Apply filters to SQL queries

Project description

My organization is working to make their system more secure. It is my job to ensure the system is safe, investigates all potential security issues, and updates employee computers as needed. The following steps provide examples of how I used SQL with filters to perform security-related tasks.

Retrieve after hours failed login attempts

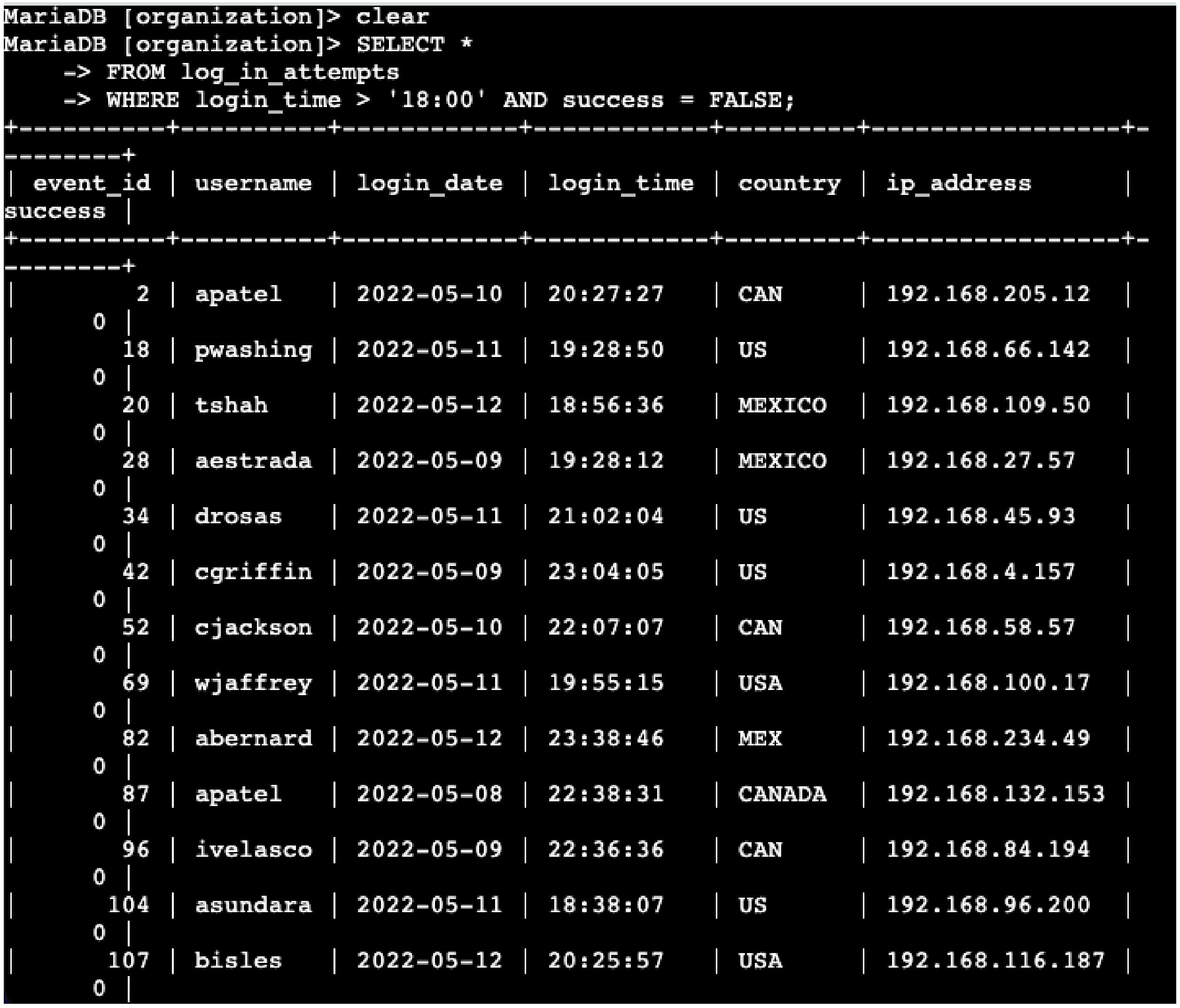
This document describes the approach to filtering and investigating failed login attempts that occurred after business hours, based on the timestamps of login attempts in the system.

Business Hours Definition:

- Business hours are defined from **9:00 AM to 6:00 PM** (18:00). Any login attempts outside this time frame are considered to have occurred after business hours.

