insightsoftware

Simba Apache Impala ODBC Data Connector

Installation and Configuration Guide

Version 2.9

November 2025

Contents

Contents **2**

Copyright **5**

About This Guide **6**

Purpose 6

Audience 6

Knowledge Prerequisites 6

Document Conventions 6

About the Simba Apache Impala ODBC Connector **7**

Platform and Data source version support **8**

Windows Connector **9**

Windows System Requirements 9

Installing the Connector in Windows 9

Creating a Data Source Name in Windows 10

Configuring Authentication in Windows 12

Configuring a Proxy Connection in Windows 20

Configuring HTTP Options in Windows 21

Configuring SSL Verification in Windows 22

Configuring Advanced Options in Windows 23

Configuring Server-Side Properties in Windows 24

Configuring Logging Options in Windows 25

Setting Connector-Wide Configuration Options in Windows 28

Configuring Kerberos Authentication for Windows 29

Verifying the Connector Version Number in Windows 32

macOS Connector 34

macOS System Requirements 34

Installing the Connector in macOS 34

Verifying the Connector Version Number in macOS 35

Linux Connector 36

Linux System Requirements 36

Installing the Connector Using the RPM File 36

Installing the Connector on Debian 37

Verifying the Connector Version Number in Linux 38

AIX Connector 40

AIX System Requirements 40

Installing the Connector on AIX 40

Verifying the Connector Version Number on AIX 41

Solaris Connector 42

Solaris System Requirements 42

Installing the Connector on Solaris 42

Verifying the Connector Version Number on Solaris 43

Configuring the ODBC Driver Manager in Non-Windows Machines 44

Specifying ODBC Driver Managers in Non-Windows Machines 44

Specifying the Locations of the Connector Configuration Files 45

Configuring ODBC Connections in Non-Windows Machine 47

Creating a Data Source Name on a Non-Windows Machine 47

Configuring a DSN-less Connection in a Non-Windows Machine 49
Configuring Authentication on a Non-Windows Machine 52
Configuring SSL Verification in a Non-Windows Machine 58
Configuring Server-Side Properties on a Non-Windows Machine 59
Configuring Logging Options in a Non-Windows Machine 59
Setting Connector-Wide Configuration Options on a Non-Windows Machine 61
Testing the Connection in Non-Windows Machine 61
Authentication Options 64
Using a Connection String 66
DSN Connection String Example 66
DSN-less Connection String Examples 66
Features 71
Data Types 71
Catalog and Schema Support 72
SQL Translation 72
Server-Side Properties 73
Active Directory 73
Write-back 73
Security and Authentication 73
Connector Configuration Options 75
Configuration Options Appearing in the User Interface 75
Configuration Options Having Only Key Names 99
Third-Party Trademarks 107

Copyright

Contact Us

www.insightsoftware.com

About This Guide

Purpose

The *Simba Apache Impala ODBC Data Connector Installation and Configuration Guide* explains how to install and configure the Simba Apache Impala ODBC Data Connector. The guide also provides details related to features of the connector.

Audience

The guide is intended for end users of the Simba Apache Impala ODBC Connector, as well as administrators and developers integrating the connector.

Knowledge Prerequisites

To use the Simba Apache Impala ODBC Connector, the following knowledge is helpful:

- Familiarity with the platform on which you are using the Simba Apache Impala ODBC Connector
- Ability to use the data source to which the Simba Apache Impala ODBC Connector is connecting
- An understanding of the role of ODBC technologies and driver managers in connecting to a data source
- Experience creating and configuring ODBC connections
- Exposure to SQL

Document Conventions

Italics is used when referring to book and document titles.

Bold is used in procedures for graphical user interface elements that a user clicks and text that a user types.

`Monospace font` indicates commands, source code, or contents of text files.

Note: A text box with a pencil icon indicates a short note appended to a paragraph.

Important: A text box with an exclamation mark indicates an important comment related to the preceding paragraph.

About the Simba Apache Impala ODBC Connector

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

The Simba Apache Impala ODBC Connector 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.

The Simba Apache Impala ODBC Connector is available for Microsoft® Windows®, Linux, Solaris, AIX, and macOS platforms.

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

Note: This is the most up-to-date version of the *Installation and Configuration Guide*, for use with version 2.9 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 Impala 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 Simba Apache Impala ODBC Connector supports Windows, macOS, and Linux operating systems. For details on the specific versions of operating systems supported as well as the versions of the data source supported, 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

Before the connector can be used, the Visual C++ Redistributable for Visual Studio 2022 with the same bitness as the connector must also be installed. If you obtained the connector from the Simba website, then your installation of the connector automatically includes this dependency. Otherwise, you must install the redistributable manually. You can download the installation packages for the redistributable at https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170.

Installing the Connector in Windows

If you did not obtain this connector from the Simba website, you might need to follow a different installation procedure. For more information, see the *Simba OEM ODBC Connector Installation Guide*.

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:

- `Simba Impala <Version Number> 32-bit.msi` for 32-bit applications
- `Simba Impala <Version Number> 64-bit.msi` for 64-bit applications

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

To install the Simba Apache Impala ODBC Connector in Windows:

1. Depending on the bitness of your client application, double-click to run **Simba Impala <Version Number> 32-bit.msi** or **Simba Impala <Version Number> 64-bit.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**.

7. If you received a license file through email, then copy the license file into the `\lib` subfolder of the installation folder you selected above. You must have Administrator privileges when changing the contents of this folder.

Creating a Data Source Name in Windows

Typically, after installing the Simba Apache Impala ODBC Connector, 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 Impala.

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 Setting Connector-Wide Configuration Options 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 Impala.

2. In the ODBC Data Source Administrator, click the **Drivers** tab, and then scroll down as needed to confirm that the Simba Apache Impala ODBC Connectorappears in the alphabetical list of ODBC Drivers 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 **Simba Apache Impala ODBC Connector**and then click **Finish**. The Simba Apache Impala ODBC ConnectorDSN 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 **Host** field, type the IP address or host name of the Impala server.
9. In the **Port** field, type the number of the TCP port that the Impala server uses to listen for client connections.

> **Note:** The default port number used by Impala is 21050.

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 Impala command prompt.

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

> **Note:**
> The default configuration of Impala requires the Simba Apache Impala ODBC Connector to be configured to use the No Authentication mechanism.

12. Optionally, if the operations against Impala 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.
13. In the **Transport Mode** drop-down list, select the Thrift transport protocol to use in the Thrift layer.

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

14. To configure a connection through a proxy server, click **Proxy Options**. For more information, see Configuring a Proxy Connection in Windows.
15. If the Transport Mode 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.
16. To configure client-server verification over SSL, click **SSL Options**. For more information, see Configuring SSL Verification in Windows.
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 logging behavior for the connector, click **Logging Options**. For more information, see Configuring Logging Options in Windows.
20. 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 Simba Impala ODBC Connector DSN Setup dialog box are correct. Contact your Impala server administrator as needed.

21. To save your settings and close the Simba Impala ODBC Connector DSN Setup dialog box, click **OK**.
22. To close the ODBC Data Source Administrator, click **OK**.

Configuring Authentication in Windows

Some Impala servers are configured to require authentication for access. To connect to an Impala server, you must configure the Simba Apache Impala ODBC Connector 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 Impala server requires, see Authentication Options.

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.

The following authentication methods are available:

- Using No Authentication
- Using Kerberos
- Using Advanced Kerberos
- Using SAML 2.0
- Using SASL User Name
- Using JSON Web Token (JWT)
- Using User Name And Password
- Using OAuth 2.0

If cookie-based authentication is enabled in your Impala 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

For this authentication mechanism, you do not need to configure any additional settings.

Note:
The default configuration of Impala requires the Simba Apache Impala ODBC Connector to be configured to use the No Authentication mechanism.

To configure a connection without 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. From the **Mechanism** drop-down list, select **No Authentication**.
3. If the Impala 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

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. To access authentication options, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
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 Impala server host is not the default, then, in the **Realm** field, type the Kerberos realm of the Impala server.
4. In the **Host FQDN** field, type the fully qualified domain name of the Impala server host.

   Note: To use the Impala 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 Impala 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. If the Impala 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.
9. 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.

10. To save your settings and close the dialog box, click **OK**.

Using Advanced Kerberos

The Advanced Kerberos authentication mechanism allows concurrent connections within the same process to use different Kerberos user principals.

This authentication mechanism is supported only when the connector is configured to handle Kerberos authentication using MIT Kerberos:

- MIT Kerberos must be installed on your machine.
- The Use Only SSPI option must be disabled. For more information, see Use Only SSPI.

When you use Advanced Kerberos authentication, you do not need to run the `kinit` command to obtain a Kerberos ticket. Instead, you use a JSON file to map your Impala user name to a Kerberos user principal name and a keytab that contains the corresponding keys. The connector obtains Kerberos tickets based on the specified mapping. As a fallback, you can specify a keytab that the connector uses by default if the mapping file is not available or if no matching keytab can be found in the mapping file.

Note:

- For information about the schema of the mapping file and how the connector handles invalid mappings, see UPN Keytab Mapping File.
- For information about how the connector searches for a keytab file if the keytab mapping and default keytab file are invalid, see Default Keytab File.

To configure Advanced Kerberos 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 **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 Impala server host is not the default, then, in the **Realm** field, type the Kerberos realm of the Impala server.

4. In the **Host FQDN** field, type the fully qualified domain name of the Impala server host.

> **Note:**
>
> To use the Impala 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 Impala 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. Select the **Use Keytab** check box.

> **Note:**
>
> If the check box is not available, make sure that MIT Kerberos is installed on your machine.

8. In the **User Name** field, type an appropriate user name for accessing the Impala server.
9. Click **Keytab Options** and then do the following in the Keytab Options dialog box:
   a. In the **UPN Keytab Mapping File** field, specify the full path to a JSON file that maps your Impala user name to a Kerberos user principal name and a keytab file.
   b. In the **Default Keytab File** field, specify the full path to a keytab file that the connector can use if the mapping file is not available or if no matching keytab can be found in the mapping file.
   c. To save your settings and close the dialog box, click **OK**.
10. If the Impala 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.
11. 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.

