Activity Overview

In this activity, you will use BigQuery to partition data and create an index. Partitions and indexes help you optimize a database by creating shortcuts to specific rows and dividing large datasets into smaller, more manageable tables.

By creating partitions and indexes, you can make faster and more efficient databases. This will make it easier to pull your data when you need to analyze or visualize it.

Be sure to complete this activity before moving on. The next course item will provide you with a completed exemplar to compare to your own work. You will not be able to access the exemplar until you have completed this activity.

Scenario

Review the following scenario. Then complete the step-by-step instructions.

You are a BI analyst for a grocery store chain that monitors dietary trends affecting in-store purchases. Your company wants you to examine which types of Hass avocados are purchased most often. The avocados are categorized as one of four sizes: small, medium, large, and extra large. In addition to the average price and total volume of each avocado, the date of each sale is also recorded.

Using this data, you will create a historical table that illustrates how indexes and partitions work. This will allow you to practice creating partitions and clustered tables and demonstrate how to use them.

Your goal is to use partitions and clusters to answer the following question: What is the distribution of avocado sales from 2015 to 2021?

Step-By-Step Instructions

Follow the instructions to complete each step of the activity. Then, answer the questions at the end of the activity before going to the next course item to compare your work to a completed exemplar.

Part 1: Set up in BigQuery

Step 1: Access the data

To use the data for this course item, download the dataset from Kaggle - Avocado Sales 2015-2021 (US centric).