SQL Query:

The following SQL query can be used to filter out failed login attempts that occurred either before 9:00 AM or after 6:00 PM.

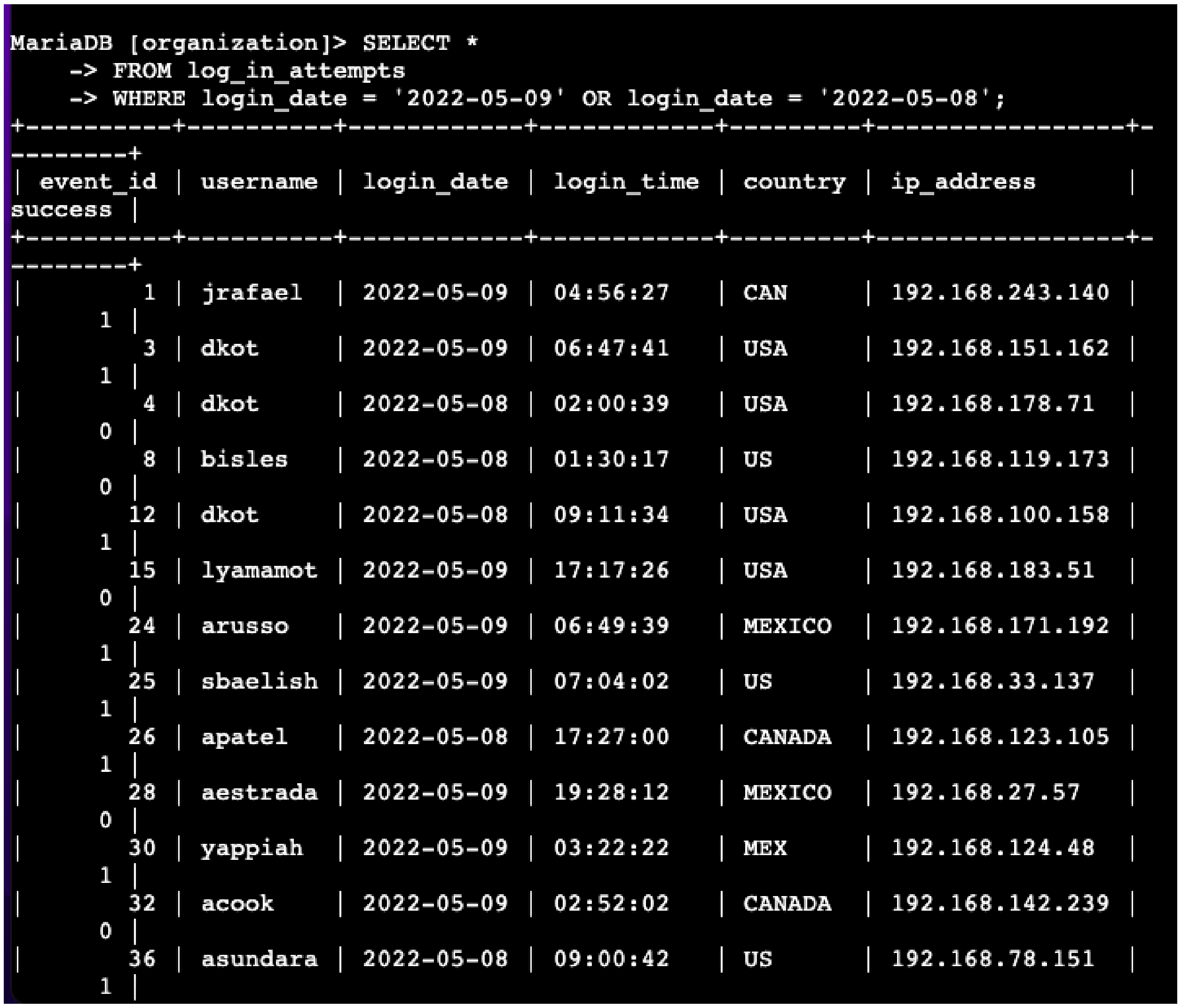


```
MariaDB [organization]> clear
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_time > '18:00' AND success = FALSE;
+----------+----------+------------+------------+---------+-----------------+-
--------+
| event_id | username | login_date | login_time | country | ip_address      |
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 |
```

SELECT * FROM login_attempts:
This gets all the data from the login_attempts table.
WHERE login_time > '18:00':
This filters the results to only include login attempts that happened after 18:00 (6:00 PM).
AND success = FALSE:
This filters the results further to only show failed login attempts (where the success field is FALSE).

