Apply filters to SQL queries

Project description

For this project I used SQL with filters to handle security tasks for my organization to help make the system more secure. I was tasked with investigating all potential security issues and updating employee computers as needed.

Retrieve after hours failed login attempts

```
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 screen shot above shows my query and output. The query imputed filters for failed login attempts that took place after business hours which is 18:00. To do this, I started by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause with an `AND` operator to filter my results to output only login attempts that occurred after 18:00 and were unsuccessful. The first condition is `login_time > '18:00'`, which filters for the login attempts that occurred after 18:00. The second condition is `success = FALSE`, which filters for the failed login attempts.

Retrieve login attempts on 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      | 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 screen shot above shows my query and output. This query returns all login attempts that occurred on 2022-05-09 or 2022-05-08. I started by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause along with the `OR` operator to filter my results to output only login attempts that occurred on either 2022-05-09 or 2022-05-08. The first condition is `login_date = '2022-05-09'`, which filters for logins with the date of

2022-05-09. The second condition is `login_date = '2022-05-08'`, which filters for logins with the date of 2022-05-08.

Retrieve 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 |       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 screen shot above shows my query and output for investigating suspicious login attempted outside of Mexico. I started by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause with `NOT` to filter for countries other than Mexico. I used `LIKE` with `MEX%` as the pattern to match because the dataset represents Mexico as `MEX` and `MEXICO`. The percentage sign (`%`) represents any number of unspecified characters when used with `LIKE`.

Retrieve employees in Marketing

```
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 |
```

The screen shot above shows my query and output used to update the computers of the employees in Marketing. The first part of the screenshot is my query, and the second part is a portion of the output. I started by selecting all data from the `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. I used `LIKE` with `East%` as the pattern to match because the data in the `office` column represents the East building with the specific office number. The first condition is the `department = 'Marketing'` portion, which filters for employees in the Marketing department. The second condition is the `office LIKE 'East%'` portion, which filters for employees working out of the East building.

Retrieve employees in Finance or Sales

```
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 screen shot above shows my query and output used to update the computers of the employees in the Finance and Sales department. I started by selecting all data from the `employees` table. Then, I used a `WHERE` clause with `OR` to filter for employees who are in the Finance and Sales departments. To get employees who work in either department I used `OR` operator instead of `AND`. The first condition is `department = 'Finance'`, which filters for employees from the Finance department. The second condition is `department = 'Sales'`, which filters for employees from the Sales department.

Retrieve all employees not in IT

```
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 screen shot above shows my query and output used to update systems for employees who are not in the IT department. I started by selecting all data from the `employees` table. Then, I used a `WHERE` clause with `NOT` to filter for employees not part of the IT department.

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.