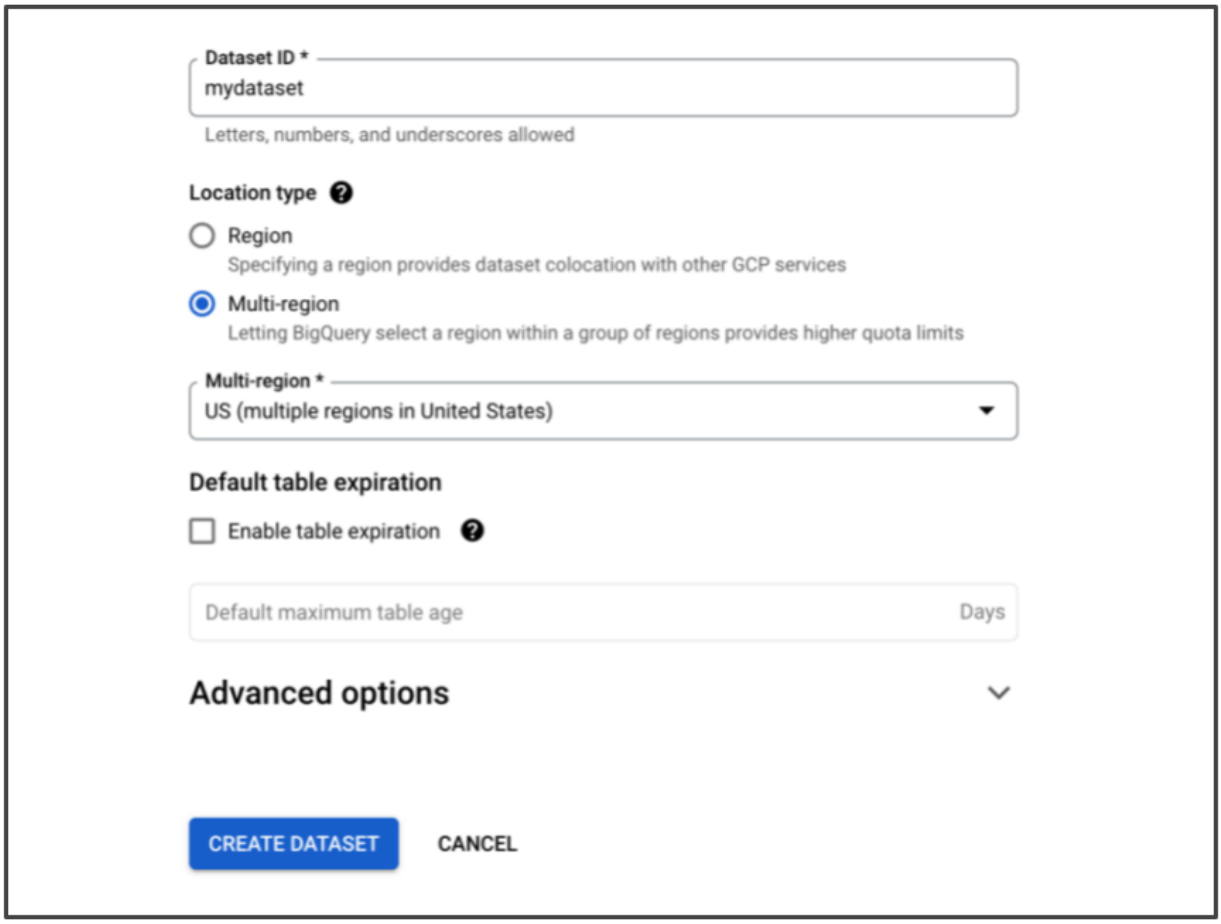
avocado cleaned
CSV File

Step 2: Open the BigQuery console

Navigate to your BigQuery console.

