Hands-On Activity: SQL sorting queries

coursera.org/learn/analyze-data/quiz/iAg3O/hands-on-activity-sql-sorting-queries/view-attempt

✓

Congratulations! You passed!

Grade received 100%

To pass 100% or higher

Activity Overview

So far, you've learned about SQL and how it's used to retrieve data from databases. In this activity, you'll practice sorting data with the *ORDER BY* clause in SQL. Sorting is a powerful tool for a data analyst. It enables you to:

- Organize and analyze your data in a meaningful way
- Find highest or lowest values in a dataset
- Compare data across different dimensions

By the time you complete this activity, you will be able to write SQL queries that sort data depending on your needs.

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

You're a public health researcher with a state government agency. For your current project, you need to identify counties in the United States that have the most and least births in the 2016-2018 time frame. To do this, you'll complete the following steps:

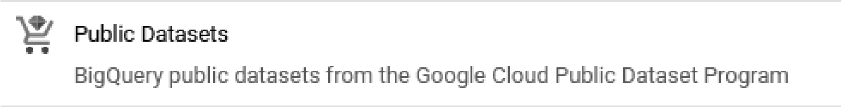
- Load the dataset.
- Query the data to explore its structure.
- Use *ORDER BY* to sort relevant data.
- Use sorted data to answer questions.

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.

1. Open the BigQuery console.

2. Select **+ADD** from the **Explorer** pane.

3. In the **Add** window, navigate to and then select **Public Datasets**.



Public Datasets

BigQuery public datasets from the Google Cloud Public Dataset Program

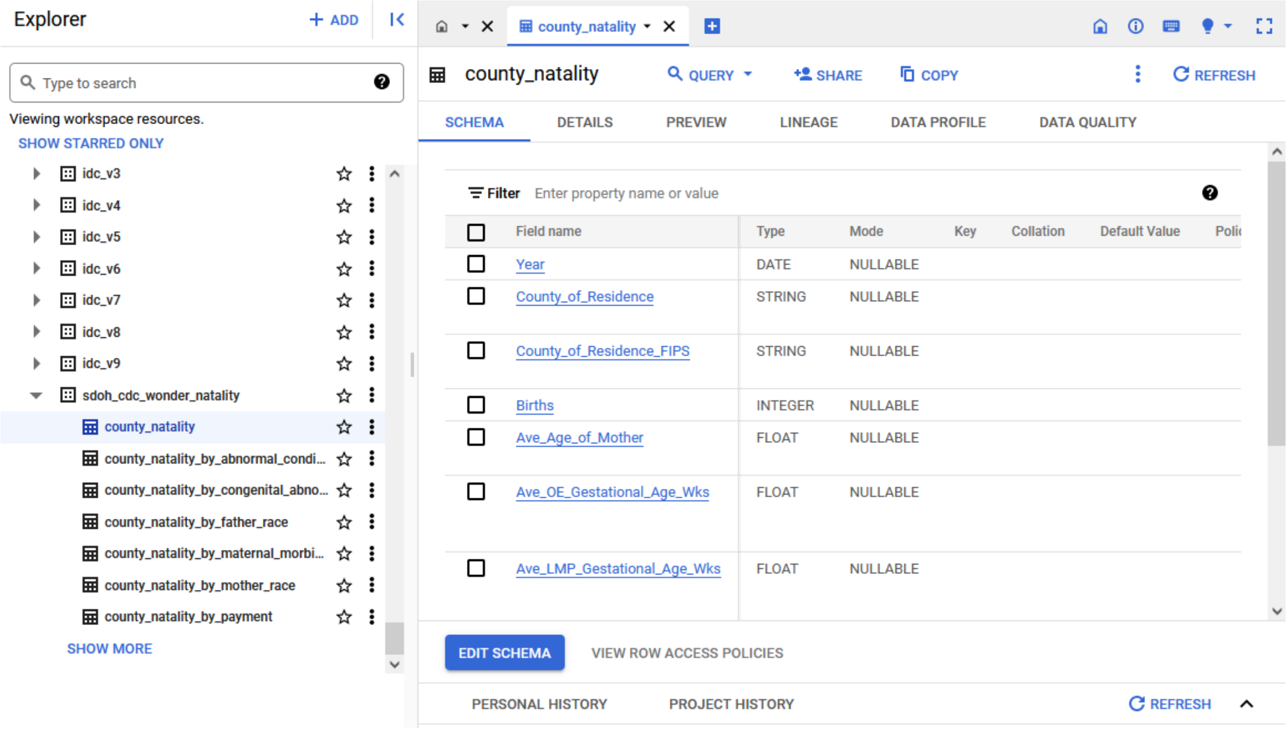
4. In the Marketplace search bar, enter **sdoh_cdc_wonder_natality** and press enter.

