

GCS to BigQuery using Bash – Complete Guide

Introduction

I will walk you through how to load data from Google Cloud Storage (GCS) to BigQuery using Bash. This process helps integrate GCS and BigQuery, making it easier to perform data analytics and streamline ETL (Extract, Transform, Load) pipelines. By the end of this guide, you will have a clear understanding of how to use the command line to automate the process of moving data from GCS to BigQuery.

Step-by-Step Process

1. Set Up Your Google Cloud Environment

To begin, I need to ensure that I have the necessary tools installed:

1.1 Install Google Cloud SDK: I followed the installation guide.

1.2 Authenticated with Google Cloud: Once the SDK is installed, I authenticated using my service account by running the following command in my terminal:

```
$ gcloud init
Welcome! This command will take you through the configuration of gcloud.

Settings from your current configuration [aramel-duality-452616-e6] are:
accessibility:
  screen_reader: 'False'
core:
  account: harshavardhan03467@gmail.com
  disable_usage_reporting: 'True'
  project: caramel-duality-452616-e6

Pick configuration to use:
[1] Re-initialize this configuration [aramel-duality-452616-e6] with new settings
[2] Create a new configuration
[3] Switch to and re-initialize existing configuration: [default]
Please enter your numeric choice: 1

Your current configuration has been set to: [aramel-duality-452616-e6]

You can skip diagnostics next time by using the following flag:
gcloud init --skip-diagnostics

Network diagnostic detects and fixes local network connection issues.
Checking network connection...done.
Reachability Check passed.
Network diagnostic passed (1/1 checks passed).

Choose the account you want to use for this configuration.
To use a federated user account, exit this command and sign in to the gcloud CLI with your login configuration file, then run this command again.

Select an account:
[1] harshavardhan03467@gmail.com
[2] Sign in with a new Google Account
[3] Skip this step
Please enter your numeric choice: 1

You are signed in as: [harshavardhan03467@gmail.com].
```

```

You are signed in as: [harshavardhan03467@gmail.com].

Pick cloud project to use:
[1] caramel-duality-452616-e6
[2] Enter a project ID
[3] Create a new project
Please enter numeric choice or text value (must exactly match list item): 1

Your current project has been set to: [caramel-duality-452616-e6].

Not setting default zone/region (this feature makes it easier to use
[gcloud compute] by setting an appropriate default value for the
--zone and --region flag).
See https://cloud.google.com/compute/docs/gcloud-compute section on how to set
default compute region and zone manually. If you would like [gcloud init] to be
able to do this for you the next time you run it, make sure the
Compute Engine API is enabled for your project on the
https://console.developers.google.com/apis page.

The Google Cloud CLI is configured and ready to use!

* Commands that require authentication will use harshavardhan03467@gmail.com by default
* Commands will reference project 'caramel-duality-452616-e6' by default
Run 'gcloud help config' to learn how to change individual settings

This gcloud configuration is called [caramel-duality-452616-e6]. You can create additional configurations if you work with multiple accounts and/or projects.
Run 'gcloud topic topic-configurations' to learn more.

Some things to try next:

* Run 'gcloud --help' to see the Cloud Platform services you can interact with. And run 'gcloud help COMMAND' to get help on any gcloud command.
* Run 'gcloud topic --help' to learn about advanced features of the CLI like arg files and output formatting
* Run 'gcloud cheat-sheet' to see a roster of go-to 'gcloud' commands.

```

Google Cloud Storage (GCS):

In the below screenshot, Here are the details:

bucket-name: covid19data202024

blob-name: WHO_COVID19_globaldata.csv

The screenshot shows the Google Cloud Storage console. The breadcrumb navigation is Buckets > covid19data202024 > WHO_COVID19_globaldata.csv. The object is a LIVE OBJECT. Below the object name, there are links for DOWNLOAD, EDIT METADATA, EDIT ACCESS, and DELETE. An Overview section shows the following details:

Overview	
Type	application/vnd.ms-excel
Size	2.2 MB
Created	Mar 4, 2025, 7:10:01 PM
Last modified	Mar 4, 2025, 7:10:01 PM
Storage class	Standard
Custom time	—
Public URL	Not applicable

Create a BigQuery Dataset

Before I can load data into BigQuery, I need to set up a dataset. If the dataset already exists, I can skip this step.

