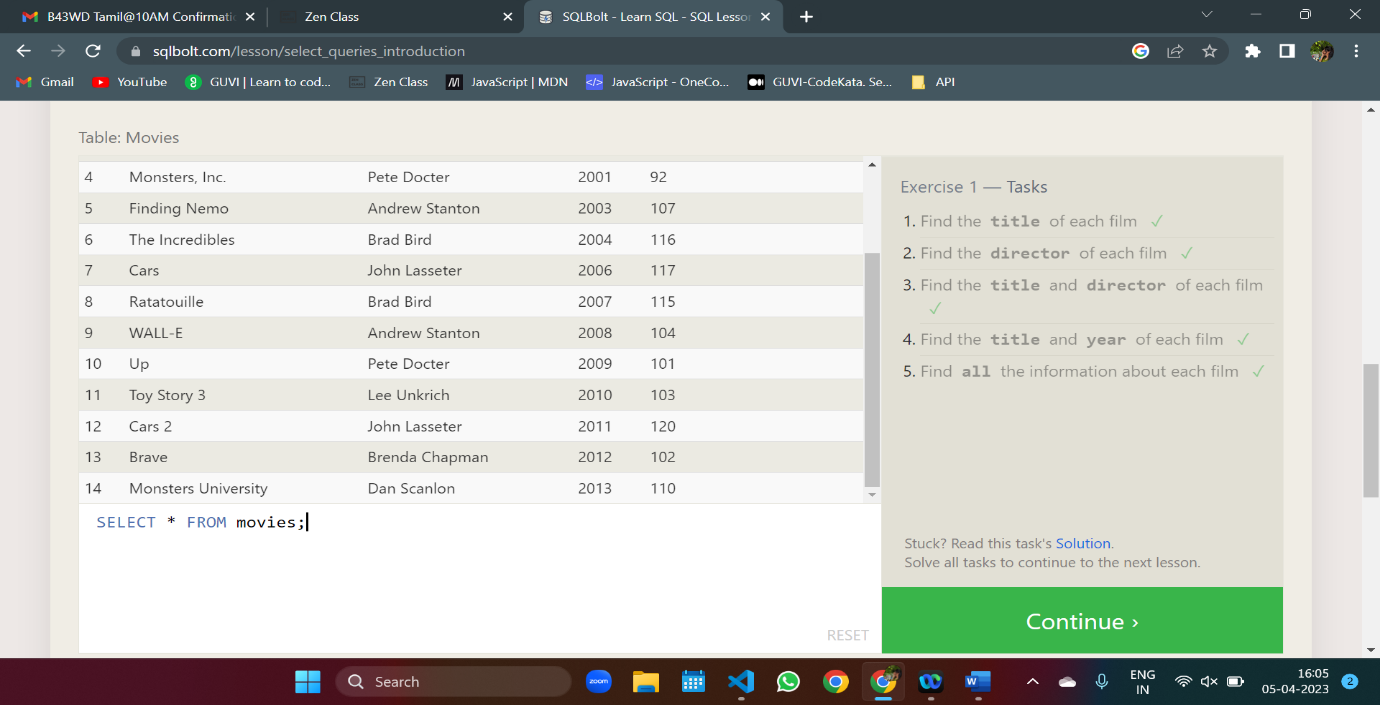
**SQL Lesson 1: SELECT queries 101**



SELECT title FROM movies;

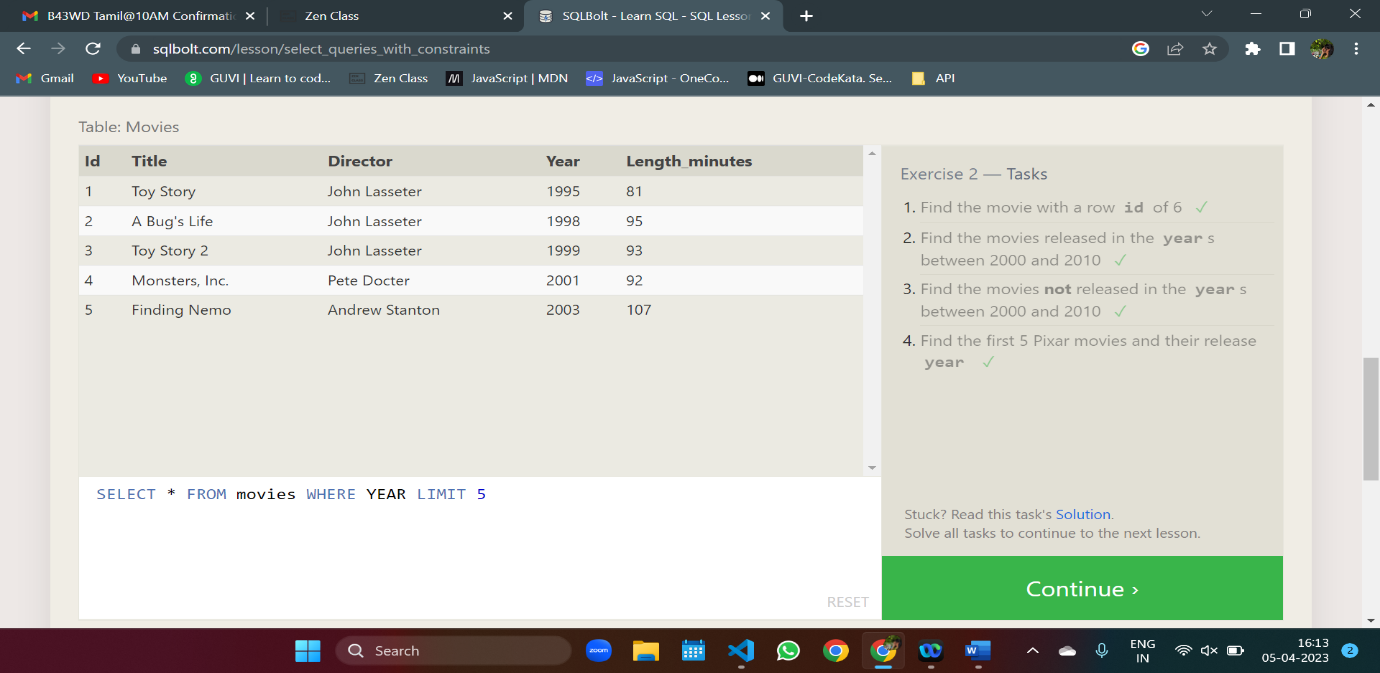
SELECT director FROM movies;

SELECT title,director FROM movies;

SELECT title,year FROM movies;

SELECT \* FROM movies;

