Apply filters to SQL queries

Project description

As a security analyst at a large organization, I investigated potential security concerns by filtering login activity and employee data using SQL queries. This project demonstrates my ability to retrieve specific information using SQL `AND`, `OR`, `NOT`, and `LIKE` operators. I analyzed login attempts by time, date, country, and success status, and filtered employee data based on department and office location. These queries support incident investigation and system update planning.

Retrieve after hours failed login attempts

```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE login_time > '18:00' AND success = 0;
```

This SQL query filters the `log_in_attempts` table to find all login attempts that were **unsuccessful** and made **after business hours (18:00)**.

- The condition `login_time > '18:00'` retrieves only those records where login attempts occurred after 6:00 PM.

- The condition `success = 0` filters for **failed login attempts** (`0` represents a Boolean FALSE in MySQL).
  By combining these filters with the `AND` operator, the query returns only the **after-hours failed login attempts**, helping identify suspicious access patterns outside of normal working hours.

```
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 (0.178 sec)
```

- **Query Result (After Hours Failed Logins)**

  This screenshot displays the **executed SQL query** along with the **retrieved results** from the `log_in_attempts` table. It confirms that the query accurately filtered **failed login attempts (success = 0)** that occurred **after 18:00 (6 PM)**. The output includes multiple rows of data, each representing a login attempt that matches the specified criteria — demonstrating successful application of SQL filtering using the `AND` operator.

Retrieve login attempts on specific dates

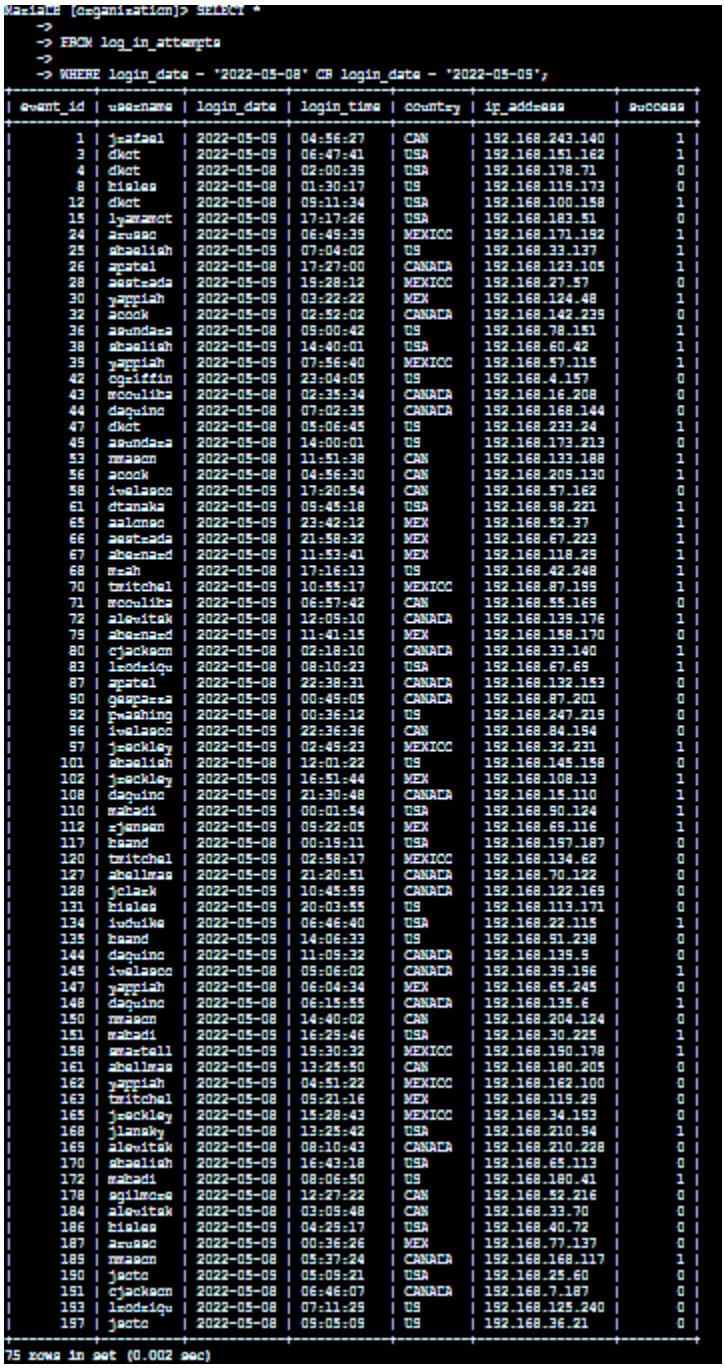
```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE login_date = '2022-05-08' OR login_date = '2022-05-09';
```

