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

1. Rewrite your queries from steps 1 and 2 of task 3.8 as CTEs.

Query Query History

```
1  WITH average_amount_cte (customer_id, last_name, city, country) AS
2  (SELECT
3          A. customer_id,
4          A. first_name,
5          A. last_name,
6          C. city,
7          D. country,
8  SUM (E.amount) AS total_amount_paid
9  FROM customer A
10 INNER JOIN payment E ON A.customer_id = E.customer_id
11 INNER JOIN address B ON A.address_id = B.address_id
12 INNER JOIN city C ON B.city_id = C.city_id
13 INNER JOIN country D ON C.country_id = D.country_id
14 WHERE C. city IN ('Aurora', 'Acua', 'Citrus Heights', 'Iwaki', 'Amb
15 GROUP BY A. customer_id,
16           A. first_name,
17           A. last_name,
18           C. city,
19           D. country
20 ORDER BY total_amount_paid DESC
21 LIMIT 5)
22 SELECT AVG (total_amount_paid)
23 FROM average_amount_cte
```

Data Output Messages Notifications

	avg numeric
1	105.5540000000000000

Total rows: 1 of 1 Query complete 00:00:00.067

Query Query History

```
1  WITH top_5_count_cte (customer_id, first_name, last_name, city, country) AS
2  (SELECT
3          A. customer_id,
4          A. first_name,
5          A. last_name,
6          C. city,
7          D. country,
8  SUM (E.amount) AS total_amount_paid
9  FROM customer A
10 INNER JOIN address B ON A.address_id = B.address_id
11 INNER JOIN city C ON B.city_id = C.city_id
12 INNER JOIN country D ON C.country_id = D.country_id
13 INNER JOIN payment E ON A.customer_id = E.customer_id
14 WHERE C. city IN ('Aurora', 'Acua', 'Citrus Heights', 'Iwaki', 'Ambattur', 'Shanwei', 'So Leo
15 GROUP BY A. customer_id,
16          A. first_name,
17          A. last_name,
18          C. city,
19          D. country
20 ORDER BY total_amount_paid DESC
21 LIMIT 5)
22 SELECT
23     D.country,
24     COUNT(DISTINCT A.customer_id) AS all_customer_count,
25     COUNT(top_5_count_cte) AS top_customer_count
26 FROM customer A
27 INNER JOIN address B ON A.address_id = B.address_id
28 INNER JOIN city C ON B.city_id = C.city_id
29 INNER JOIN country D ON C.country_id = D.country_id
30 LEFT JOIN top_5_count_cte ON A. customer_id = top_5_count_cte.customer_id
```

Total rows: 5 of 5 | Query complete 00:00:00.262 | Ln 30, Col 74

	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	India	60	1
2	China	53	1
3	United States	36	1
4	Japan	31	1
5	Mexico	30	1

The first step was putting the CTE at the beginning of the query previously used in Task 3.8. To define the CTE, the WITH clause was used, then gave the CTE a name (*top_5_count_cte*)

After having added the subquery, added the outer statements using as reference the CTE used in the first line 1.

In the second query, the trickiest part (personally) was adding a LEFT JOIN to link the CTE

Task 3.9 Ana María Tiscareno

Step 2: Compare the performance of your CTEs and subqueries

Which approach do you think will perform better and why?

In my opinion CTEs are much more readable than subqueries at least it was easier to follow the steps. A CTE can be used many times within a query, whereas a subquery can only be used once. This can make the query definition much shorter.

In cases where there is a need for Multiple nested subqueries, it can become complex and hard to follow.

Subqueries:

1. Query complete 00:00:00.127
2. Query complete 00:00:00.087

CTEs:

1. .067 msec.
3. .262 msec.

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

At first, you might think that there's almost no difference between subqueries and CTEs. We've used both a subquery and a CTE in the FROM clause and the syntax was only a little different. However, don't forget the 3.8 exercises we used a subquery in the WHERE clause there. You couldn't use a CTE there.