3.9: Common Table Expressions

```
1 WITH average_cte AS (
         SELECT
             A.customer_id,
             A.first name.
              A.last_name,
            B.address.
             C.city,
            D.country,
             SUM(E.amount) AS total_amount_paid
         FROM customer A
         INNER JOIN address B ON A.address_id = B.address_id
11
         INNER JOIN city C ON B.city_id = C.city_id
         INNER JOIN country D ON C.country_id = D.country_id
INNER JOIN payment E ON A.customer_id = E.customer_id
13
         WHERE city IN ('Aurora', 'Acua', 'Citrus Heights', 'Iwaki', 'Ambattur', 'Shanwei', 'So Leopoldo', 'Teboksary', 'Tianjin', 'Cianjur')
GROUP BY A.customer_id, A.first_name, A.last_name, B.address, C.city, D.country
15
16
         ORDER BY total_amount_paid DESC
         LIMIT 5
18
19
20
   SELECT
21
         AVG(total_amount_paid) AS avg_amount_paid
22
         FROM average_cte
                                     $ ± ~
           avg_amount_paid
           numeric
                              105.554
```

I basically took the already written inner query and built an CTE to define this query in a different way. I then queried the main statement by looking for the average of the paid amount of the top 5 customers. In the second task I proceeded similarly – I took the existing inner query and changed it to an CTE and used on the one hand to identify the top_customer_counts.

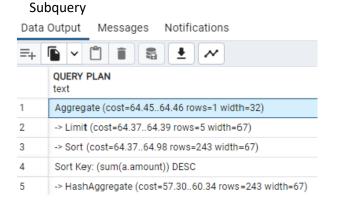
	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	India	60	1
2	China	53	1
3	United States	36	1
4	Japan	31	1
5	Mexico	30	1

2. Compare the performance of your CTEs and subqueries.

1. Which approach do you think will perform better and why?

I think the CTE performs better if the amount of variables and the complexity of the query increases. Nevertheless, the readability of the code is much easier using CTEs.

Task 1



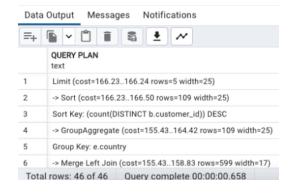
Data Output Messages Explain × Notifications QUERY PLAN

2 -> Limit (cost=65.02..65.04 rows=5 width=87)
3 -> Sort (cost=65.02..65.63 rows=244 width=87)
4 Sort Key: (sum(e.amount)) DESC
5 -> HashAggregate (cost=57.92..60.97 rows=244 width=8
6 Group Key: a.customer_id, b.address, c.city, d.country
Total rows: 22 of 22 Query complete 00:00:00.081

Aggregate (cost=65.10..65.11 rows=1 width=32)

Task 2

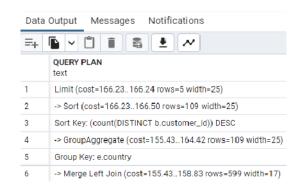
Subquery



CTE

CTE

text



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

The first task was relatively easy and it was not too difficult to replace the inner query with the CTE. However, the second task became quite complex due to the requirement of an deeper understanding of the syntax and the functionalities of the JOIN Commands. Overall, it was the most difficult task so far.