3.9: Common Table Expressions

```sql
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
    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
    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
    ORDER BY total_amount_paid DESC
    LIMIT 5
)
SELECT
    AVG(total_amount_paid) AS avg_amount_paid
    FROM average_cte
```

	avg_amount_paid numeric
1	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.

```sql
WITH top_5_customers_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
    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
    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
    ORDER BY total_amount_paid DESC
    LIMIT 5
)
SELECT
    D.country,
    COUNT(DISTINCT A.customer_id) AS all_customer_count,
    COUNT(DISTINCT top_5_customers_cte.customer_id) 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 A.customer_id = top_5_customers_cte.customer_id
GROUP BY D.country
ORDER BY all_customer_count DESC
LIMIT 5;
```

	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

Subquery

Data Output Messages Notifications

	QUERY PLAN text
1	Aggregate (cost=64.45..64.46 rows=1 width=32)
2	-> Limit (cost=64.37..64.39 rows=5 width=67)
3	-> Sort (cost=64.37..64.98 rows=243 width=67)
4	Sort Key: (sum(a.amount)) DESC
5	-> HashAggregate (cost=57.30..60.34 rows=243 width=67)

CTE

Data Output Messages Explain × Notifications

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

Task 2

Subquery

Data Output Messages Notifications

	QUERY PLAN text
1	Limit (cost=166.23..166.24 rows=5 width=25)
2	-> Sort (cost=166.23..166.50 rows=109 width=25)
3	Sort Key: (count(DISTINCT b.customer_id)) DESC
4	-> GroupAggregate (cost=155.43..164.42 rows=109 width=25)
5	Group Key: e.country
6	-> Merge Left Join (cost=155.43..158.83 rows=599 width=17)

Total rows: 46 of 46 Query complete 00:00:00.658

CTE

Data Output Messages Notifications

	QUERY PLAN text
1	Limit (cost=166.23..166.24 rows=5 width=25)
2	-> Sort (cost=166.23..166.50 rows=109 width=25)
3	Sort Key: (count(DISTINCT b.customer_id)) DESC
4	-> GroupAggregate (cost=155.43..164.42 rows=109 width=25)
5	Group Key: e.country
6	-> Merge Left Join (cost=155.43..158.83 rows=599 width=17)

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.