

Apply filters to SQL queries

Project description

The organization's security team is working to make their system more secure. The team must investigate potential security issues. The following steps provide examples of how the team used the SQL database with filters to perform the investigation.

Retrieve after hours failed login attempts

The security team discovered a potential security incident that occurred after business hours. To conclude this, the team queried the `log_in_attempts` table and review after hours activity. Office hours end at 18:00.

```
MariaDB [organization]> SELECT*
-> FROM log_in_attempts
-> WHERE login_time > "18:00:00";
```

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
23	yappiah	2022-05-10	18:11:53	MEXICO	192.168.200.48	1
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
51	jrafael	2022-05-10	22:40:01	CANADA	192.168.148.115	1
52	cjackson	2022-05-10	22:07:07	CAN	192.168.58.57	0
54	jreckley	2022-05-10	19:31:19	MEXICO	192.168.167.152	1
57	asundara	2022-05-12	21:13:02	US	192.168.211.201	1
60	acook	2022-05-11	21:46:00	CAN	192.168.54.45	1
64	apatel	2022-05-10	22:00:09	CANADA	192.168.172.71	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
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
105	cjackson	2022-05-12	19:36:42	CAN	192.168.247.153	1
107	bisles	2022-05-12	20:25:57	USA	192.168.116.187	0
108	daquino	2022-05-09	21:30:48	CANADA	192.168.15.110	1
111	aestrada	2022-05-10	22:00:26	MEXICO	192.168.76.27	0
115	ivelasco	2022-05-10	23:06:01	CAN	192.168.154.1	1
116	tmitchel	2022-05-10	20:33:27	MEXICO	192.168.119.26	1
118	smartell	2022-05-12	23:21:31	MEXICO	192.168.173.196	1
119	tmitchel	2022-05-11	23:07:13	MEXICO	192.168.110.175	1
121	btang	2022-05-10	22:00:36	US	192.168.80.143	1
126	jrafael	2022-05-12	18:47:52	CAN	192.168.22.16	1
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
132	rjensen	2022-05-12	23:26:03	MEX	192.168.9.166	1
155	cgriffin	2022-05-12	22:18:42	USA	192.168.236.176	0
158	smartell	2022-05-09	19:30:32	MEXICO	192.168.190.178	1
160	jclark	2022-05-10	20:49:00	CANADA	192.168.214.49	0
164	jclark	2022-05-12	21:15:52	CAN	192.168.18.34	1
173	asundara	2022-05-12	23:17:52	US	192.168.58.217	1
199	yappiah	2022-05-11	19:34:48	MEXICO	192.168.44.232	0

```
39 rows in set (0.001 sec)

MariaDB [organization]>
```

The **SELECT** keyword indicates which columns to return. After that we must indicate the name of the columns (E.g. "**username, login_date**"), but in this case we used an "*" that represent all the columns.

The **FROM** keyword indicates which table to query. In this case is the **log_in_attempts** table, as we mentioned before.

WHERE keyword indicates the condition for a filter. Here we wanted to filter by the **login_time**. We set that **login_time** must be greater than (>) "18:00". The ">" symbol is what we call an operator. There are some more operators such as "<" (less than), "=" (equal to), "<=" (less than or equal to), ">=" (greater than or equal to) and "<>" (not equal to).

