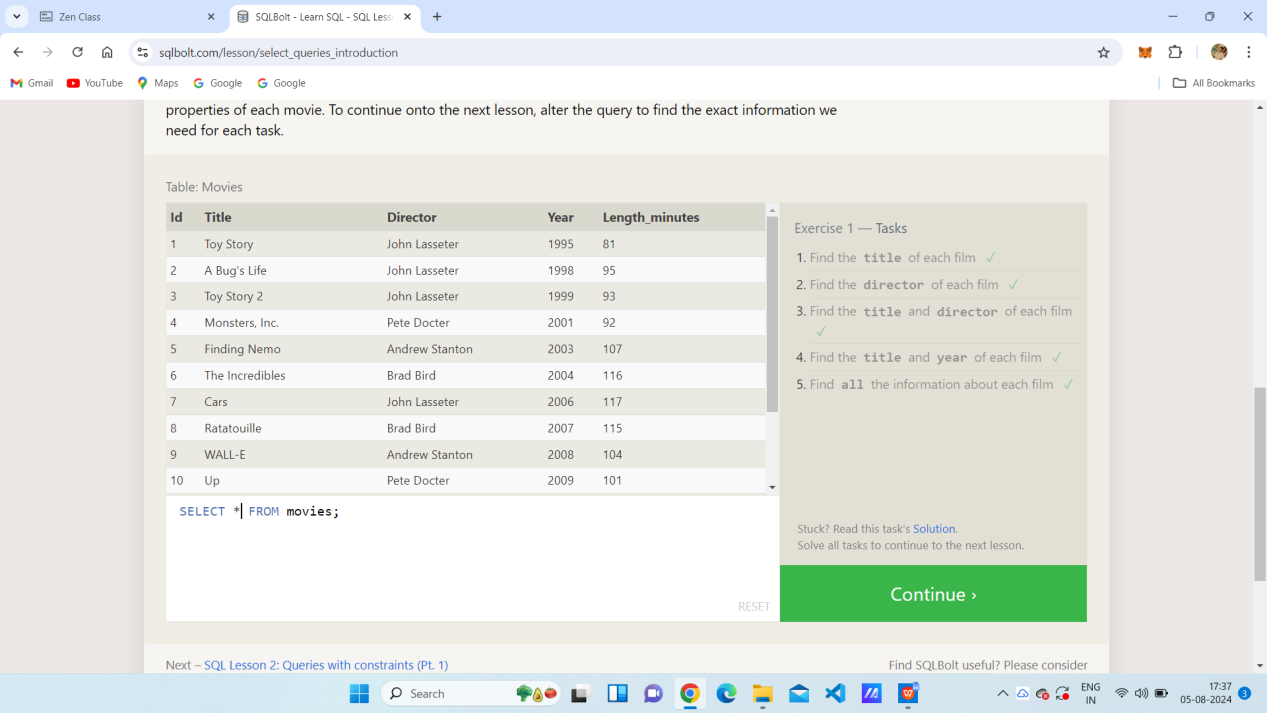
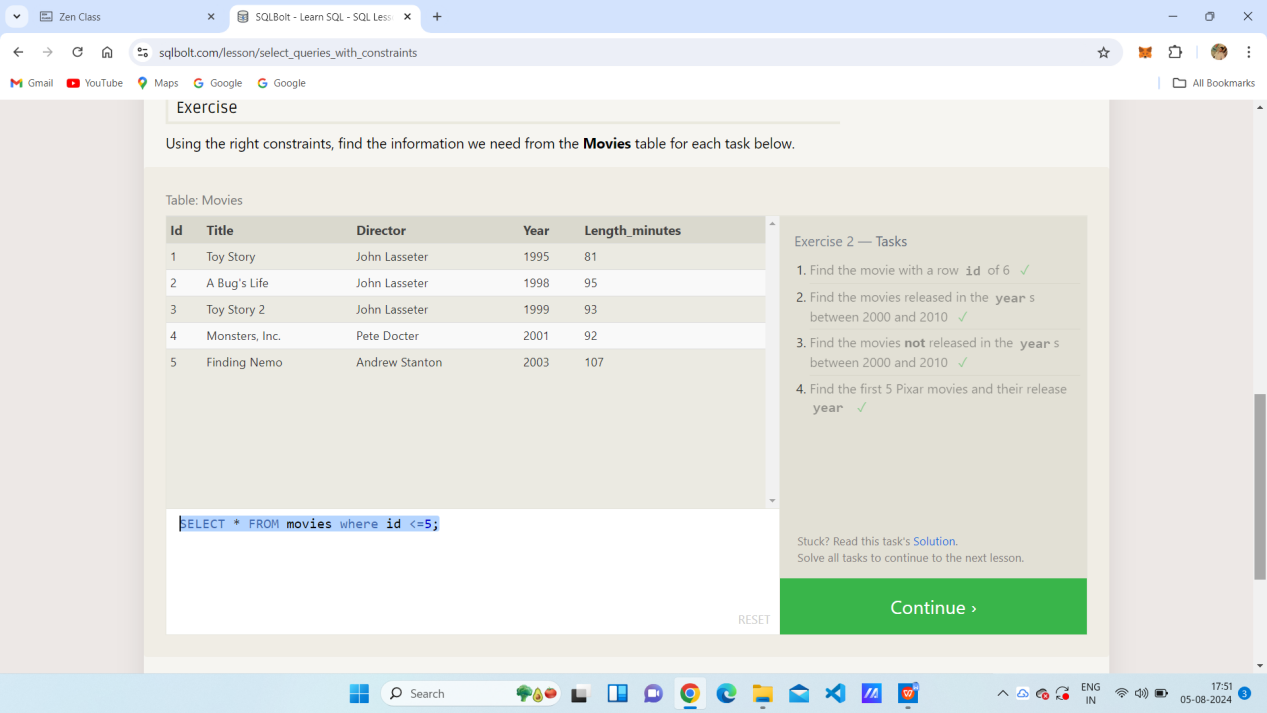
SQL Lesson 1: SELECT queries 101

1. SELECT Title FROM movies;
2. SELECT Director FROM movies;
3. SELECT Title,Director FROM movies;
4. SELECT title,year FROM movies;
5. SELECT \* FROM movies;



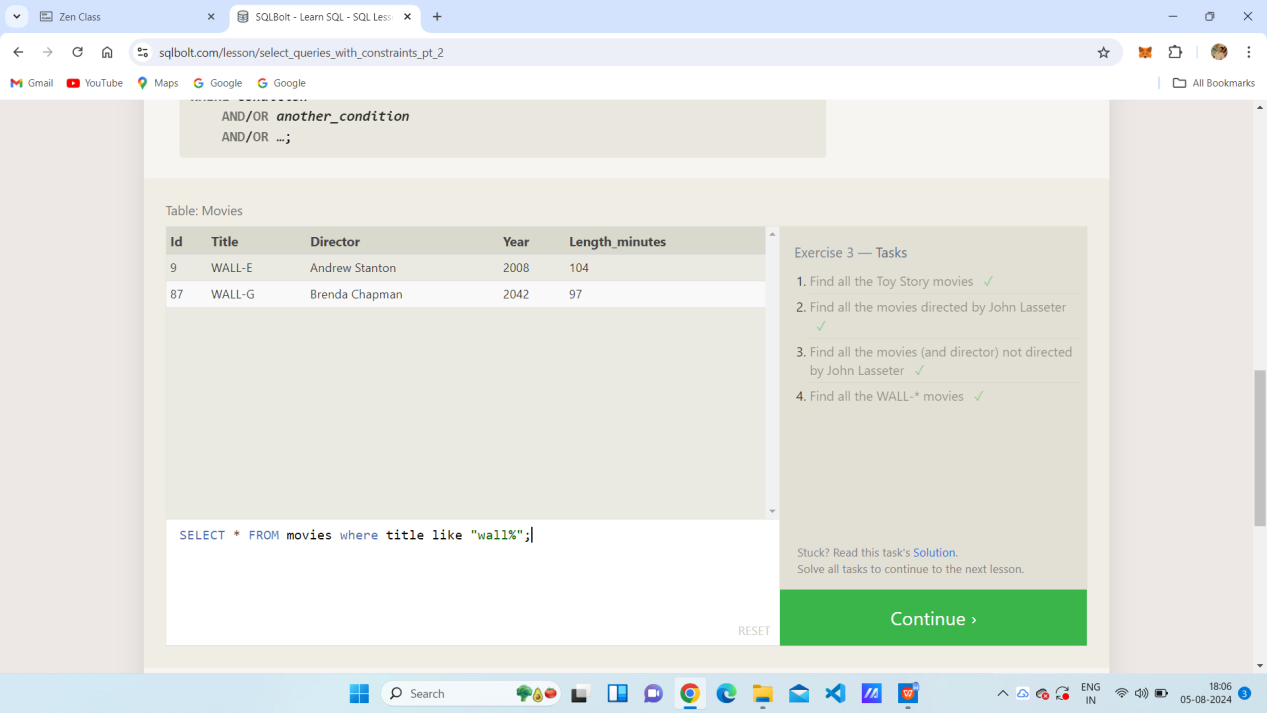
SQL Lesson 2: Queries with constraints (Pt. 1)

