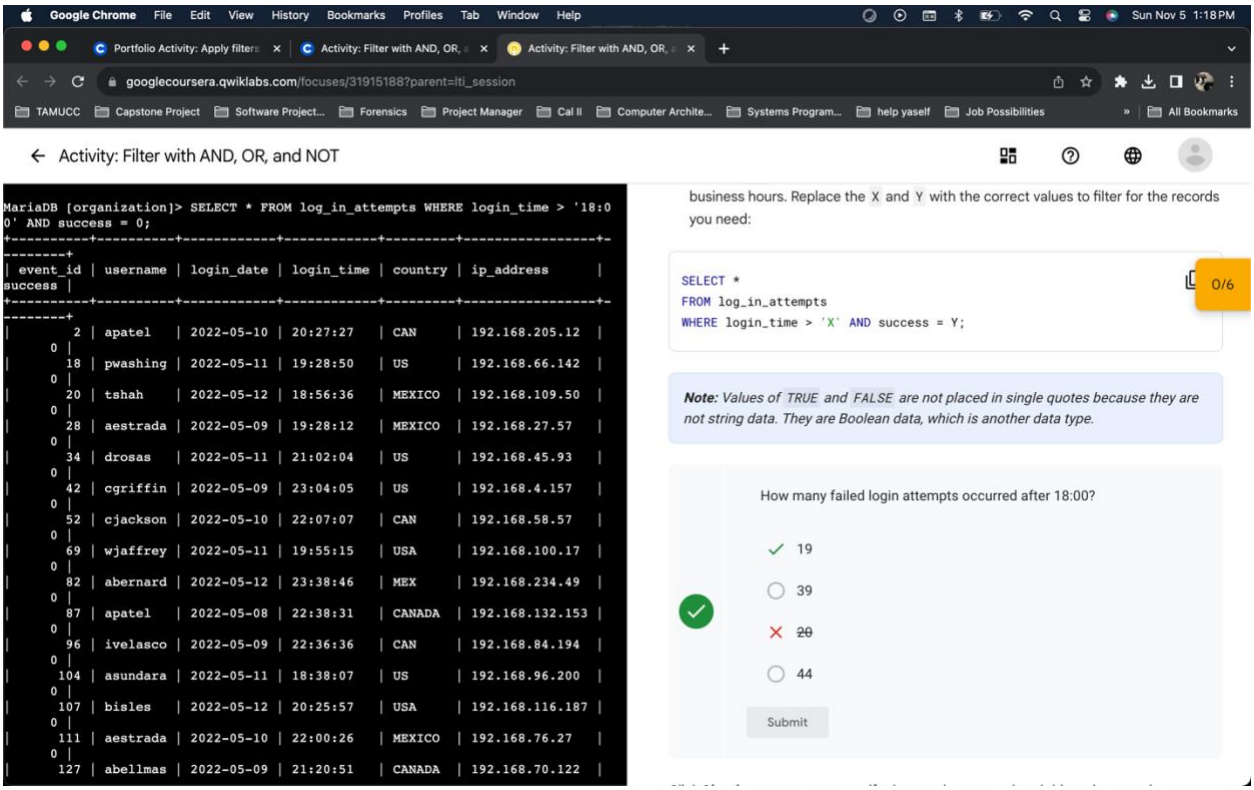
Apply filters to SQL queries

Project description

This project simulates what it's like being a security analyst by performing complex queries in SQL. We're mostly going to retrieve information from the employees database. To do this, we need to write queries that have multiple conditions and operators. For operators, we'll focus on 'NOT', 'OR', and 'AND'.