**SQL Lesson 2: Queries with constraints (Pt. 1)**

****

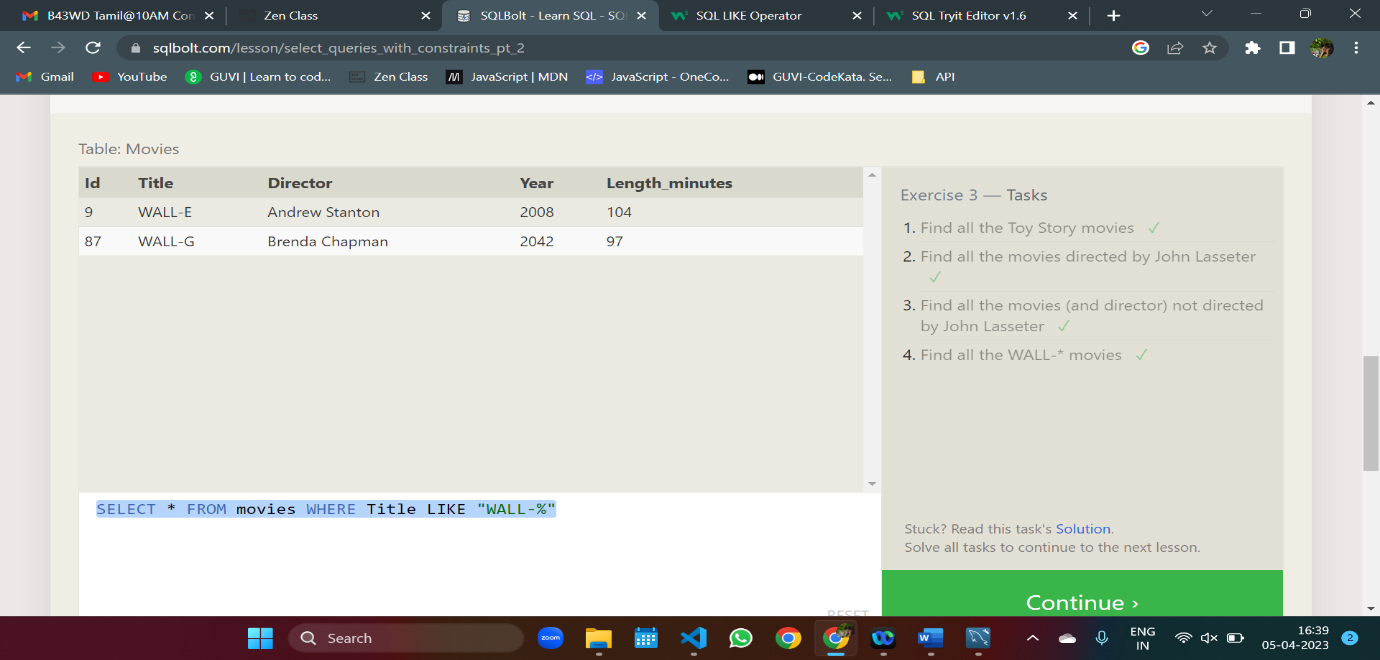
SELECT \* FROM movies WHERE id = 6

SELECT \* FROM movies WHERE year BETWEEN 2000 and 2010

SELECT \* FROM movies WHERE YEAR NOT BETWEEN 2000 AND 2010

SELECT \* FROM movies WHERE YEAR LIMIT 5

**SQL Lesson 3: Queries with constraints (Pt. 2)**



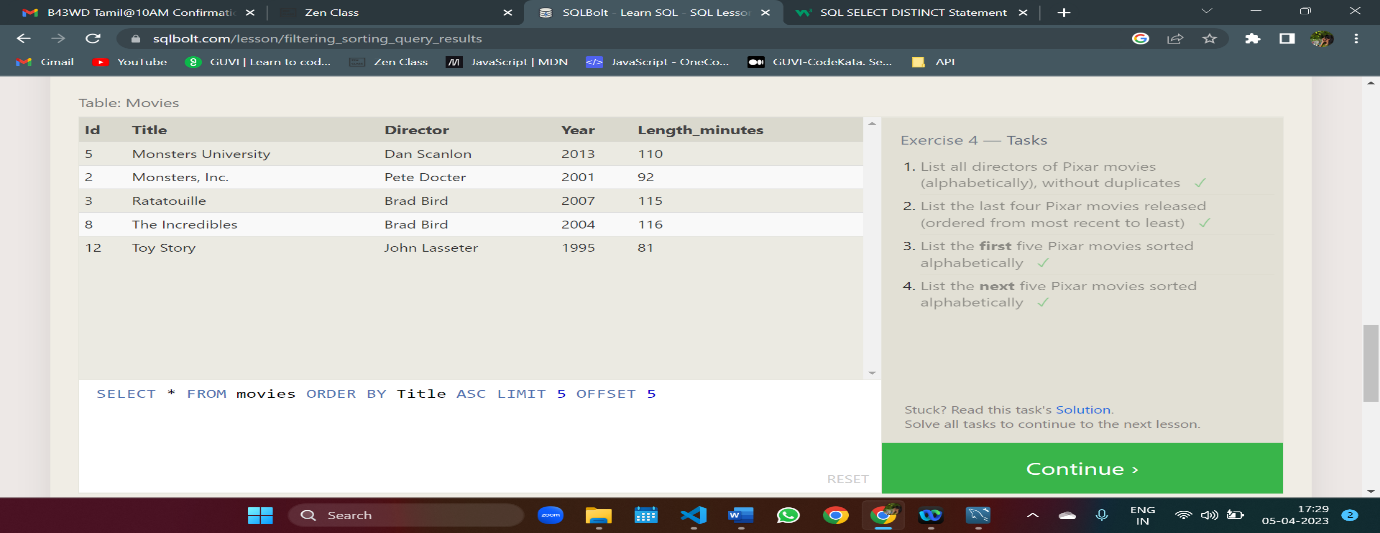
SELECT \* FROM movies WHERE Title LIKE "Toy%"

SELECT \* FROM movies WHERE Director LIKE "John Lasseter"

SELECT \* FROM movies WHERE Director != "John Lasseter"

SELECT \* FROM movies WHERE Title LIKE "WALL-%"

**SQL Lesson 4: Filtering and sorting Query results**



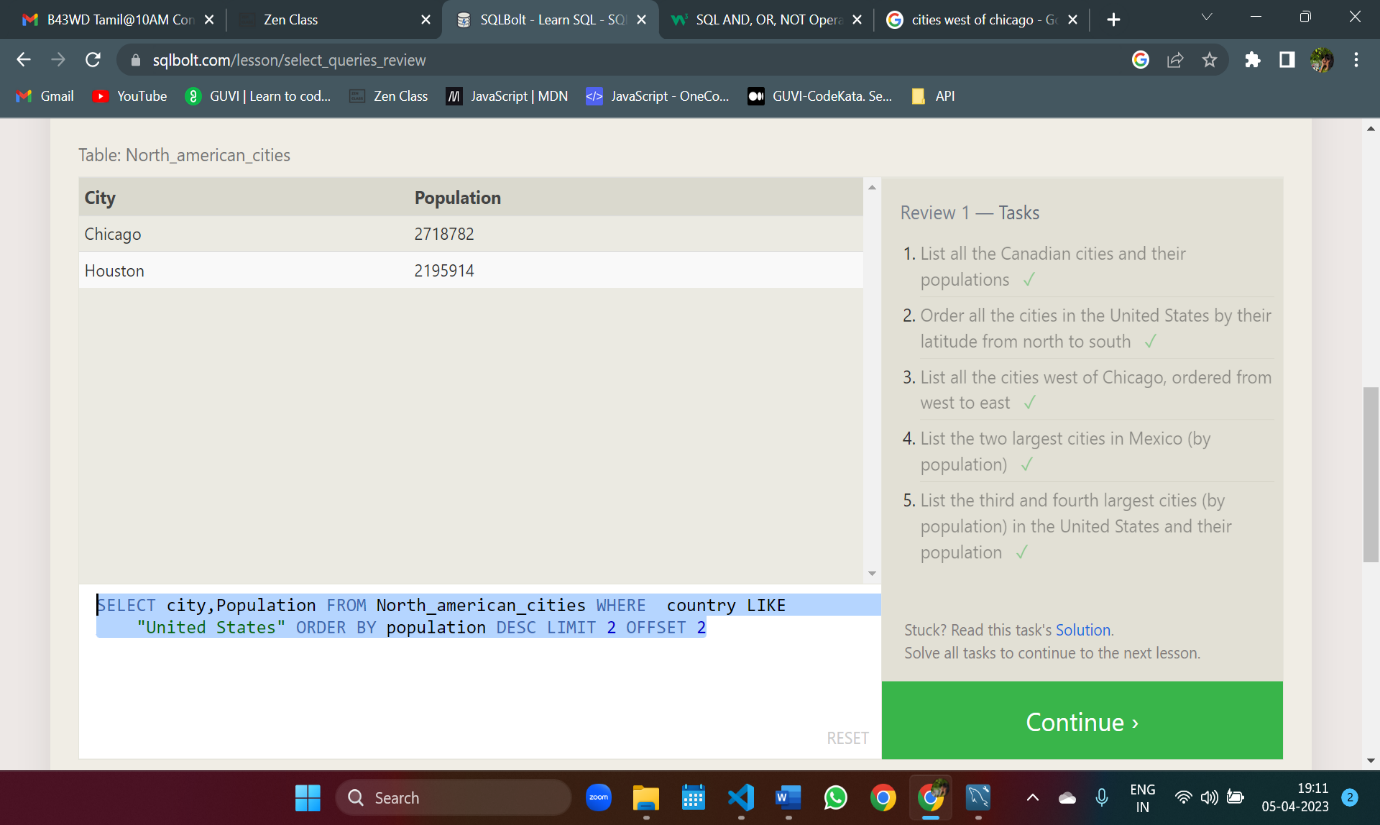
SELECT DISTINCT director FROM movies ORDER BY Director ASC

