this is the version of the Web Service API the driver is using to connect to Salesforce.
CONFIG_OPTIONS	LONGVARCHAR, NOT NULL	The configuration options used to define the remote data model to relational data model mapping.
SESSION_OPTIONS	LONGVARCHAR, NOT NULL	The options used to establish the remote connection. This typically is information needed to log into the remote data source. The password value is not displayed.

Column Name	Data Type	Description
WS_CALL_COUNT	INTEGER, NOT NULL	The number of Web service calls made through this remote session. The value of the WS_CALL_COUNT column can be reset using the ALTER SESSION statement.
WS_AGGREGATE_CALL_COUNT	INTEGER, NOT NULL	The total of all of the Web service calls made to the same remote data source by all active connections using the same server name and user ID.
REST_AGGREGATE_CALL_COUNT	INTEGER, NOT NULL	The number of REST calls made by this connection. REST calls are used for bulk operations, invoking reports, and describing report parameters.

SYSTEM_SESSIONINFO Catalog Table

The system table named SYSTEM_SESSIONINFO describes details about your connection to Salesforce.

The following table defines the keys for the SYSTEM_SESSIONINFO table. The values change based on your data source settings.

Table 79: SYSTEM_SESSIONINFO Catalog Table

Key	Description
AUTOCOMMIT	Autocommit is always enabled.
DATABASE	The location and the filename prefix for the data mapping and configuration files.
DATABASE_READONLY	Indicates whether the database the session is connected to is read only
DB_FILE_LOCATION	The fully qualified path to the directory or folder that contains the database and mapping files.
DB_FILE_PREFIX	The filename prefix of the database and mapping files the driver is using.
IDENTITY	Currently always zero
MAXROWS	Currently always zero
LOG_CONFIG_FILE	The fully qualified path to the directory or folder that contains the logging configuration file.
SCHEMA	The name of the remote Salesforce schema.
SESSION_ID	The ID for this session

SESSION_READONLY	Indicates whether the session is read only
USER	The user that is associated with this session.

SYSTEM_SESSIONS Catalog Table

The system table named SYSTEM_SESSIONS stores information about current system sessions. The values in the SYSTEM_SESSIONS table are read-only.

The following table defines the columns of the SYSTEM_SESSIONS table.

Table 80: SYSTEM_SESSIONS

Column	Data Type	Description
SESSION_ID	INTEGER, NOT NULL	A unique ID that identifies this session. The system function CURSESSIONID() returns the session ID associated with the connection. See SQL Statements and extensions for the Salesforce Driver for details on CURSESSIONID().
CONNECTED	DATETIME, NOT NULL	The date and time the session was established.
USER_NAME	VARCHAR (128), NOT NULL	The name of the embedded database that the session is using.
IS_ADMIN	BOOLEAN	For internal use only.
AUTOCOMMIT	BOOLEAN, NOT NULL	For future use.
READONLY	BOOLEAN, NOT NULL	True if the connection is in read-only mode. The READONLY status is based on whether the connection has been explicitly set to read-only mode by the Read Only connection option.
MAXROWS	INTEGER, NOT NULL	For future use.
LAST_IDENTITY	BIGINT, NULLABLE	For future use.
TRANSACTION_SIZE	INTEGER, NOT NULL	For future use.
CURRENT_SCHEMA	VARCHAR (128), NOT NULL	The current schema for the session. The current schema may be changed using the ALTER SESSION SET CURRENT_SCHEMA statement.
STMT_CALL_LIMIT	INTEGER, NOT NULL	The maximum number of Web service calls that the driver uses in attempting to execute a query to a remote data source. The statement call limit for the session may be changed via the ALTER SESSION SET STMT_CALL_LIMIT statement.

Timeouts

The following types of timeout situations can occur when connecting to Salesforce:

- **Session timeouts.** Most remote data sources impose a limit on the duration of active sessions, meaning a session can fail with a session timeout error if the session extends past the limit. This is particularly true when connection pooling is used. The driver automatically attempts to re-establish a new session if the driver receives a session timeout error from a data source. The driver uses the initial servername, port (if appropriate), remote user ID, and remote password (encrypted) to re-establish the session. If the attempt fails, the driver returns an error indicating that the session timed out and the attempt to re-establish the session failed.
- **Web service request timeouts.** You can configure the driver to never time out while waiting for a response to a Web service request or to wait for a specified interval before timing out by setting the connection option [WSTimeout](#) on page 974. For fetch requests only, if the request times out, you can configure driver to retry the request a specified number of times by setting the [WSRetry Count](#) on page 973. connection option. If all subsequent attempts to retry a request fails, the driver returns an error indicating that the service request timed out and the subsequent requests failed. See [Connection Option Descriptions](#) on page 939 for details on the WS Timeout and WS Retry Count connection options.

Views and Remote/Local Tables

You can create views with the Create View statement. A view is like a named query. The view can refer to any combination of remote and local tables as well as other views.

You can create a remote or local table using the Create Table statement. A remote table is a Salesforce object and is exposed in the SFORCE schema. A local table is maintained by the driver and is local to the machine on which the driver is running. A local table is exposed in the PUBLIC schema.

See [SQL Statements and Extensions for the Salesforce Driver](#) for details on the Create View and Create Table statements and other SQL statements supported by the driver.

Using Identifiers

Identifiers are used to refer to objects exposed by the driver, such as tables, columns, or caches. The driver supports both unquoted and quoted identifiers for naming objects. An unquoted identifier must start with an ASCII alpha character and can be followed by zero or more ASCII alpha or numeric characters. Unquoted identifiers are converted to uppercase before being used.

Quoted identifiers must be enclosed in double quotation marks ("""). A quoted identifier can contain any Unicode character, including the space character, and is case-sensitive. The Salesforce driver recognizes the Unicode escape sequence \uxxxx as a Unicode character. You can specify a double quotation mark in a quoted identifier by escaping it with a double quotation mark.

The maximum length of both quoted and unquoted identifiers is 128 characters.

Note: When object names are passed as arguments to catalog functions, the case of the value must match the case of the name in the database. If an unquoted identifier name was used when the object was created, the value passed to the catalog function must be uppercase because unquoted identifiers are converted to uppercase before being used. If a quoted identifier name was used when the object was created, the value passed to the catalog function must match the case of the name as it was defined. Object names in results returned from catalog functions are returned in the case that they are stored in the database.

Database Configuration File

You can configure an embedded database and data mapping using a database configuration file in XML format. Some of these values you can set in the file are the same as those you can set using the [Config Options](#) on page 945 connection option (see [Connection Option Descriptions](#) on page 939). Some database configuration values can be set only using a configuration file.

The name of the database configuration file has the format:

```
dbname.config
```

where:

```
dbname
```

is the name of the database to be configured. For example, if your environment has a database named mydb or a database configuration file named mydb.config, when the driver establishes a connection, it performs the following tasks:

- Checks to see if an embedded database named mydb exists (or a database using the default *dbname* if one is not specified). If mydb exists, the driver connects to the remote data source using the mydb database.
- If mydb does not exist and the driver is configured to create a database, the driver looks for a database configuration file named mydb.config. If the database configuration file exists, the driver creates the database and mapping using the properties specified in the database configuration file.
- If mydb.config does not exist, the driver generates a database configuration file with default settings and uses those settings to create the database and its mapping.

Example Database Configuration File

The following is an example database configuration file:

```
<?xml version='1.0' encoding='UTF-8'?>
<Database xmlns="http://datadirect.com/cloud/config/1.0">
  <User name="CONNECT2" defaultSchema="SFORCE">
    <UseSchema name="SFORCE"/>
    <UseSchema name="PUBLIC"/>
  </User>
  <Schema name="SFORCE" type="Salesforce">
    <ConfigOptions>uppercaseidentifiers=true;localtables=0;auditcolumns=all;
    customsuffix=strip;KeywordConflictSuffix=;</ConfigOptions>
    <SessionOptions>loginhost=test.salesforce.com;userid=
    connect2@progress.com</SessionOptions>
  </Schema>
  <Schema name="PUBLIC" type="local">
  </Schema>
</Database>
```

The following are descriptions of the elements of the database configuration file.

Database

Child Elements

User, Schema

Purpose

The Database element is the root element of the database configuration file. It does not define any configuration; it contains all of the elements that do define the database configuration. One and only one Database element must exist.

User

Parent Element

Database

Child Element

UseSchema

Purpose

Specifies the User ID used by the driver. At least one User element must exist.

Attributes

name [required]: The user name is a string with a maximum length of 128 characters. The default is `name=userid`, where `userid` is the User ID used by the driver.

defaultSchema: The name of the schema to use for unqualified table and column identifiers. If `defaultSchema` is not specified, the schema specified in the first `useSchema` child element is used as the default schema. The default is `defaultSchema=SFORCE`.

UseSchema

Parent Element

User

Child Element

None

Purpose

The UseSchema element specifies a schema that is visible to the user of this element. A schema contains the mapping between the remote data model and the relational tables the driver exposes. Multiple schemas can be associated with a user. At least one UseSchema element must exist.

A basic User definition typically has two UseSchema elements: one that specifies the mapping to the remote data source and one for the local schema.

Attributes

name [required]: The name of the schema to associate with the user. The schema name is a string with a maximum length of 128 characters. The defaults are `name=SFORCE` [remote] and `name=PUBLIC` [local].

Schema

Parent Element

Database

Child Elements

ConfigOptions, SessionOptions

Purpose

The Schema element defines the schema that contains the mapping for a remote data source. The database configuration file must contain at least one schema definition and may contain multiple schema definitions. Each schema definition defines the type of the data source to which the schema maps, the information to connect to the remote database (except password), and the information needed to configure the remote data model to relational table mapping. At least one Schema element must exist.

Attributes

name [required]: The name of the schema that defines the data model to relational mapping. This attribute can be any valid identifier name. The defaults are name=SFORCE for the remote data source and name=PUBLIC for the local database.

type [required]: The type of remote data source for which the schema defines mapping. This attribute must be type=Salesforce for the remote data source and type=local for the local database.

ConfigOptions

Parent Element

Schema

Child Element

None

Purpose

The ConfigOptions element is a string that specifies the configuration options used to define how the remote data source data model is mapped to relational tables. The ConfigOption string has the same keys, values, and syntax as the Config Options connection option (see [Connection Option Descriptions](#) on page 939) except that the enclosing curly brackets are not required (see [Database Configuration File](#) on page 988). The default is an empty string.

Attributes

None

SessionOptions

Parent Element

Schema

Child Element

None

Purpose

The SessionOptions element is a string of key value pairs that specifies the information needed to connect to the remote data source. SessionOptions includes the server name and remote user id, for example:

```
loginhost=login.salesforce.com;userid=john.public@abccorp.com
```

Attributes

None

Reports

The Salesforce driver exposes reports defined on a Salesforce instance as stored procedures. An application can obtain a list of the reports defined on a Salesforce instance by calling the SQLProcedures catalog function. The names of the reports that can be invoked through the driver are listed in the PROCEDURE_NAME name column of the SQLProcedures results.

Salesforce organizes reports into folders. The Salesforce driver incorporates the folder name and report name into the procedure name reported by SQLProcedures. The driver creates the reported procedure name by prepending the folder name to the report name using an underscore to join them. Additionally, any spaces in the report or folder names are replaced with an underscore character. Like all identifier name metadata returned by the driver, the procedure name is uppercase. For example, if a report named Opportunity Pipeline is in the folder Opportunity Reports, it would be rendered as:

```
OPPORTUNITY_REPORTS_OPPORTUNITY_PIPELINE
```

An application invokes a report using the standard Call escape syntax, `{call report name}`, and ODBC mechanisms for calling a stored procedure that returns a resultset. The following example shows one way to invoke the Opportunity Pipeline report:

```
SQLRETURN      retVal;
HSTMT          hStmt = NULL;
SQLWCHAR*      sql;
sql = L"{call OPPORTUNITY_REPORTS_OPPORTUNITY_PIPELINE}";
retVal = SQLExecDirect(hStmt, sql, SQL_NTS);
if (SQL_SUCCESS == retVal) {
    //      process results
}
```

Note: The API used by the driver to obtain the list of reports and execute the reports is not an API that is documented by Salesforce. This API may change or may not be supported in the future.

Note: When passing parameters to stored procedures, reports are not supported.

Connecting Through a Proxy Server

In some environments, your application may need to connect through a proxy server, for example, if your application accesses an external resource such as a Web service. At a minimum, your application needs to provide the following connection information when you invoke the JVM if the application connects through a proxy server:

- Server name or IP address of the proxy server
- Port number on which the proxy server is listening for HTTP/HTTPS requests

In addition, if authentication is required, your application may need to provide a valid user ID and password for the proxy server. Consult with your system administrator for the required information.

For example, the following command invokes the JVM while specifying a proxy server named `pserver`, a port of 808, and provides a user ID and password for authentication:

```
java -Dhttp.proxyHost=pserver -Dhttp.proxyPort=808 -Dhttp.proxyUser=smith  
-Dhttp.proxyPassword=secret -cp sforce.jar com.acme.myapp.Main
```

Alternatively, you can use the Proxy Host, Proxy Port, Proxy User, and Proxy Password connection attributes, but these options are applied only for the first connection. See [Connection Option Descriptions](#) on page 939 for details about these attributes.

Configuring the SQL Engine Server

Some applications may experience problems loading the JVM required for the SQL engine because the process exceeds the available heap space. If your application experiences problems loading the JVM, you can configure the Salesforce driver to operate in server mode.

By default, the Salesforce driver operates in direct mode, with the SQL engine and JVM running in a single 32-bit process within the same JVM. In server mode, the driver's SQL engine runs in a separate 32-bit process with its own JVM instead of trying to load the SQL engine and JVM in the same process used by the driver.

Note: You must be an administrator to start or stop the service, or to configure any settings for the service.

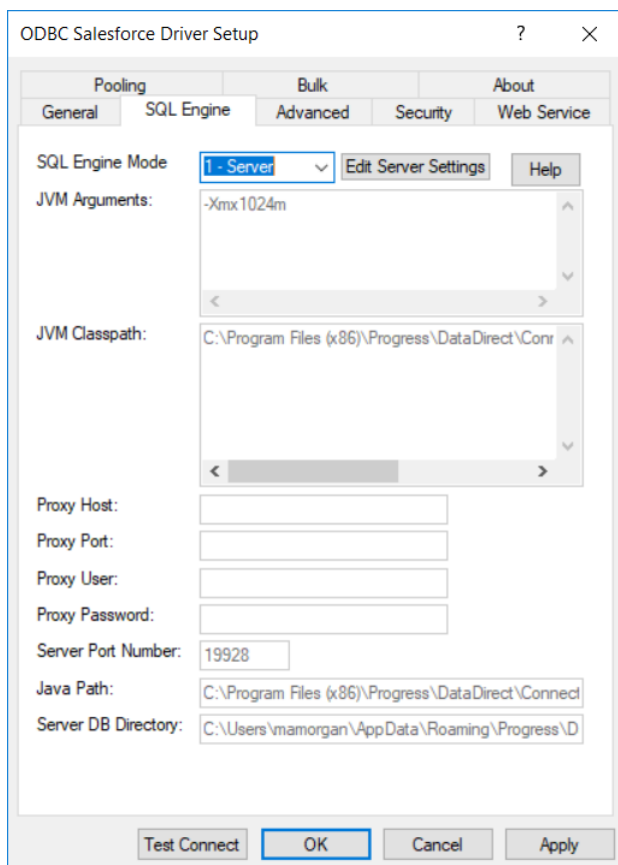
Configuring the SQL Engine Server on Windows

The following sections describe how to configure, start, and stop the SQL Engine Server on Windows platforms. By default, the driver is configured to run in Auto direct mode on Windows platforms.

Configuring Server Mode

1. Set the SQL Engine Mode connection property to a value of `1 - Server`. All fields on the SQL Engine tab become readonly, and the **Edit Server Settings** button appears.
2. Click **Edit Server Setting** to display the ODBC Salesforce SQL Engine Service Setup dialog box. Use this dialog box to define settings for Server Mode and to start and stop the Progress DataDirect Salesforce SQL Engine service.

The SQL Engine Service Setup dialog box appears.



JVM Arguments: A string that contains the arguments that are passed to the JVM that the driver is starting. The location of the JVM must be specified on your PATH. See [JVM Arguments](#) on page 956.

JVM Class Path: Specifies the CLASSPATH for the JVM used by the driver. See [JVM Classpath](#) on page 957.

Proxy Host: Specifies the Hostname and possibly the Domain of the Proxy Server. See [Proxy Host](#) on page 962.

Proxy Port: Specifies the port needed to connect to the Proxy Server. See [Proxy Port](#) on page 964.

Proxy User: Specifies the user name needed to connect to the Proxy Server. See [ProxyUser](#).

Proxy Password: Specifies the password needed to connect to the Proxy Server. See [Proxy Password](#) on page 963.

Server Port Number: Specifies a valid port on which the SQL engine listens for requests from the driver. By default, the server listens on port 19928. See [Server Port Number](#) on page 969 for more information.

Java Path: Specifies fully qualified path to the Java SE 8 or higher JVM executable that you want to use to run the SQL Engine Server. The path must not contain double quotation marks.

Server DB Directory: Specifies the path of the working directory for the SQL engine service to use to store the newly created database files or locate the existing database files. If the Database connection option contains a file name prefix, the user's local database is created at the path specified by Server DB Directory. However, if the Database connection option contains a fully qualified path, the user's local database is created using that path; the path specified by Server DB Directory is ignored.

Services: Shows the Salesforce ODBC SQL engine service that runs as a separate process instead of being loaded within the process of an ODBC application.

Start (Stop): Starts or stops the Salesforce service. A message window is displayed, confirming that the Salesforce service was started or stopped.

Apply: Applies the changes.

3. When you complete your changes, click **Apply**.
4. Click **OK** to save the changes and return to the SQL Engine tab or click **Cancel**.

Starting the SQL Engine Server

In server mode, you must start the SQL engine server before using the driver. Before starting the SQL engine server, choose a directory to store the local database files. Make sure that you have the correct permissions to write to this directory.

By default, the JVM Classpath is set to the sforce.jar file in the installation directory.

To start the SQL engine server:

1. Start the ODBC Administrator by selecting its icon from the DataDirect Connect program group.
2. Select a tab:
 - **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

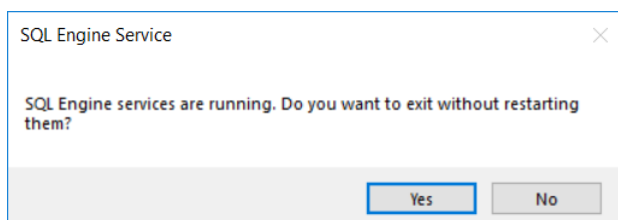
 - **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.

 - **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.
3. On the ODBC Salesforce Driver Setup dialog box, select the **SQL Engine** tab; then, select **1- Server** from the SQL Engine Mode drop-down list.
4. Click **Edit Server Settings**.
5. When you complete your changes, click **Apply**.
6. Verify that Progress DataDirect Salesforce SQL Engine is selected in the Services drop-down list, and then, click **Start** to start the service. A message window appears to confirm that the service is running. Click **OK**.
7. Click **OK** to close the ODBC Salesforce SQL Engine Service Setup dialog box.

Note: If you made changes after starting the service, a message window is displayed:



If you want the service to run with the new settings, click **No**. Then, click **Stop** to stop the service, and then click **Start** to restart the service. Then, click **OK** to close the ODBC Salesforce SQL Engine Service Setup dialog box.

Stopping the SQL Engine Server

To stop the SQL engine server:

1. Open the ODBC Salesforce Driver Setup dialog box and select the SQL Engine tab.
2. Select **1- Server** from the SQL Engine Mode drop-down list. Then, click **Edit Server Settings**.
3. Click **Stop** to stop the service. A message window appears to confirm that the service is stopped. Click **OK**.
4. Click **OK** to close the ODBC Salesforce SQL Engine Service Setup dialog box.

Configuring the SQL Engine Server on UNIX/Linux

The following sections describe how to configure, start, and stop the SQL Engine Server on UNIX and Linux platforms.

By default, the driver operates in direct mode by default on UNIX and Linux platforms.

Configuring and Starting the SQL Engine Server on UNIX/Linux

In server mode, you must start the SQL engine server before using the driver. Before starting the SQL engine server, verify that you have the correct permissions to write to the directory specified by the SchemaMap option.

To configure the SQL engine server, specify values for the Java options in the following JVM argument to suit your environment. See the "SQL Engine Server Java Options" table for a description of these options.

```
java -Xmx<heap_size>m -cp "<jvm_classpath>" com.ddtek.sforcecloud.sql.Server
-port <port_number> -Dhttp.proxyHost=<proxy_host> -Dhttp.proxyPort=<proxy_port>
-Dhttp.proxyUser=<proxy_user> -Dhttp.proxyPassword=<proxy_password>
```

For example:

```
java -Xmx1024m -cp "/opt/Progress/DataDirect/ODBC_71_64bit/java/lib/sforce.jar"
com.ddtek.sforcecloud.sql.Server -port 19938 -Dhttp.proxyHost=myhost@mydomain.com
-Dhttp.proxyPort=12345 -Dhttp.proxyUser=JohnQPublic -Dhttp.proxyPassword=secret
```

To start the SQL engine service, execute the JVM Argument after configuring the Java options. A confirmation message is returned once the server is online.

Table 81: SQL Engine Server Java Options

Java Option	Description
Required Java Options	
-cp	Specifies the CLASSPATH for the Java Virtual Machine (JVM) used by the driver. The CLASSPATH is the search string the JVM uses to locate the Java jar files the driver needs. The Salesforce driver's JVM is located on the following path: <i>install_dir/java/lib/sforce.jar</i>

Java Option	Description
-port	Specifies a valid port on which the SQL engine listens for requests from the driver. We recommend specifying one of the following values: <ul style="list-style-type: none"> • 19938 (32-bit drivers) • 19937 (64-bit drivers)
Optional Java Options	
-Xmx	Specifies the maximum memory heap size, in megabytes, for the JVM. The default size is determined by your JVM. We recommend specifying a size no smaller than 1024. Note: Although this option is not required to start the SQL engine server, we highly recommend specifying a value.
-Dhttp.proxyHost	Specifies the Hostname of the Proxy Server. The value specified can be a host name, a fully qualified domain name, or an IPv4 or IPv6 address.
-Dhttp.proxyPort	Specifies the port number where the Proxy Server is listening for HTTP and/or HTTPS requests.
-Dhttp.proxyUser	Specifies the user name needed to connect to the Proxy Server.
-Dhttp.proxyPassword	Specifies the password needed to connect to the Proxy Server.

Stopping the SQL Engine Server

To stop the SQL engine server, choose one of the following:

- Using an application, execute SHUTDOWN SQL.
- From a command line, press `Ctrl + C`.

A message is returned to confirm that the service is stopped.

Configuring Java Logging for the SQL Engine Server

Java logging can be configured by placing a logging configuration file named `ddlog.properties` in the Server DB directory (see [Configuring Server Mode](#) on page 992 for information on configuring Server DB Directory). The simple way to create one of these is to make a copy of the `ddlog.properties` file, which is located in your driver installation directory, in the `install_dir/Sample/Example` subdirectory.

For more information on logging, refer to "Loggers and logging levels" in the *Progress DataDirect for ODBC Drivers Reference*.

Unicode Support

The Salesforce driver is fully Unicode enabled. On UNIX and Linux platforms, the driver supports both UTF-8 and UTF-16. On Windows platforms, the Salesforce driver supports UCS-2/UTF-16 only.

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Advanced Features

The driver supports the following advanced features:

- Failover
- Security
- Connection Pooling

Failover

The Salesforce driver supports reconnection after a session timeout. Salesforce, like most web-based connections has a session timeout associated with it. The Salesforce driver will reconnect to Salesforce if it receives an error from Salesforce indicating the session has timed out. No configuration is needed.

You can configure the Salesforce driver to retry web service fetch operations if the web service operation timed out; Insert, Update and Delete operations are not retried. the WSRetry Count connection option specifies whether the Salesforce driver retries fetch operations and the number of times it retries. The WSTimeout connection option specifies Web Service timeout period.

Security

No configuration is required to use SSL. By default, all communication using the driver is SSL-encrypted. SSL secures the integrity of your data by encrypting information and providing authentication. See [Data Encryption Across the Network](#) on page 91 for an overview.

Depending on how the Salesforce instance is configured, a security token may need to be included with the user id and password. The [Security Token](#) on page 968 connection option specifies the token.

Connection Pooling

The driver supports connection pooling and its related connection options. Connection pooling connection options are located on the [Pooling tab](#) of the driver Setup dialog box. See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling and its implementation.

Parameter Metadata Support

The Salesforce driver supports returning parameter metadata as described in this section.

Insert and Update Statements

The Salesforce driver supports returning parameter metadata for the following forms of Insert and Update statements:

- `INSERT INTO foo VALUES(?, ?, ?)`
- `INSERT INTO foo (col1, col2, col3) VALUES(?, ?, ?)`
- `UPDATE foo SET col1=?, col2=?, col3=? WHERE col1 operator ? [{AND | OR} col2 operator ?]`

where:

operator

is any of the following SQL operators:

`=, <, >, <=, >=, and <>`

Select Statements

The Salesforce driver supports returning parameter metadata for Select statements that contain parameters in ANSI SQL 92 entry-level predicates, for example, such as COMPARISON, BETWEEN, IN, LIKE, and EXISTS predicate constructs. Refer to the ANSI SQL reference for detailed syntax.

Parameter metadata can be returned for a Select statement if one of the following conditions is true:

- The statement contains a predicate value expression that can be targeted against the source tables in the associated FROM clause. For example:

```
SELECT * FROM foo WHERE bar > ?
```

In this case, the value expression "bar" can be targeted against the table "foo" to determine the appropriate metadata for the parameter.

- The statement contains a predicate value expression part that is a nested query. The nested query's metadata must describe a single column. For example:

```
SELECT * FROM foo WHERE (SELECT x FROM y WHERE z = 1) < ?
```

The following Select statements show further examples for which parameter metadata can be returned:

```
SELECT col1, col2 FROM foo WHERE col1 = ? and col2 > ?
SELECT ... WHERE colname = (SELECT col2 FROM t2 WHERE col3 = ?)
SELECT ... WHERE colname LIKE ?
SELECT ... WHERE colname BETWEEN ? and ?
SELECT ... WHERE colname IN (?, ?, ?)
SELECT ... WHERE EXISTS(SELECT ... FROM T2 WHERE col1 < ?)
```

ANSI SQL 92 entry-level predicates in a WHERE clause containing GROUP BY, HAVING, or ORDER BY statements are supported. For example:

```
SELECT * FROM t1 WHERE col = ? ORDER BY 1
```

Joins are supported. For example:

```
SELECT * FROM t1,t2 WHERE t1.col1 = ?
```

Fully qualified names and aliases are supported. For example:

```
SELECT a, b, c, d FROM T1 AS A, T2 AS B WHERE A.a = ? and B.b = ?"
```

Using DataDirect Bulk Load With the Salesforce Driver

The driver supports DataDirect bulk load. Bulk load connection options are located on the [Bulk tab](#) of the driver Setup dialog box. The driver sends data to a Salesforce instance using the Salesforce Bulk API instead of the Web Service API. Using the Bulk API significantly reduces the number of Web service calls the driver uses to transfer data and may improve performance.

See [Using DataDirect Bulk Load](#) on page 101 for a general description of DataDirect bulk load and its implementation.

Error Handling

The Salesforce driver reports errors to the application by returning SQL_ERROR to the failing ODBC API call. The application can then call SQLGetDiagRec to obtain the error details which consist of the following information:

- Description of the probable cause of the error, prefixed by the component that generated the error
- Vendor error code (if applicable)
- String containing the XOPEN SQLState

Driver Errors

An error generated by the driver has the following format:

Syntax

```
[DataDirect][ODBC Salesforce Driver]message
```

Example

```
[DataDirect][ODBC Salesforce Driver]Timeout expired.
```

See also

Check the last ODBC call made by your application for possible problems or contact your ODBC application vendor.

Data Source Errors

An error generated by the remote or local data source has the following format:

Syntax

```
[DataDirect][ODBC Salesforce Driver][Salesforce] message
```

Example

```
[DataDirect][ODBC Salesforce Driver][Salesforce] Invalid Object Name.
```

Refer to your Salesforce documentation for details on the returned message.

Isolation and Lock Levels Supported

Salesforce supports isolation level 0 (read uncommitted).

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

See [SQL Statements and Extensions for the Salesforce Driver](#) for information about the SQL statements and extensions supported by the Salesforce driver.

ODBC Conformance Level

The Salesforce driver extends the standard results returned by the SQLColumns ODBC function to include the IS_EXTERNAL_ID column, as shown in the following table.

Table 82: Extended Functionality for the SQLColumns Function

Column	Data Type	Description
IS_EXTERNAL_ID	VARCHAR (3), NOT NULL	Provides an indication of whether the column can be used as an External ID. External ID columns can be used as the lookup column for insert and upsert operations and foreign-key relationship values. Valid values are: <ul style="list-style-type: none"> YES: The column can be used as an external ID. NO: The column cannot be used as an external ID. The standard catalog table SYSTEM_COLUMNS is also extended to include the IS_EXTERNAL_ID column.

The Salesforce driver supports only the following Level 2 functions:

- SQLColumnPrivileges
- SQLDescribeParam
- SQLForeignKeys
- SQLPrimaryKeys
- SQLProcedures

- SQLTablePrivileges

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The driver supports multiple connections and multiple statements per connection to the Force.com Web Services API.

The Sybase IQ Wire Protocol Driver

The DataDirect Connect XE for ODBC and DataDirect Connect64 XE for ODBC Sybase IQ Wire Protocol driver (the Sybase IQ Wire Protocol driver) support the following database servers:

- SAP IQ
- Sybase IQ

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The Sybase IQ Wire Protocol driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect product for the file name of the Sybase IQ Wire Protocol driver.

Driver Requirements

The driver has no client requirements.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 1009 and [Connection Option Descriptions](#) on page 1011 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment-Specific Information](#) on page 58 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

You can configure and modify data sources by editing the `odbc.ini` file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions](#) on page 1011 lists driver connection string attributes that must be used in the `odbc.ini` file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Sybase IQ)


On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Sybase IQ data source:

1. Start the ODBC Administrator by selecting its icon from the DataDirect Connect program group.

Select a tab:

-  **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.

If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.

If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 106: General tab

ODBC Sybase IQ Wire Protocol Driver Setup

Failover Pooling About
General Advanced Connection Performance

Data Source Name: Help

Description:

Network Address:

User Name:

Database Name:

Use Interfaces File for Connection Information (Optional)

Interfaces File:

Server Name:

Test Connect OK Cancel Apply

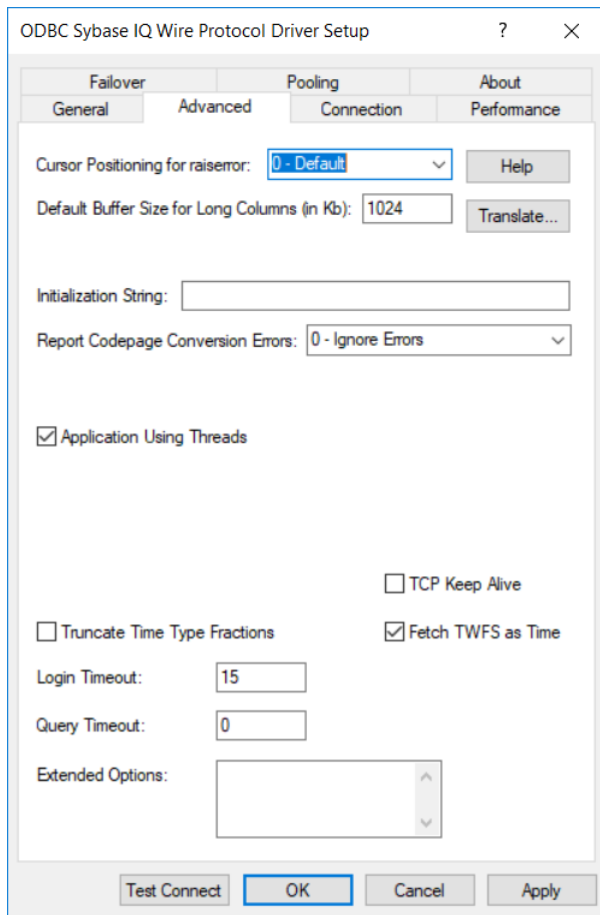
Note: The General tab displays the only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 1019	None
Description on page 1020	None
Network Address on page 1032	None
User Name on page 1037	None
Database Name on page 1020	None
Interfaces File on page 1027	None
Server Name on page 1036	None

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 107: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Cursor Positioning for Raiserror on page 1018	0 - Default
Default Buffer Size for Long Columns (in Kb) on page 1021	1024
Initialization String on page 1027	None
Report Codepage Conversion Errors on page 1035	0 - Ignore Errors
Application Using Threads on page 1014	Enabled
Fetch TWFS as Time on page 1024	Enabled
Truncate Time Type Fractions on page 1037	Disabled
Login Timeout on page 1030	15
Query Timeout on page 1034	0



Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.



Translate : Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

4. Optionally, click the **Connection** tab to specify data source settings.

Figure 108: Connection tab

The screenshot shows the 'ODBC Sybase IQ Wire Protocol Driver Setup' dialog box with the 'Connection' tab selected. The dialog has a title bar with a question mark and a close button. Below the title bar are four tabs: 'Failover', 'Pooling', 'About', and 'Performance'. The 'Connection' tab is active, showing a 'General' sub-tab. The 'General' sub-tab contains the following fields and buttons:

- Database List:** A text input field with a 'Help' button to its right.
- Workstation ID:** A text input field.
- Charset:** A text input field.
- Application Name:** A text input field.
- Language:** A text input field.

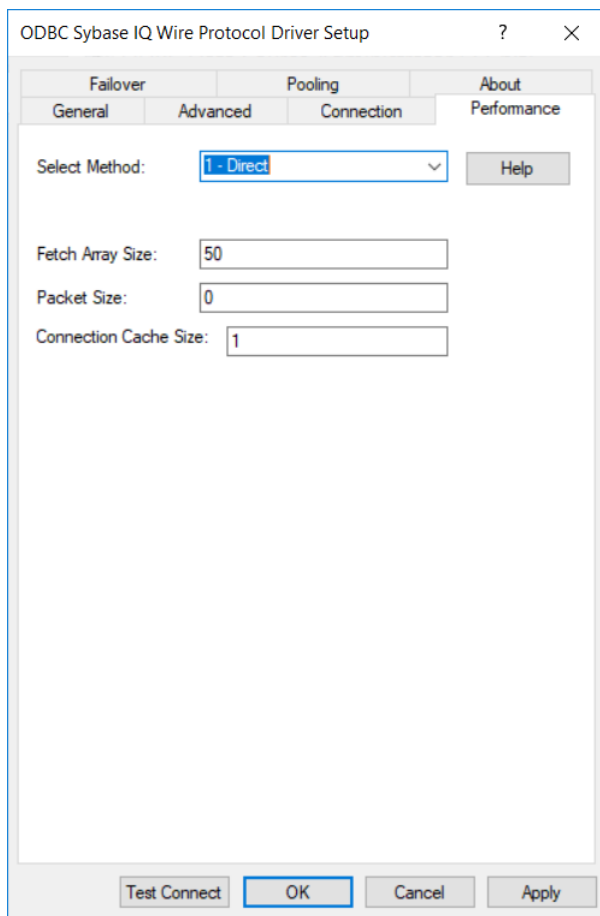
At the bottom of the dialog are four buttons: 'Test Connect', 'OK', 'Cancel', and 'Apply'. The 'OK' button is highlighted with a blue border.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Connection	Default
Database List on page 1019	None
Workstation ID on page 1038	None
Charset on page 1014	None
Application Name on page 1013	None
Language on page 1028	None

- Optionally, click the **Performance** tab to specify performance data source settings.