1. SELECT \* FROM movies where id = 6;
2. SELECT \* FROM movies where year between 2000 and 2010;
3. SELECT \* FROM movies where year not between 2000 and 2010;
4. SELECT \* FROM movies where id <=5;



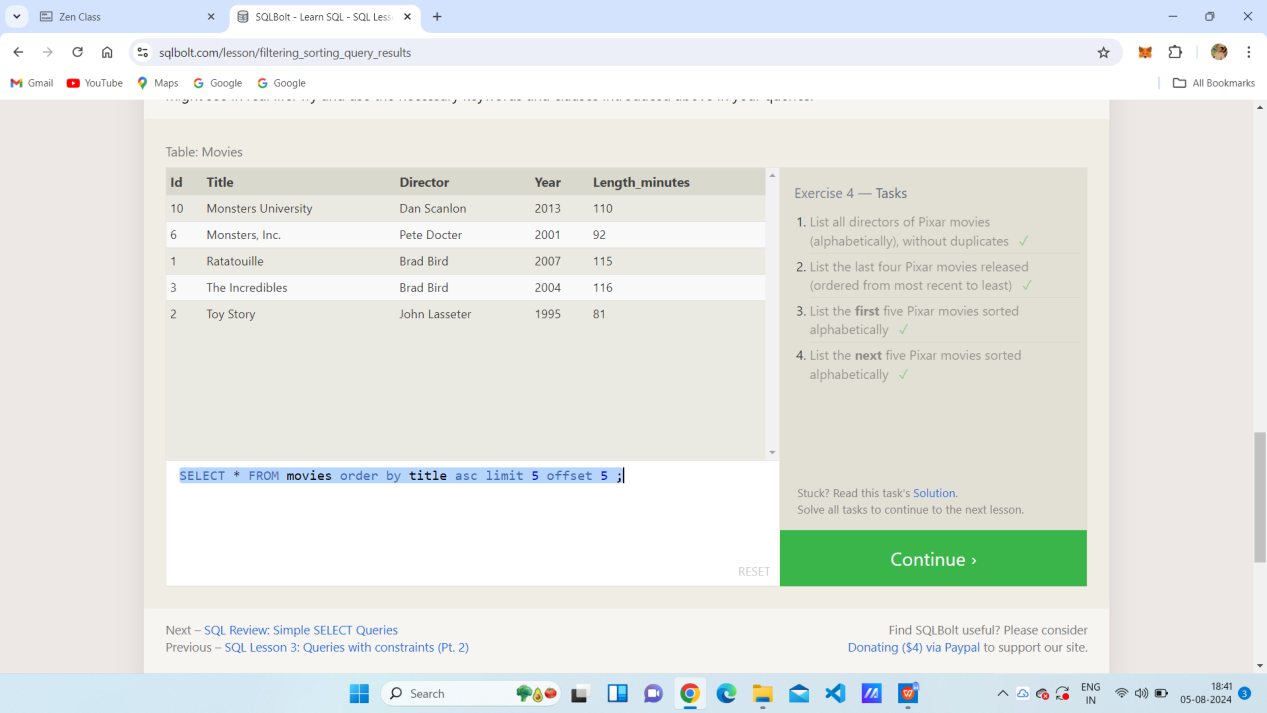
SQL Lesson 3: Queries with constraints (Pt. 2)

1. SELECT \* FROM movies where title like "%toy story%";
2. SELECT \* FROM movies where director like "%john lasseter%";
3. SELECT \* FROM movies where director not like "%john lasseter%";
4. SELECT \* FROM movies where title like "wall%";



SQL Lesson 4: Filtering and sorting Query results

1. SELECT distinct director FROM movies order by director asc;
2. SELECT \* FROM movies order by year desc limit 4 ;
3. SELECT \* FROM movies order by title asc limit 5 ;
4. SELECT \* FROM movies order by title asc limit 5 offset 5 ;

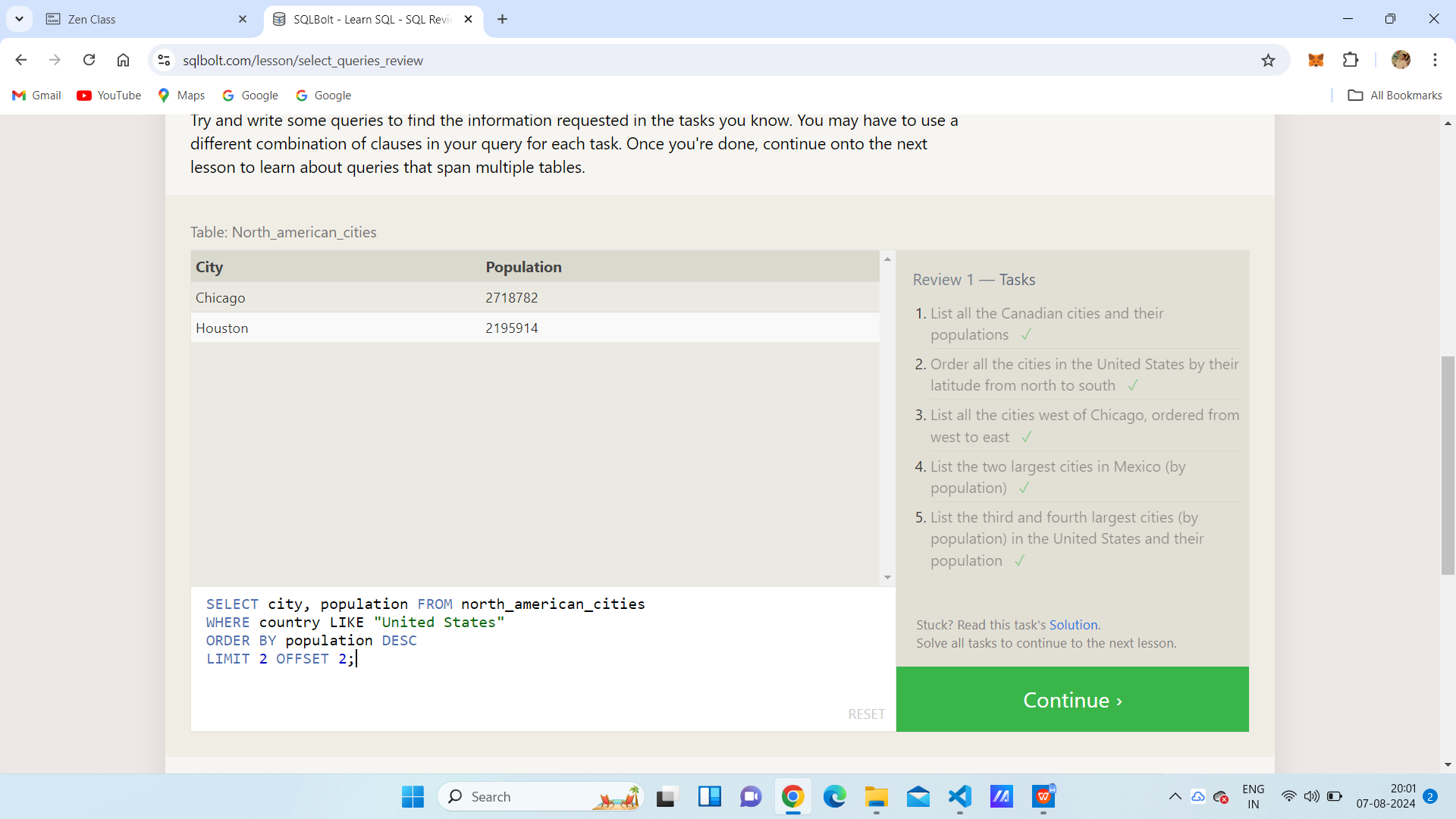


SQL Review: Simple SELECT Queries

1. select city, population from north\_american\_cities where country = "Canada";
2. select city , latitude from North\_american\_cities where country = "United States" order by latitude desc;
3. SELECT city, longitude FROM north\_american\_cities WHERE longitude < -87.629798 ORDER BY longitude ASC;
4. select city,population from North\_american\_cities where country = "Mexico" order by population desc

