TASK 3.9

Step 1

**QUERY 1:**

with Top10\_country(Country)

as

(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),

Top10\_City

as

(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 \* from Top10\_country )),

Total\_amount

as

(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 \* from Top10\_city)

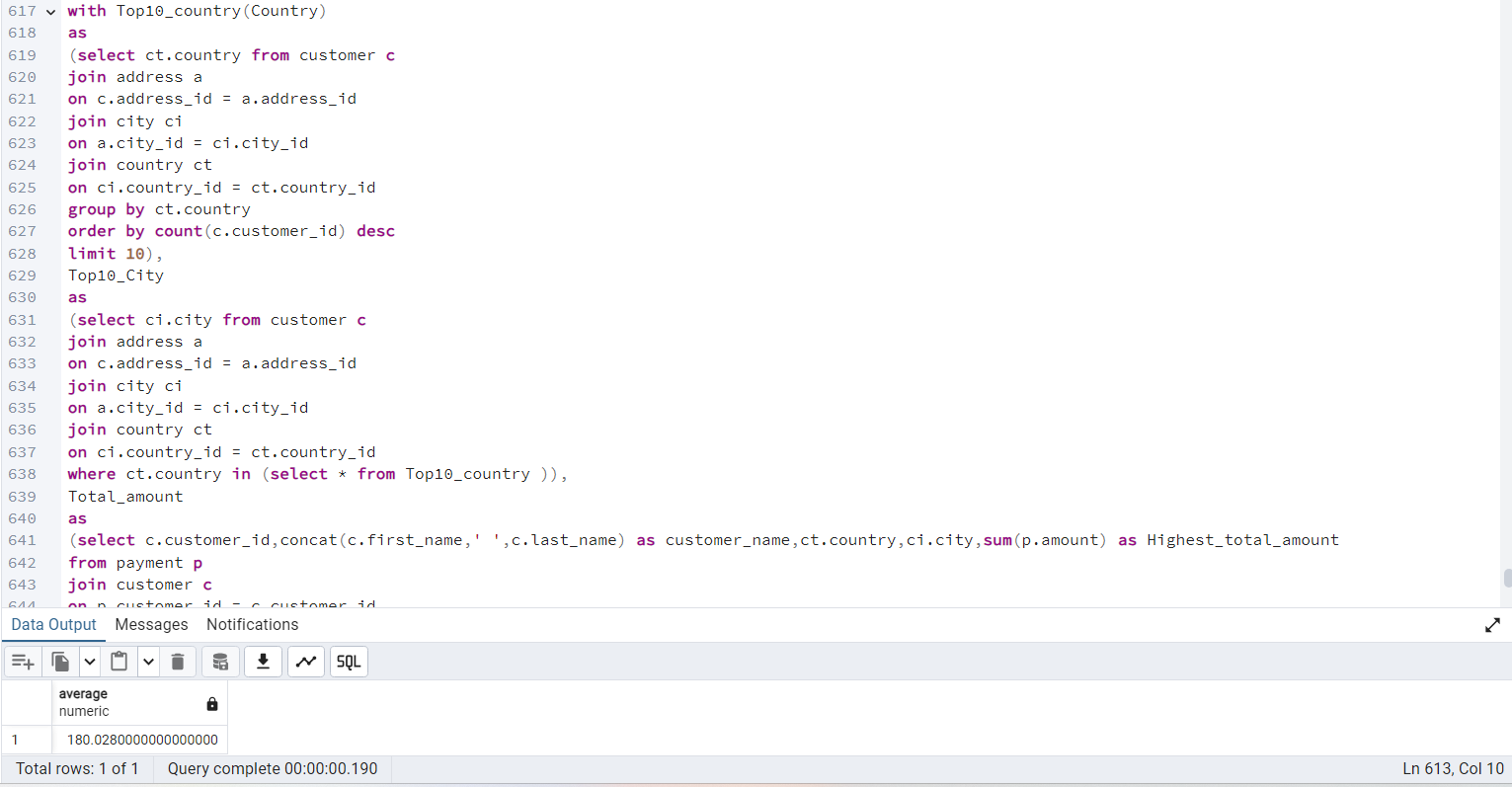
GROUP BY c.customer\_id,ct.country, ci.city

ORDER BY Highest\_total\_amount DESC

LIMIT 5)

select avg (Highest\_total\_amount) as Average

from Total\_amount;



