



Magnitude Simba Snowflake ODBC Data Connector

Installation and Configuration Guide

Version 2.23.2

May 31, 2021

Copyright © 2021 Magnitude Software, Inc. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without prior written permission from Magnitude.

The information in this document is subject to change without notice. Magnitude strives to keep this information accurate but does not warrant that this document is error-free.

Any Magnitude product described herein is licensed exclusively subject to the conditions set forth in your Magnitude license agreement.

Simba, the Simba logo, SimbaEngine, and Simba Technologies are registered trademarks of Simba Technologies Inc. in Canada, the United States and/or other countries. All other trademarks and/or servicemarks are the property of their respective owners.

All other company and product names mentioned herein are used for identification purposes only and may be trademarks or registered trademarks of their respective owners.

Information about the third-party products is contained in a third-party-licenses.txt file that is packaged with the software.

Contact Us

Magnitude Software, Inc.

www.magnitude.com

About This Guide

Purpose

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

Knowledge Prerequisites

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

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

Table of Contents

About the Simba Snowflake ODBC Connector	7
Windows Connector	8
Windows System Requirements	8
Installing the Connector on Windows	8
Creating a Data Source Name on Windows	9
Setting Connector-Wide Configuration Options on Windows	10
Configuring Authentication on Windows	11
Configuring Logging Options on Windows	13
Verifying the Connector Version Number on Windows	20
macOS Connector	21
macOS System Requirements	21
Installing the Connector on macOS	21
Verifying the Connector Version Number on macOS	22
Linux Connector	23
Linux System Requirements	23
Installing the Connector Using the Tarball Package	23
Verifying the Connector Version Number on Linux	24
Configuring the ODBC Driver Manager on Non-Windows Machines	25
Specifying ODBC Driver Managers on Non-Windows Machines	25
Specifying the Locations of the Connector Configuration Files	26
Configuring ODBC Connections on a Non-Windows Machine	28
Creating a Data Source Name on a Non-Windows Machine	28
Setting Connector-Wide Configuration Options on a Non-Windows Machine	31
Configuring a DSN-less Connection on a Non-Windows Machine	32
Configuring Authentication on a Non-Windows Machine	34
Configuring Logging Options on a Non-Windows Machine	35
Testing the Connection on a Non-Windows Machine	40
Using a Connection String	42
DSN Connection String Example	42
DSN-less Connection String Examples	42
Features	45
Data Types	45

Installation and Configuration Guide

Snowflake-Specific DML Commands	49
Snowflake-Specific SQL Attributes	52
Multiple Statements	53
Proxy Server	53
Security and Authentication	54
Connector Configuration Options	56
Configuration Options Appearing in the User Interface	56
Configuration Options Having Only Key Names	62
Third-Party Trademarks	82

About the Simba Snowflake ODBC Connector

Snowflake is a cloud-based data storage and analytics service. The Simba Snowflake ODBC Connector is used for direct SQL access to Snowflake data stores, enabling Business Intelligence (BI), analytics, and reporting on Snowflake-based data. The connector supports standard SQL as well as the Snowflake-specific PUT and GET statement syntax.

The Simba Snowflake ODBC Connector complies with the ODBC 3.80 data standard and adds important functionality such as Unicode and 32- and 64-bit support for high-performance computing environments.

ODBC is one of the most established and widely supported APIs for connecting to and working with databases. At the heart of the technology is the ODBC connector, which connects an application to the database. For more information about ODBC, see *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>.

For more information about the Snowflake ODBC Driver, see *ODBC Driver* from the Snowflake documentation: <https://docs.snowflake.com/en/user-guide/odbc.html>.

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

Note:

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

Windows Connector

Windows System Requirements

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

- Administrator rights on your machine.
- A machine that meets the following system requirements:
 - One of the following operating systems:
 - Windows 10, 8.1, or 7 SP1
 - Windows Server 2016, 2012, or 2008 R2 SP1
 - 150 MB of available disk space

Before the connector can be used, the Visual C++ Redistributable for Visual Studio 2015 with the same bitness as the connector must also be installed. If you obtained the connector from the Simba website, then your installation of the connector automatically includes this dependency. Otherwise, you must install the redistributable manually. You can download the installation packages for the redistributable at <https://www.microsoft.com/en-ca/download/details.aspx?id=48145>.

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- 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 Snowflake 2.23 32-bit.msi` for 32-bit applications
- `Simba Snowflake 2.23 64-bit.msi` for 64-bit applications

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

To install the Simba Snowflake ODBC Connector on Windows:

1. Depending on the bitness of your client application, double-click to run **Simba Snowflake 2.23 32-bit.msi** or **Simba Snowflake 2.23 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**.

Creating a Data Source Name on Windows

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

Alternatively, you can specify connection settings in a connection string. Settings in the connection string take precedence over settings in the DSN.

The following instructions describe how to create a DSN. For information about specifying settings in a connection string, see [Using a Connection String](#) on page 42.

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 Snowflake.

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

 **Note:**

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

4. Click **Add**.
5. In the Create New Data Source dialog box, select **Simba Snowflake ODBC Connector** and then click **Finish**. The **Snowflake Configuration** dialog box opens.

6. Unless you are using OAuth 2.0 to authenticate your connection, in the **User** field, type the user name that you are using to connect to the data store.
7. In the **Server** field, type the host name or IP address of the Snowflake server.
8. Optionally, in the **Database** field, type the name of the database to use when a database is not specified in a query.
9. Optionally, in the **Schema** field, type the name of the schema to use when a schema is not specified in a query.
10. Optionally, in the **Warehouse** field, type the name of the warehouse to use when a warehouse is not specified during a session.
11. Optionally, in the **Role** field, type the name of the role to use when a role is not specified during a session.
12. Optionally, to configure the connector to connect to Snowflake through a proxy server:

- a. In the **Proxy** field, type the URL for the proxy server, in one of the following formats:

```
http://[hostname]:[portnumber]/
```

```
[hostname]:[portnumber]
```

- b. Optionally, in the **NoProxy** field, type a comma-separated list of host name endings that are allowed to bypass the proxy server.

 **Note:**

- The NoProxy field does not support wildcard characters.
- The proxy server settings can also be configured as connector-wide settings. For more information, see [Proxy](#) on page 58 and [NoProxy](#) on page 57.

13. To authenticate your credentials using an authenticator other than the internal Snowflake authenticator, see [Configuring Authentication on Windows](#) on page 11.
14. To configure logging behavior for the connector, see [Configuring Event Tracing on Windows](#) on page 13.
15. To save your settings and close the Snowflake Configuration dialog box, click **OK**.
16. To close the ODBC Data Source Administrator, click **OK**.

Setting Connector-Wide Configuration Options on Windows

When you specify connection settings in a DSN or connection string, those settings apply only when you connect to Snowflake using that particular DSN or string. As an alternative, you can specify certain settings that apply to every connection that uses the

Simba Snowflake ODBC Connector by configuring them in 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 set connector-wide configuration options on Windows:

1. Choose one:
 - If you are using Windows 7 or earlier, click **Start** , then type **regedit** in the **Search** field, and then click **regedit.exe** in the search results.
 - Or, if you are using Windows 8 or later, on the Start screen, type **regedit**, and then click the **regedit** search result.
2. Navigate to the appropriate registry key for the bitness of your connector and your machine:
 - If you are using the 32-bit connector on a 64-bit machine, then browse to the following registry key:
`HKEY_LOCAL_MACHINE\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Driver\Driver`
 - Otherwise, browse to the following registry key:
`HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Driver\Driver`
3. For each configuration option that you want to configure as a connector-wide setting, create a value by doing the following:
 - a. Right-click the **Driver** subkey and then select **New > String Value**.
 - b. Type the key name of the configuration option and then press **Enter**.

To confirm the key names for each configuration option, see [Connector Configuration Options](#) on page 56.
 - c. Right-click the new value and then click **Modify**.
 - d. In the Edit String dialog box, in the **Value Data** field, type the value for the configuration option.
4. Close the Registry Editor.
5. Restart your ODBC application to make sure that the new settings take effect.

Configuring Authentication on Windows

To access data from Snowflake, you must authenticate the connection. You can configure the Simba Snowflake ODBC Connector to provide your credentials and authenticate the connection using one of the following methods:

- The internal Snowflake authenticator.
- An external web browser and a SAML 2.0-compliant identify provider (IdP) that has been defined for your Snowflake account, such as Okta or ADFS.
- Key pair authentication with a JSON Web Token (JWT).
- OAuth 2.0.
- Native Okta. This method can only be used if your authentication endpoint is Okta.

