CLOUDERA

Cloudera ODBC Driver for Apache Hive

Important Notice

Cloudera, Inc.
1001 Page Mill Road, Building 2
Palo Alto, CA 94304-1008
info@cloudera.com
US: 1-888-789-1488
Intl: 1-650-843-0595
www.cloudera.com

Release Information

Version: 2.9

Date: November 2025

Contents

About the Cloudera ODBC Driver for Apache Hive

The Cloudera ODBC Driver for Apache Hive is used for direct SQL and HiveQL access to Apache Hadoop / Hive distributions, enabling Business Intelligence (BI), analytics, and reporting on Hadoop / Hive-based data. The connector efficiently transforms an application's SQL query into the equivalent form in HiveQL, which is a subset of SQL-92. If an application is Hive-aware, then the connector is configurable to pass the query through to the database for processing. The connector interrogates Hive to obtain schema information to present to a SQL-based application. Queries, including joins, are translated from SQL to HiveQL. For more information about the differences between HiveQL and SQL, see SQL Connector for HiveQL.

The Cloudera ODBC Driver for Apache Hive complies with the ODBC 3.80 data standard and adds important functionality such as Unicode and 32- and 64-bit support for high-performance computing environments.

ODBC is one of the most established and widely supported APIs for connecting to and working with databases. At the heart of the technology is the ODBC connector, which connects an application to the database. For more information about ODBC, see: https://insightsoftware.com/blog/what-is-odbc/. For complete information about the ODBC specification, see the *ODBC API Reference* from the Microsoft documentation: https://docs.microsoft.com/en-us/sql/odbc/reference/syntax/odbc-api-reference.

Note: The AIX and Solaris connectors are not available through the Cloudera website. To get these connectors, contact the Sales & Solutions team:

- Phone number: +1.604.633.0008 ext 2
- Email: solutions@simba.com

Note: This is the most up-to-date version of the *Installation and Configuration Guide*, for use with version <Version Number> of the connector. If you are using an older version of the connector, certain features may not be available and certain settings may behave in unexpected ways. Please consult the PDF version of the *Installation and Configuration Guide* that was installed with your connector.

The *Installation and Configuration Guide* is suitable for users who are looking to access data residing within Hive from their desktop environment. Application developers might also find the information helpful. Refer to your application for details on connecting via ODBC.

Note: For basic configuration instructions that allow you to quickly set up the Windows connector so that you can evaluate and use it, see the *Simba ODBC Connectors Quick Start Guide for Windows*. The Quick Start Guide also explains how to use the connector in various applications.

Platform and Data source version support

The Cloudera ODBC Driver for Apache Hive supports Windows, macOS, and Linux operating systems. For detailed information on supported operating systems and data source versions, please refer to the connector's release notes.

Windows Connector

This section provides an overview of the Connector in the Windows platform, outlining the required system specifications and the steps for installing and configuring the connector in Windows environments.

Windows System Requirements

Install the connector on client machines where the application is installed. Before installing the connector, make sure that you have the following:

- Administrator rights on your machine.
- 100 MB of available disk space
- Visual C++ Redistributable for Visual Studio 2022 installed (with the same bitness as the connector that you are installing).
  You can download the installation packages at https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170

Installing the Connector in Windows

On 64-bit Windows operating systems, you can execute both 32-bit and 64-bit applications. However, 64-bit applications must use 64-bit connectors, and 32-bit applications must use 32-bit connectors. Make sure that you use a connector whose bitness matches the bitness of the client application:

- `ClouderaHiveODBC32.msi` for 32-bit applications
- `ClouderaHiveODBC64.msi` for 64-bit applications

You can install both versions of the connector on the same machine.

To install the Cloudera ODBC Driver for Apache Hive in Windows:

1. Depending on the bitness of your client application, double-click to run **ClouderaHiveODBC32.msi** or **ClouderaHiveODBC64.msi**.
2. Click **Next**.
3. Select the check box to accept the terms of the License Agreement if you agree, and then click **Next**.
4. To change the installation location, click **Change**, then browse to the desired folder, and then click **OK**. To accept the installation location, click **Next**.
5. Click **Install**.
6. When the installation completes, click **Finish**.

Creating a Data Source Name in Windows

Typically, after installing the Cloudera ODBC Driver for Apache Hive, you need to create a Data Source Name (DSN). A DSN is a data structure that stores connection information so that it can be used by the connector to connect to Hive.

Alternatively, you can specify connection settings in a connection string or as connector-wide settings. Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

The following instructions describe how to create a DSN. For information about specifying settings in a connection string, see Using a Connection String. For information about connector-wide settings, see Configuring a DSN-less Connection in Windows.

To create a Data Source Name in Windows:

1. From the Start menu, go to **ODBC Data Sources**.

   > **Note:** Make sure to select the ODBC Data Source Administrator that has the same bitness as the client application that you are using to connect to Hive.

2. In the ODBC Data Source Administrator, click the **Drivers** tab, and then scroll down as needed to confirm that the Cloudera ODBC Driver for Apache Hiveappears in the alphabetical list of ODBC Connectors that are installed on your system.
3. Choose one:
   - To create a DSN that only the user currently logged into Windows can use, click the **User DSN** tab.
   - Or, to create a DSN that all users who log into Windows can use, click the **System DSN** tab.

     > **Note:** It is recommended that you create a System DSN instead of a User DSN. Some applications load the data using a different user account, and might not be able to detect User DSNs that are created under another user account.

4. Click **Add**.
5. In the Create New Data Source dialog box, select **Cloudera ODBC Driver for Apache Hive**and then click **Finish**. The Cloudera ODBC Driver for Apache HiveDSN Setup dialog box opens.
6. In the **Data Source Name** field, type a name for your DSN.
7. Optionally, in the **Description** field, type relevant details about the DSN.
8. In the **Hive Server Type** drop-down list, select **Hive Server 1** or **Hive Server 2**.

   > **Note:** If you are connecting through Apache ZooKeeper, then **Hive Server 1** is not supported.

9. Specify whether the connector uses the ZooKeeper service when connecting to Hive, and provide the necessary connection information:
   - To connect to Hive without using the Apache ZooKeeper service, do the following:
     a. From the **Service Discovery Mode** drop-down list, select **No Service Discovery**.
     b. In the **Host(s)** field, type the IP address or host name of the Hive server.

c. In the **Port** field, type the number of the TCP port that the Hive server uses to listen for client connections.

- Or, to discover Hive Server 2 services via the ZooKeeper service, do the following:

a. From the **Service Discovery Mode** drop-down list, select **ZooKeeper**.

b. In the **Host(s)** field, type a comma-separated list of ZooKeeper servers. Use the following format, where *[ZK_Host]* is the IP address or host name of the ZooKeeper server and *[ZK_Port]* is the number of the TCP port that the ZooKeeper server uses to listen for client connections:

```
[ZK_Host1]:[ZK_Port1],[ZK_Host2]:[ZK_Port2]
```

c. In the **ZooKeeper Namespace** field, type the namespace on ZooKeeper under which Hive Server 2 znodes are added.

10. In the **Database** field, type the name of the database schema to use when a schema is not explicitly specified in a query.

> **Note:** You can still issue queries on other schemas by explicitly specifying the schema in the query. To inspect your databases and determine the appropriate schema to use, type the `show databases` command at the Hive command prompt.

11. In the Authentication area, configure authentication as needed. For more information, see Configuring Authentication in Windows.

> **Note:** Hive Server 1 does not support authentication. Most default configurations of Hive Server 2 require User Name authentication. To verify the authentication mechanism that you need to use for your connection, check the configuration of your Hadoop / Hive distribution. For more information, see Authentication Mechanisms.

12. Optionally, if the operations against Hive are to be done on behalf of a user that is different than the authenticated user for the connection, type the name of the user to be delegated in the **Delegation UID** field. For more information, see Delegating Authentication to a Specific User.

> **Note:** This option is applicable only when connecting to a Hive Server 2 instance that supports this feature.

13. In the **Thrift Transport** drop-down list, select the transport protocol to use in the Thrift layer.

> **Note:** For information about how to determine which Thrift transport protocols your Hive server supports, see Authentication Mechanisms.

14. If the Thrift Transport option is set to HTTP, then to configure HTTP options such as custom headers, click **HTTP Options**. For more information, see Configuring HTTP Options in Windows.

15. To configure the connector to connect to Hive through a proxy server, click **Proxy Options**. For more information, see Configuring a Proxy Connection in Windows.
16. To configure client-server verification over SSL, click **SSL Options**. For more information, see Configuring SSL Verification in Windows.

    Note: If you selected User Name as the authentication mechanism, SSL is not available.

17. To configure advanced connector options, click **Advanced Options**. For more information, see Configuring Advanced Options in Windows.
18. To configure server-side properties, click **Advanced Options** and then click **Server Side Properties**. For more information, see Configuring Server-Side Properties in Windows.
19. To configure the Temporary Table feature, click **Advanced Options** and then click **Temporary Table Configuration**. For more information, see Configuring the Temporary Table Feature and Temporary Tables.

    Note: When connecting to Hive 0.14 or later, the Temporary Tables feature is always enabled and you do not need to configure it in the connector.

20. To configure logging behavior for the connector, click **Logging Options**. For more information, see Configuring Logging Options in Windows.
21. To test the connection, click **Test**. Review the results as needed, and then click **OK**.

    Note: If the connection fails, then confirm that the settings in the Cloudera ODBC Driver for Apache Hive DSN Setup dialog box are correct. Contact your Hive server administrator as needed.

22. To save your settings and close the Cloudera ODBC Driver for Apache Hive DSN Setup dialog box, click **OK**.
23. To close the ODBC Data Source Administrator, click **OK**.

Configuring a DSN-less Connection in Windows

Some client applications provide support for connecting to a data source using a connector without a Data Source Name (DSN). To configure a DSN-less connection, you can use a connection string or the Driver Configuration tool that is installed with the Cloudera ODBC Driver for Apache Hive. Settings in a connection string apply only when you connect to Hive using that particular string, while settings in the connector configuration tool apply to every connection that uses the Cloudera ODBC Driver for Apache Hive.

The following section explains how to use the connector configuration tool. For information about using connection strings, see Using a Connection String.

Note:

. Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

. The drop-down lists in the connector configuration tool only display one option at a time. Use the scroll arrows on the right side of the drop-down list to view and select other options.

To configure a DSN-less connection using the connector configuration tool:

1. Choose one:
   - If you are using Windows 7 or earlier, click **Start** > **All Programs** > **Cloudera ODBC Driver for Apache Hive <Version Number>** > **Driver Configuration**.
   - Or, if you are using Windows 8 or later, click the arrow button at the bottom of the Start screen, and then click **Cloudera ODBC Driver for Apache Hive <Version Number>** > **Driver Configuration**.

     Note: Make sure to select the Driver Configuration Tool that has the same bitness as the client application that you are using to connect to Hive.

2. If you are prompted for administrator permission to make modifications to the machine, click **OK**.

   Note: You must have administrator access to the machine to run this application because it makes changes to the registry.

3. In the **Hive Server Type** drop-down list, select **Hive Server 1** or **Hive Server 2**.

   Note: If you are connecting through Apache ZooKeeper, then **Hive Server 1** is not supported.

4. Specify whether the connector uses the ZooKeeper service when connecting to Hive:
   - To connect to Hive without using the Apache ZooKeeper service, from the **Service Discovery Mode** drop-down list, select **No Service Discovery**.
   - Or, to discover Hive Server 2 services via the ZooKeeper service, do the following:
     - From the **Service Discovery Mode** drop-down list, select **ZooKeeper**.
     - In the **Host(s)** field, type a comma-separated list of ZooKeeper servers. Use the following format, where *[ZK_Host]* is the IP address or host name of the ZooKeeper server and *[ZK_Port]* is the number of the TCP port that the ZooKeeper server uses to listen for client connections:

       ```
       [ZK_Host1]:[ZK_Port1],[ZK_Host2]:[ZK_Port2]
       ```

- In the **ZooKeeper Namespace** field, type the namespace on ZooKeeper under which Hive Server 2 znodes are added.

5. In the Authentication area, configure authentication as needed. For more information, see Configuring Authentication in Windows.

   Note: Hive Server 1 does not support authentication. Most default configurations of Hive Server 2 require User Name authentication. To verify the authentication mechanism that you need to use for your connection, check the configuration of your Hadoop / Hive distribution. For more information, see Authentication Mechanisms.

6. Optionally, if the operations against Hive are to be done on behalf of a user that is different than the authenticated user for the connection, then in the **Delegation UID** field, type the name of the user to be delegated. For more information, see Delegating Authentication to a Specific User.

   Note: This option is applicable only when connecting to a Hive Server 2 instance that supports this feature.

7. In the **Thrift Transport** drop-down list, select the transport protocol to use in the Thrift layer.

   Note: For information about how to determine which Thrift transport protocols your Hive server supports, see Authentication Mechanisms.

8. If the Thrift Transport option is set to HTTP, then to configure HTTP options such as custom headers, click **HTTP Options**. For more information, see Configuring HTTP Options in Windows.

9. To configure the connector to connect to Hive through a proxy server, click **Proxy Options**. For more information, see Configuring a Proxy Connection in Windows.

10. To configure client-server verification over SSL, click **SSL Options**. For more information, see Configuring SSL Verification in Windows.

    Note: If you selected User Name as the authentication mechanism, SSL is not available.

11. To configure advanced options, click **Advanced Options**. For more information, see Configuring Advanced Options in Windows.

12. To configure server-side properties, click **Advanced Options** and then click **Server Side Properties**. For more information, see Configuring Server-Side Properties in Windows.

13. To configure the Temporary Table feature, click **Advanced Options** and then click **Temporary Table Configuration**. For more information, see Temporary Tables and Configuring the Temporary Table Feature.

    Important: When connecting to Hive 0.14 or later, the Temporary Tables feature is always enabled and you do not need to configure it in the connector.

14. To save your settings and close the Driver Configuration tool, click **OK**.

Configuring Authentication in Windows

Some Hive Server 2 instances are configured to require authentication for access. To connect to a Hive server, you must configure the Cloudera ODBC Driver for Apache Hive to use the authentication mechanism that matches the access requirements of the server and provides the necessary credentials.

For information about how to determine the type of authentication your Hive server requires, see Authentication Mechanisms.

You can specify authentication settings in a DSN, in a connection string, or as connector-wide settings. Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

If cookie-based authentication is enabled in your Hive Server 2 database, you can specify a list of authentication cookies in the `HTTPAuthCookies` connection property. In this case, the connector authenticates the connection once based on the provided authentication credentials. It then uses the cookie generated by the server for each subsequent request in the same connection. For more information, see HTTPAuthCookies.

Note: In Windows, the `HTTPAuthCookies` property must be set in a connection string.

Using No Authentication

When connecting to a Hive server of type Hive Server 1, you must use No Authentication. When you use No Authentication, Binary is the only Thrift transport protocol that is supported.

To configure a connection without authentication:

1. Choose one:
   - To access authentication options for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
   - Or, to access authentication options for a DSN-less connection, open the Driver Configuration tool.
2. From the **Mechanism** drop-down list, select **No Authentication**.
3. If the Hive server is configured to use SSL, then click **SSL Options** to configure SSL for the connection. For more information, see Configuring SSL Verification in Windows.
4. To save your settings and close the dialog box, click **OK**.

Using Kerberos

This authentication mechanism is available only for Hive Server 2. When you use Kerberos authentication, the Binary transport protocol is not supported.

If the Use Only SSPI advanced option is disabled, then Kerberos must be installed and configured before you can use this authentication mechanism. For information about configuring Kerberos on your machine, see Configuring Kerberos Authentication for Windows. For information about setting the Use Only SSPI advanced option, see Configuring Advanced Options in Windows.

To configure Kerberos authentication:

1. Choose one:
   - To access authentication options for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
   - Or, to access authentication options for a DSN-less connection, open the Driver Configuration tool.
2. From the **Mechanism** drop-down list, select **Kerberos**.
3. Choose one:
   - To use the default realm defined in your Kerberos setup, leave the **Realm** field empty.
   - Or, if your Kerberos setup does not define a default realm or if the realm of your Hive Server 2 host is not the default, then, in the **Realm** field, type the Kerberos realm of the Hive Server 2.
4. In the **Host FQDN** field, type the fully qualified domain name of the Hive Server 2 host.

   Note: To use the Hive server host name as the fully qualified domain name for Kerberos authentication, in the **Host FQDN** field, type **_HOST**.
5. In the **Service Name** field, type the service name of the Hive server.
6. Optionally, if you are using MIT Kerberos and a Kerberos realm is specified in the **Realm** field, then choose one:
   - To have the Kerberos layer canonicalize the server's service principal name, leave the **Canonicalize Principal FQDN** check box selected.
   - Or, to prevent the Kerberos layer from canonicalizing the server's service principal name, clear the **Canonicalize Principal FQDN** check box.
7. To allow the connector to pass your credentials directly to the server for use in authentication, select **Delegate Kerberos Credentials**.
8. From the **Thrift Transport** drop-down list, select the transport protocol to use in the Thrift layer.

