Progress DataDirect

Progress DataDirect for ODBC for Apache Hive Wire Protocol Driver

User's Guide

Release 8.0.1

Copyright

Table of Contents

1

Welcome to the Progress DataDirect for ODBC for Apache Hive Wire Protocol Driver

The Progress® DataDirect® for ODBC for Apache Hive™ Wire Protocol driver supports Apache Hive against the following distributions:

- Apache Hive
- Cloudera Data Platform (CDP)
- HPE Ezmeral Data Fabric

The driver is supported in the Windows, UNIX, and Linux environments. See "Support for multiple environments" for detailed information about the environments supported by this driver.

See "Driver file names for Windows" and "Driver file names for UNIX/Linux" for the file name of the driver.

The documentation for the driver 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.

For details, see the following topics:

What's new in this release?

Support and Certifications

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

- Release Notes: https://www.progress.com/odbc/release-history/
- DataDirect Product Compatibility Guide: https://docs.progress.com/bundle/datadirect-product-compatibility/resource/datadirect-product-compatibility.pdf

Changes Since 8.0.1 GA

- **Driver Enhancements**
  - The default version of the OpenSSL library has been upgraded to 3.5.6. As part of this upgrade, earlier version of the OpenSSL 3.0 library continues to be supported to provide the best protection for your data. The upgrade is available in the following OpenSSL library files:
    - Windows: `ivopenssl.dll` and `ddopenssl.dll`
    - Unix: `ivopenssl.so` and `ddopenssl.so`
  - The driver is now compiled with a Visual Studio 2022 compiler for the Windows platforms. As a result, you must have Microsoft Visual C/C++ runtime version 14.40.33810 or higher on your machine to run the driver.
  - The driver now supports describing Char and Varchar columns as either SQL_CHAR and SQL_VARCHAR types or SQL_WCHAR and SQL_WVARCHAR types in the metadata. You can use the new Use Unicode Char Types (UseUnicodeCharTypes) connection option to configure how the driver maps Char and Varchar columns. See Use Unicode Char Types on page 126 for details.
  - The default version of the OpenSSL library has been upgraded to 3.0. As part of this upgrade, earlier versions of the OpenSSL library are no longer supported to provide the best protection for your data. The upgrade is available in the following OpenSSL library files: *xx*`openssl30.dll` (for Windows) and *xx*`openssl30.so` [`.sl`] (for UNIX/Linux).

    The OpenSSL 3.0 library uses a set of shared libraries called providers to implement different types of cryptographic algorithms. The driver supports the following OpenSSL 3.0 providers: FIPS and default. See TLS/SSL server authentication on page 68 and TLS/SSL client authentication on page 73 for details.
  - The driver has been enhanced to support the Windows certificate store for TLS/SSL server authentication. See TLS/SSL server authentication on page 68 for details.
  - The driver has been enhanced to support TLS/SSL server authentication for the applications deployed in a serverless environment. The driver stores the TLS/SSL certificates in memory and lets applications use TLS/SSL server authentication without storing the truststore file on the disk. To use this enhancement, specify the content of the certificate in the refreshed Trust Store (`Truststore`) connection option or

the new SQL_COPT_INMEMORY_TRUSTSTORECERT pre-connection attribute. See Truststore on page 123 and Using SQL_COPT_INMEMORY_TRUSTSTORECERT on page 71 for details.

- A Password Encryption Tool, called ddencpwd, is now included with the product package. It encrypts passwords for secure handling in connection strings and odbc.ini files. At connection, the driver decrypts these passwords and passes them to the data source as required. See Password Encryption Tool (UNIX/Linux only) on page 59 for details.
- A Power BI connector is now included with the product package. You can use this connector to access your Hive data with Power BI. See Accessing data in Power BI (Windows only) on page 29 for details.
- The driver has been enhanced to include timestamp in the internal packet logs by default. If you want to disable the timestamp logging in packet logs, set PacketLoggingOptions=1. The internal packet logging is not enabled by default. To enable it, set EnablePacketLogging=1.
- The new AllowedOpenSSLVersions option allows you to determine which version of the OpenSSL library file the driver uses for data encryption.
- The driver has been enhanced to support ACID operations for Inserts, Updates, and Deletes on servers that are configured to use them. See Limitations on Apache Hive functionality on page 83 for more information on limitations and restrictions.
- The driver has been enhanced to support Apache ZooKeeper, which can be configured using the refreshed Host Name connection option and the new ZookeeperDiscovery and ZookeeperNamespace connection options. See Apache ZooKeeper on page 79, Host Name on page 106, Zookeeper Discovery on page 129, and Zookeeper Namespace on page 129 for details.

  Also, the driver supports Kerberos authentication for Apache ZooKeeper. See Configuring Apache ZooKeeper for Kerberos authentication on page 81 for details.

- **Changed Behavior**
  - The String Describe Type (StringDescribeType) connection option now supports describing String columns as SQL_LONGVARCHAR and SQL_VARCHAR as well, in addition to SQL_WLONGVARCHAR and SQL_WVARCHAR. To support this behavior, two new valid values have been added to the connection option: -1 (SQL_LONGVARCHAR) and 12 (SQL_VARCHAR). See String Describe Type on page 120 for details.
  - The Allowed OpenSSL Versions (AllowedOpenSSLVersions) connection option has been deprecated.
  - The product no longer includes version 1.1.1 of the OpenSSL library. The library will reach the end of its product life cycle in September 2023 and will not receive any security updates after that. Note that continuing to use the library after September 2023 can potentially expose you to security vulnerabilities.

    Note: As a result of this change, when installing a new version of the product, the installer program will automatically remove version 1.1.1 of the library from the install directory, which will impact all the DataDirect ODBC drivers installed on a machine. Therefore, if you are using multiple drivers, upgrade all your drivers to the latest version.

  - The product no longer includes version 1.0.2 of the OpenSSL library. The library has reached the end of its product life cycle and is not receiving security updates anymore. Note that continuing to use the library could potentially expose you to security vulnerabilities.

    Note: As a result of this change, when installing a new version of the driver, the installer program will automatically remove version 1.0.2 of the library from the install directory.

  - The crypto protocol versions prior to TLSv1 are no longer supported.

Changes for 8.0.1 GA

- **Driver Enhancements**
  - The driver has been enhanced to support HTTP mode, which allows you to access Apache Hive data stores using HTTP/HTTPS requests. HTTP mode can be configured using the new Transport Mode and HTTP Path connection options. See HTTP mode on page 61, Transport Mode on page 122, and HTTP Path on page 108 for details.
  - The driver has been enhanced to support session cookie based authentication for HTTP connections. Cookie based authentication can be configured using the new Enable Cookie Authentication and Cookie Name connection options. See Enable Cookie Authentication on page 102, and Cookie Name on page 96.
  - The driver has been enhanced to support HTTP connections to Apache Knox gateways. See Apache Knox on page 78 for details.
  - The new Array Insert Size connection option provides a workaround for memory and server issues that can sometimes occur when inserting a large number of rows that contain large values. See Array Insert Size on page 93 for details.
- **Changed Behavior**
  - The Array Size (ArraySize) connection option has been renamed Array Fetch Size (ArrayFetchSize). The ArraySize attribute will continue to be supported for this release, but will be deprecated in subsequent versions of the product. See Array Fetch Size on page 92 for details.

Changes for 8.0.0 GA

- **Driver Enhancements**
  - The driver has been enhanced to optimize the performance of fetches.
  - 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, such as SQL Server Linked Server. See Min Long Varchar Size on page 113 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, such as SQL Server Linked Server. See Varchar Threshold on page 128 for details.
  - The new Max String Size connection option allows you to determine the maximum size of columns of the String data type that the driver describes through result set descriptions and catalog functions. This option replaces the Max Varchar Size connection option. See Max String Size on page 112 for details.
  - The new Catalog Mode connection option allows you to determine whether the driver uses native catalog functions to retrieve information returned by the SQLTables, SQLColumns, and SQLStatistics catalog functions. In the default setting, the driver employs a balance of native functions and driver-discovered information for the optimal balance of performance and accuracy when retrieving catalog information. This option replaces the Use Native Catalog Functions option. See Catalog Mode on page 95 for details.
  - The driver includes a new Tableau data source file (Windows only) that provides improved functionality when accessing your data with Tableau. Refer to the Windows Quick Start for details.
  - Refer to the "Character encoding in the odbc.ini and odbcinst.ini files" in *Progress DataDirect for ODBC Drivers Reference* for details.
- **Changed Behavior**
  - The driver supports the HiveServer2 protocol and higher, and as a result:
    - Support for the HiveServer1 protocol has been deprecated

    - The Wire Protocol Version connection option has been deprecated

- The Use Native Catalog Functions connection option has been replaced by the new Catalog Mode connection option. The UseNativeCatalogFunctions attribute will continue to be supported for this release, but will be deprecated in subsequent versions of the product. See Catalog Mode on page 95 for details.
- The Max Varchar Size connection option has been replaced by the new Max String Size connection option. The MaxVarcharSize attribute will continue to be supported for this release, but will be deprecated in subsequent versions of the product. See Max String Size on page 112 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 94 for details.

  Note: Specifying the legacy setting for enabling Kerberos (`AuthenticationMethod=1`) will return an error message at connection.

- The default value for Crypto Protocol Version has been updated to `TLSv1.2,TLSv1.1,TLSv1`. This change improves the security of the driver by employing only the most secure cryptographic protocols as the default behavior. See Crypto Protocol Version on page 97 for details.
- The valid and default values for the String Describe Type connection have been updated:
  - Valid values: `-10` (SQL_WLONGVARCHAR) | `-9` (SQL_WVARCHAR)
  - Default value: `-9` (SQL_WVARCHAR)

  See String Describe Type on page 120 for details.

Driver requirements

Data source and platform requirements

For the latest support information, visit the DataDirect Product Compatibility Guide: https://docs.progress.com/bundle/datadirect-product-compatibility/resource/datadirect-product-compatibility.pdf.

Windows requirements for 32-bit drivers

- All required network software that is supplied by your database system vendors must be 32-bit compliant.
- You must have Microsoft Visual C/C++ runtime version 14.40.33810 or higher.
- You must have ODBC header files to compile your application. For example, Microsoft Visual Studio includes these files.

Windows requirements for 64-bit drivers

- All required network software that is supplied by your database system vendors must be 64-bit compliant.
- You must have Microsoft Visual C/C++ runtime version 14.40.33810 or higher.
- You must have ODBC header files to compile your application. For example, Microsoft Visual Studio includes these files.

Linux requirements for 32-bit drivers

- If your application was built with 32-bit system libraries, you must use 32-bit drivers. The database to which you are connecting can be either 32-bit or 64-bit enabled.
- 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).

Linux requirements for 64-bit drivers

- 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).

AIX requirements for 32-bit and 64-bit drivers

- IBM POWER processor
- An application compatible with components that were built using Visual Age C++ 6.0.0.0 and the AIX native threading model.

HP-UX requirements for 32-bit drivers

- The following processors are supported:
  - PA-RISC
  - Intel Itanium II (IPF)
- 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).
- 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).

HP-UX requirements for 64-bit drivers

- Intel Itanium II (IPF) processor
- HP aC++ v. 5.36 and the HP-UX 11 native (kernel) threading model (posix draft 10 threads).

Oracle Solaris requirements for 32-bit drivers

- The following processors are supported:
  - Oracle SPARC
  - x86: Intel
  - x64: Intel and AMD
- For Oracle SPARC: An application compatible with components that were built using Oracle Workshop version 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.

Oracle Solaris requirements for 64-bit drivers

- The following processors are supported:
  - Oracle SPARC
  - x64: Intel and AMD
- For Oracle SPARC: An application compatible with components that were built using Oracle Workshop version 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.

ODBC compliance

The Apache Hive Wire Protocol driver is compliant with the Open Database Connectivity (ODBC) specification. 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.

Version string information

The driver has a version string of the format:

XX.YY.ZZZZ(B*AAAA*, U*BBBB*)

or

XX.YY.ZZZZ(b*AAAA*, u*BBBB*)

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

XX.YY.ZZZZ(U*BBBB*)

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.

For example:

```
08.00.0002 (b0001, u0002)
      |__|  |___|  |___|
     Driver  Bas    Utl
```


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 driver on Linux:

```
ivtestlib ivhive28.so
```

returns:

```
08.00.0001 (B0002, U0001)
```

For example, for the Driver Manager on Linux:

```
ivtestlib libodbc.so
```

returns:

```
08.00.0001 (U0001)
```

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

```
ddtestlib libodbc.so
```

returns:

```
08.00.0001 (U0001)
```

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

```
ivtestlib libivicu28.so
08.00.0001
```

Note: Only the HP-UX version of the tool requires specifying the full path for the test loading tool. The full path does not need to be specified for other platforms.

---

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 the driver's shared object:

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

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

Data types

The following table shows how the Apache Hive data types are mapped to the standard ODBC data types.

Table 1: Apache Hive Data Types

Apache Hive	ODBC
Bigint	SQL_BIGINT
Boolean	SQL_BIT
Char[1]	SQL_WCHAR
Date	SQL_DATE
Decimal	SQL_DECIMAL
Double	SQL_DOUBLE
Float	SQL_REAL
Int	SQL_INTEGER
Smallint	SQL_SMALLINT
String[2]	SQL_WVARCHAR[3]
Timestamp	SQL_TYPE_TIMESTAMP
Tinyint	SQL_TINYINT
Varchar[4]	SQL_WVARCHAR

[1] Setting of the UseUnicodeCharTypes connection option determines where this data type maps. For example, if set to `0` (SQL_CHAR), this data type maps to SQL_CHAR.

[3] Maximum of 2 GB

[2] Setting of the StringDescribeType connection option determines where this data type maps. For example, if set to `12` (SQL_VARCHAR), this data type maps to SQL_VARCHAR.

[4] Setting of the UseUnicodeCharTypes connection option determines where this data type maps. For example, if set to `0` (SQL_VARCHAR), this data type maps to SQL_VARCHAR.

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 all platforms, 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 int     columnSize;

    SQLLEN           strlenTypeName,
                     strlenSqlDataType,
                     strlenColumnSize;

    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, &columnSize,
              (SDWORD)sizeof(columnSize), &strlenColumnSize);

// 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, columnSize);
             }

        } while (rc != SQL_NO_DATA);
    }
}
```

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 a more detailed information.

See also

SQL functionality on page 131

Additional information

In addition to the content provided in this guide, the documentation set also contains detailed conceptual and reference information that applies to all the drivers. For more information in these topics, refer the *Progress DataDirect for ODBC Drivers Reference* or use the links below to view some common topics:

- "Code page values" lists supported code page values, along with a description, for the Progress DataDirect for ODBC drivers.
- "ODBC API and scalar functions" lists the ODBC API functions supported by Progress DataDirect for ODBC drivers. In addition, it documents the scalar functions that you use in SQL statements.
- "Internationalization, localization, and Unicode" provides an overview of how internationalization, localization, and Unicode relate to each other. It also includes a background on Unicode, and how it is accommodated by Unicode and non-Unicode ODBC drivers.
- "Security best practices for ODBC applications" describes the security best practices you should employ when developing and deploying your application with the driver.

Troubleshooting

The *Progress DataDirect for ODBC Drivers Reference* provides information on troubleshooting problems should they occur.

Refer to the "Troubleshooting" section in the *Progress DataDirect for ODBC Drivers Reference* for details.

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.

April 2019, Release 8.0.1 for the Progress DataDirect for ODBC for Apache Hive Wire Protocol Driver, Version 0001

2

Getting started

This section provides basic information about configuring your driver immediately after installation, testing your connection, and accessing your data with some commonly used third-party applications. To take full advantage of the features of the driver, see "Using the driver".

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

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.

Configuring a data source

To configure a data source:

1. From the Progress 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 a default DataDirect ODBC user data source 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. On the General tab, provide the following information; then, click **Apply**.

   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 port number for the Apache Hive server is `10000`.

   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.

   Transport Mode: If you are connecting to an HTTP endpoint, set to **1 - HTTP**. By default, the driver attempts to establish a binary connection (TCP mode).

   HTTPPath: If Transport Mode is set to **1 - HTTP**, type the path of the HTTP/HTTPS endpoint used for connections. The default is `cliservice`.

3. 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.

4. For Microsoft Access and Tableau users, click on the Advanced tab. Provide the following required value for the **Max String Size** option; then, click **Apply**.

   - For Microsoft Access: Type `255`
   - For Tableau: Type a value from `255` to `4000` that suits your environment.

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. The driver returns a message indicating success or failure. A logon dialog box appears as described in "Using a logon dialog box."
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 "Using a connection string" for information about using connection strings.

See also

Using a logon dialog box on page 60
Using a connection string on page 59

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 and connecting to data sources" 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.

See also

Configuring and connecting to data sources on page 38

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 driver on Solaris, where *`xx`* represents the version number of the driver:

```
ivtestlib /opt/odbc/lib/ivhive<i>xx</i>.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.

