Apply filters to SQL queries

Project description

My organization is enhancing its system security, and my role involves ensuring system safety, investigating potential security issues, and updating employee computers as needed. Here are examples of how I used SQL with filters to perform security-related tasks.

Retrieve after hours failed login attempts

A potential security incident occurred after business hours (after 18:00), and all failed login attempts during this time need to be investigated. The following code demonstrates how I created a SQL query to filter for these failed login attempts.

A screen shot of a computer

Description automatically generated

The first part of the screenshot shows my query, and the second part displays a portion of the output. This query filters for failed login attempts that occurred after 18:00. I started by selecting all data from the log\_in\_attempts table. Then, I used a WHERE clause with an AND operator to filter the results, showing only login attempts that occurred after 18:00 and were unsuccessful. The first condition, login\_time > '18:00', filters for login attempts made after 18:00. The second condition, success = FALSE, filters for the failed login attempts.

Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. Any login activity that happened on 2022-05-09or on the day before needs to be investigated.

A screen shot of a computer

Description automatically generated

The first part of the screenshot shows my query, and the second part displays a portion of the output. This query returns all login attempts that occurred on 2022-05-09 or 2022-05-08. I started by selecting all data from the log\_in\_attempts table. Then, I used a WHERE clause with an OR operator to filter the results to show only login attempts that occurred on either 2022-05-09 or 2022-05-08. The first condition, login\_date = '2022-05-09', filters for logins on 2022-05-09, while the second condition, login\_date = '2022-05-08', filters for logins on 2022-05-08

Retrieve login attempts outside of Mexico

After investigating the organization’s data on login attempts, I identified a potential issue with login attempts that occurred outside of Mexico. These attempts should be investigated. The following code demonstrates how I created a SQL query to filter for login attempts that occurred outside of Mexico.

A screen shot of a computer

Description automatically generated

The first part of the screenshot shows my query, and the second part displays a portion of the output. This query returns all login attempts that occurred in countries other than Mexico. I started by selecting all data from the log\_in\_attempts table. Then, I used a WHERE clause with NOT to filter out entries where the country is Mexico. I used LIKE with the pattern 'MEX%' because the dataset represents Mexico as both MEX and MEXICO. The percentage sign (%) acts as a wildcard, matching any number of unspecified characters.

Retrieve employees in Marketing

My team needs to update the computers for certain employees in the Marketing department. To identify which employee machines to update, I created a SQL query to filter for employees in the Marketing department located in the East building. The following code demonstrates how I did this.

A screenshot of a computer

Description automatically generated

The first part of the screenshot shows my query, and the second part displays a portion of the output. This query returns all employees in the Marketing department located in the East building. I started by selecting all data from the employees table. Then, I used a WHERE clause with AND to filter for employees in the Marketing department and in the East building. I used LIKE with 'East%' as the pattern to match because the office column data represents the East building with specific office numbers. The first condition, department = 'Marketing', filters for employees in the Marketing department. The second condition, office LIKE 'East%', filters for employees in the East building.

Retrieve employees in Finance or Sales

The machines for employees in the Finance and Sales departments also need to be updated. Since a different security update is required, I need information on employees from only these two departments. The following code demonstrates how I created a SQL query to filter for employee machines from employees in the Finance or Sales departments.

A screenshot of a computer screen

Description automatically generated

The first part of the screenshot shows my query, and the second part displays a portion of the output. This query returns all employees in the Finance and Sales departments. I started by selecting all data from the employees table. Then, I used a WHERE clause with the OR operator to filter for employees in either the Finance or Sales departments. The first condition, department = 'Finance', filters for employees in the Finance department. The second condition, department = 'Sales', filters for employees in the Sales department.

Retrieve all employees not in IT

My team needs to make one more security update on employees who are not in the Information Technology department. To gather information on these employees, I created a SQL query to filter for employee machines from employees not in the Information Technology department. Here's how I did it.

A screenshot of a computer screen

Description automatically generated

The first part of the screenshot displays my query, while the second part shows a portion of the output. This query retrieves all employees not in the Information Technology department. Initially, I selected all data from the employees table. Subsequently, I employed a WHERE clause with NOT to filter out employees not in this department.

Summary

I utilized filters in SQL queries to retrieve specific information on login attempts and employee machines from two distinct tables: log\_in\_attempts and employees. Employing the AND, OR, and NOT operators allowed me to target the precise information required for each task. Additionally, I utilized LIKE with the percentage sign (%) wildcard to filter for patterns in the data.