Retrieve login attempts on specific dates

A suspicious event took place on **2022-05-09**, and we need to investigate any login attempts that occurred **on that day or the day before** (2022-05-08). The SQL query below filters for login attempts that happened on these two 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 |
|       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 |
|       36 | asundara | 2022-05-08 | 09:00:42   | US      | 192.168.78.151  |
      1 |
```

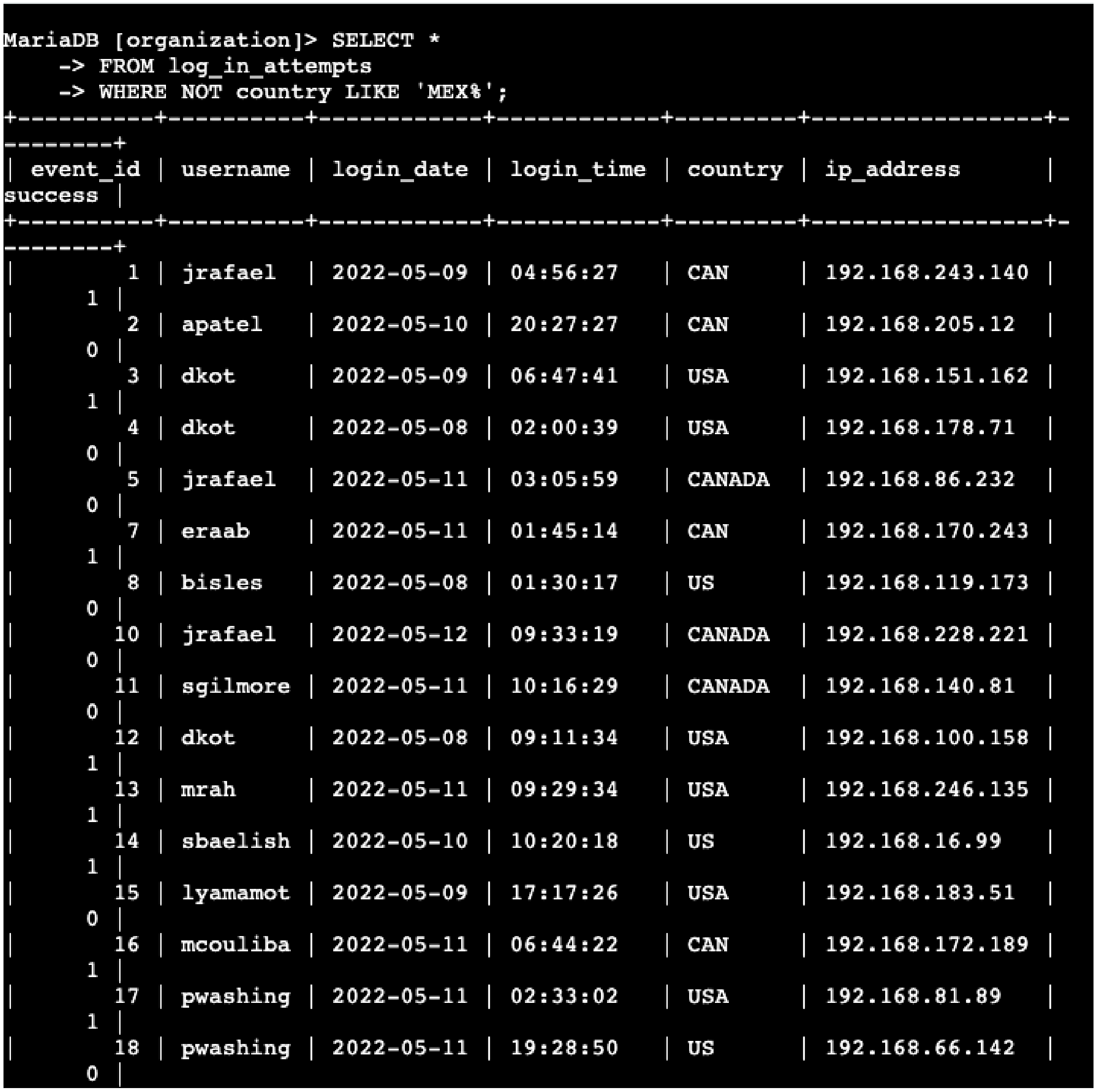
Data:
The query pulls all the data from the login_attempts table.
Filter for Specific Dates:
The WHERE clause makes sure we only see logins that happened on **2022-05-09** or **2022-05-08**.

- login_date = '2022-05-09': Shows logins from **2022-05-09**.
- login_date = '2022-05-08': Shows logins from **2022-05-08**.

Retrieve login attempts outside of Mexico

During my investigation of login attempts, I noticed that some of the login attempts were made from locations **outside of Mexico**, which may be a security concern. These login attempts need to be reviewed further.

To filter for login attempts that occurred outside of Mexico, I used the following SQL query.