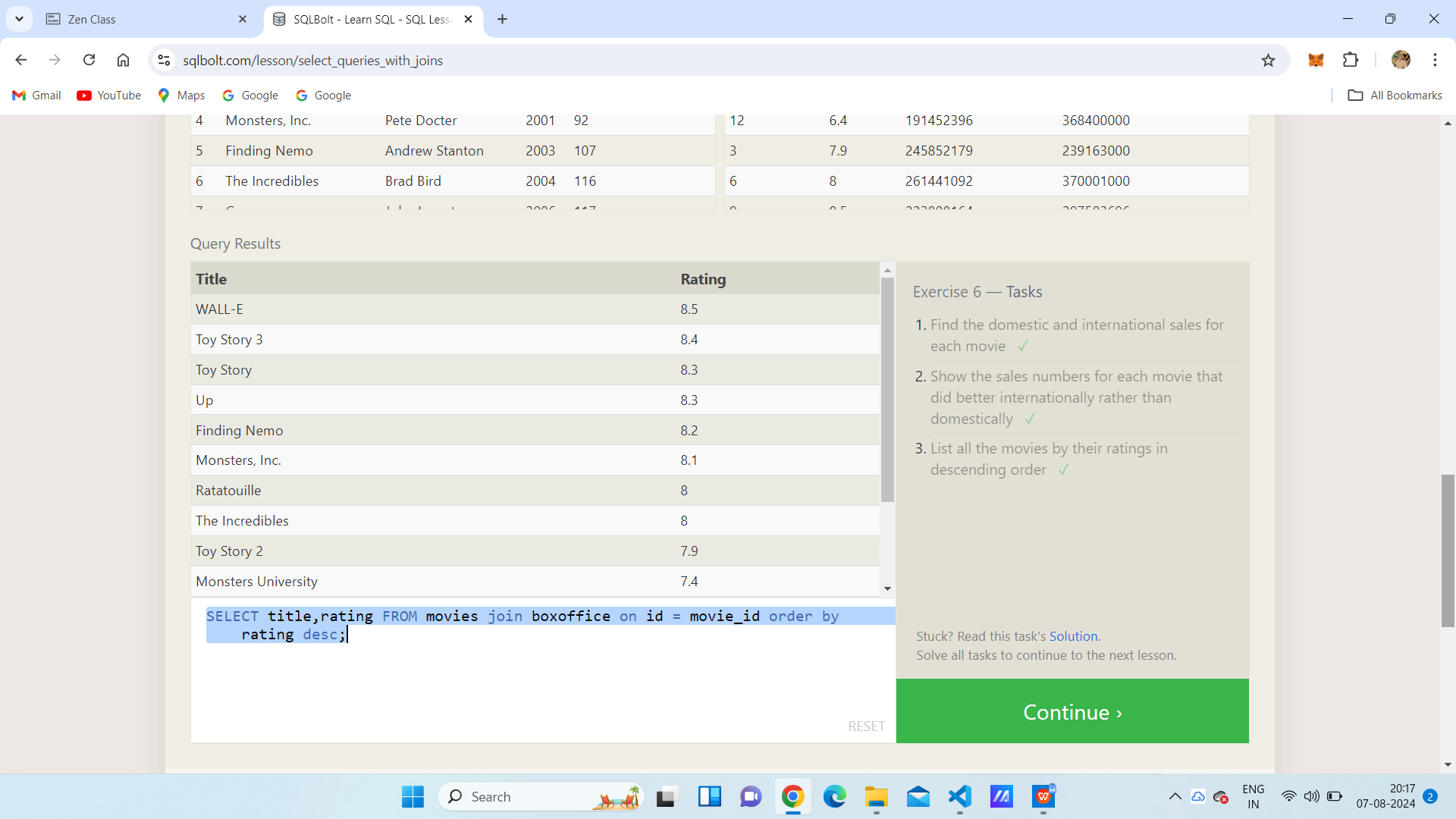
limit 2;

1. select city, population from north\_american\_cities where country LIKE "United States" order by population desc limit 2 offset 2;



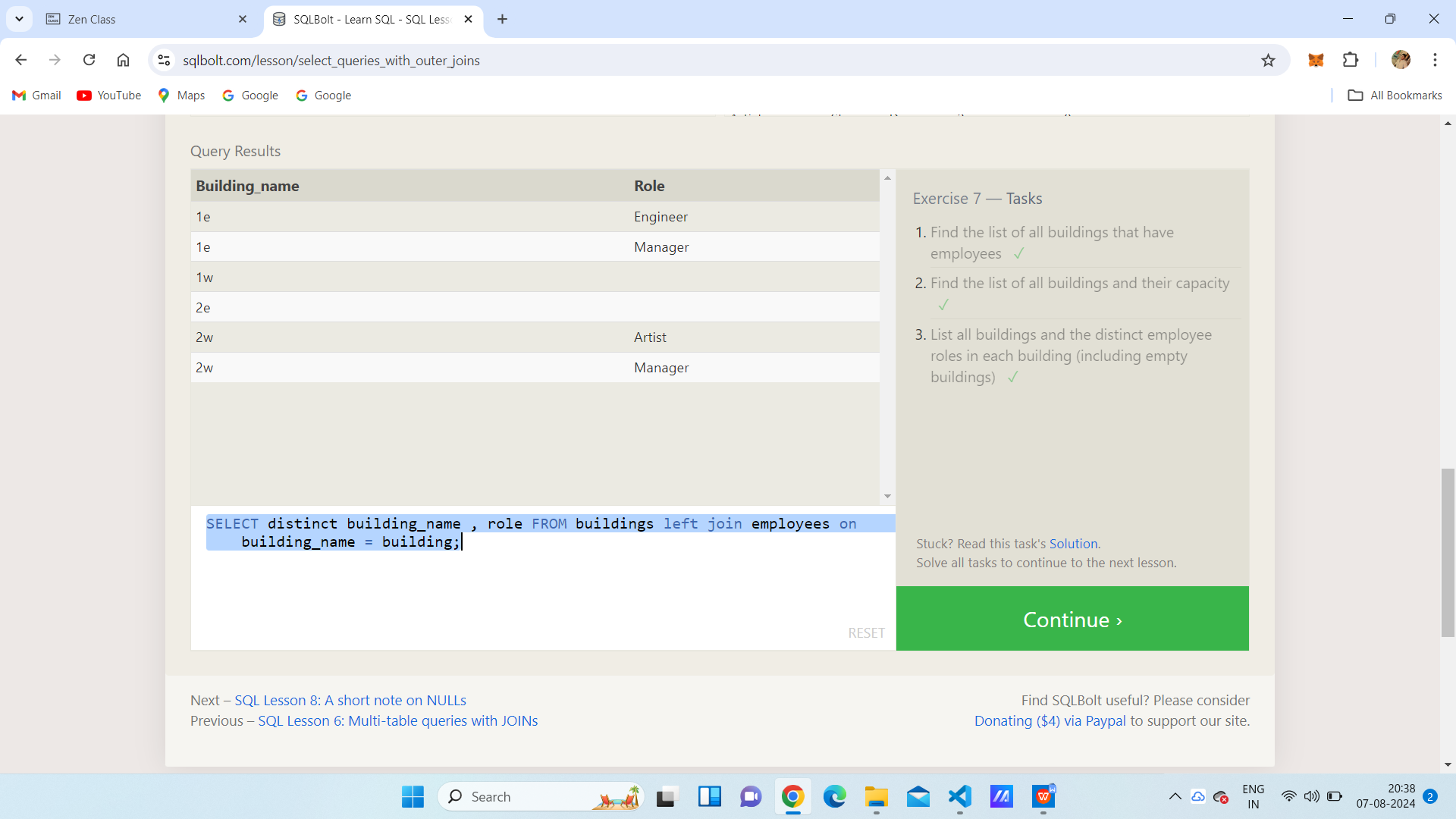
SQL Lesson 6: Multi-table queries with JOINs

