Common Table Expressions

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

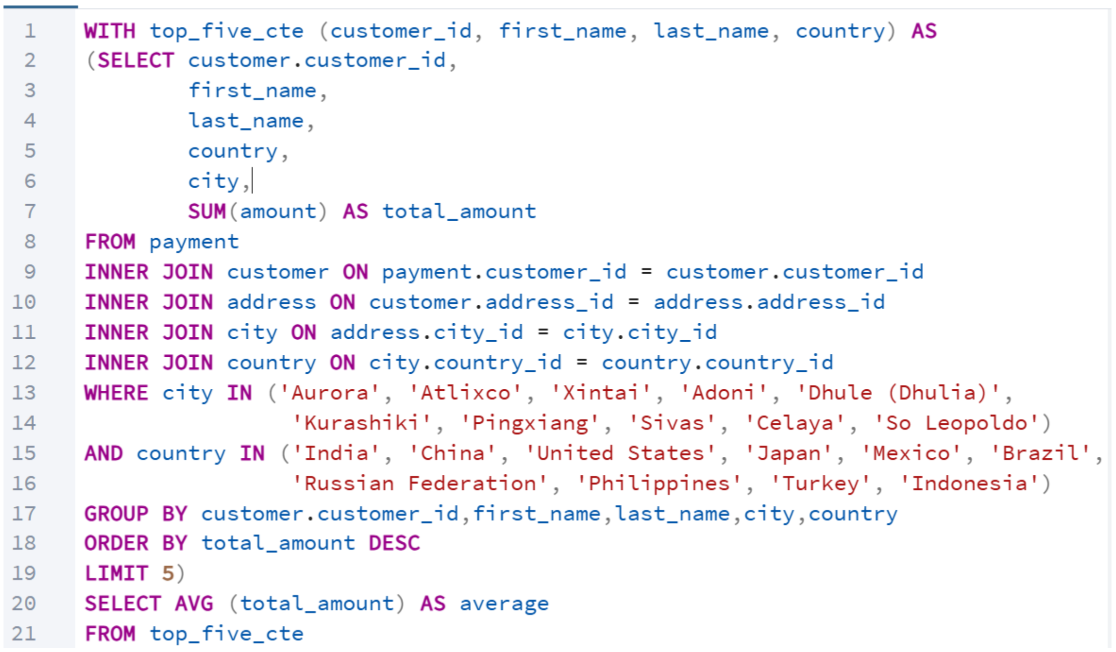
1. **Rewrite your queries from steps 1 and 2 of task 3.8 as CTEs.**
2. **Copy-paste your CTEs and their outputs into your answers document.**
3. **Write 2 to 3 sentences explaining how you approached this step, for example, what you did first, second, and so on.**

CTE commands:

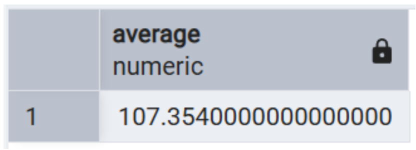
```
WITH top_five_cte (customer_id, first_name, last_name, country) AS

(SELECT customer.customer_id,

            first_name,

            last_name,

            country,

            city,

            SUM(amount) AS total_amount

FROM payment

INNER JOIN customer ON payment.customer_id = customer.customer_id

INNER JOIN address ON customer.address_id = address.address_id

INNER JOIN city ON address.city_id = city.city_id

INNER JOIN country ON city.country_id = country.country_id

WHERE city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)',  'Kurashiki', 'Pingxiang', 'Sivas',

                    'Celaya', 'So Leopoldo')

AND country IN ('India', 'China', 'United States', 'Japan', 'Mexico', 'Brazil',  'Russian Federation',

                     'Philippines', 'Turkey', 'Indonesia')

GROUP BY customer.customer_id, first_name, last_name, city, country

ORDER BY total_amount DESC

LIMIT 5)

SELECT AVG (total_amount) AS average
```

```
FROM top_five_cte
```

CTE screenshots:



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



	average numeric
1	107.3540000000000000

I used a CTE (Common Table Expression) to extract the top five customers. Then, in the main query, I calculated the average payment for these customers.