SELECT \* FROM movies ORDER BY Year DESC LIMIT 4

SELECT \* FROM movies ORDER BY Title ASC LIMIT 5

SELECT \* FROM movies ORDER BY Title ASC LIMIT 5 OFFSET 5

**SQL Review: Simple SELECT Queries**



SELECT city,population FROM North\_american\_cities WHERE country LIKE "Canada"

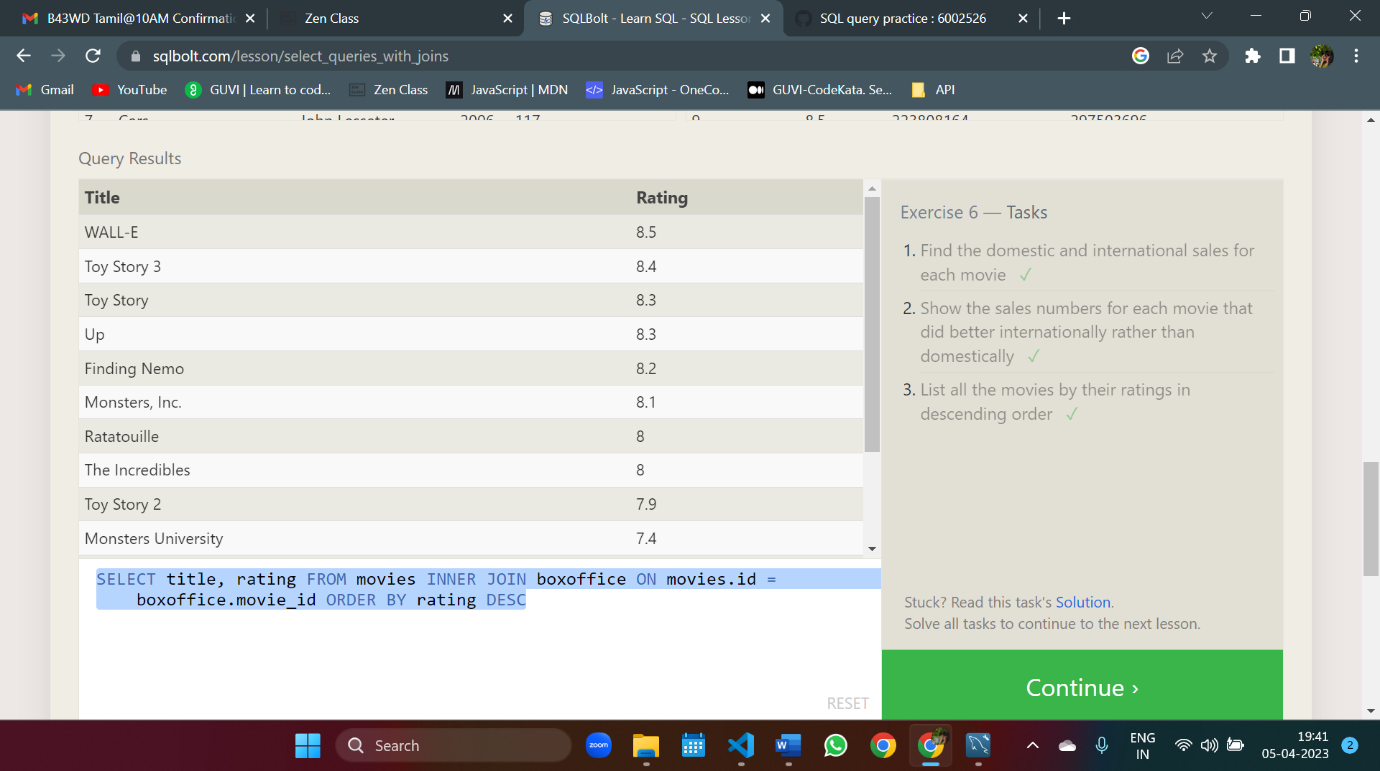
SELECT \* FROM North\_american\_cities WHERE country LIKE "United States" ORDER BY Latitude DESC

SELECT \* FROM North\_american\_cities WHERE Longitude < -87.69 ORDER BY Longitude ASC;

SELECT \* FROM North\_american\_cities WHERE country = "Mexico" ORDER BY population DESC LIMIT 2

SELECT city,Population FROM North\_american\_cities WHERE country LIKE "United States" ORDER BY population DESC LIMIT 2 OFFSET 2

**SQL Lesson 6: Multi-table queries with JOINs**

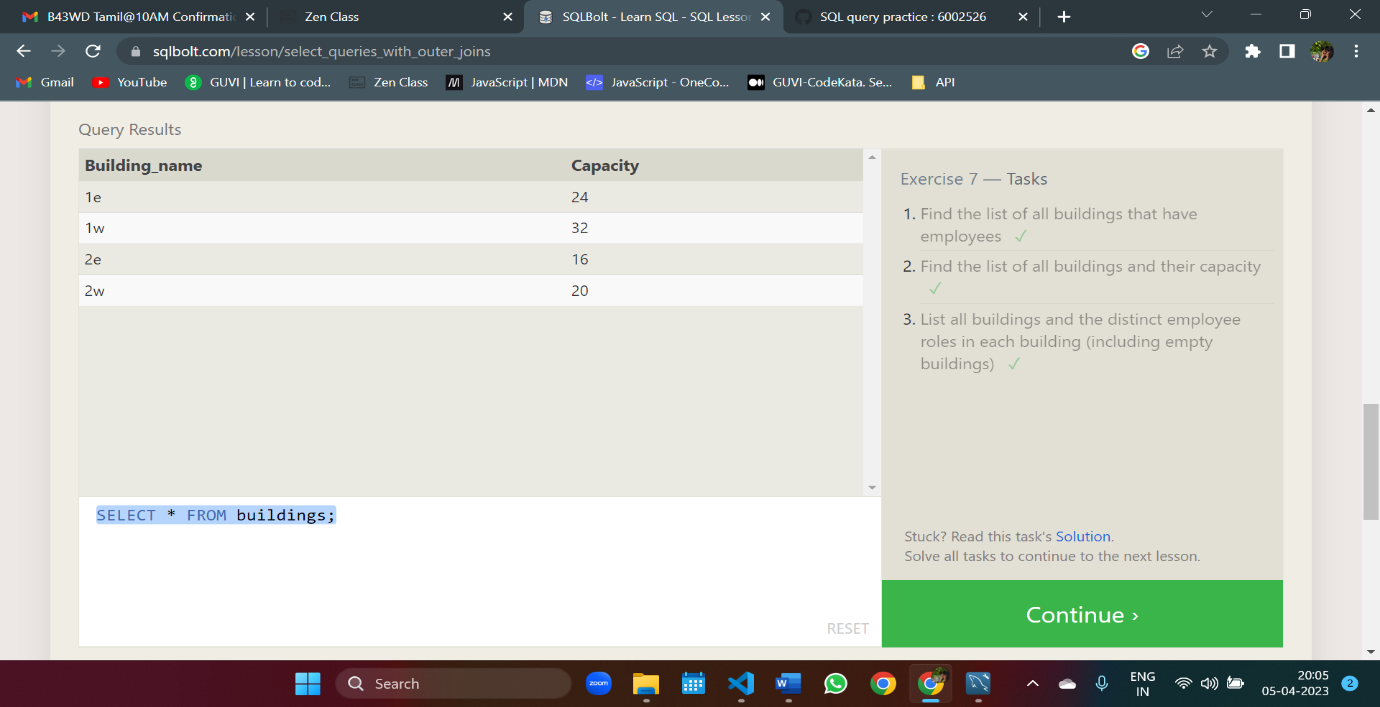


SELECT m.Title, b.Domestic\_sales, b.International\_sales FROM movies m JOIN Boxoffice b ON m.id = b.Movie\_id

