

SQL Exercise 8: subqueries

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

Query Query History

```
1  SELECT AVG (total_payment_amount) AS avg_paid_top5
2  FROM
3  (SELECT customer.customer_id,
4      customer.first_name,
5      customer.last_name,
6      city.city,
7      country.country,
8      SUM(payment.amount) AS total_payment_amount
9  FROM payment
10 INNER JOIN customer ON payment.customer_id = customer.customer_id
11 INNER JOIN address ON customer.address_id = address.address_id
12 INNER JOIN city ON address.city_id = city.city_id
13 INNER JOIN country ON city.country_id = country.country_id
14 WHERE city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni',
15                 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang',
16                 'Sivas', 'Celaya', 'So Leopoldo')
17     AND country IN ('India', 'China', 'United States', 'Japan',
18                      'Mexico', 'Brazil', 'Russian Federation',
19                      'Philippines', 'Turkey', 'Indonesia')
20 GROUP BY customer.customer_id, customer.first_name, customer.last_name,
21          city.city, country.country
22 ORDER BY total_payment_amount DESC
23 LIMIT 5) AS avg_amount_paid
```

Data Output Messages Notifications

avg_paid_top5
numeric

	avg_paid_top5
1	107.35400000000000

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

```

Query Query History
26 SELECT country.country,
27     COUNT(DISTINCT customer.customer_id) AS all_customer_count,
28     COUNT(DISTINCT top_5_customers.customer_id) AS top_customer_count
29 FROM customer
30 INNER JOIN address ON customer.address_id = address.address_id
31 INNER JOIN city ON address.city_id = city.city_id
32 INNER JOIN country ON city.country_id = country.country_id
33 LEFT JOIN (
34     SELECT customer.customer_id, country.country
35     FROM payment
36     INNER JOIN customer ON payment.customer_id = customer.customer_id
37     INNER JOIN address ON customer.address_id = address.address_id
38     INNER JOIN city ON address.city_id = city.city_id
39     INNER JOIN country ON city.country_id = country.country_id
40     WHERE city.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang',
41                     'Sivas', 'Celaya', 'So Leopoldo')
42     AND country.country IN ('India', 'China', 'United States', 'Japan', 'Mexico', 'Brazil', 'Russian Federation',
43                     'Philippines', 'Turkey', 'Indonesia')
44     GROUP BY customer.customer_id, country.country
45     ORDER BY SUM(payment.amount) DESC
46     LIMIT 5
47 ) AS top_5_customers
48     ON top_5_customers.country = country.country
49     GROUP BY country.country
50     HAVING COUNT(DISTINCT top_5_customers.customer_id) >= 0
51     ORDER BY top_customer_count DESC, all_customer_count DESC LIMIT 5;

```

Data Output Messages Notifications

The screenshot shows a database query results interface. At the top, there are several icons for file operations (New, Open, Save, Print, Copy, Paste, Find, Refresh, SQL). To the right of the icons is a "Show" button. Below the toolbar is a table with 5 rows of data. The table has four columns: "country" (character varying(50)), "all_customer_count" (bigint), "top_customer_count" (bigint), and an unnamed fourth column. The data rows are: 1. Mexico (30, 2), 2. India (60, 1), 3. United States (36, 1), 4. Turkey (15, 1), 5. China (53, 0). At the bottom of the table area, it says "Total rows: 5" and "Query complete 00:00:00.134".

	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	

Step 3:

1. Write 1 to 2 short paragraphs on the following:
 - a. Do you think steps 1 and 2 could be done without using subqueries?

Well, by searching and reviewing, I assume that steps 1 and 2 could be done without subqueries by using multiple JOINS, filtering, or grouping directly in a single query.

For example, we could join all the relevant tables, group the data by customer or country, and then use ORDER BY and LIMIT to identify the top customers before counting them by country. However, this approach would make the query harder to read and maintain because the logic for identifying the top 5 customers and the logic for counting them would be mixed together in one long query.

b. When do you think subqueries are useful?

I think they are useful when we want to break a problem into smaller, more understandable parts. They let us create an intermediate result set—like the list of top 5 customers; that can be reused in a larger query without repeating the same calculations. They also make it easier to apply limits or filters to a specific part of data before joining it to other tables. This way our main query will be simpler and more readable.