Apply filters to SQL queries (My Responses)

Project description

With the given scenario provided, it will teach me how to make a system more secure, safe, investigate all potential security issues, and update employee computers as needed. I will learn to provide examples of how I used SQL with filters to perform security-related tasks. And so, in this activity, I will focus on filtering data from a database using SQL to help with security analysis and system updates. The goal is to practice creating complex queries with multiple conditions using operators like AND, OR, and NOT. The project will require me to retrieve specific data, such as failed login attempts after business hours, logins from certain dates, and logins that didn't come from Mexico. I will also need to filter employee information, such as retrieving details of those in the Marketing department or employees who work in either the Finance or Sales departments. Additionally, I will practice excluding employees from the IT department. By completing these tasks, I will strengthen my ability to apply SQL filters to real-world scenarios and improve my data analysis skills in a security context.

Retrieve after hours failed login attempts

In this task, my team is investigating failed login attempts that occurred after business hours. The objective is to retrieve all failed login attempts that happened after 18:00. By filtering the log_in_attempts table based on the login_time column and using the success column to identify unsuccessful logins, I can isolate the necessary data.

Goal: Use the AND operator to filter out failed login attempts that occurred after 18:00.
Solution: The SQL query uses login_time > '18:00' and success = FALSE to retrieve the data.

The following code demonstrates how I created a SQL query to filter for failed login attempts that occurred after business hours.

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

Q & A: How many failed login attempts occurred after 18:00? There are 19 failed login attempts that occurred after 18:00.

```
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 (0.001 sec)

MariaDB [organization]>
```

The first part of the screenshot is my query, and the second part is a portion of the output. This query filters for failed login attempts that occurred after 18:00. First, I started by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause with an `AND` operator to filter my results to output only login attempts that occurred after 18:00 and were unsuccessful. The first condition is `login_time > '18:00'`, which filters for the login attempts that occurred after 18:00. The second condition is `success = FALSE`, which filters for the failed login attempts.

Retrieve login attempts on specific dates

My team is investigating a suspicious event that occurred on '2022-05-09'. We need to retrieve login attempts from both '2022-05-09' and the previous day, '2022-05-08'. By applying the OR operator, I can pull login attempts for these two specific dates.

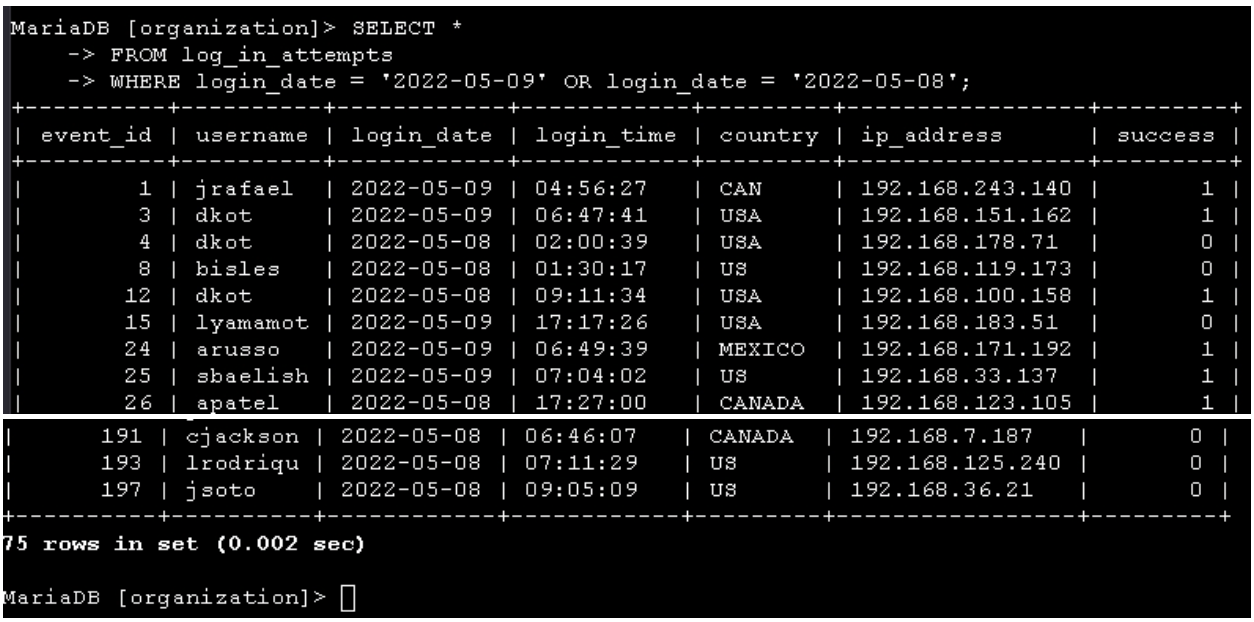
Goal: Retrieve login attempts made on '2022-05-09' and '2022-05-08'.
Solution: The SQL query uses the OR operator to combine both dates.

