# **Apply filters to SQL queries**

# 

# 

[**Apply filters to SQL queries**](#_adnh333husy)

[Project description 1](#_uakdmh2r7kug)

[Retrieve after hours failed login attempts 1](#_j19vqnsr688d)

[Retrieve login attempts on specific dates 2](#_oox49ujy9cxg)

[Retrieve login attempts outside of Mexico 3](#_ij482iei0lry)

[Retrieve employees in Marketing 4](#_kffkm7d57ava)

[Retrieve employees in Finance or Sales 5](#_lfrxh2nnynwp)

[Retrieve all employees not in IT 6](#_pzhwo5g7pcpy)

[Summary 6](#_6z42dbgzl5pb)

## 

## 

## **Project description**

My team is focused on enhancing our system's security, and I'm responsible for maintaining its safety. My duties include scrutinizing potential security vulnerabilities and updating staff computers as required. Below are instances illustrating how I've employed SQL and filters to execute tasks related to security.

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

There was a potential security incident that occurred after business hours (after 18:00). All after

hours login attempts that failed need to be investigated.

The following query selects all failed login attempts from the log\_in\_attempts table where the login time is after 18:00. The results are ordered by login time.

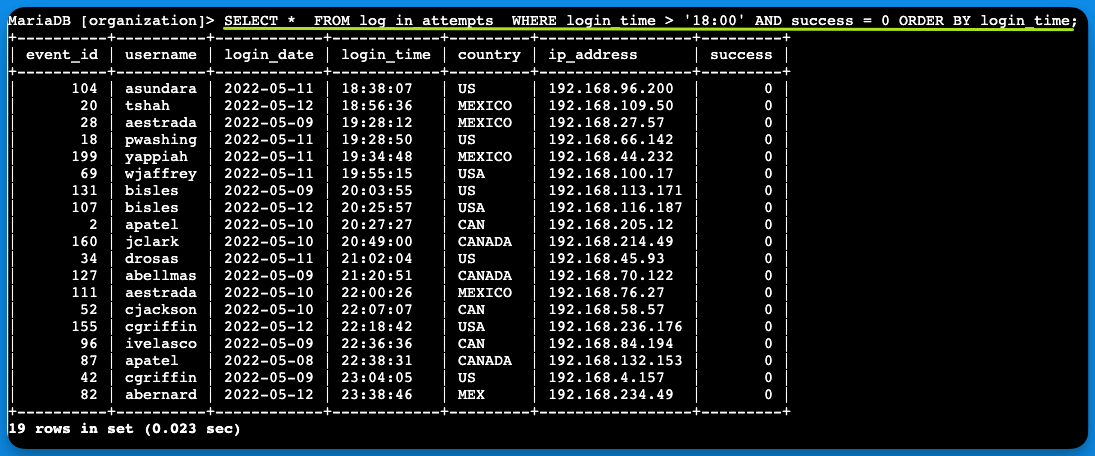
**SELECT \***

**FROM log\_in\_attempts**

**WHERE login\_time > ‘18:00’ AND success = 0;**

explanation of the query:

1. The SELECT statement selects all columns from the log\_in\_attempts table.
2. The WHERE clause filters the results to include only failed login attempts (where success = 0) that occurred after 18:00 (where login\_time > '18:00').
3. The ORDER BY clause orders the results by login time.

****

## 

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

A suspicious event occurred on 2022-05-09. Any login activity that happened on 2022-05-09

or on the day before needs to be investigated.

I created a SQL query to filter for login attempts that occurred on specific dates.

