# Apply filters to SQL queries

Scenario:

You are a security professional at a large organization. Part of your job is to investigate security issues to help keep the system secure. You recently discovered some potential security issues that involve login attempts and employee machines.

Your task is to examine the organization’s data in their **employees** and **log\_in\_attempts** tables. You’ll need to use SQL filters to retrieve records from different datasets and investigate the potential security issues.

## Project description

This project involves analyzing employee login activity using SQL queries to identify potential security risks. The focus is on tracking failed after-hours logins, access from outside approved locations, and specific departmental employee data to support investigations and system updates. By filtering and examining records, the project helps strengthen the organization’s security posture. The following steps were used to perform the security related task:

## Retrieve after hours failed login attempts

Failed login attempts occurred after business hours (18:00). To investigate the incidents, following SQL code were used to attain failed login attempts after business hours:



The first part of the screenshot is my query, and the second part is a portion of the output. First, all data is selected from the table, then the WHERE clause is used with the AND operator to narrow down the output to login attempts that happened after 18:00 and that were unsuccessful. The login\_time > '18:00' condition catches all attempts after-hours, while success = FALSE pinpoints only failed attempts.

## Retrieve login attempts on specific dates

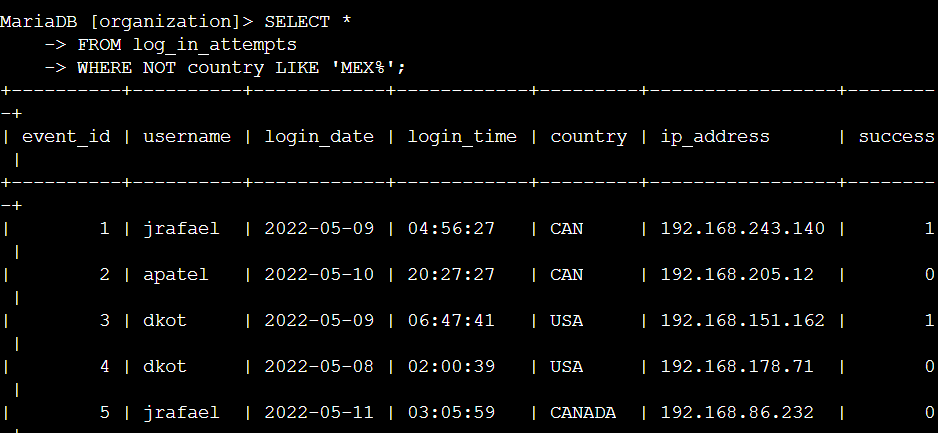
A suspicious event occurred on '2022-05-09'. To retrieve all login attempts that occurred on this day and the day before ('2022-05-08'), the following SQL code was used:



It selects all information from the log\_in\_attempts table and then uses the WHERE clause with an OR clause to show only rows where the login\_date is either '2022-05-09' or '2022-05-08'. By doing this, the output focuses just on login activity that happened on those days, making it easier to check details like usernames, times, countries, IP addresses, and whether the attempts were successful or not.

## Retrieve login attempts outside of Mexico

After investigating the organization’s data on login attempts, I believe there is an issue with the login attempts that occurred outside of Mexico. To investigate this, the following SQL code is used:

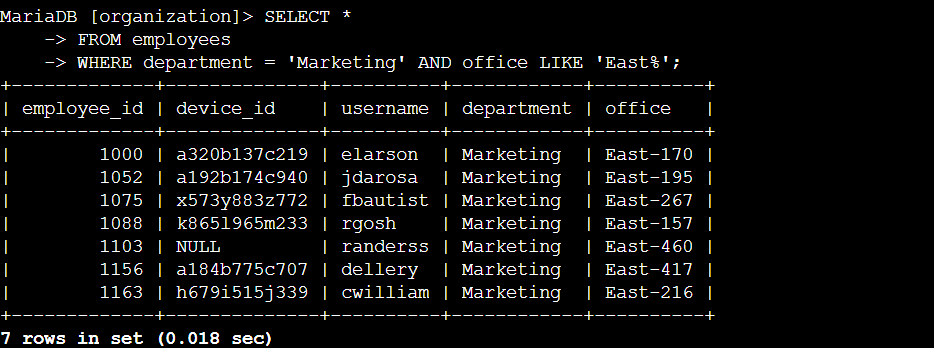


The first part is the query and the second part is the output. This query displays login attempts from the database but excludes all records where the country starts with 'MEX', which means any login from Mexico will not appear in the results. It selects every row from the log\_in\_attempts table and uses the condition WHERE NOT country LIKE 'MEX%' to filter out logins from Mexico, showing only attempts made from other countries along with their details like username, date, time, IP address, and success status.

