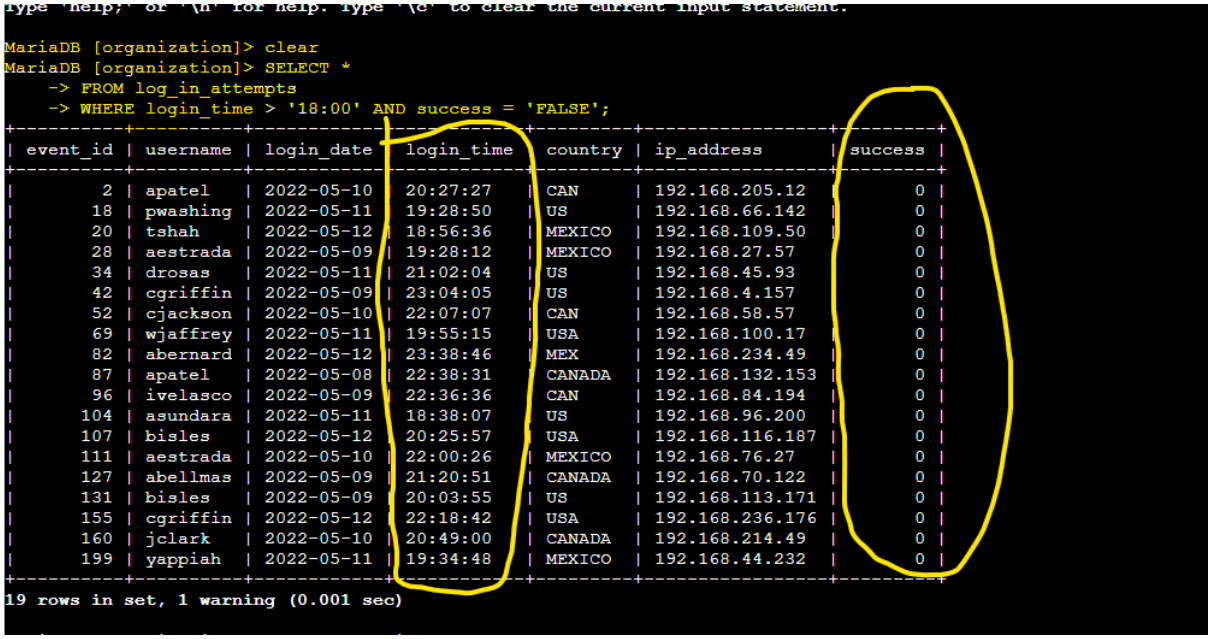
Apply filters to SQL queries

Project description

I'm a security professional at a large organization. Part of my job is to investigate security issues to help keep the system secure. I recently discovered some potential security issues that involve login attempts and employee machines. I will go through the database using SQL to solve and fix these issues.

Retrieve after hours failed login attempts



```
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, 1 warning (0.001 sec)
```

FROM log_in_attempts: This part of the query specifies the table from which the data will be retrieved. In this case, the table name is "log_in_attempts."