Figure 109: Performance tab



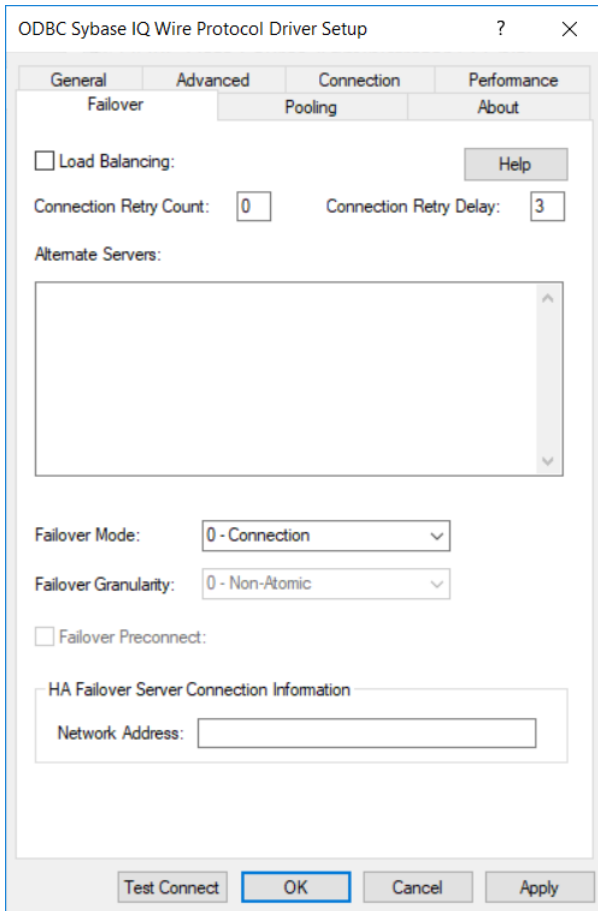
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Performance	Default
Select Method on page 1035	1 - Direct

Connection Options: Performance	Default
Fetch Array Size on page 1024	50
Packet Size on page 1033	0
Connection Cache Size on page 1015	1

6. Optionally, click the **Failover** tab to specify failover data source settings.

Figure 110: Failover tab



See [Using Failover](#) on page 78 for a general description of failover and its related connection options.

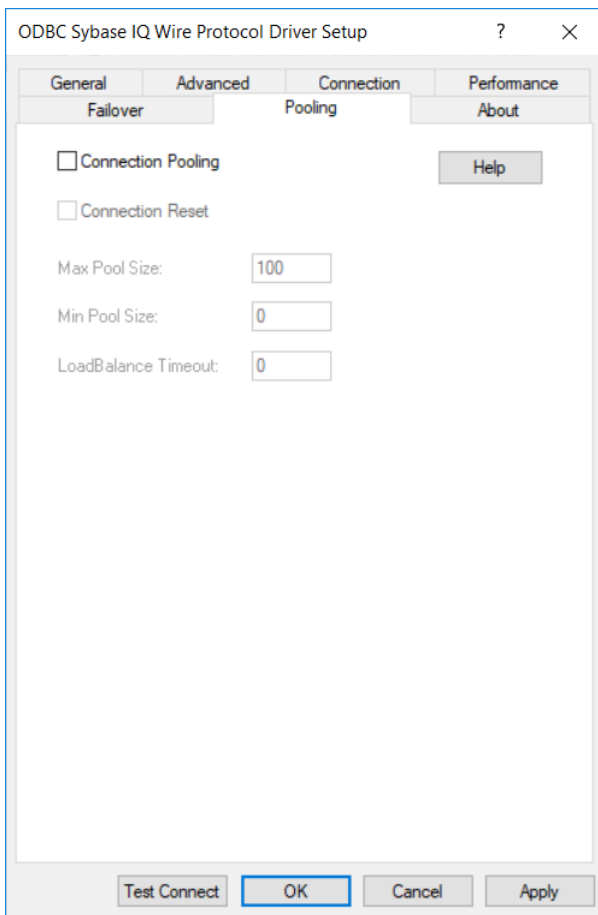
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Failover	Default
Load Balancing on page 1029	Disabled
Connection Retry Count on page 1017	0
Connection Retry Delay on page 1018	3
Alternate Servers on page 1012	None

Connection Options: Failover	Default
Failover Mode on page 1022	0 - Connection
Failover Granularity on page 1022	0 - Non-Atomic
Failover Preconnect on page 1023	Disabled
HA Failover Server Connection Information/Network Address on page 1025	None

- Optionally, click the **Pooling** tab to specify connection pooling data source settings.

Figure 111: Pooling tab



See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling.

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Pooling	Default
Connection Pooling on page 1016	Disabled
Connection Reset on page 1016	Disabled
Max Pool Size on page 1031	100

Connection Options: Pooling	Default
Min Pool Size on page 1031	0
Load Balance Timeout on page 1029	0

8. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Sybase IQ\)](#) on page 1010 for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
- If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

9. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the `DSN=`, `FILEDSN=`, or the `DRIVER=` keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{ }driver_name[]][;attribute=value[;attribute=value]...]
```

[Connection Option Descriptions](#) on page 1011 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Sybase IQ is:

```
DSN=SYBIQTABLES;DB=PAYROLL;UID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=SYBIQ.dsn;DB=PAYROLL;UID=JOHN;PWD=XYZZY
```

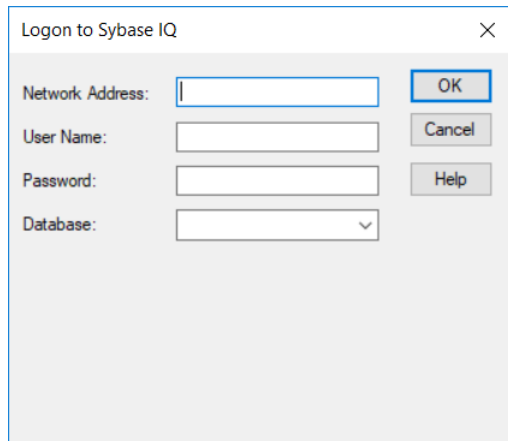
A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Sybase IQ Wire Protocol};NA=123.456.78.90,2638;  
DB=SYBIQACCT;UID=JOHN;PWD=XYZZY
```

Using a Logon Dialog Box (Sybase IQ)

Some ODBC applications display a Logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Figure 112: Logon to SybaseIQ dialog box



In the Logon dialog box, provide the following information:

1. In the Network Address field, specify an IP address for the Sybase IQ server as follows: *IP address,port_number*. For example, you might enter *199.226.224.34,2638*. If your network supports named servers, you can specify an address as: *servername,port_number*. For example, you might enter *SybIQserver,2638*.

The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details concerning these formats.

2. If required, type your case-sensitive login ID.
3. If required, type your case-sensitive password for the system.
4. In the Database field, type the name of the database you want to access (case-sensitive). Or, select the name from the Database drop-down list, which displays the names that you specified on the Connection tab of the ODBC Sybase IQ Wire Protocol driver Setup dialog box.

Note: If you are connecting through the **Test Connect** button of the Setup dialog box, only the default database specified on the General tab of the Setup dialog box is available in the Database drop-down list. The database names specified on the Connection tab are not available.

5. Click **OK** to complete the logon and to update the values in the Registry.

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the Sybase IQ Wire Protocol driver.

Table 83: Sybase IQ Wire Protocol Attribute Names

Attribute (Short Name)	Default
AlternateServers (ASRV)	None
ApplicationName (APP)	None
ApplicationUsingThreads (AUT)	1 (Enabled)
ArraySize (AS)	50
Charset (CS)	None
ConnectionReset (CR)	0 (Disabled)
ConnectionRetryCount (CRC)	0
ConnectionRetryDelay (CRD)	3
ConnectionCacheSize (CCS)	1
Database (DB)	None
Database List	None
DataSourceName (DSN)	None
DefaultLongDataBuffLen (DLDBL)	1024
Description (n/a)	None
FailoverGranularity (FG)	0 (Non-Atomic)
FailoverMode (FM)	0 (Connection)
FailoverPreconnect (FP)	0 (Disabled)
FetchTWFSasTime (FTWFSAT)	1 (Enabled)

Attribute (Short Name)	Default
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
InitializationString (IS)	None
InterfacesFile (IF)	None
InterfacesFileServerName (IFSN)	None
Language (LANG)	None
LoadBalanceTimeout (LBT)	0
LoadBalancing (LB)	0 (Disabled)
LoginTimeout (LT)	15
LogonID (UID)	None
MaxPoolSize (MXPS)	100
MinPoolSize (MNPS)	0
NetworkAddress (NA)	None
PacketSize (PS)	0
Password (PWD)	None
Pooling (POOL)	0 (Disabled)
QueryTimeout (QT)	0
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
SelectMethod (SM)	1 (Direct)
TruncateTimeTypeFractions (TTTF)	0 (Disabled)
WorkstationID (WKID)	None

Alternate Servers

Attribute

AlternateServers (ASRV)

Purpose

A list of alternate database servers to which the driver tries to connect if the primary database server is unavailable. Specifying a value for this option enables connection failover for the driver. The value you specify must be in the form of a string that defines the physical location of each alternate server. All of the other required connection information for each alternate server is the same as what is defined for the primary server connection.

Valid Values

```
({NetworkAddress=addressvalue | InterfacesFileServerName=sectionvalue}[, ...])
```

NetworkAddress and InterfacesFileServerName can be used in the same string.

Notes

- An alternate server address in IPv6 format must be enclosed in double quotation marks.
- You must specify the network address of each alternate database server or the section in the Interfaces file that contains the network connection information for the Sybase IQ database server you want to access (InterfacesFileServerName).
- The Alternate Servers option and the HA Failover Server Connection Information option are mutually exclusive.

Example

The following example Alternate Servers values define three alternate database servers for connection failover:

```
(InterfacesFileServerName=Accounting, NetworkAddress="255.125.1.11, 4200",  
NetworkAddress="SybaseIQ2, 4200")
```

In this example, the network address of the last two alternates contain commas. In this case, enclose the network address with double quotation marks as shown.

Default

None

GUI Tab

[Failover tab](#)

Application Name

Attribute

ApplicationName (APP)

Purpose

The name used by Sybase IQ to identify your application.

Valid Values

string

where:

string

is a valid application name.

Default

None

GUI Tab

[Connection tab](#)

Application Using Threads

Attribute

ApplicationUsingThreads (AUT)

Purpose

Determines whether the driver works with applications using multiple ODBC threads.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

Notes

- This connection option can affect performance.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Charset

Attribute

Charset (CS)

Purpose

The name of a character set installed on the Sybase IQ server to be used by the driver.

This option is not a substitute for the `IANAAppCodePage` option. See [IANAAppCodePage](#) on page 1026 for details.

Valid Values

charset

where:

charset

is the name of a character set installed on the Sybase IQ server.

Behavior

If unspecified, the character set setting on the Sybase IQ server is used.

For the driver to return Unicode SQL types for connections to Sybase IQ 15.0 and higher, use a value of UTF-8. Refer to the Sybase IQ server documentation for a list of valid character sets.

Example

If your client needs to receive data in iso-8859-1 from a non-Unicode Sybase IQ server, you would specify a value of `iso_1`.

Default

None

GUI Tab

[Connection tab](#)

Connection Cache Size

Attribute

`ConnectionCacheSize` (CCS)

Purpose

The number of connections that the connection cache can hold.

Valid Values

x

where:

x

is a positive integer representing the number of connections that the connection cache can hold.

To enable the connection cache, you must set the `Select Method` option to 1 (Direct). Increasing the connection cache may increase performance of some applications but requires additional database resources.

Default

1

GUI Tab

[Performance tab](#)

Connection Pooling

Attribute

Pooling (POOL)

Purpose

Specifies whether to use the driver's connection pooling.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses connection pooling.

If set to 0 (Disabled), the driver does not use connection pooling.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Connection Reset

Attribute

ConnectionReset (CR)

Purpose

Determines whether the state of connections that are removed from the connection pool for reuse by the application is reset to the initial configuration of the connection.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the state of connections removed from the connection pool for reuse by an application is reset to the initial configuration of the connection. Resetting the state can negatively impact performance because additional commands must be sent over the network to the server to reset the state of the connection.

If set to 0 (Disabled), the state of connections is not reset.

Notes

- This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Connection Retry Count

Attribute

ConnectionRetryCount (CRC)

Purpose

The number of times the driver retries connection attempts to the primary database server, and if specified, alternate servers until a successful connection is established.

This option and the Connection Retry Delay connection option, which specifies the wait interval between attempts, can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, the driver does not try to connect after the initial unsuccessful attempt.

If set to x, the driver retries connection attempts the specified number of times. If a connection is not established during the retry attempts, the driver returns an error that is generated by the last server to which it tried to connect.

Default

0

GUI Tab

[Failover tab](#)

Connection Retry Delay

Attribute

ConnectionRetryDelay (CRD)

Purpose

Specifies the number of seconds the driver waits between connection retry attempts when Connection Retry Count is set to a positive integer.

This option and the Connection Retry Count connection option can be used in conjunction with failover.

Valid Values

0 | x

where:

x

is a positive integer from 1 to 65535.

Behavior

If set to 0, there is no delay between retries.

If set to x, the driver waits the specified number of seconds between connection retry attempts.

Default

3

GUI Tab

[Failover tab](#)

Cursor Positioning for Raiserror

Attribute

RaiseErrorPositionBehavior (REPB)

Purpose

Determines whether the driver returns raiserrors when the next statement is executed or handles them separately.

Valid Values

0 | 1

Behavior

If set to 0 (Default), raiserrors are handled separately from surrounding statements. The error is returned when a raiserror is processed (for example, resulting from SQLExecute, SQLExecDirect, or SQLMoreResults). The result set is empty.

If set to 1 (Microsoft compatible), raiserrors are returned when the next statement is processed, and the cursor is positioned on the first row of the subsequent result set. This could result in multiple raiserrors being returned on a single execute.

Default

0 (Default)

GUI Tab

[Advanced tab](#)

Data Source Name

Attribute

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database List

Attribute

n/a

Description

A list of database names that will appear in the drop-down list of the logon dialog box (see [Using a Logon Dialog Box \(Sybase IQ\)](#) on page 1010 for a description).

Valid Values

database_list

where:

database_list

is a comma-separated list of database names that will appear in the drop-down list of the logon dialog box.

Default

None

GUI Tab

[Connection tab](#)

Database Name

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_name

where:

database_name

is the name of a valid database. If you do not specify a value, the default is the database defined by the system administrator for each user.

Default

None

GUI Tab

[General tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Default Buffer Size for Long Columns (in Kb)

Attribute

DefaultLongDataBuffLen (DLDBL)

Purpose

The maximum length of data (in KB) the driver can fetch from long columns in a single round trip and the maximum length of data that the driver can send using the SQL_DATA_AT_EXEC parameter.

This option also applies to binding long parameters in chunks. The driver truncates any data passed in a Long/LOB SQL_DATA_AT_EXEC parameter to the size specified.

Valid Values

An integer in multiples of 1024

The value must be in multiples of 1024 (for example, 1024, 2048). You need to increase the default value if the total size of any Long data exceeds 1 MB. This value is multiplied by 1024 to determine the total maximum length of fetched data. For example, if you enter a value of 2048, the maximum length of data would be 1024 x 2048, or 2097152 (2 MB).

Notes

- This connection option can affect performance.

Default

1024

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Failover Granularity

Attribute

FailoverGranularity (FG)

Purpose

Determines whether the driver fails the entire failover process or continues with the process if errors occur while trying to reestablish a lost connection.

This option applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select).

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2 | 3

Behavior

If set to 0 (Non-Atomic), the driver continues with the failover process and posts any errors on the statement on which they occur.

If set to 1 (Atomic) the driver fails the entire failover process if an error is generated as the result of anything other than executing and repositioning a Select statement. If an error is generated as a result of repositioning a result set to the last row position, the driver continues with the failover process, but generates a warning that the Select statement must be reissued.

If set to 2 (Atomic Including Repositioning), the driver fails the entire failover process if any error is generated as the result of restoring the state of the connection or the state of work in progress.

If set to 3 (Disable Integrity Check), the driver does not verify that the rows that were restored during the failover process match the original rows. This value applies only when Failover Mode is set to 2 (Select).

Default

0 (Non-Atomic)

GUI Tab

[Failover tab](#)

Failover Mode

Attribute

FailoverMode (FM)

Purpose

Specifies the type of failover method the driver uses.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Connection), the driver provides failover protection for new connections only.

If set to 1 (Extended Connection), the driver provides failover protection for new and lost connections, but not any work in progress.

If set to 2 (Select), the driver provides failover protection for new and lost connections. In addition, it preserves the state of work performed by the last Select statement executed.

Notes

- This connection option can affect performance.

Default

0 (Connection)

GUI Tab

[Failover tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Failover Preconnect

Attribute

FailoverPreconnect (FP)

Purpose

Specifies whether the driver tries to connect to the primary and an alternate server at the same time.

This attribute applies only when Failover Mode is set to 1 (Extended Connection) or 2 (Select) and at least one alternate server is specified.

The Alternate Servers option specifies one or multiple alternate servers for failover and is required for all failover methods.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver tries to connect to an alternate server only when failover is caused by an unsuccessful connection attempt or a lost connection. This value provides the best performance, but your application typically experiences a short wait while the failover connection is attempted.

If set to 1 (Enabled), the driver tries to connect to the primary and an alternate server at the same time. This can be useful if your application is time-sensitive and cannot absorb the wait for the failover connection to succeed.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Fetch Array Size

Attribute

ArraySize (AS)

Purpose

The number of rows the driver retrieves from the server for a fetch. This is not the number of rows given to the user. You should use Fetch Array Size in conjunction with [Select Method](#) on page 1035.

Valid Values

x

where:

x

is a positive integer specifying the number of rows.

Notes

- This connection option can affect performance.

Default

50

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Fetch TWFS as Time

Attribute

FetchTWFSasTime (FTWFSAT)

Purpose

Determines whether the driver returns column values with the time data type as the ODBC data type SQL_TYPE_TIME or SQL_TYPE_TIMESTAMP.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIME`. The fractional seconds portion of the value is truncated.

If set to 0 (Disabled), the driver returns column values with the time data type as the ODBC data type `SQL_TYPE_TIMESTAMP`. The fractional seconds portion of the value is preserved. Time columns are not searchable when they are described and fetched as timestamp.

Notes

- When returning time with fractional seconds data as `SQL_TYPE_TIMESTAMP`, the Year, Month and Day parts of the timestamp must be set to zero.

Default

1 (Enabled)

GUI Tab

[Advanced tab](#)

HA Failover Server Connection Information/Network Address

Attribute

FailoverNetworkAddress (FNA)

Purpose

The network address of the High Availability (HA) Failover server to be used in the event of a connection loss. The driver detects the dropped connection and automatically reconnects to the specified HA Failover server. This option is valid only for Sybase IQ servers that have the High Availability Failover feature enabled.

Valid Values

IP_address , *port_number* | *server_name* , *port_number*

where:

IP_address

is the IP address that uniquely identifies the HA Failover server.

port_number

is the port number assigned to the listener process on the HA Failover server.

server_name

is a name that uniquely identifies the HA Failover server. You can use this format if your environment supports named servers.

Notes

- The HA Failover Server Connection Information option and the Alternate Servers option are mutually exclusive.

Example

```
199.226.224.34, 2638
```

```
Sybaseiqserver, 2638
```

Default

None

GUI Tab

[Failover tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

N/A

Initialization String**Attribute**

InitializationString (IS)

Purpose

A SQL command that is issued immediately after connecting to the database to manage session settings.

Valid Values

SQL_command

where:

SQL_command

is a valid SQL command that is supported by the database.

Notes

- If the statement fails to execute, the connection fails and the driver reports the error returned from the server.

Example

To allow delimited identifiers, specify:

```
InitializationString=set QUOTED_IDENTIFIER on
```

Default

None

GUI Tab

[Advanced tab](#)

Interfaces File**Attribute**

InterfacesFile (IF)

Purpose

The directory to the Interfaces file.

Valid Values

file_dir

where:

file_dir

is the directory to the Interfaces file.

Behavior

If unspecified and a value is specified for the Server Name option, the driver looks for the path name of the Interfaces file in the Registry under HKEY_LOCAL_MACHINE\SOFTWARE\DataDirect\InterfacesFile. If this Registry value is empty, the driver will try to open the SQL.INI file found in the same directory where the driver is located and use it as the Interfaces file.

Notes

- This option and the Network Address option are mutually exclusive.

Default

None

GUI Tab

[General tab](#)

Language

Attribute

Language (LANG)

Purpose

The national character set installed on the Sybase IQ server.

Valid Values

charset

where:

charset

is the national character set installed on the Sybase IQ server.

Default

None (English)

GUI Tab

[Connection tab](#)

Load Balance Timeout

Attribute

LoadBalanceTimeout (LBT)

Purpose

The number of seconds to keep inactive connections open in a connection pool. An inactive connection is a database session that is not associated with an ODBC connection handle, that is, a connection in the pool that is not in use by an application.

Valid Values

0 | x

where

x

is a positive integer that specifies a number of seconds.

Behavior

If set to 0, inactive connections are kept open.

If set to x , inactive connections are closed after the specified number of seconds passes.

Notes

The Min Pool Size option may cause some connections to ignore this value.

This connection option can affect performance.

Default

0 (Disabled)

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Load Balancing

Attribute

LoadBalancing (LB)

Purpose

Determines whether the driver uses client load balancing in its attempts to connect to the database servers (primary and alternate). You can specify one or multiple alternate servers by setting the Alternate Servers option.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver uses client load balancing and attempts to connect to the database servers (primary and alternate servers) in random order.

If set to 0 (Disabled), the driver does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

Notes

- This option has no effect unless alternate servers are defined for the Alternate Servers connection option.

Default

0 (Disabled)

GUI Tab

[Failover tab](#)

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x, the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

15

GUI Tab[Advanced tab](#)**Max Pool Size****Attribute**

MaxPoolSize (MXPS)

Purpose

The maximum number of connections allowed within a single connection pool. When the maximum number of connections is reached, no additional connections can be created in the connection pool.

Valid Values

An integer from 1 to 65535

For example, if set to 20, the maximum number of connections allowed in the pool is 20.

Notes

- This connection option can affect performance.

Default

100

GUI Tab[Pooling tab](#)**See Also**

See [Performance Considerations](#) on page 1038 for details.

Min Pool Size**Attribute**

MinPoolSize (MNPS)

Purpose

The minimum number of connections that are opened and placed in a connection pool, in addition to the active connection, when the pool is created. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Valid Values

0 | x

Behavior

If set to 0, no connections are opened in addition to the current existing connection.

If set to x , the start-up number of connections in the pool is 5 in addition to the current existing connection.

Notes

- This connection option can affect performance.

Example

If set to 5, the start-up number of connections in the pool is 5 in addition to the current existing connection.

Default

0

GUI Tab

[Pooling tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Network Address

Attribute

NetworkAddress (NA)

Purpose

A unique identifier assigned to the Sybase IQ server machine.

Valid Values

server_name | *IP_address*

where:

server_name

is the Sybase IQ server name specified as: *named_server, port_number*. For example, you can enter `SyIQserver, 2638`.

IP_address

is the Sybase IQ server address specified as: *IP_address, port_number*. For example, you can enter `199.226.224.34, 2638`. The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details about these formats.

Notes

- This option is mutually exclusive with the Interfaces File and the Server Name option.

Default

None

GUI Tab

[General tab](#)

Packet Size

Attribute

PacketSize (PS)

Purpose

Determines the number of bytes for each database protocol packet that is transferred from the database server to the client machine. Adjusting the packet size can improve performance. The optimal value depends on the typical size of data that is inserted, updated, or returned by the application and the environment in which it is running. Typically, larger packet sizes work better for large amounts of data. For example, if an application regularly returns character values that are 10,000 characters in length, using a value of 32 (16 KB) typically results in improved performance.

Valid Values

-1 | 0 | x

Behavior

If set to -1, the driver uses the maximum packet size that is set by the database server.

If set to 0, the driver uses the default packet size that is used by the database server.

If set to x, an integer from 1 to 127, the driver uses a packet size that is a multiple of 512 bytes. For example, PacketSize=8 means to set the packet size to 8 * 512 bytes (4096 bytes).

Notes

- The ODBC connection attribute `SQL_ATTR_PACKET_SIZE` provides the same functionality as the Packet Size option; however, `SQL_ATTR_PACKET_SIZE` and the Packet Size option are mutually exclusive. If Packet Size is specified, the driver returns the message `Driver Not Capable` if an application attempts to call `SQLSetConnectAttr()` for `SQL_ATTR_PACKET_SIZE`. If you do not set the Packet Size option, application calls to `SQLSetConnectAttr()` for `SQL_ATTR_PACKET_SIZE` are accepted by the driver.
- This connection option can affect performance.

Default

0

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Query Timeout

Attribute

QueryTimeout (QT)

Purpose

The number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the SQL_ATTR_QUERY_TIMEOUT statement attribute on the SQLSetStmtAttr() function.

Valid Values

-1 | 0 | *x*

where:

x

is a positive integer that specifies a number of seconds.

Behavior

If set to -1, the query does not time out. The driver silently ignores the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to 0, the query does not time out, but the driver responds to the SQL_ATTR_QUERY_TIMEOUT attribute.

If set to *x*, all queries time out after the specified number of seconds unless the application overrides this value by setting the `SQL_ATTR_QUERY_TIMEOUT` attribute.

Default

0

GUI Tab[Advanced tab](#)

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where *x* is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab[Advanced tab](#)

Select Method

Attribute

SelectMethod (SM)

Purpose

Determines whether database cursors are used for Select statements.

Valid Values

0 | 1

Behavior

If set to 0 (Cursor), database cursors are used. In some cases performance degradation can occur when performing large numbers of sequential Select statements because of the amount of overhead associated with creating database cursors.

If set to 1 (Direct), Select statements are run directly without using database cursors, and the data source is limited to one active statement.

Notes

- This connection option can affect performance.

Default

1 (Direct)

GUI Tab

[Performance tab](#)

See Also

See [Performance Considerations](#) on page 1038 for details.

Server Name

Attribute

InterfacesFileName (IFSN)

Purpose

The name of the section in the Interfaces file containing the network connection information for the Sybase IQ server. Typically, the section name is the host name of the Sybase IQ server.

Valid Values

section_name

where:

section_name

is a section in the Interfaces file containing the network connection information for the Sybase IQ server.

Notes

- The Network Address option and the Server Name option are mutually exclusive.

Default

None

GUI Tab

[General tab](#)

Truncate Time Type Fractions**Attribute**

TruncateTimeTypeFractions (TTTTF)

Purpose

Determines whether the driver sets fractional seconds to zero (0) when converting data from the TIME data type to TIMESTAMP, CHAR, or WCHAR data types.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver converts fractional seconds to zero when converting the TIME data type.

If set to 0 (Disabled), the driver does not set fractional seconds to zero when converting the TIME data type.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

User Name**Attribute**

LogonID (UID)

Description

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

GUI Tab

[General tab](#)

Workstation ID

Attribute

WorkstationID (WKID)

Purpose

An identifier for the client machine.

Valid Values

ID

where:

ID

is workstation ID use by the client machine.

Default

None

GUI Tab

[Connection tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Application Using Threads (ApplicationUsingThreads): The driver coordinates concurrent database operations (operations from different threads) by acquiring locks. Although locking prevents errors in the driver, it also decreases performance. If your application does not make ODBC calls from different threads, the driver has no reason to coordinate operations. In this case, the ApplicationUsingThreads attribute should be disabled (set to 0).

Note: If you are using a multi-threaded application, you must enable the Application Using Threads option.

Connection Pooling (Pooling): If you enable the driver to use connection pooling, you can set additional options that affect performance:

- **Load Balance Timeout (LoadBalanceTimeout):** You can define how long to keep connections in the pool. The time that a connection was last used is compared to the current time and, if the timespan exceeds the value of the Load Balance Timeout option, the connection is destroyed. The Min Pool Size option can cause some connections to ignore this value.

- **Connection Reset (ConnectionRetryCount)**: Resetting a re-used connection to the initial configuration settings impacts performance negatively because the connection must issue additional commands to the server.
- **Max Pool Size (MaxPoolSize)**: Setting the maximum number of connections that the pool can contain too low might cause delays while waiting for a connection to become available. Setting the number too high wastes resources.
- **Min Pool Size (MinPoolSize)**: A connection pool is created when the first connection with a unique connection string connects to the database. The pool is populated with connections up to the minimum pool size, if one has been specified. The connection pool retains this number of connections, even when some connections exceed their Load Balance Timeout value.

Default Buffer Size for Long/LOB Columns (DefaultLongDataBuffLen): To improve performance when your application fetches images, pictures, or long text or binary data, a buffer size can be set to accommodate the maximum size of the data. The buffer size should only be large enough to accommodate the maximum amount of data retrieved; otherwise, performance is reduced by transferring large amounts of data into an oversized buffer. If your application retrieves more than 1 MB of data, the buffer size should be increased accordingly.

Failover Mode (FailoverMode): Although high availability that replays queries after a failure provides increased levels of protection, it can adversely affect performance because of increased overhead.

Fetch Array Size (ArraySize): If the Select Method connection option is set to 0 and your application fetches more than 50 rows at a time, you should set Fetch Array Size to the approximate number of rows being fetched. This reduces the number of round trips on the network, thereby increasing performance. For example, if your application normally fetches 200 rows, it is more efficient for the driver to fetch 200 rows at one time over the network than to fetch 50 rows at a time during four round trips over the network. You should use Fetch Array Size in conjunction with Select Method.

Note: The ideal setting for your application will vary. To calculate the ideal setting for this option, you must know the size in bytes of the rows that you are fetching and the size in bytes of your Network Packet. Then, you must calculate the number of rows that will fit in your Network Packet, leaving space for packet overhead. For example, suppose your Network Packet size is 1024 bytes and the row size is 8 bytes. Dividing 1024 by 8 equals 128; however, the ideal setting for Fetch Array Size is 127, not 128, because the number of rows times the row size must be slightly smaller than the Network Packet size.

Packet Size (PacketSize): Typically, it is optimal for the client to use the maximum packet size that the database server allows. This reduces the total number of round trips required to return data to the client, thus improving performance. Therefore, performance can be improved if the PacketSize attribute is set to the maximum packet size of the Sybase IQ server.

Select Method (SelectMethod): If your application often executes a SQL statement before processing or closing the previous result set, then it uses multiple active statements per connection. An active statement is defined as a statement where all the result rows or result sets have not been fetched. Using multiple active statements can cause high overhead on the server. The default setting (1) of this option causes the driver to execute statements directly without the use of database cursors and limits the application to one active statement per connection. If your application requires multiple active statements, then set Select Method to 0 (Cursor). Keep in mind that you may see a negative impact in performance. If this option is set to 0, it should be used in conjunction with Fetch Array Size (ArraySize). If this option is set to 1, Fetch Array Size (ArraySize) has no effect.

Data Types

The following table shows how the Sybase IQ data types are mapped to the standard ODBC data types. [Unicode Support](#) on page 1041 lists Sybase IQ to Unicode data type mappings.

Table 84: Sybase IQ Data Type Mapping

Sybase IQ Data Type...	Maps to ODBC Data Type
BIGINT	SQL_BIGINT
BINARY	SQL_BINARY
BIT	SQL_BIT
CHAR	SQL_CHAR
DATE	SQL_TYPE_DATE
DATETIME	SQL_TYPE_TIMESTAMP
DECIMAL	SQL_DECIMAL
DOUBLE	SQL_DOUBLE
IMAGE	SQL_LONGVARBINARY
INT	SQL_INTEGER
LONG BINARY	SQL_LONGVARBINARY
LONG VARCHAR	SQL_LONGVARCHAR
MONEY	SQL_DECIMAL
NUMERIC	SQL_NUMERIC
REAL	SQL_REAL
SMALLDATETIME	SQL_TYPE_TIMESTAMP
SMALLINT	SQL_SMALLINT
SMALLMONEY	SQL_DECIMAL
TEXT	SQL_LONGVARCHAR
TIME	SQL_TYPE_TIME
TIMESTAMP	SQL_TYPE_TIMESTAMP
TINYINT	SQL_TINYINT
UNIQUEIDENTIFIER	SQL_BINARY
UNIQUEIDENTIFIERSTR	SQL_CHAR
UNSIGNED BIGINT	SQL_BIGINT

Sybase IQ Data Type...	Maps to ODBC Data Type
UNSIGNED INT	SQL_INTEGER
VARBINARY	SQL_VARBINARY
VARCHAR	SQL_VARCHAR

Note: The Sybase IQ Wire Protocol driver supports extended new limits (XNL) for character and binary columns—columns with lengths greater than 255.

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

Unicode Support

When connected to a Unicode database, the Sybase IQ Wire Protocol driver supports the Unicode data types listed in the following table, in addition to standard ODBC data types listed in [Data Types](#) on page 1039.

Table 85: Mapping Sybase IQ Data Types to Unicode Data Types

Sybase IQ Data Type. . .	Maps to Unicode Data Type. . .
CHAR ⁷⁹	SQL_WCHAR
LONG VARCHAR	SQL_WLONGVARCHAR
TEXT ⁷⁹	SQL_WLONGVARCHAR
UNIQUEIDENTIFIERSTR	SQL_WCHAR
VARCHAR ⁷⁹	SQL_WVARCHAR

For data types that require the UTF-8 character set, set the Charset connection string attribute. See [Charset](#) on page 1014 for information about using this connection string attribute.

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Advanced Features

The driver supports the following advanced features:

- Failover

⁷⁹ This data type is available only if the data source is configured to use the UTF-8 character set.

- Connection Pooling

Failover

The driver supports failover and its related connection options. Failover connection options are located on the [Failover tab](#) of the driver Setup dialog box. See [Using Failover](#) on page 78 for a general description of failover and its implementation.

Connection Pooling

The driver supports connection pooling and its related connection options. Connection pooling connection options are located on the [Pooling tab](#) of the driver Setup dialog box. See [Using DataDirect Connection Pooling](#) on page 97 for a general description of connection pooling and its implementation.

Unexpected Characters

Users are sometimes surprised when they insert a character into a database, only to have a different character displayed when they fetch it from the database. There are many reasons this can happen, but it most often involves code page issues, not driver errors.

Client and server machines in a database system each use code pages, which can be identified by a name or a number, such as Shift_JIS (Japanese) or cp1252 (Windows English). A code page is a mapping that associates a sequence of bits, called a code point, with a specific character. Code pages include the characters and symbols of one or more languages. Regardless of geographical location, a machine can be configured to use a specific code page. Most of the time, a client and database server would use similar, if not identical, code pages. For example, a client and server might use two different Japanese code pages, such as Shift_JIS and EUC_JP, but they would still share many Japanese characters in common. These characters might, however, be represented by different code points in each code page. This introduces the need to convert between code pages to maintain data integrity. In some cases, no one-to-one character correspondence exists between the two code points. This causes a substitution character to be used, which can result in displaying an unexpected character on a fetch.

When the driver on the client machine opens a connection with the database server, the driver determines the code pages being used on the client and the server. This is determined from the Active Code Page on a Windows-based machine.

If the client and server code pages are compatible, the driver transmits data in the code page of the server. Even though the pages are compatible, a one-to-one correspondence for every character may not exist. If the client and server code pages are completely dissimilar, for example, Russian and Japanese, then many substitutions occur because very few, if any, of the characters are mapped between the two code pages.