1. SELECT title,domestic\_sales,international\_sales FROM movies join boxoffice on id = movie\_id;
2. SELECT title,domestic\_sales,international\_sales FROM movies join boxoffice on id = movie\_id where domestic\_sales<international\_sales;
3. SELECT title,rating FROM movies join boxoffice on id = movie\_id order by rating desc;



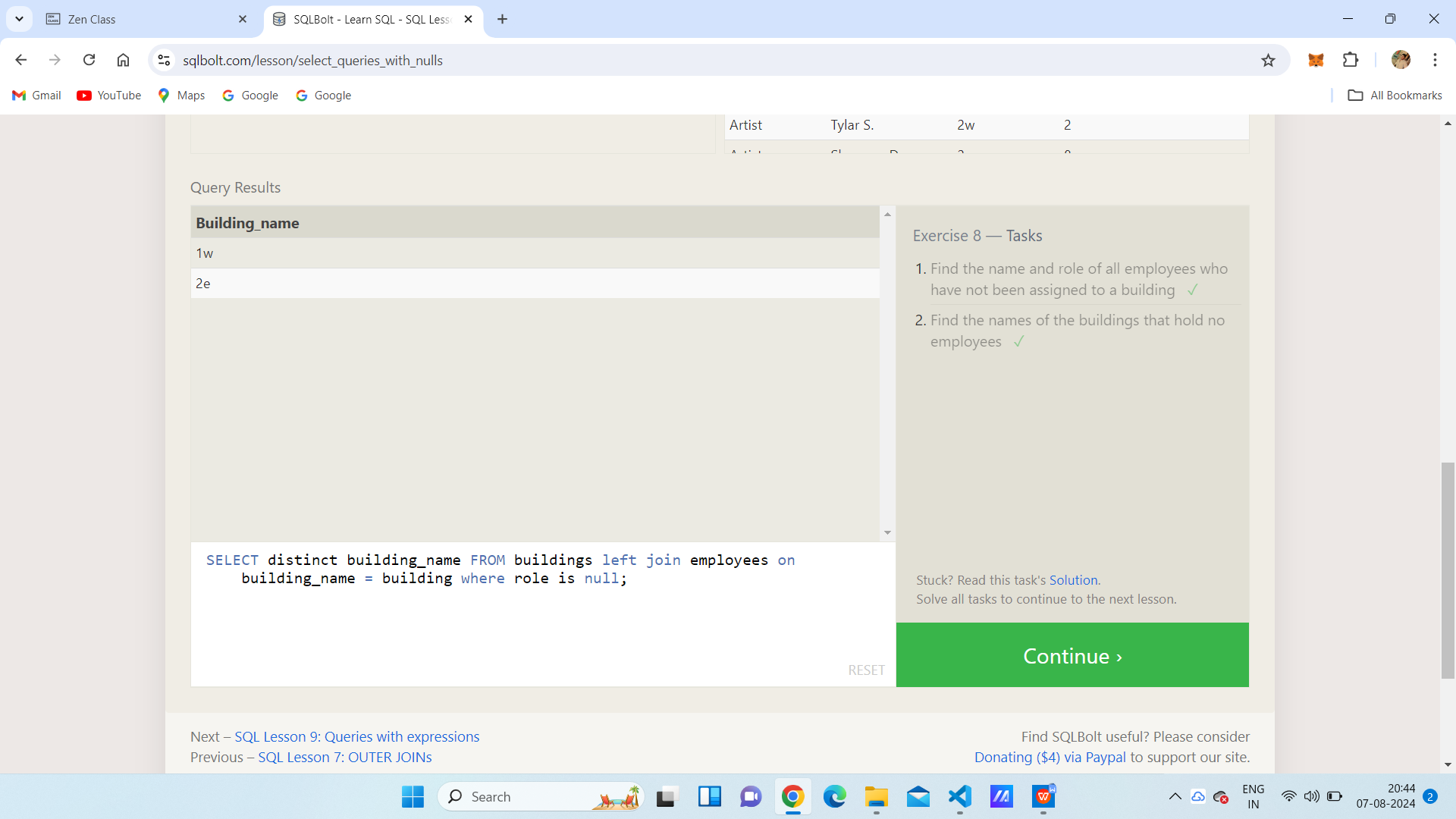
SQL Lesson 7: OUTER JOINs

1. SELECT distinct building FROM employees;
2. SELECT \* FROM buildings;
3. SELECT distinct building\_name , role FROM buildings left join employees on building\_name = building;



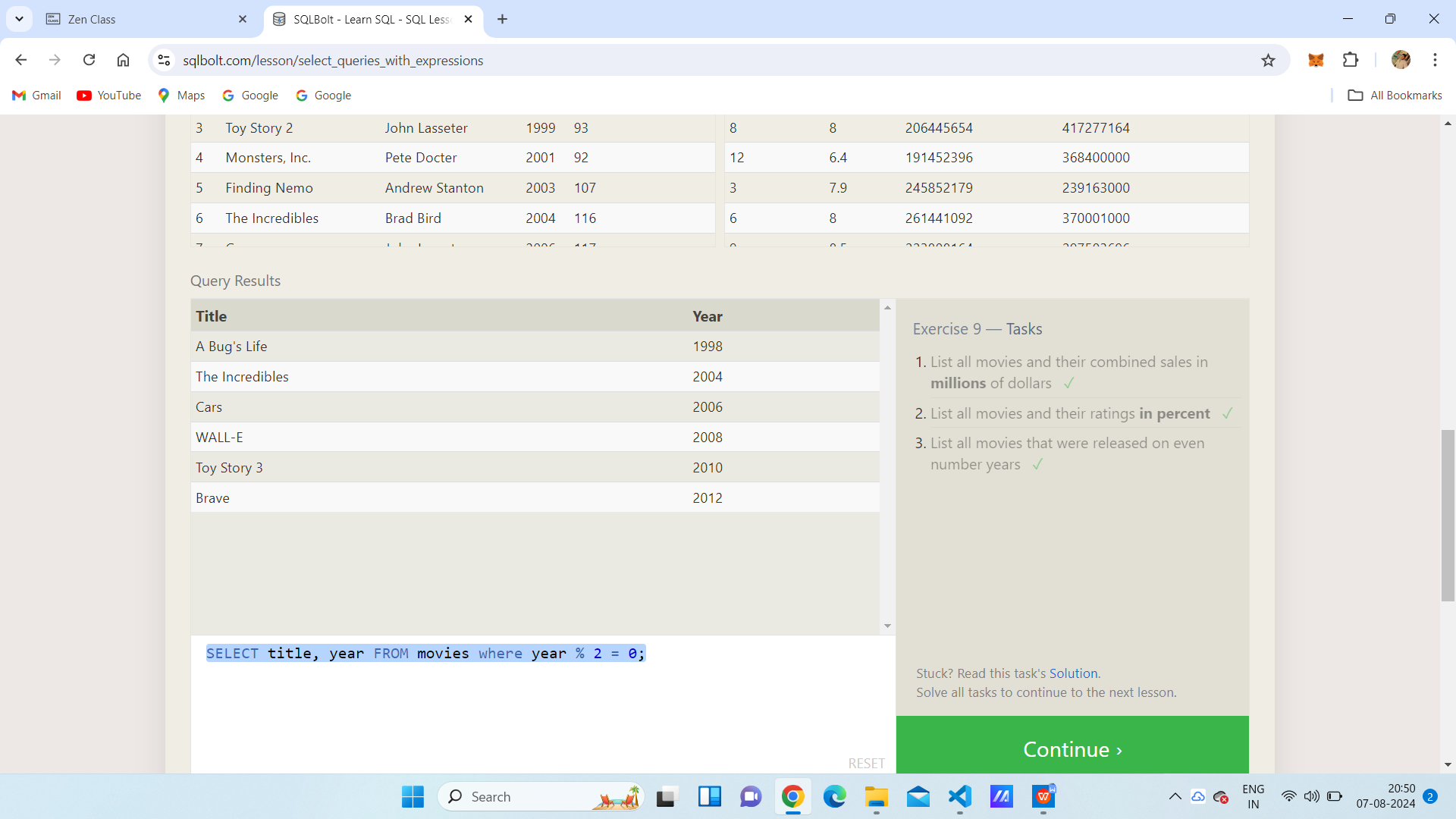
SQL Lesson 8: A short note on NULLs

1. select name, role from employees where building is null;
2. SELECT distinct building\_name FROM buildings left join employees on building\_name = building where role is null;