Configuring a data source in the system information file

The default `odbc.ini` file installed in the installation directory is a template in which you create data source definitions. You enter your site-specific database connection information using a text editor. Each data source definition must include the keyword `Driver=`, which is the full path to the driver.

The following examples show the minimum connection string options that must be set to complete a test a binary (TCP mode) or HTTP connection, where *xx* represents `iv` for 32-bit or `dd` for 64-bit drivers, *yy* represents the driver level number, and *zz* represents the extension. The values for the options are samples and are not necessarily the ones you would use.

Binary (TCP Mode) Connections

```
[ODBC Data Sources]
Apache Hive Wire Protocol=DataDirect 8.0 Apache Hive Wire Protocol

[Apache Hive Wire Protocol]
Driver=ODBCHOME/lib/xxhiveyy.zz
Database=default
HostName=HiveServer
LoginID=yourid
PortNumber=10000
MaxStringSize=2147483647
```

Connection option descriptions:

`Database`: 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.

`HostName`: Either the name or the IP address of the server to which you want to connect.

`LogonID`: The default user ID that is used to connect to your database.

`PortNumber`: The port number of the server listener. The default port number for the Apache Hive server is `10000`.

`MaxStringSize`: (This option is required only for Microsoft Access and Tableau users.) The maximum size of columns of the String data type that the driver describes through result set descriptions and catalog functions. Specify the following value that corresponds to your environment:

- For Microsoft Access users, specify a value of `255`.
- For Tableau users, specify a value from `255` to `4000` that suits your environment.

HTTP Connections

```
[ODBC Data Sources]
Apache Hive Wire Protocol=DataDirect 8.0 Apache Hive Wire Protocol

[Apache Hive Wire Protocol]
Driver=ODBCHOME/lib/xxhiveyy.zz
Database=default
HostName=HiveServer
HTTPPath=cliservice
LoginID=yourid
MaxStringSize=2147483647
PortNumber=10000
TransportMode=1
```

Connection option descriptions:

`Database`: 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.

`HostName`: Either the name or the IP address of the server to which you want to connect.

`HTTPPath`: The path of the HTTP/HTTPS endpoint used for connections. The default is `cliservice`.

`LogonID`: The default user ID that is used to connect to your database.

`MaxStringSize`: (This option is required only for Microsoft Access and Tableau users.) The maximum size of columns of the String data type that the driver describes through result set descriptions and catalog functions. Specify the following value that corresponds to your environment:

- For Microsoft Access users, specify a value of `255`.
- For Tableau users, specify a value from `255` to `4000` that suits your environment.

`PortNumber`: The port number of the server listener. The default port number for the Apache Hive server is `10000`.

`TransportMode`: Specify a value of `1` to enable HTTP mode.

Testing the connection

The driver installation includes an ODBC application called example that can be used to connect to a data source and execute SQL. The application is located in the *`installation_directory`*`/samples/example` directory.

To run the program after setting up a data source in the `odbc.ini`, 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`*. If example is unable to connect, the appropriate error message is returned.

3

Tutorials

The following sections guide you through using the driver to access your data with some common third-party applications:

- Accessing data in Tableau (Windows only) on page 29
- Accessing data in Microsoft Excel (Windows only) on page 32
- Accessing data in Microsoft Excel from the Query Wizard (Windows only) on page 34

For details, see the following topics:

- The Example application
- Accessing data in Power BI (Windows only)
- Accessing data in Tableau (Windows only)
- Accessing data in Microsoft Excel (Windows only)
- Accessing data in Microsoft Excel from the Query Wizard (Windows only)

The Example application

The driver installation includes an ODBC application called Example that can be used for:

- Testing any type of SQL statement
- Testing database connections

- Verifying your database environment

It can also be used to demonstrate ODBC function calls, including the following:

- SQLAllocHandle
- SQLBindCol
- SQLBindParameter
- SQLColAttribute
- SQLConnect
- SQLDescribeCol
- SQLDescribeParam
- SQLDisconnect
- SQLDriverConnect
- SQLExecDirect
- SQLFetch
- SQLFreeHandle
- SQLFreeStmt
- SQLGetDiagRec
- SQLGetInfo
- SQLNumResultCols
- SQLPrepare
- SQLSetEnvAttr
- SQLSetStmtAttr

The Example application can be built using the files located in the `\samples\examples` directory of the DataDirect for ODBC Drivers installation directory.

Note:

- For Windows, you can build the Windows app for ANSI and Unicode.
- For UNIX/Linux, instructions for building the Example application are contained inside the file `example.mak`, which can be read with a text editor.

To use the Example application:

1. After you have configured the data source, navigate to the `instal_dir\samples\example` directory.
2. Open the application using one of the following methods:
   - Running the application executable or binary:
     - On Windows, double-click the `Example.exe` file.
     - On UNIX/Linux, run the `example` application.
   - Executing a command-line argument. For example:
     - `example` *`connection_string`*
     - `example "DSN" "UID" "PWD"`
     - `example` *`connection_string`* `"`*`sql_command_1`*`" ["`*`sql_command_2`*`" ...]`

   Results: A command prompt opens.
3. Follow the prompts to enter your data source name, user name, and password. If successful, a SQL> prompt appears.
4. At the prompt, enter SQL statements to test your connection. For example:

   ```
   SELECT * FROM INFORMATION_SCHEMA.TABLES
   ```

The results of your query are displayed. If example is unable to connect, the appropriate error message is returned.

Accessing data in Power BI (Windows only)

After you have configured your data source, you can use the driver to access your 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.

1. Navigate to the `\tools\Power BI` subdirectory of the Progress DataDirect installation directory; then, locate the installation batch file `install.bat`.
2. Run the `install.bat` file. The following operations are executed by running the `install.bat` file:
   - The Power BI connector file, `DataDirectHive.pqx`, 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 > Progress DataDirect Apache Hive Connector**.
5. Click **Connect**. Then, from the **Progress DataDirect Apache Hive Connector** window, provide the following information. Then, click **OK**.
   - **Data Source:** Enter a name for the data source. For example, `Apache Hive 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.

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.

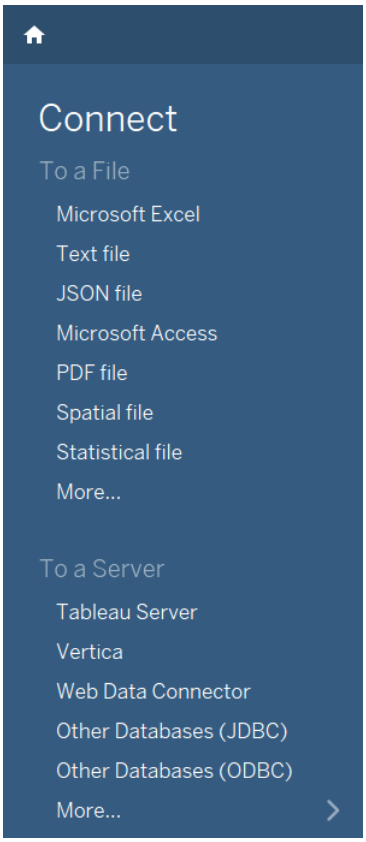
Accessing data in Tableau (Windows only)

After you have configured your data source, you can use the driver to access your Apache Hive data with Tableau. Tableau is a business intelligence software program that allows you to easily create reports and visualized representations of your data. By using the driver with Tableau, you can improve performance when retrieving data while leveraging the driver's relational mapping tools.

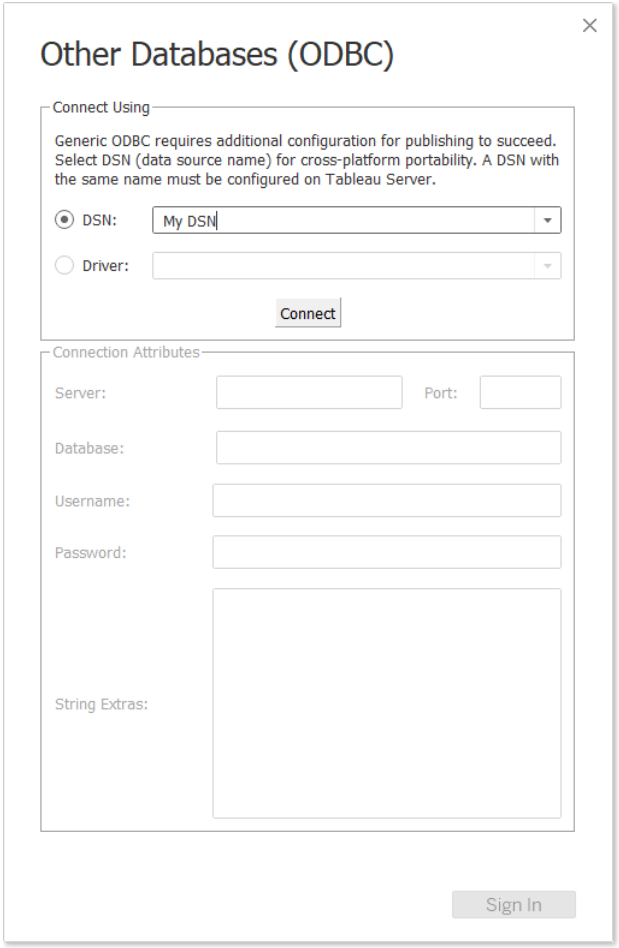
To use the driver to access data with Tableau:

1. Navigate to the `\tools\Tableau` subdirectory of the Progress DataDirect installation directory; then, locate the Tableau data source file, `DataDirect Apache Hive.tdc`.
2. Copy the `DataDirect Apache Hive.tdc` into the following directory:

   `C:\Users\`*`user_name`*`\Documents\My Tableau Repository\Datasources`
3. Open Tableau. If the **Connect** menu does not open by default, select **Data > New Data Source** or the Add New Data Source button to open the menu.

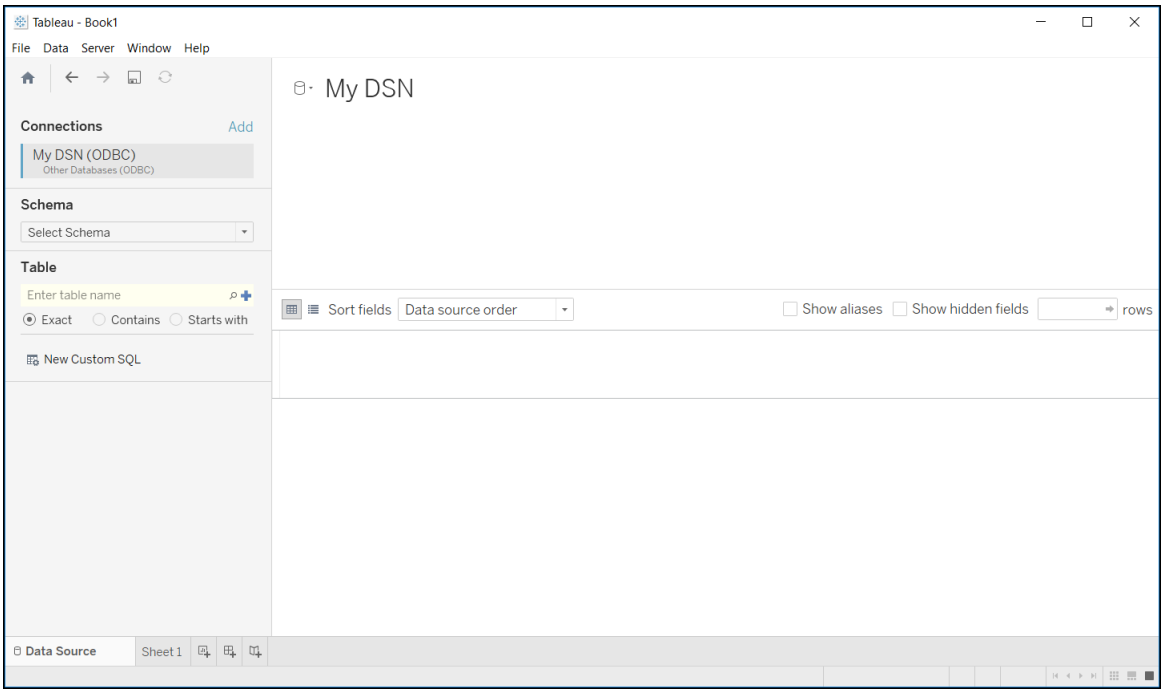


4. From the **Connect** menu, select **Other Databases (ODBC)**.
5. The **Server Connection** dialog appears.



In the DSN field, select the data source you want to use from the drop down menu. For example, **My DSN**. Then, click **Connect**. The **Logon to Apache Hive** dialog appears pre-populated with the connection information you provided in your data source.

6. If required, type your user name and password; then, click **OK**. The Logon dialog closes. Then, click **OK** on the Server Connection dialog.
7. The **Data Source** window appears.



By default, Tableau connects live, or directly, to your data. We recommend that you use the default settings to avoid extracting all of your data. However, if you prefer, you can import your data by selecting the **Extract** option at the top of the dialog.

8. In the Schema field, select the database you want to use. The tables stored in this database are now available for selection in the Table field.

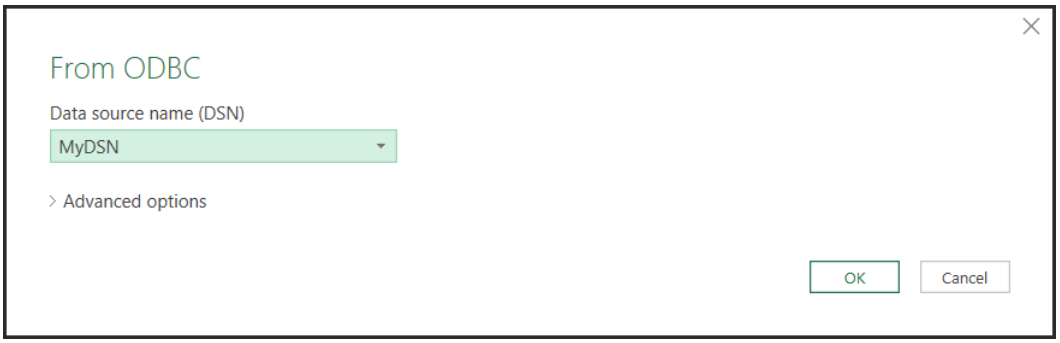
You have successfully accessed your data and are now ready to create reports with Tableau. For more information, refer to the Tableau product documentation at: http://www.tableau.com/support/help.

Accessing data in Microsoft Excel (Windows only)

After you have configured your data source, you can use the driver to access your data with Microsoft Excel from the Data Connection Wizard. Using the driver with Excel provides improved performance when retrieving data, while leveraging the driver's relational-mapping tools.

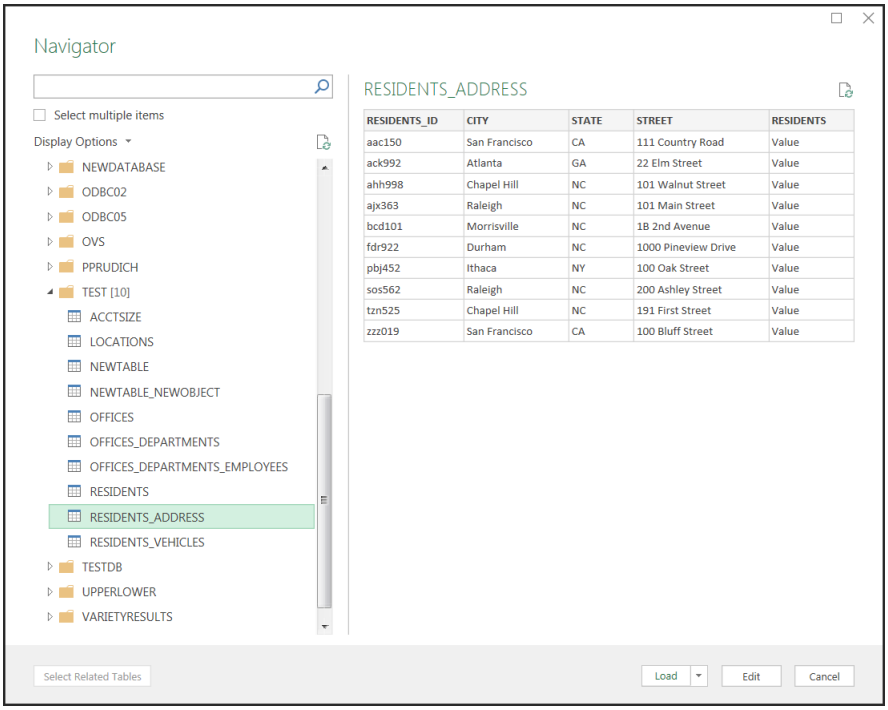
To use the driver to access data with Excel from the Data Connection Wizard:

1. Open your workbook in Excel.
2. From the **Data** menu, select **Get Data**>**From Other Sources**>**From ODBC**.
3. The **From ODBC** dialog appears.



