OUTER JOINs

by Sophia

WHAT'S COVERED

In this lesson, you will explore using OUTER JOINs to query data from two or more tables, in two parts. Specifically, this lesson will cover:

1. FULL OUTER JOINs

OUTER JOINs can identify data that doesn't exist in one table by checking for NULL values in columns from the table where there is no match. In OUTER JOINs, there are unmatched rows from one or both tables, and these unmatched rows are represented in the result set with NULL values in the columns that correspond to the missing data.

In SQL, a **FULL OUTER JOIN** combines data from two tables based on a specified condition while including all records from both tables, regardless of whether they have matching counterparts. All rows from both tables are included in the result set, with NULL values filling in columns where there are no matches. A FULL OUTER JOIN provides a complete view of the combined data and allows for thorough analysis and reporting when merging data from two sources without omitting any details.

For example, you could do a FULL OUTER JOIN between the Customers and Orders tables. To identify customers without orders, you would search for NULL values in the OrderID column from the Orders table.

```
SELECT Customers.CustomerID, Customers.Name
FROM Customers
FULL OUTER JOIN Orders ON Customers.CustomerID = Orders.CustomerID
WHERE Orders.OrderID IS NULL;
```

The FULL OUTER JOIN retrieves all customers and their orders (if any), and the WHERE clause filters the results to include only those customers for whom there is no matching order (Orders.OrderID is NULL). Knowing what customers are not placing orders can help you backtrack that customer through the website and figure out where they are abandoning the website or offer an incentive to shop, like a 10% off coupon.

TERM TO KNOW

FULL OUTER JOIN

A JOIN that retrieves all records from both tables, even if there is no matching record in the other table; the missing column entries appear as NULL.

2. Examples

Let's come back to our initial dataset with the representative and department tables to help illustrate the OUTER JOINs:

```
CREATE TABLE representative ( representative_id INT PRIMARY KEY, first_name VARCHAR (30) NOT NULL, last_name VARCHAR (30) NOT NULL );
CREATE TABLE department (
department_id INT PRIMARY KEY,
department_name VARCHAR(100) NOT NULL,
manager_id INT,
CONSTRAINT fk_manager FOREIGN KEY (manager_id) REFERENCES representative(representative_id)
);
INSERT INTO representative (representative_id, first_name, last_name)
VALUES (1, 'Bob', 'Evans'), (2, 'Tango', 'Rushmore'), (3, 'Danika', 'Arkane'), (4, 'Mac', 'Anderson');
INSERT INTO department (department_id, department_name, manager_id)
VALUES (1, 'Sales', 1), (2, 'Marketing', 3), (3, 'IT', 4), (4, 'Finance', NULL), (5, 'Support', NULL);
```

HINT

In this lesson, we are working with FULL OUTER JOIN, but there are other types of OUTER JOINs as well, such as LEFT and RIGHT OUTER JOIN. You will learn about them in upcoming lessons.

The FULL OUTER JOIN returns the data from all rows in both the left and right tables. If they match, it will return data from both sides. If they don't match, the columns of the table will be filled with NULL.

Let's first look at the structure of the statement:

```
SELECT <columnlist>
FROM <table1>
FULL OUTER JOIN <table2> ON <table1>.<table1column1> = <table2>.<table2column1>;
```

Notice that this is similar to a regular JOIN ON statement, with only the FULL OUTER added. Using our two tables, we can run the FULL OUTER JOIN like this:

```
SELECT *
FROM representative
FULL OUTER JOIN department ON representative.representative_id = department.manager_id;
```

This will return the three records that match the representative_id from the representative table and the department_id from the department table. It will also return the rows from the department table that do not have a matching row in the representative table (4th and 5th row in the result set). It will also return the rows from the representative table that do not have a match in the department table (6th row):