```
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 |
|       14 | sbaelish | 2022-05-10 | 10:20:18   | US      | 192.168.16.99   |
      1 |
|       15 | lyamamot | 2022-05-09 | 17:17:26   | USA     | 192.168.183.51  |
      0 |
|       16 | mcouliba | 2022-05-11 | 06:44:22   | CAN     | 192.168.172.189 |
      1 |
|       17 | pwashing | 2022-05-11 | 02:33:02   | USA     | 192.168.81.89   |
      1 |
|       18 | pwashing | 2022-05-11 | 19:28:50   | US      | 192.168.66.142  |
      0 |
```

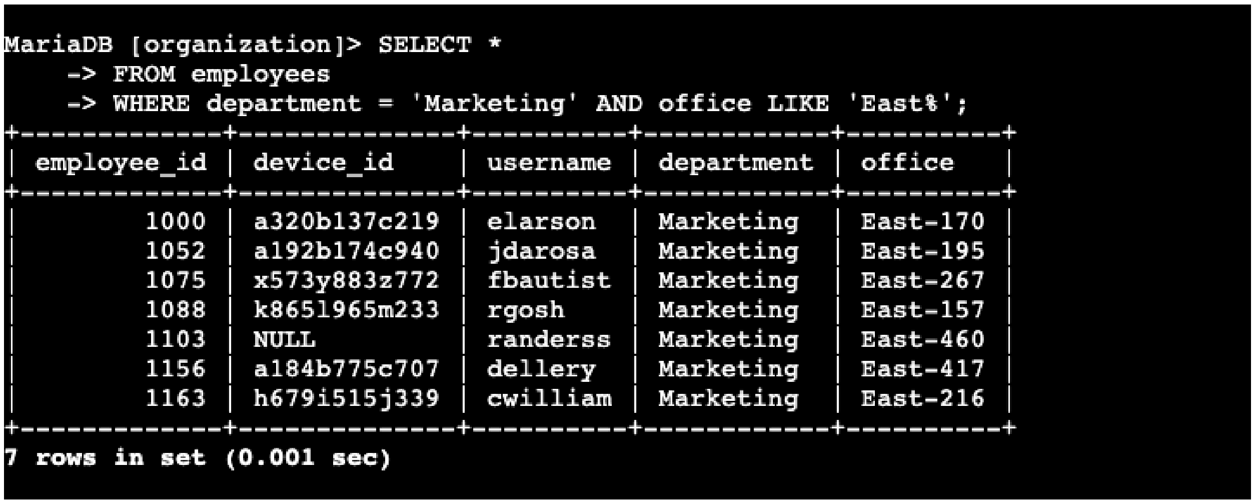
The first part of the screenshot shows my SQL query, and the second part shows the results. This query returns all login attempts that occurred in countries other than **Mexico**.

1. **Selecting All Data**:
   I started by selecting all data from the login_attempts table.
2. **Filtering with the WHERE Clause**:
   I used a WHERE clause to filter the results to exclude login attempts from **Mexico**.

3. **Using LIKE with MEX%**:
   To handle different ways Mexico is listed (like "MEX" or "MEXICO"), I used LIKE 'MEX%'. The % means "any characters after MEX", so this will match both "MEX" and "MEXICO".

Retrieve employees in Marketing

I need to gather information about which employee machines to update for certain employees in the **Marketing department** in the **East building**. The following SQL query helps filter out the relevant data.



