Apply filters to SQL queries

Project description

My organization 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

My team is investigating failed login attempts that were made after business hours. To do this, we retrieved the information from the login activity and identify all unsuccessful attempts after 18:00.

The following code demonstrates how I created a SQL query to filter for failed login attempts that occurred after business hours.

```
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 part of the screenshot is my query, and the second part is a portion of the output. This query filters for failed login attempts that occurred after 18:00. First, I SELECT all data FROM `log_in_attempts` table. `WHERE` is a clause where I included the filter for the query, which is `login_time` after office hours at 18:00. `> '18:00'` indicated any time after 18:00. The `success` column in the `log_in_attempts` table contains values of TRUE or FALSE to indicate whether the login was successful. MySQL stores Boolean values as 1 for TRUE, and 0 for FALSE. This means that TRUE is represented as 1, and FALSE represented as 0 in the success column. Therefore, to retrieve unsuccessful attempts, we need to put `success` = 0. After the commands are entered, it gave me a result of 19 failed login attempts occurred after 18:00.

Retrieve login attempts on specific dates

Next, I would like to filter the result according to specific dates, which is on '2022-05-09'. I also want to retrieve all login attempts that occurred on this day and the day before ('2022-05-08'). The following code demonstrates how I created a SQL query to filter for failed login attempts that occurred on '2022-05-09' or '2022-05-08'.

```
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 part of the screenshot is my query, and the second part is a portion of the output. This query filters for failed login attempts that occurred on '2022-05-09' or '2022-05-08'. First, I SELECT all data from `log_in_attempts` table. `WHERE` is a clause where I included the filter for the query, which is `login_date` on '2022-05-09' or '2022-05-08'. Since there are 2 dates, the OR operator is used to retrieve the failed login attempts on the specified days. The first condition is `login_date = '2022-05-09'`, which filters for logins on 2022-05-09. The second condition is `login_date = '2022-05-08'`, which filters for logins on 2022-05-08.

Retrieve login attempts outside of Mexico

Next, our team investigated on logins that did not originate in Mexico, since our team believes there is a login attempt issue coming from countries outside of Mexico.

The following code demonstrates how I created a SQL query to filter for login attempts that occurred 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 |       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 part of the screenshot is my query, and the second part is a portion of the output. This query filters for failed login attempts for countries outside of Mexico. First I `SELECT` all

data from `log_in_attempts` table. `WHERE` is a clause where I included the filter for the query, which is `country`. Since within `country` field it can include entries with 'MEX' and 'MEXICO', we need to use the `NOT` and `LIKE` operators and the matching pattern '`MEX%`'. The result will bring out all entries within `country` that have `MEX`.
From this command, the results came back with 144 login attempts made outside of Mexico.

Retrieve employees in Marketing

On a separate task, my team is updating employee machines, and need to obtain the information about employees in the 'Marketing' `department` who are located in all offices in the East building (such as 'East-170' or 'East-320').

The following code demonstrates how I created a SQL query to filter for employees machines from employees, within Marketing department and located at East office.

```
MariaDB [organization]> SELECT * FROM employees WHERE department = 'Mark
eting' 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)
```

The first part of the screenshot is my query, and the second part is a portion of the output. This query filters for employees within Marketing department and located at the East building. First I `SELECT` all data from `employees` table. Then, I used a `WHERE` clause with `AND` to filter for employees who work in the Marketing department and in the East building. Since some values in the `office` column have East with different numbers, we will need to use the `LIKE` keyword with % to filter for all entries that with East = `East%`.
The first condition, is `department = 'Marketing',` which filters employees working in the Marketing. `office` in the East building. The second condition is `office LIKE 'East%',` which filters employees within Marketing that is located in the East building.

Retrieve employees in Finance or Sales

Next, our team needs to perform a different update to the computers of all employees in the Finance or the Sales `department`. The following code demonstrates how I created a SQL query to filter for employees machines from employees, within Finance and Sales department.

```
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 part of the screenshot is my query, and the second part is a portion of the output. This query filters for employees within Finance and Sales department. First I `SELECT` all data from `employees` table. Then, I used a `WHERE` clause with `OR` to filter for employees who work in the Marketing department or Finance department. Even though both conditions are based on the same column, both full conditions must be keyed in during the query. This means that each department must be specified as the column in both conditions. The first condition, is `department = 'Finance',` which filters employees working in the Finance department. The second condition is `department = 'Sales',` which filters employees working in the Sales department.

Retrieve all employees not in IT

Final update is made to employee computers, which is not in the Information Technology `department`. The team needs information about employees who are not in that `department`. The following demonstrates how I created a SQL query to filter for employee machines from 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   |
```

The first part of the screenshot is my query, and the second part is a portion of the output. This query filters for employees which are not within Information Technology department. First, I `SELECT` all data from `employees` table. Then, I used a `WHERE` followed by `NOT` to filter for employees who does not work in the Information Technology department.
The result will give the entries which includes all department except, Information Technology.

Summary

I filtered SQL queries to retrieve specific details about login attempts and employee machines. This involved working with two tables: `log_in_attempts` and `employees`. I applied the `AND`, `OR`, and `NOT` operators to refine the data for each task. Additionally, I used the `LIKE` operator along with the `%` wildcard to filter based on patterns.