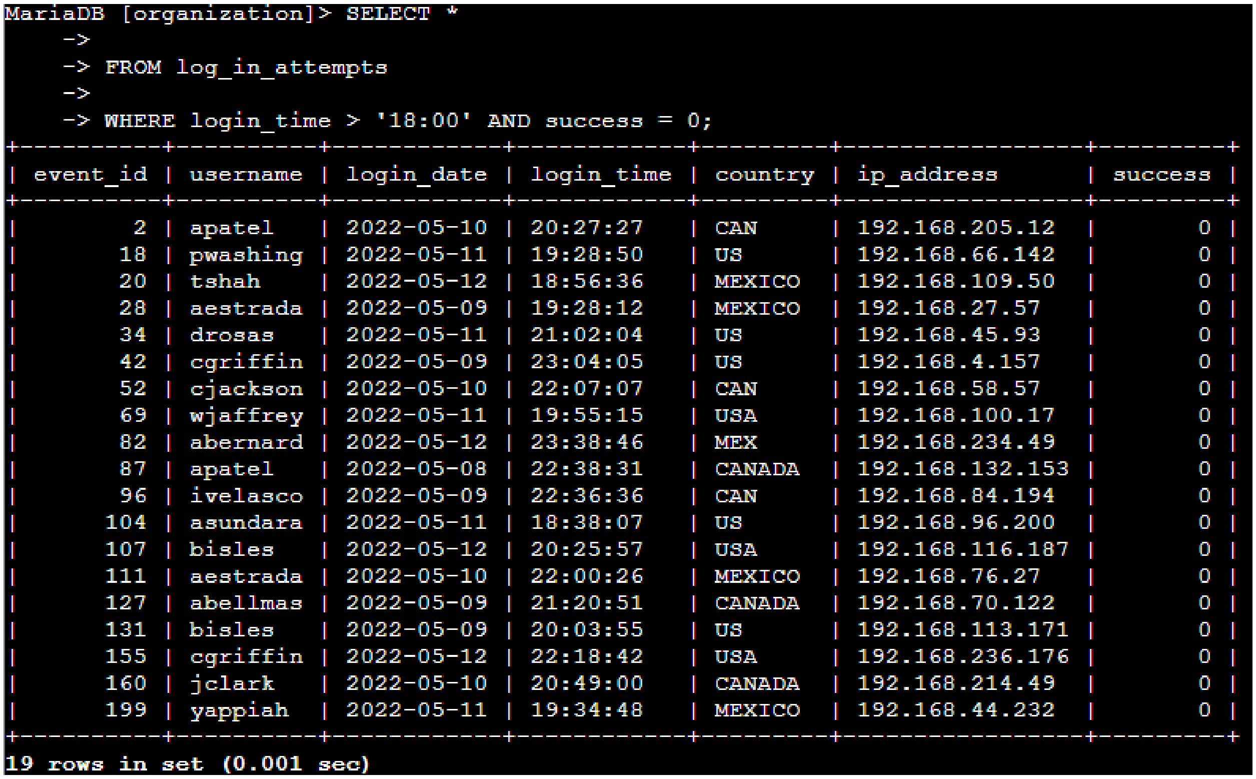
Apply filters to SQL queries

Project description

As a security analyst for this project, I use a combination of SQL queries to filter and find specific information needed for security purposes.