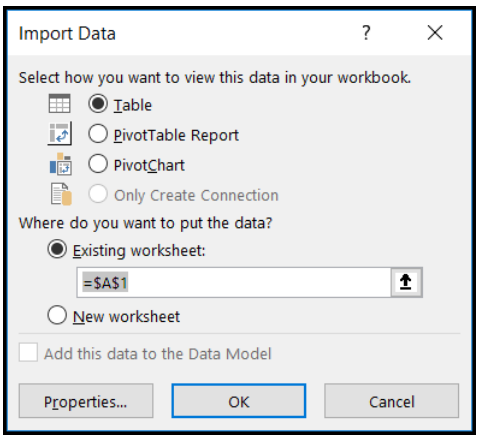
Select your data source from the Data Source Name (DSN) drop down; then, click **OK**.

4. You are prompted for logon credentials for your data source:
   - If your data source does not require logon credentials or if you prefer to specify your credentials using a connection string, select **Default or Custom** from the menu on the left. Optionally, specify your credential-related properties using a connection string in the provided field. Click **Connect** to proceed.
   - If your data source uses Windows credentials, select **Windows** from the menu; then, provide your credentials. Optionally, specify a connection string with credential-related properties in the provided field. Click **Connect** to proceed.
   - If your data source uses credentials stored on the database, select **Database**; then, provide your user name and password. Optionally, specify a connection string in the provided field. Click **Connect** to proceed.
5. The **Navigator** window appears.



From the list, select the tables you want to access. A preview of your data will appear in the pane on the right. Optionally, click **Edit** to modify the results using the Query Editor. Refer to the Microsoft Excel product documentation for detailed information on using the Query Editor.

6. Load your data:
   - Click **Load** to import your data into your work sheet. Skip to the end.
   - Click **Load**>**Load To** to specify a location to import your data. Proceed to the next step.
7. The **Import Data** window appears.



Select the desired view and insertion point for the data. Click **OK**.

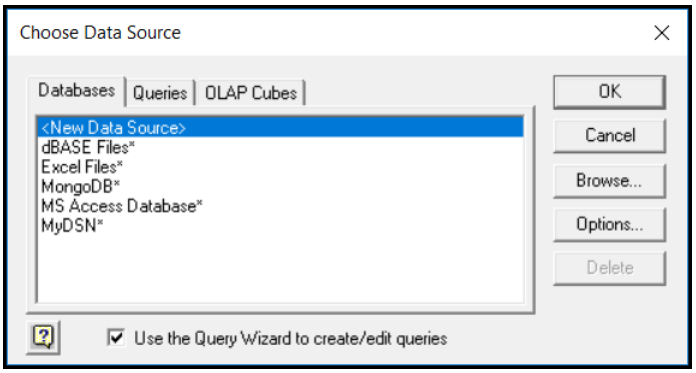
You have successfully accessed your data in Excel. For more information, refer to the Microsoft Excel product documentation at: https://support.office.com/.

Accessing data in Microsoft Excel from the Query Wizard (Windows only)

After you have configured your data source, you can use the driver to access your data with Microsoft Excel from the Query Wizard. Using the driver with Excel provides improved performance when retrieving data, while leveraging the driver's relational-mapping tools.

To use the driver to access data with Excel from the Query Wizard:

1. Open your workbook in Excel.
2. From the **Data** menu, select **Get Data**>**From Other Sources**>**From Microsoft Query**.
3. The **Choose Data Source** dialog appears.



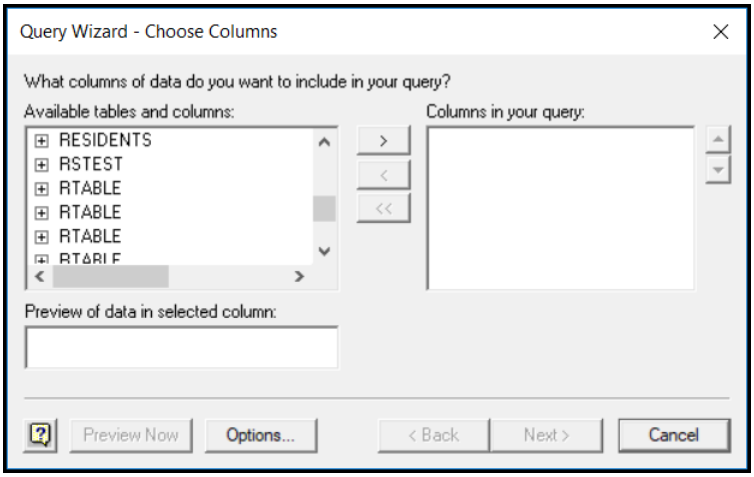
From the Databases list, select your data source. For example, **MyDSN**. Click **OK**.

4. The logon dialog appears pre-populated with the connection information you provided in your data source. If required, type your password. Click **OK** to proceed.

Note: The logon dialog may reappear if Excel needs to access additional information from the data source. If this occurs, re-enter your password; then, click **OK** to proceed to the next step.

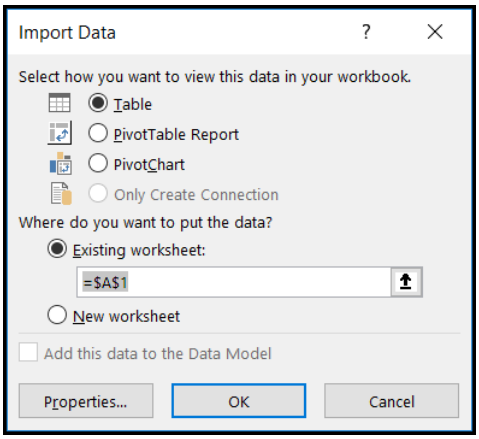
---

5. The **Query Wizard - Choose Columns** window appears.



Choose the columns you want to import into your workbook. To add a column, select the column name in Available tables and columns pane; then, click the **>** button. After you add the columns you want to include, click **Next** to continue.

6. Optionally, filter your data using the drop-down menus; then, click **Next**.
7. Optionally, sort your data using the drop-down menus; then, click **Next**.
8. Select "Return Data to Microsoft Excel"; then, click **Finish**.
9. The **Import Data** window appears.



Select the desired view and insertion point for your data. Click **OK**.

You have successfully accessed your data in Excel using the Query Wizard. For more information, refer to the Microsoft Excel product documentation at: https://support.office.com/.

4

Using the driver

This chapter guides you through the configuring and connecting to data sources. In addition, it explains how to use the functionality supported by your driver.

For details, see the following topics:

Configuring and connecting to data sources

After you install the driver, you configure data sources to connect to the database. See "Getting started" 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 "Connection option descriptions" for an alphabetical list of driver connection string attributes and their initial default values.

See also

Getting started on page 21
Connection option descriptions on page 87

Configuring the product on UNIX/Linux

This chapter contains specific information about using your driver in the UNIX and Linux environments.

See "Environment variables" for additional platform information.

See also

Environment variables on page 38

Environment variables

The first step in setting up and configuring the driver 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 Progress DataDirect *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 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.

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 UNIX/Linux" 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.

See also

Data source configuration on UNIX/Linux on page 41

ODBCINST

Setup installs in the product installation directory a default file, named odbcinst.ini, for use with DSN-less connections. See "DSN-less connections" 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.

See also

DSN-less connections on page 44

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" 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.

See also

Configuration through the system information (odbc.ini) file on page 41

The test loading tool

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

The ivtestlib (32-bit driver) and ddtestlib (64-bit driver) 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 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 driver on Solaris, where *xx* represents the version number of the driver:

```
ivtestlib /opt/odbc/lib/ivhivexx.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" for details about version strings.

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

See also

Version string information on page 15

Data source configuration on UNIX/Linux

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. This is a plain text file that contains data source definitions.

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:

```
Apache Hive=DataDirect 8.0 Apache Hive Wire Protocol Driver.
```

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 the driver. Each data source definition begins with a data source name in square brackets, for example, `[Apache Hive]`. The data source definitions contain connection string *attribute=value* pairs with default values. You can modify these values as appropriate for your system. "Connection option descriptions" describes these attributes. See "Sample default odbc.ini file" 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").

Note: This section is not included in the default `odbc.ini` file that is installed by the product installer. You must add this section manually.

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

```
[ODBC]
IANAAppCodePage=4
InstallDir=/opt/odbc
Trace=0
TraceFile=odbctrace.out
TraceDll=/opt/odbc/lib/ivtrc28.so
ODBCTraceMaxFileSize=102400
ODBCTraceMaxNumFiles=10
```

The IANAAppCodePage keyword defines the default value that the UNIX/Linux driver uses if individual data sources have not specified a different value. See "IANAAAppCodePage" in "Connection option descriptions". The default value is `4`.

For supported code page values, refer to "Code page values" in the *Progress DataDirect for ODBC Drivers Reference*.

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 `odbcinst.ini` file and do not have an `odbc.ini` file, then you must provide `[ODBC]` section information in the `[ODBC]` section of the `odbcinst.ini` file. The driver 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 `odbcinst.ini` file. See "DSN-Less connections" for details.

ODBC tracing allows you to trace calls to the ODBC driver 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*.

See also

Connection option descriptions on page 87
Sample default odbc.ini file on page 42
File data sources on page 45
IANAAppCodePage on page 109
DSN-less connections on page 44

Sample default odbc.ini file

The following is a sample `odbc.ini` file that Setup installs in the installation directory. All occurrences of ODBCHOME 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 a 32-bit driver with the driver file name beginning with `iv`. A 64-bit driver file would be identical except that driver name would begin with `dd` and the list of data sources would include only the 64-bit drivers.

```
[ODBC Data Sources]
Apache Hive=DataDirect 8.0 Apache Hive Wire Protocol

[Apache Hive]
Driver=ODBCHOME/lib/ivhive28.so
Description=DataDirect 8.0 Apache Hive Wire Protocol
ArrayFetchSize=150000
ArrayInsertSize=16384
AuthenticationMethod=0
BatchMechanism=2
CatalogMode=0
```

```
CookieName=
CryptoProtocolVersion=TLSv1.2, TLSv1.1, TLSv1
Database=<database_name>
EnableCookieAuthentication=0
EnableDescribeParam=0
EnableFIPS=1
EncryptionMethod=0
GSSClient=native
HostName=
HostNameInCertificate=
HTTPPath=cliservice
KeepAlive=0
KeyPassword=
KeystorePassword=
LoginTimeout=30
LogonID=
MaxStringSize=2147483647
MinLongVarcharSize=
PortNumber=10000
ProxyUser=
RemoveColumnQualifiers=0
ServicePrincipalName=
StringDescribeType=12
TransactionMode=0
TransportMode=0
Truststore=
TruststorePassword=
UseCurrentSchema=0
ValidateServerCertificate=1
VarcharThreshold=
ZooKeeperNamespace=/hiveserver2
ZooKeeperDiscovery=0


[ODBC]
IANAAppCodePage=4
InstallDir=ODBCHOME
Trace=0
TraceFile=odbctrace.out
TraceDll=ODBCHOME/lib/ivtrc28.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:**

  a) Using a text editor, open the `odbc.ini` file.

  b) 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 "Attribute Names for the Driver for Apache Hive" table in the "Connection Options Descriptions" for other specific attribute values.

  c) After making all modifications, save the `odbc.ini` file and close the text editor.

  Important: The "Connection option descriptions" section 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:**

a) Using a text editor, open the `odbc.ini` file.

b) Copy an appropriate existing default data source definition and paste it to another location in the file.

c) 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, `[Apache Hive Wire Protocol]`.

d) 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 "Attribute Names for the Driver for Apache Hive" table in the "Connection option descriptions" for other specific attribute values.

e) 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.

f) After making all modifications, save the `odbc.ini` file and close the text editor.

Important: The "Attribute Names for the Driver for Apache Hive" table in the "Connection option descriptions" section 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.

See also

Connection option descriptions on page 87

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" 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 8.0 Apache Hive 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" 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.

See also

ODBCINST on page 39
Configuration through the system information (odbc.ini) file on page 41

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 a 32-bit driver with the driver file name beginning with `iv`; a 64-bit driver file would be identical except that driver names would begin with `dd`.

```
[ODBC Drivers]
DataDirect 8.0 Apache Hive Wire Protocol=Installed

[DataDirect 8.0 Apache Hive Wire Protocol]
Driver=ODBCHOME/lib/ivhive28.so
APILevel=1
ConnectFunctions=YYY
DriverODBCVer=3.52
FileUsage=1
HelpRootDirectory=ODBCHOME/HiveHelp
Setup=ODBCHOME/lib/ivhive28.so
SQLLevel=0

[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/ivtrc28.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 "Getting started" 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 with a text editor. The file normally has an extension of `.dsn`.

For example, a file data source for the driver would be similar to the following:

```
[ODBC]
Driver=DataDirect 8.0 Apache Hive Wire Protocol
Database=default
Port=10000
HostName=Hive2
LogonID=JOHN
```

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").

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\Hive2.dsn
```

or, on UNIX and Linux:

```
FILEDSN=/home/users/john/filedsn/Hive2.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 "Data source configuration on UNIX/Linux" 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/Hive2.dsn;UID=james;PWD=test01
```

See also

Getting started on page 21
DSN-less connections on page 44
Data source configuration on UNIX/Linux on page 41

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);
    ```

Data source configuration through a GUI

 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 Hive data source:

1. Start the ODBC Administrator by selecting its icon from the Progress DataDirect for ODBC 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.

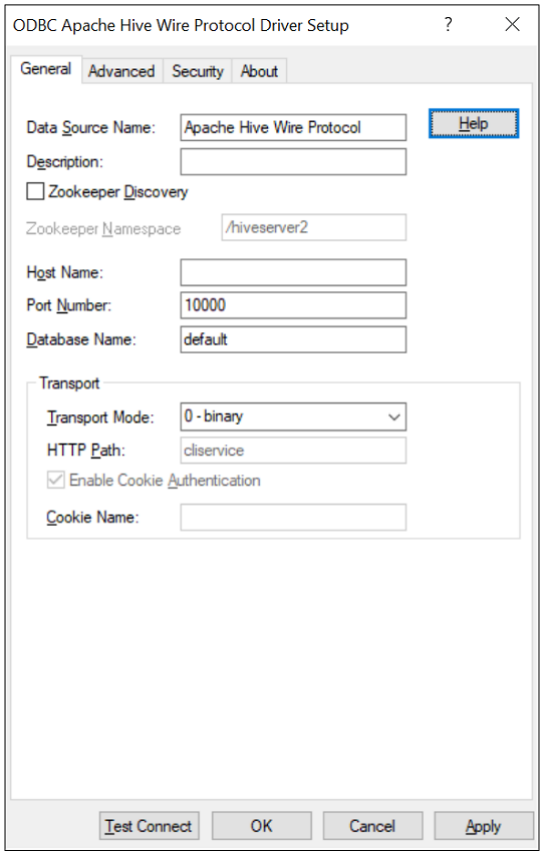
     If you are configuring a new user data source, click **Add** to display a list of installed drivers. Select the driver and click **Finish** to display the driver Setup dialog box.
   - **System DSN**: If you are configuring an existing system data source, select the data source name and click **Configure** 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. The General tab of the Setup dialog box appears by default.

Figure 1: General tab



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. The General tab displays fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.

