Apply filters to SQL queries

Project description

My organization is committed to enhancing its system security. As part of this effort, I was responsible for investigating potential security issues and managing employee data to ensure the system's safety. The following examples demonstrate how I used SQL queries in a Linux shell, applying filters with logical operators to accomplish these tasks effectively.

First, I addressed failed login attempts that occurred outside regular business hours. By using the `AND` operator, I crafted a query to retrieve records where the login status was marked as "failed" and the time fell outside the defined business hours. This enabled my team to identify potential unauthorized access attempts for further investigation.

Next, I analysed login attempts from specific dates. Using the `OR` operator, I constructed a query to filter logins that occurred on multiple dates of interest, ensuring that all relevant activity within the specified timeframe was captured for review.

To narrow my analysis further, I applied the `NOT` operator to exclude login attempts originating from Mexico. This allowed me to focus on login data from other countries, which was crucial for identifying suspicious activity across diverse regions.

I then moved on to department-specific tasks. To retrieve data on employees in the Marketing department, I used the `AND` operator in a query that filtered results based on department and associated machines. Similarly, I employed the `OR` operator to identify employees in either the Finance or Sales departments, ensuring that all personnel from both areas were included in the results.

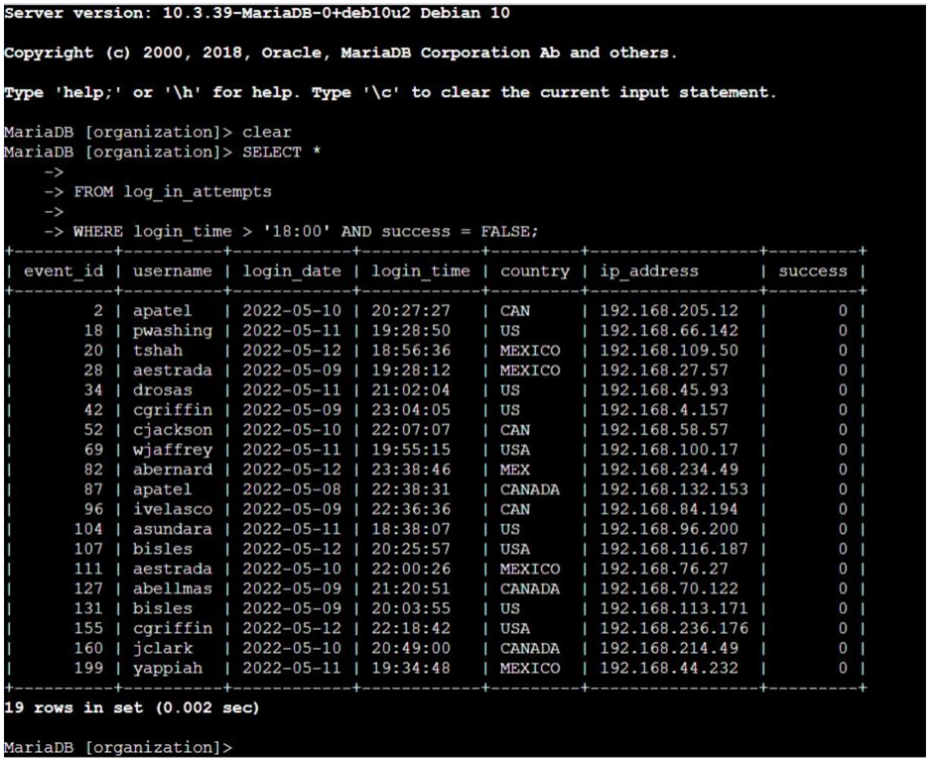
Finally, I used the `NOT` operator to extract details of employees who were not part of the Information Technology department. This query was essential for updating machines and managing devices associated with non-IT teams.

By leveraging complex SQL queries with logical operators in MySQL, I efficiently analysed data to address security concerns and support department-specific needs. These actions demonstrated my ability to apply advanced filtering techniques to real-world security and data management challenges.

Retrieve after hours failed login attempts

There was a potential security incident involving failed login attempts that occurred after business hours (after 18:00). To investigate this, I created a SQL query to filter for failed login attempts during that time.

