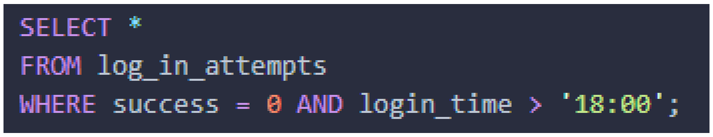
Apply filters to SQL queries

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.

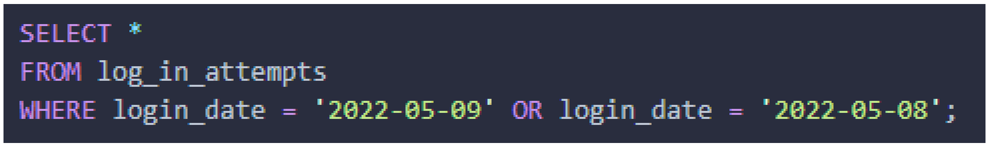
Retrieve after hours failed login attempts



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

I recently discovered a potential security incident that occurred after business hours. To investigate, I ran a filter on the “success” where it is set to 0 (1 is a successful login attempt) and the login time is greater than (aka after) 6:00 which is when the office is closed. This returns all failed login attempts that occurred after hours.

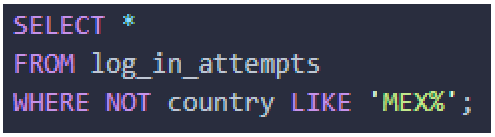
Retrieve login attempts on specific dates



```sql
SELECT *
FROM log_in_attempts
WHERE login_date = '2022-05-09' OR login_date = '2022-05-08';
```

There was a suspicious event on 2022-05-09. To investigate, I got all of the login attempts on that date and the one before using the OR operator.

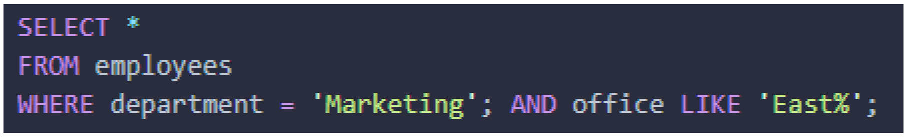
Retrieve login attempts outside of Mexico



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

The security team determined that the suspicious activity did not originate from Mexico. Using filtering I removed entries that contains “MEX" or anything that starts with “MEX” using the % wildcard.

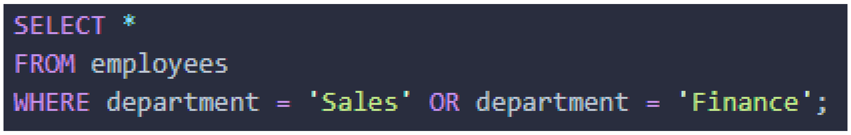
Retrieve employees in Marketing



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

My team wants to perform security updates on specific employee machines in the Marketing department. To get information on employee machines I ran a query that returned all employees who are in the marketing department in the East building. I used a wildcard here because some entries in the office column include the room numbers like “East-103”.

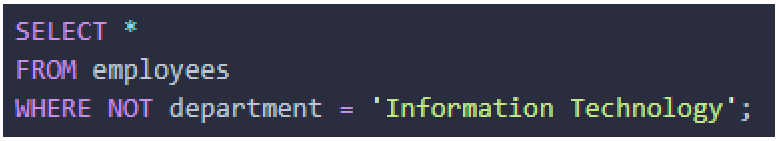
Retrieve employees in Finance or Sales



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

My team now needs to perform a different security update on machines for employees in the Sales and Finance departments. To get a report on all employees in Sales or Finance, I ran the above query.

Retrieve all employees not in IT



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

My team needs to make one more update to employee machines for those who are not in I.T. because I.T. has already received the update. Using the NOT operator this returns all employees who are not in I.T.

Summary

I applied filters to SQL queries to get specific information on login attempts and employee machines. I used two different tables, “log_in_attempts” and “employees”. I used the AND, OR, and NOT operators to filter for the specific information needed for each task. I also used LIKE and the percentage sign (%) wildcard to filter for patterns.