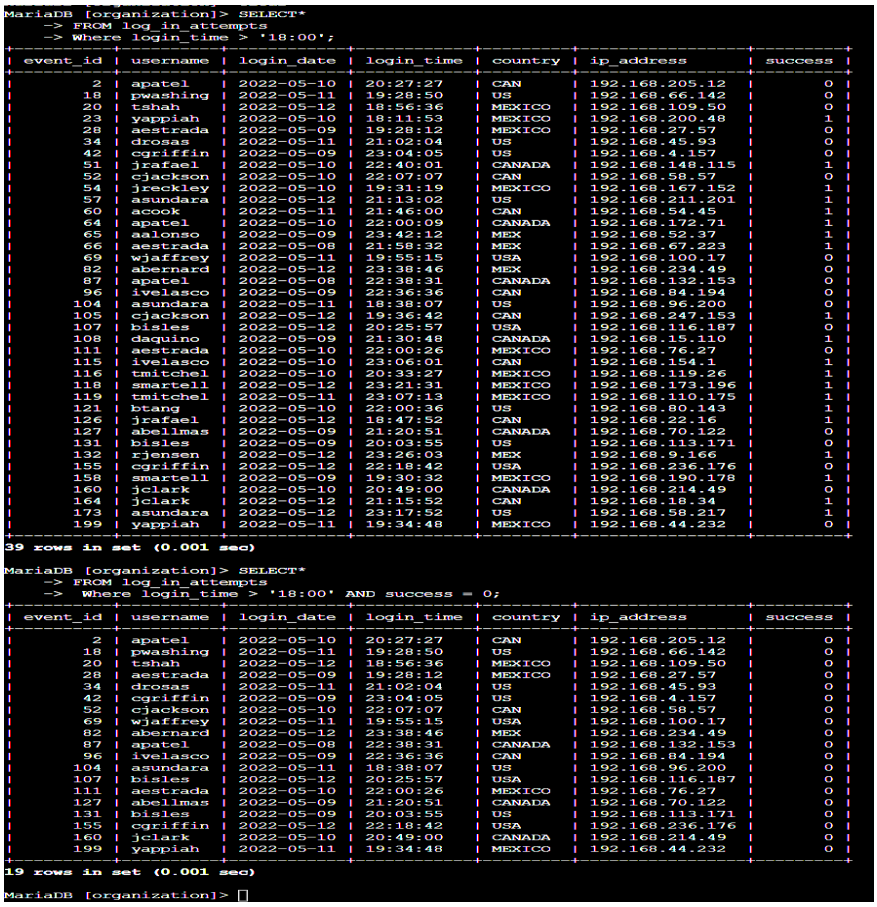
# Apply filters to SQL queries

Name: Brian Trujillo

## Project description

In this project, I will employ SQL to investigate and analyze security-related events within my organization's database, particularly focusing on login attempts and employee machine activities. By constructing targeted SQL queries, I will filter and retrieve pertinent records from the 'employees' and 'log_in_attempts' tables. My goal is to identify patterns or anomalies that indicate potential security breaches or vulnerabilities.This analysis is essential for upholding our security protocols, quickly resolving any issues, and strengthening a system's protection.