```
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)
```

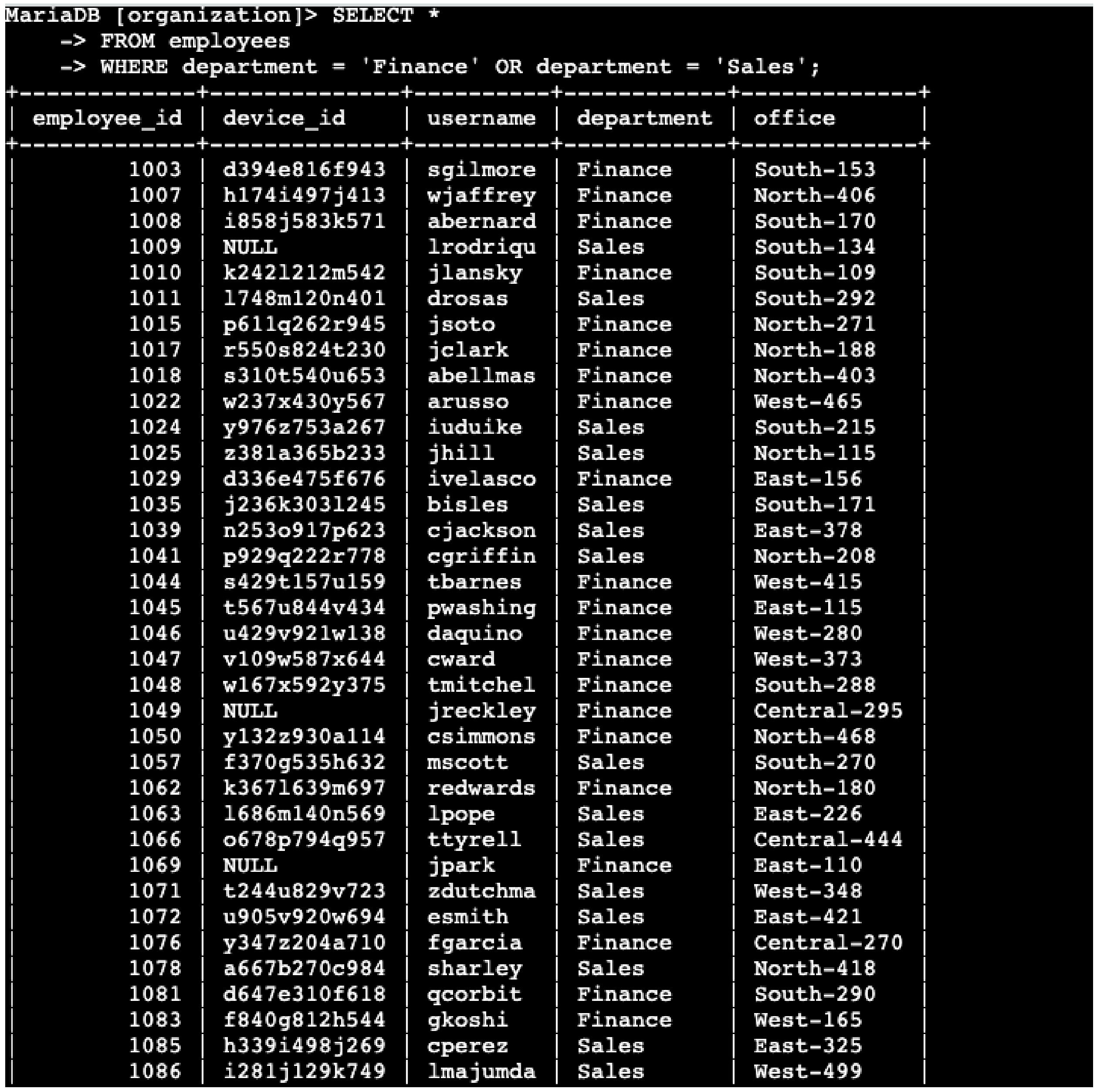
he first part of the screenshot shows the SQL query, and the second part shows the results. This query finds employees in the **Marketing department** who work in the **East building**.

How It Works:

1. **Data**:
   The query pulls all employee information from the employees table.
2. **Filter by Department and Location**:
   It looks for employees who:
   - Work in the **Marketing department**.
   - Have an office in the **East building** (using the LIKE 'East%' to match any office starting with "East").

Retrieve employees in Finance or Sales

The machines for employees in the **Finance** and **Sales** departments need to be updated with a different security update. To get this information, I used the following SQL query to find employees from these two departments.



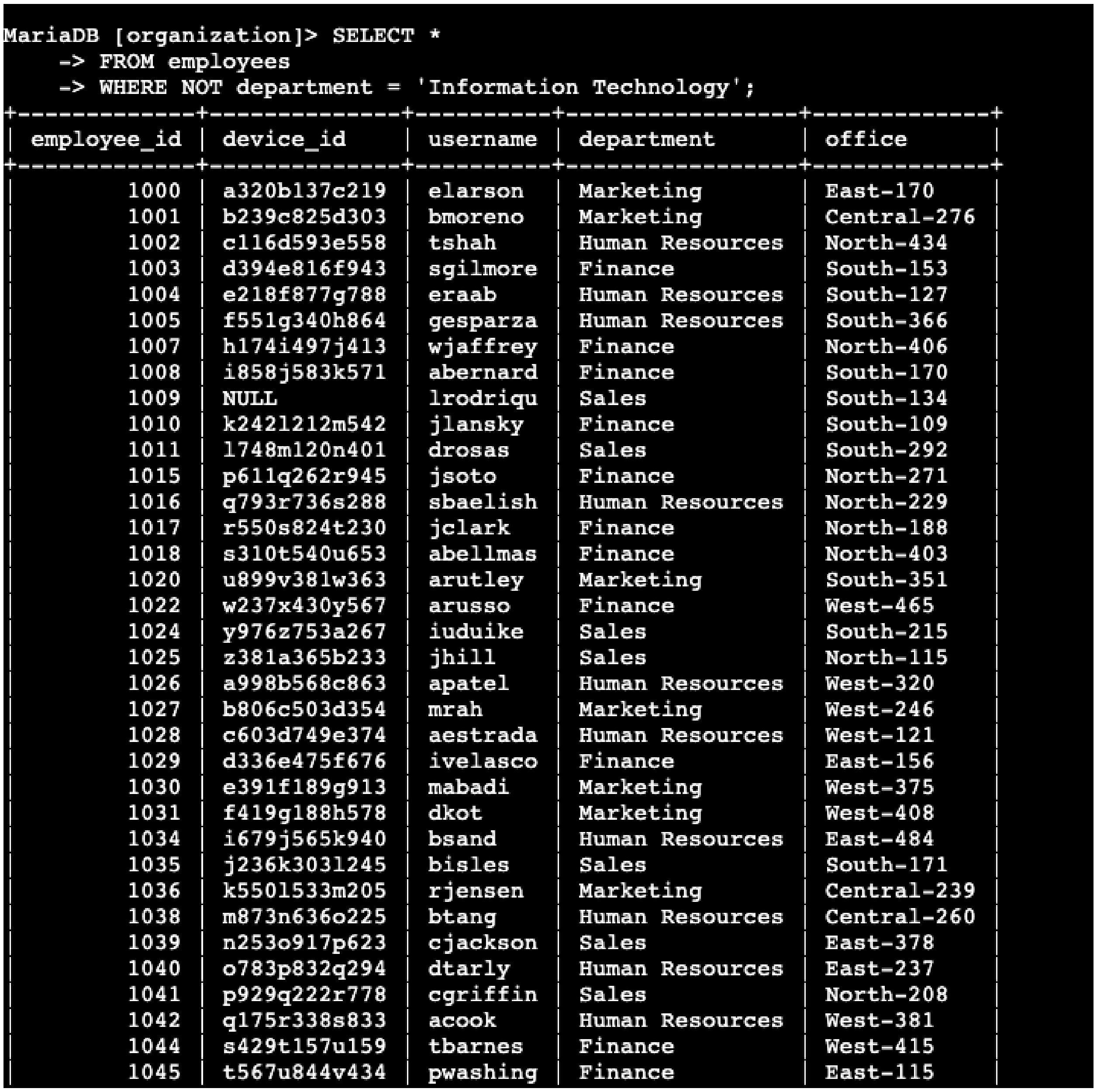
```
MariaDB [organization]> SELECT *
    -> FROM employees
    -> WHERE department = 'Finance' OR department = 'Sales';
+-------------+--------------+----------+------------+-------------+
| employee_id | device_id    | username | department | office      |
+-------------+--------------+----------+------------+-------------+
|        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    |
|        1024 | y976z753a267 | iuduike  | Sales      | South-215   |
|        1025 | z381a365b233 | jhill    | Sales      | North-115   |
|        1029 | d336e475f676 | ivelasco | Finance    | East-156    |
|        1035 | j236k3031245 | bisles   | Sales      | South-171   |
|        1039 | n253o917p623 | cjackson | Sales      | East-378    |
|        1041 | p929q222r778 | cgriffin | Sales      | North-208   |
|        1044 | s429t157u159 | tbarnes  | Finance    | West-415    |
|        1045 | t567u844v434 | pwashing | Finance    | East-115    |
|        1046 | u429v921w138 | daquino  | Finance    | West-280    |
|        1047 | v109w587x644 | cward    | Finance    | West-373    |
|        1048 | w167x592y375 | tmitchel | Finance    | South-288   |
