Apply filters to SQL queries

Project description

During the routine system test, a potential security incident was uncovered, which took place outside of regular business hours. I examined the SQL table to identify the unsuccessful login attempts that occurred after 18:00.

Retrieve after hours failed login attempts

There was a potential security incident that occurred after business hours (after 18:00).

This query allows me to retrieve the failed login attempts made after regular business hours from the SQL database:

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

Let's break down how this query works:

- The `SELECT *` statement retrieves all columns of data from the table log_in_attempts for the matching records that satisfy the specified conditions.

- The `FROM log_in_attempts` statement specifies the table from which we want to retrieve the data. In this case, it's the `log_in_attempts`table.

- The WHERE clause is used to filter the results based on specific conditions. In our case, we have two conditions: `login_time` `'18:00'` compares the `login_time` column with 18:00 which is the regular business hours end. This condition ensures that only login attempts made after business hours are included in the results.
- `success = FALSE` filters the results further to only include failed login attempts. You can modify this condition to match the specific value or criteria you are using to determine a failed login attempt.
- By combining these conditions with the logical operator AND, the query retrieves the failed login attempts made after regular business hours from the `log_in_attempts` table.

Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. I conduct an investigation into any login activities that occurred on May 9, 2022, as well as those that took place on the preceding day.

This query allows us to review all login attempts which occurred on this day and the day before:

```
SELECT *
FROM log_in_attempts
WHERE login_date = '2022-05-09' OR login_date = '2022-05-8';
```

- The `WHERE` clause specifies the conditions for filtering the data. In this case, it uses the equality operator (`=`)to compare the values of the `login_date` column with the specified dates. The logical operator `OR` is used to retrieve the login attempts that match either of the specified dates.

Retrieve login attempts outside of Mexico

A specious login attempt originating from a location outside of Mexico was captured. To investigate

I used the following query to retrieve all login attempts that occurred outside of the country.

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

- `NOT country LIKE 'MEX%'`: This condition excludes login attempts where the country starts with 'MEX'. The 'LIKE' keyword is used for pattern matching, and the % symbol is a wildcard that matches any number of characters. So, 'MEX%' matches any country that starts with 'MEX'. By using the NOT operator, we select login attempts from all countries except Mexico.

Retrieve employees in Marketing

My team wants to update certain computers of employees in the Marketing department.

To do so, I used this query to filter for employee machines from employees in the Marketing department in the East building.

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

- The `WHERE` clause is used to filter the results based on specific conditions. In our case, we have two conditions:
- `department = 'Marketing'` checks if the department column in the table is equal to the string value 'Marketing'. This condition selects the employees who work in the marketing department.
- `AND office LIKE 'East%'` combines with the previous condition using the logical operator AND. It further filters the results based on the value of the office column. The LIKE operator is used to perform pattern matching. The pattern `'East%'` means that the office value should start with the string 'East'.

Retrieve employees in Finance or Sales

The team also wants to update the machines for employees in the Sales and Finance departments.

The following query retrieves records for employees in the `'Finance'` or the `'Sales'` department.

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

This query returns all employees in the Finance and Sales departments.

- The first condition, `department = 'Finance'`, narrows down the results to employees from the Finance department.

- The second condition, `department = 'Sales'`, filters the results to include employees from the Sales department.

By using the `OR` operator, I ensure that the query fetches all employees who are in either of these departments, rather than requiring them to be in both departments simultaneously.

Retrieve all employees not in IT

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

This query retrieves records for employees who are not in the `'Information Technology'` department.

By using the NOT keyword in conjunction with the equality operator =, the query retrieves all employee records from the employees table where the department is not equal to 'Information Technology'. This filter allows us to focus on employees who are part of departments other than Information Technology.

Summary

I used SQL queries to get specific information about login attempts and employee machines.