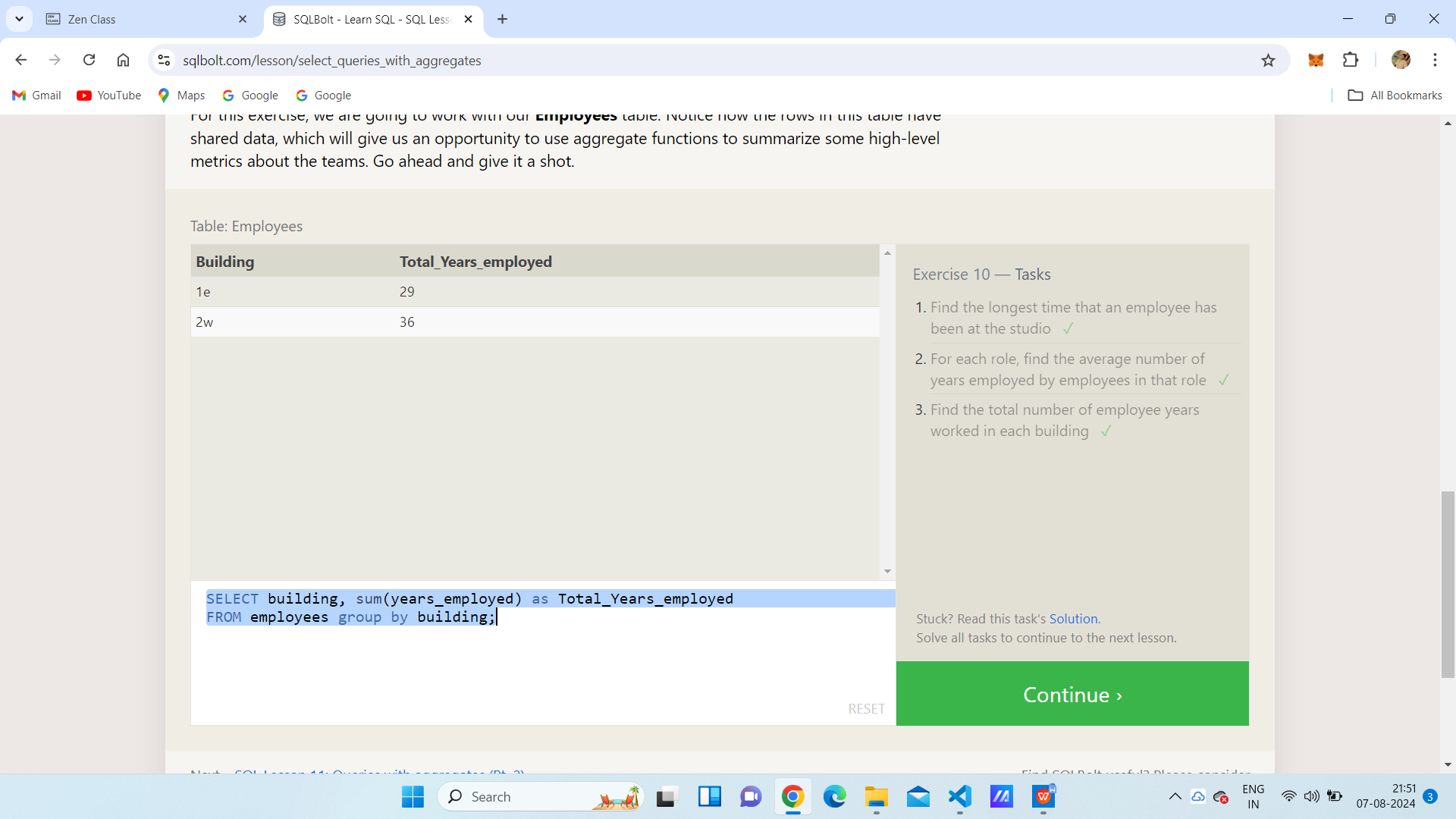
SQL Lesson 9: Queries with expressions

1. SELECT title, (domestic\_sales + international\_sales) / 1000000 as gross\_sales\_millions FROM movies join boxoffice on movies.id = boxoffice.movie\_id;
2. SELECT title, rating \* 10 as rating\_percent FROM movies JOIN boxoffice on movies.id = boxoffice.movie\_id;
3. SELECT title, year FROM movies where year % 2 = 0;



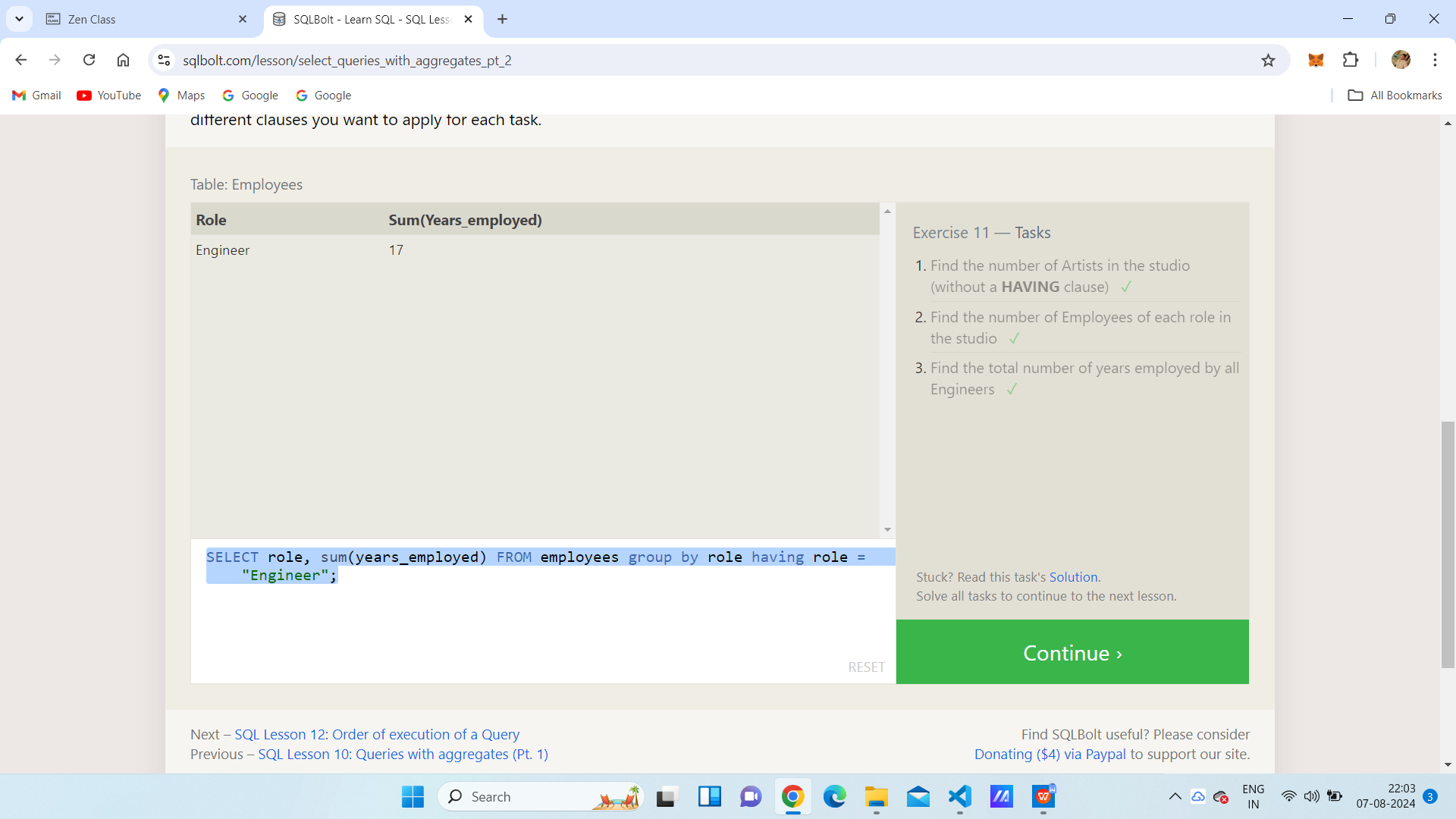
SQL Lesson 10: Queries with aggregates (Pt. 1)