The following is a specific example of an unexpected character:

- The Windows client machine is running code page cp1252.
- The Sybase IQ server is running code page cp850.
- You insert decimal literals for character data. You think you are inserting LATIN SMALL LETTER I WITH ACUTE (*i*) and BOX DRAWINGS DOUBLE VERTICAL (||) in the database. When you fetch the data, you see INVERTED EXCLAMATION MARK (*¡*) and MASCULINE ORDINAL INDICATOR (*º*) displayed on the client instead.

This occurs because the code points do not correspond in the two code pages. An example of syntax you would use to insert the decimal literals is:

```
CREATE table cp850chars(val text)
INSERT INTO cp850chars values(CHAR(161)+CHAR(186))
```

This effectively inserts the hexadecimal bytes for the numbers 161 (0xA1) and 186 (0xBA) into the text column. Each of these hexadecimal bytes is treated as the single byte code point for the character it represents. The problem is that the character representation for these two particular hexadecimal values is different from code page cp850 to code page cp1252. On cp850, these hexadecimal values represent í (0xA1) and || (0xBA), which is what you thought you were inserting by using the previously described syntax. When you fetch these hexadecimal values, however, the characters displayed on your client machine are ÿ (0xA1) and ° (0xBA), because that is what the hexadecimal values represent in code page cp1252. This is not a matter of data corruption or substitution; these hexadecimal values simply represent different values in the two different code pages.

This is not a driver error. It occurs because the code points map differently and because some characters do not exist in a code page. The best way to avoid these problems is to use the same code page on both the client and server machines.

NULL Values

When the Sybase IQ Wire Protocol driver establishes a connection, the driver sets the Sybase database option `ansinull` to `on`. Setting `ansinull` to `on` ensures that the driver is compliant with the ANSI SQL standard, which makes developing cross-database applications easier.

By default, Sybase IQ does not evaluate NULL values in SQL equality (`=`), inequity (`<>`), or aggregate function comparisons in an ANSI SQL-compliant manner. For example, the ANSI SQL specification defines that `col1=NULL` always evaluates to `false`:

```
SELECT * FROM table WHERE col1 = NULL
```

Using the default database setting (`ansinull=off`), the same comparison evaluates to `true` instead of `false`.

Setting `ansinull` to `on` changes the default database behavior so that SQL statements must use `IS NULL` instead of `=NULL`. For example, using the Sybase IQ Wire Protocol driver, if the value of `col1` in the following statement is `NULL`, the comparison evaluates to `true`:

```
SELECT * FROM table WHERE col1 IS NULL
```

In your application, you can restore the default Sybase IQ behavior for a connection in the following ways:

- Use the Initialization String option to specify the SQL command `set ANSINULL off`. For example, the following connection string ensures that the handling of NULL values is restored to the Sybase IQ default for the current connection:

```
DSN=SYB TABLES;DB=PAYROLL;IS=set ANSINULL off
```

- Explicitly execute the following statement after the connection is established:

```
SET ANSINULL OFF
```

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

The Sybase IQ database system supports isolation levels 0 (read uncommitted), 1 (read committed, the default), 2 (repeatable read), and 3 (serializable). It supports page-level locking.

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the minimum SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions. In addition, the driver supports the following Level 2 functions:

- SQLColumnPrivileges
- SQLForeignKeys
- SQLTablePrivileges

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The Sybase IQ database system supports multiple connections and multiple statements per connection. If the Select Method option on the Performance tab or the connection string attribute SelectMethod is set to 1 (Direct), Sybase IQ data sources are limited to one active statement in manual commit mode.

Using Arrays of Parameters

When designing an application, using parameter arrays for bulk inserts or updates, for example, can improve performance.

Refer to "Designing ODBC applications for performance optimization" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Because Sybase IQ databases do not support parameter arrays natively, the Sybase IQ Wire Protocol driver emulates them by sending T-SQL batches of Insert or Update statements to the database, which will improve performance.

The Driver for Apache Hive

Note: This section documents the features and functionality of the 7.1 version of the driver. For the current version of the driver, visit Progress DataDirect Connectors Documentation page:

<https://docs.progress.com/datadirect-connectors/page/DataDirect-Connectors-by-data-source.html>.

The DataDirect Connect XE *for* ODBC and DataDirect Connect64 XE *for* ODBC for Apache Hive™ Wire Protocol driver each support the following Apache Hive versions and distributions:

- Amazon Elastic MapReduce (Amazon EMR)
- Apache Hadoop Hive
- Cloudera Distribution for Apache Hadoop (CDH)
- Hortonworks Distribution for Apache Hadoop (HDP)
- IBM BigInsights
- MapR Distribution for Apache Hadoop
- Pivotal Enterprise HD

For the latest support information, visit the Progress DataDirect Supported Configurations page:

<https://www.progress.com/supported-configurations/datadirect>.

The driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect driver for the file name of the driver.

Driver Requirements

The driver has no client requirements.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 1051 and [Connection Option Descriptions for Apache Hive](#) on page 1052 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX odbc.ini File

UNIX® On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, `odbc.ini`). You can configure and modify data sources directly by editing the `odbc.ini` file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions for Apache Hive](#) on page 1052 lists driver connection string attributes that must be used in the `odbc.ini` file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Data Source Configuration through a GUI (Hive)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.



On UNIX and Linux, data sources are stored in the `odbc.ini` file.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.



To configure a data source for Apache Hive:

1. Start the ODBC Administrator:
 - On Windows, start the ODBC Administrator by selecting its icon from the Datadirect Connect program group.
2. Select a tab:
 - **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
 - **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
 - **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.
If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 113: General tab

The screenshot shows the 'ODBC Apache Hive Wire Protocol Driver Setup' dialog box with the 'General' tab selected. The fields are as follows:

- Data Source Name: Apache Hive Wire Protocol
- Description: (empty)
- Host Name: (empty)
- Port Number: 10000
- Database Name: default
- Wire Protocol Version: 0 - AutoDetect

Buttons at the bottom include 'Test Connect', 'OK', 'Cancel', and 'Apply'.

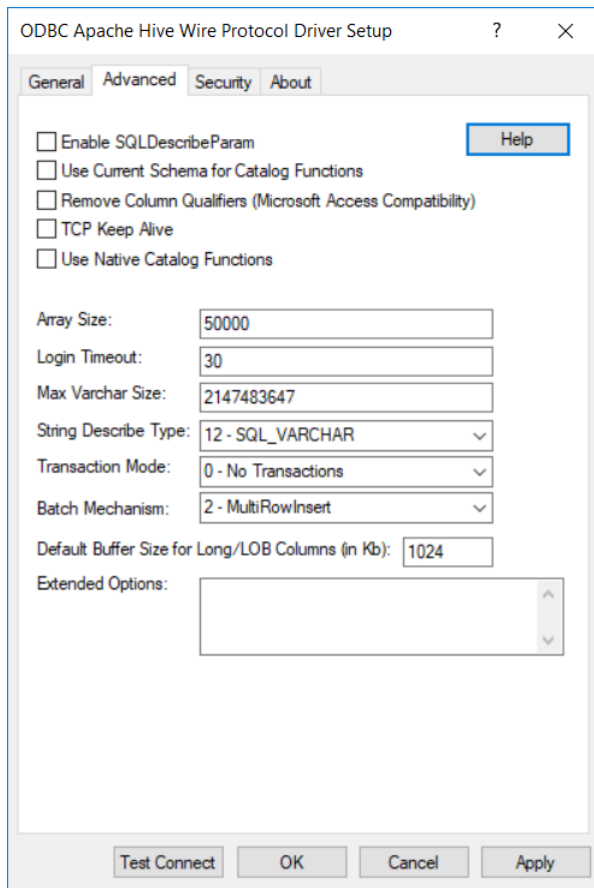
Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

- On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Data Source Name on page 1059	None
Description on page 1061	None
Host Name on page 1063	None
Port Number on page 1068	10000
Database on page 1060	default
Wire Protocol Version on page 1077	0 - AutoDetect

- Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 114: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Enable SQLDescribeParam on page 1061	Disabled
Use Current Schema for Catalog Functions on page 1075	Disabled
Remove Column Qualifiers on page 1069	Disabled
TCP Keep Alive on page 1073	Disabled
Use Native Catalog Functions on page 1076	Disabled
Array Size on page 1055	50000
Login Timeout on page 1066	30
Max Varchar Size on page 1067	2147483647
String Describe Type on page 1072	12 - SQ_VARCHAR
Transaction Mode on page 1073	0 - No Transactions

Connection Options: Advanced	Default
Batch Mechanism on page 1056	2- MultiRowInsert
Default Buffer Size for Long/LOB Columns (in Kb) on page 1060	1024

Extended Options: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect customer support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1; UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Note: Do not specify the Extended Options configuration option in a connection string, or the driver will return an error. Instead, applications should specify the individual undocumented connection options in the connection string.

- Optionally, click the **Security** tab to specify security settings.

Figure 115: Security tab

ODBC Apache Hive Wire Protocol Driver Setup

General Advanced **Security** About

Help

Authentication

Authentication Method: 0 - User ID/Password

User Name:

Proxy User:

Service Principal Name:

GSS Client Library: native

Encryption

Encryption Method: 0 - No Encryption

Crypto Protocol Version

TLSv1.2 TLSv1.1 TLSv1

SSLv3 SSLv2

Validate Server Certificate

Trust Store:

Trust Store Password:

Key Store:

Key Store Password:

Key Password:

Host Name In Certificate:

Test Connect OK Cancel Apply

On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options:Security	Default
Authentication Method on page 1056	0 - User ID/Password
User Name on page 1076	None
Proxy User on page 1069	None
Service Principal Name on page 1070	None
GSS Client Library on page 1062	native
Encryption Method on page 1062	0 (No Encryption)
Crypto Protocol Version on page 1057	TLSv1.2, TLSv1.1, TLSv1, SSLv3
Validate Server Certificate on page 1077	1 (Enabled)
Truststore on page 1074	None
Trust Store Password on page 1075	None
Key Store on page 1065	None
Keystore Password on page 1066	None
Key Password on page 1065	None
Host Name In Certificate on page 1064	None

6. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Hive\)](#) on page 1051) for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
 - If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message. Click **OK**.

Note: If you are configuring alternate servers for use with the connection failover feature, be aware that the Test Connect button tests only the primary server, not the alternate servers.

7. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]. . .]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]. . .]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{driver_name}] [;attribute=value[;attribute=value]. . .]
```

[Connection Option Descriptions for Apache Hive](#) on page 1052 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Hive is:

```
DSN=Accounting;UID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Hive.dsn;UID=JOHN;PWD=XYZZY
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Hive};HOST=server1;PORT=10000;UID=JOHN;PWD=XYZZY;
```

Using a Logon Dialog Box (Hive)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Note: A user name and password are not required to connect to Hive.

Figure 116: Logon to Apache Hive dialog box

Note: To configure a standard connection, complete the first two fields and skip to Step 4.

In this dialog box, provide the following information:

1. In the Host field, type either the name or the IP address of the server to which you want to connect.
2. The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. See [Using IP Addresses](#) on page 67 for details concerning these formats.
3. In the Port Number field, type the port number that your Hive server is listening on. Check with your Hive administrator for the correct number.
4. Click **OK** to log on to the Apache Hive server you specified and to update the values in the Registry.

Note: The User Name and Password fields are not used at this time to connect to Apache Hive Server.

Connection Option Descriptions for Apache Hive

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

The following table lists the connection string attributes supported by the driver.

Table 86: Attribute Names for the Driver for Apache Hive

Attribute (Short Name)	Default
AllowedOpenSSLVersions (AOV)	1.1.1,1.0.2
ArraySize (AS)	50000
AuthenticationMethod (AM)	0 - User ID/Password

Attribute (Short Name)	Default
BatchMechanism (BM)	2 (MultiRowInsert)
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	TLSv1.2, TLSv1.1, TLSv1,SSLv3
Database (DB)	default
DataSourceName (DSN)	None
Description (n/a)	None
DefaultLongDataBuffLen (DLDBL)	1024
EnableDescribeParam (EDP)	0 (Disabled)
EncryptionMethod (EM)	0 (No Encryption)
GSSClient (GSSC)	native
HostName (HOST)	None
HostNameInCertificate (HNIC)	None
KeepAlive (KA)	Disabled
KeyPassword (KP)	None
Keystore (KS)	None
KeystorePassword (KSP)	None
LoginTimeout (LT)	30
LogonID (UID)	None
MaxVarcharSize (MVS)	2147483647
Password (PWD)	None
PortNumber (PORT)	10000
ProxyUser (PU)	None
RemoveColumnQualifiers (RCQ)	0 (Disabled)
ServicePrincipalName (SPN)	None
SSLlibName (SLN)	Empty string
StringDescribeType (SDT)	12 - SQL_VARCHAR

Attribute (Short Name)	Default
TransactionMode (TM)	0 (No Transactions)
Truststore (TS)	None
TruststorePassword (TSP)	None
UseCurrentSchema (UCS)	0 (Disabled)
UseNativeCatalogFunctions (UNCF)	0 (Disabled)
ValidateServerCertificate (VSC)	1 (Enabled)
WireProtocolVersion (WPV)	0 - AutoDetect

AllowedOpenSSLVersions

Attribute

AllowedOpenSSLVersions (AOV)

Purpose

Important: Version 1.0.2 of the OpenSSL library has reached the end of its product life cycle and is no longer receiving security updates. Best security practices dictate that you use the latest version of the library.

Determines which version of the OpenSSL library file the driver uses for data encryption. Although the latest version of the OpenSSL library is the most secure, some characteristics of the library can cause connections to certain databases to fail. This option allows you to continue using older versions of the OpenSSL library while you transition your environment to support the latest version.

Valid Values

`latest | openssl_version_number[[, openssl_version_number] ...]`

where:

openssl_version_number

is the version number for the OpenSSL library file to be loaded by the driver, for example, 1.0.2. When more than one version is specified, the driver will first attempt to load the first version listed. If the driver is unable to locate and load this file, it will attempt to load the next version in the value. The driver currently supports versions 1.1.1 and 1.0.2. Refer to the installed readme for latest supported versions.

Behavior

If set to `latest`, the driver loads the latest installed version of the OpenSSL library file provided by Progress.

If set to `openssl_version_number`, the driver loads the specified version of the OpenSSL library file. This value is used to specify a version other than the latest.

Notes

- This option is ignored if OpenSSL library files are specified using the CryptoLibName and SSLLibName options.
- This option works only with OpenSSL library files provided by Progress and user supplied OpenSSL library files that match Progress's naming convention and installation location.
- This option works only for installations using the default directory structure.
- Consult your database administrator concerning the security settings of your server.

Default

1.1.1,1.0.2

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
AllowedOpenSSLVersions=1.0.2
```

See also

- [Advanced tab](#)

Array Size

Attribute

ArraySize (AS)

Purpose

The number of cells the driver retrieves from a server for a fetch. When executing a fetch, the driver divides the value specified by the number columns in a particular table to determine the number of rows to retrieve. By determining the fetch size based on the number of cells, the driver can avoid out of memory errors when fetching from tables containing a large number of columns while continuing to provide improved performance when fetching from tables containing a small number of columns.

Valid Values

x

where:

x

is a positive integer specifying the number of cells the driver retrieves for a fetch.

Notes

- You can improve performance by increasing the value specified for this option; however, if the number of cells specified exceeds the available buffer memory for the Apache Hive server, an out of memory error will be returned. If you receive this error, decrease the value specified until fetches are successfully executed.
- This connection option can affect performance.

Default

50000

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 1078 for details.

Authentication Method

Attribute

AuthenticationMethod (AM)

Purpose

Specifies the method the driver uses to authenticate the user to the server when a connection is established. If the specified authentication method is not supported by the database server, the connection fails and the driver generates an error.

Valid Values

0 | 4 | -1

Behavior

If set to 0 (User ID/Password), the driver sends the user ID in clear text and an encrypted password to the server for authentication.

If set to 4 (Kerberos Authentication), the driver uses Kerberos authentication. This method supports both Windows Active Directory Kerberos and MIT Kerberos environments.

Note: This setting is supported only for HiveServer2 connections.

If set to -1 (No Authentication), the driver sends the user ID and password in clear text to the server for authentication.

Default

0 (User ID/Password)

GUI Tab

[Security tab](#)

Batch Mechanism

Attribute

BatchMechanism (BM)

Purpose

Determines the mechanism that is used to execute batch operations.

Valid Values

1 | 2

Behavior

If set to 1 (SingleInsert), the driver executes an insert statement for each row contained in a parameter array. Select this setting if you are experiencing out-of-memory errors when performing batch inserts.

If set to 2 (MultiRowInsert), the driver attempts to execute a single insert statement for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the driver, the driver executes multiple statements. Select this setting for substantial performance gains when performing batch inserts.

Default

2 (MultiRowInsert)

Notes

- This connection option can affect performance.

GUI Tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 1078 for details.

Crypto Protocol Version

Attribute

CryptoProtocolVersion (CPV)

Purpose

Specifies a comma-separated list of the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (EncryptionMethod=1). When multiple protocols are specified, the driver uses the highest version supported by the server. If none of the specified protocols are supported by the database server, behavior is determined by the setting of the EncryptionMethod connection option.

Valid Values

```
cryptographic_protocol [, cryptographic_protocol ]...
```

where:

```
cryptographic_protocol
```

is one of the following cryptographic protocols:

TLSv1.2 | TLSv1.1 | TLSv1 | SSLv3 | SSLv2

Caution: Good security practices recommend using TLSv1 or higher, due to known vulnerabilities in the SSLv2 and SSLv3 protocols.

Example

If your security environment is configured to use TLSv1.2 and TLSv1.1, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.1
```

Notes

- This option is ignored if Encryption Method is set to 0 - No Encryption.
- Consult your database administrator concerning the data encryption settings of your server.

Default

```
TLSv1.2, TLSv1.1, TLSv1
```

GUI Tab

[Security tab](#)

See also

[Encryption Method](#) on page 1062

CryptoLibName

Attribute

CryptoLibName (CLN)

Purpose

The absolute path for the OpenSSL library file containing the cryptographic library to be used by the data source or connection when TLS/SSL is enabled. The cryptographic library contains the implementations of cryptographic algorithms the driver uses for data encryption.

This option allows you to designate a different cryptographic library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the cryptographic library to be used by your data source or connection.

Example

```
C:\Progress\DataDirect\Connect64_for_ODBC_71\  
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

Empty string

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
CryptoLibName=C:\Program Files\Progress\DataDirect\  
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[SSLibName](#) on page 1071

Data Source Name

Attribute

`DataSourceName` (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or `odbc.ini` file.

Valid Values

string

where:

string

is the name of a data source.

Default

None

GUI Tab

[General tab](#)

Database

Attribute

Database (DB)

Purpose

Specifies the name of the Hive database. The database must exist, or the connection attempt will fail.

Valid Values

database_name

where:

database_name

is the name of the Hive database.

Default

default

GUI Tab

[General tab](#)

Default Buffer Size for Long/LOB Columns (in Kb)

Attribute

DefaultLongDataBuffLen (DLDBL)

Purpose

The maximum length of data (in KB) the driver can fetch from long columns in a single round trip and the maximum length of data that the driver can send using the SQL_DATA_AT_EXEC parameter.

Valid Values

An integer in multiples of 1024

The value must be in multiples of 1024 (for example, 1024, 2048). You need to increase the default value if the total size of any Long data exceeds 1 MB. This value is multiplied by 1024 to determine the total maximum length of fetched data. For example, if you enter a value of 2048, the maximum length of data would be 1024 x 2048, or 2097152 (2 MB).

Notes

- This connection option can affect performance.

Default

1024

GUI tab

[Advanced tab](#)

See Also

See [Performance Considerations](#) on page 1078 for details.

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable SQLDescribeParam

Attribute

EnableDescribeParam (EDP)

Purpose

Determines whether the driver uses the SQLDescribeParam function, which describes parameters as a data type of SQL_VARCHAR with a length of 255 for statements.

Valid Values

0 | 1

Behavior

If set to 1 (enabled), the SQLDescribeParam function describes parameters as a data type of SQL_VARCHAR with a length of 255 for statements.

If set to 0 (disabled), the SQLDescribeParam function returns the standard ODBC error IM001.

Default

0 (Disabled)

GUI tab

[Advanced tab](#)

Encryption Method

Attribute

EncryptionMethod (EM)

Purpose

The method the driver uses to encrypt data sent between the driver and the database server. If the specified encryption method is not supported by the database server, the connection fails and the driver returns an error.

Valid Values

0 | 1

Behavior

If set to 0 (No Encryption), data is not encrypted.

If set to 1 (SSL), data is encrypted using the SSL protocols specified in the Crypto Protocol Version connection option.

Notes

- This connection option can affect performance.

Default

0 (No Encryption)

GUI Tab

[Security tab](#)

See also

[Performance Considerations](#) on page 1078

GSS Client Library

Attribute

GSSClient (GSSC)

Purpose

The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC).

The driver uses the path defined by the PATH environment variable for loading the specified client library.

Valid Values

native | *client_library*

where:

client_library

is a GSS client library installed on the client.

Behavior

If set to *client_library*, the driver uses the specified GSS client library.

Note: For MIT Kerberos distributions, you must provide a full path to the MIT Library. For example, the 64-bit version for Windows would use the following value: C:\Program Files\MIT\Kerberos\bin\gssapi64.dll.

If set to *native*, the driver uses the GSS client for Windows Kerberos. All other users must provide the full path to the library name.

Default

native

GUI Tab

[Security tab](#)

Host Name

Attribute

HostName (HOST)

Purpose

The name or the IP address of the server to which you want to connect.

Valid Values

host_name | *IP_address*

where:

hostname

is the name of the Apache Hive server to which you want to connect

IP_address

is the IP address of the server to which you want to connect.

Default

None

GUI Tab

[General tab](#)

Host Name In Certificate

Attribute

HostNameInCertificate (HNIC)

Purpose

A host name for certificate validation when SSL encryption is enabled (Encryption Method=1) and validation is enabled (Validate Server Certificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid Values

`host_name` | `#SERVERNAME#`

where:

`host_name`

is the host name specified in the certificate. Consult your SSL administrator for the correct value.

Behavior

If `host_name` is specified, the driver examines the `subjectAltName` values included in the certificate. If a `dnsName` value is present in the `subjectAltName` values, then the driver compares the value specified for Host Name In Certificate with the `dnsName` value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the `dnsName` value.

If no `subjectAltName` values exist or a `dnsName` value is not in the list of `subjectAltName` values, then the driver compares the value specified for Host Name In Certificate with the `commonName` part of the Subject name in the certificate. The `commonName` typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the `commonName`. If multiple `commonName` parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the `commonName` parts.

If `#SERVERNAME#` is specified, the driver compares the host server name specified as part of a data source or connection string to the `dnsName` or the `commonName` value.

Default

None

GUI Tab

[Security tab](#)

Key Password

Attribute

KeyPassword (KP)

Purpose

Specifies the password used to access the individual keys in the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. Keys stored in a keystore can be individually password-protected. To extract the key from the keystore, the driver must have the password of the key.

Valid Values

key_password

where:

key_password

is the password of a key in the keystore.

Default

None

GUI Tab

[Security tab](#)

Key Store

Attribute

Keystore (KS)

Purpose

The name of the directory containing the keystore file to be used when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request. If you do not specify a directory, the current directory is used.

Valid Values

keystore_directory

where:

keystore_directory

is the location of the keystore file.

Notes

- The keystore and truststore files can be the same file.

Default

None

GUI Tab

[Security tab](#)

Keystore Password

Attribute

KeystorePassword (KSP)

Purpose

The password used to access the keystore file when SSL is enabled (Encryption Method=1) and SSL client authentication is enabled on the database server. The keystore file contains the certificates that the client sends to the server in response to the server's certificate request.

Valid Values

keystore_password

where:

keystore_password

is the password of the keystore file.

Notes

- The keystore and truststore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Login Timeout

Attribute

LoginTimeout (LT)

Purpose

The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

Valid Values

-1 | 0 | x

where:

x

is a positive integer that represents a number of seconds.

Behavior

If set to -1, the connection request does not time out. The driver silently ignores the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to 0, the connection request does not time out, but the driver responds to the SQL_ATTR_LOGIN_TIMEOUT attribute.

If set to x , the connection request times out after the specified number of seconds unless the application overrides this setting with the SQL_ATTR_LOGIN_TIMEOUT attribute.

Default

30

GUI Tab

[Advanced tab](#)

Max Varchar Size

Attribute

MaxVarcharSize (MVS)

Purpose

Specifies the maximum size of columns of type SQL_VARCHAR that the driver describes through result set descriptions and catalog functions.

Valid Values

A positive integer from 255 to x

where:

x

is maximum size of the SQL_VARCHAR data type.

Default

2147483647

GUI Tab

[Advanced tab](#)

Password

Attribute

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Purpose

Specifies the port number of the server listener.

Valid Values

port_number

where:

port_number

is the port number of the server listener. Check with your database administrator for the correct number.

Notes

- The default port number for the Apache Hive server is 10000. Because of reported concurrency issues, you might want to use a different port number.

Default

10000

GUI Tab

[General tab](#)

Proxy User

Attribute

ProxyUser (PU)

Purpose

Specifies the UserID used for HiveServer2 Impersonation and HiveServer2 Trusted Impersonation. When impersonation is enabled on the server, this value determines your identity and access rights to files when executing queries. If no value is provided for this option or if impersonation is disabled, you will execute queries as the user who initiated the HiveServer process.

Impersonation provides a method for administrators to control access to data. Administrators set access rights to files by using HDFS and directory permissions on the server.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Default

None

Notes

- Impersonation is not supported for HiveServer1 connections.

GUI Tab

[Security tab](#)

Remove Column Qualifiers

Attribute

RemoveColumnQualifiers (RCQ)

Purpose

Specifies whether the driver removes 3-part column qualifiers and replaces them with alias.column qualifiers. Microsoft Access executes a Select statement using this syntax when an index is specified on a linked table.

Valid Values

0 | 1

Behavior

If set to 1 (enabled) the driver removes 3-part column qualifiers and replaces them with alias.column qualifiers. Column qualifiers are Microsoft Access compatible in this setting.

If set to 0, the driver does not modify the request.

Notes

- When using the driver with Microsoft Access in creating a linked table, it is highly recommended that you do not specify an index. Specifying an index causes Access to execute a Select statement for each row, which results in very slow performance.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Service Principal Name

Attribute

ServicePrincipalName (SPN)

Purpose

This option is supported only for HiveServer2 connections. The service principal name to be used by driver for Kerberos authentication.

Valid Values

ServicePrincipalName

where:

ServicePrincipalName

is the three-part service principal name registered with the key distribution center (KDC).

Note: Your service principal name is the value of the `hive.server2.authentication.kerberos.principal` property in the `hive-site.xml` file.

You must specify the service principal name using the following format:

Service_Name/Fully_Qualified_Domain_Name@REALM.COM

where:

Service_Name

is the name of the service hosting the instance. For example, `yourservicename`.

Depending on the Hive distribution you use, the name of the service is defined either automatically by the server or manually by the user who created the service. For instance, CDH distributions automatically generate a service name of `hive`, while Apache Hadoop distributions require that the service name be manually defined by the user. Refer to your distribution's documentation for additional information.

Fully_Qualified_Domain_Name

is the fully qualified domain name of the host machine. For example, `yourserver.example.com`.

REALM.COM

is the domain name of the host machine. This part of the value must be specified in upper-case characters. For example, `EXAMPLE.COM`.

Example

The following is an example of a valid service principal name:

```
your servicename/yourserver.example.com@EXAMPLE.COM
```

Notes

- If unspecified, the value of the Network Address option is used as the service principal name.
- If Authentication Method is set to 0 or -1, the value of the Service Principal Name option is ignored.

Default

None

GUI Tab

[Security tab](#)

SSLibName

Attribute

SSLibName (SLN)

Purpose

The absolute path for the OpenSSL library file containing the TLS/SSL library to be used by the data source or connection when TLS/SSL is enabled. The SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different SSL library if you encounter issues with the default version or want to use a library that you provide. Common issues that require designating a different library include security vulnerabilities with specific libraries or compatibility issues with your server or application.

Valid Values

absolute_path\openssl_filename

where:

absolute_path

is the absolute path to where the OpenSSL file is located

openssl_filename

is the name of the OpenSSL library file containing the TLS/SSL Library to be used by your data source or connection.

Example

```
C:\Program Files\Progress\DataDirect\ODBC_71\
Drivers\OpenSSL\1.0.0r\ddssl27.dll (64-bit Windows)
```

Notes

- The OpenSSL library files provided by Progress combine the cryptographic and TLS/SSL libraries into a single file; therefore, when your drivers are using a Progress library file, the values specified for the `CryptoLibName` and `SSLibName` options should be the same. For non-Progress library files, the libraries may use separate files, which would require unique values to be specified.
- This option can be used to designate OpenSSL libraries not installed by the product; however, the drivers are only certified against libraries provided by Progress.

Default

No default value

GUI Tab

The value for this option is specified as an option-value pair in the Extended Options field on the Advanced tab. For example:

```
SSLibName=C:\Program Files\Progress\DataDirect\
Connect64_for_ODBC_71\Drivers\OpenSSL\1.0.0r\ddssl27.dll;
```

See [Advanced tab](#) for details.

See also

[CryptoLibName](#) on page 1058

String Describe Type

Attribute

StringDescribeType (SDT)

Purpose

Specifies whether all string columns are described as `SQL_VARCHAR`. This connection option affects `SQL_Columns`, `SQLDescribeCol`, `SQLColAttributes`, etc. It does not affect `SQLGetTypeInfo`.

Valid Values

-10 | -9 | -1 | 12

Behavior

If set to -10 -(SQL_WLONGVARCHAR), all strings are described as SQL_WLONGVARCHAR

If set to -9 -(SQL_WVARCHAR), all string columns are described as SQL_WVARCHAR.

If set to -1 -(SQL_LONGVARCHAR), all string columns are described as SQL_LONGVARCHAR.

If set to 12 -(SQL_VARCHAR), all string columns are described as SQL_VARCHAR.

Default

12 - SQL_VARCHAR

GUI Tab

[Advanced tab](#)

TCP Keep Alive

Attribute

KeepAlive (KA)

Purpose

Specifies whether the driver enables TCPKeepAlive. TCPKeepAlive maintains idle TCP connections by periodically passing packets between the client and server. If either the client or server does not respond to a packet, the connection is considered inactive and is terminated. In addition, TCPKeepAlive prevents valid idle connections from being disconnected by firewalls and proxies by maintaining network activity.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver does not enable TCPKeepAlive.

If set to 1 (Enabled), the driver enables TCPKeepAlive.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Transaction Mode

Attribute

TransactionMode (TM)

Purpose

Specifies how the driver handles manual transactions.

Valid Values

0 | 1

Behavior

If set to 1 (Ignore), the data source does not support transactions and the driver always operates in auto-commit mode. Calls to set the driver to manual commit mode and to commit transactions are ignored. Calls to rollback a transaction cause the driver to return an error indicating that no transaction is started. Metadata indicates that the driver supports transactions and the ReadUncommitted transaction isolation level.

If set to 0 (No Transactions), the data source and the driver do not support transactions. Metadata indicates that the driver does not support transactions.

Default

0 (No Transactions)

GUI Tab

[Advanced tab](#)

Truststore

Attribute

Truststore (TS)

Purpose

The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (Encryption Method=1) and server authentication is used. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. If you do not specify a directory, the current directory is used.

Valid Values

truststore_directory\filename

where:

truststore_directory

is the directory where the truststore file is located

filename

is the file name of the truststore file.

Notes

- The truststore and keystore files may be the same file.

Default

None

GUI Tab

[Security tab](#)

Trust Store Password

Attribute

TruststorePassword (TSP)

Purpose

Specifies the password that is used to access the truststore file when SSL is enabled (`Encryption Method=1`) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

Valid Values

truststore_password

where:

truststore_password

is a valid password for the truststore file.

Notes

- The truststore and keystore files may be the same file; therefore, they may have the same password.

Default

None

GUI Tab

[Security tab](#)

Use Current Schema for Catalog Functions

Attribute

UseCurrentSchema (UCS)

Purpose

Specifies whether results are restricted to the tables and views in the current schema if a catalog function call is made without specifying a schema or if the schema is specified as the wildcard character %. Restricting results to the tables and views in the current schema improves performance of catalog calls that do not specify a schema.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), results of catalog function calls are restricted to the tables and views in the current schema.

If set to 0 (Disabled), results of catalog function calls are not restricted.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

Use Native Catalog Functions

Attribute

UseNativeCatalogFunctions (UNCF)

Purpose

This option is supported only for HiveServer2 connections. Specifies whether the driver uses native catalog functions to retrieve information returned by the SQLTables, SQLColumns, and SQLStatistics catalog functions.

Valid Values

0 | 1

Behavior

If set to 0 (Disabled), the driver uses ODBC catalog functions to retrieve data source information.

If set to 1 (Enabled), the driver uses native catalog functions to retrieve information returned by the SQLTables, SQLColumns, and SQLStatistics catalog functions.

Default

0 (Disabled)

GUI Tab

[Advanced tab](#)

User Name

Attribute

LogonID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

N/A

GUI Tab

[Security tab](#)

Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=1). When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the driver to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Valid Values

0 | 1

Behavior

If set to 1 (Enabled), the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate option is specified, the driver also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to 0 (Disabled), the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Trust Store and Trust Store Password options.

Notes

- Truststore information is specified using the TrustStore and TrustStorePassword options.

Default

1 (Enabled)

GUI Tab

[Security tab](#)

Wire Protocol Version

Attribute

WireProtocolVersion (WPV)

Purpose

Indicates which protocol to use when connecting to the Apache Hive server.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (AutoDetect), the driver tries to connect using the HiveServer1 protocol and the HiveServer2 protocol simultaneously. Whichever server socket accepts the connection is the one that the driver continues to use for subsequent connections. For example, if the server socket accepts a connection request using the HiveServer2 protocol, subsequent connections from the driver use the same socket and protocol.

If set to 1 (HiveServer1), the driver only uses the HiveServer1 protocol to communicate with the server.

If set to 2 (HiveServer2), the driver only uses the HiveServer2 protocol to communicate with the server.

Notes

- If the protocol specified for `WireProtocolVersion` is not accepted by the Hive server to which you are connecting, the connection fails with a login timeout error, if `LoginTimeout` is enabled. For example, a login timeout error would be returned if you specify `WireProtocolVersion=2` and the driver attempts to connect to an Apache Hive server that only accepts connections using the HiveServer1 protocol.

Default

0 (AutoDetect)

GUI Tab

[General tab](#)

Performance Considerations

The following connection options can enhance driver performance.

Array Size (ArraySize): To improve throughput, consider increasing the value of Array Size. By increasing the value of Array Size, you increase the number of rows the driver will retrieve from the server for a fetch. In turn, increasing the number of rows that the driver can retrieve reduces the number, and expense, of network round trips. For example, if an application attempts to fetch 100,000 rows, it is more efficient for the driver to retrieve 2000 rows over the course of 50 round trips than to retrieve 500 rows over the course of 200 round trips. Note that improved throughput does come at the expense of increased demands on memory and slower response time. Furthermore, if the fetch size exceeds the available buffer memory of the server, an out of memory error is returned when attempting to execute a fetch. If you receive this error, decrease the value specified until fetches are successfully executed.

Batch Mechanism (BatchMechanism): If your application does not require individual update counts for each statement or parameter set in the batch, then `BatchMechanism` should be set to 2 (MultiRowInsert). Unlike the native batch mechanism, the multi-row insert mechanism only returns the total number of update counts for batch inserts. Therefore, setting `BatchMechanism` to `MultiRowInsert` offers substantial performance gains when performing batch inserts.

Encryption Method (EncryptionMethod): Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) that is required to encrypt and decrypt data.

Data Types

The following table shows how the Apache Hive data types are mapped to the standard ODBC data types.