## Retrieve after hours failed login attempts



```
MariaDB [organization]> SELECT*
    -> FROM log_in_attempts
    -> Where login_time > '18:00';
+----------+----------+------------+------------+---------+----------------+---------+
| event_id | username | login_date | login_time | country | ip_address     | success |
+----------+----------+------------+------------+---------+----------------+---------+
|        2 | apatel   | 2022-05-10 | 20:27:27   | CAN     | 192.168.205.12 |       0 |
|       18 | pwashing | 2022-05-11 | 19:28:50   | US      | 192.168.66.142 |       0 |
|       20 | tshah    | 2022-05-12 | 18:56:36   | MEXICO  | 192.168.109.50 |       0 |
|       23 | yappiah  | 2022-05-10 | 18:11:53   | MEXICO  | 192.168.200.48 |       1 |
|       28 | aestrada | 2022-05-09 | 19:28:12   | MEXICO  | 192.168.27.57  |       0 |
|       34 | drosas   | 2022-05-11 | 21:02:04   | US      | 192.168.45.93  |       0 |
|       42 | cgriffin | 2022-05-09 | 23:04:05   | US      | 192.168.4.157  |       0 |
|       51 | jrafael  | 2022-05-10 | 22:40:01   | CANADA  | 192.168.148.115|       1 |
|       52 | cjackson | 2022-05-10 | 22:07:07   | CAN     | 192.168.58.57  |       0 |
|       54 | jreckley | 2022-05-10 | 19:31:19   | MEXICO  | 192.168.167.152|       1 |
|       57 | asundara | 2022-05-12 | 21:13:02   | US      | 192.168.211.201|       1 |
|       60 | acook    | 2022-05-11 | 21:45:00   | CAN     | 192.168.54.45  |       1 |
|       64 | apatel   | 2022-05-10 | 22:00:09   | CANADA  | 192.168.172.71 |       1 |
|       65 | aalonso  | 2022-05-09 | 23:42:12   | MEX     | 192.168.52.37  |       1 |
|       66 | aestrada | 2022-05-08 | 21:58:32   | MEX     | 192.168.67.223 |       1 |
|       69 | wjaffrey | 2022-05-11 | 19:55:15   | USA     | 192.168.100.17 |       0 |
|       82 | abernard | 2022-05-12 | 23:38:46   | MEX     | 192.168.234.49 |       0 |
|       87 | apatel   | 2022-05-08 | 22:38:31   | CANADA  | 192.168.132.153|       0 |
|       96 | ivelasco | 2022-05-09 | 22:36:36   | CAN     | 192.168.84.194 |       0 |
|      104 | asundara | 2022-05-11 | 18:38:07   | US      | 192.168.96.200 |       0 |
|      105 | cjackson | 2022-05-12 | 19:36:42   | CAN     | 192.168.247.153|       1 |
|      107 | bisles   | 2022-05-12 | 20:25:57   | USA     | 192.168.116.187|       0 |
|      108 | daquino  | 2022-05-09 | 21:30:48   | CANADA  | 192.168.15.110 |       1 |
|      111 | aestrada | 2022-05-10 | 22:00:26   | MEXICO  | 192.168.76.27  |       0 |
|      115 | ivelasco | 2022-05-10 | 23:06:01   | CAN     | 192.168.154.1  |       1 |
|      116 | tmitchel | 2022-05-10 | 20:33:27   | MEXICO  | 192.168.119.26 |       1 |
|      118 | smartell | 2022-05-12 | 23:21:31   | MEXICO  | 192.168.173.196|       1 |
|      119 | tmitchel | 2022-05-11 | 23:07:13   | MEXICO  | 192.168.110.175|       1 |
|      121 | btang    | 2022-05-10 | 22:00:36   | US      | 192.168.80.143 |       1 |
|      126 | jrafael  | 2022-05-12 | 18:47:52   | CAN     | 192.168.22.16  |       1 |
|      127 | abellmas | 2022-05-09 | 21:20:51   | CANADA  | 192.168.70.122 |       0 |
|      131 | bisles   | 2022-05-09 | 20:03:55   | US      | 192.168.113.171|       0 |
|      132 | rjensen  | 2022-05-12 | 23:26:03   | MEX     | 192.168.9.166  |       1 |
|      155 | cgriffin | 2022-05-12 | 22:18:42   | USA     | 192.168.236.176|       0 |
|      158 | smartell | 2022-05-09 | 19:30:32   | MEXICO  | 192.168.190.178|       1 |
|      160 | jclark   | 2022-05-10 | 20:49:00   | CANADA  | 192.168.214.49 |       0 |
|      164 | jclark   | 2022-05-12 | 21:15:52   | CAN     | 192.168.18.34  |       1 |
|      173 | asundara | 2022-05-12 | 23:17:52   | US      | 192.168.58.217 |       1 |
|      199 | yappiah  | 2022-05-11 | 19:34:48   | MEXICO  | 192.168.44.232 |       0 |
+----------+----------+------------+------------+---------+----------------+---------+
39 rows in set (0.001 sec)

MariaDB [organization]> SELECT*
    -> FROM log_in_attempts
    ->  Where login_time > '18:00' AND success = 0;
+----------+----------+------------+------------+---------+----------------+---------+
| event_id | username | login_date | login_time | country | ip_address     | success |
+----------+----------+------------+------------+---------+----------------+---------+
|        2 | apatel   | 2022-05-10 | 20:27:27   | CAN     | 192.168.205.12 |       0 |
|       18 | pwashing | 2022-05-11 | 19:28:50   | US      | 192.168.66.142 |       0 |
|       20 | tshah    | 2022-05-12 | 18:56:36   | MEXICO  | 192.168.109.50 |       0 |
|       28 | aestrada | 2022-05-09 | 19:28:12   | MEXICO  | 192.168.27.57  |       0 |
|       34 | drosas   | 2022-05-11 | 21:02:04   | US      | 192.168.45.93  |       0 |
|       42 | cgriffin | 2022-05-09 | 23:04:05   | US      | 192.168.4.157  |       0 |
|       52 | cjackson | 2022-05-10 | 22:07:07   | CAN     | 192.168.58.57  |       0 |
|       69 | wjaffrey | 2022-05-11 | 19:55:15   | USA     | 192.168.100.17 |       0 |
|       82 | abernard | 2022-05-12 | 23:38:46   | MEX     | 192.168.234.49 |       0 |
|       87 | apatel   | 2022-05-08 | 22:38:31   | CANADA  | 192.168.132.153|       0 |
|       96 | ivelasco | 2022-05-09 | 22:36:36   | CAN     | 192.168.84.194 |       0 |
|      104 | asundara | 2022-05-11 | 18:38:07   | US      | 192.168.96.200 |       0 |
|      107 | bisles   | 2022-05-12 | 20:25:57   | USA     | 192.168.116.187|       0 |
|      111 | aestrada | 2022-05-10 | 22:00:26   | MEXICO  | 192.168.76.27  |       0 |
|      127 | abellmas | 2022-05-09 | 21:20:51   | CANADA  | 192.168.70.122 |       0 |
|      131 | bisles   | 2022-05-09 | 20:03:55   | US      | 192.168.113.171|       0 |
|      155 | cgriffin | 2022-05-12 | 22:18:42   | USA     | 192.168.236.176|       0 |
|      160 | jclark   | 2022-05-10 | 20:49:00   | CANADA  | 192.168.214.49 |       0 |
|      199 | yappiah  | 2022-05-11 | 19:34:48   | MEXICO  | 192.168.44.232 |       0 |
+----------+----------+------------+------------+---------+----------------+---------+
19 rows in set (0.001 sec)

MariaDB [organization]>
```