Retrieve login attempts on specific dates

A suspicious event occurred on 2022-05-09. To investigate this event, the team wants to review all the login attempts which occurred on this day and the day before.

```
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
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	mmason	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	mrh	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

This time, we filtered by the dates that we were looking for with the help of the "OR" and "=" operators. "OR" connects the conditions we provide, in this case the dates that we set after "login_date=". The output will show only the login attempts on the date 2022-05-08 and 2022-05-09.

Retrieve login attempts outside of Mexico

The team concluded that the suspicious activity didn't originate from Mexico. This time, the team must investigate all login attempts that occurred 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	iuduke	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
46	eraab	2022-05-11	11:29:27	CAN	192.168.24.12	0
47	dkot	2022-05-08	05:06:45	US	192.168.233.24	1
48	asundara	2022-05-11	03:18:45	USA	192.168.72.10	1
49	asundara	2022-05-08	14:00:01	US	192.168.173.213	0
50	jclark	2022-05-10	10:48:02	CANADA	192.168.174.117	0
51	jrafael	2022-05-10	22:40:01	CANADA	192.168.148.115	1
52	cjackson	2022-05-10	22:07:07	CAN	192.168.58.57	0
53	nmason	2022-05-08	11:51:38	CAN	192.168.133.188	1
55	jlansky	2022-05-11	05:15:34	US	192.168.6.170	0
56	acook	2022-05-08	04:56:30	CAN	192.168.209.130	1
57	asundara	2022-05-12	21:13:02	US	192.168.211.201	1
58	ivelasco	2022-05-09	17:20:54	CAN	192.168.57.162	0
60	acook	2022-05-11	21:46:00	CAN	192.168.54.45	1

With the help of “**NOT**” and “**LIKE**” we managed to exclude Mexico from the output. Placing “**NOT**” operator right next to the “**WHERE**” keyword will negate the condition that comes after. Here the condition is the “**country**”. In the country column, Mexico appears as either “**MEXICO**” or “**MEX**”, so we have to use the “**LIKE**” operator. “**LIKE**” operator works with the use of a wildcard that can be either “**%**” or “**_**”. The “**%**” sign substitutes any number of characters, and on the contrary the “**_**” symbol only substitutes one character. These wildcards can be placed after a string (like the example), before a string or in both locations.

Retrieve employees in Marketing

The team needs to perform a security update on the machines of the Marketing department employees who are in the east building. We need to filter in a way to show only the employees in that department and in the east building. This time to achieve this we have to query the “employees” table.

```
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 | randers | Marketing | East-460 |
| 1156 | a184b775c707 | dellery | Marketing | East-417 |
| 1163 | h679i515j339 | cwilliam | Marketing | East-216 |
+-----+-----+-----+-----+-----+
7 rows in set (0.074 sec)

MariaDB [organization]> 
```



This output had to combine two conditions. For this to happen we have to use the “AND” operator, and in this case we had to combine that the department had to be from marketing and the office from the East building.

Retrieve employees in Finance or Sales

The team needs to perform a security update on the machines for employees in the Sales and Finance departments. To meet this requirement, the output must only show employees from Sales and Finance departments in order to see which machines need the update.

```
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	drozas	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
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	k367l639m697	redwards	Finance	North-180
1063	l686m140n569	lpope	Sales	East-226
1066	o678p794q957	ttyrell	Sales	Central-444
1069	NULL	jpark	Finance	East-110
1071	t244u829v723	zdutchma	Sales	West-348
1072	u905v920w694	esmith	Sales	East-421
1076	y347z204a710	fgarcia	Finance	Central-270
1078	a667b270c984	sharley	Sales	North-418
1081	d647e310f618	qcorbit	Finance	South-290
1083	f840g812h544	gkoshi	Finance	West-165
1085	h339i498j269	cperez	Sales	East-325
1086	i281j129k749	lmajumda	Sales	West-499
1089	l358m929n154	jpark2	Sales	West-251
1091	n378o313p469	rtran	Sales	Central-230
1092	o391p779q935	lpark	Sales	West-227
1098	u671v146w618	tarchamb	Sales	North-423
1099	v283w690x104	anaser	Finance	West-357
1105	b551c837d758	kmei	Finance	Central-232
1107	d168e758f876	akajwara	Sales	North-471
1109	f229g533h679	nlocklea	Sales	East-196
1110	g567h376i314	pchaudhu	Sales	Central-428

The team once again used the “” and “” operator. This time we used to filter a string of data and not numeric data.

Retrieve all employees not in IT

The team needs to make one more update to employee machines, that update was already installed in the IT department machines. The filter must exclude the IT department from the output.

```
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	jlsansky	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	j236k303l245	bisles	Sales	South-171
1036	k550l533m205	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
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
1051	z451a308b518	itraora	Marketing	Central-134
1052	a192b174c940	jdarosa	Marketing	East-195
1053	b979c871d361	nemmanue	Human Resources	Central-259
1055	d831e972f553	awilliam	Marketing	Central-256
1056	e782f537g683	ankala	Marketing	North-139
1057	f370g535h632	mccott	Sales	South-270

Once again, the team used the “**NOT**” operator to, this time, exclude the IT department from the output.

Summary

The security team applied filters to SQL queries in two different tables, **log_in_attempts** and **employees**. With the use of proper operators such as “**AND**”, “**OR**”, “**NOT**” and the use of “**LIKE**”

with the “%” wildcard, the team managed to filter the database in a way to get an easier and smaller output.