**3.8: Performing Subqueries**
**Prepared By: Rob Rowland**

**Step 1:**

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

| | avg<br>numeric |
|---|---|
| 1 | 107.3540000000000000 |

**Step 2:**

```
SELECT country.country,
COUNT(DISTINCT customer.customer_id) AS all_customer_count,
COUNT(DISTINCT country.country) AS top_customer_count
FROM
(SELECT A.customer_id, B.first_name, B.last_name, D.city, E.country,
SUM(amount) AS highest_amount_paid
FROM payment A
INNER JOIN customer B ON A.customer_id = B.customer_id
INNER JOIN address C ON B.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 country IN ('India',
                  'China',
                  'United States',
                  'Japan',
                  'Mexico',
                  'Brazil',
                  'Russian Federation',
                  'Philippines',
                  'Turkey',
                  'Indonesia')
AND city IN ('Aurora',
             'Atlixco',
             'Xintai',
             'Adoni',
             'Dhule (Dhulia)',
             'Kurashiki',
             'Pingxiang',
             'Sivas',
             'Celaya',
             'So Leopoldo')
GROUP BY A.customer_id, B.first_name, B.last_name, D.city, E.country
ORDER BY highest_amount_paid DESC
LIMIT 5) AS top_5_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
ORDER BY COUNT (country.country) DESC
LIMIT 5;
```

| | country<br>character varying (50) | all_customer_count<br>bigint | top_customer_count<br>bigint |
|---|---|---|---|
| 1 | India | 60 | 1 |
| 2 | China | 53 | 1 |
| 3 | United States | 36 | 1 |
| 4 | Japan | 31 | 1 |
| 5 | Mexico | 30 | 1 |

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

Step 1 likely could have been completed without a subquery since it only added one additional layer to the original query. Step 2 likely could as well, but with a higher degree of difficulty. I believe using a subquery, as shown above, would ultimately be the best way to accomplish step 2.

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

When an applicable query already exists, using a subquery can avoid the hassle of rebuilding the query or constantly refreshing it to get the most current information. They are most useful when they don't add a significant layer of expense, but do save a significant amount of time in executing the correct query for the information needed.