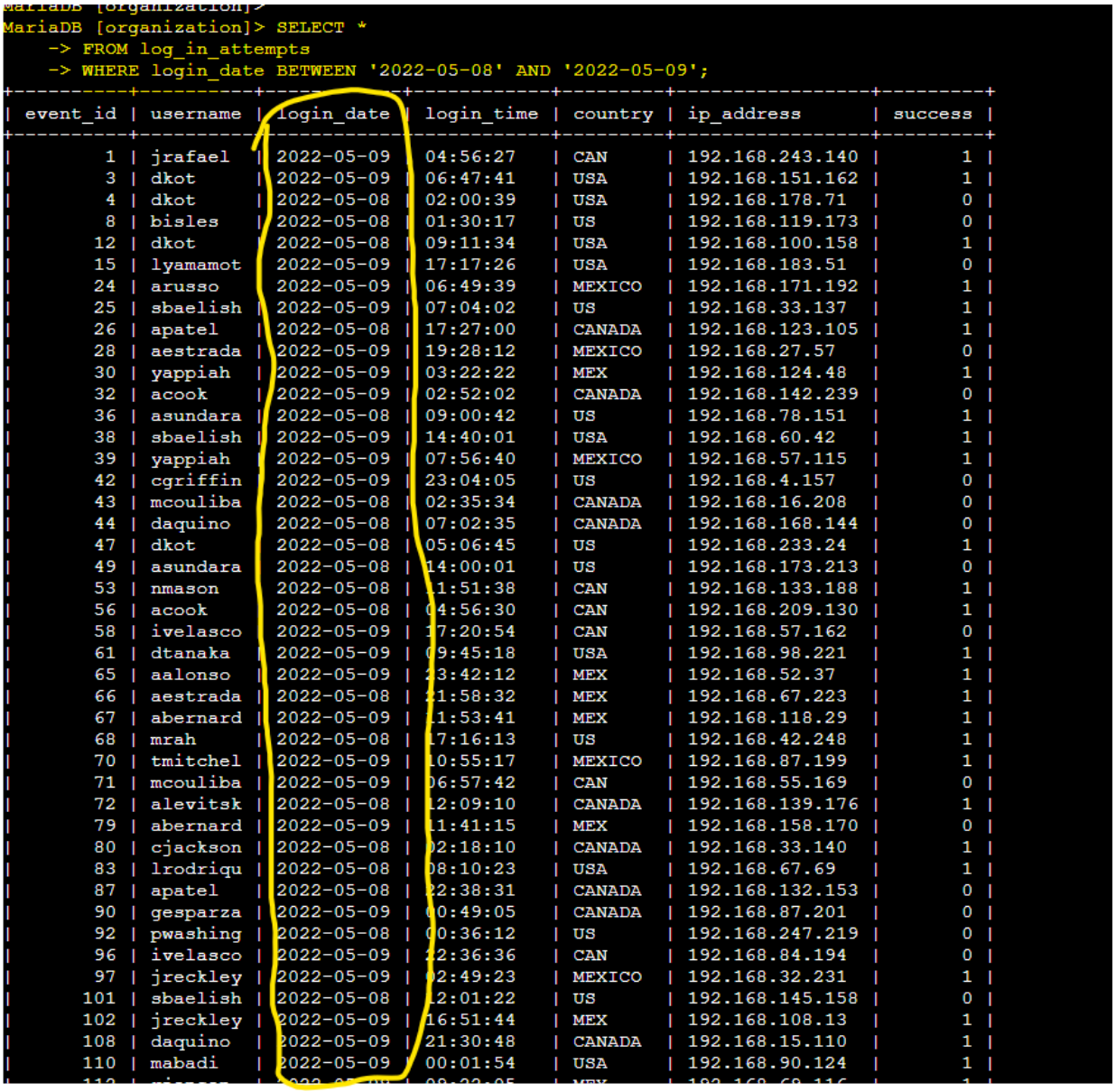
WHERE login_time > '18:00' AND success = 'FALSE': This part of the query includes the conditions for filtering the data. It tells the database to only retrieve rows where two conditions are met:

login_time > '18:00': This condition checks the "login_time" column and selects only those rows where the login time is greater than 6:00 PM (18:00 in 24-hour format).

success = 'FALSE': This condition checks the "success" column and selects only those rows where the value in the "success" column is 'FALSE'. The assumption here is that the "success" column stores a boolean value indicating whether the login attempt was successful or not.

In summary, the query is retrieving rows from the "log_in_attempts" table where the login time is after 6:00 PM and the login attempt was unsuccessful (success = 'FALSE'). The result will include all login attempts that meet these conditions, which might help analyze failed login patterns or investigate issues related to login attempts after a certain time of day.