To configure authentication on Windows:

1. To access authentication options, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
2. In the **Authenticator** field, type the value that corresponds to the authentication method that you use to connect to Snowflake, as described in the table below:

Authentication Method	Authenticator Field Value
The internal Snowflake authenticator <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;">  Note: This is the default authenticator. </div>	snowflake
An external web browser and a SAML 2.0-compliant identify provider (IdP)	Externalbrowser
Key pair authentication with a JSON Web Token (JWT) <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;">  Note: If you are using this authenticator, you must specify the PRIV_KEY_FILE configuration option. For more information, see PRIV_KEY_FILE on page 76. </div>	snowflake_jwt
OAuth 2.0	Oauth

Authentication Method	Authenticator Field Value
<p>Native Okta</p> <div data-bbox="305 338 826 537" style="border: 1px solid gray; padding: 5px;"> <p> Note:</p> <p>This method can only be used if your authentication endpoint is Okta.</p> </div>	<p><code>https://[okta_account].okta.com</code>, where <i>[okta_account]</i> is your Okta account name.</p>

3. Unless you are using OAuth 2.0 as your authentication method, in the **User** field, type the user name that you use to authenticate your connection.
4. To save your settings and close the dialog box, click **OK**.

 **Note:**

On Windows, the Snowflake Configuration dialog box contains a Password field; however, the connector does not store any values entered in the field. Instead, the connector requires credentials to be provided at connection time.

Configuring Logging Options on Windows

To help troubleshoot issues, you can enable logging in the Simba Snowflake ODBC Connector.

! 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.
- The settings for logging apply to every connection that uses the Simba Snowflake ODBC Connector, so make sure to disable the feature after you are done using it.

You can configure the connector to provide several kinds of logging functionality:

- [Configuring Event Tracing on Windows](#) on page 13
- [Configuring Connection Logging on Windows](#) on page 15
- [Configuring Verbose cURL Logging on Windows](#) on page 18

Configuring Event Tracing on Windows

You can use event tracing to troubleshoot issues with the connector.

To enable event tracing on Windows:

1. To access tracing options, open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
2. In the **Tracing** field, type the number that corresponds to the amount of information that you want to include in log files:

Tracing 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.

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

The Simba Snowflake ODBC Connector produces log files named `snowflake_odbc_generic[Number].log`, where `[Number]` is a number that identifies each log file. This file is stored in the path specified in the `LogPath` configuration option, which must be set in the Windows registry. If this option is not specified, the log file is written to the text terminal (STDOUT).

Note:

If `EnablePidLogFileNames` is set to `true`, the connector adds the process ID to the log file name, producing a log file named `snowflake_odbc_generic_[ProcessId]_[Number].log`.

To disable event tracing on Windows:

1. Open the ODBC Data Source Administrator where you created the DSN, then select the DSN, and then click **Configure**.
2. In the **Tracing** field, type 0.
3. Click **OK**.
4. Restart your ODBC application to make sure that the new settings take effect.

Configuring Connection Logging on Windows

You can use connection logging to troubleshoot issues with the connection.

On Windows, connection logging options are configured through 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 enable connection logging on Windows:

1. Choose one:
 - If you are using Windows 7 or earlier, click **Start** , then type **regedit** in the **Search** field, and then click **regedit.exe** in the search results.
 - Or, if you are using Windows 8 or later, on the Start screen, type **regedit**, and then click the **regedit** search result.
2. Navigate to the appropriate registry key for the bitness of your connector and your machine:
 - If you are using the 32-bit connector on a 64-bit machine, then browse to the following registry key:

```
HKEY_LOCAL_MACHINE\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Driver\Driver
```
 - Otherwise, browse to the following registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Driver\Driver
```
3. Create the **LogLevel** value by doing the following:
 - a. Right-click the **Driver** subkey and then select **New > String Value**.
 - b. Type **LogLevel**, and then press **Enter**.
 - c. Right-click the **LogLevel** value and then click **Modify**.
 - d. In the Edit String dialog box, in the **Value Data** field, type the number corresponding to the amount of information to include in the log files, and then click **OK**.

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.

4. Optionally, repeat the process described in the previous step to create the following string values in the `Driver` subkey:

Value Name	Value Data
<code>LogPath</code>	The full path to the folder where you want to save log files. If this option is not specified, the log files are written to the text terminal (STDOUT).
<code>LogFileCount</code>	<p>The maximum number of log files to keep.</p> <div style="border: 1px solid gray; padding: 5px; background-color: #f0f0f0;"> <p> Note:</p> <ul style="list-style-type: none"> • If this key is not set, the default value is 50. • After the maximum number of log files is reached, each time an additional file is created, the connector deletes the oldest log file. </div>

Value Name	Value Data
LogFileSize	<p>The maximum size of each log file, in bytes.</p> <div style="border: 1px solid black; padding: 5px; background-color: #f0f0f0;"> <p> Note:</p> <ul style="list-style-type: none"> • If this key is not set, the default value is 20971520. • After the maximum file size is reached, the connector creates a new file and continues logging. </div>
UseLogPrefix	<p>Set this key to 1 to prefix the log file name with the user name and process ID associated with the connection.</p> <p>When this option is set to 0 (the default value), the log prefix is not used.</p>

5. Close the Registry Editor.
6. Restart your ODBC application to make sure that the new settings take effect.

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

- A `simbasnowflakeodbcdriver.log` file that logs connector activity that is not specific to a connection.
- A `simbasnowflakeodbcdriver_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 the `LogPath` key is not specified, the log files are written to the text terminal (STDOUT).

If you set the `UseLogPrefix` key 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.

 **Note:**

If `EnablePidLogFileNames` is set to `true`, the connector adds the process ID to the log file name, producing a log file named `snowflake_odbc_generic_[ProcessId]_[Number].log`.

To disable connection logging on Windows:

1. Choose one:
 - If you are using Windows 7 or earlier, click **Start** , then type **regedit** in the **Search** field, and then click **regedit.exe** in the search results.
 - Or, if you are using Windows 8 or later, on the Start screen, type **regedit**, and then click the **regedit** search result.
2. Navigate to the appropriate registry key for the bitness of your connector and your machine:
 - If you are using the 32-bit connector on a 64-bit machine, then browse to the following registry key:
`HKEY_LOCAL_MACHINE\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Driver\Driver`
 - Otherwise, browse to the following registry key:
`HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Driver\Driver`
3. Right-click the **LogLevel** value and then click **Modify**.
4. In the Edit String dialog box, in the **Value Data** field, type 0.
5. Close the Registry Editor.
6. Restart your ODBC application to make sure that the new settings take effect.

Configuring Verbose cURL Logging on Windows

The Simba Snowflake ODBC Connector uses libcurl as the HTTP and SSL library. The libcurl library supports verbose cURL logging, which you can use to diagnose network issues.

On Windows, cURL logging options are configured through 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 enable cURL verbose logging on Windows:

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

2. Navigate to the appropriate registry key for the bitness of your connector and your machine:
 - If you are using the 32-bit connector on a 64-bit machine, then browse to the following registry key:

```
HKEY_LOCAL_MACHINE\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Driver\Driver
```
 - Otherwise, browse to the following registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Driver\Driver
```
3. Create the `CURLVerboseMode` value by doing the following:
 - a. Right-click the **Driver** subkey and then select **New > String Value**.
 - b. Type `CURLVerboseMode`, and then press **Enter**.
 - c. Right-click the **CURLVerboseMode** value and then click **Modify**.
 - d. In the Edit String dialog box, in the **Value Data** field, type `true`.
4. Close the Registry Editor.
5. Restart your ODBC application to make sure that the new settings take effect.

The Simba Snowflake ODBC Connector produces a log file named `snowflake_odbc_curl.dmp`, at the location specified in the `LogPath` key. If the `LogPath` key is not specified, the log file is written to the text terminal (STDOUT).

