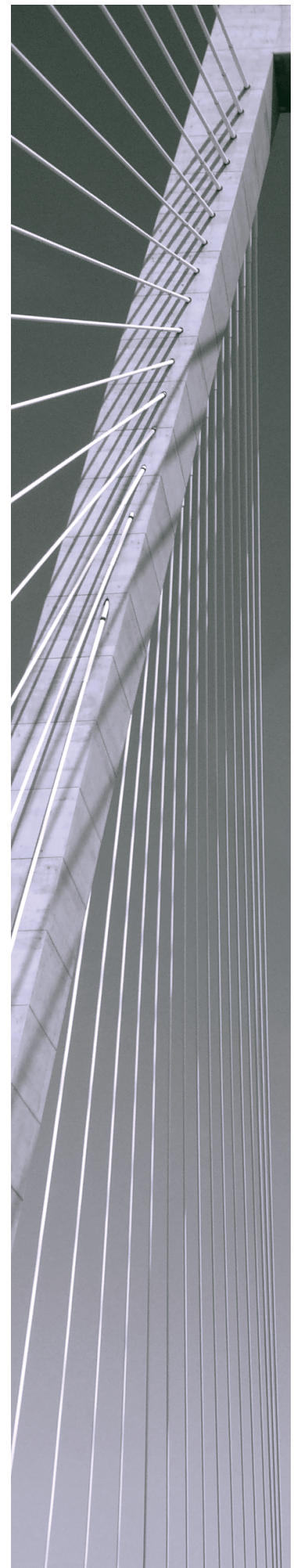# Simba JDBC Driver with SQL Connector for Google BigQuery

Installation and Configuration Guide

Simba Technologies Inc.

Version 1.2.13

February 1, 2021

Contact Us

Simba Technologies Inc.
938 West 8th Avenue
Vancouver, BC Canada
V5Z 1E5

Tel: +1 (604) 633-0008

Fax: +1 (604) 633-0004

www.simba.com

About This Guide

Purpose

The *Simba JDBC Driver with SQL Connector for Google BigQuery Installation and Configuration Guide* explains how to install and configure the Simba JDBC Driver with SQL Connector for Google BigQuery on all supported platforms. The guide also provides details related to features of the driver.

Audience

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

Knowledge Prerequisites

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

- Familiarity with the platform on which you are using the Simba JDBC Driver for Google BigQuery
- Ability to use the data store to which the Simba JDBC Driver for Google BigQuery 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 JDBC Driver for Google BigQuery

The Simba JDBC Driver for Google BigQuery enables Business Intelligence (BI), analytics, and reporting on data that is stored in BigQuery. The driver 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 driver, 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 JDBC Driver for Google BigQuery must have
Java Runtime Environment (JRE) 8.0 or 11.0 installed.

Simba JDBC Driver for Google BigQuery Files

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

The archive contains the driver 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 JDBC Driver for Google BigQuery

To install the Simba JDBC Driver for Google BigQuery 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 driver JAR file before you can use the Simba JDBC Driver for Google BigQuery.

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

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

Referencing the JDBC Driver Libraries

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

Using the Driver in a JDBC Application

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

Using the Driver in Java Code

You must include all the driver 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 Driver Class

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

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

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

The driver 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;
```

```
}
```

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 JDBC Driver for Google BigQuery:

```
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 driver. For more information, see OAuthType on page 40 and Configuring Authentication on page 13.
- *[Property1..N]* and *[Value1..N]* are additional connection properties supported by the driver. For a list of the properties available in the driver, see Driver Configuration Options on page 29.

! **Important:**

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

Configuring Authentication

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

- Using a Google User Account on page 13
- Using a Google Service Account on page 14
- Using Pre-Generated Access and Refresh Tokens on page 15
- Using Application Default Credentials on page 15

