

Apply filters to SQL queries

Project description

The security team at my organization needs to investigate security issues that involve login attempts and employee machines. There are different tables in the organization's database to view the **employees** and **log_in_attempts** so I will use SQL queries to filter different sections of these tables to find irregular events and systems.

Retrieve after hours failed login attempts

It was recently discovered that a potential security incident occurred after business hours. To investigate this, I ran a SQL query to view all failed login attempts made after hours.

```
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)

The first three lines of the above screenshot display the SQL query used to select all columns from the **log_in_attempts** table where the *login_time* and *login success* columns were after hours and false. The following results displayed 19 rows of failed login attempts made after hours, with their corresponding details such as who made the login attempt, what country it originated from, and the IP address of the requester.

Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09 and to investigate this event I reviewed all login attempts which occurred on this day and 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

In the query above, I selected all columns from the **log_in_attempts** table and filtered the results to the attempts that only occurred on 2022-05-09 and 2022-05-08 using SQL's **OR** operator.

Retrieve login attempts outside of Mexico

There was also suspicious activity with login attempts that did not originate from Mexico. Using SQL I created a filtered all login attempts that occurred from all other countries than 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	mrh	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

In the query, I combined SQL's **NOT** operator with the **LIKE** operator and % wildcard to find all logins outside of Mexico. The *country* column in the table could potentially have values **LIKE** "MEX" or "MEXICO", so I needed the % wildcard to find all values that started with "MEX". Then, the **NOT** operator ensured that I got every record that did not satisfy the wildcard filter.

Retrieve employees in Marketing

The security team wanted to perform security updates on specific employee machines in the marketing department. I needed to query for all employees in the marketing department that were in all offices 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 |
| 1088 | k865l965m233 | 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 (0.032 sec)

```

Using SQL's **AND** and **LIKE** operators I was able to filter the employees table to find all employees that were assigned to the "Marketing" department and were in the east office building. The office building could contain many different values such as "East-170" or "East-460", so I again utilized the % operator to search for values in the office column that begin with "East".

Retrieve employees in Finance or Sales

Machines for employees in the Finance and Sales departments needed different security updates. Using SQL, I filtered the employee table to find all employees in these 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 | k242l212m542 | 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 | j236k303l245 | bisles | Sales | South-171 |
| 1039 | n253o917p623 | cjackson | Sales | East-378 |
| 1041 | p929q222r778 | cgriffin | Sales | North-208 |

```

In the above query I used SQL's **OR** operator to filter the *department* column for all employees in either the Finance or Sales department.

Retrieve all employees not in IT

Employees in the IT department already had received a new security update, but all other departments needed to receive it. I used SQL to find all employees and systems that were not a part of IT. Using the

```

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 | k242l212m542 | 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 |

```

In this query, I filter the *department* column using the **NOT** operator to find all rows where the department is not “Information Technology”.

Summary

Using SQL queries, I was able to report back to my security team with all the relevant login events and employees that related to suspicious login attempts and system security updates. With SQL operators such as **AND**, **NOT**, **LIKE**, and **%** I was able to filter thousands of records to find login attempts on certain days, login attempts from specific countries, failed login attempts, and employees from specific departments. In this project I was able to utilize SQL to retrieve records from the database that would have taken hours to complete by manually searching for them.