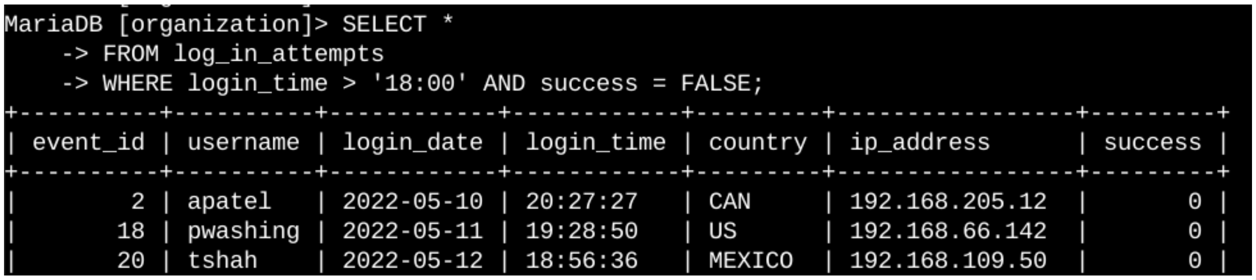
Apply filters to SQL queries

Project description

My organization is focused on enhancing system security, and my role involves ensuring the system remains secure, investigating potential security threats, and updating employee computers when necessary. Below are examples of how I utilized SQL with filters to carry out various security-related tasks.

Retrieve after hours failed login attempts

A potential security incident took place outside of regular business hours (after 6:00 PM). It is necessary to investigate all failed login attempts during this time.

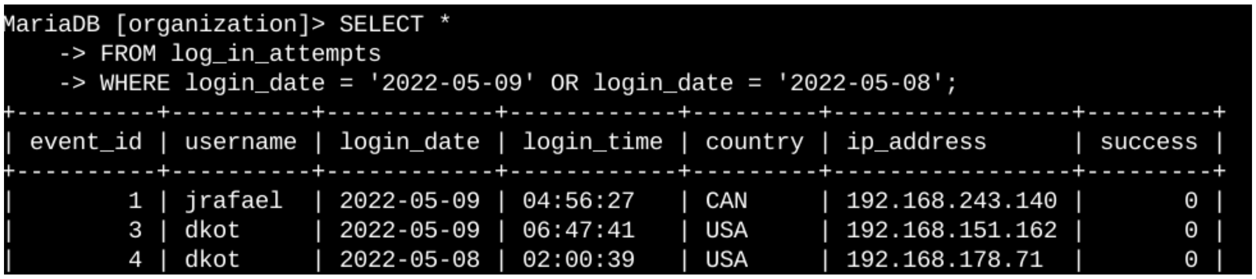


```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_time > '18:00' AND success = FALSE;
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 section of the screenshot shows my query, while the second section displays a portion of the output. This query identifies failed login attempts that occurred after 6:00 PM. To achieve this, I began by selecting all data from the `log_in_attempts` table. Next, I applied a `WHERE` clause combined with an `AND` operator to filter the results. The first condition, `login_time > '18:00'`, filters for login attempts made after 6:00 PM. The second condition, `success = FALSE`, filters for unsuccessful login attempts.

Retrieve login attempts on specific dates

A suspicious event took place on May 9, 2022. It is essential to investigate all login activity from that date and the day prior.

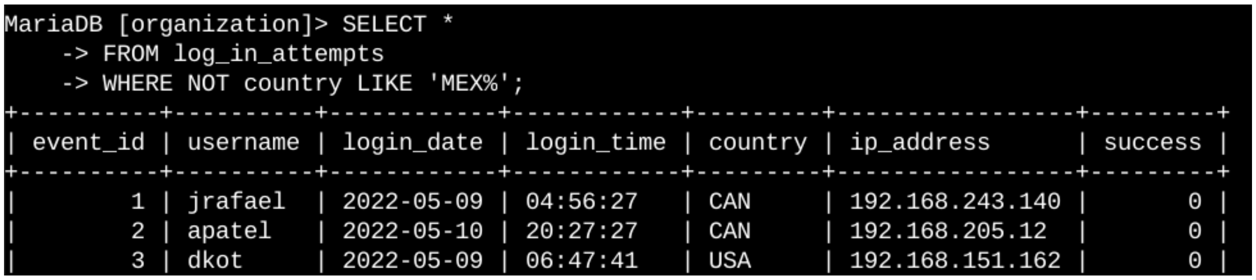


```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 section of the screenshot displays my query, while the second section shows a portion of the output. This query retrieves all login attempts made on May 9, 2022, or May 8, 2022. I started by selecting all data from the `log_in_attempts` table. Then, I applied a `WHERE` clause with an `OR` operator to filter the results to include only login attempts from those specific dates. The first condition, `login_date = '2022-05-09'`, filters for logins on May 9, 2022, while the second condition, `login_date = '2022-05-08'`, filters for logins on May 8, 2022.

Retrieve login attempts outside of Mexico

After reviewing the organization's login attempt data, I identified a potential issue with login attempts originating from outside of Mexico. These attempts require further investigation.