Connection Options: General	Description
Data Source Name on page 99	Specifies the name of a data source in your Windows Registry or `odbc.ini` file. **Default:** None
Description on page 101	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. **Default:** None

Connection Options: General	Description
Zookeeper Discovery on page 129	Determines whether the driver uses Apache ZooKeeper when connecting to a database server. If disabled, the driver does not use ZooKeeper when connecting to a database server. By default, the driver's behavior is determined by the connection options settings. If enabled, the driver attempts to connect to the member servers of a ZooKeeper ensemble that are specified by the Host Name connection option. At connection, the driver retrieves configuration information from the ZooKeeper service that determines the behavior of the driver for the connection. The retrieved configuration information takes precedent over any values specified using connection options. **Default:** Disabled
Zookeeper Namespace on page 129	Specifies the name of the Apache ZooKeeper name space to which you want to retrieve configuration information. **Default:** `/hiveserver2`
Host Name on page 106	The name or the IP address of the server to which you want to connect. When Apache ZooKeeper support is enabled (`ZooKeeperDiscovery=1`), this option specifies a comma-seperated list of member servers of the ZooKeeper ensemble to which you want to connect. For example: `server1:10000,255.125.1.11:2818,server3:2828` **Default:** None
Port Number on page 116	Specifies the port number of the server listener. **Default:** `10000`
Database Name on page 100	Specifies the name of the Hive database. The database must exist, or the connection attempt will fail. **Default:** `default`
Transport Mode on page 122	Specifies whether binary (TCP) mode or HTTP mode is used to access Apache Hive data sources. If set to **0 - binary**, Thrift RPC requests are sent directly to data sources using a binary connection (TCP mode). If set to **1 - HTTP**, Thrift RPC requests are sent using HTTP transport (HTTP mode). HTTP mode is typically used when connecting to a proxy server, such as a gateway, for improved security, or a load balancer. **Default: 0 - binary**
HTTP Path on page 108	Specifies the path of the HTTP/HTTPS endpoint used for connections when HTTP mode is enabled (`TransportMode=1`). **Default:** `cliservice`

Connection Options: General	Description
Enable Cookie Authentication on page 102	Determines whether the driver attempts to use session cookie based authentication for requests to an HTTP endpoint after the initial authentication to the server. Cookie based authentication improves response time by eliminating the need to re-authenticate with the server for each request. If disabled, the driver does not use cookie based authentication for HTTP requests after the initial authentication. If enabled, the driver attempts to use cookie based authentication for requests to an HTTP endpoint after the initial authentication to the server. The cookie used for authentication is specified by the Cookie Name option. **Default:** Enabled
Cookie Name on page 96	Specifies the name of the cookie used for authenticating HTTP requests when HTTP mode (`TransportMode=1`) and cookie based authentication are enabled (`EnableCookieAuthentication=1`). **Default:** If no value is specified, the driver attempts to use the following cookie names by default: • `hive.server2.auth` (Hive connections) • `hadoop.auth` (Apache Knox connections) • `JSESSIONID` (Apache Knox connections)

4. 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" for details). Note that the information you enter in the logon dialog box during a test connect is not saved.
5. To further configure your driver, click on the following tabs. The corresponding sections provide details on the fields specific to each configuration tab:
   - Advanced tab allows you to configure advanced behavior.
   - Security tab allows you to specify security data source settings.
6. Click **OK**. When 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.

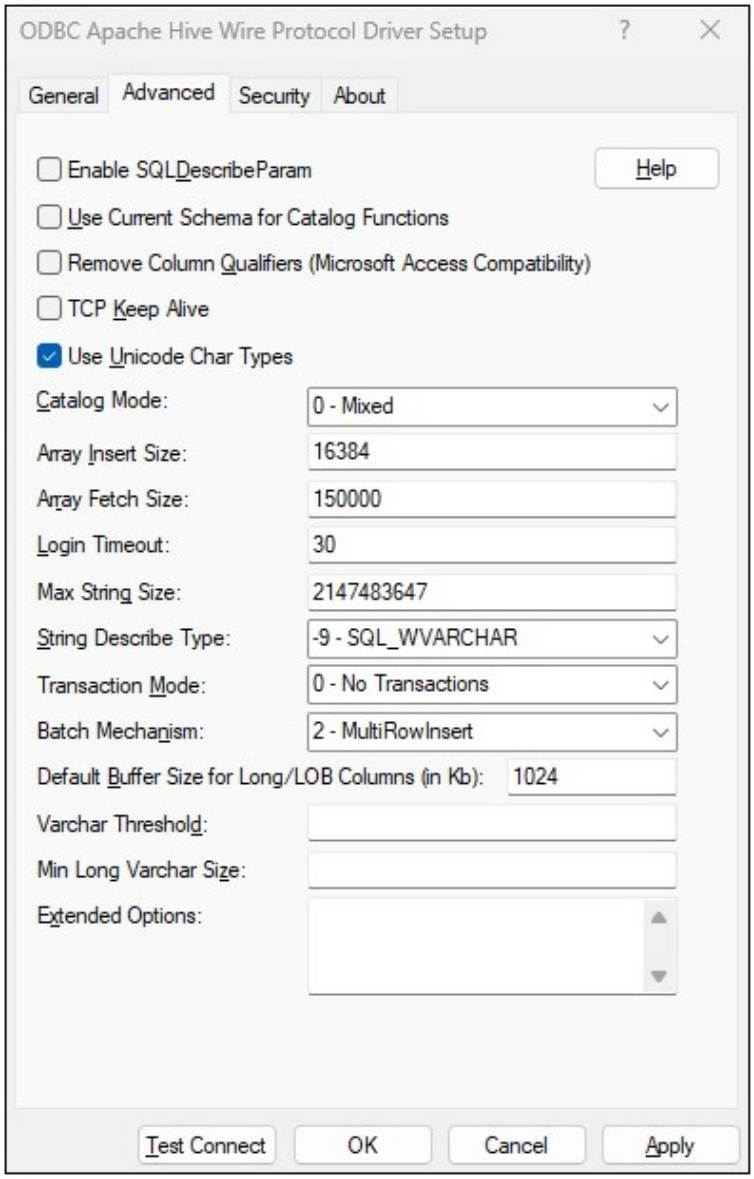
See also

Using a logon dialog box on page 60

Advanced tab

The Advanced tab allows you to specify additional data source settings. The fields are optional unless otherwise noted. On this tab, provide values for the options in the following table; then, click **Apply**.

Figure 2: Advanced tab



Connection Options: Advanced	Description
Enable SQLDescribeParam on page 103	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. If enabled, the SQLDescribeParam function describes parameters as a data type of SQL_VARCHAR with a length of 255 for statements. If disabled, the SQLDescribeParam function returns the standard ODBC error IM001. **Default:** Disabled
Use Current Schema for Catalog Functions on page 125	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. If enabled, results of catalog function calls are restricted to the tables and views in the current schema. If disabled, results of catalog function calls are not restricted. **Default:** Disabled
Remove Column Qualifiers on page 117	Specifies whether the driver removes 3-part column qualifiers and replaces them with alias.column qualifiers. If 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 disabled, the driver does not modify the request. **Default:** Disabled
TCP Keep Alive on page 121	Specifies whether the driver enables TCPKeepAlive. If disabled, the driver does not enable TCPKeepAlive. If enabled, the driver enables TCPKeepAlive. **Default:** Disabled
Use Unicode Char Types on page 126	Determines whether char and varchar columns are described as SQL_CHAR and SQL_VARCHAR types or SQL_WCHAR and SQL_WVARCHAR types. If disabled, Char columns are described as SQL_CHAR and Varchar columns are described as SQL_VARCHAR. If enabled, Char columns are described as SQL_WCHAR and Varchar columns are described as SQL_WVARCHAR. **Default:** Enabled

Connection Options: Advanced	Description
Catalog Mode on page 95	Specifies whether the driver uses native catalog functions to retrieve information returned by the SQLTables, SQLColumns, and SQLStatistics catalog functions. If set to **0 - Mixed**, the driver uses a combination of ODBC catalog functions and driver-discovered information to retrieve catalog information. Select this option for the optimal balance of performance and accuracy. **Note:** In this setting, the driver uses the optimal techniques for retrieving information. These techniques vary depending on the server used, which may result in differences in performance. If set to **1 - Native**, the driver uses native catalog functions to retrieve information returned by the SQLTables, SQLColumns, and SQLStatistics catalog functions. This option provides the best performance, but at the expense of less-accurate catalog information. If set to **2 - Query Based**, the driver uses driver-discovered information to retrieve catalog information. This option provides a high accuracy of catalog information, but at the expense of slower performance. **Default: 0 - Mixed**
Array Insert Size on page 93	Specifies the maximum buffer size, in KB, the driver uses for a packet when executing a multi-row insert. In most scenarios, the default setting provides the ideal driver behavior; however, you may need to reduce the value specified if you encounter either of the following: • Performance or memory issues when inserting larger values. • The following error when inserting larger values while using Apache Knox: `HTTP/1.1 500 Server Error.` Default: `16384`
Array Fetch Size on page 92	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. **Default:** `150000`
Login Timeout on page 111	The number of seconds the driver waits for a connection to be established before returning control to the application and generating a timeout error. **Default:** `30`
Max String Size on page 112	Specifies the maximum size of columns of the String data type that the driver describes through result set descriptions and catalog functions. **Default:** `2147483647`

Connection Options: Advanced	Description
String Describe Type on page 120	Specifies whether all string columns are described as SQL_WVARCHAR. 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. **Default: -9 - SQL_WVARCHAR**
Transaction Mode on page 121	Specifies how the driver handles manual transactions. 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**
Batch Mechanism on page 95	Determines the mechanism that is used to execute batch operations. 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**
Default Buffer Size for Long/LOB Columns (in Kb) on page 100	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. 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). **Default:** `1024`

Connection Options: Advanced	Description
Varchar Threshold on page 128	Specifies the threshold at which the driver describes columns of the data type VARCHAR as LONGVARCHAR. If the size of the VARCHAR column exceeds the value specified, the driver will describe the column as LONGVARCHAR when calling SQLDescribeCol and SQLColumns. This option allows you to fetch columns that would otherwise exceed the upper limit of the VARCHAR type for some third-party applications, such as SQL Server Linked Server. **Default:** None. If no value is specified, the driver will not change the described type for VARCHAR columns.
Min Long Varchar Size on page 113	Specifies the minimum count of characters the driver reports for columns mapped as LONGVARCHAR. If the size of a 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 LONGVARCHAR columns whose size is smaller than the minimum imposed by some third-party applications, such as SQL Server Linked Server. **Default:** None. If no value is specified, the driver will not change the column size reported for LONGVARCHAR columns.

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=myhive;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.

If you finished configuring your driver, proceed to Step 6 in "Data source configuration through a GUI". Optionally, you can further configure your driver by clicking on the following tabs. The following sections provide details on the fields specific to each configuration tab:

- General tab allows you to configure options that are required for creating a data source.
- Security tab allows you to specify security data source settings.

See also

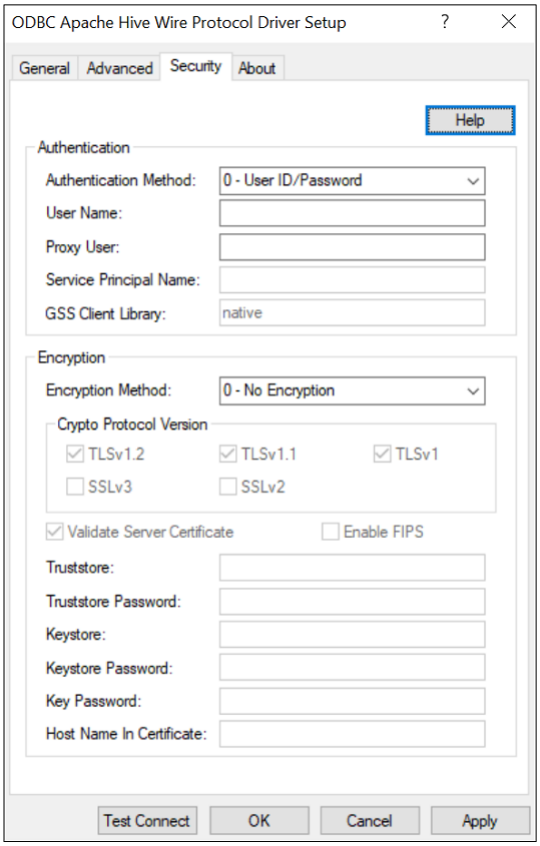
Data source configuration through a GUI on page 47

Security tab

The Security tab allows you to specify your security settings. The fields are optional unless otherwise noted. On this tab, provide values for the options in the following table; then, click **Apply**.

See "Using security" for a general description of authentication and encryption and their configuration requirements.

Figure 3: Security tab



Connection Options: Security	Description
Authentication Method on page 94	Specifies the method the driver uses to authenticate the user to the server when a connection is established. 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 **-1 - No Authentication**, 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- User ID/Password**
User Name on page 127	The default user ID that is used to connect to your database. **Default:** None

Connection Options: Security	Description
Proxy User on page 116	Specifies the UserID used for Impersonation and Trusted Impersonation. When impersonation is enabled on the server, this value determines your identity and access rights to files when executing queries. **Default:** None. 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.
Service Principal Name on page 118	The service principal name to be used by driver for Kerberos authentication. **Default:** None.
GSS Client Library on page 105	The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC). **Default:** `native` (the driver uses the GSS client for Windows Kerberos.)
Encryption Method on page 104	The method the driver uses to encrypt data sent between the driver and the database server. 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. **Default: 0 - No Encryption**
Crypto Protocol Version on page 97	Specifies the cryptographic protocols to use when SSL is enabled using the Encryption Method connection option (`EncryptionMethod=1`). **Default: TLSv1.2**, **TLSv1.1**, **TLSv1**
Validate Server Certificate on page 127	Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (`Encryption Method=1`). If 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 disabled, the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information specified by the Truststore and Truststore Password options. **Default:** Enabled

Connection Options: Security	Description
Enable FIPS on page 102	Determines whether the OpenSSL library uses cryptographic algorithms from the FIPS provider or the default provider when TLS/SSL encryption is enabled (`Encryption Method=1`). If disabled, the OpenSSL library uses cryptographic algorithms from the default provider. If enabled, the OpenSSL library uses cryptographic algorithms from the FIPS provider. **Default:** Disabled
Truststore on page 123	The directory that contains the truststore file and the truststore file name to be used when SSL is enabled (`EncryptionMethod=1`) and server authentication is used. **Default:** None
Truststore Password on page 124	Specifies the password that is used to access the truststore file when SSL is enabled (`EncryptionMethod=1`) and server authentication is used. **Default:** None
Keystore on page 110	The name of the directory containing the keystore file to be used when SSL is enabled (`EncryptionMethod=1`) and SSL client authentication is enabled on the database server. **Default:** None
Keystore Password on page 111	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. **Default:** None
Key Password on page 109	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. **Default:** None
Host Name In Certificate on page 107	A host name for certificate validation when SSL encryption is enabled (`Encryption Method=1`) and validation is enabled (`Validate Server Certificate=1`). **Default:** None

If you finished configuring your driver, proceed to Step 6 in "Data source configuration through a GUI." Optionally, you can further configure your driver by clicking on the following tabs. The following sections provide details on the fields specific to each configuration tab:

- General tab allows you to configure options that are required for creating a data source.
- Advanced tab on page 51 allows you to configure advanced behavior.

See also

Using security on page 63
Data source configuration through a GUI on page 47

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" 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 Apache Hive for Linux/UNIX/Windows is:

```
DSN=Hive;UID=JOHN;PWD=XYZZY
```

A FILEDSN connection string is similar except for the initial keyword:

```
FILEDSN=Hive;UID=JOHN;PWD=XYZZY
```

A DSN-less connection string must provide all necessary connection information:

```
DRIVER=DataDirect 8.0 Apache Hive Wire
Protocol;HOST=HiveServer;PORT=10000;UID=JOHN;PWD=XYZZY;DB=HIVE2
```

See also

Connection option descriptions on page 87

Password Encryption Tool (UNIX/Linux only)

On UNIX and Linux, Progress DataDirect provides a Password Encryption Tool, called ddencpwd, that encrypts passwords for secure handling in connection strings and odbc.ini files. At connection, the driver decrypts these passwords and passes them to the data source as required. Passwords can be encrypted for any option, including:

- KeyPassword
- KeyStorePassword
- TrustStorePassword
- Password

To use the Password Encryption Tool:

1. From a command line, navigate to the directory containing the `ddencpwd` application. By default, this is *`install_directory`*`/tools`.
2. Enter the following command:

   `ddencpwd` *`password`*

   where:

   `password`

   is the password you want to encrypt.

3. The tool returns an encrypted password value. Specify the returned value for the corresponding attribute in the connection string or `odbc.ini` file. For example, if you encrypted the password for `KeyPassword`, specify the following in your connection string or datasource definition:

   `KeyPassword=`*`returned_value`*

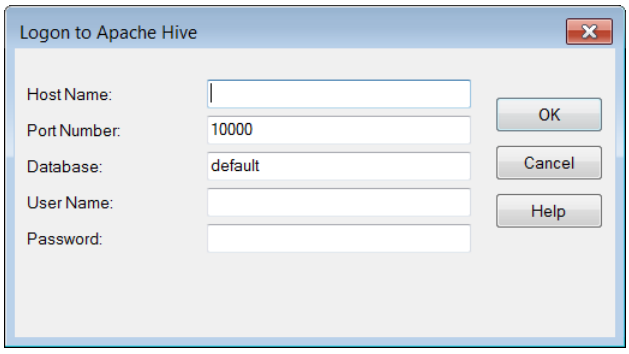
4. Repeat Steps 2 and 3 to encrypt additional passwords.
5. If using an `odbc.ini` file, save your file.

