Simba Oracle ODBC Data Connector

Installation and Configuration Guide

Version 2.0.10

May 2024

Copyright

Contact Us

insightsoftware

www.insightsoftware.com

About This Guide

Purpose

The *Simba Oracle ODBC Data Connector Installation and Configuration Guide* explains how to install and configure the Simba Oracle 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 Oracle ODBC Connector, as well as administrators and developers integrating the connector.

Knowledge Prerequisites

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

- Familiarity with the platform on which you are using the Simba Oracle ODBC Connector
- Ability to use the data source to which the Simba Oracle 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 are 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.

Contents

About the Simba Oracle ODBC Connector

The Simba Oracle ODBC Connector enables Business Intelligence (BI), analytics, and reporting on data that is stored in Oracle databases. The connector complies with the ODBC 3.52 data standard and adds important functionality such as Unicode, as well as 32- and 64-bit support for high-performance computing environments on Windows and Linux. For macOS, the connector provides 64-bit support.

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 *Data Access Standards* on the Simba Technologies website: https://www.simba.com/resources/data-access-standards-glossary. 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 *Installation and Configuration Guide* is suitable for users who are looking to access Oracle data from their desktop environment. Application developers might also find the information helpful. Refer to your application for details on connecting via ODBC.

Note:

For information about how to use the connector in various BI tools, see the *Simba ODBC Connectors Quick Start Guide for Windows*: http://cdn.simba.com/docs/ODBC_QuickstartGuide/content/quick_start/intro.htm.

Windows Connector

Windows System Requirements

The Simba Oracle ODBC Connector supports Oracle versions 11g, 12c, 18c, and 19c, 23c, as well as autonomous data sources.

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.
- A machine that meets the following system requirements:
  - One of the following operating systems:
    - Windows 10 or 8.1
    - Windows Server 2016 or 2012 R2
  - 250 MB of available disk space; 2 GB of RAM

Before the connector can be used, the following dependencies (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 these dependencies. Otherwise, you must install these dependencies manually.

- Visual C++ Redistributable for Visual Studio 2015-2022. Download and run the installation packages available at https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170
- Oracle Instant Client 19.3. These library files must be installed in the `\lib` subfolder in the connector's installation directory. For detailed instructions, see Installing the Oracle Instant Client on Windows on page 8.

Installing the Connector on 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 Connectors 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 Oracle 2.0 32-bit.msi` for 32-bit applications
- `Simba Oracle 2.0 64-bit.msi` for 64-bit applications

To install the Simba Oracle ODBC Connector on Windows:

1. Depending on the bitness of your client application, double-click to run **Simba Oracle 2.0 32-bit.msi** or **Simba Oracle 2.0 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.

Installing the Oracle Instant Client on Windows

The Simba Oracle ODBC Connector requires Oracle Instant Client 19.3. If you obtained the connector from the Simba website, then your installation of the connector automatically includes this dependency. Otherwise, you must manually install Oracle Instant Client in the `\lib` subfolder in the installation directory of the connector.

To install the Oracle Instant Client on Windows:

1. In a web browser, navigate to https://www.oracle.com/database/technologies/instant-client/downloads.html.
2. Download the 19.3 version of the Oracle Instant Client that matches the bitness of your platform. You can use the Basic package or Basic Light package depending upon your disk space.
3. Extract the archive that you downloaded to a temporary location.
4. Copy the files from the temporary location to the `\lib` subfolder in the installation directory of the connector.

Creating a Data Source Name on Windows

Typically, after installing the Simba Oracle ODBC Connector, you need to create a Data Source Name (DSN).

Alternatively, for information about DSN-less connections, see Using a Connection String on page 46.

To create a Data Source Name on 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 Oracle.

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

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

4. Click **Add**.
5. In the Create New Data Source dialog box, select **Simba Oracle ODBC Connector** and then click **Finish**. The Simba Oracle ODBC Connector DSN 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. To specify the Oracle database that you want to connect to, do one of the following:
   - To use server information that is defined in your `tnsnames.ora` configuration file, do the following:
     a. Select the **Use TNS Service Name** check box.
     b. In the **TNS Name** field, type the net service name that you want to use.

c. On your Windows machine, update the TNS_ADMIN environment variable to point to the path of the `tnsnames.ora` file.

> **Note:**
>
> For more information about the `tnsnames.ora` file, see "Local Naming Parameters in the tnsnames.ora File" in the *Oracle Database Net Services Reference*: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/local-naming-parameters-in-tnsnames-ora-file.html.

- Or, to use simultaneous cloud connections, specify the path to an Oracle Cloud Wallet configuration file or folder by doing the following:
  a. Extract cloud wallets within separate locations. For example, `[Wallet1Dir]` and `[Wallet2Dir]`.
  b. Copy the `tnsnames.ora` contents of each wallet and paste them within your `tnsnames.ora` file (located in TNS_ADMIN).
  c. Add the respective wallet locations under the security of the respective TNS alias:

```
NameOfTNSEntry =
(DESCRIPTION=
(ADDRESS_LIST =
(ADDRESS=(PROTOCOL=tcps)
(HOST=xyz.somewhere.c0m)(PORT=12345678))
)
(CONNECT_DATA=(SERVICE_NAME=NameOfService))
(SECURITY =
(MY_WALLET_DIRECTORY = [Wallet1Dir])
)
)
```

  d. Establish the connection using TNS.

> **Note:**
>
> For more information about multiple wallet connections, see "One client needing multiple wallets and/or multiple SQLNET.ORA files - (Solved/workaround)" in the *Oracle Applications and Infrastructure Community*: https://community.oracle.com/tech/apps-infra/discussion/4301296/one-client-needing-multiple-wallets-and-or-multiple-sqlnet-ora-files-solved-workaround.

- Or, to specify server information directly in the DSN, do the following:
  a. In the **Host** field, type the name or IP address of the Oracle server.
  b. In the **Port** field, type the number of the TCP port that the server uses to listen for client connections.

     > **Note:**
     >
     > The default port used by Oracle is 1521.

  c. In the **Service Name** field, type the service name of the Oracle database that you want to access.

