# Tools

<https://cloud.google.com/sdk/docs/install>

Google Cloud CLI(Linux) includes the gcloud, gsutil and bq command-line tools.

Run from git bash

# Permissions

<https://cloud.google.com/bigquery/docs/access-control#bigquery>

#### BigQuery Admin

#### BigQuery Data Editor

#### BigQuery Data Viewer

#### BigQuery Data Owner

#### Roles assign using terraform

#### <https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/bigquery_dataset_iam>

# GCP BQ Steps

1. You can list the active account name with this command:

gcloud auth login

gcloud config set project <project name>

**Task 1. Examine a table**

BigQuery offers a number of [sample tables](https://cloud.google.com/bigquery/public-data#sample_tables) that you can run queries against. In this lab, you'll run queries against the shakespeare table, which contains an entry for every word in every play.

To examine the schema of the Shakespeare table in the samples dataset, run:

bq show bigquery-public-data:samples.shakespeare

In this command you're doing the following:

* bq to invoke the BigQuery command line tool
* show is the action
* Then you're listing the name of the project:public dataset.table in BigQuery that you want to see.

Output:

Last modified Schema Total Rows Total Bytes Expiration Time Partitioning Clustered Fields Labels

----------------- ------------------------------------ ------------ ------------- ------------ ------------------- ------------------ --------

14 Mar 13:16:45 |- word: string (required) 164656 6432064

|- word\_count: integer (required)

|- corpus: string (required)

|- corpus\_date: integer (required)

**Task 2. Run the help command**

When you include a command name with the help commands, you get information about that specific command.

1. For example, the following call to bq help retrieves information about the query command:

bq help query

1. To see a list of all of the commands bq uses, run just bq help.

**Task 3. Run a query**

Now you'll run a query to see how many times the substring "raisin" appears in Shakespeare's works.

1. To run a query, run the command bq query "[SQL\_STATEMENT]":

* Escape any quotation marks inside the [SQL\_STATEMENT] with a \ mark, or
* Use a different quotation mark type than the surrounding marks ("versus").

1. Run the following standard SQL query in Cloud Shell to count the number of times that the substring "raisin" appears in all of Shakespeare's works:

bq query --use\_legacy\_sql=false \

'SELECT

word,

SUM(word\_count) AS count

FROM

`bigquery-public-data`.samples.shakespeare

WHERE

word LIKE "%raisin%"

GROUP BY

word'

In this command:

* --use\_legacy\_sql=false makes standard SQL the default query syntax.

Output:

Waiting on job\_e19 ... (0s) Current status: DONE

+---------------+-------+

| word | count |

+---------------+-------+

| praising | 8 |

| Praising | 4 |

| raising | 5 |

| dispraising | 2 |

| dispraisingly | 1 |

| raisins | 1 |

The table demonstrates that although the actual word **raisin** doesn't appear, the letters appear in order in several of Shakespeare's works.

Test completed task

Click **Check my progress** to verify your performed task. If you have successfully run a query against a public dataset, you will see an assessment score.

Assessment Completed!

Run a query (dataset: samples, table: shakespeare, substring: raisin)

Check my progress

*Assessment Completed!*

If you search for a word that isn't in Shakespeare's works, no results are returned.

* Run the following search for "huzzah", returns no matches:

bq query --use\_legacy\_sql=false \

'SELECT

word

FROM

`bigquery-public-data`.samples.shakespeare

WHERE

word = "huzzah"'

Test completed task

Click **Check my progress** to verify your performed task. If you have successfully run a query against a public dataset, you will see an assessment score.

Assessment Completed!

Run a query (dataset: samples, table: shakespeare, substring: huzzah)

Check my progress

*Assessment Completed!*

**Task 4. Create a new table**

Now create your own table. Every table is stored inside a dataset. A *dataset* is a group of resources, such as tables and views.

**Create a new dataset**

1. Use the bq ls command to list any existing datasets in your project:

bq ls

You will be brought back to the command line since there aren't any datasets in your project yet.

1. Run bq ls and the bigquery-public-data Project ID to list the datasets in that specific project, followed by a colon (:):

bq ls bigquery-public-data:

Output:

datasetId

-----------------------------

austin\_311

austin\_bikeshare

austin\_crime

austin\_incidents

austin\_waste

baseball

bitcoin\_blockchain

bls

census\_bureau\_construction

census\_bureau\_international

census\_bureau\_usa

census\_utility

chicago\_crime

...