This SQL query is used to retrieve all login attempts that occurred on two specific dates: May 8, 2022, and May 9, 2022. These dates are important because a suspicious event was detected on May 9, and investigating the day before can help uncover any related activity.

- The `SELECT *` command retrieves **all columns** from the `log_in_attempts` table.

- The `WHERE` clause applies a **filter** to return only the rows where the `login_date` is **either** `'2022-05-08'` **OR** `'2022-05-09'`.

- The `OR` operator is used to include **multiple specific conditions** for the same column.



```
MariaDB [organization]> SELECT *
    -> 
    -> FROM log_in_attempts
    -> 
    -> WHERE login_date = '2022-05-08' OR login_date = '2022-05-09';
+----------+-----------+------------+------------+---------+----------------+---------+
| 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 |
|       38 | sbaelish  | 2022-05-09 | 14:40:01   | USA     | 192.168.60.42   |       1 |
|       39 | yappiah   | 2022-05-09 | 07:56:40   | MEXICO  | 192.168.57.115  |       1 |
|       42 | cgriffin  | 2022-05-09 | 23:04:05   | US      | 192.168.4.157   |       0 |
|       43 | mcouliba  | 2022-05-08 | 02:35:34   | CANADA  | 192.168.16.208  |       0 |
|       44 | daquino   | 2022-05-08 | 07:02:35   | CANADA  | 192.168.168.144 |       0 |
|       47 | dkot      | 2022-05-08 | 05:06:45   | US      | 192.168.233.24  |       1 |
|       45 | asundara  | 2022-05-08 | 14:00:01   | US      | 192.168.173.213 |       0 |
|       53 | nmason    | 2022-05-08 | 11:51:38   | CAN     | 192.168.133.188 |       1 |
|       56 | acook     | 2022-05-08 | 04:56:30   | CAN     | 192.168.205.130 |       1 |
|       58 | ivelasco  | 2022-05-09 | 17:20:54   | CAN     | 192.168.57.162  |       0 |
|       61 | dtanaka   | 2022-05-09 | 09:45:18   | USA     | 192.168.98.221  |       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 |
|       67 | ahernand  | 2022-05-08 | 11:53:41   | MEX     | 192.168.118.25  |       1 |
|       68 | mzah      | 2022-05-08 | 17:16:13   | US      | 192.168.42.248  |       1 |
|       70 | tmitchel  | 2022-05-09 | 10:55:17   | MEXICO  | 192.168.87.155  |       1 |
|       71 | mcoulibs  | 2022-05-09 | 06:57:42   | CAN     | 192.168.55.165  |       0 |
|       72 | alevitsk  | 2022-05-08 | 12:09:10   | CANADA  | 192.168.139.176 |       1 |
|       75 | ahernand  | 2022-05-09 | 11:41:15   | MEX     | 192.168.158.170 |       0 |
|       80 | cjackson  | 2022-05-08 | 02:18:10   | CANADA  | 192.168.33.140  |       1 |
|       83 | lrodrigu  | 2022-05-08 | 08:10:23   | USA     | 192.168.67.65   |       1 |
|       87 | apatel    | 2022-05-08 | 22:38:31   | CANADA  | 192.168.132.153 |       0 |
|       90 | garzaxx   | 2022-05-09 | 00:45:05   | CANADA  | 192.168.87.201  |       0 |
|       92 | fwashing  | 2022-05-08 | 00:36:12   | US      | 192.168.247.215 |       0 |
|       96 | ivelasco  | 2022-05-09 | 22:36:36   | CAN     | 192.168.84.154  |       0 |
|       97 | jpeckley  | 2022-05-09 | 02:45:23   | MEXICO  | 192.168.32.231  |       1 |
|      101 | sbaelish  | 2022-05-08 | 12:01:22   | US      | 192.168.145.158 |       0 |