9. Configure authentication as needed. For more information, see Configuring Authentication on Windows on page 12.
10. Configure SSL encryption as needed. For more information, see Configuring SSL Verification on a Windows Machine on page 14.
11. Optionally, to configure advanced options such as statement caching or to configure the connector to recognize table type information from the data source, click **Optimization Options** or **Metadata Options**. For more information, see Configuring Optimization Options on Windows on page 13 and Configuring Metadata Options on Windows on page 14.
12. Optionally, to configure logging behavior for the connector, click **Logging Options**. For more information, see Configuring Logging Options on Windows on page 16.
13. 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 Oracle ODBC Driver DSN Setup dialog box are correct. Contact your Oracle server administrator as needed.

14. To save your settings and close the Simba Oracle ODBC Connector DSN Setup dialog box, click **OK**.
15. To close the ODBC Data Source Administrator, click **OK**.

Configuring Authentication on Windows

All Oracle databases require authentication. You can configure the Simba Oracle ODBC Connector to provide your credentials and authenticate the connection to the database using one of the following methods:

- Using Your Oracle Database Credentials on page 12
- Using Kerberos on page 13
- Using JWT on page 13

Important:

You cannot use the Simba Oracle ODBC Connector to connect to an Oracle Autonomous Data Warehouse instance using Kerberos authentication.

Using Your Oracle Database Credentials

You can configure the connector to authenticate the connection using your database credentials.

To configure authentication using your database credentials on Windows:

1. To access authentication options, open the ODBC Data Source Administrator where you created the DSN, select the DSN, and then click **Configure**.
2. Make sure that the **Use External Credentials** check box is cleared. If that check box is selected, then the User and Password fields are not available, and the connector uses Kerberos authentication instead.
3. In the **User** field, type your user name for accessing the Oracle database.
4. In the **Password** field, type the password corresponding to the user name you typed above.
5. To save your settings and close the dialog box, click **OK**.

Using Kerberos

You can configure the connector to use the Kerberos protocol to authenticate the connection. The connector retrieves and uses a Kerberos ticket based on the settings in the `sqlnet.ora` configuration file.

Before you can use this authentication mechanism, you must specify the necessary Kerberos settings in the `sqlnet.ora` file, and set the TNS_ADMIN environment variable on your machine to point to the path of the `sqlnet.ora` file. For more information, see the following:

- For information about configuring Kerberos for your Oracle database, including details about the settings required in the `sqlnet.ora` file, see "Configuring Kerberos Authentication" in the *Oracle Database Security Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-kerberos-authentication.html.
- For general information about the `sqlnet.ora` file, see "Parameters for the sqlnet.ora File" in the *Oracle Database Net Services Reference*: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/parameters-for-the-sqlnet-ora-file.html.

To configure Kerberos authentication on Windows:

1. To access authentication options, open the ODBC Data Source Administrator where you created the DSN, select the DSN, and then click **Configure**.
2. Select the **Use External Credentials** check box.
3. To save your settings and close the dialog box, click **OK**.

Using JWT

To configure the connector to authenticate the connection using your database credentials on Windows, see Using JWT on page 39.

Configuring Optimization Options on Windows

You can configure the options such as **Enable Statement Caching** and **Fetch Buffer Size** to modify the optimization behavior of the connector.

To configure optimization options on Windows:

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

2. To enable statement caching, select the **Enable Statement Caching** check box and, in the **Statement Cache Size** field, specify the number of statements to cache.
3. To set the buffer size that the connector uses for data retrieval, in the **Fetch Buffer Size** field, type the size of the buffer in bytes.
4. To configure the round-trip timeout using the OCI_ATTR_CALL_TIMEOUT attribute, in the **Query Timeout** field, type the number of seconds that the connector waits for a round-trip call between the client and the server to complete before timing out and returning an error.
5. To determine the bind buffer length dynamically, select the **Use Dynamic Bind Buffer** check box.
6. To save your settings and close the Simba Oracle ODBC Connector DSN Setup dialog box, click **OK**.

Configuring Metadata Options on Windows

You can configure options such as the **Enable Table Types** and **Enable User Filtration** options to modify the metadata behavior of the connector.

To configure metadata options on Windows:

1. To access the metadata options, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, then click **Configure**, and then click **Metadata Options**.
2. To configure the connector to recognize table type information from the data source, select the **Enable Table Types** check box.
3. To disable automatic implementation parameter database (IPD) population, unselect the **Auto IPD** check box.
4. To retrieve varchar type as unicode characters, select the **Force SQL_WCHAR support** check box.
5. To return the details of user-defined tables only, select **UserDefinedTables** from the **Catalog Details** drop-down list.
6. To save your settings and close the Simba Oracle ODBC Connector DSN Setup dialog box, click **OK**.

Configuring SSL Verification on a Windows Machine

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

The procedure for configuring SSL in your connection differs depending on whether or not you are connecting through TNS. For more information, see below:

- Using TNS on page 15
- Without Using TNS on page 15

Using TNS

If you have configured the connector to connect using server information that is defined in a `tnsnames.ora` configuration file, then you must make sure that the necessary SSL settings are specified in the `tnsnames.ora` file. For more information, see the following:

- For information about configuring SSL for your Oracle database, including details about the settings required in the `tnsnames.ora` file, see "Configuring Secure Sockets Layer Authentication" in the *Oracle Database Security Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-secure-sockets-layer-authentication.html.
- For general information about the `tnsnames.ora` configuration file, see "Local Naming Parameters in the tnsnames.ora File" in the *Oracle Database Net Services Reference*: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/parameters-for-the-sqlnet.ora.html.

Without Using TNS

If you have specified your server information directly in a DSN or connection string instead of using the `tnsnames.ora` configuration file, then you must configure the connector to use the TCPS protocol, which enables SSL encryption on a TCP/IP connection. The connector then encrypts the connection using the SSL settings defined in the `sqlnet.ora` configuration file.

Before configuring the connector to use TCPS, you must specify the necessary SSL settings in the `sqlnet.ora` file, and set the TNS_ADMIN environment variable on your machine to point to the path of the `sqlnet.ora` file. For more information, see the following:

- For information about configuring SSL for your Oracle database, including details about the settings required in the `sqlnet.ora` file, see "Configuring Secure Sockets Layer Authentication" in the *Oracle Database Security Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-secure-sockets-layer-authentication.html.
- For general information about the `sqlnet.ora` file, see "Parameters for the sqlnet.ora File" in the *Oracle Database Net Services Reference*: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/parameters-for-the-sqlnet.ora.html.

To enable TCPS on Windows:

1. Open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
2. Select the **Use TCPS** check box.
3. To save your settings and close the dialog box, click **OK**.

