**13. INSERTING ROWS:**

1. Add the studio’s new production, **Toy Story 4** to the list of Movies(you can use any director)

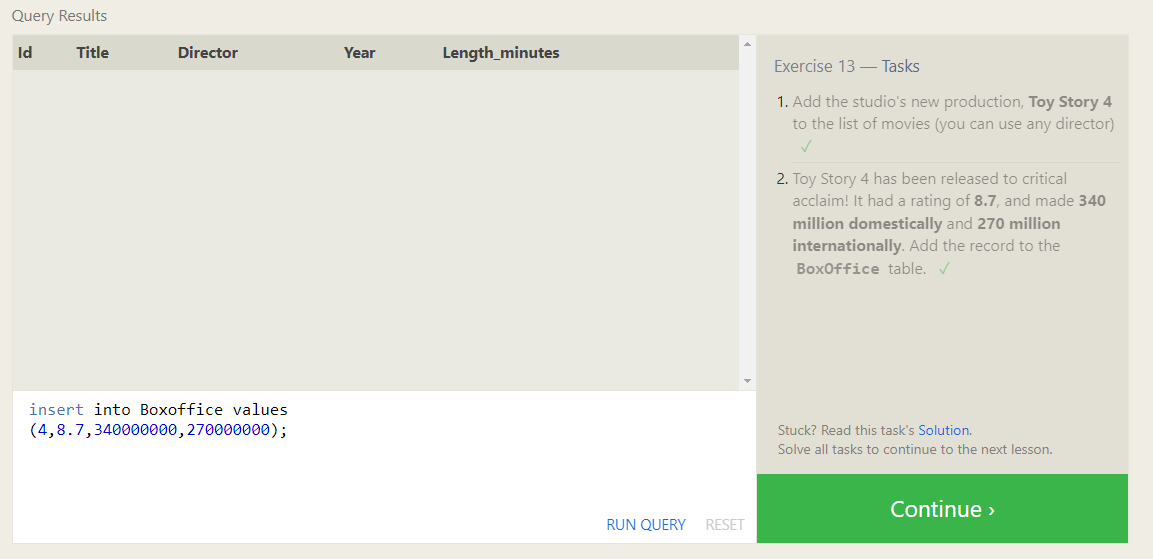
Ans ; **insert into Movies (id, title, director) values**

**(4,'Toy Story4','John Lasseter');**

1. Toy Story 4 has been released to critical acclaim! It had a rating of **8.7**, and made **340 million domestically** and **270 million internationally**. Add the record to the **Box Office** table.

Ans; **insert into Boxoffice values**

**(4,8.7,340000000,270000000);**

****

**14. UPDATING ROWS:**

1. The director for A Bug's Life is incorrect, it was actually directed by **John Lasseter**

Ans: **update Movies**

**SET Director = 'John Lasseter'**

**where Id=2;**

1. The year that Toy Story 2 was released is incorrect, it was actually released in **1999**

Ans: **update Movies**

**SET year = 1999**

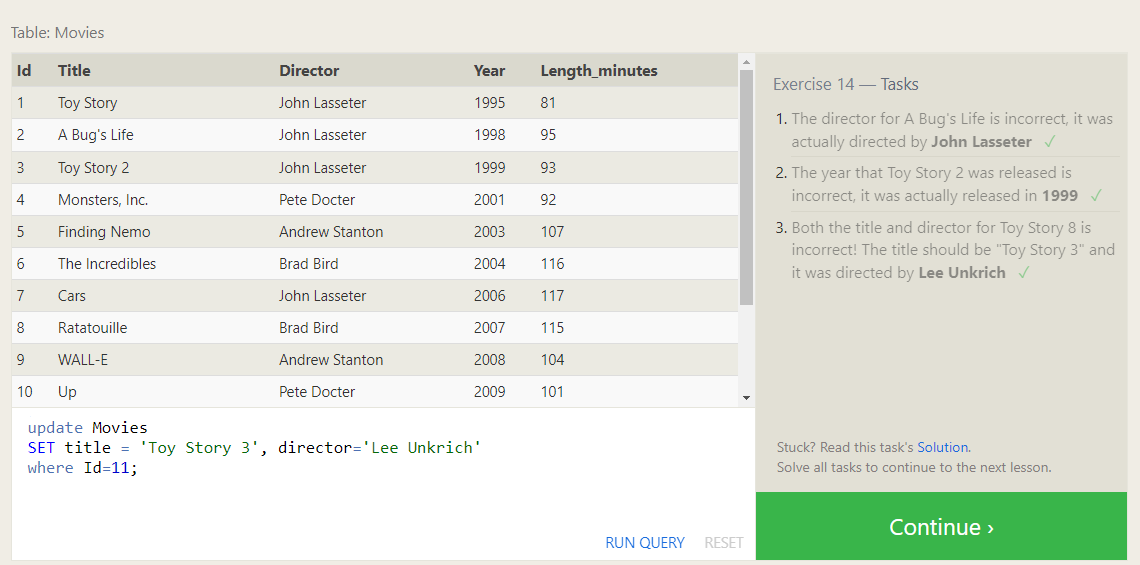
**where Id=3;**

1. Both the title and director for Toy Story 8 is incorrect! The title should be “Toy Story 3” and it was directed by **Lee Unkrich**