   Important:
   When using this authentication mechanism, the Binary transport protocol is not supported.
9. If the Hive server is configured to use SSL, then click **SSL Options** to configure SSL for the connection. For more information, see Configuring SSL Verification in Windows.
10. To save your settings and close the dialog box, click **OK**.

Using SAML 2.0

This authentication mechanism enables you to authenticate via Single Sign-On using SAML 2.0 against supported servers.

Important: In order to use SAML 2.0 for authentication, Thrift Transport must be set to HTTP and SSL must be enabled.

To configure SAML 2.0 authentication:

1. To access authentication options, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
2. In the **Mechanism** drop-down list, select **SAML_2.0**.
3. In the **Host** field, type the fully qualified domain name of the Hive server host.
4. In the **Port** field, type the number of the TCP port that the Hive server uses to listen for client connections.
5. Optionally, in the **Transport Buffer Size** field, type the number of bytes to reserve in memory for buffering unencrypted data from the network.

   Note: In most circumstances, the default value of 1000 bytes is optimal.
6. In the **Thrift Transport** drop-down list, select **HTTP**.
7. Optionally, click **SAML Options** and select the **Ignore SQL_DRIVER_NOPROMPT** check box. When the application is making a SQLDriverConnect call with a SQL_DRIVER_NOPROMPT flag, this option displays the web browser used to complete the browser based authentication flow.
8. Optionally, click **SAML Options** and select the **Enable Auth Cookie Caching** check box. When establishing a new connection using SAML SSO authentication, the connector caches the authorization cookie and does not repeatedly open a new browser.
9. Click **HTTP Options** and in the **HTTP Path** field, type the partial URL corresponding to the Hive server. For more information, see Configuring HTTP Options in Windows.
10. Click **SSL Options** and select the **Enable SSL** check box. For more information, see Configuring SSL Verification in Windows
11. To save your settings and close the dialog box, click **OK**.

Note:
Tableau does not currently support SAML_2.0 in the UI. To use SAML_2.0 with the Tableau application:

- In the registry, `HKEY_LOCAL_MACHINE\SOFTWARE\Cloudera\Cloudera ODBC Driver for Apache Hive\Driver`, add the following connector-wide configurations:

```
DriverConfigTakePrecedence=1,AuthMech=12;ThriftTransport=2
;HttpPath=
cliservice;SSL=1;SSOIgnoreDriverNoPrompt=1
```

After adding the above settings in the registry, all the connections with the Cloudera ODBC Driver for Apache Hive on this machine will use SAML_2.0 authentication.

Using User Name

This authentication mechanism requires a user name but not a password. The user name labels the session, facilitating database tracking.

This authentication mechanism is available only for Hive Server 2. Most default configurations of Hive Server 2 require User Name authentication. When you use User Name authentication, SSL is not supported and SASL is the only Thrift transport protocol available.

To configure User Name authentication:

1. Choose one:
   - To access authentication options for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
   - Or, to access authentication options for a DSN-less connection, open the Driver Configuration tool.
2. From the **Mechanism** drop-down list, select **User Name**.
3. In the **User Name** field, type an appropriate user name for accessing the Hive server.
4. To save your settings and close the dialog box, click **OK**.

Using User Name And Password

This authentication mechanism requires a user name and a password.

This authentication mechanism is available only for Hive Server 2.

To configure User Name And Password authentication:

1. Choose one:
   - To access authentication options for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
   - Or, to access authentication options for a DSN-less connection, open the Driver Configuration tool.
2. From the **Mechanism** drop-down list, select **User Name And Password**.
3. In the **User Name** field, type an appropriate user name for accessing the Hive server.
4. In the **Password** field, type the password corresponding to the user name you typed above.
5. To save the password, select the **Save Password (Encrypted)** check box.

   Important:
   The password is obscured, that is, not saved in plain text. However, it is still possible for the encrypted password to be copied and used.
6. From the **Thrift Transport** drop-down list, select the transport protocol to use in the Thrift layer.

7. If the Hive server is configured to use SSL, then click **SSL Options** to configure SSL for the connection. For more information, see Configuring SSL Verification in Windows.
8. To save your settings and close the dialog box, click **OK**.

Delegating Authentication to a Specific User

Some Hive Server 2 instances support the ability to delegate all operations against Hive to the specified user, rather than to the authenticated user for the connection.

To delegate all operations to a specified user:

1. Choose one:

- To access authentication options for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
- Or, to access authentication options for a DSN-less connection, open the Driver Configuration tool.

2. In the **Delegation UID** field, type the name of the user to be delegated.
3. To save your settings and close the dialog box, click **OK**.

   If the server returns an error message such as **Failed to validate proxy privilege of *[RealUser]* for *[DelegationUID]***, you may need to modify the server's `core-site.xml` configuration file, as follows:

   a. In the server's `core-site.xml` configuration file, add the following properties:

```
hadoop.proxyuser.[RealUser].groups=*

hadoop.proxyuser.[RealUser].hosts=*
```

   Where *[RealUser]* is the authenticated user for the connection.

   b. If you are using Kerberos authentication, then in the server's `core-site.xml` configuration file, add the following properties:

```
hadoop.proxyuser.[Principal].groups=*

hadoop.proxyuser.[Principal].hosts=*
```

   Where *[Principal]* is the primary Kerberos principal user. For example, if the primary Kerberos principal user is `kerbuser@example.com`, replace *[Principal]* with `kerbuser`.

For more information on resolving this error, see your Hive Server documentation.

Configuring Advanced Options in Windows

You can configure advanced options to modify the behavior of the connector.

The following instructions describe how to configure advanced options in a DSN and in the connector configuration tool. You can specify the connection settings described below in a DSN, in a connection string, or as connector-wide settings. Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

To configure advanced options in Windows:

1. Choose one:
   - To access advanced options for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, and then click **Advanced Options**.
   - Or, to access advanced options for a DSN-less connection, open the Driver Configuration tool, and then click **Advanced Options**.
2. To disable the SQL Connector feature, select the **Use Native Query** check box.

   Important:
   - When this option is enabled, the connector cannot execute parameterized queries.
   - By default, the connector applies transformations to the queries emitted by an application to convert the queries into an equivalent form in HiveQL. If the application is Hive-aware and already emits HiveQL, then turning off the translation avoids the additional overhead of query transformation.
3. To defer query execution to SQLExecute, select the **Fast SQLPrepare** check box.
4. To allow connector-wide configurations to take precedence over connection and DSN settings, select the **Driver Config Take Precedence** check box.
5. To use the asynchronous version of the API call against Hive for executing a query, select the **Use Async Exec** check box.
6. To retrieve table names from the database by using the SHOW TABLES query, select the **Get Tables With Query** check box.

   Note: This option is applicable only when connecting to Hive Server 2.
7. To enable the connector to return SQL_WVARCHAR instead of SQL_VARCHAR for STRING and VARCHAR columns, and SQL_WCHAR instead of SQL_CHAR for CHAR columns, select the **Unicode SQL Character Types** check box.
8. To enable the connector to return the hive_system table for catalog function calls such as SQLTables and SQLColumns, select the **Show System Table** check box.
9. To specify which mechanism the connector uses by default to handle Kerberos authentication, do one of the following:

- To use the SSPI plugin by default, select the **Use Only SSPI** check box.
- To use MIT Kerberos by default and only use the SSPI plugin if the GSSAPI library is not available, clear the **Use Only SSPI** check box.

10. To enable the connector to automatically open a new session when the existing session is no longer valid, select the **Invalid Session Auto Recover** check box.

    Note: This option is applicable only when connecting to Hive Server 2.

11. To have the connector automatically attempt to reconnect to the server if communications are lost, select **AutoReconnect**.
12. To enable the connector to perform a query translation for the CREATE TABLE AS SELECT (CTAS syntax), select **Enable Transaction For CTAS**.
13. To enable the connector to ignore SQLTables API calls that request metadata from all schemas and returns an empty result set, select **Ignore Tables Metadata From All Schemas**.
14. In the **Rows Fetched Per Block** field, type the number of rows to be fetched per block.
15. In the **Max Bytes Per Fetch Request** field, type the maximum number of bytes to be fetched.

    Note: This option is applicable only when connecting to a server that supports result set data serialized in arrow format.The value must be specified in one of the following:

    - B (bytes)
    - KB (kilobytes)
    - MB (megabytes)
    - GB (gigabytes)

    By default, the file size is in B (bytes).

16. In the **Default String Column Length** field, type the maximum data length for STRING columns.
17. In the **Binary Column Length** field, type the maximum data length for BINARY columns.
18. In the **Decimal Column Scale** field, type the maximum number of digits to the right of the decimal point for numeric data types.
19. In the **Socket Timeout** field, type the number of seconds that an operation can remain idle before it is closed.

    Note: This option is applicable only when asynchronous query execution is being used against Hive Server 2 instances.

20. In the **Query Timeout Override** field, type the number of seconds that a query can run before it is timed out.

> **Note:** When the value passed is an empty string, the connector does not attempt to override the `SQL_ATTR_QUERY_TIMEOUT` attribute.

21. To save your settings and close the Advanced Options dialog box, click **OK**.

Configuring a Proxy Connection in Windows

If you are connecting to the data source through a proxy server, you must provide connection information for the proxy server.

To configure a proxy server connection in Windows:

1. To access proxy server options, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, and then click **Proxy Options**.
2. In the **Proxy Host** field, type the host name or IP address of the proxy server.
3. In the **Proxy Port** field, type the number of the TCP port that the proxy server uses to listen for client connections.
4. In the **Proxy Username** field, type your user name for accessing the proxy server.
5. In the **Proxy Password** field, type the password corresponding to the user name.
6. To save your settings and close the Proxy Options dialog box, click **OK**.

Configuring HTTP Options in Windows

You can configure options such as custom headers when using the HTTP transport protocol in the Thrift layer. For information about how to determine if your Hive server supports the HTTP transport protocol, see Authentication Mechanisms.

The following instructions describe how to configure HTTP options in a DSNand in the connector configuration tool. You can specify the connection settings described below in a DSN, in a connection string, or as connector-wide settings. Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

To configure HTTP options in Windows:

1. Choose one:
   - If you are configuring HTTP for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, and then make sure that the Thrift Transport option is set to **HTTP**.
   - Or, if you are configuring HTTP for a DSN-less connection, open the Driver Configuration tool and then make sure that the Thrift Transport option is set to **HTTP**.
2. To access HTTP options, click **HTTP Options**.

> **Note:** The HTTP options are available only when the Thrift Transport option is set to HTTP.

3. In the **HTTP Path** field, type the partial URL corresponding to the Hive server.
4. To create a custom HTTP header, click **Add**, then type appropriate values in the **Key** and **Value** fields, and then click **OK**.
5. To edit a custom HTTP header, select the header from the list, then click **Edit**, then update the **Key** and **Value** fields as needed, and then click **OK**.
6. To delete a custom HTTP header, select the header from the list, and then click **Remove**. In the confirmation dialog box, click **Yes**.
7. To save your settings and close the HTTP Properties dialog box, click **OK**.

Configuring SSL Verification in Windows

If you are connecting to a Hive server that has Secure Sockets Layer (SSL) enabled, you can configure the connector to connect to an SSL-enabled socket. When using SSL to connect to a server, the connector supports identity verification between the client (the connector itself) and the server.

The following instructions describe how to configure SSL in a DSN and in the connector configuration tool. You can specify the connection settings described below in a DSN, in a connection string, or as connector-wide settings. Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

Note: If you selected User Name as the authentication mechanism, SSL is not available.

To configure SSL verification in Windows:

1. Choose one:
   - To access SSL options for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, and then click **SSL Options**.
   - Or, to access advanced options for a DSN-less connection, open the Driver Configuration tool, and then click **SSL Options**.
2. Select the **Enable SSL** check box.
3. To allow authentication using self-signed certificates that have not been added to the list of trusted certificates, select the **Allow Self-signed Server Certificate** check box.
4. To allow the common name of a CA-issued SSL certificate to not match the host name of the Hive server, select the **Allow Common Name Host Name Mismatch** check box.
5. To specify the CA certificates that you want to use to verify the server, do one of the following:
   - To verify the server using the trusted CA certificates from a specific `.pem` file, specify the full path to the file in the **Trusted Certificates** field and clear the **Use System Trust Store** check box.
   - Or, to use the trusted CA certificates `.pem` file that is installed with the connector , leave the **Trusted Certificates** field empty, and clear the **Use System Trust Store** check box.

- Or, to use the Windows trust store, select the **Use System Trust Store** check box.

  Important:
  - If you are using the Windows trust store, make sure to import the trusted CA certificates into the trust store.
  - If the trusted CA supports certificate revocation, select the **Check Certificate Revocation** check box.

6. From the **Minimum TLS Version** drop-down list, select the minimum version of TLS to use when connecting to your data store.
7. To configure two-way SSL verification, select the **Two-Way SSL** check box and then do the following:
   a. In the **Client Certificate File** field, specify the full path of the PEM file containing the client's certificate.
   b. In the **Client Private Key File** field, specify the full path of the file containing the client's private key.
   c. If the private key file is protected with a password, type the password in the **Client Private Key Password** field. To save the password, select the **Save Password (Encrypted)** check box.

      Important: The password is obscured, that is, not saved in plain text. However, it is still possible for the encrypted password to be copied and used.
8. To save your settings and close the SSL Options dialog box, click **OK**.

Configuring the Temporary Table Feature

You can configure the driver to create temporary tables. For more information about this feature, including details about the statement syntax used for temporary tables, see Temporary Tables.

Important:
When connecting to Hive 0.14 or later, the Temporary Tables feature is always enabled and you do not need to configure it in the driver.

To configure the Temporary Table feature:

1. Choose one:
   - To configure the temporary table feature for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN and click **Configure**, then click **Advanced Options**, and then click **Temporary Table Configuration**.
   - Or, to configure the temporary table feature for a DSN-less connection, open the Driver Configuration tool, then click **Advanced Options**, and then click **Temporary Table Configuration**.
2. To enable the Temporary Table feature, select the **Enable Temporary Table** check box.

3. In the **Web HDFS Host** field, type the host name or IP address of the machine hosting both the namenode of your Hadoop cluster and the WebHDFS service. If this field is left blank, then the host name of the Hive server is used.
4. In the **Web HDFS Port** field, type the WebHDFS port for the namenode.
5. In the **HDFS User** field, type the name of the HDFS user that the driver uses to create the necessary files for supporting the Temporary Table feature.
6. In the **Data File HDFS Dir** field, type the HDFS directory that the driver uses to store the necessary files for supporting the Temporary Table feature.

   Note:
   Due to a known issue in Hive (see https://issues.apache.org/jira/browse/HIVE-4554), HDFS paths with space characters do not work with versions of Hive prior to 0.12.0.

7. In the **Temp Table TTL** field, type the number of minutes that a temporary table is guaranteed to exist in Hive after it is created.
8. To save your settings and close the Temporary Table Configuration dialog box, click **OK**.

Configuring Server-Side Properties in Windows

You can use the connector to apply configuration properties to the Hive server.

The following instructions describe how to configure server-side properties in a DSN and in the connector configuration tool. You can specify the connection settings described below in a DSN, in a connection string, or as connector-wide settings. Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

To configure server-side properties in Windows:

1. Choose one:

   - To configure server-side properties for a DSN, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, then click **Advanced Options**, and then click **Server Side Properties**.
   - Or, to configure server-side properties for a DSN-less connection, open the Driver Configuration tool, then click **Advanced Options**, and then click **Server Side Properties**.

2. To create a server-side property, click **Add**, then type appropriate values in the **Key** and **Value** fields, and then click **OK**. For example, to set the value of the `mapreduce.job.queuename` property to `myQueue`, type **mapreduce.job.queuename** in the **Key** field and then type **myQueue** in the **Value** field.

   Note: For a list of all Hadoop and Hive server-side properties that your implementation supports, type `set -v` at the Hive CLI command lineor Beeline. You can also execute the `set -v` query after connecting using the connector.

3. To edit a server-side property, select the property from the list, then click **Edit**, then update the **Key** and **Value** fields as needed, and then click **OK**.

4. To delete a server-side property, select the property from the list, and then click **Remove**. In the confirmation dialog box, click **Yes**.
5. To configure the connector to convert server-side property key names to all lower-case characters, select the **Convert Key Name To Lower Case** check box.
6. To change the method that the connector uses to apply server-side properties, do one of the following:
   - To configure the connector to apply each server-side property by executing a query when opening a session to the Hive server, select the **Apply Server Side Properties With Queries** check box.
   - Or, to configure the connector to use a more efficient method for applying server-side properties that does not involve additional network round-tripping, clear the **Apply Server Side Properties With Queries** check box.

   Note: The more efficient method is not available for Hive Server 1, and it might not be compatible with some Hive Server 2 builds. If the server-side properties do not take effect when the check box is clear, then select the check box.
7. To save your settings and close the Server Side Properties dialog box, click **OK**.

Configuring Logging Options in Windows

To help troubleshoot issues, you can enable logging. In addition to functionality provided in the Cloudera Cloudera ODBC Driver for Apache Hive, the ODBC Data Source Administrator provides tracing functionality.

