

Find the average amount paid by the top 5 customers.

The screenshot shows a PostgreSQL query editor interface. The query is as follows:

```
12 INNER JOIN address C ON B.address_id = C.address_id
13 INNER JOIN city D ON C.city_id = D.city_id
14 INNER JOIN country E ON D.country_id = E.country_id
15 WHERE D.city IN (
16     'London', 'Aurora', 'Tokat', 'Mukateve', 'Pontianak',
17     'Gatineau', 'Molodetno', 'Saint-Denis', 'Yingkou', 'Atlixco'
18 )
19 ),
20 total_amount_per_customer AS (
21     SELECT
22         customer_id,
23         first_name,
24         last_name,
25         country,
26         city,
27         SUM(amount) AS total_amount_paid
28     FROM customer_payment_data
29     GROUP BY customer_id, first_name, last_name, country, city
30 )
31 SELECT
32     AVG(total_amount_paid) AS avg_amount_paid_top_five_customers
33 FROM total_amount_per_customer;
```

The "Data Output" tab shows the result of the query:

	avg_amount_paid_top_five_customers
1	116.63916666666667

WITH customer_payment_data AS (

SELECT

A.amount,

B.customer_id,

B.first_name,

B.last_name,

C.address_id,

D.city,

E.country

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 D.city IN ( 'London', 'Aurora', 'Tokat', 'Mukateve', 'Pontianak',  
                  'Gatineau', 'Molodetno', 'Saint-Denis', 'Yingkou', 'Atlixco' )),  
total_amount_per_customer AS (  
  SELECT  
    customer_id,  
    first_name,  
    last_name,  
    country,  
    city,  
    SUM(amount) AS total_amount_paid  
  FROM customer_payment_data  
  GROUP BY customer_id, first_name, last_name, country, city)  
SELECT  
  AVG(total_amount_paid) AS avg_amount_paid_top_five_customers  
FROM total_amount_per_customer;
```

Find out how many of the top 5 customers you identified in step 1 are based within each country.

File Object Tools Edit View Window Help

Welcome Rockbuster/postgres@PostgreSQL 17* X

Rockbuster/postgres@PostgreSQL 17

No limit

Query Query History

```

26 all_customers AS (
27     SELECT
28         A.customer_id,
29         D.country
30     FROM customer A
31     INNER JOIN address B ON A.address_id = B.address_id
32     INNER JOIN city C ON B.city_id = C.city_id
33     INNER JOIN country D ON C.country_id = D.country_id
34 )
35 SELECT
36     all_customers.country,
37     COUNT(DISTINCT top_5_customers.customer_id) AS top_customer_count,
38     COUNT(DISTINCT all_customers.customer_id) AS all_customer_count
39 FROM all_customers all_customers
40 LEFT JOIN top_5_customers top_5_customers ON all_customers.country = top_5_customers.country
41 GROUP BY all_customers.country
42 ORDER BY top_customer_count DESC;

```

Data Output Messages Notifications

Showing

	country character varying (50)	top_customer_count bigint	all_customer_count bigint
1	Runion	1	1
2	Belarus	1	2
3	Turkey	1	15
4	Indonesia	1	14
5	Mexico	1	30
6	Argentina	0	13
7	Armenia	0	1

```

WITH customer_payment_data AS (

SELECT

    B.customer_id,

    B.first_name,

    B.last_name,

    E.country,

    D.city,

    SUM(A.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 D.city IN (

```

```

'London', 'Aurora', 'Tokat', 'Mukateve', 'Pontianak',
'Gatineau', 'Molodetno', 'Saint-Denis', 'Yingkou', 'Atlixco'
)
GROUP BY B.customer_id, B.first_name, B.last_name, E.country, D.city
),
top_5_customers AS (
SELECT *
FROM customer_payment_data
ORDER BY total_amount_paid DESC
LIMIT 5
),
all_customers AS (
SELECT
A.customer_id,
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
)
SELECT
all_customers.country,
COUNT(DISTINCT top_5_customers.customer_id) AS top_customer_count,
COUNT(DISTINCT all_customers.customer_id) AS all_customer_count
FROM all_customers all_customers
LEFT JOIN top_5_customers top_5_customers ON all_customers.country = top_5_customers.country
GROUP BY all_customers.country
ORDER BY top_customer_count DESC;

```

One of the main challenges I faced was understanding how to logically break down the original query into modular, named blocks. Subqueries often work seamlessly within a query, but when converting them into CTEs, I had to ensure that each step produced the necessary intermediate data to be used in the next step. Another challenge was ensuring that the CTEs executed in the correct order and maintained the original query's logic.