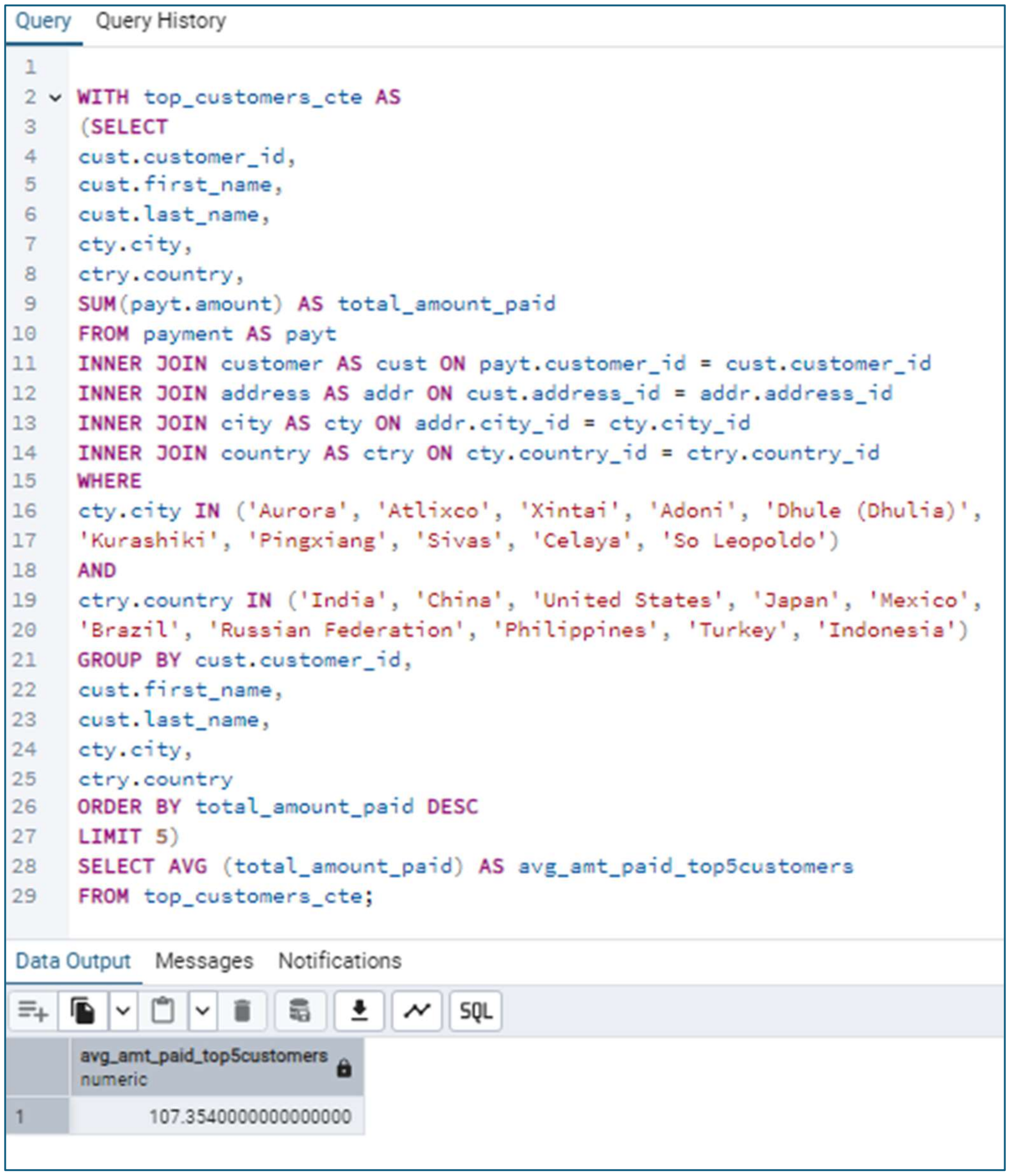
A3 - 3.9

Step 1.1 and 1.2 - Answer the business questions from steps 1 and 2 of task 3.8 using CTEs