1. SELECT max(years\_employed) as Maximum\_years-employed from employees;
2. SELECT role, avg(years\_employed) as Average\_years\_employed FROM employees group by role;
3. SELECT building, sum(years\_employed) as Total\_Years\_employed FROM employees group by building;



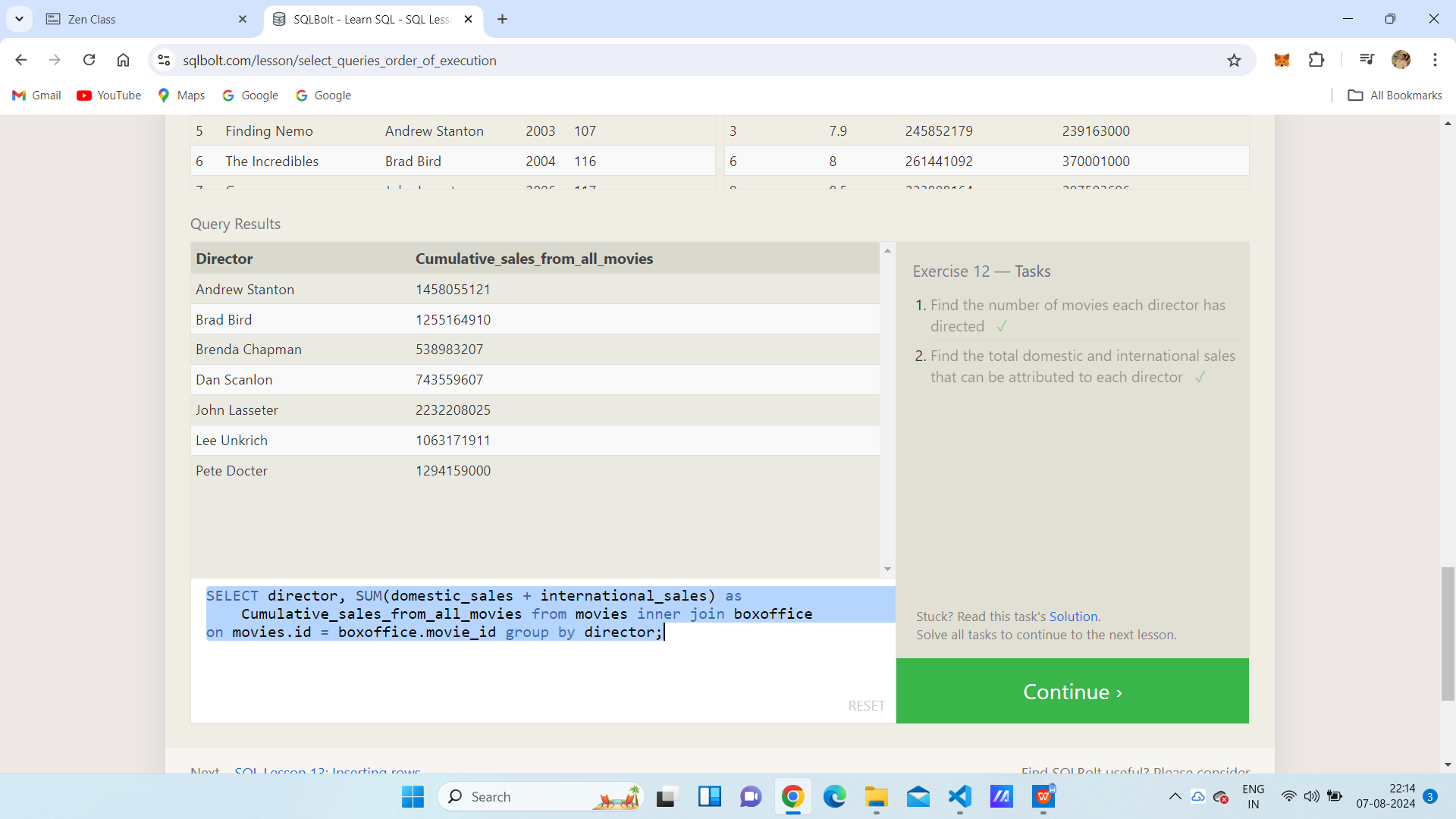
SQL Lesson 11: Queries with aggregates (Pt. 2)

1. select role , count(\*) as Number\_of\_artist from employees where role="Artist";
2. SELECT role, COUNT(\*) as number\_of\_employees FROM employees group by role;
3. SELECT role, sum(years\_employed) FROM employees group by role having role = "Engineer";



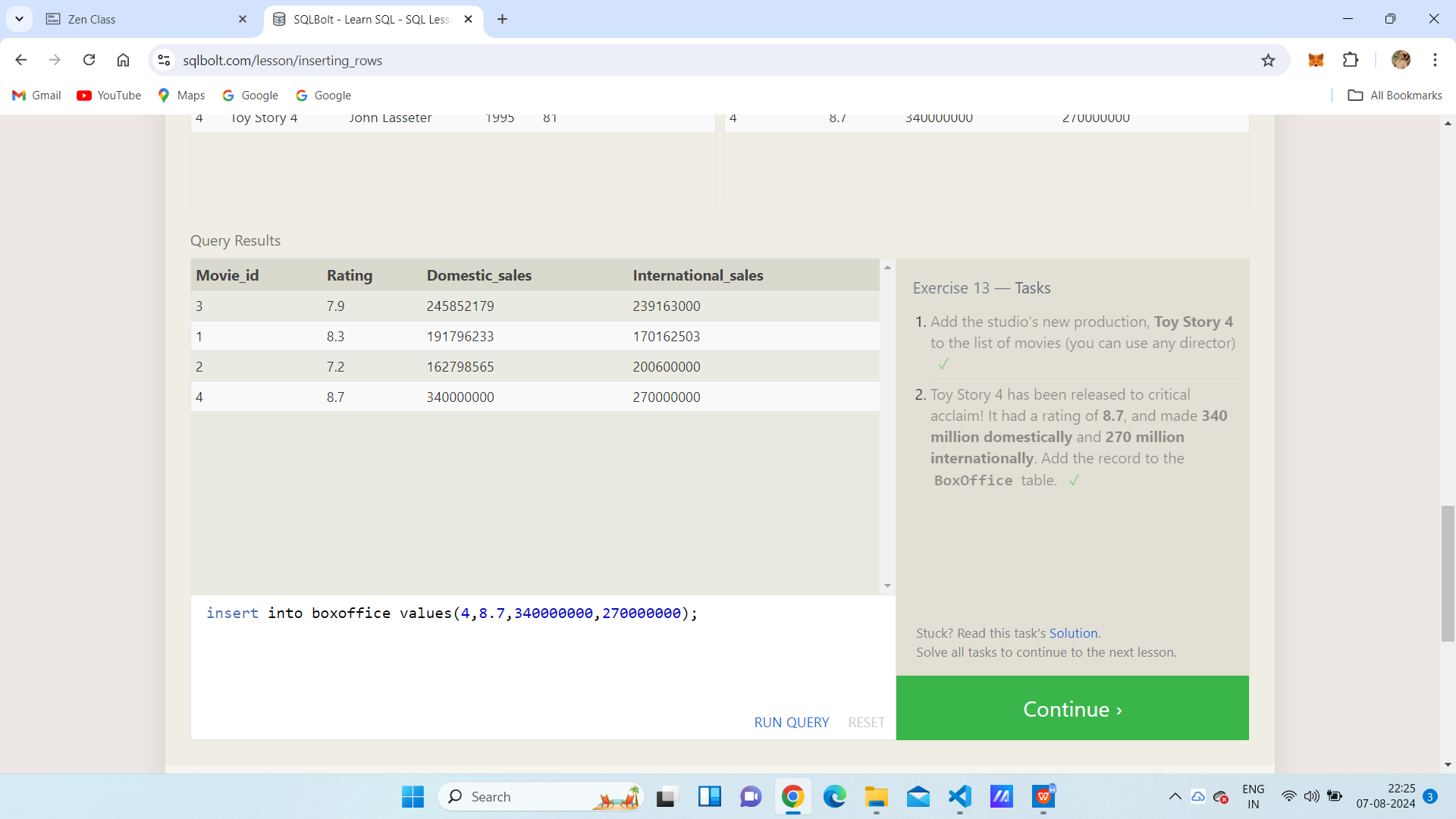
SQL Lesson 12: Order of execution of a Query

