3.8 Performing Subqueries

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.

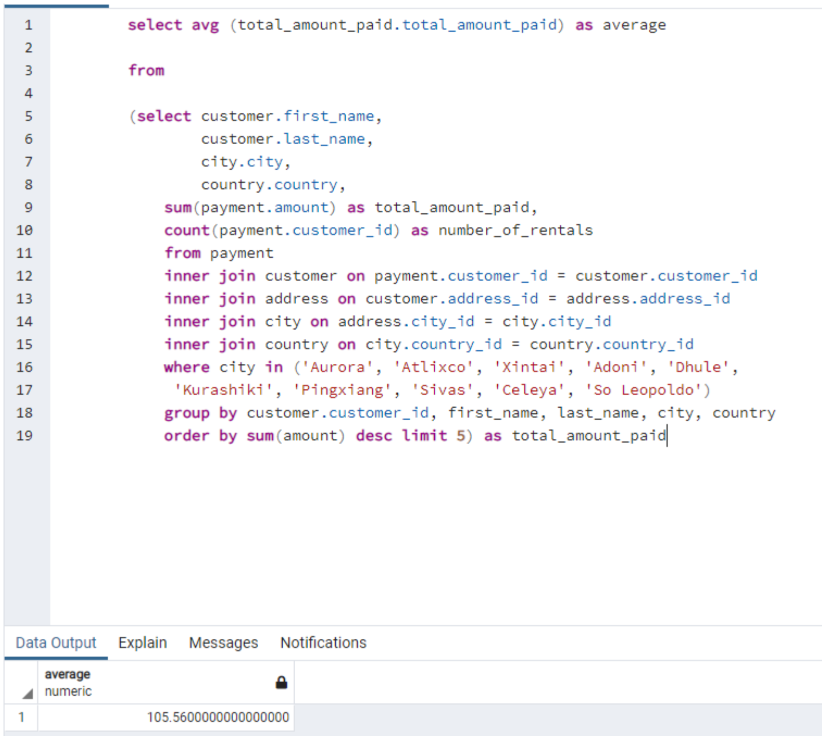
1. Copy the query you wrote in step 3 of the task from Exercise 3.7: Joining Tables of Data into the Query Tool. This will be your subquery, so give it an alias, "total_amount_paid," and add parentheses around it.
2. Write an outer statement to calculate the average amount paid.
3. Add your subquery 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".

```
select avg (total_amount_paid.total_amount_paid) as average

        from

        (select customer.first_name,
                        customer.last_name,
                        city.city,
                        country.country,
            sum(payment.amount) as total_amount_paid,
            count(payment.customer_id) as number_of_rentals
            from payment
            inner join customer on payment.customer_id = customer.customer_id
            inner join address on customer.address_id = address.address_id
            inner join city on address.city_id = city.city_id
            inner join country on city.country_id = country.country_id
            where city in ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule',
             'Kurashiki', 'Pingxiang', 'Sivas', 'Celeya', 'So Leopoldo')
            group by customer.customer_id, first_name, last_name, city, country
            order by sum(amount) desc limit 5) as total_amount_paid
```

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



```
        select avg (total_amount_paid.total_amount_paid) as average

        from

        (select customer.first_name,
                customer.last_name,
                city.city,
                country.country,
            sum(payment.amount) as total_amount_paid,
            count(payment.customer_id) as number_of_rentals
            from payment
            inner join customer on payment.customer_id = customer.customer_id
            inner join address on customer.address_id = address.address_id
            inner join city on address.city_id = city.city_id
            inner join country on city.country_id = country.country_id
            where city in ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule',
             'Kurashiki', 'Pingxiang', 'Sivas', 'Celeya', 'So Leopoldo')
            group by customer.customer_id, first_name, last_name, city, country
            order by sum(amount) desc limit 5) as total_amount_paid
```

```
   average
   numeric
1      105.5600000000000000
```