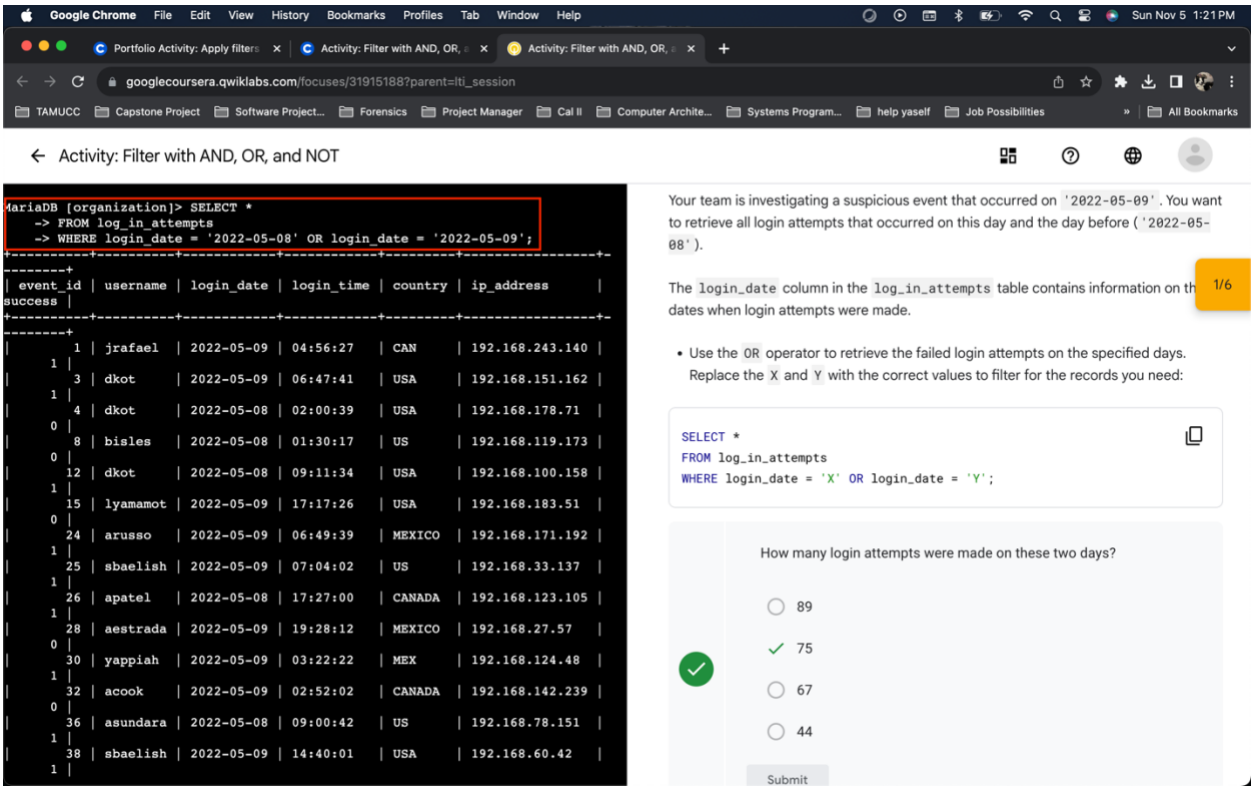
Retrieve after hours failed login attempts

So, I 'discovered' a potential security incident that occurred after business hours. To investigate, we need a query to get to the log_in_attempts table and review after hours login activity. We'll need to specifically look for failed login attempts



Retrieve login attempts on specific dates

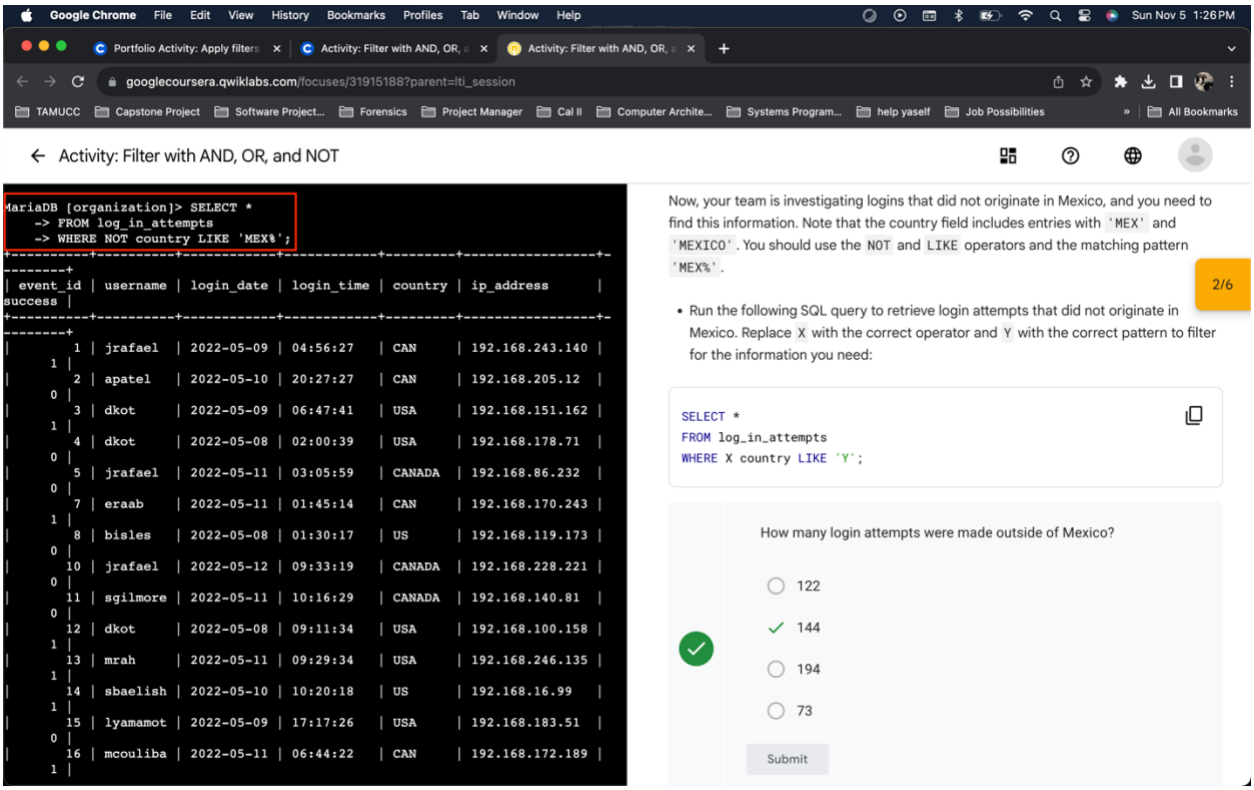
So we noticed that a suspicious event occurred on 2022-05-09. To look into this further, we need to set up a query that will review all login attempts that happened on that day and the day before.
First you can see that we're selecting all columns with '*' just to get info like the employee name and their id, along with the date. Then, we'll use the 'OR' operator to act as a filter to remove any other date that's not 2022-05-09 or 2022-05-08.



