Apply filters to SQL queries

1. Project Description

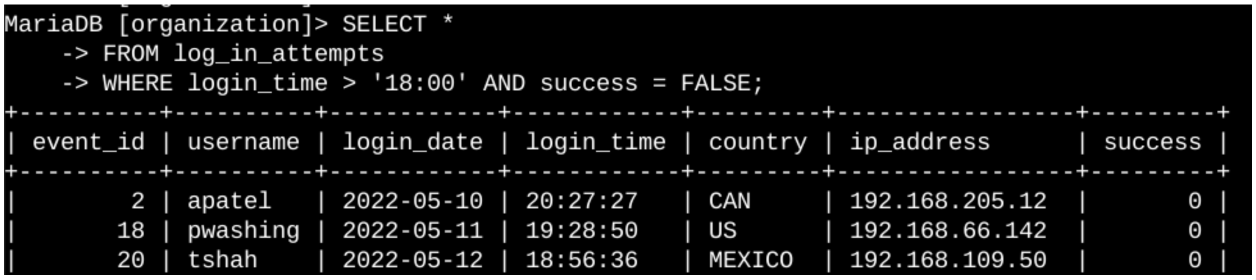
My company is working to increase the security of their system. My responsibility is to make sure the system is secure, look into any potential security problems, and update employee computers as necessary. The steps that follow show how I carried out security-related tasks using SQL and filters.

2. Retrieve after hours failed login attempt:

```
SELECT *

FROM log_in_attempts

WHERE login_time > '18:00:00'

AND success = 0;
```

This query first selects all rows from the `log_in_attempts` table. Then, it uses the `WHERE` clause to filter the results to only include rows where the `login_time` column is greater than `18:00:00`. Finally, it uses the `AND` operator to filter the results further to only include rows where the `success` column is equal to `false`. This means that the query will only return rows that represent failed login attempts that occurred after 18:00.

Here is a screenshot of the SQL query:



```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_time > '18:00' AND success = FALSE;
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        2 | apatel   | 2022-05-10 | 20:27:27   | CAN     | 192.168.205.12  |       0 |
|       18 | pwashing | 2022-05-11 | 19:28:50   | US      | 192.168.66.142  |       0 |
|       20 | tshah    | 2022-05-12 | 18:56:36   | MEXICO  | 192.168.109.50  |       0 |
```

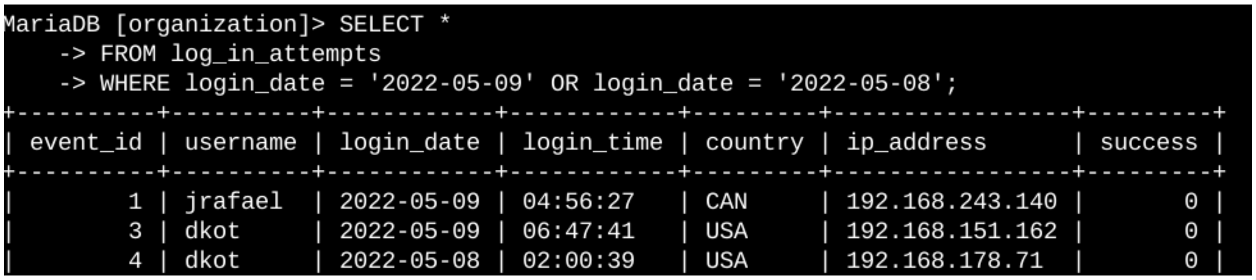
The query works by first selecting all rows from the `log_in_attempts` table. Then, it uses the `WHERE` clause to filter the results to only include rows where the `login_time` column is greater than `18:00:00`. This means that the query will only return rows that represent login attempts that occurred after 18:00. Finally, it uses the `AND` operator to filter the results further to only include rows where the `success` column is equal to `false`. This means that the query will only return rows that represent failed login attempts.