|        1049 | NULL         | jreckley | Finance    | Central-295 |
|        1050 | y132z930a114 | csimmons | Finance    | North-468   |
|        1057 | f370g535h632 | mscott   | Sales      | South-270   |
|        1062 | k3671639m697 | redwards | Finance    | North-180   |
|        1063 | 1686m140n569 | lpope    | Sales      | East-226    |
|        1066 | o678p794q957 | ttyrell  | Sales      | Central-444 |
|        1069 | NULL         | jpark    | Finance    | East-110    |
|        1071 | t244u829v723 | zdutchma | Sales      | West-348    |
|        1072 | u905v920w694 | esmith   | Sales      | East-421    |
|        1076 | y347z204a710 | fgarcia  | Finance    | Central-270 |
|        1078 | a667b270c984 | sharley  | Sales      | North-418   |
|        1081 | d647e310f618 | qcorbit  | Finance    | South-290   |
|        1083 | f840g812h544 | gkoshi   | Finance    | West-165    |
|        1085 | h339i498j269 | cperez   | Sales      | East-325    |
|        1086 | i281j129k749 | lmajumda | Sales      | West-499    |
```

Data:
The query starts by selecting all the information from the employees table.
Filter for Finance and Sales Employees:
The query uses the WHERE clause with OR to find employees in either the **Finance** or **Sales** departments:

- department = 'Finance' finds employees in the **Finance** department.
- department = 'Sales' finds employees in the **Sales** department.

The OR makes sure we get employees from **either** department, not just both.

My teams needs to update the security for employees who are **not** in the **Information Technology (IT)** department. Here's the SQL query I used to find those employees.es 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 | 1748m120n401 | 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   |
|        1022 | w237x430y567 | arusso   | Finance         | West-465    |
|        1024 | y976z753a267 | iuduike  | Sales           | South-215   |
|        1025 | z381a365b233 | jhill    | Sales           | North-115   |
|        1026 | a998b568c863 | apatel   | Human Resources | West-320    |