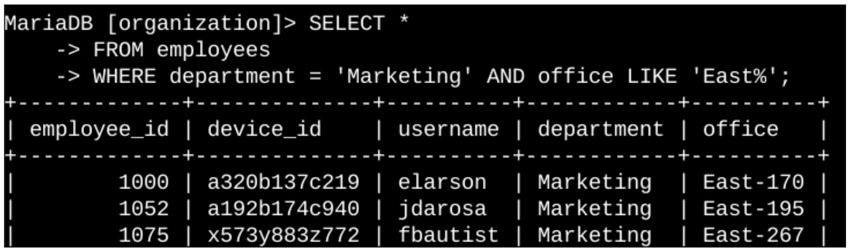


```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE NOT country LIKE 'MEX%';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 first section of the screenshot contains my query, while the second section shows a portion of the output. This query retrieves all login attempts from countries other than Mexico. I began by selecting all data from the `log_in_attempts` table. Then, I applied a `WHERE` clause with `NOT` to exclude Mexico from the results. To account for variations in how Mexico is represented in the dataset (`MEX` and `MEXICO`), I used the `LIKE` operator with the pattern `MEX%`. The percentage sign (`%`) serves as a wildcard to match any number of unspecified characters.

Retrieve employees in Marketing

My team plans to update the computers for specific employees in the Marketing department. To accomplish this, I need to gather information on which employee machines require updates.

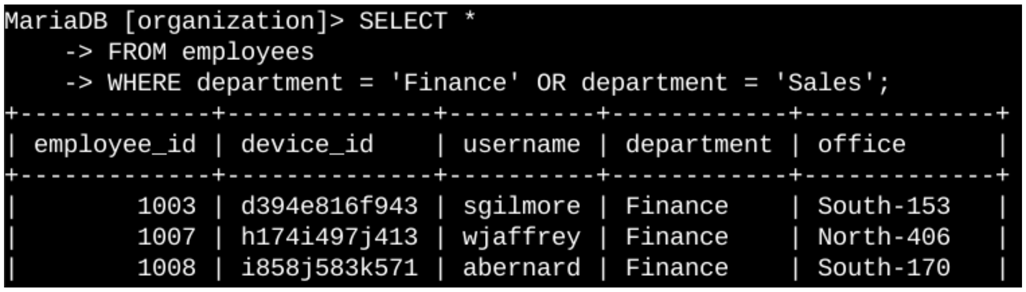


```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE department = 'Marketing' AND office LIKE 'East%';
+-------------+-------------+----------+------------+----------+
| employee_id | device_id   | username | department | office   |
+-------------+-------------+----------+------------+----------+
|        1000 | a320b137c219 | elarson  | Marketing  | East-170 |
|        1052 | a192b174c940 | jdarosa  | Marketing  | East-195 |
|        1075 | x573y883z772 | fbautist | Marketing  | East-267 |
```

The first section of the screenshot shows my query, while the second section displays a portion of the output. This query retrieves all employees in the Marketing department who are located in the East building. I began by selecting all data from the `employees` table. Next, I applied a `WHERE` clause with an `AND` operator to filter for employees working in the Marketing department and the East building. To account for the way the office data represents the East building (including specific office numbers), I used the `LIKE` operator with the pattern `East%`. The first condition, `department = 'Marketing'`, filters for employees in the Marketing department, while the second condition, `office LIKE 'East%'`, filters for those located in the East building.

Retrieve employees in Finance or Sales

The machines for employees in the Finance and Sales departments also require updates. However, since a different security update is needed, I must gather information specifically on employees from these two departments.



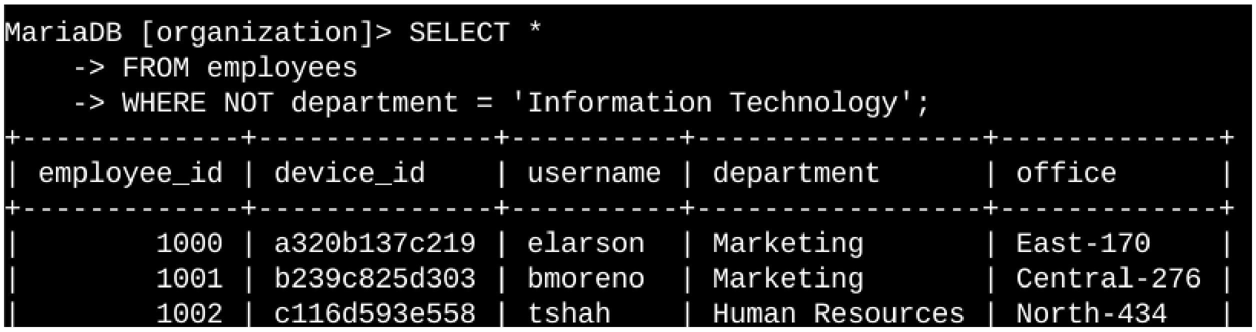
```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE department = 'Finance' OR department = 'Sales';
+-------------+-------------+----------+------------+-------------+
| employee_id | device_id   | username | department | office      |
+-------------+-------------+----------+------------+-------------+
|        1003 | d394e816f943 | sgilmore | Finance    | South-153   |
|        1007 | h174i497j413 | wjaffrey | Finance    | North-406   |
|        1008 | i858j583k571 | abernard | Finance    | South-170   |
```

The first section of the screenshot shows my query, while the second section 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 applied a `WHERE` clause with the `OR` operator to filter for employees in either the Finance or Sales department. I used `OR` instead of `AND` because I wanted to include employees from both departments. The first condition, `department = 'Finance'`, filters for employees in the Finance department, and the second condition, `department = 'Sales'`, filters for employees in the Sales department.

Retrieve all employees not in IT

My team needs to perform an additional security update for employees who are not in the Information Technology department. To do so, I first need to gather information on these employees.

The first section of the screenshot shows my query, and the second section 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 applied a `WHERE` clause with `NOT` to filter for employees outside of this 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   |
```

Summary

I applied filters to SQL queries to retrieve specific data on login attempts and employee machines, utilizing two distinct tables: `log_in_attempts` and `employees`. I employed the `AND`, `OR`, and `NOT` operators to refine the results for each task. Additionally, I used the `LIKE` operator combined with the percentage sign (%) wildcard to filter for specific patterns in the data.