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

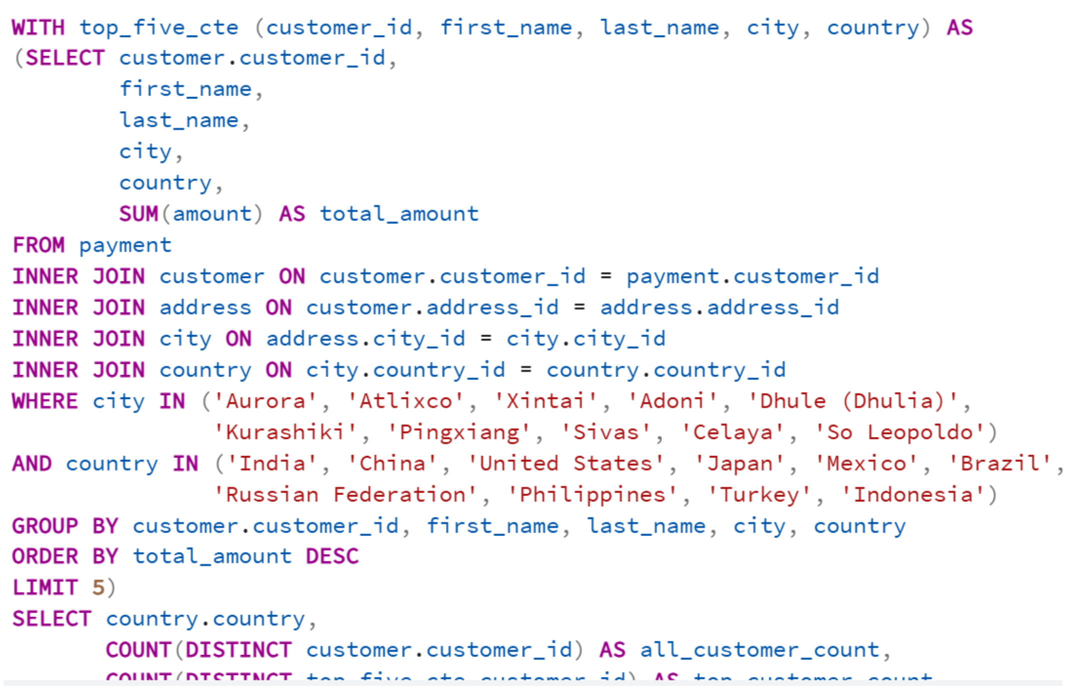
1. **Which approach do you think will perform better and why?**
2. **Compare the costs of all the queries by creating query plans for each one.**
3. **The EXPLAIN command gives you an *estimated* cost. To find out the actual speed of your queries, run them in pgAdmin 4. After you've run each query, a popup window will display its speed in milliseconds.**
4. **Did the results surprise you? Write a few sentences to explain your answer.**

CTE commands:

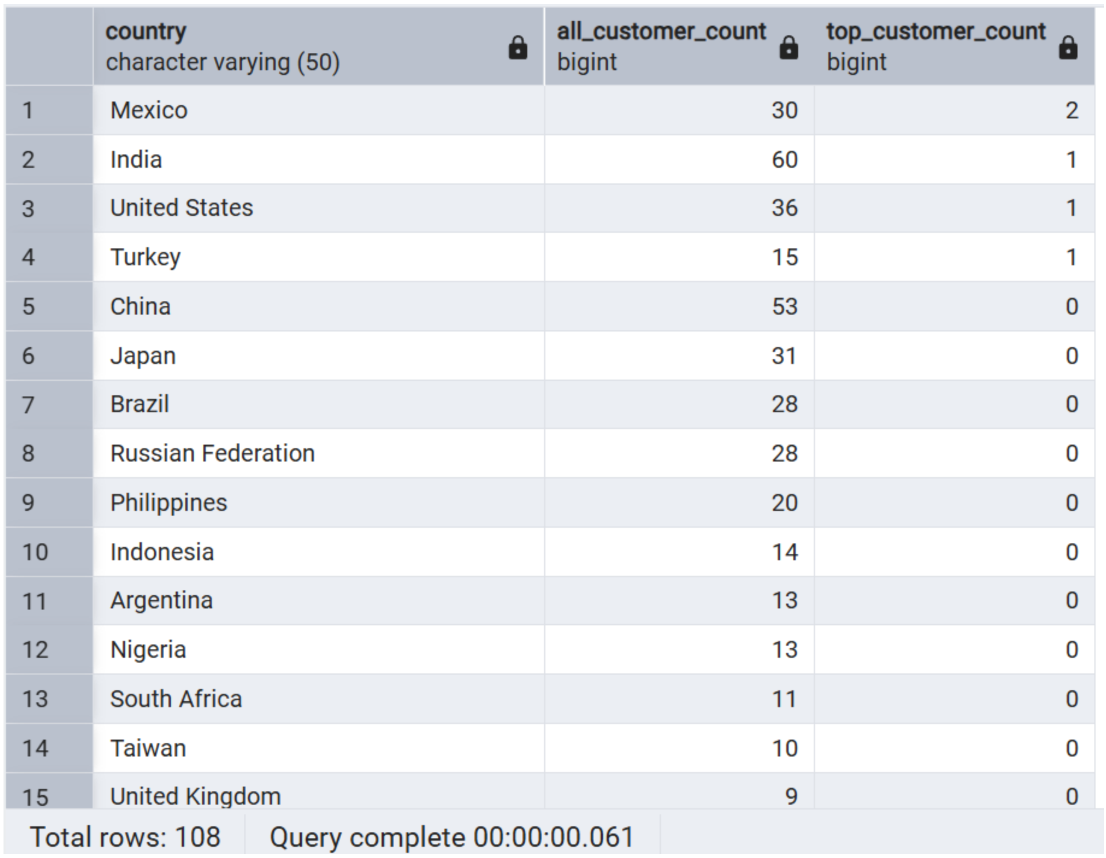
```
WITH top_five_cte (customer_id, first_name, last_name, city, country) AS
```

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

CTE screenshots:



