Examining Billing data with BigQuery

In this lab, you learn how to use BigQuery to analyze billing data.

Objectives

In this lab, you learn how to perform the following tasks:

- Sign in to BigQuery from the GCP Console
- Create a dataset
- Create a table
- Import data from a billing CSV file stored in a bucket
- Run complex queries on a larger dataset

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click Start Lab, shows how long Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access the Google Cloud Platform for the duration of the lab.

What you need

To complete this lab, you need:

- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.
  Note: If you already have your own personal GCP account or project, do not use it for this lab.

Task 1: Use BigQuery to import data

Sign in to BigQuery and create a dataset

1. In the GCP Console, on the **Navigation menu** (☰), click **BigQuery**.
2. If prompted, click **Done**.
3. click on to your **Project ID** (starts with qwiklabs-gcp) and click **Create Dataset**.

You can export billing data directly to BigQuery as outlined here. However, for the purposes of this lab, a sample CSV billing file has been prepared for you. It is located in a Cloud Storage bucket where it is accessible to your student account. You will import this billing information into a BigQuery table and examine it.

4. Specify the following:

Property	**Value (type value or select option as specified)**
Dataset ID:	**imported_billing_data**
Data location:	**US**
Default table expiration > Number of days after table creation:	In **1** days

5. Click **Create Dataset**. You should see **imported_billing_data** in the left pane.

Create a table and import

1. Point to **imported_billing_data**, and then click **Create Table** to create a new table.

2. For **Source**, specify the following, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Create table from:	**Google Cloud Storage**
Select file from GCS bucket	`gs://cloud-training/archinfra/export-billing-example.csv`
File format	**CSV**

3. For **Destination**, specify the following, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Table name	**sampleinfotable**
Table type	**Native table**

4. Under **Schema** for **Auto detect** click **Schema and input parameters** .
5. Open **Advanced options**
6. Under **Header rows to skip** specify 1
7. Click **Create Table**. After the job is completed, the table appears below the dataset in the left pane.

Click *Check my progress* to verify the objective.

Use BigQuery to import data

Check my progress

Task 2: Examine the table

1. Click **sampleinfotable**.

This displays the schema that BigQuery automatically created based on the data it found in the imported CSV file. Notice that there are strings, integers, timestamps, and floating values.

2. Click **Details**. As you can see in **Number of Rows**, this is a relatively small table with 44 rows.
3. Click **Preview**.
4. Locate the row that has the **Description:** Network Internet Ingress from EMEA to Americas.

What was the total consumption and units consumed?

○

9,738 bytes

○

9,738,199 bytes

○

9 bytes

○

9,738,199,000 bytes

Submit

5. Scroll to the **Cost** column.

The cost was 0.0, so with an ingress of 9.7 Mbytes, traffic from EMEA to the Americas had no charge.

6. Locate the row that has the **Description:** Network Internet Egress from Americas to China.

Can you interpret the information?

○

5,542 bytes exited the Americas and was transferred to China at a charge of 1e-06.

○

5,542,000 bytes exited the Americas and was transferred to China at a charge of 1e-06.

○

5,542 bytes exited China and was transferred to the Americas at a charge of 1e-06.

Submit

Task 3: Compose a simple query

When you reference a table in a query, both the dataset ID and table ID must be specified; the project ID is optional.

If the project ID is not specified, BigQuery will default to the current project.
All the information you need is available in the BigQuery interface. In the column on the left, you see the dataset ID (imported_billing_data) and table ID (sampleinfotable).

Recall that clicking on the table name brings up the **Schema** with all of the field names.
Now construct a simple query based on the **Cost** field.

1. click **Compose New Query**.
2. Paste the following in Query Editor:

```
SELECT * FROM `imported_billing_data.sampleinfotable`
WHERE Cost > 0
```

3. Click **Run**.

How many rows had cost greater than 0?

○ 104 rows

○ 10 rows

○ 20 rows

○ 44 rows

Submit

How many rows involved non-zero charges?

The table shows 20 rows and they all have non-zero charges.
Click *Check my progress* to verify the objective.

Compose a simple query

Check my progress

Task 4: Analyze a large billing dataset with SQL

In the next activity, you use BigQuery to analyze a sample dataset with 22,537 lines of billing data.

