Creating Date-Partitioned Tables in BigQuery

GSP414

Overview

BigQuery is Google's fully managed, NoOps, low cost analytics database. With BigQuery you can query terabytes and terabytes of data without having any infrastructure to manage or needing a database administrator. BigQuery uses SQL and can take advantage of the pay-as-you-go model. BigQuery allows you to focus on analyzing data to find meaningful insights.
The dataset you'll use is an ecommerce dataset that has millions of Google Analytics records for the Google Merchandise Store loaded into BigQuery. You have a copy of that dataset for this lab and will explore the available fields and row for insights.
In this lab you will query partitioned datasets and create your own dataset partitions to improve query performance and reduce cost.

Setup and requirements

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 Google 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 Google Cloud for the duration of the lab.

What you need

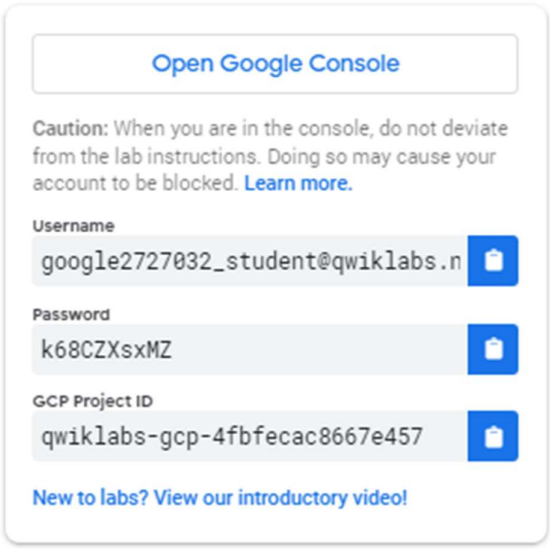
To complete this lab, you need:

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

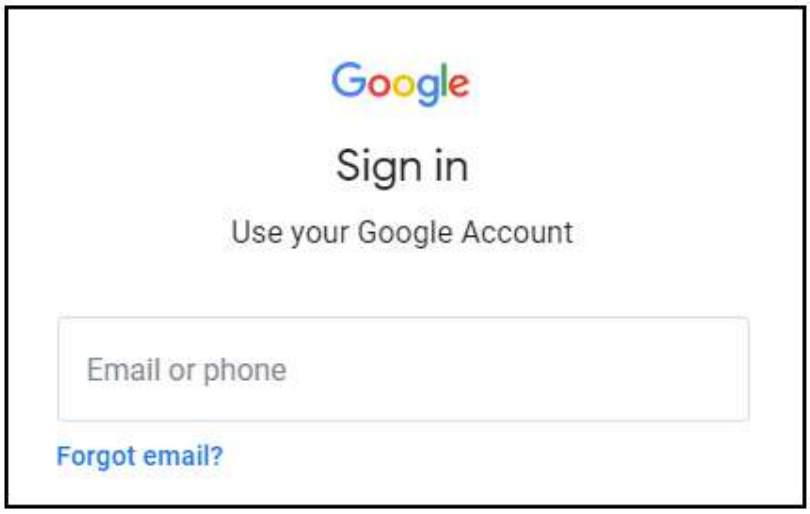
Note: If you are using a Pixelbook, open an Incognito window to run this lab.

How to start your lab and sign in to the Google Cloud Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.

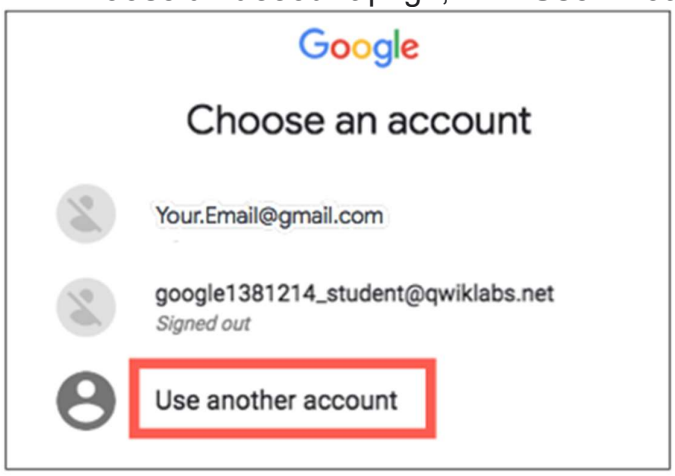


2. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.