12. 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, Transport Mode 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 Impala server host.
4. In the **Port** field, type the number of the TCP port that the Impala 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 **Transport Mode** 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. Optionally, click **SAML Options**, and select the **Enable Routing To Same Coordinator** check box. It is active only when the **Enable Auth Cookie Caching** check box is already selected. The connector uses the cached SAML auth cookie to route to the same coordinator in subsequent connections. For more information, see the configuration option Enable Routing To Same Coordinator For Cached SAML Auth Cookie.
10. Click **HTTP Options** and in the **HTTP Path** field, type the partial URL corresponding to the Impala server. For more information, see Configuring HTTP Options in Windows.
11. Click **SSL Options** and select the **Enable SSL** check box. For more information, see Configuring SSL Verification in Windows
12. To save your settings and close the dialog box, click **OK**.

Using JSON Web Token (JWT)

This authentication mechanism enables you to authenticate via JSON Web Token against supported servers.

To configure JWT 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. From the **Mechanism** drop-down list, select **JWT**.
3. In the **JWT String** field, type a valid jwt string for accessing the Impala server.
4. To encrypt your credentials, click **Encryption Options** and then select one of the following:
   - If the credentials are used only by the current Windows user, select **Current User Only**.
   - Or, if the credentials are used by all users on the current Windows machine, select **All Users Of This Machine**.

   To confirm your choice and close the Encryption Options dialog box, click **OK**.
5. To save your settings and close the dialog box, click **OK**.

Important:
In order to use `JWT` for authentication, `TransportMode` must be set to `HTTP` and `JWTString` must be specified in the connection string.

Using SASL User Name

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

To configure SASL User Name 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. From the **Mechanism** drop-down list, select **SASL User Name**.
3. In the **User Name** field, type an appropriate user name for accessing the Impala server.
4. 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.
5. 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.

Note: This authentication mechanism should not be used with an Impala configuration that does not have LDAP enabled.

To configure User Name And Password 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. 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 Impala server.
4. In the **Password** field, type the password corresponding to the user name you typed above.
5. To encrypt your credentials, click **Password Options** and then select one of the following:
   - If the credentials are used only by the current Windows user, select **Current User Only**.
   - Or, if the credentials are used by all users on the current Windows machine, select **All Users Of This Machine**.

   To confirm your choice and close the Password Options dialog box, click **OK**.
6. If the Impala 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.
7. 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.
8. Optionally, to use SASL to handle authentication, select the **Use Simple Authentication and Security Layer (SASL)** check box.

   Note: If the Transport Mode property is specified, it takes precedence over this property.
9. To save your settings and close the dialog box, click **OK**.

Using OAuth 2.0

Three types of authentication work flow are available when using OAuth 2.0, token pass-through, client credentials, or browser based authentication.

When you use OAuth 2.0 authentication, HTTP is the only transport mode protocol available.

Token Pass-through

This authentication mechanism requires a valid OAuth 2.0 access token. Be aware that access tokens typically expire after a certain amount of time, after which you must either refresh the token or obtain a new one from the server. To obtain a new access token, see Using OAuth 2.0.

To configure OAuth 2.0 token pass-though authentication:

1. 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**
2. From the **Mechanism** drop-down list, select **OAuth 2.0**.

3. Click **OAuth Options**, and then do the following:
   a. From the **Authentication Flow** drop-down list, select **Token Passthrough**.
   b. In the **Access Token** field, type your access token.
   c. Optionally, select Token or Client Secret Encryption Options and choose the encryption password for **Current User Only** or **All Users of this Machine**. Click **OK**.
   d. To save your settings and close the OAuth Options dialog box, click **OK**.
4. To save your settings and close the DSN Setup dialog box or the Driver Configuration tool, click **OK**.

Providing a New Access Token

Once an access token expires, you can provide a new access token for the connector.

Note: When an access token expires, the connector returns a "SQLState 08006" error.

To obtain a new access token:

1. In the connection string, set the `Auth_AccessToken` property with a new access token.
2. Call the SQLSetConnectAttr function with SQL_ATTR_CREDENTIALS (122) as the attribute and the new connection string as the value. The connector will update the current connection string with the new access token.

   Note: Calling the SQLGetConnectAttr function with SQL_ATTR_CREDENTIALS (122) returns the entire connection string used during connection.
3. Call the SQLSetConnectAttr function with SQL_ATTR_REFRESH_CONNECTION (123) as the attribute and SQL_REFRESH_NOW (-1) as the value. This signals the connector to update the access token value.
4. Retry the previous ODBC API call. After obtaining the new access token, the open connection, statements, and cursors associated with it remain valid for use.

Client Credentials

This authentication mechanism requires SSL to be enabled.

You can use client secret as the client credentials.

To configure OAuth 2.0 client credentials authentication using the client secret:

1. 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**.
2. From the **Mechanism** drop-down list, select **OAuth 2.0**.
3. Click **OAuth Options**, and then do the following:
   a. From the **Authentication Flow** drop-down list, select **Client Credentials**.
   b. In the **Client ID** field, type your client ID.
   c. In the **Client Secret** field, type your client secret.

d. In the **Audience** field, type your OAuth audience.
e. In the **Token Endpoint** field, type your OAuth token endpoint.
f. Optionally, select **Token** or **Client Secret Encryption Options** and choose the encryption password for **Current User Only** or **All Users of this Machine**. Click **OK**.
g. Optionally, in the **Scope** field, type your OAuth scope.
h. To save your settings and close the OAuth Options dialog box, click **OK**.

4. To save your settings and close the DSN Setup dialog box or the Driver Configuration tool, click **OK**.

Browser Based

This authentication mechanism requires SSL to be enabled.

To configure OAuth 2.0 browser based authentication:

1. 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**.
2. From the **Mechanism** drop-down list, select **OAuth 2.0**.
3. Click **OAuth Options**, and then do the following:
   a. From the **Authentication Flow** drop-down list, select **Browser Based Authorization Code**.
   b. In the **Client ID** field, type your client ID.
   c. In the **Client Secret** field, type your client secret.
   d. Optionally, select **Token** or **Client Secret Encryption Options** and choose the encryption password for **Current User Only** or **All Users of this Machine**. Click **OK**.
   e. In the **Audience** field, type your OAuth audience.
   f. In the **Token Endpoint** field, type your OAuth token endpoint.
   g. In the **Authorization Endpoint** field, type your OAuth authorization endpoint.
   h. Optionally, in the **Scope** field, type your OAuth scope.
   i. Optionally, 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.
   j. To save your settings and close the OAuth Options dialog box, click **OK**.
4. To save your settings and close the DSN Setup dialog box or the Driver Configuration tool, click **OK**.

> **Note:** When the browser based authentication flow completes, the access token and refresh token are saved in the token cache and the connector does not need to authenticate again. For more information, see Enable Token Cache.

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. Select the **Use Proxy** check box.
3. In the **Proxy Host** field, type the host name or IP address of the proxy server.
4. In the **Proxy Port** field, type the number of the TCP port that the proxy server uses to listen for client connections.
5. In the **Proxy Username** field, type your user name for accessing the proxy server.
6. In the **Proxy Password** field, type the password corresponding to the user name.
7. To encrypt your credentials, click **Password Options** and then select one of the following:
   - If the credentials are used only by the current Windows user, select **Current User Only**.
   - Or, if the credentials are used by all users on the current Windows machine, select **All Users Of This Machine**.

   To confirm your choice and close the Password Options dialog box, click **OK**.
8. To save your settings and close the HTTP 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 Impala server supports the HTTP transport protocol, see Authentication Options.

The following instructions describe how to configure HTTP options in a DSN. 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. Open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, and then make sure that the Transport Mode option is set to **HTTP**.
2. To access HTTP options, click **HTTP Options**.

   Note: The HTTP options are available only when the Transport Mode option is set to HTTP.
3. In the **HTTP Path** field, type the partial URL corresponding to the Impala 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 an Impala 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 can be configured to verify the identity of the server.

The following instructions describe how to configure SSL in a DSN. 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.

> **Important:**
> If Check Certificate Revocation is enabled, make sure that the connector has access to the CRL/OCSP server. When using a proxy between the connector and the CRL/OCSP server, make sure that the proxy is properly configured.
> If the proxy uses LDAP authentication, save the proxy credential to the Windows system. This is because the connector does not display a credential dialog when checking the revocation. Therefore, if the credential is not saved, the connector does not check revocation and returns an SSL error.

To configure SSL verification in Windows:

1. To access SSL options, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, 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 Impala 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 save your settings and close the SSL Options dialog box, click **OK**.

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. 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. To access advanced options, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, and then click **Advanced Options**.
2. To disable translation from ODBC SQL to Impala SQL, 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 Impala SQL. If the application is Impala-aware and already emits Impala SQL, then turning off the translation avoids the additional overhead of query transformation.

3. To enable the connector to successfully run queries that contain transaction statements, select the **Ignore Transactions** check box.

   Note: The transaction statements are not executed, because ODBC does not support them. Enabling this option allows the connector to run the query without returning error messages.

4. 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 **Use SQL Unicode Types** check box.

5. To have the connector automatically attempt to reconnect to the server if communications are lost, select **Enable Auto Reconnect**.
6. To have the connector restrict catalog queries to the current schema when no schema is specified, or when the schema is specified with the wildcard character %, select **Restrict Metadata with Current Schema**.
7. In the **Rows Fetched Per Block** field, type the number of rows to be fetched per block.
8. In the **Socket Timeout** field, type the number of seconds that the TCP socket waits for a response from the server before timing out the request and returning an error message.

   Note:
   Setting the Socket Timeout value to 0 disables the timeout feature.

9. In the **String Column Length** field, type the maximum data length for STRING columns.
10. In the **Async Exec Poll Interval** field, type the time in milliseconds between each poll for the query execution status.
11. To handle Kerberos authentication using the SSPI plugin instead of MIT Kerberos by default, select one or both of the check boxes under the **Use Only SSPI** option:

- To configure the current DSN to use the SSPI plugin by default, select **Enable For This DSN**.
- To configure all DSN-less connections to use the SSPI plugin by default, select **Enable For DSN-less Connections**.
- To configure all connections that use the Simba Apache Impala ODBC Connector to use the SSPI plugin by default, select both check boxes.

