Quickstart using the bq command-line tool

cloud.google.com/bigquery/docs/quickstarts/quickstart-command-line

This quickstart explains how to use the bq command-line tool to run queries and load data into BigQuery.

Before you begin

1. In the Google Cloud Console, on the project selector page, select or create a Google Cloud project.

   Note: If you don't plan to keep the resources that you create in this procedure, create a project instead of selecting an existing project. After you finish these steps, you can delete the project, removing all resources associated with the project.
   Go to project selector

2. If you are using a preexisting project, enable the BigQuery API. BigQuery is automatically enabled in new projects.

   Enable the API

3. Optional: Enable billing for the project. If you don't want to enable billing or provide a credit card, the steps in this document still work. BigQuery provides a sandbox to perform the steps.
4. In the Cloud Console, activate Cloud Shell.

   Activate Cloud Shell

   In this quickstart, you run all the `bq` tool commands in Cloud Shell from the Cloud Console.

Examine a table

BigQuery offers several sample tables that you can query. In this quickstart, you run some queries against the `shakespeare` table, which contains an entry for every word in every Shakespeare play.

Examine the `shakespeare` table in the `samples` dataset:

```
bq show bigquery-public-data:samples.shakespeare
```

This example command examines the schema of a specific table. If the project and dataset IDs are the default values for your `bq` tool, then you can omit them in the `bq show` command and just specify the table ID:

```
bq show shakespeare
```

The output is similar to the following:

```
Table bigquery-public-data:samples.shakespeare

 Last modified                  Schema                 Total Rows   Total
Bytes   Expiration
----------------- ------------------------------------ ------------ ---------
---- ------------
26 Aug 14:43:49   |- word: string (required)           164656       6432064
                  |- word_count: integer (required)
                  |- corpus: string (required)
                  |- corpus_date: integer (required)
```

Run the help command

1. View detailed information about the `bq` tool:

```
bq help
```

2. View information about a specific command:

```
bq help query
```

In this example, the call to `bq help` retrieves information about the `bq query` command.

Run a query

1. To see how many times the substring `raisin` appears in Shakespeare's works, run a query using the `bq query` command:

   ```
   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'
   ```

   The output is similar to the following:

   ```
   Waiting on job_dcda37c0bbed4c669b04dfd567859b90 ... (0s) Current status:
   DONE
   +---------------+-------+
   |     word      | count |
   +---------------+-------+
   | Praising      |   4   |
   | raising       |   5   |
   | raisins       |   1   |
   | praising      |   8   |
   | dispraising   |   2   |
   | dispraisingly |   1   |
   +---------------+-------+
   ```

2. To see how many times the substring `huzzah` appears in Shakespeare's works, run the following query:

   ```
   bq query --use_legacy_sql=false \
   'SELECT
    word
   FROM
    `bigquery-public-data`.samples.shakespeare
   WHERE
    word = "huzzah"'
   ```

   Since the substring doesn't appear in Shakespeare's works, no results are returned. The output is similar to the following:

   ```
   Waiting on job_e19 ... (4s) Current status: DONE
   ```

Create a new table

In the next sections, you create a new table and place it in a new dataset.

Download the sample data

The sample data is provided by the US Social Security Administration and contains approximately 7 MB of data about popular baby names.

1. Download and extract the baby names zip file.

   The zip file contains a file named `NationalReadMe.pdf` that describes the dataset schema. Learn more about the dataset.

2. Open the file `yob2010.txt` to see what it looks like. The file contains comma-separated values for the following three columns: name, sex (`M` or `F`), and number of children with that name. The file has no header row.

3. Copy or move the file `yob2010.txt` into the directory where you are running the `bq` command-line tool. If you are running the `bq` command-line tool in Cloud Shell, upload the `yob2010.txt` file. For more information, see Managing files with Cloud Shell.

