Apply filters to SQL queries

Project description

My company is committed to enhancing system security. My role involves safeguarding the system, probing into possible security concerns, and keeping staff computers updated. Below are illustrations of how I employed SQL queries with filtering to execute security-related operations.

Retrieve after hours failed login attempts

A possible security event happened outside regular business hours (post-18:00). It's necessary to examine each failed login attempt made during this time.
Below is the code I used for a SQL query, designed to isolate and identify failed login attempts occurring after standard business hours.

```
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 |
```

In the screenshot, the initial segment showcases my query, while the latter part displays a sample of the results. This query is tailored to single out failed login attempts post 18:00. I commenced by extracting all data from the log_in_attempts table. Subsequently, a WHERE clause combined with an AND operator was employed to refine the results, focusing exclusively on login attempts post 18:00 that were not successful. The first criterion, login_time > '18:00', isolates login attempts after 18:00. The second criterion, success = FALSE, targets only the unsuccessful login attempts.

Retrieve login attempts on specific dates

An unusual incident was detected on 2022-05-09. It is necessary to scrutinize all login activities from 2022-05-09 and the preceding day.
Below is the code I used for a SQL query, aimed at filtering login attempts on designated 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 |
```

The screenshot's first segment displays my query, while the second shows a sample of the output. This query identifies all login attempts made on either 2022-05-09 or 2022-05-08. Initially, I selected all records from the log_in_attempts table. Next, I employed a WHERE clause with an OR operator to refine the results, isolating only the login attempts from 2022-05-09 or 2022-05-08. The first criterion, login_date = '2022-05-09', isolates logins on 2022-05-09. The second, login_date = '2022-05-08', targets logins on 2022-05-08.

Retrieve login attempts outside of Mexico

Upon examining the company's data regarding login attempts, it appears there's a concern with attempts made from locations outside Mexico. These instances warrant further investigation. Below is the code I wrote for a SQL query, designed specifically to identify login attempts originating from outside 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 |
```

In the screenshot, the first section presents my query, while the second displays a segment of the results. This query is designed to retrieve login attempts from countries excluding Mexico. Initially, I retrieved all data from the log_in_attempts table. Following this, a WHERE clause combined with NOT was applied to focus on locations outside Mexico. I utilized the LIKE operator with the pattern 'MEX%', as our dataset denotes Mexico with both 'MEX' and 'MEXICO'. The '%' symbol in conjunction with LIKE signifies a wildcard for an arbitrary sequence of characters.

Retrieve employees in Marketing

My team is tasked with upgrading computers for select staff in the Marketing department. For this purpose, I need to identify the specific employee computers requiring updates.
Below is the code I composed for a SQL query, aimed at filtering out employee machines belonging to staff in the Marketing department 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 |
```

Query for Marketing Department in East Building
The first segment of the screenshot shows my query, and the second part reveals a section of the output. This query identifies all Marketing department employees located in the East building. I began by extracting data from the employees table. Next, a WHERE clause with AND was utilized to pinpoint employees in the Marketing department and those in the East building. The pattern 'East%' was used with LIKE, matching office data that indicates the East building with a specific office number. The criteria include 'department = 'Marketing'' for selecting Marketing department employees, and 'office LIKE 'East%'' to isolate those in the East building.

Retrieve employees in Finance or Sales

Computers of employees in the Finance and Sales departments require updates for a different security measure. To prepare, I need data on employees from these 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   |
```

The code shown initially is my query, followed by a portion of the output. This query fetches all employees from the Finance and Sales departments. After selecting data from the employees table, I applied a WHERE clause with OR to filter employees from either the Finance or Sales departments. The OR operator was chosen over AND to include employees from either department. The conditions are 'department = 'Finance'' for Finance department employees, and 'department = 'Sales'' for Sales department employees.

Retrieve all employees not in IT

A final security update is needed for employees outside 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   |
```

The initial part of the screenshot presents my query, with the latter part showing some of the output. The query locates all employees not in the Information Technology department. It begins with selecting all data from the employees table, followed by a WHERE clause with NOT to exclude employees from this department.

Summary

II applied various filters to SQL queries for specific data on login attempts and employee machines, using two distinct tables: log_in_attempts and employees. By employing AND, OR, and NOT operators, I efficiently filtered the necessary information for each task. Additionally, the LIKE operator and '%' wildcard were instrumental in filtering for specific patterns.