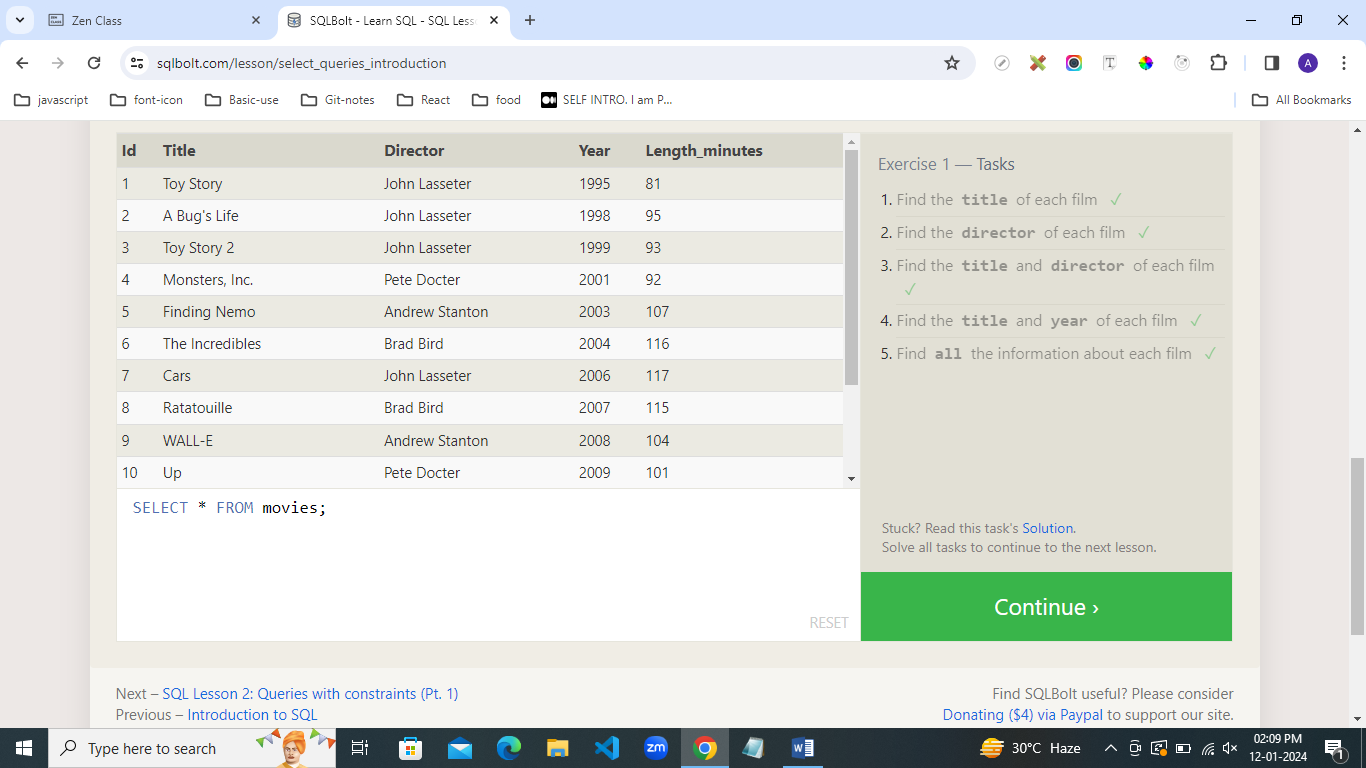
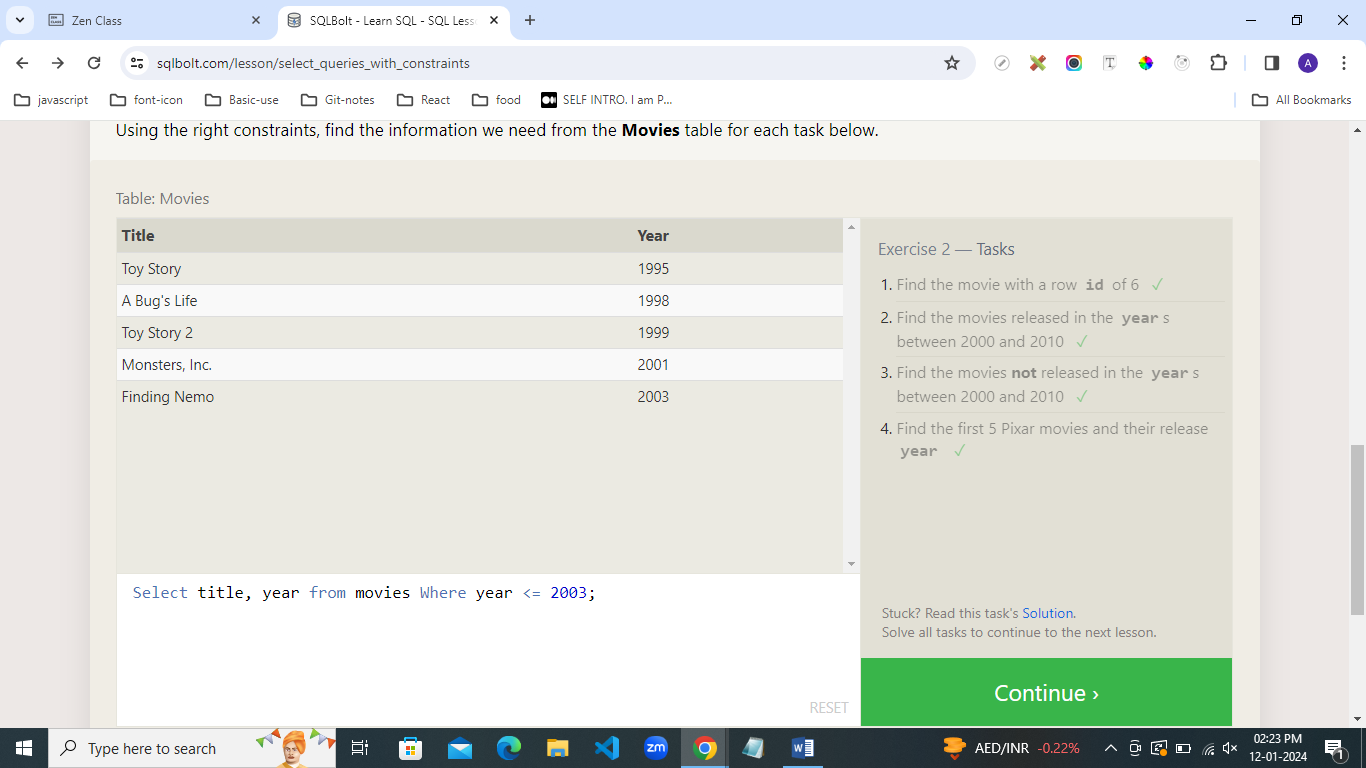
My Sql Task -1

