Task 3.9 Chris Arnold

Step 1

Rewrite your queries as CTEs from steps 1 and 2 from 3.8

Step 1

```
WITH average_amount_paid_cte (customer_id,
    first_name,
        last_name,
        city,
        total_amount_paid)
AS
(SELECT A.customer_id,
        A.first_name,
        A.last_name,
        C.city,
        SUM(E.amount) AS total_amount_paid
FROM payment E
INNER JOIN customer A ON E.customer_id=A.customer_id
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 C.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang',
                 'Sivas', 'Celaya', 'So Leopoldo')
GROUP BY A.customer_id, D.country, C.city
ORDER BY total_amount_paid DESC
LIMIT 5)
SELECT  AVG (total_amount_paid)
FROM average_amount_paid_cte
```

ANSWER displayed

Data Output | Explain | Messages | Notifications

	avg numeric
1	107.3540000000000000

Step 2

```
WITH top_customer_count_cte AS  (SELECT A.customer_id,
        A.first_name,
        A.last_name,
        C.city,
        D. country_id,
        SUM(E.amount) AS total_amount
FROM payment E
INNER JOIN customer A ON E.customer_id=A.customer_id
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 C.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)',
                 'Kurashiki', 'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
GROUP BY A.customer_id, D.country_id, C.city
ORDER BY total_amount DESC
LIMIT 5),
all_customer_count_cte AS(SELECT  D.country,
        COUNT (DISTINCT A.customer_id) AS all_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
GROUP BY D.country)

SELECT  D.country,
       COUNT (DISTINCT A.customer_id) AS all_customer_count,
       COUNT (DISTINCT top_customer_count_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_customer_count_cte ON D.country_id = top_customer_count_cte.country_id
GROUP BY D.country
ORDER BY top_customer_count DESC
LIMIT 5;
```

	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	Mexico	30	2
2	United States	36	1
3	India	60	1
4	Turkey	15	1
5	American Samoa	1	0

- Write 2 to 3 sentences explaining how you approached this step, for example, what you did first, second, and so on.

I first referenced the data mat and location of each piece that was needed in the puzzle. The query was then written to get the top 5 countries or to get the number of customers in all countries. The CTE was then defined as the average amount paid for step one and the top customer count/all customer count for step 2. Finally the query was finished to seek what the question in the task was asking.

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

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

   For this situation I would have originally guessed that it wouldn't make that big of a difference because we are not doing anything super complicated. Subqueries in general are slower because we have to define them every time we reference them, so the CTE should perform better.

2. Compare the costs of all the queries by creating query plans for each one.

QUERY	COST
step 1 (3.9.)	cost=66.09..66.10 rows=1 width=32
step 2 (3.9.)	cost=168.49..168.50 rows=5 width=25
step 1 (3.8.)	cost=66.09..66.10 rows=1 width=32
step 2 (3.8.)	cost=168.49..168.50 rows=5 width=25

3. The `EXPLAIN` command gives you an *estimated* cost. To find out the actual speed of your queries, run them in pgAdmin 4. After each query has been run, a pop-up window will display its speed in milliseconds.

QUERY	TOTAL QUERY RUNTIME
step 1 (3.9.)	41 msec. 22 rows affected.
step 2 (3.9.)	33 msec. 45 rows affected.
step 1 (3.8.)	42 msec. 22 rows affected.
step 2 (3.8.)	44 msec. 45 rows affected.

4. Did the results surprise you? Write a few sentences to explain your answer.

I in general was not surprised that the results did not differ that much. One thing to note, is that for step 2, the query time difference from 3.8-3.9 took ¾ the amount of time in 3.9 vs 3.8. Extrapolate this to something much bigger and the CTE wins by a long shot.

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

In general, the idea of the CTE was not very complicated for me. The step two was the most complicated because of there being 2 CTEs, however even that was just a step by step process that was much simpler than any proof I had to do in graduate level mathematics courses I took. It did take a bit of time to think everything out, but I imagine that the more practice that I get, the easier things will come to me.