

Apply filters to SQL queries

Apply filters to SQL queries	1
Project description	1
Retrieve after hours failed login attempts.....	1
Retrieve login attempts on specific dates.....	2
Retrieve login attempts outside of Mexico	3
Retrieve employees in Marketing	4
Retrieve employees in Finance or Sales.....	5
Retrieve all employees not in IT	6
Summary	6

Project description

My team is focused on enhancing our system's security, and I'm responsible for maintaining its safety. My duties include scrutinizing potential security vulnerabilities and updating staff computers as required. Below are instances illustrating how I've employed SQL and filters to execute tasks related to security.

Retrieve after hours failed login attempts

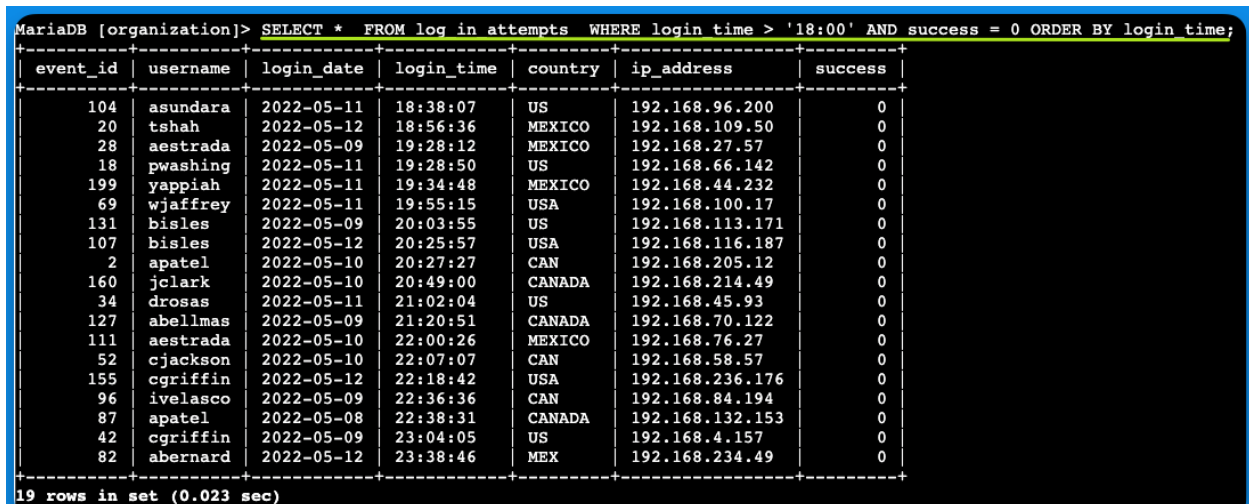
There was a potential security incident that occurred after business hours (after 18:00). All after hours login attempts that failed need to be investigated.

The following query selects all failed login attempts from the log_in_attempts table where the login time is after 18:00. The results are ordered by login time.

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

explanation of the query:

1. The SELECT statement selects all columns from the log_in_attempts table.
2. The WHERE clause filters the results to include only failed login attempts (where success = 0) that occurred after 18:00 (where login_time > '18:00').
3. The ORDER BY clause orders the results by login time.



MariaDB [organization]> SELECT * FROM log_in_attempts WHERE login_time > '18:00' AND success = 0 ORDER BY login_time;