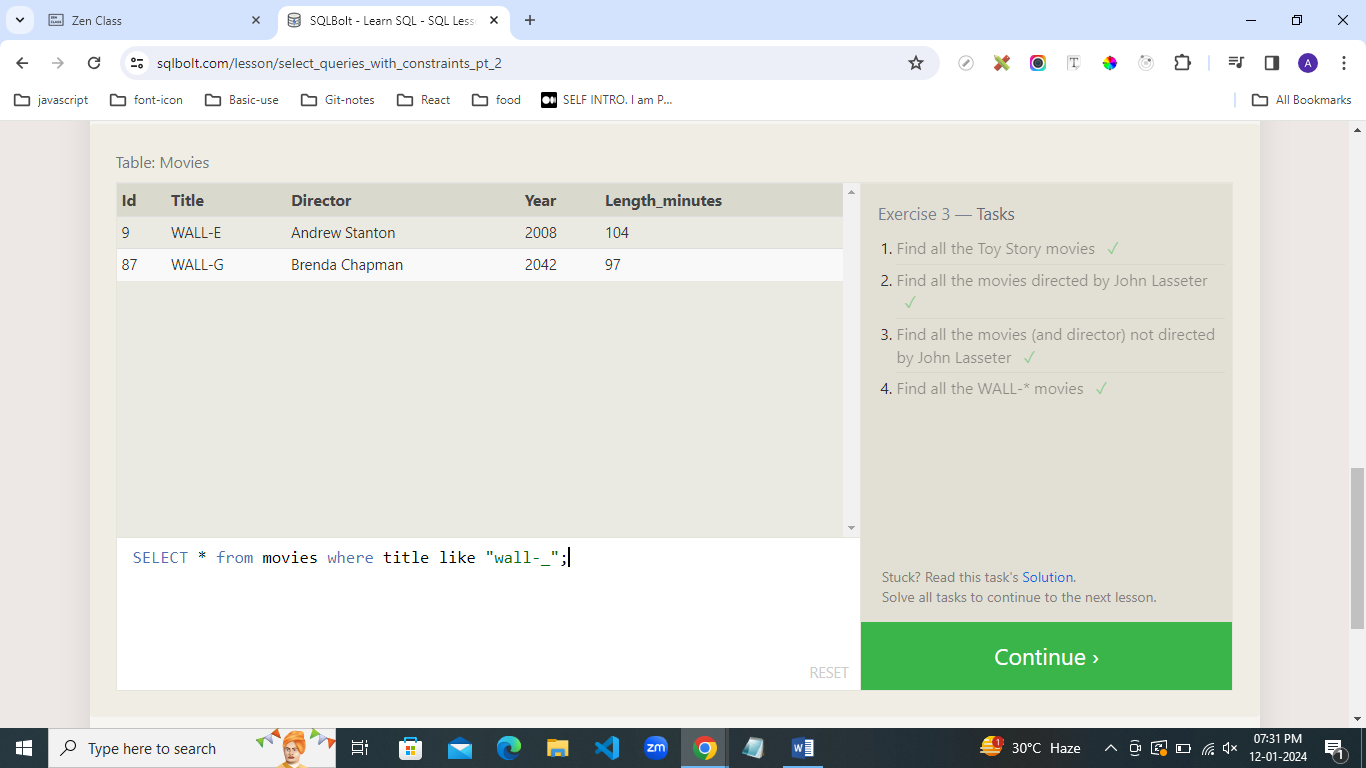
1. We will be using a database with data about some of Pixar's classic movies for most of our exercises. This first exercise will only involve the **Movies** table, and the default query below currently shows all the properties of each movie. To continue onto the next lesson, alter the query to find the exact information we need for each task



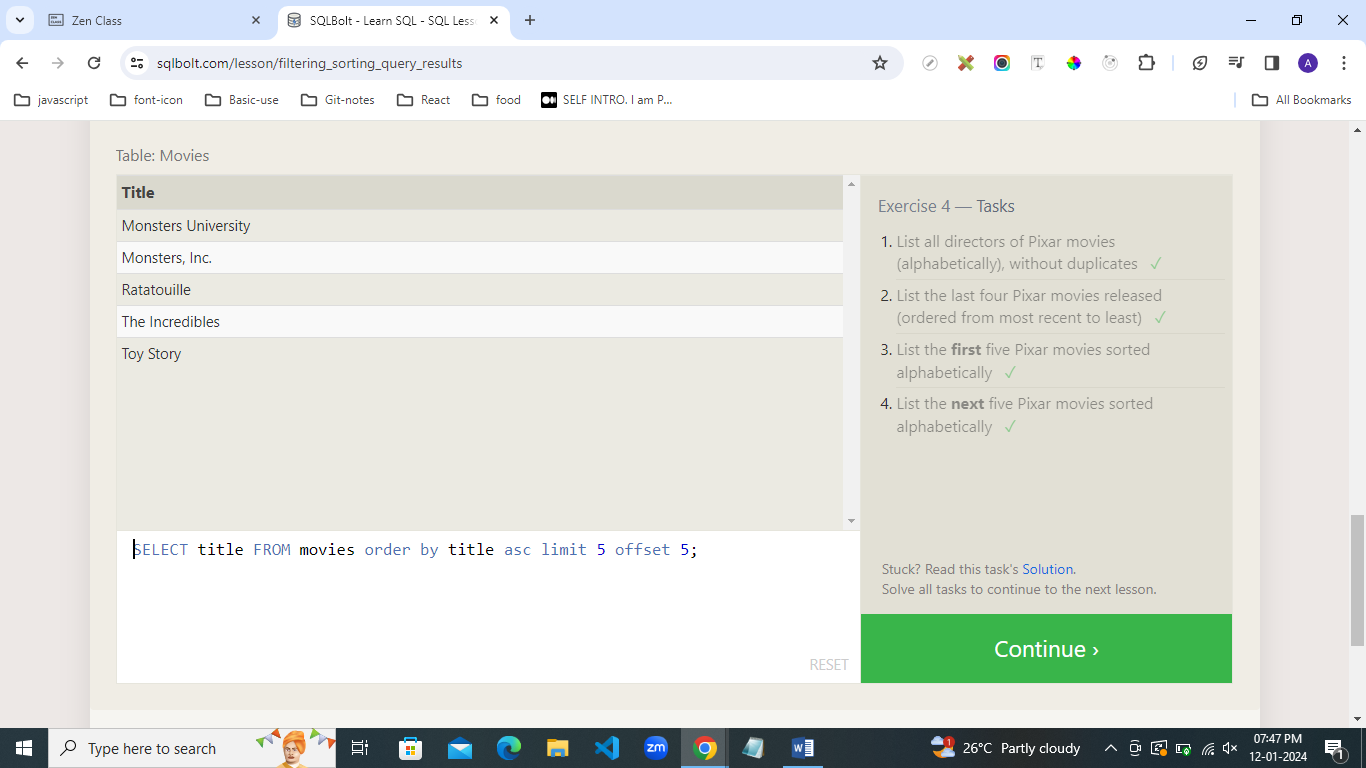
1. Using the right constraints, find the information we need from the **Movies** table for each task below.



