Apply filters to SQL queries

Project description

My organization is working to enhance the security of our system. As part of this initiative, my role is to investigate potential security issues and update employee computers when needed. The following steps demonstrate how I used SQL filters to perform security-related tasks, such as reviewing failed login attempts, identifying login attempts from specific dates and locations, and gathering information about employees in specific departments.

Retrieve after hours failed login attempts

There was a potential security incident that occurred after business hours (after 18:00). All failed login attempts that occurred after this time need to be investigated.
The following SQL query filters for failed login attempts that occurred after business hours:

```
SELECT *
FROM log_in_attempts
WHERE success = FALSE
AND login_time > '18:00';
```

Explanation: This query filters for failed login attempts (success = FALSE) that occur after 18:00. The login_time > '18:00' condition ensures that only login attempts made after business hours are retrieved.

Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. Any login attempts that happened on 2022-05-09 or the day before need to be investigated.
The following SQL query filters for login attempts that occurred on either 2022-05-09 or 2022-05-08:

```
SELECT *
FROM log_in_attempts
WHERE login_date = '2022-05-09'
OR login_date = '2022-05-08';
```

Explanation: This query filters for login attempts that occurred on the specified dates: 2022-05-09 and 2022-05-08. The OR operator is used to include login attempts from both days.

Retrieve login attempts outside of Mexico

Upon reviewing the organization's login attempts, I identified a potential issue with login attempts outside of Mexico. These attempts need to be investigated.
The following SQL query filters for login attempts that occurred outside of Mexico:

```
SELECT *
FROM log_in_attempts
WHERE country NOT LIKE '%MEX%';
```

Explanation: This query filters for login attempts that occurred outside of Mexico. The LIKE '%MEX%' pattern matches any records where the country is represented as MEX or MEXICO. By using NOT LIKE, we exclude records from Mexico and focus on login attempts from other countries.

Retrieve employees in Marketing

```
SELECT *
FROM employees
WHERE department = 'Marketing'
AND office LIKE 'East%';
```

Explanation: This query filters for employees in the Marketing department (department = 'Marketing') who are located in offices in the East building. The office LIKE 'East%' condition matches any office values that start with "East," such as East-170 or East-320.

Retrieve employees in Finance or Sales

The machines for employees in the Finance and Sales departments also need to be updated. The security update for these departments differs, so I need to identify employees only from these two departments.
The following SQL query filters for employees in the Finance or Sales departments:

```
SELECT *
FROM employees
WHERE department = 'Sales'
OR department = 'Finance';
```

Retrieve all employees not in IT

My team needs to perform a security update on employees who are not in the Information Technology (IT) department. To identify these employees, I created the following SQL query:

```
SELECT *
FROM employees
WHERE department NOT LIKE 'Information Technology';
```

Explanation: This query filters for employees who are not in the Information Technology department. The NOT LIKE operator is used to exclude employees whose department is Information Technology.

Summary

I applied SQL filters to gather specific information about login attempts and employee machines. By using the log_in_attempts and employees tables, I filtered data with the AND, OR, and NOT operators, allowing me to focus on the required information. I also used the LIKE operator with wildcards to search for patterns, such as matching office locations or countries. These queries helped in identifying security issues and employee machines that needed updates.