Apply filters to SQL queries

Project description

In this scenario, my organization is committed to enhancing system security, and I'm tasked with the responsibility of safeguarding the system, investigating potential security concerns, and making necessary updates to employee computers. The following steps outline instances where I utilized SQL with filters to execute security-related actions.

Retrieve after hours failed login attempts

In the provided screenshot, you'll find my query and a segment of the resulting output. This query is designed to isolate failed login attempts that took place after 18:00. To achieve this, I began by selecting all data from the 'log_in_attempts' table. Then, I applied a WHERE clause with an AND operator to narrow down the results, showing only those login attempts that occurred after 18:00 and were unsuccessful. The first condition, 'login_time > '18:00',' filters for attempts after 18:00, while the second condition, 'success = 0,' where 0 represents a failed login attempt.

```
MariaDB [organization]> clear
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE login_time > '18:00' 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 |
|       34 | drosas   | 2022-05-11 | 21:02:04   | US      | 192.168.45.93   |       0 |
|       42 | cgriffin | 2022-05-09 | 23:04:05   | US      | 192.168.4.157   |       0 |
|       52 | cjackson | 2022-05-10 | 22:07:07   | CAN     | 192.168.58.57   |       0 |
|       69 | wjaffrey | 2022-05-11 | 19:55:15   | USA     | 192.168.100.17  |       0 |
|       82 | abernard | 2022-05-12 | 23:38:46   | MEX     | 192.168.234.49  |       0 |
|       87 | apatel   | 2022-05-08 | 22:38:31   | CANADA  | 192.168.132.153 |       0 |
|       96 | ivelasco | 2022-05-09 | 22:36:36   | CAN     | 192.168.84.194  |       0 |
|      104 | asundara | 2022-05-11 | 18:38:07   | US      | 192.168.96.200  |       0 |
|      107 | bisles   | 2022-05-12 | 20:25:57   | USA     | 192.168.116.187 |       0 |
|      111 | aestrada | 2022-05-10 | 22:00:26   | MEXICO  | 192.168.76.27   |       0 |
|      127 | abellmas | 2022-05-09 | 21:20:51   | CANADA  | 192.168.70.122  |       0 |
|      131 | bisles   | 2022-05-09 | 20:03:55   | US      | 192.168.113.171 |       0 |
|      155 | cgriffin | 2022-05-12 | 22:18:42   | USA     | 192.168.236.176 |       0 |
|      160 | jclark   | 2022-05-10 | 20:49:00   | CANADA  | 192.168.214.49  |       0 |
|      199 | yappiah  | 2022-05-11 | 19:34:48   | MEXICO  | 192.168.44.232  |       0 |
+----------+----------+------------+------------+---------+-----------------+---------+
19 rows in set (0.013 sec)
```

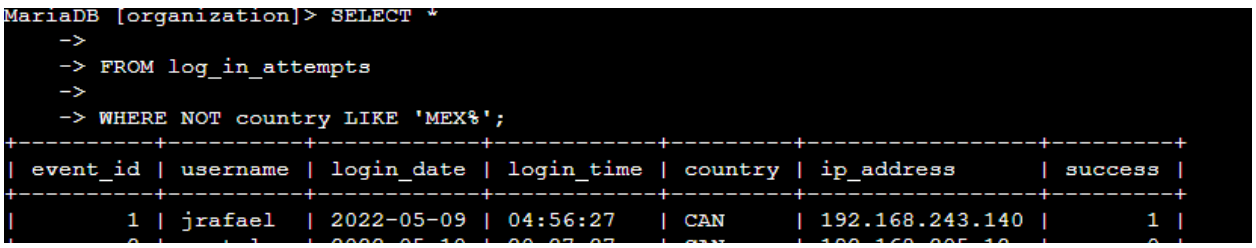
Retrieve login attempts on specific dates

