Apply filters to SQL queries

Project description

My organization is enhancing system security, and my role is to ensure its safety by identifying potential security issues and updating employee computers as needed. The following steps demonstrate how I used SQL filters to perform various security-related tasks.

Retrieve after hours failed login attempts

A potential security incident occurred after business hours (after 18:00). All failed login attempts during this time need to be investigated. The following code demonstrates how I created a SQL query to filter for failed login attempts that occurred after business hours.

```
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 41
Server version: 10.3.39-MariaDB-0+deb10u2 Debian 10

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input stat
ement.

MariaDB [organization]> clear
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      | 192.168.66.1
42  |       0 |
|       20 | tshah    | 2022-05-12 | 18:56:36   | MEXICO  | 192.168.109.
50  |       0 |
|       28 | aestrada | 2022-05-09 | 19:28:12   | MEXICO  | 192.168.27.5
```

The first part of the screenshot shows my query, while the second part displays a portion of the output. This query filters for failed login attempts that occurred after 18:00.

I began by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause with an `AND` operator to refine the results, ensuring that only login attempts after 18:00 and unsuccessful attempts were included.

- The condition `login_time > '18:00'` filters for login attempts that occurred after 18:00.
- The condition `success = FALSE` filters for failed login attempts.

Retrieve login attempts on specific dates

A suspicious event occurred on **2022-05-09**, and any login activity on **2022-05-09** or the day before needs to be investigated.

The following code demonstrates how I created a SQL query to filter for login attempts that occurred on 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 |       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 |
```

The first part of the screenshot shows my query, while the second part displays a portion of the output. This query retrieves all login attempts that occurred on **2022-05-09** or **2022-05-08**.

I began by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause with an `OR` operator to filter the results, ensuring only login attempts from the specified dates were included.

- The condition `login_date = '2022-05-09'` filters for logins on **2022-05-09**.
- The condition `login_date = '2022-05-08'` filters for logins on **2022-05-08**.

Retrieve login attempts outside of Mexico

After analyzing the organization's login attempt data, I found a potential issue with login attempts made outside of Mexico. These attempts need further investigation.

The following code demonstrates how I created a SQL query to filter for login attempts that occurred 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 |
```

The first part of the screenshot shows my query, while the second part displays a portion of the output. This query retrieves all login attempts that occurred in countries other than Mexico.

I began by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause with `NOT` to filter out records where the country is Mexico.

- I used `LIKE 'MEX%'` as the pattern to match because the dataset represents Mexico as both MEX and MEXICO.
- The percentage sign (%) acts as a wildcard, allowing the query to match any variation of "MEX" .

Retrieve employees in Marketing

My team wants to update the computers for certain employees in the Marketing department. To do this, I need to gather information on which employee machines to update.

The following code demonstrates how I created a SQL query to filter for employee machines from employees 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 first part of the screenshot shows my query, while the second part displays a portion of the output. This query retrieves all employees in the Marketing department who are located in the East building.

I started by selecting all data from the `employees` table. Then, I used a `WHERE` clause with `AND` to filter for employees who work in the Marketing department and are in the East building.

- The condition `department = 'Marketing'` filters for employees in the Marketing department.
- The condition `office LIKE 'East%'` filters for employees in the East building, as the office column includes both the building name and the specific office number.

Retrieve employees in Finance or Sales

The machines for employees in the Finance and Sales departments also need to be updated. Since a different security update is required, I need to gather information only on employees from these two departments.

The following code demonstrates how I created a SQL query to filter for employee machines from employees in 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 first part of the screenshot shows my query, while the second part displays a portion of the output. This query retrieves all employees in the Finance and Sales departments.

I began by selecting all data from the `employees` table. Then, I used a `WHERE` clause with the `OR` operator to filter for employees who are in either the Finance or Sales department. I used `OR` instead of `AND` because I want to include employees from either department.

- The condition `department = 'Finance'` filters for employees from the Finance department.
- The condition `department = 'Sales'` filters for employees from the Sales department.

Retrieve all employees not in IT

My team needs to make one more security update for employees who are not in the Information Technology department. To proceed, I first need to gather information on these employees.

The following demonstrates how I created a SQL query to filter for employee machines from employees not in 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 first part of the screenshot shows my query, while the second part displays a portion of the output. This query retrieves all employees who are not in the Information Technology department.

I began by selecting all data from the `employees` table. Then, I used a `WHERE` clause with `NOT` to filter for employees who are not in the Information Technology department.

Summary

I applied filters to SQL queries to retrieve specific information on login attempts and employee machines. I worked with two different tables: `log_in_attempts` and `employees`. To narrow down the results, I used the `AND`, `OR`, and `NOT` operators. Additionally, I used the `LIKE` operator with the percentage sign (%) wildcard to filter for patterns as needed for each task.