To disable curl logging on Windows:

1. Choose one:
 - If you are using Windows 7 or earlier, click **Start** , then type **regedit** in the **Search** field, and then click **regedit.exe** in the search results.
 - Or, if you are using Windows 8 or later, on the Start screen, type **regedit**, and then click the **regedit** search result.
2. Navigate to the appropriate registry key for the bitness of your connector and your machine:
 - If you are using the 32-bit connector on a 64-bit machine, then browse to the following registry key:

```
HKEY_LOCAL_MACHINE\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Driver\Driver
```
 - Otherwise, browse to the following registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Driver\Driver
```
3. Right-click the **CURLVerboseMode** value and then click **Modify**.
4. In the Edit String dialog box, in the **Value Data** field, type `false`.
5. Close the Registry Editor.
6. 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 Snowflake 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 Snowflake.

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

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 10.14 or 10.15
- 150MB of available disk space
- One of the following ODBC driver managers installed:
 - iODBC 3.52.9 or later
 - unixODBC 2.2.14 or later

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 Snowflake ODBC Connector is available for macOS as a `.dmg` file named `Simba Snowflake 2.23.dmg`. The connector supports both 32- and 64-bit client applications.

To install the Simba Snowflake ODBC Connector on macOS:

1. Double-click **Simba Snowflake 2.23.dmg** to mount the disk image.
2. Double-click **Simba Snowflake 2.23.pkg** to run the installer.
3. In the installer, click **Continue**.
4. On the Software License Agreement screen, click **Continue**, and when the prompt appears, click **Agree** if you agree to the terms of the License Agreement.
5. Optionally, to change the installation location, click **Change Install Location**, then select the desired location, and then click **Continue**.

 **Note:**

By default, the connector files are installed in the `/Library/simba/snowflake` directory.

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

Next, configure the environment variables on your machine to make sure that the ODBC driver manager can work with the connector. For more information, see [Configuring the ODBC Driver Manager on Non-Windows Machines](#) on page 25.

Verifying the Connector Version Number on macOS

If you need to verify the version of the Simba Snowflake 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 simba.snowflakeodbc
```

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

Linux Connector

Linux System Requirements

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

- One of the following distributions:
 - Red Hat® Enterprise Linux® (RHEL) 7
 - CentOS 7
 - SUSE Linux Enterprise Server (SLES) 12 or 15
 - Debian 8 or 9
 - Ubuntu 16.04 or 18.04
- 150 MB of available disk space
- One of the following ODBC driver managers installed:
 - iODBC 3.52.9 or later
 - unixODBC 2.2.14 or later

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

Installing the Connector 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 Snowflake ODBC Connector is available as a tarball package named `SimbaSnowflakeODBC-[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 Snowflake ODBC Connector files are installed in the `/opt/simba/snowflake` directory.

Next, configure the environment variables on your machine to make sure that the ODBC driver manager can work with the connector. For more information, see [Configuring the ODBC Driver Manager on Non-Windows Machines](#) on page 25.

Verifying the Connector Version Number on Linux

If you need to verify the version of the Simba Snowflake ODBC Connector that is installed on your Linux machine, you can search the connector's binary file for version number information.

To verify the driver 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/snowflake/lib`.
2. Open the driver'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 Snowflake 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 25.
- 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 26.

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.snowflakeodbc.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 `SIMBASNOWFLAKEINI` to the full path and file name of the `simba.snowflakeodbc.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 `SIMBASNOWFLAKEINI` to the full path and file name of the `simba.snowflakeodbc.ini` file.

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

For unixODBC:

```
export ODBCINI=/usr/local/odbc/odbc.ini
export ODBCSYSINI=/usr/local/odbc
```

```
export SIMBASNOWFLAKEINI=/etc/simba.snowflakeodbc.ini
```

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

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

Configuring ODBC Connections on a Non-Windows Machine

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

- [Creating a Data Source Name on a Non-Windows Machine](#) on page 28
- [Configuring a DSN-less Connection on a Non-Windows Machine](#) on page 32
- [Configuring Authentication on a Non-Windows Machine](#) on page 34
- [Configuring Logging Options on a Non-Windows Machine](#) on page 35
- [Testing the Connection on a Non-Windows Machine](#) on page 40

Creating a Data Source Name on a Non-Windows Machine

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

You can specify connection settings in a DSN (in the `odbc.ini` file) or in a connection string. In addition, some settings can be specified as connector-wide settings in the `simba.snowflakeodbc.ini` file. Settings in the connection string take precedence over settings in the DSN, and settings in the DSN take precedence over connector-wide settings.

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

For information about specifying settings in a connection string, see [Configuring a DSN-less Connection on a Non-Windows Machine](#) on page 32 and [Using a Connection String](#) on page 42. For information about connector-wide settings, see [Setting Connector-Wide Configuration Options on a Non-Windows Machine](#) on page 31.

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 Snowflake ODBC Connector
```

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

```
[ODBC Data Sources]
Sample DSN=Simba Snowflake 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/snowflake/lib/libsnowflakeodbc_
sbu.dylib
```

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

```
Driver=/opt/simba/snowflake/lib/32/libsnowflakeodbc_
sb32.so
```

- b. Set the `Server` property to the host name for your Snowflake account, in the format `[account_name].snowflakecomputing.com`.
- c. If authentication is required to access the Snowflake server, then specify the authentication mechanism and your credentials. For more information, see [Configuring Authentication on a Non-Windows Machine](#) on page 34.
- d. Optionally, set additional key-value pairs as needed to specify other connection settings. For detailed information about all the configuration options supported by the Simba Snowflake ODBC Connector, see [Connector Configuration Options](#) on page 56.

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 26.

For example, the following is an `odbc.ini` configuration file for macOS containing a DSN that authenticates the connection using an external web browser and identity provider, and has logging enabled at the Error level:

```
[ODBC Data Sources]
Sample DSN=Simba Snowflake ODBC Connector
[Sample DSN]
Driver=/Library/simba/snowflake/lib/libsnowflakeodbc_sbu.dylib
Server=yourserver.snowflakecomputing.com
Authenticator=Externalbrowser
UID=skroob
Tracing=2
```

As another example, the following is an `odbc.ini` configuration file for a 32-bit connector on a Linux machine, containing a DSN that authenticates the connection using an external web browser and identity provider, and has logging enabled at the Error level:

```
[ODBC Data Sources]
Sample DSN=Simba Snowflake ODBC Connector 32-bit
[Sample DSN]
Driver=/opt/simba/snowflake/lib/32/libsnowflakeodbc_sb32.so
Server=yourserver.snowflakecomputing.com
Authenticator=Externalbrowser
UID=skroob
Tracing=2
```

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

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

When you specify connection settings in a DSN or connection string, those settings apply only when you connect to Snowflake using that particular DSN or string. As an alternative, you can specify certain settings that apply to every connection that uses the Simba Snowflake ODBC Connector by configuring them in the `simba.snowflakeodbc.ini` file.

Note:

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

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

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

For example, to specify a proxy server and to log information from the libcurl library, type the following:

```
Proxy=http://proxy.mycompany.ca:1066
CURLVerboseMode=true
```

For detailed information about the configuration options supported by the connector at the connector-wide level, see below.

3. Save the `simba.snowflakeodbc.ini` configuration file.
4. Restart your ODBC application to make sure that the new settings take effect.

The following connector-wide properties must be configured in the `simba.snowflakeodbc.ini` file, or as Windows Registry key values on Windows.

- [CURLVerboseMode](#) on page 66
- [EnableAutoIpdByDefault](#) on page 68
- [EnablePidLogFileNames](#) on page 69
- [LogFileCount](#) on page 70
- [LogFileSize](#) on page 70
- [LogLevel](#) on page 71
- [LogPath](#) on page 72
- [OCSPFailOpen](#) on page 74

- [UseLogPrefix](#) on page 80
- [UseURandomDevice](#) on page 81

In addition, the following properties can be configured either as per-session properties (in a connection string or `odbc.ini` file), or as connector-wide properties (in the `simba.snowflakeodbc.ini` file or in the Windows Registry):

- [CABundleFile](#) on page 63
- [CLIENT_METADATA_REQUEST_USE_CONNECTION_CTX](#) on page 64
- [CLIENT_SESSION_KEEP_ALIVE](#) on page 65
- [CLIENT_TIMESTAMP_TYPE_MAPPING](#) on page 65
- [DEFAULT_BINARY_SIZE](#) on page 67
- [DEFAULT_VARCHAR_SIZE](#) on page 67
- [PUT_FASTFAIL](#) on page 77
- [PUT_MAXRETRIES](#) on page 77
- [PUT_TEMPDIR](#) on page 78
- [TIMEZONE](#) on page 79

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 Snowflake ODBC Connector=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/snowflake/lib/libsnowflakeodbc_sbu.dylib
```

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

```
Driver=/opt/simba/snowflake/lib/32/libsnowflakeodbc_sb32.so
```

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

For example:

```
Description=Simba Snowflake 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 26.

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

```
[ODBC Drivers]
Simba Snowflake ODBC Connector=Installed
[Simba Snowflake ODBC Connector]
Description=Simba Snowflake ODBC Connector
Driver=/Library/simba/snowflake/lib/libsnowflakeodbc_sbu.dylib
```

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

```
[ODBC Drivers]
Simba Snowflake ODBC Connector 32-bit=Installed
Simba Snowflake ODBC Connector 64-bit=Installed
[Simba Snowflake ODBC Connector 32-bit]
Description=Simba Snowflake ODBC Connector (32-bit)
Driver=/opt/simba/snowflake/lib/32/libsnowflakeodbc_sb32.so
[Simba Snowflake ODBC Connector 64-bit]
Description=Simba Snowflake ODBC Connector (64-bit)
Driver=/opt/simba/snowflake/lib/64/libsnowflakeodbc_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 42.

