ANSWERS 3.9

In this task, you'll convert your subqueries from task 3.8 into CTEs to make your code easier to read.

Directions

Create a new text document and call it "Answers 3.9." You'll save your queries, outputs, and written answers in this document.

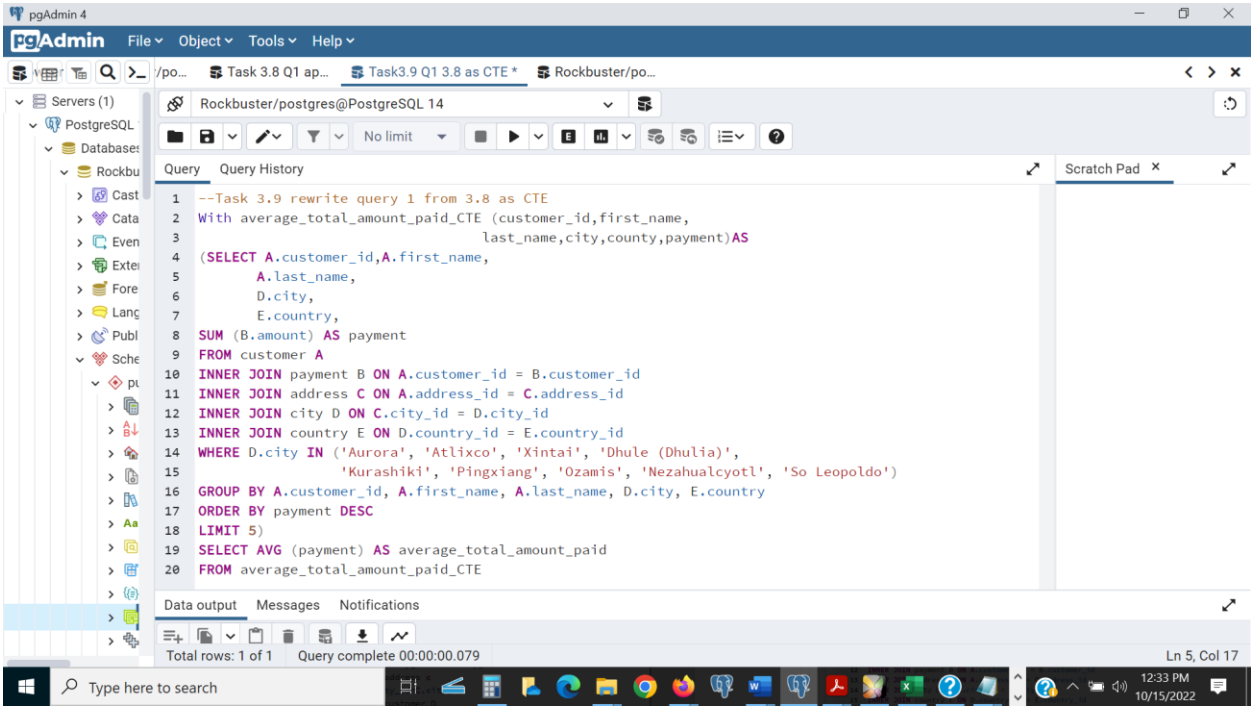
Step 1: Answer the business questions from step 1 and 2 of task 3.8 using CTEs