The first part of the screenshot displays my query, and the second part shows a portion of the output. This query focuses on retrieving failed login attempts that occurred after 18:00 from the `log_in_attempts` table.



```
Server version: 10.3.39-MariaDB-0+deb10u2 Debian 10

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

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 |
|      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 (0.002 sec)

MariaDB [organization]>
```

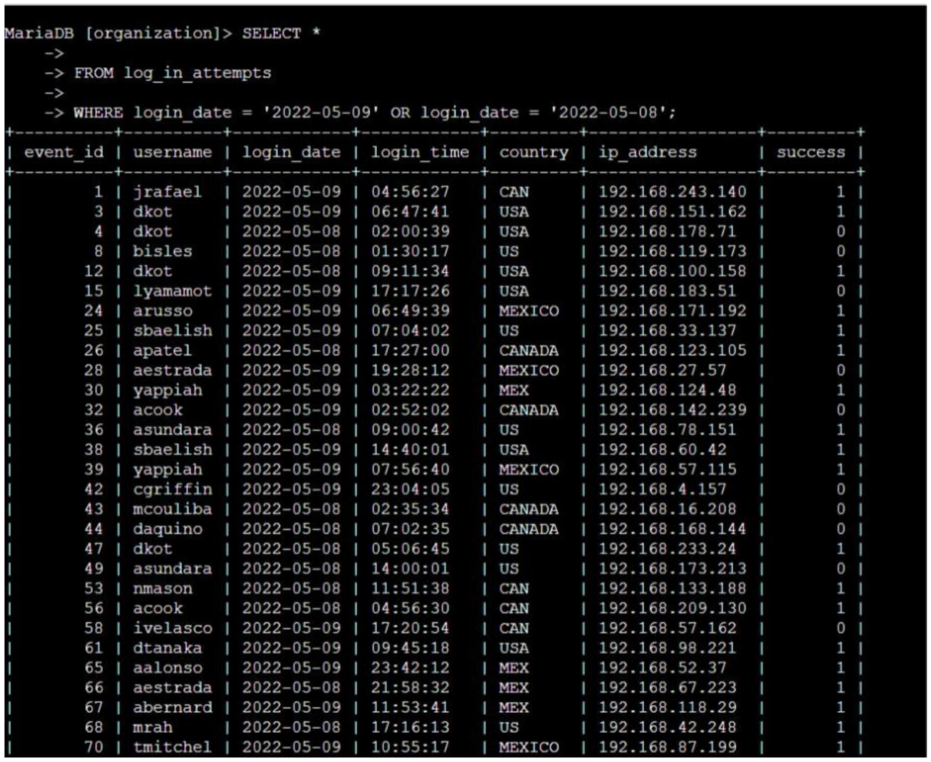
I began by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause with an `AND` operator to refine the results. The first condition, `login_time > '18:00'`, filters for login attempts that occurred after business hours. The second condition, `success = FALSE`, ensures that only failed login attempts are included in the output.

This query provided crucial details about suspicious login activities occurring outside regular working hours, enabling further investigation into potential unauthorized access attempts.

Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. To assist with the investigation, I needed to retrieve all login attempts that occurred on either 2022-05-09 or the previous day, 2022-05-08. The following code demonstrates how I created a SQL query to filter for login attempts on these specific dates.

The first part of the screenshot displays my query, and the second part shows a portion of the output. In this query, I used the `OR` operator within the `WHERE` clause to filter for login attempts based on the `login_date` column. The first condition, `login_date = '2022-05-09'`, identifies login attempts on May 9, 2022. The second condition, `login_date = '2022-05-08'`, retrieves login attempts from the day before.



```
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 |
|       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 |
|       49 | 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     | 192.168.209.130 |       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 | abernard | 2022-05-09 | 11:53:41   | MEX     | 192.168.118.29  |       1 |
|       68 | mrah     | 2022-05-08 | 17:16:13   | US      | 192.168.42.248  |       1 |
|       70 | tmitchel | 2022-05-09 | 10:55:17   | MEXICO  | 192.168.87.199  |       1 |
```

