



# Magnitude Simba Google BigQuery JDBC Data Connector

## Installation and Configuration Guide

Version 1.2.19

July 30, 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](http://www.magnitude.com)

## About This Guide

### Purpose

The *Magnitude Simba Google BigQuery JDBC Data Connector Installation and Configuration Guide* explains how to install and configure the Magnitude Simba Google BigQuery JDBC Data Connector on all supported platforms. The guide also provides details related to features of the connector.

### Audience

The guide is intended for end users of the Simba Google BigQuery JDBC Connector.

### Knowledge Prerequisites

To use the Simba Google BigQuery JDBC Connector, the following knowledge is helpful:

- Familiarity with the platform on which you are using the Simba Google BigQuery JDBC Connector
- Ability to use the data store to which the Simba Google BigQuery JDBC Connector is connecting
- An understanding of the role of JDBC technologies in connecting to a data store
- Experience creating and configuring JDBC 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 Google BigQuery JDBC Connector .....	7
System Requirements .....	8
Simba Google BigQuery JDBC Connector Files .....	9
Installing and Using the Simba Google BigQuery JDBC Connector .....	10
Referencing the JDBC Connector Libraries .....	10
Registering the Connector Class .....	11
Building the Connection URL .....	12
Configuring Authentication .....	14
Using a Google User Account .....	14
Using a Google Service Account .....	15
Using Pre-Generated Access and Refresh Tokens .....	16
Using Application Default Credentials .....	16
Configuring Logging .....	18
Features .....	20
SQL Connector .....	20
Data Types .....	20
Nested and Repeated Records .....	22
Security and Authentication .....	25
Catalog and Schema Support .....	26
Large Result Set Support .....	26
Dataset Locations .....	27
Write-Back .....	27
Positional Parameters .....	28
BigQuery Storage API .....	28
Connector Configuration Options .....	30
AdditionalProjects .....	30
AllowLargeResults .....	30
DefaultDataset .....	31
EnableHighThroughputAPI .....	31
FilterTablesOnDefaultDataset .....	32
HighThroughputActivationRatio .....	34
HighThroughputMinTableSize .....	34

KMSKeyName .....	34
LargeResultDataset .....	35
LargeResultsDatasetExpirationTime .....	36
LargeResultTable .....	36
Location .....	37
LogLevel .....	37
LogPath .....	38
MaxResults .....	39
MetaDataFetchThreadCount .....	39
OAuthAccessToken .....	39
OAuthClientId .....	40
OAuthClientSecret .....	40
OAuthPvtKey .....	40
OAuthPvtKeyPath .....	41
OAuthRefreshToken .....	41
OAuthServiceAcctEmail .....	41
OAuthType .....	42
ProjectId .....	42
ProxyHost .....	43
ProxyPort .....	43
ProxyPwd .....	43
ProxyUid .....	44
QueryDialect .....	44
QueryProperties .....	44
RequestGoogleDriveScope .....	45
SSLTrustStore .....	45
SSLTrustStorePwd .....	45
StringColumnLength .....	46
Timeout .....	46
TimestampFallback .....	46
useQueryCache .....	47
Third-Party Trademarks .....	48

## About the Simba Google BigQuery JDBC Connector

The Simba Google BigQuery JDBC Connector enables Business Intelligence (BI), analytics, and reporting on data that is stored in BigQuery. The connector complies with the JDBC 4.2 data standard.

JDBC 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 JDBC connector, which connects an application to the database. For more information about JDBC, see *Data Access Standards* on the Simba Technologies website: <https://www.simba.com/resources/data-access-standards-glossary>.

This guide is suitable for users who want to access data residing within BigQuery from their desktop environment. Application developers might also find the information helpful. Refer to your application for details on connecting via JDBC.

## System Requirements

Each machine where you use the Simba Google BigQuery JDBC Connector must have Java Runtime Environment (JRE) 8.0 or 11.0 installed.



## Simba Google BigQuery JDBC Connector Files

The Simba Google BigQuery JDBC Connector is delivered in a ZIP archive named `SimbaBigQueryJDBC42-[Version].zip`, where *[Version]* is the version number of the connector.

The archive contains the connector supporting the JDBC API version indicated in the archive name, as well as release notes and third-party license information.

## Installing and Using the Simba Google BigQuery JDBC Connector

To install the Simba Google BigQuery JDBC Connector on your machine, extract the files from the ZIP archive to the directory of your choice.

### ! Important:

If you received a license file through email, then you must copy the file into the same directory as the connector JAR file before you can use the Simba Google BigQuery JDBC Connector.

To access a BigQuery data store using the Simba Google BigQuery JDBC Connector, you need to configure the following:

- The list of connector library files (see [Referencing the JDBC Connector Libraries](#) on page 10)
- The `Driver` or `DataSource` class (see [Registering the Connector Class](#) on page 11)
- The connection URL for the connector (see [Building the Connection URL](#) on page 12)

## Referencing the JDBC Connector Libraries

Before you use the Simba Google BigQuery JDBC Connector, the JDBC application or Java code that you are using to connect to your data must be able to access the connector JAR files. In the application or code, specify all the JAR files that you extracted from the ZIP archive.

### Using the Connector in a JDBC Application

Most JDBC applications provide a set of configuration options for adding a list of connector library files. Use the provided options to include all the JAR files from the ZIP archive as part of the connector configuration in the application. For more information, see the documentation for your JDBC application.