Business questions step 1 of task 3.8 using CTE's

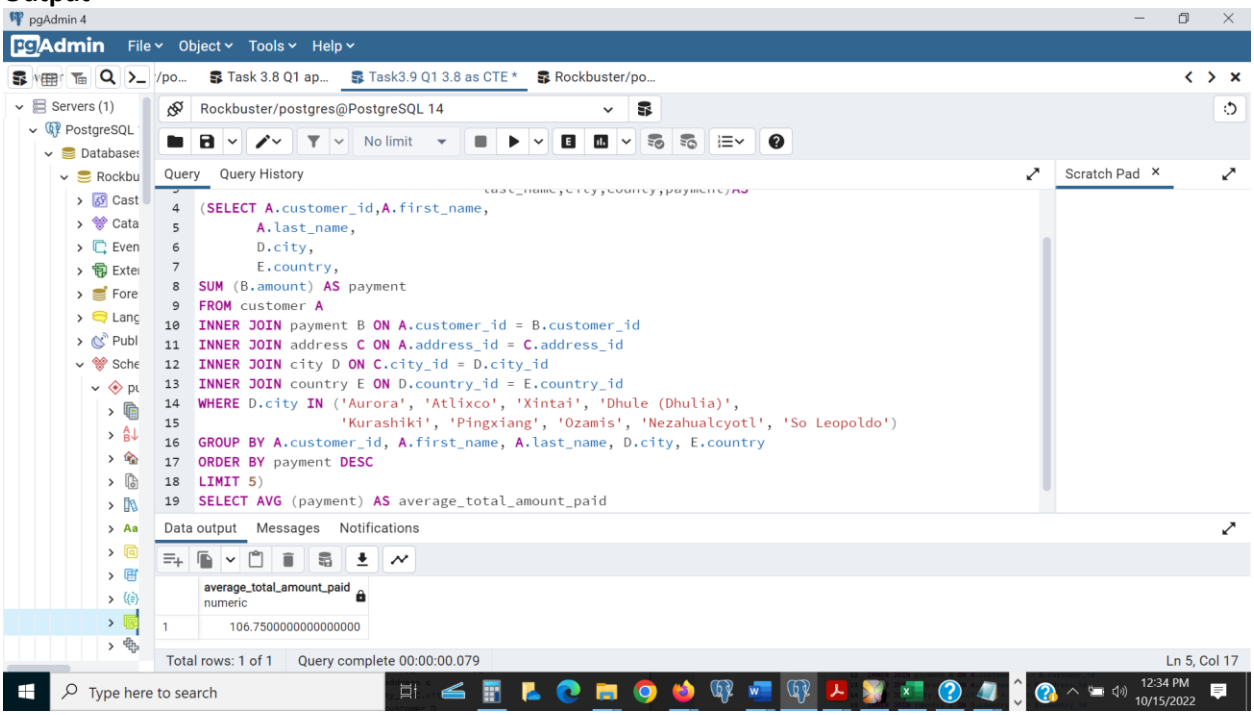
1. Rewrite your queries from steps 1 and 2 of task 3.8 as CTEs.
2. Copy-paste your CTEs and their outputs into your answers document.
3. Write 2 to 3 sentences explaining how you approached this step, for example, what you did first, second, and so on.

QUERY-1 REWRITTEN AS CTE - "Find the average amount paid by the top 5 customers".

QUERY



Output

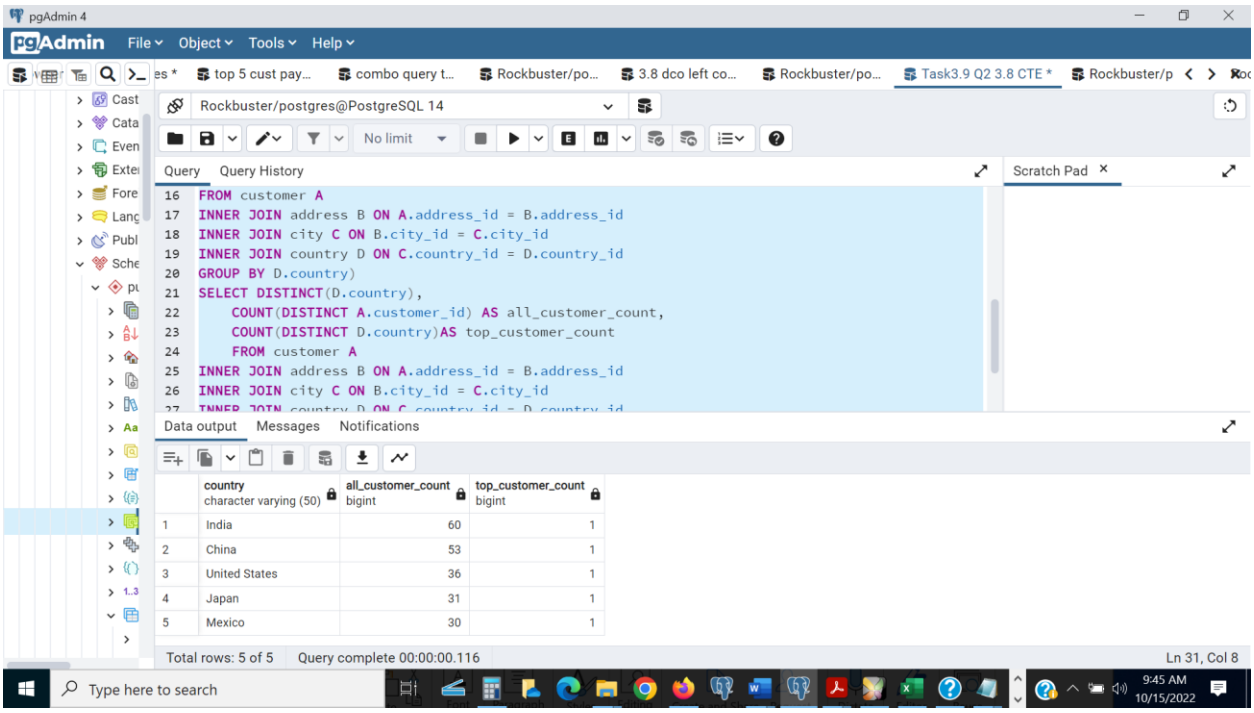


QUERY2 REWRITTEN AS CTE "**Find out how many of the top 5 customers are based within each country".**