12. Optionally, if you want the connector to retry queries that fail, select the **Enable Query Retry** check box and then do the following:
    a. In the **Max Retries** field, type the maximum number of times that the connector retries each query.
    b. In the **Result Set Cache Size** field, type the maximum amount of memory that the result set cache can occupy. Values must be specified in: B (bytes), KB (kilobytes), MB (megabytes), or GB (gigabytes).
    c. In the **Retry Interval** field, type the amount of time that the connector waits between query retry attempts. Values must be specified in S (seconds) or MS (milliseconds).
13. To save your settings and close the Advanced Options dialog box, click **OK**.

Configuring Server-Side Properties in Windows

When connecting to a server that is running Impala 2.0 or later, you can use the connector to apply configuration properties to the server.

Important:
This feature is not supported for earlier versions of Impala, where the SET statement can only be executed from within the Impala shell.

The following instructions describe how to configure server-side properties in a DSN. 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. To configure server-side properties, 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**.
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 MEM_LIMIT query option to 1 GB, type **MEM_LIMIT** in the **Key** field and then type **1000000000** in the **Value** field.
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 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 Simba Simba Apache Impala ODBC Connector, 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 Simba Apache Impala ODBC Connector, 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**.
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.

Logging Level	Description
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 Simba Apache Impala ODBC Connector produces the following log files at the location you specify in the Log Path field:

- A `simbaimpalaodbcdriver.log` file that logs connector activity that is not specific to a connection.
- A `simbaimpalaodbcdriver_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.

If you enable the `UseLogPrefix` connection property, the connector prefixes the log file name with the user name associated with the connection and the process ID of the application through which the connection is made. For more information, see UseLogPrefix.

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.

Setting Connector-Wide Configuration Options in Windows

When you specify connection settings in a DSN or connection string, those settings apply only when you connect to Impala using that particular DSN or string. As an alternative, you can specify settings that apply to every connection that uses the Simba Apache Impala ODBC Connector by configuring them in the Windows Registry.

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 in Windows:

1. Choose one:

   - If you are using Windows 7 or earlier, click **Start**, then type **regedit** in the **Search** field, and then click **regedit.exe** in the search results.
   - Or, if you are using Windows 8 or later, 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:

   - If you are using the 32-bit connector on a 64-bit machine, then browse to the following registry key:

     HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Simba\Simba Impala ODBC Driver\Driver

   - Otherwise, browse to the following registry key:

     HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Impala ODBC Driver\Driver

3. For each connection property that you want to configure, do the following:
   a. Right-click the **Driver** subkey and then select **New > String Value**.
   b. Type the key name of the connection property, and then press **Enter**.

      For example, to specify the authentication mechanism to use, type `AuthMech`. To verify the supported key name for each connector configuration option, refer to the "Key Name" column in the description of the option in Connector Configuration Options.
   c. Right-click the value that you created in the previous steps and then click **Modify**.
   d. In the Edit String dialog box, in the **Value Data** field, type the value that you want to set the connection property to and then click **OK**.

      For example, to specify the Kerberos authentication mechanism, type `1`.

4. Close the Registry Editor.

Configuring Kerberos Authentication for Windows Active Directory

The Simba Apache Impala ODBC Connector 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, according to Apache's documentation, 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 Impala server.
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 Impala server.
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 Simba Apache Impala ODBC Connector 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 Impala.

2. Click the **Drivers** tab and then find the Simba Apache Impala ODBC Connector 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 2.3.6 or later

Installing the Connector in macOS

If you did not obtain this connector from the Simba website, you might need to follow a different installation procedure. For more information, see the *Simba OEM ODBC Connectors Installation Guide*.

To install the Simba Apache Impala ODBC Connector in macOS:

1. Double-click **Simba Impala <Version Number> .dmg** to mount the disk image.
2. Double-click **SimbaImpalaODBC.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 `/Library/simba/impalaodbc` directory.

6. To accept the installation location and begin the installation, click **Install**.
7. When the installation completes, click **Close**.
8. If you received a license file through email, then copy the license file into the `/lib` subfolder in the connector installation directory. You must have root privileges when changing the contents of this folder.

   For example, if you installed the connector to the default location, you would copy the license file into the `/Library/simba/impalaodbc/lib` folder.

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 Simba Apache Impala ODBC Connector 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 simba.impalaodbc
```

The command returns information about the Simba Apache Impala ODBC Connector 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:

- 50MB 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

If you did not obtain this connector from the Simba website, you might need to follow a different installation procedure. For more information, see the *Simba OEM ODBC Connectors Installation Guide*.

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:

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

On 64-bit ARM editions of Linux, applications must use 64-bit ARM connectors:

- `simbaimpala-[Version]-[Release].aarch64.rpm` for the 64-bit ARM 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.

To install the Simba Apache Impala ODBC Connector 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, Amazon 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 Simba Apache Impala ODBC Connector files are installed in the `/opt/simba/impalaodbc` 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.

4. If you received a license file through email, then copy the license file into the `/opt/simba/impalaodbc/lib/32` or `/opt/simba/impalaodbc/lib/64` folder, depending on the version of the connector that you installed.

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:

- `simbaimpala_[Version]-[Release]_i386.deb` for the 32-bit connector
- `simbaimpala_[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 Simba Apache Impala ODBC Connector 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 **simbaimpala_*[Version]*-*[Release]*_i386.deb** or **simbaimpala_*[Version]*-*[Release]*_amd64.deb**.
3. Follow the instructions in the installer to complete the installation process.

   The Simba Apache Impala ODBC Connector files are installed in the `/opt/simba/impalaodbc` 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.

4. If you received a license file via email, then copy the license file into the `/opt/simba/impalaodbc/lib/32` or `/opt/simba/impalaodbc/lib/64` folder, depending on the version of the connector that you installed. You must have root privileges when changing the contents of this folder.

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 Simba Apache Impala ODBC Connector 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. 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 SimbaImpalaODBC
  - rpm -qa | grep SimbaImpalaODBC

The command returns information about the Simba Apache Impala ODBC Connector 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/simba/impalaodbc/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:

- `SimbaImpalaODBC-32bit-[Version]-[Release].ppc.rpm` for the 32-bit connector
- `SimbaImpalaODBC-[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 Simba Apache Impala ODBC Connector 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 Simba Apache Impala ODBC Connector files are installed in the `/opt/simba/impalaodbc` directory.
3. If you received a license file via email, then copy the license file into the `/opt/simba/impalaodbc/lib/32` or `/opt/simba/impalaodbc/lib/64` folder, depending on the version of the connector that you installed. You must have root privileges when changing the contents of this folder.

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 Simba Apache Impala ODBC Connector 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 SimbaImpalaODBC
```

The command returns information about the Simba Apache Impala ODBC Connector that is installed on your machine, including the version number.

Solaris Connector

Solaris 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:

- 50 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 Solaris

The Simba Apache Impala ODBC Connector is available for Solaris as a tarball package named `Simba Apache Impala ODBC Connector_Solaris-_gcc_[Version].[Release]_Solaris.tar.gz`, where *[Version]* is the version number of the connector and *[Release]* is the release number for this version of the connector. The package contains both the 32- bit and 64-bit versions of the connector.

On sparc64 editions of Solaris, you can execute both sparc and sparc64 applications. However, sparc64 applications must use 64-bit connectors, and sparc applications must use 32-bit connectors. Make sure that you use the version of the connector that matches the bitness of the client application. You can install both versions of the connector on the same machine.

To install the Simba Apache Impala ODBC Connector on Solaris:

1. Log in as the root user, and then navigate to the folder containing the tarball package.
2. Run the following command to extract the package and install the connector:

   ```
   tar --directory=/opt -zxvf [TarballName]
   ```

   Where `[TarballName]` is the name of the tarball package containing the connector.

   The Simba Apache Impala ODBC Connector files are installed in the `/opt/simba/impalaodbc/` directory.
3. If you received a license file via email, then copy the license file into the `/opt/simba/impalaodbc/lib/32` or `/opt/simba/impalaodbc/lib/64` folder, depending on the version of the connector that you installed. You must have root privileges when changing the contents of this folder.

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 Solaris

If you need to verify the version of the Simba Apache Impala ODBC Connector that is installed on your Solaris machine, you can query the version number through the command-line interface.

To verify the connector version number on Solaris:

- At the command prompt, run the following command:

```
rpm -qa | grep SimbaImpalaODBC
```

The command returns information about the Simba Apache Impala ODBC Connector 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 Simba Apache Impala ODBC Connector, 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.

Solaris

If you are using a Solaris machine, then set the LD_LIBRARY_PATH environment variable to include the paths to the ODBC driver manager libraries and the third-party libraries that are installed with the connector. For example, if the driver manager libraries are installed in `/usr/local/lib` and the 32-bit connector is installed in `/opt/simba/impala`, 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:/opt/simba/impala/lib/32
```

For information about setting an environment variable permanently, refer to the Solaris 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 `simba.impalaodbc.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 SIMBAIMPALAINI to the full path and file name of the `simba.impalaodbc.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 SIMBAIMPALAINI to the full path and file name of the `simba.impalaodbc.ini` file.

For example, if your `odbc.ini` and `odbcinst.ini` files are located in `/usr/local/odbc` and your `simba.impalaodbc.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 SIMBAIMPALAINI=/etc/simba.impalaodbc.ini
```

For unixODBC:

```
export ODBCINI=/usr/local/odbc/odbc.ini
export ODBCSYSINI=/usr/local/odbc
export SIMBAIMPALAINI=/etc/simba.impalaodbc.ini
```

To locate the `simba.impalaodbc.ini` file, the connector uses the following search order:

1. If the SIMBAIMPALAINI 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 `simba.impalaodbc.ini`.
3. The connector searches the current working directory of the application for a file named `simba.impalaodbc.ini`.
4. The connector searches the home directory for a hidden file named `simba.impalaodbc.ini` (prefixed with a period).
5. The connector searches the `/etc` directory for a file named `simba.impalaodbc.ini`.

Configuring ODBC Connections in Non-Windows Machine

The following sections describe how to configure ODBC connections when using the Simba Apache Impala ODBC Connector 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 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 in a Non-Windows Machine
- Setting Connector-Wide Configuration Options on a Non-Windows Machine
- Testing the Connection in Non-Windows Machine

Creating a Data Source Name on a Non-Windows Machine

Typically, after installing the Simba Apache Impala ODBC Connector, 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 Impala.