### Using the Connector in Java Code

You must include all the connector library files in the class path. This is the path that the Java Runtime Environment searches for classes and other resource files. For more information, see "Setting the Class Path" in the appropriate Java SE Documentation.

- For Windows:  
<http://docs.oracle.com/javase/8/docs/technotes/tools/windows/classpath.html>
- For Linux and Solaris:  
<http://docs.oracle.com/javase/8/docs/technotes/tools/unix/classpath.html>

## Registering the Connector Class

Before connecting to your data, you must register the appropriate class for your application.

The following classes are used to connect the Simba Google BigQuery JDBC Connector to BigQuery data stores:

- The `Driver` classes extend `java.sql.Driver`.
- The `DataSource` classes extend `javax.sql.DataSource` and `javax.sql.ConnectionPoolDataSource`.

The connector supports the following fully-qualified class names (FQCNs) that are independent of the JDBC version:

- `com.simba.googlebigquery.jdbc.Driver`
- `com.simba.googlebigquery.jdbc.DataSource`

The following sample code shows how to use the `DriverManager` class to establish a connection for JDBC 4.2:

```
private static Connection connectViaDM() throws Exception
{
    Connection connection = null;
    connection = DriverManager.getConnection(CONNECTION_URL);
    return connection;
}
```

The following sample code shows how to use the `DataSource` class to establish a connection:

```
private static Connection connectViaDS() throws Exception
{
    Connection connection = null;
    DataSource ds = new
    com.simba.googlebigquery.jdbc.DataSource();
    ds.setURL(CONNECTION_URL);
    connection = ds.getConnection();
    return connection;
}
```

```
}
```

The following sample code shows how to use service authentication to establish a connection:

```
private static Connection connectViaDS() throws Exception
{
    Connection connection = null;
    DataSource ds = new
    com.simba.bigquery.jdbc.DataSource();
    ds.setURL(CONNECTION_URL);
    ds.setProjectId(PROJECT);
    ds.setOAuthType(0); // Service Authentication
    ds.setOAuthServiceAcctEmail(EMAIL);
    ds.setOAuthPvtKeyFile(KEYFILE);
    connection = ds.getConnection();
    return connection;
}
```

 **Note:**

When using the `DataSource` class to establish a connection, all the required properties of `CONNECTION_URL` should be configured. For service authentication, the properties can be configured separately.

## Building the Connection URL

Use the connection URL to supply connection information to the data store that you are accessing. The following is the format of the connection URL for the Simba Google BigQuery JDBC Connector:

```
jdbc:bigquery://[Host]:[Port];ProjectId=[Project];OAuthType=
[AuthValue];[Property1]=[Value1];[Property2]=[Value2];...
```

The variables are defined as follows:

- *[Host]* is the DNS or IP address of the server.
- *[Port]* is the number of the TCP port to connect to. Specifying the port number is optional if you are connecting to port 443.
- *[Project]* is the name of your BigQuery project.

- *[AuthValue]* is a number that specifies the type of authentication used by the connector. For more information, see [OAuthType](#) on page 42 and [Configuring Authentication](#) on page 14.
- *[Property1..N]* and *[Value1..N]* are additional connection properties supported by the connector. For a list of the properties available in the connector, see [Connector Configuration Options](#) on page 30.

**! Important:**

- Properties are case-sensitive.
- Do not duplicate properties in the connection URL.

## Configuring Authentication

The Simba Google BigQuery JDBC Connector uses the OAuth 2.0 protocol for authentication and authorization. It authenticates your connection through Google OAuth APIs. You can configure the connector to provide your credentials and authenticate the connection to the database using one of the following methods:

- [Using a Google User Account](#) on page 14
- [Using a Google Service Account](#) on page 15
- [Using Pre-Generated Access and Refresh Tokens](#) on page 16
- [Using Application Default Credentials](#) on page 16

### Using a Google User Account

You can configure the connector to authenticate the connection with a Google user account.

You must provide your Google user account credentials to connect to the server. For more information about authenticating through OAuth 2.0 with a Google user account, see "Using OAuth 2.0 to Access Google APIs" in the Google Identity Platform documentation: <https://developers.google.com/identity/protocols/OAuth2>.

**To configure user account authentication:**

1. Connect to the server using a connection URL written in the following format:

```
jdbc:bigquery://[Host]:[Port];ProjectId=[Project];
OAuthType=1;
```

The variables are defined as follows:

- *[Host]* is the DNS or IP address of the server.
- *[Port]* is the number of the TCP port to connect to. Specifying the port number is optional if you are connecting to port 443.
- *[Project]* is the name of your BigQuery project.

For example:

```
jdbc:bigquery
://https://www.googleapis.com/bigquery/v2:443;ProjectId=
MyBigQueryProject;OAuthType=1;
```

The connector returns a connection URL, and requests an access token.

2. In a web browser, navigate to the connection URL, and provide your Google account name and password for authentication.

The browser returns an access token.

3. In the connector, type or paste the access token and press **ENTER**.

For more information about connection URL syntax, see [Building the Connection URL](#) on page 12.

## Using a Google Service Account

You can configure the connector to authenticate the connection with a Google service account. The service account can handle the authentication process so that no user input is required.

You must provide a Google service account email address and the full path to a private key file for the service account. You can download the private key file from the Google API console web page. For more information about OAuth authentication using a service account, see "Using OAuth 2.0 for Server to Server Applications" in the Google Identity Platform documentation:

<https://developers.google.com/identity/protocols/OAuth2ServiceAccount>.

