Asawer Maknoon

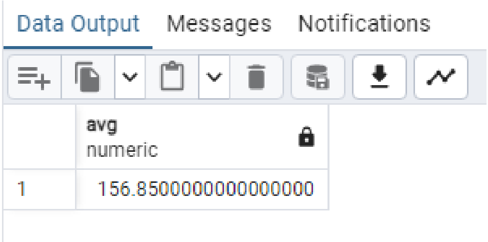
Task 3.9 Common Table Expressions

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

- **Step 1, task 3.8**

```
WITH average_amount_cte (customer_id, first_name, last_name, country, city) AS (
  SELECT
    A.customer_id,
    A.first_name,
    A.last_name,
    D.country,
    C.city,
    SUM(B.amount) AS total_amount_paid
  FROM
    customer A
  JOIN payment B ON A.customer_id = B.customer_id
  JOIN address E ON A.address_id = E.address_id
  JOIN city C ON E.city_id = C.city_id
  JOIN country D ON C.country_id = D.country_id
  WHERE C.city IN (
             SELECT city
             FROM customer A
            JOIN address E ON A.address_id = E.address_id
            JOIN city C ON E.city_id = C.city_id
            JOIN country D ON C.country_id = D.country_id
      GROUP BY C.city
      ORDER BY COUNT(A.customer_id) DESC
      LIMIT 10
    )
```

```
 GROUP BY
     A.customer_id,
     A.first_name,
     A.last_name,
     D.country,
     C.city
  ORDER BY
     total_amount_paid DESC
  LIMIT 5
)
SELECT AVG(total_amount_paid) FROM average_amount_cte;
```



	avg numeric
1	156.8500000000000000

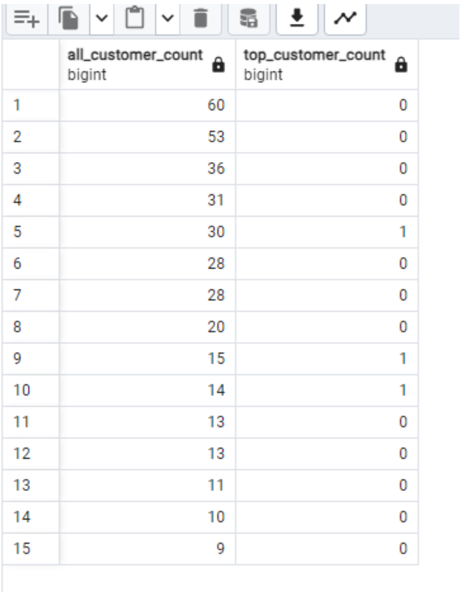
Step 2, task 3.8

```
WITH top_5_customers_cte AS (
  SELECT
     D.country,
     COUNT(DISTINCT A.customer_id) AS all_customer_count,
     COUNT(DISTINCT top_5_customers.customer_id) AS top_customer_count
  FROM customer A
  JOIN address E ON A.address_id = E.address_id
  JOIN city C ON E.city_id = C.city_id
  JOIN country D ON C.country_id = D.country_id
  LEFT JOIN (
     SELECT
```

```
            A.customer_id,
            A.first_name,
            A.last_name,
            D.country,
            C.city,
            SUM(B.amount) AS total_amount_paid
        FROM customer A
        JOIN payment B ON A.customer_id = B.customer_id
        JOIN address E ON A.address_id = E.address_id
        JOIN city C ON E.city_id = C.city_id
        JOIN country D ON C.country_id = D.country_id
        WHERE C.city IN (
            SELECT city
            FROM customer A
            JOIN address E ON A.address_id = E.address_id
            JOIN city C ON E.city_id = C.city_id
            JOIN country D ON C.country_id = D.country_id
            GROUP BY C.city
            ORDER BY COUNT(A.customer_id) DESC
            LIMIT 10
        )
        GROUP BY A.customer_id, A.first_name, A.last_name, D.country, C.city
        ORDER BY total_amount_paid DESC
        LIMIT 5
    ) AS top_5_customers ON A.customer_id = top_5_customers.customer_id
    GROUP BY D.country
    ORDER BY all_customer_count DESC
    LIMIT 15
)
```

SELECT all_customer_count,top_customer_count

FROM top_5_customers_cte;



	all_customer_count bigint	top_customer_count bigint
1	60	0
2	53	0
3	36	0
4	31	0
5	30	1
6	28	0
7	28	0
8	20	0
9	15	1
10	14	1
11	13	0
12	13	0
13	11	0
14	10	0
15	9	0

- To create a CTE from the previous subqueries (task 3.8 /step 1 and 2). I grabbed the same queries from the previous task as follows:
  - First, I used "**WITH**" clause to create a temporary result set.
  - Then I defined this CTE with a name and specified the columns contained within it.
  - After that I added the subqueries from the previous tasks in parenthesis.
  - Finally, I added an outer query (main query) and referred back to the CTE I made at the beginning.

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

Query 1 task 3.8: "Aggregate (cost=134.93..134.94 rows=1 width=32)" 94 msec.
Query 2 task 3.8: "Limit (cost=235.84..235.84 rows=1 width=25)" 257 msec.
Query 1 task 3.9: "Aggregate (cost=134.93..134.94 rows=1 width=32)" 267 msec.
Query 2 task 3.9: " Limit (cost=237.97..238.01 rows=15 width=25)"354 msec.

- In my opinion, CTEs have better performance than subqueries since they make the query more readable and much easier to understand compared to subqueries specially when we are dealing with long sql codes.
- CTEs also have to be define only one time. Conversely, subqueries have to be defined every time we want to refer to them.
- In fact, I was expected to get opposite results, means CTEs will perform the processing in less time than subqueries. But after using EXPLAIN function, I realized that there is something wrong with my queries.

Step 3: challenges I faced when replacing subqueries with CTEs:

There are many issues I met when replacing subqueries with CTEs. One of them was understanding how sql processing the query in both CTE and subquery. Second thing, I was expected to get fewer subqueries after replacing with CTE which make it a bit longer. In addition, as a beginner I was so confused when reading the queries in both CTE and subquery.