

Apply Filters to SQL Queries for Company's Cybersecurity

Project description

My organization is working to make their system more secure. It is my job to ensure the system is safe, investigate all potential security issues, and update employee computers as needed. The following steps provide examples of how I used SQL with filters to perform security-related tasks.

Retrieve after hours failed login attempts.

To investigate a potential security incident that occurred after business hours, I needed to query the `log_in_attempts` table and review login activity that happened after 18:00. The following code demonstrates how I created a SQL query to filter for failed login attempts that occurred after business 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

I selected all columns from the `log_in_attempts` table. I used the `WHERE` clause with the `AND` operator to filter for two conditions. In the first instance, I used `login_time > '18:00'`, because this filters for login attempts that occurred after 6:00 PM. Also I used, `success = 0`, due to these filters for failed login attempts where success is FALSE. Both conditions must be true to identify failed attempts that happened after hours.

Retrieve login attempts on specific dates.

A suspicious event occurred on 2022-05-09. Any login activity that happened on 2022-05-09 or on the day before needs to be investigated.

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

```
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	lyamamoto	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

I selected all columns from the `log_in_attempts` table. I used the `WHERE` clause with the `OR` operator. This was necessary because a login attempt cannot happen on two different dates simultaneously. The query returns all records where the `login_date` was either May 9th, 2022, OR May 8th, 2022.

Retrieve login attempts outside of Mexico.

After determining that the suspicious activity did not originate in Mexico, I needed to filter the data to find login attempts that occurred outside that country.

```
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

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.

The team wanted to perform security updates on specific employee machines in the Marketing department, specifically those located 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

I selected all columns from the `employee's` table. I used the `AND` operator to strictly filter for rows that met both criteria, as the `department` column must equal 'Marketing' and the `office` column must start with 'East' (captured by `LIKE 'East%'`), covering all offices in the East building.

Retrieve employees in Finance or Sales

A different security update was required for employees in the Sales and Finance departments.

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

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	k242l212m542	jlansky	Finance	South-109
1011	l748m120n401	drosas	Sales	South-292
1015	p611q262r945	jsoto	Finance	North-271

I selected all columns from the **employee's** table. I used the **OR** operator because I needed to create a list comprising employees from either department. The query returns to a record if the **department** is 'Sales' OR if it is 'Finance'.

Retrieve all employees not in IT.

Finally, the team needed to make one more update for all employees who are *not* 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
1007	h174i497j413	wjaffrey	Finance	North-406

The query returns all employees not in the Information Technology department. First, I started by selecting all data from the **employee's** table. Then, I used a **WHERE** clause with **NOT** to filter employees not in this department.

Summary

In this activity, I used SQL to investigate potential security issues and isolate specific data for updates. I applied the **AND** operator to filter for records meeting multiple specific criteria (like failed logins after a certain time). I used the **OR** operator to broaden the search to include multiple possibilities (like different dates or departments). Finally, I used the **NOT** operator to exclude specific data points (like excluding a country or department) to focus the investigation on the relevant targets.