5. Select the result **Births Data Summary** from the CDC.

6. Select **VIEW DATASET**. This will bring you back to the console and open a **Dataset info** tab about the CDC dataset in the **Details** pane.

7. Select **sdoh_cdc_wonder_natality** in the **Explorer** pane to examine the tables available within the dataset.

8. Select the table **county_natality** and explore this table's schema, details, and preview.



□	Field name	Type	Mode	Key	Collation	Default Value	Poli
□	Year	DATE	NULLABLE				
□	County_of_Residence	STRING	NULLABLE				
□	County_of_Residence_FIPS	STRING	NULLABLE				
□	Births	INTEGER	NULLABLE				
□	Ave_Age_of_Mother	FLOAT	NULLABLE				
□	Ave_OE_Gestational_Age_Wks	FLOAT	NULLABLE				
□	Ave_LMP_Gestational_Age_Wks	FLOAT	NULLABLE				

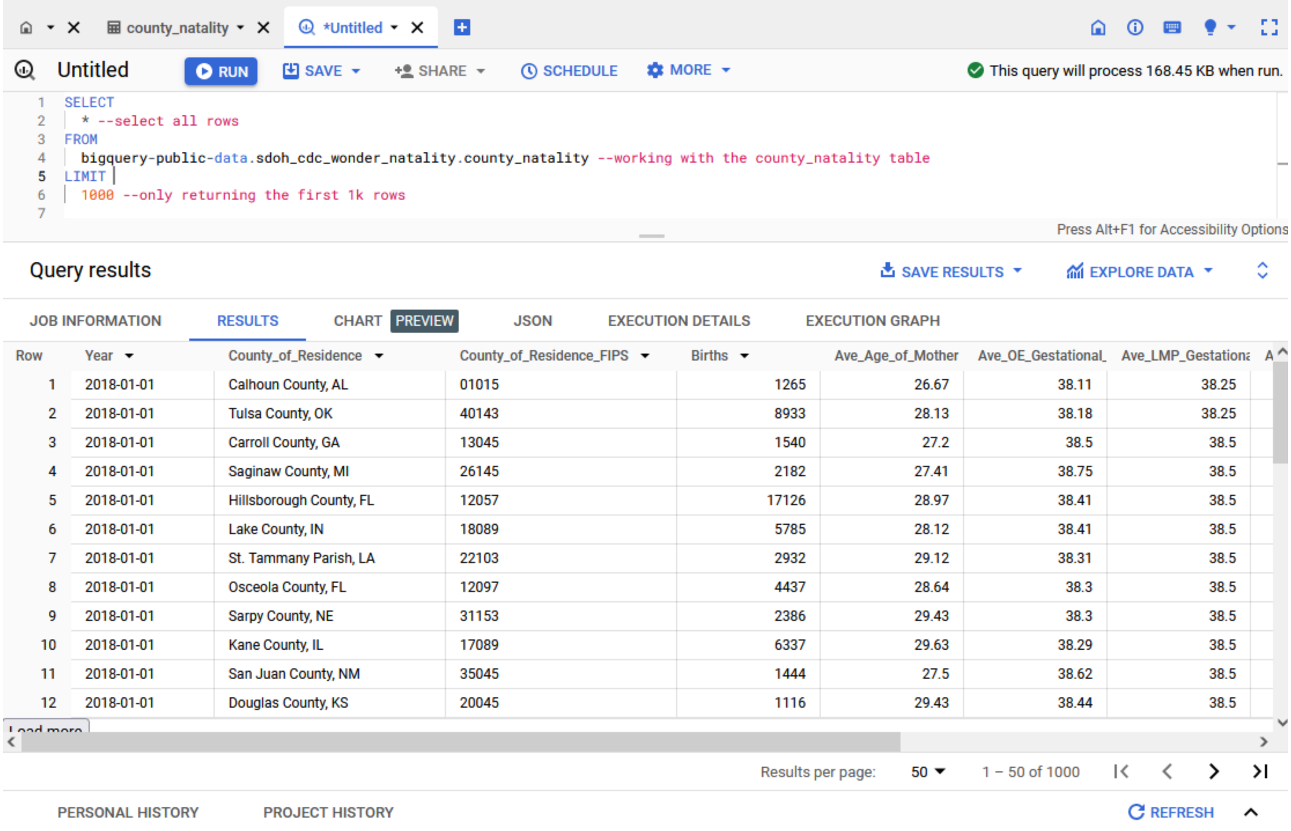
Now, it's time to start working with the CDC births data. First, run a query to examine the dataset without sorting it.

