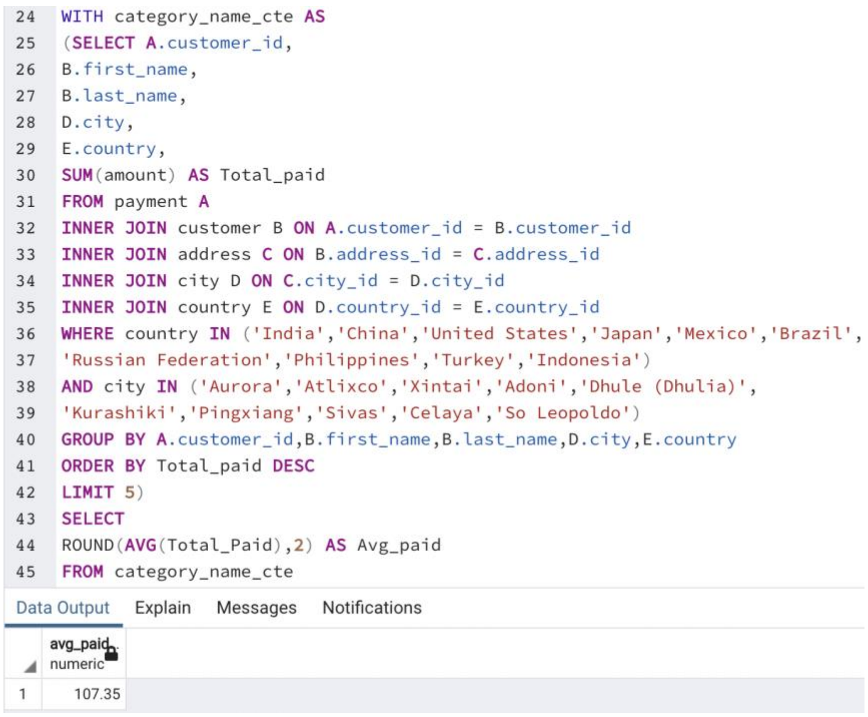
Ex. 3.9 Common Table Expressions

Step 1: Find the average amount paid by the top 5 customers (Using a CTE)

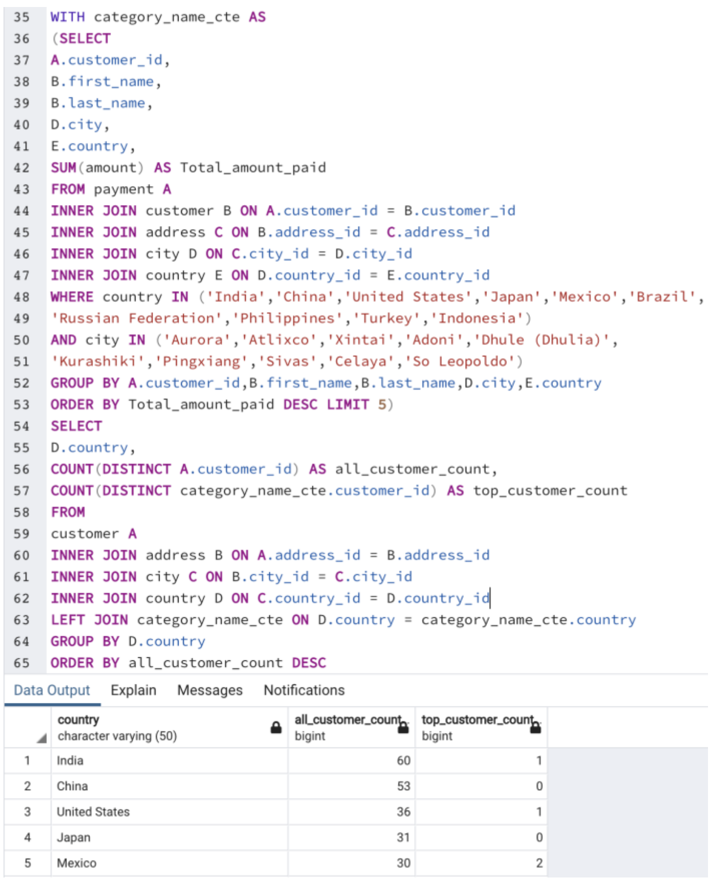


```sql
WITH category_name_cte AS
(SELECT A.customer_id,
B.first_name,
B.last_name,
D.city,
E.country,
SUM(amount) AS Total_paid
FROM payment A
INNER JOIN customer B ON A.customer_id = B.customer_id
INNER JOIN address C ON B.address_id = C.address_id
INNER JOIN city D ON C.city_id = D.city_id
INNER JOIN country E ON D.country_id = E.country_id
WHERE country IN ('India','China','United States','Japan','Mexico','Brazil',
'Russian Federation','Philippines','Turkey','Indonesia')
AND city IN ('Aurora','Atlixco','Xintai','Adoni','Dhule (Dhulia)',
'Kurashiki','Pingxiang','Sivas','Celaya','So Leopoldo')
GROUP BY A.customer_id,B.first_name,B.last_name,D.city,E.country
ORDER BY Total_paid DESC
LIMIT 5)
SELECT
ROUND(AVG(Total_Paid),2) AS Avg_paid
FROM category_name_cte
```

```
   avg_paid
   numeric
1  107.35
```

