Apply filters to SQL queries

Project description

This project aimed to investigate potential security issues by examining the organization's data with SQL filters. Specifically, the focus was on analyzing login attempts and employee machines to detect anomalies and vulnerabilities. The project aimed to proactively address these issues and strengthen the organization's security measures.

Retrieve after hours failed login attempts

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

- I used the AND operator to retrieve the failed login attempts that occurred after business hours. By combining the relevant fields in the SQL query, I was able to filter the login attempts and identify those with unsuccessful login attempts that took place outside regular business hours. This allowed me to investigate potential unauthorized access attempts during non-operational periods, contributing to the overall security assessment.

Retrieve login attempts on specific dates

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

- I used the OR operator to retrieve the failed login attempts on the specified days. By employing the OR operator in the SQL query, I was able to filter the login attempts and identify all instances of unsuccessful login attempts that occurred on any of the specified days. This enabled me to analyze potential patterns or security issues related to failed login activities on those particular days, contributing valuable insights to the investigation process.

Retrieve login attempts outside of Mexico

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

- I ran the provided SQL query to retrieve login attempts that did not originate in Mexico. By using the NOT operator in conjunction with the LIKE operator and the pattern 'MEX%', I filtered the login attempts data to exclude any records where the login location started with 'MEX'. This allowed me to identify and investigate login activities originating from locations other than Mexico.

Retrieve employees in Marketing

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

- I used this SQL query to obtain information about employees in the 'Marketing' department located in offices within the East building, such as 'East-170' or 'East-320'. The query included filters on the 'department' and 'office' columns to return only the relevant records, and all columns were selected to retrieve comprehensive employee details meeting the specified criteria. This allowed the team to efficiently update the employee machines for the targeted group in the East building's offices.

Retrieve employees in Finance or Sales

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

- This SQL query retrieves records for employees in either the 'Finance' or the 'Sales' department. By using the OR operator in the query's filter on the 'department' column, I efficiently obtained information about employees belonging to either of these departments. This enabled your team to perform the necessary update to the computers of employees in the 'Finance' or 'Sales' department with ease.

Retrieve all employees not in IT

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

- This query filters the 'employees' table to only include records where the 'department' column does not contain 'Information Technology', allowing you to obtain information about employees from all other departments except IT.

Summary

The project involved utilizing SQL queries with filters to extract specific information on login attempts and employee machines from two tables, 'log_in_attempts' and 'employees'. The implementation of AND, OR, and NOT operators allowed for tailored filtering to meet the requirements of each task. Additionally, the use of the LIKE operator with the percentage sign (%) wildcard enabled filtering based on patterns in the data.