# Apply filters to SQL queries

## Project description

## In this portfolio project, I have demonstrated SQL skills by applying various filters to retrieve specific data from a database. The scenario involves working with employee data for login attempts, and I have retrieved login attempt data based on different criteria.

This document describes how the tables used for this portfolio activity are organized. The organization database contains the following two tables:

* log\_in\_attempts
* employees

## **log\_in\_attempts**

The log\_in\_attempts table has the following columns:

* event\_id: The identification number assigned to each login event
* username: The username of the employee
* login\_date: The date the login attempt was recorded
* login\_time: The time the login attempt was recorded
* country: The country where the login attempt occurred
* ip\_address: The IP address of that employee’s machine
* success: The success of the login attempt; FALSE indicates a failed attempt

In the MariaDB shell, these columns are returned as:



## **employees**

The employees table has the following columns:

* employee\_id: The identification number assigned to each employee
* device\_id: The identification number assigned to each device used by the employee
* username: The username of the employee
* department: The department the employee is in
* office: The office the employee is located in

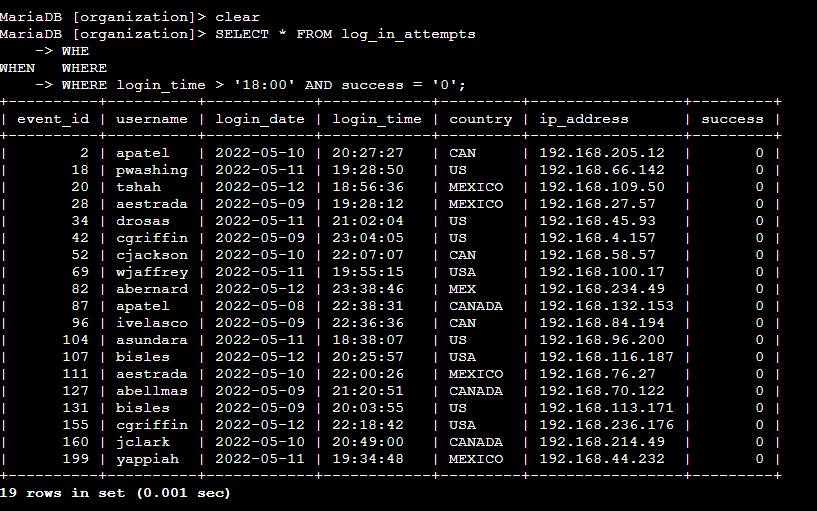
In the MariaDB shell, these columns are returned as:



## Retrieve after hours failed login attempts

## The login\_time column in the log\_in\_attempts table contains information on when login attempts were made. Office hours end at '18:00'.

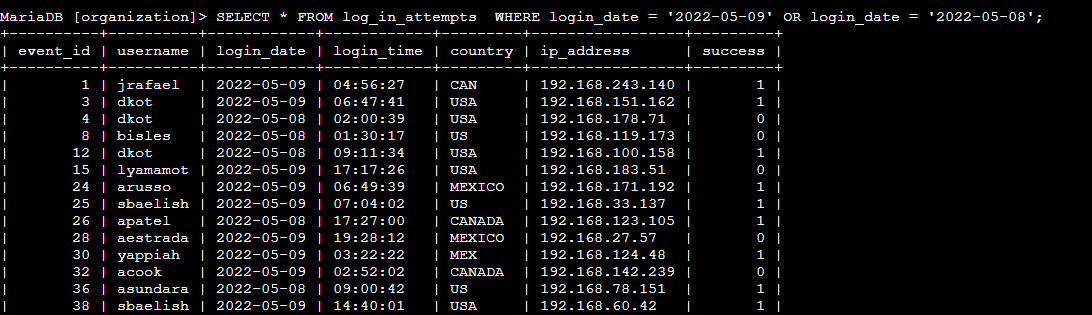
## The success column in the log\_in\_attempts table contains values of TRUE or FALSE to indicate whether the login was successful. MySQL stores Boolean values as 1 for TRUE, and 0 for FALSE. This means that TRUE is represented as 1, and FALSE represented as 0 in the success column.



