Apply filters to SQL queries

Project description

In this project I use SQL filters to investigate suspicious login activity and to identify specific groups of employees for security updates. I query the `log_in_attempts` and `employees` tables using `WHERE`, `AND`, `OR`, `NOT`, and `LIKE` to return only the records that match a given security scenario.

Retrieve after hours failed login attempts

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

Description

This query selects every column from the `log_in_attempts` table. The `WHERE` clause filters the results to only the rows where the `success` column is `FALSE`, which indicates a failed login attempt, and where `login_time` is later than 18:00. This returns all failed login attempts that happened after business hours so the security team can review them.

Retrieve login attempts on specific dates

```
SELECT *
FROM log_in_attempts
WHERE login_date IN ('2022-05-09', '2022-05-08');
```

Description

This query returns all records from `log_in_attempts` where the `login_date` is either `2022-05-09` or `2022-05-08`. The `IN` operator acts like multiple `OR` conditions and makes it easy to filter on several specific dates. This helps investigate all login activity that occurred on the suspicious date and the day before.

Retrieve login attempts outside of Mexico

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

In the `log_in_attempts` table, the `country` column can contain values such as `MEX` or `MEXICO` for Mexico. The `NOT LIKE 'MEX%'` condition filters out any rows where the country starts with `MEX`. As a result, the query returns only login attempts that occurred outside of Mexico, which allows the team to focus on foreign activity.

Retrieve employees in Marketing

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

This query pulls all columns from the `employees` table but only for employees who belong to the `Marketing` department and whose `office` value starts with `East` (for example, `East-170` or `East-320`). The `AND` operator combines both conditions so that only Marketing employees located in the East building are returned. These results can be used to plan department specific security updates for that building.

Retrieve employees in Finance or Sales

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

Description

This query selects all rows from `employees` where the `department` is either `Finance` or `Sales`. The `OR` operator ensures that records meeting either condition are included. This allows the team to identify all employees in these two departments so they can receive a different type of security update on their machines.

Retrieve all employees not in IT

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

Description

Here the query returns every employee whose `department` is not `Information Technology`. The `<>` operator means “not equal to.” Since IT employees already received their update, this filter helps locate all remaining employees in other departments who still need the security update.

Summary

Across these queries, I applied SQL filters to narrow large tables down to the exact records needed for security tasks. Using comparison operators, `IN`, `LIKE`, `AND`, `OR`, and `NOT`, I was able to:

- Isolate failed login attempts that occurred after business hours.
- Review login activity on specific suspicious dates.
- Focus on login events from outside a particular country.
- Identify employees by department and building.
- Separate employees who still require security updates from those who already received them.

These queries demonstrate how SQL can be used to support cybersecurity investigations and day to day security operations.