event_id	username	login_date	login_time	country	ip_address	success
104	asundara	2022-05-11	18:38:07	US	192.168.96.200	0
20	tshah	2022-05-12	18:56:36	MEXICO	192.168.109.50	0
28	astrada	2022-05-09	19:28:12	MEXICO	192.168.27.57	0
18	pwashing	2022-05-11	19:28:50	US	192.168.66.142	0
199	yappiah	2022-05-11	19:34:48	MEXICO	192.168.44.232	0
69	wjaffrey	2022-05-11	19:55:15	USA	192.168.100.17	0
131	bisles	2022-05-09	20:03:55	US	192.168.113.171	0
107	bisles	2022-05-12	20:25:57	USA	192.168.116.187	0
2	apatel	2022-05-10	20:27:27	CAN	192.168.205.12	0
160	jclark	2022-05-10	20:49:00	CANADA	192.168.214.49	0
34	drosas	2022-05-11	21:02:04	US	192.168.45.93	0
127	abellmas	2022-05-09	21:20:51	CANADA	192.168.70.122	0
111	astrada	2022-05-10	22:00:26	MEXICO	192.168.76.27	0
52	cjackson	2022-05-10	22:07:07	CAN	192.168.58.57	0
155	cgriffin	2022-05-12	22:18:42	USA	192.168.236.176	0
96	ivelasco	2022-05-09	22:36:36	CAN	192.168.84.194	0
87	apatel	2022-05-08	22:38:31	CANADA	192.168.132.153	0
42	cgriffin	2022-05-09	23:04:05	US	192.168.4.157	0
82	abernard	2022-05-12	23:38:46	MEX	192.168.234.49	0

19 rows in set (0.023 sec)

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.

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';
```

This query selects all login attempts from the log_in_attempts table where the login date is either May 8th, 2022 or May 9th, 2022.

Here is a step-by-step explanation of the query:

1. The SELECT statement selects all columns from the log_in_attempts table.
2. The WHERE clause filters the results to include only login attempts that occurred on either May 8th, 2022 or May 9th, 2022.
3. The OR operator is used to combine the two date filters.

```
mysql [organization]> SELECT *  
-> FROM log_in_attempts  
-> WHERE login_date = '2022-05-09' OR login_date = '2022-05-08'  
-> ORDER BY login_date;
```

event_id	username	login_date	login_time	country	ip_address	success
117	bsand	2022-05-08	00:19:11	USA	192.168.197.187	0
56	acook	2022-05-08	04:56:30	CAN	192.168.209.130	1
169	alevitsk	2022-05-08	08:10:43	CANADA	192.168.210.228	0
168	jlansky	2022-05-08	13:25:42	USA	192.168.210.94	1
66	aestrada	2022-05-08	21:58:32	MEX	192.168.67.223	1
165	jreckley	2022-05-08	15:28:43	MEXICO	192.168.34.193	0
68	mrah	2022-05-08	17:16:13	US	192.168.42.248	1
163	tmitchel	2022-05-08	09:21:16	MEX	192.168.119.29	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
147	yappiah	2022-05-08	06:04:34	MEX	192.168.65.245	0
92	pwashing	2022-05-08	00:36:12	US	192.168.247.219	0
197	jsoto	2022-05-08	09:05:09	US	192.168.36.21	0
150	nmason	2022-05-08	14:40:02	CAN	192.168.204.124	0
101	sbaelish	2022-05-08	12:01:22	US	192.168.145.158	0
148	daquino	2022-05-08	06:15:55	CANADA	192.168.135.6	1
145	ivelasco	2022-05-08	09:06:02	CANADA	192.168.39.196	1
53	nmason	2022-05-08	11:51:38	CAN	192.168.133.188	1
49	asundara	2022-05-08	14:00:01	US	192.168.173.213	0
72	alevitsk	2022-05-08	12:09:10	CANADA	192.168.139.176	1
172	mabadi	2022-05-08	08:06:50	US	192.168.180.41	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
12	dkot	2022-05-08	09:11:34	USA	192.168.100.158	1
8	bisles	2022-05-08	01:30:17	US	192.168.119.173	0
4	dkot	2022-05-08	02:00:39	USA	192.168.178.71	0
47	dkot	2022-05-08	05:06:45	US	192.168.233.24	1
189	nmason	2022-05-08	05:37:24	CANADA	192.168.168.117	1

Retrieve login attempts outside of Mexico

After investigating the organization's data on login attempts, I believe there is an issue with the login attempts that occurred outside of Mexico. These login attempts should be investigated.

The following query selects all login attempts from the `log_in_attempts` table where the country is not Mexico.

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

Here is a step-by-step explanation of the query:

1. The `SELECT` statement selects all columns from the `log_in_attempts` table.
2. The `WHERE` clause filters the results to include only login attempts where the country is not Mexico.
3. The `NOT` operator is used to negate the country filter.
4. The `LIKE` operator is used to compare the country column to the string `'MEX%'`. The `%` wildcard character matches any number of characters.