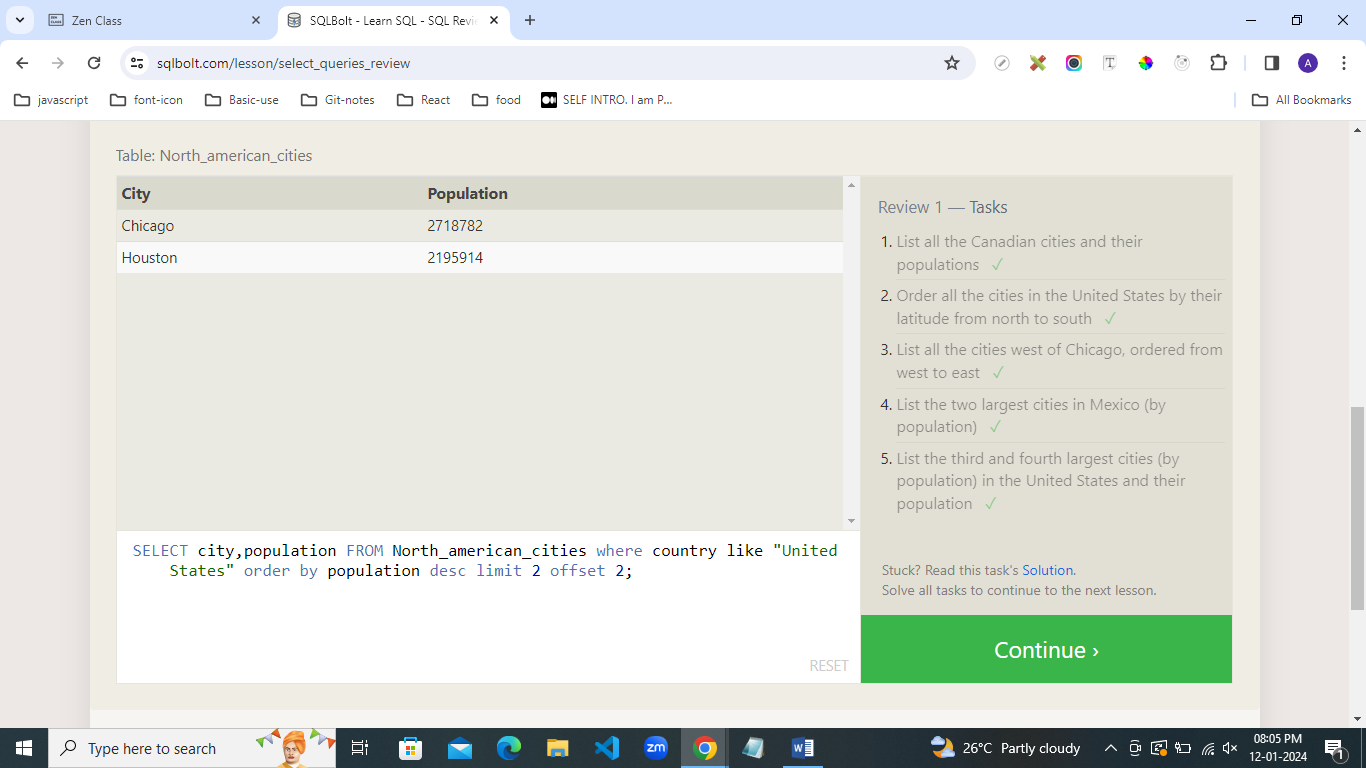
1. Here's the definition of a query with a WHERE clause again, go ahead and try and write some queries with the operators above to limit the results to the information we need in the tasks below.



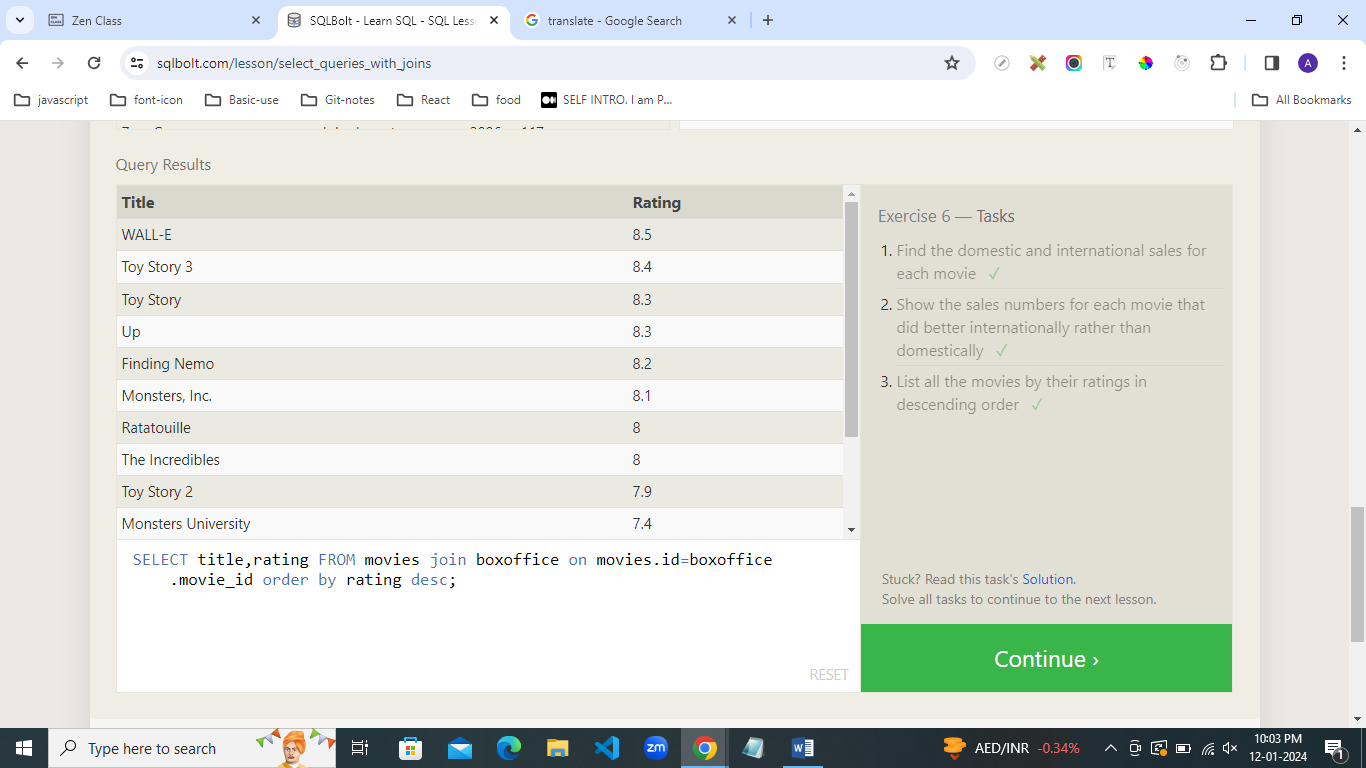
1. There are a few concepts in this lesson, but all are pretty straight-forward to apply. To spice things up, we've gone and scrambled the Movies table for you in the exercise to better mimic what kind of data you might see in real life. Try and use the necessary keywords and clauses introduced above in your queries.



1. Try and write some queries to find the information requested in the tasks you know. You may have to use a different combination of clauses in your query for each task. Once you're done, continue onto the next lesson to learn about queries that span multiple tables.

