

LEFT JOIN Query Example

This task demonstrates the use of a LEFT JOIN to retrieve data from two tables: 'cities' and 'countries'. The query retrieves all records from the 'cities' table, even if there is no matching record in the 'countries' table. This is particularly useful when analyzing unmatched data.

Below is the instruction and query setup:

Learn / Courses / Joining Data in SQL

< Course Outline >

Light Mode

Exercise

This is a LEFT JOIN, right?

Nice work getting to grips with the structure of joins! In this exercise, you'll explore the differences between `INNER JOIN` and `LEFT JOIN`. This will help you decide which type of join to use.

As before, you will be using the `cities` and `countries` tables.

You'll begin with an `INNER JOIN` with the `cities` table (left) and `countries` table (right). This helps if you are interested only in records where a country is present in both tables.

You'll then change to a `LEFT JOIN`. This helps if you're interested in returning all countries in the `cities` table, whether or not they have a match in the `countries` table.

Instructions 2/2

50 XP

✓

Perform an inner join with `cities AS c1` on the left and `countries AS c2` on the right. Use `code` as the field to merge your tables on.

2

Change the code to perform a `LEFT JOIN` instead of an `INNER JOIN`. After executing this query, have a look at how many records the query result contains.

Take Hint (-15 XP)

query.sql

```
1 SELECT
2   c1.name AS city,
3   code,
4   c2.name AS country,
5   region,
6   city_proper_pop
7 FROM cities AS c1
8 -- Join right table (with alias)
9 ---
10 ON c1.country_code = c2.code
11 ORDER BY code DESC;
```

⏮ Loading... ⏭ Loading...

query result

No query executed yet...

Showing 0 out of 0 rows

-- SQL Query Using LEFT JOIN

```
SELECT
  c1.name AS city,
```

```
code,  
c2.name AS country,  
region,  
city_proper_pop  
FROM cities AS c1  
-- Join right table (with alias)  
LEFT JOIN countries AS c2  
ON c1.country_code = c2.code  
ORDER BY code DESC;
```

Explanation: This query uses a LEFT JOIN to include all rows from the 'cities' table ('c1'), even if there are no matching rows in the 'countries' table ('c2'). The SELECT statement retrieves:

1. 'c1.name' (aliased as 'city'): The name of the city.
 2. 'code': The country code.
 3. 'c2.name' (aliased as 'country'): The name of the country (null if unmatched).
 4. 'region': The region of the country (null if unmatched).
 5. 'city_proper_pop': The proper population of the city.
- The results are sorted by the 'code' column in descending order.