Query 1



Query Query History

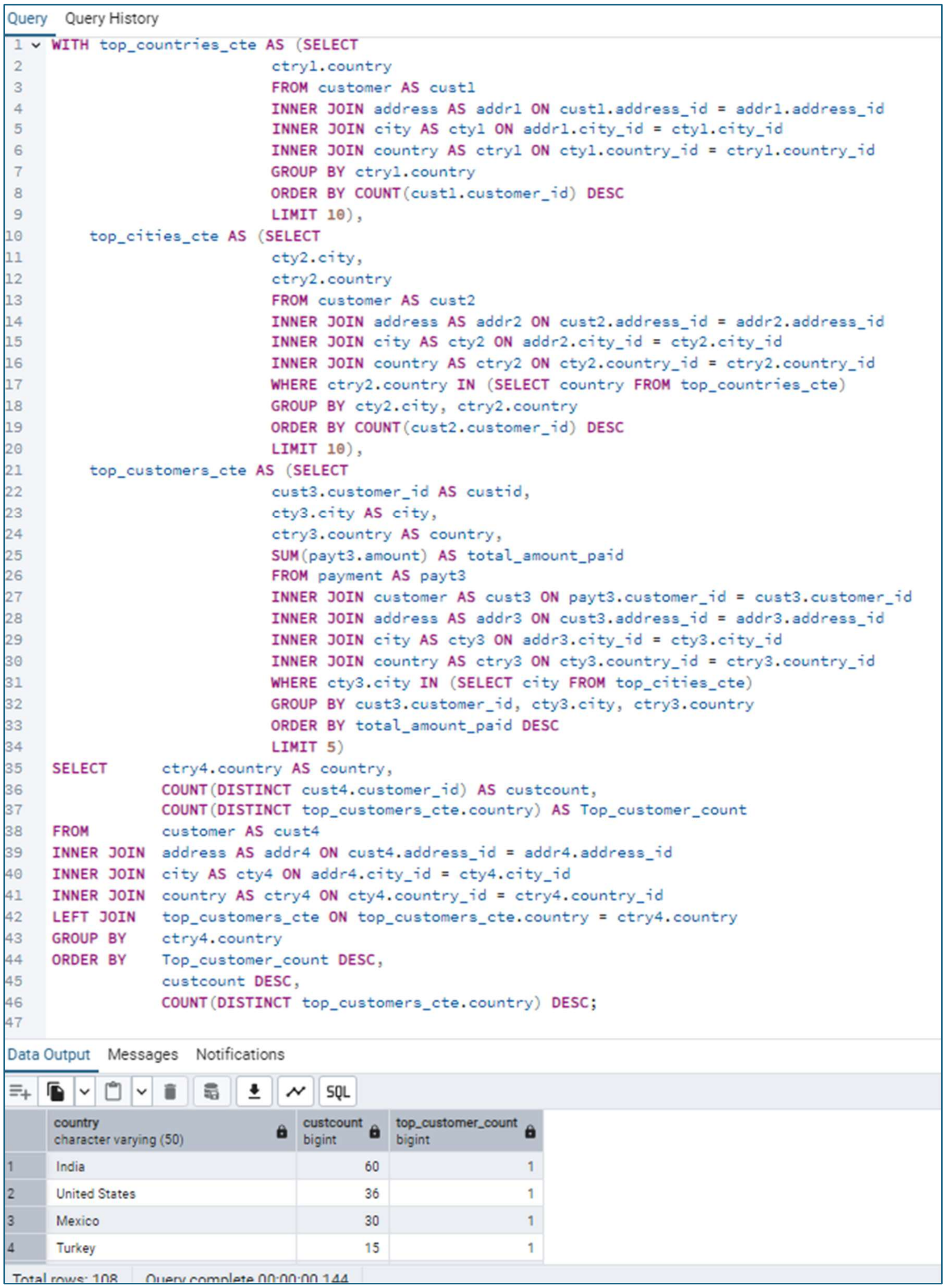
```sql

WITH top_customers_cte AS
(SELECT
cust.customer_id,
cust.first_name,
cust.last_name,
cty.city,
ctry.country,
SUM(payt.amount) AS total_amount_paid
FROM payment AS payt
INNER JOIN customer AS cust ON payt.customer_id = cust.customer_id
INNER JOIN address AS addr ON cust.address_id = addr.address_id
INNER JOIN city AS cty ON addr.city_id = cty.city_id
INNER JOIN country AS ctry ON cty.country_id = ctry.country_id
WHERE
cty.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)',
'Kurashiki', 'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
AND
ctry.country IN ('India', 'China', 'United States', 'Japan', 'Mexico',
'Brazil', 'Russian Federation', 'Philippines', 'Turkey', 'Indonesia')
GROUP BY cust.customer_id,
cust.first_name,
cust.last_name,
cty.city,
ctry.country
ORDER BY total_amount_paid DESC
LIMIT 5)
SELECT AVG (total_amount_paid) AS avg_amt_paid_top5customers
FROM top_customers_cte;
```