### To configure service account authentication:

1. Set the `OAuthType` property to 0.
2. Set the `ProjectID` property to the name of your BigQuery project.
3. Set the `OAuthServiceAcctEmail` property to your Google service account email address.
4. Set the `OAuthPvtKeyPath` property to the full path to the key file that is used to authenticate the service account email address. This parameter supports keys in `.p12` or `.json` format.

For example, the following connection URL authenticates the connection using a service account:

```
jdbc:bigquery://https://www.googleapis.com/bigquery/v2:443;  
ProjectId=MyBigQueryProject;OAuthType=0;  
OAuthServiceAcctEmail=bqtest1@data-driver-  
testing.iam.gserviceaccount.com;  
OAuthPvtKeyPath=C:\SecureFiles\ServiceKeyFile.p12;
```

For more information about connection URL syntax, see [Building the Connection URL](#) on page 12.

## Using Pre-Generated Access and Refresh Tokens

You can configure the connector to authenticate the connection using access or refresh tokens that have already been generated from the Google Authorization Server. When using this method, you can authenticate your connection by providing an access token, or by providing a refresh token along with a client ID and client secret.

For information about obtaining access and refresh tokens, see "Using OAuth 2.0 to Access Google APIs" in the Google Identity Platform documentation:

<https://developers.google.com/identity/protocols/OAuth2>.

### ! Important:

When generating the tokens to access BigQuery, you must specify the `https://www.googleapis.com/auth/bigquery` scope. If you are working with federated tables, you should also specify the `https://www.googleapis.com/auth/cloud-platform` scope.

To configure authentication using an access or refresh token:

1. Set the `OAuthType` property to 2.
2. Set the `ProjectID` property to the name of your BigQuery project.
3. Do one or both of the following:
  - Set `OAuthAccessToken` to your access token.
  - Or, set `OAuthRefreshToken` to your refresh token.
4. If you are using a refresh token, set the `OAuthClientId` property to your client ID and set the `OAuthClientSecret` property to your client secret.

For example, the following connection URL authenticates the connection using a refresh token:

```
jdbc:bigquery://https://www.googleapis.com/bigquery/v2:443;
OAuthType=2;ProjectId=MyBigQueryProject;
OAuthAccessToken=a25c7cfd36214f94a79d;OAuthRefreshToken=1jt9
Pcyq8pr3lvu143pfl4r86;OAuthClientId=11b5516f132211e6;OAuthCl
ientSecret=bCD+E1f2Gxhi3J4klmN;
```

For more information about connection URL syntax, see [Building the Connection URL](#) on page 12.

## Using Application Default Credentials

You can configure the connector to authenticate the connection using credentials obtained through Application Default Credentials on the environment, if they are



available. For information about how to configure Application Default Credentials, see "Google Application Default Credentials" in the Google Identity Platform documentation: <https://developers.google.com/identity/protocols/application-default-credentials>.

For more information about authenticating through OAuth 2.0, see "Using OAuth 2.0 to Access Google APIs" in the Google Identity Platform documentation: <https://developers.google.com/identity/protocols/OAuth2>.

**To configure authentication using Application Default Credentials:**

1. Set the `OAuthType` property to 3.
2. Set the `ProjectID` property to the name of your BigQuery project.

For example:

```
jdbc:bigquery://https://www.googleapis.com/bigquery/v2:443;  
OAuthType=3;ProjectId=MyBigQueryProject;
```

For more information about connection URL syntax, see [Building the Connection URL](#) on page 12.

## Configuring Logging

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

### ! Important:

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

The settings for logging apply to every connection that uses the Simba Google BigQuery JDBC Connector, so make sure to disable the feature after you are done using it.

In the connection URL, set the `LogLevel` key to enable logging at the desired level of detail. The following table lists the logging levels provided by the Simba Google BigQuery JDBC Connector, in order from least verbose to most verbose.

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

### To enable logging:

1. Set the `LogLevel` property to the desired level of information to include in log files.
2. Set the `LogPath` property to the full path to the folder where you want to save log files. To make sure that the connection URL is compatible with all

JDBC applications, escape the backslashes (\) in your file path by typing another backslash.

For example, the following connection URL enables logging level 3 and saves the log files in the `C:\temp` folder:

```
jdbc:bigquery://localhost;LogLevel=3;LogPath=C:\\temp
```

3. To make sure that the new settings take effect, restart your JDBC application and reconnect to the server.

The Simba Google BigQuery JDBC Connector produces the following log files in the location specified in the `LogPath` property:

- A `BigQuery_driver.log` file that logs connector activity that is not specific to a connection.
- A `BigQuery_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` value is invalid, then the connector sends the logged information to the standard output stream (`System.out`).

#### To disable logging:

1. Set the `LogLevel` property to 0.
2. To make sure that the new setting takes effect, restart your JDBC application and reconnect to the server.

## Features

More information is provided on the following features of the Simba Google BigQuery JDBC Connector:

- [SQL Connector](#) on page 20
- [Data Types](#) on page 20
- [Nested and Repeated Records](#) on page 22
- [Security and Authentication](#) on page 25
- [Catalog and Schema Support](#) on page 26
- [Large Result Set Support](#) on page 26
- [Dataset Locations](#) on page 27
- [Write-Back](#) on page 27
- [Positional Parameters](#) on page 28
- [BigQuery Storage API](#) on page 28

## SQL Connector

The SQL Connector feature of the connector enables applications to execute standard SQL queries or legacy BigQuery SQL queries against the database.