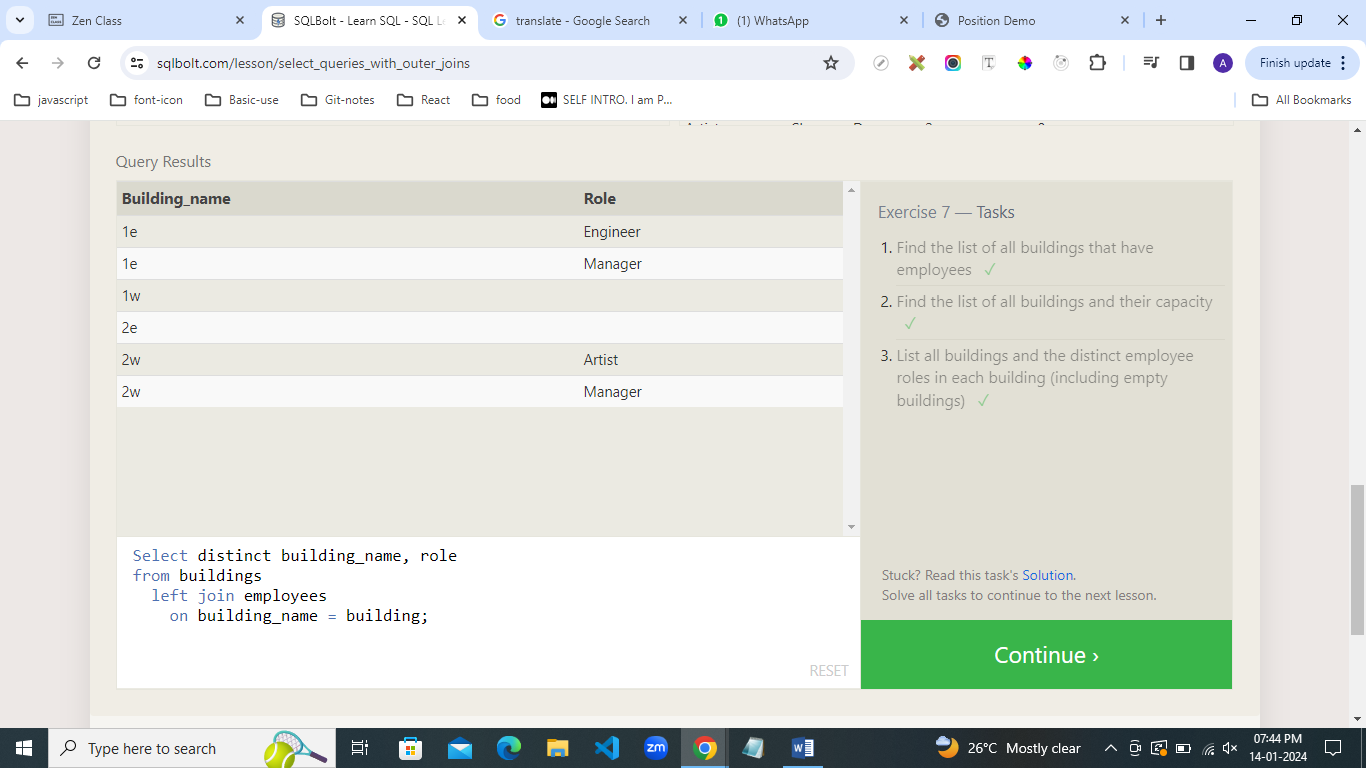


1. We've added a new table to the Pixar database so that you can try practicing some joins. The **BoxOffice** table stores information about the ratings and sales of each particular Pixar movie, and the **Movie\_id** column in that table corresponds with the **Id** column in the **Movies** table 1-to-1. Try and solve the tasks below using the **INNER JOIN** introduced above

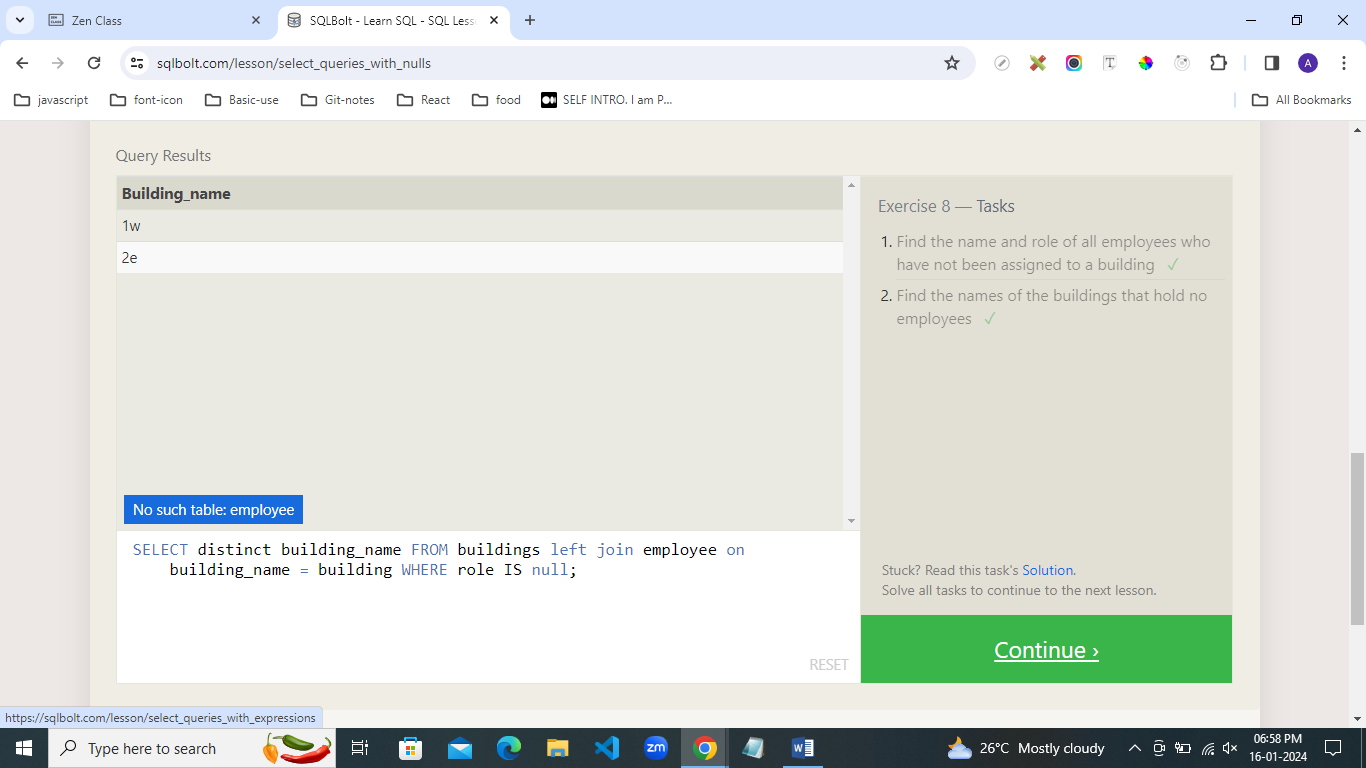


7. In this exercise, you are going to be working with a new table which stores fictional data about **Employees** in the film studio and their assigned office **Buildings**. Some of the buildings are new, so they don't have any employees in them yet, but we need to find some information about them regardless.

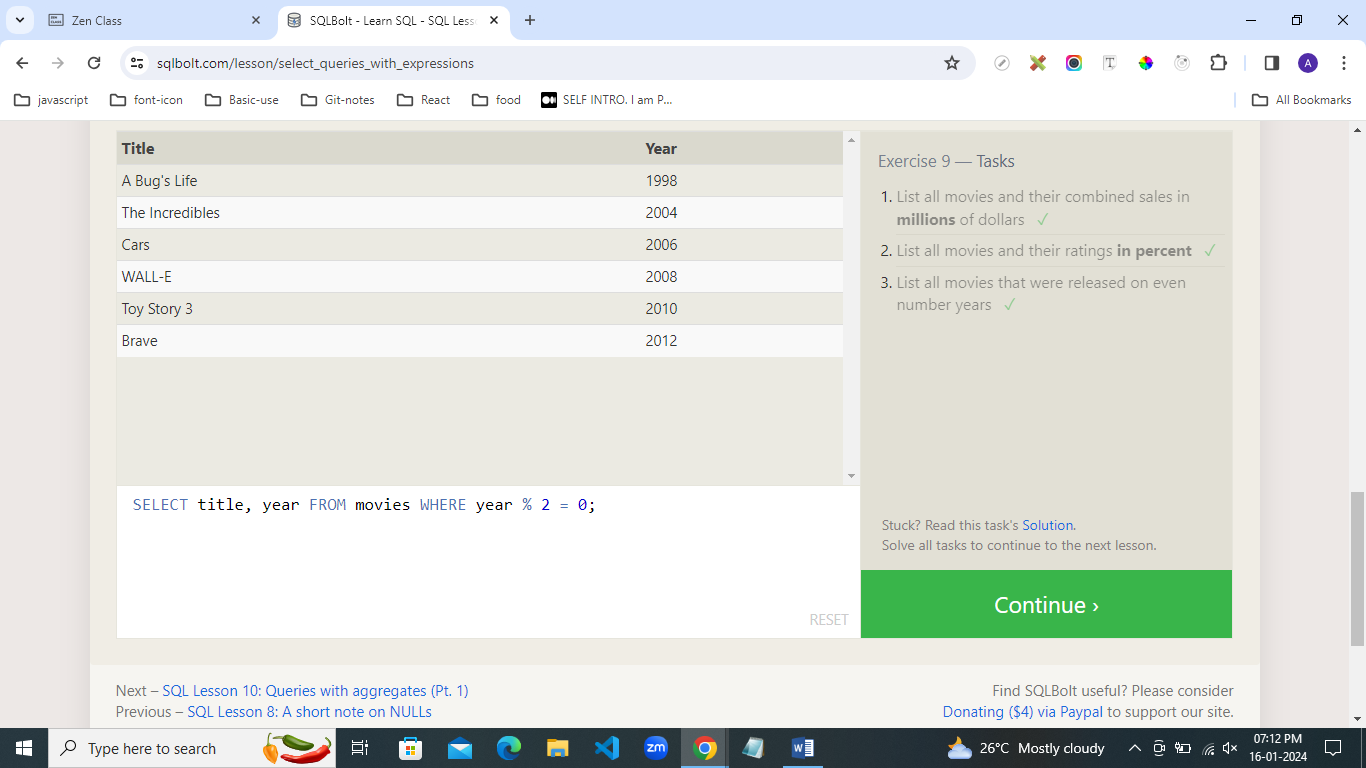
Since our browser SQL database is somewhat limited, only the **LEFT JOIN** is supported in the exercise below.