1. Used query2 from task 3.8 rewrote at CTE

```
With top_5_customers_CTE (amount,customer_id,first_name,last_name,city,country,payment)AS
(SELECT A.amount,B.customer_id,B.first_name,B.last_name,
                D.city,E.country,
SUM(A.amount) AS payment
FROM payment A
INNER JOIN customer B ON B.customer_id = A.customer_id
INNER JOIN address C ON C.address_id = B.address_id
INNER JOIN city D ON D.city_id = C.city_id
INNER JOIN country E ON E.country_id = D.country_id
GROUP BY B.customer_id, B.first_name, B.last_name, D.city, E.country, A.amount
ORDER BY payment DESC
LIMIT 5),
customer_count_CTE AS (SELECT DISTINCT (D.country),
   COUNT(DISTINCT A.customer_id) AS all_customer_count,
          COUNT(DISTINCT D.country) AS top_customer_count
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)
SELECT DISTINCT(D.country),
```

```
        COUNT(DISTINCT A.customer_id) AS all_customer_count,
        COUNT(DISTINCT D.country)AS top_customer_count
        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
LEFT JOIN top_5_customers_CTE ON D.country = top_5_customers_CTE.country
GROUP BY D.country
ORDER BY all_customer_count DESC
LIMIT 5
```