Ans: **update Movies**

**SET title = 'Toy Story 3', director='Lee Unkrich'**

**where Id=11;**

****

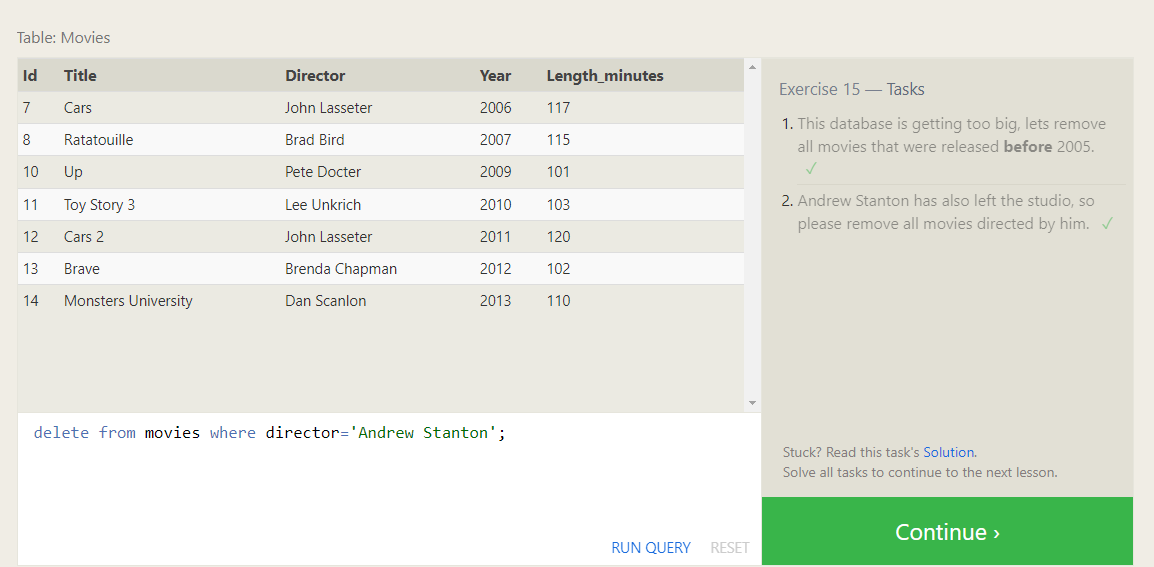
**15. DELETING ROWS:**

1. The data base is getting too big, lets remove all movies that were released **before** 2005

Ans: **delete from movies where year <= 2005;**

1. Andrew Stanton has also left the studio, so please remove all movies directed by him

Ans: **delete from movies where director='Andrew Stanton';**



**16. CREATING TABLES:**

1. Create a table named Database with the following columns;

-Name a string describing the name of the database

-Version a number of the latest version of this database

-Downoad\_count an integer count of the number of times this database was downloaded

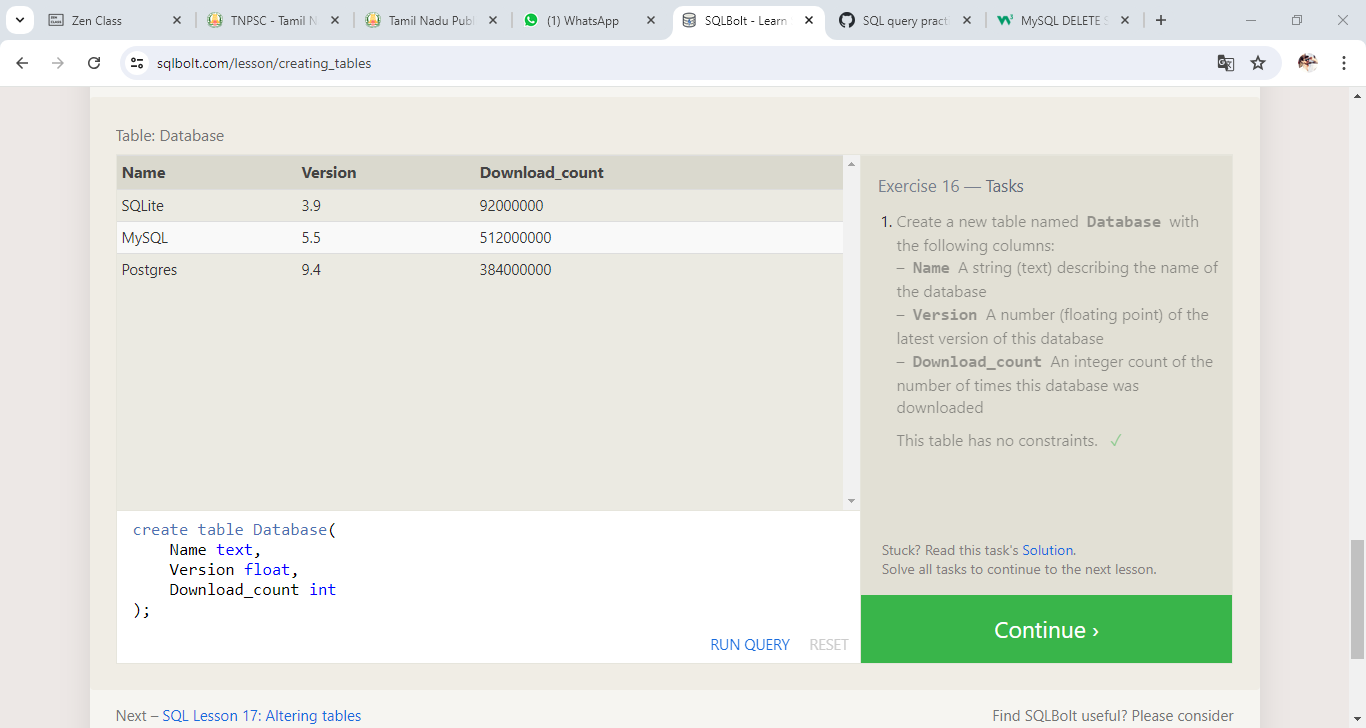
Ans: **create table Database(**

**Name text,**

**Version float,**

**Download\_count int**

**);**

****

**17. ALTERING TABLE:**

1. Add a column named **Aspect \_ratio** with a **FLOAT** data type to store the aspect-ratio each movie was released in.  
     