Important: Only enable logging or tracing long enough to capture an issue. Logging or tracing decreases performance and can consume a large quantity of disk space.

Configuring Connector-wide Logging Options

The settings for logging apply to every connection that uses the Cloudera ODBC Driver for Apache Hive, so make sure to disable the feature after you are done using it. To configure logging for the current connection, see Configuring Logging for the Current Connection.

To enable connector-wide logging in Windows:

1. To access logging options, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, and then click **Logging Options**.

   Note: A warning message appears when users without the administrator permissions click **Logging Options**. After acknowledging the warning, the Logging Options dialog opens with all settings disabled (greyed out) to prevent unauthorized modifications.
2. From the **Log Level** drop-down list, select the logging level corresponding to the amount of information that you want to include in log files:

Logging Level	Description
OFF	Disables all logging.
FATAL	Logs severe error events that lead the connector to abort.
ERROR	Logs error events that might allow the connector to continue running.
WARNING	Logs events that might result in an error if action is not taken.
INFO	Logs general information that describes the progress of the connector.
DEBUG	Logs detailed information that is useful for debugging the connector.
TRACE	Logs all connector activity.

3. In the **Log Path** field, specify the full path to the folder where you want to save log files.
4. If requested by Technical Support, type the name of the component for which to log messages in the **Log Namespace** field. Otherwise, do not type a value in the field.
5. In the **Max Number Files** field, type the maximum number of log files to keep.

   Note: After the maximum number of log files is reached, each time an additional file is created, the connector deletes the oldest log file.

6. In the **Max File Size** field, type the maximum size of each log file in megabytes (MB).

   Note: After the maximum file size is reached, the connector creates a new file and continues logging.

7. Click **OK**.
8. Restart your ODBC application to make sure that the new settings take effect.

The Cloudera ODBC Driver for Apache Hive produces the following log files at the location you specify in the Log Path field:

- A `clouderaodbcdriverforapachehive.log` file that logs connector activity that is not specific to a connection.
- A `clouderaodbcdriverforapachehive_connection_[Number].log` file for each connection made to the database, where *[Number]* is a number that identifies each log file. This file logs connector activity that is specific to the connection.

To disable connector logging in Windows:

1. Open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, and then click **Logging Options**.
2. From the **Log Level** drop-down list, select **LOG_OFF**.
3. Click **OK**.
4. Restart your ODBC application to make sure that the new settings take effect.

Configuring Logging for the Current Connection

You can configure logging for the current connection by setting the logging configuration properties in the DSN or in a connection string. For information about the logging configuration properties, see Configuring Logging Options in Windows. Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

Note: If the LogLevel configuration property is passed in via the connection string or DSN, the rest of the logging configurations are read from the connection string or DSN and not from the existing connector-wide logging configuration.

To configure logging properties in the DSN, you must modify the Windows registry. For information about the Windows registry, see the Microsoft Windows documentation.

Important: Editing the Windows Registry incorrectly can potentially cause serious, system-wide problems that may require re-installing Windows to correct.

To add logging configurations to a DSN in Windows:

1. On the Start screen, type **regedit**, and then click the **regedit** search result.
2. Navigate to the appropriate registry key for the bitness of your connector and your machine:
   - 32-bit System DSNs: **HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\ODBC\ODBC.INI*[DSN Name]***
   - 64-bit System DSNs: **HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBC.INI*[DSN Name]***
   - 32-bit and 64-bit User DSNs: **HKEY_CURRENT_USER\SOFTWARE\ODBC\ODBC.INI*[DSN Name]***
3. For each configuration option that you want to configure for the current connection, create a value by doing the following:
   a. If the key name value does not already exist, create it. Right-click the *[DSN Name]* and then select **New > String Value**, type the key name of the configuration option, and then press **Enter**.
   b. Right-click the key name and then click **Modify**.

      To confirm the key names for each configuration option, see Connector Configuration Options.
   c. In the Edit String dialog box, in the **Value Data** field, type the value for the configuration option.
4. Close the Registry Editor.
5. Restart your ODBC application to make sure that the new settings take effect.

Configuring Kerberos Authentication for Windows

Active Directory

The Cloudera ODBC Driver for Apache Hive supports Active Directory Kerberos in Windows. There are two prerequisites for using Active Directory Kerberos in Windows:

- MIT Kerberos is not installed on the client Windows machine.
- The MIT Kerberos Hadoop realm has been configured to trust the Active Directory realm so that users in the Active Directory realm can access services in the MIT Kerberos Hadoop realm.

MIT Kerberos

Downloading and Installing MIT Kerberos for Windows 4.0.1

For information about Kerberos and download links for the installer, see the MIT Kerberos website: http://web.mit.edu/kerberos/.

To download and install MIT Kerberos for Windows 4.0.1:

1. Download the appropriate Kerberos installer:
   - For a 64-bit machine, use the following download link from the MIT Kerberos website: http://web.mit.edu/kerberos/dist/kfw/4.0/kfw-4.0.1-amd64.msi.
   - For a 32-bit machine, use the following download link from the MIT Kerberos website: http://web.mit.edu/kerberos/dist/kfw/4.0/kfw-4.0.1-i386.msi.

   > **Note:** The 64-bit installer includes both 32-bit and 64-bit libraries. The 32-bit installer includes 32-bit libraries only.

2. To run the installer, double-click the `.msi` file that you downloaded above.
3. Follow the instructions in the installer to complete the installation process.
4. When the installation completes, click **Finish**.

Setting Up the Kerberos Configuration File

Settings for Kerberos are specified through a configuration file. You can set up the configuration file as an `.ini` file in the default location, which is the `C:\ProgramData\MIT\Kerberos5` directory, or as a `.conf` file in a custom location.

Normally, the `C:\ProgramData\MIT\Kerberos5` directory is hidden. For information about viewing and using this hidden directory, refer to Microsoft Windows documentation.

> **Note:** For more information on configuring Kerberos, refer to the MIT Kerberos documentation.

To set up the Kerberos configuration file in the default location:

1. Obtain a `krb5.conf` configuration file. You can obtain this file from your Kerberos administrator, or from the `/etc/krb5.conf` folder on the machine that is hosting the Hive Server 2 instance.
2. Rename the configuration file from `krb5.conf` to `krb5.ini`.
3. Copy the `krb5.ini` file to the `C:\ProgramData\MIT\Kerberos5` directory and overwrite the empty sample file.

To set up the Kerberos configuration file in a custom location:

1. Obtain a `krb5.conf` configuration file. You can obtain this file from your Kerberos administrator, or from the `/etc/krb5.conf` folder on the machine that is hosting the Hive Server 2 instance.
2. Place the `krb5.conf` file in an accessible directory and make note of the full path name.
3. Open the System window:
   - Click **Start**, then right-click **Computer**, and then click **Properties**.
   - Or right-click **This PC** on the Start screen, and then click **Properties**.
4. Click **Advanced System Settings**.
5. In the System Properties dialog box, click the **Advanced** tab and then click **Environment Variables**.
6. In the Environment Variables dialog box, under the System Variables list, click **New**.
7. In the New System Variable dialog box, in the **Variable Name** field, type **KRB5_CONFIG**.
8. In the **Variable Value** field, type the full path to the `krb5.conf` file.
9. Click **OK** to save the new variable.
10. Make sure that the variable is listed in the System Variables list.
11. Click **OK** to close the Environment Variables dialog box, and then click **OK** to close the System Properties dialog box.

Setting Up the Kerberos Credential Cache File

Kerberos uses a credential cache to store and manage credentials.

To set up the Kerberos credential cache file:

1. Create a directory where you want to save the Kerberos credential cache file. For example, create a directory named `C:\temp`.
2. Open the System window:
   - Click **Start**, then right-click **Computer**, and then click **Properties**.
   - Or right-click **This PC** on the Start screen, and then click **Properties**.
3. Click **Advanced System Settings**.
4. In the System Properties dialog box, click the **Advanced** tab and then click **Environment Variables**.
5. In the Environment Variables dialog box, under the System Variables list, click **New**.

6. In the New System Variable dialog box, in the **Variable Name** field, type **KRB5CCNAME**.
7. In the **Variable Value** field, type the path to the folder you created above, and then append the file name `krb5cache`. For example, if you created the folder `C:\temp`, then type `C:\temp\krb5cache`.

   > **Note:** `krb5cache` is a file (not a directory) that is managed by the Kerberos software, and it should not be created by the user. If you receive a permission error when you first use Kerberos, make sure that the `krb5cache` file does not already exist as a file or a directory.

8. Click **OK** to save the new variable.
9. Make sure that the variable appears in the System Variables list.
10. Click **OK** to close the Environment Variables dialog box, and then click **OK** to close the System Properties dialog box.
11. To make sure that Kerberos uses the new settings, restart your machine.

Obtaining a Ticket for a Kerberos Principal

A principal refers to a user or service that can authenticate to Kerberos. To authenticate to Kerberos, a principal must obtain a ticket by using a password or a keytab file. You can specify a keytab file to use, or use the default keytab file of your Kerberos configuration.

To obtain a ticket for a Kerberos principal using a password:

1. Open MIT Kerberos Ticket Manager.
2. In MIT Kerberos Ticket Manager, click **Get Ticket**.
3. In the Get Ticket dialog box, type your principal name and password, and then click **OK**.

   If the authentication succeeds, then your ticket information appears in MIT Kerberos Ticket Manager.

   To obtain a ticket for a Kerberos principal using a keytab file:

   a. Open a command prompt:

      - Click **Start**, then click **All Programs**, then click **Accessories**, and then click **Command Prompt**.
      - Click the arrow button at the bottom of the Start screen, then find the Windows System program group, and then click **Command Prompt**.

   b. In the Command Prompt, type a command using the following syntax:

      ```
      kinit -k -t [KeytabPath][Principal]
      ```

      [KeytabPath] is the full path to the keytab file. For example: `C:\mykeytabs\myUser.keytab`.

[Principal] is the Kerberos user principal to use for authentication. For example: `myUser@EXAMPLE.COM`.

c. If the cache location KRB5CCNAME is not set or used, then use the `-c` option of the `kinit` command to specify the location of the credential cache. In the command, the `-c` argument must appear last. For example:

```
kinit -k -t C:\mykeytabs\myUser.keytab myUser@EXAMPLE.COM -c
C:\ProgramData\MIT\krbcache
```

`Krbcache` is the Kerberos cache file, not a directory.

To obtain a ticket for a Kerberos principal using the default keytab file:

Note: For information about configuring a default keytab file for your Kerberos configuration, refer to the MIT Kerberos documentation.

1. Open a command prompt:
   - Click **Start**, then click **All Programs**, then click **Accessories**, and then click **Command Prompt**.
   - Click the arrow button at the bottom of the Start screen, then find the Windows System program group, and then click **Command Prompt**.
2. In the Command Prompt, type a command using the following syntax:

```
kinit -k [principal]
```

[principal] is the Kerberos user principal to use for authentication. For example: `MyUser@EXAMPLE.COM`.

3. If the cache location KRB5CCNAME is not set or used, then use the `-c` option of the `kinit` command to specify the location of the credential cache. In the command, the `-c` argument must appear last. For example:

```
kinit -k -t C:\mykeytabs\myUser.keytab myUser@EXAMPLE.COM -c
C:\ProgramData\MIT\krbcache
```

`Krbcache` is the Kerberos cache file, not a directory.

Verifying the Connector Version Number in Windows

If you need to verify the version of the Cloudera ODBC Driver for Apache Hive that is installed on your Windows machine, you can find the version number in the ODBC Data Source Administrator.

To verify the connector version number in Windows:

1. From the Start menu, go to **ODBC Data Sources**.

   Note: Make sure to select the ODBC Data Source Administrator that has the same bitness as the client application that you are using to connect to Hive.

2. Click the **Drivers** tab and then find the Cloudera ODBC Driver for Apache Hive in the list of ODBC Connectors that are installed on your system. The version number is displayed in the **Version** column.

macOS Connector

This section provides an overview of the Connector in the mac OS platform, outlining the required system specifications and the steps for installing and configuring the connector in mac OS environments.

macOS System Requirements

Install the connector on client machines where the application is installed. Each client machine that you install the connector on must meet the following minimum system requirements:

- 100MB of available disk space
- One of the following ODBC driver managers installed:
  - iODBC 3.52.9 or later
  - unixODBC 2.2.14 or later

Installing the Connector in macOS

If you received the connector as a macOS tarball package, see the *Simba OEM ODBC Connectors Installation Guide*.

The Cloudera ODBC Driver for Apache Hive is available for macOS as a `.dmg` file named `ClouderaHiveODBC.dmg`. The connector supports both 32- and 64-bit client applications.

To install the Cloudera ODBC Driver for Apache Hive in macOS:

1. Double-click **ClouderaHiveODBC.dmg** to mount the disk image.
2. Double-click **ClouderaHiveODBC.pkg** to run the installer.
3. In the installer, click **Continue**.
4. On the Software License Agreement screen, click **Continue**, and when the prompt appears, click **Agree** if you agree to the terms of the License Agreement.
5. Optionally, to change the installation location, click **Change Install Location**, then select the desired location, and then click **Continue**.

   Note: By default, the connector files are installed in the `/opt/cloudera/hiveodbc` directory.

6. To accept the installation location and begin the installation, click **Install**.
7. When the installation completes, click **Close**.

Next, configure the environment variables on your machine to make sure that the ODBC Driver manager can work with the connector. For more information, see Configuring the ODBC Driver Manager in Non-Windows Machines.

Verifying the Connector Version Number in macOS

If you need to verify the version of the Cloudera ODBC Driver for Apache Hive that is installed on your macOS machine, you can query the version number through the Terminal.

To verify the connector version number in macOS:

- At the Terminal, run the command:

```
pkgutil --info cloudera.hiveodbc
```

The command returns information about the Cloudera ODBC Driver for Apache Hive that is installed on your machine, including the version number.

Linux Connector

This section provides an overview of the Connector in the Linux platform, outlining the required system specifications and the steps for installing and configuring the connector in Linux environments.

For most Linux distributions, you can install the connector using the RPM file. If you are installing the connector on a Debian machine, you must use the Debian package.

Linux System Requirements

Install the connector on client machines where the application is installed. Each client machine that you install the connector on must meet the following minimum system requirements:

- 150MB of available disk space
- One of the following ODBC driver managers installed:
  - iODBC 3.52.9 or later
  - unixODBC 2.2.14 or later
- All of the following `libsasl` libraries installed:
  - `cyrus-sasl-2.1.22-7` or later
  - `cyrus-sasl-gssapi-2.1.22-7` or later
  - `cyrus-sasl-plain-2.1.22-7` or later

> **Note:** If the package manager in your Linux distribution cannot resolve the dependencies automatically when installing the connector, then download and manually install the packages.

To install the connector, you must have root access on the machine.

Installing the Connector Using the RPM File

On 64-bit editions of Linux, you can execute both 32- and 64-bit applications. However, 64-bit applications must use 64-bit connectors, and 32-bit applications must use 32-bit connectors. Make sure that you use a connector whose bitness matches the bitness of the client application:

- `ClouderaHiveODBC-32bit-`*`[Version]`*`-`*`[Release]`*`.i686.rpm` for the 32-bit connector
- `ClouderaHiveODBC-`*`[Version]`*`-`*`[Release]`*`.x86_64.rpm` for the 64-bit connector

The placeholders in the file names are defined as follows:

- *[Version]* is the version number of the connector.
- *[Release]* is the release number for this version of the connector.

You can install both the 32-bit and 64-bit versions of the connector on the same machine.

To install the Cloudera ODBC Driver for Apache Hive using the RPM File:

1. Log in as the root user.
2. Navigate to the folder containing the RPM package for the connector.
3. Depending on the Linux distribution that you are using, run one of the following commands from the command line, where *[RPMFileName]* is the file name of the RPM package:
   - If you are using Red Hat Enterprise Linux, or CentOS, run the following command:

     ```
     yum --nogpgcheck localinstall [RPMFileName]
     ```

   - Or, if you are using SUSE Linux Enterprise Server, run the following command:

     ```
     zypper install [RPMFileName]
     ```

   The Cloudera ODBC Driver for Apache Hive files are installed in the `/opt/cloudera/hiveodbc` directory.

   Note: If the package manager in your Linux distribution cannot resolve the `libsasl` dependencies automatically when installing the connector, then download and manually install the packages.

Next, configure the environment variables on your machine to make sure that the ODBC Driver manager can work with the connector. For more information, see Configuring the ODBC Driver Manager in Non-Windows Machines

Installing the Connector on Debian

To install the connector on a Debian machine, use the Debian package instead of the RPM file or tarball package.

On 64-bit editions of Debian(Non-ARM machine), you can execute both 32- and 64-bit applications. In this case, 64-bit applications must use 64-bit connectors, and 32-bit applications must use 32-bit connectors. Make sure that you use the version of the connector that matches the bitness of the client application:

- `clouderahiveodbc-32bit[Version]-[Release]_i386.deb` for the 32-bit connector
- `clouderahiveodbc[Version]-[Release]_amd64.deb` for the 64-bit connector

`[Version]` is the version number of the connector, and *`[Release]`* is the release number for this version of the connector.