You can specify connection settings in a DSN (in the `odbc.ini` file), in a connection string, or as connector-wide settings (in the `simba.impalaodbc.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=Simba Impala ODBC Driver

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

[ODBC Data Sources]

Sample DSN=Simba Impala ODBC Driver 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=/Library/simba/impalaodbc/lib/libimpalaodbc_sb64-universal.dylib

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

      Driver=/opt/simba/impalaodbc/lib/32/libimpalaodbc_sb32.so
   b. Set the `Host` property to the IP address or host name of the server.

      For example:

      Host=192.168.222.160
   c. Set the `Port` property to the number of the TCP port that the server uses to listen for client connections.

      For example:

      Port=21050
   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.
   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 Simba Apache Impala ODBC Connector, 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 an Impala server that does not require authentication:

[ODBC Data Sources]

Sample DSN=Simba Impala ODBC Driver

[Sample DSN]

Driver=/Library/simba/impalaodbc/lib/libimpalaodbc_sb64-universal.dylib

Host=192.168.222.160

Port=21050

As another example, the following is an `odbc.ini` configuration file for a 32-bit connector on a Linux/AIX/Solaris machine, containing a DSN that connects to an Impala server that does not require authentication:

[ODBC Data Sources]

Sample DSN=Simba Impala ODBC Driver 32-bit

[Sample DSN]

Driver=/opt/simba/impalaodbc/lib/32/libimpalaodbc_sb32.so

Host=192.168.222.160

Port=21050

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:

   ```
   Simba Impala ODBC Driver=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=/Library/simba/impalaodbc/lib/libimpalaodbc_sb64-
   universal.dylib.
   ```

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

   ```
   Driver=/opt/simba/impalaodbc/lib/32/libimpalaodbc_sb32.so
   ```

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

   ```
   Description=Simba Impala ODBC Driver
   ```

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]
   ```

   ```
   Simba Impala ODBC Driver=Installed
   ```

```
[Simba Impala ODBC Driver]
```

```
Description=Simba Impala ODBC Driver
```

```
Driver=/Library/simba/impalaodbc/lib/libimpalaodbc_sb64-universal.dylib
```

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

```
[ODBC Drivers]
```

```
Simba Impala ODBC Driver 32-bit=Installed
```

```
Simba Impala ODBC Driver 64-bit=Installed
```

```
[Simba Impala ODBC Driver 32-bit]
```

```
Description=[Simba Impala ODBC Driver 32-bit]
```

```
Driver=/opt/simba/impalaodbc/lib/32/libimpalaodbc_sb32.so
```

```
[Simba Impala ODBC Driver 64-bit]
```

```
Description=[Simba Impala ODBC Driver 64-bit]
```

```
Driver=/opt/simba/impalaodbc/lib/64/libimpalaodbc_sb64.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 Authentication on a Non-Windows Machine

Some Impala servers are configured to require authentication for access. To connect to an Impala server, you must configure the Simba Apache Impala ODBC Connector 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 Impala server requires, see Authentication Options.

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 `simba.impalaodbc.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 authentication methods are available:

- Using No Authentication
- Using Kerberos
- Using Advanced Kerberos
- Using SAML 2.0
- Using SASL User Name
- Using JSON Web Token (JWT)
- Using User Name And Password
- Using OAuth 2.0

If cookie-based authentication is enabled in your Impala 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

For this authentication mechanism, you do not need to configure any additional settings.

Note: The default configuration of Impala requires the Simba Apache Impala ODBC Connector to be configured to use the No Authentication mechanism.

To configure a connection without authentication:

1. Set the `AuthMech` connection attribute to `No Authentication`.
2. If the Impala 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/.

To configure Kerberos authentication:

1. Set the `AuthMech` connection attribute to `Kerberos`.
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 Impala 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 `KrbFQDN` attribute to the fully qualified domain name of the Impala server host.

   Note: To use the Impala server host name as the fully qualified domain name for Kerberos authentication, set `KrbFQDN` to `_HOST`.
5. Set the `KrbServiceName` attribute to the service name of the Impala server.
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 Advanced Kerberos

This authentication mechanism allows concurrent connections within the same process to use different Kerberos user principals.

When you use Advanced Kerberos authentication, you do not need to run the `kinit` command to obtain a Kerberos ticket. Instead, you use a JSON file to map your Impala user name to a Kerberos user principal name and a keytab that contains the corresponding keys. The connector obtains Kerberos tickets based on the specified mapping. As a fallback, you can specify a keytab that the connector uses by default if the mapping file is not available or if no matching keytab can be found in the mapping file.

Note:

- For information about the schema of the mapping file and how the connector handles invalid mappings, see UPN Keytab Mapping File.
- For information about how the connector searches for a keytab file if the keytab mapping and default keytab file are invalid, see Default Keytab File.

To configure Advanced Kerberos authentication:

1. Set the `AuthMech` connection attribute to `Kerberos`.
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 Impala 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 `KrbFQDN` attribute to the fully qualified domain name of the Impala server host.

Note:

To use the Impala server host name as the fully qualified domain name for Kerberos authentication, set `KrbFQDN` to `_HOST`.

5. Set the `KrbServiceName` attribute to the service name of the Impala server.
6. Set the `UseKeytab` attribute to `1`.
7. Set the `UID` attribute to an appropriate user name for accessing the Impala server.
8. Set the `UPNKeytabMappingFile` attribute to the full path to a JSON file that maps your Impala user name to a Kerberos user principal name and a keytab file.
9. Set the `DefaultKeytabFile` attribute to the full path to a keytab file that the connector can use if the mapping file is not available or if no matching keytab can be found in the mapping file.
10. If the Impala server is configured to use SSL, then configure SSL for the connection. For more information, see Configuring SSL Verification in a Non-Windows Machine.
11. 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 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 `TransportMode` attribute must be set to `HTTP` and the `SSL` attribute must be set to `1`.

To configure SAML 2.0 authentication:

1. Set the `AuthMech` connection attribute to `SAML_2.0`.
2. Set the `TransportMode` attribute to `HTTP`.
3. Set the `HttpPath` attribute to the partial URL corresponding to the Impala 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 SASL User Name

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

To configure SASL User Name authentication:

1. Set the `AuthMech` connection attribute to `SASL User Name`.
2. Set the `UID` attribute to an appropriate user name for accessing the Impala server.
3. 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 JSON Web Token (JWT)

This authentication mechanism enables you to authenticate via JSON Web Token against supported servers.

Important:
In order to use `JWT` for authentication, `TransportMode` must be set to `HTTP` and `JWTString` must be specified in the connection string.

To configure JWT authentication:

1. Set the `AuthMech` connection attribute to `JWT or 8`.
2. Set the `TransportMode` attribute to `HTTP`.
3. Set the `JWTString` attribute in the connection string.

You can now use the connector to authenticate through JWT and connect to your Impala server.

Using User Name And Password

This authentication mechanism requires a user name and a password.

Note: This authentication mechanism should not be used with an Impala configuration that does not have LDAP enabled.

To configure User Name And Password authentication:

1. Set the `AuthMech` connection attribute to `User Name and Password`.
2. Set the `UID` attribute to an appropriate user name for accessing the Impala server.
3. Set the `PWD` attribute to the password corresponding to the user name you provided above.
4. 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.

5. Optionally, to use SASL to handle authentication, set the `UseSASL` attribute to `1`.

   Note: If the Transport Mode property is specified, it takes precedence over this property.

Using OAuth 2.0

Three types of authentication work flow are available when using OAuth 2.0, token pass-through, client credentials, or browser based authentication.

When you use OAuth 2.0 authentication, HTTP is the only transport mode protocol available.

Token Pass-through

This authentication mechanism requires a valid OAuth 2.0 access token. Be aware that access tokens typically expire after a certain amount of time, after which you must either refresh the token or obtain a new one from the server. To obtain a new access token, see Obtaining a New Access Token.

To configure OAuth 2.0 token pass-though authentication:

1. Set the `AuthMech` property to `OAuth 2.0`.
2. Set the `Auth_Flow` property to `0`.
3. Set the `Auth_AccessToken` property to your access token.

Obtaining a New Access Token

Once an access token expires, you can obtain a new access token for the connector.

Note: When an access token expires, the connector returns a "SQLState 08006" error.

To obtain a new access token:

1. In the connection string, set the `Auth_AccessToken` property with a new access token.
2. Call the SQLSetConnectAttr function with SQL_ATTR_CREDENTIALS (122) as the attribute and the new connection string as the value. The connector will update the current connection string with the new access token.

   Note: Calling the SQLGetConnectAttr function with SQL_ATTR_CREDENTIALS (122) returns the entire connection string used during connection.

3. Call the SQLSetConnectAttr function with SQL_ATTR_REFRESH_CONNECTION (123) as the attribute and SQL_REFRESH_NOW (-1) as the value. This signals the connector to update the access token value.
4. Retry the previous ODBC API call. After obtaining the new access token, the open connection, statements, and cursors associated with it remain valid for use.

Client Credentials

This authentication mechanism requires SSL to be enabled.

To configure OAuth 2.0 client credentials authentication:

1. Set the `AuthMech` property to `OAuth 2.0`.
2. Set the `Auth_Flow` property to `1`.
3. Set the `Auth_Client_ID` to your client ID.
4. Set the `Auth_Client_Secret` to your client secret.
5. Set the `Auth_Audience` to your OAuth audience.
6. Set the `OAuth2TokenEndPoint` to your token endpoint.
7. Optionally, set the `Auth_Scope` to your OAuth scope.

Browser Based

This authentication mechanism requires SSL to be enabled.

To configure OAuth 2.0 browser based authentication:

1. Set the `AuthMech` property to `11`.
2. Set the `Auth_Flow` property to `2`.
3. Set the `Auth_Client_ID` to your client ID.
4. Set the `Auth_Client_Secret` to your client secret.
5. Set the `Auth_Audience` to your OAuth audience.
6. Set the `OAuth2TokenEndPoint` to your token endpoint.
7. Set the `OAuth2AuthorizationEndPoint` to your authorization endpoint.
8. Set the `TokenCachePassPhrase` property to a password of your choice. This is the key used for refresh token encryption.
9. Optionally, set the `Auth_Scope` to your OAuth scope.