   Ans: **ALTER table Movies**

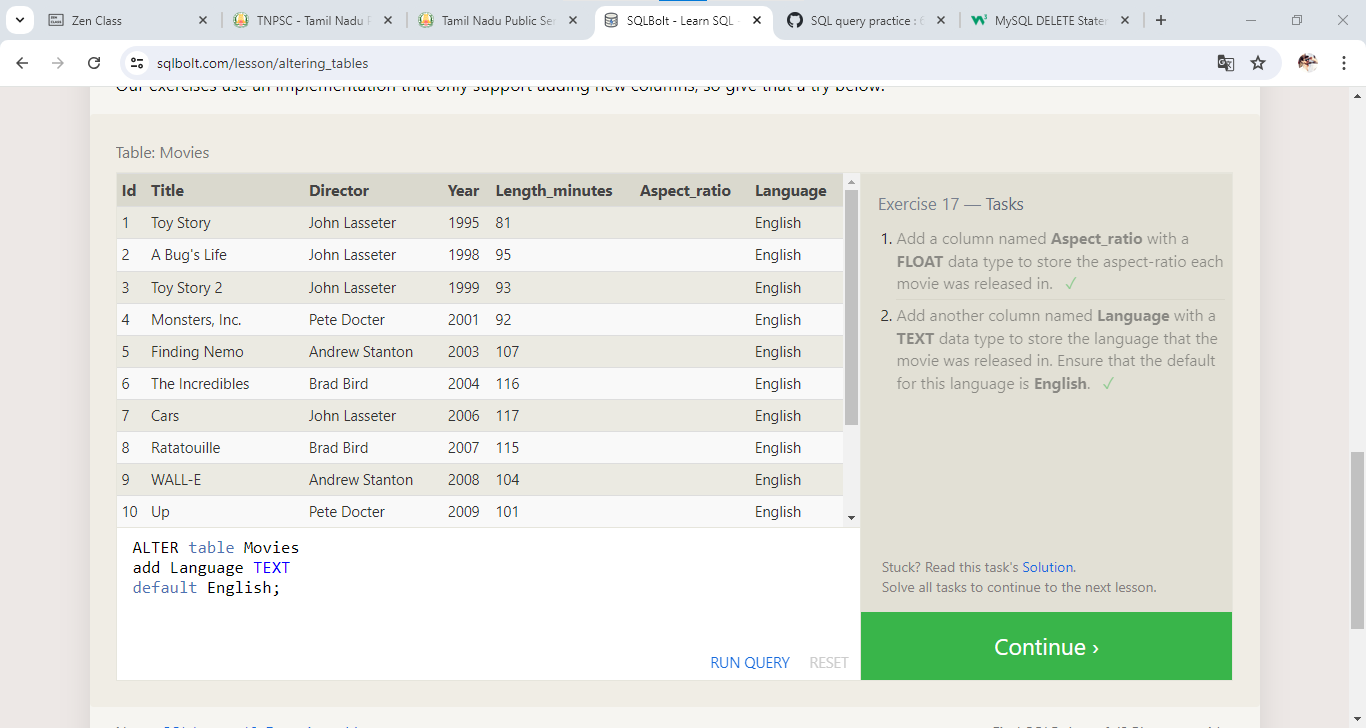
**add Aspect\_ratio float;**

1. Add another column named **Language** with a **TEXT** data type to store the language that the movie was released in. Ensure that the default for this language is **English**