Table 87: Apache Hive Data Types

Apache Hive	ODBC
Bigint	SQL_BIGINT
Boolean	SQL_BIT
Char ⁸⁰	SQL_CHAR
Date	SQL_DATE
Decimal ^{82, 83}	SQL_DECIMAL
Double	SQL_DOUBLE
Float	SQL_REAL
Int	SQL_INTEGER
Smallint	SQL_SMALLINT
String ⁸⁴	SQL_VARCHAR ⁸⁵
Timestamp ⁸⁵	SQL_TYPE_TIMESTAMP
Tinyint	SQL_TINYINT
Varchar ⁸¹	SQL_VARCHAR

Advanced Features

The driver supports the following advanced features:

- Security

Security

The driver supports authentication and encryption. Security connection options are located on the [Security tab](#) of the driver Setup dialog box. See [Using Security](#) on page 89 for a general description of security and its implementation. The following security-related information is specific to the Driver for Apache Hive.

⁸⁰ Supported only with Apache Hive 0.13.0 and higher.

⁸¹ Supported only with Apache Hive 0.12.0 and higher

⁸² Supported only with Apache Hive 0.11.0 and higher.

⁸³ The default precision and scale for the DECIMAL data type changed in Hive 0.13. For versions prior to Hive 0.13, the precision and scale are fixed and limited to 38 digits each. For Hive 0.13 and higher, the precision and scale are flexible with a default of 10 digits for precision and 0 digits for scale. When upgrading to Hive 0.13, you may need to change the length of precision and scale to avoid the rounding of values that exceed the default length. Refer to the Apache Hive user documentation for more information: <https://cwiki.apache.org/confluence/display/Hive/Home>.

⁸⁵ Maximum of 2 GB

⁸⁴ The StringDescribeType connection option setting determines where this data type maps. For example, if set to `sql_varchar` (the default), this data type maps to SQL_VARCHAR

⁸⁶ Supported only with Apache Hive 0.8.0 and higher.

Apache Sentry

Apache Sentry is a modular security system that enables HiveServer2 administrators to control access to data and metadata stored on an Apache Hadoop cluster by defining user roles and permissions. The driver works transparently with Sentry and does not require further configuration. To use Sentry, Kerberos authentication must be enabled, and a Kerberos logon must be provided at connection.

Note: When establishing a connection, the driver attempts to set the user's default database to be used for the session. In environments using Sentry, the user must be granted access to this database; otherwise, the connection will fail.

For more information, refer to the Apache Sentry documentation at <https://sentry.incubator.apache.org/>.

Materialized Views

Apache Hive supports views but purely as logical objects with no associated storage. As such, there is no support for materialized views in Hive; therefore, the driver does not support materialized views.

Stored Procedures

Apache Hive has no concept of stored procedures. Therefore, they are not supported in the driver.

Unicode Support

The driver is fully Unicode enabled. On UNIX and Linux platforms, the driver supports both UTF-8 and UTF-16. On Windows platforms, the Hive driver supports UCS-2/UTF-16 only.

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Isolation and Lock Levels Supported

Apache Hive supports isolation level 0 (read uncommitted).

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the core SQL grammar.

Refer to the [Hive Language Manual](#) for information about using HiveQL.

Also, see [SQL Functionality for the Driver for Apache Hive](#).

ODBC Conformance Level

The driver supports ODBC API Conformance Level 1.

Note: SQLCancel and SQLTransact execute successfully but perform no functions.

Note: SQLStatistics always returns an empty result set.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Using Arrays of Parameters

By default, the driver supports multi-row inserts for parameterized arrays. For a multi-row insert, the driver attempts to execute a single insert for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the driver, the driver executes multiple statements. This behavior provides substantial performance gains for batch inserts.

The driver modifies the HQL statement to perform a multi-row insert. Therefore, the default multi-row insert behavior may not be desirable in all scenarios. You can disable this behavior by setting the Batch Mechanism connection option to 1 (SingleInsert). When `BatchMechanism=1`, Hive's native batch mechanism is used to execute batch operations, and an insert statement is executed for each row contained in a parameter array.

Refer to "Designing ODBC applications for performance optimization" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Limitations on Apache Hive Functionality

The following restrictions are based on using Apache Hive 0.10.0:

- No support for row-level updates or deletes
- No difference between "NULL" and null values
- For HiverServer1 connections, no support for user-level authentication
- For HiveServer1 connections, no support for canceling a running query
- For HiveServer1 connections, no support for multiple simultaneous connections per port

For a more complete listing of Apache Hive known issues and limitations for your version of Hive, refer to the Apache Hive user documentation:

<https://cwiki.apache.org/confluence/display/Hive/Home>

Note: Note that Apache Hive is not designed for OLTP workloads and does not offer real-time queries or row-level updates. Instead, Hive is designed for batch type jobs over large data sets with high latency. This means that queries such as "SELECT * FROM mytable" return quickly. However, other SELECT statements are much slower.

Note: Because Apache HiveServer1 servers do not currently handle multiple connections well, consider using a single Apache HiveServer1 for each connection.

The Driver for the Teradata Database

The DataDirect Connect XE *for* ODBC and DataDirect Connect64 XE *for* ODBC driver for the Teradata database support Teradata database servers when using the appropriate client software.

For the latest support information, visit the Progress DataDirect Supported Configurations page: <https://www.progress.com/supported-configurations/datadirect>.

The driver is supported in the Windows, UNIX, and Linux environments. See [Environment-Specific Information](#) on page 58 for detailed information about the environments supported by this driver.

Refer to the readme file shipped with your DataDirect Connect product for the file name of the driver.

Driver Requirements

The driver requires Teradata Tools and Utilities (TTU) 8.2 or higher, which includes CLIV2, TGSS, and ICU client software, on all platforms. It requires TTU 12.0 to support 12.0 functionality.

Note: TTU 12.0 is not available for the Itanium II platform. You can use TTU 8.2 on an Itanium II client to connect to a Teradata 12.0 database, but functionality is limited to that of TTU 8.2.

Configuring and Connecting to Data Sources

After you install the driver, you configure data sources to connect to the database. See [Quick Start Connect](#) on page 33 for an explanation of different types of data sources. The data source contains connection options that allow you to tune the driver for specific performance. If you want to use a data source but need to change some of its values, you can either modify the data source or override its values at connection time through a connection string.

If you choose to use a connection string, you must use specific connection string attributes. See [Using a Connection String](#) on page 1083 and [Connection Option Descriptions](#) on page 1089 for an alphabetical list of driver connection string attributes and their initial default values.

Data Source Configuration in the UNIX/Linux odbc.ini File

UNIX[®] On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. See [Environment Configuration](#) on page 41 for basic setup information and [Environment Variables](#) on page 112 for more detail about this procedure.

Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini). If you have a Motif GUI environment on Linux, you can configure and modify data sources through the DataDirect ODBC Data Source Administrator for Linux (the Linux ODBC Administrator) using a driver Setup dialog box. (See [Configuration Through the Administrator](#) on page 115 for a detailed explanation of the Administrator.)

If you do not have a GUI environment, you can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See [Configuration Through the System Information \(odbc.ini\) File](#) on page 117 for detailed information about the specific steps necessary to configure a data source.

[Connection Option Descriptions](#) on page 1089 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Using a Connection String

If you want to use a connection string for connecting to a database, or if your application requires it, you must specify either a DSN (data source name), a File DSN, or a DSN-less connection in the string. The difference is whether you use the DSN=, FILEDSN=, or the DRIVER= keyword in the connection string, as described in the ODBC specification. A DSN or FILEDSN connection string tells the driver where to find the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in the data source.

The DSN connection string has the form:

```
DSN=data_source_name[;attribute=value[;attribute=value]...]
```

The FILEDSN connection string has the form:

```
FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
```

The DSN-less connection string specifies a driver instead of a data source. All connection information must be entered in the connection string because the information is not stored in a data source.

The DSN-less connection string has the form:

```
DRIVER=[{ }driver_name[ ]][;attribute=value[;attribute=value]...]
```

[Connection Option Descriptions](#) on page 1089 lists the long and short names for each attribute, as well as the initial default value when the driver is first installed. You can specify either long or short names in the connection string.

An example of a DSN connection string with overriding attribute values for Teradata is:

```
DSN=Teradata Tables;AS=User2;EnableDataEncryption=Yes
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Teradata.dsn;AS=User2;EnableDataEncryption=Yes
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER={DataDirect 7.1 Teradata};DBCSN=123.456.78.90;UIS=YES
```

Data Source Configuration through a GUI (Teradata)



On Windows, data sources are stored in the Windows Registry. You can configure and modify data sources through the ODBC Administrator using a driver Setup dialog box, as described in this section.


UNIX[®] On UNIX and Linux, data sources are stored in the `odbc.ini` file. On Linux, you can configure and modify data sources through the Linux ODBC Administrator using a driver Setup dialog box, as described in this section.

Note: This book shows dialog box images that are specific to Windows. If you are using the drivers in the Linux environment, the dialog box that you see may differ slightly from the Windows version. Windows-only and UNIX-only connection options are specifically noted by icons in the Setup dialog box descriptions.

When the driver is first installed, the values of its connection options are set by default. These values appear on the driver Setup dialog box tabs when you create a new data source. You can change these default values by modifying the data source. In the following procedure, the description of each tab is followed by a table that lists the connection options for that tab and their initial default values. This table links you to a complete description of the options and their connection string attribute equivalents. The connection string attributes are used to override the default values of the data source if you want to change these values at connection time.

To configure a Teradata data source:


1. Start the ODBC Administrator:

-  On Windows, start the ODBC Administrator by selecting its icon from the Progress DataDirect program group.
- **UNIX**[®] On Linux, change to the *install_dir/tools* directory and, at a command prompt, enter:

```
odbcadmin
```

where *install_dir* is the path to the product installation directory.

2. Select a tab:

- **User DSN:** If you are configuring an existing user data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
-  **System DSN:** If you are configuring an existing system data source, select the data source name and click **Configure** to display the driver Setup dialog box.
If you are configuring a new system data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
- **File DSN:** If you are configuring an existing file data source, select the data source file and click **Configure** to display the driver Setup dialog box.
If you are configuring a new file data source, click **Add** to display a list of installed drivers; then, select a driver. Click **Advanced** if you want to specify attributes; otherwise, click **Next** to proceed. Specify a name for the data source and click **Next**. Verify the data source information; then, click **Finish** to display the driver Setup dialog box.

The General tab of the Setup dialog box appears by default.

Figure 117: General tab

The screenshot shows the 'Teradata Driver Setup' dialog box with the 'General' tab selected. The 'Data Source' section has 'Name' set to 'Teradata'. The 'Teradata Server Info' section has 'DBCName or Alias' and 'DBCName List' fields. The 'Integrated Security' checkbox is unchecked. Under 'Authentication', 'Security Mechanism' is a dropdown menu, and 'Security Parameter', 'UserID', 'Default Database', and 'Account String' are text input fields. The 'Session Character Set' is set to 'ASCII'. At the bottom, there are buttons for 'Test Connect', 'OK', 'Cancel', and 'Apply'.

Note: The General tab displays only fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

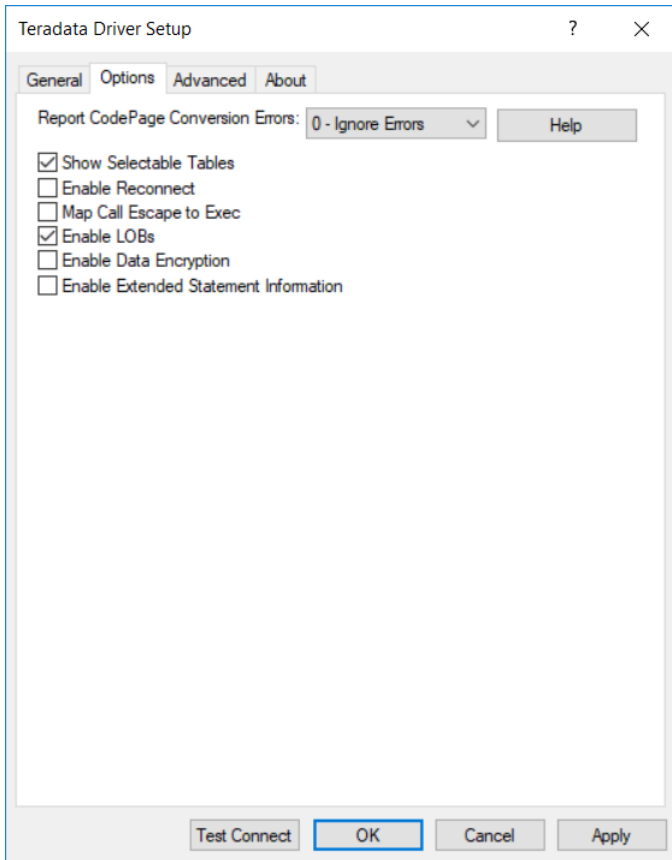
- On this tab, provide values for the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: General	Default
Name on page 1101	None
Description on page 1095	None
DBCName or Alias on page 1093	None
DBCName List on page 1092	None
Integrated Security on page 1099	Disabled
Security Mechanism on page 1105	None
Security Parameter on page 1106	None
UserID on page 1108	None

Connection Options: General	Default
Default Database on page 1094	None
Account String on page 1091	None
Session Character Set on page 1107	ASCII

4. Click the **Options** tab to specify additional configuration options.

Figure 118: Options tab



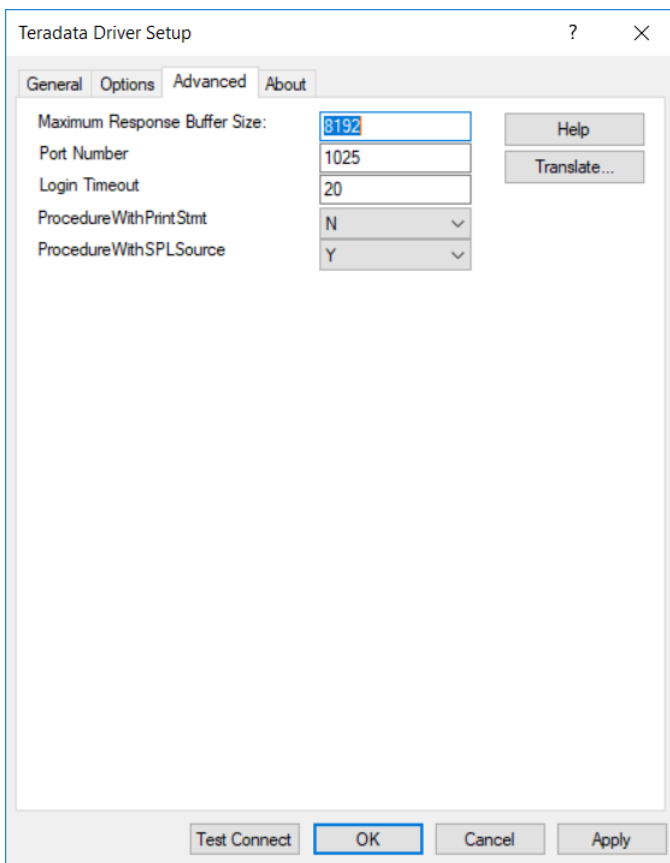
On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Options	Default
Report Codepage Conversion Errors on page 1104	0 - Ignore Errors
Show Selectable Tables on page 1107	Enabled
Enable Reconnect on page 1097	Disabled
Map Call Escape To Exec on page 1100	Disabled
Enable LOBs on page 1097	Enabled

Connection Options: Options	Default
Enable Data Encryption on page 1095	Disabled
Enable Extended Statement Information on page 1096	Disabled

5. Optionally, click the **Advanced** tab to specify additional data source settings.

Figure 119: Advanced tab



On this tab, provide values for any of the options in the following table; then, click **Apply**. The table provides links to descriptions of the connection options.

Connection Options: Advanced	Default
Maximum Response Buffer Size on page 1100	8192
Port Number on page 1102	1025
Login Timeout on page 1099	20
ProcedureWithPrintStmt on page 1102	N (No)
ProcedureWithSPLSource on page 1103	Y (Yes)
IANAAppCodePage on page 1098 UNIX ONLY	4 (ISO 8559-1 Latin-1)



Translate: Click **Translate** to display the Select Translator dialog box, which lists the translators specified in the ODBC Translators section of the Registry. Progress DataDirect provides a translator named OEM to ANSI that translates your data from the IBM PC character set to the ANSI character set.

Select a translator; then, click **OK** to close this dialog box.

6. At any point during the configuration process, you can click **Test Connect** to attempt to connect to the data source using the connection options specified in the driver Setup dialog box. A logon dialog box appears (see [Using a Logon Dialog Box \(Teradata\)](#) on page 1088 for details). The information you enter in the logon dialog box during a test connect is not saved.
 - If the driver can connect, it releases the connection and displays a `Connection Established` message. Click **OK**.
 - If the driver cannot connect because of an incorrect environment or connection value, it displays an appropriate error message.

Verify that all required client software is properly installed. If it is not, you will see the message:

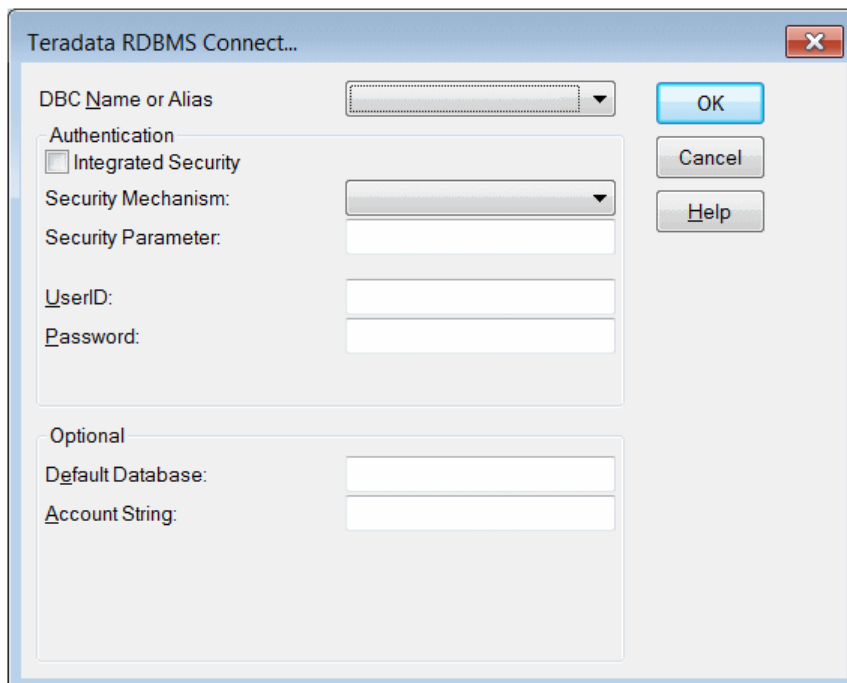
Specified driver could not be loaded due to system error [xxx].

Click **OK**.
7. Click **OK** or **Cancel**. If you click **OK**, the values you have specified become the defaults when you connect to the data source. You can change these defaults by using this procedure to reconfigure your data source. You can override these defaults by connecting to the data source using a connection string with alternate values.

Using a Logon Dialog Box (Teradata)

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

Figure 120: Teradata RDBMS Connect dialog box



In this dialog box, provide the following information:

1. Select an alias name or IP address of a Teradata server from the DBC Name or Alias drop-down list. The choices for this list are determined by the entries in DBC Name or Alias and DBCName List on the General tab of the driver Setup dialog box.
2. Select the Integrated Security check box to enable the user to connect to the database through Single Sign On (SSO) using one of the authentication mechanisms that support SSO. In this case, User Name, Password, and Domain are not required and are not available fields.
3. If you do not use Integrated Security, select a value from the Security Mechanism drop-down list to specify the authentication mechanism used for connections to the data source.
4. Type a string of characters in the Security Parameter field that is to be regarded as a parameter to the authentication mechanism. The string is ignored by the ODBC driver and is passed on to the TeraSSO function that is called to set the authentication mechanism.

The characters `[] {} () , ; ? * = ! @` must be enclosed in curly braces.

5. Other options that are displayed on the Logon Dialog box depend on the authentication mechanism selected. See the descriptions of these options under [Security Mechanism](#) on page 1105.
6. Type the domain name for Third Party Sign On along with the username and password. If a domain name is not provided, then the local domain is assumed.
7. Type a default Teradata database (optional).
8. Type an account string to be used during the creation of a user account in the Teradata Database instead of providing account information during configuration of ODBC (optional).
9. Click **OK** to complete the logon and to update these values in the Registry.

Connection Option Descriptions

The following connection option descriptions are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name, along with its short name, is listed immediately underneath the GUI name.

In most cases, the GUI name and the attribute name are the same; however, some exceptions exist. If you need to look up an option by its connection string attribute name, please refer to the alphabetical table of connection string attribute names.

Also, a few connection string attributes, for example, Password, do not have equivalent options that appear on the GUI. They are in the list of descriptions alphabetically by their attribute names.

The following table lists the connection string attributes supported by the driver.

Table 88: Teradata Attribute Names

Attribute (Short Name)	Default
AccountString (AS)	None
AuthenticationDomain (AD)	None
AuthenticationPassword (AP)	None
AuthenticationUserId (AUI)	None
CharacterSet (CS)	ASCII

Attribute (Short Name)	Default
Database (DB)	None
DataSourceName (DSN)	None
DBCName (DBCN)	None
DBCName List	None
Description (n/a)	None
EnableDataEncryption (EDE)	No (Disabled)
EnableExtendedStmtInfo (EESI)	No (Disabled)
EnableLOBs (EL)	Yes (Enabled)
EnableReconnect (ER)	No (Disabled)
IANAAppCodePage (IACP) UNIX ONLY	4 (ISO 8559-1 Latin-1)
IntegratedSecurity (IS)	No (Disabled)
LoginTimeout (LTO)	20
MapCallEscapeToExec (MCETE)	No (Disabled)
MaxRespSize (MRS)	8192
Password (PWD)	None
PortNumber (PORT)	1025
PrintOption (PO)	N (No)
ProcedureWithSPLSource (PWSS)	Y (Yes)
ReportCodepageConversionErrors (RCCE)	0 (Ignore Errors)
SecurityMechanism (SECM)	None
SecurityParameter (SP)	None
ShowSelectableTables (SST)	Yes (Enabled)
TDPProfile (TDP)	None
TDRole (TDR)	None
TDUserName (TDUN)	None
UserID (UID)	None

Account String

Attribute

AccountString (AS)

Purpose

An account string. For a complete description of account strings, refer to the *Teradata Database Administration Guide*.

Valid Values

string

where:

string

is an account string.

Default

None

GUI Tab

[General tab](#)

Authentication Password

Attribute

AuthenticationPassword (AP)

Purpose

The password for the Kerberos, LDAP, NTLM, and TD authentication mechanisms. The Authentication Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values

pwd

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Authentication UserId

Attribute

AuthenticationUserId (AUI)

Purpose

The user ID for the Kerberos, LDAP, NTLM, and TD authentication mechanisms.

Valid Values

userid

where:

userid

is a valid user ID.

Default

None

GUI Tab

[General tab](#)

DBCName List

Attribute

DBCName List

Attribute

n/a

Purpose

A list of IP addresses or aliases to appear in the drop-down list of the Logon dialog box (see [Using a Logon Dialog Box \(Teradata\)](#) on page 1088 for a description). The DBCName List option is not used as a runtime connection attribute.

Valid Values

ip_address | *alias* [, *ip_address* | *alias*][...]

where:

ip_address

is an IP address to appear in the drop-down list of the Logon dialog box.

alias

is an alias to appear in the drop-down list of the Logon dialog box.

Separate multiple IP addresses or aliases with commas. The same restrictions apply as described for the DBCName or Alias option.

Default

None

GUI Tab

[General tab](#)

DBCName or Alias

Attribute

DBCName (DBCN)

Purpose

The IP address or alias of the Teradata server.

Valid Values

IP_address | *alias*

where:

IP_address

is the IP address of the Teradata server.

alias

is the alias of the Teradata server.

Behavior

If set to *IP_address*, the time the driver waits for connections to be established is faster. The disadvantage is that if the server designated by that IP address is unavailable, the connection fails and the driver does not attempt to fail over to another IP address.

If set to *alias*, the time the driver waits for connections to be established is slower because the driver must search a local hosts file to resolve the alias to an IP address. The advantage is that the driver fails over the connection to an alternate IP address if the first address fails.

To use aliases, a local hosts file that maps aliases to IP addresses is required. Aliases cannot be more than eight characters. In the hosts file, you must specify the aliases and map each of them to an IP address in the order that you want the driver to attempt the connections. For example:

```
167.56.78.1 (NCR5100COP1)
167.56.78.2 (NCR5100COP2)
167.56.78.3 (NCR5100COP3)
```

where `NCR5100` is an alias and `COP n` (where $n = 1, 2, 3, \dots, 128$) is a suffix that sets the order of failover connection attempts. The eight-character limit on the alias does not include the suffix. You can enter a maximum of 128 COP (communications processor) entries per host.

Notes

- Although you must add a COP suffix to the alias in the hosts file, do *not* specify the suffix when entering the alias in the DBCName or Alias field of the Setup dialog box. Only specify the alias.

Default

None

GUI Tab

[General tab](#)

Default Database

Attribute

Database (DB)

Purpose

Specifies the name of the database to which you want to connect.

Valid Values

database_directory

where:

database_directory

is the full path name of the directory in which the data files are stored. If no directory is specified, the current working directory is used.

Default

None

GUI Tab

[General tab](#)

Default Role

Attribute

TDRole (TDR)

Purpose

Specifies the Teradata role for the LDAP authentication mechanism.

Valid Values

string

where:

string

is a valid role.

Default

None

GUI Tab

[General tab](#)

Description

Attribute

Description (n/a)

Purpose

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the ODBC.INI section of the Registry and in the odbc.ini file.

Valid Values

string

where:

string

is a description of a data source.

Default

None

GUI Tab

[General tab](#)

Enable Data Encryption

Attribute

EnableDataEncryption (EDE)

Purpose

Determines whether the driver uses data encryption.

Valid Values

1 | 0

Behavior

If set to 1 (Enabled), the driver encrypts data and communicates with the Teradata gateway using encryption.

If set to 0 (Disabled), the driver does not encrypt data except for logon information.

Notes

- Before you use this value, verify that the server is encryption capable. Data encryption may adversely affect performance because of the additional overhead (mainly CPU usage) required to encrypt and decrypt data.

Default

0 (Disabled)

GUI Tab

[Options tab](#)

Enable Extended Statement Information

Attribute

EnableExtendedStmtInfo (EESI)

Purpose

Determines whether the driver supports extended statement information.

Valid Values

1 | 0

Behavior

If set to 1 (Enabled), the driver queries the server to see if it supports the Statement Information parcel. If the server supports the Statement Information parcel, the driver requests the Statement Information parcel and enables auto-generated key retrieval and SQLDescribeParam support. Use this value if you want to enable the Return Generated Keys option.

If set to 0 (Disabled), the driver does not attempt to expose auto-generated key retrieval or SQLDescribeParam.

Default

0 (Disabled)

GUI Tab

[Options tab](#)

Enable LOBs

Attribute

EnableLOBs (EL)

Purpose

Determines whether the driver enforces native LOB data type mapping.

Valid Values

1 | 0

Behavior

If set to 1 (Enabled), the driver enforces native LOB data type mapping as described:

- ODBC data type SQL_LONGVARBINARY is mapped to the Teradata BLOB feature.
- ODBC data type SQL_LONGVARCHAR is mapped to the Teradata CLOB feature.

If set to 0 (Disabled), the driver provides backward compatibility for applications without LOB support that are using a version of Teradata Database prior to V2R5.1. The mappings are:

- ODBC data type SQL_LONGVARBINARY is mapped to the Teradata VARBYTE(32000) feature.
- ODBC data type SQL_LONGVARCHAR is mapped to the Teradata LONG VARCHAR feature.

This value can improve performance if your application does not send data to, or retrieve it from, LOB columns. You may receive an error if you disable this option and try to retrieve data from a LOB column.

Default

1 (Enabled)

GUI Tab

[Options tab](#)

Enable Reconnect

Attribute

EnableReconnect (ER)

Purpose

Determines whether the driver will reconnect after a system crash or reset is detected.

Valid Values

1 | 0

Behavior

If set to 1 (Enabled), the driver attempts to reconnect to the saved sessions; however, sessions cannot be reconnected until the Teradata system is available. After a session has been reconnected, applications can expect to receive error messages describing why the ODBC function failed, as well as a status report describing the post-recovery state.

If set to 0 (Disabled), the driver does not attempt to reconnect to the saved sessions.

Default

0 (Disabled)

GUI Tab

[Options tab](#)

IANAAppCodePage

Attribute

IANAAppCodePage (IACP)

Purpose

An Internet Assigned Numbers Authority (IANA) value. You must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. The value you specify must match the database character encoding and the system locale.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

Valid Values

IANA_code_page

where:

IANA_code_page

is one of the valid values listed in "IANAAppCodePage values" in the *Progress DataDirect for ODBC Drivers Reference*. The value must match the database character encoding and the system locale.

Notes

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Default

4 (ISO 8559-1 Latin-1)

GUI Tab

[Advanced tab](#)

Integrated Security

Attribute

IntegratedSecurity (IS)

Purpose

Determines whether the driver allows the user to connect to the database using Single Sign On (SSO) through an authentication mechanism that supports SSO.

Valid Values

Yes | No

Behavior

If set to Yes (Enabled), SSO is allowed. The driver uses the operating system user ID and password.

If set to No (Disabled), you must specify a value for the UserID option.

Default

No (Disabled)

GUI Tab

[General tab](#)

Login Timeout

Attribute

LoginTimeout (LTO)

Purpose

The number of seconds to wait when establishing a virtual circuit with Teradata for login.

Valid Values

x

where:

x

is a positive integer.

Behavior

If set to x, the driver waits the specified number of seconds.

Default

20

GUI Tab

[Advanced tab](#)

Map Call Escape To Exec

Attribute

MapCallEscapeToExec (MCETE)

Purpose

Determines whether the driver converts the `{CALL <name>(. . .)}` statement to `EXEC name(. . .)`.

Valid Values

Yes | No

Behavior

If set to Yes (Enabled), the driver considers the `{CALL <name>(. . .)}` statement as the SQL for MACRO execution and converts it to `EXEC name(. . .)`.

If set to No (Disabled), the driver does not convert `{CALL name(. . .)}` statements to `EXEC name(. . .)`, and considers them as CALL statements for Stored Procedure Execution.

Default

No (Disabled)

GUI Tab

[Options tab](#)

Maximum Response Buffer Size

Attribute

MaxRespSize (MRS)

Purpose

The size of the Teradata response buffer used for SQL requests. This value may be adjusted dynamically if Teradata cannot send a result within the defined size.

Valid Values

A positive integer from 1 to 65477

Behavior

If using a slow TCP/IP interface, such as PPP or SLIP, enter a smaller value. If you expect to retrieve large result sets in a LAN environment, set a larger value.

Default

8192

GUI Tab[Advanced tab](#)**Name****Attribute**

DataSourceName (DSN)

Purpose

Specifies the name of a data source in your Windows Registry or odbc.ini file.

Valid Values*string*

where:

string

is the name of a data source.

Default

None

GUI Tab[General tab](#)**Password****Attribute**

Password (PWD)

Purpose

The password that the application uses to connect to your database. The Password option cannot be specified through the driver Setup dialog box and should not be stored in a data source. It is specified through the Logon dialog box or a connection string.

Valid Values*pwd*

where:

pwd

is a valid password.

Default

None

GUI Tab

n/a

Port Number

Attribute

PortNumber (PORT)

Purpose

The port number of the server listener.

Valid Values

port_name

where:

port_name

is the port number of the server listener. Check with your database administrator for the correct number.

Default

1025

GUI Tab

[Advanced tab](#)

ProcedureWithPrintStmt

Attribute

PrintOption (PO)

Purpose

Determines whether the driver activates the print option when creating stored procedures.

Valid Values

P | N

Behavior

If set to P (Print), the driver activates the print option.

If set to N (No), the driver does not activate the print option.

Default

N (No)

GUI Tab

[Advanced tab](#)

ProcedureWithSPLSource**Attribute**

ProcedureWithSPLSource (PWSS)

Purpose

Determines whether the drive specifies SPL text when creating stored procedures.

Valid Values

Y | N

Behavior

If set to Y (Yes), the driver specifies SPL text.

If set to N (No), the driver does not specify SPL text.

Default

Y (Yes)

GUI Tab

[Advanced tab](#)

Profile**Attribute**

TDProfile (TDP)

Purpose

Specifies the Teradata profile for the LDAP authentication mechanism.

Valid Values

string

where:

string

is a valid profile.

Default

None

GUI Tab

[General tab](#)

Realm

Attribute

AuthenticationDomain (AD)

Purpose

Specifies the domain appropriate to the selected authentication mechanism

Valid Values

string

where:

string

is a valid domain.

Default

None

GUI Tab

None

Report Codepage Conversion Errors

Attribute

ReportCodepageConversionErrors (RCCE)

Purpose

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x`, where `x` is the parameter number. The standard rules for returning specific row and column errors for bulk operations apply.

Valid Values

0 | 1 | 2

Behavior

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

Default

0 (Ignore Errors)

GUI Tab

[Options tab](#)

Security Mechanism

Attribute

SecurityMechanism (SECM)

Purpose

The authentication method to be used by the driver for connections to the database.

Valid Values

TD1 | TD2 | ldap | KRB5 | KRB5C | NTLM | NTLMC

Behavior

If set to TD1, the driver uses Teradata 1.

If set to TD2, the driver uses Teradata 2.

If set to ldap, the driver uses LDAP.

If set to KRB5, the driver uses Kerberos on Windows clients working with Windows servers if the server is V2R6.0.

If set to KRB5C, the driver uses Kerberos Compatibility on Windows clients working with Windows servers if the server is pre-V2R6.0.

If set to NTLM, the driver uses NTLM on Windows clients working with Windows servers if the server is V2R6.0.

If set to NTLMC, the driver uses NTLM Compatibility on Windows clients working with Windows servers if the server is pre-V2R6.0.

Note: Kerberos and NTLM are enabled through the Teradata client. See your Teradata documentation for requirements.

The following options may appear, based on the selected method:

- *No mechanism selected*
- **User name:** A user name for the default Teradata database. If TeraSSO allows fully qualified user names, the user name may contain a domain or realm, for example, {judy@linedata}. Values containing a character like @ must be enclosed in braces.

- *KRB5 and KRB5C*
 - **Authentication UserID:** The Kerberos user ID.
 - **Realm:** The Kerberos domain. (The equivalent connection string attribute is AuthenticationDomain.)
- *LDAP*
 - **Authentication UserID:** The LDAP user ID.
 - **Realm:** The LDAP domain. (The equivalent connection string attribute is AuthenticationDomain.)
 - **TD User name:** The Teradata user name.
 - **Profile:** The Teradata Profile. (The equivalent connection string attribute is TDProfile.)
 - **Default Role:** The Teradata Role. (The equivalent connection string attribute is TDRole.)
- *NTLM and NTLMC*
 - **Authentication UserID:** The NTLM user ID.
 - **Realm:** The NTLM domain. (The equivalent connection string attribute is AuthenticationDomain.)
- *TD1 and TD2*
 - **Authentication UserID:** The TD1 or TD2 user ID.

Other parameters for the authentication mechanism can be entered in the Security Parameter field.

Default

None

GUI Tab

[General tab](#)

Security Parameter

Attribute

SecurityParameter (SP)

Purpose

A string that is passed as a parameter to the authentication method. The string is ignored by the ODBC driver and is passed to the TeraSSO function that is called to set the authentication method.

Valid Values

string

where:

string

is a string of characters. The characters [] { } () , ; ? * = ! @ must be enclosed in curly braces.

Default

None

GUI Tab

[General tab](#)

Session Character Set**Attribute**

CharacterSet (CS)

Purpose

A character set used to override the Teradata character set.