Retrieve after-hours failed login attempts

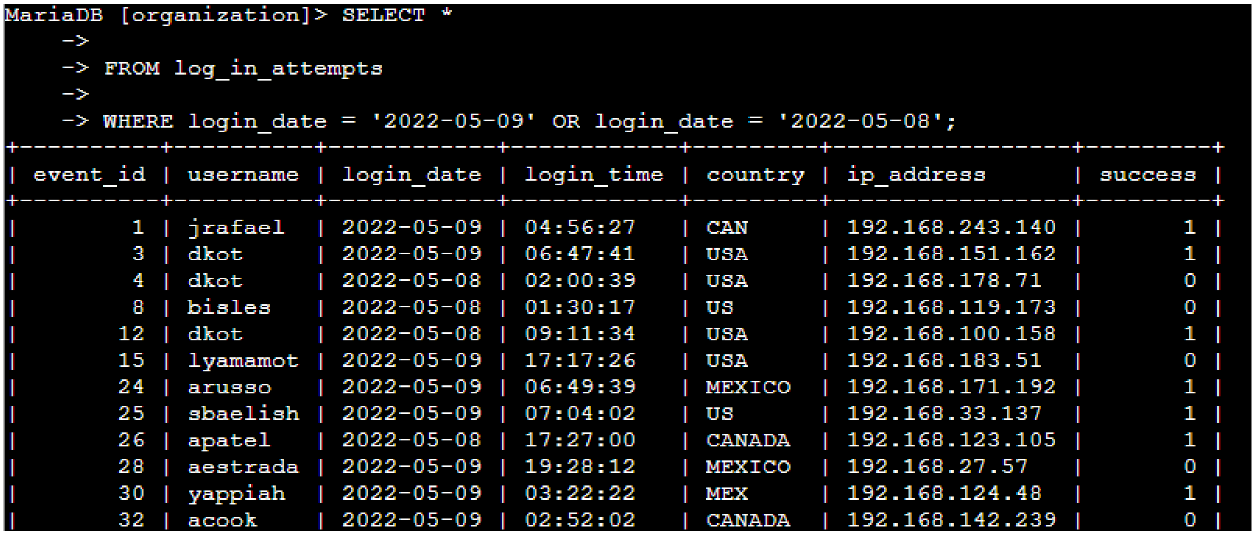
A potential security incident occurred after hours and I needed to investigate the failed log_in_attempts after hours. My query works by retrieving data from the log_in_attemps table. This is done by filtering the login_time and success columns by entries after office hours at ‘18:00’ and successful logins with a 1 representing TRUE and a 0 representing FALSE. The results are below where you can see I used the WHERE clause with an AND operator to filter the correct results.



```
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)
```

Retrieve login attempts on specific dates

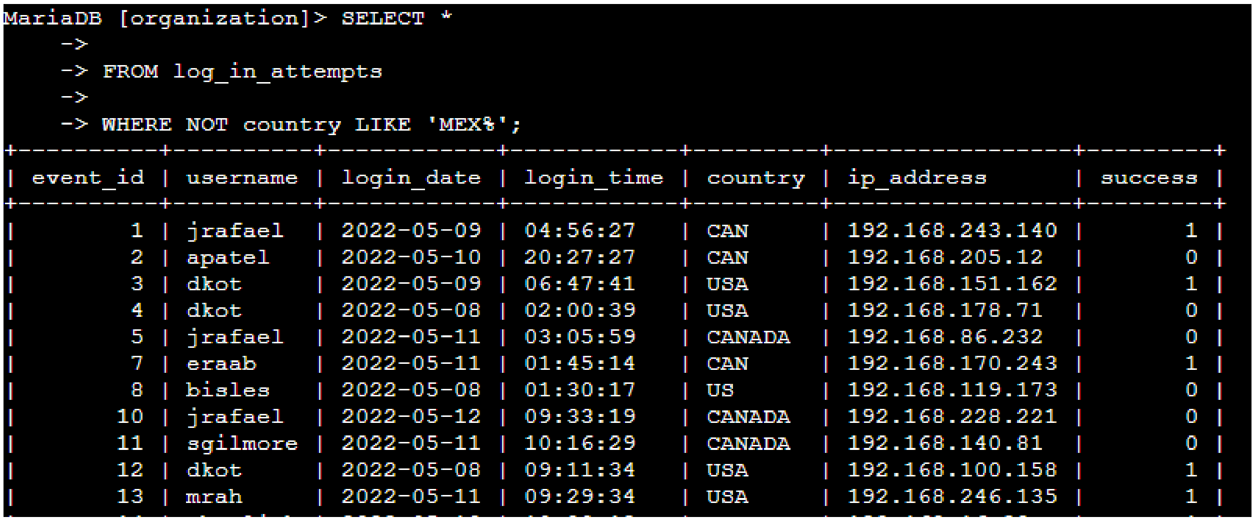
Suspicious activity occurred on ‘2022-05-09’ and I was tasked with viewing the login attempts on this day and the previous day. To do so I use the same WHERE clause before on the log_in_attempts table along with the OR operator to gather entries that are on the 8th or the 9th.



```
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 |
|       26 | apatel   | 2022-05-08 | 17:27:00   | CANADA  | 192.168.123.105 |       1 |
|       28 | aestrada | 2022-05-09 | 19:28:12   | MEXICO  | 192.168.27.57   |       0 |
|       30 | yappiah  | 2022-05-09 | 03:22:22   | MEX     | 192.168.124.48  |       1 |
|       32 | acook    | 2022-05-09 | 02:52:02   | CANADA  | 192.168.142.239 |       0 |
```