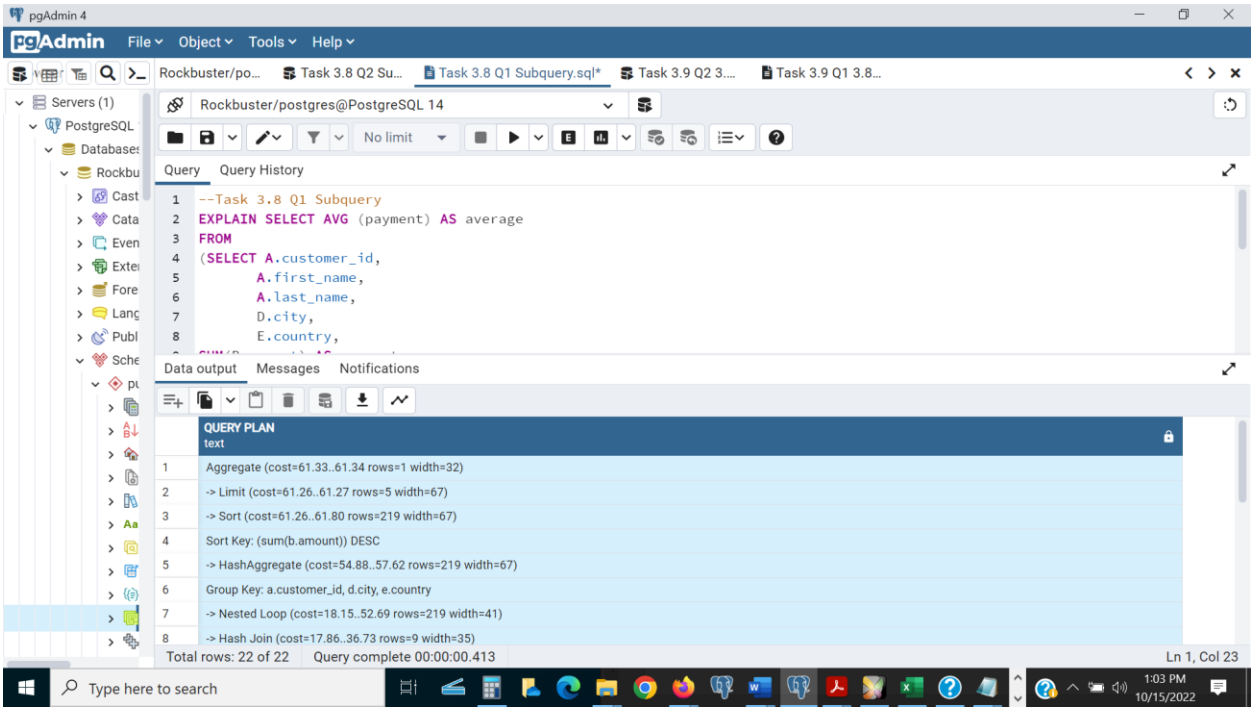


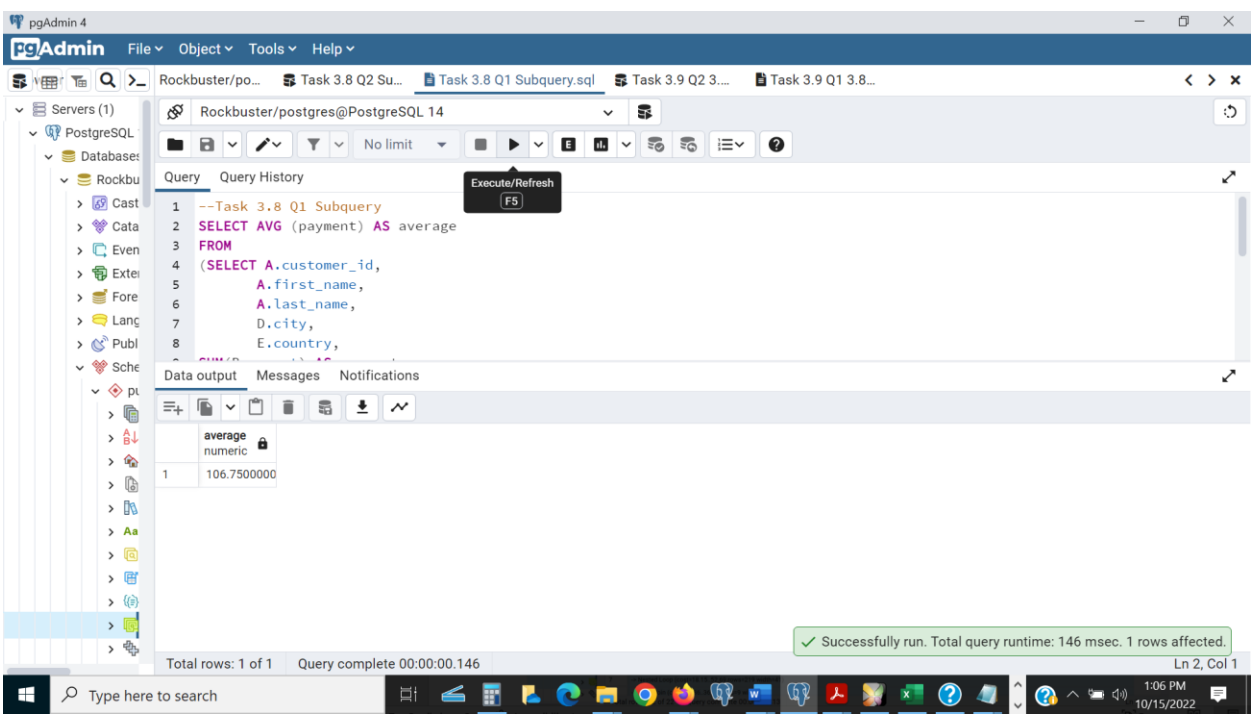
Step 2: Compare the performance of your CTEs and subqueries.

