

3.8: Performing Subqueries

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

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

```
1 SELECT AVG(total_amount_paid) AS average
2 FROM
3 (SELECT B.customer_id, B.first_name, B.last_name,
4  D.city, E.country,
5  SUM(A.amount) AS Total_Amount_Paid FROM customer B
6  INNER JOIN payment A ON B.customer_id = A.customer_id INNER JOIN address C ON B.address_id = C.address_id INNE
7  INNER JOIN country E ON D.country_id = E.country_id
8  WHERE D.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang', 'Sivas',
9  GROUP BY B.customer_id, B.first_name, B.last_name, D.city, E.country ORDER BY Total_Amount_Paid DESC
10 LIMIT 5) AS total_amount_paid
11
```

The query is executed, and the results are displayed in the 'Data output' tab. The results show a single row with the average amount paid by the top 5 customers.

	average numeric
1	107.354

Step 2: Find out how many of the top 5 customers are based within each country.

The screenshot shows a PostgreSQL query editor with a query that identifies the top 5 customers by total amount paid and then counts how many of those top customers are in each country. The query uses multiple JOINs and subqueries. The results pane shows the output of the query, which lists the top 5 countries by the number of top customers.

```
1 SELECT DISTINCT(A.country),
2 COUNT(DISTINCT D.customer_id) AS all_customer_count, COUNT(DISTINCT A.country) AS top_cutomer_count
3 FROM country A
4 INNER JOIN city B
5 ON A.country_id = B.country_id
6 INNER JOIN address C
7 ON B.city_id = C.city_id
8 INNER JOIN customer D
9 ON C.address_id = D.address_id
10 LEFT JOIN (SELECT B.customer_id, B.first_name, B.last_name,
11 D.city, E.country,
12 SUM(A.amount) AS Total_Amount_Paid FROM customer B
13 INNER JOIN payment A ON B.customer_id = A.customer_id
14 INNER JOIN address C ON B.address_id = C.address_id
15 INNER JOIN city D ON C.city_id = D.city_id
16 INNER JOIN country E ON D.country_id = E.country_id
17 WHERE D.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang', 'Sivas',
18 GROUP BY B.customer_id, B.first_name, B.last_name, D.city, E.country ORDER BY Total_Amount_Paid DESC
19 LIMIT 5) AS top_5_customers
20 ON A.country=top_5_customers.COUNTRY
21 GROUP BY A.country, top_5_customers
22 ORDER BY all_customer_count DESC
```

	country character varying (50)	all_customer_count bigint	top_cutomer_count bigint
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 00:00:00.164 Ln 20, Col 37

Step 3:

- Do you think steps 1 and 2 could be done without using subqueries?

Step 1 could have been done without a subquery by using HAVING syntax along with aggregation functions. However, step 2 needed a subquery as multiples tables and columns from databases are used to pull results from different tables.

- When do you think subqueries are useful?

Subqueries are useful where the use of INNER and OUTER functions are needed as commands for complex queries to filter out the data. Using subqueries gives enables one to compare data points, rename or make new columns within a table without affecting the original database, or when analysing a complex query.