Tip: Open the tabs in separate windows, side-by-side.

If you see the **Choose an account** page, click **Use Another Account**.



3. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.

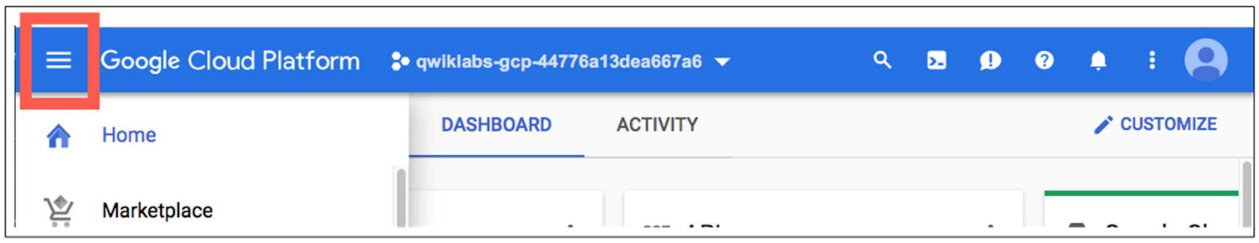
   Important: You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).

4. Click through the subsequent pages:

   - Accept the terms and conditions.
   - Do not add recovery options or two-factor authentication (because this is a temporary account).
   - Do not sign up for free trials.

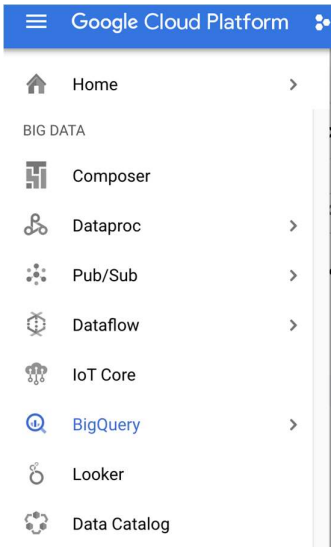
After a few moments, the Cloud Console opens in this tab.

Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left.



