Apply filters to SQL queries

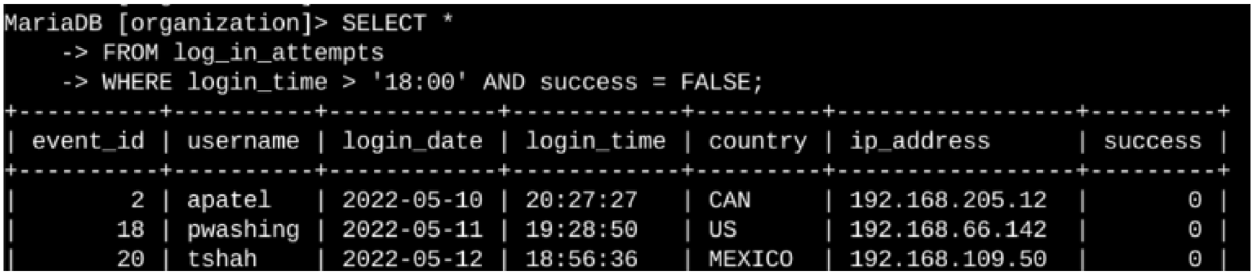
Project description

In this project I examined an organization's data in their **Employees** and **Log_in_attempts** tables. I used SQL filters and operators to retrieve records from different datasets and investigated potential security issues. The organization database contains the following two tables:

- **Log_in_attempts**
- **Employees**

Retrieve after hours failed login attempts

It was suspected that there was a potential security incident after work hours (after 18:00:00). I used the following code to create a SQL query that filtered to only display failed login attempts that happened after business hours (18:00:00)

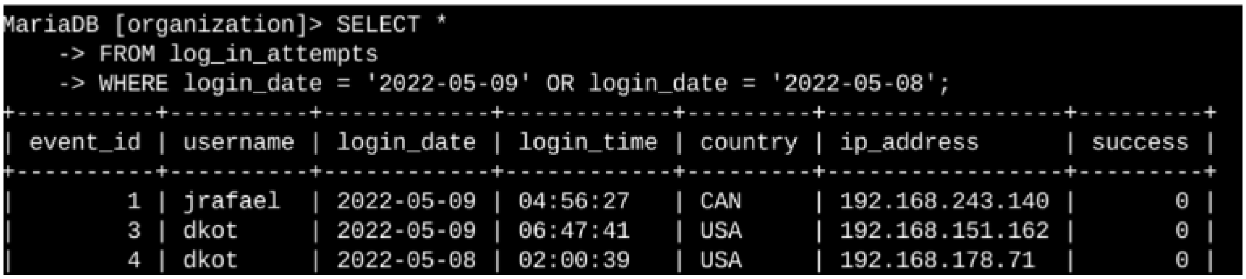


```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_time > '18:00' AND success = FALSE;
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 first part of the query selects all data from the log_in_attempts table.
I Then used a WHERE clause with an AND operator to filter my results to output only login attempts that occurred after 18:00 and were unsuccessful.
The first condition is login_time > '18:00', which filters for the login attempts that occurred after 18:00.
The second condition is success = FALSE, which filters for the failed login attempts.

Retrieve login attempts on specific dates

There was a suspicious event that occurred on 2022-05-09. I investigated any login activity that happened on 2022-05-09 or the day before. The code in the below screenshot demonstrates how I created a SQL query to filter for login attempts that occurred on the specific dates required.

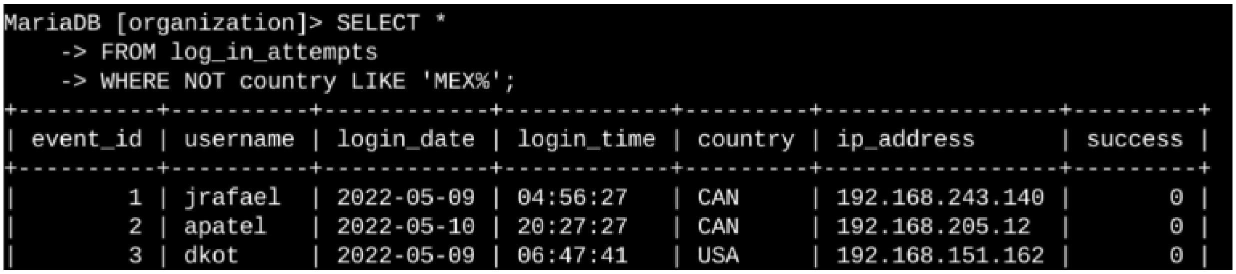


