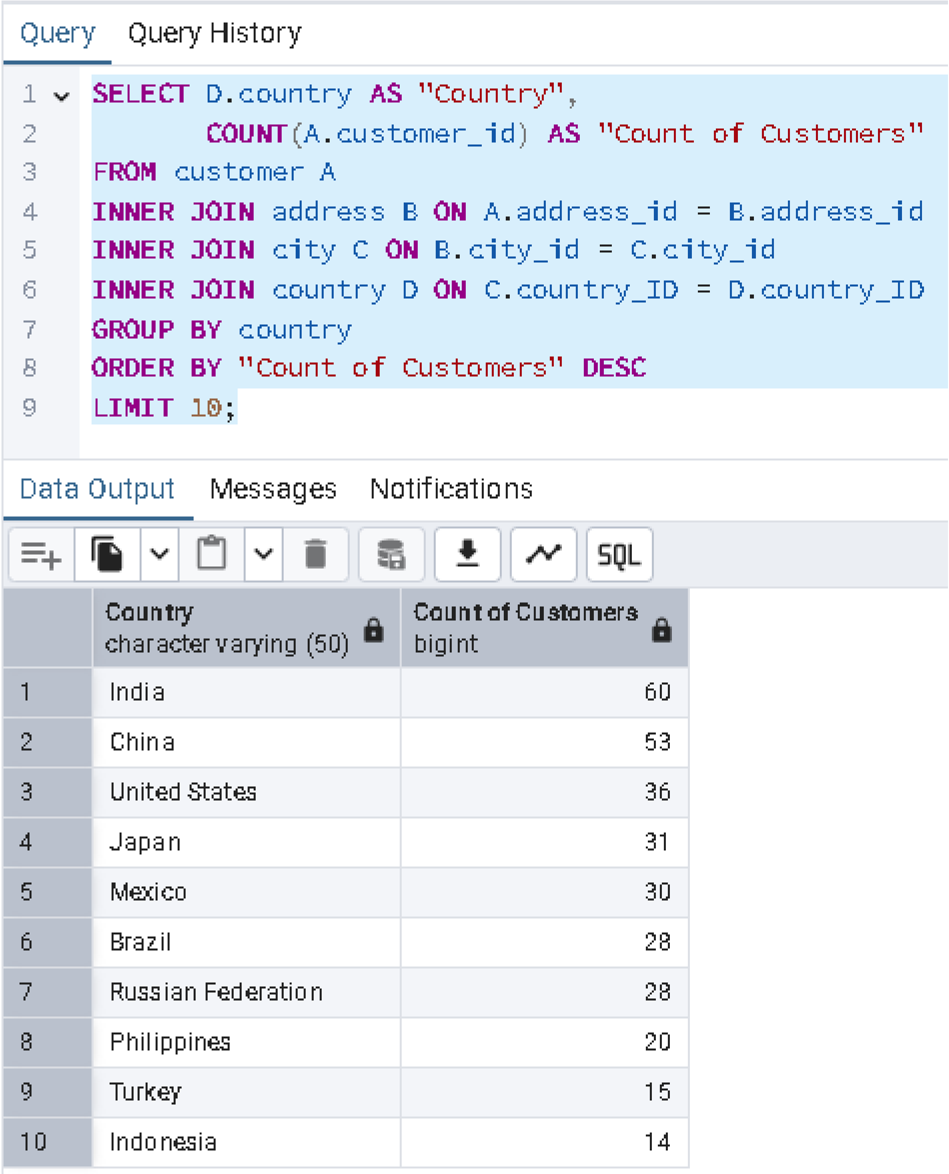
Exercise 3.7 Joining Tables of Data

1. A query to find the top 10 countries for Rockbuster in terms of customer numbers using GROUP BY and ORDER BY :

QUERY :

```
SELECT D.country AS "Country",
      COUNT(A.customer_id) AS "Count of Customers"
FROM customer A
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_ID = D.country_ID
GROUP BY country
ORDER BY "Count of Customers" DESC
LIMIT 10;
```



```sql
SELECT D.country AS "Country",
       COUNT(A.customer_id) AS "Count of Customers"
FROM customer A
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_ID = D.country_ID
GROUP BY country
ORDER BY "Count of Customers" DESC
LIMIT 10;
```

	Country character varying (50)	Count of Customers bigint
1	India	60
2	China	53
3	United States	36
4	Japan	31
5	Mexico	30
6	Brazil	28
7	Russian Federation	28
8	Philippines	20
9	Turkey	15
10	Indonesia	14

Detailed explanation of my approach :

1. **Analyzing the ERD** :
   I began by studying the Entity-Relationship Diagram (ERD) for Rockbuster, which helped me understand the relationships between tables and the keys available to connect them. I identified that the required flow to link the

customer table to the country table was :
customer > address > city > country.