Open BigQuery Console

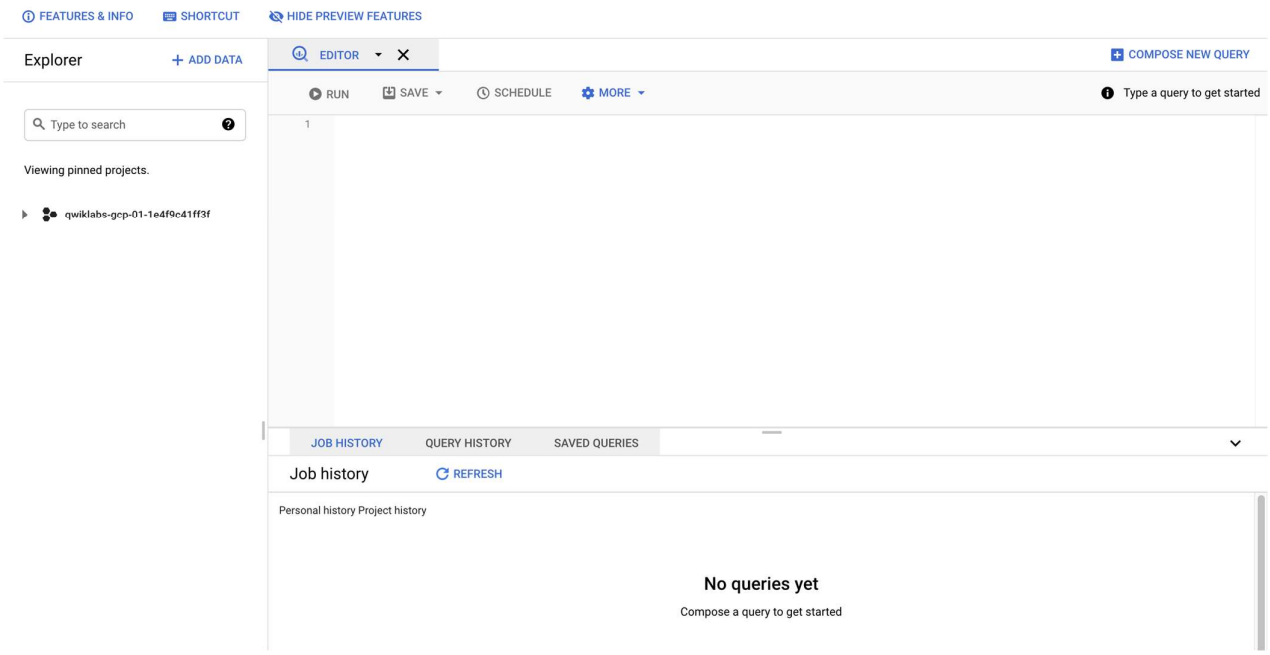
In the Google Cloud Console, select **Navigation menu** > **BigQuery**:



The **Welcome to BigQuery in the Cloud Console** message box opens. This message box provides a link to the quickstart guide and the release notes.

Click **Done**.

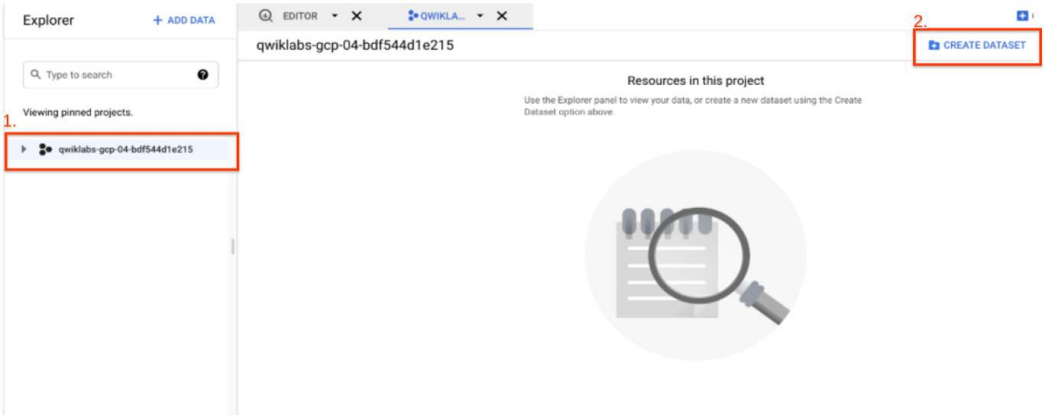
The BigQuery console opens.



Create a new dataset

First, you will create a dataset to store your tables.

Click on your project name, then click **Create Dataset**.



Name your dataset **ecommerce**. Leave the other options at their default values (Data Location, Default table Expiration).

Click **Create dataset**.

Click **Check my progress** to verify the objective.

Create a dataset named ecommerce

Check my progress

Creating tables with date partitions

A partitioned table is a table that is divided into segments, called partitions, that make it easier to manage and query your data. By dividing a large table into smaller partitions, you can improve query performance, and control costs by reducing the number of bytes read by a query.

Now you will create a new table and bind a date or timestamp column as a partition. Before we do that, let's explore the data in the non-partitioned table first.

