

Performing Subqueries.

Step 1: Find the average amount paid by the top 5 customers

```
SELECT AVG(total_payment) AS total_amount_paid
FROM (
    SELECT B.customer_id,
           B.first_name,
           B.last_name,
           D.city,
           E.country,
           SUM(A.amount) AS total_payment
    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

        (SELECT D.city
         FROM customer B
              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 E.country IN

              (SELECT E.country
               FROM customer B
                    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
               GROUP BY E.country
               ORDER BY COUNT(B.customer_id) DESC
               LIMIT 10)

        )
    GROUP BY E.country, D.city
    ORDER BY COUNT(B.customer_id) DESC
    LIMIT 10)
GROUP BY E.country, D.city, B.customer_id, B.first_name, B.last_name
ORDER BY SUM(A.amount) DESC
LIMIT 5)
```

Object Explorer

- customer
 - Columns (10)
 - customer_id
 - store_id
 - first_name
 - last_name
 - email
 - address_id
 - activebool
 - create_date
 - last_update
 - active
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
- employees
- film
 - film_actor
 - film_category
 - films
 - inventory
 - language
- payment
 - Columns (6)
 - payment_id
 - customer_id
 - staff_id
 - rental_id
 - amount
 - payment_date
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers

Dashboard x Properties x SQL x Statistics x Dependencies x Dependents x Processes x Untitled.sql* x

Rockbuster/postgres@PostgreSQL 17

Query Query History

```
1 SELECT AVG(total_payment) AS total_amount_paid
2 FROM (
3 SELECT B.customer_id,
4       B.first_name,
5       B.last_name,
6       D.city,
7       E.country,
8       SUM(A.amount) AS total_payment
9 FROM payment A
10 INNER JOIN customer B ON A.customer_id = B.customer_id
11 INNER JOIN address C ON B.address_id = C.address_id
12 INNER JOIN city D ON C.city_id = D.city_id
13 INNER JOIN country E ON D.country_id = E.country_id
14 WHERE D.city IN
15
16       (SELECT D.city
17        FROM customer B
18         INNER JOIN address C ON B.address_id = C.address_id
19         INNER JOIN city D ON C.city_id = D.city_id
20         INNER JOIN country E ON D.country_id = E.country_id
21        WHERE E.country IN
22
23       (SELECT E.country
24        FROM customer B
25         INNER JOIN address C ON B.address_id = C.address_id
26         INNER JOIN city D ON C.city_id = D.city_id
27         INNER JOIN country E ON D.country_id = E.country_id
28        GROUP BY E.country
29        ORDER BY COUNT(B.customer_id) DESC
```

Scratch Pad x

```
SELECT B.customer_id,
       B.first_name,
       B.last_name,
       D.city,
       E.country,
       SUM(A.amount) AS
total_payment
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
       (SELECT D.city
        FROM customer B
         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 E.country
IN
       (SELECT
```

Data Output Messages Notifications

	total_amount_paid
1	105.554000000000000000

Showing rows: 1 to 1 Page No: 1 of 1

Total rows: 1 Query complete 00:00:00.080 LF Ln 42, Col 27

Output 3.8.S1.1

total_amount_paid
105.554000000000000000

Step 2: Find out how many of the top 5 customers you identified in step 1 are based within each country.

```
SELECT E.country, COUNT (DISTINCT B.customer_id) AS all_customer_count,
       COUNT(DISTINCT top_5_customer.country) AS top_customer_count
FROM customer B
      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
LEFT JOIN
  (SELECT B.customer_id,
        B.first_name,
        B.last_name,
        D.city,
        E.country,
        SUM(A.amount) AS total_payment
   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

  (SELECT D.city
   FROM customer B
        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 E.country IN

    (SELECT E.country
     FROM customer B
          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
     GROUP BY E.country
     ORDER BY COUNT(B.customer_id) DESC
     LIMIT 10)

  )
  GROUP BY
    E.country,
    D.city
  ORDER BY COUNT(B.customer_id) DESC
  LIMIT 10)
  GROUP BY
    E.country,
    D.city,
    B.customer_id,
    B.first_name,
    B.last_name
  ORDER BY SUM(A.amount) DESC
  LIMIT 5) AS top_5_customer ON top_5_customer.country = E.country
GROUP BY E.country
ORDER BY all_customer_count DESC
LIMIT 5;
```

Object Explorer

- Rules
- Triggers
- Columns (3)
 - country_id
 - country
 - last_update
- Constraints
- Indexes
- RLS Policies
- Rules
- Triggers
- customer
 - Columns (10)
 - customer_id
 - store_id
 - first_name
 - last_name
 - email
 - address_id
 - activebool
 - create_date
 - last_update
 - active
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
- employees
- film
- film_actor
- film_category
- films
- inventory
- language

Dashboard x Properties x SQL x Statistics x Dependencies x Dependents x Processes x Untitled.sql x

Rockbuster/postgres@PostgreSQL 17

Query Query History

```

1 SELECT E.country, COUNT (DISTINCT B.customer_id) AS all_customer_count,
2 COUNT (DISTINCT top_5_customer.country) AS top_customer_count
3 FROM customer B
4 INNER JOIN address C ON B.address_id = C.address_id
5 INNER JOIN city D ON C.city_id = D.city_id
6 INNER JOIN country E ON D.country_id = E.country_id
7 LEFT JOIN
8 (SELECT B.customer_id,
9 B.first_name,
10 B.last_name,
11 D.city,
12 E.country,
13 SUM(A.amount) AS total_payment
14 FROM payment A
15 INNER JOIN customer B ON A.customer_id = B.customer_id
16 INNER JOIN address C ON B.address_id = C.address_id
17 INNER JOIN city D ON C.city_id = D.city_id
18 INNER JOIN country E ON D.country_id = E.country_id
19 WHERE D.city IN
20
21 (SELECT D.city
22 FROM customer B
23 INNER JOIN address C ON B.address_id = C.address_id
24 INNER JOIN city D ON C.city_id = D.city_id
25 INNER JOIN country E ON D.country_id = E.country_id
26 WHERE E.country IN
27
28 (SELECT E.country
29 FROM customer B

```

Scratch Pad

```

SELECT B.customer_id,
B.first_name,
B.last_name,
D.city,
E.country,
SUM(A.amount) AS
total_payment
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
(
SELECT D.city
FROM customer B
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 E.country
IN
(
SELECT

```

Data Output Messages Notifications

Showing rows: 1 to 5 Page No: 1 of 1

country	all_customer_count	top_customer_count
India	60	1
China	53	1
United States	36	1

Total rows: 5 Query complete 00:00:00.082 LF Ln 28, Col 52

Output A.3.8.S.2

country	all_customer_count	top_customer_count
India	60	1
China	53	1
United States	36	1
Japan	31	1
Mexico	30	1

Step 3:

1. Write 1 to 2 short paragraphs on the following:

When we need to isolate steps with filtering and aggregating data, correlate values row by row and encapsulate results with temporary tables for reuse.

I think it made easier to this case because of following a way of getting different specific answers to different questions in the same common root.

Probably if we would need more general answers or we were making exploratory analysis, we could have come to the same with a wider view.

This saved us time, and allowed us to go straight to the point.