Data Immersion Ach 03.09

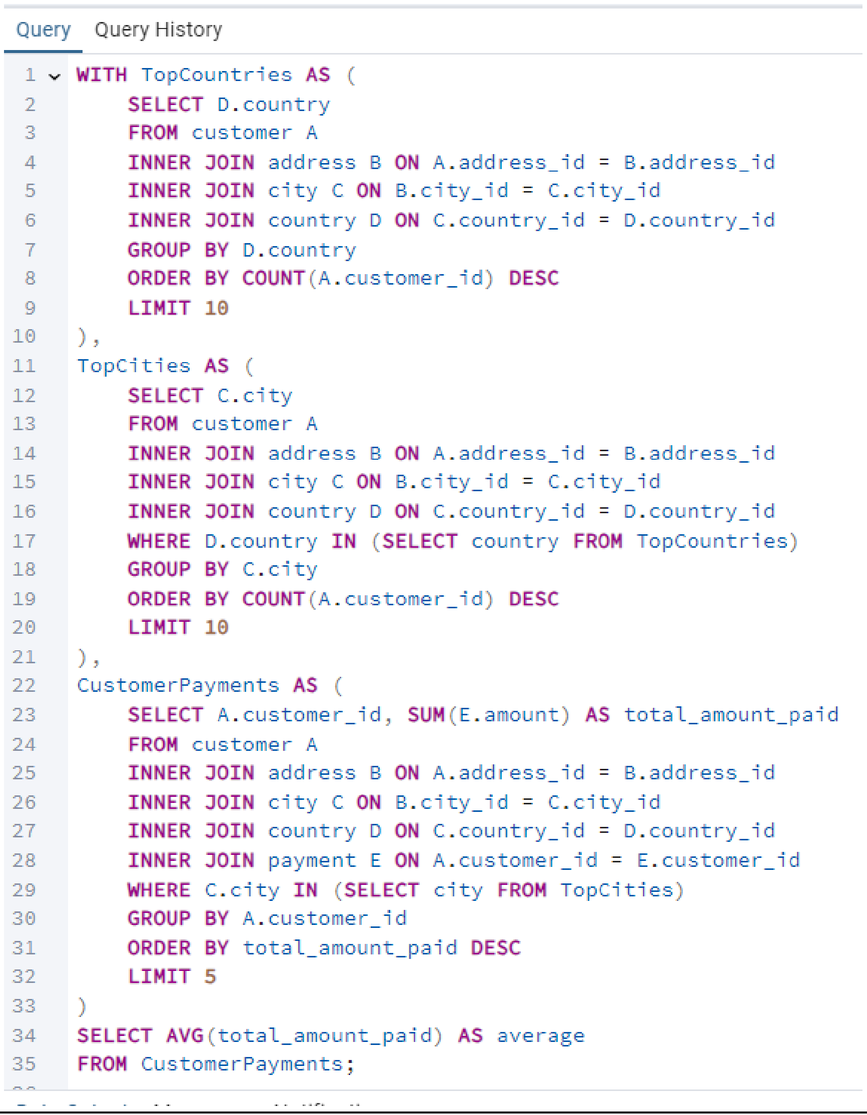
Ryan Wick – 02/07/2025

Directions

- Create a new text document and call it "Answers 3.9." You'll save your queries, outputs, and written answers in this document.

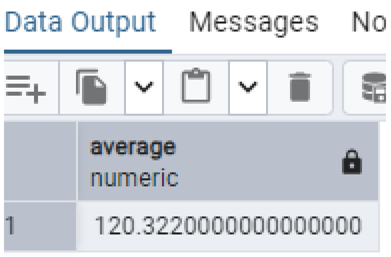
- **Step 1: Answer the business questions from steps 1 and 2 of task 3.8 using CTEs**
- Rewrite your queries from steps 1 and 2 of task 3.8 as CTEs.
    - **3.8-Task 1**

QUERY


Query Query History

```
WITH TopCountries AS (
    SELECT D.country
    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
    GROUP BY D.country
    ORDER BY COUNT(A.customer_id) DESC
    LIMIT 10
),
TopCities AS (
    SELECT C.city
    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
    WHERE D.country IN (SELECT country FROM TopCountries)
    GROUP BY C.city
    ORDER BY COUNT(A.customer_id) DESC
    LIMIT 10
),
CustomerPayments AS (
    SELECT A.customer_id, 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 C.city IN (SELECT city FROM TopCities)
    GROUP BY A.customer_id
    ORDER BY total_amount_paid DESC
    LIMIT 5
)
SELECT AVG(total_amount_paid) AS average
FROM CustomerPayments;
```