2. **Selecting Columns** :
   I determined which columns were necessary for the analysis. To minimize the resources used (time and cost), I selected only the columns required to achieve the objective. These included customer identifier from the customer table, country (names) from the country table.

3. **Formulating the Query** :
   To get the count of customers per country, I grouped the data by country. The GROUP BY country clause ensured that the customer count was aggregated for each country.

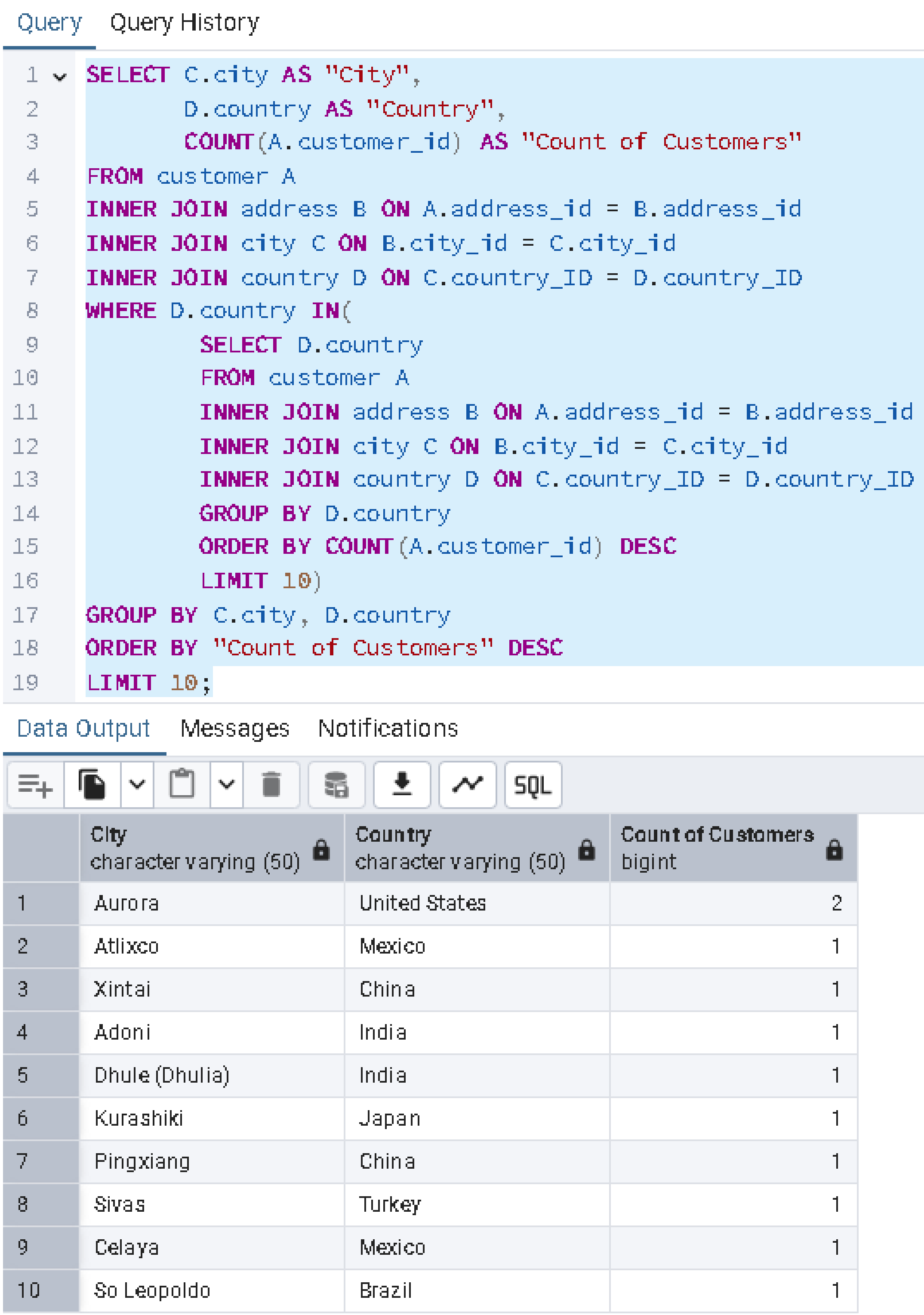
4. **Sorting and Limiting** :
   Since the goal was to find the top 10 countries, I added an ORDER BY clause to sort the results in descending order of customer count. Finally, I used LIMIT 10 to retrieve only the top 10 countries with the highest customer count.