```
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	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	agook	2022-05-12	17:36:45	CANADA	192.168.58.232	0

Retrieve employees in Marketing

My team wants to update the computers for certain employees in the Marketing department. To do this, I have to get information on which employee machines to update.

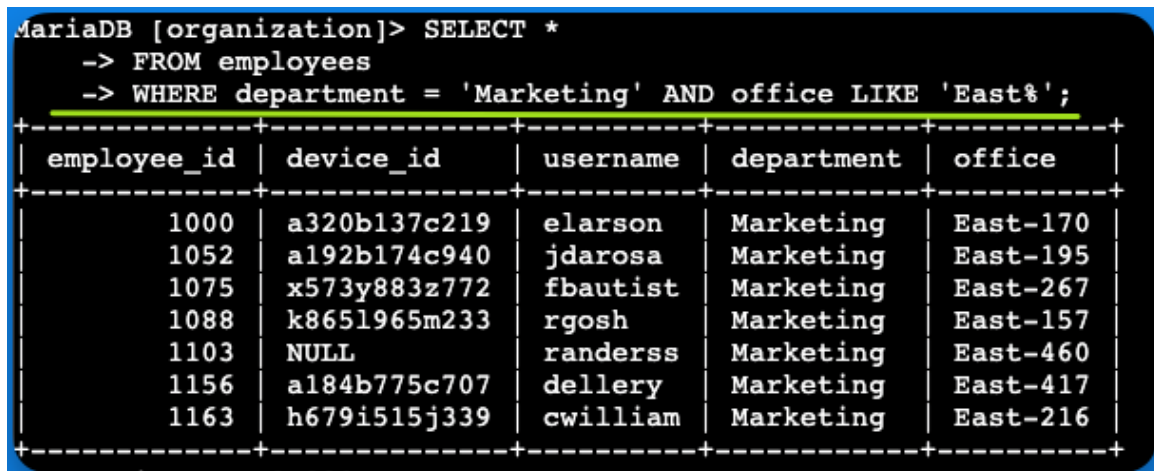
The following code demonstrates how I created a SQL query to filter for employee machines from employees in the Marketing department in the East building.

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

This query returns all columns from the employees table for employees who work in the Marketing department and have an office in the eastern region.

Here is a step-by-step explanation of the query:

1. The SELECT statement selects all columns from the employees table.
2. The WHERE clause filters the results to include only employees where the department is 'Marketing' and the office is in the eastern region.
3. The AND operator is used to combine the two filters.
4. The LIKE operator is used to compare the office column to the string 'East%'. The % wildcard character matches any number of characters



The screenshot shows a terminal window with the following text:

```
MariaDB [organization]> SELECT *  
-> FROM employees  
-> WHERE department = 'Marketing' AND office LIKE 'East%';
```

The results are displayed in a table with 5 columns: employee_id, device_id, username, department, and office. The table is enclosed in a box with dashed lines.

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

Retrieve employees in Finance or Sales

The machines for employees in the Finance and Sales departments also need to be updated. Since a different security update is needed, I have to get information on employees only from these two departments.

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

The query retrieves all staff members from either the Finance or Sales departments.