The connector does not support query prefixes, and instead determines which dialect to use based on the `QueryDialect` connection setting. By default, `QueryDialect` is set to `SQL` so that the connector executes queries using standard SQL syntax. For more information, see [QueryDialect](#) on page 44.


## Data Types

The Simba Google BigQuery JDBC Connector supports many common data formats, converting between BigQuery, SQL, and Java data types.

The following table lists the supported data type mappings.

BigQuery Type	SQL Type	Java Type
ARRAY	SQL_VARCHAR	STRING

BigQuery Type	SQL Type	Java Type
BIGNUMERIC For BIGNUMERIC data, the connector always returns 77 for the precision and 38 for the scale.	SQL_NUMERIC	BIGDECIMAL
BOOL	SQL_BOOLEAN	BOOLEAN
BYTES	SQL_VARBINARY	BYTE[]
DATE	SQL_DATE	DATE
DATETIME	SQL_TIMESTAMP	STRING
FLOAT64	SQL_DOUBLE	DOUBLE
GEOGRAPHY (See note below)	SQL_VARCHAR	STRING
INT64	SQL_BIGINT	BIGINTEGER
NUMERIC For NUMERIC data, the connector always returns 38 for the precision and 9 for the scale.	SQL_NUMERIC	BIGDECIMAL
STRING	SQL_VARCHAR	STRING
STRUCT	SQL_VARCHAR	STRING
TIME	SQL_TIME	TIME
TIMESTAMP	SQL_TIMESTAMP	TIMESTAMP

 **Note:**

GEOGRAPHY data cannot be used for a GEOGRAPHY column parameter. To insert or filter on GEOGRAPHY data, you must use the generating function described in "Geography Functions in Standard SQL" in the Google BigQuery documentation: [https://cloud.google.com/bigquery/docs/reference/standard-sql/geography\\_functions](https://cloud.google.com/bigquery/docs/reference/standard-sql/geography_functions).

## Nested and Repeated Records

The Simba Google BigQuery JDBC Connector fully supports nested and repeated records. The connector returns the base type as a text representation of the JSON object.

### Querying STRUCT Data

Standard SQL syntax represents the sub-components of record data as nested sub-types. The dot operator (.) is used to select sub-components. In the examples below, `city` and `years` belong to the base record type of `address`.

If the record column is specified in a query projection list, the connector returns the base record as a text representation of the JSON record object, and no flattening occurs.

See the following examples to see how to retrieve base records and sub-components from STRUCT data.

### Selecting Base Record

Sample query to retrieve a base record from a STRUCT column:

```
select (STRUCT("Vancouver" as city, 5 as years)) as address
```

The connector returns the results as a text reinterpretation of the JSON object, as shown here:

```
{
  "v": {
    "f": [
      {
        "v": "Vancouver"
      },
      {
        "v": "5"
      }
    ]
  }
}
```

```

    ]
  }
}

```

The BigQuery console would represent the query results as a table, as shown here:

Row	f0_.city	f0_.years
1	Vancouver	5

### Select Sub-Components

Sample query to retrieve a sub-component from a STRUCT column:

```

select address.city from (select (STRUCT("Vancouver" as city,
5 as years)) as address)

```

The connector returns the results as a text reinterpretation of the JSON object, as shown here:

```

[
  {
    "city": "Vancouver"
  }
]

```

The BigQuery console would represent the query results as a table, as shown here:

Row	city
1	Vancouver

### Querying Arrays

The Simba Google BigQuery JDBC Connector fully supports array data types. The connector returns the base array type as a text representation of the JSON array object.

#### Selecting Arrays of Primitive Type

Sample query to select a primitive array:

```

SELECT [1,2,3]

```

The connector returns the results as a text reinterpretation of the JSON object, as shown here:

```
{
  "v": [
    {
      "v": "1"
    },
    {
      "v": "2"
    },
    {
      "v": "3"
    }
  ]
}
```

The BigQuery console would represent the query results as a table, as shown here:

Row	f0_
1	1
	2
	3

## Selecting Arrays of STRUCT Data

Sample query to select multiple objects from a STRUCT array:

```
SELECT [STRUCT("Vancouver" as city, 5 as years), STRUCT
("Boston" as city, 10 as years)]
```

The connector returns the results as a text reinterpretation of the JSON object, as shown here:

```
{
  "v": [
    {
      "v": {
        "f": [
          {
            "v": "Vancouver"
          },
          {

```



```

    "v": "5"
  }
]
},
{
  "v": {
    "f": [
      {
        "v": "Boston"
      },
      {
        "v": "10"
      }
    ]
  }
}
]
}
}

```

The BigQuery console would represent the query results as a table, as shown here:

Row	f0_.city	f0_.years
1	Vancouver	5
	Boston	10

## Security and Authentication

To protect data from unauthorized access, BigQuery data stores require all connections to be authenticated using the OAuth 2.0 protocol. The Simba Google BigQuery JDBC Connector provides mechanisms that allow you to complete an OAuth 2.0 authentication flow using a personal Google account, a Google service account, or Application Default Credentials. You can also specify an access token or refresh token that you have already generated from the Google Authorization Server, and use those credentials to connect to Google BigQuery.

When you connect to BigQuery using a personal Google account, a Google service account, or Application Default Credentials, the connector automatically initiates an OAuth 2.0 authentication flow. The connector retrieves an access token based on the credentials specified in the connection URL, and then uses the token to authenticate

the connection to the database. When you connect using an access token or refresh token, the connector authenticates the connection to BigQuery without going through an OAuth 2.0 authentication flow.