## Retrieve login attempts on specific dates

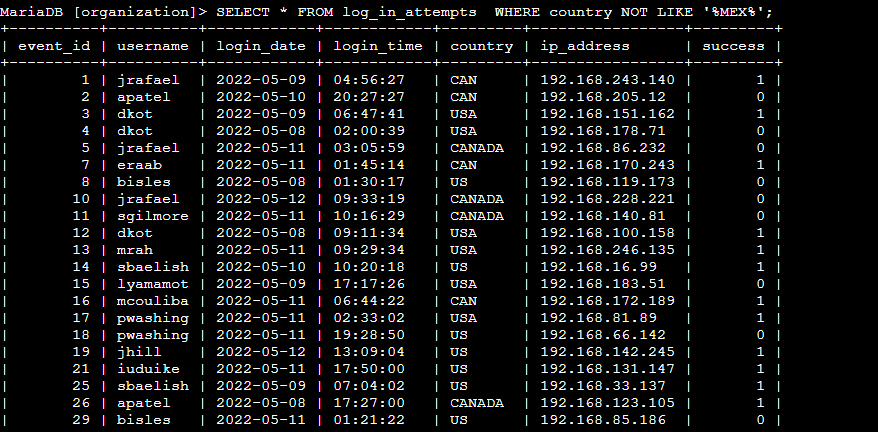
Investigating a suspicious event that occurred on '2022-05-09', retrieved all login attempts that occurred on this day and the day before ('2022-05-08').

The login\_date column in the log\_in\_attempts table contains information on the dates when login attempts were made.



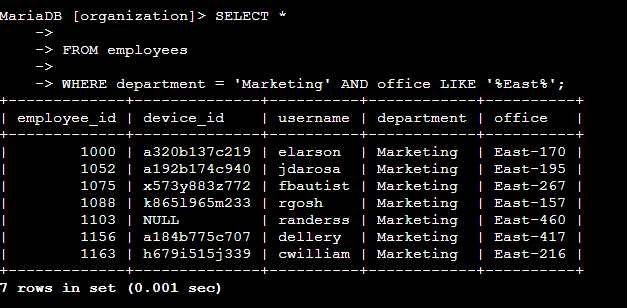
## Retrieve login attempts outside of Mexico

This query retrieves all login attempts that did not originate in Mexico. The country column is checked using the NOT LIKE operator with '%MEX%' to filter out entries with 'MEX' or 'MEXICO' in the country name.



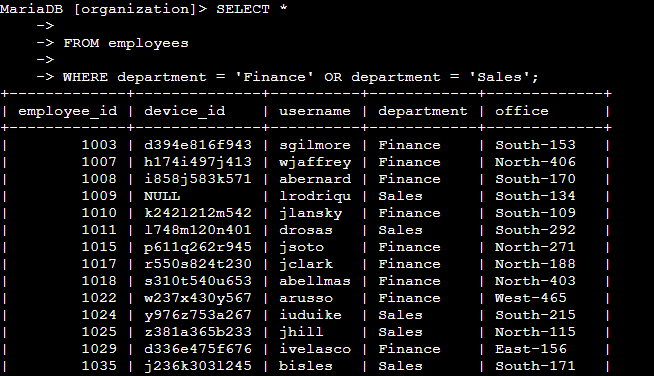
## Retrieve employees in Marketing

This query retrieves all employees in the Marketing department who work in offices located in the East building. The department column is used to filter for employees in Marketing, and the office column is filtered using the LIKE operator with '%East%' to include all offices with names containing 'East' in the value. The % sign in the LIKE operator is used as a wildcard to include any characters before or after 'East' in the office name. This way, it covers all offices in the East building regardless of their specific office number.



## Retrieve employees in Finance or Sales

This query selects all columns (\*) from the "employees" table and retrieves records where the department column has a value of either 'Finance' or 'Sales'. The OR logical operator allows us to combine the two conditions, and the query will return all rows where the department is either 'Finance' or 'Sales'.



## Retrieve all employees not in IT

This query retrieves all employees who do not belong to the Information Technology department. The department column is checked using the NOT LIKE operator with '%Information Technology%' to exclude IT-related entries.



## Filtering data using JOINS

INNER JOIN:

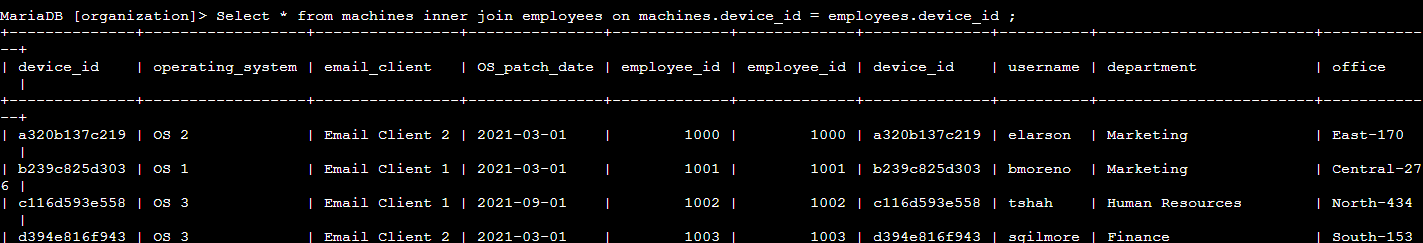
The INNER JOIN is a type of SQL join that combines rows from two or more tables based on a related column between them. It retrieves only the rows that have matching values in both tables. The result set contains only the records where the join condition is satisfied in all the participating tables. If there is no match for a particular row in one of the tables, that row will be excluded from the result set.

