# Apply filters to SQL queries

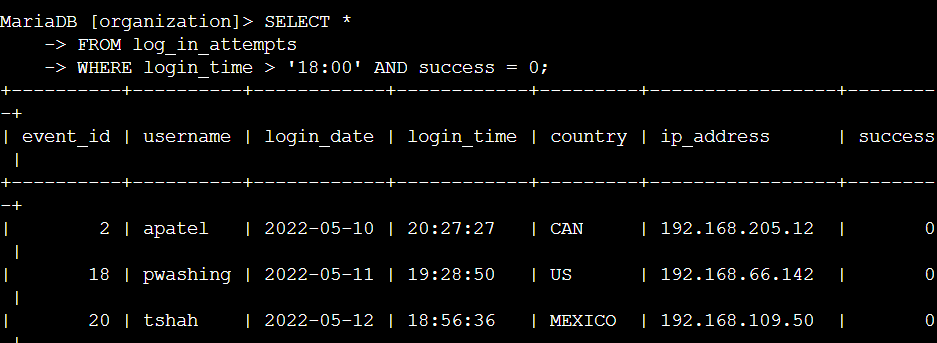
## Project description

As a security professional at a large organization, part of my job is to investigate security issues to keep our systems secure. Recently, I discovered some potential security concerns involving login attempts and employee machines.

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

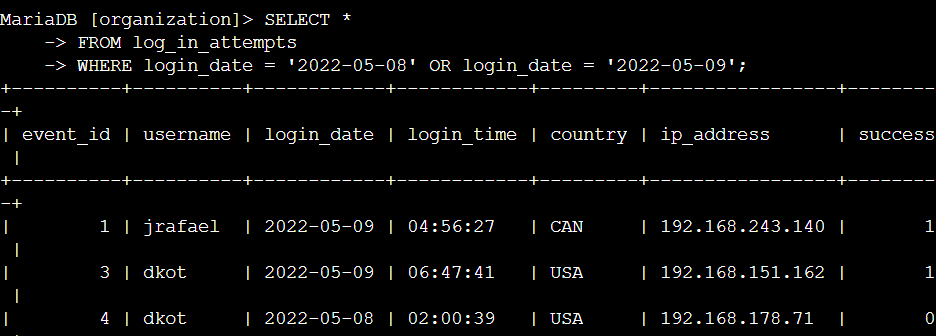
## Retrieve after hours failed login attempts

I recently discovered a potential security incident that occurred after business hours. To investigate this, I need to query the **log\_in\_attempts** table and review after-hours login activity. I’ll use SQL filters to create a query that identifies all failed login attempts that occurred after 18:00. The time of the login attempt is found in the **login\_time** column, and the **success** column contains a value of 0 when a login attempt failed. I can use either a value of 0 or FALSE in my query to identify failed login attempts, I used 0. I needed to use the **AND** operator to filter on multiple conditions to search login attempts that failed after 18:00.



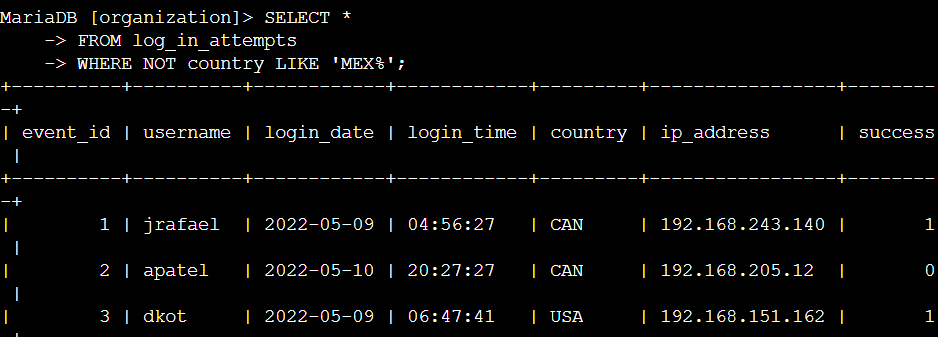
## Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. To investigate this, I want to review all login attempts that happened on this day and the day before. I’ll use SQL filters to create a query that identifies all login attempts that occurred on 2022-05-09 or 2022-05-08. The date of the login attempt is found in the **login\_date** column. In order to do so, I use the operator **OR** to output login attempts on either the 8th or the 9th, because it allows me to specify that either condition can be met.