For detailed connector configuration instructions, see [Configuring Authentication](#) on page 14.

For more information about OAuth 2.0, see "Using OAuth 2.0 to Access Google APIs" in the Google Identity Platform documentation:

<https://developers.google.com/identity/protocols/OAuth2>.

## Catalog and Schema Support

The Simba Google BigQuery JDBC Connector supports both catalogs and schemas to make it easy for the connector to work with various JDBC applications. Projects are mapped to catalogs, and table datasets are mapped to schemas. For more information, see [ProjectId](#) on page 42. The connector provides access to all of the schemas/databases that are listed under this catalog, ensuring compatibility with standard BI tools.

## Large Result Set Support

BigQuery imposes a maximum response size on all requests. If you do not enable large result set support, when executing queries, you might encounter an error message such as "Response too large to return".

If you expect your query to return a large result set, do the following to make sure that the query results can be returned as expected:

- If you are using standard SQL, specify a destination dataset and table for storing the query results. To do this, set the `LargeResultDataset` and `LargeResultTable` properties, respectively.
- If you are using legacy SQL, enable support for large result sets, and then specify a destination dataset and table for storing the query results. To do this, enable the `AllowLargeResults` property, and then set the `LargeResultDataset` and `LargeResultTable` properties to the destination dataset and table, respectively.

**! Important:**

- When the `LargeResultDataset` and `LargeResultTable` properties are set, all query results are written to and read from the those tables regardless of the query and its result size. Because of this, the result cache is not available for subsequent queries, and you are billed for every query that you make.
- Working with large data sets may cause you to reach the query limits defined in Google BigQuery. For information about query limits, see "Quotas & Limits" in the Google BigQuery documentation: <https://cloud.google.com/bigquery/quotas>.

In both standard and legacy SQL, if you do not specify a destination dataset or table, the connector stores large result sets in a temporary, default location. The default dataset is hidden, and is named "\_simba\_jdbc". The default table has a name consisting of the prefix "temp\_table" followed by the time of table creation and a randomized ID. These datasets and tables are deleted after 24 hours.

## Dataset Locations

The Simba Google BigQuery JDBC Connector supports auto-routing for regional dataset locations. If multiple datasets are available in different geographic regions, the connector automatically queries a dataset in the correct region.

For more information about dataset locations, see "Dataset Locations" in the Google BigQuery documentation: <https://cloud.google.com/bigquery/docs/locations>.

**! Important:**

If you are using a large result set, make sure that the `LargeResultDataset` option is specified, and that the specified dataset is in the same region as the queried table.

## Write-Back

The Simba Google BigQuery JDBC Connector supports Data Manipulation Language (DML) statements such as INSERT, UPDATE, and DELETE.

For example, the following INSERT statement is supported:

```
INSERT INTO MyTable (Col1, Col2) VALUES ("Key", "Value");
```

The connector also supports Data Definition Language (DDL) statements. Be aware that BigQuery supports specific syntax for DDL statements, and your statements must

be written in that syntax. For more information, see "Using Data Definition Language Statements" in Google BigQuery's *Standard SQL Query Reference*:  
<https://cloud.google.com/bigquery/docs/data-definition-language>.

## Positional Parameters

A parameterized query contains placeholders that are used for parameters. The values of those parameters are supplied at execution time.

Query parameters can be used as substitutes for arbitrary expressions. Parameters cannot be used as substitutes for identifiers, column names, table names, or other parts of the query.

The Simba Google BigQuery JDBC Connector supports SQL positional parameters. Parameters are specified in queries with a question mark (?).

For example, the following parameterized query is supported:

```
SELECT * FROM MyTable WHERE Col1=?
```

## BigQuery Storage API

The connector can leverage the BigQuery Storage API, which allows higher throughput than the standard API. This enables the connector to handle large result sets more efficiently. For more information about the API, see "BigQuery Storage API Overview" in the Google BigQuery documentation:  
<https://cloud.google.com/bigquery/docs/reference/storage/>.

If this feature is enabled, the connector checks the number of rows in an incoming result set table and the number of pages needed to retrieve all the results. If the number of rows and pages exceeds the defined threshold, the connector switches to using the BigQuery Storage API. If the connector encounters any issues initializing the storage API for retrieval, it falls back to using the standard API, unless this is a permissions issue. To ensure best performance, do not use this feature with a named destination dataset or table.

You can customize the thresholds for using the BigQuery Storage API. For information about the configuration options used to determine when the API is used, see the following:

- [EnableHighThroughputAPI](#) on page 31
- [HighThroughputActivationRatio](#) on page 34
- [HighThroughputMinTableSize](#) on page 34

In order to take advantage of this feature, the BigQuery project you are querying must have the BigQuery Storage API enabled. For more information, see "Enabling the API" in the Google BigQuery documentation:

[https://cloud.google.com/bigquery/docs/reference/storage/#enabling\\_the\\_api](https://cloud.google.com/bigquery/docs/reference/storage/#enabling_the_api).

**! Important:**


Pricing for the BigQuery Storage API is different than pricing for the standard API. For more information, see "BigQuery Storage API Pricing" in the Google BigQuery documentation: <https://cloud.google.com/bigquery/pricing#storage-api>.

Using the large result set support feature can reduce some of the performance gains from the BigQuery Storage API. If you enable the BigQuery Storage API, we recommend setting `AllowLargeResults` to 0.

## Connector Configuration Options

Connector Configuration Options lists and describes the properties that you can use to configure the behavior of the Simba Google BigQuery JDBC Connector.