SELECT title, domestic\_sales, international\_sales FROM movies INNER JOIN boxoffice ON movies.id = boxoffice.movie\_id WHERE international\_sales > domestic\_sales;

SELECT title, rating FROM movies INNER JOIN boxoffice ON movies.id = boxoffice.movie\_id ORDER BY rating DESC

**SQL Lesson 7: OUTER JOINs**

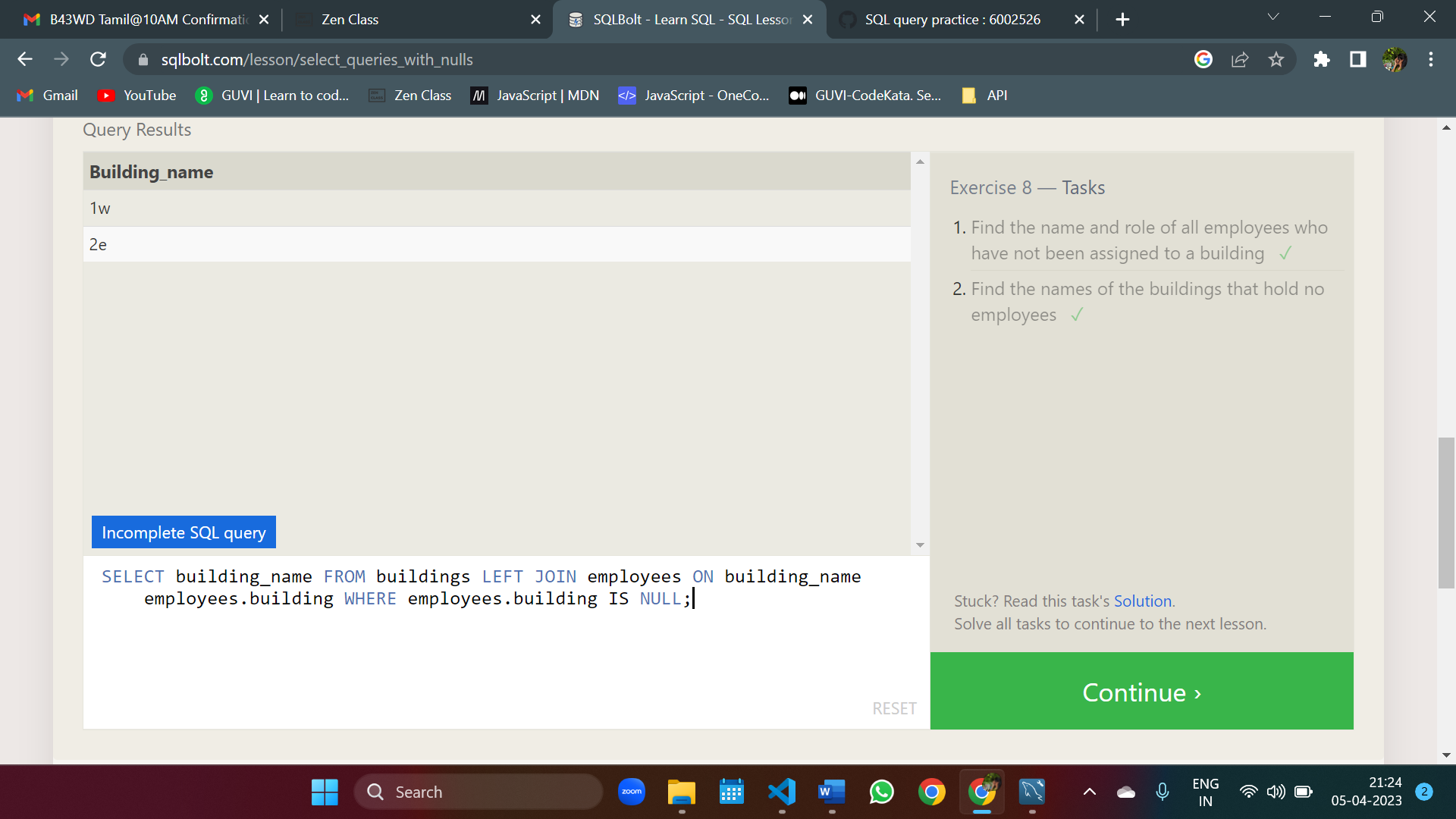
****

SELECT DISTINCT building from employees

SELECT \* FROM buildings;

SELECT DISTINCT building\_name, role FROM buildings LEFT JOIN employees ON buildings.building\_name = employees.building

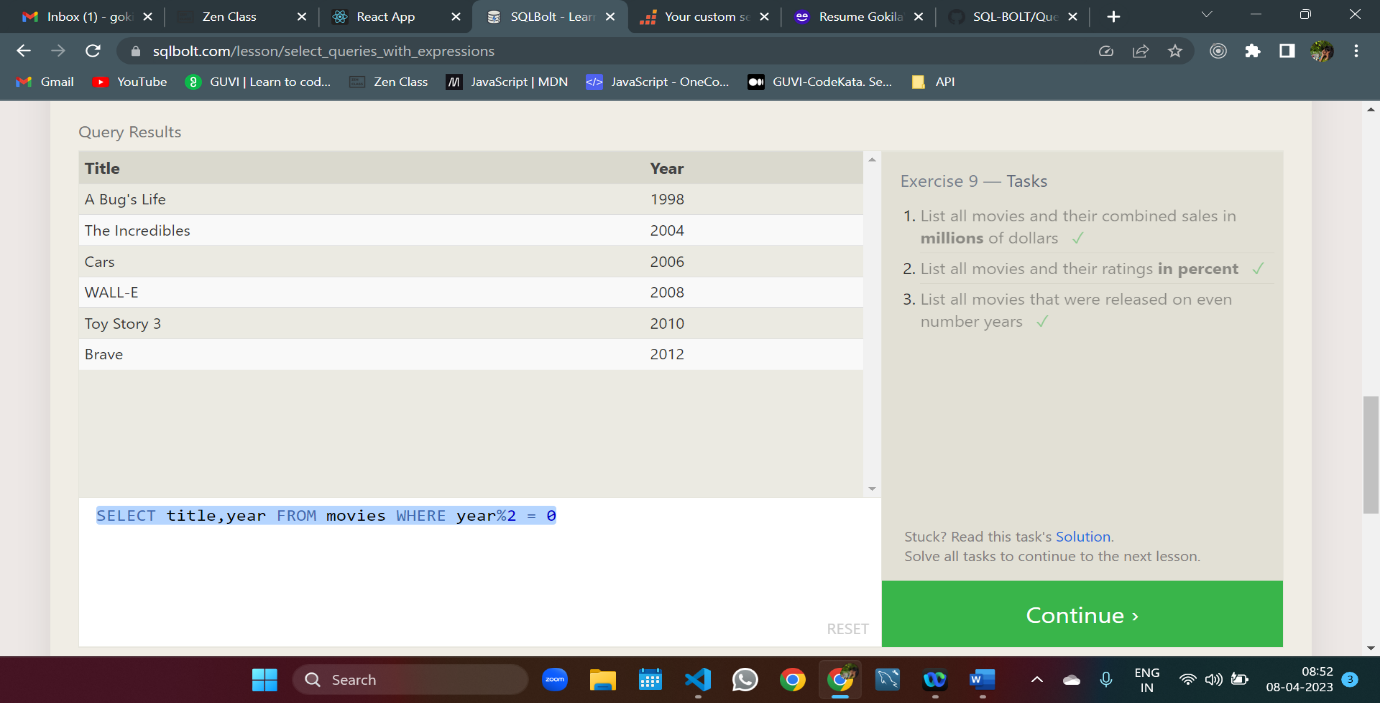
**SQL Lesson 8: A short note on NULLs**



SELECT name,role FROM employees WHERE building is null

SELECT building\_name FROM buildings LEFT JOIN employees ON building\_name employees.building WHERE employees.building IS NULL;