**Ans: ALTER table Movies**

**add Language TEXT**

**default English;**

****

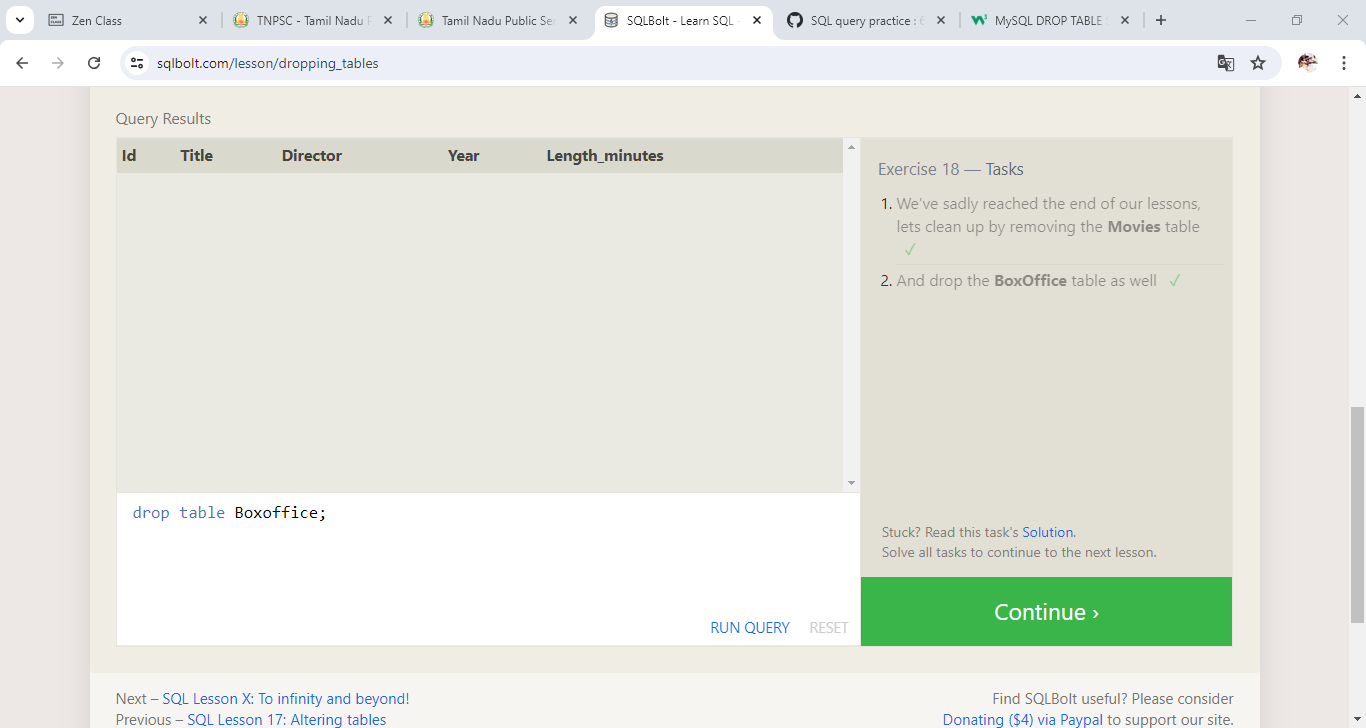
**18. DROPING TABLE:**

1. We’ve sadly reached the end of our lessons, lets clean up by removing the **Movies** table

Ans: **drop table Movies;**

1. Drop the box-office table as well

Ans: **drop table Boxoffice;**



**1.SELECT QUERIES 101:**

1. Find the title of each film

Ans; **select title from movies;**

1. Find the director of each film

**Ans: select director from movies;**

1. Find the title and director of each film

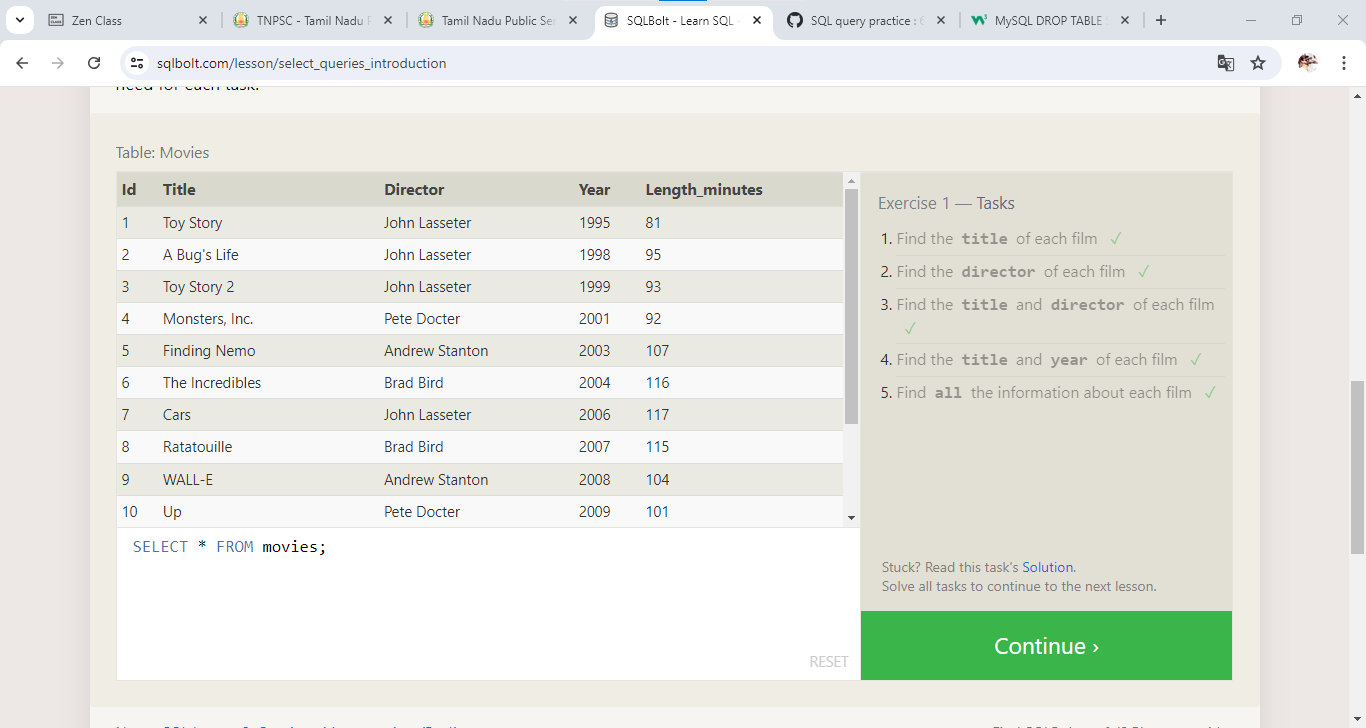
**Ans: select title, director from movies;**

1. Find the title and year of each film

**Ans: select title, year from movies;**

1. Find all the information about each film

**Ans: select \* from movies;**

****

**2.QUERIES WITH CONSTRAINTS:**

=> Find a movie with a row **id** of 6

Ans**: SELECT \* FROM movies where id =6;**

=> Find the movie released in the years between 2000 and 2010

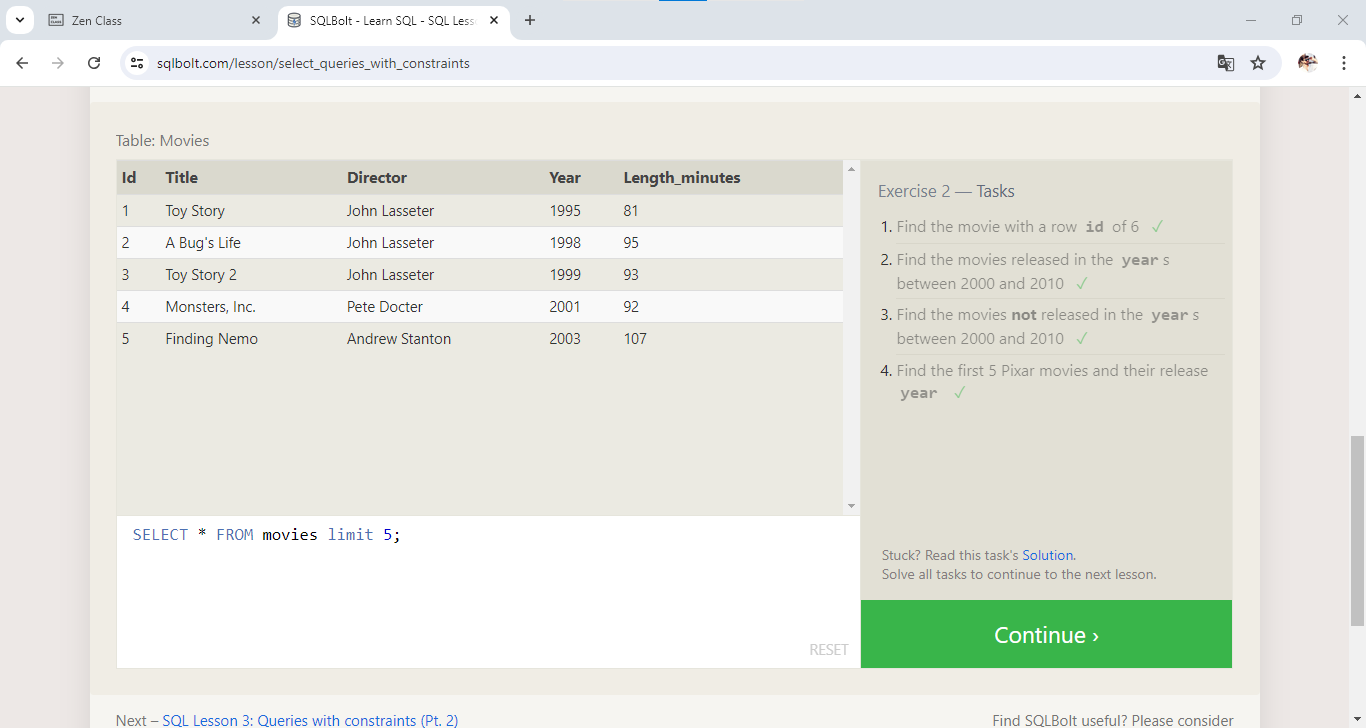
Ans **: SELECT \* FROM movies where year between 2000 and 2010;**