```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 |
```

Once again I selected all items from the Log_in_attempts table using *
I then used a **WHERE** statement to filter only Login_dates that were equal to ‘2022-05-9’ **OR** ‘2022-05-08’

Retrieve login attempts outside of Mexico

After I had finished investigating the organization’s login attempts, I believed there was an issue with the login attempts that occurred outside of Mexico. The code in the following screenshot demonstrates how I created a SQL query to filter for login attempts that occurred outside of Mexico.

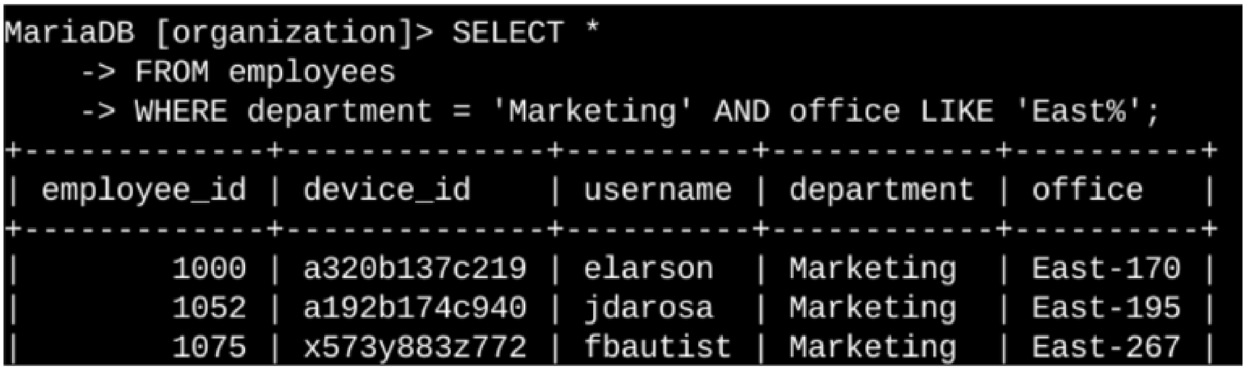


```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE NOT country LIKE 'MEX%';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 |
```

Again I selected all items from the Log_in_attempts table using *
I then used a **WHERE** statement to filter and display only countries that were not **LIKE ‘**MEX%’

Retrieve employees in Marketing

I was then asked to update computers for certain employees in the Marketing department.
To complete this task I needed to find information on which computers I needed to update.
The following code demonstrates how I created a SQL query to filter for employee machines from employees in the Marketing department in the East building.

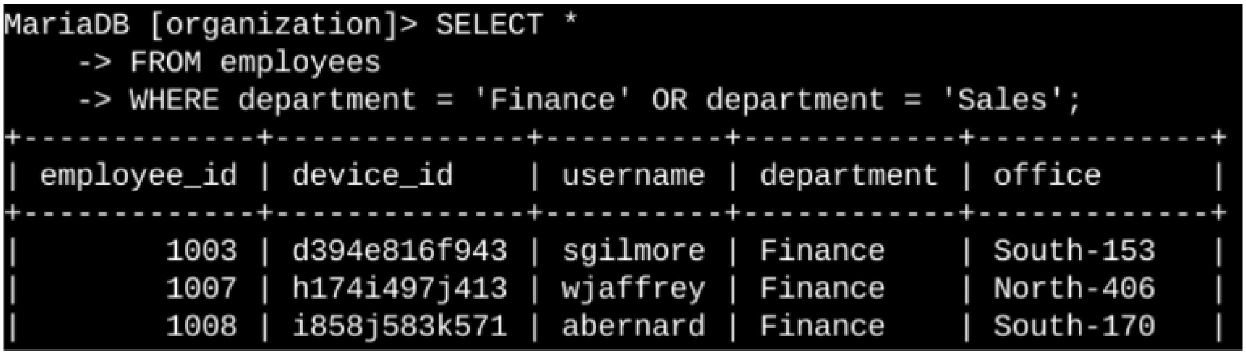