5. **Enhancing Readability** :
   To make the output easier to interpret, I used the AS keyword to assign clear, meaningful aliases to the output columns.

2. A query to identify the top 10 cities that fall within the top 10 countries we identified in step 1.

QUERY :

```
SELECT C.city AS "City",
       D.country AS "Country",
       COUNT(A.customer_id) AS "Count of Customers"
FROM customer A
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_ID = D.country_ID
WHERE D.country IN(
        SELECT D.country
        FROM customer A
        INNER JOIN address B ON A.address_id = B.address_id
        INNER JOIN city C ON B.city_id = C.city_id
        INNER JOIN country D ON C.country_ID = D.country_ID
        GROUP BY D.country
        ORDER BY COUNT(A.customer_id) DESC
        LIMIT 10)
GROUP BY C.city, D.country
ORDER BY "Count of Customers" DESC
LIMIT 10;
```



```sql
SELECT C.city AS "City",
       D.country AS "Country",
       COUNT(A.customer_id) AS "Count of Customers"
FROM customer A
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_ID = D.country_ID
WHERE D.country IN(
        SELECT D.country
        FROM customer A
        INNER JOIN address B ON A.address_id = B.address_id
        INNER JOIN city C ON B.city_id = C.city_id
        INNER JOIN country D ON C.country_ID = D.country_ID
        GROUP BY D.country
        ORDER BY COUNT(A.customer_id) DESC
        LIMIT 10)
GROUP BY C.city, D.country
ORDER BY "Count of Customers" DESC
LIMIT 10;
```

	City character varying (50)	Country character varying (50)	Count of Customers bigint
1	Aurora	United States	2
2	Atlixco	Mexico	1
3	Xintai	China	1
4	Adoni	India	1
5	Dhule (Dhulia)	India	1
6	Kurashiki	Japan	1
7	Pingxiang	China	1
8	Sivas	Turkey	1
9	Celaya	Mexico	1
10	So Leopoldo	Brazil	1

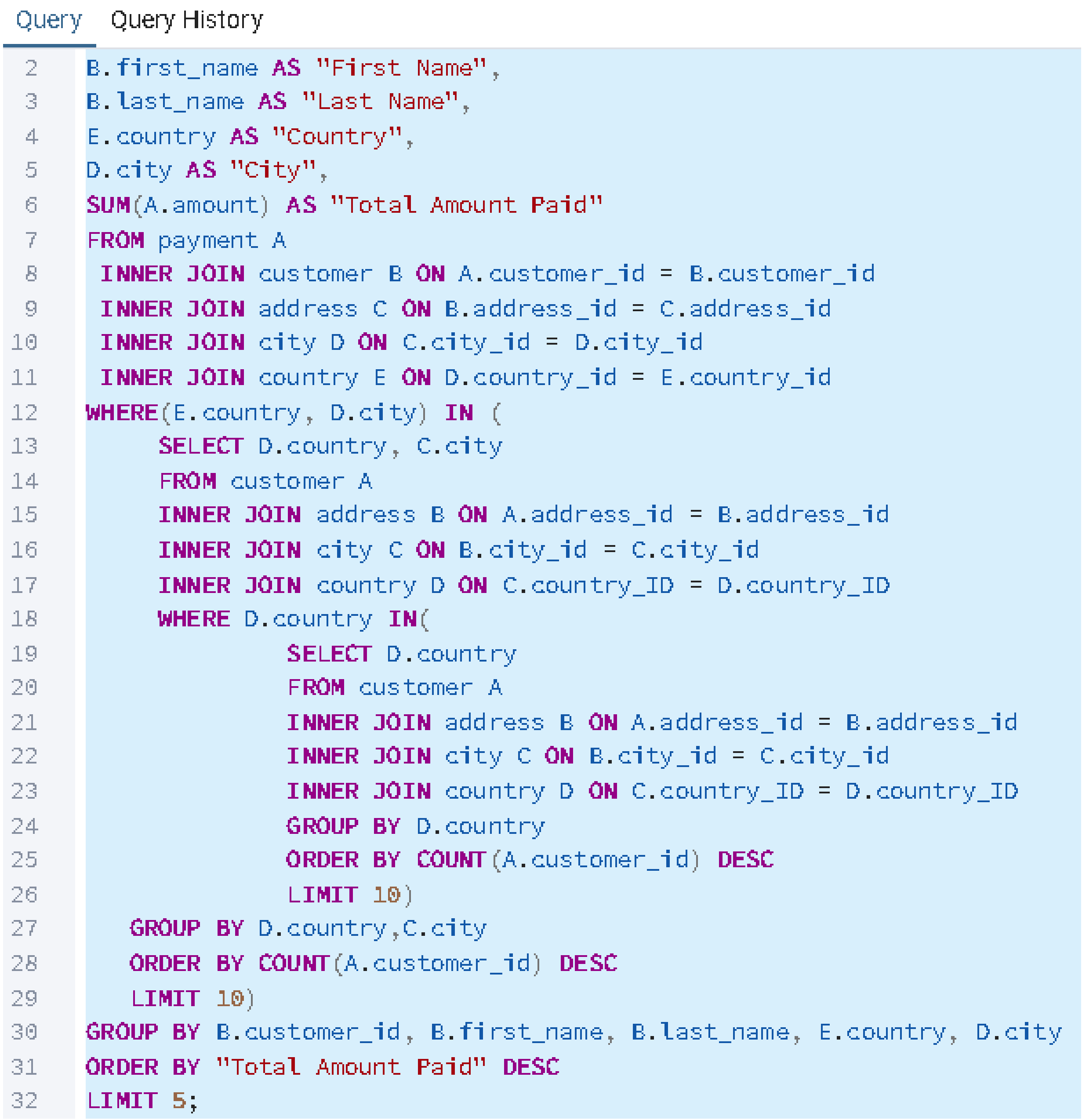
Detailed explanation of my approach :