The screenshot captures two SQL queries I ran on our MariaDB database to scrutinize login attempts made after 6:00 PM. I needed to sift through the 'log_in_attempts' table to spot any unusual or unauthorized login attempts that could indicate a security threat.

For the initial query, I pulled all records of login attempts made post-6:00 PM using
SELECT *
FROM log_in_attempts
WHERE login_time > '18:00';

This gave me a comprehensive view of all evening login activities, including successful and unsuccessful attempts, along with pertinent information like usernames, IP addresses, and whether the login was successful.

I refined my search with a second query:
SELECT *
FROM log_in_attempts
WHERE login_time > '18:00' AND success = 0;

This time, I focused solely on the failed login attempts during the same time frame. The result, 19 rows of data, highlights the failed attempts, which now warrants a closer look to identify any patterns or anomalies such as multiple failed attempts from a single IP address or consistent failures against specific user accounts, which could potentially flag a security issue like a brute force attack.

## Retrieve login attempts on specific dates

```
MariaDB [organization]> SELECT*
    -> FROM log_in_attempts
    -> WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        1 | jrafael  | 2022-05-09 | 04:56:27   | CAN     | 192.168.243.140 |       1 |
|        3 | dkot     | 2022-05-09 | 06:47:41   | USA     | 192.168.151.162 |       1 |
|        4 | dkot     | 2022-05-08 | 02:00:39   | USA     | 192.168.178.71  |       0 |
|        8 | bisles   | 2022-05-08 | 01:30:17   | US      | 192.168.119.173 |       0 |
|       12 | dkot     | 2022-05-08 | 09:11:34   | USA     | 192.168.100.158 |       1 |
|       15 | lyamamot | 2022-05-09 | 17:17:26   | USA     | 192.168.183.51  |       0 |
|       24 | arusso   | 2022-05-09 | 06:49:39   | MEXICO  | 192.168.171.192 |       1 |
|       25 | sbaelish | 2022-05-09 | 07:04:02   | US      | 192.168.33.137  |       1 |
```

Performed a SQL query to pull records from the 'log_in_attempts' table for a targeted review of login activities on specific days. The query I used was:

```
SELECT *
FROM log_in_attempts
WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
```

This query selects all columns (*) from the log_in_attempts table where the login_date column matches either May 9th, 2022, or May 8th, 2022. The use of OR in the WHERE clause allows the query to return results for either of the two dates specified, effectively widening the search to cover a two-day period.