I use the following command to create a new dataset in BigQuery:

```
bq mk --dataset --description "Data from GCS to BigQuery" <project-id>:<dataset-name>
```

```
$ bq mk --dataset --description "Data from GCS to BigQuery" caramel-duality-452616-e6:Staging
Dataset 'caramel-duality-452616-e6:Staging' successfully created.
```

We can see in the below Screenshot that “Staging” Dataset has been created under my project ID: caramel-duality-452616-e6

The screenshot displays the Google Cloud BigQuery Studio interface. At the top, a banner indicates a free trial status with \$300.00 credit and 90 days remaining. The main header shows the Google Cloud logo, the project name 'My First Project', and a search bar. The left sidebar contains the 'Explorer' panel, which lists various BigQuery resources under the project 'caramel-duality-452616-e6'. The 'Staging' dataset is highlighted. The main content area shows a 'Welcome to BigQuery Studio!' message with options to 'Create new' (SQL QUERY, PYTHON NOTEBOOK, DATA CANVAS, DATA PREPARATION, WORKFLOW) and 'Recently opened' (covid19Data). A 'Try with sample data' section offers a 'Google Trends Demo Query'. The bottom panel shows the 'Job history' section.

Explorer

Search BigQuery resources

Show starred only

caramel-duality-452616-e6

- Queries
- Notebooks
- Data canvases
- Data preparations
- Workflows
- External connections
- Staging

Staging

Dataset info

Dataset ID	caramel-duality-452616-e6.Staging
Created	Mar 4, 2025, 7:27:25 PM UTC-6
Default table expiration	Never
Last modified	Mar 4, 2025, 7:27:25 PM UTC-6
Data location	US
Description	Data from GCS to BigQuery
Default collation	
Default rounding mode	ROUNDING_MODE_UNSPECIFIED
Time travel window	7 days
Case insensitive	false
Labels	
Tags	

Dataset replica info

Primary location	US
------------------	----

Job history

3. Define the GCS File and BigQuery Table

Next, I specify the source file in Google Cloud Storage (GCS) and the destination table in BigQuery.

- Here are the variables I set:

- GCS_FILE: This is the GCS file path I want to load into BigQuery.
- ROJECT_ID: My Google Cloud project ID.
- DATASET_ID: The dataset in BigQuery where I want to store my data.
- TABLE_ID: The name of the table in BigQuery where the data will be loaded.

```
$ GCS_FILE="gs://covid19data202024/WHO_COVID19_global_Clean_data.csv"
PROJECT_ID="caramel-duality-452616-e6"
DATASET_ID="Staging"
TABLE_ID="covid19Data"
```

4. Load Data from GCS to BigQuery

Now, I can load the data into BigQuery using the bq load command. The following command loads the CSV data from GCS into BigQuery:

```
bq load --source_format=CSV --autodetect \
```

```
$PROJECT_ID:$DATASET_ID.$TABLE_ID $GCS_FILE
```

```
$ bq load --source_format=CSV --autodetect \
  $PROJECT_ID:$DATASET_ID.$TABLE_ID $GCS_FILE
Waiting on bqjob_r7429bd075c53bbc8_0000019563ee701e_1 ... (23s) Current status: DONE
```

Explanation of Parameters:

- --source format=CSV: I specify that the source file is in CSV format.
- --autodetect: BigQuery will automatically detect the schema of the CSV file.
- \$PROJECT_ID:\$DATASET_ID.\$TABLE_ID: Specifies where in BigQuery the data will go.
- \$GCS_FILE: Specifies the path to the file in Google Cloud Storage.

View the BigQuery in the following screenshot and can see covid19Data under Staging and can see the data below:

The screenshot shows the Google Cloud BigQuery console. On the left, the 'Explorer' pane shows the project hierarchy: 'caramel-duality-452616-e6' > 'Staging' > 'covid19Data'. The main pane displays the 'covid19Data' table under the 'Staging' dataset. The 'SCHEMA' tab is active, showing a table with 5 columns: 'Date_reported' (DATE, NULLABLE), 'Country' (STRING, NULLABLE), 'Cumulative_cases' (INTEGER, NULLABLE), 'New_deaths' (FLOAT, NULLABLE), and 'Cumulative_deaths' (INTEGER, NULLABLE). Below the schema, there are buttons for 'EDIT SCHEMA' and 'VIEW ROW ACCESS POLICIES'. At the bottom, the 'Job history' section is visible with a 'REFRESH' button.

Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
<input type="checkbox"/> Date_reported	DATE	NULLABLE	-	-	-	-	-
<input type="checkbox"/> Country	STRING	NULLABLE	-	-	-	-	-
<input type="checkbox"/> Cumulative_cases	INTEGER	NULLABLE	-	-	-	-	-
<input type="checkbox"/> New_deaths	FLOAT	NULLABLE	-	-	-	-	-
<input type="checkbox"/> Cumulative_deaths	INTEGER	NULLABLE	-	-	-	-	-

Once I run the command, the data begins loading into BigQuery. The job status will be displayed in the terminal.

```
bq query --use_legacy_sql=false "SELECT * FROM \`${PROJECT_ID}.${DATASET_ID}.${TABLE_ID}` LIMIT 10"
```

```
$ bq query --use_legacy_sql=false "SELECT * FROM \`${PROJECT_ID}.${DATASET_ID}.${TABLE_ID}` LIMIT 10"
```

Date_reported	Country	Cumulative_cases	New_deaths	Cumulative_deaths
2020-01-05	Afghanistan	0	0.0	0
2020-01-12	Afghanistan	0	0.0	0
2020-01-19	Afghanistan	0	0.0	0
2020-01-26	Afghanistan	0	0.0	0
2020-02-02	Afghanistan	0	0.0	0
2020-02-09	Afghanistan	0	0.0	0
2020-02-16	Afghanistan	0	0.0	0
2020-02-23	Afghanistan	0	0.0	0
2020-03-01	Afghanistan	1	0.0	0
2020-03-08	Afghanistan	1	0.0	0

The screenshot shows the Google Cloud BigQuery console interface. The left sidebar contains the 'Explorer' panel with a search bar and a tree view of resources. The 'covid19Data' table is selected under the 'Staging' dataset. The main panel displays the 'PREVIEW' of the table, showing 16 rows of data. The columns are 'Date_reported', 'Country', 'Cumulative_cases', 'New_deaths', and 'Cumulative_deaths'. The data shows a progression of cases and deaths in Afghanistan from January to April 2020. The bottom of the console shows the 'Job history' section with a 'REFRESH' button.

Row	Date_reported	Country	Cumulative_cases	New_deaths	Cumulative_deaths
1	2020-01-05	Afghanistan	0	0.0	0
2	2020-01-12	Afghanistan	0	0.0	0
3	2020-01-19	Afghanistan	0	0.0	0
4	2020-01-26	Afghanistan	0	0.0	0
5	2020-02-02	Afghanistan	0	0.0	0
6	2020-02-09	Afghanistan	0	0.0	0
7	2020-02-16	Afghanistan	0	0.0	0
8	2020-02-23	Afghanistan	0	0.0	0
9	2020-03-01	Afghanistan	1	0.0	0
10	2020-03-08	Afghanistan	1	0.0	0
11	2020-03-15	Afghanistan	7	0.0	0
12	2020-03-22	Afghanistan	24	0.0	0
13	2020-03-29	Afghanistan	91	2.0	2
14	2020-04-05	Afghanistan	274	3.0	5
15	2020-04-12	Afghanistan	521	10.0	15
16	2020-04-19	Afghanistan	908	15.0	30

Why GCS to BigQuery in Real-Time?

Moving data from Google Cloud Storage (GCS) to BigQuery has several real-time use cases:

- 1. Data Analytics:** I can store raw data in GCS and then load it into BigQuery for complex analyses using SQL queries. This is particularly useful for large datasets that require powerful analytics tools.
- 2. Automated ETL Pipelines:** By automating the process of moving data from GCS to BigQuery, I can build efficient ETL pipelines that run at scheduled intervals, ensuring my data is always up to date.

3. Business Intelligence: BigQuery is ideal for running business intelligence queries on large datasets. I can use the data loaded from GCS to generate reports, dashboards, and other insights.

4. Machine Learning: Once the data is in BigQuery, I can use it for feature extraction and even run machine learning models directly within BigQuery using BigQuery ML.

5. Real-time Decision Making: By continuously loading data from GCS into BigQuery, I can enable real-time analytics, which helps with faster decision-making for businesses.

Conclusion

In this guide, I've demonstrated how to load data from Google Cloud Storage (GCS) into BigQuery using Git Bash. By following these steps, I can easily move large datasets into BigQuery for advanced data analysis, reporting, and business intelligence. This process is highly beneficial for automating ETL pipelines and leveraging BigQuery's powerful analytics capabilities for real-time insights.