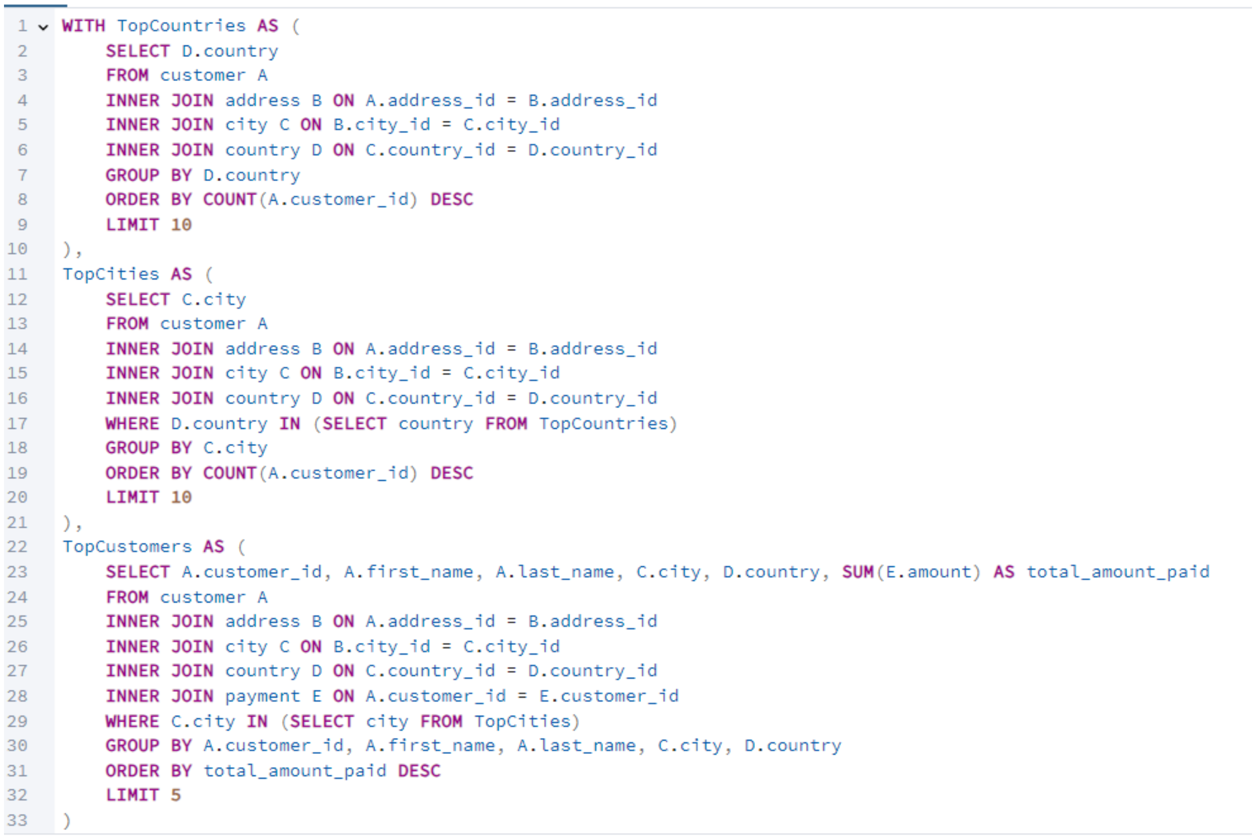
OUTPUT



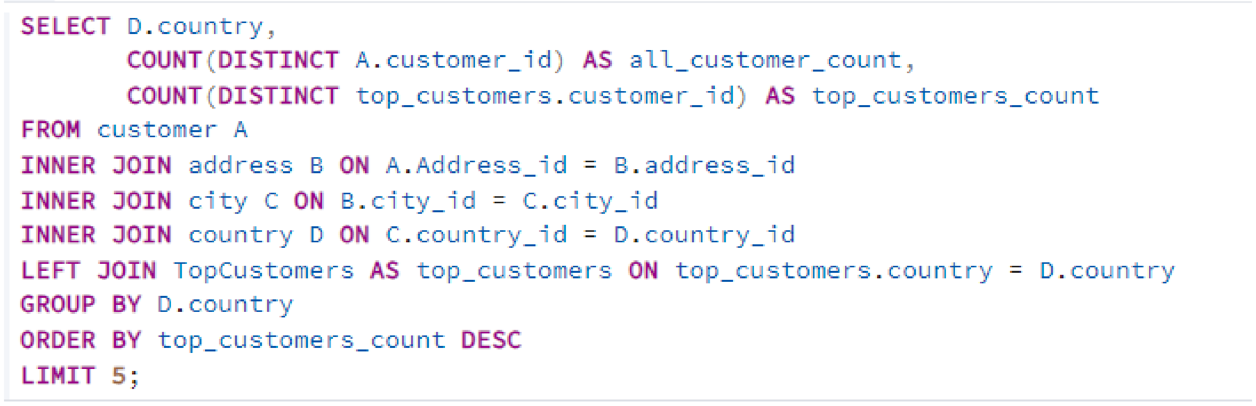
```
average
numeric
1  120.3220000000000000
```

- **3.8-Task 2**

QUERY

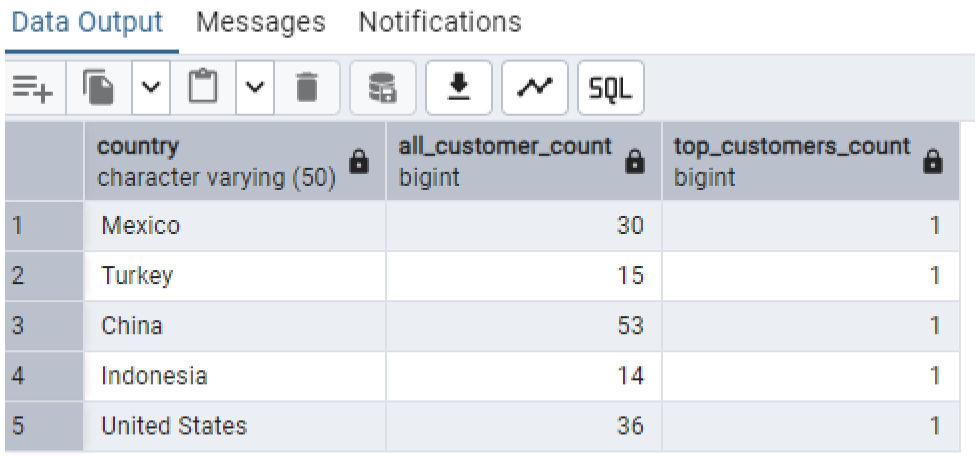


```
WITH TopCountries AS (
    SELECT D.country
    FROM customer A
    INNER JOIN address B ON A.address_id = B.address_id
    INNER JOIN city C ON B.city_id = C.city_id
    INNER JOIN country D ON C.country_id = D.country_id
    GROUP BY D.country
    ORDER BY COUNT(A.customer_id) DESC
    LIMIT 10
),
TopCities AS (
    SELECT C.city
    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
    WHERE D.country IN (SELECT country FROM TopCountries)
    GROUP BY C.city
    ORDER BY COUNT(A.customer_id) DESC
    LIMIT 10
),
TopCustomers AS (
    SELECT A.customer_id, A.first_name, A.last_name, 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 C.city IN (SELECT city FROM TopCities)
    GROUP BY A.customer_id, A.first_name, A.last_name, C.city, D.country
    ORDER BY total_amount_paid DESC
    LIMIT 5
)
```



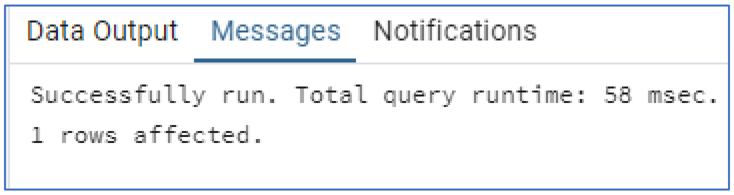
```
SELECT D.country,
       COUNT(DISTINCT A.customer_id) AS all_customer_count,
       COUNT(DISTINCT top_customers.customer_id) AS top_customers_count
FROM customer A
INNER JOIN address B ON A.Address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
LEFT JOIN TopCustomers AS top_customers ON top_customers.country = D.country
GROUP BY D.country
ORDER BY top_customers_count DESC
LIMIT 5;
```