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

- [Configuring Authentication on a Non-Windows Machine](#) on page 34
- [Configuring Event Tracing on a Non-Windows Machine](#) on page 36
- [Configuring Connection Logging on a Non-Windows Machine](#) on page 37

For detailed information about all the connection properties that the connector supports, see [Connector Configuration Options](#) on page 56.

Configuring Authentication on a Non-Windows Machine

To access data from Snowflake, you must authenticate the connection. You can configure the Simba Snowflake ODBC Connector to provide your credentials and authenticate the connection using one of the following methods:

- The internal Snowflake authenticator.
- An external web browser and a SAML 2.0-compliant identify provider (IdP) that has been defined for your Snowflake account, such as Okta or ADFS.
- Key pair authentication with a JSON Web Token (JWT).
- OAuth 2.0.
- Native Okta. This method can only be used if your authentication endpoint is Okta.

To configure authentication on a non-Windows machine:

1. Open the `odbc.ini` configuration file in a text editor.
2. Set the `Authenticator` property to one of the following values:

Authentication Method	Authenticator Value
The internal Snowflake authenticator <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;">  Note: This is the default authenticator. </div>	<code>snowflake</code>
An external web browser and a SAML 2.0-compliant identify provider (IdP)	<code>Externalbrowser</code>
Key pair authentication with a JSON Web Token (JWT)	<code>snowflake_jwt</code>
OAuth 2.0	<code>Oauth</code>
Native Okta <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;">  Note: This method can only be used if your authentication endpoint is Okta. </div>	<code>https://[okta_account].okta.com, where [okta_account] is your Okta account name.</code>

3. Unless you are using OAuth 2.0 as your authentication method, set the `UID` property to the user name that you use to authenticate your connection.
4. If you are using the internal Snowflake authenticator or native Okta as your authentication method, set the `PWD` property to the password that you use to authenticate your connection.
5. If you are using a JWT as your authentication method, set the `PRIV_KEY_FILE` property to the path to the private key file for key pair authentication.
6. Save the `odbc.ini` configuration file.

Configuring Logging Options on a Non-Windows Machine

To help troubleshoot issues, you can enable logging in the Simba Snowflake ODBC Connector.

! 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.
- The settings for logging apply to every connection that uses the Simba Snowflake ODBC Connector, so make sure to disable the feature after you are done using it.

You can configure the connector to provide several kinds of logging functionality:

- [Configuring Event Tracing on a Non-Windows Machine](#) on page 36
- [Configuring Connection Logging on a Non-Windows Machine](#) on page 37
- [Configuring Verbose cURL Logging on a Non-Windows Machine](#) on page 39

Configuring Event Tracing on a Non-Windows Machine

You can use event tracing to troubleshoot issues with the connector.

To enable event tracing on a non-Windows machine:

1. Open the `simba.snowflakeodbc.ini` configuration file in a text editor.
2. To specify the level of information to include in log files, set the `Tracing` 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.

3. Save the `simba.snowflakeodbc.ini` configuration file.
4. Restart your ODBC application to make sure that the new settings take effect.

The Simba Snowflake ODBC Connector produces log files named `snowflake_odbc_generic [Number].log`, where `[Number]` is a number that identifies each log file. This file is stored in the path specified in the `LogPath` configuration option. If this option is not specified, the log file is written to the text terminal (STDOUT).

 **Note:**

If `EnablePidLogFileNames` is set to `true`, the connector adds the process ID to the log file name, producing a log file named `snowflake_odbc_generic_[ProcessId]_[Number].log`.

To disable event tracing on a non-Windows machine:

1. Open the `simba.snowflakeodbc.ini` configuration file in a text editor.
2. Set the `Tracing` key to 0.
3. Save the `simba.snowflakeodbc.ini` configuration file.
4. Restart your ODBC application to make sure that the new settings take effect.

Configuring Connection Logging on a Non-Windows Machine

You can use connection logging to troubleshoot issues with the connection.

On non-Windows platforms, connection logging options are configured through connector-wide settings in the `simba.snowflakeodbc.ini` file, which apply to all connections that use the connector.

To enable connection logging on a non-Windows machine:

1. Open the `simba.snowflakeodbc.ini` configuration file in a text editor.
2. 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.

LogLevel Value	Description
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.

3. Set the `LogPath` property to the full path to the folder where you want to save log files.
4. Set the `LogFileCount` property 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.

5. Set the `LogFileSize` property 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.

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

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

- A `simbasnowflakeodbcdriver.log` file that logs connector activity that is not specific to a connection.
- A `simbasnowflakeodbcdriver_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 the `LogPath` property is not specified, the log files are written to the text terminal (STDOUT).

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.

 **Note:**

If `EnablePidLogFileNames` is set to `true`, the connector adds the process ID to the log file name, producing a log file named `snowflake_odbc_generic_[ProcessID]_[Number].log`.

To disable connection logging on a non-Windows machine:

1. Open the `simba.snowflakeodbc.ini` configuration file in a text editor.
2. Set the `LogLevel` property to 0.
3. Save the `simba.snowflakeodbc.ini` configuration file.
4. Restart your ODBC application to make sure that the new settings take effect.

Configuring Verbose cURL Logging on a Non-Windows Machine

The Simba Snowflake ODBC Connector uses `libcurl` as the HTTP and SSL library. The `libcurl` library supports verbose cURL logging, which you can use to diagnose network issues.

On non-Windows platforms, cURL logging options are configured through connector-wide settings in the `simba.snowflakeodbc.ini` file, which apply to all connections that use the connector.

To enable cURL verbose logging on a non-Windows machine:

1. Open the `simba.snowflakeodbc.ini` configuration file in a text editor.
2. Set the `CURLVerboseMode` property to `true`.
3. Save the `simba.snowflakeodbc.ini` configuration file.
4. Restart your ODBC application to make sure that the new settings take effect.

The Simba Snowflake ODBC Connector produces a log file named `snowflake_odbc_curl.dmp`, at the location specified in the `LogPath` configuration option. If this option is not specified, the log file is written to the text terminal (STDOUT).

To disable additional logging on a non-Windows machine:

1. Open the `simba.snowflakeodbc.ini` configuration file in a text editor.
2. Set the `CURLVerboseMode` property to `false`.

3. Save the `simba.snowflakeodbc.ini` configuration file.
4. 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 42.

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 Options](#) on page 56.

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:

- *[AuthToken]* is the OAuth access token.
- *[OktaAccount]* is your Okta account name.
- *[PrivateKeyFile]* is the path to the private key file for key pair authentication.
- *[ServerName]* is the fully qualified domain name of the Snowflake server host.
- *[YourPassword]* is the password corresponding to your user name.
- *[YourUserName]* is the user name that you use to access the Snowflake server.