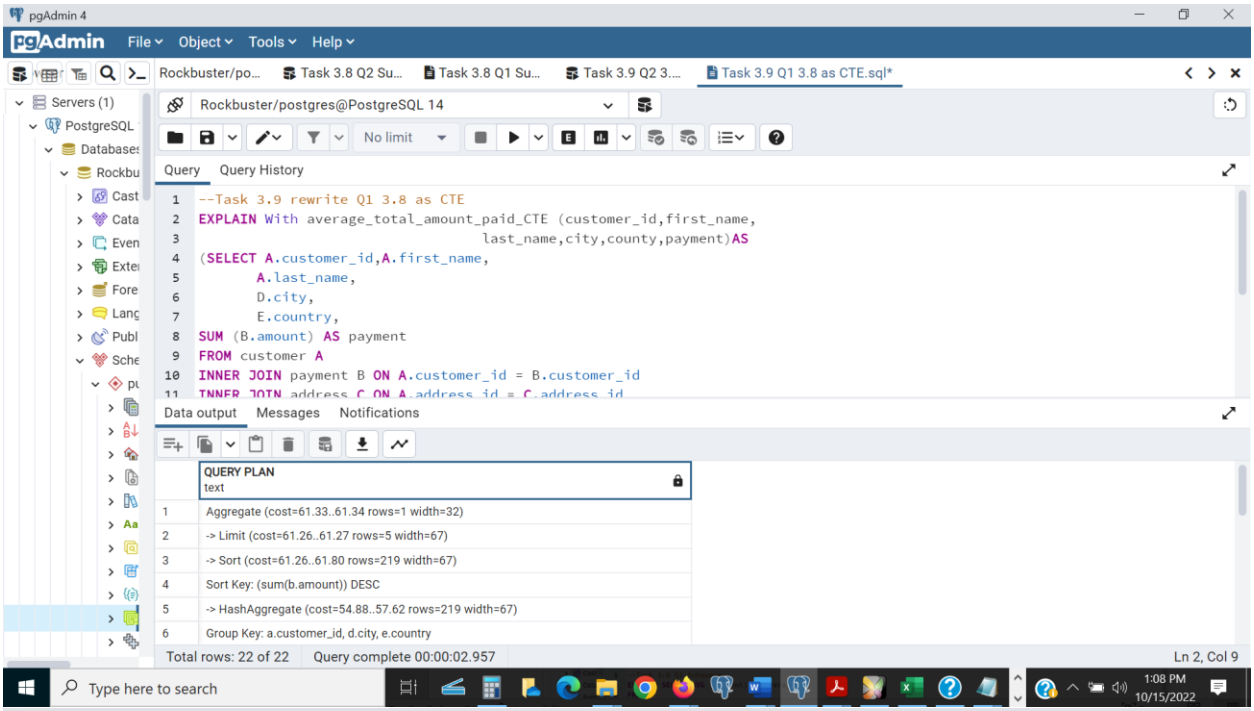
1. Which approach do you think will perform better and why? I would think that since the subquery has less lines of code it may run faster than the CTE.
2. Compare the costs of all the queries by creating query plans for each one. See below table
3. The EXPLAIN command gives you an *estimated* cost. To find out the actual speed of your queries, run them in pgAdmin 4. After each query has been run, a pop-up window will display its speed in milliseconds.
4. Did the results surprise you? Write a few sentences to explain your answer. Results were surprisingly the same for costs but run times would change each time I would rerun the query, must have to do with accessing the data and possibly my system.