|      102 | jpeckley  | 2022-05-09 | 16:51:44   | MEX     | 192.168.108.13  |       1 |
|      108 | daquino   | 2022-05-09 | 21:30:48   | CANADA  | 192.168.15.110  |       1 |
|      110 | mabadi    | 2022-05-09 | 00:01:54   | USA     | 192.168.90.124  |       1 |
|      112 | rjensen   | 2022-05-09 | 09:22:05   | MEX     | 192.168.69.116  |       1 |
|      117 | hsand     | 2022-05-08 | 00:15:11   | USA     | 192.168.197.187 |       0 |
|      120 | tmitchel  | 2022-05-09 | 02:58:17   | MEXICO  | 192.168.134.62  |       0 |
|      127 | abellmas  | 2022-05-09 | 21:20:51   | CANADA  | 192.168.70.122  |       0 |
|      128 | jclark    | 2022-05-09 | 10:45:55   | CANADA  | 192.168.122.165 |       0 |
|      131 | bisles    | 2022-05-09 | 20:03:55   | US      | 192.168.113.171 |       0 |
|      134 | iuduike   | 2022-05-09 | 06:46:40   | USA     | 192.168.22.115  |       1 |
|      135 | hsand     | 2022-05-09 | 14:06:33   | US      | 192.168.51.238  |       0 |
|      144 | daquino   | 2022-05-09 | 11:09:32   | CANADA  | 192.168.139.9   |       0 |
|      145 | ivelasco  | 2022-05-08 | 09:06:02   | CANADA  | 192.168.39.196  |       1 |
|      147 | yappiah   | 2022-05-08 | 06:04:34   | MEX     | 192.168.65.245  |       0 |
|      148 | daquino   | 2022-05-08 | 06:15:55   | CANADA  | 192.168.135.6   |       1 |
|      150 | nmason    | 2022-05-08 | 14:40:02   | CAN     | 192.168.204.124 |       0 |
|      151 | mabadi    | 2022-05-08 | 16:29:46   | USA     | 192.168.30.225  |       1 |
|      158 | smartell  | 2022-05-09 | 19:30:32   | MEXICO  | 192.168.190.178 |       1 |
|      161 | abellmas  | 2022-05-09 | 13:25:50   | CAN     | 192.168.180.205 |       0 |
|      162 | yappiah   | 2022-05-09 | 04:51:22   | MEXICO  | 192.168.162.100 |       0 |
|      163 | tmitchel  | 2022-05-08 | 09:21:16   | MEX     | 192.168.119.25  |       0 |
|      165 | jpeckley  | 2022-05-08 | 15:28:43   | MEXICO  | 192.168.34.193  |       0 |
|      168 | jlansky   | 2022-05-08 | 13:25:42   | USA     | 192.168.210.94  |       1 |
|      169 | alevitsk  | 2022-05-08 | 08:10:43   | CANADA  | 192.168.210.228 |       0 |
|      170 | sbaelish  | 2022-05-09 | 16:43:18   | USA     | 192.168.65.113  |       0 |
|      172 | mabadi    | 2022-05-08 | 08:06:50   | US      | 192.168.180.41  |       1 |
|      178 | agilmore  | 2022-05-08 | 12:27:22   | CAN     | 192.168.52.216  |       0 |
|      184 | alevitsk  | 2022-05-08 | 03:09:48   | CAN     | 192.168.33.70   |       0 |
|      186 | bisles    | 2022-05-09 | 04:29:17   | USA     | 192.168.40.72   |       0 |
|      187 | arusso    | 2022-05-09 | 00:36:26   | MEX     | 192.168.77.137  |       0 |
|      189 | nmason    | 2022-05-08 | 05:37:24   | CANADA  | 192.168.168.117 |       1 |
|      190 | jsoto     | 2022-05-09 | 05:05:21   | USA     | 192.168.25.60   |       0 |
|      191 | cjackson  | 2022-05-08 | 06:46:07   | CANADA  | 192.168.7.187   |       0 |
|      193 | lrodrigu  | 2022-05-08 | 07:11:29   | US      | 192.168.125.240 |       0 |
|      197 | jsoto     | 2022-05-08 | 09:05:09   | US      | 192.168.36.21   |       0 |
+----------+-----------+------------+------------+---------+----------------+---------+
75 rows in set (0.002 sec)
```