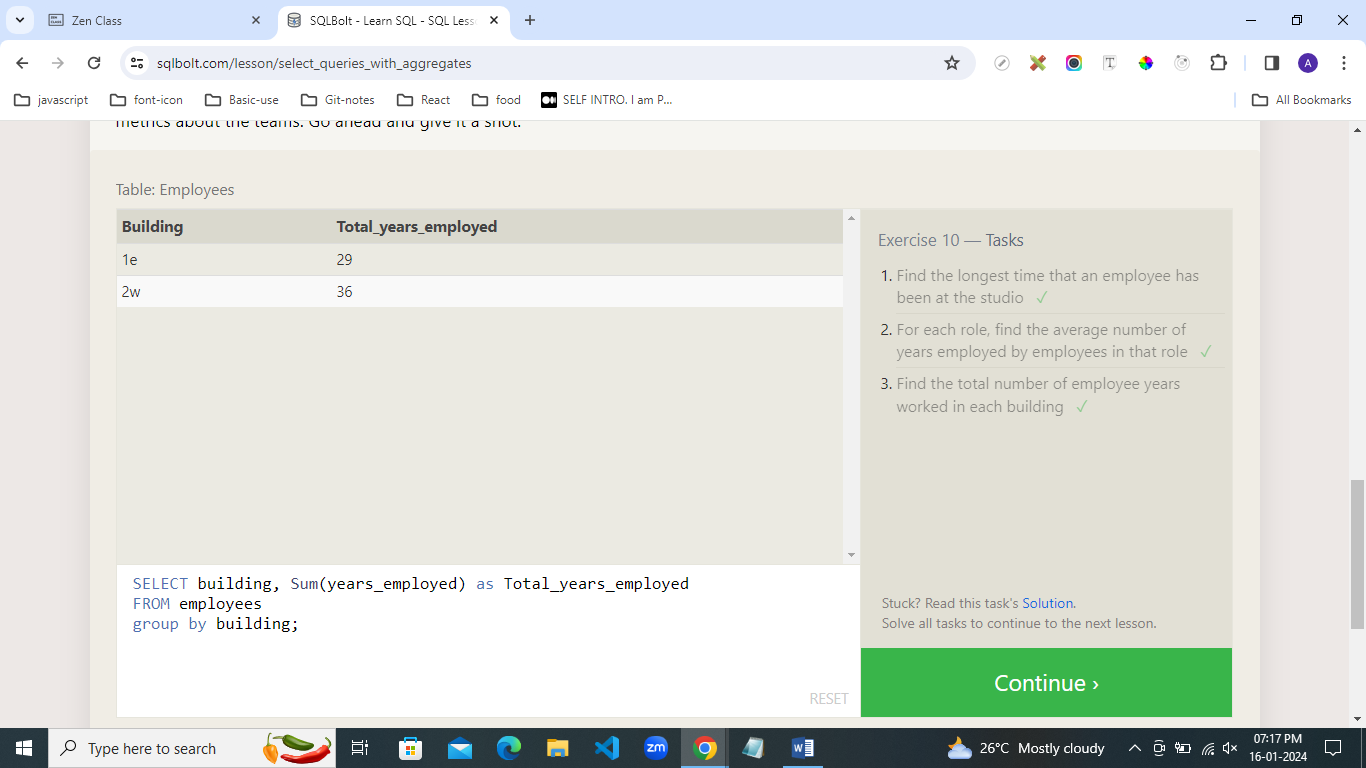
8. This exercise will be a sort of review of the last few lessons. We're using the same **Employees** and **Buildings** table from the last lesson, but we've hired a few more people, who haven't yet been assigned a building.



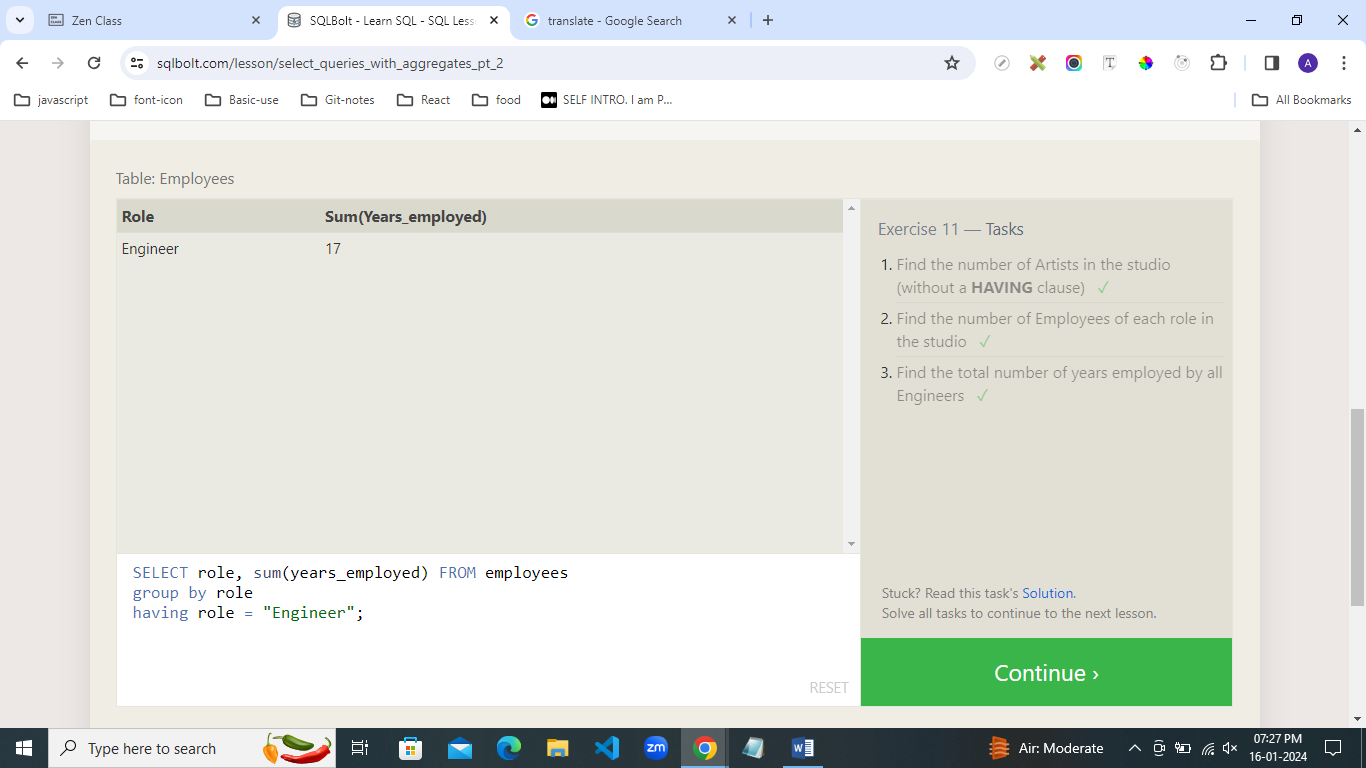
9. You are going to have to use expressions to transform the **BoxOffice** data into something easier to understand for the tasks below.