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



	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	Mexico	30	2
2	India	60	1
3	United States	36	1
4	Turkey	15	1
5	China	53	0
6	Japan	31	0
7	Brazil	28	0
8	Russian Federation	28	0
9	Philippines	20	0
10	Indonesia	14	0
11	Argentina	13	0
12	Nigeria	13	0
13	South Africa	11	0
14	Taiwan	10	0
15	United Kingdom	9	0

Total rows: 108 Query complete 00:00:00.061

- Use a CTE to return the top five customers along with their city and country.
- In the main query, compare country counts to identify the country with the largest number of

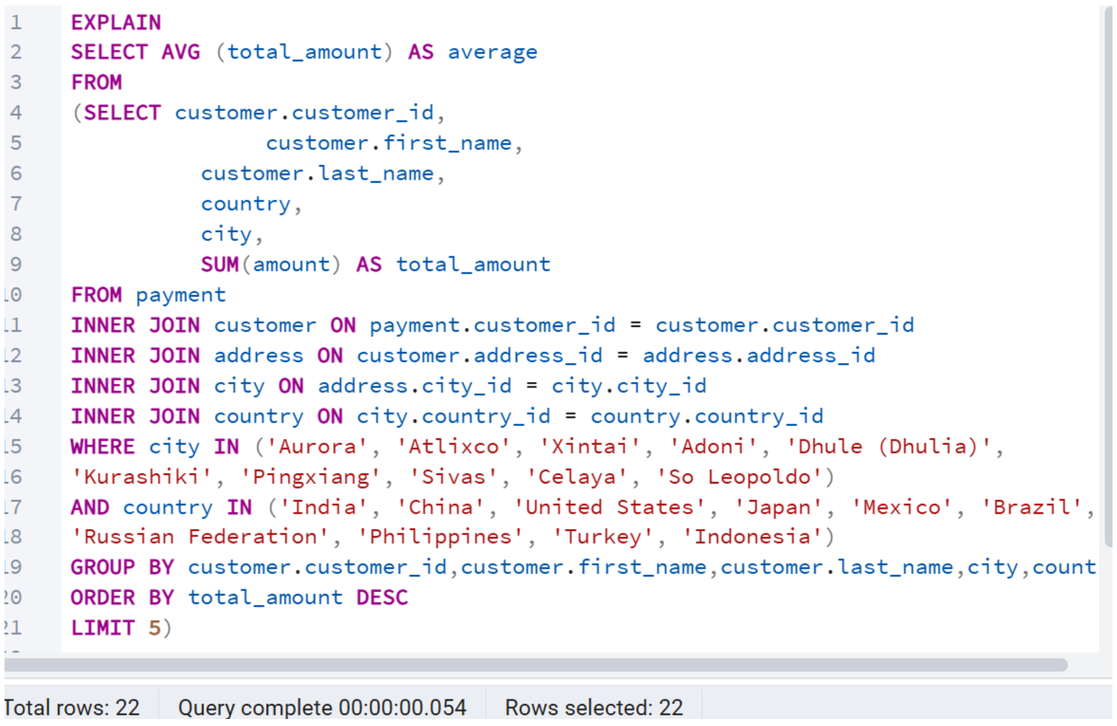
customers overall, and also show how many of the top five come from each country.

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

1. **Which approach do you think will perform better and why?**
2. **Compare the costs of all the queries by creating query plans for each one.**
3. **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.**
4. **Did the results surprise you? Write a few sentences to explain your answer.**

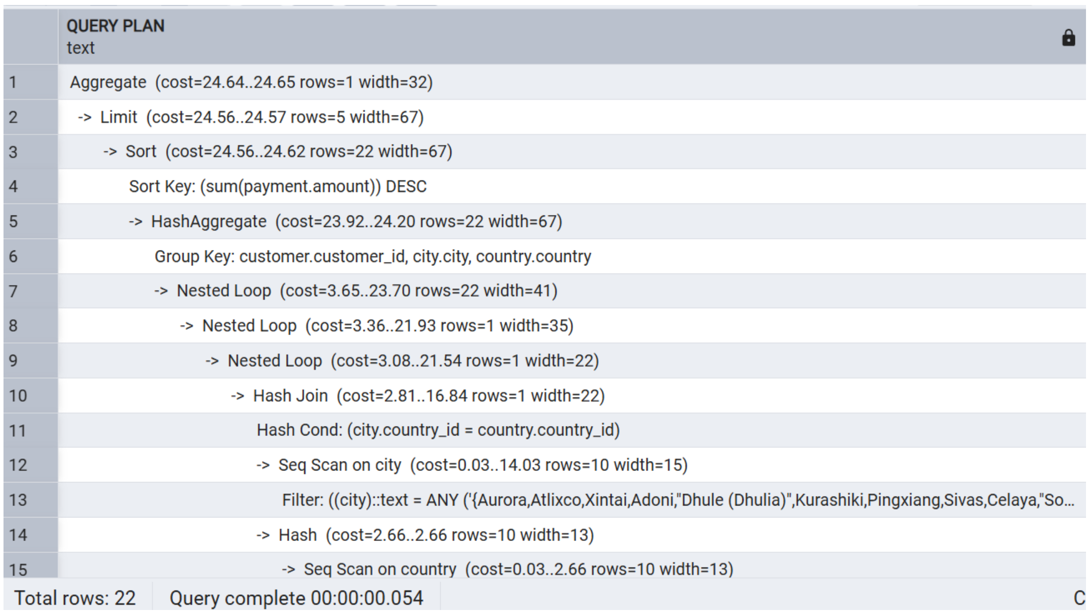
In this case, I think both CTEs and subqueries work well. But in the case when multiple subqueries are used, the query will appear long and hard to follow. In that case, I would prefer CTEs.