Configuring SSL Verification in a Non-Windows Machine

If you are connecting to an Impala 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 can be configured to verify the identity of the 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 `simba.impalaodbc.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 Impala 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 allow authentication, when the certificate's revocation status is undetermined, set the CheckCertRevocation attribute to `1`.
6. 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

When connecting to a server that is running Impala 2.0 or later, you can use the connector to apply configuration properties to the Impala 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 `simba.impalaodbc.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.

Important:
This feature is not supported for earlier versions of Impala, where the SET statement can only be executed from within the Impala shell.

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 MEM_LIMIT query option to 1 GB and the REQUEST_POOL query option to myPool, type the following in the `odbc.ini` file:

   ```
   SSP_MEM_LIMIT=1000000000
   ```

   ```
   SSP_REQUEST_POOL=myPool
   ```

   Or, to set those properties in a connection string, type the following:

   ```
   SSP_MEM_LIMIT={1000000000};SSP_REQUEST_POOL={myPool}
   ```

   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.

2. 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 in a Non-Windows Machine

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 `simba.impalaodbc.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 on a non-Windows machine:

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.
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. Optionally, to prefix the log file name with the user name and process ID associated with the connection, set the `UseLogPrefix` property to `1`.
6. Save the `simba.impalaodbc.ini` configuration file.
7. Restart your ODBC application to make sure that the new settings take effect.

The Simba Apache Impala ODBC Connector produces the following log files at the location you specify using the `LogPath` key:

- A `impalaodbcdriver.log` file that logs connector activity that is not specific to a connection.
- A `impalaodbcdriver_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.

If you set the `UseLogPrefix` property to `1`, then each file name is prefixed with `[UserName]_[ProcessID]_`, where *[UserName]* is the user name associated with the connection and *[ProcessID]*

is the process ID of the application through which the connection is made. For more information, see UseLogPrefix.

To disable logging on a non-Windows machine:

1. Set the `LogLevel` key to `0`.
2. Save the `simba.impalaodbc.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 Impala using that particular DSN or string. As an alternative, you can specify settings that apply to every connection that uses the Simba Apache Impala ODBC Connector by configuring them in the `simba.impalaodbc.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 `simba.impalaodbc.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 SASL User Name authentication using "simba" as the user name, type the following:

   ```
   AuthMech=SASL User Name
   ```

   ```
   UID=simba
   ```

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

Testing the Connection in Non-Windows Machine

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 Using a Connection String.

   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 Options

Impala supports multiple authentication mechanisms. You must determine the authentication type that your server is using. The authentication methods available in the Simba Apache Impala ODBC Connector are as follows:

- No Authentication
- Kerberos
- SAML 2.0
- SASL User Name
- JWT
- User Name And Password

Note:

- The default configuration of Impala requires the Simba Apache Impala ODBC Connector to be configured to use the No Authentication mechanism.
- In addition to regular Kerberos authentication, the connector also supports an advanced configuration of Kerberos authentication that allows concurrent connections within the same process to use different Kerberos user principals.

In addition to authentication, you can configure the connector to connect over SSL or use SASL to handle authentication.

The Impala server uses SASL (Simple Authentication and Security Layer) to support some of the authentication methods. Kerberos is supported with the SASL GSSAPI mechanism. SASL User Name and User Name And Password (with SASL enabled) are supported with the SASL PLAIN mechanism.

SASL mechanisms	Non-SASL mechanisms
■ Kerberos ■ SASL User Name ■ User Name And Password (with SASL enabled)	■ No Authentication ■ SAML 2.0 ■ JWT ■ User Name And Password (without SASL enabled)

Note:
Thrift (the layer for handling remote process communication between the Simba Apache Impala ODBC Connector and the Impala server) has a limitation where it cannot detect a mix of non-SASL and SASL mechanisms being used between the connector and the server. If this happens, the connector will appear to hang during connection establishment.

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:

- *[AccessToken]* is your access token for authenticating the connection through the OAuth 2.0 protocol.
- *[DomainName]* is the fully qualified domain name of the Impala server host.
- *[MappingFile]* is the full path to a JSON file that maps your Impala user name to a Kerberos user principal name and a keytab file.
- *[PortNumber]* is the number of the TCP port that the Impala server uses to listen for client connections.
- *[Realm]* is the Kerberos realm of the Impala server host.
- *[Server]* is the IP address or host name of the Impala server to which you are connecting.
- *[ServiceName]* is the Kerberos service principal name of the Impala server.
- *[URL]* is the partial URL corresponding to the the Impala server.

- *[YourPassword]* is the password corresponding to your user name.
- *[YourUserName]* is the user name that you use to access the Impala server.

Connecting to an Impala Server Without Authentication

The following is the format of a DSN-less connection string that connects to an Impala server that does not require authentication:

Driver=Simba Impala ODBC Driver;Host=*[Server]*;
Port=*[PortNumber]*;

For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;

If you are connecting to the server through SSL, then set the `SSL` property to `1`. For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;SSL=1;

Connecting to an Impala Server that Requires Kerberos Authentication

The following is the format of a DSN-less connection string that connects to an Impala server requiring Kerberos authentication:

Driver=Simba Impala ODBC Driver;Host=*[Server]*;
Port=*[PortNumber]*;AuthMech=Kerberos;KrbRealm=*[Realm]*;
KrbFQDN=*[DomainName]*;KrbServiceName=*[ServiceName]*;

For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;AuthMech=Kerberos;KrbRealm=SIMBA;
KrbFQDN=localhost.localdomain;KrbServiceName=impala;

If you are connecting to the server through SSL, then set the `SSL` property to `1`. For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;AuthMech=Kerberos;KrbRealm=SIMBA;
KrbFQDN=localhost.localdomain;KrbServiceName=impala;SSL=1;

Connecting to an Impala Server using Advanced Kerberos Authentication

The following is the format of a DSN-less connection string that connects to an Impala server using Advanced Kerberos authentication:

Driver=Simba Impala ODBC Driver;Host=*[Server]*;
Port=*[PortNumber]*;AuthMech=Kerberos;KrbRealm=*[Realm]*;
KrbFQDN=*[DomainName]*;KrbServiceName=*[ServiceName]*;
UseKeytab=1;UID=*[YourUserName]*;
UPNKeytabMappingFile=*[MappingFile]*;

For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;AuthMech=Kerberos;KrbRealm=SIMBA;
KrbFQDN=localhost.localdomain;KrbServiceName=impala;
UseKeytab=1;UID=simba;
UPNKeytabMappingFile=C:\Temp\simba.keytab;

If you are connecting to the server through SSL, then set the `SSL` property to `1`. For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;AuthMech=Kerberos;KrbRealm=SIMBA;
KrbFQDN=localhost.localdomain;KrbServiceName=impala;
UseKeytab=1;UID=simba;
UPNKeytabMappingFile=C:\Temp\simba.keytab;SSL=1;

Connecting to an Impala Server using SAML 2.0

The following is the format of a DSN-less connection string that connects to an Impala server using SAML 2.0 authentication:

Driver=Simba Impala ODBC Driver;Host=*[Server]*;
Port=*[PortNumber]*;AuthMech=SAML_2.0;TransportMode=http;HttpPath=*[URL]*;SSL=1

For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;AuthMech=SAML_2.0;TransportMode=http;HttpPath=cliservice;SSL=1

Connecting to a Impala Server using a JSON Web Token (JWT)

The following is the format of a DSN-less connection string for connecting to a Impala server using a JWT:

Driver=Simba Impala ODBC Driver;Host=*[Server]*;
Port=*[PortNumber]*;AuthMech=`JWT`;JWTString=`[token]`;TransportMode=`http`; HttpPath=`[URL]`;
SSL=`1`;

For example:

Driver=Simba Impala ODBC Driver;Host=`192.168.222.160`;
Port=`21050`;AuthMech=`JWT`; JWTString=`s0m3t0ken5tr1ngh3re`;TransportMode=`http`;
HttpPath=`cliservice`; SSL=`1`;

Connecting to an Impala Server that Requires User Name Authentication

The following is the format of a DSN-less connection string that connects to a Impala server requiring User Name authentication. By default, the connector uses anonymous as the user name.

Driver=Simba Impala ODBC Driver;Host=*[Server]*;
Port=*[PortNumber]*;AuthMech=`SASL User Name`;

For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;AuthMech=SASL User Name;

If you are connecting to the server through SSL, then set the SSL property to 1. For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;AuthMech=SASL User Name;SSL=1;

Connecting to an Impala Server with LDAP Authentication or other User Name and Password Authentication Enabled

The following is the format of a DSN-less connection string that connects to an Impala server with LDAP authentication, or another form of username/password authentication, enabled:

Driver=Simba Impala ODBC Driver;Host=*[Server]*;
Port=*[PortNumber]*;AuthMech=User Name and Password;UID=*[UserName]*;
PWD=*[Password]*;

For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;AuthMech=User Name and Password;UID=simba;PWD=simba;

If you are connecting to the LDAP-enabled server through SSL, then set the SSL property to 1. For example:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=21050;AuthMech=User Name and Password;UID=simba;PWD=simba;SSL=1;

Connecting to a Impala Server that Requires OAuth 2.0 Authentication

The following is the format of a DSN-less connection string that connects to an Impala server requiring OAuth 2.0 authentication:

Token pass-through

Driver=Simba Impala ODBC Driver;Host=*[Server]*;
Port=*[PortNumber]*;AuthMech=OAuth 2.0;Auth_Flow=0;Auth_AccessToken=
[AccessToken];TransportMode=http;

For example, using token pass-through authentication:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;
Port=10000;AuthMech=OAuth 2.0;Auth_Flow=0;Auth_
AccessToken=P9QcyQ7prK2LwUMZMpFQ4R+6jd;TransportMode=http;

Client Credentials

Driver=Simba Impala ODBC Driver;Host=[Server];Port=[PortNumber];AuthMech=OAuth 2.0;Auth_
Flow=1;Auth_Audience=[audience];Auth_Client_ID=[client_id];Auth_Client_Secret=[client_
secret];OAuth2TokenEndPoint=[token_endpoint];TransportMode=http;

For example, using client credentials authentication:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;Port=10000;AuthMech=OAuth 2.0;Auth_Flow=1;Auth_Audience=https://dev-xxx.us.auth0.com/api/v2/;Auth_Client_ID=KZJdh;Auth_Client_Secret=Kw41H;OAuth2TokenEndPoint=https://dev-xxx.us.auth0.com/oauth/token;TransportMode=http;

Browser based

Driver=Simba Impala ODBC Driver;Host=[Server];Port=[PortNumber];AuthMech=OAuth 2.0;Auth_Flow=2;Auth_Audience=[audience];Auth_Client_ID=[client_id];Auth_Client_Secret=[client_secret];OAuth2TokenEndPoint=[token_endpoint];OAuth2AuthorizationEndPoint={authorization_endpoint};TransportMode=http;

For example, using browser based authentication:

Driver=Simba Impala ODBC Driver;Host=192.168.222.160;Port=10000;AuthMech=OAuth 2.0;Auth_Flow=2;Auth_Audience=https://dev-xxx.us.auth0.com/api/v2/;Auth_Client_ID=KZJdh;Auth_Client_Secret=Kw41H;OAuth2TokenEndPoint=https://dev-xxx.us.auth0.com/oauth/token;OAuth2AuthorizationEndPoint=https://dev-xxx.us.auth0.com/authorize;TransportMode=http;

Features

For more information on the features of the Simba Apache Impala ODBC Connector, see the following:

- Data Types
- Catalog and Schema Support
- SQL Translation
- Server-Side Properties
- Active Directory
- Write-back
- Security and Authentication

Data Types

The Simba Apache Impala ODBC Connector supports many common data formats, converting between Impala data types and SQL data types.

