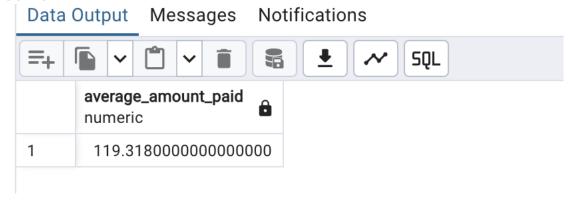
3.8: Performing Subqueries

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

```
SELECT
 AVG(total_amount_paid.total_amount_paid) AS average_amount_paid
FROM (
 -- Subquery to find the top 5 customers and their total payments
 WITH TopCities AS (
   SELECT
     city.city id
   FROM
     customer
   JOIN
     address ON customer.address id = address.address id
   JOIN
     city ON address.city_id = city.city_id
   JOIN
     country ON city.country_id = country.country_id
   GROUP BY
     city.city_id
   ORDER BY
     COUNT(customer.customer_id) DESC
   LIMIT 10
 )
 SELECT
   customer.customer_id,
   customer.first name,
   customer.last name,
   country.country,
   city.city,
   SUM(payment.amount) AS total_amount_paid
 FROM
   customer
 JOIN
   address ON customer.address_id = address.address_id
 JOIN
   city ON address.city_id = city.city_id
 JOIN
   country ON city.country_id = country.country_id
 JOIN
   rental ON customer.customer_id = rental.customer_id
 JOIN
   payment ON rental.rental_id = payment.rental_id
 WHERE
   city.city id IN (SELECT city id FROM TopCities)
 GROUP BY
```

```
customer.customer_id, customer.first_name, customer.last_name, country.country, city.city
ORDER BY
total_amount_paid DESC
LIMIT 5
) AS total_amount_paid;
```

OUTPUT



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

```
WITH TopCities AS (
 -- Subquery to find the top 10 cities by customer count
 SELECT
   city.city_id
 FROM
   customer
 JOIN
   address ON customer.address_id = address.address_id
 JOIN
   city ON address.city_id = city.city_id
 JOIN
   country ON city.country_id = country.country_id
 GROUP BY
   city.city_id
 ORDER BY
   COUNT(customer.customer_id) DESC
 LIMIT 10
),
TopCustomers AS (
 -- Subquery to find the top 5 customers by total amount paid
 SELECT
   customer.customer_id,
   country.country
 FROM
   customer
```

```
JOIN
        address ON customer.address_id = address.address_id
      JOIN
        city ON address.city_id = city.city_id
      JOIN
        country ON city.country_id = country.country_id
      JOIN
        rental ON customer.customer id = rental.customer id
      JOIN
        payment ON rental.rental_id = payment.rental_id
      WHERE
        city.city_id IN (SELECT city_id FROM TopCities)
      GROUP BY
        customer.customer id, country.country
      ORDER BY
        SUM(payment.amount) DESC
      LIMIT 5
     )
     -- Final query to count all customers and top customers by country
     SELECT
      country.country,
      COUNT(customer.customer_id) AS all_customer_count,
      COUNT(top_customers.customer_id) AS top_customer_count
     FROM
      customer
     JOIN
      address ON customer.address_id = address.address_id
     JOIN
      city ON address.city_id = city.city_id
     JOIN
      country ON city.country_id = country.country_id
     LEFT JOIN
      TopCustomers
                         AS
                                top_customers
                                                   ON
                                                           customer.customer_id
     top_customers.customer_id
     GROUP BY
      country.country
     ORDER BY
all_customer_count DESC;
```

OUTPUT

Data	Data Output 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	0	
4	Japan	31	0	
5	Mexico	30	0	
6	Brazil	28	0	
7	Russian Federation	28	0	
8	Philippines	20	0	
9	Turkey	15	0	
10	Indonesia	14	0	
11	Argentina	13	0	
12	Nigeria	13	0	
13	South Africa	11	0	
14	Taiwan	10	0	
Tota	l rows: 108 of 108 Query complete	e 00:00:00.161 Ln 3	335, Col 29	

Step 3. Do you think steps 1 and 2 could be done without using subqueries?

Steps 1 and 2 could technically be done without using subqueries, but it would be more complex and less readable. For example, instead of using subqueries, we could perform multiple joins and apply conditions to filter the top cities and top customers in a single query. However, this approach would make the query harder to maintain, as we would have to repeat logic like filtering for the top 10 cities and top 5 customers multiple times. Subqueries help simplify these tasks by isolating these smaller operations, making the code cleaner and easier to understand.

When do you think subqueries are useful?

Subqueries are especially useful when you need to break down a complex task into smaller, more manageable parts or when the result of one query is needed to filter or drive another. They are valuable when performing aggregations like rankings (e.g., finding the top 5 customers), filtering based on grouped data (e.g., top cities), or when joining tables multiple times with specific conditions. Subqueries also make the main query more readable by abstracting away intermediate steps.