Configuring Logging Options on Windows

To help troubleshoot issues, you can enable logging. In addition to functionality provided in the Simba Oracle 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 Oracle 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 on page 18.

To enable connector-wide logging on 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. 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.

5. 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.

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

The Simba Oracle ODBC Connector produces the following log files at the location you specify in the Log Path field:

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

- A `simbaoracleodbcdriver_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 on page 68.

To disable connector logging on 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 on Windows on page 16. 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 on 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 Properties on page 52.
   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.

Verifying the Connector Version Number on Windows

If you need to verify the version of the Simba Oracle 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 on 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 Oracle.

2. Click the **Drivers** tab and then find the Simba Oracle 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

macOS System Requirements

The Simba Oracle ODBC Connector supports Oracle versions 11g, 12c, 18c, and 19c, as well as autonomous data sources.

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:

- macOS version 10.13 or 10.14
- 250 MB of available disk space; 730 MB of RAM
- One of the following ODBC driver managers installed:
  - iODBC 3.52.9 or later
  - unixODBC 2.2.14 or later

Before the connector can be used, the 64-bit edition of Oracle Instant Client 19.3 must be installed in the `/lib` subfolder in the connector's installation directory. If you obtained the connector from the Simba website, then your installation of the connector automatically includes this dependency. Otherwise, you must install Oracle Instant Client manually. For detailed instructions, see Installing the Oracle Instant Client on macOS on page 22.

Installing the Connector on 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*.

The Simba Oracle ODBC Connector is available for macOS as a `.dmg` file named `Simba Oracle 2.0.dmg`. The connector supports 64-bit client applications only.

To install the Simba Oracle ODBC Connector on macOS:

1. Double-click **Simba Oracle 2.0.dmg** to mount the disk image.
2. In the installer, click **Continue**.
3. 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.
4. 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/oracleodbc` directory.

5. To accept the installation location and begin the installation, click **Install**.
6. When the installation completes, click **Close**.
7. 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/oracleodbc/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 on Non-Windows Machines on page 29.

Installing the Oracle Instant Client on macOS

The Simba Oracle ODBC Connector requires the 64-bit edition of Oracle Instant Client 19.3. If you obtained the connector from the Simba website, then your installation of the connector automatically includes this dependency. Otherwise, you must manually install Oracle Instant Client in the `/lib` subfolder in the installation directory of the connector.

> **Note:**
>
> Oracle Instant Client can also be installed in `/usr/lib`, or any path that is added within the DYLD_LIBRARY_PATH environment variable.

To install the Oracle Instant Client on macOS:

1. In a web browser, navigate to https://www.oracle.com/database/technologies/instant-client/downloads.html.
2. Download the 64-bit edition of Oracle Instant Client version 19.3. You can use the Basic package or Basic Light package depending upon your disk space.
3. Extract the archive that you downloaded to a temporary location.
4. Copy the files from the temporary location to the `/lib` subfolder in the installation directory of the connector.

Verifying the Connector Version Number on macOS

If you need to verify the version of the Simba Oracle ODBC Connector that is installed on your macOS machine, you can query the version number through the Terminal.

To verify the connector version number on macOS:

- At the Terminal, run the following command:

```
pkgutil --info com.simba.oracleodbc
```

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

Linux Connector

The Linux connector is available as an RPM file and as a tarball package.

Linux System Requirements

The Simba Oracle ODBC Connector supports Oracle versions 11g, 12c, 18c, and 19c, 23c, as well as autonomous data sources.

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:

- One of the following distributions:
  - Red Hat® Enterprise Linux® (RHEL) 7
  - SUSE Linux Enterprise Server (SLES) 12
  - Ubuntu 22.04
- 270 MB of available disk space; 256 MB of RAM
- One of the following ODBC driver managers installed:
  - iODBC 3.52.9 or later
  - unixODBC 2.2.14 or later

Note:

These Linux distributions are officially supported by the Oracle Call Interface (OCI), which the Simba Oracle ODBC Connector uses as a dependency. Simba also tests the connector on other Linux distributions, but can only officially support the ones listed earlier in this topic.

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

If you are using the RPM file to install the connector on Debian or Ubuntu, you must also have the `alien` utility installed. The `alien` utility is available on SourceForge: https://sourceforge.net/projects/alien-pkg-convert/.

Before the connector can be used, Oracle Instant Client 19.3 (with the same bitness as the connector) must be installed in the `/lib` subfolder in the connector's installation directory. If you obtained the connector from the Simba website, then your installation of the connector automatically includes this dependency. Otherwise, you must install Oracle Instant Client manually. For detailed instructions, see Installing the Oracle Instant Client on Linux on page 27.

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:

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

The placeholders in the file names are defined as follows:

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

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

To install the Simba Oracle ODBC Connector using the RPM File:

1. Log in as the root user.
2. If you are installing the connector on a Debian or Ubuntu machine, download and install the `alien` utility:
   a. Download the package from SourceForge: https://sourceforge.net/projects/alien-pkg-convert/.
   b. From the command line, run the following command:

      ```
      sudo apt-get install alien
      ```
3. Navigate to the folder containing the RPM package for the connector.
4. Depending on the Linux distribution that you are using, run one of the following commands from the command line, where *[RPMFileName]* is the file name of the RPM package:
   - If you are using Red Hat Enterprise Linux or CentOS, run the following command:

     ```
     yum --nogpgcheck localinstall [RPMFileName]
     ```
   - Or, if you are using SUSE Linux Enterprise Server, run the following command:

```
zypper install [RPMFileName]
```

- Or, if you are using Debian or Ubuntu, run the following command:

```
alien -i [RPMFileName]
```

The Simba Oracle ODBC Connector files are installed in the `/opt/simba/oracleodbc` directory.

5. If you received a license file through email, then copy the license file into the `/opt/simba/oracleodbc/lib/32` or `/opt/simba/oracleodbc/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 on Non-Windows Machines on page 29.

Installing the Connector Using the Tarball Package

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*.

The Simba Oracle ODBC Connector is available as a tarball package named `SimbaOracleODBC-[Version].[Release]-Linux.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 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. You can install both versions of the connector on the same machine.

To install the connector using the tarball package:

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 Oracle ODBC Connector files are installed in the `/opt/simba/oracleodbc` directory.