The table below lists the supported data type mappings.

Impala Type	SQL Type
ARRAY	SQL_VARCHAR
BIGINT	SQL_BIGINT
BOOLEAN	SQL_BOOLEAN
CHAR **Note:** Only available in CDH 5.2 or later.	SQL_CHAR **Note:** SQL_WCHAR is returned instead if the Use SQL Unicode Types configuration option (the `UseUnicodeSqlCharacterTypes` key) is enabled.
DATE **Note:** DATE data types are only supported in Impala 3.3 or later.	SQL_DATE
DECIMAL	SQL_DECIMAL

Impala Type	SQL Type
Note: Only available in CDH 5.2 or later.	
DOUBLE **Note:** REAL is an alias for DOUBLE.	SQL_DOUBLE
FLOAT	SQL_REAL
INT	SQL_INTEGER
MAP	SQL_VARCHAR
SMALLINT	SQL_SMALLINT
STRUCT	SQL_VARCHAR
TIMESTAMP	SQL_TIMESTAMP
TINYINT	SQL_TINYINT
VARCHAR **Note:** Only available in CDH 5.2 or later.	SQL_VARCHAR **Note:** SQL_WVARCHAR is returned instead if the Use SQL Unicode Types configuration option (the `UseUnicodeSqlCharacterTypes` key) is enabled.

Catalog and Schema Support

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

SQL Translation

The Simba Apache Impala ODBC Connector can parse queries locally before sending them to the Impala server. This feature allows the connector to calculate query metadata without executing the query, support query parameters, and support extra SQL features such as ODBC escape sequences and additional scalar functions that are not available in the Impala-shell tool.

Note:

The connector does not support translation for queries that reference a field contained in a nested column (an ARRAY, MAP, or STRUCT column). To retrieve data from a nested column, make sure that the query is written in valid Impala SQL syntax.

Server-Side Properties

The Simba Apache Impala ODBC Connector 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.

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.

Active Directory

The Simba Apache Impala ODBC Connector 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, according to Apache's documentation, so that users in the Active Directory realm can access services in the MIT Kerberos Hadoop realm.

Write-back

The Simba Apache Impala ODBC Connector supports translation for the following syntax:

- INSERT
- CREATE
- DROP

The connector also supports translation for UPDATE and DELETE syntax, but only when querying Kudu tables while connected to an Impala server that is running Impala 2.7 or later.

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 Impala SQL.

Security and Authentication

To protect data from unauthorized access, some Impala data stores require connections to be authenticated with user credentials or encrypted using the SSL protocol. The Simba Apache Impala ODBC Connector 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,the OAuth 2.0 protocol, your Impala user name only, or your Impala user name and password. You must use the authentication mechanism that matches the security requirements of the Impala server. For information about determining the appropriate authentication mechanism to use

based on the Impala server configuration, see Authentication Options. For detailed connector configuration instructions, see Configuring Authentication in Windows or Configuring Authentication on a Non-Windows Machine.

Additionally, the connector supports SSL connections with or without one-way authentication. If the server has an SSL-enabled socket, then you can configure the connector to connect to it.

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 Simba Apache Impala ODBC Connector 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 following dialog boxes:

- Simba Impala ODBC Connector DSN Setup
- Advanced Options
- OAuth Options
- Keytab Options
- Server Side Properties
- SSL Options

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

Note:
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 Simba Apache Impala ODBC Connector, or via the key name when using a connection string, configuring connector-wide settings, or configuring a connection from a Linux/macOS/AIX/Solaris machine:

- Audience
- Allow Common Name Host Name Mismatch
- Allow Self-Signed Server Certificate
- Async Exec Poll Interval
- Access Token
- Proxy Host
- Proxy Password
- Proxy Port
- Proxy Username
- Realm
- Restrict Metadata with Current

- Authentication Flow
- Authorization Endpoint
- Canonicalize Principal FQDN
- CheckCertificate Revocation
- Client ID
- Client Secret
- Convert Key Name to Lower Case
- Database
- Default Keytab File
- Delegation UID
- Enable Auto Reconnect
- Enable Query Retry
- Enable Token Cache
- Enable Routing To Same Coordinator For Cached SAML Auth Cookie
- Enable SAML Auth Cookie Caching
- Enable SSL
- Host
- Host FQDN
- HTTP Path
- Ignore SQL_DRIVER_NOPROMPT
- Ignore Transactions
- JWTString
- Log Level
- Log Path
- Schema
- Result Set Cache Size
- Retry Interval
- Rows Fetched Per Block
- Service Name
- Socket Timeout
- String Column Length
- TokenCachePassPhrase
- Token Endpoint
- TokenRenewLimit
- Transport Buffer Size
- Transport Mode
- Trusted Certificates
- UPN Keytab Mapping File
- Use Keytab
- Use Native Query
- Use Only SSPI
- Use Proxy
- Use Simple Authentication and Security Layer (SASL)
- Use SQL Unicode Types
- Use System Trust Store
- User Name

- Max File Size
- Max Number Files
- Maximum Retries
- Mechanism
- Minimum TLS
- OAuth Scope
- Password
- Port

Access Token

The access token for authenticating the connection through the OAuth 2.0 protocol.

Note: In case of unixODBC, when the Auth_AccessToken line length is longer than the maximum limit of 1000, add the following in your odbc.ini file:

In the ODBC section:

- `Auth_AccessToken`=(your access token)

In the DSN section:

- `ConfigsFromFileDSN`=Auth_AccessToken
- `FILEDSNPATH`=(Full path of the odbc.ini file)
- `Auth_AccessToken`=(your access token)

  If you have multiple DSN configured in your odbc.ini file and each of them require a different `Auth_AccessToken`, you can add the `Auth_AccessToken` to the ODBC section of a different ini file, and configure the FILEDSNPATH in your DSN to point to this `inifile`.

Key Name	Default Value	Required
`Auth_AccessToken`	None	Yes, if the authentication mechanism is OAuth 2.0 and the work flow is Token Passthrough (`0`).

Allow Common Name Host Name Mismatch

This option specifies whether a CA-issued SSL certificate name must match the host name of the Impala 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 Impala server.
- Disabled (`0`): The CA-issued SSL certificate name must match the host name of the Impala 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 anImpala server that uses a self-signed certificate.

- Enabled (`1`): The connector authenticates the Impala server even if the server is using a self-signed certificate.
- Disabled (`0`): The connector does not allow self-signed certificates from the server.

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

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

Async Exec Poll Interval

The time in milliseconds between each poll for the query execution status.

"Asynchronous execution" refers to the fact that the RPC call used to execute a query against Impala is asynchronous. It does not mean that ODBC asynchronous operations are supported.

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

Audience

The audience used in OAuth connections.

Key Name	Default Value	Required
Auth_Audience	None	No

Authorization Endpoint

The Authorization endpoint used in OAuth connections.

Key Name	Default Value	Required
OAuth2AuthorizationEndPoint	None	No

Authentication Flow

This option specifies the type of OAuth authentication flow that the connector uses when the Mechanism option is set to OAuth 2.0 or SAML_2.0 (or when AuthMech is set to OAuth 2.0 or SAML_2.0). When SAML_2.0 is selected, only Browser is supported for this option.

When this option is set to Token Passthrough (0), the connector uses the access token specified by the Access Token (Auth_AccessToken) option to authenticate the connection to the server. For more information, see Access Token.

When this option is set to Client Credentials (1), the connector uses the client credentials to authenticate the connection to the server.

When this option is set to Browser Based Authorization Code (2), the connector uses the browser based authorization code flow to authenticate the connection to the server.

Key Name	Default Value	Required
Auth_Flow	Token Passthrough (0) for OAuth 2.0 and Browser for SAML_2.0	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 ID

The client ID used in OAuth connections.

Key Name	Default Value	Required
`Auth_Client_ID`	None	No

Client Secret

The client secret associated with the client ID used in OAuth connections.

Key Name	Default Value	Required
`Auth_Client_Secret`	None	No

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

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 Impala command prompt, type `show databases`.

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

Default Keytab File

The full path to the keytab file that the connector uses to obtain the ticket for Kerberos authentication.

Note:

- This option is applicable only when the authentication mechanism is set to Kerberos (`AuthMech=Kerberos`) and the Use Keytab option is enabled (`UseKeytab=1`).
- If the UPN Keytab Mapping File option (the `UPNKeytabMappingFile` key) is set to a JSON file with a valid mapping to a keytab, then that keytab takes precedence.

If you do not set this option but the Use Keytab option is enabled (`UseKeytab=1`), then the MIT Kerberos library will search for a keytab using the following search order:

- The file specified by the KRB5_KTNAME environment variable.
- The `default_keytab_name` setting in the `[libdefaults]` section of the Kerberos configuration file (`krb5.conf/krb5.ini`).
- The default keytab file specified in the MIT Kerberos library. Typically, the default file is `C:\Windows\krb5kt` for Windows platforms and `/etc/krb5.keytab` for non-Windows platforms.