**SELECT \***

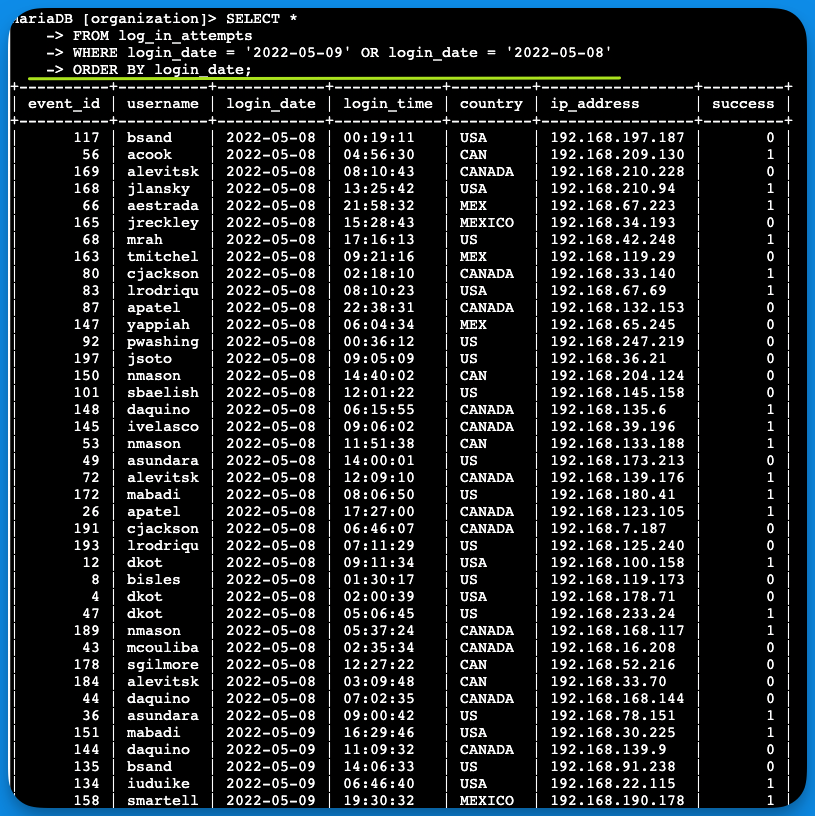
**FROM log\_in\_attempts**

**WHERE login\_date = ‘2022-05-09’ OR logni\_date= ‘2022-05-08’;**

This query selects all login attempts from the log\_in\_attempts table where the login date is either May 8th, 2022 or May 9th, 2022.

Here is a step-by-step explanation of the query:

1. The SELECT statement selects all columns from the log\_in\_attempts table.
2. The WHERE clause filters the results to include only login attempts that occurred on either May 8th, 2022 or May 9th, 2022.
3. The OR operator is used to combine the two date filters.

****

## 

## **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. These login attempts should be investigated.

The following query selects all login attempts from the log\_in\_attempts table where the country is not Mexico.

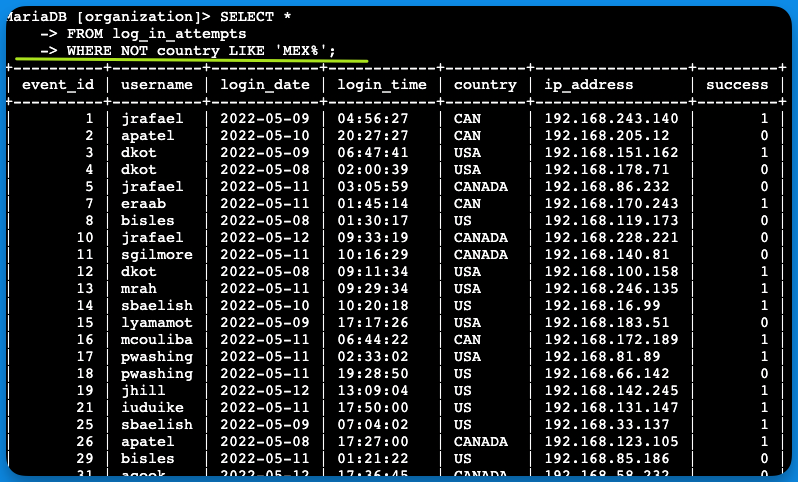
**SELECT \***

**FROM log\_in\_attempts**

**WHERE NOT country LIKE 'MEX%';**

Here is a step-by-step explanation of the query:

1. The SELECT statement selects all columns from the log\_in\_attempts table.
2. The WHERE clause filters the results to include only login attempts where the country is not Mexico.
3. The NOT operator is used to negate the country filter.
4. The LIKE operator is used to compare the country column to the string 'MEX%'. The % wildcard character matches any number of characters.

****

## 

## **Retrieve employees in Marketing**

My team wants to update the computers for certain employees in the Marketing department.

To do this, I have to get information on which employee machines to update.

The following code demonstrates how I created a SQL query to filter for employee machines

from employees in the Marketing department in the East building.