Retrieve login attempts outside of Mexico

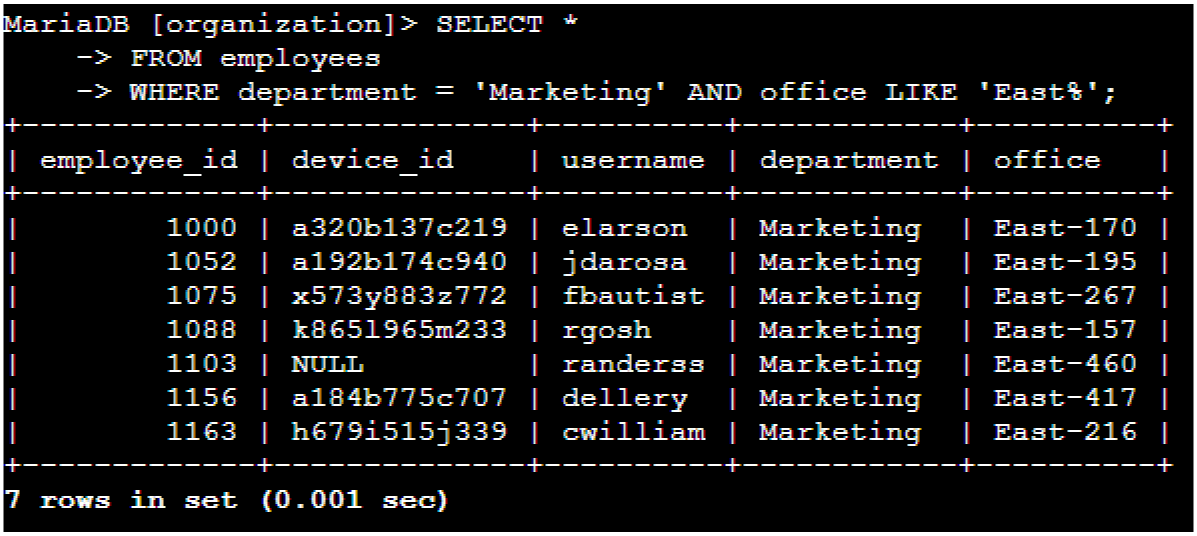
For this, my team looked into login entries that did not originate in Mexico. We needed not to include entries with ‘MEX’ and ‘MEXICO’ to do this we can use the matching pattern ‘MEX%’ which means that the string has to start with MEX but can end with anything. To disclude the countries we used the WHERE clause along with the NOT and LIKE operators.



```
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 |
```