=> Find the movies not released in the years between 2000 and 2010

Ans: **SELECT \* FROM movies where year not between 2000 and 2010;**

=>Find the first pixar movies and their release year

Ans: **SELECT \* FROM movies limit 5;**

****

1. **QUERIES WITH CONSTRAINTS 2**
2. Find all the Toy Story movies

Ans: **SELECT \* FROM movies where title like '%toy story%';**

1. Find all the movies directed by john Lasseter

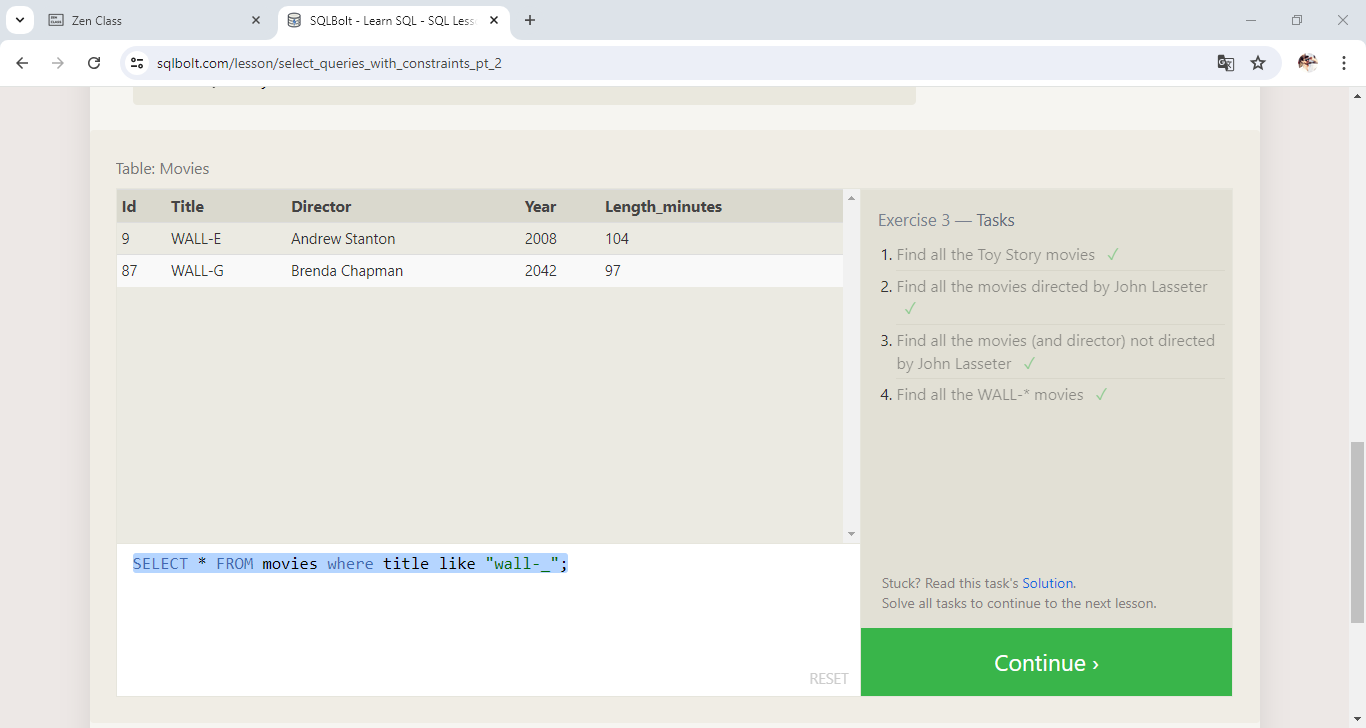
Ans**: SELECT \* FROM movies where director like 'john lasseter';**

