Apply filters to SQL queries

Project description

I am leading a security enhancement project in my organization, focusing on investigating and addressing potential security issues. My role includes updating employee computers for improved safety. I utilize SQL with filters for security tasks, employing queries to identify vulnerabilities, implement access controls, and assess databases. Through these measures, I aim to strengthen our system's security and ensure a proactive defense against evolving threats.

Retrieve after hours failed login attempts

There was a potential security incident that occurred after business hours (after 18:00). All after hours login attempts that failed 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.

```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE login_time > '18:00:00' 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 |
```

The first part of the screenshot is my query, and the second part is a portion of the output. I initiated the query by selecting all data from the `"log_in_attempts"` table. Subsequently, a `WHERE` clause was employed, incorporating an AND operator to refine the results. The first condition, `login_time > '18:00'`, was applied to isolate login

attempts after 18:00. The second condition, `success = 0 (FALSE)`, was then imposed to specifically target unsuccessful login attempts.

In summary, the query employs a two-fold filtering approach: first by time, focusing on attempts post 18:00, and secondly by outcome, concentrating on unsuccessful login attempts. This method ensures the extraction of relevant data pertaining to failed login activities occurring after the specified time.

Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. Any login activity that happened on 2022-05-09 or on 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 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 |
|        8 | bisles   | 2022-05-08 | 01:30:17   | US      | 192.168.119.173 |       0 |
```

I initiated the query by selecting all data from the "`log_in_attempts`" table. Subsequently, a `WHERE` clause was employed, incorporating an `OR` operator to refine the results. 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.

In summary, this query utilizes a logical OR operator to include login attempts that match either of the specified dates, providing a comprehensive view of login activities on 2022-05-09 or 2022-05-08.

Retrieve login attempts outside of Mexico

There's been suspicious activity with login attempts, but the team has determined that this activity didn't originate in Mexico. These login attempts should be investigated.

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 |
|        4 | dkot     | 2022-05-08 | 02:00:39   | USA     | 192.168.178.71  |       0 |
```

The first part of the screenshot is my query, and the second part is a portion of the output. I began by selecting all data from the `"log_in_attempts"` table. Following that, I utilized a `WHERE` clause with the `NOT` operator and a LIKE condition. The condition, `NOT country LIKE 'MEX%'`, filters the results to exclude entries where the country code starts with `'MEX'`. This ensures that only login attempts from countries other than those with codes starting with `'MEX'` are included in the output.

Retrieve employees in Marketing

My team wants to update the computers for certain employees in the Marketing department. To do this, I have to get 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 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 |
|        1088 | k8651965m233 | rgosh    | Marketing  | East-157 |
|        1103 | NULL         | randerss | Marketing  | East-460 |
```

The first part of the screenshot is my query, and the second part is a portion of the output. I initiated the query by selecting all data from the `"employees"` table. Then, I utilized a `WHERE` clause to filter the results based on two conditions. The first condition, `department = 'Marketing'`, narrows down the records to those within the Marketing department. The second condition, `office LIKE 'East%'`, further refines the results to include only those with offices starting with `'East'`. In summary, the query focuses on employees in the Marketing department with offices located in the East.

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 needed, I have to get information on employees only 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   |
|        1009 | NULL         | lrodriqu | Sales      | South-134   |
```

The first part of the screenshot is my query, and the second part is a portion of the output. Initiating the query by selecting all data from the `"employees"` table, I then utilized a `WHERE` clause with conditions. The first condition, `department = 'Finance'`, isolates records within the Finance department. The subsequent condition, `OR department = 'Sales'`, broadens the scope to include entries from the Sales department. In summary, the query is tailored to capture information about employees associated with either the Finance or Sales departments

Retrieve all employees not in IT

My team needs to make one more security update on employees who are not in the Information Technology department. To make the update, I first have to get 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   |
|        1003 | d394e816f943 | sgilmore | Finance         | South-153   |
|        1004 | e218f877g788 | eraab    | Human Resources | South-127   |
```

The first part of the screenshot is my query, and the second part is a portion of the output. I initiated by selecting all data from the `"employees"` table. Following that, I used a `WHERE` clause with the `NOT` operator and the condition `department = 'Information Technology'` to exclude records associated with the 'Information Technology' department. In essence, the query focuses on retrieving details of employees outside the 'Information Technology' department.

Summary

I applied filters to SQL queries to get specific information on login attempts and employee machines. I used two different tables, `log_in_attempts` and employees. I used the `AND`, `OR`, and `NOT` operators to filter for the specific information needed for each task. I also used `LIKE` and the percentage sign (`%`) wildcard to filter for patterns.