Apply filters to SQL queries

Project description

My organization is actively enhancing system security, and I am tasked with safeguarding it by investigating potential security issues and updating employee computers as required. The subsequent steps illustrate how I employed SQL with filters to execute security-related actions.

Retrieve after hours failed login attempts

A potential security incident occurred outside of business hours, specifically after 18:00. It is imperative to investigate all failed login attempts during this time period. The following code exemplifies how I formulated a SQL query to filter for failed login attempts that transpired after 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 |
```

My query filters for failed login attempts occurring after 18:00. Initially, I selected all data from the log_in_attempts table. Then, I utilized a WHERE clause with an AND operator to narrow down results to failed login attempts after 18:00. The condition login_time > '18:00' filters for attempts post-18:00, while success = FALSE isolates failed attempts.

Retrieve login attempts on specific dates

A suspicious event unfolded on 2022-05-09, prompting investigation of login activity on that date and the preceding day. The subsequent code showcases my SQL query designed to filter login attempts based on 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 query retrieves all login attempts that transpired on either 2022-05-09 or 2022-05-08. It begins by selecting all data from the log_in_attempts table. Subsequently, a WHERE clause incorporating an OR operator is employed to filter the results, isolating login attempts from either of the specified dates. The first condition, login_date = '2022-05-09', extracts logins on 2022-05-09, while the second condition, login_date = '2022-05-08', identifies logins on 2022-05-08.

Retrieve login attempts outside of Mexico

After scrutinizing the organization's login attempt data, I've identified a concern regarding login attempts originating outside of Mexico. These attempts require investigation. The subsequent code illustrates my SQL query formulated to filter for login attempts 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 |       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 |
```

The query retrieves all login attempts originating from countries other than Mexico. Initially, I selected all data from the log_in_attempts table. Subsequently, a WHERE clause with NOT was employed to filter countries other than Mexico. Utilizing LIKE with the pattern MEX% was necessary to match both "MEX" and "MEXICO" representations in the dataset. The percentage sign (%) acts as a wildcard, representing any number of unspecified characters when used with LIKE.

Retrieve employees in Marketing

To facilitate the update of computers for specific employees in the Marketing department, I generated a SQL query to extract information on employee machines. The following code exemplifies this query, filtering for employee machines belonging to 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 query retrieves all employees in the Marketing department situated in the East building. Initially, I selected all data from the employees table. Subsequently, a WHERE clause with an AND operator was employed to filter employees working in both the Marketing department and the East building. Utilizing LIKE with the pattern 'East%' was necessary to match the East building, considering the data in the office column represents it along with specific office numbers. The first condition, department = 'Marketing', isolates employees in the Marketing department, while the second condition, office LIKE 'East%', identifies employees in the East building.

Retrieve employees in Finance or Sales

To facilitate the update of machines for employees in the Finance and Sales departments, I crafted a SQL query to retrieve relevant information solely for these two departments. The following code exemplifies this query, filtering for employee machines belonging to employees in either 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 query retrieves all employees in the Finance and Sales departments. Initially, I selected all data from the employees table. Subsequently, a WHERE clause with an OR operator was utilized to filter employees belonging to either the Finance or Sales departments. The OR operator was chosen instead of AND to include all employees from either department. The first condition, department = 'Finance', filters for employees from the Finance department, while the second condition, department = 'Sales', filters for employees from the Sales department.

Retrieve all employees not in IT

To facilitate another security update on employees not belonging to the Information Technology department, I constructed a SQL query to gather relevant information on these employees. The following demonstrates this query, filtering 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 query fetches all employees not affiliated with the Information Technology department. Initially, I selected all data from the employees table. Subsequently, a WHERE clause with NOT was utilized to filter employees not belonging to this department.

Summary

I utilized SQL queries to apply filters and extract specific information from two distinct tables: log_in_attempts and employees. Employing operators such as AND, OR, and NOT enabled me to filter data according to specific criteria for each task. Additionally, I employed the LIKE operator along with the percentage sign (%) wildcard to filter for patterns within the data.