The **cloud-training-prod-bucket.arch_infra.billing_data** dataset used in this task is shared with the public. For more information on public datasets and how to share datasets with the public, refer to the documentation.

1. For New Query, paste the following in Query Editor:

```
SELECT
  product,
  resource_type,
  start_time,
  end_time,
  cost,
  project_id,
  project_name,
  project_labels_key,
  currency,
  currency_conversion_rate,
  usage_amount,
  usage_unit
FROM
  `cloud-training-prod-bucket.arch_infra.billing_data`
```

2. Click **Run**. Verify that the resulting table has 22,537 lines of billing data.
3. To find the latest 100 records where there were charges (cost > 0), for New Query, paste the following in Query Editor:

```
SELECT
  product,
  resource_type,
  start_time,
  end_time,
  cost,
  project_id,
  project_name,
  project_labels_key,
  currency,
  currency_conversion_rate,
  usage_amount,
  usage_unit
FROM
  `cloud-training-prod-bucket.arch_infra.billing_data`
WHERE
  Cost > 0
ORDER BY end_time DESC
```

```
LIMIT
  100
```

4. Click **Run**.
5. To find all charges that were more than 3 dollars, for Compose New Query, paste the following in Query Editor:

```
SELECT
  product,
  resource_type,
  start_time,
  end_time,
  cost,
  project_id,
  project_name,
  project_labels_key,
  currency,
  currency_conversion_rate,
  usage_amount,
  usage_unit
FROM
  `cloud-training-prod-bucket.arch_infra.billing_data`
WHERE
  cost > 3
```

6. Click **Run**.
7. To find the product with the most records in the billing data, for New Query, paste the following in Query Editor:

```
SELECT
  product,
  COUNT(*) AS billing_records
FROM
  `cloud-training-prod-bucket.arch_infra.billing_data`
GROUP BY
  product
ORDER BY billing_records DESC
```

8. Click **Run**.

Which product had the most billing records?

○ Cloud SQL has 10,271 records

○ Cloud Pub/Sub has 10,271 records

○ Stackdriver has 11,271 records

Submit

9. To find the most frequently used product costing more than 1 dollar, for New Query, paste the following in Query Editor:

```
SELECT
```

```
  product,
  COUNT(*) AS billing_records
FROM
  `cloud-training-prod-bucket.arch_infra.billing_data`
WHERE
  cost > 1
GROUP BY
  product
ORDER BY
  billing_records DESC
```

10. Click **Run**.

Which product had the most billing records of over $1

○ Compute Engine has 17 charges costing more than 1 dollar.

○ Kubernetes Engine has 7 charges costing more than 1 dollar.

○ Cloud SQL has 15 charges costing more than 1 dollar.

Submit

11. To find the most commonly charged unit of measure, for Compose New Query, paste the following in Query Editor:

```
SELECT
  usage_unit,
  COUNT(*) AS billing_records
FROM
  `cloud-training-prod-bucket.arch_infra.billing_data`
WHERE cost > 0
GROUP BY
  usage_unit
ORDER BY
  billing_records DESC
```

12. Click **Run**.

What was the most commonly charged unit of measure?

○ Requests were the most commonly charged unit of measure with 6,539 requests.

○ Requests were the most commonly charged unit of measure with 504 requests.

○ Byte-seconds were the most commonly charged unit of measure with 2,937 requests.

Submit

13. To find the product with the highest aggregate cost, for New Query, paste the following in Query Editor:

```
SELECT
  product,
  ROUND(SUM(cost),2) AS total_cost
FROM
  `cloud-training-prod-bucket.arch_infra.billing_data`
GROUP BY
  product
ORDER BY
  total_cost DESC
```

14. Click **Run**.

Which product has the highest total cost?

○ Compute Engine has an aggregate cost of $112.02.

○ Cloud SQL has an aggregate cost of $47.37

○ BigQuery has an aggregate cost of $114.02

Submit

Task 5: Review

In this lab, you imported billing data into BigQuery that had been generated as a CSV file. You ran a simple query on the file. Then you accessed a shared dataset containing more than 22,000 records of billing information. You ran a variety of queries on that data to explore how you can use BigQuery to ask and answer questions by running queries.