Now create a dataset. A dataset name can be up to 1,024 characters long, and consist of A-Z, a-z, 0-9, and the underscore, but it cannot start with a number or underscore, or have spaces.

1. Use the bq mk command to create a new dataset named babynames in your project:

bq mk babynames

Copied!

content\_copy

Sample output:

Dataset 'qwiklabs-gcp-ba3466847fe3cec0:babynames' successfully created.

Test completed task

Click **Check my progress** to verify your performed task. If you have successfully created a BigQuery dataset named babynames, you will see an assessment score.

Assessment Completed!

Create a new dataset (name: babynames)

Check my progress

*Assessment Completed!*

* Run bq ls to confirm that the dataset now appears as part of your project:

bq ls

Sample output:

datasetId

-------------

babynames

**Upload the dataset**

Before you can build the table, you need to add the dataset to your project. The custom data file you'll use contains approximately 7 MB of data about popular baby names, provided by the US Social Security Administration.

1. Run this command to add the [baby names zip file](http://www.ssa.gov/OACT/babynames/names.zip) to your project, using the URL for the data file:

curl -LO <http://www.ssa.gov/OACT/babynames/names.zip>

1. List the file:

ls

You can see the name of the file added to your project.

1. Now unzip the file:

unzip names.zip

1. That's a pretty big list of text files! List the files again:

ls

The bq load command creates or updates a table and loads data in a single step.

You will use the bq load command to load your source file into a new table called names2010 in the babynames dataset you just created. By default, this runs synchronously, and will take a few seconds to complete.

The bq load arguments you'll be running are:

datasetID: babynames

tableID: names2010

source: yob2010.txt

schema: name:string,gender:string,count:integer

1. Create your table:

bq load babynames.names2010 yob2010.txt name:string,gender:string,count:integer

Sample output:

Waiting on job\_4f0c0878f6184119abfdae05f5194e65 ... (35s) Current status: DONE

Test completed task

Click **Check my progress** to verify your performed task. If you have successfully loaded data into a dataset table, you will see an assessment score.

Assessment Completed! Table name(s): ["names2010"]

Load the data into a new table

Check my progress

*Assessment Completed! Table name(s): ["names2010"]*

1. Run bq ls and babynames to confirm that the table now appears in your dataset:

bq ls babynames

Output:

tableId Type

----------- -------

names2010 TABLE

1. Run bq show and your dataset.table to see the schema:

bq show babynames.names2010

Output:

Last modified Schema Total Rows Total Bytes Expiration Time Partitioning Clustered Fields Labels

----------------- ------------------- ------------ ------------- ----------------- ------------------- ------------------ --------

13 Aug 14:37:34 |- name: string 34073 654482 12 Oct 14:37:34

|- gender: string

|- count: integer

**Note:**By default, when you load data, BigQuery expects UTF-8 encoded data. If you have data that is in ISO-8859-1 (or Latin-1) encoding and are having problems with your loaded data, you can tell BigQuery to treat your data as Latin-1 explicitly, using the -E flag. Learn more about Character Encodings from the [Introduction to loading data guide](https://cloud.google.com/bigquery/docs/loading-data).

**Task 5. Run queries**

Now you're ready to query the data and return some interesting results.

1. Run the following command to return the top 5 most popular girls names:

bq query "SELECT name,count FROM babynames.names2010 WHERE gender = 'F' ORDER BY count DESC LIMIT 5"

Output:

Waiting on job\_58c0f5ca52764ef1902eba611b71c651 ... (0s) Current status: DONE

+----------+-------+

| name | count |

+----------+-------+

| Isabella | 22913 |

| Sophia | 20643 |

| Emma | 17345 |

| Olivia | 17028 |

| Ava | 15433 |

+----------+-------+

1. Run the following command to see the top 5 most unusual boys names:

bq query "SELECT name,count FROM babynames.names2010 WHERE gender = 'M' ORDER BY count ASC LIMIT 5"

**Note:** The minimum count is 5 because the source data omits names with fewer than 5 occurrences.

Output:

Waiting on job\_556ba2e5aad340a7b2818c3e3280b7a3 ... (1s) Current status: DONE

+----------+-------+

| name | count |

+----------+-------+

| Aaqib | 5 |

| Aaidan | 5 |

| Aadhavan | 5 |

| Aarian | 5 |

| Aamarion | 5 |

+----------+-------+

**Task 7. Clean up**

1. Run the bq rm command to remove the babynames dataset with the -r flag to delete all tables in the dataset:

bq rm -r babynames

1. Confirm the delete command by typing Y.