Retrieve login attempts outside of Mexico

In this part of the project, we've become aware of suspicious activity with login attempts. But we know for sure that the activity didn't originate from Mexico, so we can mark that off. To narrow the search, we need to look at login attempts outside of Mexico. We'll use a filter in a query that identifies all login attempts that occurred anywhere but Mexico. To do this, we'll use the 'NOT' operator, and the '%' to handle variations of the spelling of Mexico. It could either be 'MEX' or 'MEXICO'. We'll use it like this: 'MEX%', the percentage sign could either complete the full word Mexico, or it could just leave it as the

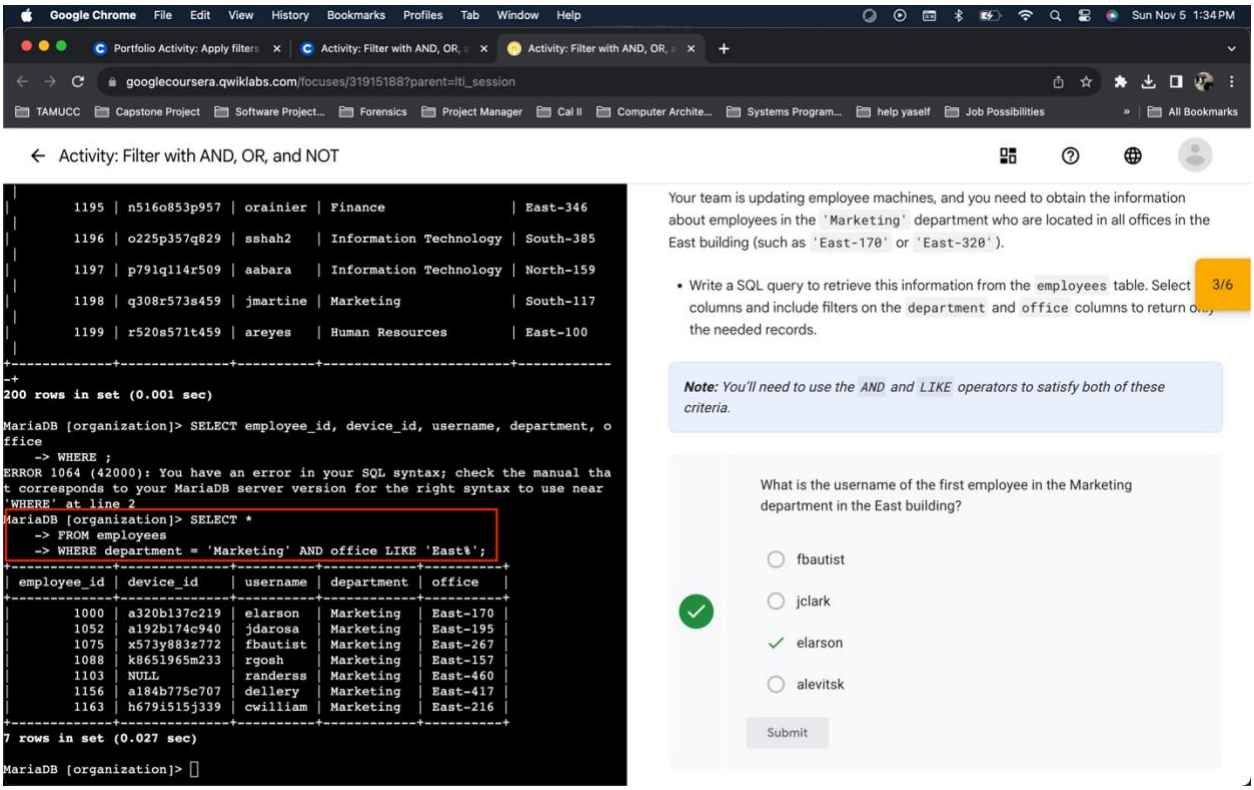
abbreviation and that will be just fine as well.