Query webpage analytics for a sample of visitors in 2017

In the **Query Editor**, add the below query. Before running, note the total amount of data it will process as indicated next to the query validator icon: "This query will process 1.74 GB when run".

```
#standardSQL
SELECT DISTINCT
  fullVisitorId,
  date,
  city,
  pageTitle
FROM `data-to-insights.ecommerce.all_sessions_raw`
WHERE date = '20170708'
LIMIT 5
```

Click **Run**.

The query returns 5 results.

Query webpage analytics for a sample of visitors in 2018

Let's modify the query to look at visitors for 2018 now.

Click **COMPOSE NEW QUERY** to clear the **Query Editor**, then add this new query. Note the `WHERE date` parameter is changed to `20180708`:

```
#standardSQL
SELECT DISTINCT
  fullVisitorId,
  date,
  city,
  pageTitle
FROM `data-to-insights.ecommerce.all_sessions_raw`
WHERE date = '20180708'
LIMIT 5
```

The **Query Validator** will tell you how much data this query will process.

Click **Run**.

Notice that the query still processes 1.74 GB even though it returns 0 results. Why? The query engine needs to scan all records in the dataset to see if they satisfy the date matching condition in the WHERE clause. It must look at each record to compare the date against the condition of '20180708'.

Additionally, the LIMIT 5 does not reduce the total amount of data processed, which is a common misconception.

Why did the previous query return 0 records but still scan through 1.74GB of data?
close~~The query was written incorrectly~~

checkBefore the query runs, the query engine does not know whether 2018 data exists to satisfy the WHERE clause condition and it needs to scan through all records in a non-partitioned table.

The query engine has the metadata for each partition stored but still needs to scan all records even if the table is partitioned.

Submit

Common use-cases for date-partitioned tables

Scanning through the entire dataset everytime to compare rows against a WHERE condition is wasteful. This is especially true if you only really care about records for a specific period of time like:

- All transactions for the last year
- All visitor interactions within the last 7 days
- All products sold in the last month

Instead of scanning the entire dataset and filtering on a date field like we did in the earlier queries, we will now setup a date-partitioned table. This will allow us to completely ignore scanning records in certain partitions if they are irrelevant to our query.

Create a new partitioned table based on date

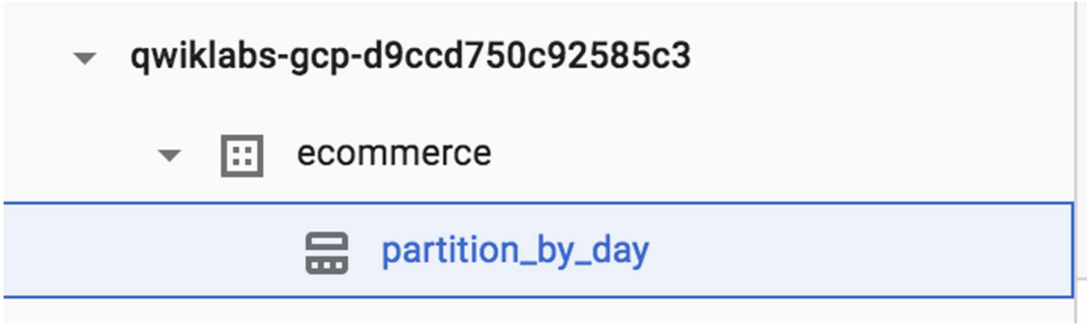
Click **COMPOSE NEW QUERY** and add the below query, then **Run**:

```
#standardSQL
 CREATE OR REPLACE TABLE ecommerce.partition_by_day
 PARTITION BY date_formatted
 OPTIONS(
   description="a table partitioned by date"
 ) AS

 SELECT DISTINCT
 PARSE_DATE("%Y%m%d", date) AS date_formatted,
 fullvisitorId
 FROM `data-to-insights.ecommerce.all_sessions_raw`
```

In this query, note the new option - PARTITION BY a field. The two options available to partition are DATE and TIMESTAMP. The PARSE_DATE function is used on the date field (stored as a string) to get it into the proper DATE type for partitioning.