You can set configuration properties using the connection URL. For more information, see [Building the Connection URL](#) on page 12.

 **Note:**

Property names and values are case-sensitive.

### AdditionalProjects

Default Value	Data Type	Required
None	String	No

#### Description

A comma-separated list of BigQuery projects that the connector can access and use as catalogs. These projects are available as catalogs in metadata functions.

### AllowLargeResults

Default Value	Data Type	Required
0	Integer	No

#### Description

This option specifies whether the connector supports query results larger than 128MB when working in legacy SQL (the `QueryDialect` property is set to `BIG_QUERY`).

- 1: The connector allows query results that are larger than 128MB in size.
- 0: The connector returns an error when query results are larger than 128MB in size.

When working in standard SQL (the `QueryDialect` property is set to `SQL`), this option is always considered to be enabled. For more information about the supported SQL dialects, see [QueryDialect](#) on page 44.

**! Important:**

When this property is enabled for legacy SQL, all query results are written to and read from the destination tables, regardless of the query and its result size. Because of this, the result cache is not available to subsequent queries, and you are billed for every query that you make.

For detailed information about how the connector stores large result sets, see [Large Result Set Support](#) on page 26.

## DefaultDataset

Default Value	Data Type	Required
None	String	No

### Description

The name of a dataset that the connector queries by default.

Specifying a default dataset enables you to use unqualified table names in SQL statements. The connector treats unqualified tables as part of the default dataset. Additionally, it treats the default dataset as part of the project that is being billed. For information about specifying the project to bill, see [ProjectId](#) on page 42.

## EnableHighThroughputAPI

Default Value	Data Type	Required
0	Integer	No

### Description

This option specifies whether the connector uses the BigQuery Storage API for large result sets.

- 1: The connector uses the Storage API for result sets the exceed the activation ratio.
- 0: The connector does not use the Storage API.

Be aware that the storage API must be enabled for the BigQuery project you are querying.

### ! Important:

Pricing for the storage API is different than pricing for the standard API. For more information, see "BigQuery Storage API Pricing" in the Google BigQuery documentation: <https://cloud.google.com/bigquery/pricing#storage-api>.

For information on the BigQuery Storage API, see [BigQuery Storage API](#) on page 28.

For information on customizing the activation ratio, see [HighThroughputMinTableSize](#) on page 34 and [HighThroughputActivationRatio](#) on page 34.

## FilterTablesOnDefaultDataset

Default Value	Data Type	Required
0	Boolean	No

### Description

This option determines whether the connector filters tables in the `DatabaseMetaData.getTables` call and columns in the `DatabaseMetaData.getColumns` call to return only tables and columns that belong to the default dataset.

- 0: The connector returns all tables in the `DatabaseMetaData.getTables` call and all columns in the `DatabaseMetaData.getColumns` call.
- 1: The connector only returns tables and columns that belong to the default dataset.

### Note:

To filter tables and columns, you must define a default dataset. For more information, see [DefaultDataset](#) on page 31.

When this option is set to 1, the connector behaves as described below for the functions `DatabaseMetaData.getTables` and `DatabaseMetaData.getColumns`.



For the function `DatabaseMetaData.getTables`:

Catalog	Schema	Table	Table Type	Returned List
NULL or %	NULL or %	NULL or %	NULL or %	All tables that belong to the default dataset under the default catalog
NULL or %	<i>[schema]</i>	NULL or %	NULL or %	All tables that belong to the specified schema under all catalogs
<i>[catalog]</i>	NULL or %	NULL or %	NULL or %	All tables that belong to the default dataset under the specified catalog
<i>[catalog]</i>	<i>[schema]</i>	NULL or %	NULL or %	All tables that belong to the specified schema under the specified catalog

For the function `DatabaseMetaData.getColumns`:

Catalog	Schema	Table	Column	Returned List
NULL or %	NULL or %	NULL or %	NULL or %	All columns of all tables that belong to the default dataset under the default catalog
NULL or %	<i>[schema]</i>	NULL or %	NULL or %	All columns of all tables that belong to the specified dataset under all catalogs
<i>[catalog]</i>	NULL or %	NULL or %	NULL or %	All columns of all tables that belong to the default dataset under the specified catalog
<i>[catalog]</i>	<i>[schema]</i>	NULL or %	NULL or %	All columns of all tables that belong to the specified dataset under the specified catalog

## HighThroughputActivationRatio

Default Value	Data Type	Required
3	Integer	No

### Description

When the number of pages in your query results exceeds this number, and the number of rows in the results exceeds the `HighThroughputMinTableSize` value, the connector switches to using the BigQuery Storage API instead of the standard API.

If you define this, you must also set the `EnableHighThroughputAPI` option to 1. For more information, see [EnableHighThroughputAPI](#) on page 31.

## HighThroughputMinTableSize

Default Value	Data Type	Required
100	Integer	No

### Description

When the number of table rows in your query results exceeds this number, and the number of pages in the results exceeds the `HighThroughputRatio` value, the connector switches to using the BigQuery Storage API instead of the standard API.

If you define this, you must also set the `EnableHighThroughputAPI` option to 1. For more information, see [EnableHighThroughputAPI](#) on page 31.

## KMSKeyName

Default Value	Data Type	Required
None. The connector uses the default encryption key from Google.	String	No

## Description

The key name of the customer-managed encryption key (CMEK) that you want the connector to use when executing queries. When this property is not set, the connector uses the default encryption key from Google.