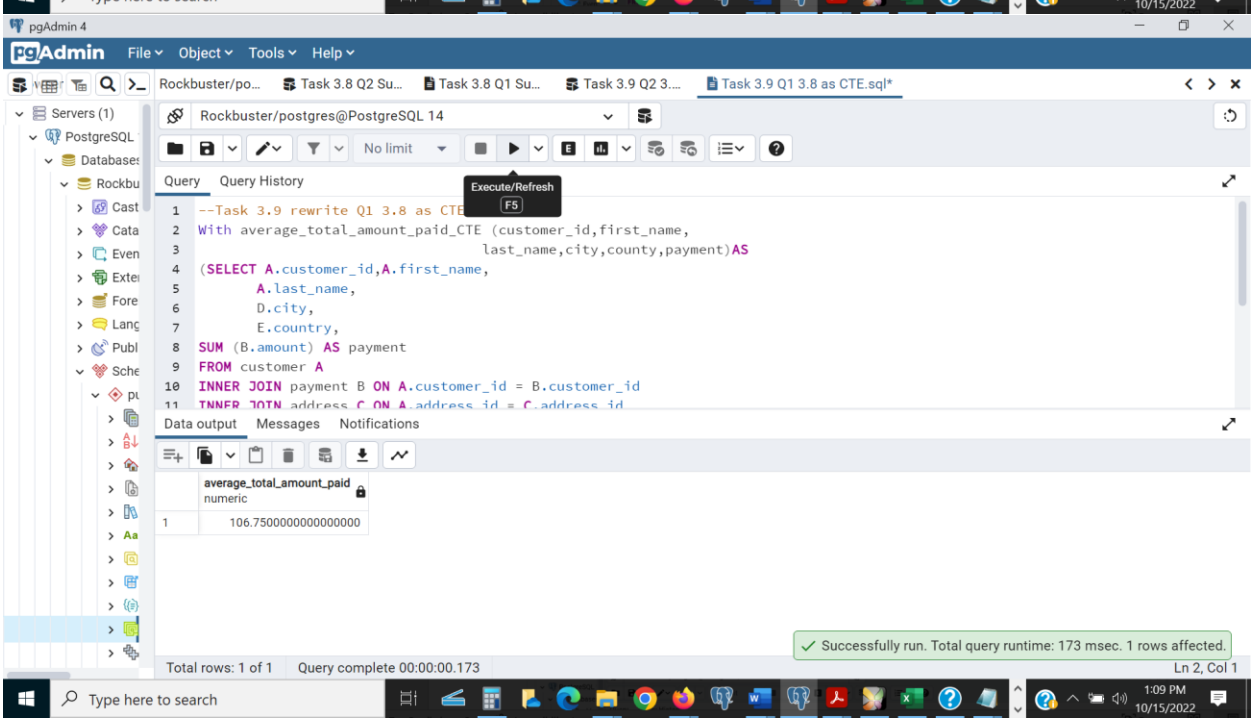
QUERY-1 3.8 Subquery / REWRITTEN AS CTE

	Subquery	CTE
Cost	"Aggregate (cost=61.33..61.34 rows=1 width=32)"	"Aggregate (cost=61.33..61.34 rows=1 width=32)"
Time	Total query runtime: 146 msec. 1 rows affected	Total query runtime: 173 msec. 1 rows affected





