**TASK 3.8**

**Step 1:**

**Find the average amount paid by the top 5 customers.**

**select avg (Highest\_total\_amount) from**

**(select c.customer\_id,concat(c.first\_name,' ',c.last\_name) as customer\_name,ct.country,ci.city,sum(p.amount) as Highest\_total\_amount**

**from payment p**

**join customer c**

**on p.customer\_id = c.customer\_id**

**join address a**

**on c.address\_id = a.address\_id**

**join city ci**

**on a.city\_id = ci.city\_id**

**join country ct**

**on ci.country\_id = ct.country\_id**

**where ci.city in**

**(select ci.city from customer c**

**join address a**

**on c.address\_id = a.address\_id**

**join city ci**

**on a.city\_id = ci.city\_id**

**join country ct**

**on ci.country\_id = ct.country\_id**

**where ct.country in**

**(select ct.country from customer c**

**join address a**

**on c.address\_id = a.address\_id**

**join city ci**

**on a.city\_id = ci.city\_id**

**join country ct**

**on ci.country\_id = ct.country\_id**

**group by ct.country**

**order by count(c.customer\_id) desc**

**limit 10)**

**group by ct.country,ci.city**

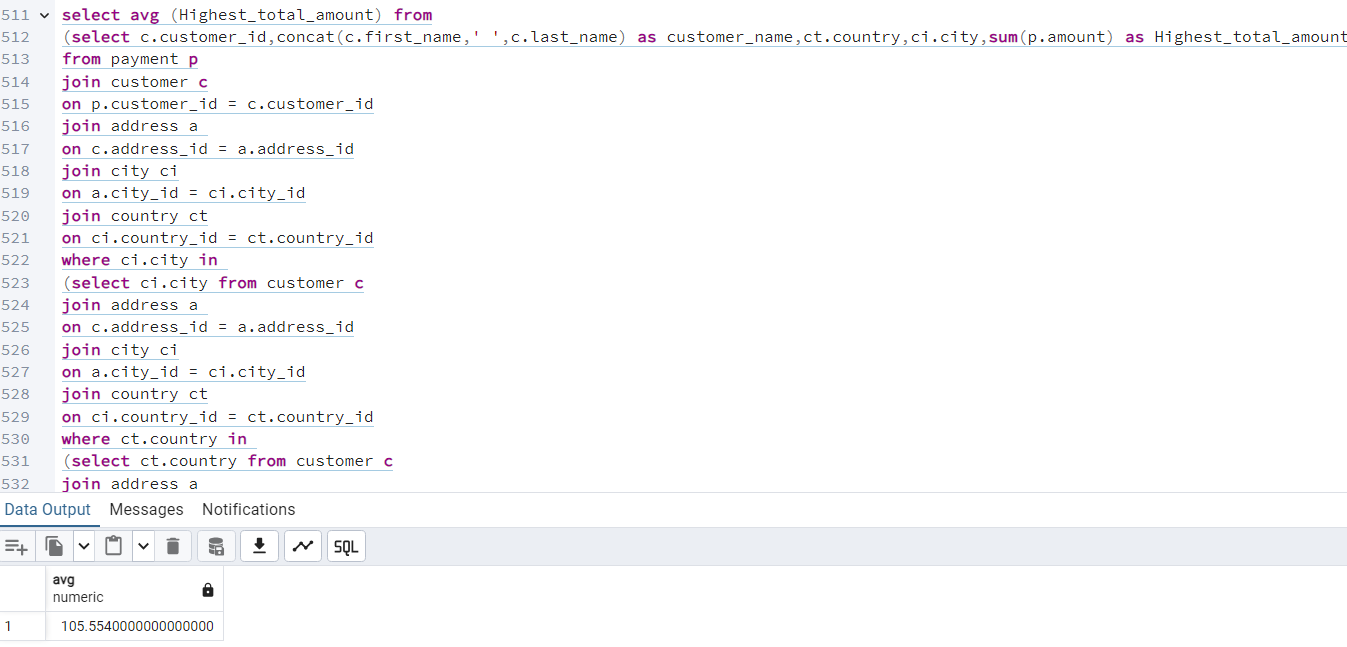
**order by count(c.customer\_id) desc**

**limit 10)**

**GROUP BY c.customer\_id,ct.country, ci.city**

**ORDER BY Highest\_total\_amount DESC**

**LIMIT 5) as total\_amount\_paid;**



**Step 2:**

**Find out how many of the top 5 customers you identified in step 1 are based within each country.**

**select ct.country,count(distinct c.customer\_id) as all\_customer\_count,count(distinct top\_5\_customers.customer\_id) as top\_customer\_count**

**from customer c**

**INNER JOIN address a ON a.address\_id = c.address\_id**

**INNER JOIN city ci ON ci.city\_id = a.city\_id**

**INNER JOIN country ct ON ct.country\_id = ci.country\_id**

**left join**

**(select c.customer\_id,concat(c.first\_name,' ',c.last\_name) as customer\_name,ct.country,ci.city,sum(p.amount) as Highest\_total\_amount**