|        1027 | b806c503d354 | mrah     | Marketing       | West-246    |
|        1028 | c603d749e374 | aestrada | Human Resources | West-121    |
|        1029 | d336e475f676 | ivelasco | Finance         | East-156    |
|        1030 | e391f189g913 | mabadi   | Marketing       | West-375    |
|        1031 | f419g188h578 | dkot     | Marketing       | West-408    |
|        1034 | i679j565k940 | bsand    | Human Resources | East-484    |
|        1035 | j236k3031245 | bisles   | Sales           | South-171   |
|        1036 | k5501533m205 | rjensen  | Marketing       | Central-239 |
|        1038 | m873n636o225 | btang    | Human Resources | Central-260 |
|        1039 | n253o917p623 | cjackson | Sales           | East-378    |
|        1040 | o783p832q294 | dtarly   | Human Resources | East-237    |
|        1041 | p929q222r778 | cgriffin | Sales           | North-208   |
|        1042 | q175r338s833 | acook    | Human Resources | West-381    |
|        1044 | s429t157u159 | tbarnes  | Finance         | West-415    |
|        1045 | t567u844v434 | pwashing | Finance         | East-115    |
```

The first part of the screenshot shows the SQL query, and the second part shows part of the results. This query finds all employees who are **not** in the **Information Technology (IT)** department.

How It Works:

1. **Get All Employee Data**:
   The query starts by selecting all information from the `employees` table.
2. **Filter Employees Not in IT**:
   The `WHERE` clause uses `=` to exclude employees in the **IT department**:
   - `department != 'Information Technology'` makes sure we only get employees who are not in IT.

SUMMARY

I used filters in my SQL queries to get specific information about login attempts and employee machines from two tables: `log_in_attempts` and `employees`. I applied **AND**, **OR**, and **NOT** to find data that met certain conditions, like employees in a specific department or building, or excluding those who didn't meet the criteria. I also used the **LIKE** operator with the `%` symbol to search for patterns, such as finding login attempts from offices starting with "East."