# Apply filters to SQL queries

## Project description

My organization is working toward enhancing its security measures. As part of my role, I am responsible for ensuring that the systems are secure, investigating potential security issues, and updating employee computers as needed. The following steps outline examples of how I used SQL with filters to perform various security-related tasks.

## Retrieve after-hours failed login attempts

A potential security incident occurred after business hours (after 6:00 PM), and all login attempts need to be investigated. To filter for failed login attempts that took place after business hours, I used the following 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 |
|       28 | aestrada | 2022-05-09 | 19:28:12   | MEXICO  | 192.168.27.57   |       0 |
```

The first three lines of the snippet represent my query, and the rest is the output. To filter for failed login attempts after business hours, I first selected all data from the `log_in_attempts` table. Then I use the `WHERE` clause and `AND` operator to apply two conditions: `login_time > '18:00'` which filters for login attempts after 6:00 PM and `success= FALSE` to capture only the failed attempts.

## Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. Any login activity that happened on that day or the day before needs to be investigated. To filter for login attempts on those specific dates, I used the following query:
The first three lines of the snippet represent my query, and the rest is the output. I started by selecting all data from the `log_in_attempts` table. Then I used the `WHERE` clause with `OR` operator to filter login attempts that occurred on 2022-05-08 and 2022-05-09.

```
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 |       1 |
|        3 | dkot     | 2022-05-09 | 06:47:41   | USA     | 192.168.151.162 |       1 |
|        4 | dkot     | 2022-05-08 | 02:00:39   | USA     | 192.168.178.71  |       0 |
|        8 | bisles   | 2022-05-08 | 01:30:17   | US      | 192.168.119.173 |       0 |
|       12 | dkot     | 2022-05-08 | 09:11:34   | USA     | 192.168.100.158 |       1 |
|       15 | lyamamot | 2022-05-09 | 17:17:26   | USA     | 192.168.183.51  |       0 |
```

## Retrieve login attempts outside of Mexico

After investigating the login attempt data, I noticed an issue with attempts originating outside of Mexico. These should be further investigated. To filter for login attempts that occurred outside of Mexico, I used the following 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 |       1 |
|        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 |       1 |
|        4 | dkot     | 2022-05-08 | 02:00:39   | USA     | 192.168.178.71  |       0 |
|        5 | jrafael  | 2022-05-11 | 03:05:59   | CANADA  | 192.168.86.232  |       0 |
```

The first three lines of the snippet represent my query, and the rest is the output. I started by selecting all data from the `log_in_attempts` table. Then I used `WHERE` clause with the `NOT` operator to exclude results that match `LIKE MEX%,` which filtered for entries starting with “MEX” including “MEX” itself.

## Retrieve employees in Marketing

My team wanted to update certain computers in the East building that are assigned to employees from the marketing department. To accomplish this, I needed to gather information on which employee machines required an update. I used the following query to retrieve this data

The first three lines of the snippet represent my query, and the rest is the output. I started by selecting all data from the `employee` table. Then I use the `WHERE` clause and `AND` operator to apply two conditions: `department = marketing` to filter for employees from the marketing department and `office = LIKE ‘East%’` to capture employees located 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 |
|        1088 | k8651965m233 | rgosh    | Marketing  | East-157 |
|        1103 | NULL         | randerss | Marketing  | East-460 |
|        1156 | a184b775c707 | dellery  | Marketing  | East-417 |
|        1163 | h679i515j339 | cwilliam | Marketing  | East-216 |
+-------------+-------------+----------+------------+----------+
```

## Retrieve employees in Finance or Sales

The Finance and Sales departments also need to be updated. Since it's a different update, I need to retrieve information only for employees from these two departments. The following code snippet was used to retrieve this data:

```
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   |
|        1009 | NULL         | lrodriqu | Sales      | South-134   |
```

The first three lines of the snippet represent my query, and the rest is the output. I started by selecting all data from the `employee` table. Then I use the `WHERE` clause and `OR` operator to select all employees from the Finance and Sales departments by using the following conditions: `department = 'Finance'` and `department = 'Sales'`.

## Retrieve all employees not in the IT

Another security update needs to be performed for all employees who do not belong to the IT department. To achieve this, I used the following query to retrieve the necessary information:

The first three lines of the snippet represent my query, and the rest is the output. I started by selecting all data from the `employee` table. Then I use the `WHERE` clause and `NOT` operator to select all employees who are not from the `Information Technology` department.

```
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   |
|        1003 | d394e816f943 | sgilmore | Finance         | South-153   |
```

## Summary

I applied filters to SQL queries to retrieve specific information on login attempts and employee machines to perform two main security tasks. The first task involved using the `log_in_attempts` table to investigate and document any suspicious activities. The second task utilized the `employee` table to identify and perform various updates. To filter the appropriate data, I used operators such as `AND`, `OR`, and `NOT`. Additionally, I refined the filters by using the `LIKE` operator with the percentage sign (`%`) wildcard for pattern matching.