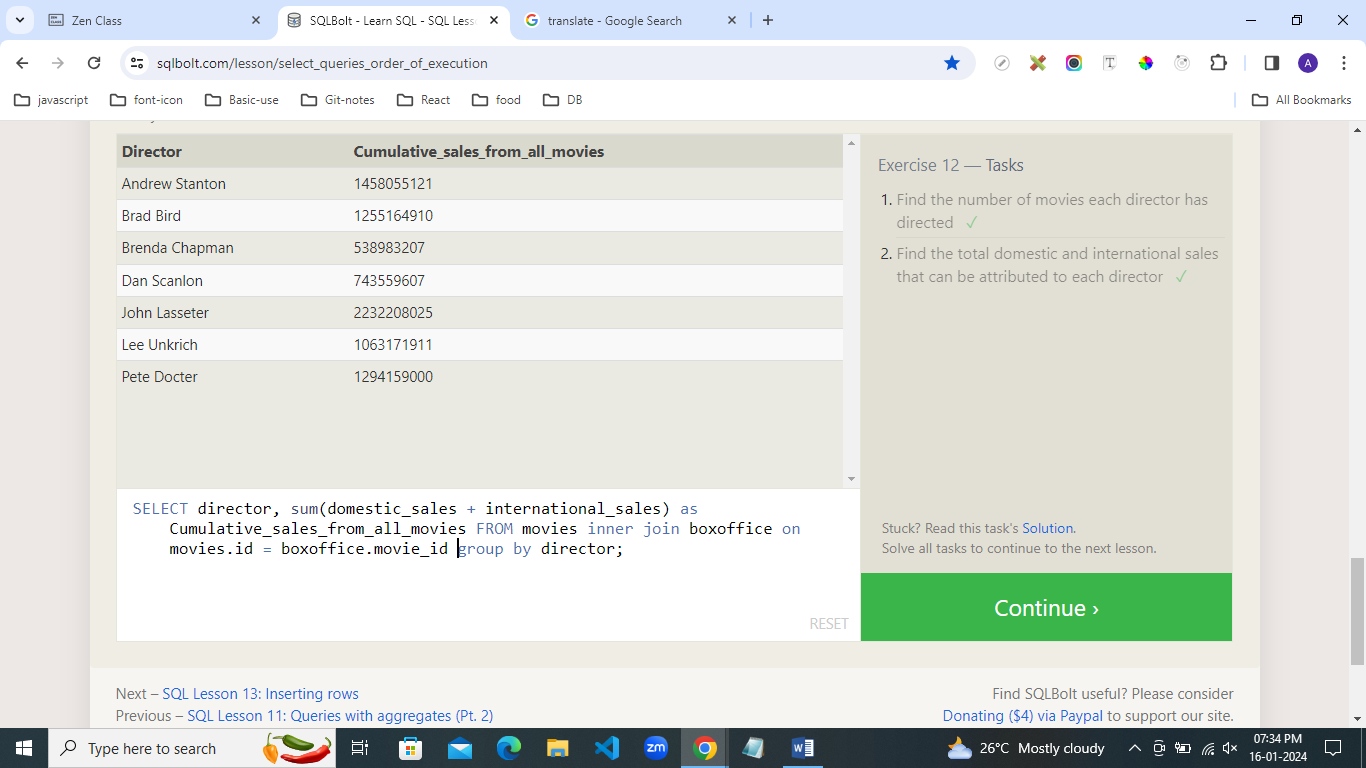
10. For this exercise, we are going to work with our **Employees** table. Notice how the rows in this table have shared data, which will give us an opportunity to use aggregate functions to summarize some high-level metrics about the teams. Go ahead and give it a shot.



11. For this exercise, you are going to dive deeper into **Employee** data at the film studio. Think about the different clauses you want to apply for each task.

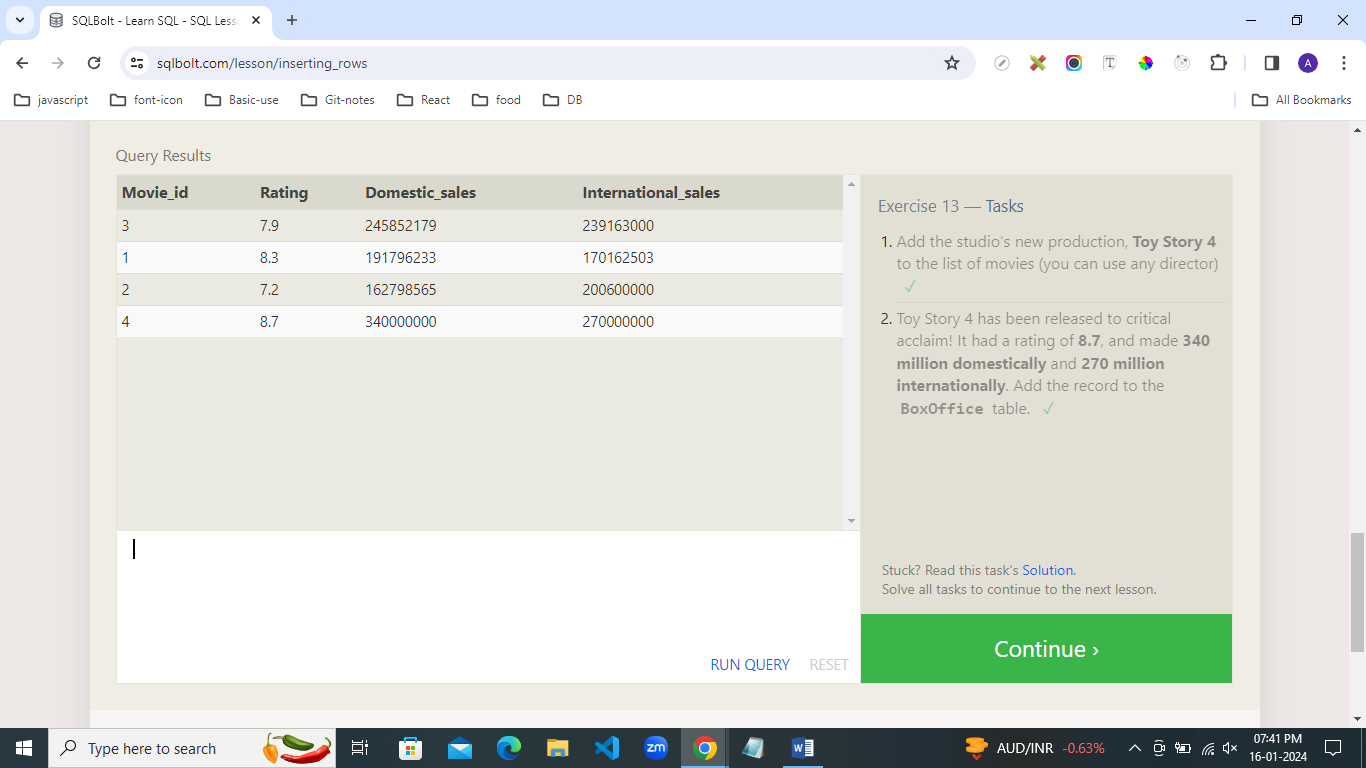


12. Here ends our lessons on **SELECT** queries, congrats of making it this far! This exercise will try and test your understanding of queries, so don't be discouraged if you find them challenging. Just try your best.