3. If you received a license file through email, then copy the license file into the `opt/simba/oracleodbc/lib/32` or `opt/simba/oracleodbc/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 on Non-Windows Machines on page 29.

Installing the Oracle Instant Client on Linux

The Simba Oracle ODBC Connector requires Oracle Instant Client 19.3. If you obtained the connector from the Simba website, then your installation of the connector automatically includes this dependency. Otherwise, you must manually install Oracle Instant Client in the `/lib` subfolder in the installation directory of the connector.

Note:

Oracle Instant Client can also be installed in `/usr/lib`, or any path that is added within the LD_LIBRARY_PATH environment variable.

To install the Oracle Instant Client on Linux:

1. In a web browser, navigate to https://www.oracle.com/database/technologies/instant-client/downloads.html.
2. Download the 19.3 version of the Oracle Instant Client that matches the bitness of your platform. You can use the Basic package or Basic Light package depending upon your disk space.
3. Extract the archive that you downloaded to a temporary location.
4. Copy the files from the temporary location to the `/lib` subfolder in the installation directory of the connector.

Verifying the Connector Version Number on Linux

If you need to verify the version of the Simba Oracle 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.

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

- Depending on your package manager, at the command prompt, run one of the following commands:
  - `yum list | grep SimbaOracleODBC`
  - `rpm -qa | grep SimbaOracleODBC`

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

To verify the connector version number on 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/oracleodbc/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.

Configuring the ODBC Driver Manager on Non-Windows Machines

To make sure that the ODBC Driver manager on your machine is configured to work with the Simba Oracle 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 on Non-Windows Machines on page 29.
- 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 on page 30.

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

Specifying ODBC Driver Managers on 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

If you are using a Linux 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 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.oracleodbc.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 _ORACLE_ODBC_INI to the full path and file name of the `simba.oracleodbc.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 _ORACLE_ODBC_INI to the full path and file name of the `simba.oracleodbc.ini` file.

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

For unixODBC:

```
export ODBCINI=/usr/local/odbc/odbc.ini
export ODBCSYSINI=/usr/local/odbc
export _ORACLE_ODBC_INI=/etc/simba.oracleodbc.ini
```

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

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

Configuring ODBC Connections on a Non-Windows Machine

The following sections describe how to configure ODBC connections when using the Simba Oracle ODBC Connector on non-Windows platforms:

Creating a Data Source Name on a Non-Windows Machine

When connecting to your data store using a DSN, you only need to configure the `odbc.ini` file. Set the properties in the `odbc.ini` file to create a DSN that specifies the connection information for your data store. For information about configuring a DSN-less connection instead, see Configuring a DSN-less Connection on a Non-Windows Machine on page 36.

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.

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 Oracle ODBC Connector
```

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

```
[ODBC Data Sources]
Sample DSN=Simba Oracle ODBC Connector 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/oracleodbc/lib/liboracleodbc_
      sb64.dylib
      ```

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

      ```
      Driver=/opt/simba/oracleodbc/lib/32/liboracleodbc_
      sb32.so
      ```

   b. To specify the Oracle database that you want to connect to, do one of the following:
      - To use server information that is defined in your `tnsnames.ora` configuration file, do the following:
        i. Set the `TNS` property to the net service name that you want to use.
        ii. On your Linux or macOS machine, update the TNS_ADMIN environment variable to point to the path of the `tnsnames.ora` file.

        For example:

        ```
        TNS=oracleconnection1
        ```

      - Or, to use simultaneous cloud connections, specify the path to an Oracle Cloud Wallet configuration file or folder by doing the following:
        i. Extract cloud wallets within separate locations. For example, `[Wallet1Dir]` and `[Wallet2Dir]`.
        ii. Copy the `tnsnames.ora` contents of each wallet and paste them within your `tnsnames.ora` file (located in TNS_ADMIN).

iii. Add the respective wallet locations under the security of the respective TNS alias:

```
NameOfTNSEntry =
(DESCRIPTION=
(ADDRESS_LIST =
(ADDRESS=(PROTOCOL=tcps)
(HOST=xyz.somewhere.c0m)(PORT=12345678))
)
(CONNECT_DATA=(SERVICE_NAME=NameOfService))
(SECURITY =
(MY_WALLET_DIRECTORY = [Wallet1Dir])
)
)
```

iv. Establish the connection using TNS.

Note:

For more information about multiple wallet connections, see "One client needing multiple wallets and/or multiple SQLNET.ORA files - (Solved/workaround)" in the *Oracle Applications and Infrastructure Community*: https://community.oracle.com/tech/apps-infra/discussion/4301296/one-client-needing-multiple-wallets-and-or-multiple-sqlnet-ora-files-solved-workaround.

- Or, to specify server information directly in the DSN, do the following:
  i. Set the `Host` property to the IP address or host name of the Oracle server.
  ii. Set the `Port` property to the number of the TCP port that the server uses to listen for client connections.
  iii. Set the SVC property to the service name of the Oracle database that you want to access.

  For example:

```
Host=192.168.222.160
Port=1521
SVC=ORCL
```

Note:

If you specify `TNS` in addition to `Host`, `Port`, and `SVC`, the `TNS` setting takes precedence and the connector connects using the server information defined in the `tnsnames.ora` configuration file.

c. Configure authentication as needed. For more information, see Configuring Authentication on a Non-Windows Machine on page 38.

d. Configure SSL encryption as needed. For more information, see Configuring SSL Verification on a Non-Windows Machine on page 40.

e. 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 Oracle ODBC Connector, see Connector Configuration Properties on page 52.

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 on page 30.