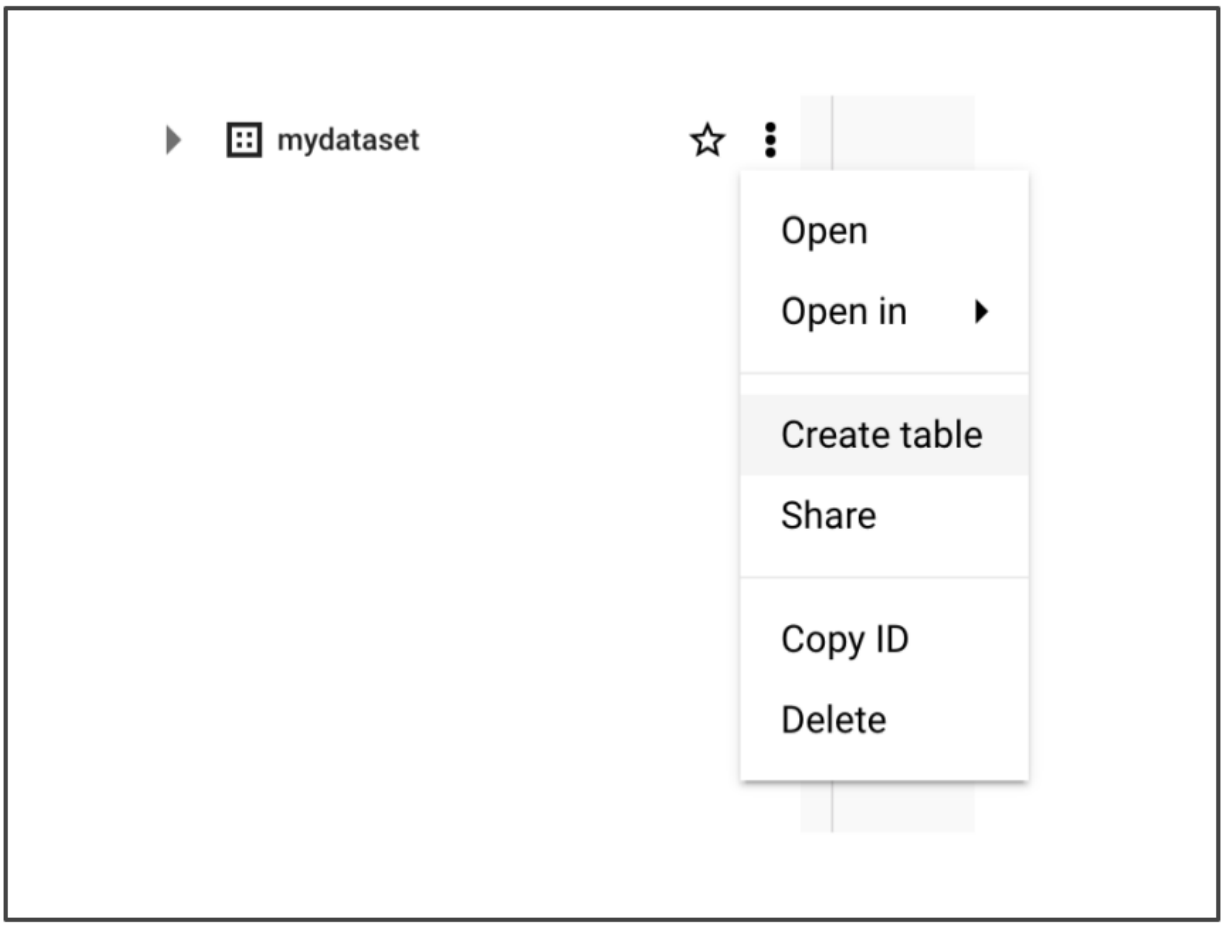
Step 3: Create a dataset

In the **Explorer** menu, find one of your projects. It may be titled "My First Project" or a title you gave it. Click the three dot icon, then select **Create dataset**. Fill in "mydataset" for the Dataset ID and set the location to "us (multiple regions in United States)." Then select **Create dataset**.