Connecting to a Snowflake Server Using the Internal Snowflake Authenticator

The following is the format of a DSN-less connection string that connects using the internal Snowflake authenticator:

```
Driver=Simba Snowflake ODBC Driver;  
Server=[ServerName];UID=[YourUserName];PWD=[YourPassword];
```

For example:

```
Driver=Simba Snowflake ODBC Driver;  
Server=mydata.snowflakecomputing.com;UID=skroob;PWD=12345;
```

Connecting to a Snowflake Server Using an External Browser

The following is the format of a DSN-less connection string that connects using an external web browser:

```
Driver=Simba Snowflake ODBC Driver;  
Server=[ServerName];Authenticator=Externalbrowser;UID=  
[YourUserName];
```

For example:

```
Driver=Simba Snowflake ODBC Driver;  
Server=mydata.  
snowflake  
computing.com;Authenticator=Externalbrowser;UID=skroob;
```

Connecting to a Snowflake Server Using a JSON Web Token

The following is the format of a DSN-less connection string that connects using a JSON Web Token (JWT):

```
Driver=Simba Snowflake ODBC Driver;  
Server=[ServerName];Authenticator=snowflake_jwt;UID=  
[YourUserName];PRIV_KEY_FILE=[PrivateKeyFile];
```

For example:

```
Driver=Simba Snowflake ODBC Driver;  
Server=mydata.snowflakecomputing.com;Authenticator=snowflake_  
jwt;UID=skroob;PRIV_KEY_FILE=C:\Snowflake\keyfile;
```

Connecting to a Snowflake Server Using OAuth 2.0

The following is the format of a DSN-less connection string that connects using an OAuth 2.0 authentication token:

```
Driver=Simba Snowflake ODBC Driver;  
Server=[ServerName];Authenticator=OAuth;Token=[AuthToken];
```

For example:

```
Driver=Simba Snowflake ODBC Driver;  
Server=mydata.snowflake  
computing.com;Authenticator=OAuth;Token=kN9PcyQ9prK4LvUMMMpFL4  
R+lVE=;
```

Connecting to a Snowflake Server Using Okta

The following is the format of a DSN-less connection string that connects using Okta for authentication:

```
Driver=Simba Snowflake ODBC Driver;  
Server=[ServerName];Authenticator=https://  
[OktaAccount].okta.com;UID=[YourUserName];PWD=[YourPassword];
```

For example:

```
Driver=Simba Snowflake ODBC Driver;  
Server=mydata.snowflake  
computing.com;Authenticator=https://myokta.okta.com;UID=skroob  
;PWD=12345;
```

Features

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

- [Data Types](#) on page 45
- [Snowflake-Specific DML Commands](#) on page 49
- [Snowflake-Specific SQL Attributes](#) on page 52
- [Multiple Statements](#) on page 53
- [Proxy Server](#) on page 53
- [Security and Authentication](#) on page 54

Data Types

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

The table below lists the supported data type mappings.

Snowflake Type	SQL Type	Comment
ARRAY	None	This is a Snowflake-specific data type.
BIGINT	SQL_DECIMAL(38,0)	Synonymous with NUMBER but with a fixed precision and length.
BINARY	SQL_BINARY	The maximum length is 8MB.
BOOLEAN	SQL_BIT	
BYTEINT	SQL_DECIMAL(38,0)	Synonymous with NUMBER but with a fixed precision and length.
CHARACTER	SQL_VARCHAR	Synonymous with VARCHAR but with a default length of 1.

Snowflake Type	SQL Type	Comment
DATE	SQL_DATE	
DATETIME	SQL_TIMESTAMP	Synonymous with TIMESTAMP_NTZ.
DECIMAL	SQL_DECIMAL	Synonymous with NUMBER.
DOUBLE PRECISION (DOUBLE)	SQL_DOUBLE	Synonymous with FLOAT.
FLOAT FLOAT4 FLOAT8	SQL_DOUBLE	Snowflake uses double- precision (64-bit) IEEE 754 floating point numbers.
INTEGER (INT)	SQL_DECIMAL(38,0)	Synonymous with NUMBER but with a fixed precision and length.
NUMBER	SQL_DECIMAL	The maximum scale is 37.
NUMERIC	SQL_DECIMAL	Synonymous with NUMBER.
OBJECT	None	This is a Snowflake- specific data type.
REAL	SQL_DOUBLE	Synonymous with FLOAT.
SMALLINT	SQL_DECIMAL(38,0)	Synonymous with NUMBER but with a fixed precision and length.
STRING TEXT	SQL_VARCHAR	Synonymous with VARCHAR.

Snowflake Type	SQL Type	Comment
TIME	SQL_TIME	See the note below on SQL_TIMESTAMP data types.
TIMESTAMP	SQL_TIMESTAMP	<p>TIMESTAMP is a user-specified alias associated with one of the TIMESTAMP_* variations. The default is TIMESTAMP_NTZ.</p> <p>In all operations where TIMESTAMP is used, the associated TIMESTAMP_* variation is automatically used. The TIMESTAMP data type is never stored in tables.</p>
TIMESTAMP_LTZ TIMESTAMPLTZ TIMESTAMP WITH LOCAL TIME ZONE	SQL_TIMESTAMP	<ul style="list-style-type: none"> • See the note below on SQL_TIMESTAMP data types. • TIMESTAMP_LTZ internally stores UTC time. • All operations are performed in the current session's time zone, controlled by the TIMEZONE session parameter.

Snowflake Type	SQL Type	Comment
TIMESTAMP_NTZ TIMESTAMPNTZ TIMESTAMP WITHOUT TIME ZONE	SQL_TIMESTAMP	<ul style="list-style-type: none"> • See the note below on SQL_TIMESTAMP data types. • TIMESTAMP_NTZ internally stores "wallclock" time. • All operations are performed without taking any time zone into account.
TIMESTAMP_TZ TIMESTAMPTZ TIMESTAMP WITH TIME ZONE	SQL_TIMESTAMP	<ul style="list-style-type: none"> • See the note below on SQL_TIMESTAMP data types. • TIMESTAMP_TZ internally stores UTC time together with an associated time zone offset. When a time zone is not provided, the session time zone offset is used. • All operations are performed with the time zone offset specific to each record.
TINYINT	SQL_DECIMAL(38,0)	Synonymous with NUMBER but with a fixed precision and length.
VARBINARY	SQL_BINARY	Synonymous with BINARY.
VARCHAR	SQL_VARCHAR	The maximum length is 16 MB.

Snowflake Type	SQL Type	Comment
VARIANT	None	This is a Snowflake-specific data type.

! Important:

For all SQL_TIME and SQL_TIMESTAMP data types:

- Time precision can range from 0 (seconds) to 9 (nanoseconds). The default precision is 9.
- All TIME values must be between 00:00:00 and 23:59:59.999999999.

Snowflake-Specific DML Commands

The connector enables you to execute the Snowflake-specific DML commands PUT, GET, COPY INTO, and REMOVE. For more information about these commands, see "DML Commands" in the Snowflake documentation: <https://docs.snowflake.net/manuals/sql-reference/sql-dml.html>.

The PUT, GET, and COPY INTO commands return a result set. These result sets are described below.

- [PUT Result Set](#) on page 49
- [GET Result Set](#) on page 50
- [COPY INTO Result Sets](#) on page 51

The REMOVE command does not return a result set.

PUT Result Set

The table below describes the result set that the connector returns after executing the PUT command.

Column Name	Data Type	Description
source	SQL_VARCHAR	The relative path and name of the source file.
target	SQL_VARCHAR	The relative path and name of the target file.

Column Name	Data Type	Description
source_size	SQL_DECIMAL	The size of the source file, in bytes.
target_size	SQL_DECIMAL	The size of the target file, in bytes.
source_compression	SQL_VARCHAR	The type of compression used for the source file.
target_compression	SQL_VARCHAR	The type of compression used for the target file.
status	SQL_VARCHAR	The status of the command, which can be one of the following: UPLOADED, SKIPPED, or ERROR.
encryption	SQL_VARCHAR	Whether the target file is encrypted. This should always be ENCRYPTED.
message	SQL_VARCHAR	Any additional information about the command.

GET Result Set

The table below describes the result set that the connector returns after executing the GET command.

Column Name	Data Type	Description
file	SQL_VARCHAR	The relative path and name of the target file.
size	SQL_DECIMAL	The size of the target file, in bytes.
status	SQL_VARCHAR	The status of the command, which can be either DOWNLOADED or ERROR.

Column Name	Data Type	Description
encryption	SQL_VARCHAR	Whether the target file is encrypted. This should always be ENCRYPTED.
message	SQL_VARCHAR	Any additional information about the command.

COPY INTO Result Sets

The tables below describe the result sets that the connector can return after executing the COPY INTO command, depending on the destination.

If you execute the COPY INTO command on a table, the connector generates the following result set:

Column Name	Data Type	Description
file	SQL_VARCHAR	The relative path and name of the source file.
status	SQL_VARCHAR	The status of the command, which can be one of the following: LOADED, LOAD FAILED, or PARTIALLY LOADED.
rows_parsed	SQL_DECIMAL	The number of rows that have been parsed from the source file.
rows_loaded	SQL_DECIMAL	The number of rows that have been loaded from the source file.
error_limit	SQL_DECIMAL	If the number of errors reaches this limit, the connector aborts the command.
error_seen	SQL_DECIMAL	The number of errors that the connector has encountered in the source file.
first_error	SQL_VARCHAR	The first error encountered in the source file.