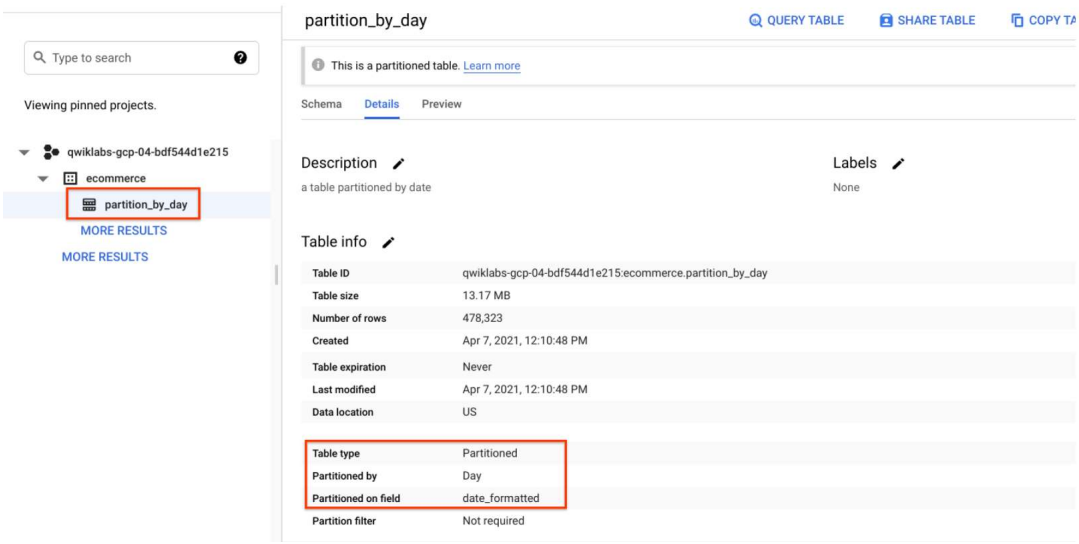
Click on the **ecommerce** dataset, then select the new **partiton_by_day** table:



Click on the **Details** tab.

Confirm that you see:

- Partitioned by: Day
- Partitioning on: date_formatted



Note: Partitions within partitioned tables on your Qwiklabs account will auto-expire after 60 days from the value in your date column. Your personal Google Cloud account with billing-enabled will let you have partitioned tables that don't expire. For the purposes of this lab, the remaining queries will be ran against partitioned tables that have already been created.

Click **Check my progress** to verify the objective.

Create a new partitioned table based on date

Check my progress

View data processed with a partitioned table

Run the below query, and note the total bytes to be processed:

```
#standardSQL
SELECT *
FROM `data-to-insights.ecommerce.partition_by_day`
WHERE date_formatted = '2016-08-01'
```

This time 25 KB or 0.025MB is processed, which is a fraction of what you queried.

Now run the below query, and note the total bytes to be processed:

```
#standardSQL
SELECT *
FROM `data-to-insights.ecommerce.partition_by_day`
WHERE date_formatted = '2018-07-08'
```

You should see `This query will process 0 B when run.`

Why was there 0 bytes processed?
close~~The query is running from query cache~~
checkThe query engine knows which date partitions exist before the query is ran (and there is no 2018 partitions)

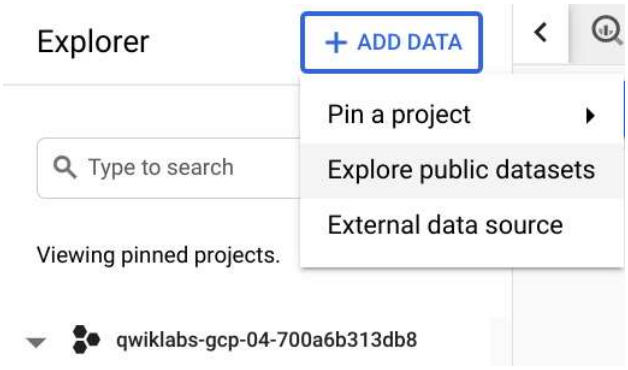
The query is running from a saved View
Submit

Creating an auto-expiring partitioned table

Auto-expiring partitioned tables are used to comply with data privacy statutes, and can be used to avoid unnecessary storage (which you'll be charged for in a production environment). If you want to create a rolling window of data, add an expiration date so the partition disappears after you're finished using it.