For example, the following is an `odbc.ini` configuration file for macOS containing a DSN that connects to Oracle, with statement caching enabled and the number of statements in the statement cache set to 20. Also, the connector uses 1000000 bytes as the buffer size for data retrieval:

```
[ODBC Data Sources]
Sample DSN=Simba Oracle ODBC Connector
[Sample DSN]
Driver=/Library/simba/oracleodbc/lib/liboracleodbc_sb64.dylib
Host=192.168.222.160
Port=1521
SVC=ORCL
UID=jsmith
PWD=simba123
ENABLESTMTCACHE=true
```

```
STMTCACHESIZE=20
MEMLIM=1000000
```

As another example, the following is an `odbc.ini` configuration file for a 32-bit connector on a Linux machine, containing a DSN that connects to Oracle, with statement caching enabled and the number of statements in the statement cache set to 20. Also, the connector uses 1000000 bytes as the buffer size for data retrieval:

```
[ODBC Data Sources]
Sample DSN=Simba Oracle ODBC Connector 32-bit
[Sample DSN]
Driver=/opt/simba/oracleodbc/lib/32/liboracleodbc_sb32.so
Host=192.168.222.160
Port=1521
SVC=ORCL
UID=jsmith
PWD=simba123
ENABLESTMTCACHE=true
STMTCACHESIZE=20
MEMLIM=1000000
```

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

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

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

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

To define a connector on a non-Windows machine:

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

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

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

   For example:

   ```
   [ODBC Drivers]
   Simba Oracle 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/oracleodbc/lib/liboracleodbc_
      sb64.dylib
      ```

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

      ```
      Driver=/opt/simba/oracleodbc/lib/32/liboracleodbc_
      sb32.so
      ```

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

      For example:

      ```
      Description=Simba Oracle ODBC Connector
      ```

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 on page 30.

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

```
[ODBC Drivers]
```

```
Description= Simba Oracle ODBC Connector
Driver=/Library/simba/oracleodbc/lib/liboracleodbc_sb64.dylib
```

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

```
[ODBC Drivers]
Description= Oracle ODBC Connector(32 bit)
Driver=/opt/simba/oracleodbc/lib/32/liboracleodbc_sb32.so
Description= Oracle ODBC Connector(64 bit)
Driver=/opt/simba/oracleodbc/lib/64/liboracleodbc_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 on page 46.

For instructions about configuring specific connection features, see the following:

- Configuring Authentication on a Non-Windows Machine on page 38
- Configuring SSL Verification on a Non-Windows Machine on page 40

For detailed information about all the connection properties that the connector supports, see Connector Configuration Properties on page 52.

Configuring Authentication on a Non-Windows Machine

All Oracle databases require authentication. You can configure the Simba Oracle ODBC Connector to provide your credentials and authenticate the connection to the database using one of the following methods:

- Using Your Oracle Database Credentials on page 39
- Using Kerberos on page 39
- Using JWT on page 39

Important:

You cannot use the Simba Oracle ODBC Connector to connect to an Oracle Autonomous Data Warehouse instance using Kerberos authentication.

You can set the connection properties described below in a connection string or in a DSN (in the `odbc.ini` file). Settings in the connection string take precedence over settings in the DSN.

Using Your Oracle Database Credentials

You can configure the connector to authenticate the connection using your database credentials.

To configure authentication using your database credentials on a non-Windows machine:

1. Set the `UseExternalCredentials` property to `false`.
2. Set the `UID` property to your user name for accessing the Oracle database.
3. Set the `PWD` property to the password corresponding to the user name you typed above.

Using Kerberos

You can configure the connector to use the Kerberos protocol to authenticate the connection. The connector retrieves and uses a Kerberos ticket based on the settings in the `sqlnet.ora` configuration file.

Before you can use this authentication mechanism, you must specify the necessary Kerberos settings in the `sqlnet.ora` file, and set the TNS_ADMIN environment variable on your machine to point to the path of the `sqlnet.ora` file. For more information, see the following:

- For information about configuring Kerberos for your Oracle database, including details about the settings required in the `sqlnet.ora` file, see "Configuring Kerberos Authentication" in the *Oracle Database Security Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-kerberos-authentication.html.
- For general information about the `sqlnet.ora` file, see "Parameters for the sqlnet.ora File" in the *Oracle Database Net Services Reference*: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/parameters-for-the-sqlnet.ora.html.

To configure Kerberos authentication on a non-Windows machine:

- Set the `UseExternalCredentials` property to `true`.

Using JWT

To configure the connector to use JWT authentication, you can set the connection properties described below in a connection string.

To configure JWT authentication on a non-Windows machine:

1. Set the `Authenticator` property to `OAuth`.
2. Set the `Token` property to the JWT token that you have generated.
3. Set the `USETCPS` property to `true`, if TCPS is required for the authentication.
4. Set the `SSL_SERVER_DN_MATCH` property to `true`, if SSL Server DN should match as client.
5. Set the `SSL_SERVER_CERT_DN` property to DN certificate of the server.

> **Note:**
> - When `Authenticator` property is not set to `OAuth`, Steps 2 to 5 are optional.
> - The `JWT` authentication method is applicable only on Windows and Linux.

Configuring SSL Verification on a Non-Windows Machine

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

The procedure for configuring SSL in your connection differs depending on whether or not you are connecting through TNS. For more information, see below:

- Using TNS on page 40
- Without Using TNS on page 41

Using TNS

If you have configured the connector to connect using server information that is defined in a `tnsnames.ora` configuration file, then you must make sure that the necessary SSL settings are specified in the `tnsnames.ora` file. For more information, see the following:

- For information about configuring SSL for your Oracle database, including details about the settings required in the `tnsnames.ora` file, see "Configuring Secure Sockets Layer Authentication" in the *Oracle Database Security Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-secure-sockets-layer-authentication.html.
- For general information about the `tnsnames.ora` configuration file, see "Local Naming Parameters in the tnsnames.ora File" in the *Oracle Database Net*

Services Reference: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/parameters-for-the-sqlnet.ora.html.

Without Using TNS

If you have specified your server information directly in a DSN or connection string instead of using the `tnsnames.ora` configuration file, then you must configure the connector to use the TCPS protocol, which enables SSL encryption on a TCP/IP connection. The connector then encrypts the connection using the SSL settings defined in the `sqlnet.ora` configuration file.

Before configuring the connector to use TCPS, you must specify the necessary SSL settings in the `sqlnet.ora` file, and set the TNS_ADMIN environment variable on your machine to point to the path of the `sqlnet.ora` file. For more information, see the following:

- For information about configuring SSL for your Oracle database, including details about the settings required in the `sqlnet.ora` file, see "Configuring Secure Sockets Layer Authentication" in the *Oracle Database Security Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-secure-sockets-layer-authentication.html.
- For general information about the `sqlnet.ora` file, see "Parameters for the sqlnet.ora File" in the *Oracle Database Net Services Reference*: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/parameters-for-the-sqlnet.ora.html.

To enable TCPS on a non-Windows machine:

- Set the `UseTCPS` property to `true`.

Configuring Logging Options on 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.oracleodbc.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.oracleodbc.ini` configuration file.
7. Restart your ODBC application to make sure that the new settings take effect.

