# Apply filters to SQL queries

## Project description

In our security team, my job as a security analyst is to investigate potential security threats, ensure that our systems are safe, identify and run updates on systems and machines used by employees. In the following steps, I have demonstrated using SQL with filters in performing security-related tasks.

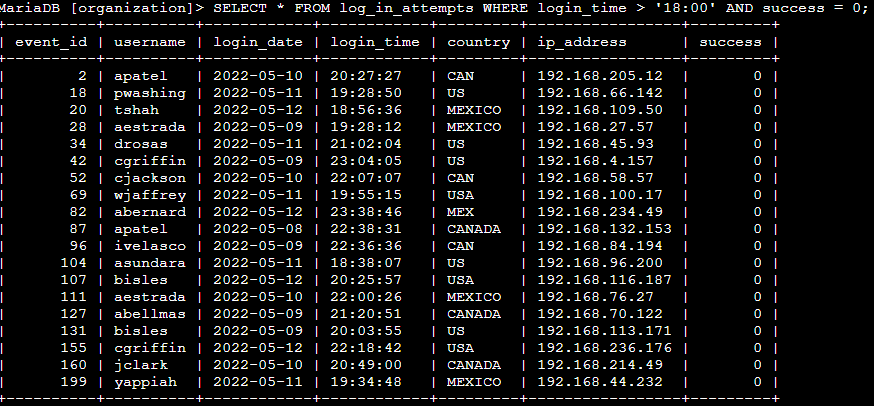
## Retrieve after hours failed login attempts

After business hours 18:00, our SIEM tool notified me of a potential security threat. The first step I took was to examine the failed log in attempts. To do this, I ran the following query using the AND operator.:



This query scanned through the log\_in\_attempts table and retrieved all failed login attempts that occurred after 18:00. I began by selecting all login failed attempts using the SELECT \* and FROM query inputs. Then I used the WHERE clause together with the AND operator to filter my results. The first condition login\_time > ’18:00’ filtered for events that were recorded after the set time, while the second condition success > 0 showed unsuccessful attempts to log into the system. Such failed attempts might be indicative of a malicious actor trying to brute force their way into the system.

The query returned the following 19 rows of failed login attempts.

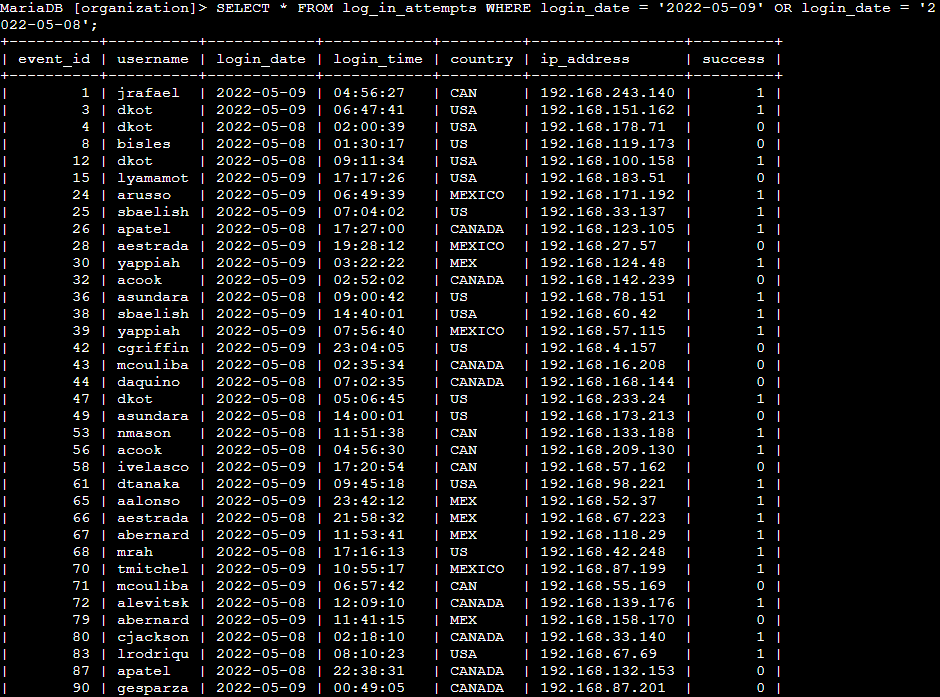


## Retrieve login attempts on specific dates

Going through the system, I also realised that a suspicious event occurred on the 2022-05-09. I launched an investigation into the event by running a query that identified all login attempts which occurred on the day and the day before. To do this, I ran the following query that identified login attempts that happened on the 2022-05-09 or on the 2022-05-08.