Subquery 1:



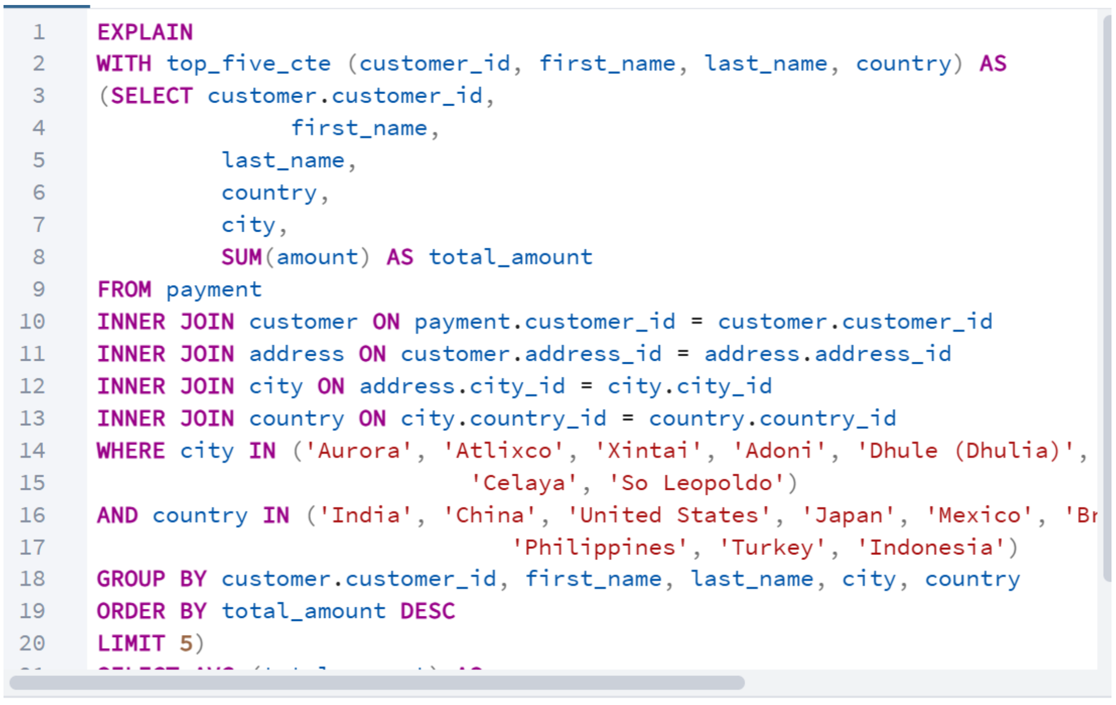
```sql
EXPLAIN
SELECT AVG (total_amount) AS average
FROM
(SELECT customer.customer_id,
               customer.first_name,
          customer.last_name,
          country,
          city,
          SUM(amount) AS total_amount
FROM payment
INNER JOIN customer ON payment.customer_id = customer.customer_id
INNER JOIN address ON customer.address_id = address.address_id
INNER JOIN city ON address.city_id = city.city_id
INNER JOIN country ON city.country_id = country.country_id
WHERE city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)',
'Kurashiki', 'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
AND country IN ('India', 'China', 'United States', 'Japan', 'Mexico', 'Brazil',
'Russian Federation', 'Philippines', 'Turkey', 'Indonesia')
GROUP BY customer.customer_id,customer.first_name,customer.last_name,city,count
ORDER BY total_amount DESC
LIMIT 5)
```