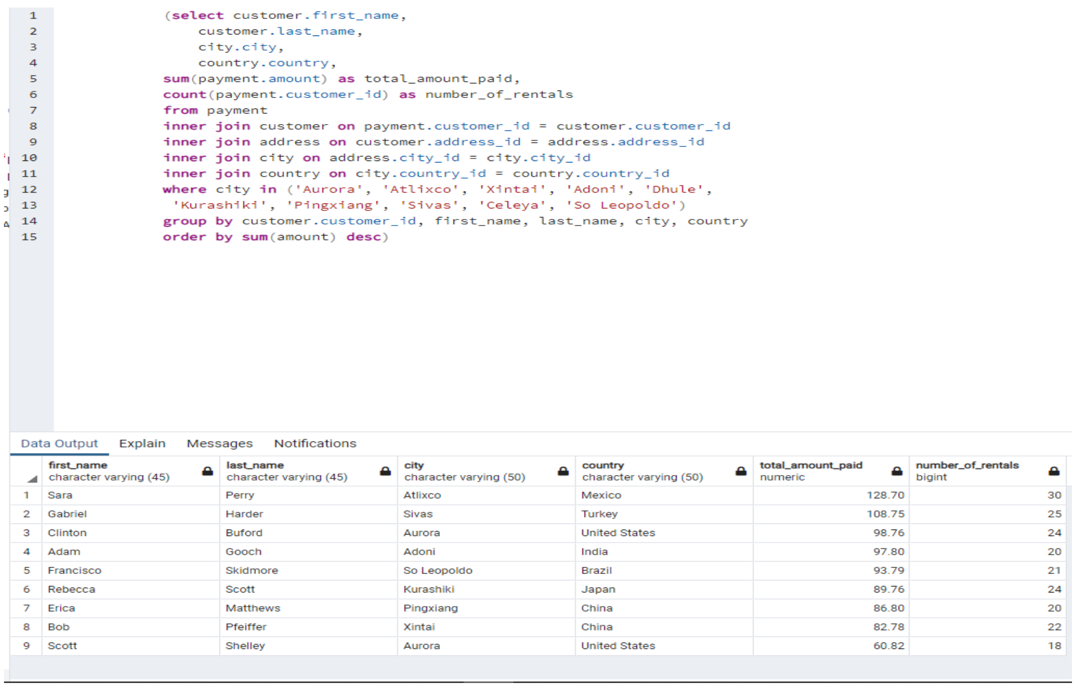
Step 2: Find out how many of the top 5 customers are based within each country.

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 below:

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.



```sql
            (select customer.first_name,
                customer.last_name,
                city.city,
                country.country,
            sum(payment.amount) as total_amount_paid,
            count(payment.customer_id) as number_of_rentals
            from payment
            inner join customer on payment.customer_id = customer.customer_id
            inner join address on customer.address_id = address.address_id
            inner join city on address.city_id = city.city_id
            inner join country on city.country_id = country.country_id
            where city in ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule',
             'Kurashiki', 'Pingxiang', 'Sivas', 'Celeya', 'So Leopoldo')
            group by customer.customer_id, first_name, last_name, city, country
            order by sum(amount) desc)
```

Data Output   Explain   Messages   Notifications

	first_name character varying (45)	last_name character varying (45)	city character varying (50)	country character varying (50)	total_amount_paid numeric	number_of_rentals bigint
1	Sara	Perry	Atlixco	Mexico	128.70	30
2	Gabriel	Harder	Sivas	Turkey	108.75	25
3	Clinton	Buford	Aurora	United States	98.76	24
4	Adam	Gooch	Adoni	India	97.80	20
5	Francisco	Skidmore	So Leopoldo	Brazil	93.79	21
6	Rebecca	Scott	Kurashiki	Japan	89.76	24
7	Erica	Matthews	Pingxiang	China	86.80	20
8	Bob	Pfeiffer	Xintai	China	82.78	22
9	Scott	Shelley	Aurora	United States	60.82	18

I changed the A, B, C nomenclature as it became too confusing to work with instead opting for the "table.specific_column_in_table" style.

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.

```sql
select

        country.country,

        count(distinct customer.customer_id) as all_customer_count

from customer

left join address on customer.address_id = address.address_id

left join city on address.city_id = city.city_id

left join country on city.country_id = country.country_id

group by 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.

4. Add a left join after your outer query, followed by the subquery in parentheses.
5. Give your subquery an alias so you can refer to it in your outer query, for example, “top_5_customers”.
6. Remember to specify which columns to join the two tables on using ON. Both ON and the column names should follow the alias.
7. Count the top 5 customers for the third column using GROUP BY and COUNT (DISTINCT). Give this column the alias “top_customer_count”.