1. Select **Query**, then **In new tab**.

2. Enter the following query into the Query Editor to display the first 1,000 rows of the **county_natality** table.

```
SELECT
  *
FROM
  bigquery-public-data.sdoh_cdc_wonder_natality.county_natality
LIMIT
  1000
```

3. Select **RUN**.



This query will process 168.45 KB when run.

```
SELECT
  * --select all rows
FROM
  bigquery-public-data.sdoh_cdc_wonder_natality.county_natality --working with the county_natality table
LIMIT
  1000 --only returning the first 1k rows
```

Row	Year	County_of_Residence	County_of_Residence_FIPS	Births	Ave_Age_of_Mother	Ave_OE_Gestational_	Ave_LMP_Gestationa
1	2018-01-01	Calhoun County, AL	01015	1265	26.67	38.11	38.25
2	2018-01-01	Tulsa County, OK	40143	8933	28.13	38.18	38.25
3	2018-01-01	Carroll County, GA	13045	1540	27.2	38.5	38.5
4	2018-01-01	Saginaw County, MI	26145	2182	27.41	38.75	38.5
5	2018-01-01	Hillsborough County, FL	12057	17126	28.97	38.41	38.5
6	2018-01-01	Lake County, IN	18089	5785	28.12	38.41	38.5
7	2018-01-01	St. Tammany Parish, LA	22103	2932	29.12	38.31	38.5
8	2018-01-01	Osceola County, FL	12097	4437	28.64	38.3	38.5
9	2018-01-01	Sarpy County, NE	31153	2386	29.43	38.3	38.5
10	2018-01-01	Kane County, IL	17089	6337	29.63	38.29	38.5
11	2018-01-01	San Juan County, NM	35045	1444	27.5	38.62	38.5
12	2018-01-01	Douglas County, KS	20045	1116	29.43	38.44	38.5