Retrieve login attempts on specific dates



```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_date BETWEEN '2022-05-08' AND '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 |
|       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 |
|       71 | mcouliba | 2022-05-09 | 06:57:42   | CAN     | 192.168.55.169  |       0 |
|       72 | alevitsk | 2022-05-08 | 12:09:10   | CANADA  | 192.168.139.176 |       1 |
|       79 | abernard | 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 | lrodriqu | 2022-05-08 | 08:10:23   | USA     | 192.168.67.69   |       1 |
|       87 | apatel   | 2022-05-08 | 22:38:31   | CANADA  | 192.168.132.153 |       0 |
|       90 | gesparza | 2022-05-09 | 00:49:05   | CANADA  | 192.168.87.201  |       0 |
|       92 | pwashing | 2022-05-08 | 00:36:12   | US      | 192.168.247.219 |       0 |
|       96 | ivelasco | 2022-05-09 | 22:36:36   | CAN     | 192.168.84.194  |       0 |
|       97 | jreckley | 2022-05-09 | 02:49:23   | MEXICO  | 192.168.32.231  |       1 |
|      101 | sbaelish | 2022-05-08 | 12:01:22   | US      | 192.168.145.158 |       0 |
|      102 | jreckley | 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 |
```

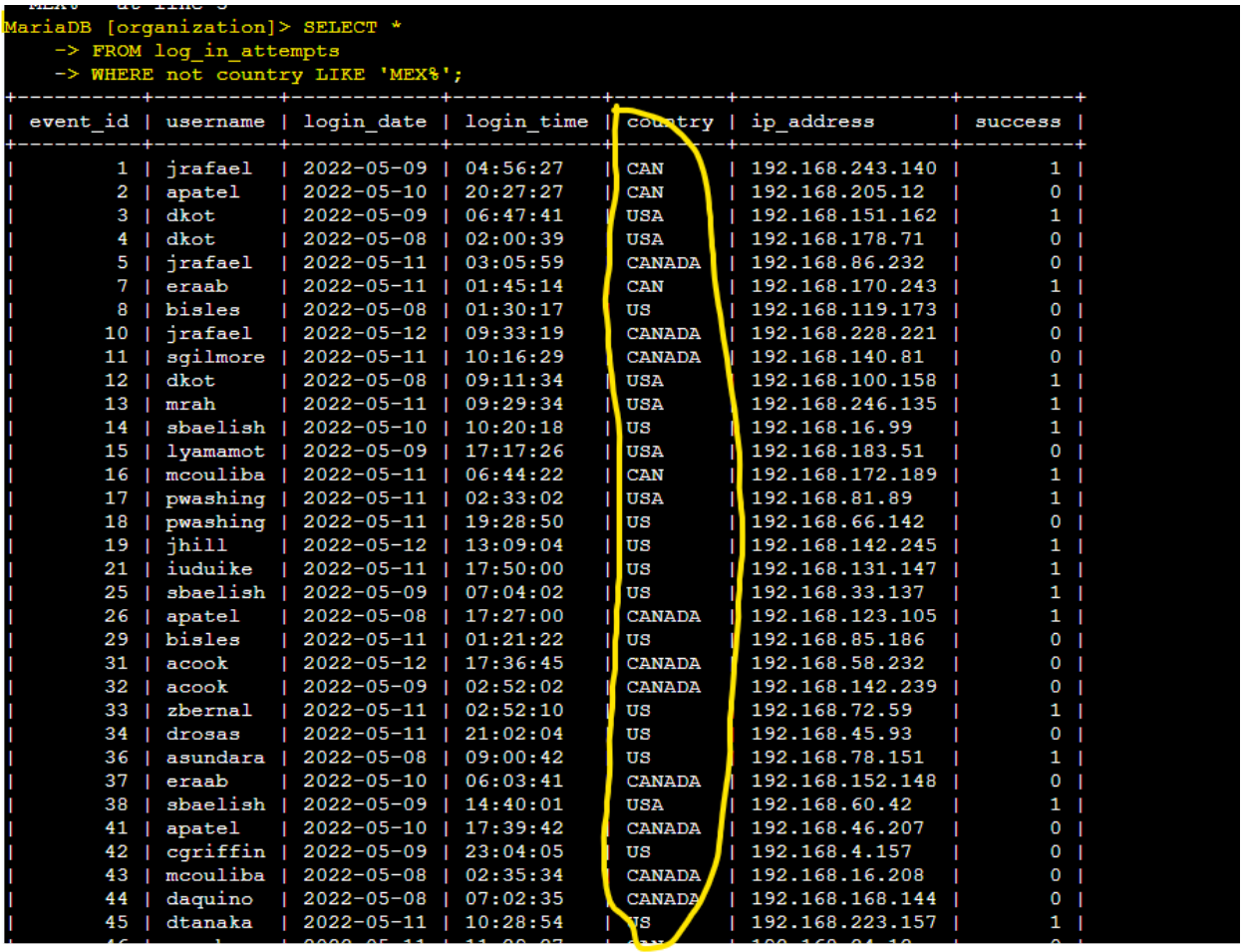
This SQL query aims to gather data from the "log_in_attempts" table with a focus on login activities within a specific time frame. By using the SELECT statement with an asterisk (*), the query ensures that all available columns will be included in the result.