Create a new dataset

1. Check whether your default project has existing datasets:

   ```
   bq ls
   ```

   The output is similar to the following:

   ```
     datasetId
    -------------
     olddataset
   ```

2. List the datasets in a specific project by including the project ID followed by a colon (`:`):

   ```
   bq ls publicdata:
   ```

   This example lists the datasets in the `publicdata` project.

   The output is similar to the following:

   ```
     datasetId
    -----------
     samples
   ```

3. In the project you selected for this quickstart, create a new dataset named `babynames` :

   ```
   bq mk babynames
   ```

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

   The output is similar to the following:

   ```
   Dataset 'myprojectid:babynames' successfully created.
   ```

4. Confirm that the dataset now appears as part of the default project:

   ```
   bq ls
   ```

   The output is similar to the following:

   ```
     datasetId
    -------------
     olddataset
     babynames
   ```

Upload the table

1. In the `babynames` dataset that you created, load your source file `yob2010.txt` into a new table called `names2010` :

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

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

   The command includes the following arguments:

   - datasetID: `babynames`
   - tableID: `names2010`
   - source: `yob2010.txt` (if necessary, include the full path)
   - schema: `name:string,gender:string,count:integer`

   The output is similar to the following:

   ```
   Upload complete.
   Waiting on job_4f0c0878f6184119abfdae05f5194e65 ... (35s)
   Current status: DONE
   ```

2. Confirm that the table now appears in the dataset:

   ```
   bq ls babynames
   ```

   The output is similar to the following:

   ```
      tableId     Type
    ----------- -------
     names2010    TABLE
   ```

3. View the schema:

   ```
   bq show babynames.names2010
   ```

   The output is similar to the following

   ```
   Table myprojectid:babynames.names2010

      Last modified         Schema         Total Rows   Total Bytes   Expiration
    ----------------- ------------------- ------------ ------------- -----------
   -
     13 Mar 15:31:00   |- name: string     34041        653855
                       |- gender: string
                       |- count: integer
   ```

By default, when you load data, BigQuery expects UTF-8 encoded data. If you have data in ISO-8859-1 (or Latin-1) encoding and you're having problems with it, instruct BigQuery to treat your data as Latin-1 using the `-E` flag. For more information, see Encoding.

Run queries

1. Return the most popular girls' names:

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

   The output is similar to the following:

   ```
   Waiting on job_58c0f5ca52764ef1902eba611b71c651 ... (0s) Current status: DONE
   +----------+-------+
   |   name   | COUNT |
   +----------+-------+
   | Isabella | 22731 |
   | Sophia   | 20477 |
   | Emma     | 17179 |
   | Olivia   | 16860 |
   | Ava      | 15300 |
   +----------+-------+
   ```

2. See the most unusual boys' names:

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

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

   The output is similar to the following:

   ```
   Waiting on job_556ba2e5aad340a7b2818c3e3280b7a3 ... (1s) Current status: DONE
   +----------+-------+
   |   name   | COUNT |
   +----------+-------+
   | Aarian   |     5 |
   | Aaidan   |     5 |
   | Aamarion |     5 |
   | Aadhavan |     5 |
   | Aaqib    |     5 |
   +----------+-------+
   ```

Clean up

To avoid incurring charges to your Google Cloud account for the resources used in this quickstart, follow these steps.

1. Remove the `babynames` dataset:

   ```
   bq rm --recursive=true babynames
   ```

   The `--recursive` flag deletes all tables in the dataset, including the `names2010` table.

2. To confirm the delete command, type `y` .

However, if you followed this quickstart in a new project, then you can delete the project.

What's next

- To learn more about the `bq` command-line tool, see Using the bq command-line tool.
- To learn more about loading data into BigQuery, see Introduction to loading data.
- To learn more about querying data, see Overview of querying BigQuery data.
- To learn how to export data out of BigQuery, see Exporting table data.
- To learn more about accessing BigQuery programmatically, see the REST API reference or the BigQuery API client libraries page.