Data Immersion
Exercise 3.9
Common Table Expressions

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

Query:

Welcome | Rockbuster/postgr... | Rockbuster/postgr... | Rockbuster/postgr... | Rockbuster/postgres@PostgreSQL 17*

Rockbuster/postgres@PostgreSQL 17

No limit

Query | Query History

```sql
With Top_Ten_Countries AS
(select cntry.country
    from customer cust
    inner join address addr on cust.address_id = addr.address_id
    inner join city city on addr.city_id = city.city_id
    inner join country cntry on city.country_id = cntry.country_id
    group by cntry.country
    order by count(cust.customer_id) desc limit 10),
Top_Ten_Cities AS
(SELECT city.city
    FROM customer cust
    INNER JOIN address addr ON cust.address_id = addr.address_id
    INNER JOIN city city ON addr.city_id = city.city_id
    INNER JOIN country cntry ON city.country_id = cntry.country_id
    where cntry.country IN
        (select * from Top_Ten_Countries)
    group by cntry.country, city.city
    order by count(cust.customer_id) desc limit 10),
Top_Five_Customer AS
(SELECT SUM(pay.amount) AS total_amount
    FROM payment pay
    INNER JOIN customer cust ON pay.customer_id = cust.customer_id
    INNER JOIN address addr ON cust.address_id = addr.address_id
    INNER JOIN city city ON addr.city_id = city.city_id
    INNER JOIN country cntry ON city.country_id = cntry.country_id
    where city.city in
        ( select * from Top_Ten_Cities)
    group by cust.first_name, cust.last_name, city.city, cntry.country order by total_amount desc limit 5)
select avg(total_amount)
from Top_Five_Customer
```

Output:

Data Output | Messages | Notifications

	avg numeric
1	105.5540000000000000

Step 1b: How many of the top five customers identified in step one are based in each country?

Query:

```
With Top_Ten_Countries AS
(SELECT cntry.country
	from customer cust
	INNER JOIN address addr on cust.address_id = addr.address_id
	INNER JOIN city city on addr.city_id = city.city_id
	INNER JOIN country cntry on city.country_id = cntry.country_id
	GROUP BY cntry.country
	ORDER BY COUNT (cust.customer_id) DESC LIMIT 10),
Top_Ten_Cities AS
(SELECT city.city
	FROM customer cust
	INNER JOIN address addr ON cust.address_id = addr.address_id
	INNER JOIN city city ON addr.city_id = city.city_id
	INNER JOIN country cntry ON city.country_id = cntry.country_id
	WHERE cntry.country IN
		(SELECT * FROM Top_Ten_Countries)
	GROUP BY cntry.country, city.city
	ORDER BY COUNT (cust.customer_id) DESC LIMIT 10),
Top_Customer AS
(SELECT SUM(pay.amount) AS total_amount, cntry.country
	FROM payment pay
	INNER JOIN customer cust ON pay.customer_id = cust.customer_id
	INNER JOIN address addr ON cust.address_id = addr.address_id
	INNER JOIN city city ON addr.city_id = city.city_id
	INNER JOIN country cntry ON city.country_id = cntry.country_id
	where city.city in
		( SELECT * FROM Top_Ten_Cities)
	GROUP BY cust.first_name, cust.last_name, city.city, cntry.country ORDER BY
total_amount DESC LIMIT 5)
SELECT cntry.country,
	COUNT (DISTINCT cust.customer_id) AS all_customer_count,
	COUNT (DISTINCT Top_Customer) AS top_customer_count
FROM customer cust
INNER JOIN address addr ON cust.address_id = addr.address_id
INNER JOIN city city ON addr.city_id = city.city_id
INNER JOIN country cntry ON city.country_id = cntry.country_id
LEFT JOIN Top_Customer ON cntry.country = Top_Customer.country
GROUP BY cntry.country
```

ORDER BY top_customer_count DESC
LIMIT 5;

Output:

Welcome | Rockbuster/postgr... | Rockbuster/postgr... | Rockbuster/postgr... | Rock

Rockbuster/postgres@PostgreSQL 17

No limit

Query | Query History

```sql
    INNER JOIN address addr ON cust.address_id = addr.address_id
    INNER JOIN city city ON addr.city_id = city.city_id
    INNER JOIN country cntry ON city.country_id = cntry.country_id
    where city.city in
        ( SELECT * FROM Top_Ten_Cities)
    GROUP BY cust.first_name, cust.last_name, city.city, cntry.country OR
SELECT cntry.country,
    COUNT (DISTINCT cust.customer_id) AS all_customer_count,
    COUNT (DISTINCT Top_Customer) AS top_customer_count
FROM customer cust
INNER JOIN address addr ON cust.address_id = addr.address_id
INNER JOIN city city ON addr.city_id = city.city_id
INNER JOIN country cntry ON city.country_id = cntry.country_id
LEFT JOIN Top_Customer ON cntry.country = Top_Customer.country
GROUP BY cntry.country
ORDER BY top_customer_count DESC
LIMIT 5;
```

Data Output | Messages | Notifications

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

Write 2 to 3 sentences explaining how you approached this step

→ The idea was to rewrite the subqueries into CTE to find the top ten countries with the most of Rockbuster's customers. I began the query with the WITH clause and gave the name "Top Ten Countries". Then, I used AS followed by the query to make sure that I created a table with the top ten countries. I used the same process for top ten cities and top five customers.

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

1. ***Which approach do you think will perform better and why?***
    a. I think the CTE approach will perform better because of the length of the query. It's also easier to read and understand, both for myself and those who are from different industries. It also helps eliminate repetitive logic, making the query more efficient.
2. ***Compare the costs of all the queries by creating query plans for each one. 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.***

Query Type:	**CTE**	**Subquery**
Query 1	Cost:170.21	Cost: 127.90
Query 1	Speed:71	Speed: 73
Query 2	Cost:272.60	Cost:271.90
Query 2	Speed:77	Speed:79

    a.
3. ***Did the results surprise you? Write a few sentences to explain your answer.***
    a. The results are mostly similar, except for the first query. I was surprised by how much higher the cost of the CTE was compared to the subquery. In the second query, the costs are nearly the same. The execution speed is also quite similar, with the CTE being slightly faster than the subquery.

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

- One of the biggest challenges I faced when converting subqueries to CTEs was maintaining their structure while ensuring the results stayed the same. Troubleshooting and making adjustments took time since each subquery had to be adapted to the CTE format.