1. SELECT director, count(id) as Num\_movies\_directed FROM movies group by director;
2. SELECT director, SUM(domestic\_sales + international\_sales) as Cumulative\_sales\_from\_all\_movies from movies inner join boxoffice on movies.id = boxoffice.movie\_id group by director;



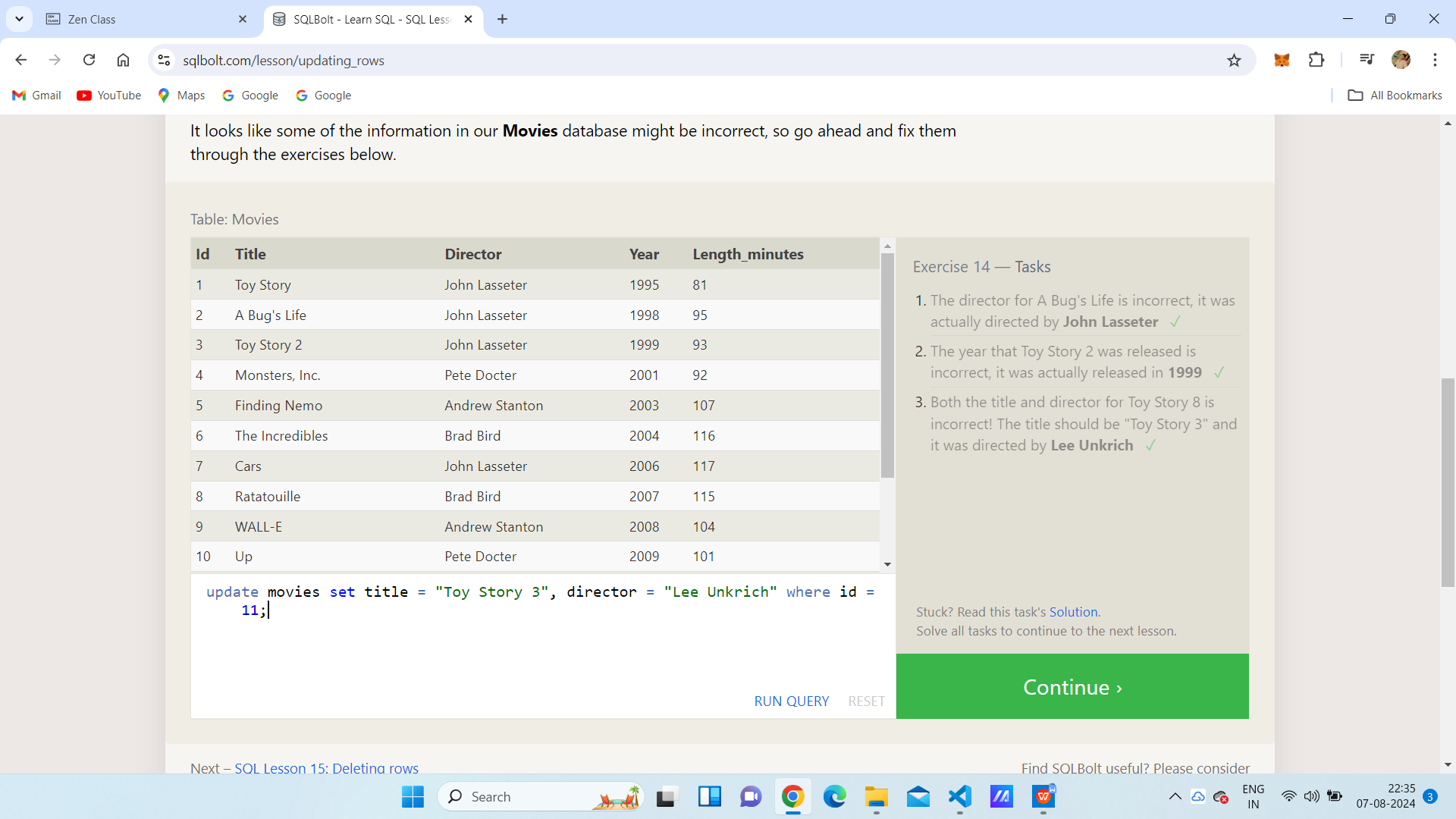
SQL Lesson 13: Inserting rows

1. Insert into movies values(4,”Toy Story 4”,”John Lasseter”,1995,85);
2. insert into boxoffice values(4,8.7,340000000,270000000);



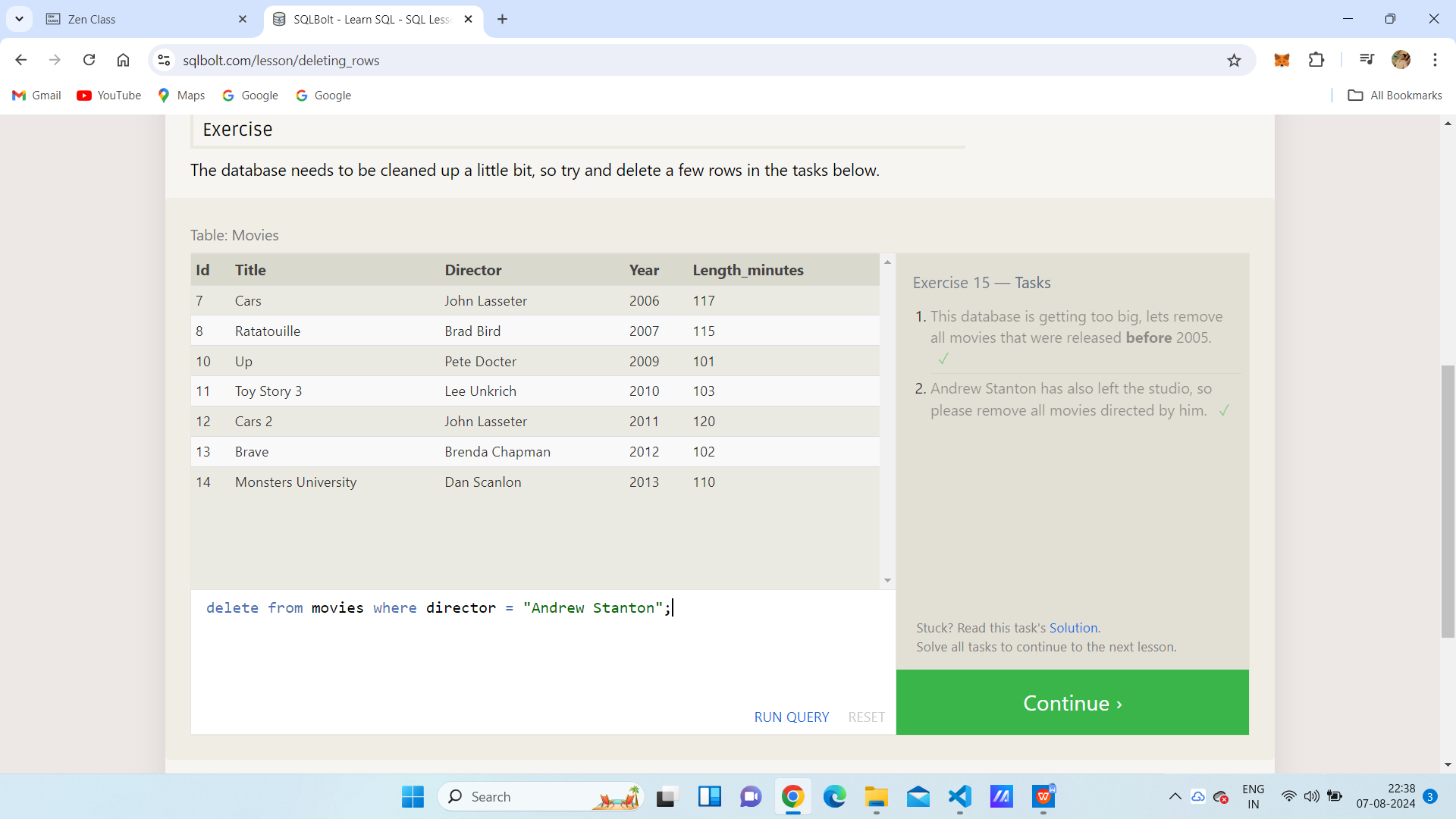
SQL Lesson 14: Updating rows

1. UPDATE movies set director = "John Lasseter" where id = 2;
2. update movies set year = 1999 where id = 3;
3. update movies set title = "Toy Story 3", director = "Lee Unkrich" where id = 11;