Retrieve employees in Marketing

So now we want to see what machines in the Marketing department need to be updated for better security. We'll need to create a query that identifies all employees in the Marketing department in all offices in the East building. This time we only want the Marketing column, and offices in the East wing. We'll use the LIKE keyword instead of the equals sign, and we'll use '%' to help with filtering for the East wing. Notice that for the 'WHERE' part, it's appropriate to use the '=' operator, but when we need to look for a certain set of characters, this is incorrect syntax. Instead we use the 'LIKE' operator paired

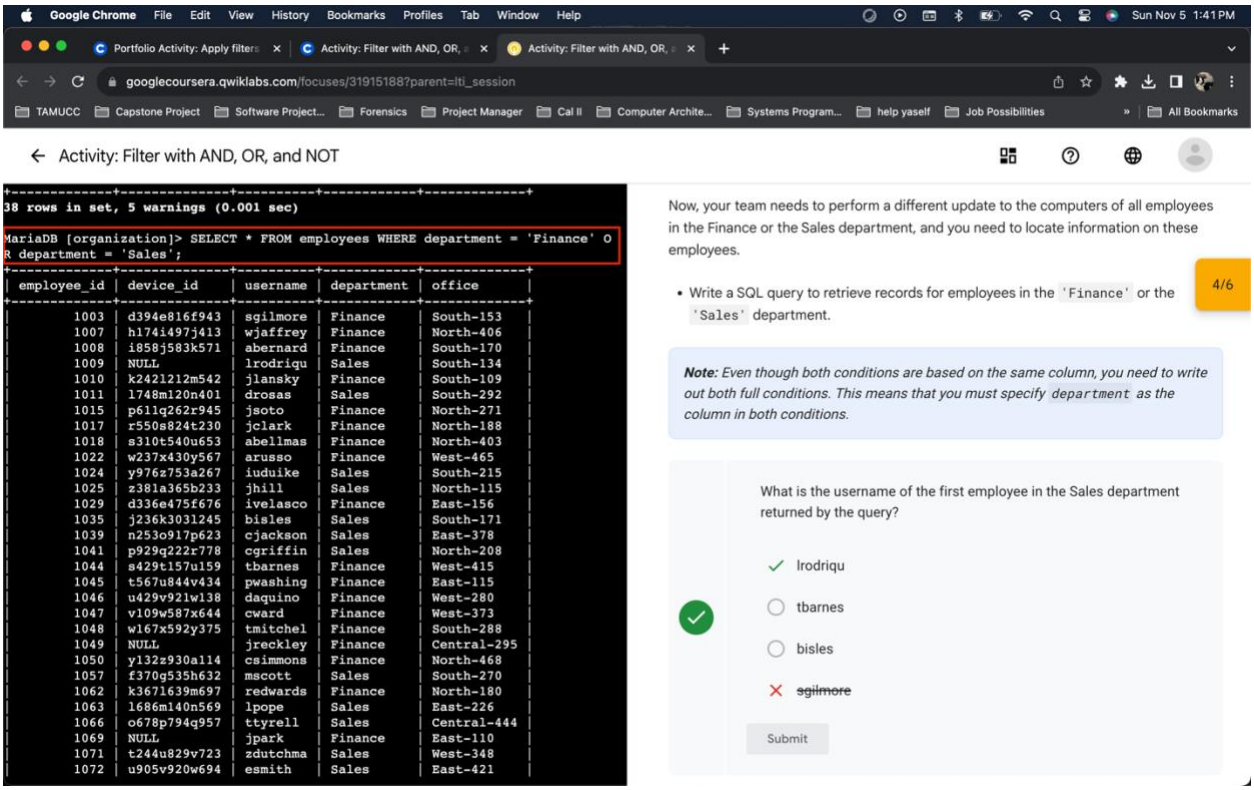
with the '%' sign.