```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE department = 'Marketing' AND office LIKE 'East%';
+-------------+-------------+----------+------------+----------+
| employee_id | device_id   | username | department | office   |
+-------------+-------------+----------+------------+----------+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

 I selected all items from the Log_in_attempts table using *
I then used the **WHERE** statement to filter results to only display employees in the **Marketing** department **AND** any office in the East building using **LIKE** ‘east%’ as the pattern to match.

Retrieve employees in Finance or Sales

I also needed to retrieve the data for employee machines in the Finance or Sales departments. The code below demonstrates how I created a SQL query to filter for employee machines only from the Finance and Sales departments.

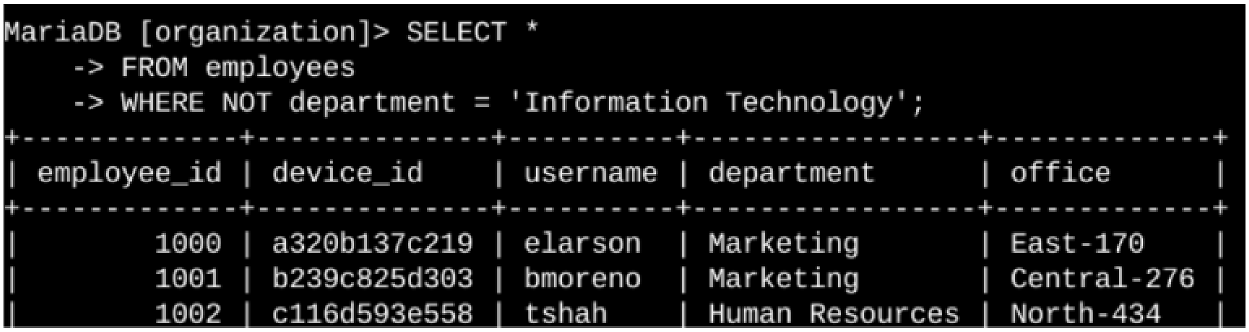


```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE department = 'Finance' OR department = 'Sales';
+-------------+-------------+----------+------------+-------------+
| employee_id | device_id   | username | department | office      |
+-------------+-------------+----------+------------+-------------+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

Again I used **SELECT *** to select all data from the **employees** table.
Then i used a **WHERE** clause to filter for employees in ‘Finance’ **OR** ‘Sales’
Here I used the **OR** operator instead of **AND** because I wanted all employees in either department.

Retrieve all employees not in IT

Lastly I had to make one more security update. This was for all employees who are not in the Information Technology department. The following code demonstrates the SQL query I used in order to retrieve the data I required.



```
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   |
```

Again I used **SELECT *** to select all data from the **employees** table.
Then I used a **WHERE** clause with a **NOT** operator to filter for all employees that are not in the Information Technology department.

Summary

In this project I applied varies filters to SQL queries to gain specific data on employee end devices and login attempts. I used the **Employees** and **Log_in_attempts** tables. I used the following operators:

- **AND**
- **OR**
- **NOT**

I also used **LIKE** and the **%** wildcard to filter for patterns.