Total rows: 22   Query complete 00:00:00.054   Rows selected: 22

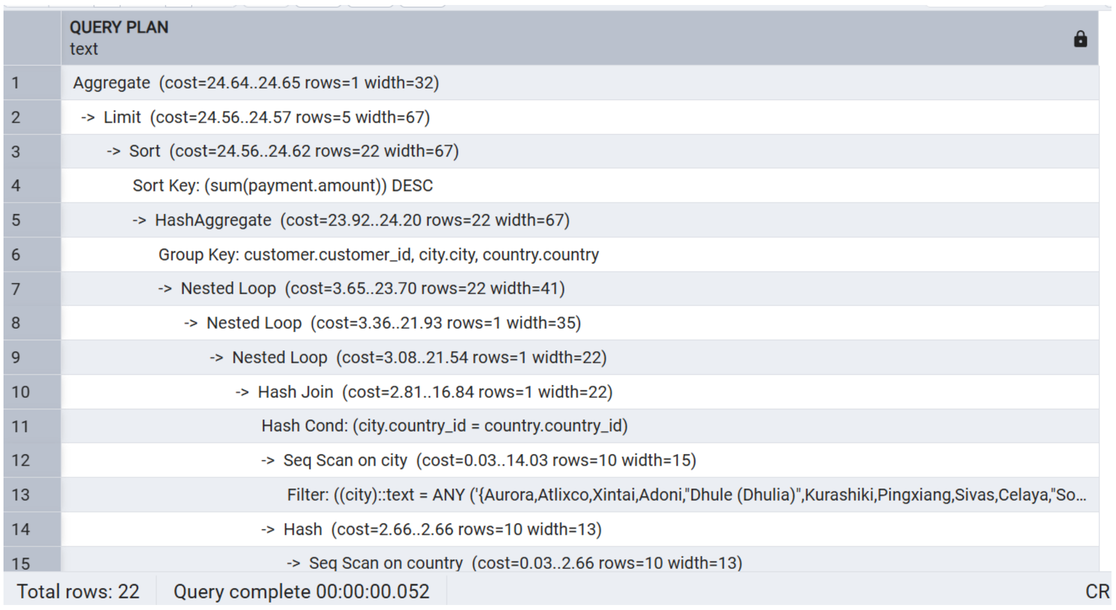


```
QUERY PLAN
text
Aggregate  (cost=24.64..24.65 rows=1 width=32)
  ->  Limit  (cost=24.56..24.57 rows=5 width=67)
        ->  Sort  (cost=24.56..24.62 rows=22 width=67)
              Sort Key: (sum(payment.amount)) DESC
              ->  HashAggregate  (cost=23.92..24.20 rows=22 width=67)
                    Group Key: customer.customer_id, city.city, country.country
                    ->  Nested Loop  (cost=3.65..23.70 rows=22 width=41)
                          ->  Nested Loop  (cost=3.36..21.93 rows=1 width=35)
                                ->  Nested Loop  (cost=3.08..21.54 rows=1 width=22)
                                      ->  Hash Join  (cost=2.81..16.84 rows=1 width=22)
                                            Hash Cond: (city.country_id = country.country_id)
                                            ->  Seq Scan on city  (cost=0.03..14.03 rows=10 width=15)
                                                  Filter: ((city)::text = ANY ('{Aurora,Atlixco,Xintai,Adoni,"Dhule (Dhulia)",Kurashiki,Pingxiang,Sivas,Celaya,"So...
                                            ->  Hash  (cost=2.66..2.66 rows=10 width=13)
                                                  ->  Seq Scan on country  (cost=0.03..2.66 rows=10 width=13)
Total rows: 22    Query complete 00:00:00.054
```

CTE 1:

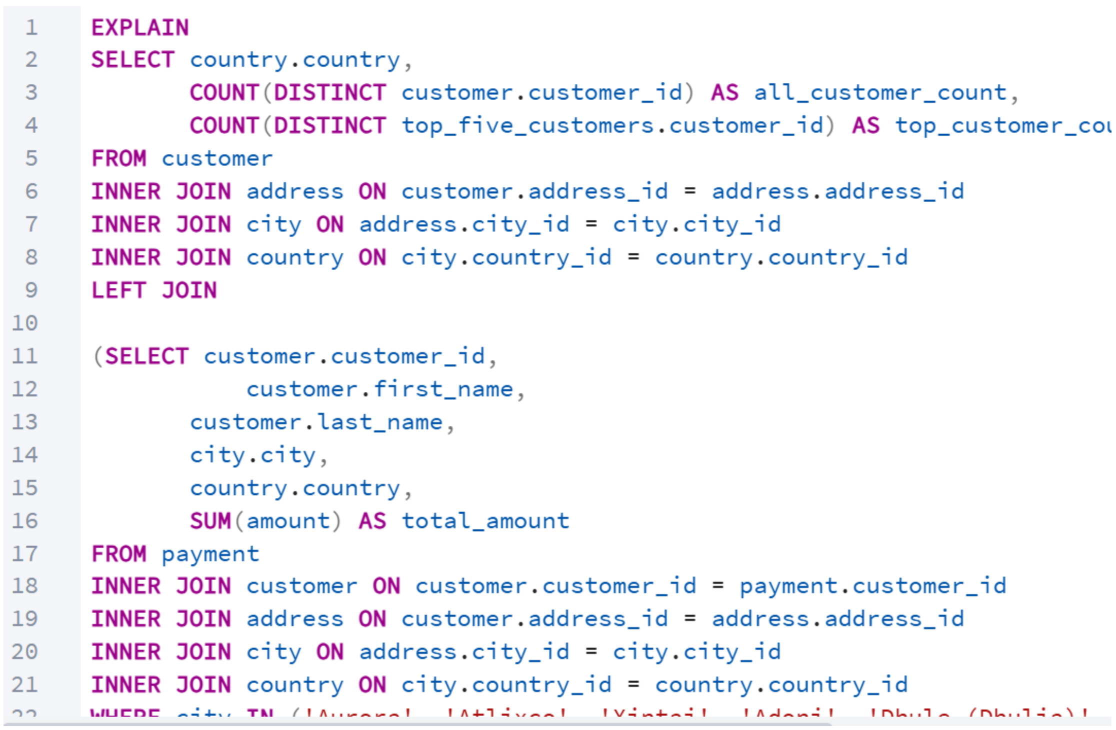


```
EXPLAIN
WITH top_five_cte (customer_id, first_name, last_name, country) AS
(SELECT customer.customer_id,
              first_name,
         last_name,
         country,
         city,
         SUM(amount) AS total_amount
FROM payment
INNER JOIN customer ON payment.customer_id = customer.customer_id
INNER JOIN address ON customer.address_id = address.address_id
INNER JOIN city ON address.city_id = city.city_id
INNER JOIN country ON city.country_id = country.country_id
WHERE city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)',
                          'Celaya', 'So Leopoldo')
AND country IN ('India', 'China', 'United States', 'Japan', 'Mexico', 'Br
                             'Philippines', 'Turkey', 'Indonesia')
GROUP BY customer.customer_id, first_name, last_name, city, country
ORDER BY total_amount DESC
LIMIT 5)
```