Valid Values

ASCII | UTF16 (valid only for V2R6.x servers) | LATIN1252_0A | LATIN9_0A | LATIN1_0A | Shift-JIS | EUC | BIG5 | GB | NetworkKorean

The specified character set must be installed on the database.

Default

ASCII

GUI Tab

[General tab](#)

Show Selectable Tables**Attribute**

ShowSelectableTables (SST)

Purpose

Determines whether the driver supports X views.

Valid Values

Yes | No

Behavior

If set to Yes (Enabled), `SQLTables()` and `SQLProcedures()` use `dbc.tablesX` and `dbc.databasesX` instead of `dbc.tables` and `dbc.databases`. Also, `SQLColumns()` and `SQLProcedureColumns()` use `dbc.columnsX` instead of `dbc.columns`. `SqlStatistics()` uses `dbc.tablesizeX` instead of `dbc.tablesize`. The X tables only contain information that the user has permission to access. These tables are optional for Teradata, so verify that they exist.

If set to No (Disabled), `SQLTables()` and `SQLProcedures()` use `dbc.tables` and `dbc.databases`. Also, `SQLColumns()` and `SQLProcedureColumns()` use `dbc.columns`. `SqlStatistics()` uses `dbc.tablesize`.

Default

Yes (Enabled)

GUI Tab

[Options tab](#)

TDUserName

Attribute

TDUserName (TDUN)

Purpose

Specifies the Teradata user name for the LDAP authentication mechanism.

Valid Values

user_name

where:

user_name

is a valid user name.

Default

None

GUI Tab

[General tab](#)

UserID

Attribute

UserID (UID)

Purpose

The default user ID that is used to connect to your database. Your ODBC application may override this value or you may override it in the logon dialog box or connection string.

Valid Values

userid

where:

userid

is a valid user ID with permissions to access the database.

Behavior

The user name is interpreted in the context of the authentication mechanism. If, for example, the authentication mechanism is NTLM, the user name is assumed to be a Windows user name.

If TeraSSO allows fully qualified user names, the user name may contain a domain or realm, for example, {judy@linedata}. Values containing a character such as @ must be enclosed in curly braces.

SSO is indicated by the absence of a UserID.

Default

None

GUI Tab

[General tab](#)

Data Types

The following table shows how the Teradata data types map to the standard ODBC data types.

Table 89: Teradata Data Types

Teradata	ODBC
Blob ⁸⁷	SQL_LONGVARBINARY
Bigint ⁸⁸	SQL_BIGINT
Byte	SQL_BIT
Byteint	SQL_TINYINT
Char	SQL_CHAR
Clob ⁸⁹	SQL_LONGVARCHAR
Date	SQL_TYPE_DATE
Decimal ⁹⁰	SQL_DECIMAL
Double	SQL_DOUBLE
Float	SQL_FLOAT
Integer	SQL_INTEGER
Interval day	SQL_INTERVAL_DAY
Interval day to hour	SQL_INTERVAL_DAY_TO_HOUR

⁸⁷ If no LOB support, VARBYTE(32000).

⁸⁸ Supported only on Teradata 6.2 and higher.

⁸⁹ If no LOB support, LONGVARCHAR.

⁹⁰ Precision of 18 unless on a Teradata 6.2 or higher server that supports large decimal types.

Teradata	ODBC
Interval day to minute	SQL_INTERVAL_DAY_TO_MINUTE
Interval day to second	SQL_INTERVAL_DAY_TO_SECOND
Interval hour	SQL_INTERVAL_HOUR
Interval hour to minute	SQL_INTERVAL_HOUR_TO_MINUTE
Interval hour to second	SQL_INTERVAL_HOUR_TO_SECOND
Interval minute ⁹¹	SQL_INTERVAL_MINUTE
Interval minute to second	SQL_INTERVAL_MINUTE_TO_SECOND
Interval month ⁹¹	SQL_INTERVAL_MONTH
Interval second	SQL_INTERVAL_SECOND
Interval year	SQL_INTERVAL_YEAR
Interval year to month	SQL_INTERVAL_YEAR_TO_MONTH
Number	SQL_DOUBLE
Number (p)	SQL_BIGINT SQL_DECIMAL ⁹²
Number (p, s)	SQL_DECIMAL
Numeric	SQL_NUMERIC
Real	SQL_REAL
Smallint	SQL_SMALLINT
Time	SQL_TYPE_TIME
Timestamp	SQL_TYPE_TIMESTAMP
Varchar	SQL_VARCHAR

See [Retrieving Data Type Information](#) on page 72 for information about retrieving data types.

Unicode Support

The driver supports Unicode data types. The following table shows how the Teradata data types map to the Unicode data types, but only when CharacterSet is set to UTF-16.

⁹¹ Supported only on Teradata 6.2 and higher when EnableExtendedStmtInfo is enabled.

⁹² When precision is less than or equal to 19, Number maps to SQL_BIGINT. When precision is greater than 19, it maps to Decimal.

Table 90: Teradata Unicode Data Types

Teradata	Unicode
char () charset Unicode	SQL_WCHAR
clob charset Unicode	SQL_WLONGVARCHAR
varchar () charset Unicode	SQL_WVARCHAR

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See [UTF-16 Applications on UNIX and Linux](#) on page 137 for related details.

Also, refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for a more detailed explanation of Unicode.

Persisting a Result Set as an XML Data File

The driver allows you to persist a result as an XML data file with embedded schema. See [Persisting a Result Set as an XML Data File](#) on page 73 for details about implementation.

Isolation and Lock Levels Supported

Teradata supports isolation levels 0 (read uncommitted) and 3 (serializable).

Refer to "Locking and isolation levels" in the *Progress DataDirect for ODBC Drivers Reference* for details.

SQL Support

The driver supports the minimum SQL grammar.

ODBC Conformance Level

The driver is Level 1 compliant, that is, it supports all ODBC Core and Level 1 functions.

Refer to "ODBC API and scalar functions" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported API functions.

Number of Connections and Statements Supported

The driver supports multiple connections and 16 statements per connection to the Teradata database system.

Supported SQL Statements and Extensions

This chapter includes information on supported SQL functionality for the Apache Hive, Impala Wire Protocol, Salesforce, and flat-file drivers.

For details, see the following topics:

- [SQL Functionality for the Driver for Apache Hive](#)
- [SQL Statements for Flat-File Drivers](#)
- [SQL Functionality for the Impala Wire Protocol Driver](#)
- [SQL Statements and Extensions for the Salesforce Driver](#)

SQL Functionality for the Driver for Apache Hive

The DataDirect Connect XE *for* ODBC and DataDirect Connect64 XE *for* ODBC for Apache Hive Wire Protocol driver support an extended set of SQL 92. in addition to the syntax for Apache HiveQL, which is a subset of SQL 92.

Refer to the [Hive Language Manual](#) for information about using HiveQL.

Data Definition Language (DDL)

The Driver for Apache Hive supports a broad set of DDL, including (but not limited to) the following:

- CREATE Database and DROP Database

- CREATE Table and DROP Table
- ALTER Table and Alter Partition statements
- CREATE View and Drop View
- CREATE Function and Drop Function

Refer to the [Hive Data Definition Language manual](#) for information about using HiveQL.

Selecting Data With the Driver

Select List

The following sections apply to the way the Select list can be used with the driver.

Column Name Qualification

A column can only be qualified with a single name, which must be a table alias. Furthermore, a table can be qualified with a database (ODBC schema) name in the FROM clause, and in some cases, must also be aliased. Aliasing may not be necessary if the database qualifier is not the current database.

The driver can work around these limitations using the Remove Column Qualifiers connection option.

- If set to 1, the driver removes three-part column qualifiers and replaces them with alias.column qualifiers.
- If set to 0, the driver does not do anything with the request.

Suppose you have the following ANSI SQL query:

```
SELECT schema.table1.col1,schema.table2.col2 FROM schema.table1,schema.table2
WHERE schema.table1.col3=schema.table2.col3
```

If the Remove Column Qualifiers connection option is enabled, the driver replaces the three-part column qualifiers:

```
SELECT table1.col1, table2.col2 FROM schema.table1 table1 JOIN schema.table2 table2
WHERE table1.col3 = table2.col3
```

From Clause

LEFT, RIGHT, and FULL OUTER JOINS are supported, as are LEFT SEMI JOINS and CROSS JOINS using the equal comparison operator, as shown in the following examples

```
SELECT a.* FROM a JOIN b ON (a.id = b.id AND a.department = b.department)
```

```
SELECT a.val, b.val, c.val FROM a JOIN b ON (a.key = b.key1) JOIN c ON
(c.key = b.key2)
```

```
SELECT a.val, b.val FROM a LEFT OUTER JOIN b ON (a.key=b.key)
WHERE a.ds='2009-07-07' AND b.ds='2009-07-07'
```

However, the following syntax fails because of the use of non-equal comparison operators.

```
SELECT a.* FROM a JOIN b ON (a.id <> b.id)
```

HiveQL does not support join syntax in the form of a comma-separated list of tables. The driver, however, overcomes this limitation by translating the SQL into HiveQL, as shown in the following examples.

ANSI SQL 92 Query

```
SELECT * FROM t1, t2 WHERE a = b
```

```
SELECT * FROM t1 y, t2 x WHERE a = b
```

```
SELECT * FROM t2, (SELECT * FROM t1) x
```

Driver for Apache Hive Wire HiveQL Translation

```
SELECT * FROM t1 t1 JOIN t2 t2 WHERE a = b
```

```
SELECT * FROM t1 y JOIN t2 x WHERE a = b
```

```
SELECT * FROM t2 JOIN (SELECT * FROM t1 t1) x
```

Group By Clause

The Group By clause is supported, with the following Entry SQL level restrictions:

- The COLLATE clause is not supported.
- SELECT DISTINCT is not supported.
- The grouping column reference cannot be an alias. Both of the following queries fail, because `fc` is an alias for the `intcol` column:

```
SELECT intcol AS fc, COUNT (*) FROM p_gtable GROUP BY fc
```

```
SELECT f(col) as fc, COUNT (*) FROM table_name GROUP BY fc
```

Having Clause

The Having Clause is supported, with the following Entry SQL level restriction: a GROUP BY clause is required.

Insert

Insert statements are supported using the following syntax:

```
INSERT INTO TABLE <table_name> VALUES (<expression> [,<expression>]...)
```

where:

`table_name`

is the name of the table into which you want to insert rows.

`expression`

is a literal, a parameterized array, or null.

Notes

- The following conditions apply for the successful execution of an insert:
 - Values for all columns must be specified in order.
 - Column lists cannot be used.
 - Casts and other functions cannot be used.
 - String values must be enclosed in single quotation marks (').

- By default, the driver supports multi-row inserts for parameterized arrays. For a multi-row insert, the driver attempts to execute a single insert for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the driver, the driver executes multiple statements. This behavior provides substantial performance gains for batch inserts.
- The driver modifies the HQL statement to perform a multi-row insert. Therefore, the default multi-row insert behavior may not be desirable in all scenarios. You can disable this behavior by setting the Batch Mechanism connection option to SingleInsert. When Batch Mechanism is set to SingleInsert, Hive's native batch mechanism is used to execute batch operations, and an insert statement is executed for each row contained in a parameter array.

Order By Clause

The Order By clause is supported, with the following Entry SQL level restrictions:

- An integer sort key is not allowed.
- The COLLATE clause is not supported.

For Update Clause

Not supported in this release. If present, the driver strips the For Update clause from the query.

Set Operators

Supported, with the following Entry SQL level restrictions:

- UNION is not supported.
- UNION ALL is supported only in a subquery.

The following example fails with Apache Hive because UNION ALL is used in a standard syntax:

```
SELECT * FROM t1 UNION ALL SELECT * FROM t2
```

To make the query work with Apache Hive, use a subquery:

```
SELECT * FROM (SELECT * FROM t1 UNION ALL SELECT * FROM t2) sq
```

In addition, INTERSECT or EXCEPT are not supported.

Subqueries

A query is an operation that retrieves data from one or more tables or views. In this reference, a top-level query is called a Select statement, and a query nested within a Select statement is called a subquery.

Subqueries are supported, with the following Entry SQL level restriction: subqueries can only exist in the FROM clause, that is, in a derived table. In the following example, the second Select statement is a subquery:

```
SELECT * FROM (SELECT * FROM t1 UNION ALL SELECT * FROM t2) sq
```

Although Apache Hive currently does not support IN or EXISTS subqueries, you can efficiently implement the semantics by rewriting queries to use LEFT SEMI JOIN.

SQL Expressions

An expression is a combination of one or more values, operators, and SQL functions that evaluate to a value. You can use expressions in the *Where* and *Having* clauses of *Select* statements.

Expressions enable you to use mathematical operations as well as character string manipulation operators to form complex queries.

Valid expression elements are:

- [Constants](#) on page 1117
- [Numeric Operators](#) on page 1117
- [Character Operator](#) on page 1118
- [Relational Operators](#) on page 1118
- [Logical Operators](#) on page 1118
- [Functions](#) on page 1119

Constants

Apache Hive servers prior to Apache Hive 0.8 do not support literal values expressed in scientific notation.

Numeric Operators

You can use a numeric operator in an expression to negate, add, subtract, multiply, and divide numeric values. The result of this operation is also a numeric value. The + and - operators are also supported in date/time fields to allow date arithmetic.

The following table lists the supported arithmetic operators.

Table 91: Numeric Operators

Entry SQL Level Operator	HiveQL Operator
N/A	% (Mod)
N/A	& (bitwise AND)
*	Supported
+	Supported
-	Supported
/	Supported
N/A	^ (XOR)

Character Operator

The concatenation operator (||) is not supported; however, the CONCAT function is supported by HiveQL.

```
SELECT CONCAT('Name is', '(ename FROM emp)')
```

Relational Operators

Relational operators compare one expression to another.

The following table lists the supported relational operators.

Table 92: Relational Operators Supported with Apache Hive

Entry SQL Level Operator	Support in HiveQL
<>	Supported
<	Supported
<=	Supported
=	Supported
<=>	Supported (Hive versions 0.9 and higher)
>	Supported
>=	Supported
IS [NOT] NULL	Supported
[NOT] BETWEEN x AND y	Supported (Hive versions 0.9 and higher)
[NOT] IN	Supported
EXISTS	Supported
[NOT] LIKE	Supported, except that no collate clause is allowed
RLIKE	Supported
REGEXP	Supported

Logical Operators

A logical operator combines the results of two component conditions to produce a single result or to invert the result of a single condition. The following table lists the supported logical operators.

Table 93: Logical Operators

Operator	Support in HiveQL
NOT !	Supported
AND &&	Supported
OR	Supported

Functions

The driver supports a number of functions that you can use in expressions, as listed in the following tables.

Table 94: Set Functions Supported

Set Function	Support in HiveQL
Count	Supported
AVG	Supported
MIN	Supported
MAX	Supported
SUM	Supported
DISTINCT	Supported
ALL	Supported

Table 95: Numeric Functions Supported

Numeric Function	Support in HiveQL
CHAR_LENGTH CHARACTER_LENGTH	Not supported. Use LENGTH(string) instead.
Position...In	Not supported
BIT_LENGTH(s)	Not supported
OCTET_LENGTH(str)	Not supported
EXTRACT...FROM	Not supported
TIMEZONE_HOUR	Not supported
TIMEZONE_MINUTE	Not supported

Table 96: String Functions Supported

String Function	Support in HiveQL
Substring	Supported
Convert ... using	Not supported
TRIM	Supported.
Leading	Not supported. Use LTRIM.
Trailing	Not supported. Use RTRIM.
Both	Not supported (default behavior of TRIM)

Table 97: Date/Time Functions Supported

Date/Time Function	Support in HiveQL
CURRENT_DATE()	Not supported
CURRENT_TIME()	Not supported
CURRENT_TIMESTAMP	Not supported. Use UNIX_TIMESTAMP().

Table 98: System Functions Supported

System Function	Support in HiveQL
CASE ... END	Supported.
COALESCE	Supported.
NULLIF	Not supported.
CAST	Supported.

Restrictions

This section describes some of the functional restrictions of Apache Hive.

Insert and Update Restrictions

Apache Hive does not support row level INSERT, UPDATE, and DELETE. Hive-specific syntax provides limited INSERT support:

- INSERT INTO is supported.
- INSERT OVERWRITE is supported.
- The name of the target table for an Insert must be prefixed by `table`, for example, `INSERT INTO table gtable ...`

Column lists are not supported. Values for all columns must be specified.

Stored Procedures

Apache Hive has no concept of stored procedures. Therefore, they are not supported by the driver.

Views

Apache Hive supports views but purely as logical objects with no associated storage. As such, there is no support for materialized views in Hive; therefore, the Apache Hive Wire Protocol driver does not support materialized views.

Apache Hive does not automatically update the view's schema after it is created; therefore, subsequent changes to underlying tables are not reflected in the view's schema. Any modifications that render the view incompatible will cause queries on the view to fail. Views are intended for Read Only access. LOAD, INSERT, and ALTER statements on a view will return an error.

Other Restrictions

The Apache Hive server has the following restrictions:

- Column values and parameters are always nullable
- No ROWID support
- No support for synonyms
- Primary and foreign keys are not supported.
- The length of a SQL string is limited to 2 GB.
- Support for indexes is incomplete.

SQL Statements for Flat-File Drivers

This chapter describes the SQL statements that you can use with the flat-file drivers (Btrieve, dBASE, and Text). Any exceptions to the supported SQL functionality described in this chapter are documented in the individual flat-file driver chapters in the *DataDirect Connect Series for ODBC User's Guide*.

The database drivers parse SQL statements and translate them into a form that the database can understand. The SQL statements described in this chapter let you:

- Read, insert, update, and delete rows from a database
- Create new tables
- Drop existing tables

These SQL statements allow your application to be portable across other databases.

Select Statement

The form of the Select statement supported by the flat-file drivers is:

```
SELECT [DISTINCT] { * | column_expression, ... }
FROM table_names [table_alias] ...
[ WHERE expr1rel_operatorexpr2 ]
[ GROUP BY {column_expression, ...} ]
[ HAVING expr1rel_operatorexpr2 ]
[ UNION [ALL] (SELECT...) ]
[ ORDER BY {sort_expression [DESC | ASC]}, ... ]
[ FOR UPDATE [OF {column_expression, ...}] ]
```

Select Clause

Follow Select with a list of column expressions you want to retrieve or an asterisk (*) to retrieve all fields.

```
SELECT [DISTINCT] { * | column_expression, [[AS] column_alias]. . . }
```

column_expression can be simply a field name (for example, LAST_NAME). More complex expressions may include mathematical operations or string manipulation (for example, SALARY * 1.05). See [SQL Expressions](#) on page 1125 for details.

column_alias can be used to give the column a descriptive name. For example, to assign the alias DEPARTMENT to the column DEP:

```
SELECT dep AS department FROM emp
```

Separate multiple column expressions with commas (for example, LAST_NAME, FIRST_NAME, HIRE_DATE).

Field names can be prefixed with the table name or alias. For example, EMP.LAST_NAME or E.LAST_NAME, where E is the alias for the table EMP.

The Distinct operator can precede the first column expression. This operator eliminates duplicate rows from the result of a query. For example:

```
SELECT DISTINCT dep FROM emp
```

Aggregate Functions

Aggregate functions can also be a part of a Select clause. Aggregate functions return a single value from a set of rows. An aggregate can be used with a field name (for example, AVG(SALARY)) or in combination with a more complex column expression (for example, AVG(SALARY * 1.07)). The column expression can be preceded by the Distinct operator. The Distinct operator eliminates duplicate values from an aggregate expression. For example:

```
COUNT (DISTINCT last_name)
```

In this example, only distinct last name values are counted.

The following table lists valid aggregate functions.

Table 99: Aggregate Functions

Aggregate	Returns
SUM	The total of the values in a numeric field expression. For example, SUM(SALARY) returns the sum of all salary field values.

AVG	The average of the values in a numeric field expression. For example, AVG(SALARY) returns the average of all salary field values.
COUNT	The number of values in any field expression. For example, COUNT(NAME) returns the number of name values. When using COUNT with a field name, COUNT returns the number of non-NULL field values. A special example is COUNT(*), which returns the number of rows in the set, including rows with NULL values.
MAX	The maximum value in any field expression. For example, MAX(SALARY) returns the maximum salary field value.
MIN	The minimum value in any field expression. For example, MIN(SALARY) returns the minimum salary field value.

From Clause

The From clause indicates the tables to be used in the Select statement. The format of the From clause is:

```
FROM table_names [table_alias]
```

table_names can be one or more simple table names in the current working directory or complete path names.

table_alias is a name used to refer to a table in the rest of the Select statement. Database field names may be prefixed by the table alias. Given the table specification:

```
FROM emp E
```

you may refer to the LAST_NAME field as E.LAST_NAME. Table aliases must be used if the Select statement joins a table to itself. For example:

```
SELECT * FROM emp E, emp F WHERE E.mgr_id = F.emp_id
```

The equal sign (=) includes only matching rows in the results.

If you are joining more than one table, you can use LEFT OUTER JOIN, which includes non-matching rows in the first table you name. For example:

```
SELECT * FROM T1 LEFT OUTER JOIN T2 on T1.key = T2.key
```

Where Clause

The Where clause specifies the conditions that rows must meet to be retrieved. The Where clause contains conditions in the form:

```
WHERE expr1rel_operatorexpr2
```

expr1 and *expr2* can be field names, constant values, or expressions.

rel_operator is the relational operator that links the two expressions. See [SQL Expressions](#) on page 1125 for details.

For example, the following Select statement retrieves the names of employees that make at least \$20,000.

```
SELECT last_name,first_name FROM emp WHERE salary >= 20000
```

Group By Clause

The Group By clause specifies the names of one or more fields by which the returned values should be grouped. This clause is used to return a set of aggregate values. It has the following form:

GROUP BY *column_expressions*

column_expressions must match the column expression used in the Select clause. A column expression can be one or more field names of the database table, separated by a comma (,) or one or more expressions, separated by a comma (,). See [SQL Expressions](#) on page 1125 for details.

The following example sums the salaries in each department:

```
SELECT dept_id, sum(salary) FROM emp GROUP BY dept_id
```

This statement returns one row for each distinct department ID. Each row contains the department ID and the sum of the salaries of the employees in the department.

Having Clause

The Having clause enables you to specify conditions for groups of rows (for example, display only the departments that have salaries totaling more than \$200,000). This clause is valid only if you have already defined a Group By clause. It has the following form:

```
HAVING expr1rel_operatorexpr2
```

expr1 and *expr2* can be field names, constant values, or expressions. These expressions do not have to match a column expression in the Select clause.

rel_operator is the relational operator that links the two expressions. See [SQL Expressions](#) on page 1125 for details.

The following example returns only the departments whose sums of salaries are greater than \$200,000:

```
SELECT dept_id, sum(salary) FROM emp
GROUP BY dept_id HAVING sum(salary) > 200000
```

Union Operator

The Union operator combines the results of two Select statements into a single result. The single result is all the returned rows from both Select statements. By default, duplicate rows are not returned. To return duplicate rows, use the All keyword (UNION ALL). The form is:

```
SELECT statement
UNION ALL
SELECT statement
```

When using the Union operator, the Select lists for each Select statement must have the same number of column expressions with the same data types, and must be specified in the same order. For example:

```
SELECT last_name, salary, hire_date FROM emp
UNION
SELECT name, pay, birth_date FROM person
```

This example has the same number of column expressions, and each column expression, in order, has the same data type.

The following example is *not* valid because the data types of the column expressions are different (salary from emp has a different data type than last_name from raises). This example does have the same number of column expressions in each Select statement but the expressions are not in the same order by data type.

```
SELECT last_name, salary FROM emp
UNION
SELECT salary, last_name FROM raises
```

Order By Clause

The Order By clause indicates how the rows are to be sorted. The form is:

```
ORDER BY {sort_expression [DESC | ASC]}, ...
```

sort_expression can be field names, expressions, or the positioned number of the column expression to use.

The default is to perform an ascending (ASC) sort.

For example, to sort by `last_name` and then by `first_name`, you could use either of the following Select statements:

```
SELECT emp_id, last_name, first_name FROM emp
```

```
ORDER BY last_name, first_name
```

or

```
SELECT emp_id, last_name, first_name FROM emp
```

```
ORDER BY 2,3
```

In the second example, `last_name` is the second column expression following Select, so Order By 2 sorts by `last_name`.

For Update Clause

The For Update clause locks the rows of the database table selected by the Select statement. The form is:

```
FOR UPDATE OF column_expressions
```

column_expressions is a list of field names in the database table that you intend to update, separated by a comma (,).

The following example returns all rows in the employee database that have a salary field value of more than \$20,000. When each record is fetched, it is locked. If the record is updated or deleted, the lock is held until you commit the change. Otherwise, the lock is released when you fetch the next record.

```
SELECT * FROM emp WHERE salary > 20000   FOR UPDATE OF last_name, first_name,
salary
```

SQL Expressions

Expressions are used in the Where clauses, Having clauses, and Order By clauses of SQL Select statements.

Expressions enable you to use mathematical operations as well as character string and date manipulation operators to form complex database queries.

The most common expression is a simple field name. You can combine a field name with other expression elements.

Valid expression elements are as follows:

- Field Names
- Constants
- Exponential notation
- Numeric operators

- Character operators
- Date operators
- Relational operators
- Logical operators
- Functions

Constants

Constants are values that do not change. For example, in the expression `PRICE * 1.05`, the value 1.05 is a constant.

You must enclose character constants in pairs of single (') or double (") quotation marks. To include a single quotation mark in a character constant enclosed by single quotation marks, use two single quotation marks together (for example, 'Don"t'). Similarly, if the constant is enclosed by double quotation marks, use two double quotation marks to include one.

You must enclose date and time constants in braces ({}), for example, {01/30/89} and {12:35:10}. The form for date constants is MM/DD/YY or MM/DD/YYYY. The form for time constants is HH:MM:SS.

The logical constants are .T. and 1 for True and .F. and 0 for False. For portability, use 1 and 0.

Exponential Notation

You can include exponential notation in expression elements. For example:

```
SELECT col1, 3.4E+7 FROM table1 WHERE calc < 3.4E-6 * col2
```

Numeric Operators

You can include the following operators in numeric expressions:

Operator	Meaning
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Exponentiation
^	Exponentiation

The following table shows examples of numeric expressions. For these examples, assume salary is 20000.

Example	Resulting value
<code>salary + 10000</code>	30000
<code>salary * 1.1</code>	22000
<code>2 ** 3</code>	8

You can precede numeric expressions with a unary plus (+) or minus (-). For example, `-(salary * 1.1)` is -22000.

Character Operators

Character expressions can include the following operators:

Operator	Meaning
+	Concatenation, keeping trailing blanks.
-	Concatenation, moving trailing blanks to the end.

The following table shows examples of character expressions. In the examples, last_name is 'JONES ' and first_name is 'ROBERT '.

Example	Resulting Value
first_name + last_name	'ROBERT JONES '
first_name - last_name	'ROBERTJONES '

Note: Some flat-file drivers return character data with trailing blanks as shown in the table; however, you cannot rely on the driver to return blanks. If you want an expression that works regardless of whether the drivers return trailing blanks, use the TRIM function before concatenating strings to make the expression portable. For example:

```
TRIM(first_name) + ' ' + TRIM(last_name)
```

Date Operators

You can include the following operators in date expressions:

Operator	Meaning
+	Add a number of days to a date to produce a new date.
-	The number of days between two dates, or subtract a number of days from a date to produce a new date.

The following table shows examples of date expressions. In these examples, hire_date is {01/30/1990}.

Example	Resulting Value
hire_date + 5	{02/04/1990}
hire_date - {01/01/1990}	29
hire_date - 10	{01/20/1990}

Relational Operators

Relational operators separating any two expressions can be any one of those listed in the following table.

Table 100: Relational Operators

Operator	Meaning
=	Equal.
<>	Not Equal.

Operator	Meaning
>	Greater Than.
>=	Greater Than or Equal.
<	Less Than.
<=	Less Than or Equal.
Like	Matching a pattern.
Not Like	Not matching a pattern.
Is NULL	Equal to NULL.
Is Not NULL	Not Equal to NULL.
Between	Range of values between a lower and upper bound.
In	A member of a set of specified values or a member of a subquery.
Exists	True if a subquery returned at least one record.
Any	Compares a value to each value returned by a subquery. Any must be prefaced by =, <>, >, >=, <, or <=. Any is equivalent to In.
All	Compares a value to each value returned by a subquery. All must be prefaced by =, <>, >, >=, <, or <=.

The following list shows some examples of relational operators:

```

salary <= 40000
dept = 'D101'
hire_date > {01/30/1989}
salary + commission >= 50000
last_name LIKE 'Jo%'
salary IS NULL
salary BETWEEN 10000 AND 20000
WHERE salary = ANY (SELECT salary FROM emp WHERE dept = 'D101')
WHERE salary > ALL (SELECT salary FROM emp WHERE dept = 'D101')

```

Logical Operators

Two or more conditions may be combined to form more complex criteria. When two or more conditions are present, they must be related by AND or OR. For example:

```
salary = 40000 AND exempt = 1
```

The logical NOT operator is used to reverse the meaning. For example:

```
NOT (salary = 40000 AND exempt = 1)
```


Operator Precedence

As expressions become more complex, the order in which the expressions are evaluated becomes important. The following table shows the order in which the operators are evaluated. The operators in the first line are evaluated first, then those in the second line, and so on. Operators in the same line are evaluated left to right in the expression.

Table 101: Operator Precedence

Precedence	Operator
1	Unary -, Unary +
2	**
3	*, /
4	+, -
5	=, <>, <, <=, >, >=, LIKE, NOT LIKE, IS NULL, IS NOT NULL, BETWEEN, IN, EXISTS, ANY, ALL
6	NOT
7	AND
8	OR

The following example shows the importance of precedence:

```
WHERE salary > 40000 OR
hire_date > {01/30/1989} AND
dept = 'D101'
```

Because AND is evaluated first, this query retrieves employees in department D101 hired after January 30, 1989, as well as every employee making more than \$40,000, no matter what department or hire date.

To force the clause to be evaluated in a different order, use parentheses to enclose the conditions to be evaluated first. For example:

```
WHERE (salary > 40000 OR hire_date > {01/30/1989})
AND dept = 'D101'
```

retrieves employees in department D101 that either make more than \$40,000 or were hired after January 30, 1989.

Functions

The flat-file drivers support a number of functions that you may use in expressions. In the following tables, the functions are grouped according to the type of result they return.

Table 102: Functions that Return Character Strings

Function	Description
CHR	Converts an ASCII code into a one-character string. CHR(67) returns C.
RTRIM	Removes trailing blanks from a string. RTRIM('ABC ') returns ABC.
TRIM	Removes trailing blanks from a string. TRIM('ABC ') returns ABC.
LTRIM	Removes leading blanks from a string. LTRIM(' ABC') returns ABC.
UPPER	Changes each letter of a string to uppercase. UPPER('Allen') returns ALLEN.
LOWER	Changes each letter of a string to lowercase. LOWER('Allen') returns allen.
LEFT	Returns leftmost characters of a string. LEFT('Mattson',3) returns Mat.
RIGHT	Returns rightmost characters of a string. RIGHT('Mattson',4) returns tson.
SUBSTR	Returns a substring of a string. Parameters are the string, the first character to extract, and the number of characters to extract (optional). SUBSTR('Conrad',2,3) returns onr. SUBSTR('Conrad',2) returns onrad.
SPACE	Generates a string of blanks. SPACE(5) returns ' '.

Function	Description
DTCO	<p>Converts a date to a character string. An optional second parameter determines the format of the result:</p> <p>0 (the default) returns MM/DD/YY. 1 returns DD/MM/YY. 2 returns YY/MM/DD. 10 returns MM/DD/YYYY. 11 returns DD/MM/YYYY. 12 returns YYYY/MM/DD.</p> <p>An optional third parameter specifies the date separator character. If not specified, a slash (/) is used.</p> <p>DTCO({01/30/1997}) returns 01/30/97. DTCO({01/30/1997}, 0) returns 01/30/97. DTCO({01/30/1997}, 1) returns 30/01/97. DTCO({01/30/1997}, 2, '-') returns 97-01-30.</p>
DTOS	<p>Converts a date to a character string using the format YYYYMMDD.</p> <p>DTOS({01/23/1990}) returns 19900123.</p>
IIF	<p>Returns one of two values, true or false. Parameters are a logical expression, the true value, and the false value. If the logical expression evaluates to true, the function returns the true value. Otherwise, it returns the false value.</p> <p>IIF(salary>20000, 'BIG', 'SMALL') returns BIG if salary is greater than 20000. If not, it returns SMALL.</p>
STR	<p>Converts a number to a character string. Parameters are the number, the total number of output characters (including the decimal point), and optionally the number of digits to the right of the decimal point.</p> <p>STR(12.34567, 4) returns 12. STR(12.34567, 4, 1) returns 12.3. STR(12.34567, 6, 3) returns 12.346.</p>
STRVAL	<p>Converts a value of any type to a character string.</p> <p>STRVAL('Woltman') returns Woltman. STRVAL({12/25/1953}) returns 12/25/1953. STRVAL (5 * 3) returns 15. STRVAL (4 = 5) returns 'False'.</p>
TIME	<p>Returns the time of day as a character string. At 9:49 PM, TIME() returns 21:49:00.</p>

Function	Description
TTOC	<p>Note: This function applies only to flat-file drivers that support SQL_TIMESTAMP: the Btrieve driver and the dBASE (access to FoxPro 3.0) driver.</p> <p>Converts a timestamp to a character string. An optional second parameter determines the format of the result:</p> <p>When set to 0 or none (the default), MM/DD/YY HH:MM:SS AM is returned.</p> <p>When set to 1, YYYYMMDDHHMMSS is returned, which is a suitable format for indexing.</p> <p>TTOC({1992-04-02 03:27:41}) returns 04/02/92 03:27:41 AM.</p> <p>TTOC({1992-04-02 03:27:41, 1}) returns 19920402032741</p>
USERNAME	For Btrieve, the logon ID specified at connect time is returned. For all other flat-file drivers, an empty string is returned.

Table 103: Functions that Return Numbers

Function	Description
MOD	Divides two numbers and returns the remainder of the division. MOD(10 , 3) returns 1.
LEN	Returns the length of a string. LEN('ABC') returns 3.
MONTH	Returns the month part of a date. MONTH({01/30/1989}) returns 1.
DAY	Returns the day part of a date. DAY({01/30/1989}) returns 30.
YEAR	Returns the year part of a date. YEAR({01/30/1989}) returns 1989.
MAX	Returns the larger of two numbers. MAX(66 , 89) returns 89.
DAYOFWEEK	Returns the day of week (1-7) of a date expression. DAYOFWEEK({05/01/1995}) returns 5.
MIN	Returns the smaller of two numbers. MIN(66 , 89) returns 66.

Function	Description
POW	Raises a number to a power. POW(7,2) returns 49.
INT	Returns the integer part of a number .INT(6.4321) returns 6.
ROUND	Rounds a number. ROUND(123.456, 0) returns 123. ROUND(123.456, 2) returns 123.46. ROUND(123.456, -2) returns 100.
NUMVAL	Converts a character string to a number. If the character string is not a valid number, a zero (0) is returned. NUMVAL('123') returns the number 123.
VAL	Converts a character string to a number. If the character string is not a valid number, a zero (0) is returned. VAL('123') returns the number 123.

Table 104: Functions that Return Dates

Function	Description
DATE	Returns today's date. If today is 12/25/1999, DATE() returns {12/25/1999}.
TODAY	Returns today's date. If today is 12/25/1999, TODAY() returns {12/25/1999}.
DATEVAL	Converts a character string to a date. DATEVAL('01/30/1989') returns {01/30/1989}.
CTOD	Converts a character string to a date. An optional second parameter specifies the format of the character string: 0 (the default) returns MM/DD/YY, 1 returns DD/MM/YY, and 2 returns YY/MM/DD. CTOD('01/30/1989') returns {01/30/1989}. CTOD('01/30/1989',1) returns {30/01/1989}.