Column Name	Data Type	Description
first_error_line	SQL_DECIMAL	The line number of the first error encountered in the source file.
first_error_position	SQL_DECIMAL	The character position of the first error encountered in the source file.
first_error_column_name	SQL_VARCHAR	The name of the column containing the first error encountered in the source file.

If you execute the COPY INTO command on a location, the connector generates the following result set:

Column Name	Data Type	Description
rows_uploaded	SQL_DECIMAL	The number of rows that have been uploaded from the source file.
input_bytes	SQL_DECIMAL	The size of the data uploaded from the source file.
output_bytes	SQL_DECIMAL	The size of the target file.

Snowflake-Specific SQL Attributes

Certain connector configuration options can be defined through the use of Snowflake-specific SQL attributes, as follows:

- The `APPLICATION` option can be defined using the `SQL_SF_CONN_ATTR_APPLICATION` attribute.
- The `PRIV_KEY_FILE` and `PRIV_KEY_FILE_PWD` options can be defined using the `SQL_SF_CONN_ATTR_PRIV_KEY` attribute.

For more information, see the "*Snowflake-Specific Behavior*" section in the *ODBC Driver API Support* documentation: <https://docs.snowflake.net/manuals/user-guide/odbc-api.html#snowflake-specific-behavior>.

The connector supports the Snowflake-specific statement attribute `SQL_SF_STMT_ATTR_LAST_QUERY_ID`, which you can retrieve by calling `SQLGetStmtAttr`.

For more information, see the "*Retrieving The Last Query ID*" section in the *ODBC Driver API Support* documentation: <https://docs.snowflake.net/manuals/user-guide/odbc-api.html#retrieving-last-query-id>.

The connector supports the Snowflake-specific statement attribute `SQL_SF_STMT_ATTR_MULTI_STATEMENT_COUNT`, which you can set by calling `SqlSetStmtAttr`.

For more details, see the "*Executing a Batch of SQL Statements (Multi-Statement Support)*" section in the *Using the ODBC Driver* documentation: <https://docs.snowflake.com/en/user-guide/odbc-using.html#specifying-the-number-of-statements-in-a-batch>.

Multiple Statements

You can configure the Simba Snowflake ODBC Connector to execute multiple statements in a single request. To do so, separate each statement with a semicolon (;).

For example:

```
// Sending a batch of SQL statements to be executed
rc = SQLExecDirect(hstmt, (SQLCHAR *) "select c1 from t1;
select c2 from t2; select c3 from t3",
SQL_NTS);
```

You must also specify the exact number of statements in the batch. To do so, call `SqlSetStmtAttr` to set the `SQL_SF_STMT_ATTR_MULTI_STATEMENT_COUNT` attribute to the number of statements in the batch.

! Important:

The Snowflake database requires the exact number of statements in order to guard against SQL injection attacks.

For more details, see the "*Executing a Batch of SQL Statements (Multi-Statement Support)*" section in the *Using the ODBC Driver* documentation: <https://docs.snowflake.com/en/user-guide/odbc-using.html#specifying-the-number-of-statements-in-a-batch>.

Proxy Server

You can configure the Simba Snowflake ODBC Connector to connect to Snowflake through a proxy server.

You specify proxy server settings in the connector using the `Proxy` and `No_Proxy` configuration options. These settings can be configured in the following ways:

- As per-session properties, which can be specified in a DSN or a connection string or in the `odbc.ini` file for macOS or Linux.
- As connector-wide properties, which can be specified in the Windows Registry or in the `simba.snowflakeodbc.ini` file for macOS or Linux.

You can also specify proxy settings using the environment variables `https_proxy`, `http_proxy`, and `no_proxy`.

Proxy settings in a DSN or connection string take precedence over connector-wide settings, and connector-wide settings take precedence over environment variables. If you are using environment variables, `https_proxy` takes precedence over `http_proxy`.

Security and Authentication

To protect data from unauthorized access, Snowflake data stores require connections to be authenticated with user credentials and the SSL protocol. The Simba Snowflake 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 up to TLS 1.2. The SSL version used for the connection is the highest version that is supported by both the connector and the server.

The connector provides mechanisms that enable you to authenticate your connection using one of the following methods:

- The internal Snowflake authenticator.
- An external web browser and a SAML 2.0-compliant identify provider (IdP) that has been defined for your account, such as Okta or ADFS.
- Key pair authentication with a JSON Web Token (JWT).
- OAuth 2.0.
- Native Okta, if your endpoint is Okta.

You must use the authentication mechanism that matches the security requirements of the Snowflake server. For detailed connector configuration instructions see [Configuring Authentication on Windows](#) on page 11 or [Configuring Authentication on a Non-Windows Machine](#) on page 34.

Additionally, the connector automatically applies SSL encryption to all connections. SSL encryption protects data and credentials when they are transferred over the network, and provides stronger security than authentication alone. You can configure

the minimum SSL version that the connector accepts by setting the [SSL_Version](#) on page [79](#) configuration property in a connection string.

Connector Configuration Options

Connector Configuration Options lists the configuration options available in the Simba Snowflake 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 , the fields and buttons described below are available in the following dialog boxes:

- Logging Options

When using a connection string, 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 Snowflake ODBC Connector, or via the key name when using a connection string or configuring a connection from a Linux/macOS machine:

- [Authenticator](#) on page 57
- [Database](#) on page 57
- [Password](#) on page 58
- [Role](#) on page 59
- [Schema](#) on page 60
- [Server](#) on page 60
- [Tracing](#) on page 60
- [User](#) on page 61
- [Warehouse](#) on page 61

In addition, the following properties can be configured either as per-session properties or as connector-wide properties. Per-session properties can be configured in a DSN or a connection string, or the `odbc.ini` file for macOS or Linux. Driver-wide properties can be configured in the Windows Registry or in the `simba.snowflakeodbc.ini` file for macOS or Linux.

- [NoProxy](#) on page 57
- [Proxy](#) on page 58

Authenticator

Key Name	Default Value	Required
Authenticator	snowflake	No

Description

This property specifies the authenticator to use to verify your login credentials.

Set this property to one of the following authenticators:

- `snowflake`: The internal Snowflake authenticator.
- `Externalbrowser`: An external web browser and a SAML 2.0-compliant identity provider (IdP) that has been defined for your account, such as Okta or ADFS
- `snowflake_jwt`: Key pair authentication with a JSON Web Token (JWT).
- `OAuth`: OAuth 2.0.
- `https://[your_okta_account_name].okta.com`: Native Okta. This method can only be used if your authentication endpoint is Okta.

Database

Key Name	Default Value	Required
Database	None	No

Description

The name of the default Snowflake database that you want to access.

NoProxy

Key Name	Default Value	Required
No_Proxy	None	No

Description

Set this property to a comma-separated list of IP addresses or host name endings that should be allowed to bypass the proxy server.

Note:

This parameter does not support wildcard characters.

This property is available as either a per-session setting or a connector-wide setting.

To configure this property as a connector-wide setting 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 Snowflake ODBC Connector\Driver
- Otherwise: HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver

Use `No_Proxy` as the value name, and a comma-separated list of host name endings or IP addresses as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file. For more information, see [Setting Connector-Wide Configuration Options on a Non-Windows Machine](#) on page 31.

Password

Key Name	Default Value	Required
PWD	None	Yes, if Authenticator is set to snowflake or Native Okta.

Description

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

Note:

For security reasons, passwords are not saved in the DSN. When you connect to your data, you are prompted to type your password again.

Proxy

Key Name	Default Value	Required
Proxy	None	No

Description

The proxy server URL. When this property is set, all ODBC communications with Snowflake use this server, with the exception of communications from servers specified in the NoProxy property (see [NoProxy](#) on page 57).

The proxy server URL must be written in one of the following formats:

```
http://[hostname]:[portnumber]/
```

```
[hostname]:[portnumber]
```

Where:

- *[hostname]* is the host name of the proxy server.
- *[portnumber]* is the port used by the proxy server.

This property is available as either a per-session or a connector-wide setting.

To configure this property as a connector-wide setting 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\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Connector\Driver
- Otherwise: HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver

Use `Proxy` as the value name, and the proxy server URL as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file. For more information, see [Setting Connector-Wide Configuration Options on a Non-Windows Machine](#) on page 31.

