Apply filters to SQL queries

Project description

My organization is focused on enhancing our system's security. My role involves safeguarding the system, investigating any potential security threats, and updating employee computers as necessary. Here are some examples of how I utilized SQL with filters to carry out security-related tasks.

Retrieve after hours failed login attempts

A potential security incident occurred after business hours, specifically after 6:00 PM. It is necessary to investigate all failed login attempts that happened during this time.

The following SQL query demonstrates how I filtered for these after-hours failed login attempts.

```
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 screenshot shows my SQL query, and the second part displays a portion of the results. This query identifies failed login attempts that happened after 6:00 PM. I began by selecting all data from the `log_in_attempts` table. Then, I applied a WHERE clause with an AND operator to filter the results, so it only includes login attempts after 6:00 PM that were unsuccessful. The first condition, `login_time > '18:00'`, filters for attempts after 6:00 PM. The second condition, `success = FALSE`, filters for failed login attempts.

Retrieve login attempts on specific dates

A suspicious event took place on May 9, 2022. We need to investigate any login activity from that day or the day before.

The following SQL query shows how I filtered login attempts for these 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 |
```

The first part of the screenshot shows my SQL query, and the second part displays a portion of the results. This query retrieves all login attempts that happened on May 9, 2022, or May 8, 2022. I started by selecting all data from the `log_in_attempts` table. Then, I used a WHERE clause with an OR operator to filter the results, so it includes only login attempts from those two dates. The first condition, `login_date = '2022-05-09'`, filters for logins on May 9, 2022. The second condition, `login_date = '2022-05-08'`, filters for logins on May 8, 2022.

Retrieve login attempts outside of Mexico

After examining the organization's data on login attempts, I suspect there may be a problem with those that occurred outside of Mexico. These attempts need further investigation.

The following SQL query shows how I filtered for login attempts that took place 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 |
```

I queried the log_in_attempts table to retrieve all login attempts excluding those from Mexico. I applied a WHERE clause with NOT to filter out entries where the country code didn't match 'MEX%'—'MEX%' matches both 'MEX' and 'MEXICO' in the dataset, using the '%' wildcard to represent any sequence of characters.

Retrieve employees in Marketing

My team aims to upgrade computers for selected employees within the Marketing department. To proceed, I need to gather details about which employee machines require updates.

Below is the SQL query I developed to filter for employee machines used by individuals 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 |
```

The screenshot consists of my query and a segment of its output. This query retrieves all employees located in the Marketing department within the East building. Initially, I selected all data from the employees table. Then, I applied a WHERE clause using AND to filter for employees who belong to the Marketing department and are situated in the East building. I utilized LIKE with 'East%' as the pattern for matching because the office column denotes the East building along with specific office numbers. The first condition 'department = 'Marketing'' filters for Marketing department employees, while the second condition 'office LIKE 'East%'' filters for those in the East building.

Retrieve employees in Finance or Sales

We also need to update machines for employees in the Finance and Sales departments. Because these departments require a different security update, I need to gather information specifically for employees in these two departments.

Below is the SQL query I used to filter for employee machines from the Finance or 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   |
```

The screenshot shows my query and part of its output. This query retrieves employees from the Finance and Sales departments. Initially, I selected all data from the employees table. Then, I applied a WHERE clause using OR to filter for employees in either the Finance or Sales departments. I opted for the OR operator instead of AND because I wanted to include all employees from either department. The first condition 'department = 'Finance'' filters for employees in the Finance department, while the second condition 'department = 'Sales'' filters for employees in the Sales department.

Retrieve all employees not in IT

We need to apply an additional security update to employees who are not part of the Information Technology department. To begin, I need to gather information about these employees.

Below is the SQL query I used to filter for employee machines belonging to employees outside of 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 screenshot includes my query and a segment of its output. This query retrieves employees who are not part of the Information Technology department. Initially, I selected all data from the employees table. Then, I applied a WHERE clause using NOT to filter out employees from this department.

Summary

I utilized SQL filters to extract targeted details from login attempts and employee machine data stored in two tables: log_in_attempts and employees. Employing operators such as AND, OR, and NOT enabled me to refine queries for distinct information requirements. Additionally, I employed the LIKE operator along with the '%' wildcard to filter data based on specific patterns.