Explore the available NOAA weather data tables

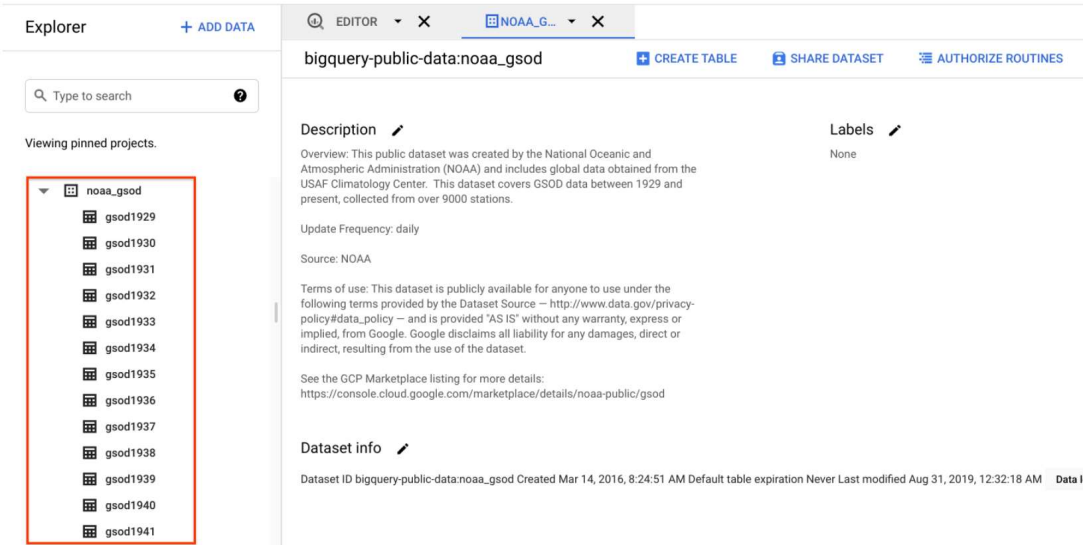
In the left menu, in Explorer, click on **Add Data** and select **Explore public datasets**.



Search for "GSOD NOAA" then select the dataset.

Click on **View Dataset**.

Scroll through the tables in the **noaa_gsod** dataset (which are manually sharded and not partitioned)



Your goal is to create a table that:

- Queries on weather data from 2018 onward
- Filters to only include days that have had some precipitation (rain, snow, etc.)
- Only stores each partition of data for 90 days from that partition's date (rolling window)

First, **copy and paste** this below query:

```
#standardSQL
 SELECT
   DATE(CAST(year AS INT64), CAST(mo AS INT64), CAST(da AS INT64)) AS date,
   (SELECT ANY_VALUE(name) FROM `bigquery-public-data.noaa_gsod.stations` AS stations
    WHERE stations.usaf = stn) AS station_name,  -- Stations may have multiple names
   prcp
 FROM `bigquery-public-data.noaa_gsod.gsod*` AS weather
 WHERE prcp < 99.9  -- Filter unknown values
   AND prcp > 0      -- Filter stations/days with no precipitation
   AND CAST(_TABLE_SUFFIX AS int64) >= 2018
 ORDER BY date DESC -- Where has it rained/snowed recently
 LIMIT 10
```

Note that the table wildcard * used in the FROM clause to limit the amount of tables referred to in the *TABLE_SUFFIX* filter.

Note that although a LIMIT 10 was added, this still does not reduce the total amount of data scanned (about 1.83 GB) since there are no partitions yet.

Click **Run**.

Confirm the date is properly formatted and the precipitation field is showing non-zero values.