Role

Key Name	Default Value	Required
Role	None	No

Description

Specifies the default role to use for sessions initiated by the connector.

The specified role should be a role that has been assigned to the specified user for the connector. If the specified role does not match any of the roles assigned to the user,

then sessions initiated by the connector start without any roles. In this case, you can still specify a role from within the session.

Schema

Key Name	Default Value	Required
Schema	public	No

Description

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

Server

Key Name	Default Value	Required
Server	None	Yes

Description

The host name for your Snowflake account, in the following format:

```
[account_name].snowflakecomputing.com
```

Tracing

Key Name	Default Value	Required
Tracing	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.

Set the property to one of the following values:

- 0: Disable 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.

When logging is enabled, the Simba Snowflake ODBC Connector produces a log file named `snowflake_odbc_generic[Number].log`, where `[Number]` is a number that identifies each log file. This file is stored in the path specified in the `LogPath` configuration option. If this option is not specified, the log file is written to the text terminal (STDOUT).

User

Key Name	Default Value	Required
UID	None	Yes, if <code>Authenticator</code> is set to anything other than <code>Oauth</code> .

Description

The user name that you use to access the Snowflake instance.

Warehouse

Key Name	Default Value	Required
Warehouse	None	No

Description

The default warehouse to use for sessions initiated by the connector.

Configuration Options Having Only Key Names

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

- [Application](#) on page 63
- [Driver](#) on page 68
- [JWT_TIMEOUT](#) on page 69
- [KeepLeadingTrailingZeros](#) on page 69
- [LOGIN_TIMEOUT](#) on page 71
- [NETWORK_TIMEOUT](#) on page 73
- [OCSP_FAIL_OPEN](#) on page 74
- [OCSPFailOpen](#) on page 74
- [Passcode](#) on page 75
- [PasscodeInPassword](#) on page 75
- [Port](#) on page 76
- [PRIV_KEY_FILE](#) on page 76
- [PRIV_KEY_FILE_PWD](#) on page 76
- [QUERY_TIMEOUT](#) on page 79
- [SSL_Version](#) on page 79
- [Token](#) on page 80

The following connector-wide properties must be configured as Windows Registry key values, or in the `simba.snowflakeodbc.ini` file for macOS or Linux.

- [CURLVerboseMode](#) on page 66
- [EnableAutoIpdByDefault](#) on page 68
- [EnablePidLogFileNames](#) on page 69
- [LogFileCount](#) on page 70
- [LogFileSize](#) on page 70
- [LogLevel](#) on page 71
- [LogPath](#) on page 72
- [NoExecuteInSQLPrepare](#) on page 73
- [OCSPFailOpen](#) on page 74
- [UseLogPrefix](#) on page 80
- [UseURandomDevice](#) on page 81

In addition, the following properties can be configured either as per-session properties or as connector-wide properties. Per-session properties can be configured in a connection string, or the `odbc.ini` file for macOS or Linux. Driver-wide properties can

be configured in the Windows Registry or in the `simba.snowflakeodbc.ini` file for macOS or Linux.

- [CABundleFile](#) on page 63
- [CLIENT_METADATA_REQUEST_USE_CONNECTION_CTX](#) on page 64
- [CLIENT_SESSION_KEEP_ALIVE](#) on page 65
- [CLIENT_TIMESTAMP_TYPE_MAPPING](#) on page 65
- [DEFAULT_BINARY_SIZE](#) on page 67
- [DEFAULT_VARCHAR_SIZE](#) on page 67
- [PUT_FASTFAIL](#) on page 77
- [PUT_MAXRETRIES](#) on page 77
- [PUT_TEMPDIR](#) on page 78
- [TIMEZONE](#) on page 79

Application

Key Name	Default Value	Required
Application	None	No

Description

The name of a Snowflake partner application to connect to.

This property can also be set through the Snowflake-specific SQL attribute `SQL_SF_CONN_ATTR_APPLICATION`. For more information, see "Snowflake-Specific Behavior" in the Snowflake documentation: <https://docs.snowflake.net/manuals/user-guide/odbc-api.html#snowflake-specific-behavior>.

CABundleFile

Key Name	Default Value	Required
CABundleFile	<code>cacert.pem</code>	No

Description

The path to the Certificate Authority (CA) bundle file. This can be either the full path, or the path relative to the connector binary.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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 Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `CABundleFile` as the value name, and the full path to the CA bundle as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

CLIENT_METADATA_REQUEST_USE_CONNECTION_CTX

Key Name	Default Value	Required
<code>CLIENT_METADATA_REQUEST_USE_CONNECTION_CTX</code>	<code>false</code>	No

Description

This property overwrites the Snowflake session parameter `CLIENT_METADATA_REQUEST_USE_CONNECTION_CTX`. For more information, see the Snowflake documentation.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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 Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `CLIENT_METADATA_REQUEST_USE_CONNECTION_CTX` as the value name, and `true` or `false` as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

CLIENT_SESSION_KEEP_ALIVE

Key Name	Default Value	Required
CLIENT_SESSION_KEEP_ALIVE	false	No

Description

This property overwrites the Snowflake session parameter `CLIENT_SESSION_KEEP_ALIVE`. For more information, see the Snowflake documentation.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `CLIENT_SESSION_KEEP_ALIVE` as the value name, and `true` or `false` as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

CLIENT_TIMESTAMP_TYPE_MAPPING

Key Name	Default Value	Required
CLIENT_TIMESTAMP_TYPE_MAPPING	timestamp_ltz	No

Description

This property overwrites the Snowflake session parameter `CLIENT_TIMESTAMP_TYPE_MAPPING`. For more information, see the Snowflake documentation.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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 Snowflake ODBC Connector\Driver
- Otherwise: HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver

Use `CLIENT_TIMESTAMP_TYPE_MAPPING` as the value name, and one of the following as the value data:

- `timestamp_ltz`
- `timestamp_ntz`

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

CURLVerboseMode

Key Name	Default Value	Required
CURLVerboseMode	false	No

Description

This connector-wide option specifies whether to enable cURL verbose logging.

The Snowflake ODBC connector uses cURL as the HTTP and SSL library. The cURL verbose log can be useful for diagnosing network issues.

Set this property to one of the following values:

- `true`: The Simba Snowflake ODBC Connector produces a log file named `snowflake_odbc_curl.dmp`, at the location specified in the `LogPath` configuration option. If this option is not specified, the log file is written to the text terminal (STDOUT).
- `false`: The connector does not enable cURL verbose logging.

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 Snowflake ODBC Connector\Driver
- Otherwise: HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver

Use `CURLVerboseMode` as the value name, and `true` or `false` as the value data.

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

DEFAULT_BINARY_SIZE

Key Name	Default Value	Required
DEFAULT_BINARY_SIZE	8388608	No

Description

The default size that the connector uses when retrieving and converting values from BINARY columns of undetermined sizes.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Connector\Driver
- Otherwise: HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver

Use `DEFAULT_BINARY_SIZE` as the value name, and the size as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

DEFAULT_VARCHAR_SIZE

Key Name	Default Value	Required
DEFAULT_VARCHAR_SIZE	16777216	No

Description

The default size that the connector uses when retrieving and converting values from VARCHAR columns of undetermined sizes.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `DEFAULT_VARCHAR_SIZE` as the value name, and the size as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

Driver

Key Name	Default Value	Required
<code>Driver</code>	Simba Snowflake 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 Snowflake 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.

EnableAutoIpdByDefault

Key Name	Default Value	Required
<code>EnableAutoIpdByDefault</code>	<code>true</code>	No

Description

This connector-wide setting specifies whether to enable automatic population of the IPD feature by default.

- `true`: The connector enables automatic population of the IPD feature by default.
- `false`: The connector disables automatic population of the IPD feature by default.

 **Note:**

For details on automatic population of the IPD feature, see the Microsoft documentation: <https://docs.microsoft.com/en-us/sql/odbc/reference/develop-app/automatic-population-of-the-ipd?view=sql-server-ver15>.

EnablePidLogFileNames

Key Name	Default Value	Required
EnablePidLogFileNames	false	No

Description

This property specifies whether the connector adds the process ID to the log file name.

- `true`: The connector adds the process ID to the log file name, producing a log file named `snowflake_odbc_generic_[ProcessId]_[Number].log`.
- `false`: The connector does not add the process ID to the log file name.

JWT_TIMEOUT

Key Name	Default Value	Required
JWT_TIMEOUT	30	No

Description

The length of time, in seconds, that the Snowflake service waits to receive the JSON Web Token before returning an error.