- **Output of Query for Specific Dates**

  This screenshot displays the output of the SQL query that filtered login attempts from the `log_in_attempts` table based on two specific dates: **2022-05-08** and **2022-05-09**. The query successfully returned all records matching the specified `login_date` values, including relevant fields such as:

- `event_id`
- `username`
- `login_date` and `login_time`
- `country`
- `ip_address`
- `success` (where 0 indicates failed login)

This data will help the security team investigate activity around the date of a suspicious event and understand user behavior leading up to and during the incident.

Retrieve login attempts outside of Mexico

```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE NOT country LIKE 'MEX%';
```

In this step, the objective was to retrieve all login attempts **that did not originate in Mexico**. The `country` column in the database includes both `"MEX"` and `"MEXICO"` for Mexican-origin attempts. To filter out these, I used the `NOT` operator combined with `LIKE 'MEX%'`.

- `LIKE 'MEX%'` matches any country value starting with "MEX", covering both `"MEX"` and `"MEXICO"`.
- The `NOT` operator excludes all these matching entries from the result.

This ensures that **only login attempts from countries other than Mexico** are returned. It helps narrow down the investigation to external sources of suspicious activity.

```
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 |
|       19 | jhill    | 2022-05-12 | 13:09:04   | US      | 192.168.142.245 |       1 |
|       21 | iuduike  | 2022-05-11 | 17:50:00   | US      | 192.168.131.147 |       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 |
|       29 | bisles   | 2022-05-11 | 01:21:22   | US      | 192.168.85.186  |       0 |
|       31 | acook    | 2022-05-12 | 17:36:45   | CANADA  | 192.168.58.232  |       0 |
|       32 | acook    | 2022-05-09 | 02:52:02   | CANADA  | 192.168.142.239 |       0 |
|       33 | zbernal  | 2022-05-11 | 02:52:10   | US      | 192.168.72.59   |       1 |
|       34 | drosas   | 2022-05-11 | 21:02:04   | US      | 192.168.45.93   |       0 |
|       36 | asundara | 2022-05-08 | 09:00:42   | US      | 192.168.78.151  |       1 |
|       37 | eraab    | 2022-05-10 | 06:03:41   | CANADA  | 192.168.152.148 |       0 |
|       38 | sbaelish | 2022-05-09 | 14:40:01   | USA     | 192.168.60.42   |       1 |
|       41 | apatel   | 2022-05-10 | 17:39:42   | CANADA  | 192.168.46.207  |       0 |
|       42 | cgriffin | 2022-05-09 | 23:04:05   | US      | 192.168.4.157   |       0 |
|       43 | mcouliba | 2022-05-08 | 02:35:34   | CANADA  | 192.168.16.208  |       0 |
|       44 | daquino  | 2022-05-08 | 07:02:35   | CANADA  | 192.168.168.144 |       0 |
|       45 | dtanaka  | 2022-05-11 | 10:28:54   | US      | 192.168.223.157 |       1 |
|       46 | eraab    | 2022-05-11 | 11:29:27   | CAN     | 192.168.24.12   |       0 |
|       47 | dkot     | 2022-05-08 | 05:06:45   | US      | 192.168.233.24  |       1 |
|       48 | asundara | 2022-05-11 | 03:18:45   | USA     | 192.168.72.10   |       1 |
|       49 | asundara | 2022-05-08 | 14:00:01   | US      | 192.168.173.213 |       0 |
|       50 | jclark   | 2022-05-10 | 10:48:02   | CANADA  | 192.168.174.117 |       0 |
```

The output displayed is the result of a SQL query that filtered all login attempts which **did not originate from Mexico**. To achieve this, the `NOT` and `LIKE` operators were used with the pattern `'MEX%'`, which successfully excluded both `"MEX"` and `"MEXICO"` entries from the results.

