William Su

Task 3.8 – Performing Subqueries

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

```
SELECT AVG (total_amount) AS average
FROM
    (SELECT B.customer_id, B.first_name, B.last_name, E.country, D.city, SUM(A.amount) AS total_amount
    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
    D.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
    GROUP BY B.customer_id, D.city, E.country
    ORDER BY total_amount DESC
    LIMIT 5) AS total_amount_paid;
```

Data Output | Messages | Notifications

	average numeric
1	107.3540000000000000

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

Query | Query History

```
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 B.customer_id, B.first_name, B.last_name, E.country, D.city, SUM(A.amount) AS total_amount
    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
    D.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki', 'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
    GROUP BY B.customer_id, D.city, E.country
    ORDER BY total_amount DESC
    LIMIT 5) AS total_amount_paid
ON A.country = total_amount_paid.country
GROUP BY A.country, total_amount_paid
ORDER BY all_customer_count DESC
LIMIT 5
```

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

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?

Step 1 could have been done without a subquery by using the HAVING clause and using aggregate functions. Step 2, however, would need subqueries as we needed data from multiple different data tables. If all the data was congregated into one giant data table, then it would not have needed subqueries.

Subqueries are very useful when we need to compare and use different data points, as well as being very good at filtering data. It is very useful when used to combine multiple steps together, as well as being able to constantly update itself. For example, instead of using one query to find out information for another query, we would be able to use the first query as a subquery for the second, allowing it to update itself constantly.