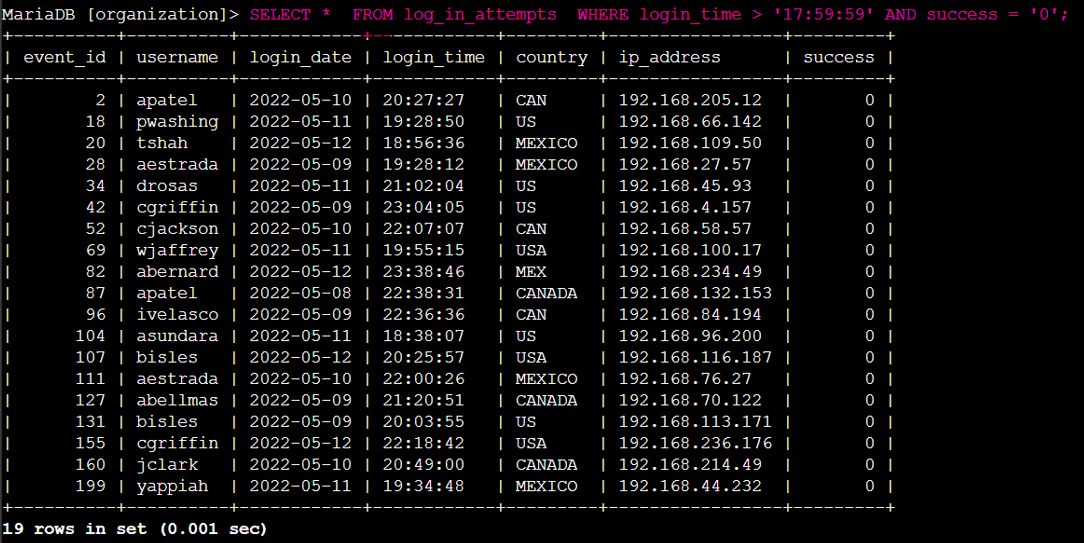
# **Apply filters to SQL queries**

## **Project description**

## I'm responsible for enhancing my organization's system security. My role involves safeguarding the system, conducting thorough investigations into potential security concerns, and maintaining employee computers as required. The following steps illustrate instances where I used SQL filters to accomplish security-related objectives.

## **Retrieve after hours failed login attempts**

## I recently discovered a potential security incident that occurred after business hours. To investigate this, I queried the log\_in\_attempts table and reviewed the after-hours login activity (after 18:00).

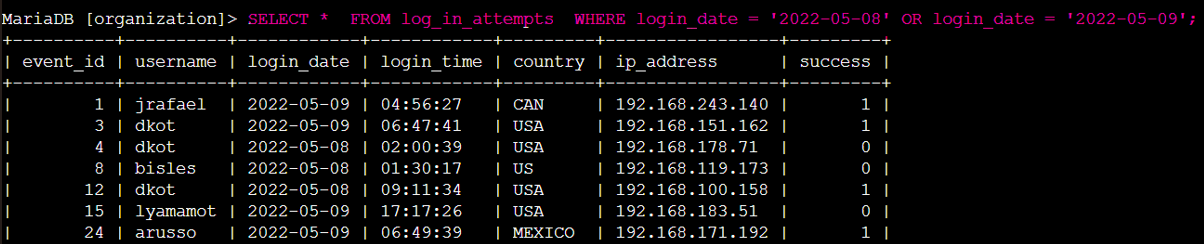


The initial segment of the screenshot represents my query, while the subsequent part displays a segment of the output. This query is designed to filter failed login attempts occurring after 18:00. To begin, I initiated the query by selecting all data from the log\_in\_attempts table. Next, I employed a WHERE clause in conjunction with an AND operator to refine my results, exclusively displaying login attempts that transpired post 18:00 and were unsuccessful. The initial condition, login\_time > '17:59:59', filters for login attempts occurring after 18:00, while the second condition, success = ‘0’ (1 for TRUE, 0 for FALSE), isolates the failed login attempts.

## **Retrieve login attempts on specific dates**

## A suspicious event occurred on May 9, 2022. As a result, I had to investigate all login activities from May 9, 2022, as well as activities from the day preceding it.

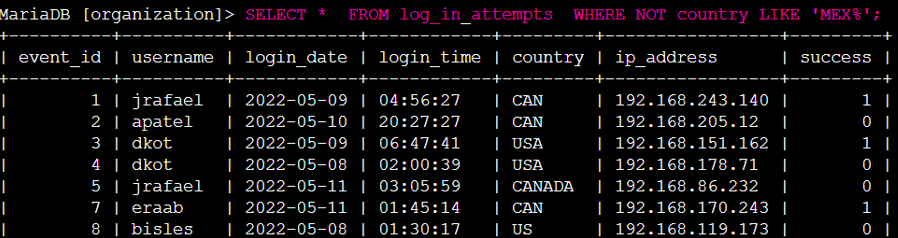
## The code below shows how I made a SQL query to pinpoint login attempts on specific dates for further scrutiny.