The following examples use some of the number and date functions.

Retrieve all employees that have been with the company at least 90 days:

```
SELECT first_name, last_name FROM emp
```

```
WHERE DATE() - hire_date >= 90
```

Retrieve all employees hired in January of this year or last year:

```
SELECT first_name, last_name FROM emp
WHERE MONTH(hire_date) = 1
AND (YEAR(hire_date) = YEAR(DATE())
OR YEAR(hire_date) = YEAR(DATE()) - 1)
```

Create and Drop Table Statements

The flat-file drivers support SQL statements to create and delete database files. The Create Table statement is used to create files and the Drop Table statement is used to delete files.

Create Table

The form of the Create Table statement is:

```
CREATE TABLE table_name (col_definition[,col_definition,...])
```

table_name can be a simple table name or a full path name. A table name is preferred for portability to other SQL data sources. If a table name is used, the file is created in the directory you specified as the database directory in the connection string. If you did not specify a database directory in the connection string, the file is created in the directory specified as the database directory in .odbc.ini. If you did not specify a database directory in either place, the file is created in the current working directory at connect time.

col_definition is the column name, followed by the data type, followed by an optional column constraint definition. Values for column names are database specific. The data type specifies a column's data type.

The only column constraint definition currently supported by some flat-file drivers is "not NULL." Not all flat-file tables support "not NULL" columns. In the cases where "not NULL" is not supported, this restriction is ignored and the driver returns a warning if "not NULL" is specified for a column. The "not NULL" column constraint definition is allowed in the driver so that you can write a database-independent application (and not be concerned about the driver raising an error on a Create Table statement with a "not NULL" restriction).

A sample Create Table statement to create an employee database table is:

```
CREATE TABLE emp (last_name CHAR(20) NOT NULL,
first_name CHAR(12) NOT NULL,
salary NUMERIC (10,2) NOT NULL,
hire_date DATE NOT NULL)
```

Drop Table

The form of the Drop Table statement is:

```
DROP TABLE table_name
```

table_name can be a simple table name (emp) or a full path name. A table name is preferred for portability to other SQL data sources. If a table name is used, the file is dropped from the directory you specified as the database directory in the connection string. If you did not specify a database directory in the connection string, the file is deleted from the directory specified as the database directory in .odbc.ini. If you did not specify a database directory in either of these places, the file is dropped from the current working directory at connect time.

A sample Drop Table statement to delete the emp table is:

```
DROP TABLE emp
```

Insert Statement

The Insert statement is used to add new rows to a database table. With it, you can specify either of the following options:

- A list of values to be inserted as a new record
- A Select statement that copies data from another table to be inserted as a set of new rows

The form of the Insert statement is:

```
INSERT INTO table_name [(col_name, ...)]
{VALUES (expr, ...) | select_statement}
```

table_name can be a simple table name or a full path name. A table name is preferred for portability to other SQL data sources.

col_name is an optional list of column names giving the name and order of the columns whose values are specified in the Values clause. If you omit *col_name*, the value expressions (*expr*) must provide values for all columns defined in the file and must be in the same order that the columns are defined for the file.

expr is the list of expressions giving the values for the columns of the new record. Usually, the expressions are constant values for the columns. Character string values must be enclosed in single (') or double (") quotation marks, date values must be enclosed in braces {}, and logical values that are letters must be enclosed in periods (for example, .T. or .F.).

An example of an Insert statement that uses a list of expressions is:

```
INSERT INTO emp (last_name, first_name, emp_id, salary, hire_date)
VALUES ('Smith', 'John', 'E22345', 27500, {4/6/1999})
```

Each Insert statement adds one record to the database table. In this case a record has been added to the employee database table, emp. Values are specified for five columns. The remaining columns in the table are assigned a blank value, meaning NULL.

select_statement is a query that returns values for each *col_name* value specified in the column name list. Using a Select statement instead of a list of value expressions lets you select a set of rows from one table and insert it into another table using a single Insert statement.

An example of an Insert statement that uses a Select statement is:

```
INSERT INTO emp1 (first_name, last_name, emp_id, dept, salary)
SELECT first_name, last_name, emp_id, dept, salary from emp
WHERE dept = 'D050'
```

In this type of Insert statement, the number of columns to be inserted must match the number of columns in the Select statement. The list of columns to be inserted must correspond to the columns in the Select statement just as it would to a list of value expressions in the other type of Insert statement. That is, the first column inserted corresponds to the first column selected; the second inserted to the second, and so forth.

The size and data type of these corresponding columns must be compatible. Each column in the Select list should have a data type that the driver accepts on a regular Insert/Update of the corresponding column in the Insert list. Values are truncated when the size of the value in the Select list column is greater than the size of the corresponding Insert list column.

The Select statement is evaluated before any values are inserted. This query cannot be made on the table into which values are inserted.

Update Statement

The Update statement is used to change rows in a database file. The form of the Update statement supported for flat-file drivers is:

```
UPDATE table_name SET col_name = expr, ...  
[ WHERE { conditions | CURRENT OF cursor_name } ]
```

table_name can be a simple table name or a full path name. A table name is preferred for portability to other SQL data sources.

col_name is the name of a column whose value is to be changed. Several columns can be changed in one statement.

expr is the new value for the column. The expression can be a constant value or a subquery. Character string values must be enclosed with single (') or double (") quotation marks, date values must be enclosed by braces {}, and logical values that are letters must be enclosed by periods (for example, .T. or .F.). Subqueries must be enclosed in parentheses.

The Where clause (any valid clause described in [Select Statement](#) on page 1122) determines which rows are to be updated.

The Where Current Of *cursor_name* clause can be used only by developers coding directly to the ODBC API. It causes the row at which *cursor_name* is positioned to be updated. This is called a "positioned update." You must first execute a Select...For Update statement with a named cursor and fetch the row to be updated.

An example of an Update statement on the emp table is:

```
UPDATE emp SET salary=32000, exempt=1  
WHERE emp_id = 'E10001'
```

The Update statement changes every record that meets the conditions in the Where clause. In this case, the salary and exempt status are changed for all employees having the employee ID E10001. Because employee IDs are unique in the emp table, only one record is updated.

An example using a subquery is:

```
UPDATE emp SET salary = (SELECT avg(salary) FROM emp)  
WHERE emp_id = 'E10001'
```

In this case, the salary is changed to the average salary in the company for the employee having employee ID E10001.

Delete Statement

The Delete statement is used to delete rows from a database table. The form of the Delete statement supported for flat-file drivers is:

```
DELETE FROM table_name  
[ WHERE { conditions | CURRENT OF cursor_name } ]
```

table_name can be a simple table name or a full path name. A table name is preferred for portability to other SQL data sources.

The Where clause determines which rows are to be deleted. If you include only the keyword Where, all rows in the table are deleted, but the file is left intact.

The Where Current Of *cursor_name* clause can be used only by developers coding directly to the ODBC API. It causes the row at which *cursor_name* is positioned to be deleted. This is called a "positioned delete." You must first execute a Select...For Update statement with a named cursor and fetch the row to be deleted.

An example of a Delete statement on the emp table is:

```
DELETE FROM emp WHERE emp_id = 'E10001'
```

Each Delete statement removes every record that meets the conditions in the Where clause. In this case, every record having the employee ID E10001 is deleted. Because employee IDs are unique in the employee table, at most, one record is deleted.

Reserved Keywords

The following words are reserved for use in SQL statements. If they are used for file or column names in a database that you use, you must enclose them in double (") quotation marks in any SQL statement where they appear as file or column names.

ALL	FROM	LIKE	OR
AND	FULL	NATURAL	ORDER
BETWEEN	GROUP	NOT	RIGHT
COMPUTE	HAVING	NULL	UNION
CROSS	INNER	ON	WHERE
DISTINCT	INTO	OPTIONS	
FOR	LEFT	OR	

SQL Functionality for the Impala Wire Protocol Driver

The DataDirect Connect XE *for* ODBC and DataDirect Connect64 XE *for* ODBC for Impala Wire Protocol driver support an extended set of SQL 92, in addition to the syntax for Impala SQL, which is a subset of SQL 92.

Refer to the [Impala Language Reference](#) documentation for information about using Impala SQL.

Data Definition Language (DDL)

The Impala Wire Protocol driver supports a broad set of DDL, including (but not limited to) the following:

- CREATE Database and DROP Database
- CREATE Table and DROP Table
- ALTER Table and Alter Partition statements

Refer to the [Impala Language Reference](#) documentation for information about using Impala SQL.

Selecting Data With the Driver

Select List

The following sections apply to the way the Select list can be used with the driver.

Column Name Qualification

A column can only be qualified with a single name. Furthermore, a table can be qualified with a database (ODBC schema) name in the FROM clause, and in some cases, must also be aliased. Aliasing may not be necessary if the database qualifier is not the current database.

The driver can work around these limitations using the Remove Column Qualifiers connection option.

- If set to 1, the driver removes three-part column qualifiers and replaces them with alias.column qualifiers.
- If set to 0, the driver does not do anything with the request.

Suppose you have the following ANSI SQL query:

```
SELECT schema.table1.col1,schema.table2.col2 FROM schema.table1,schema.table2
WHERE schema.table1.col3=schema.table2.col3
```

If the Remove Column Qualifiers connection option is enabled, the driver replaces the three-part column qualifiers:

```
SELECT table1.col1,
table2.col2 FROM schema.table1 table1 JOIN schema.table2 table2
WHERE table1.col3 = table2.col3
```

From Clause

LEFT, RIGHT, and FULL OUTER JOINS are supported, as are LEFT SEMI JOINS and CROSS JOINS using the equal comparison operator, as shown in the following examples:

```
SELECT a.* FROM a JOIN b ON (a.id = b.id AND a.department = b.department)
```

```
SELECT a.val, b.val, c.val FROM a JOIN b ON (a.key = b.key1) JOIN c ON
(c.key = b.key2)
```

```
SELECT a.val, b.val FROM a LEFT OUTER JOIN b ON (a.key=b.key)
WHERE a.ds='2009-07-07' AND b.ds='2009-07-07'
```

However, the following syntax fails because of the use of non-equal comparison operators.

```
SELECT a.* FROM a JOIN b ON (a.id <> b.id)
```

Impala SQL does not support join syntax in the form of a comma-separated list of tables. The driver, however, overcomes this limitation by translating the SQL into Impala SQL, as shown in the following examples.

ANSI SQL 92 Query

```
SELECT * FROM t1, t2 WHERE a = b
```

```
SELECT * FROM t1 y, t2 x WHERE a = b
```

Impala SQL Translation

```
SELECT * FROM t1 JOIN t2 WHERE a = b
```

```
SELECT * FROM t1 y JOIN t2 x WHERE a = b
```

ANSI SQL 92 Query

```
SELECT * FROM t2, (SELECT *
FROM t1) x
```

Impala SQL Translation

```
SELECT * FROM t2 t2 JOIN (SELECT * FROM t1 t1) x
```

Group By Clause

The Group By clause is supported, with the following Entry SQL level restrictions:

- The COLLATE clause is not supported.
- SELECT DISTINCT is not supported.

Having Clause

The Having Clause is supported, with the following Entry SQL level restriction: a GROUP BY clause is required.

Order By Clause

The Order By clause is supported, with the following Entry SQL level restrictions:

- An integer sort key is not allowed.
- The COLLATE clause is not supported.
- A LIMIT clause or a default ORDER BY limit for result sets is required.

Default ORDER BY Limit

Impala will not return result sets for statements containing an order by clause unless a limit clause is included or a default limit is specified for result sets. A default limit can be set in the server; however, this limits the result sets for all queries, not just statements containing the ORDER BY clause.

The driver can work around these limitations by using the Default Order By Limit connection option.

where:

x

is a positive integer that represents maximum number of rows returned.

If set to x , the number of rows returned by a SQL statement containing an ORDER BY clause are limited to the specified number of rows for the session. To override this value, specify a new value in a LIMIT clause in the statement that is being executed.

If set to -1 (disabled), there is no default limit the number of rows returned by a statement containing an ORDER BY clause. The application must limit the number of rows returned by SQL statements that contain an ORDER BY clause, or Impala will return an error.

For Update Clause

Not supported in this release. If present, the driver strips the For Update clause from the query.

Set Operators

Supported, with the following restrictions: INTERSECT or EXCEPT are not supported.

Subqueries

A query is an operation that retrieves data from one or more tables or views. In this reference, a top-level query is called a Select statement, and a query nested within a Select statement is called a subquery.

Subqueries are supported, with the following Entry SQL level restriction: subqueries can only exist in the FROM clause, that is, in a derived table. In the following example, the second Select statement is a subquery:

```
SELECT * FROM (SELECT * FROM t1 UNION ALL SELECT * FROM t2) sq
```

Although Impala currently does not support IN or EXISTS subqueries, you can efficiently implement the semantics by rewriting queries to use LEFT SEMI JOIN.

SQL Expressions

An expression is a combination of one or more values, operators, and SQL functions that evaluate to a value. You can use expressions in the Where and Having clauses of Select statements.

Expressions enable you to use mathematical operations as well as character string manipulation operators to form complex queries.

Valid expression elements are:

- [Constants](#) on page 1140
- [Numeric Operators](#) on page 1140
- [Character Operator](#) on page 1141
- [Relational Operators](#) on page 1141
- [Logical Operators](#) on page 1142
- [Functions](#) on page 1142

Constants

Impala uses binary literals for internal functions. Although the driver supports binary literals, no useful information is returned.

Numeric Operators

You can use a numeric operator in an expression to negate, add, subtract, multiply, and divide numeric values. The result of this operation is also a numeric value. The + and - operators are also supported in date/time fields to allow date arithmetic.

The following table lists the supported arithmetic operators.

Table 105: Numeric Operators

Entry SQL Level Operator	Impala SQL Operator
N/A	% (Mod)
N/A	& (bitwise AND)
*	Supported
+	Supported
-	Supported
/	Supported
N/A	^ (XOR)

Character Operator

The concatenation operator (||) is not supported; however, the CONCAT function is supported by Impala SQL.

```
SELECT CONCAT('Name is', '(ename FROM emp)')
```

Relational Operators

Relational operators compare one expression to another.

The following table lists the supported relational operators.

Table 106: Relational Operators Supported with Impala

Entry SQL Level Operator	Support in Impala SQL
<>	Supported
<	Supported
<=	Supported
=	Supported
<=>	Supported
>	Supported
>=	Supported
IS [NOT] NULL	Supported
[NOT] BETWEEN x AND y	Supported
[NOT] IN	Supported

Entry SQL Level Operator	Support in Impala SQL
EXISTS	Supported
[NOT] LIKE	Supported, except that no collate clause is allowed
RLIKE	Supported
REGEXP	Supported

Logical Operators

A logical operator combines the results of two component conditions to produce a single result or to invert the result of a single condition. The following table lists the supported logical operators.

Table 107: Logical Operators

Operator	Support in Impala HQL
NOT !	Supported
AND &&	Supported
OR	Supported

Functions

The following tables show how SQL-92 functions are supported in the Impala Query Language. Additional methods may be supported with ODBC Escapes.

Table 108: Set Functions Supported

Set Function	Support in Impala SQL
Count	Supported
AVG	Supported
MIN	Supported
MAX	Supported
SUM	Supported
DISTINCT	Supported
ALL	Supported

Table 109: Numeric Functions Supported

Numeric Function	Support in Impala SQL
CHAR_LENGTH CHARACTER_LENGTH	Not supported. Use LENGTH(string) instead.
Position...In	Not supported
BIT_LENGTH(s)	Not supported
OCTET_LENGTH(str)	Not supported
EXTRACT TIMEZONE_HOUR FROM	Not supported
EXTRACT TIMEZONE_MINUTE FROM	Not supported

Table 110: String Functions Supported

String Function	Support in Impala SQL
Substring	Supported
Convert ... using	Not supported
TRIM	Supported.
Leading	Not supported. Use LTRIM.
Trailing	Not supported. Use RTRIM.
Both	Not supported (default behavior of TRIM)

Table 111: Date/Time Functions Supported

Date/Time Function	Support in Impala SQL
CURRENT_DATE()	Not supported ⁹³
CURRENT_TIME()	Not supported ⁹³
CURRENT_TIMESTAMP	Not supported. Use UNIX_TIMESTAMP() ⁹³ .

Table 112: System Functions Supported

System Function	Support in Impala SQL
CASE . . . END	Supported.
COALESCE	Supported.

⁹³ Supported by ODBC Escapes

System Function	Support in Impala SQL
NULLIF	Not supported. ⁹³
CAST	Supported.

Restrictions

This section describes some of the functional restrictions of Impala.

Insert and Update Restrictions

Impala does not support row level INSERT, UPDATE, and DELETE. Impala-specific syntax provides limited INSERT support:

- INSERT INTO is supported.
- INSERT OVERWRITE is supported.
- The name of the target table for an Insert must be prefixed by `table`, for example, `INSERT INTO table gtable ...`

Column lists are not supported. Values for all columns must be specified.

Stored Procedures

Impala has no concept of stored procedures. Therefore, they are not supported by the driver.

Views

Impala has no concept of views. Therefore, they are not supported by the driver.

Other Restrictions

The Impala server has the following restrictions:

- Column values and parameters are always nullable
- No ROWID support
- No support for synonyms
- Primary and foreign keys are not supported.
- The length of a SQL string is limited to 2 GB.
- Support for indexes is incomplete.

SQL Statements and Extensions for the Salesforce Driver

The Salesforce driver provides support for standard SQL (primarily SQL 92). In addition, the product supports a set of SQL extensions. For example, the product supports extensions that allow you to change the default schema or set the maximum number of Web service calls the driver can make when executing a SQL statement.

This chapter describes both the standard SQL statements and the SQL extensions. SQL extensions are denoted by an (EXT) in the topic title.

Alter Cache (EXT)

Purpose

The Alter Cache statement changes the definition of a cache on a remote table or view. An error is returned if the remote table or view specified does not exist.

Syntax

```
ALTER CACHE ON {remote_table | view}
  [REFERENCING (remote_table_ref[,remote_table_ref]...)]
  [REFRESH_INTERVAL {0 | -1 | interval_value [{M, H, D}]}]
  [INITIAL_CHECK [ONFIRSTCONNECT | FIRSTUSE | DEFAULT]]
  [PERSIST {TEMPORARY | MEMORY | DISK | DEFAULT}]
  [ENABLED {YES | TRUE | NO | FALSE}]
  [CALL_LIMIT {0 | -1 | max_calls}]
  [FILTER (expression)]
```

where:

remote_table

is the name of the remote table cache definition to be modified. The remote table name can be a two-part name: *schemaname.tablename*. When specifying a two-part name, the specified remote table must be defined in the specified schema, and you must have the privilege to alter objects in the specified schema. When altering a relational cache, *remote_table* must specify the primary table of the relational cache.

view

is the name of the view cache definition to be modified. The view name can be a two-part name: *schemaname.viewname*. When specifying a two-part name, the specified view must be defined in the specified schema, and you must have the privilege to alter objects in the specified schema. Caches on views are not currently supported in the product.

REFERENCING

is an optional clause that specifies the name of the remote table(s) for which a relationship cache is to be created. See [Relational Caches](#) on page 1146 and [Referencing Clause](#) on page 1156 for a complete explanation.

REFRESH_INTERVAL

is an optional clause that specifies the length of time the data in the cached table can be used before being refreshed. See [Refresh Interval Clause](#) on page 1157 for a complete explanation.

INITIAL_CHECK

is an optional clause that specifies when the driver initially checks whether the data in the cache needs refreshed. See [Initial Check Clause](#) on page 1157 for a complete explanation.

PERSIST

is an optional clause that specifies the life span of the data in the cached table or view. See [Persist Clause](#) on page 1158 for a complete explanation.

ENABLED

is an optional clause that specifies whether the cache is enabled or disabled for use with SQL statements. See [Enabled Clause](#) on page 1158 for a complete explanation.

CALL_LIMIT

is an optional clause that specifies the maximum number of Web service calls that can be used to populate or refresh the cache. See [Call Limit Clause](#) on page 1159 for a complete explanation.

FILTER

is an optional clause that specifies a filter for the primary table to limit the number of rows that are cached in the primary table. See [Filter Clause](#) on page 1160 for a complete explanation.

Notes

- At least one of the optional clauses must be used. If two or more are specified, they must be specified in the order shown in the grammar description.

Relational Caches

If the Referencing clause is specified, the Alter Cache statement drops the existing cache and any referenced caches and creates a new set of related caches, one for each of the tables specified in the statement. The cache attributes for the existing cache are the default cache attributes for the new relational cache. Any attributes specified in the Alter Cache statement override the default attributes. If the Referencing clause is not specified, the existing cache references, if any, are used.

If the cache being altered is a relational cache, the attributes specified in the Alter Cache statement apply to all of the caches that comprise the relational cache.

Alter Index

Purpose

The Alter Index statement changes the name of an existing index. Index names must not conflict with other user-defined or system-defined names.

Syntax

```
ALTER INDEX index_name RENAME TO new_name
```

where:

index_name

specifies an existing index name.

new_name

specifies the new index name.

Alter Sequence

Purpose

The Alter Sequence statement resets the next value of an existing sequence.

Syntax

```
ALTER SEQUENCE sequence_name RESTART WITH value
```

where:

sequence_name

specifies an existing sequence.

value

specifies the next value to be returned through the Next Value For clause (see [Next Value For Clause](#) on page 1162).

Notes

- Indexes on remote tables cannot be created, altered or dropped. Indexes can only be defined on local tables by the driver.

Alter Session (EXT)

Purpose

The Alter Session statement changes various attributes of a database session or a remote session. A database session maintains the state of the overall connection. A remote session maintains the state that pertains to a particular remote data source connection.

Syntax

```
ALTER SESSION SET attribute_name=value
```

where:

attribute_name

specifies the name of the attribute to be changed. Attributes apply to either database sessions or remote sessions.

value

refers to the specific value setting for that attribute.

The following table lists the database and remote session attributes, and provides their descriptions.

Table 113: Alter Session Attributes

Attribute Name	Session Type	Description
Current_Schema	Database	Sets the current schema for the database session. The current schema is the schema used when an identifier in a SQL statement is unqualified. The string value must be the name of a schema visible in the database session. For example: <code>ALTER SESSION SET CURRENT_SCHEMA=sforce</code>
Stmt_Call_Limit	Database	Sets the maximum number of Web service calls the driver can make in executing a statement. Setting the Stmt_Call_Limit attribute has the same effect as setting the StmtCallLimit connection option. It sets the default Web service call limit used by any statement on the connection. Executing this command on a statement overrides the previously set StmtCallLimit for the connection. The value specified must be a positive integer or 0. The value 0 means that no call limit exists. For example: <code>ALTER SESSION SET STMT_CALL_LIMIT=10</code>
Ws_Call_Count	Remote	Resets the Web service call count of a remote session to the value specified. The value must be zero or a positive integer. WS_Call_Count represents the total number of Web service calls made to the remote data source instance for the current session. For example: <code>ALTER SESSION SET sforce.WS_CALL_COUNT=0</code> The current value of WS_Call_Count can be obtained by referring to the System_Remote_Sessions system table (see SYSTEM_REMOTE_SESSIONS Catalog Table in "The Salesforce Driver" chapter for details). For example: <code>SELECT * FROM information_schema.system_remote_sessions WHERE session_id = cursessionid()</code>

Alter Table

Purpose

The Alter Table statement adds or removes a column. The table being altered can be either a remote or local table. A remote table is a Salesforce object and is exposed in the SFORCE schema. A local table is maintained by the driver and is local to the machine on which the driver is running. A local table is exposed in the PUBLIC schema.

For information on...

Altering a remote table

Altering a local table

See

[Altering a Remote Table](#) on page 1149

[Altering a Local Table](#) on page 1151

Altering a Remote Table

Syntax

```
ALTER TABLE table_name[add_clause] [drop_clause]
```

where:

table_name

specifies an existing remote table.

add_clause

specifies a column or a foreign key constraint to be added to the table. See [Add Clause: Columns](#) on page 1149 and [Add Clause: Constraints](#) on page 1150 for a complete explanation.

drop_clause

specifies a column to be dropped from the table. See [Drop Clause: Columns](#) on page 1151 for a complete explanation.

Notes

- You cannot drop a constraint from a remote table.

Add Clause: Columns

Purpose

Use the Add clause to add a column to an existing table. It is optional.

This clause adds a column to the table. It defines a column with the same syntax as the Create Table command (see [Column Definition for Remote Tables](#) on page 1163).

Syntax

```
ADD [COLUMN] column_nameDatatype ... [DEFAULT default_value] [[NOT]NULL] [EXT_ID]
[PRIMARY KEY] [START WITH starting_value]
```

default_value

is the default value to be assigned to the column. See [Column Definition for Remote Tables](#) on page 1163 for details.

starting_value

is the starting value for the Identity column. The default start value is 0.

Notes

- If `NOT NULL` is specified and the table is not empty, a default value must be specified. In all other respects, this command is the equivalent of a column definition in a Create Table statement.
- You cannot specify `ANYTYPE`, `BINARY`, `COMBOBOX`, or `TIME` data types in the column definition of Alter Table statements.
- If a SQL view includes `SELECT * FROM` for the table to which the column was added in the view's Select statement, the new column is added to the view.

Example A

Assuming the current schema is `SFORCE`, this example adds the `status` column with a default value of `ACTIVE` to the `test` table.

```
ALTER TABLE test ADD COLUMN status TEXT(30) DEFAULT 'ACTIVE'
```

Example B

Assuming the current schema is `SFORCE`, this example adds a `deptId` column that can be used as a foreign key column.

```
ALTER TABLE test ADD COLUMN deptId TEXT(18)
```

Add Clause: Constraints

Purpose

Use the Add clause to add a constraint to an existing table. It is optional.

This command adds a constraint using the same syntax as the Create Table command (see [Column Definition for Remote Tables](#) on page 1163).

Syntax

```
ADD [CONSTRAINT constraint_name] ...
```

Notes

- The only type of constraint you can add is a foreign key constraint.
- When adding a foreign key constraint, the table that contains the foreign key must be empty.

Example A

Assuming the current schema is `SFORCE`, a foreign key constraint is added to the `deptId` column of the `test` table, referencing the `rowId` of the `dept` table. For the operation to succeed, the `dept` table must be empty.

```
ALTER TABLE test ADD FOREIGN KEY (deptId) REFERENCES dept(rowId)
```

Drop Clause: Columns

Purpose

Use the Drop clause to drop a column from an existing table. It is optional.

Syntax

```
DROP {[COLUMN] column_name | [CONSTRAINT] constraint_name}
```

where:

column_name

specifies an existing column in an existing table.

Notes

- The column being dropped cannot have a constraint defined on it.
- Drop fails if a SQL view includes the column.

Example A

This example drops the `status` column. For the operation to succeed, the `status` column cannot have a constraint defined on it and cannot be used in a SQL view.

```
ALTER TABLE test DROP COLUMN status
```

Altering a Local Table

Syntax

```
ALTER TABLE table_name [add_clause] [drop_clause] [rename_clause]
```

where:

table_name

specifies an existing local table.

add_clause

specifies a column or constraint to be added to the table. See [Add Clause: Columns](#) on page 1152 and [Add Clause: Constraints](#) on page 1150 for a complete explanation.

drop_clause

specifies a column or constraint to be dropped from the table. See [Drop Clause: Columns](#) on page 1151 and [Drop Clause: Constraints](#) on page 1153 for a complete explanation.

rename_clause

specifies a new name for the table. See [Rename Clause](#) on page 1154 for a complete explanation.

Add Clause: Columns

Purpose

Use the Add clause to add a column to an existing table. It is optional.

This clause adds a column to the end of the column list. It defines a column with the same syntax as the Create Table command (see [Column Definition for Remote Tables](#) on page 1163). If `NOT NULL` is specified and the table is not empty, a default value must be specified. In all other respects, this command is the equivalent of a column definition in a Create Table statement.

Syntax

```
ADD [COLUMN] column_nameDatatype ... [BEFORE existing_column]
```

Notes

- You cannot specify ANYTYPE, BINARY, COMBOBOX, or TIME data types in the column definition of Alter Table statements.
- The optional *Before existing_column* can be used to specify the name of an existing column so that the new column is inserted in a position just before the existing column.
- The optional *Before existing_column* can be used to specify the name of an existing column so that the new column is inserted in a position just before the existing column.
- If a SQL view includes `SELECT * FROM` for the table to which the column was added in the view's Select statement, the new column is added to the view.

Example A

Assuming the current schema is PUBLIC, this example adds the *status* column with a default value of `ACTIVE` to the *test* table.

```
ALTER TABLE test ADD COLUMN status VARCHAR(30) DEFAULT 'ACTIVE'
```

Example B

Assuming the current schema is PUBLIC, this example adds a *deptId* column that can be used as a foreign key column.

```
ALTER TABLE test ADD COLUMN deptId VARCHAR(18)
```

Add Clause: Constraints

Purpose

Use the Add clause to add a constraint to an existing table. It is optional.

This command adds a constraint using the same syntax as the Create Table command (see [Constraint Definition for Local Tables](#) on page 1169).

Syntax

```
ADD [CONSTRAINT constraint_name] ...
```

Notes

- You cannot add a Unique constraint if one is already assigned to the same column list. A Unique constraint works only if the values of the columns in the constraint columns list for the existing rows are unique or include a Null value.

- Adding a foreign key constraint to the table fails if, for each existing row in the referring table, a matching row (with equal values for the column list) is not found in the referenced table.

Example A

Assuming the current schema is PUBLIC, this example adds a foreign key constraint to the `deptId` column of the `test` table that references the `rowId` of the `dept` table.

```
ALTER TABLE test ADD CONSTRAINT test_fk FOREIGN KEY (deptId) REFERENCES dept(id)
```

Drop Clause: Columns

Purpose

Use the Drop clause to drop a column from an existing table. It is optional.

Syntax

```
DROP {[COLUMN] column_name}
```

where:

column_name

specifies an existing column in an existing table.

Notes

- Drop fails if a SQL view includes the column.

Example A

This example drops the `status` column. For the operation to succeed, the `status` column cannot have a constraint defined on it and cannot be used in a SQL view.

```
ALTER TABLE test DROP COLUMN status
```

Drop Clause: Constraints

Purpose

Use the Drop clause to drop a constraint from an existing table. It is optional.

Syntax

```
DROP {[CONSTRAINT] constraint_name}
```

where:

constraint_name

specifies an existing constraint.

Notes

- The specified constraint cannot be a primary key constraint or unique constraint.

Example A

This example drops the `test_fk` constraint.

```
ALTER TABLE test DROP CONSTRAINT test_fk
```

Rename Clause

Purpose

Use the Rename clause to rename an existing table. It is optional.

Syntax

```
RENAME TO new_name
```

where:

```
new_name
```

specifies the new name for the table.

Example A

This example renames the table to `test2`.

```
ALTER TABLE test RENAME TO test2
```

Checkpoint

Purpose

The Checkpoint statement ensures that all database changes in memory are committed to disk. Executing the Checkpoint statement closes the database files, rewrites the script file, deletes the log file, and reopens the database.

Syntax

```
CHECKPOINT [DEFRAG]
```

Notes

- If `DEFRAG` is specified, this statement evaluates abandoned space in the database data (.data) file and shrinks the data file to its minimum size.

Create Cache (EXT)

Purpose

The Create Cache statement creates a cache that holds the data of a remote table. The data is not loaded into the cache when the Create Cache statement is executed; the data is loaded the first time that the remote table is executed or when a Refresh Cache statement on the remote table is executed. An error is returned if the remote table specified does not exist.

Syntax

```
CREATE CACHE ON {remote_table}  
  [REFERENCING (remote_table_ref[,remote_table_ref]...)]  
  [REFRESH_INTERVAL {0 | -1 | interval_value [{M, H, D}]}]  
  [INITIAL_CHECK [{ONFIRSTCONNECT | FIRSTUSE | DEFAULT}]]
```

```
[PERSIST {TEMPORARY | MEMORY | DISK | DEFAULT}]
[ENABLED {YES | TRUE | NO | FALSE}]
[CALL_LIMIT {0 | -1 | max_calls}]
[FILTER (expression)]
```

where:

remote_table

is the name of the remote table from which data is to be cached on the client. The name of the cached table is the same as the name of the remote table. When the table name is specified in a query, the cached table is accessed, not the remote table.

The remote table name can be a two-part name: *schemaname.tablename*. When specifying a two-part name, the specified remote table must be defined in the specified schema, and you must have the privilege to create objects in the specified schema.

REFERENCING

is an optional clause that specifies the name of the remote table(s) for which a relationship cache is to be created. See [Relational Caches](#) on page 1146 and [Referencing Clause](#) on page 1156 for a complete explanation.

REFRESH_INTERVAL

is an optional clause that specifies the length of time the data in the cached table can be used before being refreshed. See [Refresh Interval Clause](#) on page 1157 for a complete explanation.

INITIAL_CHECK

is an optional clause that specifies when the driver initially checks whether the data in the cache needs refreshed. See [Initial Check Clause](#) on page 1157 for a complete explanation.

PERSIST

is an optional clause that specifies the life span of the data in the cached table or view. See [Persist Clause](#) on page 1158 for a complete explanation.

ENABLED

is an optional clause that specifies whether the cache is enabled or disabled for use with SQL statements. See [Enabled Clause](#) on page 1158 for a complete explanation.

CALL_LIMIT

is an optional clause that specifies the maximum number of Web service calls that can be used to populate or refresh the cache. See [Call Limit Clause](#) on page 1159 for a complete explanation.

FILTER

is an optional clause that specifies a filter for the primary table to limit the number of rows that are cached in the primary table. See [Filter Clause](#) on page 1160 for a complete explanation.

Notes

- Caches on views are not supported.

- If two or more optional clauses are specified, they must be specified in the order shown in the grammar description.

Relational Caches

If the Referencing clause is specified, the Create Cache statement creates a set of related caches, one for each of the tables specified in the statement. This set of caches is referred to as a related or relational cache. The set of caches in a relational cache is treated as a single entity. They are refreshed, altered, and dropped as a unit. Any attributes specified in the Create Cache statement apply to the cache created for the primary table and to the caches created for all of the referenced tables specified.

A database session can have both standalone and relational caches defined, but only one cache can be defined on a table. If a table is referenced in a relational cache definition, a standalone cache cannot be created on that table.

Referencing Clause

Purpose

The Referencing clause specifies the name of the remote table(s) for which a relationship cache is to be created; it is optional. The specified remote table must be related to either the primary table being cached or one of the other specified related tables. The remote table name cannot include a schema name. The referenced tables must exist in the same schema as the primary table.

Syntax

```
REFERENCING (remote_table_ref[,remote_table_ref]...)]
```

where:

remote_table_ref

represents *remote_table*[.*foreign_key_name*]

remote_table

specifies one or more tables related to the primary table that are to be cached in conjunction with the primary table.

foreign_key_name

specifies the name of the foreign key relationship between the remote table and the primary table (or, optionally, another related table). If a foreign key name is not specified, the driver attempts to find a relationship between the remote table and one of the other tables specified in the relational cache. The driver first looks for a relationship to the primary table. If a relationship to the primary table does not exist, the driver then looks for a relationship to other referenced tables.

Refresh Interval Clause

Purpose

The Refresh Interval clause specifies the length of time the data in the cached table can be used before being refreshed; it is optional. The driver maintains a timestamp of when the data in a table was last refreshed. When a cached table is used in a query, the driver checks if the current time is greater than the last refresh time plus the value of Refresh_Interval. If it is, the driver refreshes the data in the cached table before processing the query.