The following code demonstrates how I created a SQL query to filter for login attempts that occurred on specific dates.

```
SELECT *
FROM log_in_attempts
```

```
WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
```

Q & A: How many login attempts were made on these two days? There are 75 login attempts in these two days.



```
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 |
|      191 | cjackson | 2022-05-08 | 06:46:07   | CANADA  | 192.168.7.187   |       0 |
|      193 | lrodriqu | 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)

MariaDB [organization]>
```

The first part of the screenshot is my query, and the second part is a portion of the output. This query returns all login attempts that occurred on 2022-05-09 or 2022-05-08. First, I started by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause with an `OR` operator to filter my results to output only login attempts that occurred on either 2022-05-09 or 2022-05-08. The first condition is `login_date = '2022-05-09'`, which filters for logins on 2022-05-09. The second condition is `login_date = '2022-05-08'`, which filters for logins on 2022-05-08.

Retrieve login attempts outside of Mexico

Now, we need to identify login attempts that didn't originate from Mexico. The country field contains entries with values like 'MEX' and 'MEXICO'. To filter out logins originating in Mexico, I will use the NOT operator with a pattern match using the LIKE operator.

Goal: Retrieve login attempts from countries other than Mexico by excluding any entry that matches 'MEX%' in the country column.
Solution: The query uses the NOT operator with LIKE 'MEX%' to exclude Mexico-based logins.

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

```
SELECT *
FROM log_in_attempts
WHERE NOT country LIKE 'MEX%';
```

Q & A: How many login attempts were made outside of Mexico? There are 144 login attempts made 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 |
|      196 | acook    | 2022-05-10 | 09:56:48   | CAN     | 192.168.52.90   |       0 |
|      197 | jsoto    | 2022-05-08 | 09:05:09   | US      | 192.168.36.21   |       0 |
|      200 | jclark   | 2022-05-12 | 01:11:45   | CANADA  | 192.168.91.103  |       1 |
+----------+----------+------------+------------+---------+-----------------+---------+
144 rows in set (0.025 sec)

MariaDB [organization]>
```

The first part of the screenshot is my query, and the second part is a portion of the output. This query returns all login attempts that occurred in countries other than Mexico. First, I started by selecting all data from the `log_in_attempts` table. Then, I used a `WHERE` clause with `NOT` to filter for countries other than Mexico. I used `LIKE` with `MEX%` as the pattern to match because the dataset represents Mexico as `MEX` and `MEXICO`. The percentage sign (`%`) represents any number of unspecified characters when used with `LIKE`.

Retrieve employees in Marketing

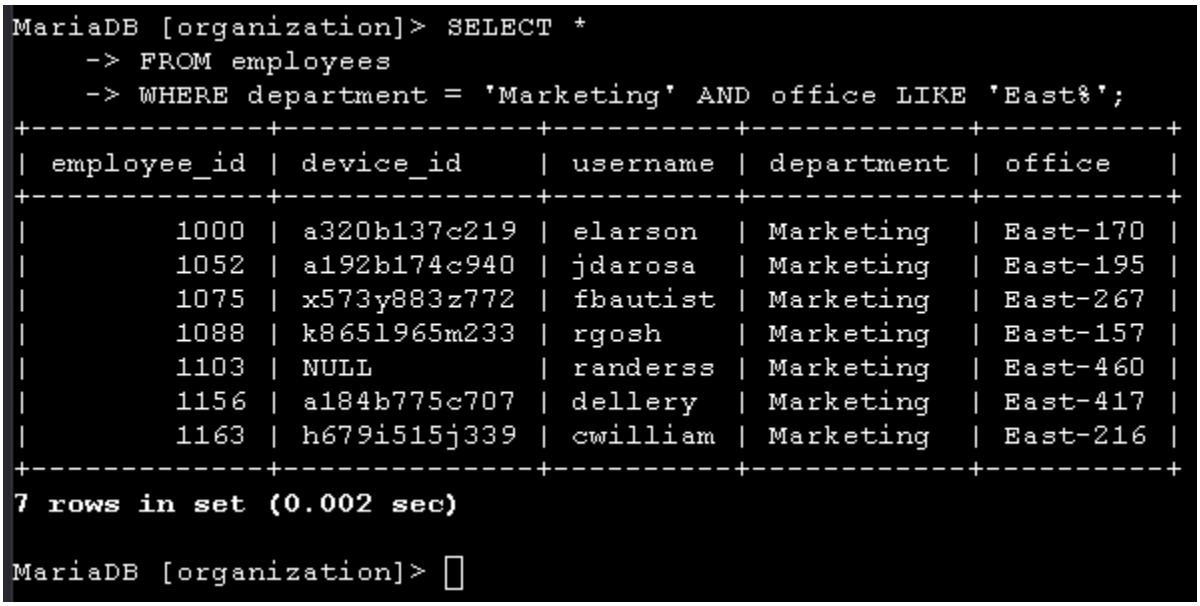
For this task, I need to obtain information about employees in the Marketing department who are located in offices in the East building (like 'East-170' or 'East-320'). Using the AND and LIKE operators, I can filter the records based on the department and office.

Goal: Retrieve employee data for those in the 'Marketing' department and located in offices starting with 'East'.
Solution: The SQL query uses the AND operator for filtering by department and LIKE 'East%' for the office location.

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

```
SELECT *
FROM employees
WHERE department = 'Marketing' AND office LIKE 'East%';
```

Q & A: What is the username of the first employee in the Marketing department in the East building? The username of the first employee in the Marketing department in the East building is elarson.