1. Find all the movies (and director) not directed by john Lasseter

Ans: **SELECT \* FROM movies where director not like 'john lasseter';**

1. Find all the WALL-\* movies

Ans: **SELECT \* FROM movies where title like "wall-\_";**



**4. Filtering and sorting query results:**

1. List all directors of pixar movies(alphabetically), without duplicates

Ans: **SELECT distinct director from movies order by director asc;**

2. List all four pixar movies released (order from most recent to lease)

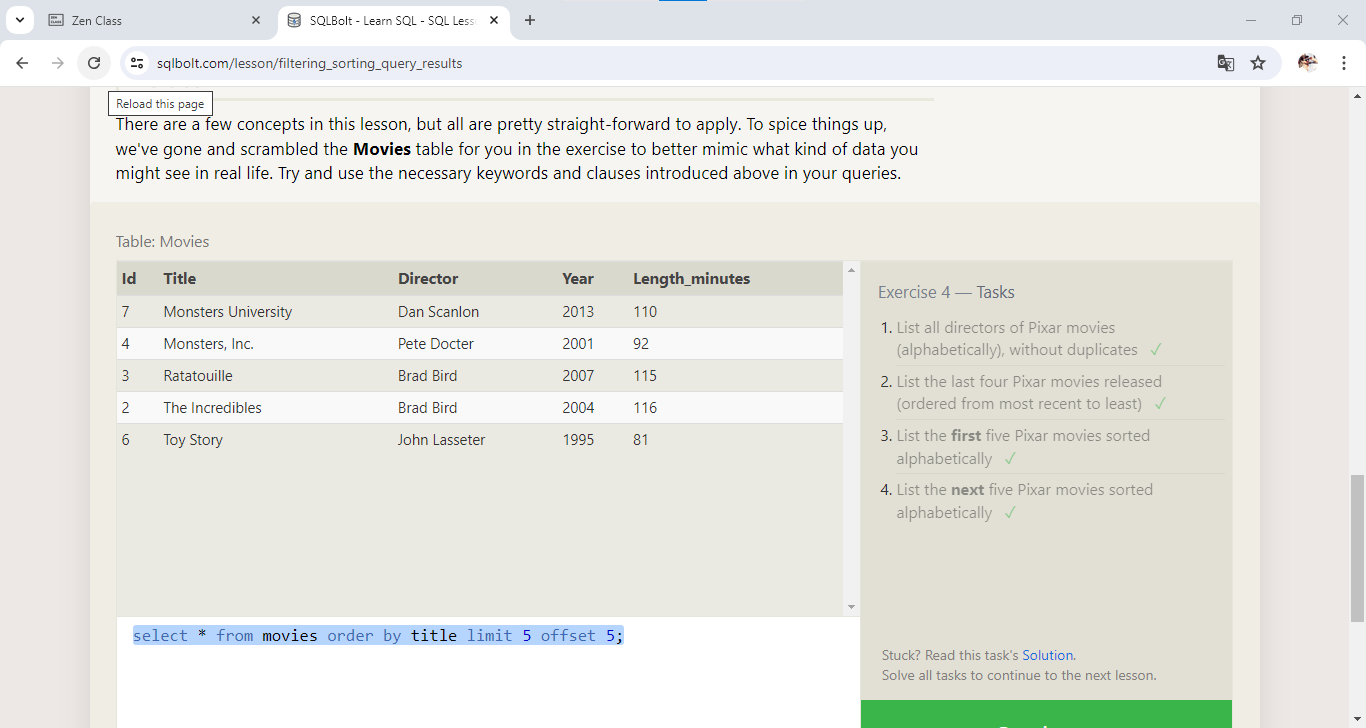
Ans**: SELECT \* from movies order by year desc limit 4;**

3. List the first five pixar movies sorted alphabetically

Ans**: SELECT \* from movies order by title limit 5;**

4. List the next five pixar movies sorted alphabetically

Ans: **SELECT \* from movies order by title limit 5 offset 5;**



**5. simple SELECT queries:**

1. list all the Canadian cities and their population

Ans: **SELECT city, population from North\_american\_cities where country = 'Canada';**

2. Order all the cities in the United States by their latitude from north to south

Ans: **SELECT \* from North\_american\_cities where country = 'United States' order by latitude desc;**

3. List all the cities west of Chicago, ordered from west to east

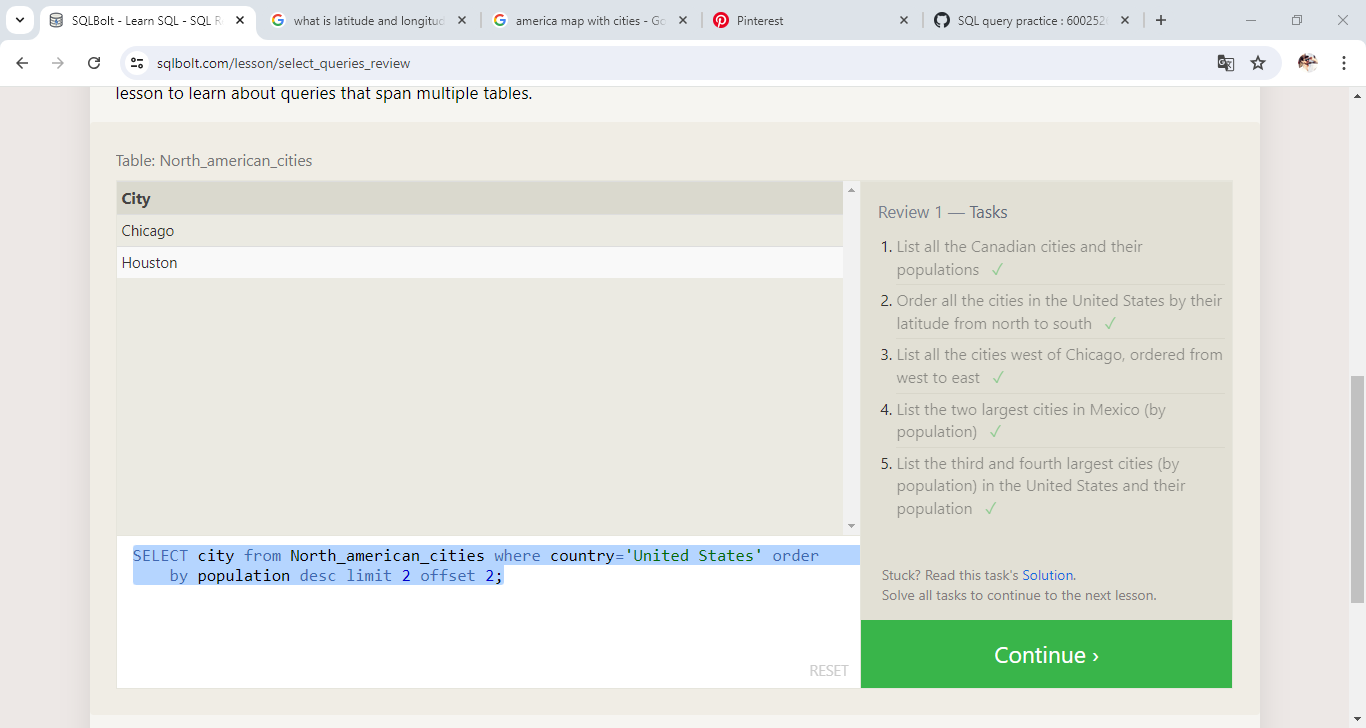
Ans: **SELECT \* from North\_american\_cities where longitude < -87.629798 order by longitude;**