## Retrieve employees in Marketing

To update the computers for certain employees in the Marketing department, I have to get information on which employee machines to update.

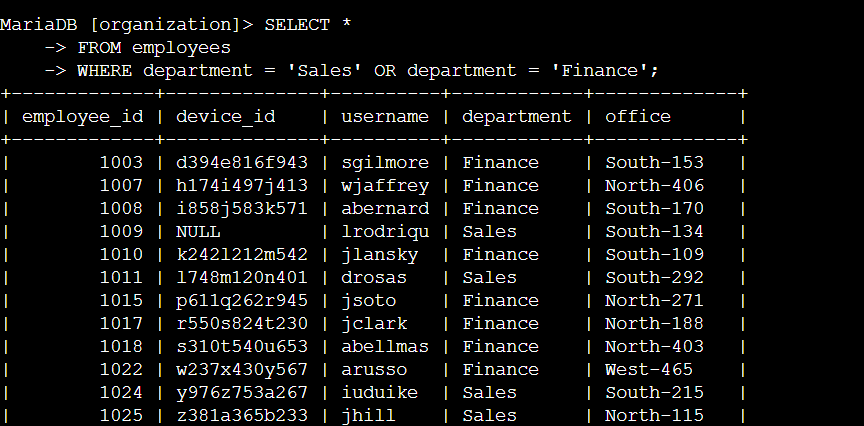
For this, the following query was used:



The first part of the screenshot is my query, and the second part is a portion of the output. This query retrieves detailed information about employees who belong specifically to the Marketing department and work in offices with names that start with "East." It uses the statement SELECT \* to select all columns from the employees table. The filtering is done by the WHERE clause with two conditions combined by the AND operator. The first condition, department = 'Marketing', selects only those employees whose department value matches Marketing exactly. The second condition, office LIKE 'East%', uses the LIKE operator with a wildcard % to match any office names starting with "East".  This way, the query focuses on employees who meet both criteria, showing employee ID, device ID, username, department, and office, which is helpful for managing and analyzing team assignments within specific locations.

## Retrieve employees in Finance or Sales

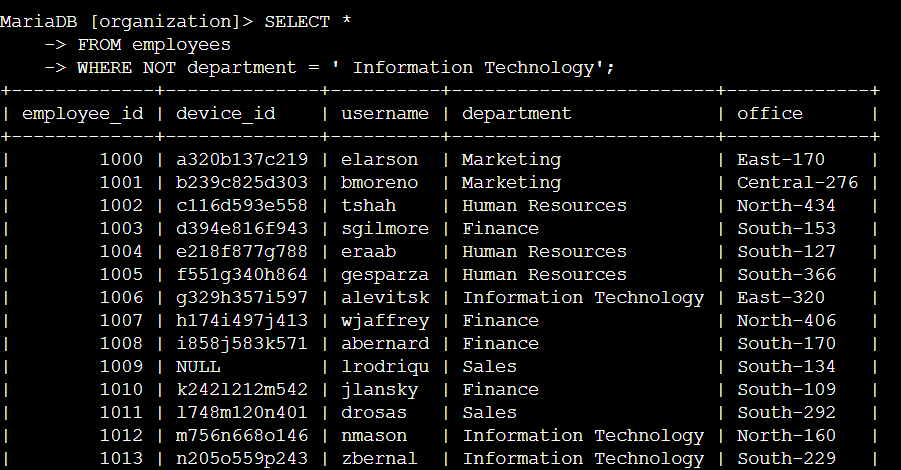
The machines for employees in the Finance and Sales departments also need to be updated. To get information on employees only from these two departments, the following codes where used:



This query retrieves employees from the database who work in either the Sales or Finance departments. It uses SELECT \* FROM employees to choose all columns from the employees table, then the WHERE clause with department = 'Sales' OR department = 'Finance' to filter the results, showing only those rows where the department is either Sales or Finance. The output provides information including employee ID, device ID, username, department, and office for each matched employee, making it easy to review and analyze staff from these two specific departments.

## 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 make the update, I first have to get information on these employees. The following codes where used for this:



This query is used to display all employees except those who work in the Information Technology department. It uses SELECT \* FROM employees to select every column from the employees table and applies a filter with WHERE NOT department = 'Information Technology', which removes any records where the department is 'Information Technology'. As a result, the output shows the employee ID, device ID, username, department, and office for staff in all other departments such as Marketing, Human Resources, Finance, and Sales, making it easy to focus on non-IT personnel.

## Summary

I applied filters to SQL queries to get specific information on login attempts and employee machines. I used two different tables, log\_in\_attempts and employees. I used the AND, OR, and NOT operators to filter for the specific information needed for each task. I also used LIKE and the percentage sign (%) wildcard to filter for patterns.