Dataflow: Qwik Start – Templates

Activate Cloud Shell

Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Cloud Shell provides command-line access to your Google Cloud resources.

1. Click **Activate Cloud Shell** at the top of the Google Cloud console.

2. Click through the following windows:

   - Continue through the Cloud Shell information window.
   - Authorize Cloud Shell to use your credentials to make Google Cloud API calls.

When you are connected, you are already authenticated, and the project is set to your **Project_ID**, `qwiklabs-gcp-04-3b88e90c83b0`. The output contains a line that declares the **Project_ID** for this session:

```
Your Cloud Platform project in this session is set to qwiklabs-gcp-04-3b88e90c83b0
```

`gcloud` is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell and supports tab-completion.

3. (Optional) You can list the active account name with this command:

```
gcloud auth list
```

Copied!

content_copy

4. Click **Authorize**.

Output:

```
ACTIVE: *
ACCOUNT: student-01-c3311f1dcbcf@qwiklabs.net

To set the active account, run:
    $ gcloud config set account `ACCOUNT`
```

5. (Optional) You can list the project ID with this command:

```
gcloud config list project
```

Copied!

content_copy

Output:

```
[core]
project = qwiklabs-gcp-04-3b88e90c83b0
```

Note: For full documentation of `gcloud`, in Google Cloud, refer to the gcloud CLI overview guide.

Task 1. Ensure that the Dataflow API is successfully re-enabled

To ensure access to the necessary API, restart the connection to the Dataflow API.

1. In the Cloud Console, enter "Dataflow API" in the top search bar. Click on the result for **Dataflow API**.

2. Click **Manage**.

3. Click **Disable API**.

If asked to confirm, click **Disable**.

4. Click **Enable**.

When the API has been enabled again, the page will show the option to disable.

Task 2. Create a BigQuery dataset, BigQuery table, and Cloud Storage bucket using Cloud Shell

Let's first create a BigQuery dataset and table.

Note: This task uses the `bq` command-line tool. **Skip down** to Task 3 if you want to complete these steps using the Cloud console.

1. Run the following command to create a dataset called `taxirides`:

```
bq mk taxirides
```

Copied!

content_copy

Your output should look similar to:

```
Dataset '<myprojectid:taxirides>' successfully created
</myprojectid:taxirides>
```

Now that you have your dataset created, you'll use it in the following step to instantiate a BigQuery table.

2. Run the following command to do so:

```
bq mk \
--time_partitioning_field timestamp \
--schema
ride_id:string,point_idx:integer,latitude:float,longitude:float,\
timestamp:timestamp,meter_reading:float,meter_increment:float,ride_stat
us:string,\
passenger_count:integer -t taxirides.realtime
```

Copied!

content_copy

Your output should look similar to:

```
Table 'myprojectid:taxirides.realtime' successfully created
```

On its face, the `bq mk` command looks a bit complicated. However, with some assistance from the BigQuery command-line documentation, we can break down what's going on here. For example, the documentation tells us a little bit more about **schema**:

- Either the path to a local JSON schema file or a comma-separated list of column definitions in the form `[FIELD]:[DATA_TYPE],[FIELD]:[DATA_TYPE]`.

In this case, we are using the latter—a comma-separated list.

Create a Cloud Storage bucket using Cloud Shell

Now that we have our table instantiated, let's create a bucket.

Use the Project ID as the bucket name to ensure a globally unique name: `qwiklabs-gcp-04-3b88e90c83b0`

- Run the following commands to do so:
```
export BUCKET_NAME=qwiklabs-gcp-04-3b88e90c83b0
```
Copied!

content_copy

```
gsutil mb gs://$BUCKET_NAME/
```

Task 3. Create a BigQuery dataset, BigQuery table, and Cloud Storage bucket using the Google Cloud console

Note: Do not complete Task 3 if you completed Task 2, which includes the same tasks in the command line!

1. From the left-hand menu, in the Big Data section, click on **BigQuery**.
2. Then click **Done**.
3. Click on the three dots next to your project name under the **Explorer** section, then click **Create dataset**.
4. Input `taxirides` as your dataset ID:
5. Select **us (multiple regions in United States)** in Data location.
6. Leave all of the other default settings in place and click **CREATE DATASET**.

