Apply filters to SQL queries

Project description

In this project I am tasked with finding the failed login attempts that occurred after normal work hours. I will use SQL to filter the log retrieving only those that I need.

The purpose of this is to show how you can mitigate the data returned from a database to only those columns needed. This is vital for a security analyst as being able to efficiently filter through logs will allow me to complete my tasks quickly and effectively by retrieving reputable data and translating the results easily.

I will use SQL to:

- Filter for login attempts that occurred after hours
- Filter for login attempts on specific dates
- Filter for login attempts from specific locations
- Filter for information on employees in specific departments
- Filter for information on employees not in a specific department

Retrieve after hours failed login attempts

To begin with, I specified specific rows I wanted to return as i do not want to return redundant information that is not needed for this task , I chose to filter for username - so I can see which user account failed, login time, - so I can specify after hours attempts and success - as this will return the success rate of the logins.

I then had to filter based on 2 conditions. First, I wanted the `login_time` column to only show me logs after 6pm as this is the standard logoff time for most employees.

Secondly, I had to specify the `success` rate of login attempts, the 0 indicates a failure to login and a 1 indicated a successfully login.

```
 [organization]> SELECT username, login_time, success
FROM log_in_attempts
WHERE login_time > '18:00' AND success = 0;
```

This had returned the following:

username	login_time	success
apatel	20:27:27	0
pwashing	19:28:50	0
tshah	18:56:36	0
aestrada	19:28:12	0
drosas	21:02:04	0
cgriffin	23:04:05	0
cjackson	22:07:07	0
wjaffrey	19:55:15	0
abernard	23:38:46	0
apatel	22:38:31	0
ivelasco	22:36:36	0
asundara	18:38:07	0
bisles	20:25:57	0
aestrada	22:00:26	0
abellmas	21:20:51	0
bisles	20:03:55	0
cgriffin	22:18:42	0
jclark	20:49:00	0
yappiah	19:34:48	0

Retrieve login attempts on specific dates

My team is investigating a suspicious event that occurred on `'2022-05-09'`. I want to retrieve all login attempts that occurred on this day and the day before (`'2022-05-08'`).

```
 [organization]> SELECT *
FROM log_in_attempts
WHERE login_date BETWEEN '2022-05-08' AND '2022-05-09';
```

I had chosen to return all database columns as I want to know all the events that occured during this time frame.

To specify this I used the BETWEEN operator and the AND operator to filter between two dates.

The WHERE operator always comes before as this indicates the column I want to apply the filter to

This has returned a large list, partial below.

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	nmason	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	mrah	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

Retrieve login attempts outside of Mexico

I had not specified the column so I used the asterisk wildcard which returns all rows, I then specified the database to be used.

Lastly, to filter for all login attempts apart from Mexico I had used the NOT and LIKE operator with the percentage sign wildcard.

The NOT operator will filter out the specified condition after it,

The LIKE operator will search for a matching pattern of the specified string.

The issue was there are countries listed as ‘MEX’ and ‘MEXICO’ so to make the query as efficient as possible I used the percent wildcard which tells the database to match the pattern of anything beginning with ‘MEX’.

```
; [organization]> SELECT *
FROM log_in_attempts
WHERE NOT country LIKE 'MEX%';
```

Retrieve employees in Marketing

In the following query I had to search for all employee information who work in the east building marketing department.

I again specified to return all columns in the employees table.

I then used the WHERE operator to specify the exact columns I want to filter against, specific the marketing department and used the LIKE operator with the percent wildcard to return all the devices in the east office building.

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

The results look like this:

```
+-------------+--------------+----------+------------+----------+
| 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 |
+-------------+--------------+----------+------------+----------+
```

Retrieve employees in Finance or Sales

In this query I had used the OR operator to filter for two specific departments.

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

employee_id	device_id	username	department	office
1009	NULL	lrodriqu	Sales	South-134
1011	l748m120n401	drosas	Sales	South-292
1024	y976z753a267	iuduike	Sales	South-215
1025	z381a365b233	jhill	Sales	North-115
1035	j236k3031245	bisles	Sales	South-171
1039	n253o917p623	cjackson	Sales	East-378
1041	p929q222r778	cgriffin	Sales	North-208
1057	f370g535h632	mscott	Sales	South-270
1063	l686m140n569	lpope	Sales	East-226
1066	o678p794q957	ttyrell	Sales	Central-444
1071	t244u829v723	zdutchma	Sales	West-348
1072	u905v920w694	esmith	Sales	East-421
1078	a667b270c984	sharley	Sales	North-418
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
1107	d168e758f876	akajwara	Sales	North-471
1109	f229g533h679	nlocklea	Sales	East-196
1110	g567h376i314	pchaudhu	Sales	Central-428
1111	h835i179j862	jlee	Sales	West-309
1116	m272n572o874	nzhao	Sales	South-100
1117	n683o758p820	dahmad	Sales	West-405
1118	o305p208q337	jpark3	Sales	South-329
1119	p164q780r999	omubarak	Sales	West-409
1121	r628s557t397	mrojas	Sales	East-288
1130	a317b635c465	tsnow	Sales	Central-451
1169	NULL	mmitchel	Sales	Central-250
1176	u849v569w521	nliu	Sales	West-220
1185	d790e839f461	revens	Sales	North-330
1186	e281f433g404	sacosta	Sales	North-460

Summary

In Summary, I was able to retrieve Logs using SQL to filter for information I needed from the database by applying AND, OR and NOT operators to SQL queries as well as utilizing wildcards to return accurate information.