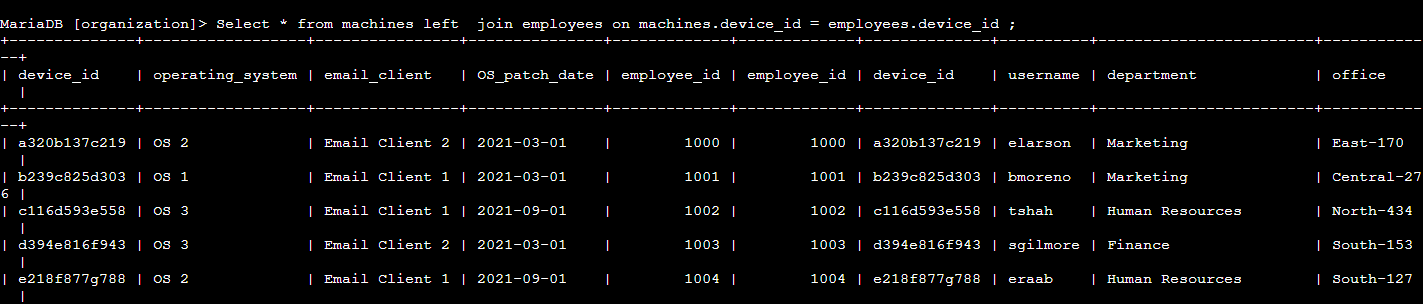
LEFT JOIN (or LEFT OUTER JOIN):

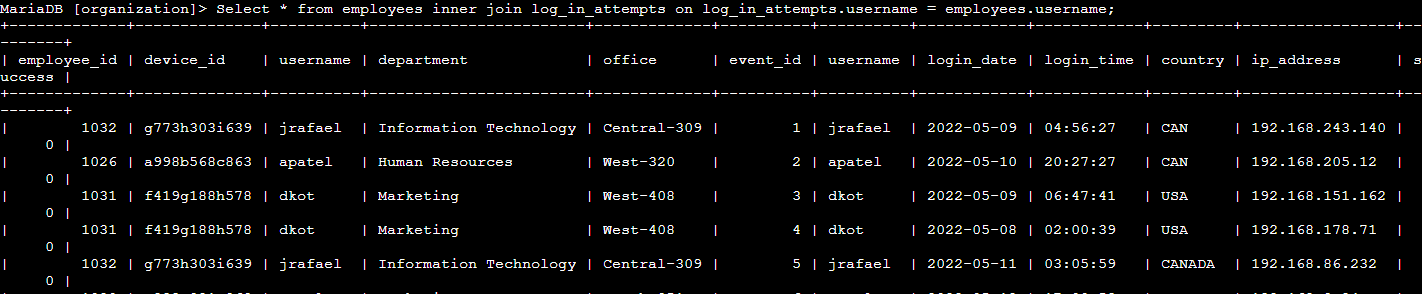
The LEFT JOIN retrieves all the rows from the left table and matching rows from the right table based on the specified join condition. If there is no match for a row in the right table, the result set will still include the row from the left table, and the columns from the right table will be filled with NULL values.

RIGHT JOIN (or RIGHT OUTER JOIN):

The RIGHT JOIN is similar to the LEFT JOIN, but it retrieves all the rows from the right table and matching rows from the left table based on the specified join condition. If there is no match for a row in the left table, the result set will still include the row from the right table, and the columns from the left table will be filled with NULL values.







## Summary

The project involves using SQL queries to filter and retrieve specific data from the "login\_attempts" and "employees" tables. We performed various tasks, including finding failed login attempts after business hours, retrieving login attempts on specific dates, identifying login attempts outside of Mexico, and filtering employees by department and office. By effectively applying SQL filters with AND, OR, and NOT operators, we gained valuable insights into the data, enabling us to investigate security incidents and support our team's security updates on employee machines.