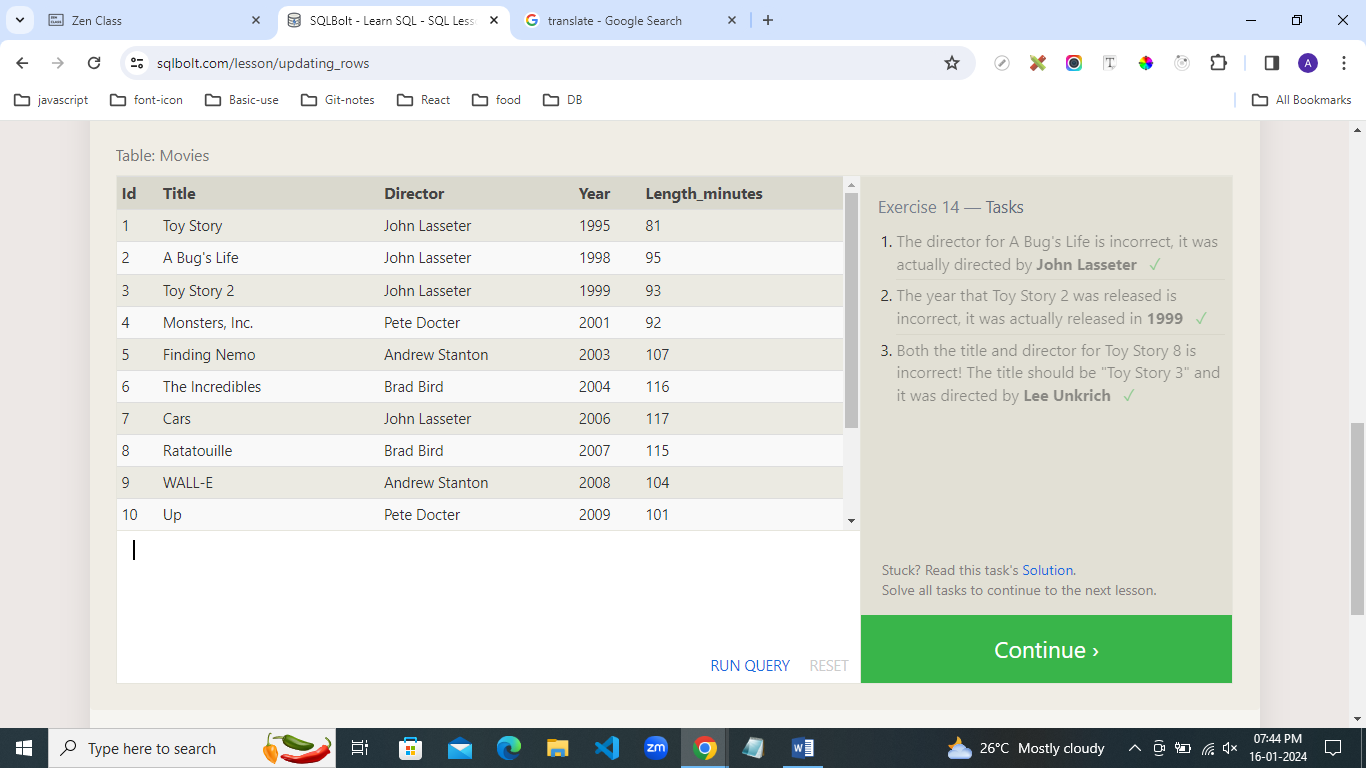


13. In this exercise, we are going to play studio executive and add a few movies to the **Movies** to our portfolio. In this table, the **Id** is an auto-incrementing integer, so you can try inserting a row with only the other columns defined.

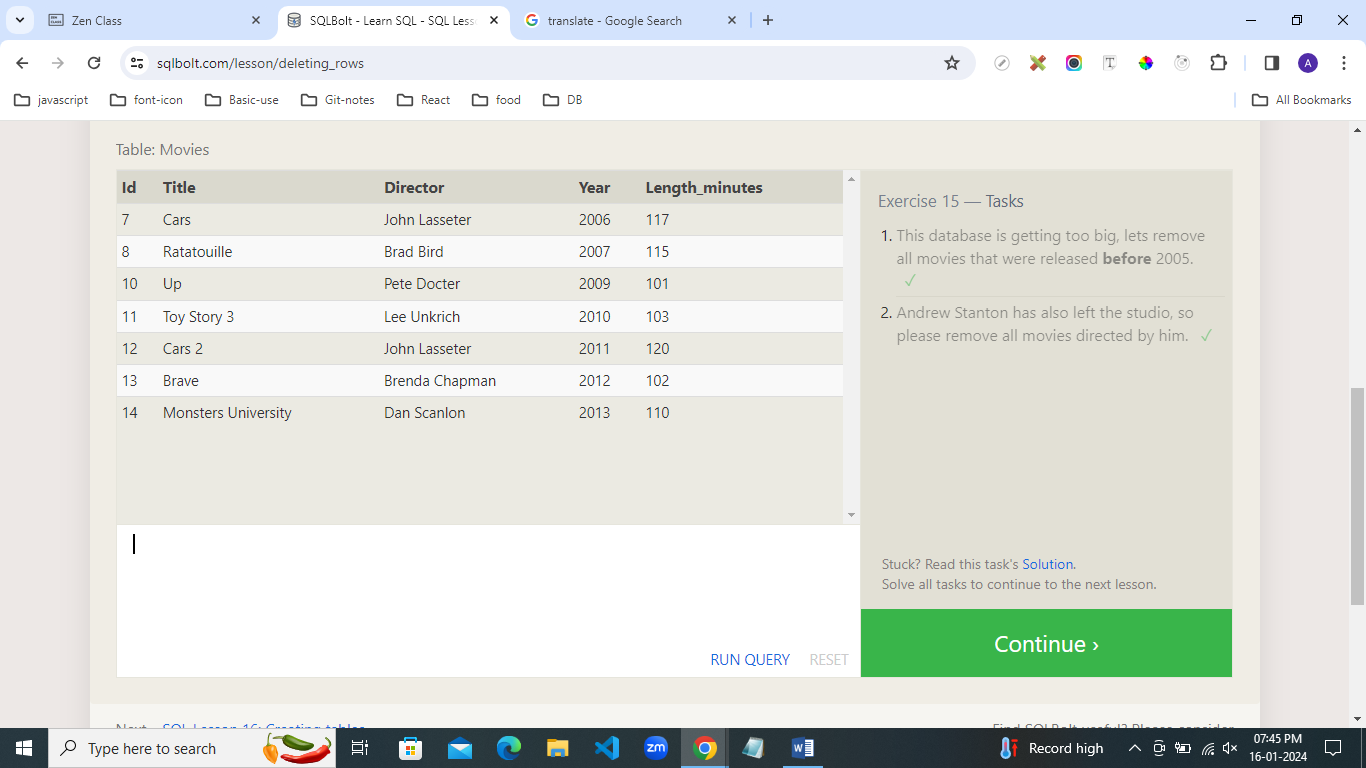
Since the following lessons will modify the database, you'll have to manually run each query once they are ready to go.