Retrieve employees in Marketing

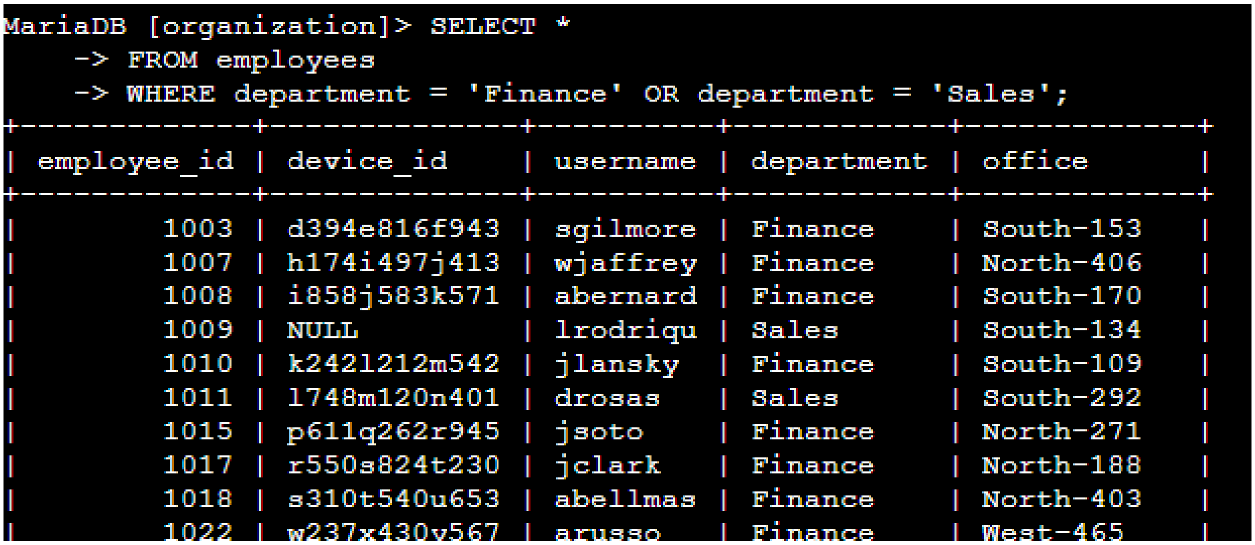
My team needed information to help update employees in the Marketing department, which is located in the East building. To do so we use the WHERE clause, AND, and LIKE operators to filter the department and office locations.



```
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 (0.001 sec)
```

Retrieve employees in Finance or Sales

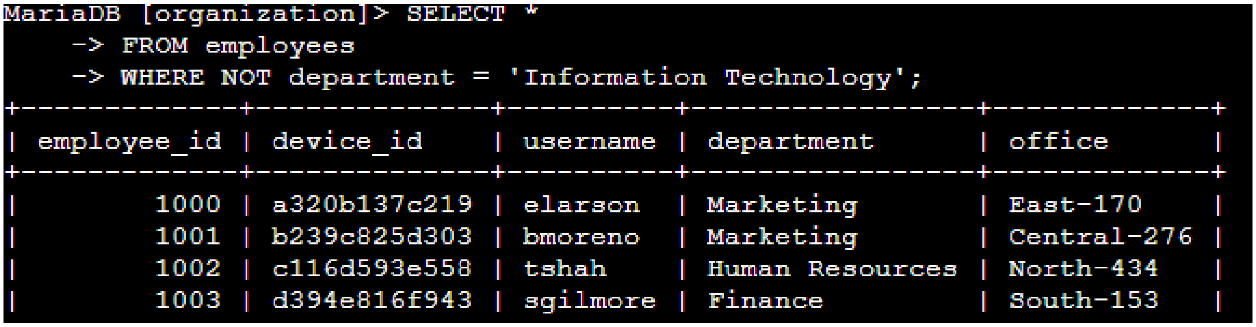
My team needed to retrieve all employees in the Finance or Sales department to perform some updates. To do this my team filters the employee's table using the WHERE clause and OR operator to find all employees in either department.



```
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 | 1748m120n401  | drosas   | Sales       | South-292     |
|        1015 | p611q262r945  | jsoto    | Finance     | North-271     |
|        1017 | r550s824t230  | jclark   | Finance     | North-188     |
|        1018 | s310t540u653  | abellmas | Finance     | North-403     |
|        1022 | w237x430y567  | arusso   | Finance     | West-465      |
```

Retrieve all employees not in IT

My team needed to make an update to all employee computers. However, the update has been made to all the IT employees so we need to filter out all employees that are not in the IT departments. To do so we use a simple NOT operator to filter employees.



```
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   |
```

Summary

I used various SQL queries to filter and gather information about information on login attempts and employee records. I used the log_in_attempts and employees tables to do so. I used many SQL operators like AND, OR, and NOT to filter information. Additionally using the LIKE and ‘%” wildcard to filter for various patterns.