**QUERY 2:**

with Top10\_country(Country)

as

(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),

Top10\_City

as

(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 \* from Top10\_country )

group by ct.country, ci.city

order by count(c.customer\_id) desc

Limit 10),

total\_amount\_paid as (

select c.customer\_id, ct.country, sum(P.amount) as total\_amount\_payment

from payment p

inner join customer as c on C.customer\_id = P.customer\_id

inner join address as a on A.address\_id = C.address\_id

inner join city as ci on CI.city\_id = A.city\_id

inner join country as Ct on Ct.country\_id = CI.country\_id

where CI.city in (select city from top10\_city)

group by C.customer\_id, Ct.country

order by sum(p.amount) desc

limit 5)

select ct.country,

count(distinct C.customer\_id) as all\_customer\_count,

count(distinct T5.customer\_id) as top\_customer\_count

from customer as c

inner join address as A on A.address\_id = C.address\_id

inner join city as ci on ci.city\_id = a.city\_id

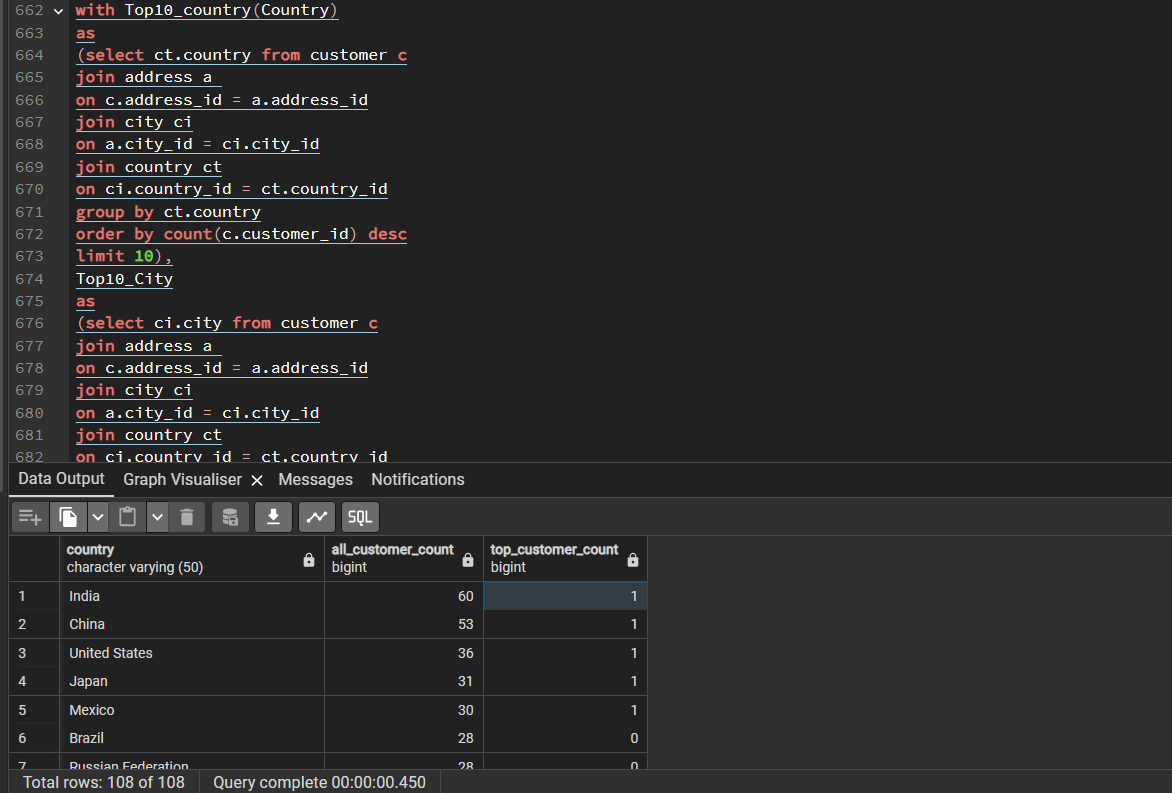
inner join country as ct on ct.country\_id = ci.country\_id

left join total\_amount\_paid as T5 on T5.customer\_id = C.customer\_id and T5.country =

ct.country

group by Ct.country

order by all\_customer\_count desc



Step 2

I think the CTE queries will perform better than the subqueries. A CTE only has to be defined once, while with a subquery you have to define it every time referring to it and this is why a subquery also often is longer with more lines.

**Comparing costs:**

Query 1 task 3.8:"Aggregate (cost=167.66..167.67 rows=1 width=32)" 84msec

Query 2 task 3.8: "Sort (cost=271.93..272.20 rows=109 width=25)" 80msec

Query 1 task 3.9: Aggregate (cost=729.14..729.15 rows=1 width=32) 92msec

Query 2 task 3.9: Sort (cost=271.80..272.08 rows=109 width=25) 105msec

The results surprised me a bit, but after reading the comparisons between CTE’s and subqueries again I understand more that one does not always perform better than the other. And that it is better to always check this.

Step 3

I got easily confused with these queries as they are more complicated than the examples in the exercise before the task. When I start transforming the queries I make mistakes because I am still learning how to use the right format. But once I made the queries correct and read them back, I do understand the logic and I do understand that CTE’s and subqueries are very similar and that it is a matter of having the same details. The difference is with the CTE query I create the CTE immediately, after this comes the inner statement that is the same as the subquery and then the outer statement, which is adapted to link with the CTE.