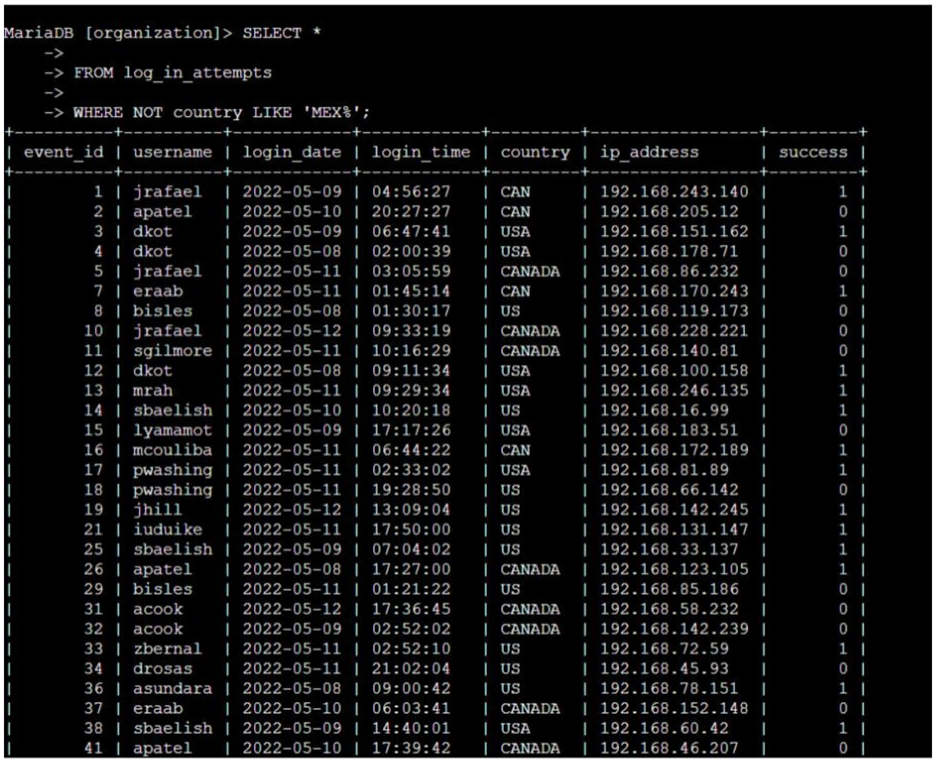
This query efficiently returned all relevant records for further analysis of activity on these specific dates.

Retrieve login attempts outside of Mexico

The following code demonstrates how I created a SQL query to filter for login attempts that occurred outside of Mexico.

The first part of the screenshot displays my query, and the second part shows a portion of the output. In this query, I used the `NOT` operator along with `LIKE` to exclude entries in the `country` column that begin with "MEX." The pattern `MEX%` accounts for both "MEX" and "MEXICO," as the percentage sign (`%`) represents any number of characters when used with the `LIKE` operator.

This query returned all login attempts from countries other than Mexico, allowing me to concentrate on international access and identify potential security issues.



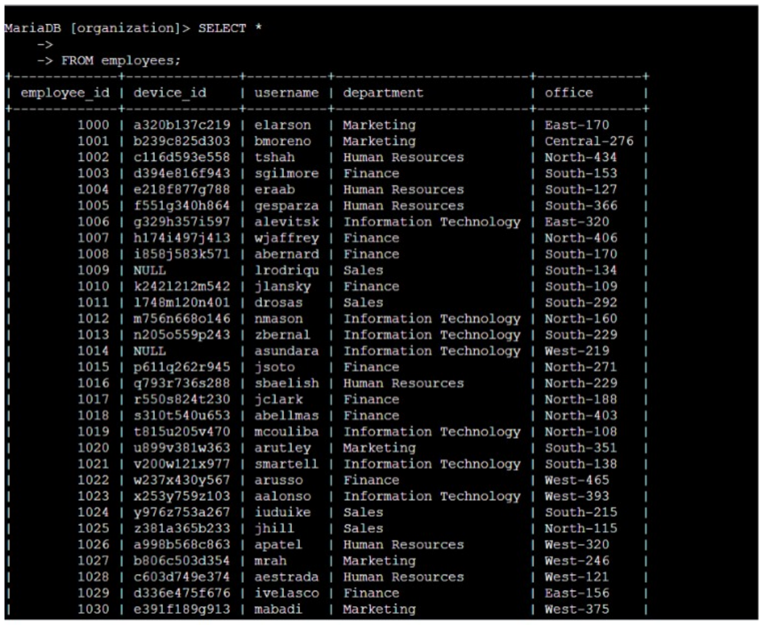
```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> WHERE NOT country LIKE 'MEX%';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 |
```