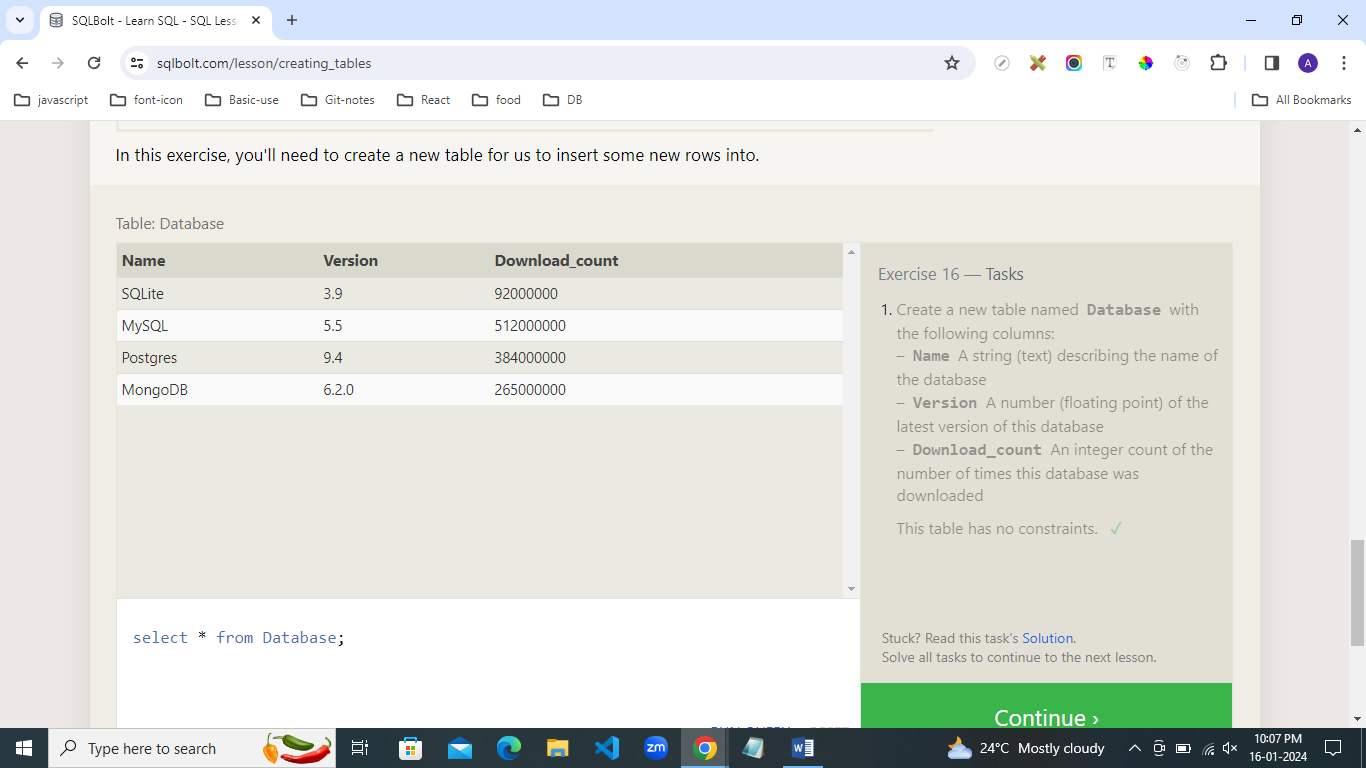
14. It looks like some of the information in our **Movies** database might be incorrect, so go ahead and fix them through the exercises below.



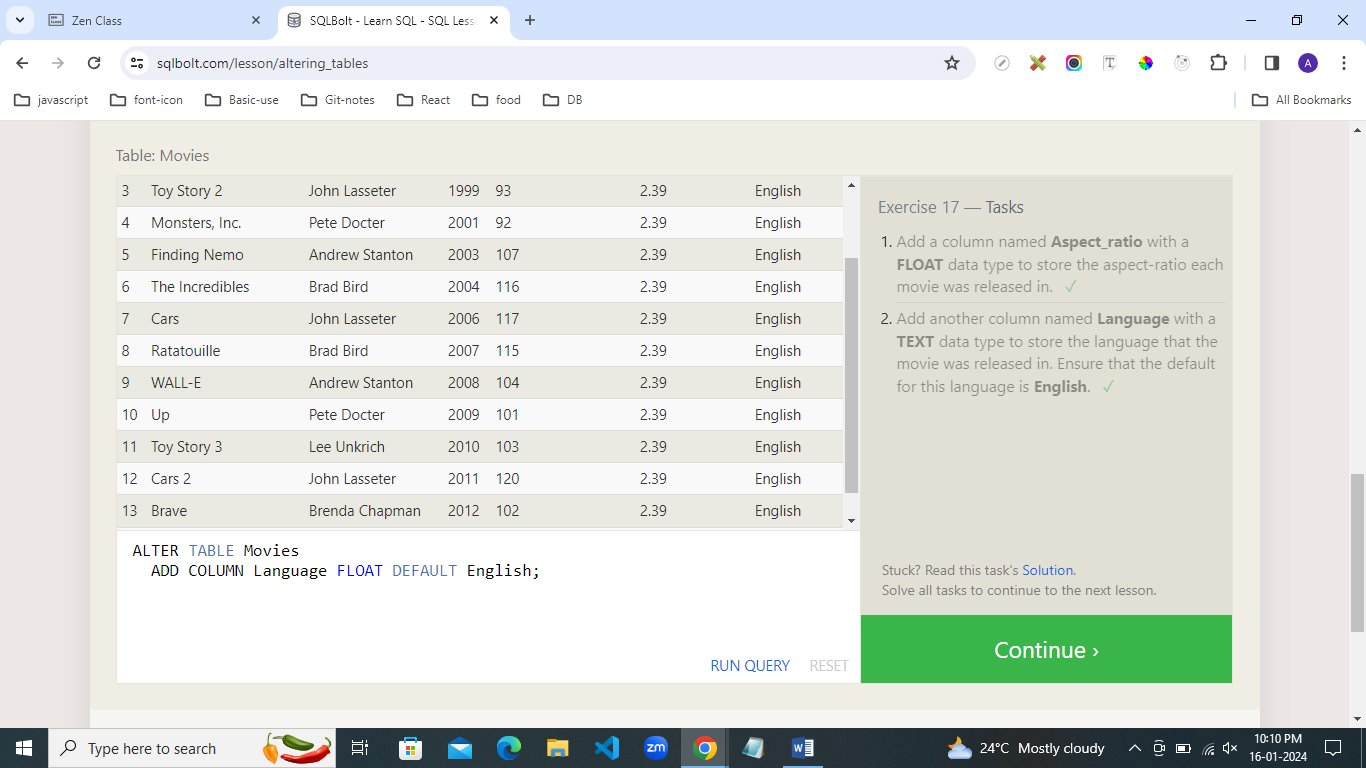
15. The database needs to be cleaned up a little bit, so try and delete a few rows in the tasks below.



16. In this exercise, you'll need to create a new table for us to insert some new rows into.



17. Our exercises use an implementation that only support adding new columns, so give that a try below.



18. We've reached the end of our exercises, so lets clean up by removing all the tables we've worked with.