**from payment p**

**join customer c**

**on p.customer\_id = c.customer\_id**

**join address a**

**on c.address\_id = a.address\_id**

**join city ci**

**on a.city\_id = ci.city\_id**

**join country ct**

**on ci.country\_id = ct.country\_id**

**where ci.city in**

**(select ci.city from customer c**

**join address a**

**on c.address\_id = a.address\_id**

**join city ci**

**on a.city\_id = ci.city\_id**

**join country ct**

**on ci.country\_id = ct.country\_id**

**where ct.country in**

**(select ct.country from customer c**

**join address a**

**on c.address\_id = a.address\_id**

**join city ci**

**on a.city\_id = ci.city\_id**

**join country ct**

**on ci.country\_id = ct.country\_id**

**group by ct.country**

**order by count(c.customer\_id) desc**

**limit 10)**

**group by ct.country,ci.city**

**order by count(c.customer\_id) desc**

**limit 10)**

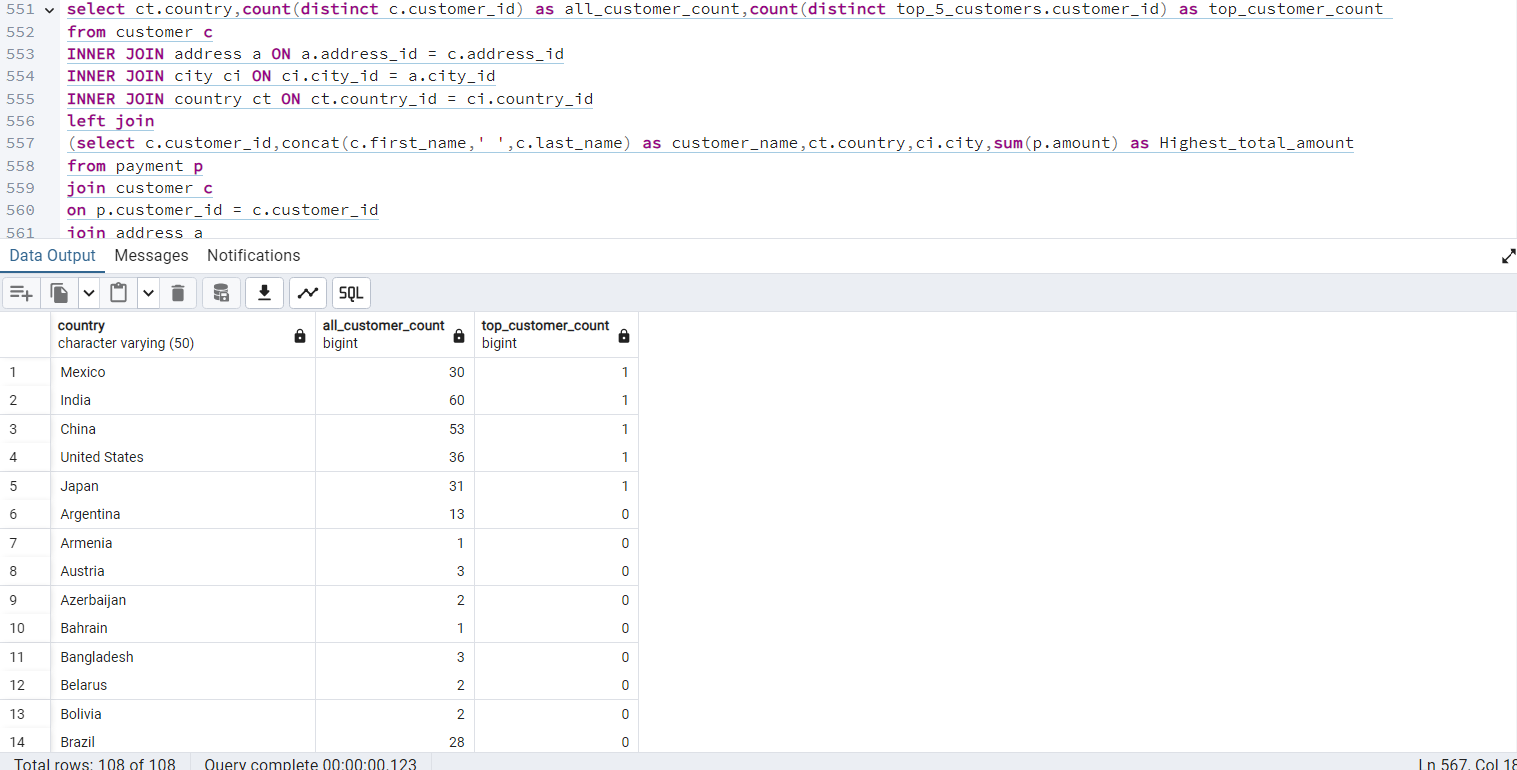
**group by c.customer\_id,ct.country, ci.city**

**order by Highest\_total\_amount desc**

**limit 5)as top\_5\_customers on top\_5\_customers.country = ct.country**

**group by ct.country**

**order by top\_customer\_count desc;**

****

**Step 3:**

Do you think steps 1 and 2 could be done without using subqueries?

When do you think subqueries are useful?

**Even though my experience with PostgreSQL is limited at the moment, I think the subqueries I used to complete steps 1 and 2 are quite long and complicated, making them difficult to read and interpret for other users. Therefore, I guess more expert analysts could find more efficient and simpler ways to produce the same results, maybe by using a CTE. Another way would be to run a query whose results build upon a separate query, but this procedure is inefficient if the data is constantly changing. Subqueries are still useful in various scenarios where we need to perform complex queries or retrieve data based on conditions that involve data from multiple tables, or furthermore where the data stored in the database is always changing and we need to get the latest and most up-to-date information.**