Apply filters to SQL queries

Project description

My organisation is working to make their system more secure. It is my job to ensure the system is safe, investigate all potential security issues, and update employee computers as needed. The following steps provide examples of how I used SQL with filters to perform security-related tasks.

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.

```
MariaDB [organization]> select * from log_in_attempts where success = '0' and login_time > '18:00';
+----------+----------+------------+------------+---------+-----------------+---------+
| 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 (0.001 sec)

MariaDB [organization]>
```

The first line in the screenshot above is the query I used in order to filter all of the failed login attempts after 6pm. The query can be broken down as follows:

- SELECT * FROM log_in_attempts - selects all entries from the log_in_attempts table.
- WHERE success = '0' - filters out successful login attempts and shows only failed attempts.
- AND login_time > '18:00'; - additional filter (uses AND keyword to connect filters) to retrieve login attempts that took place after 6pm.

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.

```
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 |
|       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 |
|       25 | sbaelish | 2022-05-09 | 07:04:02   | US      | 192.168.33.137  |       1 |
|       26 | apatel   | 2022-05-08 | 17:27:00   | CANADA  | 192.168.123.105 |       1 |
|       28 | aestrada | 2022-05-09 | 19:28:12   | MEXICO  | 192.168.27.57   |       0 |
```

The screenshot above shows the result of a query which fetches all login attempts that happened on the specified date as well as the day before. It can be broken down as follows:

- SELECT * FROM log_in_attempts - selects all entries from the log_in_attempts table.
- WHERE login_date = ‘2022-05-09’ - selects login attempts that occurred on that date.
- OR login_date = ‘2022-05-08’; - OR keyword is used to select attempts that happened on either day.

Retrieve login attempts outside of Mexico

After investigating the organization’s data on login attempts, I believe there is an issue with the login attempts that occurred outside of Mexico. These login attempts should be investigated.

```
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 |
|        5 | jrafael  | 2022-05-11 | 03:05:59   | CANADA  | 192.168.86.232  |       0 |
|        7 | eraab    | 2022-05-11 | 01:45:14   | CAN     | 192.168.170.243 |       1 |
```

The screenshot above shows the result of a query which fetches all login attempts that occurred from locations outside of Mexico. It can be broken down as follows:

- SELECT * FROM log_in_attempts - selects all entries from the log_in_attempts table.
- WHERE NOT country LIKE ‘MEX%’ - selects only the rows in which the country column does not contain the prefix ‘MEX’. The ‘%’ wildcard is used at the end of the string along with the LIKE keyword to search for any sequence as long as it starts with ‘MEX’.

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.

```
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 |
```

The screenshot above shows a query that retrieves all employees in the marketing department in the east office. It can be broken down as follows:

- SELECT * FROM log_in_attempts - selects all entries from the log_in_attempts table.
- WHERE department = 'Marketing' - selects only the rows in which the department column contains the string 'Marketing'.
- AND office LIKE 'East%'; - a second filter is applied as well as the first to select rows where the office column starts with the string 'East'. The '%' wildcard is used again.

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.

```
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 above retrieves all employees whose department is either finance or sales. It can be broken down as follows:

- SELECT * FROM log_in_attempts - selects all entries from the log_in_attempts table.
- WHERE department = 'Finance' - selects only the rows in which the department column contains the string 'Finance'.
- OR department = 'Sales'; - the OR keyword is used to retrieve the rows in which the department column contains the string 'Sales'. The returned table consists of all rows where either condition is met.

Retrieve all employees not in IT

My team needs to make one more security update on employees who are not in the IT department. To make the update, I first have to get information on these employees.

```
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   |
```

The query above allows me to retrieve all rows which the department column is not Information Technology. It can be broken down as follows:

- SELECT * FROM log_in_attempts - selects all entries from the log_in_attempts table.
- WHERE NOT department = ‘Information Technology’; - selects every row in which the department column does not contain the string ‘Information Technology’.

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.