Retrieve employees in Finance or Sales

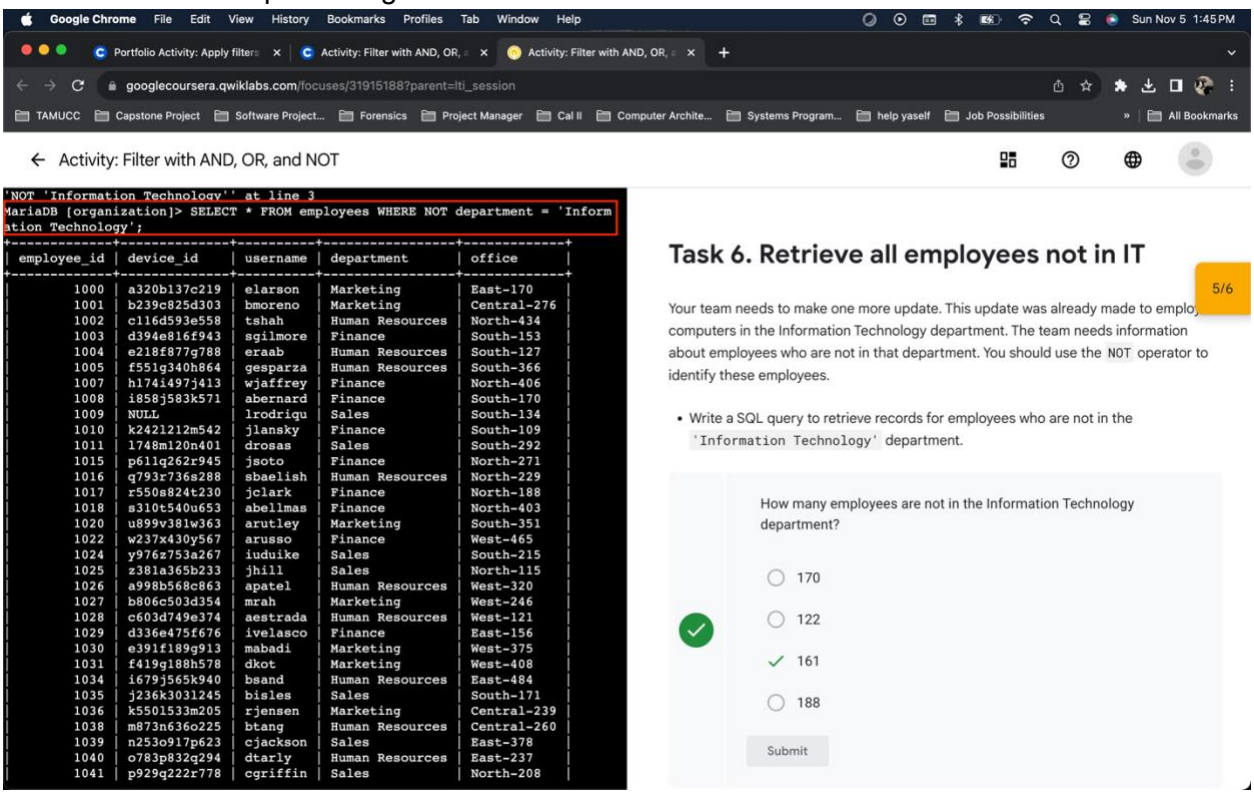
So now we'd like to see who needs the IT department to perform security updates on every employee's machine that's in either the Sales or Finance department. Simple enough. We'll go ahead and use the 'OR' operator to grab the employees' information from either the Sales or Finance department. Even though we're gathering data from one department, we need to make sure our conditions are specific enough. For the query to have correct syntax, we need write out both full conditions. We need to

specify the department for the two conditions.



Retrieve all employees not in IT

Now we need to look at providing security updates for every employee not in the IT department. Let's use the NOT operator again.



Summary

So now I can say I have practical experience using SQL to run queries to retrieve information from a database. Along with using filters using the AND, OR, and NOT operator to curate the data.