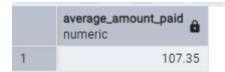
Step 1: Answer the business questions from steps 1 and 2 of task 3.8 using CTEs

WITH total_amount_paid_cte (customer_id,full_name,country,city,total_payment)

```
AS
      (SELECT
      c.customer id.
      CONCAT(c.first_name, ',c.last_name) AS full_name,
      co.country AS country,
      ci.city AS city,
      SUM(p.amount) AS total_payment
      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 co ON co.country_id = ci.country_id
      INNER JOIN payment AS p ON p.customer_id = c.customer_id
      WHERE ci.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki',
'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
      GROUP BY 1,2,3,4--,5,6
      ORDER BY total_payment DESC
      LIMIT 5)
```

SELECT

ROUND(AVG(total_payment), 2) AS average_amount_paid FROM total_amount_paid_cte



WITH TopCustomers_cte AS (

```
c.customer_id,
CONCAT(c.first_name, ' ', c.last_name) AS full_name,
co.country AS country,
ci.city AS city,
SUM(p.amount) AS total_payment
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 co ON co.country_id = ci.country_id
  INNER JOIN payment AS p ON p.customer_id = c.customer_id
  WHERE ci.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)',
             'Kurashiki', 'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
  GROUP BY c.customer_id, co.country, ci.city
  ORDER BY total_payment DESC
  LIMIT 5
).
TopCustomersCount_cte AS (
  SELECT
     abc.country AS abccountry,
     COUNT(DISTINCT abc.customer_id) AS top_customer_count
  FROM TopCustomers_cte AS abc
  GROUP BY abc.country
)
SELECT
  co.country AS country,
  COUNT(DISTINCT c.customer_id) AS all_customer_count,
  COALESCE(MAX(abcd.top_customer_count), 0) 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 co ON co.country_id = ci.country_id
LEFT JOIN TopCustomersCount_cte AS abcd ON abcd.abccountry = co.country
GROUP BY co.country
```

ORDER BY all customer count DESC

	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	India	60	1
2	China	53	0
3	United States	36	1
4	Japan	31	0
5	Mexico	30	2
6	Brazil	28	0
7	Russian Federation	28	0
8	Philippines	20	0
9	Turkey	15	1

Write 2 to 3 sentences explaining how you approached this step, for example, what you did first, second, and so on.

To approach this step, I first reviewed the queries from steps 1 and 2 and identified the subqueries that could be replaced with Common Table Expressions (CTEs). I then rewrote each query using CTEs to improve readability and structure, ensuring that the logic and results remained consistent. Finally, I tested the CTE-based queries to confirm their correctness and copied the outputs for inclusion in the answers document.

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

Which approach do you think will perform better and why?

CTEs are generally preferable due to their readability, reusability, and structured approach. They are particularly effective in scenarios where intermediate results need to be referenced multiple times. Subqueries, on the other hand, might perform slightly better in simpler cases where modularity and reusability are not as critical.

In this scenario, the performance difference is negligible because of the dataset size and query structure. However, for better maintainability and scalability, the CTE-based approach is more suitable.

Compare the costs of all the queries by creating query plans for each one.

o CTE



Subquery



Did the results surprise you? Write a few sentences to explain your answer

Yes, the results did surprise me. I expected the CTE approach to perform significantly better due to its structured design and reusability. However, the execution times for both queries were almost identical, showing that the database engine optimizes both CTEs and subqueries efficiently for this dataset size and complexity. This highlights that the performance difference is often minimal in simpler scenarios, even though I still find CTEs more beneficial for readability and maintainability.

Step 3:

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

When replacing subqueries with CTEs, the main challenge was ensuring that the rewritten queries produced the same results. Subqueries are directly embedded in the main query, so breaking them into separate CTEs required careful attention to the logic and how each part connected. It took extra effort to test the changes and confirm everything worked correctly.

Another difficulty was understanding how CTEs are processed compared to subqueries. While CTEs made the queries easier to read and manage, I had to test them to ensure they performed well. Overall, it was a bit tricky at first, but it helped me better understand how to structure queries.