**SQL Lesson 9: Queries with expressions**

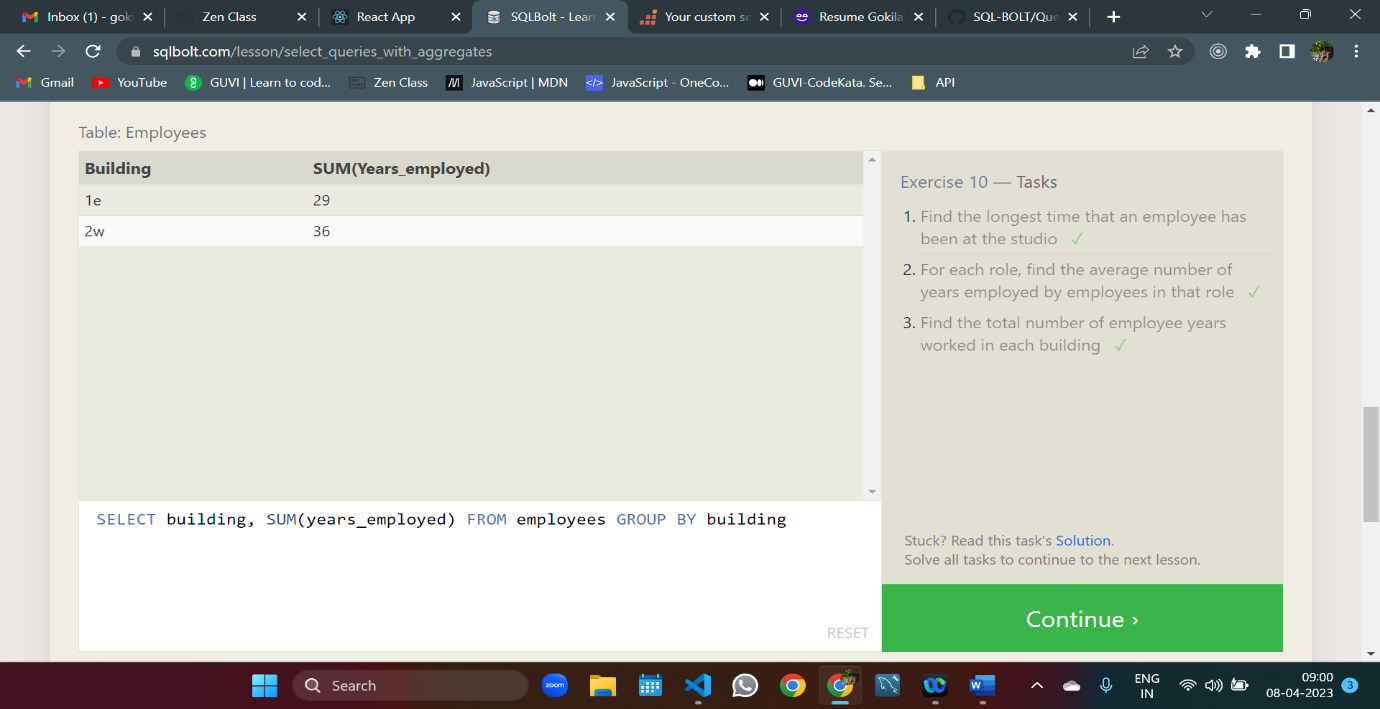


SELECT title,(domestic\_sales+international\_sales)/1000000 AS combined\_sales FROM movies LEFT JOIN boxoffice ON id = movie\_id

SELECT title,(rating\*10)AS Percent FROM movies LEFT JOIN boxoffice ON id = movie\_id

SELECT title,year FROM movies WHERE year%2 = 0

**SQL Lesson 10: Queries with aggregates (Pt. 1)**

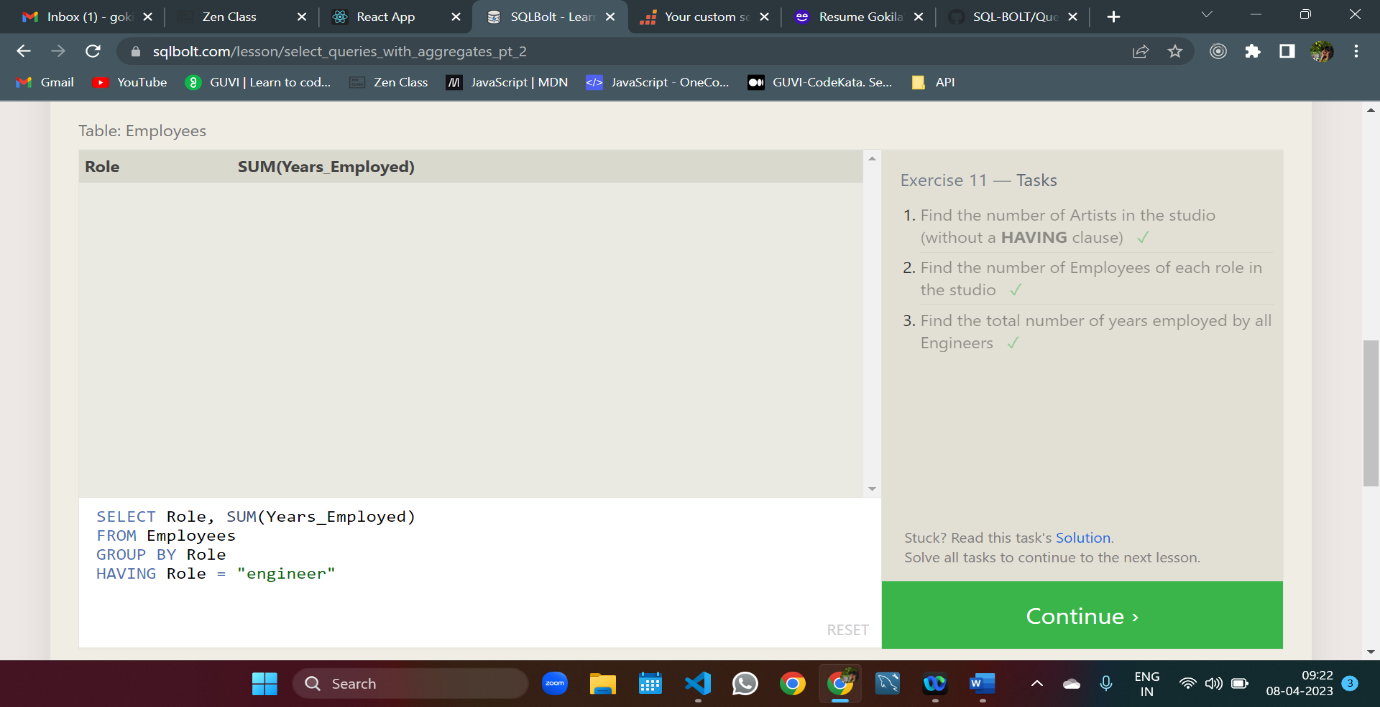
****

SELECT MAX(years\_employed) FROM employees;