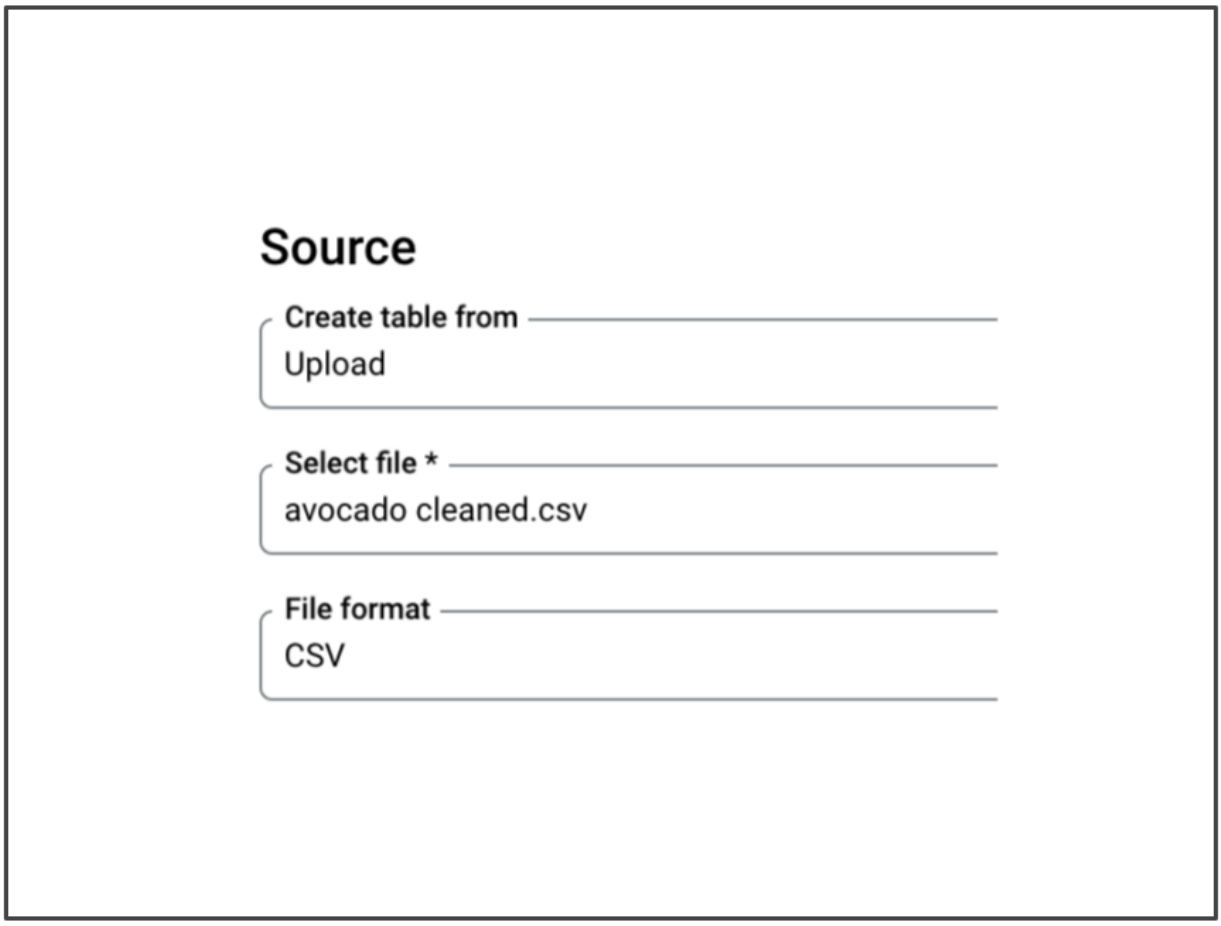


Step 4: Load the avocado data into a table