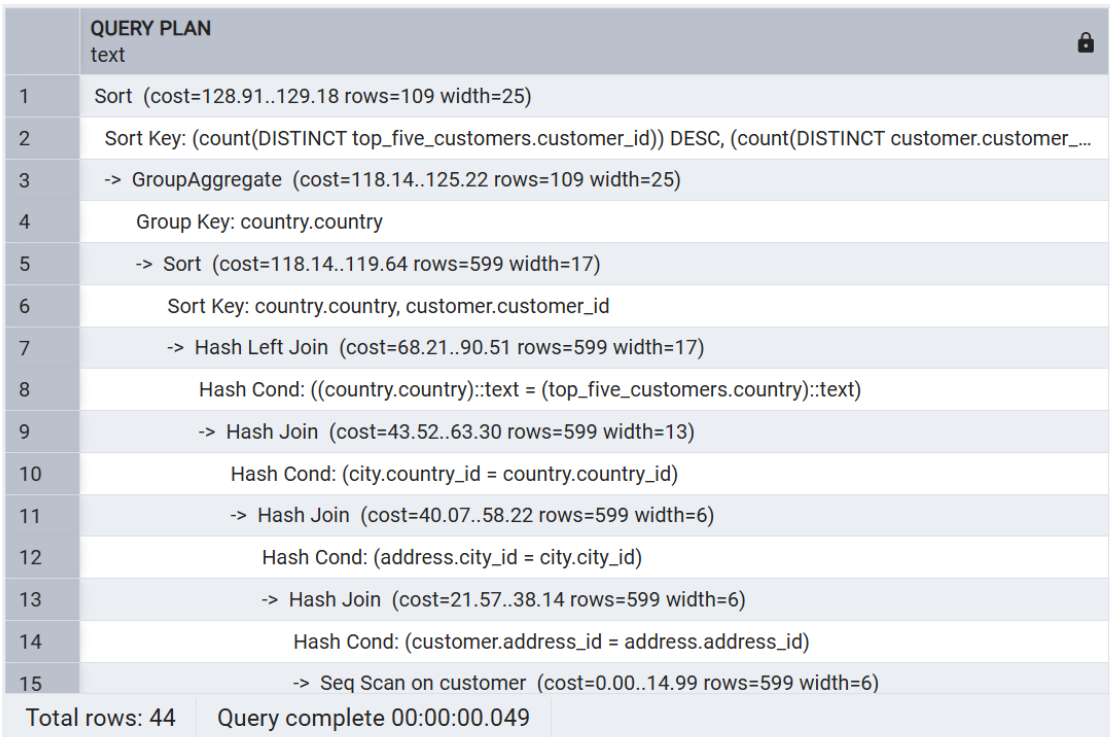


```
QUERY PLAN
text
Aggregate  (cost=24.64..24.65 rows=1 width=32)
  ->  Limit  (cost=24.56..24.57 rows=5 width=67)
        ->  Sort  (cost=24.56..24.62 rows=22 width=67)
              Sort Key: (sum(payment.amount)) DESC
              ->  HashAggregate  (cost=23.92..24.20 rows=22 width=67)
                    Group Key: customer.customer_id, city.city, country.country
                    ->  Nested Loop  (cost=3.65..23.70 rows=22 width=41)
                          ->  Nested Loop  (cost=3.36..21.93 rows=1 width=35)
                                ->  Nested Loop  (cost=3.08..21.54 rows=1 width=22)
                                      ->  Hash Join  (cost=2.81..16.84 rows=1 width=22)
                                            Hash Cond: (city.country_id = country.country_id)
                                            ->  Seq Scan on city  (cost=0.03..14.03 rows=10 width=15)
                                                  Filter: ((city)::text = ANY ('{Aurora,Atlixco,Xintai,Adoni,"Dhule (Dhulia)",Kurashiki,Pingxiang,Sivas,Celaya,"So...
                                            ->  Hash  (cost=2.66..2.66 rows=10 width=13)
                                                  ->  Seq Scan on country  (cost=0.03..2.66 rows=10 width=13)
Total rows: 22    Query complete 00:00:00.052    CR
```