SELECT role, AVG(years\_employed) FROM employees GROUP BY Role

SELECT building, SUM(years\_employed) FROM employees GROUP BY building

**SQL Lesson 11: Queries with aggregates (Pt. 2)**

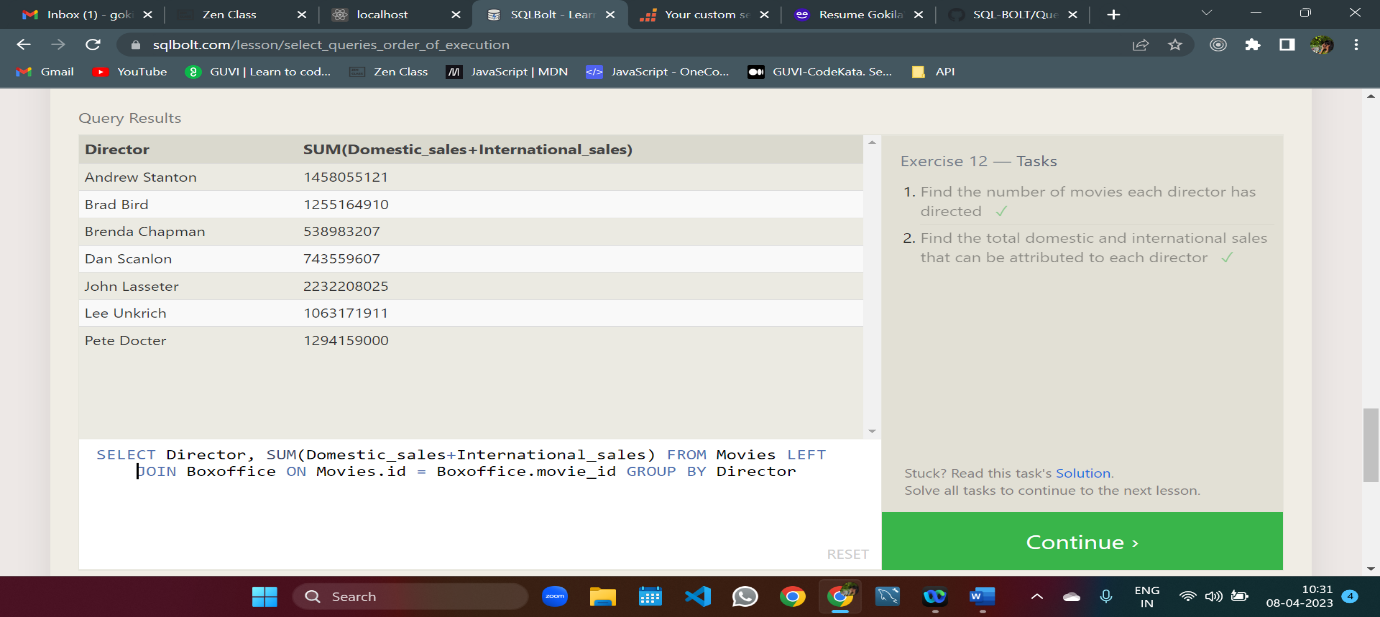
****

SELECT Role, COUNT(\*) AS Number\_of\_ArtistsFROM Employees WHERE Role = "Artist";

SELECT role,count(\*) FROM employees GROUP BY role

SELECT Role, SUM(Years\_Employed) FROM Employees GROUP BY Role HAVING Role = "engineer"

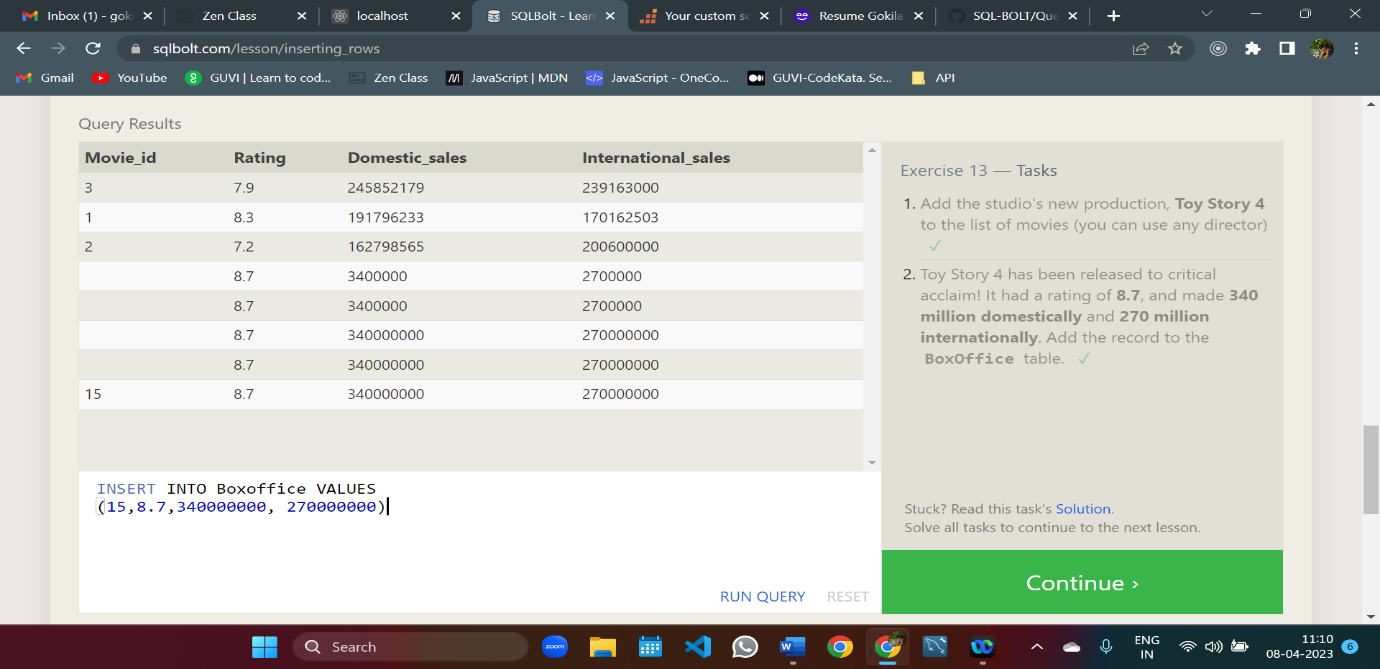
**SQL Lesson 12: Order of execution of a Query**



SELECT COUNT(\*) title,director FROM movies GROUP BY director

SELECT Director, SUM(Domestic\_sales+International\_sales) FROM Movies LEFT JOIN Boxoffice ON Movies.id = Boxoffice.movie\_id GROUP BY Director