This completes this tutorial. You are now ready to connect using encrypted passwords.

Using a logon dialog box

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 4: Logon to Apache Hive dialog box



In this dialog box, provide the following information:

1. In the Host Name field, type the name or the IP address of the server to which you want to connect to which you want to connect.
2. In the Port Number field, type the port number of the server listener.

3. In the Database field, type the name of the database to which you want connect.
4. In the User Name field, type your logon ID.
5. In the Password field, type your password.
6. Click **OK** to complete the logon.

HTTP mode

In addition to the default thrift protocol (binary mode), the driver also supports HTTP mode, which allows you to access Apache Hive data stores using HTTP/HTTPS requests. When HTTP mode is enabled, thrift RPC messages are sent to an endpoint using HTTP transport. HTTP mode is typically employed when there is a need to access data through a proxy server, such as when connecting to a load balancer or a gateway server. Unless otherwise noted, the same features and functionality are supported for both the thrift and HTTP protocols.

To connect to a server using HTTP Mode:

1. Configure the minimum required options required for a connection:
   - Set the Database Name option to provide the name of the Apache Hive database to which you want to connect.
   - Set the Host Name option to provide the name or the IP address of the server to which you want to connect.
   - Set the Port Number option to provide the TCP port of the primary database server that is listening for connections to the Apache Hive database. The default is `10000`.
2. Set the Transport Mode option to `1` (HTTP).
3. Optionally, if not using the default HTTP endpoint, set the HTTP Path option to provide the path of the endpoint to be used for HTTP/HTTPS requests. The default is `cliservice`.
4. Optionally, if you are sending requests to HTTPS endpoints, set the Encryption Method option to `1` (SSL) to enable SSL data encryption. Data encryption behavior can be further configured using the connection properties described in "TLS/SSL encryption".

The following examples demonstrate a basic configuration of HTTP mode with SSL enabled.

Using a connection URL:

```
DRIVER=DataDirect 8.0 Apache Hive Wire
Protocol;HostName=HiveServer;PortNumber=10001;
DatabaseName=mydb1;EncryptionMethod=1;HTTPPath=mywarehouse;TransportMode=1;
```

Using the `odbc.ini` file:

```
Driver=ODBCHOME/lib/ivhivexx.so
Description=DataDirect Apache Hive Wire Protocol
DatabaseName=mydb1
EncryptionMethod=1
HTTPPath=mywarehouse
HostName=HiveServer
PortNumber=10001
TransportMode=1
```

See also

Database Name on page 100

Host Name on page 106
Port Number on page 116
Transport Mode on page 122
HTTP Path on page 108
Encryption Method on page 104
TLS/SSL encryption on page 67

Performance considerations

The following connection options can enhance driver performance.

Array Fetch Size (ArrayFetchSize): To improve throughput, consider increasing the value of Array Fetch Size. By increasing the value, 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.

Array Insert Size (ArrayInsertSize): You can improve performance when executing batch inserts by increasing the max buffer size using Array Insert Size. By increasing the value for Array Insert Size, you increase the size of the packet sent to the server, therefore reducing the expense associated with multiple network round trips. The tradeoff for improved throughput is a greater demand on memory and slower response times. If you encounter memory issues or memory related server errors, you may need to decrease the value for this option.

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.

Catalog Mode (CatalogMode): Apache Hive's native catalog functions return incorrect information in certain scenarios. To address this issue, by default, the driver uses a combination of driver-discovered information and native functions to retrieve more accurate catalog information than native functions alone. While using driver-discovered information improves accuracy, it does so at an expense to performance. If accurate catalog information is not required, you can improve performance by setting Catalog Mode connection option to 1 (Native).

Default Buffer Size for Long/LOB Columns (in Kb) (**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.

EnableCookieAuthentication (EnableCookieAuthentication): To improve response time when using HTTP mode (`TransportMode=1`), enable session cookie based authentication (`EnableCookieAuthentication=1`). When cookie based authentication is enabled (the default), the driver uses cookies authenticate requests to the server after the initial authentication that occurs at connection. This eliminates the overhead associated with executing a standard re-authentication attempt for each request to the server.

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.

String Describe Type (StringDescribeType): To fetch String as SQL_WLONGVARCHAR, the String Describe Type connection option must be set to `-10` (SQL_WLONGVARCHAR). When String Describe Type is set to `-10`, the driver not only maps String to SQL_WLONGVARCHAR, but also allocates more space to cache the long data. Because more space is allocated for the long data, your application will incur a performance penalty.

See also

Array Fetch Size on page 92
Batch Mechanism on page 95
Catalog Mode on page 95
Default Buffer Size for Long/LOB Columns (in Kb) on page 100
Enable Cookie Authentication on page 102
Encryption Method on page 104
String Describe Type on page 120

Using security

The driver supports 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 driver supports 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. The Driver for Apache Hive supports both Windows Active Directory Kerberos and MIT Kerberos implementations.

Kerberos authentication

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.

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/.

Summary of authentication-related options

The following tables describe the connection options used for User ID/Password and Kerberos Authentication. The connection options are listed alphabetically by the GUI name that appears on the driver Setup dialog box. The connection string attribute name is listed in parentheses. See "Connection option descriptions" for details about configuring the options.

Table 2: Summary: User ID/Password Authentication Connection Options

Option	Description
User ID/Password Authentication	

Option	Description
Authentication Method (AuthenticationMethod)	Specifies the method the driver uses to authenticate the user to the server when a connection is established. 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 `-1` (No Authentication), 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` User ID/Password
Proxy User (ProxyUser)	Specifies the UserID used for Impersonation and Trusted Impersonation. When impersonation is enabled on the server, this value determines your identity and access rights to files when executing queries. **Default:** None. 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.
User Name (LogonID)	The default user ID that is used to connect to your database. **Default:** None
Additional Properties for Session Cookie Based Authentication (HTTP Mode only)	
Cookie Name (CookieName)	Specifies the name of the cookie used for authenticating HTTP requests when HTTP mode (`TransportMode=1`) and session cookie based authentication are enabled (`EnableCookieAuthentication=1`). **Default:** If no value is specified, the driver attempts to use the following cookie names by default: • `hive.server2.auth` (Hive connections) • `hadoop.auth` (Apache Knox connections) • `JSESSIONID` (Apache Knox connections)
Enable Cookie Authentication (EnableCookieAuthentication)	Determines whether the driver attempts to use session cookie based authentication for requests to an HTTP endpoint after the initial authentication to the server. Cookie based authentication improves response time by eliminating the need to re-authenticate with the server for each request. If set to `0` (Disabled), the driver does not use cookie based authentication for HTTP requests after the initial authentication. If set to `1` (Enabled), the driver attempts to use cookie based authentication for requests to an HTTP endpoint after the initial authentication to the server. The cookie used for authentication is specified by the Cookie Name option. If the name does not match, or authentication fails, the driver attempts to authenticate according to the setting of the Authentication Method option. **Default:** `1` (Enabled)

Table 3: Summary: Kerberos Authentication Connection Options

Option	Description
Authentication Method (AuthenticationMethod)	Specifies the method the driver uses to authenticate the user to the server when a connection is established. 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 `-1` (No Authentication), 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` User ID/Password
GSS Client Library (GSSClient)	The name of the GSS client library that the driver uses to communicate with the Key Distribution Center (KDC). **Default:** `native` (the driver uses the GSS client for Windows Kerberos.)
Service Principal Name (ServicePrincipalName)	The service principal name to be used by driver for Kerberos authentication. **Default:** None.
User Name (LogonID)	The default user ID that is used to connect to your database. **Default:** None

See also

Connection option descriptions on page 87

Connection string examples for configuring authentication

The following connection string configures the Apache Hive 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=DataDirect 8.0 Apache Hive Wire Protocol;HostName=HiveServer;
AuthenticationMethod=4;Database=mydb1;GSSClient=native;PortNumber=10000;LogonID=JohnSmith
```

odbc.ini file examples for configuring authentication

The following example `odbc.ini` file configures the 32-bit Apache Hive 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/ivhivexx.so
Description=DataDirect Apache Hive Wire Protocol driver
AuthenticationMethod=4
Database=Hivedb1
GSSClient=native
HostName=HiveServer
PortNumber=10000
LogonID=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.

Your Progress DataDirect *for* ODBC driver supports Transport Layer Security (TLS) and Secure Sockets Layer (SSL). TLS/SSL are industry-standard protocols for sending encrypted data over database connections. TLS/SSL secures the integrity of your data by encrypting information and providing client/server authentication.

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

TLS/SSL encryption

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

The encryption cipher suite defines the type of encryption that is used for any data exchanged through a TLS/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.

The handshake involves the following types of authentication:

- *TLS/SSL server authentication* requires the server to authenticate itself to the client.
- *TLS/SSL client authentication* is optional and requires the client to authenticate itself to the server after the server has authenticated itself to the client.

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.

Certificates

TLS/SSL encryption 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. Your Progress DataDirect *for* ODBC drivers supports many popular formats. Supported formats include:

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

TLS/SSL server authentication

When the client makes a connection request, the server presents its certificate for the client to accept or deny. The client checks the issuer of the certificate against a list of trusted Certificate Authorities (CAs) whose root certificates reside in one or both of the following stores on the client:

- On Windows operating systems: A permanent storage known as *Windows certificate store*. To learn how to import the required root certificates into the Windows certificate store, see "Importing root certificates into the Windows certificate store."
- On both Windows and non-Windows operating systems: An encrypted file known as *truststore file*. Most truststore files are password-protected. The driver must be able to locate the truststore file and unlock it with the appropriate password. Two connection options are available to the driver to provide this information: Trust Store (Truststore) and Trust Store Password (TruststorePassword).

If the server certificate matches a root certificate in either of the stores, an encrypted connection is established between the client and the server. If the certificate does not match, the connection fails and the client generates an error.

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. Setting the Validate Server Certificate (ValidateServerCertificate) connection option to false allows the driver to accept any certificate returned from the server regardless of whether the issuer of the certificate is a trusted CA.

To configure the driver to use data encryption via TLS/SSL server authentication:

- Set the Host Name (HostName) option to specify the name or the IP address of the server to which you want to connect.
- Set the Port Number (PortNumber) option to specify the port number of the server listener. The default is `10000`.
- Set the Database Name (Database) option to specify the name of the database to which you want to connect.
- Set the Encryption Method (EncryptionMethod) option to `1`.
- Set the Validate Server Certificate (ValidateServerCertificate) option to determine whether the driver validates the certificates sent by the server. When it is set to `1`, the driver validates the certificates. When it is set to `0`, the driver does not validate the certificates.
- Set the Host Name In Certificate (HostNameInCertificate) option to specify the host name that is specified in the Subject of the certificate. 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. Consult your SSL administrator for the correct value.
- Set the Trust Store (Truststore) option to specify either the full path of the truststore file or the contents of the TLS/SSL certificates.

  Note: To allow the client to use TLS/SSL server authentication without storing the truststore file on the disk, you can specify the contents of the root certificates using the Trust Store connection option. Alternatively, you can use the pre-connection attribute, `SQL_COPT_INMEMORY_TRUSTSTORECERT`, to specify the certificate content. For more information, see "Trust Store" and "Using SQL_COPT_INMEMORY_TRUSTSTORECERT".

- Set the Truststore Password (TruststorePassword) option to specify the password that is used to access the truststore file.
- Optionally, set the Enable FIPS (EnableFIPS) connection option to `1` to allow the driver to load the FIPS provider. The FIPS provider contains a set of approved cryptographic algorithms that conform to the Federal Information Processing Standards (FIPS) specified in FIPS 140-2. If you do not specify a value for Enable FIPS, the driver uses its default value (`0`) and loads the default provider.

  Note:

  - The FIPS provider is supported only on the following platforms: Windows 64-bit, Linux 64-bit, and AIX 64-bit.
  - Do not set the Truststore Password connection option when using the FIPS provider. The truststore password uses the PKCS12KDF algorithm, which is not an approved FIPS algorithm. Hence, it must not be specified when using the FIPS provider.
  - For using the FIPS and default providers, the certificates must be generated using the OpenSSL 3.5-compliant cryptographic algorithms. See "Generating TLS/SSL certificates using OpenSSL 3.5-compliant algorithms" for more information.

The following examples show how to configure the driver to establish a connection via user ID/password authentication and use data encryption via TLS/SSL server authentication. In these examples, since `ValidateServerCertificate=1` and `EnableFIPS=1`, the driver validates the certificate sent by the server and the host name specified by the HostNameInCertificate option, and loads the FIPS provider for data encryption.

Connection string

Truststore:

```
DRIVER=DataDirect 8.0 Apache Hive Wire Protocol;
EnableFIPS=1;EncryptionMethod=1;Database=db1;HostName=YourServer;
HostNameInCertificate=MySubjectAltName;PortNumber=10000;
Truststore=TrustStoreName;ValidateServerCertificate=1
```

Note: On Windows, the driver validates the server certificate against the root certificates available in both truststore and Windows certificate store. If a matching certificate is found in either of the stores, the connection is established.

Windows certificate store:

```
DRIVER=DataDirect 8.0 Apache Hive Wire Protocol;
EnableFIPS=1;EncryptionMethod=1;Database=db1;HostName=YourServer;
HostNameInCertificate=MySubjectAltName;PortNumber=10000;
ValidateServerCertificate=1
```

Note: The LogonID and Password options are not required to be stored in the connection string. They can also be sent separately by the application using the SQLConnect ODBC API. For SQLDriverConnect and SQLBrowseConnect, they will need to be specified in the connection string.

odbc.ini

Truststore:

```
Driver=ODBCHOME/lib/ivhivexx.so
Description=DataDirect Apache Hive Wire Protocol driver
...
EnableFIPS=1
...
EncryptionMethod=1
...
Database=db1
...
HostName=YourServer
...
HostNameInCertificate=MySubjectAltName
...
PortNumber=10000
...
Truststore=TrustStoreName
...
ValidateServerCertificate=1
...
```

Note: On Windows, the driver validates the server certificate against the root certificates available in both truststore and Windows certificate store. If a matching certificate is found in either of the stores, the connection is established.

Windows certificate store:

```
Driver=ODBCHOME/lib/ivhivexx.so
Description=DataDirect Apache Hive Wire Protocol driver
...
EnableFIPS=1
...
EncryptionMethod=1
```

```
...
Database=db1
...
HostName=YourServer
...
HostNameInCertificate=MySubjectAltName
...
PortNumber=10000
...
ValidateServerCertificate=1
...
```

Note: The LogonID and Password options are not required to be stored in the data source. They can also be sent separately by the application using the SQLConnect ODBC API. For SQLDriverConnect and SQLBrowseConnect, they will need to be specified in the data source or connection string.

See also

Importing root certificates into the Windows certificate store on page 72
Truststore on page 123
Using SQL_COPT_INMEMORY_TRUSTSTORECERT on page 71
Connection option descriptions on page 87
Generating TLS/SSL certificates using OpenSSL 3.5-compliant algorithms on page 75

Using SQL_COPT_INMEMORY_TRUSTSTORECERT

SQL_COPT_INMEMORY_TRUSTSTORECERT is a pre-connection attribute that specifies the contents of the TLS/SSL certificates for TLS/SSL server authentication. When using SQL_COPT_INMEMORY_TRUSTSTORECERT, the driver stores the certificate content in memory, which eliminates the need to store the truststore file on the disk and lets applications use TLS/SSL server authentication without any disk dependency.

Note: The certificate content can be specified using the Trust Store (Truststore) connection option as well. However, if it is specified using both Trust Store and SQL_COPT_INMEMORY_TRUSTSTORECERT, SQL_COPT_INMEMORY_TRUSTSTORECERT takes precedence over Trust Store.

The following example shows how to specify the contents of 3 certificates using SQL_COPT_INMEMORY_TRUSTSTORECERT:

```
SQLCHAR certificate[] = "
-----BEGIN CERTIFICATE-----12345abc-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----abcd123456-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----aabbcc11----END CERTIFICATE-----";
//The content of each certificate must be specified between -----BEGIN CERTIFICATE-----
 and -----END CERTIFICATE-----. Also, the number of dashes (-----) must be the same
before and after both BEGIN CERTIFICATE and END CERTIFICATE.

