Juliana Lee

Task 3.8: Performing Subqueries

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

Inner Query (subquery):

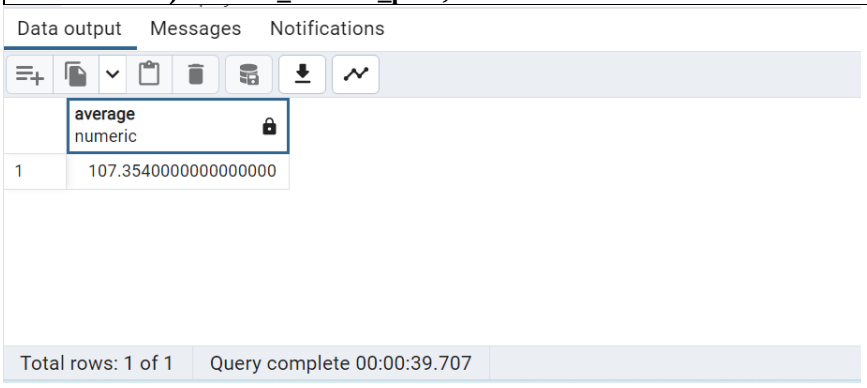
```
(SELECT A.customer_id,
A.first_name,
A.last_name,
D.city,
E.country,
SUM(B.amount) AS total_amount_paid
FROM customer A
INNER JOIN payment B on A.customer_id = B.customer_id
INNER JOIN address C on A.address_id = C.address_id
INNER JOIN city D on C.city_id = D.city_id
INNER JOIN country E on D.country_id = E.country_id
WHERE D.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang',
'Sivas', 'Celaya', 'So Leopoldo')
GROUP BY A.customer_id, A.first_name, A.last_name, D.city, E.country
ORDER BY total_amount_paid
DESC limit 5) AS total_amount_paid;
```

Outer Query:

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

Query:

```
SELECT AVG(total_amount_paid) AS average
FROM
(SELECT A.customer_id,
A.first_name,
A.last_name,
D.city,
E.country,
SUM(B.amount) AS total_amount_paid
FROM customer A
INNER JOIN payment B on A.customer_id = B.customer_id
INNER JOIN address C on A.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 ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang',
'Sivas', 'Celaya', 'So Leopoldo')
GROUP BY A.customer_id, A.first_name, A.last_name, D.city, E.country
ORDER BY total_amount_paid
DESC limit 5) AS total_amount_paid;
```



Juliana Lee

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

Inner Query (subquery):

```
(SELECT A.customer_id,
A.first_name,
A.last_name,
B.city,
E.country,
SUM(C.amount) AS total_amount_paid
FROM customer A
INNER JOIN payment C on A.customer_id = C.customer_id
INNER JOIN address D on A.address_id = D.address_id
INNER JOIN city B on D.city_id = B.city_id
INNER JOIN country E on B.country_id = E.country_id
WHERE E.country IN ('India', 'China', 'United States', 'Japan', 'Mexico', 'Brazil', 'Russian Federation',
'Philippines', 'Turkey', 'Indonesia')
AND B.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang', 'Sivas',
'Celaya', 'So Leopoldo')
GROUP BY A.customer_id, E.country, B.city
ORDER BY total_amount_paid DESC limit 5) AS top_5_customers
```

Outer Query:

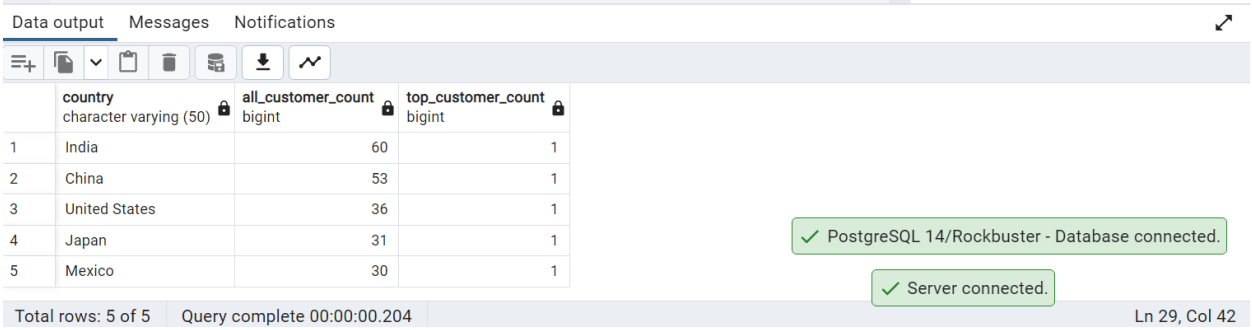
```
SELECT DISTINCT(A.country),
COUNT (DISTINCT D.customer_id) AS all_customer_count,
COUNT (DISTINCT A.country) AS top_customer_count
FROM country A
INNER JOIN city B
ON A.country_id=B.country_id
INNER JOIN address C
ON B.city_id=C.city_id
INNER JOIN customer D
ON C.address_id=D.address_id
LEFT JOIN
```

Query:

```
SELECT DISTINCT(A.country),
COUNT (DISTINCT D.customer_id) AS all_customer_count,
COUNT (DISTINCT A.country) AS top_customer_count
FROM country A
INNER JOIN city B
ON A.country_id=B.country_id
INNER JOIN address C
ON B.city_id=C.city_id
INNER JOIN customer D
ON C.address_id=D.address_id
LEFT JOIN (SELECT A.customer_id,
A.first_name,
A.last_name,
B.city,
E.country,
SUM(C.amount) AS total_amount_paid
FROM customer A
INNER JOIN payment C on A.customer_id = C.customer_id
INNER JOIN address D on A.address_id = D.address_id
INNER JOIN city B on D.city_id = B.city_id
INNER JOIN country E on B.country_id = E.country_id
```

Juliana Lee

```
WHERE E.country IN ('India', 'China', 'United States', 'Japan', 'Mexico', 'Brazil', 'Russian Federation',
'Philippines', 'Turkey', 'Indonesia')
AND B.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang', 'Sivas',
'Celaya', 'So Leopoldo')
GROUP BY A.customer_id, E.country, B.city
ORDER BY total_amount_paid DESC limit 5) AS top_5_customers
ON A.country=top_5_customers.COUNTRY
GROUP BY A.country, top_5_customers
ORDER BY all_customer_count DESC
LIMIT 5;
```



Step 3: Step 1 could be done without using subqueries. Instead, I could have used HAVE along with the aggregation functions. However, step 2 requires subqueries since I am using multiple columns from multiple tables. Subqueries are useful when you have to filter data that is constantly changing and need to analyze results of complex queries.