For information about CMEKs and Cloud KMS encryption, see "Protecting Data with Cloud KMS Keys" in the Google BigQuery documentation:

<https://cloud.google.com/bigquery/docs/customer-managed-encryption>.

### ! Important:

- Do not set this property unless you are certain that you are specifying the correct CMEK. If you execute an INSERT statement with an incorrect CMEK, the connector returns an error or corrupts the table.
- When this property is set, the connector uses the specified CMEK for all queries.

## LargeResultDataset

Default Value	Data Type	Required
<p><code>_simba_jdbc</code>, if <code>QueryDialect</code> is set to <code>BIG_QUERY</code>, or if <code>QueryDialect</code> is set to <code>SQL</code> and <code>LargeResultTable</code> is specified.</p> <p>None, if <code>QueryDialect</code> is set to <code>SQL</code> and no value is provided for <code>LargeResultTable</code>.</p>	String	No

## Description

A persistent destination dataset for storing query results. For more information, see [Large Result Set Support](#) on page 26.

**! Important:**

- If you specify a persistent destination table and dataset, all query results are written to and read from the destination tables regardless of the query and its result size. Because of this, the result cache is not available for subsequent queries, and you are billed for every query that you make.
- If you are using different datasets for different locations, make sure that this option is specified, and that the specified dataset is in the same region as the queried table.

If `QueryDialect` is set to `BIG_QUERY`, this option is only used when the `AllowLargeResults` property is enabled.

## LargeResultsDatasetExpirationTime

Default Value	Data Type	Required
3600000	Long	No

### Description

The length of time, in milliseconds, before the tables in a user-specified large result dataset expire.

This expiration time is applied to all tables created in this dataset.

## LargeResultTable

Default Value	Data Type	Required
<p>A value consisting of the prefix <code>temp_table_</code> followed by the time of table creation, if <code>QueryDialect</code> is set to <code>BIG_QUERY</code>, or if <code>QueryDialect</code> is set to <code>SQL</code> and <code>LargeResultDatasetSet</code> is specified.</p> <p>None, if <code>QueryDialect</code> is set to <code>SQL</code> and no value is provided for <code>LargeResultDataSet</code>.</p>	String	No

## Description

A persistent destination dataset for storing query results. For more information, see [Large Result Set Support](#) on page 26.

### ! Important:

If you specify a persistent destination table and dataset, all query results are written to and read from the destination tables regardless of the query and its result size. Because of this, the result cache is not available for subsequent queries, and you are billed for every query that you make.

If `QueryDialect` is set to `BIG_QUERY`, this option is only used when the `AllowLargeResults` property is enabled.

## Location

Default Value	Data Type	Required
None	String	No

## Description

The location where the BigQuery datasets are stored. If this property is specified, the connector can only query datasets that are in this location.

For a list of locations, see "Dataset Locations" in the Google BigQuery documentation: <https://cloud.google.com/bigquery/docs/locations>.

## LogLevel

Default Value	Data Type	Required
0	Integer	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 Google BigQuery JDBC Connector, so make sure to disable the feature after you are done using it.

Set the property to one of the following numbers:

- 0: Disable all logging.
- 1: Enable logging on the FATAL level, which logs very severe error events that will lead the connector to abort.
- 2: Enable logging on the ERROR level, which logs error events that might still allow the connector to continue running.
- 3: Enable logging on the WARNING level, which logs events that might result in an error if action is not taken.
- 4: Enable logging on the INFO level, which logs general information that describes the progress of the connector.
- 5: Enable logging on the DEBUG level, which logs detailed information that is useful for debugging the connector.
- 6: Enable logging on the TRACE level, which logs all connector activity.

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

- A `BigQuery_driver.log` file that logs connector activity that is not specific to a connection.
- A `BigQuery_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` value is invalid, then the connector sends the logged information to the standard output stream (`System.out`).

## LogPath

Default Value	Data Type	Required
The current working directory	String	No

## Description

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



### Note:

To make sure that the connection URL is compatible with all JDBC applications, it is recommended that you escape the backslashes (\) in your file path by typing another backslash.

## MaxResults

Default Value	Data Type	Required
10000	Integer	No

## Description

The maximum number of results that are displayed per page.

## MetaDataFetchThreadCount

Default Value	Data Type	Required
32	Integer	No

## Description

The number of threads used to call a `DatabaseMetaData` method.

## OAuthAccessToken

Default Value	Data Type	Required
None	String	No

## Description

The pre-generated access token you are using to authenticate into BigQuery. For details, see [Using Pre-Generated Access and Refresh Tokens](#) on page 16.

## OAuthClientId

Default Value	Data Type	Required
None	String	Yes, if using a refresh token.

## Description

The Client ID you are using to authenticate into BigQuery with a pre-generated refresh token. For details, see [Using Pre-Generated Access and Refresh Tokens](#) on page 16.

## OAuthClientSecret

Default Value	Data Type	Required
None	String	Yes, if using a refresh token.

## Description

The client secret you are using to authenticate into BigQuery with a pre-generated refresh token. For details, see [Using Pre-Generated Access and Refresh Tokens](#) on page 16.

## OAuthPvtKey

Default Value	Data Type	Required
None	String	No



## Description

The keyfile used for a service account. This can be a path to the `.p12` or `.json` keyfile, or a raw JSON keyfile object.

## OAuthPvtKeyPath

Default Value	Data Type	Required
None	String	Yes, if <code>OAuthUserAuth=0</code> .

## Description

The full path to the `.p12` or `.json` key file that is used to authenticate the service account email address, if you are authenticating your connection using a service account. For more information, see [Using a Google Service Account](#) on page 15.

