## 3.8: Performing Subqueries

# Innocent Bayai

### 13 June 2024

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

Copy the query you wrote in step 3 of the task from Exercise 3.7: Joining Tables of Data into the Query Tool. This will be your subquery, so give it an alias, "total\_amount\_paid," and add parentheses around it.

Write an outer statement to calculate the average amount paid.

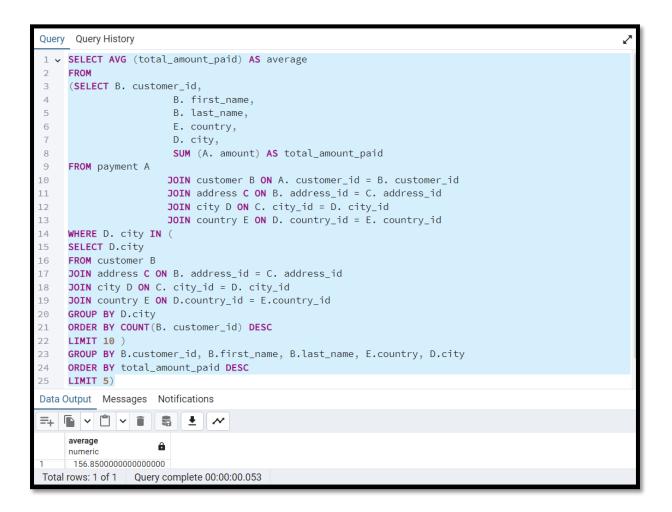
Add your subquery to the outer statement. It will go in either the SELECT, WHERE, or FROM clause. (Hint: When referring to the subquery in your outer statement, make sure to use the subquery's alias, "total\_amount\_paid".)

If you've done everything correctly, pgAdmin 4 will require you to add an alias after the subquery. Go ahead and call it "average".

Copy-paste your queries and the final data output from pgAdmin 4 into your answers document.

```
SELECT AVG (total_amount_paid) AS average
```

```
FROM
(SELECT B. customer_id,
        B. first_name,
        B. last_name,
        E. country,
        D. city,
        SUM (A. amount) AS total_amount_paid
FROM payment A
       JOIN customer B ON A. customer_id = B. customer_id
       JOIN address C ON B. address_id = C. address_id
       JOIN city D ON C. city_id = D. city_id
       JOIN country E ON D. country_id = E. country_id
WHERE
D. city IN (
SELECT D.city
FROM customer B
JOIN address C ON B. address_id = C. address_id
JOIN city D ON C. city_id = D. city_id
JOIN country E ON D.country_id = E.country_id
GROUP BY D.city
ORDER BY COUNT(B. customer_id) DESC
LIMIT 10
)
GROUP BY B.customer_id, B.first_name, B.last_name, E.country, D.city
ORDER BY total_amount_paid DESC
LIMIT 5)
```



NB: to check if my output is correct, I manually calculated the mean for the total payments made by top 5 customers from Ex. 3.7 and confirmed that it is 156.8

**Step 2:** Find out how many of the top 5 customers you identified in step 1 are based within each country. Your final output should include 3 columns:

"country", "all\_customer\_count" with the total number of customers in each country

"top\_customer\_count" showing how many of the top 5 customers live in each country

You'll notice that this step is quite difficult. We've broken down each part and provided you with some helpful hints:

- 1. Copy the query from step 3 of task 3.7 into the Query Tool and add parentheses around it. This will be your inner query.
- 2. Write an outer statement that counts the number of customers living in each country. You'll need to refer to your entity relationship diagram or data dictionary in order to do this. The information you need is in different tables, so you'll have to use a JOIN. To get the count for each country, use COUNT(DISTINCT) and GROUP BY. Give your second column the alias "all\_customer\_count" for readability.
- 3. Place your inner query in the outer query. Since you want to merge the entire output of the outer query with the information from your inner query, use a left join to connect the