Subquery 2:

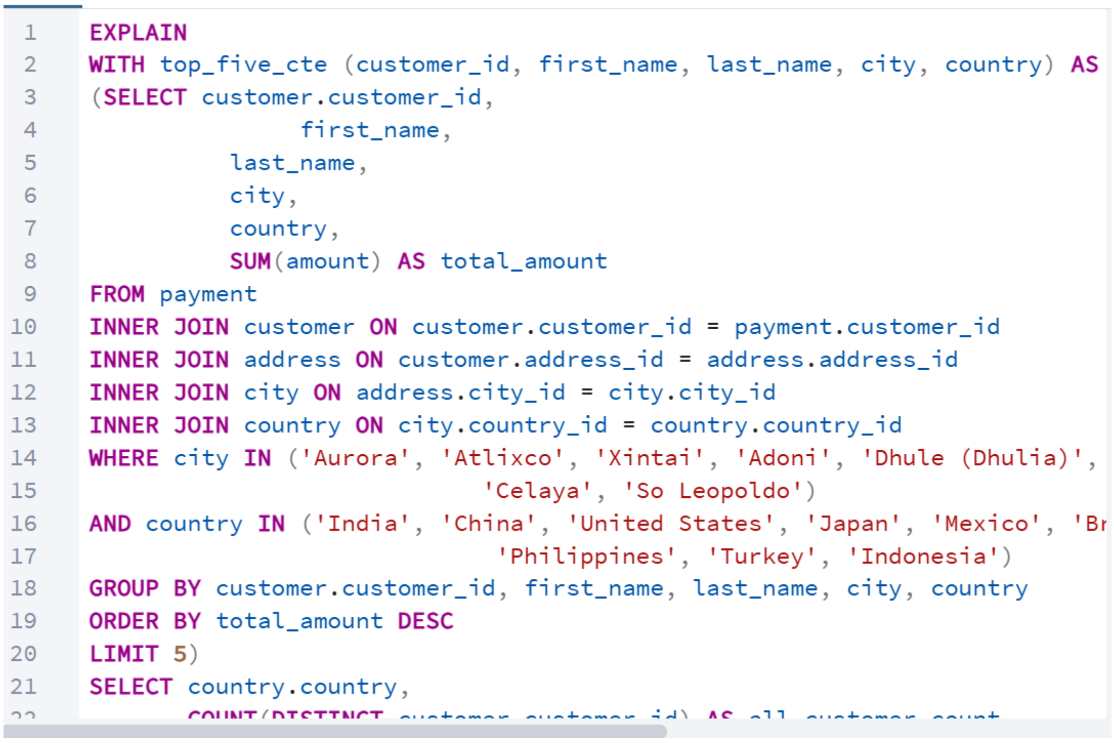


```sql
EXPLAIN
SELECT country.country,
       COUNT(DISTINCT customer.customer_id) AS all_customer_count,
       COUNT(DISTINCT top_five_customers.customer_id) AS top_customer_cou
FROM customer
INNER JOIN address ON customer.address_id = address.address_id
INNER JOIN city ON address.city_id = city.city_id
INNER JOIN country ON city.country_id = country.country_id
LEFT JOIN

(SELECT customer.customer_id,
           customer.first_name,
       customer.last_name,
       city.city,
       country.country,
       SUM(amount) AS total_amount
FROM payment
INNER JOIN customer ON customer.customer_id = payment.customer_id
INNER JOIN address ON customer.address_id = address.address_id
INNER JOIN city ON address.city_id = city.city_id
INNER JOIN country ON city.country_id = country.country_id
WHERE city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)'
```

