STEP 1

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.

```
--query from exercise 3.7 prepared into subquery to use later with the outer statement
SELECT A.customer_id, A.first_name, A.last_name, c.city, d.country, SUM(amount) AS
total_amount_paid
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 payment E ON A.customer id = E.customer id
WHERE c.city IN(SELECT C.city
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
WHERE D.country IN
(SELECT 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
GROUP BY country
ORDER BY COUNT (customer_id) DESC
LIMIT 10)
GROUP BY D.country, C.city
ORDER BY COUNT (customer_id) DESC
LIMIT 10)
GROUP BY A.customer_id, A.first_name, A.last_name, c.city, d.country
ORDER BY total_amount_paid DESC
LIMIT 5
) AS total_amount_paid;
```

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

```
--outer statement for calculating average amount paid
SELECT AVG(amount)
FROM payment
```

3. Add your subquery to the outer statement.

```
-- joint statement
SELECT AVG(total_amount_paid) as average_top_five
FROM (
SELECT A.customer_id, A.first_name, A.last_name, C.city, D.country, SUM(amount) AS
total amount paid
 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 payment E ON A.customer_id = E.customer_id
WHERE C.city IN (
 SELECT C.city
 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
 WHERE D.country IN (
  SELECT 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
  GROUP BY D.country
  ORDER BY COUNT(customer_id) DESC
  LIMIT 10
 )
 GROUP BY D.country, C.city
 ORDER BY COUNT(customer_id) DESC
 LIMIT 10
 )
 GROUP BY A.customer_id, A.first_name, A.last_name, C.city, D.country
 ORDER BY total_amount_paid DESC
 LIMIT 5
) AS total_amount_paid;
```

Query Query History

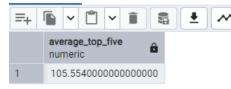
```
2 🗸 (
3 SELECT A.customer_id, A.first_name, A.last_name, c.city, d.country, SUM(amount) AS total_amount_paid
5 INNER JOIN address B ON A.address_id = B.address_id
 6 INNER JOIN city C ON b.city_id = c.city_id
7 INNER JOIN country D ON c.country_id = d.country_id
8 INNER JOIN payment E ON A.customer_id = E.customer_id
9 WHERE c.city IN(SELECT C.city
10 FROM customer A
11 INNER JOIN address B ON A.address_id = B.address_id
12 INNER JOIN city C ON B.city_id = C.city_id
13 INNER JOIN country D on C.country_id = D.country_id
14 WHERE D.country IN
15 (SELECT D.country
16 FROM customer A
17 INNER JOIN address B ON A.address_id = B.address_id
18 INNER JOIN city C ON B.city_id = C.city_id
19  INNER JOIN country D on C.country_id = D.country_id
20 GROUP BY country
21 ORDER BY COUNT (customer_id) DESC
22 LIMIT 10)
23 GROUP BY D.country, C.city
24 ORDER BY COUNT (customer_id) DESC
25 LIMIT 10)
26 GROUP BY A.customer_id, A.first_name, A.last_name, c.city, d.country
27 ORDER BY total amount paid DESC
28 LIMIT 5
29 ) AS total amount paid;
30
```

1 --query from exercise 3.7 prepared into subquery to use later with the outer statement

```
31 -- outer statement for calculating average amount paid
32 - SELECT AVG(amount)
33 FROM payment
35 -- joint statement
     SELECT AVG(total_amount_paid) as average_top_five
37
    FROM (
38 SELECT A.customer_id, A.first_name, A.last_name, C.city, D.country, SUM(amount) AS total_amount_paid
      FROM customer A
40
      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
42
       INNER JOIN payment E ON A.customer_id = E.customer_id
44
       WHERE C.city IN (
45
         SELECT C.city
46
         FROM customer A
47
         INNER JOIN address B ON A.address id = B.address id
48
         INNER JOIN city C ON B.city_id = C.city_id
49
         INNER JOIN country D ON C.country_id = D.country_id
50
         WHERE D.country IN (
51
          SELECT D.country
52
          FROM customer A
53
          INNER JOIN address B ON A.address_id = B.address_id
54
          INNER JOIN city C ON B.city_id = C.city_id
55
          INNER JOIN country D ON C.country_id = D.country_id
56
          GROUP BY D. country
57
          ORDER BY COUNT(customer_id) DESC
58
          LIMIT 10
59
60
         GROUP BY D.country, C.city
61
         ORDER BY COUNT(customer_id) DESC
62
63
64
       GROUP BY A.customer_id, A.first_name, A.last_name, C.city, D.country
65
      ORDER BY total_amount_paid DESC
      LIMIT 5
67 ) AS total_amount_paid;
```

Data Output Messages Notifications

SQL.