- two queries on the "country" column. You'll need to add a LEFT JOIN after your outer query, followed by the subquery in parentheses.
- 4. Give your subquery an alias so you can refer to it in your outer query, for example, "top\_5\_customers".
- 5. Remember to specify which columns to join the two tables on using ON. Both ON and the column names should follow the alias.
- 6. Count the top 5 customers for the third column using GROUP BY and COUNT (DISTINCT). Give this column the alias "top\_customer\_count".
- 7. Copy-paste your query and the data output into your "Answers 3.8" document.

```
SELECT E. country,
   COUNT (DISTINCT B. customer_id) AS all_customer_count,
   COUNT (DISTINCT top_5_customers) AS top_customer_count
FROM customer B
JOIN address C ON B. address_id = C. address_id
JOIN city D ON C. city_id = D. city_id
JOIN country E ON D. country_id = E. country_id
       JOIN
(SELECT B. customer_id,B. first_name, B. last_name, E. country, D. city,
   SUM (A. amount) AS total_amount_paid
FROM payment A
JOIN customer B ON A. customer_id = B. customer_id
JOIN address C ON B. address_id = C. address_id
JOIN city D ON C. city_id = D. city_id
JOIN country E ON D. country_id = E. country_id
WHERE D. city IN (
SELECT D.city
FROM customer B
JOIN address C ON B. address_id = C. address_id
JOIN city D ON C. city_id = D. city_id
JOIN country E ON D.country_id = E.country_id
                       IN
                              (SELECT
                                              E.country
WHERE E.country
FROM customer
JOIN
       address C
                       ON
                              B.address_id
                                                     C.address id
               D
JOIN
                       ON
       city
                              C.city_id
                                                     D.city_id
                                              =
                       ON
                                                     E.country_id
JOIN
       country E
                              D.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
               B.customer_id, B.first_name, B.last_name, E.country, D.city
ORDER BY
               SUM(A.amount) DESC
LIMIT
       5)
               AS
                      top_5_customers
ON
       B.customer_id =
                              top_5_customers. customer_id
GROUP BY
               E.country
ORDER BY
               all_customer_count
                                      DESC
LIMIT 5
```

```
Query Query History
 1 V SELECT E. country,
2
           COUNT (DISTINCT B. customer_id) AS all_customer_count,
3
           COUNT (DISTINCT top_5_customers) AS top_customer_count
     FROM customer B
4
     JOIN address C ON B. address_id = C. address_id
 5
     JOIN city D ON C. city_id = D. city_id
 6
     JOIN country E ON D. country_id = E. country_id
     LEFT JOIN
9
     (SELECT B. customer_id, B. first_name, B. last_name, E. country, D. city,
10
            SUM (A. amount) AS total_amount_paid
11
     FROM payment A
     JOIN customer B ON A. customer_id = B. customer_id
12
     JOIN address C ON B. address_id = C. address_id
13
     JOIN city D ON C. city_id = D. city_id
14
     JOIN country E ON D. country_id = E. country_id
15
16
    WHERE D. city IN (
17
    SELECT D.city
    FROM customer B
18
    JOIN address C ON B. address_id = C. address_id
19
20
    JOIN city D ON C. city_id = D. city_id
    JOIN country E ON D.country_id = E.country_id
21
     WHERE E.country IN (SELECT E.country
22
     FROM
23
            customer
                       В
           address C ON B.address_id = C.address_id
24
     JOIN
                      ON C.city_id = D.city_id
          city D
25
     JOIN
26
     JOIN
            country E ON D.country_id = E.country_id
27
     GROUP BY E.country
28
     ORDER
           BY COUNT(B.customer_id) DESC
29
     LIMIT
            10)
30
    GROUP
            BY E.country, D.city
```

```
LIMIT 10)
30
    GROUP BY E.country, D.city
31
    ORDER BY COUNT(B.customer_id)
                                     DESC
32
    LIMIT 10)
33
    GROUP
           BY B.customer_id, B.first_name, B.last_name, E.country, D.city
34
    ORDER
           BY SUM(A.amount) DESC
    LIMIT 5) AS top_5_customers
35
    ON B.customer_id = top_5_customers.customer_id
36
37
    GROUP BY E.country
    ORDER BY all_customer_count DESC
38
     LIMIT
```

Data Output Messages Notifications				
	country character varying (50)	all_customer_count bigint	top_customer_count bigint	â
1	India	60		1
2	China	53		1
3	United States	36		1
4	Japan	31		1
5	Mexico	30		1
Total rows: 5 of 5 Query complete 00:00:00.082				

## Step 3:

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

I am a fresher with respect to SQL but I think that subqueries must be used when and where they are the only way out to get output, otherwise simpler queries must be adopted. The complexity associated with steps 1 and 2 took me lots of time to figure out and I made mistakes too that took long to resolve. This makes it difficult for non-technical individuals to understand the meaning imbedded in the syntax.

However, subqueries remain vital where data is required under certain conditions. Subqueries under these circumstances allows conditioned data to be selected. Also, where data keeps changing, it is important to use subqueries to pick up to date data.