WITH category_name_cte AS
(SELECT A.customer_id,
B.first_name,
B.last_name,
D.city,
E.country,
SUM(amount) AS Total_paid
FROM payment A
INNER JOIN customer B ON A.customer_id = B.customer_id
INNER JOIN address C ON B.address_id = C.address_id
INNER JOIN city D ON C.city_id = D.city_id
INNER JOIN country E ON D.country_id = E.country_id
WHERE country IN ('India','China','United States','Japan','Mexico','Brazil',
'Russian Federation','Philippines','Turkey','Indonesia')
AND city IN ('Aurora','Atlixco','Xintai','Adoni','Dhule (Dhulia)',
'Kurashiki','Pingxiang','Sivas','Celaya','So Leopoldo')
GROUP BY A.customer_id,B.first_name,B.last_name,D.city,E.country
ORDER BY Total_paid DESC
LIMIT 5)
SELECT
ROUND(AVG(Total_Paid),2) AS Avg_paid
FROM category_name_cte

Step 2: Find out how many of the top 5 customers are based within each country (using a CTE)



```
WITH category_name_cte AS
(SELECT
A.customer_id,
B.first_name,
B.last_name,
D.city,
E.country,
SUM(amount) AS Total_amount_paid
FROM payment A
INNER JOIN customer B ON A.customer_id = B.customer_id
INNER JOIN address C ON B.address_id = C.address_id
INNER JOIN city D ON C.city_id = D.city_id
INNER JOIN country E ON D.country_id = E.country_id
WHERE country IN ('India','China','United States','Japan','Mexico','Brazil',
'Russian Federation','Philippines','Turkey','Indonesia')
AND city IN ('Aurora','Atlixco','Xintai','Adoni','Dhule (Dhulia)',
'Kurashiki','Pingxiang','Sivas','Celaya','So Leopoldo')
GROUP BY A.customer_id,B.first_name,B.last_name,D.city,E.country
ORDER BY Total_amount_paid DESC LIMIT 5)
SELECT
D.country,
COUNT(DISTINCT A.customer_id) AS all_customer_count,
COUNT(DISTINCT category_name_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 category_name_cte ON D.country = category_name_cte.country
GROUP BY D.country
ORDER BY all_customer_count DESC
```

Data Output | Explain | Messages | Notifications

	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	India	60	1
2	China	53	0
3	United States	36	1
4	Japan	31	0
5	Mexico	30	2

WITH category_name_cte AS
(SELECT
A.customer_id,
B.first_name,
B.last_name,
D.city,
E.country,
SUM(amount) AS Total_amount_paid
FROM payment A
INNER JOIN customer B ON A.customer_id = B.customer_id
INNER JOIN address C ON B.address_id = C.address_id
INNER JOIN city D ON C.city_id = D.city_id
INNER JOIN country E ON D.country_id = E.country_id
WHERE country IN ('India','China','United States','Japan','Mexico','Brazil',
'Russian Federation','Philippines','Turkey','Indonesia')
AND city IN ('Aurora','Atlixco','Xintai','Adoni','Dhule (Dhulia)',
'Kurashiki','Pingxiang','Sivas','Celaya','So Leopoldo')
GROUP BY A.customer_id,B.first_name,B.last_name,D.city,E.country
ORDER BY Total_amount_paid DESC LIMIT 5)
SELECT
D.country,
COUNT(DISTINCT A.customer_id) AS all_customer_count,
COUNT(DISTINCT category_name_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 category_name_cte ON D.country = category_name_cte.country
GROUP BY D.country
ORDER BY all_customer_count DESC

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

1. Which approach do you think will perform better and why?
   As both queries are relatively small, I would not expect any major differences between the different approaches.

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

   - Average amount paid by the top 5 customers:
     - Subquery: "Aggregate (cost=29.22..29.24 rows=1 width=32)"
     - CTE: "Aggregate (cost=29.22..29.24 rows=1 width=32)"

   - How many of the top 5 customers are based within each country:
     - Subquery: "Sort (cost=132.88..133.15 rows=109 width=25)"
     - CTE: "Sort (cost=132.88..133.15 rows=109 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.

   - Average amount paid by the top 5 customers:
     - Subquery: Total query runtime: 51 msec
     - CTE: Total query runtime: 160 msec

   - How many of the top 5 customers are based within each country:
     - Subquery: Total query runtime: 56 msec
     - CTE: Total query runtime: 49 msec

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

   The runtime for the first query (average paid by the top 5 customers) was quicker as a subquery (51 msec vs. 160 msec). This surprised me as I thought both runtimes would be closer. My assumption is that as it's a small query, using a subquery is quicker.
   For the second query (countries with to 5 customers column), the CTE was minimally quicker (56 msec vs. 49 msec). I assume as it is a longer query, there is a benefit to using the CTE approach.

Step 3:

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

The exercise forced me to think about how the CTE is set out, specifically the joins. Furthermore, on the second query, I spent some time working out that I had to change the subquery name to the CTE reference when referring to the count of the top 5 customers.