You can install both versions of the connector on the same machine.

To install the Cloudera ODBC Driver for Apache Hive on Debian:

1. Log in as the root user, and then navigate to the folder containing the Debian package for the connector.

2. Double-click **clouderahiveodbc-32bit*[Version]-[Release]*_i386.deb** or **clouderahiveodbc *[Version]-[Release]*_amd64.deb**.
3. Follow the instructions in the installer to complete the installation process.

   The Cloudera ODBC Driver for Apache Hive files are installed in the `/opt/cloudera/hiveodbc` directory.

> **Note:** If the package manager in your Ubuntu distribution cannot resolve the `libsasl` dependencies automatically when installing the connector, then download and manually install the packages required by the version of the connector that you want to install.

Next, configure the environment variables on your machine to make sure that the ODBC driver manager can work with the connector. For more information, see Configuring the ODBC Driver Manager in Non-Windows Machines.

Verifying the Connector Version Number in Linux

If you need to verify the version of the Cloudera ODBC Driver for Apache Hive that is installed on your Linux machine, you can query the version number through the command-line interface if the connector was installed using an RPM file or Debian package. Alternatively, you can search the connector's binary file for version number information.

To verify the connector version number in Linux using the command-line interface:

- Depending on your package manager, at the command prompt, run one of the following commands:
  - yum list | grep ClouderaHiveODBC
  - rpm -qa | grep ClouderaHiveODBC
  - dpkg -l | grep clouderahiveodbc

The command returns information about the Cloudera ODBC Driver for Apache Hive that is installed on your machine, including the version number.

To verify the connector version number in Linux using the binary file:

1. Navigate to the `/lib` subfolder in your connector installation directory. By default, the path to this directory is: `/opt/cloudera/`hiveodbc`/lib`.
2. Open the connector's `.so` binary file in a text editor, and search for the text `$driver_version_sb$:`. The connector's version number is listed after this text.

AIX Connector

This section provides an overview of the Connector in the AIX platform, outlining the required system specifications and the steps for installing and configuring the connector in AIX environments.

AIX System Requirements

Install the connector on client machines where the application is installed. Each machine that you install the connector on must meet the following minimum system requirements:

- 150 MB of available disk space
- One of the following ODBC driver managers installed:
  - iODBC 3.52.9 or later
  - unixODBC 2.2.14 or later

To install the connector, you must have root access on the machine.

Installing the Connector on AIX

On 64-bit editions of AIX, you can execute both 32- and 64-bit applications. However, 64-bit applications must use 64-bit connectors, and 32-bit applications must use 32-bit connectors. Make sure that you use the version of the connector that matches the bitness of the client application:

- `ClouderaHiveODBC-32bit-[Version]-[Release].ppc.rpm` for the 32-bit connector
- `ClouderaHiveODBC-[Version]-[Release].ppc.rpm` for the 64-bit connector

[Version] is the version number of the connector, and *[Release]* is the release number for this version of the connector.

You can install both versions of the connector on the same machine.

To install the Cloudera ODBC Driver for Apache Hive on AIX:

1. Log in as the root user, and then navigate to the folder containing the RPM package for the connector.
2. Run the following command from the command line, where *[RPMFileName]* is the file name of the RPM package:

   ```
   rpm --install [RPMFileName]
   ```

   The Cloudera ODBC Driver for Apache Hive files are installed in the `/opt/cloudera/hiveodbc` directory.

Next, configure the environment variables on your machine to make sure that the ODBC driver manager can work with the connector. For more information, see Configuring the ODBC Driver Manager in Non-Windows Machines.

Verifying the Connector Version Number on AIX

If you need to verify the version of the Cloudera ODBC Driver for Apache Hive that is installed on your AIX machine, you can query the version number through the command-line interface.

To verify the connector version number on AIX:

- At the command prompt, run the following command:

```
rpm -qa | grep ClouderaHiveODBC
```

The command returns information about the Cloudera ODBC Driver for Apache Hive that is installed on your machine, including the version number.

Configuring the ODBC Driver Manager in Non-Windows Machines

To make sure that the ODBC Driver manager on your machine is configured to work with the Cloudera ODBC Driver for Apache Hive, do the following:

- Set the library path environment variable to make sure that your machine uses the correct ODBC Driver manager. For more information, see Specifying ODBC Driver Managers in Non-Windows Machines .
- If the connector configuration files are not stored in the default locations expected by the ODBC driver manager, then set environment variables to make sure that the Driver manager locates and uses those files. For more information, see Specifying the Locations of the Connector Configuration Files.

  After configuring the ODBC Driver manager, you can configure a connection and access your data store through the connector.

Specifying ODBC Driver Managers in Non-Windows Machines

You need to make sure that your machine uses the correct ODBC Driver manager to load the connector. To do this, set the library path environment variable.

macOS

If you are using a macOS machine, then set the DYLD_LIBRARY_PATH environment variable to include the paths to the ODBC driver manager libraries. For example, if the libraries are installed in `/usr/local/lib`, then run the following command to set DYLD_LIBRARY_PATH for the current user session:

```
export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:/usr/local/lib
```

For information about setting an environment variable permanently, refer to the macOS shell documentation.

Linux or AIX

If you are using a Linux or AIX machine, then set the LD_LIBRARY_PATH environment variable to include the paths to the ODBC driver manager libraries. For example, if the libraries are installed in `/usr/local/lib`, then run the following command to set LD_LIBRARY_PATH for the current user session:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
```

For information about setting an environment variable permanently, refer to the Linux or AIX shell documentation.

Specifying the Locations of the Connector Configuration Files

By default, ODBC Driver managers are configured to use hidden versions of the `odbc.ini` and `odbcinst.ini` configuration files (named `.odbc.ini` and `.odbcinst.ini`) located in the home directory, as well as the `cloudera.hiveodbc.ini` file in the `lib` subfolder of the connector installation directory. If you store these configuration files elsewhere, then you must set the environment variables described below so that the driver manager can locate the files.

If you are using iODBC, do the following:

- Set ODBCINI to the full path and file name of the `odbc.ini` file.
- Set ODBCINSTINI to the full path and file name of the `odbcinst.ini` file.
- Set CLOUDERAHIVEINI to the full path and file name of the `cloudera.hiveodbc.ini` file.

If you are using unixODBC, do the following:

- Set ODBCINI to the full path and file name of the `odbc.ini` file.
- Set ODBCSYSINI to the full path of the directory that contains the `odbcinst.ini` file.
- Set CLOUDERAHIVEINI to the full path and file name of the `cloudera.hiveodbc.ini` file.

For example, if your `odbc.ini` and `odbcinst.ini` files are located in `/usr/local/odbc` and your `cloudera.hiveodbc.ini` file is located in `/etc`, then set the environment variables as follows:

For iODBC:

```
export ODBCINI=/usr/local/odbc/odbc.ini
export ODBCINSTINI=/usr/local/odbc/odbcinst.ini
export CLOUDERAHIVEINI=/etc/cloudera.hiveodbc.ini
```

For unixODBC:

```
export ODBCINI=/usr/local/odbc/odbc.ini
export ODBCSYSINI=/usr/local/odbc
export CLOUDERAHIVEINI=/etc/cloudera.hiveodbc.ini
```

To locate the `cloudera.hiveodbc.ini` file, the connector uses the following search order:

1. If the CLOUDERAHIVEINI environment variable is defined, then the connector searches for the file specified by the environment variable.
2. The connector searches the directory that contains the connector library files for a file named `cloudera.hiveodbc.ini`.
3. The connector searches the current working directory of the application for a file named `cloudera.hiveodbc.ini`.

4. The connector searches the home directory for a hidden file named `cloudera.hiveodbc.ini` (prefixed with a period).
5. The connector searches the `/etc` directory for a file named `cloudera.hiveodbc.ini`.

Configuring ODBC Connections in Non-Windows Machine

The following sections describe how to configure ODBC connections when using the Cloudera ODBC Driver for Apache Hive on non-Windows platforms:

- Creating a Data Source Name on a Non-Windows Machine
- Configuring a DSN-less Connection in a Non-Windows Machine
- Configuring Service Discovery Mode on a Non-Windows Machine
- Configuring Authentication on a Non-Windows Machine
- Configuring SSL Verification in a Non-Windows Machine
- Configuring Server-Side Properties on a Non-Windows Machine
- Configuring Logging Options
- Setting Connector-Wide Configuration Options on a Non-Windows Machine
- Testing the Connection

Creating a Data Source Name on a Non-Windows Machine

Typically, after installing the Cloudera ODBC Driver for Apache Hive, you need to create a Data Source Name (DSN). A DSN is a data structure that stores connection information so that it can be used by the connector to connect to Hive.

You can specify connection settings in a DSN (in the `odbc.ini` file), in a connection string, or as connector-wide settings (in the `cloudera.hiveodbc.ini` file). Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

The following instructions describe how to create a DSN by specifying connection settings in the `odbc.ini` file. If your machine is already configured to use an existing `odbc.ini` file, then update that file by adding the settings described below. Otherwise, copy the `odbc.ini` file from the `Setup` subfolder in the connector installation directory to the home directory, and then update the file as described below.

For information about specifying settings in a connection string, see Configuring a DSN-less Connection in a Non-Windows Machine and Using a Connection String. For information about connector-wide settings, see Setting Connector-Wide Configuration Options on a Non-Windows Machine.

To create a Data Source Name on a non-Windows machine:

1. In a text editor, open the `odbc.ini` configuration file.

   Note: If you are using a hidden copy of the `odbc.ini` file, you can remove the period (`.`) from the start of the file name to make the file visible while you are editing it.

2. In the `[ODBC Data Sources]` section, add a new entry by typing a name for the DSN, an equal sign (=), and then the name of the connector.

   For example, on a macOS machine:

   ```
   [ODBC Data Sources]

   Sample DSN=Cloudera ODBC Driver for Apache Hive
   ```

   As another example, for a 32-bit connector on a Linux/AIX/Debian machine:

   ```
   [ODBC Data Sources]

   Sample DSN=Cloudera ODBC Driver for Apache Hive 32-bit
   ```

3. Create a section that has the same name as your DSN, and then specify configuration options as key-value pairs in the section:
   a. Set the `Driver` property to the full path of the connector library file that matches the bitness of the application.

      For example, on a macOS machine:

      ```
      Driver=/opt/cloudera/hiveodbc/lib/universal/libcloudera
      hiveodbc.dylib
      ```

      As another example, for a 32-bit connector on a Linux/AIX/Debian machine:

      ```
      Driver=/opt/cloudera/hiveodbc/lib/32/libclouderahiveodbc32.so
      ```

   b. Set the `HiveServerType` property to one of the following values:

- If you are running Hive Server 1, set the property to `1`.
- If you are running Hive Server 2, set the property to `2`.

For example:

```
HiveServerType=1
```

c. Specify whether the connector uses the ZooKeeper service when connecting to Hive, and provide the necessary connection information:

- To connect to Hive without using the ZooKeeper service do the following:
  a. Set the `ServiceDiscoveryMode` property to `0`.
  b. Set the `Host` property to the IP address or host name of the server.
  c. Set the `Port` property to the number of the TCP port that the server uses to listen for client connections

  For example:

```
ServiceDiscoveryMode=0

Host=192.168.222.160

Port=10000
```

- Or, to discover Hive Server 2 services through the ZooKeeper service, set properties as described in Configuring Service Discovery Mode on a Non-Windows Machine.

d. If authentication is required to access the server, then specify the authentication mechanism and your credentials. For more information, see Configuring Authentication on a Non-Windows Machine.

e. If you want to connect to the server through SSL, then enable SSL and specify the certificate information. For more information, see Configuring SSL Verification in a Non-Windows Machine.

> **Note:** If the `AuthMech` property is set to `2`, SSL is not available.

f. If you want to configure server-side properties, then set them as key-value pairs using a special syntax. For more information, see Configuring Server-Side Properties on a Non-Windows Machine.

g. Optionally, set additional key-value pairs as needed to specify other optional connection settings. For detailed information about all the configuration options supported by the Cloudera ODBC Driver for Apache Hive, see Connector Configuration Options.

4. Save the `odbc.ini` configuration file.

> **Note:** If you are storing this file in its default location in the home directory, then prefix the file name with a period (.) so that the file becomes hidden. If you are storing this file in another location, then save it as a non-hidden file (without the prefix), and make sure that the ODBCINI environment variable specifies the location. For more information, see Specifying the Locations of the Connector Configuration Files.

For example, the following is an `odbc.ini` configuration file for macOS containing a DSN that connects to a Hive Thrift Server instance directly and authenticates the connection using a user name and password:

```
[ODBC Data Sources]

Sample DSN=Cloudera ODBC Driver for Apache Hive

[Sample DSN]

Driver=/opt/cloudera/hiveodbc/lib/universal/libclouderahiveodbc
.dylib

HiveServerType=2

ServiceDiscoveryMode=0

Host=192.168.222.160

Port=10000

UID=username

PWD=userpassword
```

As another example, the following is an `odbc.ini` configuration file for a 32-bit connector on a Linux/AIX/Debian machine, containing a DSN that connects to a HiveThrift Server instance directly and authenticates the connection using a user name and password:

```
[ODBC Data Sources]

Sample DSN=Cloudera ODBC Driver for Apache Hive 32-bit

[Sample DSN]

Driver=/opt/cloudera/hiveodbc/lib/32/libclouderahiveodbc32.so

HiveServerType=2

ServiceDiscoveryMode=0

Host=192.168.222.160

Port=10000

UID=username

PWD=userpassword
```

You can now use the DSN in an application to connect to the data store.

Configuring a DSN-less Connection in a Non-Windows Machine

To connect to your data store through a DSN-less connection, you need to define the connector in the `odbcinst.ini` file and then provide a DSN-less connection string in your application.

If your machine is already configured to use an existing `odbcinst.ini` file, then update that file by adding the settings described below. Otherwise, copy the `odbcinst.ini` file from the `Setup` subfolder in the connector installation directory to the home directory, and then update the file as described below.

To define a connector on a non-Windows machine:

1. In a text editor, open the `odbcinst.ini` configuration file.

   Note: If you are using a hidden copy of the `odbcinst.ini` file, you can remove the period (`.`) from the start of the file name to make the file visible while you are editing it.

2. In the `[ODBC Drivers]` section, add a new entry by typing a name for the connector, an equal sign (=), and then `Installed`.For example:

   ```
   [ODBC Drivers]
   ```

   ```
   Cloudera ODBC Driver for Apache Hive=Installed
   ```

3. Create a section that has the same name as the connector (as specified in the previous step), and then specify the following configuration options as key-value pairs in the section:
   a. Set the `Driver` property to the full path of the connector library file that matches the bitness of the application.

   For example, on a macOS machine:

   ```
   Driver=/opt/cloudera/hiveodbc/lib/universal/libclouderahiveodbc.dylib
   ```

   As another example, for a 32-bit connector on a Linux/AIX/Debian machine:

   ```
   Driver=/opt/cloudera/hiveodbc/lib/32/libclouderahiveodbc32.so
   ```

   b. Optionally, set the `Description` property to a description of the connector. For example:

   ```
   Description=Cloudera ODBC Driver for Apache Hive
   ```

4. Save the `odbcinst.ini` configuration file.

   Note: If you are storing this file in its default location in the home directory, then prefix the file name with a period (`.`) so that the file becomes hidden. If you are storing this file in another location, then save it as a non-hidden file (without the prefix), and make sure that the ODBCINSTINI or ODBCSYSINI environment variable specifies the location. For more information, see Specifying the Locations of the Connector Configuration Files.

   For example, the following is an `odbcinst.ini` configuration file for macOS:

   ```
   [ODBC Drivers]
   ```

   ```
   Description=Cloudera ODBC Driver for Apache Hive
   ```

```
Driver=/opt/cloudera/hiveodbc/lib/universal/libclouderahiveodbc.dylib
```

As another example, the following is an `odbcinst.ini` configuration file for both the 32- and 64-bit connectors in Linux/AIX/Debian:

```
[ODBC Drivers]

Cloudera ODBC Driver for Apache Hive 32-bit=Installed

Cloudera ODBC Driver for Apache Hive 64-bit=Installed

[Cloudera ODBC Driver for Apache Hive 32-bit]

Description=Cloudera ODBC Driver for Apache Hive (32-bit)

Driver=/opt/cloudera/hiveodbc/lib/32/libclouderahiveodbc32.so

[Cloudera ODBC Driver for Apache Hive 64-bit]

Description=Cloudera ODBC Driver for Apache Hive (64-bit)

