Apply filters to SQL queries

Project description

On this project, I used SQL filters to investigate potential security issues within an organization. The scenario involved querying the `log_in_attempts` table to analyze suspicious login behavior, particularly focusing on failed login attempts after hours, on specific dates, and outside of Mexico. Additionally, I queried the `employees` table to identify employees in specific departments and office locations for machine updates. These queries helped assess system vulnerabilities, detect security risks, and ensure proper updates for employee machines across various departments.

Retrieve after hours failed login attempts

```
SELECT *
FROM log_in_attempts
WHERE login_time > '18:00' AND success = 0;
```

This query filters the `log_in_attempts` table to retrieve failed login attempts (where the `success` column equals 0) that occurred after 18:00 hours (after business hours). By specifying the time condition and using the `AND` operator, I ensured that only records meeting both criteria (failed login and after hours) were returned.

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 uses the `OR` operator to filter login attempts that occurred on either May 9, 2022, or May 8, 2022. It helps identify login attempts from the specified dates for further investigation into a suspicious event. The `login_date` column was used to filter for these specific days.

Retrieve login attempts outside of Mexico

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

This query uses the `NOT LIKE` operator to filter login attempts originating outside of Mexico. The `LIKE 'MEX%'` pattern is used to match both "MEX" and "MEXICO" values in the `country` column. The query excludes any records where the `country` field contains "MEX" or "MEXICO", helping identify login attempts that did not originate in Mexico.

Retrieve employees in Marketing

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

This query retrieves employees who are in the Marketing department and located in offices in the East building. The `LIKE 'East%'` condition is used to match any office located in the East building. The `AND` operator ensures that both conditions are met (Marketing department and East building office).

Retrieve employees in Finance or Sales

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

This query retrieves employees who work in either the Finance or Sales department by using the `OR` operator. The filter looks for records where the `department` column contains either "Finance" or "Sales".

Retrieve all employees not in IT

```
SELECT *
```

```
FROM employees
WHERE department <> 'Information Technology';
```

This query uses the `<>` (not equal to) operator to retrieve all employees who are not in the Information Technology (IT) department. It filters out employees in the IT department, identifying employees in other departments who may need updates.

Summary

I applied various SQL filtering techniques to investigate security incidents and perform system updates. The tasks involved querying the `log_in_attempts` and `employees` tables, using operators such as `AND`, `OR`, and `NOT` to filter login attempts and employee data. The queries helped identify potential vulnerabilities such as suspicious login attempts outside of normal business hours and identify employees who need updates based on department and office location. This analysis is essential for securing the system and ensuring that the right employees receive necessary updates.