Your turn: Create a Partitioned Table

Modify the previous query to create a table with the below specifications:

- Table name: ecommerce.days_with_rain
- Use the date field as your PARTITION BY
- For OPTIONS, specify partition_expiration_days = 60
- Add the table description = "weather stations with precipitation, partitioned by day"

Your query should look like this:

```
#standardSQL
 CREATE OR REPLACE TABLE ecommerce.days_with_rain
 PARTITION BY date
 OPTIONS (
   partition_expiration_days=60,
   description="weather stations with precipitation, partitioned by day"
 ) AS


 SELECT
   DATE(CAST(year AS INT64), CAST(mo AS INT64), CAST(da AS INT64)) AS date,
   (SELECT ANY_VALUE(name) FROM `bigquery-public-data.noaa_gsod.stations` AS stations
    WHERE stations.usaf = stn) AS station_name,  -- Stations may have multiple names
   prcp
 FROM `bigquery-public-data.noaa_gsod.gsod*` AS weather
 WHERE prcp < 99.9  -- Filter unknown values
   AND prcp > 0      -- Filter
   AND CAST( _TABLE_SUFFIX AS int64) >= 2018
```

Click **Check my progress** to verify the objective.

Your turn: Create a Partitioned Table

Check my progress

Confirm data partition expiration is working

To confirm you are only storing data from 60 days in the past up until today, run the DATE_DIFF query to get the age of your partitions, which are set to expire after 60 days.

Below is a query which tracks the average rainfall for the NOAA weather station in Wakayama, Japan which has significant precipitation.
Add this query and run it:

```
#standardSQL
# avg monthly precipitation
SELECT
  AVG(prcp) AS average,
  station_name,
  date,
  CURRENT_DATE() AS today,
  DATE_DIFF(CURRENT_DATE(), date, DAY) AS partition_age,
  EXTRACT(MONTH FROM date) AS month
FROM ecommerce.days_with_rain
WHERE station_name = 'WAKAYAMA' #Japan
GROUP BY station_name, date, today, month, partition_age
ORDER BY date DESC; # most recent days first
```

Confirm the oldest partition_age is at or below 60 days

Update the ORDER BY clause to show the oldest partitions first. The date you see there
Add this query and run it:

```
#standardSQL
# avg monthly precipitation

SELECT
  AVG(prcp) AS average,
  station_name,
  date,
  CURRENT_DATE() AS today,
  DATE_DIFF(CURRENT_DATE(), date, DAY) AS partition_age,
  EXTRACT(MONTH FROM date) AS month
FROM ecommerce.days_with_rain
WHERE station_name = 'WAKAYAMA' #Japan
GROUP BY station_name, date, today, month, partition_age
ORDER BY partition_age DESC
```

Note: Your results will vary if you re-run the query in the future, as the weather data, and your partitions, are continuously updated.

Congratulations!

You've successfully created and queried partitioned tables in BigQuery.

Finish Your Quest

This self-paced lab is part of the Qwiklabs BigQuery for Data Warehousing Quest. A Quest is a series of related labs that form a learning path. Completing this Quest earns you the badge above, to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. Enroll in this Quest and get immediate completion credit if you've taken this lab. See other available Qwiklabs Quests.

Take Your Next Lab

Continue your Quest with Troubleshooting and Solving Data Join Pitfalls, or check out these suggestions:

- Working with JSON, Arrays, and Structs in BigQuery
- Predict Taxi Fare with a BigQuery ML Forecasting Model

Next Steps / Learn More

If you are curious about how to create ingestion-time partitioned tables that are not bound to a specific date or timestamp column, refer to the BigQuery Partition documentation and examples.
Already have a Google Analytics account and want to query your own datasets in BigQuery? Follow this export guide.

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. Our classes include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. Certifications help you validate and prove your skill and expertise in Google Cloud technologies.
Manual Last Updated: April 19, 2021
Lab Last Tested: April 19, 2021