Apply Filters to SQL Queries

Project Description

As part of my role as a security professional in my organization, I was tasked to investigate all potential security issues to ensure system safety. The following SQL queries with filters were completed to perform security-related tasks towards this goal.

Retrieve after hours failed login attempts

Due to a potential security incident that occurred after business hours (i.e. after 18:00), all failed login attempts needed to be investigated.

```
MariaDB [organization]> SELECT * FROM log_in_attempts WHERE login_time > '18:00' AND success = 0;
+----------+----------+------------+------------+---------+-----------------+---------+
| 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 |
```

Using a SQL query of the `log_in_attempts` table, I filtered my results to include only login times greater than (i.e. after) 18:00 from the `login_time` column in the table. The second condition `success=0` checks a boolean value to filter for only failed login attempts during this time.

Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. Therefore, we should filter for any login activity that happened on 2022-05-09 or on the day before.

```
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 |
```

The SQL query above shows the filter that was applied to search in the `login_time` column in the `log_in_attempts` table for the dates either on `2022-05-09` or

`2022-05-08`. The `OR` operator implies that both conditions do not need to be true (i.e. we are filtering for either day).

Retrieve login attempts outside of Mexico

While investigating further, our team suspects the suspicious activity with login attempts is likely occurring 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 |       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 |
```

The SQL query above returns all login attempts in countries other than Mexico. This is again completed by selecting all data from the `log_in_attempts` table. Since the data in the `country` column could be represented as `MEX` or `MEXICO`, we use the percentage sign (`%`) to represent any country that is not starting with the string `'MEX'` by combining this string with the `NOT` operator.

Retrieve employees in Marketing

Additionally, my team wants to perform security updates on specific employee machines in the Marketing department. To perform this task, I needed to retrieve the information on which employee machines to update via 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 |
|        1088 | k8651965m233  | rgosh    | Marketing  | East-157 |
```

The SQL query above filters for employee machines from employees in the Marketing department in the East building. We complete this by looking in the `employees` table and searching the `department` column for the `Marketing` department. Additionally, I was tasked to query for employees only in the East building, so I added the `office LIKE 'East%'` at

the end of my query to ensure that these employees were both in the Marketing department and in the East office, searching for office values starting with 'East' only.

Retrieve employees in Finance or Sales

After updating these employee's machines, my team now needs to perform a different security update on machines for employees in the Sales and Finance 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   |
|        1009 | NULL         | lrodriqu | Sales      | South-134   |
```

To complete this task, I queried in the table: `employees`, `WHERE` employees were either in `department = 'Marketing'` or `department = 'Sales'`. If either condition were true, we would retrieve this data from the SQL query and now know which machines for which employee to update.

Retrieve all employees not in IT

Finally I was tasked to update employee machines who are not in the Information Technology department, as employees in the IT department already had the update.

```
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   |
```

To complete this task, I created another SQL query to filter employee machines from employees not in the Information Technology department by searching for all data in the `employees` table, `WHERE` the employees were `NOT` in the IT department.

Summary

In conclusion, I applied filters to SQL queries to retrieve specific information on login attempts and employee machines, utilizing two different tables `employees` and `log_in_attempts`. These tables were filtered using `AND`, `OR`, and `NOT` operators as well as the `LIKE` operator in conjunction with the percent sign (`%`) to further refine the SQL queries.