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

- A `simbaoracleodbcdriver.log` file that logs connector activity that is not specific to a connection.
- A `simbaoracleodbcdriver_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 on page 68.

To disable logging on a non-Windows machine:

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

Testing the Connection on a 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 on page 46.

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.

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 Properties on page 52.

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:

- *[DBService]* is service name of the database that you want to access.
- *[PortNumber]* is the number of the TCP port that the Oracle server uses to listen for client connections.
- *[Server]* is the IP address or host name of the Oracle server to which you are connecting.
- *[TNSName]* is the net service name from your `tnsnames.ora` file that you want to use for your connection.
- *[YourPassword]* is the password corresponding to your user name.
- *[YourUserName]* is the user name that you use to access the Oracle server.

Connecting to Oracle Using Your Oracle Database Credentials

The following is the format of a DSN-less connection string for connecting to Oracle using your database credentials:

```
Driver=Simba Oracle ODBC Connector;Host=[Server];
Port=[PortNumber];SVC=[DBService];UID=[YourUserName];
PWD=[YourPassword];
```

For example:

```
Driver=Simba Oracle ODBC Connector;Host=192.168.222.160;
Port=1521;SVC=ORCL;UID=jsmith;PWD=simba123;
```

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

```
Driver=Simba Oracle ODBC Connector;Host=192.168.222.160;
Port=2484;SVC=ORCL;UID=jsmith;PWD=simba123;
UseTCPS=true;
```

Connecting to Oracle Using Kerberos

The following is the format of a DSN-less connection string for connecting to Oracle using the Kerberos protocol:

```
Driver=Simba Oracle ODBC Connector;Host=[Server];
Port=[PortNumber];SVC=
[DBService];UseExternalCredentials=true;
```

For example:

```
Driver=Simba Oracle ODBC Connector;Host=192.168.222.160;
Port=1521;SVC=ORCL;UseExternalCredentials=true;
```

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

```
Driver=Simba Oracle ODBC Connector;Host=192.168.222.160;
Port=2484;SVC=ORCL;UseExternalCredentials=true;
UseTCPS=true;
```

Important:

You cannot use the Simba Oracle ODBC Connector to connect to an Oracle Autonomous Data Warehouse instance using Kerberos authentication.

Connecting to Oracle Through TNS

The following is the format of a DSN-less connection string for connecting to a Oracle through TNS. In this example, the connector authenticates the connection using Oracle database credentials; however, you can configure the connector to authenticate through Kerberos instead, as shown in the examples above.

```
Driver=Simba Oracle ODBC Driver;TNS=[TNSName];
UID=[YourUserName];PWD=[YourPassword];
```

For example:

```
Driver=Simba Oracle ODBC Driver;TNS=oracleconnection1;
UID=jsmith;PWD=simba123;
```

If you are connecting to the server through SSL, make sure that the TNS_ADMIN environment variable on your machine points to a `tnsnames.ora` configuration file that contains the necessary SSL settings. For more information, see Configuring SSL Verification on a Windows Machine on page 14 or Configuring SSL Verification on a Non-Windows Machine on page 40.

Features

The Simba Oracle ODBC Connector supports the following features:

Data Types

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

The table below lists the supported data type mappings.

Oracle Type	SQL Type
BFILE	SQL_LONGVARBINARY
BINARY_DOUBLE	SQL_DOUBLE
BINARY_FLOAT	SQL_REAL
BLOB	SQL_LONGVARBINARY
CHAR	SQL_CHAR
CLOB	SQL_LONGVARCHAR
DATE	SQL_TYPE_TIMESTAMP
DECIMAL	SQL_DECIMAL
DOUBLE_PRECISION	SQL_DOUBLE
FLOAT	SQL_FLOAT
INTEGER	SQL_DECIMAL
INTERVAL_DAY_TO_SECOND	SQL_INTERVAL_DAY_TO_SECOND
INTERVAL_YEAR_TO_MONTH	SQL_INTERVAL_YEAR_TO_MONTH

Oracle Type	SQL Type
NCHAR	SQL_WCHAR
NCLOB	SQL_WLONGVARCHAR
NUMBER	SQL_DECIMAL
NUMBER([1-38])	SQL_DECIMAL
NUMBER([1-38], [0-38])	SQL_DECIMAL
NVARCHAR2	SQL_WVARCHAR
RAW	SQL_VARBINARY
REAL	SQL_DOUBLE
ROWID	SQL_WCHAR
TIMESTAMP	SQL_TYPE_TIMESTAMP
TIMESTAMP_ WITH_LOCAL_TIME_ ZONE	SQL_TYPE_TIMESTAMP
TIMESTAMP_WITH_TIME_ZONE	SQL_TYPE_TIMESTAMP
UROWID	SQL_WCHAR
VARCHAR	SQL_VARCHAR
VARCHAR2	SQL_VARCHAR

Security and Authentication

To protect data from unauthorized access, Oracle data stores require connections to be authenticated with user credentials and sometimes the SSL protocol. The Simba Oracle 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 the same SSL/TLS versions as Oracle Call Interface (OCI) 19.3.

The connector provides a mechanism that enables you to authenticate your connection using your Oracle database credentials or the Kerberos protocol. Authentication through Kerberos requires you to provide a `sqlnet.ora` configuration file that contains the necessary Kerberos settings. For detailed configuration instructions, see Configuring Authentication on Windows on page 12 or Configuring Authentication on a Non-Windows Machine on page 38.

Additionally, the connector supports SSL encryption with identity verification. For information about SSL support, see "SSL Cipher Suite Authentication, Encryption, Integrity, and TLS Versions" in the *Oracle Database Security Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-secure-sockets-layer-authentication.html#GUID-EFF4B2C9-2D25-473D-B718-A42754252347.

Note:

If you try to establish an SSL connection to a server that is using an earlier version of OCI, the connection might fail due to differences in the supported SSL features.

If you are connecting through TNS, then your SSL settings must be specified in your `tnsnames.ora` configuration file. Otherwise, you must enable TCPS support in the connector and then provide a `sqlnet.ora` configuration file that contains the necessary SSL settings. For detailed configuration instructions, see Configuring SSL Verification on a Windows Machine on page 14 or Configuring SSL Verification on a Non-Windows Machine on page 40.

It is recommended that you use 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.