SQL Lesson 15: Deleting rows

1. delete from movies where year < 2005;
2. delete from movies where director = "Andrew Stanton";



SQL Lesson 16: Creating tables

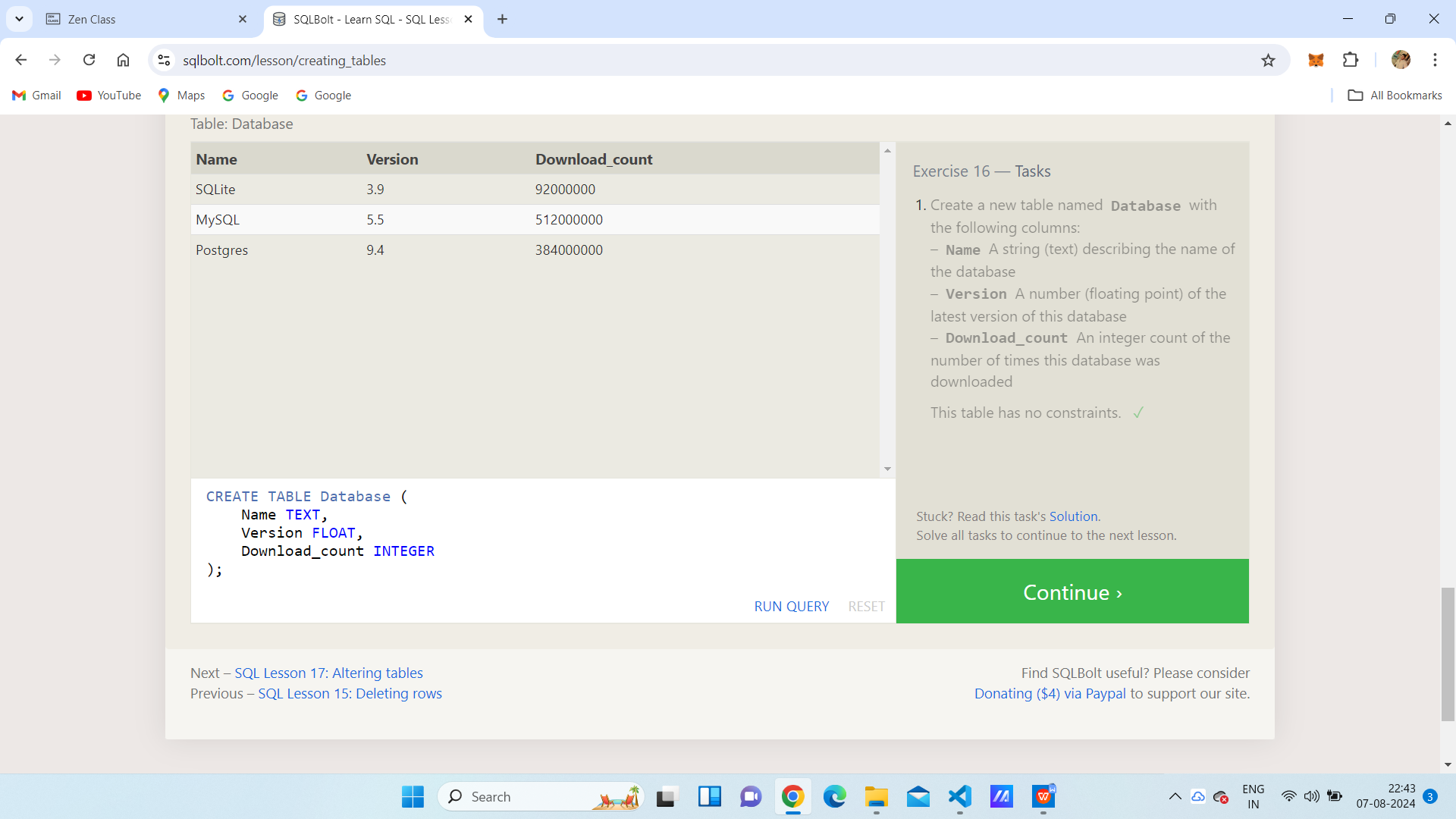
1.CREATE TABLE Database (

Name TEXT,

Version FLOAT,

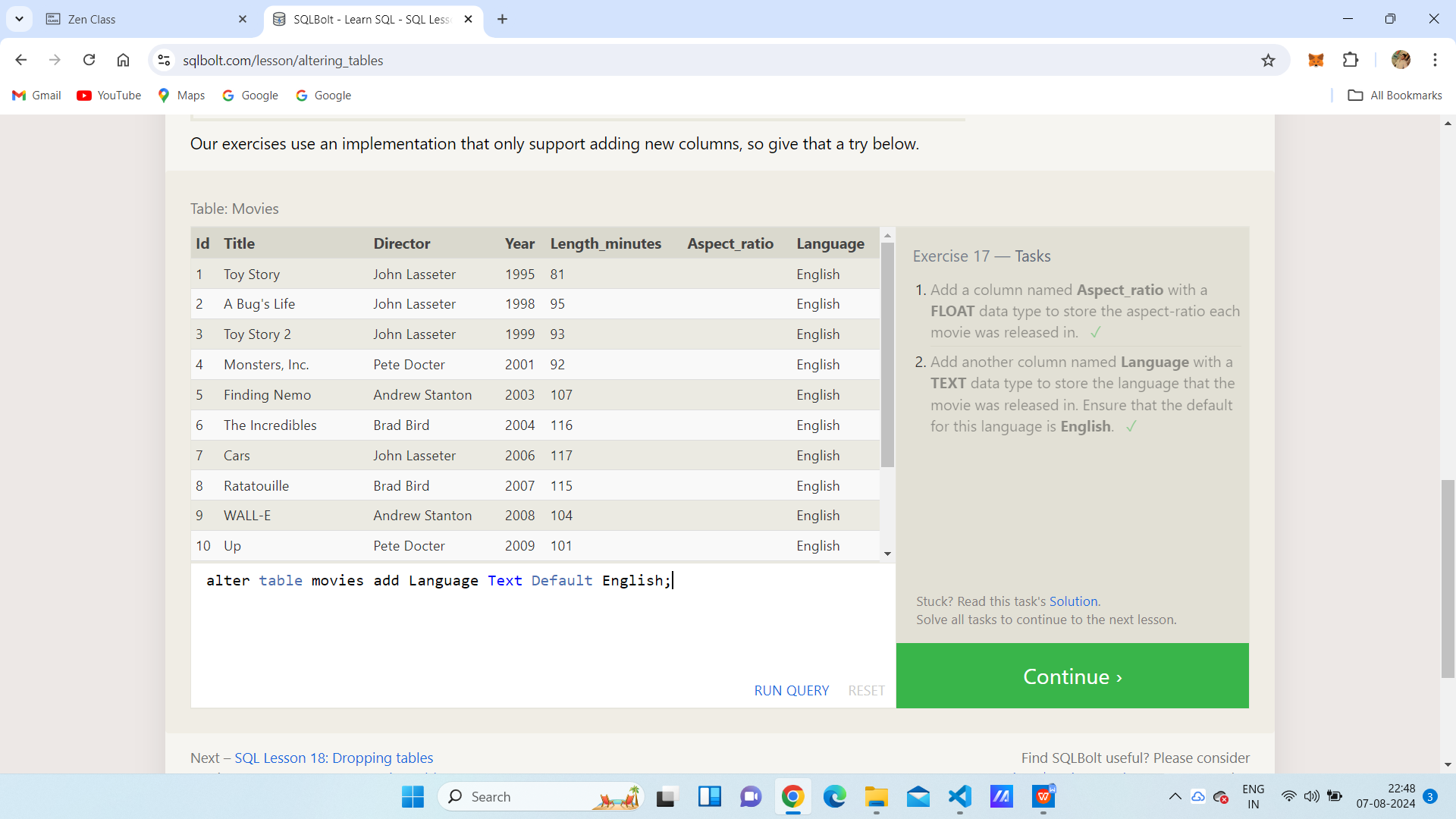
Download\_count INTEGER

);



SQL Lesson 17: Altering tables

1. Alter table movies add Aspect\_ratio Float;
2. alter table movies add Language Text Default English;



SQL Lesson 18: Dropping tables

1. Drop table Movies;
2. Drop table boxoffice;