Syntax

```
[REFRESH_INTERVAL {0 | -1 | interval_value [{M, H, D}]}]
```

where:

0

specifies that the cache is refreshed manually. You can use the Refresh Cache statement to refresh the cache manually.

-1

resets the refresh interval to the default value of 12 hours.

interval_value

is a positive integer that specifies the amount of time between refreshes. The default unit of time is hours (H). You can also specify M for minutes or D for days. For example, 60M would set the time between refreshes to 60 minutes. The default refresh interval is 12 hours.

Initial Check Clause

Purpose

The Initial Check clause specifies when the driver performs its initial check of the data in the cache to determine whether it needs to be refreshed; it is optional.

Syntax

```
[INITIAL_CHECK [ONFIRSTCONNECT | FIRSTUSE | DEFAULT]]
```

where:

ONFIRSTCONNECT

specifies that the initial check is performed the first time a connection for a user is established. Subsequently, it is performed each time the table or view is used. A driver session begins on the first connection for a user and the session is active as long as at least one connection is open for the user.

FIRSTUSE

specifies that the initial check is performed the first time the table or view is used in a query. Subsequently, it is performed each time the table or view is used.

DEFAULT

resets the value back to its default, which is `FIRSTUSE`.

Persist Clause

Purpose

The `Persist` clause specifies the life span of the data in the cached table or view; it is optional.

Syntax

```
[ PERSIST { TEMPORARY | MEMORY | DISK | DEFAULT } ]
```

where:

TEMPORARY

specifies that the data exists for the life of the driver session. When the driver session ends, the data is discarded. A driver session begins on the first connection for a user and the session is active if at least one connection is open for the user.

MEMORY

specifies that the data exists beyond the life of the connection. While the connection is active, the cached data is stored in memory. When the connection is closed, the cached data is persisted to disk. If the connection ends abnormally, changes to the cached data may not be persisted to disk. This is the default.

DISK

specifies that the data exists beyond the life of the connection. A portion of the cached data is stored in memory while the connection is active. If the size of the cached data exceeds the cache memory threshold, the remaining data is stored on disk. When the connection is closed, the portion of the cached data that is in memory is persisted to disk. If the connection ends abnormally, changes to the cached data held in memory may not be persisted to disk.

DEFAULT

resets the `PERSIST` value back to its default, which is `MEMORY`.

Notes

- You can design your application to force all cached data held in memory to be persisted to disk at any time by using the [Checkpoint](#) on page 1154 statement.
- If you specify a value of `MEMORY` or `DISK` for the `Persist` clause, the remote data remains on the client past the lifetime of the application.

Enabled Clause

Purpose

The `Enabled` clause specifies whether the cache is enabled or disabled for use with SQL statements; it is optional.

Syntax

```
[ENABLED {YES | TRUE | NO | FALSE}]
```

where:

```
YES | TRUE
```

specifies that the cache is enabled. When a cache is enabled, the driver accesses the cached data for the remote table or view when a query is executed.

The driver does not check whether the cache needs to be refreshed when the Alter Cache statement is used to enable the cache. The check occurs the next time that the cache is accessed.

```
NO | FALSE
```

specifies that the cache is disabled, which means that the driver accesses the data in the remote table or view rather than the cache when a query is executed. The driver does not update the cache when inserts, updates, and deletes are performed on a remote table or view. To use the cache, you must enable it.

All data in an existing cache is persisted on the client even when the cache is disabled, except for the case where `PERSIST` is set to `TEMPORARY`.

The default is `TRUE`.

Call Limit Clause

Purpose

The Call Limit clause specifies the maximum number of Web service calls that can be used to populate or refresh the cache; it is optional.

Syntax

```
[CALL_LIMIT {0 | -1 | max_calls}]
```

where:

```
0
```

specifies no call limit.

```
-1
```

resets the call limit back to its default, which is 0 (no call limit).

```
max_calls
```

is a positive integer that specifies the maximum number of Web service calls.

The default value is 0.

Notes

- The call limit for a cache is independent of the `Stmt_Call_Limit` set on a database session. See [Alter Session \(EXT\)](#) on page 1147 for details.

If the call limit of a cache is exceeded during the population or refresh of the cache, the cache is marked as partially initialized. At the next refresh opportunity, the driver attempts to complete the population or refresh of the cache. If the call limit (or other error) occurs during this second attempt, the cache becomes invalid and is disabled. All data in the cache is discarded after the second attempt to populate or refresh the cache fails. Before re-enabling the cache, consider altering the cache definition to allow more Web service calls or specify a more restrictive filter, or both.

Filter Clause

Purpose

Filter is an optional clause that specifies a filter for the primary table to limit the number of rows that are cached in the primary table. This clause is not supported for views.

Syntax

```
[FILTER (expression) ]
```

where:

expression

is any valid Where clause. See [Where Clause](#) on page 1186 for details. Do not include the Where keyword in the clause. The filter for an existing cache can be removed by specifying an empty string for the filter expression, for example, `FILTER ()`.

The default value is that cached data is not filtered.

Example

The following example filters by last activity date.

```
FILTER (lastactivitydate >= {d'2010-01-01'})
```

Example A

The Referencing clause allows multiple related tables to be cached as a single entity. The following example creates a cache on the remote table `account`. The cache is populated with all accounts that had activity in 2010. Additionally, caches are created for the following remote tables: `opportunity`, `contact`, and `opportunitylineitem`. These caches are populated with the opportunities and contacts that are associated with the accounts stored in the `accounts` cache and the opportunity line items associated with the opportunities stored in the `opportunity` cache.

```
CREATE CACHE ON account REFERENCING (opportunity, contact, opportunitylineitem)
FILTER (lastactivitydate >= {d'2010-01-01'})
```

Example B

The following example caches all rows of the `account` table with a refresh interval of 12 hours, checks whether data of the cached table needs to be refreshed on the first use, persists the data beyond the life of the connection, and stores the data in memory while the connection is active.

```
CREATE CACHE ON account
```

Example C

The following example caches all active accounts in the `account` table with a refresh interval of 1 day, checks whether data of the cached table needs to be refreshed when the connection is established, and discards the data when the connection is closed.


```
CREATE CACHE ON account REFRESH_INTERVAL 1d INITIAL_CHECK ONFIRSTCONNECT PERSIST
TEMPORARY FILTER(account.active = 'Yes')
```

Create Index

Purpose

The Create Index statement creates an index on one or more columns in a local table.

Syntax

```
CREATE [UNIQUE] INDEX index_name ON table_name (column_name [, ...])
```

where:

`UNIQUE`

means that key columns cannot have duplicate values.

index_name

specifies the name of the index to be created.

table_name

specifies an existing local table.

column_name

specifies an existing column.

Notes

- The driver cannot create an index in a remote table; the driver returns an error indicating that the operation cannot be performed on a remote table.
- Creating a unique constraint is the preferred way to specify that the values of a column must be unique.

Create Sequence

Purpose

The Create Sequence statement creates an auto-incrementing sequence for a local table.

Syntax

```
CREATE SEQUENCE sequence_name [AS {INTEGER | BIGINT}] [START WITH start_value]
[INCREMENT BY increment_value]
```

where:

sequence_name

specifies the name of the sequence. By default, the sequence type is INTEGER.

start_value

specifies the starting value of the sequence. The default start value is 0.

increment_value

specifies the value of the increment; the value must be a positive integer. The default increment is 1.

Next Value For Clause

Purpose

Use the Next Value For clause to specify the next value for a sequence that is used in a Select, Insert, or Update statement.

Syntax

```
NEXT VALUE FOR sequence_name
```

where:

sequence_name

specifies the name of the sequence from which to retrieve the value.

Example

The following example retrieves the next value or set of values in Sequence1:

```
SELECT NEXT VALUE FOR Sequence1 FROM Account
```

Create Table

For information on...

Creating a remote table

Creating a local table

See...

[Creating a Remote Table](#) on page 1162

[Creating a Local Table](#) on page 1167

Creating a Remote Table

Purpose

Use the Create Table statement to create a new table. You can create either a remote or local table. A remote table is a Salesforce object and is exposed in the SFORCE schema. Creating a table in the SFORCE schema creates a remote table. A local table is maintained by the driver and is local to the machine on which the driver is running. A local table is exposed in the PUBLIC schema. Creating a table in the PUBLIC schema creates a local table.

Syntax

```
CREATE TABLE table_name (column_definition [, ...] [, constraint_definition...])
```

where:

table_name

specifies the name of the new remote table. The table name can be qualified by a schema name using the format *schema.table*. If the schema is not specified, the table is created in the current schema. See [Alter Session \(EXT\)](#) on page 1147 for information about changing the current schema.

column_definition

specifies the definition of a column in the new table. See [Column Definition for Remote Tables](#) on page 1163 for a complete explanation.

constraint_definition

specifies constraints on the columns of the new table. See [Constraint Definition for Remote Tables](#) on page 1164 for a complete explanation.

Notes

- Creating tables in Salesforce is not a quick operation. It can take several minutes for Salesforce to create the table and its relationships.

Column Definition for Remote Tables

Purpose

Defines the syntax to define a column for remote tables.

Syntax

```
column_name Datatype [(precision[,scale])...] [DEFAULT
default_value][[NOT]NULL][EXT_ID][PRIMARY KEY] [START WITH starting_value]
```

where:

column_name

is the name to be assigned to the column.

Datatype

is the data type of the column to be created. See Data Types in "The Salesforce Driver" chapter of the *DataDirect Connect Series for ODBC User's Guide* for a list of supported Salesforce data types. You cannot specify ANYTYPE, BINARY, COMBOBOX, ENCRYPTEDTEXT, or TIME data types in the column definition of Create Table statements.

precision

is the total number of digits for DECIMAL columns, the number of seconds for DATETIME columns, and the length of HTML, LONGTEXTAREA, and TEXT columns.

scale

is the number of digits to the right of the decimal point for DECIMAL columns.

default_value

is the default value to be assigned to the column. The following default values are allowed in column definitions for remote tables:

- For character columns, a single-quoted string or NULL.
- For datetime columns, a single-quoted Date, Time, or Timestamp value or NULL. You can also use the following datetime SQL functions: CURRENT_DATE, CURRENT_TIMESTAMP, TODAY, or NOW.
- For boolean columns, the literals FALSE, TRUE, NULL.
- For numeric columns, any valid number or NULL.

starting_value

is the starting value for the Identity column. The default start value is 0.

[NOT]NULL

is used to specify whether NULL values are allowed or not allowed in a column. If NOT NULL is specified, all rows in the table must have a column value. If NULL is specified or if neither NULL or NOT NULL is specified, NULL values are allowed in the column.

EXT_ID

is used to specify that the column is an external ID column.

PRIMARY KEY

can only be specified when the data type of the column is ID. ID columns are always the primary key column for Salesforce.

START WITH

specifies the sequence of numbers generated for the Identity column. It can only be used when the data type of the column definition is AUTONUMBER.

Example A

Assuming the current schema is SFORCE, the remote table `Test` is created in the SFORCE schema. The `id` column has a starting value of 1000.

```
CREATE TABLE Test (id AUTONUMBER START WITH 1000, Name TEXT(30))
```

Example B

The table name is qualified with a schema name that is not the current schema, creating the `Test` table in the SFORCE schema. The table is created with the following columns: `id`, `Name`, and `Status`. The `Status` column contains a default value of `ACTIVE`.

```
CREATE TABLE SFORCE.Test (id NUMBER(9, 0), Name TEXT(30), Status TEXT(10) DEFAULT 'ACTIVE')
```

Example C

Assuming the current schema is SFORCE, the remote table `dept` is created with the `name` and `deptId` columns. The `deptId` column can be used as an external ID column.

```
CREATE TABLE dept (name TEXT(30), deptId NUMBER(9, 0) EXT_ID)
```

Constraint Definition for Remote Tables

Purpose

Defines the syntax to define a constraint for a remote table.

Syntax

```
[CONSTRAINT [constraint_name]
  {foreign_key_constraint}]
```

where:

constraint_name

Is ignored. The driver uses the Salesforce relationship naming convention to generate the constraint name.

foreign_key_constraint

Defines a link between related tables. See [Foreign Key Clause](#) on page 1166 for syntax.

A column defined as a foreign key in one table references a primary key in the related table. Only values that are valid in the primary key are valid in the foreign key. The following example is valid because the foreign key values of the dept id column in the EMP table match those of the id column in the referenced table DEPT:

Referenced Table		Main Table		
DEPT		EMP		
		(Foreign Key)		
id	name	id	name	dept id
1	Dev	1	Mark	1
2	Finance	1	Jim	3
3	Sales	1	Mike	2

The following example, however, is not valid. The value 4 in the dept id column does not match any value in the referenced id column of the DEPT table.

Referenced Table		Main Table		
DEPT		EMP		
		(Foreign Key)		

Referenced Table		Main Table		
id	name	id	name	dept id
1	Dev	1	Mark	1
2	Finance	1	Jim	3
3	Sales	1	Mike	4

Foreign Key Clause

Purpose

Defines the syntax to specify a foreign key for a constraint.

Syntax

```
FOREIGN KEY (fcolumn_name) REFERENCES ref_table (pcolumn_name)
```

where:

fcolumn_name

specifies the foreign key column to which the constraint is applied. The data type of this column must be the same as the data type of the column it references.

ref_table

specifies the table to which the foreign key refers.

pcolumn_name

specifies the primary key column in the referenced table. For Salesforce, the primary key column is always the `rowId` column.

Example

Assuming the current schema is SFORCE, the remote table `emp` is created with the `name`, `empId`, and `deptId` columns. The table contains a foreign key constraint on the `deptId` column, referencing the `rowId` in the `dept` table created in [Example C](#) on page 1164. For the operation to succeed, the data type of the `deptId` column must be the same as that of the `rowId` column.

```
CREATE TABLE emp (name TEXT(30), empId NUMBER(9, 0) EXT_ID, deptId TEXT(18),  
FOREIGN KEY(deptId) REFERENCES dept(rowId))
```

Creating a Local Table

Syntax

```
CREATE [{MEMORY | DISK | [GLOBAL] {TEMPORARY | TEMP}}]
TABLE table_name (column_definition [, ...]
[, constraint_definition...]) [ON COMMIT {DELETE | PRESERVE} ROWS
```

where:

MEMORY

creates the new table in memory. The data for a memory table is held entirely in memory for the duration of the database session. When the database is closed, the data for the memory table is persisted to disk.

DISK

creates the new table on disk. A disk table caches a portion of its data in memory and the remaining data on disk.

TEMPORARY and TEMP

are equivalent and create the new table as a global temporary table. The GLOBAL qualifier is optional. The definition of a global temporary table is visible to all connections. The data written to a global temporary table is visible only to the connection used to write the data.

Note: If MEMORY, DISK, or TEMPORARY/TEMP is not specified, the new table is created in memory.

table_name

specifies the name of the new table.

column_definition

specifies the definition of a column in the new table. See [Column Definition for Local Tables](#) on page 1168 for a complete explanation.

constraint_definition

specifies constraints on the columns of the new table. See [Constraint Definition for Local Tables](#) on page 1169 for a complete explanation.

ON COMMIT PRESERVE ROWS

preserves row values in a temporary table while the connection is open; this is the default action.

ON COMMIT DELETE ROWS

empties row values on each commit or rollback.

Column Definition for Local Tables

Purpose

Use the following syntax to define a column for local tables.

Syntax

```
column_name Datatype [(precision[,scale])]
[{DEFAULT default_value | GENERATED BY DEFAULT AS IDENTITY
(START WITH n[, INCREMENT BY m)]}] | [[NOT] NULL]
[IDENTITY] [PRIMARY KEY]
```

where:

column_name

is the name to be assigned to the column.

Datatype

is the data type of the column to be created. See Data Types in "The Salesforce Driver" chapter of the *DataDirect Connect Series for ODBC User's Guide* for a list of supported Salesforce data types. You cannot specify ANYTYPE, BINARY, COMBOBOX, or TIME data types in the column definition of Create Table statements.

precision

is the number characters for CHAR and VARCHAR columns, the number of bytes for BINARY and VARBINARY columns, and the total number of digits for DECIMAL columns.

scale

is the number of digits to the right of the decimal point for DECIMAL columns and the number of seconds for DATETIME columns.

default_value

is the default value to be assigned to the column. The following default values are allowed in column definitions for local tables:

- For character columns, a single-quoted string or NULL. The only SQL function that can be used is CURRENT_USER.
- For datetime columns, a single-quoted Date, Time, or Timestamp value or NULL. You can also use the following datetime SQL functions: CURRENT_DATE, CURRENT_TIME, CURRENT_TIMESTAMP, TODAY, or NOW.
- For boolean columns, the literals FALSE, TRUE, NULL.
- For numeric columns, any valid number or NULL.
- For binary columns, any valid hexadecimal string or NULL.

IDENTITY | GENERATED BY DEFAULT AS IDENTITY

defines an auto-increment column. Either clause can be specified only on INTEGER or BIGINT columns. Identity columns are considered primary key columns, so a table can have only one Identity column.

The `GENERATED BY DEFAULT AS IDENTITY` clause is the standard SQL syntax for specifying an Identity column.

The `IDENTITY` operator is equivalent to `GENERATED BY DEFAULT AS IDENTITY` without the optional `START WITH` clause.

```
START WITH n[, INCREMENT BY m]
```

specifies the sequence of numbers generated for the Identity column. *n* and *m* are the starting and incrementing values, respectively, for an Identity column. The default start value is 0 and the default increment value is 1.

Example A

Assuming the current schema is PUBLIC, a local table is created. `id` is an identity column with a starting value of 0 and an increment value of 1 because no Start With and Increment By clauses are specified.

```
CREATE TABLE Test (id INTEGER GENERATED BY DEFAULT AS IDENTITY, name VARCHAR(30))
```

This example is equivalent to the previous example.

```
CREATE TABLE Test (id INTEGER IDENTITY, name VARCHAR(30))
```

Example B

Assuming the current schema is PUBLIC, a local table is created. `id` is an identity column with a starting value of 2 and an increment of 2.

```
CREATE TABLE Test (id INTEGER GENERATED BY DEFAULT AS IDENTITY (START WITH 2, INCREMENT BY 2), name VARCHAR(30))
```

Constraint Definition for Local Tables

Purpose

Defines the syntax to define a constraint for a local table.

Syntax

```
[CONSTRAINT [constraint_name]
 {unique_constraint |
 primary_key_constraint |
 foreign_key_constraint}]
```

where:

constraint_name

specifies a name for the constraint.

unique_constraint

specifies a constraint on a single column in the table. See [Unique Clause](#) for syntax.

Values in the constrained column cannot be repeated, except in the case of null values. For example:

```
ColA
```

```
1
```

2
 NULL
 4
 5
 NULL

A single table can have multiple columns with unique constraints.

primary_key_constraint

specifies a constraint on one or more columns in the table. See [Primary Key Clause](#) for syntax.

Values in a single column primary key column must be unique. Values across multiple constrained columns cannot be repeated, but values within a column can be repeated. Null values are not allowed. For example:

Col A	Col B
2	1
3	1
4	2
5	2
6	2

Only one primary key constraint is allowed in the table.

foreign_key_constraint

defines a link between related tables. See [Foreign Key Clause](#) on page 1166 for syntax.

A column defined as a foreign key in one table references a primary key in the related table. Only values that are valid in the primary key are valid in the foreign key. The following example is valid because the foreign key values of the dept id column in the EMP table match those of the id column in the referenced table DEPT:

Referenced Table	Main Table
DEPT	EMP
	(Foreign Key)

Referenced Table		Main Table		
id	name	id	name	dept id
1	Dev	1	Mark	1
2	Finance	1	Jim	3
3	Sales	1	Mike	2

The following example, however, is not valid. The value 4 in the dept id column does not match any value in the referenced id column of the DEPT table.

Referenced Table		Main Table		
DEPT		EMP		
id	name	id	name	dept id
1	Dev	1	Mark	1
2	Finance	1	Jim	3
3	Sales	1	Mike	4

(Foreign Key)

Unique Clause

`UNIQUE (column_name [,column_name...])`

where:

column_name

specifies the column to which the constraint is applied. Multiple column names must be separated by commas.

Primary Key Clause

`PRIMARY KEY (column_name [,column_name...])`

where:

column_name

specifies the primary key column to which the constraint is applied. Multiple column names must be separated by commas.

Foreign Key Clause

```
FOREIGN KEY (fcolumn_name [,fcolumn_name...])  
REFERENCES ref_table (pcolumn_name [,pcolumn_name...])  
[ON {DELETE | UPDATE}  
{CASCADE | SET DEFAULT | SET NULL}]
```

fcolumn_name

specifies the foreign key column to which the constraint is applied. Multiple column names must be separated by commas.

ref_table

specifies the table to which a foreign key refers.

pcolumn_name

specifies the primary key column or columns referenced in the referenced table. Multiple column names must be separated by commas.

ON DELETE

is a clause that defines the operation performed when a row in the table referenced by a foreign key constraint is deleted. One of the following operators must be specified in the On Delete clause:

where:

- **CASCADE** specifies that all rows in the foreign key table that reference the deleted row in the primary key table are also deleted.
- **SET DEFAULT** specifies that the value of the foreign key column is set to the column default value for all rows in the foreign key table that reference the deleted row in the primary key table.
- **SET NULL** specifies that the value of the foreign key column is set to NULL for all rows in the foreign key table that reference the deleted row in the primary key table.

ON UPDATE

is a clause that defines the operation performed when the primary key of a row in the table referenced by a foreign key constraint is updated. One of the following operators must be specified in the On Update clause:

- **CASCADE** specifies that the value of the foreign key column for all rows in the foreign key table that reference the row in the primary key table that had the primary key updated are updated with the new primary key value.
- **SET DEFAULT** specifies that the value of the foreign key column is set to the column default value for all rows in the foreign key table that reference the row that had the primary key updated in the primary key table.
- **SET NULL** specifies that the value of the foreign key column is set to NULL for all rows in the foreign key table that reference the row that had the primary key updated in the primary key table.

Notes

- You must specify at least one constraint.
- Both the On Delete and On Update clauses can be used in a single foreign key definition.

Example

Assuming the current schema is PUBLIC, the `emp` table is created with the `name`, `empId`, and `deptId` columns. The table contains a foreign key constraint on the `deptId` column that references the `id` column in the `dept` table. In addition, it sets the value of any rows in the `deptId` column to NULL that point to a deleted row in the referenced `dept` table.

```
CREATE TABLE emp (name VARCHAR(30), empId INTEGER, deptId INTEGER, FOREIGN
KEY(deptId) REFERENCES dept(id)) ON DELETE SET NULL)
```

Create View

Purpose

The Create View statement creates a new view. A view is analogous to a named query. The view's query can refer to any combination of remote and local tables as well as other views. Views are read-only; they cannot be updated.

Syntax

```
CREATE VIEW view_name[(view_column,...)] AS
SELECT ... FROM ... [WHERE Expression]
  [ORDER BY order_expression [, ...]]
  [LIMIT limit [OFFSET offset]];
```

where:

view_name

specifies the name of the view.

view_column

specifies the column associated with the view. Multiple column names must be separated by commas.

The other commands used for Create View are the same as those used for Select (see [Select](#) on page 1181).

Notes

- A view can be thought of as a virtual table. A Select statement is stored in the database; however, the data accessible through a view is not stored in the database. The result set of the Select statement forms the virtual table returned by the view. You can use this virtual table by referring to the view name in SQL statements the same way you refer to a table. A view is used to perform any or all of these functions:
 - Restrict a user to specific rows in a table.
 - Restrict a user to specific columns.
 - Join columns from multiple tables so that they function like a single table.
 - Aggregate information instead of supplying details. For example, the sum of a column, or the maximum or minimum value from a column can be presented.
- Views are created by defining the Select statement that retrieves the data to be presented by the view.
- The Select statement in a View definition must return columns with distinct names. If the names of two columns in the Select statement are the same, use a column alias to distinguish between them. Alternatively, you can define a list of new columns for a view.

Example A

This example creates a view named `myOpportunities` that selects data from three database tables to present a virtual table of data.

```
CREATE VIEW myOpportunities AS
SELECT a.name AS AccountName,
       o.name AS OpportunityName,
       o.amount AS Amount,
       o.description AS Description
FROM Opportunity o INNER JOIN Account a
    ON o.AccountId = a.id
    INNER JOIN User u
    ON o.OwnerId = u.id
WHERE u.name = 'MyName'
      AND o.isClosed = 'false'
ORDER BY Amount desc
```

You can then refer to the `myOpportunities` view in statements just as you would refer to a table. For example:

```
SELECT * FROM myOpportunities;
```

Example B

The `myOpportunities` view contains a detailed description for each opportunity, which may not be needed when only a summary is required. A view can be built that selects only specific `myOpportunities` columns as shown in the following example:

```
CREATE VIEW myOpps_NoDesc as
SELECT AccountName,
       OpportunityName,
       Amount
FROM myOpportunities
```

The view selects the name column from both the opportunity and account tables. These columns are assigned the alias `OpportunityName` and `AccountName`, respectively.

Delete

Purpose

The Delete statement is used to delete rows from a table.

Syntax

```
DELETE FROM table_name [WHERE search_condition]
```

where:

table_name

specifies the name of the table from which you want to delete rows.

search_condition

is an expression that identifies which rows to delete from the table.

Notes

- The Where clause determines which rows are to be deleted. Without a Where clause, all rows of the table are deleted, but the table is left intact. See [Where Clause](#) on page 1186 for information about the syntax of Where clauses. Where clauses can contain subqueries.

Example A

This example shows a Delete statement on the emp table.

```
DELETE FROM emp WHERE emp_id = 'E10001'
```

Each Delete statement removes every record that meets the conditions in the Where clause. In this case, every record having the employee ID E10001 is deleted. Because employee IDs are unique in the employee table, at most, one record is deleted.

Example B

This example shows using a subquery in a Delete clause.

```
DELETE FROM emp WHERE dept_id = (SELECT dept_id FROM dept WHERE dept_name = 'Marketing')
```

The records of all employees who belong to the department named Marketing are deleted.

Drop Cache (EXT)

Purpose

The Drop Cache statement drops the cache defined on a remote table. To drop a relational cache, the specified table must be the primary table of the relational cache. If a relational cache is specified, the cache for the primary table and all referenced caches are dropped.

Syntax

```
DROP CACHE ON {remote_table} [IF EXISTS]
```

where:

remote_table

is the name of the remote table cache to be dropped. The remote table name can be a two-part name: *schemaname.tablename*. When specifying a two-part name, the specified remote table must be mapped in the specified schema, and you must have the privilege to drop objects in the specified schema.

```
IF EXISTS
```

specifies that an error is not to be returned if a cache for the remote table or view does not exist.

Notes

- Caches on views are not supported.

Drop Index

Purpose

The Drop Index statement drops an index for a local table.

Syntax

```
DROP INDEX index_name [IF EXISTS]
```

where:

index_name

specifies an existing index.

IF EXISTS

specifies that an error is not to be returned if the index does not exist. The Drop Index command generates an error if an index that is associated with a UNIQUE or FOREIGN KEY constraint is specified.

Notes

- Indexes on a remote table cannot be dropped. Only indexes on local tables can be created, altered, and dropped.

Drop Sequence

Purpose

The Drop Sequence statement drops a sequence for local tables.

Syntax

```
DROP SEQUENCE sequence_name [IF EXISTS] [RESTRICT|CASCADE]
```

where:

sequence_name

specifies the name of a sequence to drop.

IF EXISTS

specifies that an error is not to be returned if the sequence does not exist.

RESTRICT

is in effect by default, meaning that the drop fails if any view refers to the sequence.

CASCADE

silently drops all dependent database objects.

Drop Table

Purpose

The Drop Table statement drops (removes) a remote or local table, its data, and its indexes. A remote table is a Salesforce object and is exposed in the SFORCE schema. Dropping a table in the SFORCE schema drops a remote table. A local table is maintained by the driver and is local to the machine on which the driver is running. A local table is exposed in the PUBLIC schema. Dropping a table in the PUBLIC schema drops a local table.

Syntax

```
DROP TABLE table_name [IF EXISTS] [RESTRICT | CASCADE]
```

where:

table_name

specifies the name of an existing table to drop.

IF EXISTS

specifies that an error is not to be returned if the table does not exist.

RESTRICT

is in effect by default, meaning that the drop fails if any tables or views reference this table.

CASCADE

specifies that the drop extends to linked objects. If the specified table is a local table, it drops all dependent views and any foreign key constraints that link this table to other tables. If the specified table is a remote table, any tables that reference the specified table are dropped also.

Drop View

Purpose

The Drop View statement drops a view.

Syntax

```
DROP VIEW view_name [IF EXISTS] [RESTRICT | CASCADE]
```

where:

view_name

specifies the name of a view.

IF EXISTS

specifies that an error is not to be returned if the view does not exist.

RESTRICT

is in effect by default, meaning that the drop fails if any other view refers to this view.

CASCADE

silently drops all dependent views.

Explain Plan

Purpose

The Explain Plan statement can be used with any query to retrieve a detailed list of the elements in the execution plan. Explain Plan generates a result set with a single column named `OPERATION`. The individual elements that comprise the plan are returned as rows in the result set.

Syntax

```
EXPLAIN PLAN FOR {SELECT ... | DELETE ... | INSERT ... | UPDATE ...}
```

The returned list of elements includes the indexes used for performing the query and can be used to optimize the query.

Insert

Purpose

The Insert statement is used to add new rows to a table. You can specify either of the following options:

- List of values to be inserted as a new row
- Select statement that copies data from another table to be inserted as a set of new rows

Syntax

```
INSERT INTO table_name [(column_name[,column_name]...)] {VALUES (expression  
[,expression]...) | select_statement}
```

table_name

is the name of the table in which you want to insert rows.

column_name

is optional and specifies an existing column. Multiple column names (a column list) must be separated by commas. A column list provides the name and order of the columns, the values of which are specified in the Values clause. If you omit a *column_name* or a column list, the value expressions must provide values for all columns defined in the table and must be in the same order that the columns are defined for the table. Table columns that do not appear in the column list are populated with the default value, or with NULL if no default value is specified. See [Specifying an External ID Column](#) on page 1179 for more information.

expression

is the list of expressions that provides the values for the columns of the new record. Typically, the expressions are constant values for the columns. Character string values must be enclosed in single quotation marks ('). See [Literals](#) on page 1193 for more information.

select_statement

is a query that returns values for each *column_name* value specified in the column list. Using a Select statement instead of a list of value expressions lets you select a set of rows from one table and insert it into another table using a single Insert statement. The Select statement is evaluated before any values are inserted. This query cannot be made on the table into which values are inserted. See [Select](#) on page 1181 for information about Select statements.

Specifying an External ID Column

To specify an external ID column to look up the value of a foreign key column, use the following syntax:

```
column_name EXT_ID [schema_name.table_name.] ext_id_column
```

where:

EXT_ID

is used to specify that the column specified by *ext_id_column* is used to look up the rowid to be inserted into the column specified by *column_name*.

schema_name

is the name of the schema of the table that contains the foreign key column being specified as the external ID column.

table_name

is the name of the table that contains the foreign key column being specified as the external ID column.

ext_id_column

is the external ID column.

Example A

The following example uses a list of expressions to insert records. Each Insert statement adds one record to the database table. In this case, one record is added to the table `emp`. Values are specified for five columns. The remaining columns in the table are assigned the default value or NULL if no default value is specified.

```
INSERT INTO emp (last_name,
                 first_name,
                 emp_id,
                 salary,
                 hire_date)
VALUES ('Smith', 'John', 'E22345', 27500, {1999-04-06})
```

Example B

The following example uses a Select statement to insert records. The number of columns in the result of the Select statement must match exactly the number of columns in the table if no column list is specified, or it must match the number of column names specified in the column list. A new entry is created in the table for every row of the Select result.

```
INSERT INTO emp1 (first_name,
                 last_name,
                 emp_id,
                 dept,
                 salary)
SELECT first_name, last_name, emp_id, dept, salary FROM emp
WHERE dept = 'D050'
```

Example C

The following example uses a list of expressions to insert records and specifies an external ID column (a foreign key column) named `accountId` that references a table that has an external ID column named `AccountNum`.

```
INSERT INTO emp (last_name,
                first_name,
                emp_id,
                salary,
                hire_date,
                accountId EXT_ID AccountNum)
VALUES ('Smith', 'John', 'E22345', 27500, {1999-04-06}, 0001)
```

Refresh Cache (EXT)

Purpose

The Refresh Cache statement forces the data in the cache for the specified remote table to be refreshed.

Syntax

```
REFRESH CACHE ON {remote_table | ALL} [CLEAN]
```

where:

remote_table

is the name of the remote table cache to be refreshed. The remote table name can be a two-part name: *schemaname.tablename*. When specifying a two-part name, the specified remote table must be mapped in the specified schema, and you must have the privilege to insert, update, and delete objects in the specified schema.

ALL

forces all caches to be refreshed.

CLEAN

is optional and discards the data in the cache for the specified table or view, or all cache data if ALL is specified, and repopulates the cache with the data in the remote table or view.

Notes

- Caches on views are not supported.

Refresh Schema (EXT)

Purpose

The Refresh Schema statement updates the remote object mapping and other information contained in a remote schema.

Syntax

```
REFRESH SCHEMA schema_name
```

where:

schema_name

is the name of the schema to be refreshed.

Select

Purpose

Use the Select statement to fetch results from one or more tables. The Select statement can operate on local and remote tables in any combination.

Syntax

```
SELECT select_clause from_clause
[where_clause]
[groupby_clause]
[having_clause]
[ {UNION [ALL | DISTINCT] |
  {MINUS [DISTINCT] | EXCEPT [DISTINCT]} |
  INTERSECT [DISTINCT]} select_statement ]
[orderby_clause]
[limit_clause]
```

where:

select_clause

specifies the columns from which results are to be returned by the query. See [Select Clause](#) on page 1182 for a complete explanation.

from_clause

specifies one or more tables on which the other clauses in the query operate. See [From Clause](#) on page 1184 for a complete explanation.

where_clause

is optional and restricts the results that are returned by the query. See [Where Clause](#) on page 1186 for a complete explanation.

groupby_clause

is optional and allows query results to be aggregated in terms of groups. See [Group By Clause](#) on page 1186 for a complete explanation.

having_clause

is optional and specifies conditions for groups of rows (for example, display only the departments that have salaries totaling more than \$200,000). See [Having Clause](#) on page 1187 for a complete explanation.

UNION

is an optional operator that combines the results of the left and right Select statements into a single result. See [Union Operator](#) on page 1187 for a complete explanation.

INTERSECT

is an optional operator that returns a single result by keeping any distinct values from the results of the left and right Select statements. See [Intersect Operator](#) on page 1188 for a complete explanation.

EXCEPT | MINUS

are synonymous optional operators that returns a single result by taking the results of the left Select statement and removing the results of the right Select statement. See [Except and Minus Operators](#) on page 1189 for a complete explanation.

orderby_clause

is optional and sorts the results that are returned by the query. See [Order By Clause](#) on page 1190 for a complete explanation.

limit_clause

is optional and places an upper bound on the number of rows returned in the result. See [Limit Clause](#) on page 1190 for a complete explanation.

Select Clause

Purpose

Use the Select clause to specify with a list of column expressions that identify columns of values that you want to retrieve or an asterisk (*) to retrieve the value of all columns.

Syntax

```
SELECT [{LIMIT offsetnumber | TOP number}] [ALL | DISTINCT] {* | column_expression
[[AS] column_alias] [,column_expression [[AS] column_alias], ...}
[INTO [DISK | TEMP] new_table]
```

where:

```
LIMIT offset number
```

creates the result set for the Select statement first and then discards the first number of rows specified by *offset* and returns the number of remaining rows specified by *number*. To not discard any of

the rows, specify 0 for *offset*, for example, `LIMIT 0 number`. To discard the first *offset* number of rows and return all the remaining rows, specify 0 for *number*, for example, `LIMIT offset 0`.

