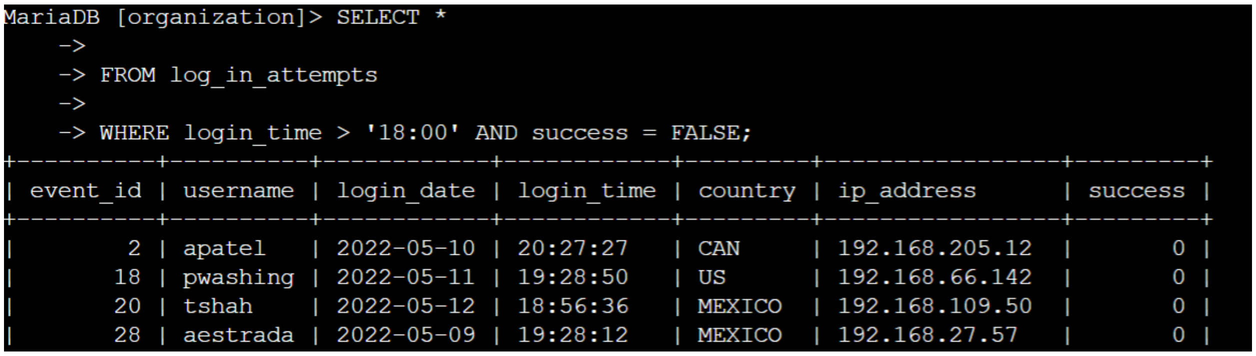
Apply filters to SQL queries

Project description

I'm a security professional at a large organization. Part of my job is to investigate security issues, update employee computers, and keep our systems secure. Recently, I noticed some suspicious login activity involving employee machines. To gain insight into the situation, I need to analyze relevant data. Below are examples of how I utilized SQL with filters to carry out security related tasks.

Retrieve after hours failed login attempts

I'm looking into failed login attempts that happened after business hours to check for any suspicious activity. To do this, I'll run an SQL query to pull all unsuccessful login attempts after 18:00 from the login activity logs.

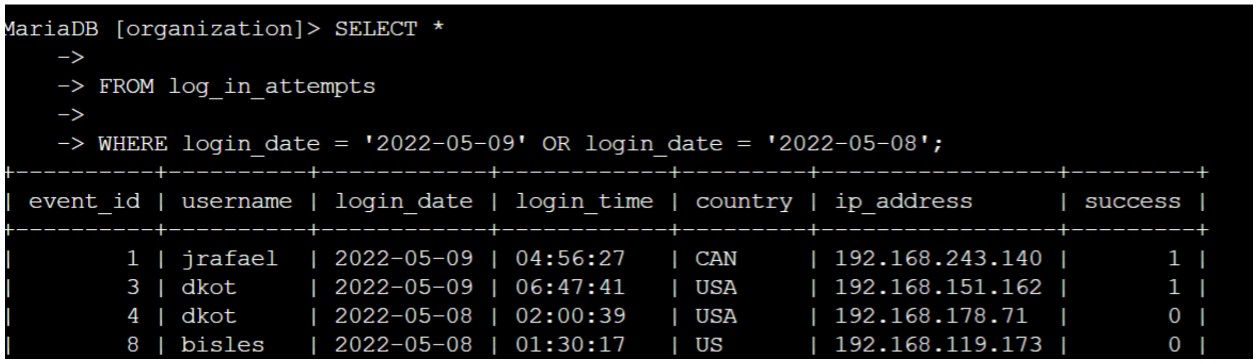


```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE login_time > '18:00' AND success = FALSE;
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 |
|       28 | aestrada | 2022-05-09 | 19:28:12   | MEXICO  | 192.168.27.57   |       0 |
```

This query filters for failed login attempts after 18:00 by selecting all data from the `log_in_attempts` table and using a `WHERE` clause with an `AND` operator. The first condition, `login_time > '18:00'`, ensures that only attempts after 18:00 are included, and the second condition, `success = FALSE`, filters for unsuccessful login attempts.