3. Retrieve login attempts on specific dates:

```
SELECT *

FROM log_in_attempts

WHERE login_date = '2022-05-09'

OR login_date = '2022-05-08';
```

This query first selects all rows from the `log_in_attempts` table. Then, it uses the `WHERE` clause to filter the results to only include rows where the `login_date` column is equal to either `2022-05-09` or `2022-05-08`. This means that the query will only return rows that represent login attempts that occurred on either of these two dates.

Here is a screenshot of the SQL query:



```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        1 | jrafael  | 2022-05-09 | 04:56:27   | CAN     | 192.168.243.140 |       0 |
|        3 | dkot     | 2022-05-09 | 06:47:41   | USA     | 192.168.151.162 |       0 |
|        4 | dkot     | 2022-05-08 | 02:00:39   | USA     | 192.168.178.71  |       0 |
```

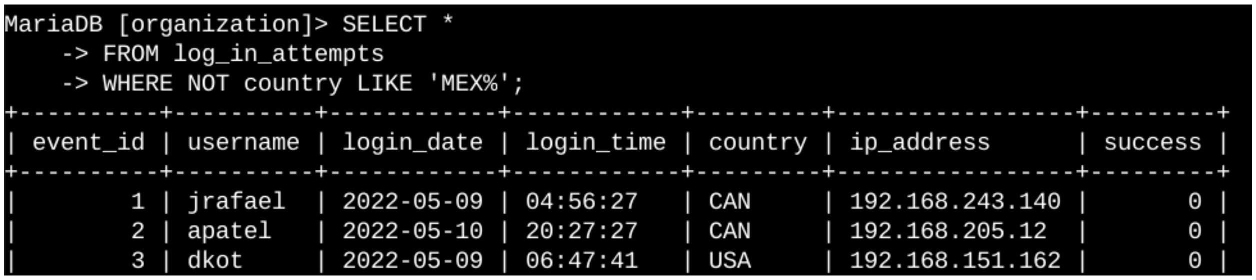
The query works by first selecting all rows from the `log_in_attempts` table. Then, it uses the `WHERE` clause to filter the results to only include rows where the `login_date` column is equal to either `2022-05-09` or `2022-05-08`. This means that the query will only return rows that represent login attempts that occurred on either of these two dates.

4. Retrieve login attempts outside of Mexico:

```
SELECT *

FROM log_in_attempts

WHERE country NOT LIKE '%MEX%';
```

This query first selects all rows from the `log_in_attempts` table. Then, it uses the `WHERE` clause to filter the results to only include rows where the `country` column does not contain the substring `MEX`. This means that the query will only return rows that represent login attempts that did not originate in Mexico.

Here is a screenshot of the SQL query:



```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE NOT country LIKE 'MEX%';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        1 | jrafael  | 2022-05-09 | 04:56:27   | CAN     | 192.168.243.140 |       0 |
|        2 | apatel   | 2022-05-10 | 20:27:27   | CAN     | 192.168.205.12  |       0 |
|        3 | dkot     | 2022-05-09 | 06:47:41   | USA     | 192.168.151.162 |       0 |
```