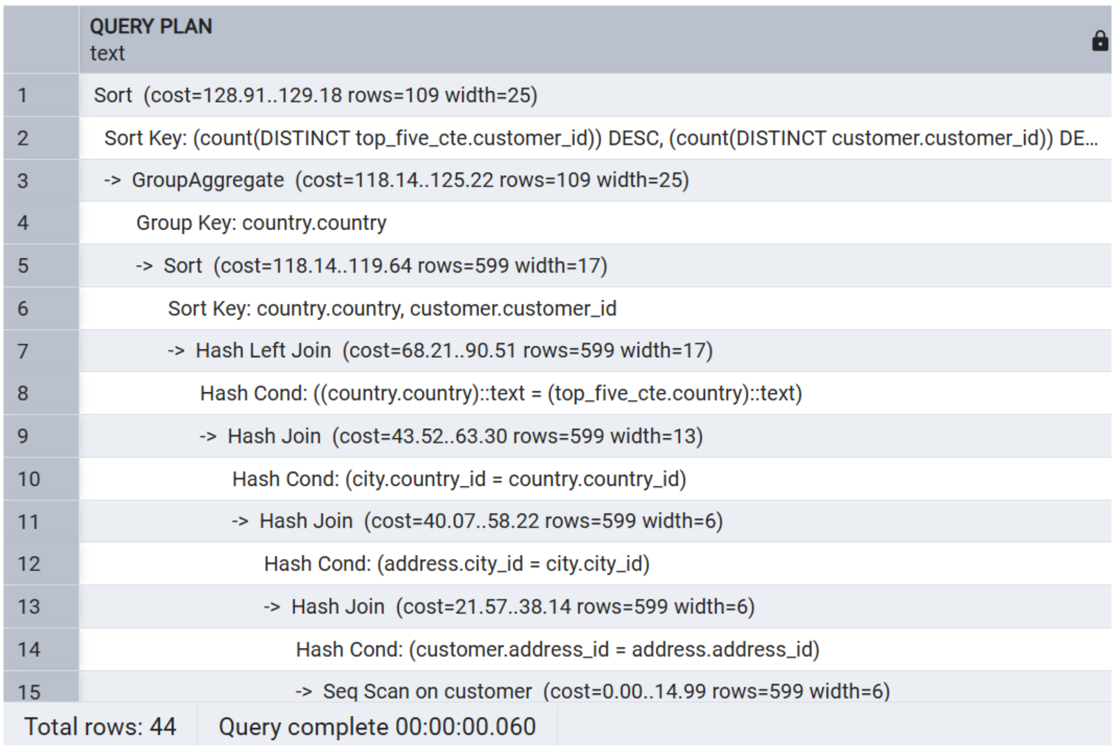


```
QUERY PLAN
text
Sort  (cost=128.91..129.18 rows=109 width=25)
 Sort Key: (count(DISTINCT top_five_customers.customer_id)) DESC, (count(DISTINCT customer.customer_...
 -> GroupAggregate  (cost=118.14..125.22 rows=109 width=25)
    Group Key: country.country
    -> Sort  (cost=118.14..119.64 rows=599 width=17)
       Sort Key: country.country, customer.customer_id
       -> Hash Left Join  (cost=68.21..90.51 rows=599 width=17)
          Hash Cond: ((country.country)::text = (top_five_customers.country)::text)
          -> Hash Join  (cost=43.52..63.30 rows=599 width=13)
             Hash Cond: (city.country_id = country.country_id)
             -> Hash Join  (cost=40.07..58.22 rows=599 width=6)
                Hash Cond: (address.city_id = city.city_id)
                -> Hash Join  (cost=21.57..38.14 rows=599 width=6)
                   Hash Cond: (customer.address_id = address.address_id)
                   -> Seq Scan on customer  (cost=0.00..14.99 rows=599 width=6)
Total rows: 44    Query complete 00:00:00.049
```

CTE2:



```sql
EXPLAIN
WITH top_five_cte (customer_id, first_name, last_name, city, country) AS
(SELECT customer.customer_id,
               first_name,
          last_name,
          city,
          country,
          SUM(amount) AS total_amount
FROM payment
INNER JOIN customer ON customer.customer_id = payment.customer_id
INNER JOIN address ON customer.address_id = address.address_id
INNER JOIN city ON address.city_id = city.city_id
INNER JOIN country ON city.country_id = country.country_id
WHERE city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)',
                            'Celaya', 'So Leopoldo')
AND country IN ('India', 'China', 'United States', 'Japan', 'Mexico', 'Br
                             'Philippines', 'Turkey', 'Indonesia')
GROUP BY customer.customer_id, first_name, last_name, city, country
ORDER BY total_amount DESC
LIMIT 5)
SELECT country.country,
       COUNT(DISTINCT customer.customer_id) AS all_customer_count
```



```
QUERY PLAN
text
1   Sort  (cost=128.91..129.18 rows=109 width=25)
2     Sort Key: (count(DISTINCT top_five_cte.customer_id)) DESC, (count(DISTINCT customer.customer_id)) DE...
3     -> GroupAggregate  (cost=118.14..125.22 rows=109 width=25)
4          Group Key: country.country
5          -> Sort  (cost=118.14..119.64 rows=599 width=17)
6               Sort Key: country.country, customer.customer_id
7               -> Hash Left Join  (cost=68.21..90.51 rows=599 width=17)
8                    Hash Cond: ((country.country)::text = (top_five_cte.country)::text)
9                    -> Hash Join  (cost=43.52..63.30 rows=599 width=13)
10                        Hash Cond: (city.country_id = country.country_id)
11                        -> Hash Join  (cost=40.07..58.22 rows=599 width=6)
12                             Hash Cond: (address.city_id = city.city_id)
13                             -> Hash Join  (cost=21.57..38.14 rows=599 width=6)
14                                  Hash Cond: (customer.address_id = address.address_id)
15                                  -> Seq Scan on customer  (cost=0.00..14.99 rows=599 width=6)
Total rows: 44    Query complete 00:00:00.060
```

For the first question, using subquery is slightly faster than CTE. But for the second question, using CTE is faster than subquery.

No clear overall winner—use whichever is clearer. But when the query is long, using CTE improves readability.

Step 3:

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

When replacing subqueries with CTEs, the main challenge is memorizing and understanding their different fixed syntactical structures. For me, the most difficult part is clarifying the logical flow between the main query and the subqueries/CTEs.