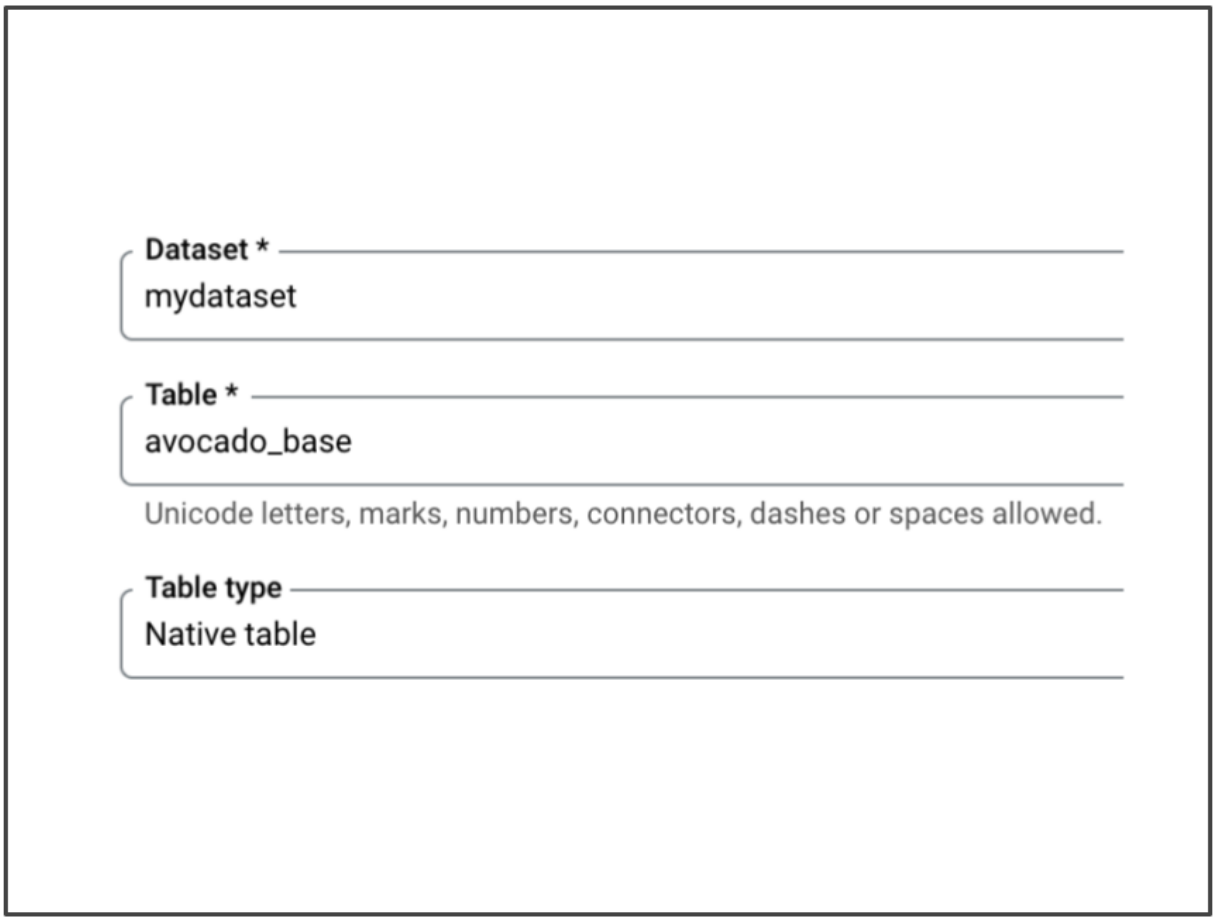
Next to **mydataset**, click the three-dot icon and select **Create table**.



