Trent Dozier

Apply filters to SQL queries

Project description

You are a security professional at a large organization. Part of your job is to investigate security issues to help keep the system secure. You recently discovered some potential security issues that involve login attempts and employee machines.

Your task is to examine the organization's data in their *employees* and *log_in_attempts* tables. You'll need to use SQL filters to retrieve records from different datasets and investigate potential security issues.

Retrieve after hours failed login attempts

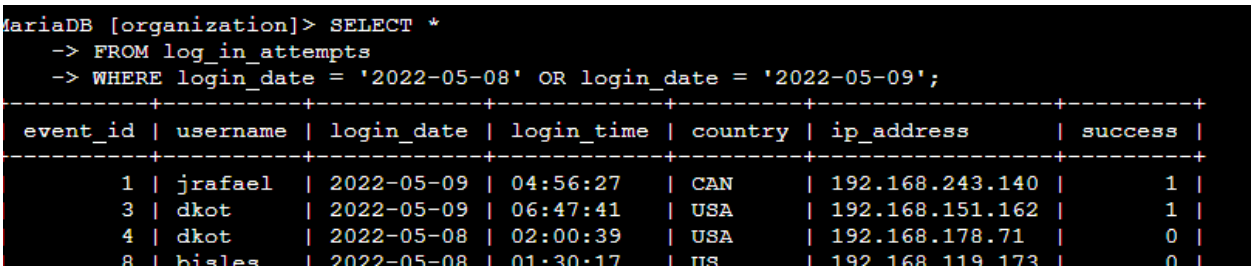
```
MariaDB [organization]> select *
    -> FROM log_in_attempts
    -> WHERE login_time > '18:00' AND success = '0';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 (0.172 sec)

MariaDB [organization]>
```

In plain English I typed select all from the database log_in_attempts where the column login_time is greater than 18:00 and the column success is set to 0. '0' is the boolean value for false meaning the login attempt failed.

SELECT * FROM log_in_attempts WHERE login_time > '18:00' AND success ='0';

Trent Dozier

Retrieve login attempts on specific dates

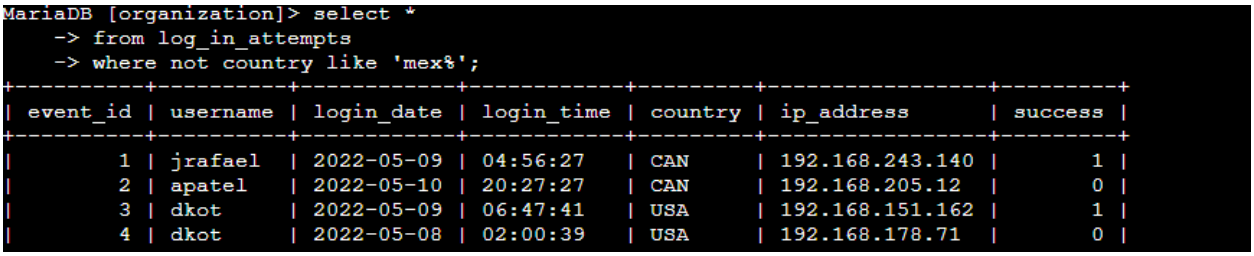


```
MariaDB [organization]> SELECT *
    -> FROM log_in_attempts
    -> WHERE login_date = '2022-05-08' OR login_date = '2022-05-09';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 |
```

Here I selected all from the database log_in_attempts where the column login_date was equal to 2022-05-09 or 2022-05-08.

SELECT * FROM log_in_attempts WHERE login_date = ‘2022-05-09’ OR ‘2022-05-08’;

Retrieve login attempts outside of Mexico

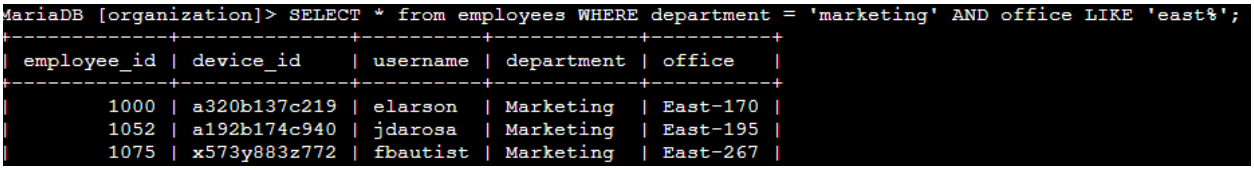


```
MariaDB [organization]> select *
    -> from log_in_attempts
    -> where not country like 'mex%';
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        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 |
```

Here, I stated select all from table log_in_attempts where the column country is not like mex%. I used the percent sign to remove any words that begin with mex in the country column because this database included both “Mexico” and “Mex”.