I began by selecting all data from the log\_in\_attempts dataset. I then used a WHERE clause with the operator OR to filter the results. The first condition login\_date = ‘2022-05-09’returned login attempts made on the 2022-05-09 whereas the second condition login\_date = ‘2022-05-08’ filtered for attempts made on 2022-05-08. Unlike the AND operator in which both conditions must be satisfied simultaneously, using the OR operator ensures that one or both conditions can be met. The query returned the following result.



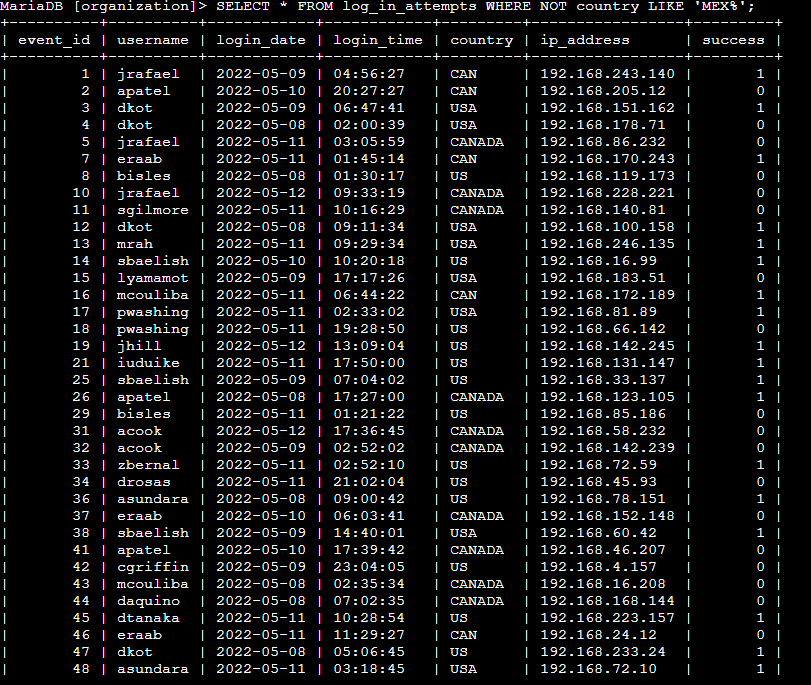
## Retrieve login attempts outside of Mexico

Here I investigated login attempts that occurred outside Mexico because the security team determined that the suspicious activity did not originate in Mexico. I used filters in SQL to identify all login attempts that occurred outside Mexico. I ran the query:



Since entries in the database represented Mexico using a mixture of MEX and MEXICO, using the % operator helped me filter for unspecified characters. Using the NOT operator reduced the query complexity by simply producing all login entries that did not originate from Mexico.

The query returned 144 rows from the log\_in\_attempts table. An excerpt is provided below:

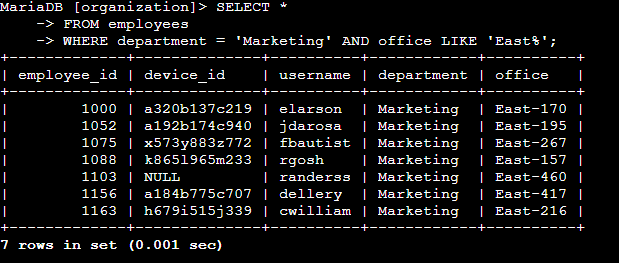


## Retrieve employees in Marketing

The following week, my team decided to perform security updates on speif employee machines in the marketing department. My responsiblilty is to get information on all the employee machines. To do this, I had to query the employees table using SQL filters to create a query that identifies all employees in the marketing department for all offices in the East building. I ran the following query on the employees table:



This query scans the employees table to output details of all employees in the marketing department. The LIKE and %(Modulo) operators are used to filter for unspecified characters. It is characteristic to use them together since the % is incompatible with the = operator. The query produced the following result:

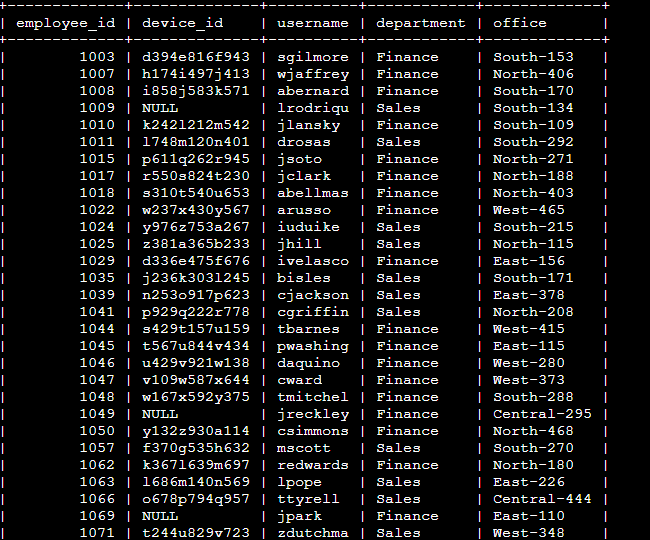


## Retrieve employees in Finance or Sales

Next, I was tasked with querying the employee table to identify the employees in the sales and finance departments as their computers need a different security update. Pretty straightforward, I ran the query:



The OR operator retrieves one or both of the conditions provided they are satisfied. In this situation, it returned data on both the employees in the sales and finance departments. A snippet of the results is shown below:



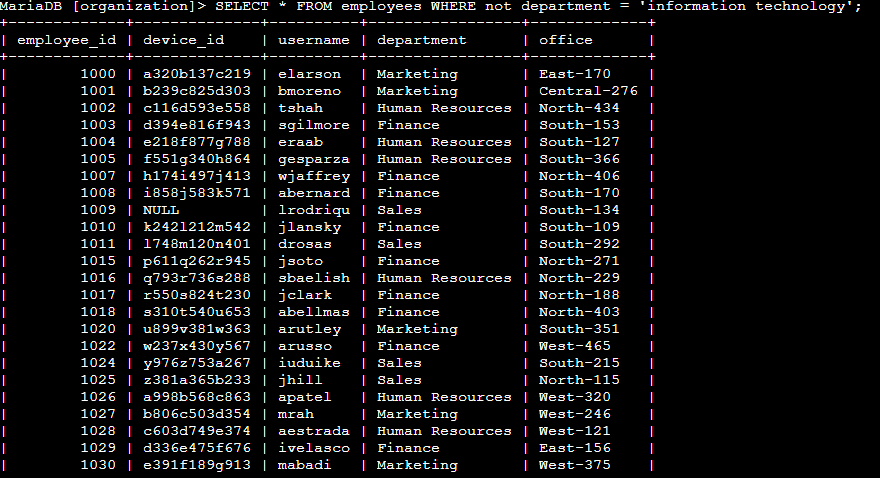
## Retrieve all employees not in IT

The team still needed to make just one more update on the machines of employees who are not in the information technology department since machines in IT had already received the updates. Hence, I needed to run a query that will return detailed information on all employees not In the IT department.



The first part of the screenshot is my query while the screenshot provided below is the output. I began by selecting all columns and rows from the employee table. I then used a WHERE clause with the NOT operator to filter for employees that are not in IT.

Running this query returned data on all other employees are not in the IT department.



## Summary

In this task, I used SQL to query a database to identify potential security threats involving login attempts and employee machines. Using the LIKE, AND, OR and % operators to query the dataset provided richer insights into the source and patterns of the event. I was able to use them in filtering on multiple conditions. The project also demonstrated my ability to use SQL to retrieve information on employees and their machines which is a key part of security audits / assessment.