Retrieve login attempts on specific dates

My team is investigating a suspicious event that happened on May 9, 2022. I need to pull all login attempts from that day and the day before (May 8) to get a clearer picture of what occurred.

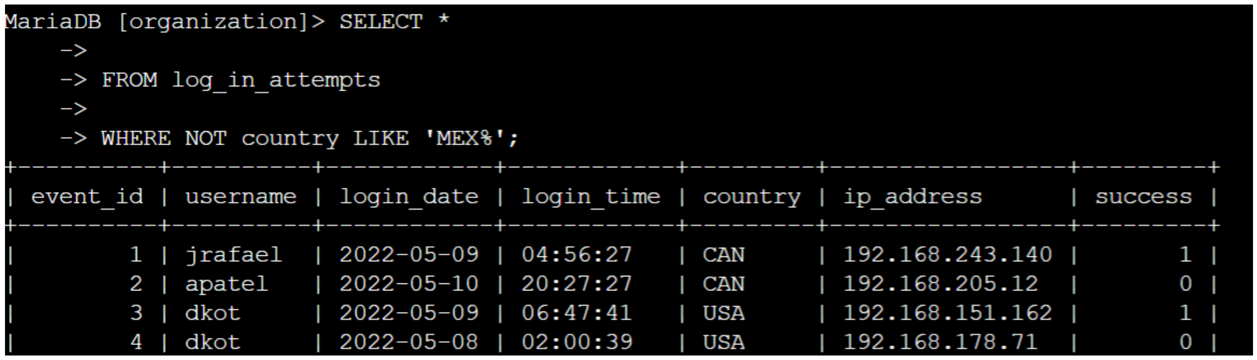


```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | 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 used the `WHERE` clause to filter out the login attempts that happened on either of those two dates. The query checks if the `login_date` is either '2022-05-09' or '2022-05-08'.

Retrieve login attempts outside of Mexico

There's been some unusual login activity, and the team confirmed it did not originate from Mexico. Now, I need to check all the login attempts that were made from outside Mexico.

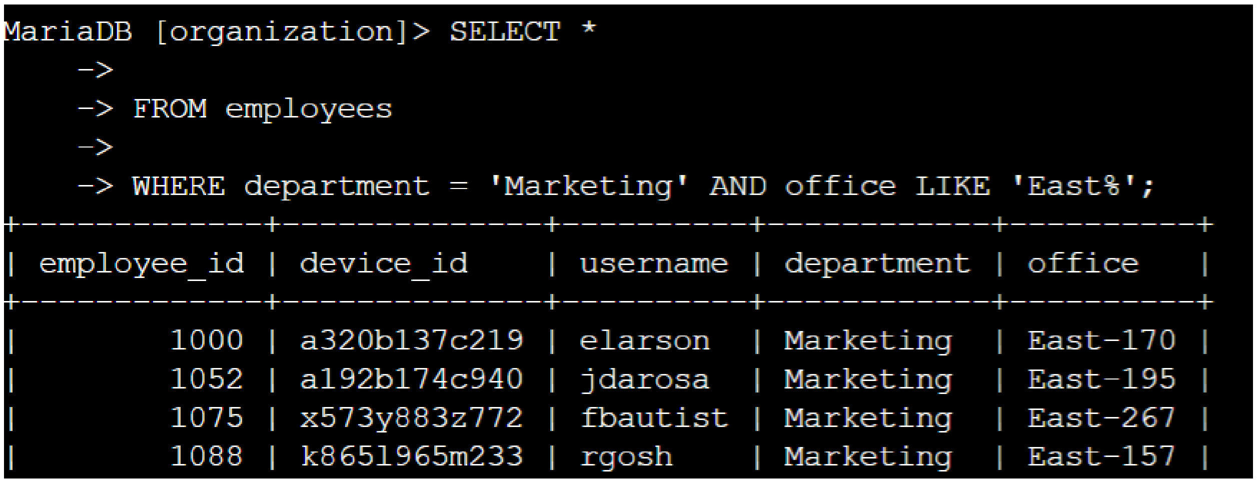


