Find Duplicate Rows

by Sophia

WHAT'S COVERED

This lesson explores finding duplicate rows of data in a table, in three parts. Specifically, this lesson will cover:

1. Introduction

Finding duplicate data in a table can be quite useful, as it can help us identify potential issues or matches. A **duplicate row** is one that refers to the same thing or person as a whole other row. However, it is important to note that not all duplicate rows will have completely identical information, as it will depend on what columns of data we want to search on. For example, we may have a large employee table that stores the employee's Social Security number that uniquely identifies the employee in the U.S.

We could run a query to identify any instances of the same Social Security number appearing in multiple records. If we had customers, we could search for duplicate accounts with an email address, as traditionally, an email address should only belong to a single customer for most e-commerce sites. If we had multiple records for a single customer, it would be difficult to get a full order history for the customer, as they would have several rows that we would have to compare against.

TERM TO KNOW

Duplicate Row

A row that refers to the same instance (such as a person, thing, or event) as another row.

2. Finding Duplicates

The structure of the query to find duplicates looks like the following:

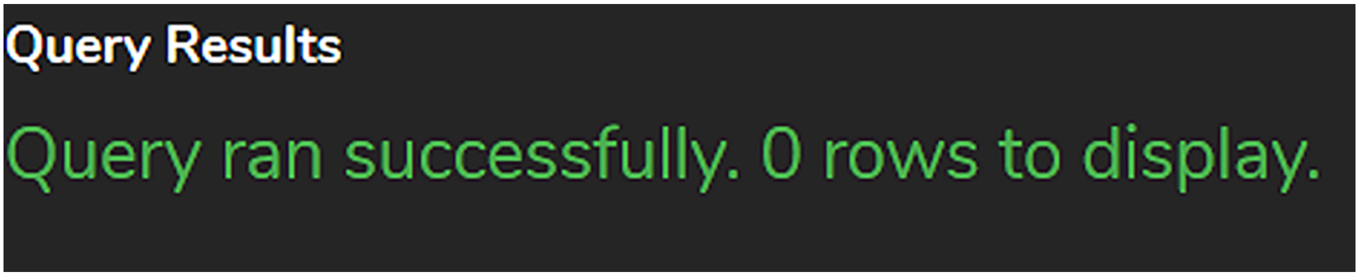
```
SELECT <columnlist>
FROM <tablename>
GROUP BY <columnlist>
HAVING COUNT(*) > 1;
```

Note that the column list in the SELECT clause should match the column list in the GROUP BY clause. In addition, we could add a COUNT(*) in the SELECT clause to identify how many duplicates there are of the same criteria that we grouped by.

For example, we want to verify that all of our customers have a unique phone number. We could do so like this:

```
SELECT phone, COUNT(*)
FROM customer
GROUP BY phone
HAVING COUNT(*) > 1;
```

We can verify that there are no customers that meet this criterion:

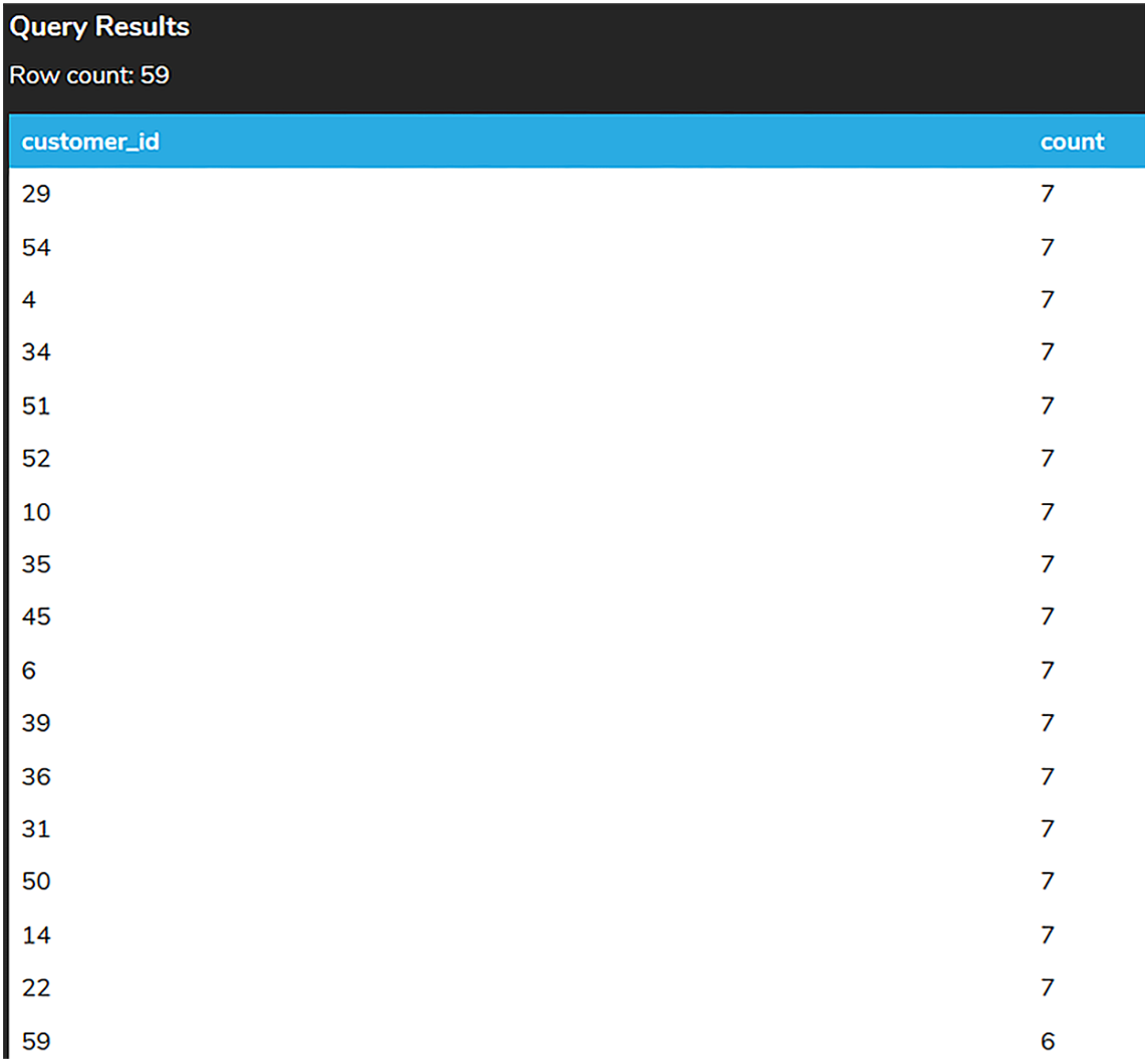


Query Results

Query ran successfully. 0 rows to display.

Similarly, we might want to know if there are customers who have placed multiple orders, and if so, how many. Here is a query that would produce that information.

```
SELECT customer_id, COUNT(*)
FROM invoice
GROUP BY customer_id
HAVING COUNT(*) > 1;
```



Query Results

Row count: 59

customer_id	count
29	7
54	7
4	7
34	7
51	7
52	7
10	7
35	7
45	7
6	7
39	7
36	7
31	7
50	7
14	7
22	7
59	6

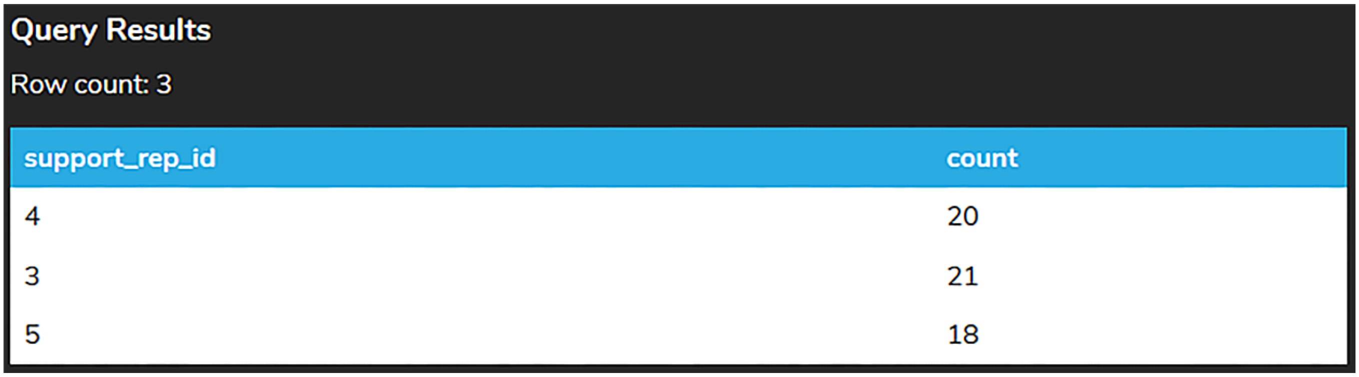
We are quickly and easily able to identify those types of scenarios.