The output from this query provides a comprehensive list of all login attempts made on those two specific dates, along with associated details such as event_id, username, login_date, login_time, country, ip_address, and a success flag indicating whether the login attempt was successful (1) or not (0).

This command is utilized to focus the investigation on login activity within a defined timeframe, which can be crucial for identifying patterns related to security breaches or pinpointing the timeframe of suspicious activities.

## Retrieve login attempts outside of Mexico

```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE NOT country LIKE 'MEX%';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        1 | jrafael  | 2022-05-09 | 04:56:27   | CAN     | 192.168.243.140 |       1 |
|        2 | apatel   | 2022-05-10 | 20:27:27   | CAN     | 192.168.205.12  |       0 |
|        3 | dkot     | 2022-05-09 | 06:47:41   | USA     | 192.168.151.162 |       1 |
|        4 | dkot     | 2022-05-08 | 02:00:39   | USA     | 192.168.178.71  |       0 |
|        5 | jrafael  | 2022-05-11 | 03:05:59   | CANADA  | 192.168.86.232  |       0 |
|        7 | eraab    | 2022-05-11 | 01:45:14   | CAN     | 192.168.170.243 |       1 |
|        8 | bisles   | 2022-05-08 | 01:30:17   | US      | 192.168.119.173 |       0 |
|       10 | jrafael  | 2022-05-12 | 09:33:19   | CANADA  | 192.168.228.221 |       0 |
|       11 | sgilmore | 2022-05-11 | 10:16:29   | CANADA  | 192.168.140.81  |       0 |
|       12 | dkot     | 2022-05-08 | 09:11:34   | USA     | 192.168.100.158 |       1 |
|       13 | mrah     | 2022-05-11 | 09:29:34   | USA     | 192.168.246.135 |       1 |
```

I executed an SQL command to filter out login records from the log_in_attempts table that did not originate from Mexico.

```
SELECT*
```

FROM log_in_attempts
WHERE NOT country LIKE 'MEX%';

With this query, I've pulled all relevant data from the table but excluded any entries where the 'country' field begins with 'MEX', effectively narrowing down the data to only include login attempts from outside Mexico. his ensures that the query results will only include login attempts that have occurred in countries other than Mexico, allowing me to focus on analyzing the suspicious activities originating from other regions

## Retrieve employees in Marketing

```
MariaDB [organization]> SELECT*
    -> FROM employees
    -> WHERE department = 'Marketing' AND office LIKE 'East%';
+-------------+--------------+----------+------------+----------+
| employee_id | device_id    | username | department | office   |
+-------------+--------------+----------+------------+----------+
|        1000 | a320b137c219 | elarson  | Marketing  | East-170 |
|        1052 | a192b174c940 | jdarosa  | Marketing  | East-195 |
|        1075 | x573y883z772 | fbautist | Marketing  | East-267 |
|        1088 | k8651965m233 | rgosh    | Marketing  | East-157 |
|        1103 | NULL         | randerss | Marketing  | East-460 |
|        1156 | a184b775c707 | dellery  | Marketing  | East-417 |
|        1163 | h679i515j339 | cwilliam | Marketing  | East-216 |
+-------------+--------------+----------+------------+----------+
7 rows in set (0.022 sec)
```

Conducting a query on employee machines specifically in the Marketing department located in the East building, I crafted an SQL query to fetch the necessary information from our 'employees' table. My query was as follows:

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

This query retrieves all records from the 'employees' table where the 'department' is exactly 'Marketing'. Additionally, it filters these results to include only those whose 'office' designation begins with 'East', which is indicated by the 'LIKE' keyword followed by 'East%'.