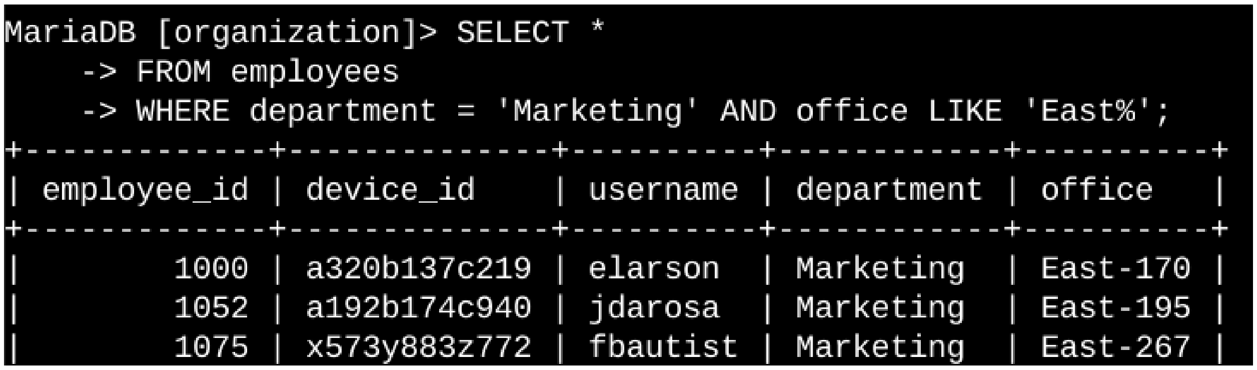
The `LIKE` keyword is used to match a substring within a column. In this case, we are using the `LIKE` keyword to match the substring `MEX` within the `country` column. The `%` wildcard is used to match any number of characters. So, the `LIKE '%MEX%'` clause will match any row where the `country` column contains the substring `MEX`, regardless of how many other characters are present in the column.

5. Retrieve employees in Marketing:

```
SELECT *

FROM employees

WHERE department LIKE '%Marketing%'

AND office LIKE '%East%';
```

This query first selects all rows from the `employees` table. Then, it uses the `WHERE` clause to filter the results to only include rows where the `department` column contains the substring `Marketing` and the `office` column contains the substring `East`. This means that the query will only return rows that represent employees who work in the Marketing department and who have an office in the East building.

Here is a screenshot of the SQL query:



```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE department = 'Marketing' AND office LIKE 'East%';
+-------------+-------------+----------+------------+----------+
| employee_id | device_id   | username | department | office   |
+-------------+-------------+----------+------------+----------+
|        1000 | a320b137c219 | elarson  | Marketing  | East-170 |
|        1052 | a192b174c940 | jdarosa  | Marketing  | East-195 |
|        1075 | x573y883z772 | fbautist | Marketing  | East-267 |
```

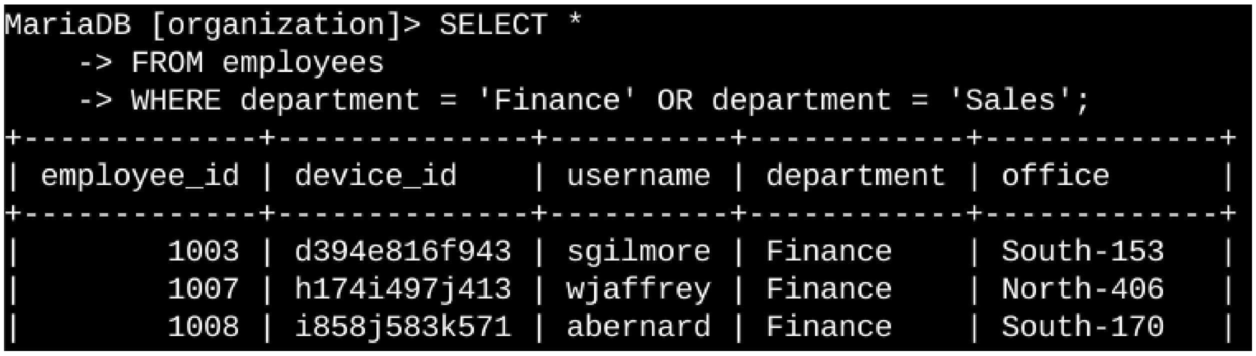
The `LIKE` keyword is used to match a substring within a column. In this case, we are using the `LIKE` keyword to match the substring `Marketing` within the `department` column and the substring `East` within the `office` column. The `%` wildcard is used to match any number of characters. So, the `LIKE '%Marketing%'` clause will match any row where the `department` column contains the substring `Marketing`, regardless of how many other characters are present in the column.