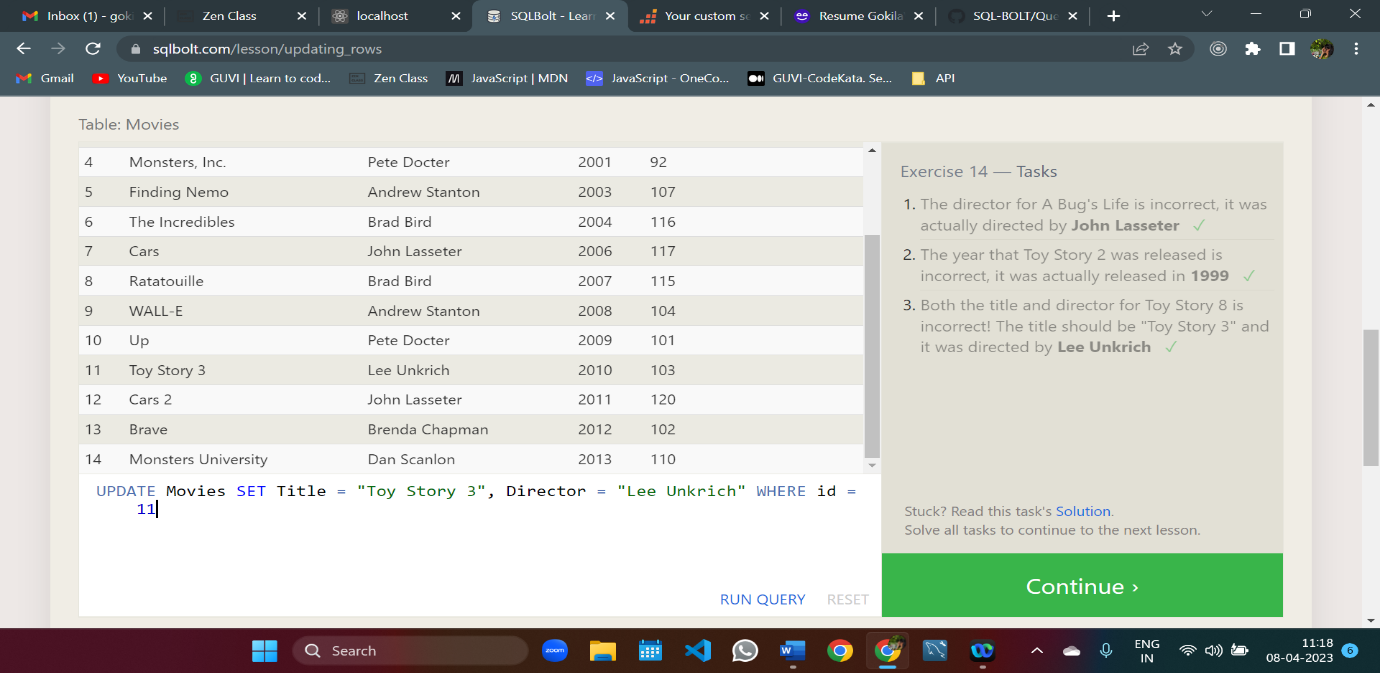
**SQL Lesson 13: Inserting rows**

****

INSERT INTO Movies(Title,Director) VALUES ('Toy Story 4','John Lasseter')

INSERT INTO Boxoffice VALUES (15,8.7,340000000, 270000000)