STEP 2

1. Copy the guery from step 3 of task 3.7 into the Query Tool and add parentheses around it. This will be your inner guery.

```
--query from exercise 3.7 prepared into subquery to use later with the outer statement
SELECT A.customer_id, A.first_name, A.last_name, c.city, d.country, SUM(amount) AS total_amount_paid
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 payment E ON A.customer_id = E.customer_id
WHERE c.city IN(SELECT C.city
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
WHERE D.country IN
(SELECT 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
GROUP BY country
ORDER BY COUNT (customer_id) DESC
LIMIT 10)
GROUP BY D.country, C.city
```

- 2. Write an outer statement that counts the number of customers living in each country.
 - -- outer statement counting the number of customers living in each country

SELECT COUNT(DISTINCT customer.customer_id) AS all_customer_count, country.country

FROM customer

INNER JOIN address ON address_id = customer.address_id

INNER JOIN city ON city.city_id = address.city_id

INNER JOIN country ON country.country_id = city.country_id

GROUP BY country.country

ORDER BY all_customer_count DESC;

3. Place your inner query in the outer query.

```
SELECT
```

all_countries.country,

all_countries.all_customer_count,

COUNT(top_customers.customer_id) AS top_5_customer_count

FROM (

-- Total number of customers per country

SELECT country, COUNT(DISTINCT customer.customer_id) AS all_customer_count

FROM customer

INNER JOIN address ON address.address_id = customer.address_id

INNER JOIN city ON city.city_id = address.city_id

INNER JOIN country ON country.country_id = city.country_id

GROUP BY country.country

) AS all_countries

LEFT JOIN (

-- Top 5 customers by total amount paid

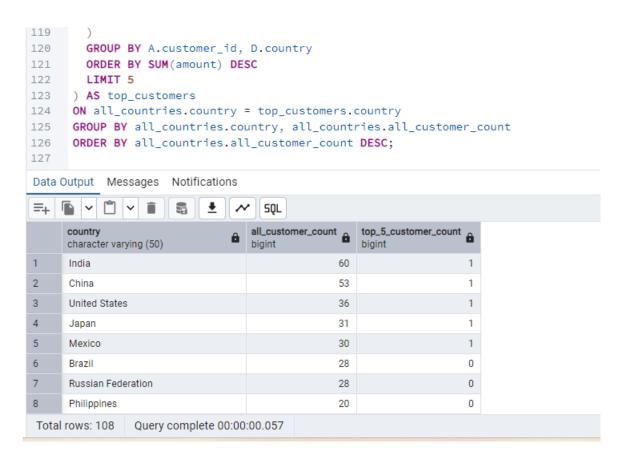
```
SELECT A.customer_id, D.country
FROM customer A
INNER JOIN address B ON A.address id = B.address id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country id = D.country id
INNER JOIN payment E ON A.customer_id = E.customer_id
WHERE C.city IN (
SELECT C.city
 FROM customer A
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
 WHERE D.country IN (
 SELECT 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
 GROUP BY D.country
 ORDER BY COUNT(customer_id) DESC
 LIMIT 10
 GROUP BY D.country, C.city
 ORDER BY COUNT(customer_id) DESC
 LIMIT 10
GROUP BY A.customer id, D.country
```

ORDER BY SUM(amount) DESC LIMIT 5

) AS top_customers

ON all_countries.country = top_customers.country
GROUP BY all_countries.country, all_countries.all_customer_count
ORDER BY all_countries.all_customer_count DESC;

```
78 -- joint statement querying the total number of customers and the number of top 5 customers in each country where Rockbuster operates
80 all_countries.country,
81 all_countries.all_customer_count,
82 COUNT(top_customers.customer_id) AS top_5_customer_count
83 FROM (
84 -- Total number of customers per country
85 SELECT country.country, COUNT(DISTINCT customer.customer_id) AS all_customer_count
87 INNER JOIN address ON address.address_id = customer.address_id
88 INNER JOIN city ON city.city_id = address.city_id
89 INNER JOIN country ON country.country_id = city.country_id
90 GROUP BY country.country
91 ) AS all_countries
92 LEFT JOIN (
93 -- Top 5 customers by total amount paid
94 SELECT A.customer_id, D.country
95 FROM customer A
96 INNER JOIN address B ON A.address id = B.address id
97 INNER JOIN city C ON B.city_id = C.city_id
98 INNER JOIN country D ON C.country_id = D.country_id
99 INNER JOIN payment E ON A.customer_id = E.customer_id
100 WHERE C.city IN (
101 SELECT C.city
102
       FROM customer A
       INNER JOIN address B ON A.address_id = B.address_id
104
       INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
106 WHERE D.country IN (
107
      SELECT D.country
108 FROM customer A
INNER JOIN country D ON C.country_id = D.country_id
112 GROUP BY D.country
113
        ORDER BY COUNT(customer_id) DESC
114
        LIMIT 10
115
116 GROUP BY D.country, C.city
ORDER BY COUNT(customer_id) DESC
118
       LIMIT 10
```



STEP 3

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?

If steps 1 and 2 could be done without using subqueries (how great would that be!), I'm not aware of it. We needed a piece of information composed of bits from different tables—basically, one piece of info from an inner query that's then filtered or used by the outer query—to get exactly what we wanted, no more and no less.