3. Duplicates for Counting

Another way to find duplicates is to count the number of rows that meet certain criteria. For example, in the following query, we are counting the number of customers assigned to each support rep and listing the reps who have more than one customer assigned.

```
SELECT support_rep_id, COUNT(*)
FROM customer
```

```
GROUP BY support_rep_id
HAVING COUNT(*) > 1;
```



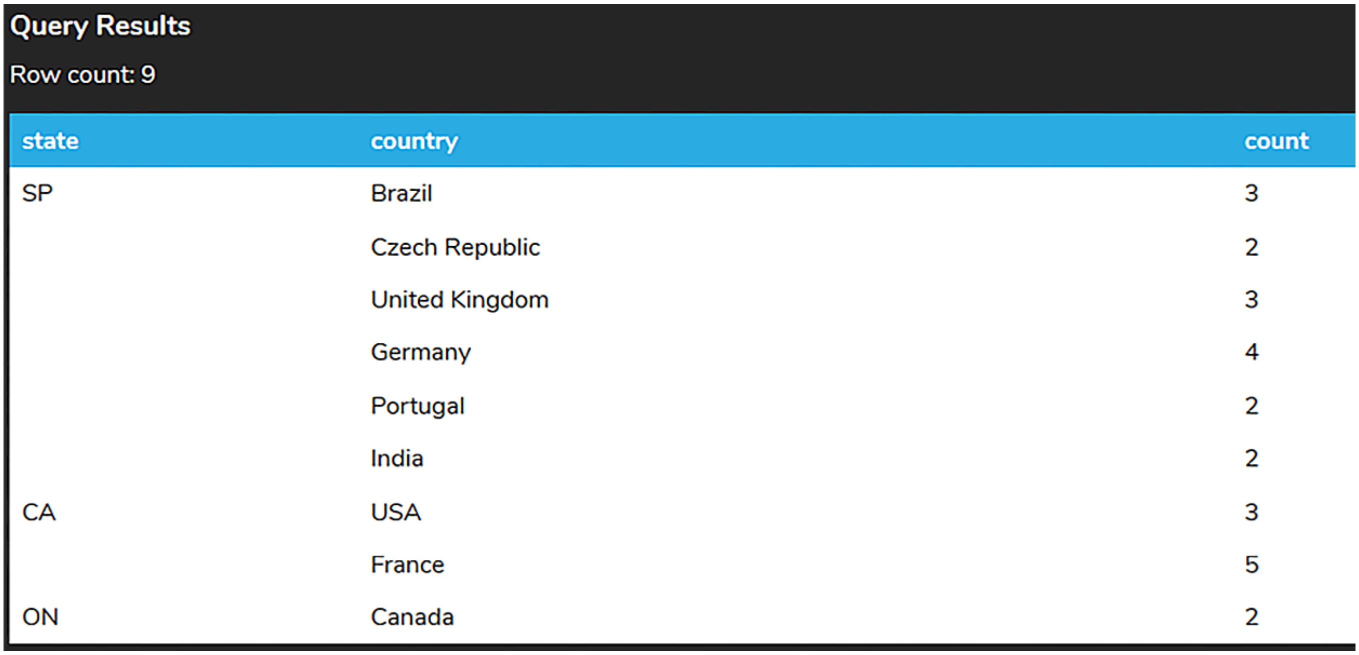
Query Results

Row count: 3

support_rep_id	count
4	20
3	21
5	18

And here's how we could list the states and countries that have more than one customer:

```
SELECT state, country, COUNT(*)
FROM customer
GROUP BY state, country
HAVING COUNT(*) > 1;
```



Query Results

Row count: 9

state	country	count
SP	Brazil	3
	Czech Republic	2
	United Kingdom	3
	Germany	4
	Portugal	2
	India	2
CA	USA	3
	France	5
ON	Canada	2

WATCH

TRY IT

Your turn! Open the SQL tool by clicking on the LAUNCH DATABASE button below. Then, enter one of the examples above and see how it works. Next, try your own choices for which columns you want the query to provide.

SUMMARY

In this lesson, in the **introduction** you learned that the process of **finding duplicates** in a table involves identifying and quantifying rows with similar values. In this process, you can gain insights into data quality, uncover potential anomalies, or support data cleansing efforts. SQL queries utilizing aggregation functions like COUNT() and GROUP BY are commonly used in conjunction with filtering conditions to isolate duplicate records. This would be accomplished by constructing a SQL query that groups the rows by the columns you're interested in and then applying the COUNT() function to each group. A group with a count greater than one represents duplicate records since the specified columns have the same values. When you execute the query, you will receive a list of duplicate values and their occurrence counts. You can also find duplicates by **counting** the number of rows that meet certain criteria. Using data patterns and potential data entry errors to identify duplicate records is useful for data analysis and decision making.

Source: THIS TUTORIAL WAS AUTHORED BY DR. VINCENT TRAN, PHD (2020) AND FAITHE WEMPEN (2024) FOR SOPHIA LEARNING. PLEASE SEE OUR **TERMS OF USE**.