Next, use the **Create table from** the dropdown menu and select **Upload**. Choose the CSV file you downloaded earlier in this activity.

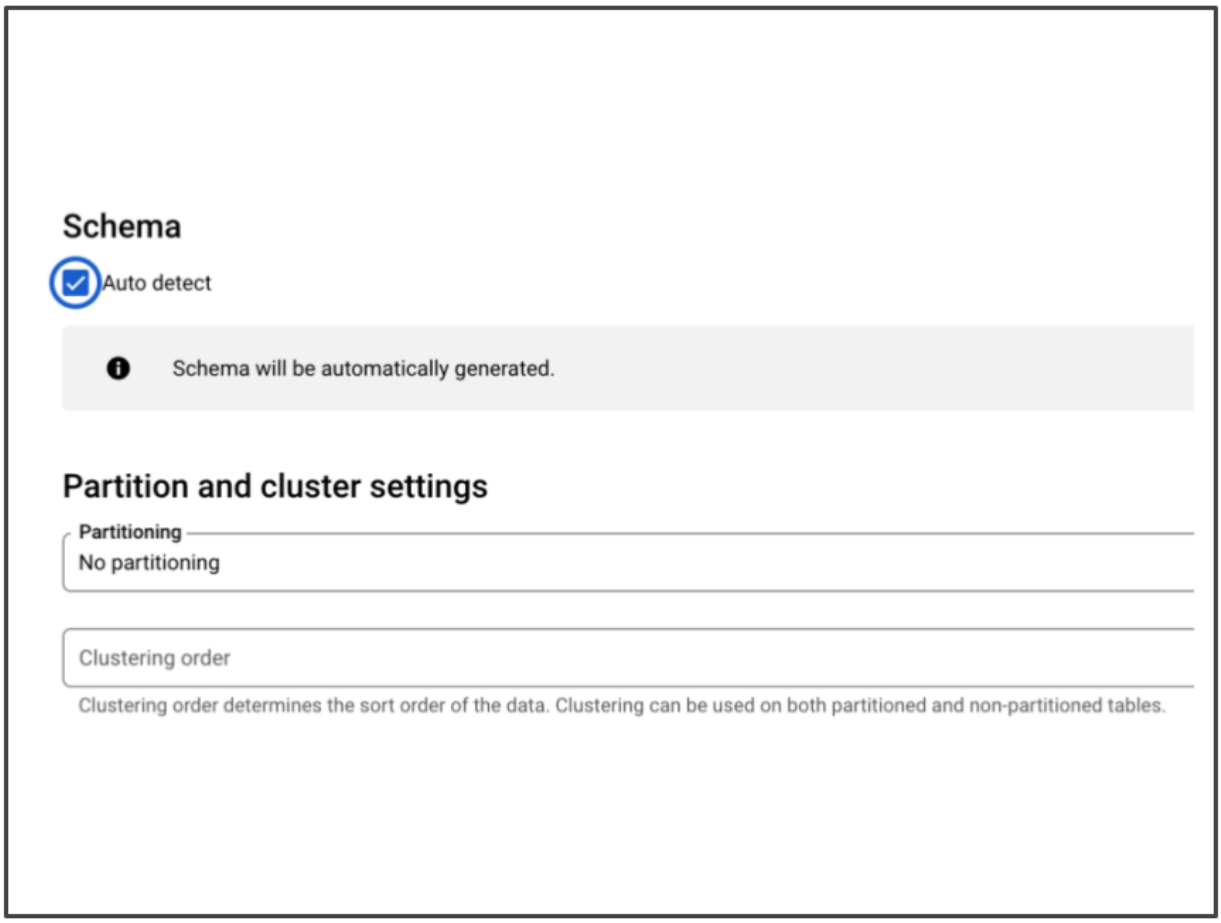


Then, name the table avocado_base." Make sure the **Dataset** field reads "mydataset" and the **Table type** field reads "Native table."