Key Name	Default Value	Required
`DefaultKeytabFile`	None	No

Delegation UID

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

Key Name	Default Value	Required
DelegationUID	None	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 Query Retry

This option specifies whether the connector automatically retries queries that are sent to the server but fail.

- Enabled (1): The connector retries failed queries.
- Disabled (0): The connector only submits each query once.

Query retry settings are configured through the following options:

- Maximum Retries
- Result Set Cache Size
- Retry Interval
- GlobalResultSetCache

Key Name	Default Value	Required
EnableQueryRetry	Clear (0)	No

Enable Token Cache

This option specifies whether the connector enables or disables token cache for OAuth2 Browser Based authentication.

- Enabled (1): The connector enables the token cache for OAuth2 Browser Based authentication.
- Disabled (0): The connector disables the token cache for OAuth2 Browser Based authentication.

Key Name	Default Value	Required
EnableTokenCache	Enabled (1)	No

Enable Routing To Same Coordinator For Cached SAML Auth Cookie

This option specifies whether the connector uses the cached SAML auth cookie to route to the same coordinator in subsequent connections.

- Enabled (1): The connector uses the cached SAML auth cookie to route to the same coordinator in subsequent connections.
- Disabled (0): The connector does not use the cached SAML auth cookie to route to the same coordinator in subsequent connections.

Note:

- This property is available only in Windows.
- This property is active only when the configuration option Enable SAML Auth Cookie Caching is enabled.
- This property addresses issues where SAML auth cookie caching may fail in an Impala cluster with active-active coordinators. If the cluster uses a coordinator-specific auth cookie mechanism, the cached SAML auth cookie is valid only for the issuing coordinator. When the cached cookie happens to be routed to a different coordinator, authentication fails. In this case, enabling this configuration ensures consistent routing to the correct coordinator. If the cluster uses a shared auth cookie mechanism across coordinators, the cached SAML auth cookie remains valid across all coordinators. In this case, disabling this configuration is recommended to avoid unnecessary routing and improve load balancing.

Key Name	Default Value	Required
EnableRoutingToSameCoordinatorForCachedSamlAuthCookie	Clear (0)	No

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 Impala server.

- Enabled (1): The client communicates with the Impala 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.

Key Name	Default Value	Required
SSL	Clear (0)	No

HTTP Path

The partial URL corresponding to the Impala 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:21050/gateway/sandbox/impala/version`, you would set HTTP Path to `/gateway/sandbox/impala/version`.

Key Name	Default Value	Required
`HTTPPath`	None	No

Host

The IP address or host name of the Impala server.

Key Name	Default Value	Required
`Host` (or `Server`)	None	Yes

Host FQDN

The fully qualified domain name of the Impala host.

When the value of Host FQDN is `_HOST`, the connector uses the Impala server host name as the fully qualified domain name for Kerberos authentication.

Key Name	Default Value	Required
`KrbFQDN`	`_HOST`	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

Ignore Transactions

This option specifies whether the connector should simulate transactions, or return an error.

- Enabled (`1`): The connector simulates transactions, enabling queries that contain transaction statements to be run successfully. The transactions are not executed.
- Disabled (`0`): The connector returns an error if it attempts to run a query that contains transaction statements.

Note:
ODBC does not support transaction statements, so they cannot be executed.

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

JWTString

The JSON Web Token used to access the server.

Note:
This option is only available when using `JWT` as the authentication mechanism.

Note:
When using unixODBC:

When the `JWTString` line length is longer than the maximum limit of `1000`, add the following in your `odbc.ini` file:

In the ODBC section:

`JWTString`=(your JWT token)

In the DSN section:

- `FILEDSNPATH`=(Full path of the `odbc.ini` file)
- `ConfigsFromFileDSN=JWTString`
- `JWTString`=(your JWT token)

If you have multiple DSN configured in your `odbc.ini` file and each of them require a different `JWTString`, you can add the `JWTString` to the ODBC section of a different `ini` file, and configure the `FILEDSNPATH` in your DSN to point to this `ini`file.

Key Name	Default Value	Required
`JWTString`	None	Yes, if the authentication mechanism is `JWT`.

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 `simbaimpalaodbcdriver.log` file that logs connector activity that is not specific to a connection.
- A `simbaimpalaodbcdriver_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.

If you enable the `UseLogPrefix` connection property, the connector prefixes the log file name with the user name associated with the connection and the process ID of the application through which the connection is made. For more information, see UseLogPrefix.

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

Maximum Retries

The maximum number of times that the connector retries a failed query, when query retry is enabled.

Also see Enable Query Retry.

Key Name	Default Value	Required
MaxNumQueryRetries	5	No

Mechanism

The authentication mechanism to use.

Select one of the following settings, or set the key to the authentication name:

- No Authentication (`No Authentication` or`0`)
- Kerberos (`Kerberos` or `1`)
- SASL User Name (`SASL User Name` or `2`)
- User Name And Password (`User Name and Password` or `3`)
- JWT (`JWT` or `8`)
- OAuth 2.0 (`OAuth 2.0` or `9`)
- SAML_2.0 (`SAML_2.0`)
- MapR SASL (`13`)

Note: The `AuthMech` property now uses the authentication mechanism's name instead of the corresponding numbers. The connector continues to recognize the corresponding numbers (`0`, `1`, `2`, `3`, and `8`) for backwards compatibility.\

Note: For connector version 2.7.1. and earlier, when using JWT to connect to an Impala virtual warehouse cluster with active-active coordinators, the connector may return a 401 error for the first operation after the connection is established.
As a workaround, add the following configuration to the DSN or connector string:
`AuthorizationHeaderReplacementCookie=impala.auth`

Key Name	Default Value	Required
`AuthMech`	No Authentication (`No Authentication` or `0`)	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

OAuth Scope

The scope used in the OAuth 2.0 connection.

Key Name	Default Value	Required
Auth_Scope	None	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 (User Name and Password).

Port

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

Key Name	Default Value	Required
Port	21050	Yes

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.

Realm

The realm of the Impala host.

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

Key Name	Default Value	Required
KrbRealm	NULL	No

Restrict Metadata with Current Schema

This option specifies whether the connector should restrict catalog function results to tables and views in the current schema if a catalog function call is made without specifying a schema, or if the schema is specified as the wildcard character %.

Note:
Restricting results to the tables and views in the current schema may improve the performance of catalog calls that do not specify a schema.

- Enabled (1): The connector restricts catalog function results to the current schema if a schema is not specified.
- Disabled (0): The connector does not restrict catalog function results to the current schema if a schema is not specified.

Key Name	Default Value	Required
CurrentSchema RestrictedMetadata	Clear (0)	No

Result Set Cache Size

The maximum amount of memory that the result set cache for the current connection can occupy. Values must be specified in: B (bytes), KB (kilobytes), MB (megabytes), or GB (gigabytes).

When query retries are enabled, the connector temporarily caches result sets. For more information about this feature, see Enable Query Retry.

For information about setting a maximum total cache size for all concurrent connections, see GlobalResultSetCache.

Key Name	Default Value	Required
ResultSetCacheSize	`20MB`	No

Retry Interval

The amount of time that the connector waits between query retry attempts, when query retry is enabled. Values can be specified in S (seconds) or MS (milliseconds).

Also see Enable Query Retry.

Key Name	Default Value	Required
QueryRetryInterval	`5S`	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

Service Name

The Kerberos service principal name of the Impala server.

Key Name	Default Value	Required
`KrbServiceName`	`impala`	No

Socket Timeout

The number of seconds that the TCP socket waits for a response from the server before timing out the request and returning an error message.

When this option is set to `0`, the TCP socket does not time out any requests.

Key Name	Default Value	Required
SocketTimeout	30	No

String Column Length

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

Key Name	Default Value	Required
StringColumnLength	32767	No

TokenCachePassPhrase

The password used to protect the token cache.

Note: This option is required only on non-Windows platform.

Key Name	Default Value	Required
TokenCachePassPhrase	None	Yes

Token Endpoint

The token endpoint used in OAuth connections.

Key Name	Default Value	Required
OAuth2TokenEndPoint	None	No

TokenRenewLimit

TokenRenewLimit sets the threshold (in minutes) for renewing the access token when its remaining lifetime is less than or equal to this value.

Key Name	Default Value	Required
TokenRenewLimit	0	No

Transport Buffer Size

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.

Key Name	Default Value	Required
TSaslTransportBufSize	1000	No

Transport Mode

The transport protocol to use in the Thrift layer.

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

- Binary (`0` or `binary`)
- SASL (`1` or `sasl`)
- HTTP (`2` or `http`)

Note:

- For information about how to determine which Thrift transport protocols your Impala server supports, see Mechanism.
- If this property is specified, it takes precedence over the Use SASL property (`UseSASL`).
- If this property is not specified and the Use SASL property is specified, the Use SASL property takes precedence over the default value of this property.

For more information, see Use Simple Authentication and Security Layer (SASL).

Key Name	Default Value	Required
`TransportMode`	Binary (`0`) if using No Authentication, otherwise SASL (`1`).	No

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

UPN Keytab Mapping File

The full path to a JSON file that maps your Impala user name to a Kerberos user principal name and a keytab file.

Note:
This option is applicable only when the authentication mechanism is set to Kerberos (AuthMech=Kerberos) and the Use Keytab option is enabled (UseKeytab=1).

The mapping in the JSON file must be written using the following schema, where *[UserName]* is the Impala user name, *[KerberosUPN]* is the Kerberos user principal name, and *[Keytab]* is the full path to the keytab file:

```
{
        "[UserName]": {
                "principal" : "[KerberosUPN]",
                "keytab": "[Keytab]"
        },
... }
```

For example, the following file maps the Impala user name **simba** to the **simba@SIMBA** Kerberos user principal name and the C:\Temp\simba.keytab file:

```
{
        "simba": {
                "principal" : "simba@SIMBA",
                "keytab": "C:\Temp\simba.keytab"
        },
... }
```

If parts of the mapping are invalid or not defined, then the following occurs:

- If the mapping file fails to specify a Kerberos user principal name, then the connector uses the Impala user name as the Kerberos user principal name.
- If the mapping file fails to specify a keytab file, then the connector uses the keytab file that is specified in the Default Keytab File setting.
- If the entire mapping file is invalid or not defined, then the connector does both of the actions described above.