In the screenshot provided, you'll find my query and a segment of the resulting output. This query retrieves all login attempts that took place on either 2022-05-09 or 2022-05-08. To achieve this, I initiated the process by selecting all data from the 'log_in_attempts' table. Subsequently, I applied a WHERE clause with an OR operator to refine the results, displaying only those login attempts that occurred on either 2022-05-09 or 2022-05-08. The first condition, 'login_date = '2022-05-08',' filters for logins on the 8th of May 2022, while the second condition, 'login_date = '2022-05-09',' filters for logins on the 9th of May 2022.There were a total of 75 login attempts.

```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE login_date = '2022-05-8' OR login_date = '2022-05-09';
+----------+----------+------------+------------+---------+-----------------+---------+
| 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 |
|       12 | dkot     | 2022-05-08 | 09:11:34   | USA     | 192.168.100.158 |       1 |
|       15 | lyamamot | 2022-05-09 | 17:17:26   | USA     | 192.168.183.51  |       0 |
|       24 | arusso   | 2022-05-09 | 06:49:39   | MEXICO  | 192.168.171.192 |       1 |
|      190 | jsoto    | 2022-05-09 | 05:09:21   | USA     | 192.168.25.60   |       0 |
|      191 | cjackson | 2022-05-08 | 06:46:07   | CANADA  | 192.168.7.187   |       0 |
|      193 | lrodriqu | 2022-05-08 | 07:11:29   | US      | 192.168.125.240 |       0 |
|      197 | jsoto    | 2022-05-08 | 09:05:09   | US      | 192.168.36.21   |       0 |
+----------+----------+------------+------------+---------+-----------------+---------+
75 rows in set (0.001 sec)
```

Retrieve login attempts outside of Mexico

In the provided screenshot, you'll see my query and a segment of the output it generated. This query is designed to retrieve all login attempts originating from countries other than Mexico. The process began by selecting all data from the 'log_in_attempts' table. Then, I utilized a WHERE clause with 'NOT' to filter out Mexico. To achieve this, I employed 'LIKE' with the pattern 'MEX%' because the dataset represents Mexico as 'MEX' and 'MEXICO.' The '%' symbol is used with 'LIKE' to match any number of unspecified characters. There were a total of 144 login attempts 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 |
```

```
|      200 | jclark    | 2022-05-12 | 01:11:45   | CANADA   | 192.168.91.103  |          1 |
+----------+-----------+------------+------------+----------+-----------------+------------+
144 rows in set (0.001 sec)
```

Retrieve employees in Marketing

In the screenshot provided, you can see my query and a portion of the resulting output. This query aims to retrieve all employees located in the East building who are part of the Marketing department. The process began by selecting all data from the 'employees' table. I then utilized a WHERE clause with 'AND' to filter for employees meeting both criteria: working in the Marketing department and being situated in the East building. To identify the East building, I used 'LIKE' with the pattern 'East%' in the 'office' column, as it represents the East building with specific office numbers. The first condition, 'department = 'Marketing',' filters for Marketing department employees, while the second condition, 'office LIKE 'East%',' filters for those in the East building.

```
MariaDB [organization]> SELECT *
    ->
    -> FROM employees
    -> WHERE department LIKE '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 |
|        1156 | a184b775c707 | dellery  | Marketing  | East-417 |
|        1163 | h679i515j339 | cwilliam | Marketing  | East-216 |
+-------------+-------------+----------+------------+----------+
7 rows in set (0.001 sec)
```

Retrieve employees in Finance or Sales

In the screenshot provided, you'll find my query and a portion of the resulting output. This query's purpose is to retrieve all employees belonging to either the Finance or Sales departments. To achieve this, I began by selecting all data from the 'employees' table. Next, I employed a WHERE clause with 'OR' to filter for employees who are members of either the Finance or Sales departments. The choice of 'OR' instead of 'AND' is deliberate, as it ensures that all employees from either department are included. The first condition, 'department = 'Finance',' targets Finance department employees, while the second condition, 'department = 'Sales',' targets Sales department employees.

```
MariaDB [organization]> clear
MariaDB [organization]> SELECT*
    -> FROM employees
    -> WHERE department LIKE 'Finance' OR Department LIKE '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   |
|        1199 | h518b859p957 | drainier | Finance    | East-348    |
+-------------+--------------+----------+------------+-------------+
71 rows in set (0.017 sec)
```

Retrieve all employees not in IT

In the provided screenshot, you'll see my query and a segment of the resulting output. This query is intended to retrieve all employees who are not part of the Information Technology department. To accomplish this, I began by selecting all data from the 'employees' table. Subsequently, I utilized a WHERE clause with 'NOT' to filter out employees who do not belong to 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 |
|        1199 | r520s571t459 | areyes   | Human Resources | East-100    |
+-------------+--------------+----------+-----------------+-------------+
161 rows in set (0.001 sec)
```

Summary

I utilized SQL queries to extract precise details regarding login attempts and employee machines by applying filters to two distinct tables, 'log_in_attempts' and 'employees.' Throughout these tasks, I employed various operators such as AND, OR, and NOT to refine the data selection based on specific criteria. Additionally, I used the LIKE operator in conjunction with the '%' wildcard to filter for patterns within the data.