Retrieve employees in Marketing

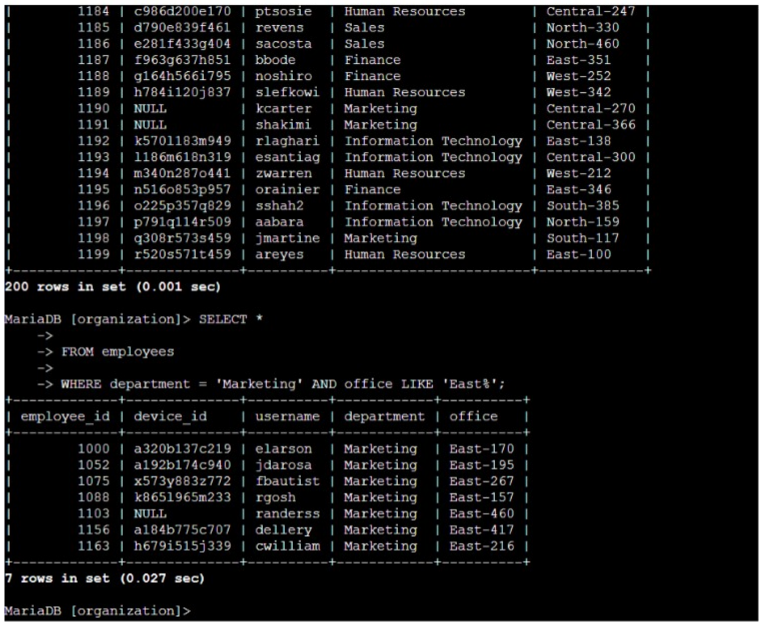
My team needs to update the computers for employees in the Marketing department located in the East building. To complete this task, I needed to retrieve information on which employees and machines to update.

The following code demonstrates how I created a SQL query to filter for employees in the Marketing department working in the East building.

The first part of the screenshot displays my query, and the second part shows a portion of the output. This query filters for all employees in the Marketing department located in the East building. First, I selected all data from the `employees` table. Then, I used a `WHERE` clause with `AND` to specify both conditions.



```
MariaDB [organization]> SELECT *
    ->
    -> FROM employees;
+-------------+--------------+----------+------------------------+-------------+
| 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   |
|        1006 | g329h357i597 | alevitsk | Information Technology | East-320    |
|        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   |
|        1012 | m756n668o146 | nmason   | Information Technology | North-160   |
|        1013 | n205o559p243 | zbernal  | Information Technology | South-229   |
|        1014 | NULL         | asundara | Information Technology | West-219    |
|        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   |
|        1019 | t815u205v470 | mcouliba | Information Technology | North-108   |
|        1020 | u899v381w363 | arutley  | Marketing              | South-351   |
|        1021 | v200w121x977 | smartell | Information Technology | South-138   |
|        1022 | w237x430y567 | arusso   | Finance                | West-465    |
|        1023 | x253y759z103 | aalonso  | Information Technology | West-393    |
|        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    |
```



```
|        1184 | c986d200e170 | ptsosie  | Human Resources        | Central-247 |
|        1185 | d790e839f461 | revens   | Sales                  | North-330   |
|        1186 | e281f433g404 | sacosta  | Sales                  | North-460   |
|        1187 | f963g637h851 | bbode    | Finance                | East-351    |
|        1188 | g164h566i795 | noshiro  | Finance                | West-252    |
|        1189 | h784i120j837 | slefkowi | Human Resources        | West-342    |
|        1190 | NULL         | kcarter  | Marketing              | Central-270 |
|        1191 | NULL         | shakimi  | Marketing              | Central-366 |
|        1192 | k5701183m949 | rlaghari | Information Technology | East-138    |
|        1193 | 1186m618n319 | esantiag | Information Technology | Central-300 |
|        1194 | m340n287o441 | zwarren  | Human Resources        | West-212    |
|        1195 | n516o853p957 | orainier | Finance                | East-346    |
|        1196 | o225p357q829 | sshah2   | Information Technology | South-385   |
|        1197 | p791q114r509 | aabara   | Information Technology | North-159   |
|        1198 | q308r573s459 | jmartine | Marketing              | South-117   |
|        1199 | r520s571t459 | areyes   | Human Resources        | East-100    |
+-------------+--------------+----------+------------------------+-------------+
200 rows in set (0.001 sec)

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 (0.027 sec)

MariaDB [organization]>
```

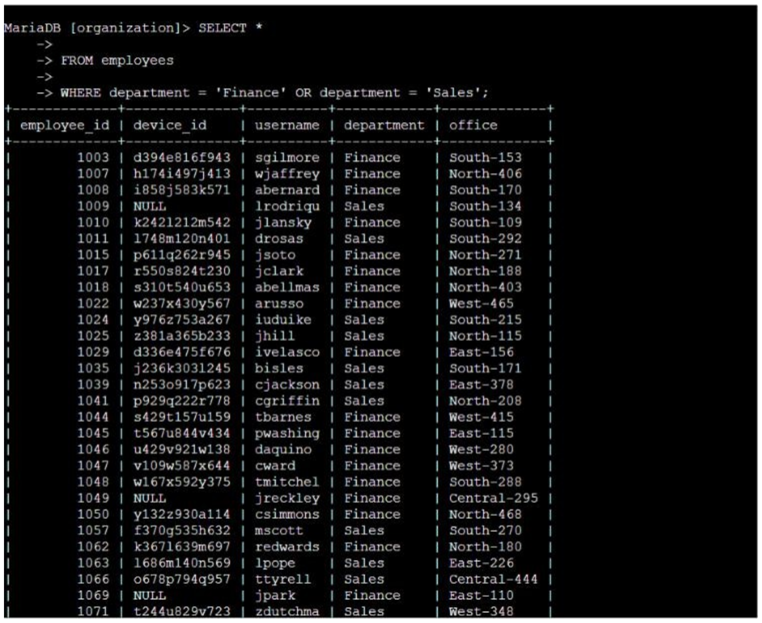
The first condition is `department = 'Marketing'`, which filters for employees in the Marketing department. The second condition is `office LIKE 'East%'`, which filters for employees working in any office within the East building (e.g., East-170, East-320). The `LIKE` operator with the pattern `East%` accounts for the building name followed by specific office numbers.