OUTPUT


Data Output Messages Notifications

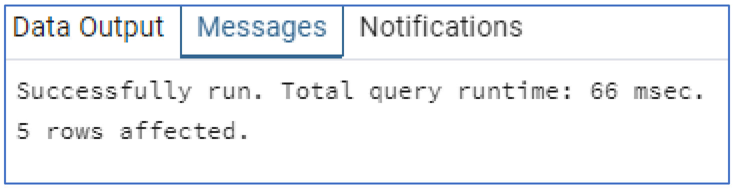
	country character varying (50)	all_customer_count bigint	top_customers_count bigint
1	Mexico	30	1
2	Turkey	15	1
3	China	53	1
4	Indonesia	14	1
5	United States	36	1

- Copy-paste your CTEs and their outputs into your answers document.
- Write 2 to 3 sentences explaining how you approached this step, for example, what you did first, second, and so on.
    - Essentially I began approaching this step by being completely lost. Next, after a short cry I went and looked over the reading again. Now while it wasn't in the reading and not 100% sure it's a constant thing I did notice in examples that the inner most sub-query is what became the first CTE. So, I imagined it as essentially flipping the code inside out at least in just the way of the syntax. From there it just took me a few to get my WHERE closes cleaned up and I was set.
- **Step 2: Compare the performance of your CTEs and subqueries.**
- Which approach do you think will perform better and why?
    - I would imagine CTE will just barely outperform subqueries right now since our code is short. I'd imagine more times than not that CTE would outperform subqueries in production environments.
- Compare the costs of all the queries by creating query plans for each one.
    - 3.8-Task 1 SUB


Data Output Messages Notifications

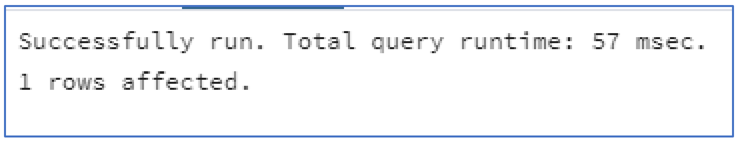
```
Successfully run. Total query runtime: 58 msec.
1 rows affected.
```

- 3.8-Task 2 SUB


Data Output Messages Notifications

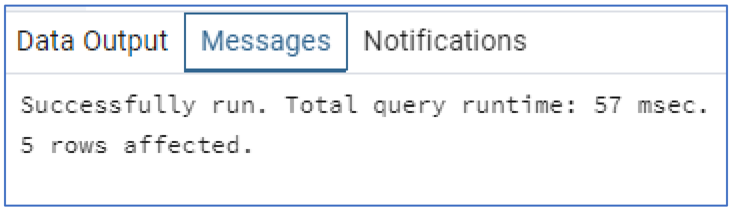
```
Successfully run. Total query runtime: 66 msec.
5 rows affected.
```

- 3.8-Task 1 CTE



```
Successfully run. Total query runtime: 57 msec.
1 rows affected.
```

- 3.8-Task 2 CTE



```
Successfully run. Total query runtime: 57 msec.
5 rows affected.
```

- The EXPLAIN command gives you an *estimated* cost. To find out the actual speed of your queries, run them in pgAdmin 4. After you've run each query, a popup window will display its speed in milliseconds.
- Did the results surprise you? Write a few sentences to explain your answer.
    - No, not really. Again, the code is small still so there wasn't much room for change. It did, however, give backing to my hypothesis of CTEs typically being the more efficient route.
- **Step 3:**
    - Write 1 to 2 paragraphs on the challenges you faced when replacing your subqueries with CTEs.
        - Well, I feel like I answered most of this in Step 1.4, but I'll give it a try. The first major challenge I faced was just being able to understand both syntax of the CTE and subqueries. Once I began to see where the items were and what was changed to make them into CTE it began to click. Again, I'm not totally sure if it always works this way but I found that the inner most subquery becomes the top CTE. From there it can be followed building the CTEs from the "inside out" of the subqueries if that makes sense. Not totally sure if CTEs can be in any order just like function in other programing languages. Though even if they can I may try and keep that pattern since it seems to help me understand and stay organized in thought.
- **Step 4:**
    - Save your "Answers 3.9" document as a PDF and upload it here for your tutor to review.