The FROM clause indicates that the data will be retrieved from the "log_in_attempts" table, which likely contains records of login attempts, possibly along with related information like login times, dates, and success statuses.

The core of the query lies in the WHERE clause, where the condition is set to filter the login attempts based on their "login_date." The BETWEEN operator is used to specify a date range. In this case, the dates '2022-05-08' and '2022-05-09' are used to establish the range. This means that only the login attempts recorded on or after May 8, 2022, and on or before May 9, 2022, will be included in the query result.

The result of this query will offer valuable insights into the login activities that occurred on the specified days. It could be particularly useful for analyzing user behavior or identifying any potential issues related to login attempts during this specific time window.

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 |
|       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 |
```

SELECT *: This part of the query instructs the database to select all columns () from the "log_in_attempts" table. Instead of specifying individual column names, the asterisk () is used as a wildcard to include all columns in the result.

FROM log_in_attempts: This part of the query specifies the table from which the data will be retrieved. In this case, the table name is "log_in_attempts."

WHERE not country LIKE 'MEX%': This part of the query includes the condition for filtering the data. It tells the database to only retrieve rows where the "country" column does not start with the characters 'MEX'.

Here's how the filtering condition works:

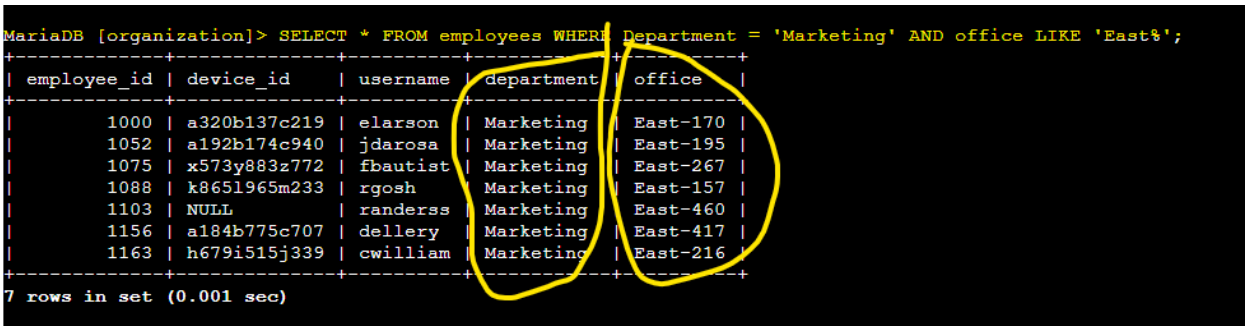
country: This refers to a column in the "log_in_attempts" table that likely contains information about the country associated with each login attempt.

LIKE 'MEX%': The LIKE operator is used for pattern matching. In this case, the condition 'MEX%' specifies that the "country" should start with the letters 'MEX', and the '%' symbol acts as a wildcard to allow any characters to follow.

not: The NOT keyword negates the condition. So, the overall condition becomes "not starting with 'MEX'".

In summary, the query is retrieving all columns from the "log_in_attempts" table for login attempts where the country associated with the attempt does not start with 'MEX'. This means the query will exclude any login attempts from Mexico, and the result will consist of login attempts from other countries. The purpose of this query might be to analyze login activities from countries other than Mexico or to focus on international login patterns.

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 (0.001 sec)
```

