Applying filters to SQL queries

This was a mock scenario investigating failed login attempts from employees and company machines at a large organization with multiple regional offices. To parse for logins, I'm using SQL to examine log files to determine an initial root cause.

Retrieve after hours failed login attempts

```
SELECT *
FROM log_in_attempts
WHERE login_time > '18:00' AND success = FALSE;
```

I used the > operator to filter out logins after 18:00 which is afterhours for the organization. Also included the AND operator to filter out the unsuccessful logins

```
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 |
```

Retrieve login attempts on specific dates

```
SELECT *
FROM log_in_attempts
WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
```

Using the OR operator, I was able to filter out login attempts down to these two specific dates

```
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 |
```

Retrieve login attempts outside of Mexico

```
SELECT *
FROM log_in_attempts
WHERE NOT country LIKE 'MEX%';
```

I used the NOT function to filter out login attempts from outside of Mexico. Because some formating of the country in the table is MEX, I used the MEX% to ensure I don't miss any filtered attempts

```
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 |
```

Retrieve employees in Marketing

```
SELECT *
FROM employees
WHERE department = 'Marketing' AND office LIKE 'East%';
```

To retrieve employees in the Marketing department of the East offices, I included the AND operator to filter out both departments and LIKE with East% to pull from all offices including East in the table itself.

```
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 |
```

Retrieve employees in Finance or Sales

```
SELECT *
FROM employees
WHERE department = 'Finance' OR department = 'Sales';
```

I used the OR function to filter out employees only in 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   |
```

Retrieve all employees not in IT

```
SELECT *
FROM employees
WHERE NOT department = 'Information Technology';
```

I used NOT to filter out employees not in the IT department from the employees database.

```
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   |
```

Summary

To investigate the security issue at the large company, I used SQL to examine the log files from all of the East regional offices. Despite the size, I was able to use SQL filters to make the investigation move swiftly.

- It was found that there were 19 failed login attempts after 18:00 on 2022-05-08 & 2022-05-09
- It was determined that the login attempts were made outside of Mexico
- Patching needs to be proposed to the change management team

This document describes how the tables used for this mock scenario are organized. The `organization` database contains the following two tables:

- `log_in_attempts`
- `employees`

log_in_attempts

The `log_in_attempts` table has the following columns:

- `event_id`: The identification number assigned to each login event
- `username`: The username of the employee
- `login_date`: The date the login attempt was recorded
- `login_time`: The time the login attempt was recorded
- `country`: The country where the login attempt occurred
- `ip_address`: The IP address of that employee's machine
- `success`: The success of the login attempt; `FALSE` indicates a failed attempt

In the MariaDB shell, these columns are returned as:

```
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
```

employees

The `employees` table has the following columns:

- `employee_id`: The identification number assigned to each employee
- `device_id`: The identification number assigned to each device used by the employee
- `username`: The username of the employee
- `department`: The department the employee is in
- `office`: The office the employee is located in

In the MariaDB shell, these columns are returned as:

```
+-------------+--------------+----------+------------------------+-------------+
| employee_id | device_id    | username | department             | office      |
+-------------+--------------+----------+------------------------+-------------+
```