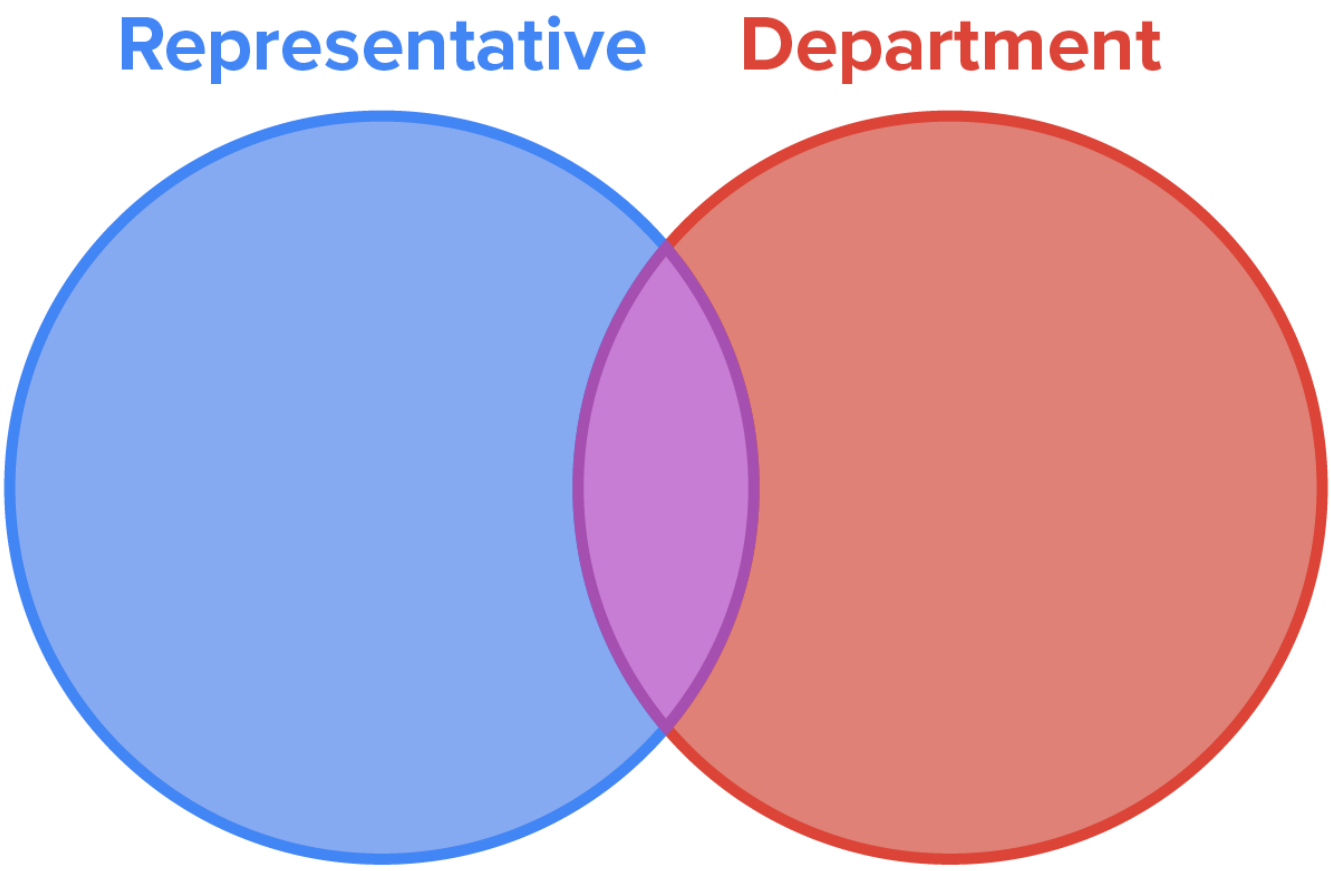
Query Results

Row count: 6

representative_id	first_name	last_name	department_id	department_name	manager_id
1	Bob	Evans	1	Sales	1
3	Danika	Arkane	2	Marketing	3
4	Mac	Anderson	3	IT	4
			4	Finance	
			5	Support	
2	Tango	Rushmore			

It's important to note that the main purpose of an OUTER JOIN is to find the areas where there are no matches. If every row in both tables matched up perfectly, a FULL OUTER JOIN would have the same result as an INNER JOIN.

The following Venn diagram shows what the data would return:



WATCH

TRY IT

Your turn! Open the SQL tool by clicking on the LAUNCH DATABASE button below. Then, enter in 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, you learned how to use a **FULL OUTER JOIN** to retrieve a result set from multiple tables that includes both matched and unmatched records. A FULL OUTER JOIN includes all the records from both tables, even if there are no corresponding records in the other table. This is in contrast to INNER JOINs, which fetch only matching records. Any columns for which there is no matching value appear with a NULL value in the result set. OUTER JOINs are particularly useful for scenarios involving incomplete data sets. In the **example**, you learned the command syntax for using a FULL OUTER JOIN to return data from the tables.

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

TERMS TO KNOW

FULL OUTER JOIN
A JOIN that retrieves all records from both tables, even if there is no matching record in the other table; the missing column entries appear as NULL.