Directions

Create a new text document and call it “Answers 3.8.” You’ll be copy-pasting your queries, outputs, and written answers into this document, as you’ve done in previous tasks.

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

1. **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.**

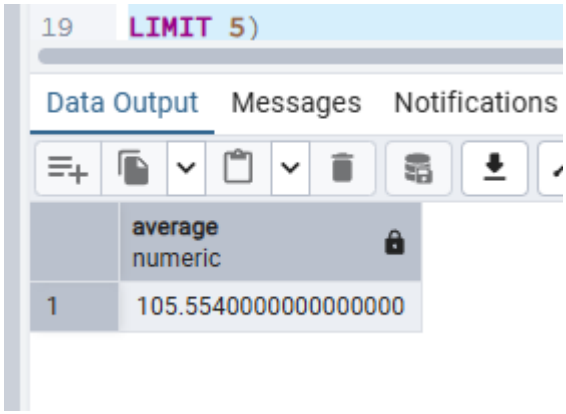
```
(
SELECT A. customer_id,
A. first_name,
A.last_name,
C. city,
D. country,
SUM(F.amount) AS total_spend
FROM customer A
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
INNER JOIN rental E ON A.customer_id = E.customer_id
INNER JOIN payment F ON E.rental_id = F.rental_id
WHERE city = 'Aurora' OR city = 'Acua' OR city = 'Citrus Heights' OR city ='Iwaki' OR city = 'Ambattur' OR city = 'Shanwei' OR city = 'So Leopoldo' OR city = 'Teboksary' OR city = 'Tianjin' OR city = 'Cianjur'
GROUP BY A. customer_id, A. first_name, A.last_name, C. city, D. country
ORDER BY total_spend DESC
LIMIT 5)
AS total_amount_paid
```

2. **Write an outer statement to calculate the average amount paid**

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

3. **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".)**
4. **If you've done everything correctly, pgAdmin 4 will require you to add an alias after the subquery. Go ahead and call it "average".**
5. **Copy-paste your queries and the final data output from pgAdmin 4 into your answers document.**

```
SELECT AVG(total_spend) AS average

FROM

(

SELECT A. customer_id,

A. first_name,

A.last_name,

C. city,

D. country,

SUM(F.amount) AS total_spend

FROM customer A

INNER JOIN address B ON A.address_id = B.address_id

INNER JOIN city C ON B.city_id = C.city_id

INNER JOIN country D ON C.country_id = D.country_id

INNER JOIN rental E ON A.customer_id = E.customer_id

INNER JOIN payment F ON E.rental_id = F.rental_id

WHERE city = 'Aurora' OR city = 'Acua' OR city = 'Citrus Heights' OR city ='Iwaki' OR city = 'Ambattur' OR city = 'Shanwei' OR city = 'So Leopoldo' OR city = 'Teboksary' OR city = 'Tianjin' OR city = 'Cianjur'

GROUP BY A. customer_id, A. first_name, A.last_name, C. city, D. country

ORDER BY total_spend DESC

LIMIT 5)

AS total_amount_paid
```


19 LIMIT 5)
Data Output
Messages
Notifications
average
numeric
1
105.5540000000000000

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.**

```
(

SELECT A. customer_id,

A. first_name,

A.last_name,

C. city,

D. country,

SUM(F.amount) AS total_spend

FROM customer A

INNER JOIN address B ON A.address_id = B.address_id

INNER JOIN city C ON B.city_id = C.city_id

INNER JOIN country D ON C.country_id = D.country_id

INNER JOIN rental E ON A.customer_id = E.customer_id

INNER JOIN payment F ON E.rental_id = F.rental_id

WHERE city = 'Aurora' OR city = 'Acua' OR city = 'Citrus Heights' OR city ='Iwaki' OR city = 'Ambattur' OR city = 'Shanwei' OR city = 'So Leopoldo' OR city = 'Teboksary' OR city = 'Tianjin' OR city = 'Cianjur'

GROUP BY A. customer_id, A. first_name, A.last_name, C. city, D. country

ORDER BY total_spend DESC

LIMIT 5)

AS total_amount_paid
```