**SELECT \***

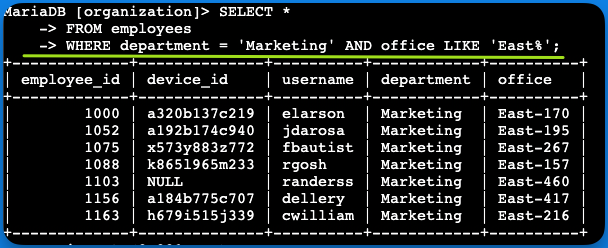
**FROM employees**

**WHERE department = 'Marketing' AND office LIKE 'East%';**

This query returns all columns from the employees table for employees who work in the Marketing department and have an office in the eastern region.

Here is a step-by-step explanation of the query:

1. The SELECT statement selects all columns from the employees table.
2. The WHERE clause filters the results to include only employees where the department is 'Marketing' and the office is in the eastern region.
3. The AND operator is used to combine the two filters.
4. The LIKE operator is used to compare the office column to the string 'East%'. The % wildcard character matches any number of characters

****

## 

## **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 needed, I have to get information on employees only from

these two departments.

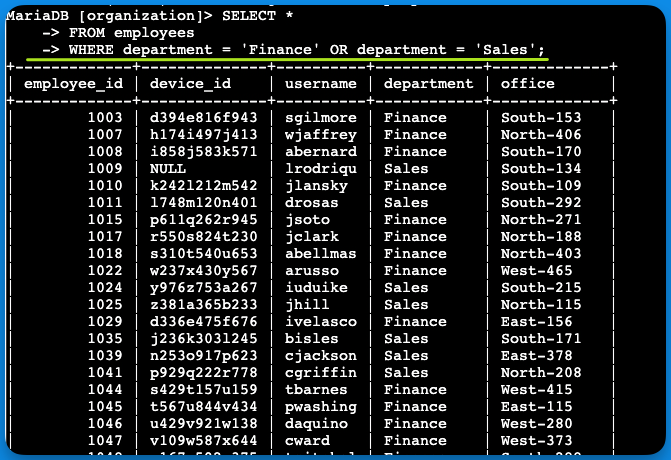
**SELECT \***

**FROM employees**

**WHERE department = 'Finance' OR department = 'Sales';**

The query retrieves all staff members from either the Finance or Sales departments.

Initially, I pulled all the information from the employees table. To narrow down the results, I employed a **WHERE** clause using the **OR** operator to isolate employees in either of the specified departments. I opted for **OR** rather than **AND** to capture employees belonging to either Finance or Sales. The first condition, **department = 'Finance'**, filters for those in the Finance sector, while the second condition, **department = 'Sales'**, does the same for those in Sales.

****

## **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.

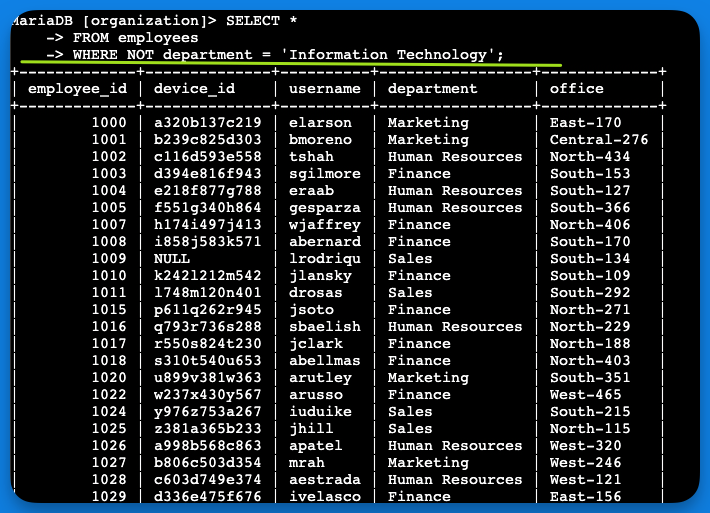
**SELECT \***

**FROM employees**

**WHERE NOT department = 'Information Technology';**

The query returns all employees not in the Information Technology department.

1. First, I started by selecting all data from the employees table.
2. Then, I used a WHERE clause with NOT to filter for employees
3. NOT in this department.

****

## **Summary**

I utilized SQL query filters to extract targeted data on login activities and staff computers from two separate tables: **log\_in\_attempts** and **employees**. I employed a range of operators—**AND**, **OR**, and **NOT**—to fine-tune the information relevant to each task. Additionally, I made use of the **LIKE** function and the percentage sign **(%)** wildcard to identify specific patterns.