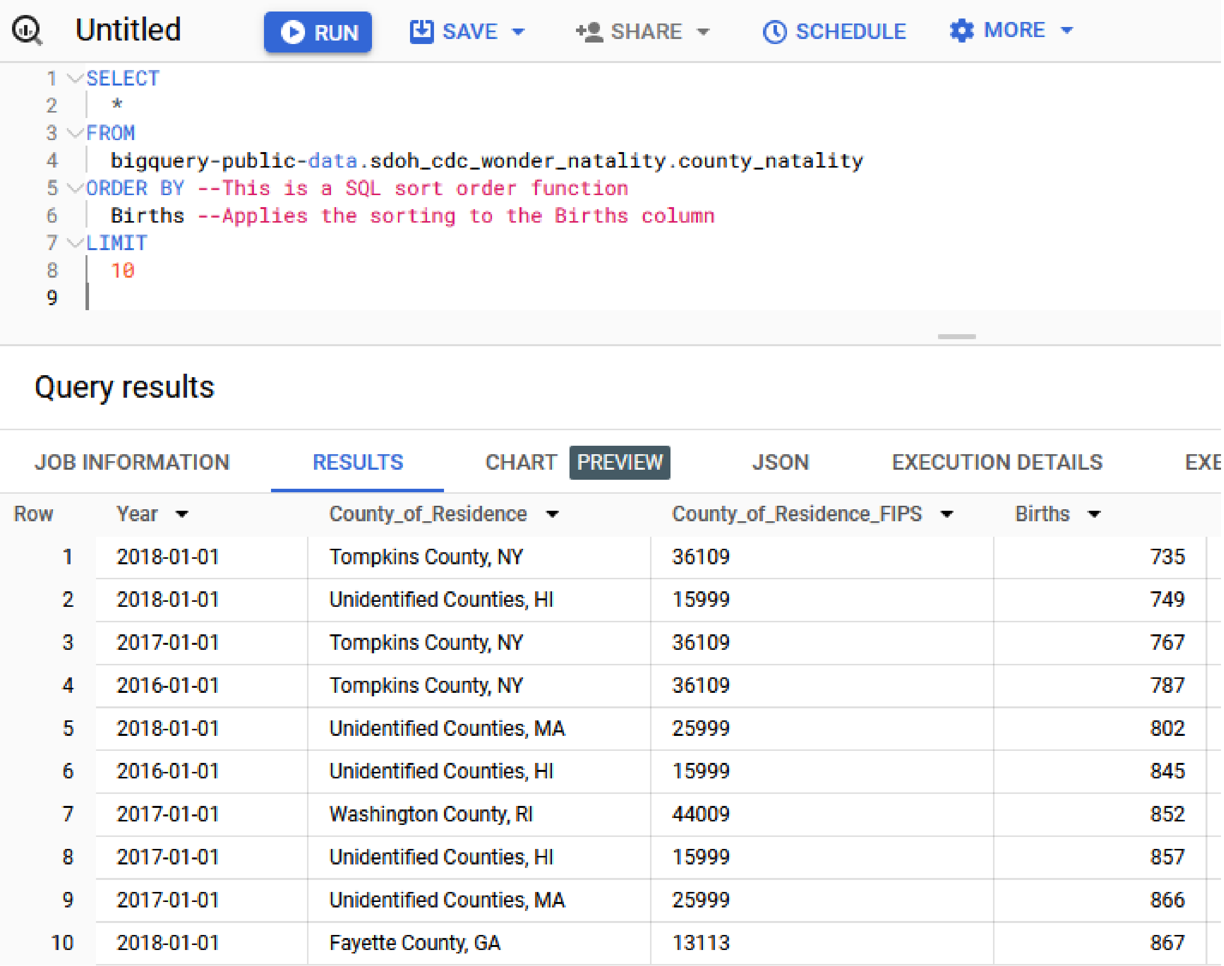
A screenshot of the Query results pane. The data returned include Year, County_of_Residence, County_of_Residence_FIPS, Births, Avg_Age_of_Mother, Ave_OE_Gestationa, and Ave_LMP_Gestational.

Examine the dataset you just loaded. Take a moment to familiarize yourself with the columns and fields available.

Note: Many of the public databases on BigQuery are living records and, as such, are periodically updated with new data. Throughout this course (and others in this certificate program), if your results differ from those you encounter in videos or screenshots, there's a good chance it is due to a data refresh.

Now, sort the data with SQL's *ORDER BY* function. Enter the following query into the Query Editor. The text preceded by two hyphens (--) are comments that explain the code. Run the Query.

```
SELECT
  *
FROM
  bigquery-public-data.sdoh_cdc_wonder_natality.county_natality
ORDER BY --This is a SQL sort order function
  Births --Applies the sorting to the Births column
LIMIT
  10
```



```
SELECT
  *
FROM
  bigquery-public-data.sdoh_cdc_wonder_natality.county_natality
ORDER BY --This is a SQL sort order function
  Births --Applies the sorting to the Births column
LIMIT
  10

```

Row	Year	County_of_Residence	County_of_Residence_FIPS	Births
1	2018-01-01	Tompkins County, NY	36109	735
2	2018-01-01	Unidentified Counties, HI	15999	749
3	2017-01-01	Tompkins County, NY	36109	767
4	2016-01-01	Tompkins County, NY	36109	787
5	2018-01-01	Unidentified Counties, MA	25999	802
6	2016-01-01	Unidentified Counties, HI	15999	845
7	2017-01-01	Washington County, RI	44009	852
8	2017-01-01	Unidentified Counties, HI	15999	857
9	2017-01-01	Unidentified Counties, MA	25999	866
10	2018-01-01	Fayette County, GA	13113	867

Examine the Births column. Notice that it's sorted from smallest to largest. When the *ORDER BY* function is applied to sort a given column, SQL will default to sorting in ascending order, which orders items from smallest to largest.