Connector Configuration Properties

Connector Configuration Options lists the configuration options available in the Simba Oracle 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 described below are available in the following dialog boxes:

- Simba Oracle ODBC Driver DSN Setup
- Logging Options

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

Configuration Options Appearing in the User Interface

The following configuration options are accessible via the Windows user interface for the Simba Oracle ODBC Connector, or via the key name when using a connection string or configuring a connection from a Linux or macOS machine:

Auto IPD

Key Name	Default Value	Required
`AutoIPD`	Enabled (`true`)	No

Description

This option indicates whether automatic implementation parameter database (IPD) population is enabled or not.

- Enabled (`true`): The connector automatically populates the IPD.
- Disabled (`false`): The connector does not automatically populate the IPD.

Catalog Details

Key Name	Default Value	Required
`CatalogDetails`	`AllTables`	No

Description

This option indicates whether the catalog functions only return the details of user-defined tables.

- `AllTables`: The catalog functions return the details of all the tables.
- `UserDefinedTables`: The catalog functions return the details of user-defined tables, and no information about other tables.

Enable Statement Caching

Key Name	Default Value	Required
`EnableStmtCache`	Clear (`false`)	No

Description

This option indicates whether statement caching is enabled or not.

- Enabled (`true`): The connector caches statements, increasing performance for parsing the same statements multiple times in the same connection. Be aware that the connector uses more memory when statement caching is enabled.
- Disabled (`false`): The connector does not cache statements. When statement caching is disabled, the connector uses less memory.

The default size of the cache is 20 statements. For more information about the cache size, see Statement Cache Size on page 60.

Enable Table Types

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

Description

This option specifies whether the connector recognizes table type information from the data source. By default, the connector only recognizes a single, generic table type.

- Enabled (`1`): The connector recognizes the following table types: TABLE, SYSTEM TABLE, and GLOBAL TEMPORARY.
- Disabled (`0`): All tables returned from the data source have the generic type TABLE.

Fetch Buffer Size

Key Name	Default Value	Required
`MEMLIM`	`104857600` (100 MB)	No

Description

The size of the buffer that the connector uses for data retrieval, in bytes. The minimum value for the buffer size is `32000` (32 KB).

This property determines the maximum number of rows that the connector can retrieve each time during array fetches. The maximum number of rows is calculated using the `MEMLIM` value and the maximum size of one row.

Note:

To confirm the number of rows that the connector retrieves at a time based on your `MEMLIM` setting, enable connector logging on the `DEBUG` level and then run a query. The log file includes information about the number of rows per fetch relative to the `MEMLIM` setting.

For information about configuring logging when using the Windows connector, see Configuring Logging Options on Windows on page 16.

Force SQL_WCHAR support

Key Name	Default Value	Required
`ForceSQLWCHARSupport`	Clear (`false`)	No

Description

This option specifies whether the varchar type is retrieved as unicode or wide characters.

- Enabled (`true`): The connector reads and displays varchar type as unicode or wide characters.
- Disabled (`false`): The connector does not read and display varchar type correctly.

Host

Key Name	Default Value	Required
`Host`	None	

Description

The IP address or host name of the Oracle server.

Note:

If you are connecting using a connection string or from a non-Windows machine, and the `TNS` property is set, the connector uses the server information defined in the specified net service name instead of this value.

Log Level

Key Name	Default Value	Required
LogLevel	OFF (0)	No

Description

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 simbaoracleodbcdriver.log file that logs connector activity that is not specific to a connection.
- A simbaoracleodbcdriver_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 on page 68.

Log Path

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

Description

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.

Max File Size

Key Name	Default Value	Required
LogFileSize	20971520	No

Description

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.

Max Number Files

Key Name	Default Value	Required
LogFileCount	50	No

Description

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.

Password

Key Name	Default Value	Required
`PWD`	None	Yes

Description

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

Port

Key Name	Default Value	Required
`Port`	None	Yes, unless connecting through TNS.

Description

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

> **Note:**
>
> If you are connecting using a connection string or from a non-Windows machine, and the `TNS` property is set, the connector uses the server information defined in the specified net service name instead of this value.

Query Timeout

Key Name	Default Value	Required
QueryTimeout	0	No

Description

The length of time, in seconds, that the connector waits for a round-trip call between the client and the server to complete before timing out and returning an error.

A value of 0 or -1 indicates that the round-trip timeout is disabled, and the client-server calls do not time out.

When this option is set to a value from 1 to 2,147,483,647 (both inclusive), the connector specifies the round-trip timeout using the OCI_ATTR_CALL_TIMEOUT attribute. For information about this attribute, see “Service Context Handle Attributes” in the *Oracle Call Interface Programmer’s Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/lnoci/handle-and-descriptor-attributes.html#GUID-CB59C987-07E7-42D4-ADDF-96142CBD3D11.

Note:

Be aware that, if you run a large query while round-trip timeout is enabled, your connection might time out.

Service Name

Key Name	Default Value	Required
SVC	None	Yes, unless connecting through TNS.

Description

The service name of the database.

Note:

If you are connecting using a connection string or from a non-Windows machine, and the TNS property is set, the connector uses the service name specified through that setting instead of this value.

Statement Cache Size

Key Name	Default Value	Required
StmtCacheSize	20	No

Description

This option indicates the number of statements that the statement cache can contain. Even though the Simba Oracle ODBC Connector does not place any restriction on the size of the statement cache, you must be aware of the maximum number of open cursors allowed in your data source. For example, if the cache size is set to a value higher than the maximum number of open cursors allowed in your data source, you may see an error such as the following:
`ORA-01000: maximum open cursors exceeded`

We recommend that you turn on auto-tuning with the connection-specific parameters such as `prefetch` and `statement_cache` for performance improvements. For more information, see the following:

- For information about OCI auto-tuning, see "Client Statement Cache Auto-Tuning" in the *Oracle Call Interface Programmer's Guide* :https://docs.oracle.com/database/121/LNOCI/oci10new.htm#LNOCI73051.
- For information about `oraaccess.xml`, see "OCI Client-Side Deployment Parameters Using oraaccess.xml" in the *Oracle Call Interface Programmer's Guide*: https://docs.oracle.com/database/121/LNOCI/oci10new.htm#LNOCI-GUID-CD599644-135A-4116-8B3B-40A9BA172E5C.
- For information about fetching results with OCI, see "About Fetching Results" in the *Oracle Call Interface Programmer's Guide*: https://docs.oracle.com/database/121/LNOCI/oci04sql.htm#LNOCI1635.