This query is designed to filter login attempts occurring on 2022-05-08 or 2022-05-09. To begin, I initiated the query by selecting all data from the log\_in\_attempts table. Next, I employed a WHERE clause in conjunction with an OR operator to refine my results, exclusively displaying login attempts that transpired on 2022-05-08 or 2022-05-09.

## **Retrieve login attempts outside of Mexico**

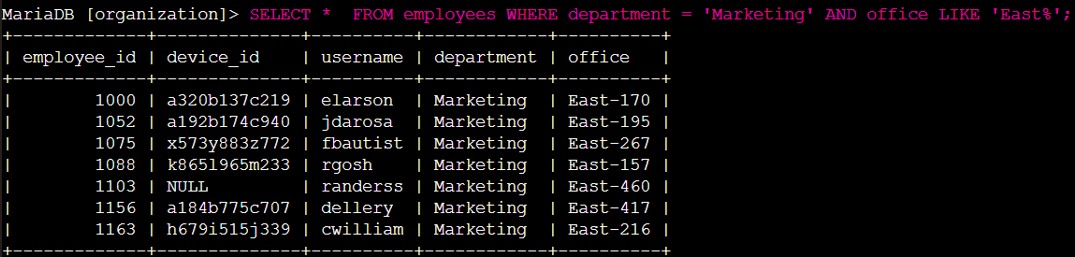
Upon conducting an in-depth examination of the organization's login attempt data, it has come to my attention that there may be a concern regarding login attempts originating from outside of Mexico. Therefore, I made the following query:



This query retrieves all login attempts originating from countries other than Mexico. I applied a WHERE clause with the NOT operator to exclusively capture countries that **are not Mexico**. I employed the LIKE operator with the pattern 'MEX%' because the dataset denotes Mexico as either 'MEX' or 'MEXICO.' The '%' symbol serves as a wildcard, accommodating any number of unspecified characters when used with LIKE.

## **Retrieve employees in Marketing**

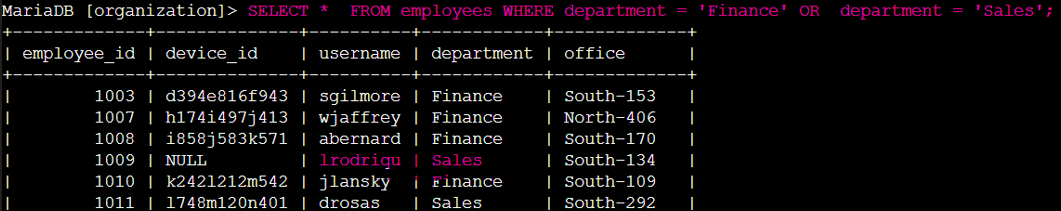
Our team aims to update computers for specific employees within the Marketing department from the different east offices. To facilitate this process, I made the following query:



The objective of this query is to retrieve a list of employees situated in the Marketing department within the East offices. A WHERE clause is used in conjunction with the AND operator to narrow down the results to employees from to both the Marketing department and the East offices. The LIKE operator is used with the pattern 'East%', as the office column contains the East buildings using specific office numbers. The first condition department = 'Marketing' filters for Marketing department employees, while the second condition office LIKE 'East%’ identifies employees in the different East offices.

## **Retrieve employees in Finance or Sales**

The computers for employees in the Finance and Sales departments also needed to be updated, so I made the following query:



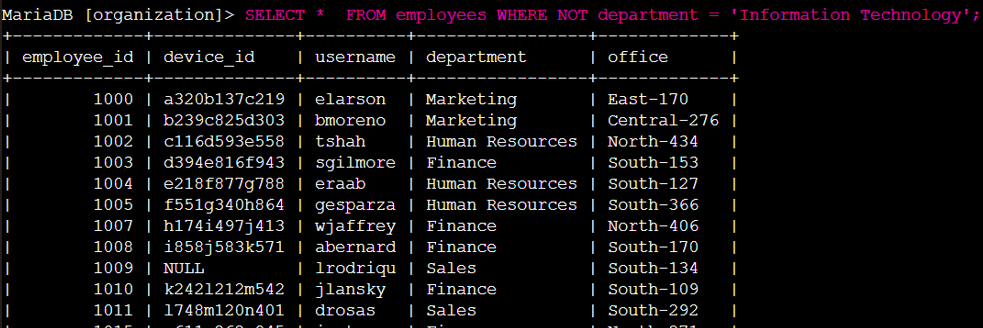
This query shows all employees in the Finance and Sales departments. In this scenario, I used the OR operator instead of AND because I wanted **all** employees who are in either department.

The first condition department = 'Finance' filters for Finance department employees, while department = 'Sales' filters for Sales department employees.

## **Retrieve all employees not in IT**

The team needed to make one more security update on employees who are not in the

Information Technology department. I filtered employee machines from employees who are not in the IT department with this query:



Here I used a WHERE in conjunction with NOT operator to filter for employees who not in the ‘Information Technology’ department.

## **Summary**

In these scenarios, I focused on SQL queries to investigate and address various security and operational concerns within the organization. I used SQL query filters to extract precise details about login attempts and employee computers. Using both the log\_in\_attempts and employees tables, I applied the AND, OR, and NOT operators to filter the specific information required. Additionally, I used the LIKE operator and the '%' wildcard to filter patterns effectively.