Using a Google User Account

You can configure the driver 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/v2:4
43;ProjectId=MyBigQueryProject;OAuthType=1;
```

The driver 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 driver, 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 driver 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 driver 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 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 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

Using Application Default Credentials

You can configure the driver 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 driver.

> ❗ **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 JDBC Driver for Google BigQuery, 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 JDBC Driver for Google BigQuery, in order from least verbose to most verbose.

LogLevel Value	Description
0	Disable all logging.
1	Log severe error events that lead the driver to abort.
2	Log error events that might allow the driver 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 driver.
5	Log detailed information that is useful for debugging the driver.
6	Log all driver 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 JDBC Driver for Google BigQuery produces the following log files in the location specified in the `LogPath` property:

- A `BigQuery_driver.log` file that logs driver 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 driver activity that is specific to the connection.

If the `LogPath` value is invalid, then the driver 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 JDBC Driver for Google BigQuery:

SQL Connector

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

The driver 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 driver executes queries using standard SQL syntax. For more information, see QueryDialect on page 42.

Data Types

The Simba JDBC Driver for Google BigQuery 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 driver 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 driver 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.

Nested and Repeated Records

The Simba JDBC Driver for Google BigQuery fully supports nested and repeated records. The driver 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 driver 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 driver 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 driver 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 JDBC Driver for Google BigQuery fully supports array data types. The driver 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 driver 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 driver 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 JDBC Driver for Google BigQuery 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 driver automatically initiates an OAuth 2.0 authentication flow. The driver 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 driver authenticates the connection to BigQuery without going through an OAuth 2.0 authentication flow.

For detailed driver configuration instructions, see Configuring Authentication on page 13.

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 JDBC Driver for Google BigQuery supports both catalogs and schemas to make it easy for the driver 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 40. The driver 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 driver 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. These datasets and tables are deleted after 24 hours.

Dataset Locations

The Simba JDBC Driver for Google BigQuery supports auto-routing for regional dataset locations. If multiple datasets are available in different geographic regions, the driver 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 JDBC Driver for Google BigQuery 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 driver 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 JDBC Driver for Google BigQuery 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 driver can leverage the BigQuery Storage API, which allows higher throughput than the standard API. This enables the driver 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 driver 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 driver switches to using the BigQuery Storage API. If the driver 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 30
- HighThroughputActivationRatio on page 32
- HighThroughputMinTableSize on page 33

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.

> **! 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`.

Driver Configuration Options

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

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 driver 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 driver supports query results larger than 128MB when working in legacy SQL (the `QueryDialect` property is set to `BIG_QUERY`).

- `1`: The driver allows query results that are larger than 128MB in size.
- `0`: The driver 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 42.

> **! 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 driver stores large result sets, see Large Result Set Support on page 25.

DefaultDataset

Default Value	Data Type	Required
None	String	No

Description

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

Specifying a default dataset enables you to use unqualified table names in SQL statements. The driver 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 40.

EnableHighThroughputAPI

Default Value	Data Type	Required
0	Integer	No

Description

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

- `1`: The driver uses the Storage API for result sets the exceed the activation ratio.
- `0`: The driver 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 27.

For information on customizing the activation ratio, see HighThroughputMinTableSize on page 33 and HighThroughputActivationRatio on page 32.

FilterTablesOnDefaultDataset

Default Value	Data Type	Required
0	Boolean	No

Description

This option determines whether the driver 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 driver returns all tables in the `DatabaseMetaData.getTables` call and all columns in the `DatabaseMetaData.getColumns` call.
- `1`: The driver 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 30.

When this option is set to `1`, the driver 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 %	[schema]	NULL or %	NULL or %	All tables that belong to the specified schema under all catalogs
[catalog]	NULL or %	NULL or %	NULL or %	All tables that belong to the default dataset under the specified catalog
[catalog]	[schema]	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 %	[schema]	NULL or %	NULL or %	All columns of all tables that belong to the specified dataset under all catalogs
[catalog]	NULL or %	NULL or %	NULL or %	All columns of all tables that belong to the default dataset under the specified catalog
[catalog]	[schema]	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 driver 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 30.

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 `HighThroughPuRatio` value, the driver 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 30.

KMSKeyName

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

Description

The key name of the customer-managed encryption key (CMEK) that you want the driver to use when executing queries. When this property is not set, the driver 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 driver returns an error or corrupts the table.
> - When this property is set, the driver uses the specified CMEK for all queries.

LargeResultDataset

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

Description

A persistent destination dataset for storing query results. For more information, see Large Result Set Support on page 25.

> ! **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
A value consisting of the prefix `temp_table_` followed by the time of table creation, if `QueryDialect` is set to `BIG_QUERY`, or if `QueryDialect` is set to `SQL` and `LargeResultDatasetSet` is specified. None, if `QueryDialect` is set to `SQL` and no value is provided for `LargeResultDataSet`.	String	No

Description

A persistent destination dataset for storing query results. For more information, see Large Result Set Support on page 25.

> ❗ **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 driver 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 driver 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 JDBC Driver for Google BigQuery, 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 driver to abort.
- `2`: Enable logging on the ERROR level, which logs error events that might still allow the driver 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 driver.
- `5`: Enable logging on the DEBUG level, which logs detailed information that is useful for debugging the driver.
- `6`: Enable logging on the TRACE level, which logs all driver activity.

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

- A `BigQuery_driver.log` file that logs driver 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 driver activity that is specific to the connection.

If the `LogPath` value is invalid, then the driver 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 driver 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
0	Integer	No

Description

The maximum number of results that are displayed per page.

The default value of 0 allows an unlimited number of results to be displayed per page.

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 15.

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 15.

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 15.

OAuthPvtKeyPath

Default Value	Data Type	Required
None	String	Yes, if `OAuthUserAuth=0`.

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 14.

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 15.

OAuthServiceAcctEmail

Default Value	Data Type	Required
None	String	Yes, if `OAuthUserAuth=0.`

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 14.

OAuthType

Default Value	Data Type	Required
0	Integer	No

Description

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

- `0`: The driver uses service-based OAuth authentication (see Using a Google Service Account on page 14).
- `1`: The driver uses user-based OAuth authentication (see Using a Google User Account on page 13).
- `2`: The driver uses pre-generated tokens for authentication (see Using Pre-Generated Access and Refresh Tokens on page 15).
- `3`: The driver uses Application Default Credentials for authentication (see Using Application Default Credentials on page 15).

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 JDBC Driver for Google BigQuery 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.

ProxyPassword

Default Value	Data Type	Required
None	String	No

Description

The password, if needed, for proxy server settings.

ProxyPort

Default Value	Data Type	Required
None	Integer	No

Description

The listening port of your proxy server.

ProxyUsername

Default Value	Data Type	Required
None	String	No

Description

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

QueryDialect

Default Value	Data Type	Required
SQL	Enumerated	No

Description

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

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

QueryProperties

Default Value	Data Type	Required
QueryProperties	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 driver requests access to Google Drive. Allowing the driver to access Google Drive enables support for federated tables that combine BigQuery data with data from Google Drive.

- `0`: The driver does not request access to Google Drive.
- `1`: The driver 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 43.

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 43.

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 driver 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 driver version 1.1.0 or earlier, while transitioning your applications to connect using driver version 1.1.1 or later.

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

- 1: The driver sets the default timezone of the JVM to UTC. This behavior is non-optimal, but consistent with the behavior from driver 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 JDBC Driver for Google BigQuery.
> - Enabling this property causes the driver 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 driver returns an error.

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

useQueryCache

Default Value	Data Type	Required
1	Integer	No

Description

This option specifies whether the driver uses cached query results.

- `1`: The driver uses cached query results.
- `0`: The driver 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.