TNS Name

Key Name	Default Value	Required
TNS	None	Yes, if connecting through TNS.

Description

The net service name from your `tnsnames.ora` file that you want to use for your connection. Set this property when you want to connect to Oracle using server

information that is defined in your `tnsnames.ora` file, instead of specifying server information directly in a DSN or connection string.

For more information about the `tnsnames.ora` configuration file, see "Local Naming Parameters in the tnsnames.ora File" in the *Oracle Database Net Services Reference*: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/local-naming-parameters-in-tns-ora-file.html.

Use Dynamic Bind Buffer

Key Name	Default Value	Required
`UseDynamicBindBuffer`	Disabled (`false`)	No

Description

This option specifies how the connector determines the bind buffer length of a parameter.

- Disabled (`false`): The connector uses column size to determine the bind buffer length.
- Enabled (`true`): The connector determines bind buffer length dynamically.

Use External Credentials

Key Name	Default Value	Required
`UseExternalCredentials`	Clear (`false`)	No

Description

This option specifies whether the connector uses the Kerberos protocol to authenticate the connection.

- Enabled (`true`): The connector uses the Kerberos protocol to authenticate the connection. The connector retrieves and uses a Kerberos ticket based on the settings in the `sqlnet.ora` configuration file. The `sqlnet.ora` configration file must be specified by the TNS_ADMIN environment variable on your machine.

Note:

- For information about configuring Kerberos on your Oracle database, including details about the settings required in the `sqlnet.ora` file, see "Configuring Kerberos Authentication" in the *Oracle Database Security Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-kerberos-authentication.html.
- For general information about the `sqlnet.ora` file, see "Parameters for the sqlnet.ora File" in the *Oracle Database Net Services Reference*: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/parameters-for-the-sqlnet.ora.html.

- Disabled (`false`): The connector does not use the Kerberos protocol.

Use TCPS

Key Name	Default Value	Required
`UseTCPS`	Clear (`false`)	No

Description

This option specifies whether the connector connects to Oracle over the TCPS protocol, which provides SSL/TLS encryption on a TCP/IP connection.

- Enabled (`true`): The connector connects over the TCPS protocol, encrypting the connection using the SSL settings defined in the `sqlnet.ora` configuration file. The `sqlnet.ora` file must be specified by the TNS_ADMIN environment variable on your machine.

Note:

- For information about configuring SSL on your Oracle database, including details about the settings required in the `sqlnet.ora` file, see "Configuring Secure Sockets Layer Authentication" in the *Oracle Database Security Guide*: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-secure-sockets-layer-authentication.html.
- For general information about the `sqlnet.ora` file, see "Parameters for the sqlnet.ora File" in the *Oracle Database Net Services Reference*: https://docs.oracle.com/en/database/oracle/oracle-database/19/netrf/parameters-for-the-sqlnet-ora-file.html.

- Disabled (`false`): The connector does not use the TCPS protocol.

User

Key Name	Default Value	Required
`UID`	None	Yes

Description

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

Configuration Options Having Only Key Names

The following configuration options do not appear in the Windows user interface for the Simba Oracle ODBC Connector. They are accessible only when you use a connection string or configure a connection on macOS or Linux.

The following configuration options do not appear in the Windows user interface for the Simba Oracle ODBC Connector. They are accessible only when you use a connection string.

- MaxColumnNameLen on page 66
- MaxSchemaNameLen on page 66
- MaxTableNameLen on page 67
- SSLServerCertDN on page 67
- SSLServerDNMatch on page 67
- Token on page 67
- UseTCPS on page 69

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

- UseLogPrefix on page 68

Authenticator

Key Name	Default Value	Required
`OAuth`	None	No

Description

This option specifies the type of authentication used by the Simba Oracle ODBC Connector.

Driver

Key Name	Default Value	Required
`Driver`	`Simba Oracle ODBC Driver` when installed on Windows, or the absolute path of the connector shared object file when installed on a non-Windows machine.	Yes

Description

On Windows, the name of the installed connector (`Simba Oracle 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.

DriverLocale

Key Name	Default Value	Required
DriverLocale	en-US	No

Description

The locale to use for error messages.

This is a connector-wide setting, and cannot be specified in a DSN or connection string. To configure the locale on a per-session basis, see Locale on page 65.

If both `Locale` and `DriverLocale` are specified, `Locale` takes precedence.

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 Oracle ODBC Connector\Driver**
- Otherwise: **HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Oracle ODBC Connector\Driver**

Use `DriverLocale` as the value name, and the locale as the value data.

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

Locale

Key Name	Default Value	Required
Locale	en-US	No

Description

The locale to use for error messages.

If both `Locale` and `DriverLocale` are specified, `Locale` takes precedence.

MaxCatalogNameLen

Key Name	Default Value	Required
MaxCatalogNameLen	0	No

Description

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`.

MaxColumnNameLen

Key Name	Default Value	Required
MaxColumnNameLen	30	No

Description

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`.

MaxSchemaNameLen

Key Name	Default Value	Required
MaxSchemaNameLen	30	No

Description

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`.

MaxTableNameLen

Key Name	Default Value	Required
MaxTableNameLen	30	No

Description

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.

SSLServerCertDN

Key Name	Default Value	Required
SSL_SERVER_CERT_DN	None	No

Description

This option specifies the DN certificate of the server.

SSLServerDNMatch

Key Name	Default Value	Required
SSL_SERVER_DN_MATCH	None	No

Description

This option specifies whether the SSL server DN should match as client value.

Token

Key Name	Default Value	Required
Token	None	No

Description

This option specifies whether the JWT token is generated for authentication.

UseLogPrefix

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

Description

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_simbaodbcdriver.log` and `jdoe_7836_simbaodbcdriver_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 Oracle ODBC Connector\Driver**
- Otherwise: **HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Oracle ODBC Connector\Driver**
- For a 32-bit connector installed on a 64-bit machine: **HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Simba\\Driver**
- Otherwise: **HKEY_LOCAL_MACHINE\SOFTWARE\\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.oracleodbc.ini` file.

UseTCPS

Key Name	Default Value	Required
USETCPS	None	No

Description

This option specifies whether the TCPS is required for authentication.

Third-Party Trademarks

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 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.

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

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.

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

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

All other trademarks are trademarks of their respective owners.