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.**

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

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 D.country,
   COUNT(DISTINCT(A.customer_id)) AS all_customer_count FROM customer A
COUNT(DISTINCT(A.customer_id)) AS top_customer_count FROM total_amount_paid
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
GROUP BY D.country
ORDER BY all_customer_count DESC
LEFT JOIN D.country = D.country
    (
    SELECT A. customer_id,
    A. first_name,
    A.last_name,
    C. city,
    D. country,
    SUM(F.amount) AS total_spend
    FROM customer A
    INNER JOIN address B ON A.address_id = B.address_id
    INNER JOIN city C ON B.city_id = C.city_id
    INNER JOIN country D ON C.country_id = D.country_id
    INNER JOIN rental E ON A.customer_id = E.customer_id
```

INNER JOIN payment F ON E.rental_id = F.rental_id

WHERE city = 'Aurora' OR city = 'Acua' OR city = 'Citrus Heights' OR city ='Iwaki' OR city = 'Ambattur' OR city = 'Shanwei' OR city = 'So Leopoldo' OR city = 'Teboksary' OR city = 'Tianjin' OR city = 'Cianjur'

GROUP BY A. customer_id, A. first_name, A.last_name, C. city, D. country

ORDER BY total_spend DESC

LIMIT 5)

AS total_amount_paid

THIS IS what I created. I knew the LEFT JOIN syntax wasn't right!

This is what Co-pilot helped me with:

```
SELECT
   D.country,
   COUNT(DISTINCT A.customer_id) AS all_customer_count,
   COUNT(DISTINCT T.customer_id) AS top_customer_count
FROM customer A
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
LEFT JOIN (
   SELECT A.customer_id, D.country
   FROM customer A
   INNER JOIN address B ON A.address_id = B.address_id
   INNER JOIN city C ON B.city_id = C.city_id
   INNER JOIN country D ON C.country_id = D.country_id
   INNER JOIN rental E ON A.customer_id = E.customer_id
   INNER JOIN payment F ON E.rental_id = F.rental_id
   WHERE city IN ('Aurora', 'Acua', 'Citrus Heights', 'Iwaki', 'Ambattur',
         'Shanwei', 'So Leopoldo', 'Teboksary', 'Tianjin', 'Cianjur')
   GROUP BY A.customer_id, D.country
   ORDER BY SUM(F.amount) DESC
   LIMIT 5
) AS T ON A.customer_id = T.customer_id
GROUP BY D.country
ORDER BY all_customer_count DESC;
```

```
SELECT
  D.country,
  COUNT(DISTINCT A.customer_id) AS all_customer_count,
  COUNT(DISTINCT T.customer_id) AS top_customer_count
FROM customer A
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
LEFT JOIN (
  SELECT A.customer_id, D.country
  FROM customer A
  INNER JOIN address B ON A.address_id = B.address_id
  INNER JOIN city C ON B.city_id = C.city_id
  INNER JOIN country D ON C.country_id = D.country_id
  INNER JOIN rental E ON A.customer_id = E.customer_id
  INNER JOIN payment F ON E.rental_id = F.rental_id
  WHERE city IN ('Aurora', 'Acua', 'Citrus Heights', 'Iwaki', 'Ambattur',
        'Shanwei', 'So Leopoldo', 'Teboksary', 'Tianjin', 'Cianjur')
  GROUP BY A.customer_id, D.country
  ORDER BY SUM(F.amount) DESC
  LIMIT 5
) AS T ON A.customer_id = T.customer_id
GROUP BY D.country
ORDER BY all_customer_count DESC;
```

This was the key step that I missed, to give the subquery a new Table identity – in this case "T"

I missed that this syntax needed to be: LEFT JOIN, then the whole Subquery, then how it is joined

I think I would have worked out to move the GROUP BY and ORDER BY to the end

Data Output | Messages | Notifications

	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	India	60	1
2	China	53	1
3	United States	36	1
4	Japan	31	1
5	Mexico	30	1
6	Brazil	28	0
7	Russian Federation	28	0
8	Philippines	20	0
9	Turkey	15	0
10	Indonesia	14	0
11	Nigeria	13	0
12	Argentina	13	0

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?**
    - **When do you think subqueries are useful?**

First ANSWER: It feels like this could have been done without a subquery.

To do that I would need to JOIN more tables, but confident I could get there that way.

In this instance, with the subquery already created, this was probably the easiest way.

{I then tried to create this query)

Second Answer: Having attempted to do thos without a subquery, I've vchnaged my mind.

Without the subquery, ALL of the data gets filtered, in the same way

So we don't see the total number of customers per country, we see the total number, but the cities we've specified.

Subquery now feels like the best way to do this.

Subqueries will be especially useful when they already exist, and perhaps if you have a very complex query, and you want to break it down, so that it's clear what is doing what, and in the event of neededing to fix query errors.

Step 4:

Save your "Answers 3.8" document as a PDF and upload it here for your tutor to review.