## Retrieve login attempts outside of Mexico

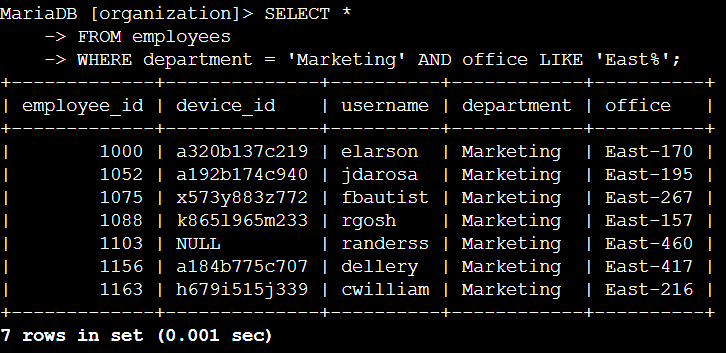
There’s been suspicious activity with login attempts, but the team has determined that this activity didn’t originate in Mexico. Now, I need to investigate login attempts that occurred outside of Mexico. I’ll use SQL filters to create a query that identifies all login attempts that occurred outside of Mexico. When referring to Mexico, the **country** column contains values of both **MEX** and **MEXICO**, so I’ll use the **LIKE** keyword with **%** to ensure my query reflects this. The reason for the **%** is to look for unspecified characters after **MEX**, while **LIKE** is used to search for patterns, which allows me to search for **MEX** and **MEXICO**. The **NOT** operator negates the condition to search all countries except for MEX/MEXICO.



## Retrieve employees in Marketing

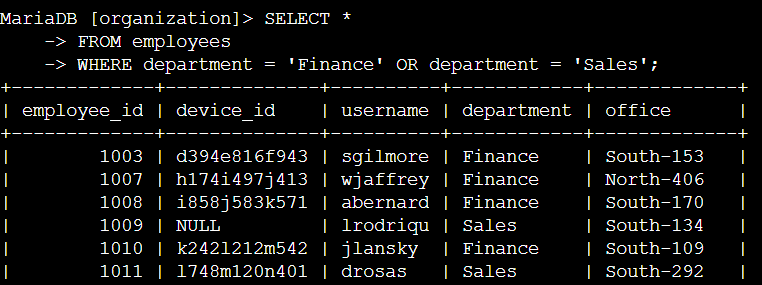
My team wants to perform security updates on specific employee machines in the Marketing department. I’m responsible for gathering information on these employee machines and will need to query the **employees** table. I’ll use SQL filters to create a query that identifies all employees in the Marketing department for all offices in the East building.

The department of the employee is found in the **department** column, which contains values that include Marketing. The office is found in the **office** column. Some examples of values in this column are **East-170**, **East-320**, and **North-434**. I’ll need to use the **LIKE** keyword with **%** to filter for the East building.



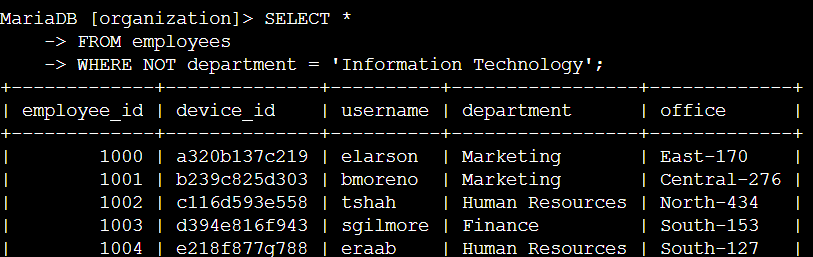
## Retrieve employees in Finance or Sales

My team now needs to perform a different security update on machines for employees in the Sales and Finance departments. I’ll use SQL filters to create a query that identifies all employees in the Sales or Finance departments. The department of the employee is found in the **department** column, which contains values that include **Sales** and **Finance**. I use the **OR** operator because it specifies that either condition can be met.



## Retrieve all employees not in IT

My team needs to make one more update to employee machines. The employees in the Information Technology department have already received this update, but employees in all other departments still need it. I’ll use SQL filters to create a query that identifies all employees not in the IT department. The department of the employee is found in the **department** column, which contains values that include **Information Technology**. I create a simple query by using the **NOT** operator to search all employees outside of the Information Technology department to update their machines.



## Summary

Recently, I used SQL filters to retrieve records and I performed several security related tasks to investigate potential issues. These included retrieving after hours failed login attempts, identifying login attempts on specific dates, and filtering login attempts that occurred outside of Mexico. Additionally, I retrieved information on employees in the Marketing department, employees in the Finance or Sales departments, and all employees not in the IT department.