6. Retrieve employees in Finance or Sales:

```
SELECT *

FROM employees

WHERE department LIKE '%Finance%'

OR department LIKE '%Sales%';
```

This query first selects all rows from the `employees` table. Then, it uses the `WHERE` clause to filter the results to only include rows where the `department` column contains the substring `Sales` or the substring `Finance`. This means that the query will only return rows that represent employees who work in either the Sales or Finance department.

Here is a screenshot of the SQL query:



```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE department = 'Finance' OR department = 'Sales';
+-------------+-------------+----------+------------+-------------+
| employee_id | device_id   | username | department | office      |
+-------------+-------------+----------+------------+-------------+
|        1003 | d394e816f943 | sgilmore | Finance    | South-153   |
|        1007 | h174i497j413 | wjaffrey | Finance    | North-406   |
|        1008 | i858j583k571 | abernard | Finance    | South-170   |
```

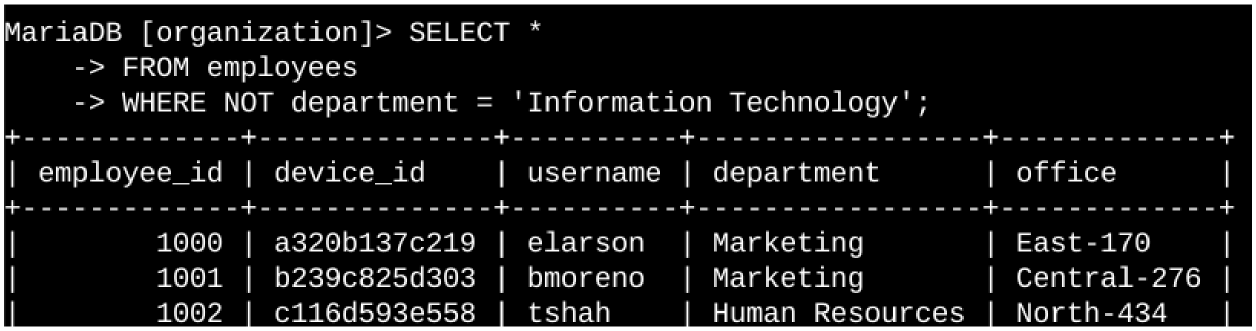
The `LIKE` keyword is used to match a substring within a column. In this case, we are using the `LIKE` keyword to match the substring `Sales` or the substring `Finance` within the `department` column. The `%` wildcard is used to match any number of characters. So, the `LIKE '%Sales%'` clause will match any row where the `department` column contains the substring `Sales`, regardless of how many other characters are present in the column.

7. Retrieve all employees not in IT:

```
SELECT *

FROM employees

WHERE department NOT ='%Information Technology%';
```

This query first selects all rows from the `employees` table. Then, it uses the `WHERE` clause to filter the results to only include rows where the `department` column does not contain the substring `Information Technology`. This means that the query will only return rows that represent employees who do not work in the IT department.

Here is a screenshot of the SQL query:



```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE NOT department = 'Information Technology';
+-------------+-------------+----------+-----------------+-------------+
| employee_id | device_id   | username | department      | office      |
+-------------+-------------+----------+-----------------+-------------+
|        1000 | a320b137c219 | elarson  | Marketing       | East-170    |
|        1001 | b239c825d303 | bmoreno  | Marketing       | Central-276 |
|        1002 | c116d593e558 | tshah    | Human Resources | North-434   |
```

The `NOT` keyword is used to negate the results of a filter. In this case, we are using the `NOT` keyword to filter out rows where the `department` column contains the substring `Information Technology`. This means that the query will only return rows that represent employees who do not work in the IT department.

8. Project Summary

I used filters on SQL queries to obtain detailed data on login attempts and employee computers. Log_in_attempts and Employees were the two different tables I used. I applied filters using the AND, OR, and NOT operators to find the precise data required for each task. To search for patterns, I also used the LIKE and percentage sign (%) wildcards.