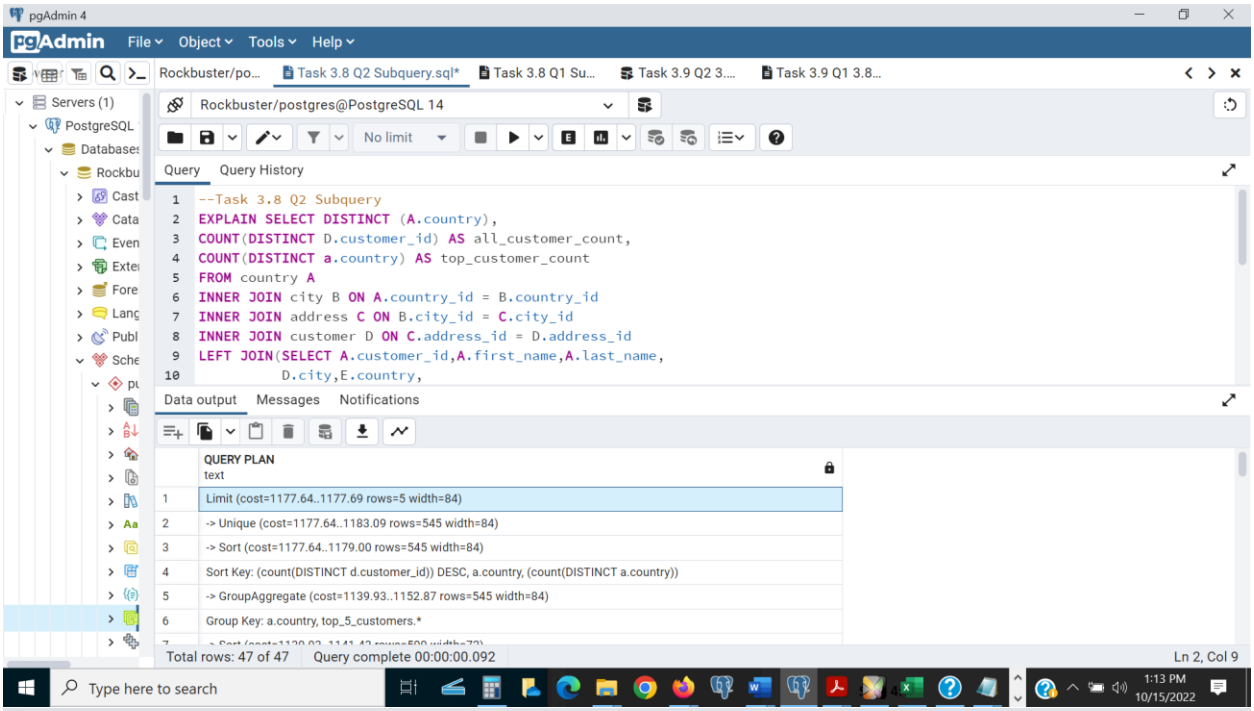


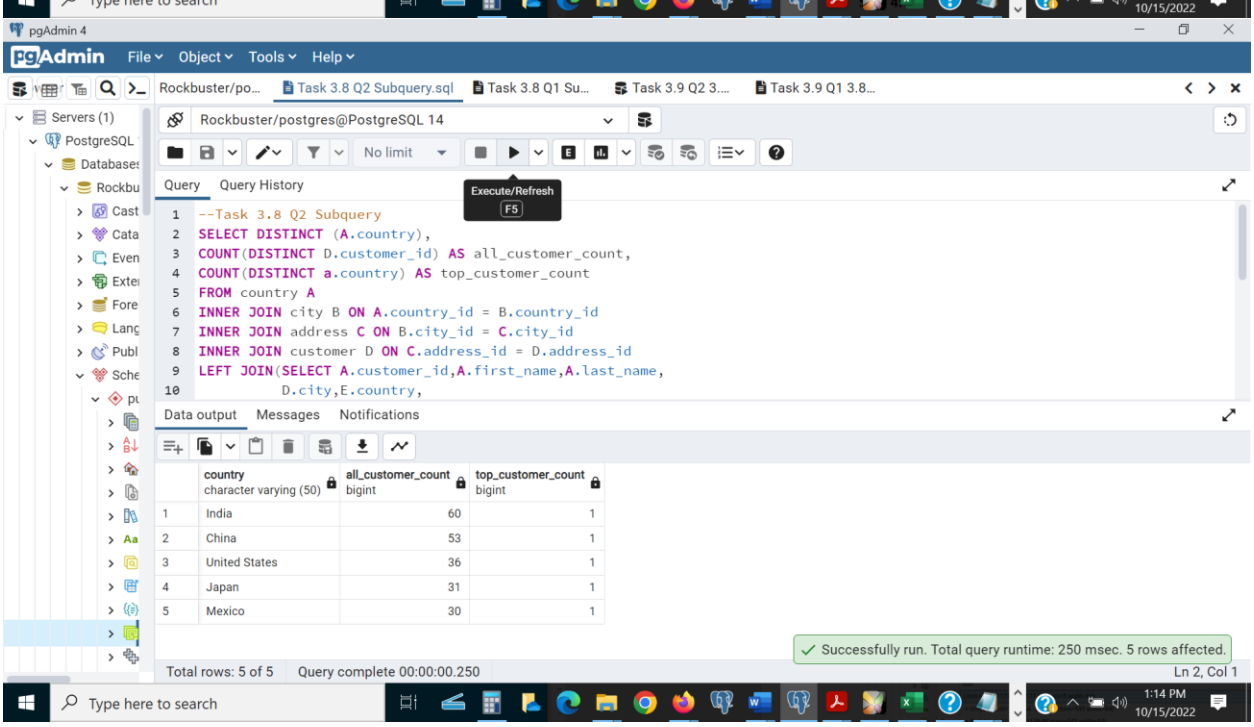


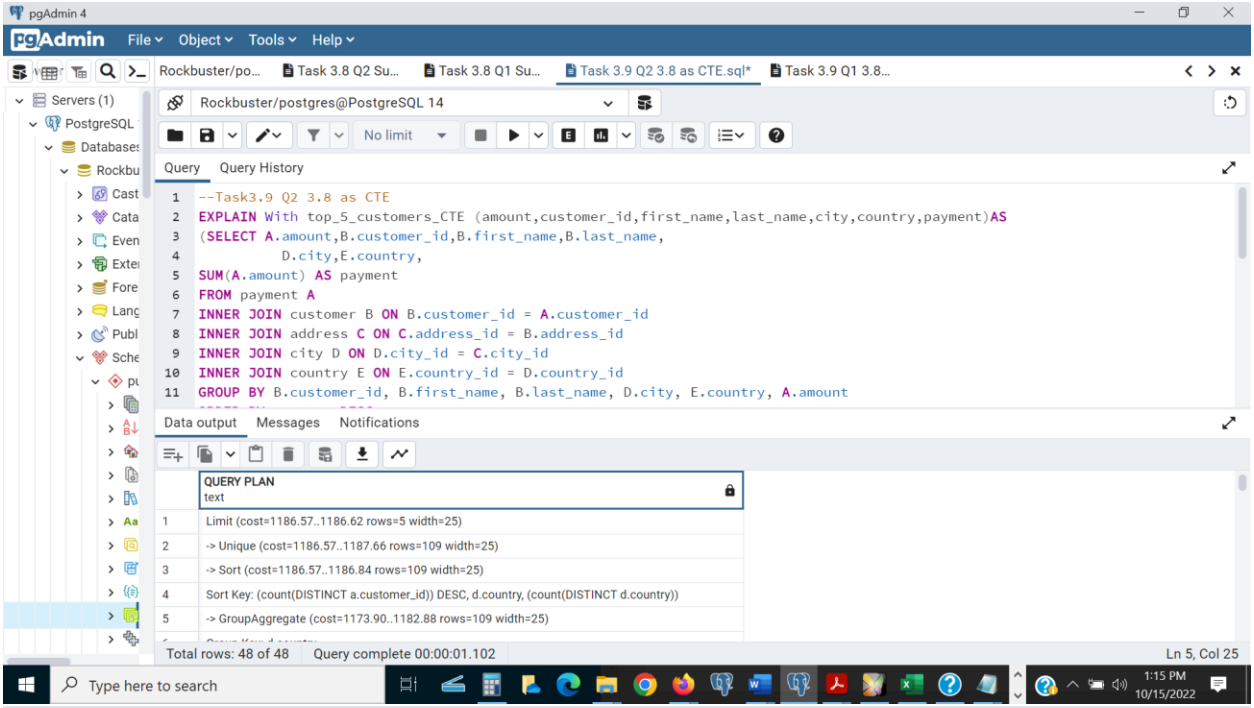
QUERY-2 3.8 Subquery / REWRITTEN AS CTE

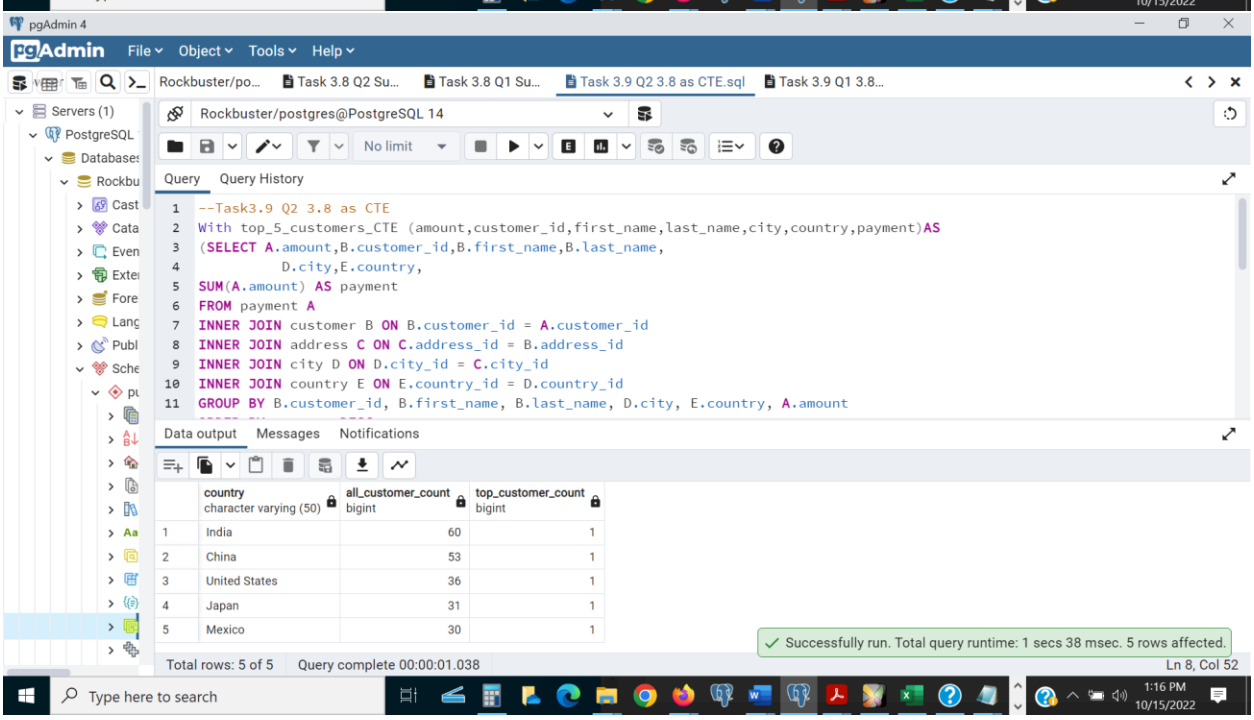
	Subquery	CTE

Cost	"Limit (cost=1177.64..1177.69 rows=5 width=84)"	"Limit (cost=1186.57..1186.62 rows=5 width=25)"
Time	Total query runtime 250 msec. 5 rows affected.	Total query runtime 1 secs. 38 msec. 5 rows affected.









Step 3:

Write 1 to 2 paragraphs on the challenges you faced when replacing your subqueries with CTEs.

With creation of the subqueries the process was relatively simple with writing inner join queries given the examples, and then understanding the table relationships. In rewriting the CTE's it was more difficult with the second task with having to create two CTE's to join the second inner query, was not as straight.

Step 4:

Save your “Answers 3.9” document as a PDF and upload it here for your tutor to review.