Initially, I pulled all the information from the employees table. To narrow down the results, I employed a **WHERE** clause using the **OR** operator to isolate employees in either of the specified departments. I opted for **OR** rather than **AND** to capture employees belonging to either Finance or Sales. The first condition, **department = 'Finance'**, filters for those in the Finance sector, while the second condition, **department = 'Sales'**, does the same for those in Sales.

```
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
1044	s429t157u159	tbarnes	Finance	West-415
1045	t567u844v434	pwashing	Finance	East-115
1046	u429v921w138	daquino	Finance	West-280
1047	v109w587x644	cward	Finance	West-373

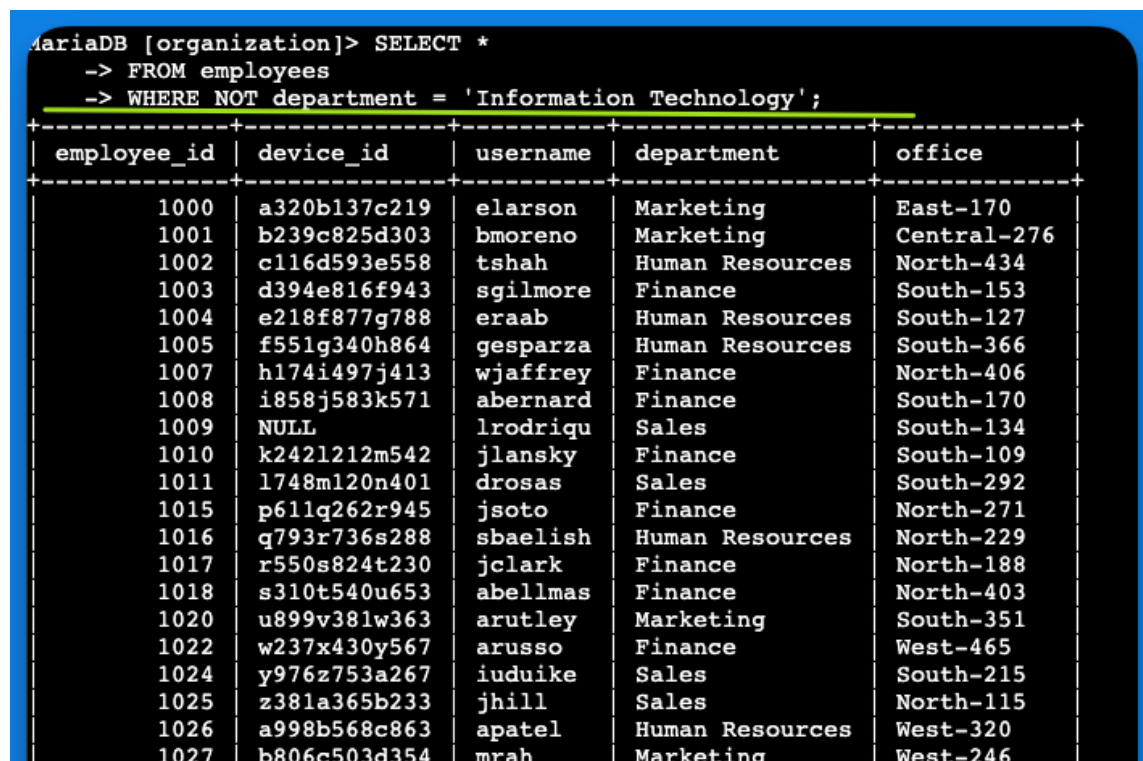
Retrieve all employees not in IT

My team needs to make one more security update on employees who are not in the Information Technology department. To make the update, I first have to get information on these employees.

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

The query returns all employees not in the Information Technology department.

1. First, I started by selecting all data from the employees table.
2. Then, I used a WHERE clause with NOT to filter for employees
3. NOT in this department.



The screenshot shows a terminal window with the following content:

```
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
1026	a998b568c863	apatel	Human Resources	West-320
1027	b806c503d354	mrah	Marketing	West-246

Summary

I utilized SQL query filters to extract targeted data on login activities and staff computers from two separate tables: `log_in_attempts` and `employees`. I employed a range of operators—**AND**, **OR**, and **NOT**—to fine-tune the information relevant to each task. Additionally, I made use of the **LIKE** function and the percentage sign (%) wildcard to identify specific patterns.