Data Output Messages Notifications

	avg_amt_paid_top5customers numeric
1	107.3540000000000000

Query 2



Query Query History

```sql
WITH top_countries_cte AS (SELECT
                        ctry1.country
                        FROM customer AS cust1
                        INNER JOIN address AS addr1 ON cust1.address_id = addr1.address_id
                        INNER JOIN city AS cty1 ON addr1.city_id = cty1.city_id
                        INNER JOIN country AS ctry1 ON cty1.country_id = ctry1.country_id
                        GROUP BY ctry1.country
                        ORDER BY COUNT(cust1.customer_id) DESC
                        LIMIT 10),
    top_cities_cte AS (SELECT
                        cty2.city,
                        ctry2.country
                        FROM customer AS cust2
                        INNER JOIN address AS addr2 ON cust2.address_id = addr2.address_id
                        INNER JOIN city AS cty2 ON addr2.city_id = cty2.city_id
                        INNER JOIN country AS ctry2 ON cty2.country_id = ctry2.country_id
                        WHERE ctry2.country IN (SELECT country FROM top_countries_cte)
                        GROUP BY cty2.city, ctry2.country
                        ORDER BY COUNT(cust2.customer_id) DESC
                        LIMIT 10),
    top_customers_cte AS (SELECT
                        cust3.customer_id AS custid,
                        cty3.city AS city,
                        ctry3.country AS country,
                        SUM(payt3.amount) AS total_amount_paid
                        FROM payment AS payt3
                        INNER JOIN customer AS cust3 ON payt3.customer_id = cust3.customer_id
                        INNER JOIN address AS addr3 ON cust3.address_id = addr3.address_id
                        INNER JOIN city AS cty3 ON addr3.city_id = cty3.city_id
                        INNER JOIN country AS ctry3 ON cty3.country_id = ctry3.country_id
                        WHERE cty3.city IN (SELECT city FROM top_cities_cte)
                        GROUP BY cust3.customer_id, cty3.city, ctry3.country
                        ORDER BY total_amount_paid DESC
                        LIMIT 5)
SELECT      ctry4.country AS country,
            COUNT(DISTINCT cust4.customer_id) AS custcount,
            COUNT(DISTINCT top_customers_cte.country) AS Top_customer_count
FROM        customer AS cust4
INNER JOIN  address AS addr4 ON cust4.address_id = addr4.address_id
INNER JOIN  city AS cty4 ON addr4.city_id = cty4.city_id
INNER JOIN  country AS ctry4 ON cty4.country_id = ctry4.country_id
LEFT JOIN   top_customers_cte ON top_customers_cte.country = ctry4.country
GROUP BY    ctry4.country
ORDER BY    Top_customer_count DESC,
            custcount DESC,
            COUNT(DISTINCT top_customers_cte.country) DESC;
```

Data Output Messages Notifications

	country character varying (50)	custcount bigint	top_customer_count bigint
1	India	60	1
2	United States	36	1
3	Mexico	30	1
4	Turkey	15	1

Total rows: 108 Query complete 00:00:00.144

<u>Step 1.3 - Write 2 to 3 sentences explaining how you approached this step, for example, what you did first, second, and so on.</u>

Converting subqueries into CTEs was relatively easier as the main components of the query were already written.

I converted three subqueries into CTEs (1) Top countries (2) Top cities and (3) Top customer counts for top cities using WITH cte_name as (subquery syntax). What I found the most useful was to create all CTE at the top in more structured manner and then writing a simple Select statement to retrieve info from the relevant CTE.

<u>Step 2: Compare the performance of your CTEs and subqueries.</u>

<u>Step 2.1</u> - Which approach do you think will perform better and why?

I believe that CTE approach performs better compared to Subqueries due to simplicity in writing CTE. It is like more like a standalone query or creating a view so that it can easily be referred back to in the query instead of creating subquery which is interwoven in the main query and reading it can be a challenge.

<u>Step 2.2 & 2.3</u> - Compare the costs of all the queries by creating query plans for each one. The EXPLAIN command gives you an *estimated* cost. To find out the actual speed of your queries, run them in pgAdmin 4. After you've run each query, a popup window will display its speed in milliseconds.

Performance Analysis

	Time	Cost
Query 1 – Subquery	71	25.44
Query 1 – CTE	71	25.44
Query 2 – Subquery	109	270
Query 2 – CTE	175	270

These queries are relatively simple compared to complex syntaxes powering the real time data updates in many organizations. Costs are the same for both CTEs and Subqueries. Time factor is also not much different given their complexity and amt of data being processed.

<u>Step 2.4</u> - Did the results surprise you? Write a few sentences to explain your answer.

Results from Explain statement comparing both CTE and Subqueries were not very surprising as (1) these are simple queries (2) Data volume being processed is very small (3) We are retrieving a very small amount of data back.

Step 3:

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

Replacing subqueries with CTEs was relatively easy to understand and perform. I would have struggled more if the ask was the other way around.