If you want the largest number to appear first, then you'd want to specify the sort order to be descending by adding a command to the *ORDER BY* clause. You can make your code easier to read by using a command to specify either sort order. Here are the corresponding commands:

- *ASC* = Ascending
- *DESC* = Descending

Next, you'll use the same query, but this time you'll explicitly state the order of your *ORDER BY* function using *ASC*. Enter and run the following SQL query:

8

5

6

7

2

3

4

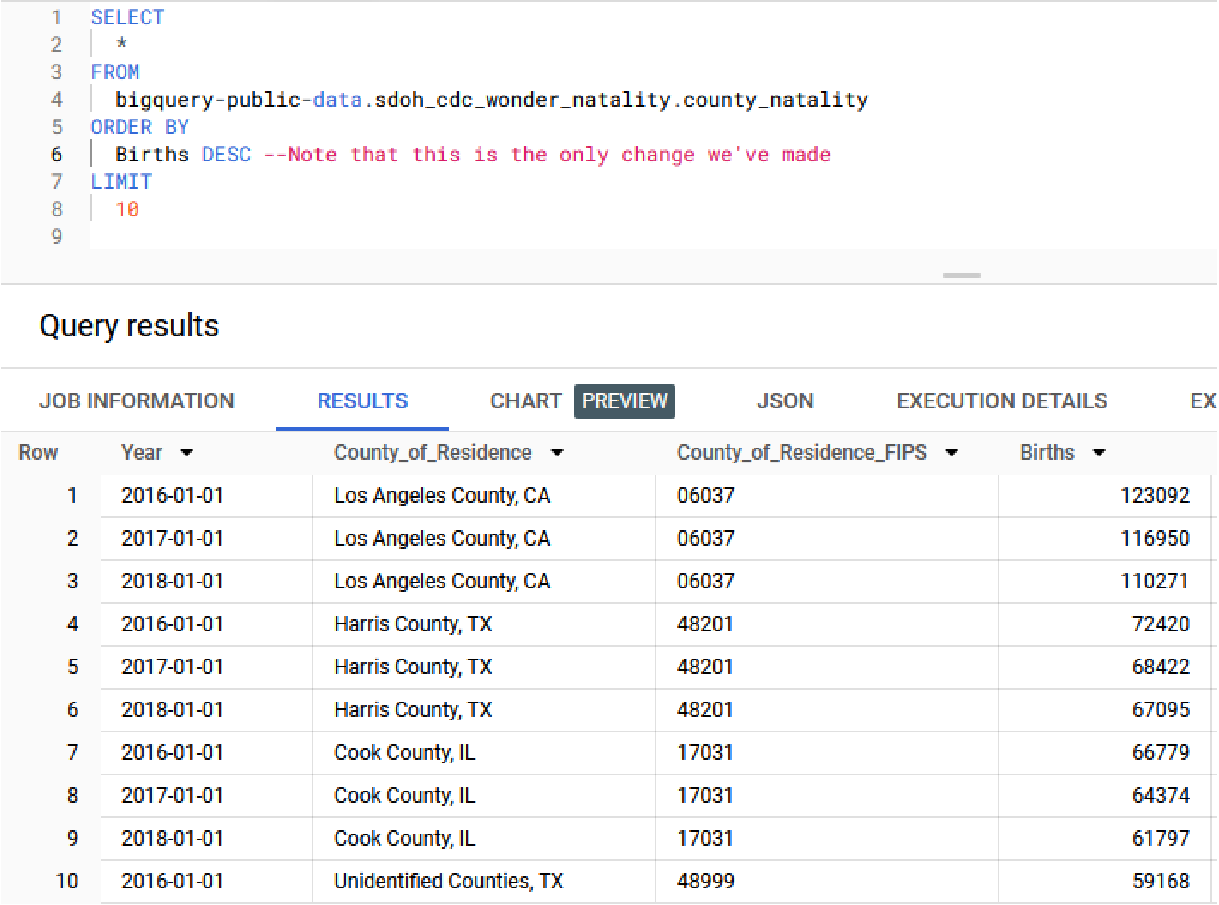
1

```
  10
ORDER BY
  Births ASC --
Place the ASC or DESC specifier directly after the column name separated by a space (no other punctuation)
LIMIT
  *
FROM
  bigquery-public-data.sdoh_cdc_wonder_natality.county_natality
SELECT
```

Notice that the results did not change. Tompkins County, NY, had just 735 births in 2018—the lowest birth count of any county in the US between 2016-2018.