4. List the two largest cities in Mexico(by population)

Ans: **SELECT city from North\_american\_cities where country='Mexico' order by population desc limit 2;**

5. List the third and fourth largest cities(by population) in the United States and their population

Ans: **SELECT city from North\_american\_cities where country='United States' order by population desc limit 2 offset 2;**

****

**6. MULTI TABLE QUERIES WITH JOINS:**

1. Find the domestic and international sales for each movie

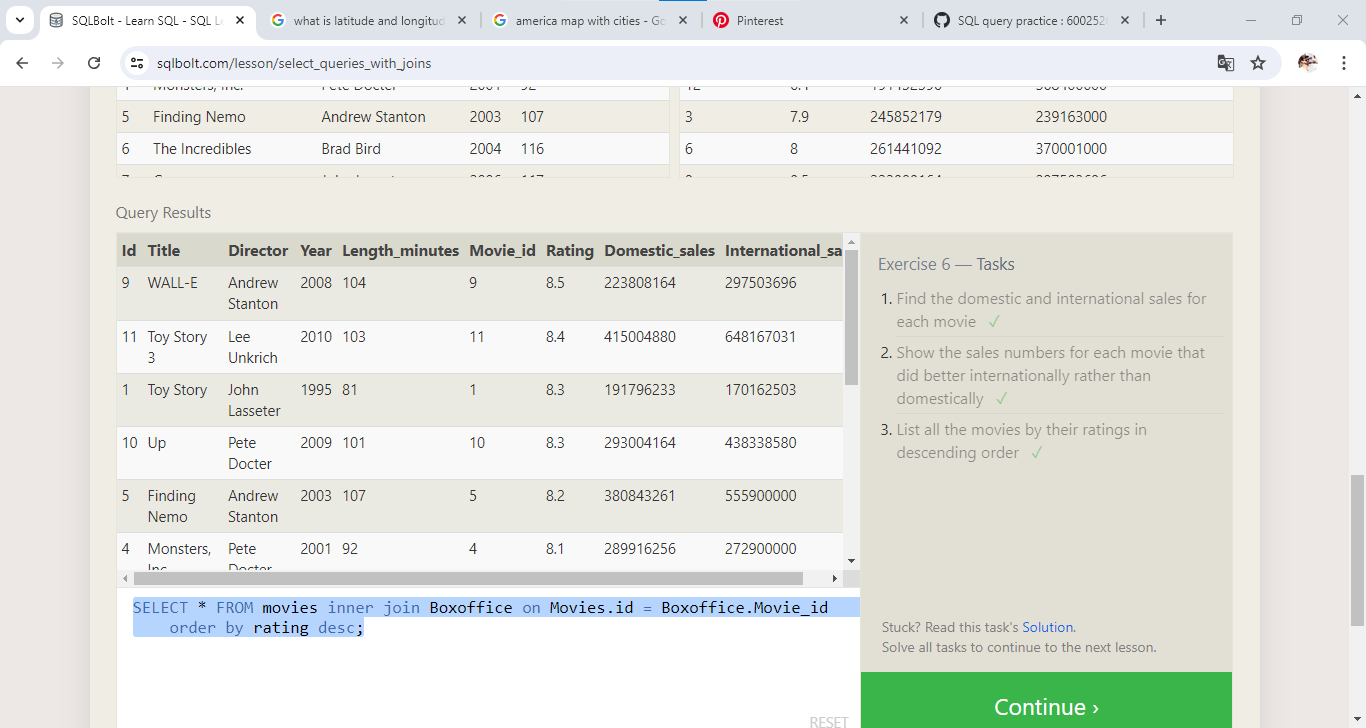
Ans: **SELECT \* FROM movies inner join Boxoffice on Movies.id = Boxoffice.Movie\_id order by id;**

2. Show the sales numbers for each movie that did better internationally rather than domestically

Ans: **SELECT \* FROM movies inner join Boxoffice on Movies.id = Boxoffice.Movie\_id where international\_sales>Domestic\_sales order by international\_sales desc;**

3. List all the movies by their ratings in desc

Ans: **SELECT \* FROM movies inner join Boxoffice on Movies.id = Boxoffice.Movie\_id order by rating desc;**