```
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)

MariaDB [organization]>
```

The first part of the screenshot is my query, and the second part is a portion of the output. This query returns all employees in the Marketing department in the East building. First, I started by selecting all data from the `employees` table. Then, I used a `WHERE` clause with `AND` to filter for employees who work in the Marketing department and in the East building. I used `LIKE` with `East%` as the pattern to match because the data in the `office` column represents the East building with the specific office number. The first condition is the `department = 'Marketing'` portion, which filters for employees in the Marketing department. The second condition is the `office LIKE 'East%'` portion, which filters for employees in the East building.

Retrieve employees in Finance or Sales

Next, I need to retrieve employees in the Finance or Sales department for an upcoming system update. The OR operator will allow me to filter records where the department is either 'Finance' or 'Sales'.

Goal: Retrieve employee data for employees in the 'Finance' or 'Sales' department.
Solution: The SQL query uses OR to combine the two department conditions.

The following code demonstrates how I created a SQL query to filter for employee machines from employees in the Finance or Sales departments.

```
SELECT *
FROM employees
WHERE department = 'Finance' OR department = 'Sales';
```

Q & A: What is the username of the first employee in the Sales department returned by the query? The username of the first employee in the Sales department is lrodriqu.

```
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   |
|        1188 | g164h566i795 | noshiro  | Finance    | West-252    |
|        1195 | n516o853p957 | orainier | Finance    | East-346    |
+-------------+--------------+----------+------------+-------------+
71 rows in set (0.001 sec)

MariaDB [organization]>
```

The first part of the screenshot is my query, and the second part is 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 the Finance and Sales departments. I used the `OR` operator instead of `AND` because I want all employees who are in either department. The first condition is `department = 'Finance'`, which filters for employees from the Finance department. The

second condition is `department = 'Sales'`, which filters for employees from the Sales department.

Retrieve all employees not in IT

Finally, I need to retrieve information on employees who are not in the Information Technology department. The NOT operator helps me filter out those who are in the IT department.

Goal: Exclude employees who belong to the 'Information Technology' department.
Solution: The SQL query uses NOT department = 'Information Technology' to filter out IT employees.

The following demonstrates how I created a SQL query to filter for employee machines from employees not in the Information Technology department.

```
SELECT *
FROM employees
WHERE NOT department = 'Information Technology';
```

Q & A: How many employees are not in the Information Technology department? There are 161 employees who aren't in the Information Technology department.

```
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   |
|        1195 | n516o853p957 | orainier | Finance         | East-346    |
|        1198 | q308r573s459 | jmartine | Marketing       | South-117   |
|        1199 | r520s571t459 | areyes   | Human Resources | East-100    |
+-------------+--------------+----------+-----------------+-------------+
161 rows in set (0.001 sec)

MariaDB [organization]>
```

The first part of the screenshot is my query, and the second part is a portion of the output. The query returns all employees not in the Information Technology department. First, I started

by selecting all data from the `employees` table. Then, I used a `WHERE` clause with `NOT` to filter for employees not in this department.

Summary

Throughout this exercise, I've gained valuable experience using SQL to solve specific data retrieval challenges. By applying operators like AND, OR, and NOT, I've been able to filter data effectively to meet the needs of a security investigation and system updates. Each task addressed a unique scenario, and by writing queries to filter for things like failed login attempts after hours, logins on specific dates, and logins outside of Mexico, I was able to focus on the data that really mattered. For example, when investigating failed login attempts after business hours, I used the AND operator to narrow the results to only those after 18:00. Similarly, I used the OR operator to pull data for login attempts on specific dates, ensuring that we could investigate key events. For identifying logins outside of Mexico, I used the NOT operator along with the LIKE clause to exclude any login attempts from Mexico, which allowed us to focus on potential foreign threats. In the case of employees in certain departments, such as Marketing or Sales, I combined operators like AND and OR to filter the data based on department and office location. This allowed the team to quickly locate employees needing updates. Finally, by using the NOT operator, I could filter out employees in the IT department, ensuring they were excluded from updates that had already been completed.
After this activity, I can comprehend and code complex SQL queries since this experience has sharpened my skills in retrieving specific data from a database and applying filters to get exactly what's needed for security tasks and system management.