## OAuthRefreshToken

Default Value	Data Type	Required
None	String	No

## Description

The pre-generated refresh token you are using to authenticate into BigQuery. For details, see [Using Pre-Generated Access and Refresh Tokens](#) on page 16.

## OAuthServiceAcctEmail

Default Value	Data Type	Required
None	String	Yes, if <code>OAuthUserAuth=0</code> .

## Description

The service account email ID that is used for authentication if you are using service authentication. For more information, see [Using a Google Service Account](#) on page 15.

## OAuthType

Default Value	Data Type	Required
0	Integer	No

## Description

This option specifies how the connector obtains or provides the credentials for OAuth 2.0 authentication.

- 0: The connector uses service-based OAuth authentication (see [Using a Google Service Account](#) on page 15).
- 1: The connector uses user-based OAuth authentication (see [Using a Google User Account](#) on page 14).
- 2: The connector uses pre-generated tokens for authentication (see [Using Pre-Generated Access and Refresh Tokens](#) on page 16).
- 3: The connector uses Application Default Credentials for authentication (see [Using Application Default Credentials](#) on page 16).

## ProjectId

Default Value	Data Type	Required
None	String	Yes

## Description

The name of your BigQuery project. This project is the default project that the Simba Google BigQuery JDBC Connector queries against, and also the project that is billed for queries that are run using the DSN.

## ProxyHost

Default Value	Data Type	Required
None	String	No

### Description

The IP address or host name of your proxy server.

## ProxyPort

Default Value	Data Type	Required
None	Integer	No

### Description

The listening port of your proxy server.

## ProxyPwd

Default Value	Data Type	Required
None	String	No

### Description

The password, if needed, for proxy server settings.

When using a proxy that requires credentials, the following JVM arguments must be used:

- `Djdk.http.auth.tunneling.disabledSchemes=`
- `Djdk.http.auth.proxying.disabledSchemes=`

## ProxyUid

Default Value	Data Type	Required
None	String	No

### Description

The user name, if needed, for proxy server settings.

When using a proxy that requires credentials, the following JVM arguments must be used:

- `Djdk.http.auth.tunneling.disabledSchemes=`
- `Djdk.http.auth.proxying.disabledSchemes=`

## QueryDialect

Default Value	Data Type	Required
SQL	Enumerated	No

### Description

This option specifies whether the connector executes queries using standard SQL syntax or the legacy BigQuery SQL syntax.

- `SQL`: The connector uses standard SQL.
- `BIG_QUERY`: The connector uses legacy SQL.

## QueryProperties

Default Value	Data Type	Required
<code>QueryProperties</code>	None	No

### Description

This option allows you to create a comma-separated list of key-value pairs, which will be passed through to the server when inserting a job. Properties set in this manner are

used for all queries in a connection. As a comma delimited list, QueryProperties must be of the following form:

```
key1=value1, key2=value2, . . . , keyN=valueN.
```

## RequestGoogleDriveScope

Default Value	Data Type	Required
0	Integer	No

### Description

This option specifies whether the connector requests access to Google Drive. Allowing the connector to access Google Drive enables support for federated tables that combine BigQuery data with data from Google Drive.

- 0: The connector does not request access to Google Drive.
- 1: The connector requests access to Google Drive.

## SSLTrustStore

Default Value	Data Type	Required
None	String	No

### Description

The full path of the Java TrustStore containing the server certificate for one-way SSL authentication.

If the trust store requires a password, provide it using the property `SSLTrustStorePwd`. See [SSLTrustStorePwd](#) on page 45.

## SSLTrustStorePwd

Default Value	Data Type	Required
None	String	Yes, if using a TrustStore.

## Description

The password for accessing the Java TrustStore that you specified using the property [SSLTrustStore](#) on page 45.

## StringColumnLength

Default Value	Data Type	Required
65535	Long	No

## Description

The maximum length for string type columns.

## Timeout

Default Value	Data Type	Required
10	Integer	No

## Description

The length of time, in seconds, that the connector waits for a query to retrieve the results of an executed job.

## TimestampFallback

Default Value	Data Type	Required
0	Integer	No

## Description

### ! Important:

This connection property is deprecated. Only enable this option if you need to temporarily support connections that used connector version 1.1.0 or earlier, while transitioning your applications to connect using connector version 1.1.1 or later.

This property specifies whether the connector sets the default timezone of the JVM to UTC.

- 1: The connector sets the default timezone of the JVM to UTC. This behavior is non-optimal, but consistent with the behavior from connector versions 1.1.0 and earlier.

 **Note:**

Before enabling `TimestampFallback`, be aware of the following:

- This property applies to all connections that use the Simba Google BigQuery JDBC Connector.
- Enabling this property causes the connector to change JVM settings, affecting other processes that are running on the same JVM.
- Once you have made a connection with this property enabled, all subsequent connections must also have this property enabled. Otherwise, the connector returns an error.

- 0: The connector does not change the timezone settings on the JVM. This is the preferred connector behavior.

## useQueryCache

Default Value	Data Type	Required
1	Integer	No

### Description

This option specifies whether the connector uses cached query results.

- 1: The connector uses cached query results.
- 0: The connector does not use cached query results.

## Third-Party Trademarks

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Google BigQuery, Google, and BigQuery are trademarks or registered trademarks of Google, Inc. or its subsidiaries in Canada, the United States and/or other countries.

All other trademarks are trademarks of their respective owners.