Apply filters to SQL queries

Project description

In this project, I demonstrate the ability to apply filters to SQL queries to retrieve specific data based on different conditions. By writing and executing various SQL queries, I filter and extract information such as failed login attempts after specific hours, login attempts on particular dates, employee details by department, and records based on location or other attributes. This project showcases how SQL can be used to query databases efficiently by utilizing conditional logic, pattern matching, and boolean filters.

Scenario:

You are a security professional at a large organization. Part of your job is to investigate security issues to help keep the system secure. You recently discovered some potential security issues that involve login attempts and employee machines.

Your task is to examine the organization's data in their **employees** and **log_in_attempts** tables. You'll need to use SQL filters to retrieve records from different datasets and investigate the potential security issues.

Note: *This scenario involves the same queries as the ones the* ***Filter with AND, OR, and NOT*** *lab. You can revisit the lab to get screenshots to include in your portfolio document. If you choose, it's also possible to complete this activity without revisiting the lab by typing your queries in the template*

Retrieve after hours failed login attempts

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

This query selects all records from the `log_in_attempts` table where the login time occurred after 6:00 PM (`'18:00'`) and the login attempt was unsuccessful (`success = FALSE`).

Retrieve login attempts on specific dates

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

This query fetches all login attempts from the `log_in_attempts` table that occurred on either May 9, 2022, or May 8, 2022 (`login_date = '2022-05-09' OR '2022-05-08'`).

Retrieve login attempts outside of Mexico

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

This query selects login attempts from the `log_in_attempts` table where the `country` field does not start with "MEX", filtering out attempts from Mexico.

Retrieve employees in Marketing

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

This query returns all employees from the `employees` table who work in the Marketing department and whose office location begins with "East".

Retrieve employees in Finance or Sales

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

This query retrieves all employees from the `employees` table who are either in the Finance or Sales departments.

Retrieve all employees not in IT

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

This query selects all employees from the `employees` table who are not part of the Information Technology department.

Summary

The project highlights essential SQL querying techniques to retrieve targeted data from a database. I use queries to filter login attempts by time, date, and location, as well as retrieve employees based on department and office location. By applying conditional statements (`AND`, `OR`, `NOT`), pattern matching (`LIKE`), and comparisons, the queries efficiently narrow down large datasets to focus on relevant records. These techniques are crucial for efficiently managing and analyzing data in real-world scenarios. This project serves as a foundational example of how SQL can be applied for data filtering and retrieval.