1. **Analyzing the Requirement** : I first understood that my goal was to find the top 10 cities based on customer count, restricted to the top 10 countries identified previously. This meant narrowing down the scope to cities within these countries while ensuring the ranking was by customer count.
2. **Planning the Query** : I began by breaking the task into smaller steps
   - First, I reused the subquery from my earlier work to identify the top 10 countries.
   - Next, I designed the main query to retrieve city-level data filtered by these top 10 countries.
   - I ensured the query included meaningful columns like city names, country names and customer counts, using aliases to make the output intuitive.
3. **Efficient Use of Joins** : I carefully joined the relevant tables (customer, address, city and country) based on the relationships in the ERD.
4. **Crafting the Subquery** : The subquery (was created to retrieve the top 10 countries) was used in the WHERE IN clause of the main query.
5. **Writing the Main Query** : I grouped data in the main query by city and country, counted the number of customers for each city and sorted the results by customer count in descending order. To get only the top 10 cities, I applied LIMIT 10.

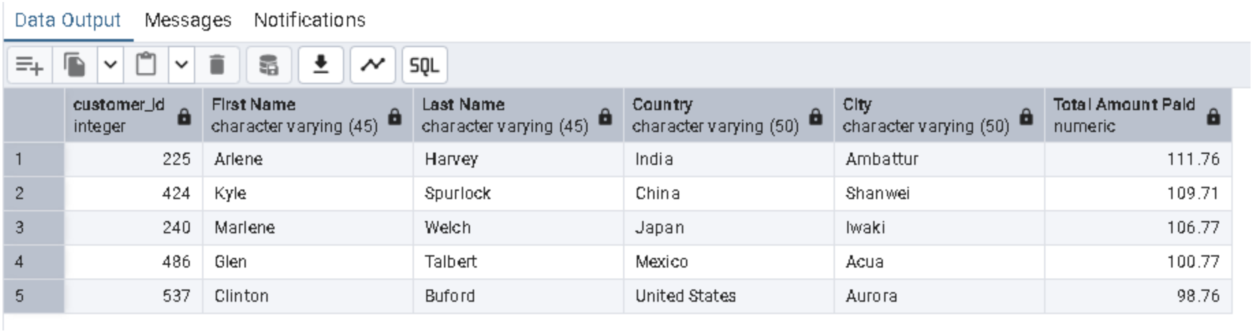
3. A query to find the top 5 customers from the top 10 cities who’ve paid the highest total amounts to Rockbuster. The customer team would like to reward them for their loyalty (we have to use the WHERE clause with an operator, followed by GROUP BY and ORDER BY).