SELECT *: This part of the query instructs the database to select all columns () from the "employees" table. Using the asterisk () as a wildcard ensures that all available columns in the "employees" table will be included in the query result.

FROM employees: This part of the query specifies the table from which the data will be retrieved. In this case, the table name is "employees."

WHERE Department = 'Marketing' AND office LIKE 'East%': This part of the query includes the conditions for filtering the data. It tells the database to only retrieve rows that meet two specific conditions:

Department = 'Marketing': This condition checks the "Department" column and selects only those rows where the value is 'Marketing'. It narrows down the results to employees who belong to the Marketing department.

office LIKE 'East%': This condition checks the "office" column and selects only those rows where the value starts with the characters 'East'. The '%' symbol acts as a wildcard to allow any

characters to follow. So, this condition selects employees whose office location starts with 'East'. For example, this could include offices like 'East Coast', 'East Wing', 'Eastside Plaza', etc.

The query is retrieving all columns from the "employees" table for employees who belong to the Marketing department and have an office location that starts with 'East'. The result will include employees who meet both of these conditions, helping to narrow down the search to a specific group of employees within the Marketing department working in offices situated in the eastern region.

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   |
|        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    |
```

SELECT *: This part of the query instructs the database to select all columns () from the "employees" table. The asterisk () serves as a wildcard, ensuring that all available columns in the "employees" table will be included in the query result.

FROM employees: This part of the query specifies the table from which the data will be retrieved. In this case, the table name is "employees."

WHERE Department = 'Finance' OR Department = 'Sales': This part of the query includes the condition for filtering the data. It tells the database to retrieve rows that meet either of the following two conditions:

Department = 'Finance': This condition checks the "Department" column and selects rows where the value is 'Finance'. It includes employees who belong to the Finance department.

OR: The OR keyword is a logical operator that allows either condition to be true for a row to be included in the result. It means the query will also include employees who meet the next condition.

Department = 'Sales': This condition checks the "Department" column and selects rows where the value is 'Sales'. It includes employees who belong to the Sales department.

In summary, the query is retrieving all columns from the "employees" table for employees who belong to either the Finance department or the Sales department. The result will include employees from both departments, providing information about the individuals working in Finance and Sales within the organization.

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   |
|        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    |
```

SELECT *: This part of the query instructs the database to select all columns () from the "employees" table. The asterisk () is used as a wildcard to include all available columns in the query result.

FROM employees: This part of the query specifies the table from which the data will be retrieved. In this case, the table name is "employees."

WHERE NOT department = 'Information Technology': This is the condition for filtering the data. It tells the database to retrieve rows where the value in the "department" column is not equal to 'Information Technology'.

Here's how the filtering works:

department: This refers to a column in the "employees" table that likely stores the department to which each employee belongs.

NOT: The NOT keyword negates the condition. So, the condition becomes "not equal to 'Information Technology'".

In summary, the query is retrieving all columns from the "employees" table for employees who do not belong to the Information Technology department. The result will include employees from all other departments except for Information Technology, providing information about the individuals working in various departments within the organization, except IT.

Summary

We explored SQL queries used to retrieve specific data from a database table named "employees." The queries demonstrated various filtering techniques, such as selecting rows based on specific conditions like login times, success status, and office locations. We also encountered the use of logical operators like OR and NOT to filter data based on multiple criteria, and the LIKE operator for pattern matching. The queries were designed to help analyze login attempts, departmental affiliations, and office locations of employees, providing valuable insights into the organization's data and aiding decision-making processes.