Apply filters to SQL queries

Project description

As a security analyst, I will perform security related tasks to help keep the system safe. There are a few security issues that involve login attempts and employee machines and it is my job to investigate all the security issues. I will also need to update employee computers to prevent them against security vulnerabilities. The following steps show how I used SQL to query a specific database that contains login information.

Retrieve after hours failed login attempts

The organization noticed a potential security incident that happened after business hours. Log in attempts that happened after business hours and failed needs to be queried.

To help investigate this issue, I queried the log_in_attemps database using a filter. This filter returns all the log in attempts after the business hours that failed.

```
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      | su
ccess |
+----------+----------+------------+------------+---------+-----------------+---
------+
|        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 |
```

I selected all columns to be displayed so we could get all the information about the log-in attempt. The table that I queried from is the log_in_attemps table. Then I used a filter.
WHERE login_time > ‘18:00’ AND success = ‘FALSE’;
This filter returns all the failed login attempts after 6 pm. There are two filters applied here. The first one represents the after business hour filter, and the second one is for a failed attempt.

Retrieve login attempts on specific dates

The organization reported that a suspicious login attempt occurred on 2022-05-09. As a security analyst, I need to investigate all the login attempts on 2022-05-09 and on the day before.
The screenshot shows how I used an SQL query to filter for login attempts on the 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      | su
ccess |
+----------+----------+------------+------------+---------+-----------------+---
------+
|        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 |
```

This query returns all the login attempts that happened on 2022-05-09 and 2022-05-08. I selected all the columns from log_in_attemps table to be displayed so we could get more information about the incident. Then I specified what table to query using ‘FROM’. Then I performed my filter.
WHERE login_date = ‘2022-05-09’ OR ‘2022-05-08’;
This filter will display login attempts that happened on either 2022-05-09 or 2022-05-08. This simple query will help us investigate the security incident that occured on 2022-05-09 by reviewing who logged in and from where.

Retrieve login attempts outside of Mexico

After gathering more evidence, we come to know that the login attempt did and the security incident did not originate from Mexico. We want to know all of the login attempts that occurred outside of Mexico to investigate them.
The following screenshot shows how I used an SQL query to filter for 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     | su
ccess |
+----------+----------+------------+------------+---------+----------------+---
------+
|        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 |
```

This query returns all of the login attempts that did not originate from Mexico. I wanted to display all the columns for the resulting data, so I chose to use *. This will let me know more about each login attempt. Then I specified the table I am querying from, which is the log_in_attempts table. Next I applied my filter.

WHERE NOT country LIKE ‘MEX%’;

This filter uses the ‘NOT’ operator because we want to get all the login attempts from countries that are not Mexico. This query does not use an equals operator but uses LIKE to identify patterns. For example, any country that does not start MEX will be displayed. This way we can investigate login attempts that did not occur in Mexico.

Retrieve employees in Marketing

My organization wants to perform security updates on specific employee machines in the Marketing department. I have to get information about which employee machines to update. This screenshot shows how I used a SQL query to filter for employee machines from employees in the Marketing team whose office is in the east building.

```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE office LIKE 'East%' AND department = 'Marketing';
+-------------+-------------+----------+------------+----------+
| 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)
```

The resulting query returns all the employee machines from employees in the marketing team whose office is in the east building. The first line of the query says that I want all the columns to be displayed. The second line specifies the table I want to query, which is the employees table. Next, I applied my filter.

WHERE office LIKE ‘East%’ AND department = Marketing;

This query first specifies that all offices that start with ‘East’ should be filtered, then it performs an additional filter within the results to filter out the ‘Marketing’ department employees. This is a double filter and the result contains information about marketing department employees whose building is in the east. This way we can get the machine_id and perform the updates easily.

Retrieve employees in Finance or Sales

My organization needs to perform security updates for employees' machines in the Sales and the Finance department. My job is to get information about employees from these two departments.

The following screenshot shows how I used a SQL query to get employees that work in the Finance or Sales department. From there, we could get the employee_id easily to patch the machines.

```
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   |
|        1010 | k2421212m542 | jlansky  | Finance    | South-109   |
|        1011 | l748m120n401 | drosas   | Sales      | South-292   |
|        1015 | p611q262r945 | jsoto    | Finance    | North-271   |
|        1017 | r550s824t230 | jclark   | Finance    | North-188   |
|        1018 | s310t540u653 | abellmas | Finance    | North-403   |
|        1022 | w237x430y567 | arusso   | Finance    | West-465    |
```

The resulting query displays all the employees in the Finance and the Sales department. First, I wanted all the columns to be displayed so I specified that with a *. Then, I specified the table I want to query, which is the employees table. Then, I applied my filter.
WHERE department = ‘Finance’ OR department = ‘Sales’ ;
This filter gets all employees from the ‘Finance’ department, and the Sales' department. From the result, we can get all the machine_id to patch the machines.

Retrieve all employees not in IT

My organization also needs to perform a security update to the employees not in the IT department. My job is to get information about those employees and their machines.

The following screenshot shows how I used a SQL query to filter 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   |
|        1005 | f551g340h864 | gesparza | Human Resources | South-366   |
|        1007 | h174i497j413 | wjaffrey | Finance         | North-406   |
|        1008 | i858j583k571 | abernard | Finance         | South-170   |
|        1009 | NULL         | lrodriqu | Sales           | South-134   |
|        1010 | k2421212m542 | jlansky  | Finance         | South-109   |
```

This query returns all the employees not in the IT department. I first selected all the data from the employees table. Then I performed my query.
WHERE NOT department = ‘Information Technology’;
This filter uses the NOT operator which is like a negation. So in this example, the filter specifies to not return employees from the IT department.

Summary

In this activity, I gained hands-on experience with SQL by performing various queries with filters. I used two tables in this activity : log_in_attemps and employees. Some of my queries involved mathematical operators like ‘=’ and ‘>’ while others were characters like ‘%’ to look for patterns in the database. I also used the AND, OR and NOT operators to apply filters to my query.