QUERY :

```
SELECT B.customer_id,
B.first_name AS "First Name",
B.last_name AS "Last Name",
E.country AS "Country",
D.city AS "City",
SUM(A.amount) AS "Total Amount Paid"
FROM payment A
 INNER JOIN customer B ON A.customer_id = B.customer_id
 INNER JOIN address C ON B.address_id = C.address_id
 INNER JOIN city D ON C.city_id = D.city_id
 INNER JOIN country E ON D.country_id = E.country_id
WHERE(E.country, D.city) IN (
    SELECT D.country, C.city
    FROM customer A
    INNER JOIN address B ON A.address_id = B.address_id
    INNER JOIN city C ON B.city_id = C.city_id
    INNER JOIN country D ON C.country_ID = D.country_ID
    WHERE D.country IN(
        SELECT D.country
        FROM customer A
        INNER JOIN address B ON A.address_id = B.address_id
        INNER JOIN city C ON B.city_id = C.city_id
        INNER JOIN country D ON C.country_ID = D.country_ID
        GROUP BY D.country
        ORDER BY COUNT(A.customer_id) DESC
        LIMIT 10)
  GROUP BY D.country,C.city
  ORDER BY COUNT(A.customer_id) DESC
  LIMIT 10)
GROUP BY B.customer_id, B.first_name, B.last_name, E.country, D.city
ORDER BY "Total Amount Paid" DESC
LIMIT 5;
```



```
B.first_name AS "First Name",
B.last_name AS "Last Name",
E.country AS "Country",
D.city AS "City",
SUM(A.amount) AS "Total Amount Paid"
FROM payment A
 INNER JOIN customer B ON A.customer_id = B.customer_id
 INNER JOIN address C ON B.address_id = C.address_id
 INNER JOIN city D ON C.city_id = D.city_id
 INNER JOIN country E ON D.country_id = E.country_id
WHERE(E.country, D.city) IN (
     SELECT D.country, C.city
     FROM customer A
     INNER JOIN address B ON A.address_id = B.address_id
     INNER JOIN city C ON B.city_id = C.city_id
     INNER JOIN country D ON C.country_ID = D.country_ID
     WHERE D.country IN(
              SELECT D.country
              FROM customer A
              INNER JOIN address B ON A.address_id = B.address_id
              INNER JOIN city C ON B.city_id = C.city_id
              INNER JOIN country D ON C.country_ID = D.country_ID
              GROUP BY D.country
              ORDER BY COUNT(A.customer_id) DESC
              LIMIT 10)
   GROUP BY D.country,C.city
   ORDER BY COUNT(A.customer_id) DESC
   LIMIT 10)
GROUP BY B.customer_id, B.first_name, B.last_name, E.country, D.city
ORDER BY "Total Amount Paid" DESC
LIMIT 5;
```



	customer_id integer	First Name character varying (45)	Last Name character varying (45)	Country character varying (50)	City character varying (50)	Total Amount Paid numeric
1	225	Arlene	Harvey	India	Ambattur	111.76
2	424	Kyle	Spurlock	China	Shanwei	109.71
3	240	Marlene	Welch	Japan	Iwaki	106.77
4	486	Glen	Talbert	Mexico	Acua	100.77
5	537	Clinton	Buford	United States	Aurora	98.76

Detailed explanation of my approach :

1. I began by studying the **relationships** between **the Tables** :

- **payment table** contains the payment amounts (amount) and is linked to customer table via customer_id.
- **customer table** provides customer details such as first_name, last_name and address_id.
- **address table** links customers to specific cities using city_id.
- **city table** provides city names and is linked to countries through country_id.
- **country table** provides country names.

By understanding this flow, I determined which tables to join and how to connect them effectively to get the necessary data.

2. I needed to find the **top 10 cities** with the highest number of customers within the **top 10 countries** (I already had this query from Question2)
   <u>**Identify** the top 10 countries</u>

- I grouped by country in the innermost subquery.
- Used COUNT(customer_id) to calculate the number of customers for each country.
- Ordered the results in descending order of customer count and limited the results to 10 using LIMIT 10.

<u>**Find the top 10 cities in those countries**</u>

- I created another subquery to group by both city and country.
- Filtered to include only cities from the top 10 countries (using the results of the previous subquery).
- Ordered the cities by the number of customers in descending order and limited the results to the top 10.

This two-step nested filtering ensures that the cities selected are relevant to the top-performing countries.

3. Aggregate Payments for Customers in the Selected Cities

- Once I had the top 10 cities and countries, I aggregated payment data (SUM(amount)) for customers in these cities.
- Grouped by customer details (customer_id, first_name, last_name) as well as city and country.
- Ordered the results by the total amount paid in descending order to find the top spenders.

4. In the main query

- I joined the payment, customer, address, city, and country tables to gather all necessary information.
- I filtered the results to include only cities and countries from the subquery (using WHERE (city, country) IN (...)).
- Grouped the data by the required attributes to calculate the total amount paid by each customer.
- Ordered the results by the total amount paid in descending order and limited the output to the top 5 customers.