This query allowed me to efficiently identify the employees and machines needing updates in the specified department and location.

Retrieve employees in Finance or Sales

The machines for employees in the Finance and Sales departments need to be updated due to a different security update requirement. To gather relevant employee information for these updates, I needed to filter employees from these two departments.

The following code demonstrates how I created a SQL query to filter for employees from the Finance and Sales 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    |
```

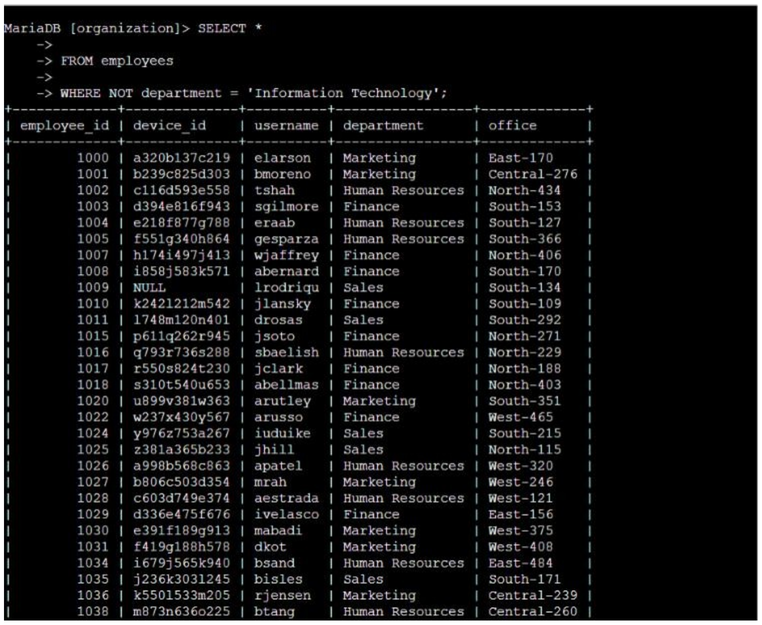
The first part of the screenshot displays my query, and the second part shows a portion of the output. This query returns all employees in the Finance and Sales departments. First, I started by selecting all data from the `employees` table. Then, I used a `WHERE` clause with `OR` to filter for employees who are in either the Finance or Sales department. The `OR` operator ensures that the query returns employees from both departments. The first condition is `department = 'Finance'`, which filters for employees in the Finance department. The second condition is `department = 'Sales'`, which filters for employees in the Sales department.

This query enabled me to efficiently gather the data necessary to update employee machines in both departments.

Retrieve all employees not in IT

My team needed to make one final security update for employees who are not in the Information Technology department. To prepare for this, I retrieved information about employees from all other departments.

The following demonstrates how I created a SQL query to filter for these 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   |
|        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 |
```

The first part of the screenshot is my query, and the second part is a portion of the output. This query returns all employees who are not in the Information Technology department. First, I started by selecting all data from the employees table. Then, I used a WHERE clause with the NOT operator to exclude any records where the department was listed as "Information Technology." This ensured that my query only returned employees from other departments, allowing my team to focus on updating their systems efficiently.

Summary

I applied filters to SQL queries to retrieve and analyse specific information from the **log_in_attempts** and **employees** tables. Using the AND, OR, and NOT operators, I created queries tailored to each task's requirements. I also used the LIKE operator with the percentage sign (%) wildcard to match patterns efficiently.

These queries allowed me to identify after-hours failed login attempts, retrieve login activity on specific dates, exclude logins originating from Mexico, and gather employee information across various departments. Through this project, I demonstrated my ability to leverage SQL queries to solve real-world cybersecurity challenges and analyse data effectively in a professional context.