In the **Schema** section of the interface, check the box for **Auto detect**.

Then select **Create table**.



Part 2: Create tables with partitions and clusters

Step 1: Create a table without a partition or cluster

To begin, create a new table without a partition or cluster. This will serve as a baseline to compare to the partitioned and clustered tables. Name it **avocados**.

Then, in the **Editor** tab, copy and paste the following SQL code and click **Run**.

```
7
4
5
6
1
2
3
);
    SELECT
```

```
        *
    FROM `mydataset.avocado_base`
CREATE TABLE
    `mydataset.avocados`
AS (
```

When you finish running the code, switch to the **Results** tab. Click **Go to table** and take note of the **Details** pane. Save the details for later by taking a screenshot or copying and pasting the information into another document. The dates on your screen might differ, but the table size, long-term storage size, and number of rows should be the same as in the following image.

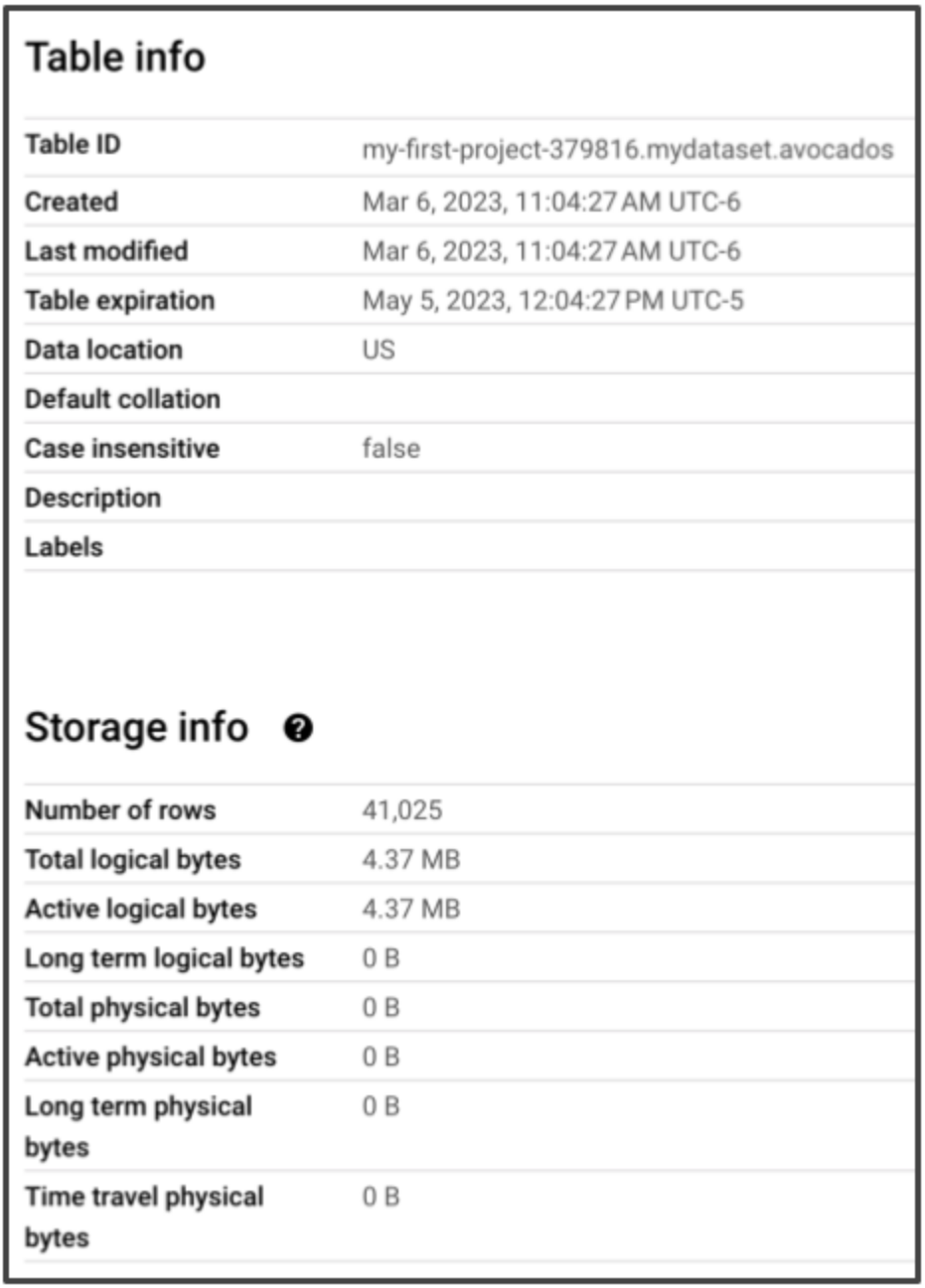


Table info

Table ID	my-first-project-379816.mydataset.avocados
Created	Mar 6, 2023, 11:04:27 AM UTC-6
Last modified	Mar 6, 2023, 11:04:27 AM UTC-6
Table expiration	May 5, 2023, 12:04:27 PM UTC-5
Data location	US
Default collation	
Case insensitive	false
Description	
Labels	

Storage info

Number of rows	41,025
Total logical bytes	4.37 MB
Active logical bytes	4.37 MB
Long term logical bytes	0 B
Total physical bytes	0 B
Active physical bytes	0 B
Long term physical bytes	0 B
Time travel physical bytes	0 B

Step 2: Create a table with a partition

Next, create a table partitioned by an integer range (the years 2015 through 2022). Name it **avocados_partitioned**.

Return to the tab you entered the SQL code into. Delete that code then copy and paste the following SQL code. Click **Run**.

```
CREATE TABLE
    `mydataset.avocados_partitioned`
```

```
PARTITION BY
    RANGE_BUCKET(Year, GENERATE_ARRAY(2015,2022,1))
AS (
    SELECT
        *
    FROM `mydataset.avocado_base`
);
```

When you finish running the code, switch to the **Results** tab. Click **Go to table** and take note of the **Details** pane. Save the details for later by taking a screenshot or copying and pasting the information into another document. After this activity, you'll compare this to the exemplar.

Step 3: Create a table with a partition and a cluster

Next, create a table partitioned by an integer range and clustered by type. Name it **avocados_clustered**.

Return to the tab where you entered the SQL code. Delete that code, then copy and paste the following SQL code. Click **Run**.

```
CREATE TABLE
    `mydataset.avocados_clustered`
PARTITION BY
    RANGE_BUCKET(Year, GENERATE_ARRAY(2015,2022,1))
CLUSTER BY
    type
AS (
    SELECT
        *
    FROM `mydataset.avocado_base`
);
```

When you finish running the code, switch to the **Results** tab. Click **Go to table** and take note of the **Details** pane. Save the details for later by taking a screenshot or copying and pasting the information into another document. After this activity, you'll compare this to the exemplar.

Part 3: Query the tables and compare performance

Step 1: Query the table without a partition or cluster

Delete the code in the **Editor** tab, then copy and paste the following code. Click **Run** to query avocados—the table without partition or cluster.