Now, change the order from ascending (*ASC*) to descending (*DESC*) to find the most births. Enter and run this query:

```
SELECT
  *
FROM
  bigquery-public-data.sdoh_cdc_wonder_natality.county_natality
ORDER BY
  Births DESC --Note that this is the only change you've made
LIMIT
  10
```



```
SELECT
  *
FROM
  bigquery-public-data.sdoh_cdc_wonder_natality.county_natality
ORDER BY
  Births DESC --Note that this is the only change we've made
LIMIT
  10

```

Query results

Row	Year	County_of_Residence	County_of_Residence_FIPS	Births
1	2016-01-01	Los Angeles County, CA	06037	123092
2	2017-01-01	Los Angeles County, CA	06037	116950
3	2018-01-01	Los Angeles County, CA	06037	110271
4	2016-01-01	Harris County, TX	48201	72420
5	2017-01-01	Harris County, TX	48201	68422
6	2018-01-01	Harris County, TX	48201	67095
7	2016-01-01	Cook County, IL	17031	66779
8	2017-01-01	Cook County, IL	17031	64374
9	2018-01-01	Cook County, IL	17031	61797
10	2016-01-01	Unidentified Counties, TX	48999	59168

The query returns the 10 rows with the largest values in the Births column. Los Angeles County takes up the top three spots.

Now that you've become familiar with the basics of sorting functions, use them to answer questions about your data. This exercise will require you to apply both your previous learnings (especially filtering with the WHERE clause) and your new understanding of sorting.

In your work as a public health researcher, you're exploring whether the birth rate trends in several counties in upstate New York have been increasing or decreasing—and whether they follow the same pattern.

To answer this, you'll need the following information:

- Results from Erie, Niagara, and Chautauqua counties in New York state
- Results ordered by county of residence and year to find the trend

The following query will filter the results by county and sort the results byear and county. This will allow you to determine if the number of births is increasing or decreasing in each county.

Enter the following query into the Query Editor, then select **RUN**.

```
SELECT
  *
FROM
bigquery-public-data.sdoh_cdc_wonder_natality.county_natality
WHERE
  County_of_Residence = 'Erie County, NY'
  OR County_of_Residence = 'Niagara County, NY'
  OR County_of_Residence = 'Chautauqua County, NY'
ORDER BY
  County_of_Residence,
  Year
```

You’ve now successfully used both *ORDER BY* (sort) and *WHERE* (filter) clauses in the same query. Based on the results of this query, are births in these three counties following the same trend?

1.

Question 1

Reflection

The last query you ran returned births in three counties, sorted by year and county. Now, you want to identify the greatest number of births in Erie, Chautauqua, or Niagara counties between 2016 and 2018. Modify the previous query to order the data by Births in descending order to make this easy to identify. How many births occurred in the county with the highest number of births in one year?

1 / 1 point

✓
Correct

A total of 9,916 births occurred in Erie County in 2016. To arrive at this answer, you needed to sort by birth instead of county and year. Adding *DESC* to the *ORDER BY* clause would sort the Births column in descending order to make this information easier to identify.

```
SELECT
  *
FROM
  bigquery-public-data.sdoh_cdc_wonder_natality.county_natality
WHERE
  Year = '2017-01-01' --This filters our results to 2017
ORDER BY
  Births DESC --This sorts Births and puts the lowest number at the bottom of our list
LIMIT
  10
```

2.

Question 2
In this activity, you practiced sorting data using SQL queries with *ORDER BY* and filtering data with *WHERE* clauses. In the text box below, write 2-3 sentences (40-60 words) in response to each of the following questions:

- How can the *ORDER BY* clause help you organize and structure your data?
- Why is it helpful to use the *ORDER BY* and *WHERE* clauses together when analyzing data?
- Describe a business question that you could answer using the *ORDER By* and *WHERE* clauses together. How would this method help you answer the question?

1 / 1 point

ORDER BY sorts your data, making it easier to find trends and patterns. Used with WHERE, it becomes even more powerful. Imagine filtering customers by city and then ordering them by high spenders. This reveals the city with the most big spenders, answering your business question quickly.

✓
Correct

An effective response would include that sorting the data you return in your queries is a crucial tool for analyzing and understanding data.

You can also answer business questions by sorting the dataset according to a given metric. For instance, a store may want to know which products they sell the most or least. Sorting helps you answer business questions that involve phrases such as “how much,” “how many,” “best,” or “worst”—which will be a valuable skill in your career as a data analyst.