Apply filters to SQL queries

Project description

My organisation is working on making their system secure, and it is my job to ensure the system is safe. My tasks were to examine the organisation's data in their `employees` and `log_in_attempts` tables. I used SQL filters to retrieve records from different datasets and investigate the potential security issues.

Retrieve after hours failed login attempts

A potential security incident that occurred after business hours has been discovered. To investigate this, I queried the `log_in_attempts` table and reviewed after hours login activity.

This is what was typed into MariaDB to get the records.

```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_time > '18:00:00' AND success = 0;
```

The first row, `SELECT *`, indicates that I want all of the columns to be returned.

The second row, `FROM log_in_attempts`, indicates that I want to query from the `log_in_attempts` table.

The third row is the filter for the data, this is accomplished with the `WHERE` clause. First, `login_time > '18:00:00'` indicates that the query will return only the login attempts made after 6PM. `login_time` is the column that is being filtered, the operator `>` is greater than which is an exclusive operator saying `login_time` has to be greater than, and to the right of the operator is 6PM in 24 hours time.

The `AND` operator was used to check for another condition to see if both are met. In this case the condition is `success = 0`. This indicates that the query will only return the failed login attempts that happened after 6PM as the `success` column is storing boolean values and `0` is equivalent to `FALSE`.

`;` indicates the end of the query.

The query returned 19 results of failed login attempts after business hours.

Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. To investigate this event, I reviewed all login attempts which occurred on this day and the day before.

This is the query that was used.

```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
```

The first 2 rows are the same as previous. `SELECT * FROM log_in_attempts` indicates that I want the query to return all columns from the `log_in_attempts` table.

The third row is filtering for data from the specific dates. The first part, `login_date = '2022-05-09'`, indicates that the query will return all the login attempts from 2022-05-09.

The `OR` operator was used to check for another condition to see if either is met. In this case the condition is `login_date = '2022-05-08'`. This indicates that the query will return all the login attempts made on either 2022-05-09 or 2022-05-08.

The query returned 75 results of login attempts on 2022-05-09 and 2022-05-08.

Retrieve login attempts outside of Mexico

There's been suspicious activity with login attempts, the security team has determined that this activity didn't originate in Mexico. To investigate this event, I reviewed all login attempts that occurred outside of Mexico.

This is the query that was used.

```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE NOT country LIKE 'MEX%';
```

The first 2 rows are the same as previous. `SELECT * FROM log_in_attempts` indicates that I want the query to return all columns from the `log_in_attempts` table.

The third row is filtering for a string pattern. The part `country LIKE 'MEX%'` indicates that the query will return all the records that start with MEX on the country column. The `%` operator in the string is a wildcard operator used to substitute for any number of other characters. The `LIKE` operator is used to search for patterns in a column.
The `NOT` operator was used right after the `WHERE` clause to negate a condition. In this case it negates `country LIKE 'MEX%'` so the query will return all login attempts made outside of Mexico.

The query returned 144 results of login attempts made outside of Mexico.

Retrieve employees in Marketing

The security team has decided to perform security updates on specific employee machines in the marketing department. I am responsible for getting information on these employee machines.

This is the query that was used.

```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE department = 'Marketing' AND office LIKE 'EAST%';
```

The first 2 rows are the same as previous. `SELECT * FROM log_in_attempts` indicates that I want the query to return all columns from the `employees` table.

The third row is filtering for a string pattern and a specific department. The first part `department = 'Marketing'`, indicates that the query will return all the employees that are in the Marketing department.

The `AND` operator was used to check for another condition to see if both are met. In this case the condition is `office LIKE 'EAST%'`. This indicates that the query will search for strings in the office column that starts with EAST as the `%` operator was used.

The query returned 7 marketing employees who work in the East building.

Retrieve employees in Finance or Sales

The security team needed to perform a different security update on machines for employees in the sales and finance departments. I was responsible for getting information on these employee machines.

This is the query that was used.

```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE department = 'Sales' OR department = 'Finance';
```

The first 2 rows are the same as previous. `SELECT * FROM log_in_attempts` indicates that I want the query to return all columns from the `employees` table.

The third row is filtering for employees from specific departments. The first part `department = 'Sales'`, indicates that the query will return all the employees that are in the sales department.

The `OR` operator was used to check for another condition to see if either is met. In this case the condition is `department = 'Finance'`. This indicates that the query will return all the employees that are from the finance department and the marketing department.

The query shows that the number of employees between the sales department and the finance department is 71.

Retrieve all employees not in IT

The security team needed to make one more update to employee machines. The employees who are in the IT department already had this update, but employees in all other departments needed it. I was responsible for getting information on these employee machines.

This is the query that was used.

```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE NOT department = 'Information Technology';
```

The first 2 rows are the same as previous. `SELECT * FROM log_in_attempts` indicates that I want the query to return all columns from the `employees` table.

The third row is filtering for a specific department. The part `department = 'Information Technology'` indicates that the query will return all the employees that are in the IT department.

The `NOT` operator was used right after the `WHERE` clause to negate a condition. In this case it negates `department = 'Information Technology'` so the query will return all the employees from other departments.

The query shows that there are 161 employees from other departments other than IT.

Summary

Through the use of SQL, I was able to investigate the potential security issues using the data returned. I was able to determine which of the employees' systems needed updates to help keep the system secure. I used two tables to accomplish this, `log_in_attempts` and `employees`. I also used the `AND`, `OR`, and `NOT` operators to filter for specific information.