```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE NOT country LIKE 'MEX%';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 |
```

This SQL query filters login attempts that did not originate from Mexico by using the `NOT country LIKE 'MEX%'` condition. The `LIKE` operator, combined with the (`%`) wildcard, ensures that any variations starting with 'MEX', such as 'MEXICO' or 'MEXICAN REPUBLIC', are excluded. This allows the query to return only login attempts from outside of Mexico.

Retrieve employees in Marketing

Our team is preparing to perform security updates on specific employee machines within the 'Marketing' department, targeting those located in the 'East' building. I am responsible for gathering the necessary information. I'll be querying the 'employees' table using SQL. I'll construct a query that utilizes filters to identify all employees whose department is 'Marketing' and whose office is situated in the 'East' building. This will provide a comprehensive list of employees, enabling us to accurately target the security updates to the correct machines.

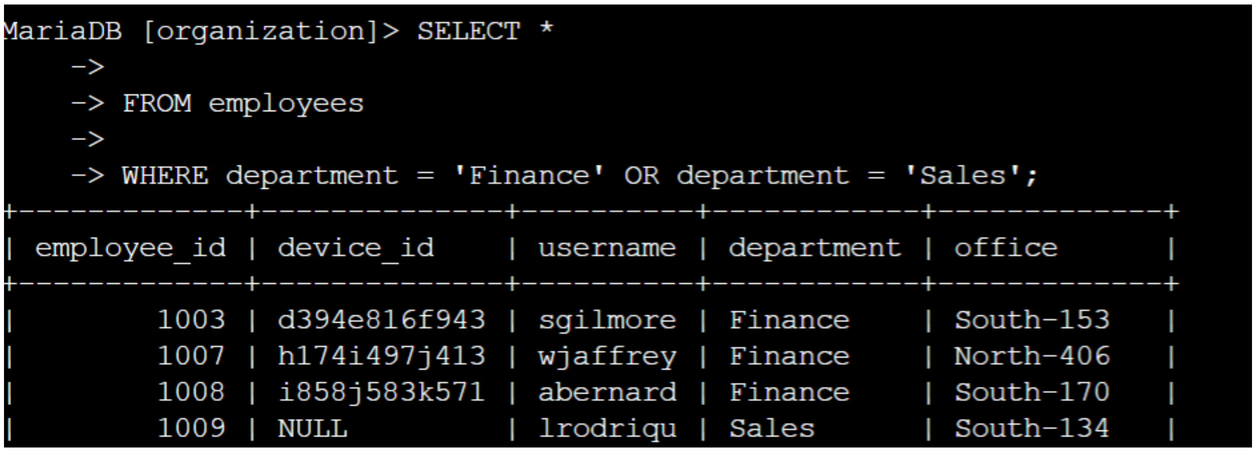