**SQL Lesson 14: Updating rows**

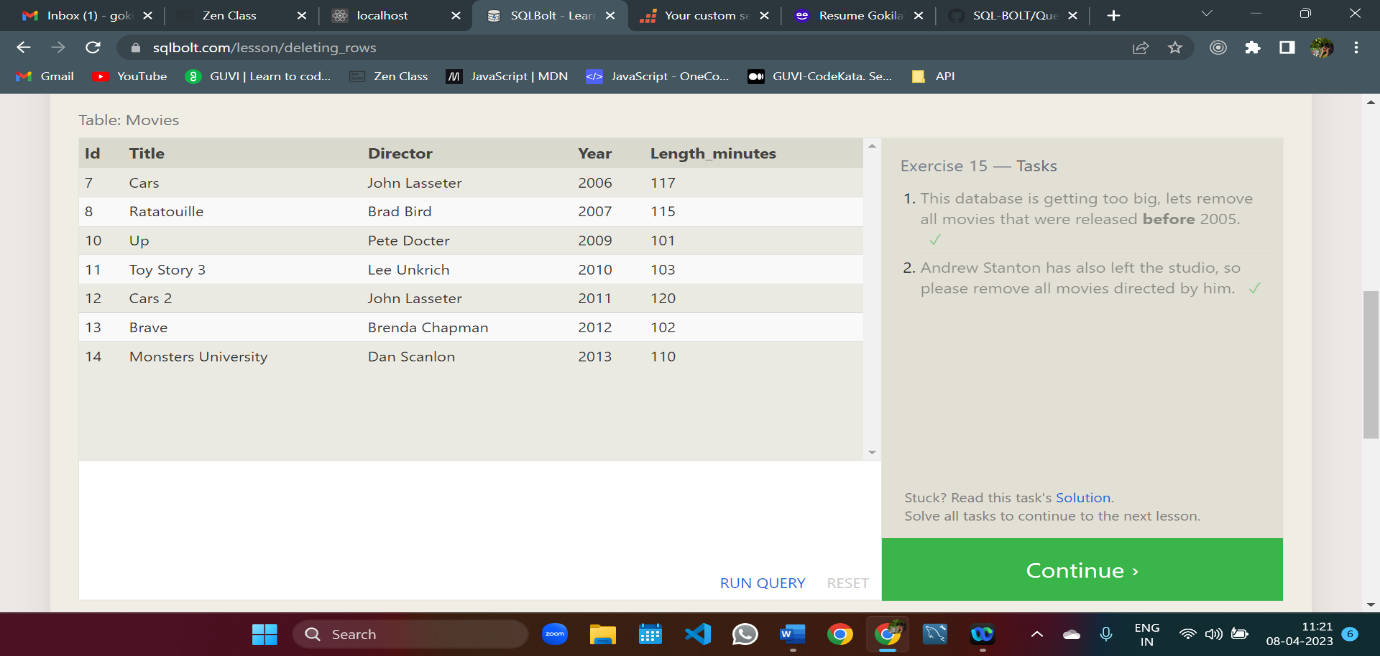
****

UPDATE Movies SET Director = "John Lasseter" WHERE id = 2

UPDATE Movies SET Year = 1999 WHERE id = 3

UPDATE Movies SET Title = "Toy Story 3", Director = "Lee Unkrich" WHERE id = 11

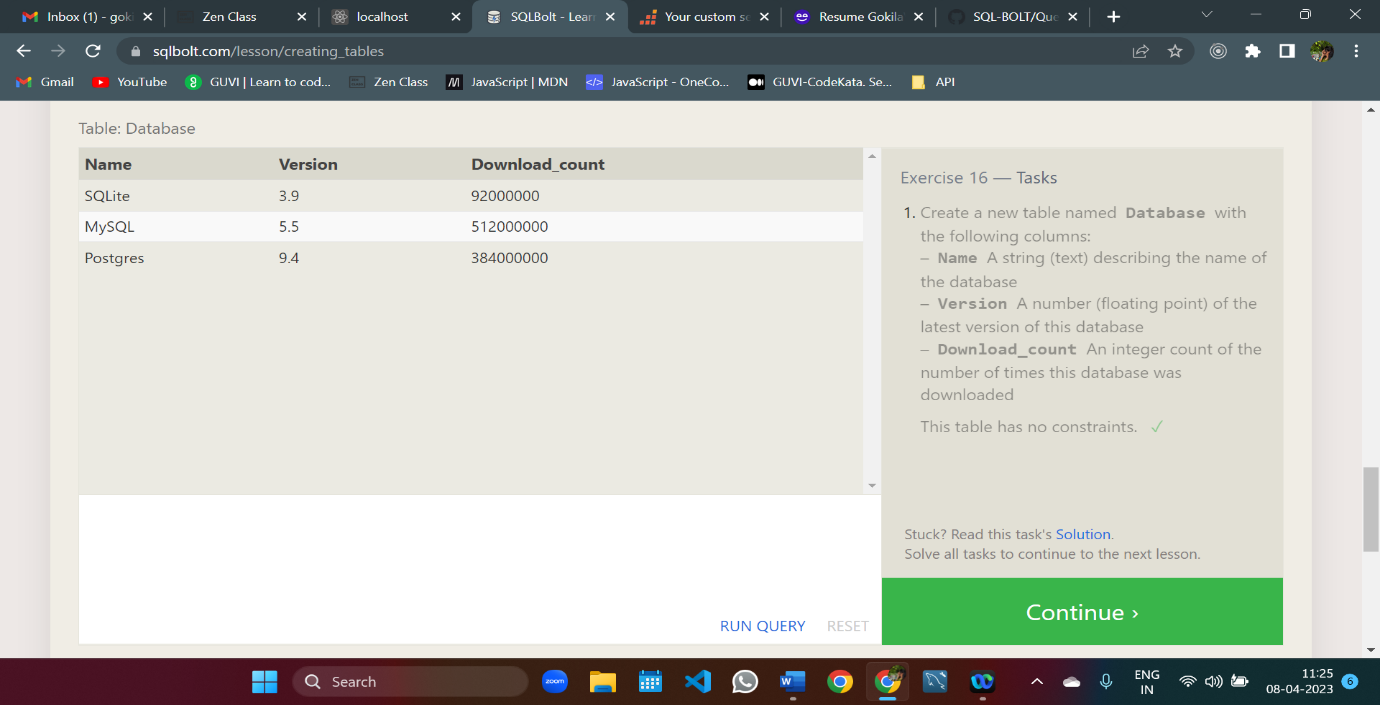
**SQL Lesson 15: Deleting rows**



DELETE FROM Movies WHERE Year < 2005

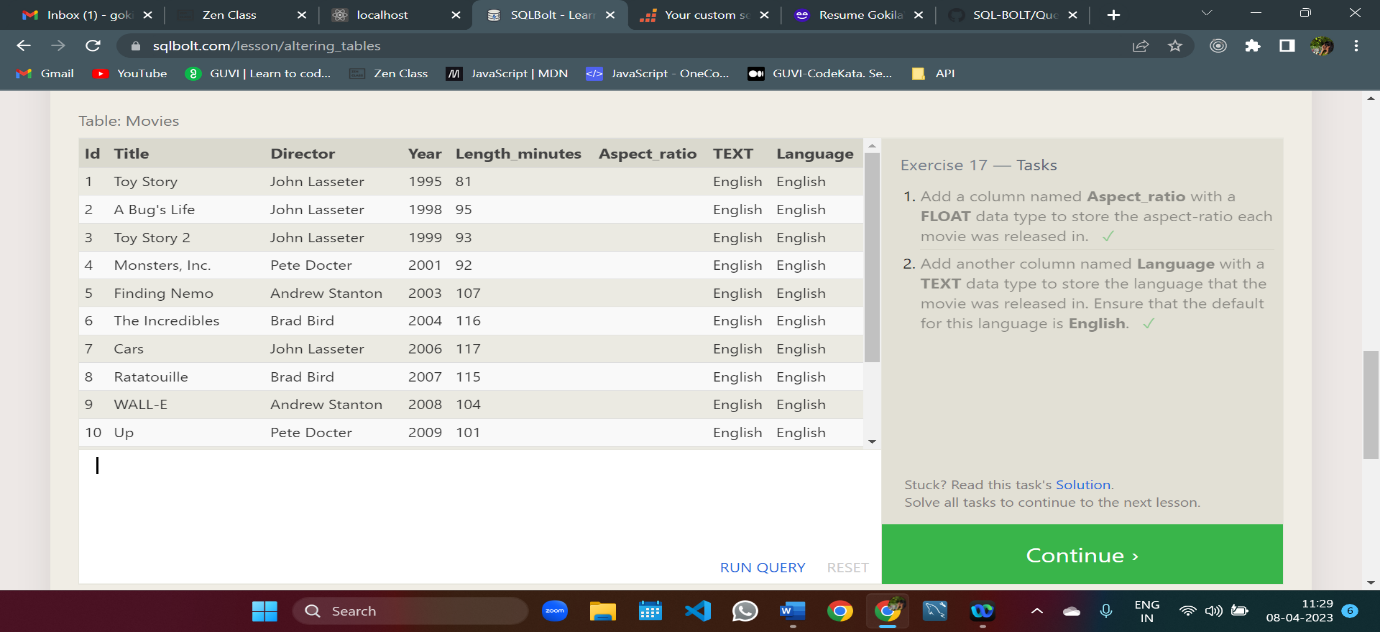
DELETE FROM Movies WHERE Director = "Andrew Stanton"

**SQL Lesson 16: Creating tables**

****

CREATE TABLE IF NOT EXISTS Database(Name varchar(255), Version Float, Download\_count Int)

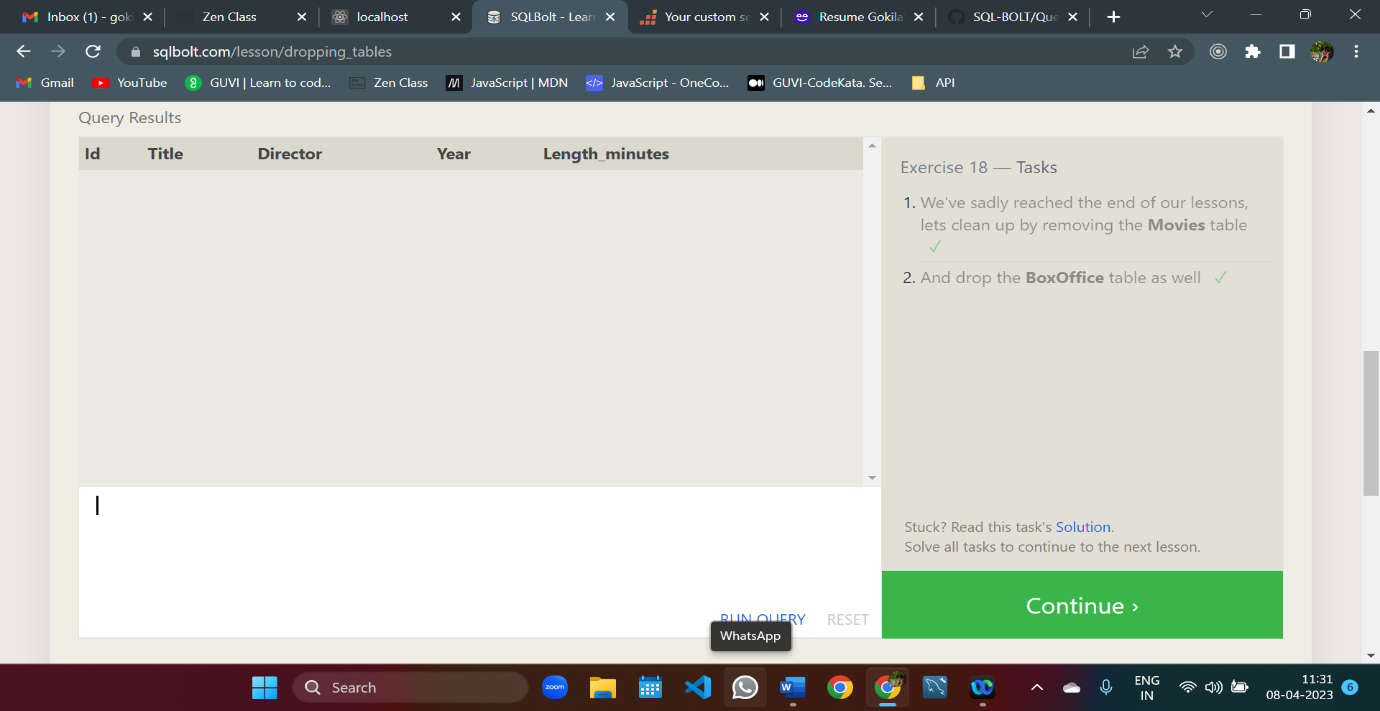
**SQL Lesson 17: Altering tables**

****

ALTER TABLE Movies ADD COLUMN Aspect\_ratio Float

ALTER TABLE Movies ADD COLUMN Language TEXT DEFAULT English

**SQL Lesson 18: Dropping tables**

****

DROP TABLE Movies

DROP TABLE IF EXISTS Boxoffice