Driver=/opt/cloudera/hiveodbc/lib/64/libclouderahiveodbc64.so
```

You can now connect to your data store by providing your application with a connection string where the `Driver` property is set to the connector name specified in the `odbcinst.ini` file, and all the other necessary connection properties are also set. For more information, see "DSN-less Connection String Examples" in Using a Connection String.

For detailed information about all the connection properties that the connector supports, see Connector Configuration Options.

Configuring Service Discovery Mode on a Non-Windows Machine

You can configure the Cloudera ODBC Driver for Apache Hive to discover Hive Server 2 services through ZooKeeper.

You can set the connection properties described below in a connection string, in a DSN (in the `odbc.ini` file), or as a connector-wide setting (in the `cloudera.hiveodbc.ini` file). Settings in the

connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

To enable Service Discovery Mode on a non-Windows machine:

1. Set the `ServiceDiscoveryMode` connection attribute to `1`.
2. Set the `ZKNamespace` connection attribute to specify the namespace on ZooKeeper under which Hive Server 2 znodes are added.
3. Set the `Host` connection attribute to specify the ZooKeeper ensemble as a comma-separated list of ZooKeeper servers. For example, type the following, where *[ZK_Host]* is the IP address or host name of the ZooKeeper server and *[ZK_Port]* is the number of the TCP port that the ZooKeeper server uses to listen for client connections:

```
[ZK_Host1]:[ZK_Port1],[ZK_Host2]:[ZK_Port2]
```

Important: When `ServiceDiscoveryMode` is set to `1`, connections to Hive Server 1 are not supported and the `Port` connection attribute is not applicable.

Configuring Authentication on a Non-Windows Machine

Some Hive Server 2 instances are configured to require authentication for access. To connect to a Hive server, you must configure the Cloudera ODBC Driver for Apache Hive to use the authentication mechanism that matches the access requirements of the server and provides the necessary credentials.

For information about how to determine the type of authentication your Hive server requires, see Authentication Mechanisms.

You can set the connection properties for authentication in a connection string, in a DSN (in the `odbc.ini` file), or as a connector-wide setting (in the `cloudera.hiveodbc.ini` file). Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

Depending on the authentication mechanism you use, there might be additional connection attributes that you must define. For more information about the attributes involved in configuring authentication, see Connector Configuration Options .

If cookie-based authentication is enabled in your Hive Server 2 database, you can specify a list of authentication cookies in the `HTTPAuthCookies` connection property. In this case, the connector authenticates the connection once based on the provided authentication credentials. It then uses the cookie generated by the server for each subsequent request in the same connection. For more information, see HTTPAuthCookies.

Using No Authentication

When connecting to a Hive server of type Hive Server 1, you must use No Authentication. When you use No Authentication, Binary is the only Thrift transport protocol that is supported.

To configure a connection without authentication:

1. Set the `AuthMech` connection attribute to `0`.
2. If the Hive server is configured to use SSL, then configure SSL for the connection. For more information, see Configuring SSL Verification in a Non-Windows Machine.

Using Kerberos

Kerberos must be installed and configured before you can use this authentication mechanism. For more information, refer to the MIT Kerberos Documentation: http://web.mit.edu/kerberos/krb5-latest/doc/.

This authentication mechanism is available only for Hive Server 2. When you use Kerberos authentication, the Binary transport protocol is not supported.

To configure Kerberos authentication:

1. Set the `AuthMech` connection attribute to `1`.
2. Choose one:
   - To use the default realm defined in your Kerberos setup, do not set the `KrbRealm` attribute.
   - Or, if your Kerberos setup does not define a default realm or if the realm of your Hive server is not the default, then set the appropriate realm using the `KrbRealm` attribute.
3. Optionally, if you are using MIT Kerberos and a Kerberos realm is specified using the `KrbRealm` connection attribute, then choose one:
   - To have the Kerberos layer canonicalize the server's service principal name, leave the `ServicePrincipalCanonicalization` attribute set to `1`.
   - Or, to prevent the Kerberos layer from canonicalizing the server's service principal name, set the `ServicePrincipalCanonicalization` attribute to `0`.
4. Set the `KrbHostFQDN` attribute to the fully qualified domain name of the Hive Server 2 host.

   Note: To use the Hive server host name as the fully qualified domain name for Kerberos authentication, set `KrbHostFQDN` to `_HOST`.
5. Set the `KrbServiceName` attribute to the service name of the Hive .
6. To allow the connector to pass your credentials directly to the server for use in authentication, set `DelegateKrbCreds` to `1`.
7. Set the `ThriftTransport` connection attribute to the transport protocol to use in the Thrift layer.

   Important:
   When using this authentication mechanism, Binary (`ThriftTransport=0`) is not supported.
8. If the Hive server is configured to use SSL, then configure SSL for the connection. For more information, see Configuring SSL Verification in a Non-Windows Machine.

Using SAML 2.0

This authentication mechanism enables you to authenticate via Single Sign-On using SAML 2.0 against supported servers.

Important: In order to use SAML 2.0 for authentication, the `ThriftTransport` attribute must be set to `2` and the `SSL` attribute must be set to `1`.

To configure SAML 2.0 authentication:

1. Set the `AuthMech` connection attribute to `12`.
2. Set the `ThriftTransport` attribute to `2`.
3. Set the `HttpPath` attribute to the partial URL corresponding to the Hive server.
4. Set the `SSL` attribute to `1`.
5. Optionally, set the `SSOIgnoreDriverNoPrompt` attribute to `true`. When the application is making a SQLDriverConnect call with a SQL_DRIVER_NOPROMPT flag, this property displays the web browser used to complete the browser based authentication flow.
6. Optionally, set the `TSaslTransportBufSize` attribute to the number of bytes to reserve in memory for buffering unencrypted data from the network.

   Note: In most circumstances, the default value of 1000 bytes is optimal.

Using User Name

This authentication mechanism requires a user name but does not require a password. The user name labels the session, facilitating database tracking.

This authentication mechanism is available only for Hive Server 2. Most default configurations of Hive Server 2 require User Name authentication. When you use User Name authentication, SSL is not supported and SASL is the only Thrift transport protocol available.

To configure User Name authentication:

1. Set the `AuthMech` connection attribute to `2`.
2. Set the `UID` attribute to an appropriate user name for accessing the Hive server.

Using User Name And Password

This authentication mechanism requires a user name and a password.

This authentication mechanism is available only for Hive Server 2.

To configure User Name And Password authentication:

1. Set the `AuthMech` connection attribute to `3`.
2. Set the `UID` attribute to an appropriate user name for accessing the Hive server.
3. Set the `PWD` attribute to the password corresponding to the user name you provided above.

4. Set the `ThriftTransport` connection attribute to the transport protocol to use in the Thrift layer.
5. If the Hive server is configured to use SSL, then configure SSL for the connection. For more information, see Configuring SSL Verification in a Non-Windows Machine.

Delegating Authentication to a Specific User

Some Hive Server 2 instances support the ability to delegate all operations against Hive to the specified user, rather than to the authenticated user for the connection.

To delegate all operations to a specified user:

- Set the `DelegationUID` configuration option to the name of the user to be delegated.

If the server returns an error message such as **Failed to validate proxy privilege of *[RealUser]* for *[DelegationUID]***, you may need to modify the server's `core-site.xml` configuration file, as follows:

1. In the server's `core-site.xml` configuration file, add the following properties:

```
hadoop.proxyuser.[RealUser].groups=*

hadoop.proxyuser.[RealUser].hosts=*
```

   Where *[RealUser]* is the authenticated user for the connection.

2. If you are using Kerberos authentication, then in the server's `core-site.xml` configuration file, add the following properties:

```
hadoop.proxyuser.[Principal].groups=*

hadoop.proxyuser.[Principal].hosts=*
```

   Where *[Principal]* is the primary Kerberos principal user. For example, if the primary Kerberos principal user is `kerbuser@example.com`, replace *[Principal]* with `kerbuser`.

For more information on resolving this error, see your Hive Server documentation.

Configuring SSL Verification in a Non-Windows Machine

If you are connecting to a Hive server that has Secure Sockets Layer (SSL) enabled, you can configure the connector to connect to an SSL-enabled socket. When using SSL to connect to a server, the connector supports identity verification between the client (the connector itself) and the server.

Note: If the `AuthMech` property is set to `2`, SSL is not available.

You can set the connection properties described below in a connection string, in a DSN (in the `odbc.ini` file), or as a connector-wide setting (in the `cloudera.hiveodbc.ini` file). Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

To configure SSL verification on a non-Windows machine:

1. To enable SSL connections, set the `SSL` attribute to `1`.
2. To allow authentication using self-signed certificates that have not been added to the list of trusted certificates, set the `AllowSelfSignedServerCert` attribute to `1`.
3. To allow the common name of a CA-issued SSL certificate to not match the host name of the Hive server, set the `AllowHostNameCNMismatch` attribute to `1`.
4. Choose one:
   - To configure the connector to load SSL certificates from a specific `.pem` file when verifying the server, set the `TrustedCerts` attribute to the full path of the `.pem` file.
   - Or, to use the trusted CA certificates `.pem` file that is installed with the connector, do not specify a value for the `TrustedCerts` attribute.
5. To configure two-way SSL verification, set the `TwoWaySSL` attribute to `1` and then do the following:
   a. Set the `ClientCert` attribute to the full path of the `.pem` file containing the client's certificate.
   b. Set the `ClientPrivateKey` attribute to the full path of the file containing the client's private key.
   c. If the private key file is protected with a password, set the `ClientPrivateKeyPassword` attribute to the password.
6. To allow authentication, when the certificate's revocation status is undetermined, set the Accept Undetermined Revocation attribute to `1`.
7. To specify the minimum version of TLS to use, set the `Min_TLS` property to the minimum version of TLS. Supported options include `1.0` for TLS 1.0, `1.1` for TLS 1.1, and `1.2` for TLS 1.2

Configuring Server-Side Properties on a Non-Windows Machine

You can use the connector to apply configuration properties to the Hive server.

You can set the connection properties described below in a connection string, in a DSN (in the `odbc.ini` file), or as a connector-wide setting (in the `cloudera.hiveodbc.ini` file). Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

To configure server-side properties on a non-Windows machine:

1. To set a server-side property, use the syntax `SSP_[SSPKey]=[SSPValue]`, where *[SSPKey]* is the name of the server-side property and *[SSPValue]* is the value to specify for that property. For example, to set the `mapreduce.job.queuename` property to `myQueue`, type the following:

```
SSP_mapreduce.job.queuename=myQueue
```

Note:

- When setting a server-side property in a connection string, it is recommended that you enclose the value in braces (`{ }`) to make sure that special characters can be properly escaped.
- For a list of all Hadoop and Hive server-side properties that your implementation supports, type `set -v` at the Hive CLI command lineor Beeline. You can also execute the `set -v` query after connecting using the connector.

2. To change the method that the connector uses to apply server-side properties, do one of the following:

- To configure the connector to apply each server-side property by executing a query when opening a session to the Hive server, set the `ApplySSPWithQueries` property to `1`.
- Or, to configure the connector to use a more efficient method for applying server-side properties that does not involve additional network round-tripping, set the `ApplySSPWithQueries` property to `0`.

Note: The more efficient method is not available for Hive Server 1, and it might not be compatible with some Hive Server builds. If the server-side properties do not take effect when the `ApplySSPWithQueries` property is set to `0`, then set it to `1`.

3. To disable the connector's default behavior of converting server-side property key names to all lower-case characters, set the `LCaseSspKeyName` property to `0`.

Configuring Logging Options

To help troubleshoot issues, you can enable logging in the connector.

Important: Only enable logging long enough to capture an issue. Logging decreases performance and can consume a large quantity of disk space.

You can set the connection properties described below in a connection string, in a DSN (in the `odbc.ini` file), or as a connector-wide setting (in the `cloudera.hiveodbc.ini` file). Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

To enable logging:

1. To specify the level of information to include in log files, set the `LogLevel` property to one of the following numbers:

LogLevel Value	Description
0	Disables all logging.

LogLevel Value	Description
1	Logs severe error events that lead the connector to abort.
2	Logs error events that might allow the connector to continue running.
3	Logs events that might result in an error if action is not taken.
4	Logs general information that describes the progress of the connector.
5	Logs detailed information that is useful for debugging the connector.
6	Logs all connector activity.

2. Set the `LogPath` key to the full path to the folder where you want to save log files.
3. Set the `LogFileCount` key to the maximum number of log files to keep.

   Note: After the maximum number of log files is reached, each time an additional file is created, the connector deletes the oldest log file.

4. Set the `LogFileSize` key to the maximum size of each log file in bytes.

   Note: After the maximum file size is reached, the connector creates a new file and continues logging.

5. Save the `cloudera.hiveodbc.ini` configuration file.
6. Restart your ODBC application to make sure that the new settings take effect.

The Cloudera ODBC Driver for Apache Hive produces the following log files at the location you specify using the `LogPath` key:

- A `clouderaodbcdriverforapachehive.log` file that logs connector activity that is not specific to a connection.
- A `clouderaodbcdriverforapachehive_connection_[Number].log`file for each connection made to the database, where *[Number]* is a number that identifies each log file. This file logs connector activity that is specific to the connection.

To disable logging:

1. Set the `LogLevel` key to `0`.
2. Save the `cloudera.hiveodbc.ini` configuration file.
3. Restart your ODBC application to make sure that the new settings take effect.

Setting Connector-Wide Configuration Options on a Non-Windows Machine

When you specify connection settings in a DSN or connection string, those settings apply only when you connect to Hive using that particular DSN or string. As an alternative, you can specify settings that apply to every connection that uses the Cloudera ODBC Driver for Apache Hive by configuring them in the `cloudera.hiveodbc.ini` file.

Note: Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

To set connector-wide configuration options on a non-Windows machine:

1. In a text editor, open the `cloudera.hiveodbc.ini` configuration file.
2. In the `[Driver]` section, specify configuration options as key-value pairs. Start a new line for each key-value pair.

   For example, to enable User Name authentication using "cloudera" as the user name, type the following:

   ```
   AuthMech=2
   ```

   ```
   UID=cloudera
   ```

   For detailed information about all the configuration options supported by the connector, see Connector Configuration Options.
3. Save the `cloudera.hiveodbc.ini` configuration file.

Testing the Connection

To test the connection, you can use an ODBC-enabled client application. For a basic connection test, you can also use the test utilities that are packaged with your driver manager installation. For example, the iODBC driver manager includes simple utilities called `iodbctest` and `iodbctestw`. Similarly, the unixODBC driver manager includes simple utilities called isql and iusql.

Using the iODBC Driver Manager

You can use the iodbctest and iodbctestw utilities to establish a test connection with your connector. Use iodbctest to test how your connector works with an ANSI application, or use iodbctestw to test how your connector works with a Unicode application.

Note: There are 32-bit and 64-bit installations of the iODBC driver manager available. If you have only one or the other installed, then the appropriate version of iodbctest (or iodbctestw) is available. However, if you have both 32- and 64-bit versions installed, then you need to make sure that you are running the version from the correct installation directory.

For more information about using the iODBC driver manager, see http://www.iodbc.org.

To test your connection using the iODBC driver manager:

1. Run **iodbctest** or **iodbctestw**.
2. Optionally, if you do not remember the DSN, then type a question mark (`?`) to see a list of available DSNs.
3. Type the connection string for connecting to your data store, and then press ENTER. For more

information, see .

If the connection is successful, then the `SQL>` prompt appears.

Using the unixODBC Driver Manager

You can use the isql and iusql utilities to establish a test connection with your connector and your DSN. isql and iusql can only be used to test connections that use a DSN. Use isql to test how your connector works with an ANSI application, or use iusql to test how your connector works with a Unicode application.

Note: There are 32-bit and 64-bit installations of the unixODBC driver manager available. If you have only one or the other installed, then the appropriate version of isql (or iusql) is available. However, if you have both 32- and 64-bit versions installed, then you need to make sure that you are running the version from the correct installation directory.

For more information about using the unixODBC driver manager, see http://www.unixodbc.org.

To test your connection using the unixODBC driver manager:

- Run isql or iusql by using the corresponding syntax:

  - `isql [DataSourceName]`
  - `iusql [DataSourceName]`

`[DataSourceName]` is the DSN that you are using for the connection.

If the connection is successful, then the `SQL>` prompt appears.

Note: For information about the available options, run isql or iusql without providing a DSN.

Authentication Mechanisms

To connect to a Hive server, you must configure the Cloudera ODBC Driver for Apache Hive to use the authentication mechanism that matches the access requirements of the server and provides the necessary credentials. To determine the authentication settings that your Hive server requires, check the server configuration and then refer to the corresponding section below.

Hive Server 1

You must use No Authentication as the authentication mechanism. Hive Server 1 instances do not support authentication.

Hive Server 2

Note: Most default configurations of Hive Server 2 require User Name authentication.

Configuring authentication for a connection to a Hive Server 2 instance involves setting the authentication mechanism, the Thrift transport protocol, and SSL support. To determine the settings that you need to use, check the following three properties in the `hive-site.xml` file in the Hive server that you are connecting to:

- `hive.server2.authentication`
- `hive.server2.transport.mode`
- `hive.server2.use.SSL`

Use the following table to determine the authentication mechanism that you need to configure, based on the `hive.server2.authentication` value in the `hive-site.xml` file:

hive.server2.authentication	Authentication Mechanism
NOSASL	No Authentication
KERBEROS	Kerberos
NONE	User Name
LDAP	User Name and Password
SAML	SAML 2.0

Use the following table to determine the Thrift transport protocol that you need to configure, based on the `hive.server2.authentication` and `hive.server2.transport.mode` values in the `hive-site.xml` file:

hive.server2.authentication	hive.server2.transport.mode	Thrift Transport Protocol
NOSASL	binary	Binary
KERBEROS	binary or http	SASL or HTTP
NONE	binary or http	SASL or HTTP

hive.server2.authentication	hive.server2.transport.mode	Thrift Transport Protocol
LDAP	binary or http	SASL or HTTP
SAML	http	HTTP

To determine whether SSL should be enabled or disabled for your connection, check the `hive.server2.use.SSL` value in the `hive-site.xml` file. If the value is true, then you must enable and configure SSL in your connection. If the value is false, then you must disable SSL in your connection.

For detailed instructions on how to configure authentication when using the Windows connector, see Configuring Authentication in Windows.

For detailed instructions on how to configure authentication when using a non-Windows connector, see Configuring Authentication on a Non-Windows Machine.

Using a Connection String

For some applications, you might need to use a connection string to connect to your data source. For detailed information about how to use a connection string in an ODBC application, refer to the documentation for the application that you are using.

The connection strings in the following sections are examples showing the minimum set of connection attributes that you must specify to successfully connect to the data source. Depending on the configuration of the data source and the type of connection you are working with, you might need to specify additional connection attributes. For detailed information about all the attributes that you can use in the connection string, see Connector Configuration Options.

DSN Connection String Example

The following is an example of a connection string for a connection that uses a DSN:

DSN=*[DataSourceName]*

[DataSourceName] is the DSN that you are using for the connection.

You can set additional configuration options by appending key-value pairs to the connection string. Configuration options that are passed in using a connection string take precedence over configuration options that are set in the DSN.

DSN-less Connection String Examples

Some applications provide support for connecting to a data source using a connector without a DSN. To connect to a data source without using a DSN, use a connection string instead.

The placeholders in the examples are defined as follows, in alphabetical order:

- *[DomainName]* is the fully qualified domain name of the Hive server host.
- *[Namespace]* is the namespace on ZooKeeper under which Hive Server 2 znodes are added.
- *[PortNumber]* is the number of the TCP port that the Hive server uses to listen for client connections.
- *[Realm]* is the Kerberos realm of the Hive server host.
- *[Server]* is the IP address or host name of the Hive server to which you are connecting.
- *[ServiceName]* is the Kerberos service principal name of the Hive server.
- *[YourPassword]* is the password corresponding to your user name.
- *[YourUserName]* is the user name that you use to access the Hive server.

Connecting to a Hive Server 1 Instance

The following is the format of a DSN-less connection string that connects to a Hive Server 1 instance:

```
Driver=Cloudera ODBC Driver for Apache Hive;HiveServerType=1;
Host=[Server];Port=[PortNumber];
```

For example:

```
Driver=Cloudera ODBC Driver for Apache Hive;HiveServerType=1;
Host=192.168.222.160;Port=10000;
```

Connecting to a Standard Hive Server 2 Instance

The following is the format of a DSN-less connection string for a standard connection to a Hive Server 2 instance. By default, the connector is configured to connect to a Hive Server 2 instance that requires User Name authentication, and the connector uses **anonymous** as the user name.

```
Driver=Cloudera ODBC Driver for Apache Hive ;Host=[Server];Port=
[PortNumber];
```

For example:

```
Driver=Cloudera ODBC Driver for Apache
Hive;Host=192.168.222.160;Port=10000;
```

Connecting using Dynamic Service Discovery

The following is the format of a DSN-less connection string that discovers Hive Server 2 services via the ZooKeeper service.

```
Driver=Cloudera ODBC Driver for Apache
Hive;ServiceDiscoveryMode=1;Host=[Server1]:[PortNumber1], [Server2]:
[PortNumber2], [Server3]:[PortNumber3];ZKNamespace=[Namespace];
```

For example:

```
Driver=Cloudera ODBC Driver for Apache Hive;ServiceDiscoveryMode=1;
Host=192.168.222.160:10000, 192.168.222.165:10000,
192.168.222.231:10000;ZKNamespace=hiveserver;
```

Connecting to a Hive Server 2 Instance Without Authentication

The following is the format of a DSN-less connection string for a Hive Server 2 instance that does not require authentication.

```
Driver=Cloudera ODBC Driver for Apache Hive;Host=[Server];Port=
[PortNumber];AuthMech=0;
```

For example:

```
Driver=Cloudera ODBC Driver for Apache
Hive;Host=192.168.222.160;Port=10000;AuthMech=0;
```

Connecting to a Hive Server that Requires Kerberos Authentication

The following is the format of a DSN-less connection string that connects to a Hive Server 2 instance requiring Kerberos authentication:

```
Driver=Cloudera ODBC Driver for Apache Hive;Host=[Server];Port=
[PortNumber];AuthMech=1;KrbRealm=[Realm];KrbHostFQDN=
[DomainName];KrbServiceName=[ServiceName];
```

For example:

```
Driver=Cloudera ODBC Driver for Apache
Hive;Host=192.168.222.160;Port=10000;AuthMech=1;KrbRealm=CLOUDERA
;KrbHostFQDN=localhost.localdomain;KrbServiceName=hive;
```

Connecting to a Hive Server using SAML 2.0

The following is the format of a DSN-less connection string that connects to a Hive Server 2 instance via Single Sign-On using SAML 2.0 authentication:

```
Driver=Cloudera ODBC Driver for Apache Hive;Host=[Server];Port=
[PortNumber];AuthMech=12;ThriftTransport=2;HTTPPath=
[ServerURL];SSL=1;
```

For example:

```
Driver=Cloudera ODBC Driver for Apache Hive
;Host=192.168.222.160;Port=10000;AuthMech=12;ThriftTransport=2;HTTPPa
th=cliservice;SSL=1;
```

Connecting to a Hive Server that Requires User Name And Password Authentication (LDAP)

The following is the format of a DSN-less connection string that connects to a Hive Server 2 instance requiring User Name And Password authentication:

```
Driver=Cloudera ODBC Driver for Apache Hive;Host=[Server];Port=
[PortNumber];AuthMech=3;UID=[YourUserName];
PWD=[YourPassword];
```

For example:

```
Driver=Cloudera ODBC Driver for Apache
Hive;Host=192.168.222.160;Port=10000;AuthMech=3;UID=cloudera;PWD=
cloudera;
```

Features

For more information on the features of the Cloudera ODBC Driver for Apache Hive, see the following:

- SQL Connector for HiveQL
- Data Types
- Catalog and Schema Support
- hive_system Table
- Server-Side Properties
- Get Tables With Query
- Active Directory
- Write-back
- Dynamic Service Discovery using ZooKeeper
- Security and Authentication

SQL Connector for HiveQL

The native query language supported by Hive is HiveQL. For simple queries, HiveQL is a subset of SQL-92. However, the syntax is different enough that most applications do not work with native HiveQL.

To bridge the difference between SQL and HiveQL, the SQL Connector feature translates standard SQL-92 queries into equivalent HiveQL queries. The SQL Connector performs syntactical translations and structural transformations. For example:

- **Quoted Identifiers**: The double quotes (") that SQL uses to quote identifiers are translated into back quotes (`) to match HiveQL syntax. The SQL Connector needs to handle this translation because even when a connector reports the back quote as the quote character, some applications still generate double-quoted identifiers.
- **Table Aliases**: Support is provided for the AS keyword between a table reference and its alias, which HiveQL normally does not support.
- **JOIN**, **INNER JOIN**, and **CROSS JOIN**: SQL JOIN, INNER JOIN, and CROSS JOIN syntax is translated to HiveQL JOIN syntax.
- **TOP N/LIMIT**: SQL TOP N queries are transformed to HiveQL LIMIT queries.

Data Types

The Cloudera ODBC Driver for Apache Hive supports many common data formats, converting between Hive data types and SQL data types.

The following table lists the supported data type mappings.

Hive Type	SQL Type
BIGINT	SQL_BIGINT
BINARY	SQL_VARBINARY
BOOLEAN	SQL_BIT
CHAR(n)	SQL_CHAR **Note:** SQL_WCHAR is returned instead if the Unicode SQL Character Types configuration option (the `UseUnicodeSqlCharacterTypes` key) is enabled.
DATE	SQL_TYPE_DATE
DECIMAL	SQL_DECIMAL
DECIMAL(p,s)	SQL_DECIMAL
DOUBLE	SQL_DOUBLE
FLOAT	SQL_REAL
INT	SQL_INTEGER
SMALLINT	SQL_SMALLINT
STRING	SQL_VARCHAR **Note:** SQL_WVARCHAR is returned instead if the Unicode SQL Character Types configuration option (the `UseUnicodeSqlCharacterTypes` key) is enabled.
TIMESTAMP	SQL_TYPE_TIMESTAMP
TINYINT	SQL_TINYINT
VARCHAR(n)	SQL_VARCHAR

Note: The aggregate types (ARRAY, MAP, and STRUCT) are not supported. Columns of aggregate types are treated as STRING columns. The interval types (YEAR TO MONTH and DAY TIME) are supported only in query expressions and predicates. Interval types are not supported as column data types in tables.

Catalog and Schema Support

The Cloudera ODBC Driver for Apache Hive supports both catalogs and schemas to make it easy for the connector to work with various ODBC applications. Since Hive only organizes tables into schemas/databases, the connector provides a synthetic catalog named HIVE under which all of the schemas/databases are organized. The connector also maps the ODBC schema to the Hive schema/database.

hive_system Table

A pseudo-table called hive_system can be used to query for Hive cluster system environment information. The pseudo-table is under the pseudo-schema called hive_system. The table has two STRING type columns, envkey and envvalue. Standard SQL can be executed against the hive_system table. For example:

```
SELECT * FROM HIVE.hive_system.hive_system WHERE envkey LIKE '%hive%'
```

The above query returns all of the Hive system environment entries whose key contains the word "hive". A special query, `set -v`, is executed to fetch system environment information. Some versions of Hive do not support this query. For versions of Hive that do not support querying system environment information, the connector returns an empty result set.

Server-Side Properties

The Cloudera ODBC Driver for Apache Hive allows you to set server-side properties via a DSN. Server-side properties specified in a DSN affect only the connection that is established using the DSN.

You can also specify server-side properties for connections that do not use a DSN. To do this, use the Driver Configuration tool that is installed with the Windows version of the connector, or set the appropriate configuration options in your connection string or the `cloudera.hiveodbc.ini` file. Properties specified in the connector configuration tool or the `cloudera.hiveodbc.ini` file apply to all connections that use the Cloudera ODBC Driver for Apache Hive.

For more information about setting server-side properties when using the Windows connector, see Configuring Server-Side Properties in Windows. For information about setting server-side properties when using the connector on a non-Windows platform, see Configuring Server-Side Properties on a Non-Windows Machine.

Temporary Tables

The driver supports the creation of temporary tables and lets you insert literal values into temporary tables. Temporary tables are only accessible by the ODBC connection that created them, and are dropped when the connection is closed.

CREATE TABLE Statement for Temporary Tables

The driver supports the following DDL syntax for creating temporary tables:

<create table statement> := CREATE TABLE <temporary table name> <left paren><column definition list><right paren>

<column definition list> := <column definition>[, <column definition>]*

<column definition> := <column name> <data type>

<temporary table name> := <double quote><number sign><table name><double quote>

<left paren> := (

<right paren> :=)

<double quote> := "

<number sign> := #

The following is an example of a SQL statement for creating a temporary table:

CREATE TABLE "#TEMPTABLE1" (C1 DATATYPE_1, C2 DATATYPE_2, ..., Cn DATATYPE_n)

The temporary table name in a SQL query must be surrounded by double quotes ("), and the name must begin with a number sign (#).

Note:

You can only use data types that are supported by Hive.

INSERT Statement for Temporary Tables

The driver supports the following DDL syntax for inserting data into temporary tables:

<insert statement> := INSERT INTO <temporary table name> <left paren><column name list><right paren> VALUES <left paren><literal value list><right paren>

<column name list> := <column name>[, <column name>]*

<literal value list> := <literal value>[, <literal value>]*

<temporary table name> := <double quote><number sign><table name><double quote>

<left paren> := (

<right paren> :=)

<double quote> := "

<number sign> := #

The following is an example of a SQL statement for inserting data into temporary tables:

INSERT INTO "#TEMPTABLE1" values (VAL(C1), VAL(C2) ... VAL(Cn))

VAL(C1) is the literal value for the first column in the table, and VAL(Cn) is the literal value for the nth column in the table.

Note:

The INSERT statement is only supported for temporary tables.

Get Tables With Query

The Get Tables With Query configuration option allows you to choose whether to use the SHOW TABLES query or the GetTables API call to retrieve table names from a database.

Active Directory

The Cloudera ODBC Driver for Apache Hive supports Active Directory Kerberos in Windows. There are two prerequisites for using Active Directory Kerberos in Windows:

- MIT Kerberos is not installed on the client Windows machine.
- The MIT Kerberos Hadoop realm has been configured to trust the Active Directory realm so that users in the Active Directory realm can access services in the MIT Kerberos Hadoop realm.

Write-back

The Cloudera ODBC Driver for Apache Hive supports translation for the following syntax when connecting to a Hive Server 2 instance:

- INSERT
- UPDATE
- DELETE
- CREATE
- DROP

If the statement contains non-standard SQL-92 syntax, then the connector is unable to translate the statement to SQL and instead falls back to using HiveQL.

Timestamp Function Support

The Cloudera ODBC Driver for Apache Hive supports the following ODBC functions for working with data of type TIMESTAMP:

- **TIMESTAMPADD**: You can call this function to increment a TIMESTAMP value by a specified interval of time.
- **TIMESTAMPDIFF**: You can call this function to calculate the interval of time between two specified TIMESTAMP values.

The types of time intervals that are supported for these functions might vary depending on the Hive server version that you are connecting to. To return a list of the intervals supported for TIMESTAMPADD, call the SQLGetInfo catalog function using SQL_TIMEDATE_ADD_INTERVALS as the argument. Similarly, to return a list of the intervals supported for TIMESTAMPDIFF, call SQLGetInfo using SQL_TIMEDATE_DIFF_INTERVALS as the argument.

Note: The SQL_TSI_FRAC_SECOND interval is not supported by Hive.

Dynamic Service Discovery using ZooKeeper

The Cloudera ODBC Driver for Apache Hive can be configured to discover Hive Server 2 services via the ZooKeeper service.

For information about configuring this feature in the Windows connector, see Creating a Data Source Name in Windows or Configuring a DSN-less Connection in Windows. For information about configuring

this feature when using the connector on a non-Windows platform, see Configuring Service Discovery Mode on a Non-Windows Machine.

Security and Authentication

To protect data from unauthorized access, some Hive data stores require connections to be authenticated with user credentials or encrypted using the SSL protocol. The Cloudera ODBC Driver for Apache Hive provides full support for these authentication protocols.

Note: In this documentation, "SSL" refers to both TLS (Transport Layer Security) and SSL (Secure Sockets Layer). The connector supports TLS 1.0, 1.1, and 1.2. The SSL version used for the connection is the highest version that is supported by both the connector and the server.

The connector provides mechanisms that enable you to authenticate your connection using the Kerberos protocol, your Hive user name only, or your Hive user name and password. You must use the authentication mechanism that matches the security requirements of the Hive server. For information about determining the appropriate authentication mechanism to use based on the Hive server configuration, see Authentication Mechanisms. For detailed connector configuration instructions, see Configuring Authentication in Windows or Configuring Authentication on a Non-Windows Machine.

Additionally, the connector supports the following types of SSL connections:

- No identity verification
- One-way authentication
- Two-way authentication

It is recommended that you enable SSL whenever you connect to a server that is configured to support it. SSL encryption protects data and credentials when they are transferred over the network, and provides stronger security than authentication alone. For detailed configuration instructions, see Configuring SSL Verification in Windows or Configuring SSL Verification in a Non-Windows Machine.

Connector Configuration Options

Connector Configuration Options lists the configuration options available in the Cloudera ODBC Driver for Apache Hive alphabetically by field or button label. Options having only key names, that is, not appearing in the user interface of the connector, are listed alphabetically by key name.

When creating or configuring a connection from a Windows machine, the fields and buttons are available in the Driver Configuration tool and the following dialog boxes:

- Cloudera ODBC Driver for Apache Hive DSN Setup
- Advanced Options
- HTTP Proxy Options
- Server Side Properties
- SSL Options
- HTTP Properties

When using a connection string or configuring a connection from a non-Windows machine, use the key names provided.

Note:
If you are using the connector on a non-Windows machine, you can set connector configuration properties in a connection string, in a DSN (in the `odbc.ini` file), or as a connector-wide setting (in the `cloudera.hiveodbc.ini` file). Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

Configuration Options Appearing in the User Interface

The following configuration options are accessible via the Windows user interface for the Cloudera ODBC Driver for Apache Hive, or via the key name when using a connection string or configuring a connection from a Linux/macOS/AIX/Debian machine:

- Allow Common Name Host Name Mismatch
- Allow Self-Signed Server Certificate
- Apply Properties With Queries
- Authentication Flow
- Binary Column Length
- Log Path
- Max File Size
- Max Number Files
- Mechanism
- Minimum TLS

- Ignore SQL_DRIVER_NOPROMPT
- Ignore Tables Metadata From All Schemas
- Invalid Session Auto Recover
- Log Level
- Web HDFS Host
- Web HDFS Port
- ZooKeeper Namespace

Allow Common Name Host Name Mismatch

This option specifies whether a CA-issued SSL certificate name must match the host name of the Hive server.

Note: The key for this option used to be `CAIssuedCertNamesMismatch`, and is still recognized by the connector under that key. If both keys are defined, `AllowHostNameCNMismatch` will take precedence.

This setting is applicable only when SSL is enabled.

- Enabled (`1`): The connector allows a CA-issued SSL certificate name to not match the host name of the Hive server.
- Disabled (`0`): The CA-issued SSL certificate name must match the host name of the Hive server.

Key Name	Default Value	Required
`AllowHostNameCNMismatch`	Clear (`0`)	No

Allow Self-Signed Server Certificate

This option specifies whether the connector allows a connection to aHive server that uses a self-signed certificate, even if this certificate is not in the list of trusted certificates. This list is contained in the Trusted Certificates file, or in the system Trust Store if the system Trust Store is used instead of a file.

- Enabled (`1`): The connector authenticates the Hive server even if the server is using a self-signed certificate that has not been added to the list of trusted certificates.
- Disabled (`0`): The connector does not allow self-signed certificates from the server unless they have already been added to the list of trusted certificates.

Note: This setting is applicable only when SSL is enabled.

Key Name	Default Value	Required
`AllowSelfSignedServer Cert`	Clear (`0`)	No

Apply Properties With Queries

This option specifies how the connector applies server-side properties.

- Enabled (`1`): The connector applies each server-side property by executing a `set *SSPKey=SSPValue*` query when opening a session to the Hive server.
- Disabled (`0`): The connector uses a more efficient method for applying server-side properties that does not involve additional network round-tripping. However, some Hive Server 2 builds are not compatible with the more efficient method.

Note: When connecting to a Hive Server 1 instance, this option is always enabled.

Key Name	Default Value	Required
`ApplySSPWithQueries`	Selected (`1`)	No

Authentication Flow

This property specifies the type of authentication flow that the connector uses when the Mechanism option is set to SAML_2.0. Currently, the only supported value is `browser`.

Note:
The browser workflow is only supported in a GUI desktop environment in Windows, Linux, and macOS.

Key Name	Default Value	Required
Auth_Flow	`browser`	No

Binary Column Length

The maximum data length for BINARY columns.

By default, the columns metadata for Hive does not specify a maximum data length for BINARY columns.

Key Name	Default Value	Required
`BinaryColumnLength`	`32767`	No

Canonicalize Principal FQDN

This option specifies whether the Kerberos layer canonicalizes the host FQDN in the server's service principal name.

- Enabled (`1`): The Kerberos layer canonicalizes the host FQDN in the server's service principal name.

- Disabled (`0`): The Kerberos layer does not canonicalize the host FQDN in the server's service principal name.

Note: This option only affects MIT Kerberos, and is ignored when using Active Directory Kerberos. It can only be disabled if the Kerberos Realm or `KrbRealm` key is specified.

Key Name	Default Value	Required
ServicePrincipal Canonicalization	Selected (`1`)	No

CheckCertificate Revocation

This option specifies whether the connector checks to see if a certificate has been revoked while retrieving a certificate chain from the Windows Trust Store.

This option is only applicable if you are using a CA certificate from the Windows Trust Store (see Use System Trust Store).

- Enabled (`1`): The connector checks for certificate revocation while retrieving a certificate chain from the Windows Trust Store.
- Disabled (`0`): The connector does not check for certificate revocation while retrieving a certificate chain from the Windows Trust Store.

Note: This option is disabled when the `AllowSelfSignedServerCert` property is set to `1`. This option is only available in Windows.

Key Name	Default Value	Required
`CheckCertRevocation`	Selected (`1`)	No

Client Certificate File

The full path to the `.pem` file containing the client's SSL certificate.

Note: This setting is applicable only when two-way SSL is enabled.

Key Name	Default Value	Required
`ClientCert`	None	No

Client Private Key File

The full path to the `.pem` file containing the client's SSL private key.

If the private key file is protected with a password, then provide the password using the connector configuration option Client Private Key Password.

Note: This setting is applicable only when two-way SSL is enabled.

Key Name	Default Value	Required
ClientPrivateKey	None	Yes, if two-way SSL verification is enabled.

Client Private Key Password

The password of the private key file that is specified in the Client Private Key File field (ClientPrivateKey).

Key Name	Default Value	Required
ClientPrivateKeyPassword	None	Yes, if two-way SSL verification is enabled and the client's private key file is protected with a password.

Convert Key Name to Lower Case

This option specifies whether the connector converts server-side property key names to all lower-case characters.

- Enabled (1): The connector converts server-side property key names to all lower-case characters.
- Disabled (0): The connector does not modify the server-side property key names.

Key Name	Default Value	Required
LCaseSspKeyName	Selected (1)	No

Data File HDFS Dir

The HDFS directory that the driver uses to store the necessary files for supporting the Temporary Table feature.

Note:
Due to a problem in Hive (see https://issues.apache.org/jira/browse/HIVE-4554), HDFS paths with space characters do not work with versions of Hive prior to 0.12.0.

This option is not applicable when connecting to Hive 0.14 or later.

Key Name	Default Value	Required
HDFSTempTableDir	/tmp/simba	No

Database

The name of the database schema to use when a schema is not explicitly specified in a query. You can still issue queries on other schemas by explicitly specifying the schema in the query.

Note: To inspect your databases and determine the appropriate schema to use, at the Hive command prompt, type `show databases`.

Key Name	Default Value	Required
`Schema`	`default`	No

Decimal Column Scale

The maximum number of digits to the right of the decimal point for numeric data types.

Key Name	Default Value	Required
`DecimalColumnScale`	`10`	`No`

Default String Column Length

The maximum number of characters that can be contained in STRING columns.

By default, the columns metadata for Hive does not specify a maximum length for STRING columns.

Key Name	Default Value	Required
`Default StringColumnLength`	`255`	No

Delegate Kerberos Credentials

This option specifies whether your Kerberos credentials are forwarded to the server and used for authentication.

Note: This option is only applicable when Authentication Mechanism is set to Kerberos (`AuthMech=1`).

Key Name	Default Value	Required
`DelegateKrbCreds`	Clear (`0`)	No

Delegation UID

If a value is specified for this setting, the connector delegates all operations against Hive to the specified user, rather than to the authenticated user for the connection.

Note: This option is applicable only when connecting to a Hive Server 2 instance that supports this feature.

Key Name	Default Value	Required
DelegationUID	None	No

Driver Config Take Precedence

This option specifies whether connector-wide configuration settings take precedence over connection and DSN settings.

- Enabled (1): Connector-wide configurations take precedence over connection and DSN settings.
- Disabled (0): Connection and DSN settings take precedence instead.

Key Name	Default Value	Required
DriverConfigTakePrecedence	Clear (0)	No

Enable Auto Reconnect

This option specifies whether the connector attempts to automatically reconnect to the server when a communication link error occurs.

- Enabled (1): The connector attempts to reconnect.
- Disabled (0): The connector does not attempt to reconnect.

Key Name	Default Value	Required
AutoReconnect	Selected (1)	Yes

Enable SAML Auth Cookie Caching

This option specifies whether the connector uses the auth cookie when using SAML SSO authentication.

- Enabled (1):The connector caches the authorization cookie and does not open a new browser every time while establishing a new connection using SAML SSO authentication.
- Disabled (0): The connector does not cache the authorization cookie and opens a new browser every time while establishing a new connection using SAML SSO authentication.

Note:

- This property is available only in Windows.
- When this configuration is enabled, the Username and Password configurations are optional.
- The Username and Password can be optionally used as parts of the key for uniquely identifying the cache entries in the cache.
- The Password is used to encrypt the key so that the key is not stored in plain text in the cache file.

Key Name	Default Value	Required
`EnableSamlCookieCaching`	Clear (`0`)	No

Enable SSL

This option specifies whether the client uses an SSL encrypted connection to communicate with the Hive server.

- Enabled (`1`): The client communicates with the Hive server using SSL.
- Disabled (`0`): SSL is disabled.

SSL is configured independently of authentication. When authentication and SSL are both enabled, the connector performs the specified authentication method over an SSL connection.

Note:

- This option is applicable only when connecting to a Hiveserver that supports SSL.
- If you selected User Name as the authentication mechanism, SSL is not available.

Key Name	Default Value	Required
`SSL`	Clear (`0`)	No

Enable Temporary Table

This option specifies whether the driver supports the creation and use of temporary tables.

- Enabled (`1`): The driver supports the creation and use of temporary tables.
- Disabled (`0`): The driver does not support temporary tables.

Important:
When connecting to Hive 0.14 or later, the Temporary Tables feature is always enabled and you do not need to configure it in the driver.

Key Name	Default Value	Required
EnableTempTable	Clear (0)	No

Enable Translation For CTAS

This property specifies whether the connector performs a query translation for the CREATE TABLE AS SELECT (CTAS) syntax.

- Enabled (1): The connector performs a query translation for the CTAS syntax.
- Disabled (0): The connector does not perform a query translation for the CTAS syntax.

Key Name	Default Value	Required
EnableTranslationForCTAS	Enabled (1)	No

Fast SQLPrepare

This option specifies whether the connector defers query execution to SQLExecute.

- Enabled (1): The connector defers query execution to SQLExecute.
- Disabled (0): The connector does not defer query execution to SQLExecute.

Note: When using Native Query mode, the connector executes the HiveQL query to retrieve the result set metadata for SQLPrepare. As a result, SQLPrepare might be slow. If the result set metadata is not required after calling SQLPrepare, then enable Fast SQLPrepare.

Key Name	Default Value	Required
FastSQLPrepare	Clear (0)	No

Get Tables With Query

This option specifies whether the connector uses the SHOW TABLES query or the GetTables Thrift API call to retrieve table names from the database.

- Enabled (1): The connector uses the SHOW TABLES query to retrieve table names.
- Disabled (0): The connector uses the GetTables Thrift API call to retrieve table names.

Note: This option is applicable only when connecting to a Hive Server 2 instance.

Key Name	Default Value	Required
GetTablesWithQuery	0	No

HDFS User

The name of the HDFS user that the driver uses to create the necessary files for supporting the Temporary Tables feature.

This option is not applicable when connecting to Hive 0.14 or later.

Key Name	Default Value	Required
HDFSUser	hdfs	No

Hive Server Type

This option specifies the type of Hive server.

- Hive Server 1 (1): The connector connects to a Hive Server 1 instance.
- Hive Server 2 (2): The connector connects to a Hive Server 2 instance.

Note: If Service Discovery Mode is enabled, then connections to Hive Server 1 are not supported.

Key Name	Default Value	Required
HiveServerType	Hive Server 2 (2)	No

Host(s)

If Service Discovery Mode is disabled, the IP address or host name of the Hive server.

If Service Discovery Mode is enabled, specify a comma-separated list of ZooKeeper servers in the following format, where *[ZK_Host]* is the IP address or host name of the ZooKeeper server and *[ZK_Port]* is the number of the TCP port that the ZooKeeper server uses to listen for client connections:

```
[ZK_Host1]:[ZK_Port1],[ZK_Host2]:[ZK_Port2]
```

Key Name	Default Value	Required
Host	None	Yes

Host FQDN

The fully qualified domain name of the Hive Server 2 host.

When the value of Host FQDN is _HOST, the connector uses the Hive server host name as the fully qualified domain name for Kerberos authentication. If Service Discovery Mode is disabled, then the

connector uses the value specified in the Host connection attribute. If Service Discovery Mode is enabled, then the connector uses the Hive Server 2 host name returned by ZooKeeper.

Key Name	Default Value	Required
KrbHostFQDN	_HOST	No

HTTP Path

The partial URL corresponding to the Hive server.

The connector forms the HTTP address to connect to by appending the HTTP Path value to the host and port specified in the DSN or connection string. For example, to connect to the HTTP address `http://localhost:10002/gateway/sandbox/hive/version`, you would set HTTP Path to `/gateway/sandbox/hive/version`.

Note: By default, Hive servers use `cliservice` as the partial URL.

Key Name	Default Value	Required
HTTPPath	`/hive2` if using Windows Azure HDInsight Service (`6`). `/` if using non-Windows Azure HDInsight Service with Thrift Transport set to HTTP (`2`).	No

Ignore Tables Metadata From All Schemas

This option specifies whether SQLTables API calls that request metadata from all schemas should be ignored and return an empty result set.

- Enabled(`1`): The connector ignores SQLTables API calls that request metadata from all schemas and returns an empty result set.
- Disabled(`0`): The connector does not ignores SQLTables API calls that request metadata from all schemas and returns the expected result set.

Key Name	Default Value	Required
IgnoreTablesMetadataFromAllSchemas	Clear (`0`)	No

Invalid Session Auto Recover

This option specifies whether the connector automatically opens a new session when the existing session is no longer valid.

- Enabled (`1`): The connector automatically opens a new session when the existing session is no longer valid.
- Disabled (`0`): The connector does not automatically open new sessions.

Note: This option is applicable only when connecting to Hive Server 2.

Key Name	Default Value	Required
`InvalidSessionAutoRecover`	Selected (`1`)	No

Log Level

Use this property to enable or disable logging in the connector and to specify the amount of detail included in log files.

Important:

- Only enable logging long enough to capture an issue. Logging decreases performance and can consume a large quantity of disk space.
- When logging with connection strings and DSNs, this option only applies to per-connection logs.

Set the property to one of the following values:

- OFF (`0`): Disable all logging.
- FATAL (`1`): Logs severe error events that lead the connector to abort.
- ERROR (`2`): Logs error events that might allow the connector to continue running.
- WARNING (`3`): Logs events that might result in an error if action is not taken.
- INFO (`4`): Logs general information that describes the progress of the connector.
- DEBUG (`5`): Logs detailed information that is useful for debugging the connector.
- TRACE (`6`): Logs all connector activity.

When logging is enabled, the connector produces the following log files at the location you specify in the Log Path (`LogPath`) property:

- A `clouderaodbcdriverforapachehive.log` file that logs connector activity that is not specific to a connection.
- A `clouderaodbcdriverforapachehive_connection_[Number].log` file for each connection made to the database, where *[Number]* is a number that identifies each log file. This file logs connector activity that is specific to the connection.

Key Name	Default Value	Required
LogLevel	OFF (0)	No

Log Path

The full path to the folder where the connector saves log files when logging is enabled.

Important: When logging with connection strings and DSNs, this option only applies to per-connection logs.

Key Name	Default Value	Required
LogPath	None	Yes, if logging is enabled.

Max File Size

The maximum size of each log file in bytes. After the maximum file size is reached, the connector creates a new file and continues logging.

If this property is set using the Windows UI, the entered value is converted from megabytes (MB) to bytes before being set.

Important: When logging with connection strings and DSNs, this option only applies to per-connection logs.

Key Name	Default Value	Required
LogFileSize	20971520	No

Max Number Files

The maximum number of log files to keep. After the maximum number of log files is reached, each time an additional file is created, the connector deletes the oldest log file.

Important: When logging with connection strings and DSNs, this option only applies to per-connection logs.

Key Name	Default Value	Required
LogFileCount	50	No

Mechanism

The authentication mechanism to use.

Select one of the following settings, or set the key to the corresponding number:

- No Authentication (0)
- Kerberos (1)
- User Name (2)
- User Name And Password (3)
- SAML_2.0 (12)

Key Name	Default Value	Required
AuthMech	No Authentication (0) if you are connecting to Hive Server 1. User Name (2) if you are connecting to Hive Server 2.	No

Minimum TLS

The minimum version of TLS/SSL that the connector allows the data store to use for encrypting connections. For example, if TLS 1.1 is specified, TLS 1.0 cannot be used to encrypt connections.

- TLS 1.0 (1.0): The connection must use at least TLS 1.0.
- TLS 1.1 (1.1): The connection must use at least TLS 1.1.
- TLS 1.2 (1.2): The connection must use at least TLS 1.2.

Key Name	Default Value	Required
Min_TLS	TLS 1.2 (1.2)	No

Password

The password corresponding to the user name that you provided in the User Name field (the UID key).

Key Name	Default Value	Required
PWD	None	Yes, if the authentication mechanism is User Name And Password (3).

Port

The number of the TCP port that the Hive server uses to listen for client connections.

Key Name	Default Value	Required
Port	10000	Yes, if Service Discovery Mode

Key Name	Default Value	Required
		is disabled.

Proxy Host

The host name or IP address of a proxy server that you want to connect through.

Key Name	Default Value	Required
ProxyHost	None	Yes, if connecting through a proxy server.

Proxy Password

The password that you use to access the proxy server.

Key Name	Default Value	Required
ProxyPWD	None	Yes, if connecting to a proxy server that requires authentication.

Proxy Port

The number of the port that the proxy server uses to listen for client connections.

Key Name	Default Value	Required
ProxyPort	None	Yes, if connecting through a proxy server.

Proxy Username

The user name that you use to access the proxy server.

Key Name	Default Value	Required
ProxyUID	None	Yes, if connecting to a proxy server that requires authentication.

QueryTimeoutOverride

The number of seconds that a query can run before it is timed out. This property overwrites the SQL_ATTR_QUERY_TIMEOUT attribute.

Note:
When the value passed is an empty string, the connector does not attempt to override the SQL_ATTR_QUERY_TIMEOUT attribute.

Key Name	Default Value	Required
QueryTimeoutOverride	Empty string	No

Realm

The realm of the Hive Server 2 host.

If your Kerberos configuration already defines the realm of the Hive Server 2 host as the default realm, then you do not need to configure this option.

Key Name	Default Value	Required
KrbRealm	Depends on your Kerberos configuration.	No

Rows Fetched Per Block

The maximum number of rows that a query returns at a time.

Valid values for this setting include any positive 32-bit integer. However, testing has shown that performance gains are marginal beyond the default value of 10000 rows.

Key Name	Default Value	Required
RowsFetchedPerBlock	10000	No

Save Password (Encrypted)

This option specifies whether the password is saved in the registry.

- Enabled: The password is saved in the registry.
- Disabled: The password is not saved in the registry.

Note: This option is available only on the Windows connector. It appears in the Cloudera ODBC Driver for Apache HiveDSN Setup dialog box and the SSL Options dialog box. The password is obscured (not saved in plain text). However, it is still possible for the encrypted password to be copied and used.

Key Name	Default Value	Required
N/A	Selected	No

Service Discovery Mode

This option specifies whether the connector uses the ZooKeeper service.

- Enabled (`1`): The connector discovers Hive Server 2 services via the ZooKeeper service.
- Disabled (`0`): The connector connects to Hive without using a discovery service.

Key Name	Default Value	Required
`ServiceDiscoveryMode`	No Service Discovery (`0`)	No

Service Name

The Kerberos service principal name of the Hive server.

Key Name	Default Value	Required
`KrbServiceName`	None	No

Show System Table

This option specifies whether the connector returns the hive_system table for catalog function calls such as `SQLTables` and `SQLColumns`.

- Enabled (`1`): The connector returns the hive_system table for catalog function calls such as `SQLTables` and `SQLColumns`.
- Disabled (`0`): The connector does not return the hive_system table for catalog function calls.

Key Name	Default Value	Required
`ShowSystemTable`	Clear (`0`)	No

Socket Timeout

The number of seconds that an operation can remain idle before it is closed.

Note: This option is applicable only when asynchronous query execution is being used against Hive Server 2 instances.

Key Name	Default Value	Required
`SocketTimeout`	`60`	No

Ignore SQL_DRIVER_NOPROMPT

This option specifies whether the connector displays a web browser when the application makes a SQLDriverConnect API call with a SQL_DRIVER_NOPROMPT flag to the connector.

- Enabled (`true`): The connector displays the web browser used to complete the browser based authentication flow even when SQL_DRIVER_NOPROMPT is passed.
- Disabled (`false`): The connector does not display a web browser.

Note:

- SSOIgnoreDriverNoPrompt configuration affects only the SAML SSO authentication.
- For other authentication mechanisms, SQL_DRIVER_NOPROMPT parameter works as expected.

Key Name	Default Value	Required
SSOIgnoreDriverNoPrompt	`true`	No

Temp Table TTL

Key Name	Default Value	Required
TempTableTTL	10	No

Description

The number of minutes a temporary table is guaranteed to exist in Hive after it is created.

This option is not applicable when connecting to Hive 0.14 or later.

Thrift Transport

The transport protocol to use in the Thrift layer.

Select one of the following settings, or set the key to the number corresponding to the desired setting:

- Binary (`0`)
- SASL (`1`)
- HTTP (`2`)

Note: For information about how to determine which Thrift transport protocols your Hive server supports, see Authentication Mechanisms.

Key Name	Default Value	Required
`ThriftTransport`	Binary (`0`) if you are connecting to Hive Server 1.	No

Key Name	Default Value	Required
	SASL (`1`) if you are connecting to Hive Server 2.	

Trusted Certificates

The full path of the `.pem` file containing trusted CA certificates, for verifying the server when using SSL.

If this option is not set, then the connector defaults to using the trusted CA certificates `.pem` file installed by the connector. To use the trusted CA certificates in the `.pem` file, set the `UseSystemTrustStore` property to `0` or clear the Use System Trust Store check box in the SSL Options dialog.

Important:
If you are connecting from a Windows machine and the Use System Trust Store option is enabled, the connector uses the certificates from the Windows trust store instead of your specified `.pem` file.

For more information, see Use System Trust Store.

Key Name	Default Value	Required
`TrustedCerts`	The `cacerts.pem` file in the `\lib` subfolder within the connector's installation directory. The exact file path varies depending on the version of the connector that is installed. For example, the path for the Windows connector is different from the path for the macOS connector.	No

Two-Way SSL

This option specifies whether two-way SSL is enabled.

- Enabled (`1`): The client and the Hive server verify each other using SSL. See also the connector configuration options Client Certificate File, Client Private Key File, and Client Private Key Password.
- Disabled (`0`): The server does not verify the client. Depending on whether one-way SSL is enabled, the client might verify the server. For more information, see Enable SSL.

Note: This option is applicable only when connecting to a Hive server that supports SSL. You must enable SSL before Two Way SSL can be configured. For more information, see Enable SSL.

Key Name	Default Value	Required
TwoWaySSL	Clear (0)	No

Unicode SQL Character Types

This option specifies the SQL types to be returned for string data types.

- Enabled (1): The connector returns SQL_WVARCHAR for STRING and VARCHAR columns, and returns SQL_WCHAR for CHAR columns.
- Disabled (0): The connector returns SQL_VARCHAR for STRING and VARCHAR columns, and returns SQL_CHAR for CHAR columns.

Key Name	Default Value	Required
UseUnicodeSqlCharacterTypes	Clear (0)	No

Use Async Exec

This option specifies whether to execute queries synchronously or asynchronously.

- Enabled (1): The connector uses an asynchronous version of the API call against Hive for executing a query.
- Disabled (0): The connector executes queries synchronously.

Note: Setting ForceSynchronousExec property to 1 replaces EnableAsyncExec.

Key Name	Default Value	Required
EnableAsyncExec	Clear (0)	No

Use Native Query

This option specifies whether the connector uses native HiveQL queries, or converts the queries emitted by an application into an equivalent form in HiveQL. If the application is Hive-aware and already emits HiveQL, then enable this option to avoid the extra overhead of query transformation.

- Enabled (1): The connector does not transform the queries emitted by an application, and executes HiveQL queries directly.
- Disabled (0): The connector transforms the queries emitted by an application and converts them into an equivalent form in HiveQL.

Important:
When this option is enabled, the connector cannot execute parameterized queries.

Key Name	Default Value	Required
UseNativeQuery	Clear (0)	No

Use Only SSPI

This option specifies how the connector handles Kerberos authentication: either with the SSPI plugin or with MIT Kerberos.

- Enabled (1): The connector handles Kerberos authentication by using the SSPI plugin instead of MIT Kerberos by default.
- Disabled (0): The connector uses MIT Kerberos to handle Kerberos authentication, and only uses the SSPI plugin if the GSSAPI library is not available.

Important: This option is only available in Windows.

Key Name	Default Value	Required
UseOnlySSPI	Clear (0)	No

Use Proxy Server

This option specifies whether the connector uses a proxy server to connect to the data store.

- Enabled (1): The connector connects to a proxy server based on the information provided in the Proxy Host, Proxy Port, Proxy Username, and Proxy Password fields or the ProxyHost, ProxyPort, ProxyUID, and ProxyPWD keys.
- Disabled (0): The connector connects directly to the Hive server.

Key Name	Default Value	Required
UseProxy	Clear (0)	No

Use System Trust Store

This option specifies whether to use a CA certificate from the system trust store, or from a specified .pem file.

- Enabled (1): The connector verifies the connection using a certificate in the system trust store.
- Disabled (0): The connector verifies the connection using a specified .pem file. For information about specifying a .pem file, see Trusted Certificates.

Important:
If you are connecting from a Windows machine and you want to specify a .pem file containing trusted CA certificates, this option must be disabled. For more information, see Trusted Certificates.

Note: This option is only available in Windows.

Key Name	Default Value	Required
UseSystemTrustStore	Clear (0)	No

User Name

The user name that you use to access Hive Server 2.

Key Name	Default Value	Required
UID	For User Name (2) authentication only, the default value is anonymous	Yes, if the authentication mechanism is User Name And Password (3). No, if the authentication mechanism is User Name (2).

Web HDFS Host

The host name or IP address of the machine hosting both the namenode of your Hadoop cluster and the WebHDFS service.

This option is not applicable when connecting to Hive 0.14 or later.

Key Name	Default Value	Required
WebHDFSHost	The Hive server host.	No

Web HDFS Port

The WebHDFS port for the namenode.

This option is not applicable when connecting to Hive 0.14 or later.

Key Name	Default Value	Required
WebHDFSPort	50070	No

ZK Force Active Passive HA

If the ZkNamespace property does not contain "ActivePassiveHA", this option forces the ZooKeeper discovery mode to Active Passive HS2 Interactive HA.

- Enabled (`1`): The connector enables the Active Passive HS2 Interactive HA mode.
- Disabled (`0`): The connector does not enable the Active Passive HS2 Interactive HA mode.

Note:
This property is ignored if the `ZkNamespace` property contains "ActivePassiveHA".

Key Name	Default Value	Required
`ZKForceActivePassiveHA`	0	No

ZooKeeper Namespace

The namespace on ZooKeeper under which Hive Server 2 znodes are added.

Note:
If the namespace contains "ActivePassiveHA", the connector will enable the Active Passive HS2 Interactive HA connection mode.

Key Name	Default Value	Required
`ZKNamespace`	None	Yes, if Service Discovery Mode is enabled.

Configuration Options Having Only Key Names

The following configuration options do not appear in the Windows user interface for the Cloudera ODBC Driver for Apache Hive. They are accessible only when you use a connection string or configure a connection from a Linux/macOS/AIX machine:

- ConfigsFromFileDSN
- DelegationUserIDCase
- Driver
- HTTPAuthCookies
- http.header.
- SSP_
- SSOWebServerTimeout

ConfigsFromFileDSN

A comma-separated list of configuration names that the connector reads from the DSN file.

The connector only attempts to read the configuration values from a file DSN if the following conditions are met:

- The FILEDSN configuration is passed in via the connection string.
- The configuration key-value pairs must be inside the ODBC section in the DSN file.
- The ConfigsFromFileDSN configuration is either passed in via the connection string or via the file DSN, and the value of the ConfigsFromFileDSN configuration must contain the names, comma-separated, of the configurations to read from the DSN file.
- The value of the FILEDSN configuration is a valid directory path pointing to the location of the file DSN.

Important:
In some cases, the configuration of FILEDSN is removed from the connection string. In this case, add a FILEDSNPATH configuration to the connection string with the same value that is passed in for the FILEDSN configuration.

Key Name	Default Value	Required
`ConfigsFromFileDSN`	None	No

DelegationUserIDCase

This option specifies whether the connector changes the Delegation UID (or `DelegationUID`) value to all upper-case or all lower-case. The following values are supported:

- `Upper`: Change the delegated user name to all upper-case.
- `Lower`: Change the delegated user name to all lower-case.
- `Unchanged`: Do not modify the delegated user name.

For more information about delegating a user name, see Delegation UID.

Key Name	Default Value	Required
`DelegationUserIDCase`	`Unchanged`	No

Driver

In Windows, the name of the installed connector for (Cloudera ODBC Driver for Apache Hive).

On other platforms, the name of the installed connector as specified in `odbcinst.ini`, or the absolute path of the connector shared object file.

Key Name	Default Value	Required
`Driver`	`Cloudera Hive ODBC Driver`	Yes

Key Name	Default Value	Required
	when installed in Windows, or the absolute path of the connector shared object file when installed on a non-Windows machine.	

HTTPAuthCookies

A comma-separated list of authentication cookies that are supported by the connector.

If cookie-based authentication is enabled in your server, the connector authenticates the connection once based on the provided authentication credentials. It then uses the cookie generated by the server for each subsequent request in the same connection.

Key Name	Default Value	Required
`HTTPAuthCookies`	`hive.server2.auth, JSessionID, KNOXSESSIONID,KNOX_ BACKEND-HIVE`	No

http.header.

Set a custom HTTP header by using the following syntax, where *[HeaderKey]* is the name of the header to set and *[HeaderValue]* is the value to assign to the header:

```
http.header.[HeaderKey]=[HeaderValue]
```

For example:

```
http.header.AUTHENTICATED_USER=john
```

After the connector applies the header, the http.header. prefix is removed from the DSN entry, leaving an entry of *[HeaderKey]=[HeaderValue]*

The example above would create the following custom HTTP header:

```
AUTHENTICATED_USER: john
```

Note:

- The `http.header.` prefix is case-sensitive.
- This option is applicable only when you are using HTTP as the Thrift transport protocol. For more information, see Thrift Transport.

Key Name	Default Value	Required
http.header.	None	No

SSP_

Set a server-side property by using the following syntax, where *[SSPKey]* is the name of the server-side property and *[SSPValue]* is the value for that property:

```
SSP_[SSPKey]=[SSPValue]
```

For example:

```
SSP_mapred.queue.names=myQueue
```

After the connector applies the server-side property, the SSP_ prefix is removed from the DSN entry, leaving an entry of *[SSPKey]=[SSPValue]*.

Note:

- The SSP_ prefix must be upper case.
- When setting a server-side property in a connection string, it is recommended that you enclose the value in braces ({ }) to make sure that special characters can be properly escaped.

Key Name	Default Value	Required
SSP_	None	No

SSOWebServerTimeout

The length of time, in seconds, for which the connector waits for a browser response before timing out. If set to 0, the connector waits for an indefinite amount of time.

Note:
If SQL_ATTR_LOGIN_TIMEOUT is set, SQL_ATTR_LOGIN_TIMEOUT takes precedence. The connector honors SQL_ATTR_LOGIN_TIMEOUT when using the SAML authentication workflow.

Key Name	Default Value	Required
SSOWebServerTimeout	120	No

ODBC API Conformance Level

The following table lists the ODBC interfaces that the Cloudera ODBC Driver for Apache Hive implements and the ODBC compliance level of each interface.

ODBC compliance levels are Core, Level 1, and Level 2. These compliance levels are defined in the ODBC Specification published with the Interface SDK from Microsoft.

Interfaces include both the Unicode and non-Unicode versions. For more information, see "Unicode Function Arguments" in the *ODBC Programmer's Reference*: http://msdn.microsoft.com/en-us/library/ms716246%28VS.85%29.aspx.

Conformance Level	INTERFACES	Conformance Level	INTERFACES
Core	SQLAllocHandle	Core	SQLGetStmtAttr
Core	SQLBindCol	Core	SQLGetTypeInfo
Core	SQLBindParameter	Core	SQLNativeSql
Core	SQLCancel	Core	SQLNumParams
Core	SQLCloseCursor	Core	SQLNumResultCols
Core	SQLColAttribute	Core	SQLParamData
Core	SQLColumns	Core	SQLPrepare
Core	SQLConnect	Core	SQLPutData
Core	SQLCopyDesc	Core	SQLRowCount
Core	SQLDescribeCol	Core	SQLSetConnectAttr
Core	SQLDisconnect	Core	SQLSetCursorName
Core	SQLDriverconnect	Core	SQLSetDescField
Core	SQLEndTran	Core	SQLSetDescRec
Core	SQLExecDirect	Core	SQLSetEnvAttr
Core	SQLExecute	Core	SQLSetStmtAttr
Core	SQLFetch	Core	SQLSpecialColumns
Core	SQLFetchScroll	Core	SQLStatistics
Core	SQLFreeHandle	Core	SQLTables
Core	SQLFreeStmt	Core	SQLBrowseConnect
Core	SQLGetConnectAttr	Core	SQLPrimaryKeys
Core	SQLGetCursorName	Core	SQLGetInfo

Conformance Level	INTERFACES		Conformance Level	INTERFACES
Core	SQLGetData		Level 1	SQLProcedureColumns
Core	SQLGetDescField		Level 1	SQLProcedures
Core	SQLGetDescRec		Level 2	SQLColumnPrivileges
Core	SQLGetDiagField		Level 2	SQLDescribeParam
Core	SQLGetDiagRec		Level 2	SQLForeignKeys
Core	SQLGetEnvAttr		Level 2	SQLTablePrivileges
Core	SQLGetFunctions			

Contact Us

If you are having difficulties using the connector, our Community Forum may have your solution. In addition to providing user to user support, our forums are a great place to share your questions, comments, and feature requests with us.

If you are a Subscription customer you may also use the Cloudera Support Portal to search the Knowledge Base or file a Case.

Important:
To help us assist you, prior to contacting Cloudera Support please prepare a detailed summary of the client and server environment including operating system version, patch level, and configuration.