7. You should now see the `taxirides` dataset underneath your project ID in the left-hand console.

8. Click on the three dots next to `taxirides` dataset and select **Open**.

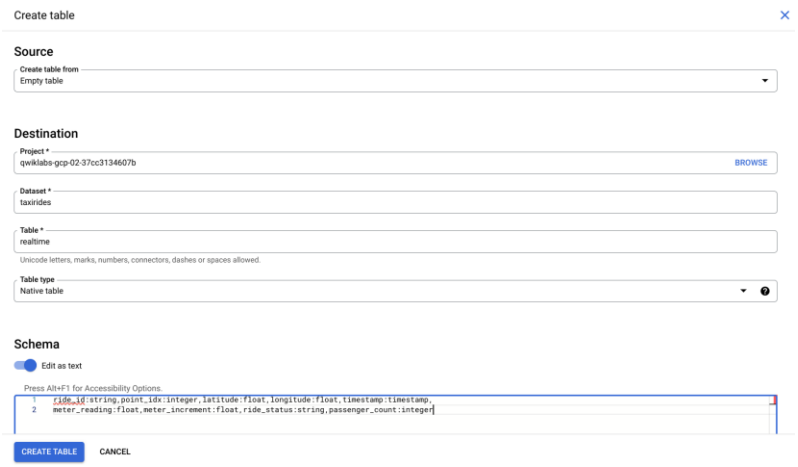
9. Then select **CREATE TABLE** in the right-hand side of the console.

10. In the **Destination** > **Table Name** input, enter `realtime`.

11. Under Schema, toggle the **Edit as text** slider and enter the following:

```
ride_id:string,point_idx:integer,latitude:float,longitude:float,timestamp:timestamp,
meter_reading:float,meter_increment:float,ride_status:string,passenger_count:integer
```

Your console should look like the following:



12. Now, click **Create table**.

Create a Cloud Storage bucket using the Cloud console

1. Go back to the Cloud Console and navigate to **Cloud Storage** > **Buckets** > **Create bucket**.

2. Use the Project ID as the bucket name to ensure a globally unique name: `qwiklabs-gcp-04-3b88e90c83b0`

3. Leave all other default settings, then click **Create**.

Task 4. Run the pipeline

Deploy the Dataflow Template:

```
gcloud dataflow jobs run iotflow \
    --gcs-location gs://dataflow-templates-europe-
west1/latest/PubSub_to_BigQuery \
    --region europe-west1 \
    --worker-machine-type e2-medium \
    --staging-location gs://qwiklabs-gcp-04-3b88e90c83b0/temp \
    --parameters inputTopic=projects/pubsub-public-
data/topics/taxirides-realtime,outputTableSpec=qwiklabs-gcp-04-
3b88e90c83b0:taxirides.realtime
```

In the **Google Cloud Console**, on the **Navigation menu**, click **Dataflow > Jobs**, and you will see your dataflow job.

Please refer the document for more information.
You'll watch your resources build and become ready for use.

Now, let's go view the data written to BigQuery by clicking on **BigQuery** found in the Navigation menu.

- When the BigQuery UI opens, you'll see the **taxirides** dataset added under your project name and **realtime** table underneath that.

Task 5. Submit a query

You can submit queries using standard SQL.

1. In the BigQuery **Editor**, add the following to query the data in your project:

```
SELECT * FROM `qwiklabs-gcp-04-3b88e90c83b0.taxirides.realtime` LIMIT
1000
```

Copied!

content_copy

2. Now click **RUN**.

If you run into any issues or errors, run the query again (the pipeline takes a minute to start up.)

3. When the query runs successfully, you'll see the output in the **Query Results** panel as shown below:

Query results SAVE AS EXPLORE IN DATA STUDIO

Query complete (2.116 sec elapsed, 0 B processed)

Job information | Results | JSON | Execution details

Row	ride_id	point_idx	latitude	longitude	timestamp
1	b0810fbd-78a8-4159-b9ff-963695e2a23d	225	40.753550000000004	-73.98504000000001	2018-07-25 23:28:20.870530 UTC
2	1a10dc8b-3623-41bf-938a-9fca26c2ae10	311	40.752930000000006	-73.96584	2018-07-25 23:24:10.608380 UTC
3	5253c100-1a30-4a3e-89ee-6c0c861cf44f	224	40.74331	-73.99172	2018-07-25 23:26:34.636480 UTC
4	3efa96c2-4695-4c0b-96b6-da33a4b74ccf	8	40.7533	-73.97832000000001	2018-07-25 23:24:06.823150 UTC
5	d6d37615-ccba-4416-9932-e956e0f0ba65	747	40.682140000000004	-74.00594000000001	2018-07-25 23:24:10.103770 UTC