Key Name	Default Value	Required
`UPNKeytabMappingFile`	None	No

Use Keytab

This option specifies whether the connector obtains the ticket for Kerberos authentication by using a keytab.

- Enabled (`1`): The connector uses a keytab to obtain a ticket before authenticating the connection using Kerberos.
- Disabled (`0`): The connector does not attempt to obtain the Kerberos ticket, and assumes that a valid ticket is already available in the credentials cache.

Note:
This option is applicable only when the authentication mechanism is set to Kerberos (`AuthMech=Kerberos`).

If you enable this option but do not set the Default Keytab File option (the `DefaultKeytabFile` key), then the MIT Kerberos library will search for a keytab file using the following search order:

1. The file specified by the KRB5_KTNAME environment variable.
2. The `default_keytab_name` setting in the `[libdefaults]` section of the Kerberos configuration file (`krb5.conf/krb5.ini`).
3. The default keytab file specified in the MIT Kerberos library. Typically, the default file is `C:\Windows\krb5kt` for Windows platforms.

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

Use Native Query

This option specifies whether the connector uses native Impala SQL queries, or converts the queries emitted by an application into an equivalent form in Impala SQL. If the application is Impala-aware and already emits Impala SQL, 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 Impala SQL queries directly.
- Disabled (`0`): The connector transforms the queries emitted by an application and converts them into an equivalent form in Impala SQL.

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.

- Enable For This DSN (`1` in the DSN entry in the registry): The connector handles Kerberos authentication in the DSN connection by using the SSPI plugin instead of MIT Kerberos by default.
- Enable For DSN-less Connections (`1` in the connector configuration section of the registry): The connector handles Kerberos authentication in DSN-less connections by using the SSPI plugin instead of MIT Kerberos by default.

If you want all connections that use the Simba Apache Impala ODBC Connector to use the SSPI plugin by default, then enable Use Only SSPI for both DSN and DSN-less connections.

- 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

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 Impala server.

Key Name	Default Value	Required
UseProxy	Clear (0)	No

Use Simple Authentication and Security Layer (SASL)

This option specifies whether the connector uses SASL to handle authentication.

- Enabled (1): The connector uses SASL to handle authentication.
- Disabled (0): The connector does not use SASL.

This option is configurable only when you are using the User Name And Password authentication mechanism. If the connector is configured to use the other authentication mechanisms, then it uses the default setting for the Use Simple Authentication and Security Layer (SASL) option.

Note:

- If the Transport Mode property (TransportMode) is specified, it takes precedence over this property.
- If the Transport Mode property is not specified and this property is specified, this property takes precedence over the default value of Transport Mode.

For more information, see Transport Mode.

Key Name	Default Value	Required
UseSASL	0 if using No Authentication. 1 if using User Name And Password or Kerberos or SASL User Name authentication.	No

Use SQL Unicode 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
UseSQLUnicodeTypes	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 the Impala server.

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 (`User Name and Password`). No, if the authentication mechanism is SASL User Name (`SASL User Name`).

Configuration Options Having Only Key Names

The following configuration options do not appear in the Windows user interface for the Simba Apache Impala ODBC Connector. They are accessible only when you use a connection string, configure connector-wide settings, or configure a connection from a Linux/macOS/AIX/Solaris machine:

- ConfigsFromFileDSN
- DelegationUserIDCase
- DisableOptimizedEncodingConverter
- Driver
- EnableMultiSocketFetch

- GlobalResultSetCache
- HTTPAuthCookies
- http.header.
- MaxCatalogNameLen
- MaxColumnNameLen
- MaxSchemaNameLen
- MaxTableNameLen
- MaxNumMultiSocketFetchHandlers
- MaxNumMultiSocketFetchRowsetPerOperation
- SSP_
- SSOWebServerTimeout

The `UseLogPrefix` property must be configured as a Windows Registry key value, or as a connector-wide property in the `simba.impalaodbc.ini` file for macOS or Linux.

- UseLogPrefix

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

DisableOptimizedEncodingConverter

This connector-wide option controls which data encoding converter the connector uses.

Note:
Enabling this option may result in decreased performance.

- true: The connector uses a standard ICU converter.
- false: The connector uses an optimized converter.

Important:
This option applies to every connection that uses the Simba Apache Impala ODBC Connector. For more information, see Setting Connector-Wide Configuration Options in Windows or Setting Connector-Wide Configuration Options on a Non-Windows Machine.

Key Name	Default Value	Required
DisableOptimizedEncodingConverter	false	No

Driver

In Windows, the name of the installed connector for (Simba Impala ODBC Driver).

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	Simba Impala ODBC Driver when installed in Windows, or the absolute path of the connector shared object file when installed on a non-Windows machine.	Yes

EnableMultiSocketFetch

This properties specifies whether to enable the multi-socket fetch feature for the connection.

1: The connector enables multi-socket fetch.

0: The connector disables multi-socket fetch.

Note:

- The multi-socket fetch feature is currently not supported when using SAML 2.0 authentication or JWT authentication.
- The multi-socket fetch feature is currently not supported when connecting to an Impala Virtual Warehouse cluster with active-active coordinators.

Key Name	Default Value	Required
EnableMultiSocketFetch	0	No

GlobalResultSetCache

The maximum total amount of memory that can be occupied by result set caches across concurrent connections. Values must be specified in: B (bytes), KB (kilobytes), MB (megabytes), or GB (gigabytes).

For example, when this property is set to the default value of 200MB, if you have 3 concurrent Impala connections, the total amount of memory consumed by the 3 result set caches for those connections cannot exceed 200MB.

These result set caches are used only when query retries are enabled. For more information about this feature, see Enable Query Retry.

For information about setting a maximum cache size for the current connection only, see Result Set Cache Size.

Key Name	Default Value	Required
GlobalResultSetCache	200MB	No

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`	`impala.auth,JSESSIONID,KNOXSESSIONID, impala.session.id`	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 Transport Mode.

Key Name	Default Value	Required
`http.header.`	None	No

MaxCatalogNameLen

The maximum number of characters that can be returned for catalog names. This option can be set to any integer from 0 to 65535, inclusive. To indicate that there is no maximum length or that the length is unknown, set this option to `0`.

Key Name	Default Value	Required
MaxCatalogNameLen	0	No

MaxColumnNameLen

The maximum number of characters that can be returned for column names. This option can be set to any integer from 0 to 65535, inclusive. To indicate that there is no maximum length or that the length is unknown, set this option to 0.

Key Name	Default Value	Required
MaxColumnNameLen	0	No

MaxNumMultiSocketFetchHandlers

The maximum number of multi-socket fetch handlers for the connection.

Key Name	Default Value	Required
MaxNumMultiSocketFetchHandlers	5	No

MaxNumMultiSocketFetchRowsetPerOperation

The maximum number of row sets to buffer in memory for each operation.

Key Name	Default Value	Required
MaxNumMultiSocketFetchRowsetPerOperation	10	No

MaxSchemaNameLen

The maximum number of characters that can be returned for schema names. This option can be set to any integer from 0 to 65535, inclusive. To indicate that there is no maximum length or that the length is unknown, set this option to 0.

Key Name	Default Value	Required
MaxSchemaNameLen	256	No

MaxTableNameLen

The maximum number of characters that can be returned for table names. This option can be set to any integer from 0 to 65535, inclusive. To indicate that there is no maximum length or that the length is unknown, set this option to 0.

Key Name	Default Value	Required
MaxTableNameLen	0	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

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_MEM_LIMIT=1000000000
```

```
SSP_REQUEST_POOL=myPool
```

Or, to set those properties in a connection string, type the following:

```
SSP_MEM_LIMIT={1000000000};SSP_REQUEST_POOL={myPool}
```

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

Important:
This property is supported only for connections to Impala 2.0 or later. In earlier versions of Impala, the SET statement can only be executed from within the Impala shell.

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

UseLogPrefix

This option specifies whether the connector includes a prefix in the names of log files so that the files can be distinguished by user and application.

Set the property to one of the following values:

- `1`: The connector prefixes log file names with the user name and process ID associated with the connection that is being logged.

  For example, if you are connecting as a user named "jdoe" and using the connector in an application with process ID 7836, the generated log files would be named `jdoe_7836_simbaimpalaodbcdriver.log` and `jdoe_7836_simbaimpalaodbcdriver_connection_[Number].log`, where *[Number]* is a number that identifies each connection-specific log file.

- `0`: The connector does not include the prefix in log file names.

  To configure this option for the Windows connector, you create a value for it in one of the following registry keys:

- For a 32-bit connector installed on a 64-bit machine: **HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Simba\Simba Impala ODBC Connector\Driver**
- Otherwise: **HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Impala ODBC Connector\Driver**

Use `UseLogPrefix` as the value name, and either `0` or `1` as the value data.

To configure this option for a non-Windows connector, you must use the `simba.impalaodbc.ini` file.

Key Name	Default Value	Required
`UseLogPrefix`	`0`	No

Third-Party Trademarks

IBM and AIX are trademarks or registered trademarks of IBM Corporation or its subsidiaries in Canada, United States, and/or other countries.

Kerberos is a trademark of the Massachusetts Institute of Technology (MIT).

Linux is the registered trademark of Linus Torvalds in Canada, United States and/or other countries.

Mac, macOS, Mac OS, and OS X are trademarks or registered trademarks of Apple, Inc. or its subsidiaries in Canada, United States and/or other countries.

Microsoft, MSDN, Windows, Windows Azure, Windows Server, Windows Vista, and the Windows start button are trademarks or registered trademarks of Microsoft Corporation or its subsidiaries in Canada, United States and/or other countries.

Red Hat, Red Hat Enterprise Linux, and CentOS are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in Canada, United States and/or other countries.

Solaris is a registered trademark of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

SUSE is a trademark or registered trademark of SUSE LLC or its subsidiaries in Canada, United States and/or other countries.

Apache Impala, Apache, and Impala are either registered trademarks or trademarks of The Apache Software Foundation in the United States and other countries.

All other trademarks are trademarks of their respective owners.