```
MariaDB [organization]> SELECT *
    ->
    -> FROM employees
    ->
    -> WHERE department = 'Marketing' AND office LIKE 'East%';
+-------------+-------------+----------+------------+----------+
| employee_id | device_id   | username | department | office   |
+-------------+-------------+----------+------------+----------+
|        1000 | a320b137c219 | elarson  | Marketing  | East-170 |
|        1052 | a192b174c940 | jdarosa  | Marketing  | East-195 |
|        1075 | x573y883z772 | fbautist | Marketing  | East-267 |
|        1088 | k8651965m233 | rgosh    | Marketing  | East-157 |
```

This SQL query filters employees in the 'Marketing' department with offices located in the 'East' building by using the `WHERE` clause with two conditions. The first condition, `department = 'Marketing'`, ensures that only employees in the 'Marketing' department are selected. The second condition, `office LIKE 'East%'`, filters for offices located in the 'East' building or any variation of that. The `AND` operator ensures that both conditions must be true for a record to be included in the result.

Retrieve employees in Finance or Sales

For the next computer update, we'll be targeting employees in the 'Finance' and 'Sales' departments. I'll be responsible for finding their employee details.

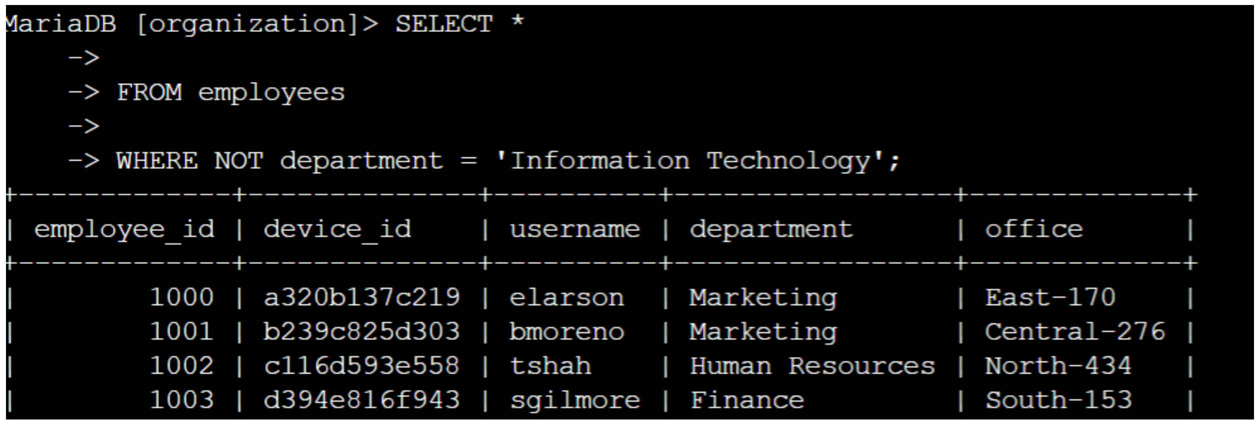


```
MariaDB [organization]> SELECT *
    ->
    -> FROM employees
    ->
    -> WHERE department = 'Finance' OR department = 'Sales';
+-------------+--------------+----------+------------+-------------+
| employee_id | device_id    | username | department | office      |
+-------------+--------------+----------+------------+-------------+
|        1003 | d394e816f943 | sgilmore | Finance    | South-153   |
|        1007 | h174i497j413 | wjaffrey | Finance    | North-406   |
|        1008 | i858j583k571 | abernard | Finance    | South-170   |
|        1009 | NULL         | lrodriqu | Sales      | South-134   |
```

This SQL query filters employees from the 'Finance' and 'Sales' departments by using the `OR` operator to apply two conditions. The first condition, `department = 'Finance'`, filters for employees in the 'Finance' department. The second condition, `department = 'Sales'`, filters for employees in the 'Sales' department. Using the `OR` operator ensures that employees from either department are included in the result set, providing a comprehensive list of employees from both departments. This allows the necessary data for targeted updates to be retrieved.

Retrieve all employees not in IT

Now the team still needs to carry out one final update. This has already been done for employees in the 'Information Technology' department. Now, the team needs details about employees in other departments to complete the process.



```
ariaDB [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   |
```

This SQL query filters out employees from the 'Information Technology' department by using the `WHERE` clause with the `NOT` operator. The condition `NOT department = 'Information Technology'` ensures that any employee in the 'Information Technology' department is excluded from the result. The query retrieves all columns (`*`) from the 'employees' table, returning a list of employees who belong to any department other than Information Technology.

Summary

During this investigation, I used SQL queries to analyze login attempts and employee data, helping identify potential security issues. By filtering information from the `log_in_attempts` and `employees` tables, I was able to focus on relevant records. I applied operators like `AND`, `OR`, and `NOT` to refine searches and used the `LIKE` operator with the (`%`) wildcard to detect patterns. These queries provided key insights into login activity and employee machines, supporting efforts to strengthen system security.