SELECT * FROM log_in_attempts WHERE NOT country like ‘mex%’:

Retrieve employees in Marketing in the East Building

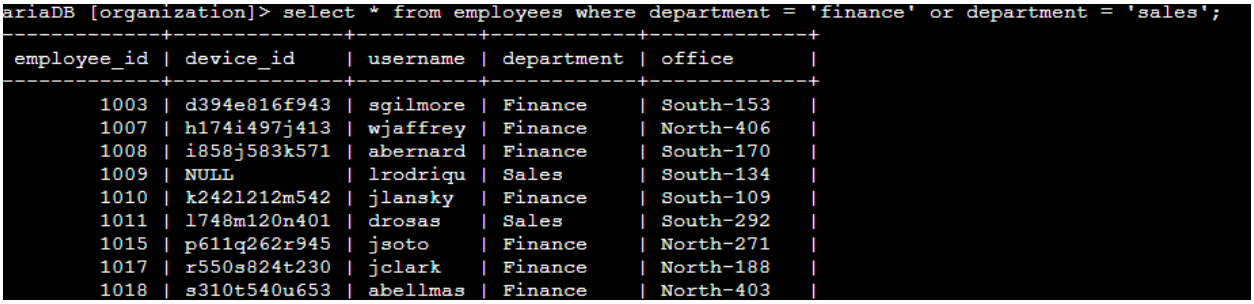


```
MariaDB [organization]> SELECT * from employees WHERE department = 'marketing' AND office LIKE 'east%';
+-------------+-------------+----------+------------+----------+
| employee_id | device_id   | username | department | office   |
+-------------+-------------+----------+------------+----------+
|        1000 | a320b137c219 | elarson  | Marketing  | East-170 |
|        1052 | a192b174c940 | jdarosa  | Marketing  | East-195 |
|        1075 | x573y883z772 | fbautist | Marketing  | East-267 |
```

Here I first ran SELECT * FROM employees; which allowed me to see the columns of the database that I need to what the column name is for departments and the buildings. Then I used select all from employees where the department equals marketing and the building is like east%. I used the wildcard % because the instructions dictate that the east building may have numbers after the name.

SELECT * FROM employees WHERE department = ‘marketing’ AND office LIKE ‘east%’;

Retrieve employees in Finance or Sales

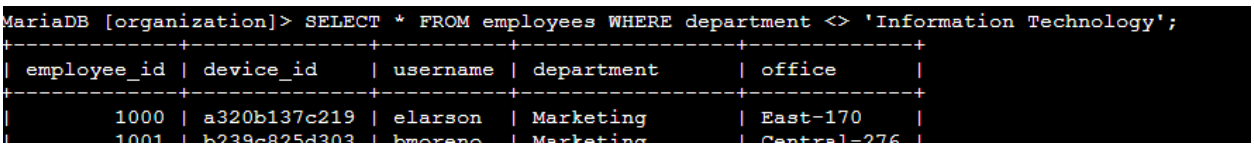


```
ariaDB [organization]> select * from employees where department = 'finance' or department = 'sales';
+-------------+-------------+----------+------------+-------------+
| employee_id | device_id   | username | department | office      |
+-------------+-------------+----------+------------+-------------+
|        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   |
```

Here, I selected all records from the employee table where the department equals finance or sales.

SELECT * FROM employees WHERE department = ‘finance’ OR department = ‘sales’;

Retrieve all employees not in IT



```
MariaDB [organization]> SELECT * FROM employees WHERE department <> 'Information Technology';
+-------------+-------------+----------+-----------------+-------------+
| employee_id | device_id   | username | department      | office      |
+-------------+-------------+----------+-----------------+-------------+
|        1000 | a320b137c219 | elarson  | Marketing       | East-170    |
```

Here, I wrote select all from employees table where department is not information technology. There are a couple of ways of using the NOT operator in this example I chose <> but I could have also used WHERE NOT.

SELECT * FROM employees WHERE department <> ‘Information technology’;

Summary

In this activity I used SQL statements to find and filter through data. I used two different tables log_in_attempts and employees. I used the AND, OR, NOT operators to filter for specific information. I also used LIKE and % wildcard to filter for patterns.