...

SQLSetConnectAttr(dbc, SQL_COPT_INMEMORY_TRUSTSTORECERT, (SQLPOINTER)certificate,
SQL_IS_POINTER);

ret = SQLDriverConnect(dbc, NULL,
(SQLCHAR*)"DSN=Apache HiveWP_SSL;UID=jsmith;PWD=secret", SQL_NTS,
NULL, 0, NULL, SQL_DRIVER_NOPROMPT);
```

Importing root certificates into the Windows certificate store

This section provides you with an overview of the steps required to import the required root certificates from a truststore file to the Windows certificate store.

You can import root certificates using either the Certificate Import Wizard or a PowerShell script.

Importing root certificates using Certificate Import Wizard

To import root certificates using Certificate Import Wizard:

1. Double-click the trustore file. The **Certificate Import Wizard** window appears.
2. Select the **Current User** radio button; then, click **Next**.
3. Verify the file path and name available in the **File name** field; then, click **Next**.
4. Enter the password to unlock the truststore file; then, click **Next**.
5. Select the **Automatically select the certificate store based on the type of certificate** radio button; then, click **Next**.
6. Click **Finish**.

The root certificates are imported into the following location in the Windows certificate store: **Certificates** > **Trusted Root Certification Authorities** > **Certificates**.

Note: At times, Windows doesn't trust the imported certificates and imports them into **Certificates** > **Intermediate Certificate Authorities** > **Certificates**. In such cases, manually copy the imported certificates from **Intermediate Certificate Authorities** to **Trusted Root Certification Authorities**.

Importing root certificates using a PowerShell script

To import root certificates using a PowerShell script:

1. Open PowerShell in Administrator mode.
2. Type the following command; then, press **ENTER**.

```
Import-PfxCertificate -Password (ConvertTo-SecureString -String "truststore_password"
 -AsPlainText -Force) -CertStoreLocation Cert:\LocalMachine\Root -FilePath
truststore_filepath
```

where:

`truststore_password`

is the password that is used to access the truststore file.

`truststore_filepath`

is the path to the directory where the truststore file is located.

The root certificates are imported into the following location in the Windows certificate store: **Certificates** > **Trusted Root Certification Authorities** > **Certificates**.

Note: At times, Windows doesn't trust the imported certificates and imports them into **Certificates** > **Intermediate Certificate Authorities** > **Certificates**. In such cases, manually copy the imported certificates from **Intermediate Certificate Authorities** to **Trusted Root Certification Authorities**.

TLS/SSL client authentication

If the server is configured for TLS/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 options are available to the driver to provide this information: Keystore (KeyStore) and Keystore Password (KeyStorePassword). The value of KeyStore is a pathname that specifies the location of the keystore file. The value of Keystore Password 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 option, Key Password (KeyPassword), allows you to specify a password for an individual key.

To configure the driver to use data encryption via TLS/SSL client authentication:

- Set the Host Name (HostName) option to specify the name or the IP address of the server to which you want to connect.
- Set the Port Number (PortNumber) option to specify the port number of the server listener. The default is `10000`.
- Set the Database Name (Database) option to specify the name of the database to which you want to connect.
- Set the Encryption Method (EncryptionMethod) option to `1`.
- Set the Validate Server Certificate (ValidateServerCertificate) option to determine whether the driver validates the certificates sent by the server. When it is set to `1`, the driver validates the certificates. When it is set to `0`, the driver does not validate the certificates.
- Set the Host Name In Certificate (HostNameInCertificate) option to specify the host name that is specified in the Subject of the certificate. 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. Consult your SSL administrator for the correct value.
- Set the Key Store (Keystore) option to specify the location of the keystore file.
- Set the Keystore Password (KeystorePassword) option to specify the password that is used to access the keystore file.
- Optionally, set the Enable FIPS (EnableFIPS) connection option to `1` to allow the driver to load the FIPS provider. The FIPS provider contains a set of approved cryptographic algorithms that conform to the Federal Information Processing Standards (FIPS) specified in FIPS 140-2. If you do not specify a value for Enable FIPS, the driver uses its default value (`0`) and loads the default provider.

Note:

- The FIPS provider is supported only on the following platforms: Windows 64-bit, Linux 64-bit, and AIX 64-bit.
- For using the FIPS and default providers, the certificates must be generated using the OpenSSL 3.5-compliant cryptographic algorithms. See "Generating TLS/SSL certificates using OpenSSL 3.5-compliant algorithms" for more information.
- Do not set the Keystore Password connection option when using the FIPS provider. The keystore password uses the PKCS12KDF algorithm, which is not an approved FIPS algorithm. Hence, it must not be specified when using the FIPS provider.

The following examples show how to configure the driver to establish a connection via user ID/password authentication and use data encryption via TLS/SSL client authentication. In these examples, since `ValidateServerCertificate=1` and `EnableFIPS=1`, the driver validates the certificate sent by the server and the host name specified by HostNameInCertificate, and loads the FIPS provider for data encryption.

Connection string

```
DRIVER=DataDirect 8.0 Apache Hive Wire Protocol;EnableFIPS=1;
EncryptionMethod=1;DatabaseName=Hivedb1;HostName=HiveServer;
HostNameInCertificate=MySubjectAltName;PortNumber=10000;
Keystore=KeyStoreName;ValidateServerCertificate=1;
```

Note: The LogonID and Password options are not required to be stored in the connection string. They can also be sent separately by the application using the SQLConnect ODBC API. For SQLDriverConnect and SQLBrowseConnect, they will need to be specified in the connection string.

odbc.ini

```
Driver=ODBCHOME/lib/ivhivexx.so
Description=DataDirect Apache Hive Wire Protocol driver
...
EnableFIPS=1
...
EncryptionMethod=1
...
DatabaseName=Hivedb1
...
HostName=HiveServer
...
HostNameInCertificate=MySubjectAltName
...
PortNumber=10000
...
KeyStore=KeyStoreName
...
ValidateServerCertificate=1
...
```

Note: The LogonID and Password options are not required to be stored in the data source. They can also be sent separately by the application using the SQLConnect ODBC API. For SQLDriverConnect and SQLBrowseConnect, they will need to be specified in the data source or connection string.

See also

Connection option descriptions on page 87
Generating TLS/SSL certificates using OpenSSL 3.5-compliant algorithms on page 75

Generating TLS/SSL certificates using OpenSSL 3.5-compliant algorithms

For using the OpenSSL 3.5 providers (FIPS and default), the certificates for TLS/SSL encryption must be generated using the OpenSSL 3.5-compliant cryptographic algorithms.

There are multiple ways of generating these certificates. The following commands demonstrate one of them. You can use these commands to generate the certificates and add them to the truststore and keystore files.

Note: The openssl.exe file is required for running these commands. You can download it from the official OpenSSL website.

Note: OpenSSL 3.5.x enforces Security Level 2, which requires all RSA/DSA keys to be at least 2048 bits. To meet these security requirements, certificates must be updated to use RSA keys of 2048 bits or higher. Any certificates that still use 1024-bit keys will be rejected during the SSL/TLS handshake.

For truststore.pfx, every CA certificate must use a 2048-bit or larger public key.

For keystore.pfx, both the private key and the corresponding certificate must be 2048 bits or greater to comply with OpenSSL Security Level 2.

Truststore:

```
openssl.exe pkcs12 -in certificate_name -export -out truststore_filename -nokeys
-keypbe cryptographic_algorithm -certpbe cryptographic_algorithm -password
pass:truststore_password -nomac
```

where:

`certificate_name`

is the name of the certificate you are generating.

`truststore_filename`

is the name of the truststore file.

`cryptographic_algorithm`

is the cryptographic algorithm you are using to generate the certificate.

`truststore_password`

is the password required for accessing the truststore file.

Example:

```
openssl.exe pkcs12 -in nc-thunder-SHA256.cer -export -out truststorepw.pfx -nokeys -keypbe
 AES-256-CBC -certpbe AES-256-CBC -password pass:MyPassW0rd -nomac
```

Keystore:

```
openssl.exe pkcs12 -in certificate_name -inkey privatekey_file -export -out
keystore_file -keypbe cryptographic_algorithm -certpbe cryptographic_algorithm
-nomac
```

where:

`certificate_name`

is the name of the certificate you are generating.

`privatekey_file`

is the name of the file that contains the private key.

`truststore_filename`

is the name of the keystore file.

`cryptographic_algorithm`

is the cryptographic algorithm you are using to generate the certificate.

Example:

```
openssl.exe pkcs12 -in nc-thunder-SHA256.cer -inkey ./file.pem -export -out keystorepw.pfx
 -keypbe AES-256-CBC -certpbe AES-256-CBC -nomac
```

Note: If you are using the Windows certificate store for TLS/SSL encryption, import the certificates generated with the OpenSSL 3.5-compliant algorithms into the store.

Designating an OpenSSL library

Important: Currently, the driver supports version 3.5.6 of the OpenSSL library by default.

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 TLS/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:

- Version 3.5: `ivopenssl.dll` and `ddopenssl.dll`

UNIX/Linux:

- Version 3.5: `ivopenssl.so` and `ddopenssl.so`

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 TLS/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` (TLS/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. See "CryptoLibName" and "SSLLibName" for details.

Note: The CryptoLibName and SSLLibName options must be configured if you are using OpenSSL version 3.0.

See also

CryptoLibName on page 98
SSLLibName on page 119

Apache Knox

Apache Knox is a gateway system that serves as a reverse proxy to Apache Hadoop clusters. The primary advantages of Apache Knox are that it provides a single point of authentication that simplifies security policy enforcement while providing REST API access to a clustered environment. The driver supports connecting to Apache Knox in a similar manner to a standard connection using HTTP mode.

To connect to an Apache Knox gateway:

1. Configure the minimum required options required for a connection:
   - Set the Database Name option to provide the name of the Apache Hive database to which you want to connect.
   - Set the Host Name option to provide the name or IP address of the Apache Knox server to which you want to connect.
   - Set the Port Number option to provide the port of the primary Apache Knox server that is listening for connections. The default for an Apache Knox gateway instance is `8443`.
2. Set the Transport Mode option to `1` (HTTP).
3. Set the HTTP Path option to provide the path of the endpoint to be used for HTTP/HTTPS requests. This value is the Hive endpoint as defined by Apache Knox and corresponds to the name of the server topology file. The default for an Apache Knox gateway is `gateway/default/hive`.
4. Optionally, if your server is configured for SSL, set the Encryption Method option to `1` (SSL) to enable SSL data encryption. Data encryption behavior can be further configured using the connection properties described in "TLS/SSL encryption".
5. Optionally, if your server is configured for Kerberos authentication:
   a) Set the Authentication Method option to `4` (Kerberos).
   b) Set the Service Principal Name option to provide the service principal name for your Apache Knox gateway to be used for Kerberos authentication. The default for an Apache Knox gateway instance is `knox/`*`servername`*`@`*`REALM`*`.COM`. For example, `knox/knoxserver1.example.com@EXAMPLE.COM`.

The following examples demonstrates a basic connection to Apache Knox using Kerberos and SSL data encryption.

Using a connection string:

```
DRIVER=DataDirect 8.0 Apache Hive Wire Protocol;AuthenticationMethod=4;
DatabaseName=hivedb1;EncryptionMethod=1;HostName=HiveServer;
HTTPPath=gateway/default/hive;PortNumber=8443;
ServicePrincipalName=knox/knoxserver1.example.com@EXAMPLE.COM;
TransportMode=1;
```

Using the odbc.ini file:

```
Driver=ODBCHOME/lib/ivhivexx.so
Description=DataDirect Apache Hive Wire Protocol driver
AuthenticationMethod=4
Database=hivedb1
EncryptionMethod=1
HostName=HiveServer
HTTPPath=gateway/default/hive
HostNameInCertificate=MySubjectAltName
```

```
PortNumber=8443
ServicePrincipalName=knox/knoxserver1.example.com@EXAMPLE.COM
TransportMode=1
```

Note: If you receive an `HTTP/1.1 500 Server Error` message when attempting to insert a large number of rows with Apache Knox, reduce the value specified for the ArrayInsertSize property until the operation succeeds.

See also

Database Name on page 100
Host Name on page 106
Port Number on page 116
Transport Mode on page 122
HTTP Path on page 108
Encryption Method on page 104
TLS/SSL encryption on page 67
Authentication Method on page 94
Service Principal Name on page 118
Array Insert Size on page 93

Apache ZooKeeper

Apache ZooKeeper is a centralized service that facilitates the coordination of distributed applications using a simple hierarchal architecture. In addition to supporting centralized administration of configuration information, ZooKeeper offers a number of features out of the box, including naming, synchronization, and group services. ZooKeeper services can be replicated onto server clusters, called ensembles, which allow for a scalable environment with a high-level of availability.

The driver supports Apache ZooKeeper on both 32-bit and 64-bit versions of the following operating systems:

- Windows
- Linux
- AIX

The driver supports retrieving connection information from Apache ZooKeeper services using the methods described in this section.

To retrieve connection information from an Apache ZooKeeper service:

- Set the Zookeeper Discovery option to `1` (Enabled).
- Set the Zookeeper Namespace option to specify the name of the ZooKeeper name space to which you want to retrieve configuration information. The default is `/hiveserver2`.
- Set the Host Name connection option to specify a list of the member servers for the ZooKeeper ensemble to which you want to connect. The value for this option takes the following form:

  `host_name``:`*`port_number`*`|`*`IP_address`*`:`*`port_number`* `[, ...]`

  Note: The driver will return an error if unable to connect to any of the ZooKeeper servers specified with the Host Name option.

The following examples demonstrate a basic connection to Apache ZooKeeper:

Using a connection string:

```
DRIVER=DataDirect 8.0 Apache Hive Wire Protocol;DatabaseName=hivedb1;
       HostName=ZKServer1:2181,255.125.1.11:2818,ZKServer3:2828;
       ZookeeperNamespace=mynamespace1;ZookeeperDiscovery=1;
```

Using the `odbc.ini` file:

```
Driver=ODBCHOME/lib/ivhiveXX.so
Description=DataDirect Apache Hive Wire Protocol driver
Database=hivedb1
HostName=ZKServer1:2181,255.125.1.11:2818,ZKServer3:2828
ZookeeperNamespace=mynamespace1
ZookeeperDiscovery=1
```

One of the primary benefits of ZooKeeper is that it provides centralized management of configuration settings. To take advantage of this feature, the driver transparently retrieves configuration information from the ZooKeeper service at connection. This information is then used to determine the behavior of the driver when establishing the session with the Hive server. However, as a result, settings for certain connection options are overridden by settings provided by the service. The following table describes the type of information typically retrieved by the driver and affected connection options.

Table 4: Configuration information retrieved from Apache ZooKeeper

Retrieved Configuration Information	Overridden Connection Options
Host name	Host Name (HostName)
Port number	Port Number (PortNumber)
Kerberos-related settings	• Authentication Method • Service Principal Name
Encryption-related settings	Encryption Method (EncryptionMethod)
Transport mode related settings	• HTTP Path (HTTPPath) • Transport Mode (TransportMode)

See also

Zookeeper Discovery on page 129
Zookeeper Namespace on page 129
Host Name on page 106

Configuring Apache ZooKeeper for Kerberos authentication

The driver supports Kerberos authentication for Apache ZooKeeper.

To configure Apache ZooKeeper for Kerberos authentication:

- Connect to Apache ZooKeeper by configuring the following connection options.
  - Set the Zookeeper Discovery option to `1` (Enabled).
  - Set the Zookeeper Namespace option to the name of the ZooKeeper name space. The default is `/hiveserver2`.
  - Set the Host Name option to the list of the member servers for the ZooKeeper ensemble to which you want to connect. The value for this option takes the following form:

    `host_name``:`*`port_number`*`|`*`IP_address`*`:`*`port_number`* `[, ...]`

- Enable Kerberos authentication by configuring the following connection options.
  1. Set the Authentication Method option to `4` (Kerberos).
  2. Set the Service Principal Name option to the service principal name for your Apache ZooKeeper to be used for Kerberos authentication. The value for this option takes the following form: `zookeeper/`*`hostname`*`@`*`REALM`*`.COM`.

     For example, `zookeeper/ZKserver1.example.com@EXAMPLE.COM`.

The following examples show how to connect to Apache ZooKeeper using Kerberos authentication.

Using a connection string:

```
DRIVER=DataDirect 8.0 Apache Hive Wire Protocol;DatabaseName=hivedb1;
       HostName=ZKServer1:2181,255.125.1.11:2818,ZKServer3:2828;
       ZookeeperNamespace=mynamespace1;ZookeeperDiscovery=1;
       ServicePrincipalName=zookeeper/ZKserver1.example.com@EXAMPLE.COM;
       AuthenticationMethod=4
```

Using the `odbc.ini` file:

```
Driver=ODBCHOME/lib/ivhivexx.so
Description=DataDirect Apache Hive Wire Protocol driver
Database=hivedb1
HostName=ZKServer1:2181,255.125.1.11:2818,ZKServer3:2828
ZookeeperNamespace=mynamespace1
ZookeeperDiscovery=1
ServicePrincipalName=zookeeper/ZKserver1.example.com@EXAMPLE.COM
AuthenticationMethod=4
```

See also

Apache ZooKeeper on page 79

Authentication Method on page 94
Service Principal Name on page 118

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.

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" for related details.

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

See also

UTF-16 applications on UNIX and Linux on page 46

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, your driver uses only the parameter binding information that is provided by the application.

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`, the driver's batch mechanism emulation 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 apply to Apache Hive:

- No difference between "NULL" and null values

The following restrictions apply to Hive ACID operations:

- Hive ACID operations are supported only on servers that are configured to use them
- Single-table operations for Updates and Deletes are supported only for tables marked as transactional
- Hive ACID operations do not currently support Begin, Commit, or Rollback statements

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.

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.

Packet logging

The driver code includes a packet logging mechanism that allows you to log TCP packets transmitted between your driver and database over the network layer. The logs compiled from can then be analyzed and used to troubleshoot issues. You can enable and configure logging using driver connection options.

Note: The packet logging mechanism is supported only for drivers that transmit TCP packets. Refer to "Packet Logging" in the *Progress DataDirect for ODBC Drivers Reference* for a list of supported drivers.

See the following "Packet Logging Connection options" section for a list of connection options used to configure packet logging.

To enable TCP packet logging:

1. Configure and enable packet logging using one of the following methods:
   - Driver setup dialog (Windows)
   - odbc.ini file (UNIX/Linux)
   - Connection string

   See the following "Configuring and enabling packet logging" section for details.
2. Start your application and reproduce the issue.
3. Stop the application and disable packet logging.
4. Send your logs to Technical Support for analysis. Optionally, you can view your logs using a text editor.

Configuring and enabling packet logging

The following driver configuration methods can be used to enable and configure packet logging. Note that only the `EnablePacketLogging` connection option is required to enable packet logging. If you do not specify values for the other connection options for packet logging, the default behavior is used.

Driver setup dialog (Windows)

You can specify connection options for packet logging in the Extended Options field of the **Advanced** tab. For example:

```
EnablePacketLogging=1;PacketLoggingFilePrefix=C:\temp\myPacketLog;
PacketLoggingMaxFileSize=7500
```

odbc.ini file (UNIX/Linux)

In your data source definition in the [ODBC Data Sources] section of the system information file, you can specify connection options that control packet logging.

```
[Apache Hive Wire Protocol]
Driver=ODBCHOME/lib/ivhive28.so
Description=DataDirect 8.0 Apache Hive Wire Protocol
...
Database=default
...
EnablePacketLogging=1
...
HostName=HiveServer
...
LogonID=JOHN
...
PacketLoggingFilePrefix=/tmp/myPacketLog
...
PacketLoggingMaxFileSize=102400
...
PacketLoggingMaxNumFiles=10
...
Password=secret
...
PortNumber=1000
...
```

Connection string

You can specify connection options that configure packet logging in connection strings.

```
DRIVER=DataDirect 8.0 Apache Hive Wire Protocol;HostName=HiveServer;PortNumber=10000;
LogonID=JOHN;Database=default;EnablePacketLogging=1;
PacketLoggingFilePrefix=C:\temp\myPacketLog;
```

Packet logging connection options

The following table describes the connection options used to configure packet logging.

Table 5: Packet Logging Connection Options

Option	Description
EnablePacketLogging	If set to 0, packet logging is disabled. This is the default. If set to 1, packet logging is enabled. If set to 2, packet logging is enabled, but the generated log file does not contain packet data. This value is typically used for performance testing. (Windows only) If set to 5, packet logging and ODBC tracing are enabled. If set to 6, packet logging and ODBC tracing are enabled, but the log file for packet logging does not contain data.

Option	Description
PacketLoggingFlush	If set to 0, the operating system determines when to write the log content stored in memory to disk. This is the default. If set 1, the driver determines when to write the log content stored in memory to disk. If set to 2, the content of memory is written to a the log file after each write. This setting provides a more complete logging history in the event of a crash, but can incur a performance penalty.
PacketLoggingFilePrefix	Specifies the path and prefix name of the log file. If no path is specified, the trace log resides in the working directory of the application you are using. For example: • /tmp/myLogFile (UNIX/Linux) • C:\temp\myLogFile (Windows) The above examples would generate a file named myLogFile*YYYYMMDDhhmmssxxx_nn*.out in the temp directory. If you do not specify a value for this option, the driver creates log files in the working directory using the following form: pkt*YYYYMMDDhhmmssxxx_nn*.out.
PacketLoggingMaxFileSize	Specifies the file size limit (in KB) of the log file. Once this file size limit is reached, a new log file is created and logging continues. The default is 102400. Note that subsequent files are named by appending sequential numbers, starting at 1, to the end of the original file name, for example, myLog<*timestamp*>_1.out, myLog<*timestamp*>_2.out, and so on.
PacketLoggingMaxNumFiles	Specifies the maximum number of log files that can be created. The default is 10. Once the maximum number of log files is created, the logging mechanism reopens the first file in the sequence, deletes the content, and continues logging in that file until the file size limit is reached, after which it repeats the process with the next file in the sequence.
PacketLoggingMemBuffSize	Specifies the maximum amount of memory, in kiloybtes, to use when writing packet logging. The default is 1024.

5

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.

Note: The driver does not support specifying values for the same connection option multiple times in a connection string or DSN. If a value is specified using the same attribute multiple times or using both long and short attributes, the connection may fail or the driver may not behave as intended.

The following table lists the connection string attributes supported by the driver.

Table 6: Attribute Names for the Driver for Apache Hive

Attribute (Short Name)	Default
ArraySize (AS) **Note:** ArraySize has been replaced by ArrayFetchSize.	None
ArrayFetchSize (AFS)	`150000`
ArrayInsertSize (AIS)	`16384`

Attribute (Short Name)	Default
AuthenticationMethod (AM)	`0` (User ID/Password)
BatchMechanism (BM)	`2` (MultiRowInsert)
CatalogMode (CM)	`0` (Mixed)
CookieName (CN)	If no value is specified, the driver attempts to use the following cookie names by default: • `hive.server2.auth` (Hive connections) • `hadoop.auth` (Apache Knox connections) • `JSESSIONID` (Apache Knox connections)
CryptoLibName (CLN)	Empty string
CryptoProtocolVersion (CPV)	`TLSv1.2, TLSv1.1, TLSv1`
Database (DB)	`default`
DataSourceName (DSN)	None
DefaultLongDataBuffLen (DLDBL)	`1024`
Description (n/a)	None
Enable Cookie Authentication (ECA)	`1` (Enabled)
EnableDescribeParam (EDP)	`0` (Disabled)
EnableFIPS (EF)	`0` (Default provider)
EncryptionMethod (EM)	`0` (No Encryption)
GSSClient (GSSC)	`native`
HostName (HOST)	None
HostNameInCertificate (HNIC)	None
HTTPPath (HP)	`cliservice`
KeepAlive (KA)	`0` (Disabled)
KeyPassword (KP)	None
Keystore (KS)	None
KeystorePassword (KSP)	None
LoginTimeout (LT)	`30`

Attribute (Short Name)	Default
LogonID (UID)	None
MaxStringSize (MSS)	2147483647
MinLongVarcharSize (MINLVS)	None. If no value is specified, the driver will not change the column size reported for SQL_LONGVARCHAR columns.
OpenSSLConfigFile (OSSLCNF)	*install_dir*\drivers\openssl.cnf (Windows) *install_dir*/lib/openssl.cnf (UNIX/Linux)
OpenSSLProviderPath (OSSLPP)	*install_dir*\drivers (Windows) *install_dir*/lib (UNIX/Linux)
Password (PWD)	None
PortNumber (PORT)	10000
ProxyUser (PU)	None
RemoveColumnQualifiers (RCQ)	0 (Disabled)
ServicePrincipalName (SPN)	None
SSLLibName (SLN)	Empty string
StringDescribeType (SDT)	-9 (SQL_WVARCHAR)
TransactionMode (TM)	0 (No Transactions)
TransportMode (TRM)	0 (binary)
Truststore (TS)	None
TruststorePassword (TSP)	None
UseCurrentSchema (UCS)	0 (Disabled)
UseNativeCatalogFunctions (UNCF) **Note:** UseNativeCatalogFunctions has been replaced by CatalogMode.	None
UseUnicodeCharTypes (UUCT)	1
ValidateServerCertificate (VSC)	1 (Enabled)

Attribute (Short Name)	Default
VarcharThreshold (VT)	None. If no value is specified, the driver will not change the described type for SQL_VARCHAR columns.
ZookeeperDiscovery (ZKD)	`0` (Disabled)
ZookeeperNamespace (ZKN)	`/hiveserver2`

For details, see the following topics:

- Array Size
- Array Fetch Size
- Array Insert Size
- Authentication Method
- Batch Mechanism
- Catalog Mode
- Cookie Name
- Crypto Protocol Version
- CryptoLibName
- Data Source Name
- Database Name
- Default Buffer Size for Long/LOB Columns (in Kb)
- Description
- Enable Cookie Authentication
- Enable FIPS
- Enable SQLDescribeParam
- Encryption Method
- GSS Client Library
- Host Name
- Host Name In Certificate
- HTTP Path
- IANAAppCodePage
- Key Password
- Keystore
- Keystore Password
- Login Timeout

Array Size

Attribute

ArraySize (AS)

Purpose

Note: The Array Size option has been renamed Array Fetch Size. The ArraySize attribute will continue to be supported for this release, but will be deprecated in subsequent versions of the product. If values are specified for both options, the value for Array Fetch Size will take precedent.

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

None

GUI Tab

Advanced tab

See Also

- Performance considerations on page 62

Array Fetch Size

Attribute

ArrayFetchSize (AFS)

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

`150000`

GUI Tab

Advanced tab

See Also

- Performance considerations on page 62

Array Insert Size

Attribute

ArrayInsertSize (AIS)

Purpose

Specifies the maximum buffer size, in KB, the driver uses for a packet when executing a multi-row insert. By limiting the packet size, the driver can avoid out of memory errors when executing inserts that require large amounts of memory, while continuing to provide improved performance when executing inserts that are smaller in size.

In most scenarios, the default setting provides the ideal driver behavior; however, you may need to reduce the value specified if you encounter either of the following:

- Performance or memory issues when inserting larger values.
- The following error when inserting larger values while using Apache Knox: `HTTP/1.1 500 Server Error`.

Valid Values

x

where:

x

is a positive integer representing the maximum buffer size in KB.

Notes

- You can improve performance by increasing the value specified for this option; however, if the value 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

`16384` (KB)

GUI Tab

Advanced tab

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

- Performance considerations on page 62

Catalog Mode

Attribute

CatalogMode (CM)

Purpose

Specifies whether the driver uses native catalog functions to retrieve information returned by the SQLTables, SQLColumns, and SQLStatistics catalog functions.

Valid Values

`0` | `1` | `2`

Behavior

If set to `0` (Mixed), the driver uses a combination of ODBC catalog functions and driver-discovered information to retrieve catalog information. Select this option for the optimal balance of performance and accuracy.

Note: In this setting, the driver uses the best techniques for retrieving information. These techniques vary depending on the server used, which may result in differences in performance.

If set to `1` (Native), the driver uses native catalog functions to retrieve information returned by the SQLTables, SQLColumns, and SQLStatistics catalog functions. This setting provides the best performance, but at the expense of less-accurate catalog information.

If set to 2 (Query Based), the driver uses driver-discovered information to retrieve catalog information. This option provides highly accurate catalog information, but at the expense of slower performance.

Default

`0` (Mixed)

GUI Tab

Advanced tab

See Also

- Performance considerations on page 62

Cookie Name

Attribute

CookieName (CN)

Purpose

Specifies the name of the cookie used for authenticating HTTP requests when HTTP mode (`TransportMode=1`) and session cookie based authentication are enabled (`EnableCookieAuthentication=1`). When preparing an HTTP request to the server, the driver will not attempt to reauthenticate if a valid cookie is present.

Valid Values

`string`

where:

`string`

is a valid cookie name.

Default

If no value is specified, the driver attempts to use the following cookie names by default:

- `hive.server2.auth` (Hive connections)
- `hadoop.auth` (Apache Knox connections)
- `JSESSIONID` (Apache Knox connections)

Notes

- If the driver is operating in binary mode (`TransportMode=0`) or cookie based authentication (`EnableCookieAuthentication=0`) is disabled, this option is ignored.

GUI Tab

General tab

See also

- Transport Mode on page 122
- Enable Cookie Authentication on page 102
- Authentication Method on page 94

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` | `TLSv1.3`

Example

If your security environment is configured to use TLSv1.2 and TLSv1.3, specify the following values:

```
CryptoProtocolVersion=TLSv1.2, TLSv1.3
```

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 104

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 cryptograpic 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\ODBC\Drivers\ddopenssl30.dll
```

Notes

- **Warning**: If you are distributing the driver with your application, you must prevent your end users from setting the value for the CryptoLibName option. The CryptoLibName option provides a method for you to specify a cryptographic library file used for TLS/SSL encryption. However, if exposed, the option can be used to specify files that execute malicious or undesirable code. Refer to "Security best practices for ODBC applications" in the *Progress DataDirect for ODBC Drivers Reference* for more information.
- The value specified for this option should be an absolute path to a mounted drive.
- 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.
- This option must be configured if you are using OpenSSL version 3.0.

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\ODBC\drivers\ddopenssl30.dll;
```

See also

- Advanced tab on page 51
- SSLLibName on page 119

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 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

- Performance considerations on page 62

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 Cookie Authentication

Attribute

EnableCookieAuthentication (ECA)

Purpose

Determines whether the driver attempts to use session cookie based authentication for requests to an HTTP endpoint after the initial authentication to the server. Cookie based authentication improves response time by eliminating the need to re-authenticate with the server for each request.

Valid Values

`0` | `1`

Behavior

If set to `0` (Disabled), the driver does not use cookie based authentication for HTTP requests after the initial authentication.

If set to `1` (Enabled), the driver attempts to use cookie based authentication for requests to an HTTP endpoint after the initial authentication to the server. The cookie used for authentication is specified by the Cookie Name option. If the name does not match, or authentication fails, the driver attempts to authenticate according to the setting of the Authentication Method option.

Notes

- If HTTP mode is disabled (`TransportMode=0`), this option is ignored.

Default

`1` (Enabled)

GUI Tab

General tab

See also

- Transport Mode on page 122
- Cookie Name on page 96
- Authentication Method on page 94

Enable FIPS

Attribute

EnableFIPS (EF)

Purpose

Determines whether the OpenSSL library uses cryptographic algorithms from the FIPS provider or the default provider when TLS/SSL encryption is enabled.

Valid Values

`0` | `1`

Behavior

If set to `0`, the OpenSSL library uses cryptographic algorithms from the default provider.

If set to `1`, the OpenSSL library uses cryptographic algorithms from the FIPS provider.

Notes

- The FIPS provider is supported only on the following platforms: Windows 64-bit, Linux 64-bit, and AIX 64-bit. On the other platforms, the driver uses the default provider of the OpenSSL 3.5 library.
- Do not set the Truststore Password (TruststorePassword) connection option when using the FIPS provider. The truststore password uses the PKCS12KDF algorithm, which is not an approved FIPS algorithm. Hence, it must not be specified when using the FIPS provider.
- For using the FIPS and default providers, the certificates must be encrypted with the OpenSSL 3.5-compliant cryptographic algorithms. See "Generating TLS/SSL certificates with OpenSSL 3.5-compliant algorithms" for more information.

Default

`0`

See also

TLS/SSL server authentication on page 68
TLS/SSL client authentication on page 73
Generating TLS/SSL certificates using OpenSSL 3.5-compliant algorithms on page 75

Enable SQLDescribeParam

Attribute

EnableDescribeParam (EDP)

Purpose

Determines whether the driver uses the SQLDescribeParam function, which describes parameters as a data type of SQL_VARCHAR and SQL_WVARCHAR 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 and SQL_WVARCHAR 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.
- To configure the driver to use HTTPS end points, set `TransportMode=1` (HTTP) and `EncryptionMethod=1` (SSL).
- When using FIPS and default providers, the certificates must be generated using the OpenSSL 3.5-compliant cryptographic algorithms.

Default

0 (No Encryption)

GUI Tab

Security tab

See also

- Performance considerations on page 62

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.

Notes

- **Warning**: If you are distributing the driver with your application, you must prevent your end users from setting the value for the GSS Client Library option. The GSS Client Library option provides a method for you to specify a library file used to communicate with the Key Distribution Center (KDC). However, if exposed, the option can be used to specify files that execute malicious or undesirable code. Refer to "Security best practices for ODBC applications" in the *Progress DataDirect for ODBC Drivers Reference* for more information.
- The value specified for this option should be an absolute path to a mounted drive.

Default

`native`

GUI Tab

Security tab

Host Name

Attribute

HostName (HOST)

Purpose

Specifies the name or the IP address of the server to which you want to connect. When Apache ZooKeeper support is enabled (`ZooKeeperDiscovery=1`), this option specifies a list of member servers of the ZooKeeper ensemble to which you want to connect.

Valid Values for a Single-Server Connection

`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.

Valid Values for a ZooKeeper Connection (`ZooKeeperDiscovery=1`)

`host_name``:`*`port_number`* | *`IP_address`*`:`*`port_number`* `[, ...]`

where:

`hostname`

is the name of an ensemble server to which you want to connect.

`port_number`

is the port number of the server listener.

`IP_address`

is the IP address of the server to which you want to connect.

For example:

```
server1:10000,255.125.1.11:2818,server3:2828
```

Notes

- When Zookeeper support is enabled, the driver will return an error if unable to connect to any of the ZooKeeper servers specified with this option.

Default

None

GUI Tab

General tab

See also

- Zookeeper Discovery on page 129
- Apache ZooKeeper on page 79

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 compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If the certificate does not have a SubjectAlternativeName, the driver compares the host name with the Common Name (CN) part of the certificate. If the values do not match, the connection fails and the driver throws an exception.

If `#SERVERNAME#` is specified, the driver compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If the certificate does not have a SubjectAlternativeName, the driver compares the host name to the CN part of the certificate's Subject name. If the values do not match, the connection fails and the driver throws an exception. If multiple CN parts are present, the driver validates the host name against each CN part. If any one validation succeeds, a connection is established.

Default

None

GUI Tab

Security tab

HTTP Path

Attribute

HTTPPath (HP)

Purpose

Specifies the path component of the HTTP/HTTPS endpoint used for connections when HTTP mode is enabled (`TransportMode=1`).

Valid Values

`string`

where:

`string`

is the path component of the URL endpoint. By default, the value specified must be an HTTP end point. To support HTTPS values, enable SSL using the Encryption Method option (`EncryptionMethod=1`).

Example

If your server was listening to the following URL:

```
https://myserver:10000/cliservice/
```

Then the following value should be specified for HTTPPath:

```
cliservice
```

Notes

- This option is ignored when HTTP mode is disabled (`TransportMode=0`).

Default

```
cliservice
```

GUI Tab

General tab

See Also

- Transport Mode on page 122
- Encryption Method on page 104
- HTTP mode on page 61

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.

Default

`4` (ISO 8559-1 Latin-1)

GUI Tab

NA

See Also

Refer to "Internationalization, localization, and Unicode" in the *Progress DataDirect for ODBC Drivers Reference* for details.

Key Password

Attribute

KeyPassword (KP)

Purpose

Specifies the password used to access the individual keys in the keystore file when SSL is enabled (`EncryptionMethod=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 (`EncryptionMethod=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

- **Warning**: If you are distributing the driver with your application, you must prevent your end users from setting the value for the Keystore option. The Keystore option provides a method for you to specify a keystore file used for TLS/SSL encryption. However, if exposed, the option can be used to specify files that execute malicious or undesirable code. Refer to "Security best practices for ODBC applications" in the *Progress DataDirect for ODBC Drivers Reference* for more information.
- The value specified for this option should be an absolute path to a mounted drive.

- 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 String Size

Attribute

MaxStringSize (MSS)

Purpose

Specifies the maximum size of columns of the String data type 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 columns defined as the String data type.

Notes

- Microsoft Access and Tableau users must configure this option to the following values:
  - For Microsoft Access, specify a value of 255.
  - For Tableau, specify a value from 255 to 4000 that suits your environment.
- The value specified for the String Describe Type connection option determines whether the String data type maps to the SQL_WVARCHAR or SQL_WLONGVARCHAR ODBC data types.
- Max String Size replaces the Max Varchar Size connection option.

Default

2147483647

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_(W)LONGVARCHAR. If the size of a SQL_(W)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_(W)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_(W)LONGVARCHAR type.

Notes

- Configuring the VarcharThreshold and MinLongVarcharSize options allows you to fetch SQL_(W)VARCHAR and SQL_(W)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_(W)LONGVARCHAR columns.

GUI Tab
Advanced tab

See also

- Varchar Threshold on page 128

OpenSSLConfigFile

Attribute

OpenSSLConfigFile (OSSLCNF)

Purpose

Specifies the absolute path to the configuration file required to load the FIPS provider when the driver is configured to use OpenSSL with FIPS provider for TLS/SSL encryption (`EnableFIPS=1`).

Valid Values

`fips_config_file`

where:

`fips_config_file`

is the absolute path to the configuration file. For example: `/opt/Progress/DataDirect/ODBC/lib/openssl.cnf`.

Notes

- The OpenSSLConfigFile option is not available on the setup dialog box. To set a value for it, use the Extended Options connection option, which is available on the Advanced tab of the setup dialog box.

Default

- *`install_dir`*`\drivers\openssl.cnf` (Windows)
- *`install_dir`*`/lib/openssl.cnf` (UNIX/Linux)

OpenSSLProviderPath

Attribute

OpenSSLProviderPath (OSSLPP)

Purpose

Specifies the path to the directory that contains the provider library when TLS/SSL encryption is enabled.

Valid Values

provider_path

where:

provider_path

is the path to the directory that contains the provider library.

Notes

- The OpenSSLProviderPath option is not available on the setup dialog box. To set a value for it, use the Extended Options connection option, which is available on the Advanced tab of the setup dialog box.

Default

- *install_dir*\drivers (Windows)
- *install_dir*/lib (UNIX/Linux)

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

Logon dialog

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

10000

GUI Tab

General tab

Proxy User

Attribute

ProxyUser (PU)

Purpose

Specifies the UserID used for Impersonation and 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

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` (disabled), 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

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, Cloudera Data Platform 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:

```
yourservicename/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

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 TLS/SSL library contains the implementations of TLS/SSL protocols the driver uses for data encryption.

This option allows you to designate a different TLS/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\Drivers\ddopenssl30.dll
```

Notes

- **Warning**: If you are distributing the driver with your application, you must prevent your end users from setting the value for the SSLLibName option. The SSLLibName option provides a method for you to specify an OpenSSL library file used for SSL encryption. However, if exposed, the option can be used to specify files that execute malicious or undesirable code. Refer to "Security best practices for ODBC applications" in the *Progress DataDirect for ODBC Drivers Reference* for more information.
- The value specified for this option should be an absolute path to a mounted drive.
- 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.
- This option must be configured if you are using OpenSSL version 3.0.

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\ODBC\Drivers\ddopenssl30.dll;
```

See also

- Advanced tab on page 51
- CryptoLibName on page 98

String Describe Type

Attribute

StringDescribeType (SDT)

Purpose

Specifies the data type used to describe String columns. This connection option affects functions that return column-related data, such as SQLColumns, SQLDescribeCol, and SQLColAttributes. It does not affect SQLGetTypeInfo.

Valid Values

`-10` | `-9` | `-1` | `12`

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.

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

-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

Transport Mode

Attribute

TransportMode (TRM)

Purpose

Specifies whether binary (TCP) mode or HTTP mode is used to access Apache Hive data sources.

Valid Values

0 | 1

Behavior

If set to 0 (binary), Thrift RPC requests are sent directly to data sources using a binary connection (TCP mode).

If set to 1 (HTTP), Thrift RPC requests are sent using HTTP transport (HTTP mode). HTTP mode is typically used when connecting to a proxy server, such as a gateway, for improved security, or a load balancer.

Notes

- The setting of this option corresponds to that of the hive.server2.transport.mode property in your `hive-site.xml` file.
- When HTTP mode is enabled (`TransportMode=1`), the HTTP/HTTPS end point for the Hive server must be specified using the HTTP Path option.
- To configure the driver to use HTTPS end points, set `TransportMode=1` (HTTP) and `EncryptionMethod=1` (SSL).
- Apache Hive currently supports using only one protocol mode per server at a time.

Default

0 (binary)

GUI Tab

General tab

See also

- Encryption Method on page 104
- HTTP Path on page 108
- HTTP mode on page 61

Truststore

Attribute

Truststore (TS)

Purpose

Specifies either the path and file name of the truststore file or the contents of the TLS/SSL certificates to be used when SSL is enabled (`Encryption Method=1`) and server authentication is used.

Valid Values

truststore_directory`\`*filename* | `data://-----BEGIN CERTIFICATE-----`*certificate_content*`-----END CERTIFICATE-----`

where:

`truststore_directory`

is the path to the directory where the truststore file is located.

`filename`

is the file name of the truststore file.

`certificate_content`

is the content of the TLS/SSL certificate.

Notes

- **Warning**: If you are distributing the driver with your application, you must prevent your end users from setting the value for the Truststore option. The Truststore option provides a method for you to specify a truststore file used for TLS/SSL encryption. However, if exposed, the option can be used to specify files that execute malicious or undesirable code. Refer to "Security best practices for ODBC applications" in the *Progress DataDirect for ODBC Drivers Reference* for more information.
- The value specified for this option should be an absolute path to a mounted drive.

- If you do not specify the path to the directory that contains the truststore file, the current directory is used for authentication.
- The keystore and truststore files may be the same file.
- When specifying content for multiple certificates, secify the content of each certificate between `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----`. For example:

  ```
  -----BEGIN CERTIFICATE-----certificatecontent1-----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----certificatecontent2-----END CERTIFICATE-----
  -----BEGIN CERTIFICATE-----certificatecontent3-----END CERTIFICATE-----
  ```

  Note that the number of dashes (`-----`) must be the same before and after both `BEGIN CERTIFICATE` and `END CERTIFICATE`.
- When specifying the certificate content for authentication, do not specify the truststore password. Since the truststore file is not required to be stored on the disk when the certificate content is specified directly, the driver need not unlock its contents.
- The Trust Store field on the Driver setup dialog supports content up to 8192 characters in length. For specifying certificate content longer than 8192 characters, edit the registry and manually add the entry to the DSN.
- On Windows platforms, if the required certificates are available in the Windows certificate store, the Trust Store and Truststore Password options need not be used.

Default

No default value

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

Use Native Catalog Functions

Attribute

UseNativeCatalogFunctions (UNCF)

Purpose

Note: The Use Native Catalog Functions option has been replaced by Catalog Mode. The UseNativeCatalogFunctions attribute will continue to be supported for this release, but will be deprecated in subsequent versions of the product.

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

None

GUI Tab

None

Use Unicode Char Types

Attribute

UseUnicodeCharTypes (UUCT)

Purpose

Determines whether Char and Varchar columns are described as SQL_CHAR and SQL_VARCHAR types or SQL_WCHAR and SQL_WVARCHAR types. This connection option affects functions that return column-related data, such as SQLColumns, SQLDescribeCol, and SQLColAttributes. It does not affect SQLGetTypeInfo.

Valid Values

`0` | `1`

Behavior

If set to `0` (disabled), Char columns are described as SQL_CHAR and Varchar columns are described as SQL_VARCHAR.

If set to `1` (enabled), Char columns are described as SQL_WCHAR and Varchar columns are described as SQL_WVARCHAR.

Default

`1`

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 Truststore and Truststore Password options.

Notes

- Truststore information is specified using the Truststore and Truststore Password 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_(W)VARCHAR as SQL_(W)LONGVARCHAR. If the size of the SQL_(W)VARCHAR column exceeds the value specified, the driver will describe the column as SQL_(W)LONGVARCHAR when calling SQLDescribeCol and SQLColumns. This option allows you to fetch columns that would otherwise exceed the upper limit of the SQL_(W)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_(W)VARCHAR.

Notes

- Configuring the VarcharThreshold and MinLongVarcharSize options allows you to fetch SQL_(W)VARCHAR and SQL_(W)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_(W)VARCHAR columns.

GUI Tab

Advanced tab

See also

- Min Long Varchar Size on page 113

Zookeeper Namespace

Attribute

ZookeeperNamespace (ZKN)

Purpose

Specifies the name of the Apache ZooKeeper name space from which you want to retrieve configuration information. The driver uses this information to determine its behavior for the connection. Settings retrieved from the service take precedence over connection property settings. For a list of affected properties, see "Apache ZooKeeper."

Valid Values

string

where:

string

is a valid name of a ZooKeeper name space.

Default

`/hiveserver2`

Notes

- If support for ZooKeeper is disabled (`ZookeeperDiscovery=0`), this option is ignored.

GUI Tab

General tab

See also

- Zookeeper Discovery on page 129
- Apache ZooKeeper on page 79

Zookeeper Discovery

Attribute

ZookeeperDiscovery (ZKD)

Purpose

Determines whether the driver uses Apache ZooKeeper when connecting to a database server.

Valid Values

`0 | 1`

Behavior

If set to `0` (Disabled), the driver does not use ZooKeeper when connecting to a database server. By default, the driver's behavior is determined by the connection options settings.

If set to `1` (Enabled), the driver attempts to connect to the member servers of a ZooKeeper ensemble that are specified by the Host Name connection option. At connection, the driver retrieves configuration information from the ZooKeeper service that determines the behavior of the driver for the connection. The retrieved configuration information takes precedent over any values specified using connection options. For additional information, see "Apache ZooKeeper."

Notes

- In addition to enabling this option (`Zookeeper Discovery=1`), you must provide the following to retrieve configuration information from a ZooKeeper service:
  - Using the Host Name connection option, specify the names (or addresses) and port numbers of the member servers of the ZooKeeper ensemble to which you want to connect. They must be specified using the following format:

    `host_name``:`*`port_number`* `|` *`IP_address`*`:`*`port_number`* `[, ...]`

    See "Host Name" for details.
  - Using the Zookeeper Namespace option, specify the name of the ZooKeeper name space to which you want to retrieve configuration information.

Default

`0` (Disabled)

GUI Tab

General tab

See also

- Apache ZooKeeper on page 79
- Host Name on page 106
- Zookeeper Namespace on page 129

6

SQL functionality

The driver supports 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.

For details, see the following topics:

- Data Definition Language (DDL)
- Selecting Data With the Driver
- From Clause
- Group By Clause
- Having Clause
- Order By Clause
- For Update Clause
- Set Operators
- Subqueries
- SQL Expressions
- Restrictions

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	Driver for Apache Hive Wire HiveQL Translation
`SELECT * FROM t1, t2 WHERE a = b`	`SELECT * FROM t1 t1 JOIN t2 t2 WHERE a = b`
`SELECT * FROM t1 y, t2 x WHERE a = b`	`SELECT * FROM t1 y JOIN t2 x WHERE a = b`
`SELECT * FROM t2, (SELECT * FROM t1) x`	`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.
- 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.

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.
- INTERSECT is not supported.
- 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
- Numeric Operators

- Character Operator
- Relational Operators
- Logical Operators
- Functions

See also

Constants on page 135
Numeric Operators on page 135
Character Operator on page 136
Relational operators on page 136
Logical Operators on page 136
Functions on page 137

Constants

The driver supports literal values.

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 7: 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 8: Relational Operators Supported with Apache Hive

Entry SQL Level Operator	Support in HiveQL
<>	Supported
<	Supported
<=	Supported
=	Supported
<=>	Supported
>	Supported
>=	Supported
IS [NOT] NULL	Supported
[NOT] BETWEEN x AND y	Supported
[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 9: 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 10: Set Functions Supported

Set Function	Support in HiveQL
Count	Supported
AVG	Supported
MIN	Supported
MAX	Supported
SUM	Supported
DISTINCT	Supported
ALL	Supported

Table 11: 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 12: 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 13: 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 14: 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.

Merge Restrictions

Apache Hive does not support the Merge statement.

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.

The ACID operations have the following restrictions:

- Hive ACID operations are supported only on servers that are configured to use them
- Single-table operations for Updates and Deletes are supported only for tables marked as transactional
- Hive ACID operations do not currently support Begin, Commit, or Rollback statements