KeepLeadingTrailingZeros

Key Name	Default Value	Required
KeepLeadingTrailingZeros	true	No

Description

This property specifies whether to keep the leading and trailing zeros when displaying the string representation of a numeric value.

- `true`: The connector keeps the leading and trailing zeroes when displaying the string representation of a numeric value.
- `false`: The connector trims leading and trailing zeroes.

LogFileCount

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.

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\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `LogFileCount` as the value name, and the number of files to keep as the value data.

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

LogFileSize

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.

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 Snowflake ODBC Connector\Driver
- Otherwise: HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver

Use `LogFileSize` as the value name, and the maximum file size in bytes as the value data.

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

LOGIN_TIMEOUT

Key Name	Default Value	Required
LOGIN_TIMEOUT	60	No

Description

The length of time, in seconds, that the connector waits for a response when connecting to the Snowflake service before returning an error.

LogLevel

Key Name	Default Value	Required
LogLevel	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.
- The settings for logging apply to every connection that uses the Simba Snowflake ODBC Connector, so make sure to disable the feature after you are done using it.

Set the property to one of the following values:

- 0: Disable 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.

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

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

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 Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `LogLevel` as the value name, and a number from 0 to 6 as the value data.

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

LogPath

Key Name	Default Value	Required
<code>LogPath</code>	STDOUT	No

Description

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

If this option is not specified, the log file is written to the text terminal (STDOUT).

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\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Connector\Driver
- Otherwise: HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver

Use `LogPath` as the value name, and the full path to the folder as the value data.

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

NETWORK_TIMEOUT

Key Name	Default Value	Required
NETWORK_TIMEOUT	0	No

Description

The length of time, in seconds, that the connector waits for a response from the Snowflake service before returning an error. A value of 0 indicates no timeout value is set.

NoExecuteInSQLPrepare

Key Name	Default Value	Required
NoExecuteInSQLPrepare	false	No

Description

This property specifies whether the connector executes queries other than DML and SELECT (such as DDL and PUT/GET) in the execution stage.

Set this property to one of the following values:

- `true`: The connector does not execute queries other than DML and Select in the execution stage.
- `false`: The connector executes queries other than DML and Select in the execution stage.

To configure this property as a connector-wide setting 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 Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `NoExecuteInSQLPrepare` as the value name, and `true` or `false` as the value data

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

OCSP_FAIL_OPEN

Key Name	Default Value	Required
<code>OCSP_FAIL_OPEN</code>	<code>true</code>	No

Description

This per-session option specifies how the connector responds during an Online Certificate Status Protocol (OCSP) event.

- `true`: The connector responds to the OCSP event with fail-open behavior. Fail-open behavior means that the connector only closes the connection if a certificate is revoked, and keeps the connection open for any other types of certificate errors
- `false`: The connector responds to the OCSP event with fail-close behavior. Fail-close behavior means that if the connector does not receive a valid OCSP CA response for any reason, the connector closes the connection.

To configure the connector's response for all connections, see [OCSPFailOpen](#) on page 74.

OCSPFailOpen

Key Name	Default Value	Required
<code>OCSPFailOpen</code>	<code>true</code>	No

Description

This connector-wide option specifies how the connector responds during an Online Certificate Status Protocol (OCSP) event.

- `true`: The connector responds to the OCSP event with fail-open behavior. Fail-open behavior means that the connector only closes the connection if a certificate is revoked, and keeps the connection open for any other types of certificate errors
- `false`: The connector responds to the OCSP event with fail-close behavior. Fail-close behavior means that if the connector does not receive a valid OCSP CA response for any reason, the connector closes the connection.

To configure the connector's response on a per-session basis, see [OCSP_FAIL_OPEN](#) on page 74.

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 Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `OCSPFailOpen` as the value name, and `true` or `false` as the value data.

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

Passcode

Key Name	Default Value	Required
Passcode	None	No

Description

The passcode to use for multi-factor authentication.

PasscodeInPassword

Key Name	Default Value	Required
PasscodeInPassword	false	No

Description

This property specifies whether the passcode for multi-factor authentication is appended to the password.

- `true`: The passcode is appended to the password.
- `false`: The passcode is not appended to the password.

Port

Key Name	Default Value	Required
Port	443	No

Description

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

PRIV_KEY_FILE

Key Name	Default Value	Required
PRIV_KEY_FILE	None	Yes, if Authenticator is set to <code>snowflake_jwt</code> .

Description

The local path to the private key file for key pair authentication.

This property can also be set through the Snowflake-specific SQL attribute `SQL_SF_CONN_ATTR_PRIV_KEY`. For more information, see the Snowflake documentation : <https://docs.snowflake.net/manuals/user-guide/odbc-api.html#snowflake-specific-behavior>.

PRIV_KEY_FILE_PWD

Key Name	Default Value	Required
PRIV_KEY_FILE_PWD	None	No

Description

The passcode to decode the private key file, if it is encrypted.

This property can also be set through the Snowflake-specific SQL attribute `SQL_SF_CONN_ATTR_PRIV_KEY`. For more information, see "Snowflake-Specific Behavior" in the Snowflake documentation : <https://docs.snowflake.net/manuals/user-guide/odbc-api.html#snowflake-specific-behavior>.

PUT_FASTFAIL

Key Name	Default Value	Required
PUT_FASTFAIL	false	No

Description

This property specifies the connector behavior if an error occurs when you are using wildcard characters with the PUT command to upload multiple files at once.

- `true`: The connector stops uploading when an error occurs.
- `false`: The connector continues to upload the rest of the files when an error occurs.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `PUT_FASTFAIL` as the value name, and `true` or `false` as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

PUT_MAXRETRIES

Key Name	Default Value	Required
PUT_MAXRETRIES	5	No

Description

The number of times that the connector should retry the PUT command if the command fails, from 0 to 100.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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 Snowflake ODBC Connector\Driver
- Otherwise: HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver

Use `PUT_MAXRETRIES` as the value name, and the number of retries as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

PUT_TEMPDIR

Key Name	Default Value	Required
PUT_TEMPDIR	/tmp/snowflakeTmp_<username>, where <username> is the username of the current user in the operating system.	No

Description

The temporary directory to use for PUT command requests. The connector uses this temporary directory to create temporary compressed files before uploading those files to Snowflake.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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 Snowflake ODBC Connector\Driver
- Otherwise: HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver

Use `PUT_TEMPDIR` as the value name, and the directory name as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

QUERY_TIMEOUT

Key Name	Default Value	Required
QUERY_TIMEOUT	0	No

Description

The length of time, in seconds, that the connector waits for a query to complete before returning an error. A value of 0 indicates that the connector might wait indefinitely.

SSL_Version

Key Name	Default Value	Required
SSL_Version	TLSv1_2	No

Description

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

- `SSLv2`: The connection must use SSLv2.
- `SSLv3`: The connection must use SSLv3.
- `TLSv1_0`: The connection must use at least TLS 1.0.
- `TLSv1_1`: The connection must use at least TLS 1.1.
- `TLSv1_2`: The connection must use at least TLS 1.2.

TIMEZONE

Key Name	Default Value	Required
TIMEZONE	America/Los_Angeles	No

Description

This property overwrites the Snowflake session parameter `TIMEZONE`. For more information, see the Snowflake documentation.

This property is available as both a per-session and a connector-wide setting.

To configure this property as a connector-wide setting 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\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

Use `TIMEZONE` as the value name, and an IANA time zone name as the value data.

To configure this property as a connector-wide setting for a non-Windows connector, you must use the `simba.snowflakeodbc.ini` file.

Token

Key Name	Default Value	Required
Token	None	Yes, if Authenticator is set to OAuth.

Description

The access token used for OAuth 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_simbasnowflakeodbcdriver.log` and `jdoe_7836_simbasnowflakeodbcdriver_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\SOFTWAREWow6432Node\Simba\Simba Snowflake ODBC Connector\Driver`
- Otherwise: `HKEY_LOCAL_MACHINE\SOFTWARE\Simba\Simba Snowflake ODBC Connector\Driver`

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

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

UseURandomDevice

Key Name	Default Value	Required
UseURandomDevice	true	No

Description

This connector-wide setting specifies the method used for generating the encryption key for file transfer (PUT or GET query).

- `true`: The driver uses `DEV_URANDOM`.
- `false`: The driver uses `DEV_RANDOM`.

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.

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.

Snowflake is a registered trademark of Snowflake Inc.

All other trademarks are trademarks of their respective owners.