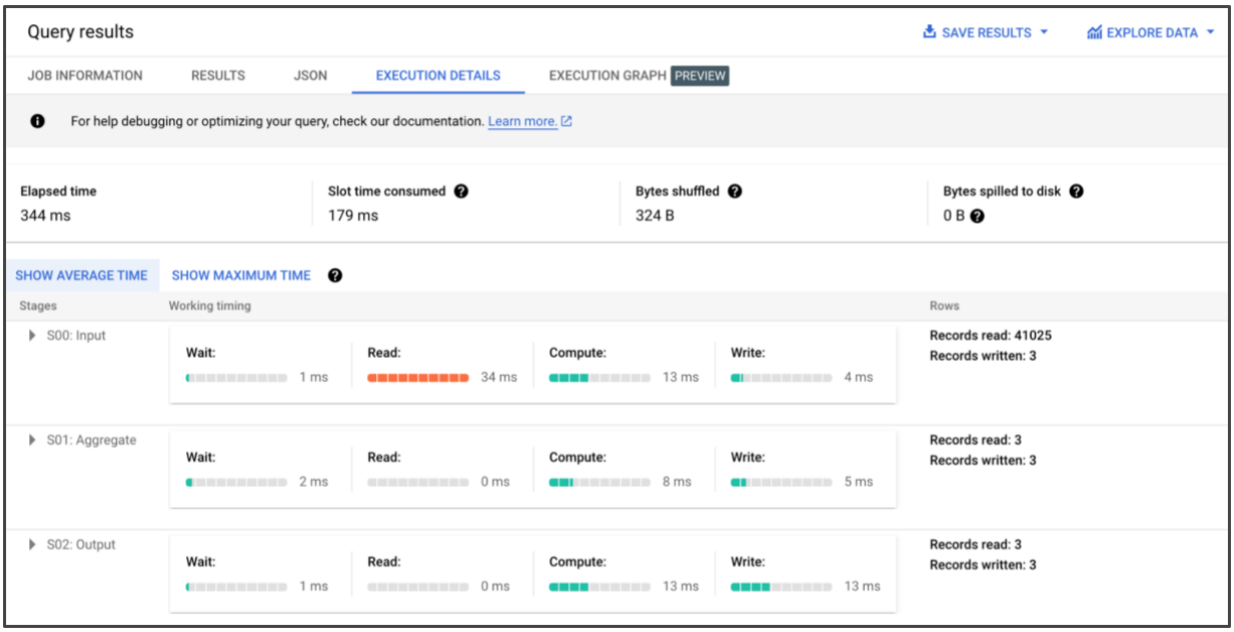
```
SELECT

    year,
```

```
    COUNT(*) AS number_avocados,
    SUM(TotalVolume) AS sum_totalVolume,
    SUM(AveragePrice) AS sum_AveragePrice
FROM `mydataset.avocados`
WHERE type = 'organic'
GROUP BY year
ORDER BY year ASC;
```

When the query has finished running, check the **Execution** details tab. This explains that the number of records read is the total number of records in the table. In this query, the database processes all records from the table. This is reflected in S00:Input.

Note: The **Working timing** section on your screen might vary in color or duration. Your SQL query might take longer or shorter to run depending on differing BigQuery engine server speeds. Your screen might not match the following screenshot, but the records read and records written should match with the **Rows** section.



In the next steps, take note of the S00 and S01 rows as described in the preceding screenshot. You will need to compare these details to the exemplar.

Step 2: Query the partitioned table

Delete the code in the **Editor** tab, then copy and paste the following code. Click **Run** to query **avocados_partitioned**—the table that is partitioned by an integr range.

```
SELECT
    year,
    COUNT(*) AS number_avocados,
    SUM(TotalVolume) as sum_TotalVolume,
```

```
    SUM(AveragePrice) as sum_AveragePrice
FROM `mydataset.avocados_partitioned`
WHERE type = 'organic'
GROUP BY year
ORDER BY year ASC;
```

When the query has finished running, check the **Execution** details tab and save a screenshot of it. You'll need to compare these details to the exemplar.

Step 3: Query the partitioned and clustered table

Delete the code in the **Editor** tab, then copy and paste the following code. Click **Run** to query **avocados_clustered**—the table that is partitioned by an integer range and clustered by type.

```
SELECT
    year,
    COUNT(*) AS number_avocados,
    SUM(TotalVolume) as sum_TotalVolume,
    SUM(AveragePrice) as sum_AveragePrice
FROM `mydataset.avocados_clustered`
WHERE type = 'organic'
GROUP BY year
ORDER BY year ASC;
```

When the query has finished running, check the **Execution** details tab and save a screenshot of it. You will need to compare these details to the exemplar.

What to Include in Your Response

You should record the following in your SQL code results:

- A screenshot of the **Details** pane of the **avocados_partitioned** table
- A screenshot of the **Details** pane of the **avocados_clustered** table
- A screenshot of the **Execution Details** pane of the **avocados_partitioned** table
- A screenshot of the **Execution Details** pane of the **avocados_clustered** table

In addition to this criteria, in a business role you might consider including a report that describes the distribution of avocados over the six-year time period and if there are any relationships between avocado size, type, and total volume sold. You could also share your recommendations based on any trends you find in the data, in order to anticipate future demand.