This query was essential to identify potentially suspicious or unauthorized login activity from outside the expected region. The `country` column values in the result confirm the presence of logins from the **US**, **Canada**, and other countries, but none from Mexico.

- Partial results of login attempts that did not originate in Mexico, retrieved using the NOT + LIKE operator. Full results returned 144 records.

Retrieve employees in Marketing

```
MariaDB [organization]> SELECT *
    ->
    -> FROM employees
    ->
    -> WHERE department = 'Marketing'
    ->
    ->   AND office LIKE 'East-%';
```

This SQL query retrieves all employee records from the `employees` table where the `department` is **Marketing** and the `office` is located in the **East building**.

- The condition `department = 'Marketing'` filters the data to include only employees in the **Marketing department**.

- The condition `office LIKE 'East-%'` uses the `LIKE` operator with a wildcard (`%`) to include all office values that **start with "East-"**, such as `East-170` or `East-320`.

- The `AND` operator ensures both conditions must be true — meaning the result will only include **Marketing employees located in East building offices**.

This helps the team target specific employee machines for security updates based on **both department and location**.

```
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 (0.002 sec)
```

The output displays **7 employees** from the `employees` table who belong to the **Marketing department** and are located in offices that begin with **"East-"**, such as East-170, East-195, and East-460.
This confirms that the query successfully applied both conditions (`department = 'Marketing'` AND `office LIKE 'East-%'`) and returned only the relevant employee records needed for the update task.

This result will help the team perform targeted security updates on machines used by these Marketing employees in the East building.

Retrieve employees in Finance or Sales

To support a new security update, I needed to identify all employees working in the **Finance** or **Sales** departments. These departments are listed in the department column of the employees table.

I used the OR operator in the SQL query to filter for both conditions. It was necessary to fully write out each condition and reference the department column each time to ensure the query worked as intended.

```
MariaDB [organization]> SELECT *
    ->
    -> FROM employees
    ->
    -> WHERE department = 'Finance'
    ->
    ->    OR department = 'Sales';
```

- The query retrieves **all columns** from the `employees` table.
- It filters results where the `department` is either **'Finance'** or **'Sales'** using the `OR` operator.
- This helps isolate relevant employees whose systems require targeted updates.

```
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   |
|        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 | l686m140n569 | 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 |
```

- The query successfully returned a list of employees from the `employees` table who work in either the **Finance** or **Sales** departments. All columns for the matched records were displayed. This output will help the team identify and apply the necessary system updates for users in these departments.
- Each record confirms the correct filtering using the `OR` condition across the `department` column. The result includes users across multiple offices, reflecting a complete dataset for both departments.

Retrieve all employees not in IT

```
MariaDB [organization]> SELECT *
    ->
    -> FROM employees
    ->
    -> WHERE department != 'Information Technology';
```

In this query, I retrieved all employee records except those who belong to the **Information Technology** department. I used the `!=` operator (which acts as the **NOT** condition) to filter out entries where the `department` column value is `'Information Technology'`.

This ensures that the result set includes only those employees **who still require the system update**, as the IT department's machines have already been updated.

Purpose:

To assist the team in identifying which employee machines still need updating by excluding those already handled (in the IT department).

```
MariaDB [organization]> SELECT *
    ->
    -> FROM employees
    ->
    -> WHERE 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   |
|        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    |
```

This SQL query successfully retrieved all employees **not** in the **Information Technology** department using the `!=` operator on the `department` column. The output includes fields like `employee_id`, `device_id`, `username`, `department`, and `office`.

Employees shown belong to various departments such as **Marketing**, **Finance**, **Sales**, and **Human Resources**, with office locations distributed across **East**, **West**, **North**, **South**, and **Central** regions.

Total rows returned: 161
This confirms that the filter was correctly applied to exclude IT department employees from the update process.

Summary

In this project, I ran a series of SQL queries to support a security investigation and system update task. I retrieved failed login attempts outside office hours, filtered activity based on specific dates, excluded login attempts from Mexico using pattern matching with `NOT LIKE`,

and identified employees based on department and location criteria. These queries demonstrate my ability to use complex SQL filters for security use cases, including time-based analysis, pattern matching, and multi-condition filtering. The outputs were validated with screenshots and documented for professional reporting.