`TOP number`

is equivalent to `LIMIT 0 number`.

column_expression

can be simply a column name (for example, `last_name`). More complex expressions may include mathematical operations or string manipulation (for example, `salary * 1.05`). See [SQL Expressions](#) on page 1193 for details. *column_expression* can also include aggregate functions. See [Aggregate Functions](#) on page 1183 for details.

column_alias

can be used to give the column a descriptive name. For example, to assign the alias `department` to the column `dep`:

```
SELECT dep AS department FROM emp
```

`DISTINCT`

eliminates duplicate rows from the result of a query. This operator can precede the first column expression. For example:

```
SELECT DISTINCT dep FROM emp
```

`INTO`

copies the result set into *new_table*. `INTO DISK` creates the new table in cached memory. `INTO TEMP` creates a temporary table.

Notes

- Separate multiple column expressions with commas (for example, `SELECT last_name, first_name, hire_date`).
- Column names can be prefixed with the table name or table alias. For example, `SELECT emp.last_name` or `e.last_name`, where `e` is the alias for the table `emp`.
- `NULL` values are not treated as distinct from each other. The default behavior is that all result rows be returned, which can be made explicit with the keyword `ALL`.

Aggregate Functions

The result of a query can be the result of one or more aggregate functions. Aggregate functions return a single value from a set of rows. An aggregate can be used with a column name (for example, `AVG(salary)`) or in combination with a more complex column expression (for example, `AVG(salary * 1.07)`). The column expression can be preceded by the `DISTINCT` operator. The `DISTINCT` operator eliminates duplicate values from an aggregate expression.

The following table lists supported aggregate functions.

Table 114: Aggregate Functions

Aggregate	Returns

AVG	The average of the values in a numeric column expression. For example, <code>AVG(salary)</code> returns the average of all salary column values.
COUNT	The number of values in any column expression. For example, <code>COUNT(name)</code> returns the number of name values. When using <code>COUNT</code> with a column name, <code>COUNT</code> returns the number of non-NULL column values. A special example is <code>COUNT(*)</code> , which returns the number of rows in the set, including rows with NULL values.
MAX	The maximum value in any column expression. For example, <code>MAX(salary)</code> returns the maximum salary column value.
MIN	The minimum value in any column expression. For example, <code>MIN(salary)</code> returns the minimum salary column value.
SUM	The total of the values in a numeric column expression. For example, <code>SUM(salary)</code> returns the sum of all salary column values.

Except for `COUNT(*)`, all aggregate functions exclude NULL values. The returned value type for `COUNT` is `INTEGER` and for `MIN`, `MAX`, and `AVG` it is the same type as the column.

Example A

In the following example, only distinct last name values are counted. The default behavior is that all duplicate values be returned, which can be made explicit with `ALL`.

```
COUNT (DISTINCT last_name)
```

Example B

The following example uses the `COUNT`, `MAX`, and `AVG` aggregate functions:

```
SELECT
    COUNT(amount) AS numOpportunities,
    MAX(amount) AS maxAmount,
    AVG(amount) AS avgAmount
FROM opportunity o INNER JOIN user u
    ON o.ownerId = u.id
WHERE o.isClosed = 'false' AND
    u.name = 'MyName'
```

From Clause

Purpose

The From clause indicates the tables to be used in the Select statement.

Syntax

```
FROM table_name [table_alias] [, ...]
```

where:

table_name

is the name of a table or a subquery. Multiple tables define an implicit inner join among those tables. Multiple table names must be separated by a comma. For example:

```
SELECT * FROM emp, dep
```


Subqueries can be used instead of table names. Subqueries must be enclosed in parentheses. See [Subquery in a From Clause](#) on page 1185 for an example.

table_alias

is a name used to refer to a table in the rest of the Select statement. When you specify an alias for a table, you can prefix all column names of that table with the table alias.

Example

The following example specifies two table aliases, e for emp and d for dep:

```
SELECT e.name, d.deptName
FROM emp e, dep d
WHERE e.deptId = d.id
```

The equal sign (=) includes only matching rows in the results.

Join in a From Clause

Purpose

You can use a Join as a way to associate multiple tables within a Select statement. Joins may be either explicit or implicit. For example, the following is the example from the previous section restated as an explicit inner join:

```
SELECT e.name, d.deptName FROM emp e INNER JOIN dep d ON e.deptId = d.id;
```

Syntax

```
FROM table_name {RIGHT OUTER | INNER | LEFT OUTER | CROSS} JOIN table.key ON search-condition
```

Example

In this example, two tables are joined using LEFT OUTER JOIN. T1, the first table named includes nonmatching rows.

```
SELECT * FROM T1 LEFT OUTER JOIN T2 ON T1.key = T2.key
```

If you use a CROSS JOIN, no ON expression is allowed for the join.

Subquery in a From Clause

Subqueries can be used in the From clause in place of table references (*table_name*).

Example

```
SELECT * FROM (SELECT * FROM emp WHERE sal > 10000) new_emp, dept WHERE
new_emp.deptno = dept.deptno
```

See also

For more information about subqueries, see [Subqueries](#) on page 1205.

Where Clause

Purpose

The Where clause specifies the conditions that rows must meet to be retrieved.

Grammar

```
WHERE expr1 rel_operator expr2
```

where:

expr1

is either a column name, literal, or expression.

expr2

is either a column name, literal, expression, or subquery. Subqueries must be enclosed in parentheses.

rel_operator

is the relational operator that links the two expressions.

Example

The following Select statement retrieves the first and last names of employees that make at least \$20,000.

```
SELECT last_name, first_name FROM emp WHERE salary >= 20000
```

See also

See [Subqueries](#) on page 1205 for complete information about subqueries.

See [SQL Expressions](#) on page 1193 for details about expressions.

Group By Clause

Purpose

The Group By clause specifies the names of one or more columns by which the returned values are grouped. This clause is used to return a set of aggregate values.

Grammar

```
GROUP BY column_expression [,...]
```

where:

column_expression

is either a column name or a SQL expression. Multiple values must be separated by a comma. If *column_expression* is a column name, it must match one of the column names specified in the Select clause. Also, the Group By clause must include all non-aggregate columns specified in the Select list.

Example

The following example totals the salaries in each department:

```
SELECT dept_id, sum(salary) FROM emp GROUP BY dept_id
```

This statement returns one row for each distinct department ID. Each row contains the department ID and the sum of the salaries of the employees in the department.

See also

[SQL Expressions](#) on page 1193

Having Clause

Purpose

The Having clause specifies conditions for groups of rows (for example, display only the departments that have salaries totaling more than \$200,000). This clause is valid only if you have already defined a Group By clause.

Grammar

```
HAVING expr1 rel_operator expr2
```

where:

```
expr1 | expr2
```

can be column names, constant values, or expressions. These expressions do not have to match a column expression in the Select clause. See [SQL Expressions](#) on page 1193 for details regarding SQL expressions.

```
rel_operator
```

is the relational operator that links the two expressions.

Example

The following example returns only the departments that have salaries totaling more than \$200,000:

```
SELECT dept_id, sum(salary) FROM emp GROUP BY dept_id HAVING sum(salary) > 200000
```

Union Operator

Purpose

The Union operator combines the results of two Select statements into a single result. The single result is all the returned rows from both Select statements. By default, duplicate rows are not returned. To return duplicate rows, use the All keyword (`UNION ALL`).

Grammar

```
select_statement  
UNION [ALL | DISTINCT] | {MINUS [DISTINCT] | EXCEPT [DISTINCT]} | INTERSECT  
[DISTINCT]select_statement
```

Notes

- When using the Union operator, the Select lists for each Select statement must have the same number of column expressions with the same data types and must be specified in the same order.

Example A

The following example has the same number of column expressions, and each column expression, in order, has the same data type.

```
SELECT last_name, salary, hire_date FROM emp
UNION
SELECT name, pay, birth_date FROM person
```

Example B

The following example is *not* valid because the data types of the column expressions are different (`salary FROM emp` has a different data type than `last_name FROM raises`). This example does have the same number of column expressions in each Select statement but the expressions are not in the same order by data type.

```
SELECT last_name, salary FROM emp
UNION
SELECT salary, last_name FROM raises
```

Intersect Operator

Purpose

Intersect operator returns a single result set. The result set contains rows that are returned by both Select statements. Duplicates are returned unless the Distinct operator is added.

Syntax

```
select_statement
INTERSECT [DISTINCT]
select_statement
```

where:

DISTINCT

eliminates duplicate rows from the results.

Notes

- When using the Intersect operator, the Select lists for each Select statement must have the same number of column expressions with the same data types and must be specified in the same order.

Example A

The following example has the same number of column expressions, and each column expression, in order, has the same data type.

```
SELECT last_name, salary, hire_date FROM emp
INTERSECT [DISTINCT]
SELECT name, pay, birth_date FROM person
```

Example B

The following example is *not* valid because the data types of the column expressions are different (`salary FROM emp` has a different data type than `last_name FROM raises`). This example does have the same number of column expressions in each Select statement but the expressions are not in the same order by data type.

```
SELECT last_name, salary FROM emp
INTERSECT
SELECT salary, last_name FROM raises
```

Except and Minus Operators

Purpose

The Except and Minus are synonymous operators that return the rows from the left Select statement that are not included in the result of the right Select statement.

Syntax

```
select_statement
{EXCEPT [DISTINCT] | MINUS [DISTINCT]}
select_statement
```

where:

DISTINCT

eliminates duplicate rows from the results.

Notes

- When using one of these operators, the Select lists for each Select statement must have the same number of column expressions with the same data types and must be specified in the same order.

Example A

The following example has the same number of column expressions, and each column expression, in order, has the same data type.

```
SELECT last_name, salary, hire_date FROM emp
EXCEPT
SELECT name, pay, birth_date FROM person
```

Example B

The following example is *not* valid because the data types of the column expressions are different (`salary FROM emp` has a different data type than `last_name FROM raises`). This example does have the same number of column expressions in each Select statement but the expressions are not in the same order by data type.

```
SELECT last_name, salary FROM emp
EXCEPT
SELECT salary, last_name FROM raises
```

Order By Clause

Purpose

The Order By clause specifies how the rows are to be sorted.

Syntax

```
ORDER BY sort_expression [DESC | ASC] [...]
```

where:

sort_expression

is either the name of a column, a column alias, a SQL expression, or the positioned number of the column or expression in the select list to use.

The default is to perform an ascending (ASC) sort.

Example

To sort by `last_name` and then by `first_name`, you could use either of the following Select statements:

```
SELECT emp_id, last_name, first_name FROM emp
ORDER BY last_name, first_name
```

or

```
SELECT emp_id, last_name, first_name FROM emp
ORDER BY 2,3
```

In the second example, `last_name` is the second item in the Select list, so `ORDER BY 2,3` sorts by `last_name` and then by `first_name`.

See also

See [SQL Expressions](#) on page 1193 for details regarding SQL expressions.

Limit Clause

Purpose

The Limit clause places an upper bound on the number of rows returned in the result.

Syntax

```
LIMIT number_of_rows [OFFSET offset_number]
```

where:

number_of_rows

specifies a maximum number of rows in the result. A negative number indicates no upper bound.

OFFSET

specifies how many rows to skip at the beginning of the result set. *offset_number* is the number of rows to skip.

Notes

- In a compound query, the Limit clause can appear only on the final Select statement. The limit is applied to the entire query, not to the individual Select statement to which it is attached.

Example

The following example returns a maximum of 20 rows.

```
SELECT last_name, first_name FROM emp WHERE salary > 20000 ORDER BY dept_idc LIMIT 20
```

Set Checkpoint Defrag

Purpose

The Set Checkpoint Defrag statement is used in conjunction with the [Checkpoint](#) on page 1154 statement. Set Checkpoint Defrag sets the threshold for triggering a Checkpoint Defrag.

Syntax

```
SET CHECKPOINT DEFRAG size
```

where:

size

specifies the threshold size.

Notes

- When a [Checkpoint](#) on page 1154 statement is performed, either as a result of the .log file reaching the limit set by [Set Logsize](#) on page 1191 or by the user issuing a Checkpoint statement, the amount of abandoned space in the database data (.data) file is checked. If it is larger than the value of *size*, a CHECKPOINT DEFRAG, which eliminates the abandoned space, is performed instead of CHECKPOINT.

Set Logsize

Purpose

The Set Logsize statement sets the maximum size to which the driver's embedded database log file can grow before a [Checkpoint](#) on page 1154 statement is performed.

Syntax

```
SET LOGSIZE size
```

where:

size

specifies the maximum size in MB of the .log file. The default is 200 MB. A value of 0 means no limit is imposed on the size of the log file.

Notes

- When the log file exceeds the specified size, the Checkpoint statement closes and then reopens the database files, resetting the .log file.

Update

Purpose

An Update statement changes the value of columns in selected rows of a table.

Syntax

```
UPDATE table_name SET column_name = expression  
[, column_name = expression] [WHERE conditions]
```

table_name

is the name of the table for which you want to update values.

column_name

is the name of a column, the value of which is to be changed. Multiple column values can be changed in a single statement.

expression

is the new value for the column. The expression can be a constant value or a subquery that returns a single value. Subqueries must be enclosed in parentheses.

Notes

- A Where clause can be used to restrict which rows are updated.

See also

See [Subqueries](#) on page 1205 for complete information about subqueries.

See [Where Clause](#) on page 1186 for details.

Example A

The following example changes every record that meets the conditions in the Where clause. In this case, the salary and exempt status are changed for all employees having the employee ID E10001. Because employee IDs are unique in the emp table, only one record is updated.

```
UPDATE emp SET salary=32000, exempt=1  
WHERE emp_id = 'E10001'
```

Example B

The following example uses a subquery. In this example, the salary is changed to the average salary in the company for the employee having employee ID E10001.

```
UPDATE emp SET salary = (SELECT avg(salary) FROM emp)  
WHERE emp_id = 'E10001'
```


SQL Expressions

An expression is a combination of one or more values, operators, and SQL functions that evaluate to a value. You can use expressions in the Where, Having, and Order By clauses of Select statements; and in the Set clauses of Update statements.

Expressions enable you to use mathematical operations as well as character string manipulation operators to form complex queries.

The Salesforce driver supports both unquoted and quoted identifiers. An unquoted identifier must start with an ASCII alpha character and can be followed by zero or more ASCII alphanumeric characters. Unquoted identifiers are converted to uppercase before being used.

Quoted identifiers must be enclosed in double quotation marks ("""). A quoted identifier can contain any Unicode character including the space character. The Salesforce driver recognizes the Unicode escape sequence \uxxxx as a Unicode character. You can specify a double quotation mark in a quoted identifier by escaping it with a double quotation mark.

The maximum length of both quoted and unquoted identifiers is 128 characters.

Valid expression elements are:

- Column names
- Literals
- Operators
- Functions

Column Names

The most common expression is a simple column name. You can combine a column name with other expression elements.

Literals

Literals are fixed data values. For example, in the expression `PRICE * 1.05`, the value 1.05 is a constant. Literals are classified into types, including the following:

- Binary
- Character string
- Date
- Floating point
- Integer
- Numeric
- Time
- Timestamp

The following table describes the literal format for supported SQL data types.

Table 115: Literal Syntax Examples

SQL Type	Literal Syntax	Example
BIGINT	<i>n</i> where <i>n</i> is any valid integer value in the range of the INTEGER data type	12 or -34 or 0
BOOLEAN	Min Value: 0 Max Value: 1	0 1
DATE	'yyyy-mm-dd'	'2010-05-21'
DATETIME	'yyyy-mm-dd hh:mm:ss.SSSSSS'	'2010-05-21 18:33:05.025'
DECIMAL	<i>n.f</i> where: <i>n</i> is the integral part <i>f</i> is the fractional part	0.25 3.1415 -7.48
DOUBLE	<i>n.fEx</i> where: <i>n</i> is the integral part <i>f</i> is the fractional part <i>x</i> is the exponent	1.2E0 or 2.5E40 or -3.45E2 or 5.67E-4
INTEGER	<i>n</i> where <i>n</i> is a valid integer value in the range of the INTEGER data type	12 or -34 or 0
LONGVARBINARY	'hex_value'	'000482ff'
LONGVARCHAR	'value'	'This is a string literal'
TIME	'hh:mm:ss'	'18:33:05'
VARCHAR	'value'	'This is a string literal'

Character String Literals

Text specifies a character string literal. A character string literal must be enclosed in single quotation marks. To represent one single quotation mark within a literal, you must enter two single quotation marks. When the data in the fields is returned to the client, trailing blanks are stripped.

A character string literal can have a maximum length of 32 KB, that is, (32*1024) bytes.

Example

```
'Hello' 'Jim''s friend is Joe'
```

Integer Literals

Integer literals are represented by a string of numbers that are not enclosed in quotation marks and do not contain decimal points.

Notes

- Integer constants must be whole numbers; they cannot contain decimals.
- Integer literals can start with sign characters (+/-).

Example

```
1994
```

```
-2
```

Numeric Literals

Unquoted numeric values are treated as numeric literals. If the unquoted numeric value contains a decimal point or exponent, it is treated as a real literal; otherwise, it is treated as an integer literal.

Example

```
+1894.1204
```

Binary Literals

Binary literals are represented with single quotation marks. The valid characters in a binary literal are 0-9, a-f, and A-F.

Example

```
'00af123d'
```

Date/Time Literals

Date and time literal values are:

- A Date literal is enclosed in single quotation marks (' '). The format is `yyyy-mm-dd`.
- A Time literal is enclosed in single quotation marks (' '). The format is `hh:mm:ss`.
- A Timestamp is enclosed in single quotation marks (' '). The format is `yyyy-mm-dd hh:mm:ss.SSSSSS`.

Operators

This section describes the operators that can be used in SQL expressions.

Unary Operator

A unary operator operates on only one operand.

operator operand

Binary Operator

A binary operator operates on two operands.

operand1 operator operand2

If an operator is given a null operand, the result is always null. The only operator that does not follow this rule is concatenation (||).

Arithmetic Operators

You can use an arithmetic operator in an expression to negate, add, subtract, multiply, and divide numeric values. The result of this operation is also a numeric value. The + and - operators are also supported in date/time fields to allow date arithmetic. The following table lists the supported arithmetic operators.

Table 116: Arithmetic Operators

Operator	Purpose	Example
+ -	Denotes a positive or negative expression. These are unary operators.	SELECT * FROM emp WHERE comm = -1
* /	Multiplies, divides. These are binary operators.	UPDATE emp SET sal = sal + sal * 0.10
+ -	Adds, subtracts. These are binary operators.	SELECT sal + comm FROM emp WHERE empno > 100

Concatenation Operator

The concatenation operator manipulates character strings. The following table lists the only supported concatenation operator.

Table 117: Concatenation Operator

Operator	Purpose	Example
	Concatenates character strings.	SELECT 'Name is' ename FROM emp

The result of concatenating two character strings is the data type VARCHAR.

Comparison Operators

Comparison operators compare one expression to another. The result of such a comparison can be TRUE, FALSE, or UNKNOWN (if one of the operands is NULL). The Salesforce driver considers the UNKNOWN result as FALSE.

The following table lists the supported comparison operators.

Table 118: Comparison Operators

Operator	Purpose	Example
=	Equality test.	SELECT * FROM emp WHERE sal = 1500
!<>	Inequality test.	SELECT * FROM emp WHERE sal != 1500
><	"Greater than" and "less than" tests.	SELECT * FROM emp WHERE sal > 1500 SELECT * FROM emp WHERE sal < 1500
>=<=	"Greater than or equal to" and "less than or equal to" tests.	SELECT * FROM emp WHERE sal >= 1500 SELECT * FROM emp WHERE sal <= 1500
ESCAPE clause in LIKE operator LIKE 'pattern string' ESCAPE 'c'	The Escape clause is supported in the LIKE predicate to indicate the escape character. Escape characters are used in the pattern string to indicate that any wildcard character that is after the escape character in the pattern string should be treated as a regular character. The default escape character is backslash (\)	SELECT * FROM emp WHERE ENAME LIKE 'J%_%' ESCAPE '\' This matches all records with names that start with letter 'J' and have the '_' character in them. SELECT * FROM emp WHERE ENAME LIKE 'JOE_JOHN' ESCAPE '\' This matches only records with name 'JOE_JOHN'.
[NOT] IN	"Equal to any member of" test.	SELECT * FROM emp WHERE job IN ('CLERK', 'ANALYST') SELECT * FROM emp WHERE sal IN (SELECT sal FROM emp WHERE deptno = 30)
[NOT] BETWEEN x AND y	"Greater than or equal to x" and "less than or equal to y."	SELECT * FROM emp WHERE sal BETWEEN 2000 AND 3000
EXISTS	Tests for existence of rows in a subquery.	SELECT empno, ename, deptno FROM emp e WHERE EXISTS (SELECT deptno FROM dept WHERE e.deptno = dept.deptno)
IS [NOT] NULL	Tests whether the value of the column or expression is NULL.	SELECT * FROM emp WHERE ename IS NOT NULL SELECT * FROM emp WHERE ename IS NULL

Logical Operators

A logical operator combines the results of two component conditions to produce a single result or to invert the result of a single condition. The following table lists the supported logical operators.

Table 119: Logical Operators

Operator	Purpose	Example
NOT	Returns TRUE if the following condition is FALSE. Returns FALSE if it is TRUE. If it is UNKNOWN, it remains UNKNOWN.	<pre>SELECT * FROM emp WHERE NOT (job IS NULL) SELECT * FROM emp WHERE NOT (sal BETWEEN 1000 AND 2000)</pre>
AND	Returns TRUE if both component conditions are TRUE. Returns FALSE if either is FALSE; otherwise, returns UNKNOWN.	<pre>SELECT * FROM emp WHERE job = 'CLERK' AND deptno = 10</pre>
OR	Returns TRUE if either component condition is TRUE. Returns FALSE if both are FALSE; otherwise, returns UNKNOWN.	<pre>SELECT * FROM emp WHERE job = 'CLERK' OR deptno = 10</pre>

Example

In the Where clause of the following Select statement, the AND logical operator is used to ensure that managers earning more than \$1000 a month are returned in the result:

```
SELECT * FROM emp WHERE jobtitle = manager AND sal > 1000
```

Operator Precedence

As expressions become more complex, the order in which the expressions are evaluated becomes important. The following table shows the order in which the operators are evaluated. The operators in the first line are evaluated first, then those in the second line, and so on. Operators in the same line are evaluated left to right in the expression. You can change the order of precedence by using parentheses. Enclosing expressions in parentheses forces them to be evaluated together.

Table 120: Operator Precedence

Precedence	Operator
1	+ (Positive), - (Negative)
2	*(Multiply), / (Division)
3	+ (Add), - (Subtract)
4	(Concatenate)
5	=, >, <, >=, <=, <>, != (Comparison operators)
6	NOT, IN, LIKE
7	AND
8	OR

Example A

The query in the following example returns employee records for which the department number is 1 or 2 and the salary is greater than \$1000:

```
SELECT * FROM emp WHERE (deptno = 1 OR deptno = 2) AND sal > 1000
```

Because parenthetical expressions are forced to be evaluated first, the OR operation takes precedence over AND.

Example B

In the following example, the query returns records for all the employees in department 1, but only employees whose salary is greater than \$1000 in department 2.

```
SELECT * FROM emp WHERE deptno = 1 OR deptno = 2 AND sal > 1000
```

The AND operator takes precedence over OR, so that the search condition in the example is equivalent to the expression `deptno = 1 OR (deptno = 2 AND sal > 1000)`.

Functions

The Salesforce driver supports a number of functions that you can use in expressions, as listed and described in the following tables.

Table 121: Numerical Functions Supported

Numerical Function	Description
ABS(d)	Returns the absolute value of a double value.
ACOS(d)	Returns the arc cosine of an angle.
ASIN(d)	Returns the arc sine of an angle.
ATAN(d)	Returns the arc tangent of an angle.
ATAN2(a,b)	Returns the tangent of a/b.
BITAND(a,b)	Returns a and b.
BITOR(a,b)	Returns a or b.
CEILING(d)	Returns the smallest integer that is not less than d.
COS(d)	Returns the cosine of an angle.
COT(d)	Returns the cotangent of an angle.
DEGREES(d)	Converts radians to degrees.
EXP(d)	Returns e (2.718... raised to the power of d).
FLOOR(d)	Returns the largest integer that is not greater than d.
LOG(d)	Returns the natural logarithm (base e).

Numerical Function	Description
LOG10(d)	Returns the logarithm (base 10).
MOD(a,b)	Returns a modulo b.
PI()	Returns pi (3.1415...).
POWER(a,b)	Returns a raised to the power of b.
RADIANS(d)	Converts degrees to radians.
RAND()	Returns a random number x bigger or equal to 0.0 and smaller than 1.0.
ROUND(a,b)	Rounds a to b digits after the decimal point.
ROUNDMAGIC(d)	Solves rounding problems such as 3.11-3.1-0.01.
SIGN(d)	Returns -1 if d is smaller than 0, 0 if d==0 and 1 if d is bigger than 0.
SIN(d)	Returns the sine of an angle.
SQRT(d)	Returns the square root.
TAN(A)	Returns the trigonometric tangent of an angle.
TRUNCATE(a,b)	Truncates a to b digits after the decimal point.

Table 122: String Functions Supported

String Function	Description
ASCII(s)	Returns the ASCII code of the leftmost character of s.
BIT_LENGTH(str)	Returns the length of the string in bits.
CHAR(c)	Returns a character that has the ASCII code c.
CHAR_LENGTH(str)	Returns the length of the string in characters.
CONCAT(str1,str2)	Returns the string that results from concatenating str1 + str2.
DIFFERENCE(s1,s2)	Returns an integer value from 0 to 4 that indicates the difference between the values returned by the SOUNDEX function for S1 and S2. A value of 4 indicates that S1 and S2 are the same, while a value of 0 indicates that the values have no similarity.
HEXTORAW(s1)	Returns the binary format of the string.
INSERT(s,start,len,s2)	Returns a string where len number of characters beginning at start has been replaced by s2.

String Function	Description
LCASE(s)	Converts s to lower case.
LEFT(s,count)	Returns the leftmost count of characters of s. If s requires double quoting, use SUBSTRING() instead.
LENGTH(s)	Returns the number of characters in s.
LOCATE(search,s,[start])	Returns the first index (1=left, 0=not found) where search is found in s, starting at start.
LTRIM(s)	Removes all leading blanks in s.
OCTET_LENGTH(str)	Returns the length of the string in bytes (twice the number of characters).
RAWTOHEX(s1)	Returns translated string.
REPEAT(s,count)	Returns s repeated count times.
REPLACE(s,replace,s2)	Returns s with all occurrences of replace replaced with s2.
RIGHT(s,count)	Returns the right-most count of characters of s.
RTRIM(s)	Removes all trailing spaces in s.
SOUNDEX(s)	Returns a 4-character code representing the sound of s.
SPACE(count)	Returns a string consisting of count spaces.
SUBSTR(s,start[,len])	Alias for substring.
SUBSTRING(s,start[,len])	Returns the substring starting at start (1=left) with length len.
UCASE(s)	Converts s to uppercase.
LOWER(s)	Converts s to lowercase.
UPPER(s)	Converts s to uppercase.

Table 123: Date/Time Functions Supported

Date/Time Function	Description
CURDATE()	Returns the current date.
CURTIME()	Returns the current time.

Date/Time Function	Description
DATEDIFF(string, datetime1, datetime2)	<p>Returns the count of units of time elapsed from datetime1 to datetime2. The string indicates the unit of time and can have the following values:</p> <ul style="list-style-type: none"> • 'ms'='millisecond' • 'ss'='second' • 'mi'='minute' • 'hh'='hour' • 'dd'='day' • 'mm'='month' • 'yy' = 'year' <p>Both the long and short form of the strings can be used.</p>
DAYNAME(date)	Returns the name of the day.
DAYOFMONTH(date)	Returns the day of the month (1-31).
DAYOFWEEK(date)	Returns the day of the week (1 means Sunday).
DAYOFYEAR(date)	Returns the day of the year (1-366).
HOUR(time)	Returns the hour (0-23).
MINUTE(time)	Returns the minute (0-59).
MONTH(date)	Returns the month (1-12).
MONTHNAME(date)	Returns the name of the month.
NOW()	Returns the current date and time as a timestamp, use CURRENT_TIMESTAMP instead.
QUARTER(date)	Returns the quarter (1-4).
SECOND(time)	Returns the second (0-59).
WEEK(date)	Returns the week of this year (1-53).
YEAR(date)	Returns the year.
CURRENT_DATE	Returns the current date.
CURRENT_TIME	Returns the current time.
CURRENT_TIMESTAMP	Returns the current timestamp.

Table 124: System/Connection Functions Supported

System/Connection Function	Description
DATABASE()	Returns the name of the database of this connection.
USER()	Returns the user name of this connection.
CURRENT_USER	SQL standard function, returns the user name of this connection.
CURSESSIONID()	Returns the ID of the session (connection) on which this function was invoked.
IDENTITY()	Returns the last identity value that was inserted by this connection.

Table 125: System Functions Supported

System Function	Description
IFNULL(<i>expr</i> , <i>value</i>)	If <i>expr</i> is NULL, then <i>value</i> is returned; otherwise the result of <i>expr</i> is returned. See COALESCE for evaluating multiple expressions.
CONVERT(<i>term</i> , <i>type</i>)	Converts <i>term</i> to another data type.
CAST(<i>term</i> AS <i>type</i>)	Converts <i>term</i> to another data type.
COALESCE(<i>expr1</i> , <i>expr2</i> , ...)	If <i>expr1</i> is not Null, then it is returned; otherwise, <i>expr2</i> is evaluated and, if not Null, it is returned, and so on. This is an ANSISQL standard system function.
NULLIF(<i>value1</i> , <i>value2</i>)	If <i>value1</i> equals <i>value2</i> , then Null is returned; otherwise, <i>value1</i> is returned.
CASE <i>value1</i> WHEN <i>value2</i> THEN <i>value3</i> [ELSE <i>value4</i>] END	When <i>value1</i> equals <i>value2</i> , then <i>value3</i> is returned; otherwise, <i>value4</i> or Null is returned in the absence of ELSE.
CASE WHEN <i>expr1</i> THEN <i>value1</i> [WHEN <i>expr2</i> THEN <i>value2</i>] [ELSE <i>value4</i>] END	When <i>expr1</i> is true, then <i>value1</i> is returned (optionally repeated for more cases); otherwise <i>value4</i> or Null is returned in the absence of ELSE.
EXTRACT ({YEAR MONTH DAY HOUR MINUTE SECOND} FROM <i>datetime_value</i>)	Any of the date and time terms can be extracted from <i>datetime_value</i> .
POSITION(<i>string_expression1</i> IN <i>string_expression2</i>)	If <i>string_expression1</i> is a sub-string of <i>string_expression2</i> , then the position of the sub-string, counting from one, is returned; otherwise, 0 is returned.

System Function	Description
SUBSTRING(<i>string_expression</i> FROM <i>numeric_expression1</i> [FOR <i>numeric_expression2</i>])	<i>string_expression</i> is returned from the <i>numeric_expression1</i> starting location. Optionally, <i>numeric_expression2</i> specifies the length of the substring.
TRIM([LEADING TRAILING BOTH]) FROM <i>string_expression</i>)	When returned, either the leading or trailing spaces, or both, are trimmed from <i>string_expression</i> .

Conditions

A condition specifies a combination of one or more expressions and logical operators that evaluates to either TRUE, FALSE, or UNKNOWN. You can use a condition in the Where clause of the Delete, Select, and Update statements; and in the Having clauses of Select statements. The following describes supported conditions.

Table 126: Conditions

Condition	Description
Simple comparison	Specifies a comparison with expressions or subquery results. = , !=, <>, < , >, <=, >=
Group comparison	Specifies a comparison with any or all members in a list or subquery. [= , !=, <>, < , >, <=, >=] [ANY, ALL, SOME]
Membership	Tests for membership in a list or subquery. [NOT] IN
Range	Tests for inclusion in a range. [NOT] BETWEEN
NULL	Tests for nulls. IS NULL, IS NOT NULL
EXISTS	Tests for existence of rows in a subquery. [NOT] EXISTS

Condition	Description
LIKE	Specifies a test involving pattern matching. [NOT] LIKE
Compound	Specifies a combination of other conditions. CONDITION [AND/OR] CONDITION

Subqueries

A query is an operation that retrieves data from one or more tables or views. In this reference, a top-level query is called a Select statement, and a query nested within a Select statement is called a subquery.

A subquery is a query expression that appears in the body of another expression such as a Select, an Update, or a Delete statement. In the following example, the second Select statement is a subquery:

```
SELECT * FROM emp WHERE deptno IN (SELECT deptno FROM dept)
```

IN Predicate

Purpose

The In predicate specifies a set of values against which to compare a result set. If the values are being compared against a subquery, only a single column result set is returned.

Syntax

```
value [NOT] IN (value1, value2,...)
```

OR

```
value [NOT] IN (subquery)
```

Example

```
SELECT * FROM emp WHERE deptno IN
(SELECT deptno FROM dept WHERE dname <> 'Sales')
```

EXISTS Predicate

Purpose

The Exists predicate is true only if the cardinality of the subquery is greater than 0; otherwise, it is false.

Syntax

```
EXISTS (subquery)
```

Example

```
SELECT empno, ename, deptno FROM emp e WHERE EXISTS
```

```
(SELECT deptno FROM dept WHERE e.deptno = dept.deptno)
```

UNIQUE Predicate

Purpose

The Unique predicate is used to determine whether duplicate rows exist in a virtual table (one returned from a subquery).

Syntax

```
UNIQUE (subquery)
```

Example

```
SELECT * FROM dept d WHERE UNIQUE  
(SELECT deptno FROM emp e WHERE e.deptno = d.deptno)
```

Correlated Subqueries

Purpose

A correlated subquery is a subquery that references a column from a table referred to in the parent statement. A correlated subquery is evaluated once for each row processed by the parent statement. The parent statement can be a Select, Update, or Delete statement.

A correlated subquery answers a multiple-part question in which the answer depends on the value in each row processed by the parent statement. For example, you can use a correlated subquery to determine which employees earn more than the average salaries for their departments. In this case, the correlated subquery specifically computes the average salary for each department.

Syntax

```
SELECT select_list  
  FROM table1 t_alias1  
  WHERE expr rel_operator  
    (SELECT column_list  
     FROM table2 t_alias2  
     WHERE t_alias1.columnrel_operatort_alias2.column)  
UPDATE table1 t_alias1  
  SET column =  
    (SELECT expr  
     FROM table2 t_alias2  
     WHERE t_alias1.column = t_alias2.column)  
DELETE FROM table1 t_alias1  
  WHERE column rel_operator  
    (SELECT expr  
     FROM table2 t_alias2  
     WHERE t_alias1.column = t_alias2.column)
```

Notes

- Correlated column names in correlated subqueries must be explicitly qualified with the table name of the parent.

Example A

The following statement returns data about employees whose salaries exceed their department average. This statement assigns an alias to `emp`, the table containing the salary information, and then uses the alias in a correlated subquery:

```
SELECT deptno, ename, sal FROM emp x WHERE sal >
  (SELECT AVG(sal) FROM emp WHERE x.deptno = deptno)
ORDER BY deptno
```

Example B

This is an example of a correlated subquery that returns row values:

```
SELECT * FROM dept "outer" WHERE 'manager' IN
  (SELECT managename FROM emp
  WHERE "outer".deptno = emp.deptno)
```

Example C

This is an example of finding the department number (`deptno`) with multiple employees:

```
SELECT * FROM dept main WHERE 1 <
  (SELECT COUNT(*) FROM emp WHERE deptno = main.deptno)
```

Example D

This is an example of correlating a table with itself:

```
SELECT deptno, ename, sal FROM emp x WHERE sal >
  (SELECT AVG(sal) FROM emp WHERE x.deptno = deptno)
```