Running this query, I successfully obtained a complete list of Marketing department employees located in the East building. The resulting data, which includes employee IDs, device IDs, usernames, and specific office codes.A search such as this can be useful in many scenarios,

one being the next step of a security update process, ensuring a team to precisely target the machines that require updates.

## Retrieve employees in Finance or Sales

```
MariaDB [organization]> SELECT*
    -> FROM employees
    -> WHERE department = 'Finance' OR department = 'Sales';
+-------------+-------------+----------+------------+-------------+
| employee_id | device_id   | username | department | office      |
+-------------+-------------+----------+------------+-------------+
|        1003 | d394e816f943 | sgilmore | Finance    | South-153   |
|        1007 | h174i497j413 | wjaffrey | Finance    | North-406   |
|        1008 | i858j583k571 | abernard | Finance    | South-170   |
|        1009 | NULL         | lrodriqu | Sales      | South-134   |
|        1010 | k2421212m542 | jlansky  | Finance    | South-109   |
|        1011 | l748m120n401 | drosas   | Sales      | South-292   |
```

To carry out necessary security updates for our team's machines in both the Sales and Finance departments, I created and ran an SQL query against our 'employees' table. The query I used was:

SELECT*
FROM employees
WHERE department = 'Finance' OR department = 'Sales';

This command allowed me to obtain a complete list of employees who belong to either the Finance or Sales departments. I used the OR condition in the WHERE clause to ensure that my result set included employees from both departments, not just one. The output from the query provided me with the employee IDs, device IDs, usernames, departments, and office locations for these individuals. With this data at hand. We can use this data to proceed with the security updates, specifically targeting the machines used by the employees in these departments to enhance our cybersecurity defenses efficiently.

## Retrieve all employees not in IT

```
MariaDB [organization]> SELECT*
    -> FROM employees
    -> WHERE NOT department = 'Information Technology';
+-------------+-------------+----------+----------------+-------------+
| employee_id | device_id   | username | department     | office      |
+-------------+-------------+----------+----------------+-------------+
|        1000 | a320b137c219 | elarson  | Marketing      | East-170    |
|        1001 | b239c825d303 | bmoreno  | Marketing      | Central-276 |
|        1002 | c116d593e558 | tshah    | Human Resources | North-434   |
|        1003 | d394e816f943 | sgilmore | Finance        | South-153   |
|        1004 | e218f877g788 | eraab    | Human Resources | South-127   |
|        1005 | f551g340h864 | gesparza | Human Resources | South-366   |
|        1007 | h174i497j413 | wjaffrey | Finance        | North-406   |
|        1008 | i858j583k571 | abernard | Finance        | South-170   |
|        1009 | NULL         | lrodriqu | Sales          | South-134   |
|        1010 | k2421212m542 | jlansky  | Finance        | South-109   |
|        1011 | l748m120n401 | drosas   | Sales          | South-292   |
|        1015 | p611q262r945 | jsoto    | Finance        | North-271   |
|        1016 | q793r736s288 | sbaelish | Human Resources | North-229   |
|        1017 | r550s824t230 | jclark   | Finance        | North-188   |
|        1018 | s310t540u653 | abellmas | Finance        | North-403   |
|        1020 | u899v381w363 | arutley  | Marketing      | South-351   |
```

To perform the necessary security updates for employee machines outside the Information Technology department, I constructed an SQL query to identify all relevant employees. The query I used was:

SELECT *
FROM employees
WHERE NOT department = 'Information Technology';

With this command, I extracted a complete dataset from the employees table that excluded anyone within the IT department. The key part of this query was the NOT operator, which ensured that my results only included departments other than IT.  Reason being is the IT department had already received the security update, and now needed to extend this update to the rest of the company.

The output from the query provided a clear list of employee IDs, device IDs, usernames, departments, and office locations, enabling me to systematically target the remaining departments for the security update. This step is vital to maintain a high level of cybersecurity across the entire organization, ensuring every department, other than IT, gets the latest security enhancements on their machines.

## Summary

Throughout this project, I successfully used SQL queries to filter for login attempts after business hours and exclude those originating from Mexico, based on the team's security focus. Additionally, I identified employees outside the IT department who needed security updates, ensuring comprehensive coverage. These tasks demonstrated my ability to apply SQL filters effectively, showcasing attention to detail and commitment to maintaining security protocols within the organization.