```
select

        country.country_id,

        country.country,

        count(distinct country.country) as top_customer_count,

        count(distinct customer.customer_id) as all_customer_count




from

                        (select customer.first_name,

                                customer.last_name,

                                city.city,

                                country.country,

                        sum(payment.amount) as total_amount_paid,

                        count(payment.customer_id) as number_of_rentals

                        from payment

                        inner join customer on payment.customer_id = customer.customer_id

                        inner join address on customer.address_id = address.address_id

                        inner join city on address.city_id = city.city_id

                        inner join country on city.country_id = country.country_id

                        where city in ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule',

                         'Kurashiki', 'Pingxiang', 'Sivas', 'Celeya', 'So Leopoldo')

                        group by customer.customer_id, first_name, last_name, city, country
```

```
                    order by sum(amount) desc

                    limit 5) as top_customers


left join customer on customer.customer_id = customer.customer_id

left join address on customer.address_id = address.address_id

left join city on address.city_id = city.city_id

left join country on city.country_id = country.country_id


group by country.country_id

order by all_customer_count desc

limit 10;
```

8. Copy-paste your query and the data output into your “Answers 3.8” document.

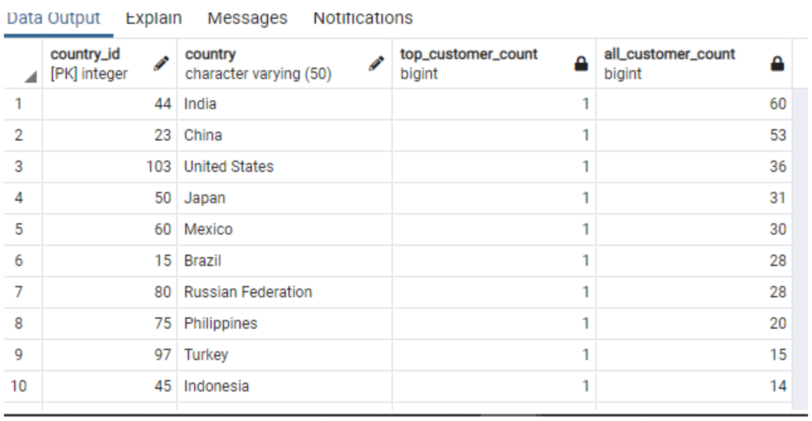
Step 3:

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

Both steps require a subquery. In step one, there is no ability to get an average of the top customer’s payments just from the tables. That is, one has to create the inner statement and take an average from that output. In this case, the totals in the top 5 were 128.7,108.75,98.76,97.8, and 93.79 averaging to 105.56. Even running a “select avg(payment.amount)” only yields the average each customer paid over their total rentals (which is just total_amount_paid divided by number_of_rentals).

Step 2 definitely requires a subquery as we are asking to find the total number of renters in the top countries from those previously found cities (countries found in question 1 of the 3.7 lesson) AS WELL AS the number of top paying customers in those countries while keeping the country and its ID intact.

The major issue I find is that if you look at the inner statement by itself, it returns two entries from China and two entries from the United States. However, when I run the full statements from above, it describes each of the top 10 countries as having 1 top customer even if I change the limiting factor within the inner statement to 10 instead of 5. I think this has to do with the ‘distinct’ command under the count in the top selection, but I’m not sure how to remedy this.


Data Output Explain Messages Notifications

	country_id [PK] integer	country character varying (50)	top_customer_count bigint	all_customer_count bigint
1	44	India	1	60
2	23	China	1	53
3	103	United States	1	36
4	50	Japan	1	31
5	60	Mexico	1	30
6	15	Brazil	1	28
7	80	Russian Federation	1	28
8	75	Philippines	1	20
9	97	Turkey	1	15
10	45	Indonesia	1	14

- When do you think subqueries are useful?

Subqueries seem to be invaluable anytime you want to find out specific information from an existing query and to combine that with other tables within the database. As described in the lesson, this is a good way to extract from someone else’s existing query which should save time for both the analyst and the company.

Step 4:

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