**7. OUTER JOIN**

1. Find the list of all buildings that have employees

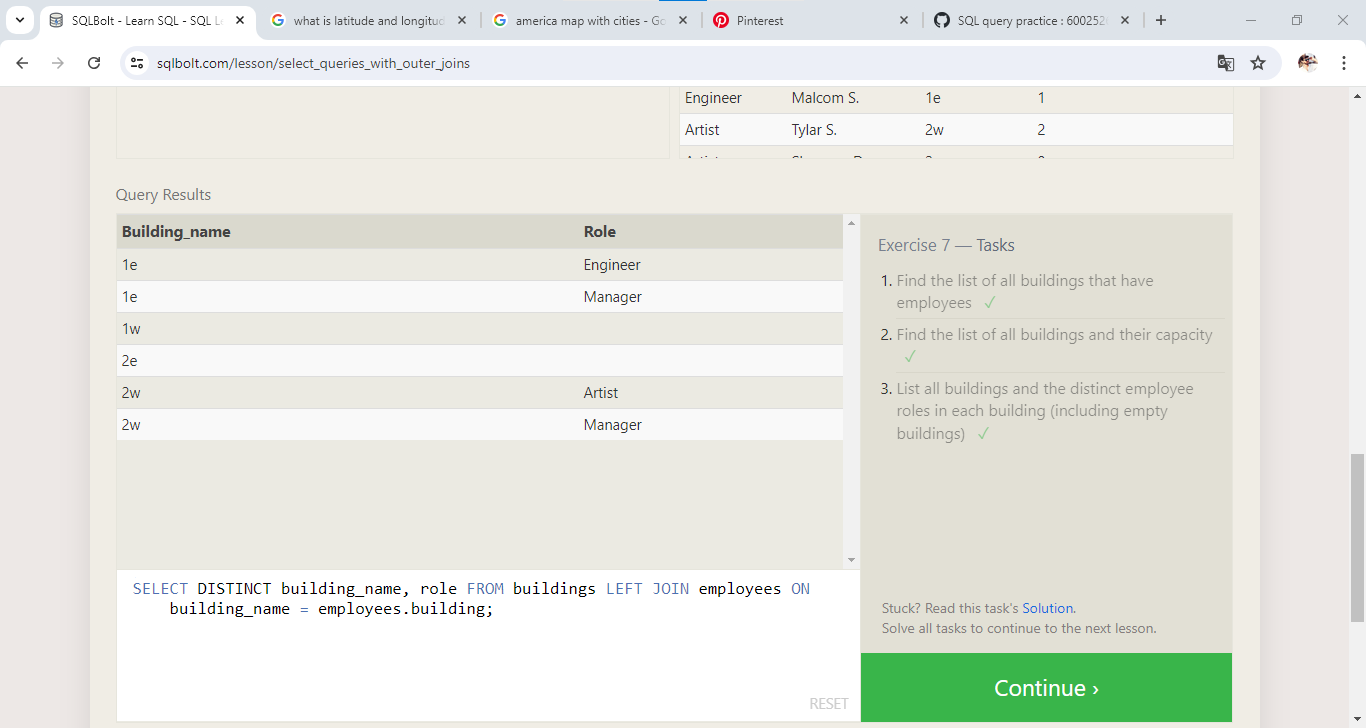
Ans: **SELECT distinct building FROM employees;**

2. Find the list of all buildings and their capacity

Ans: **SELECT \* FROM buildings;**

3. List all buildings and the distinct employee roles in each building (including empty buildings)

Ans: **SELECT DISTINCT building\_name, role FROM buildings LEFT JOIN employees ON building\_name = employees.building;**



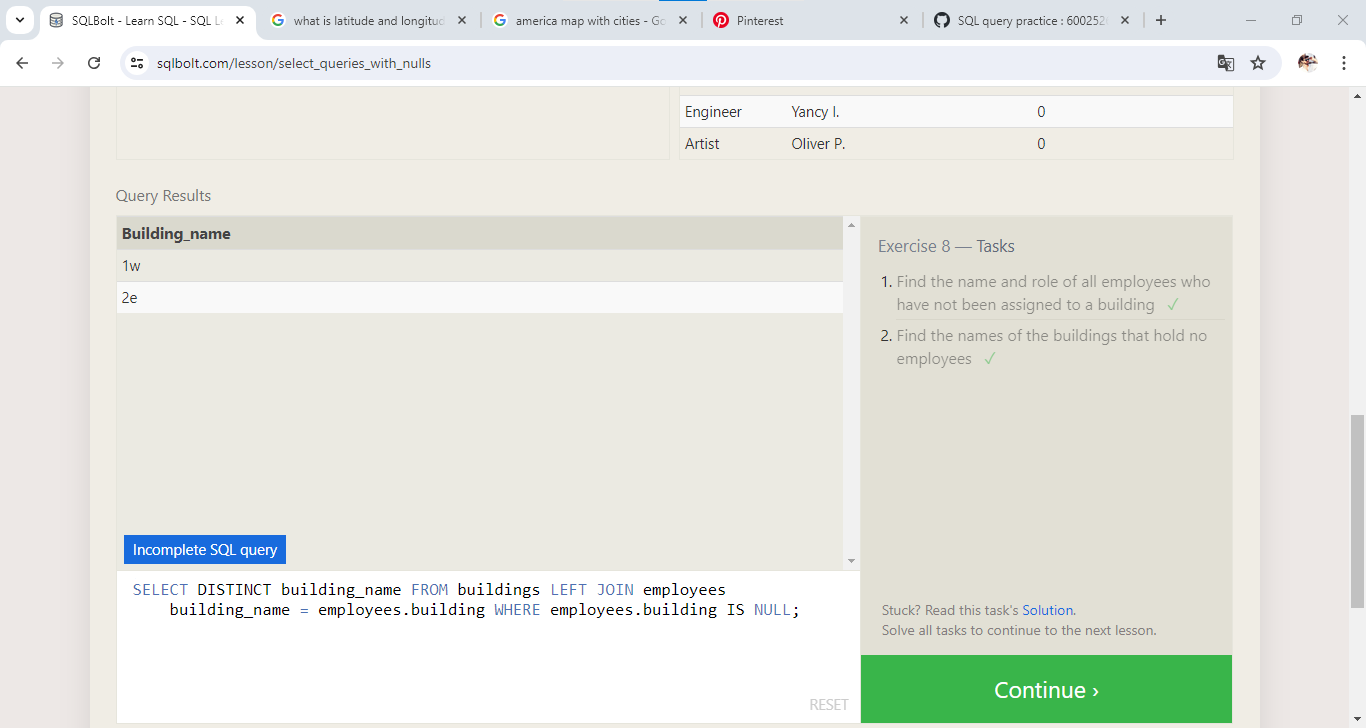
**8. SHORT NOT ON NULLS**

1. Find the name and role of all employees who have not been assigned to a building

Ans: **SELECT role, name FROM employees where building is null;**

2. Find the names of the buildings that hold no employees

Ans: **SELECT DISTINCT building\_name FROM buildings LEFT JOIN employees building\_name = employees.building WHERE employees.building IS NULL;**

****

**9. QUERIES WITH EXPRESSION**

1. List all movies and their combined sales in millions of dollars

Ans: **SELECT DISTINCT**

**title,**

**(domestic\_sales + international\_sales) / 1000000 AS sales**

**FROM movies**

**INNER JOIN boxoffice**

**ON movies.id = boxoffice.movie\_id;**

2. List all movies and their ratings in percent

Ans: **SELECT DISTINCT**

**title,**

**(rating \* 10) AS rate\_percent**

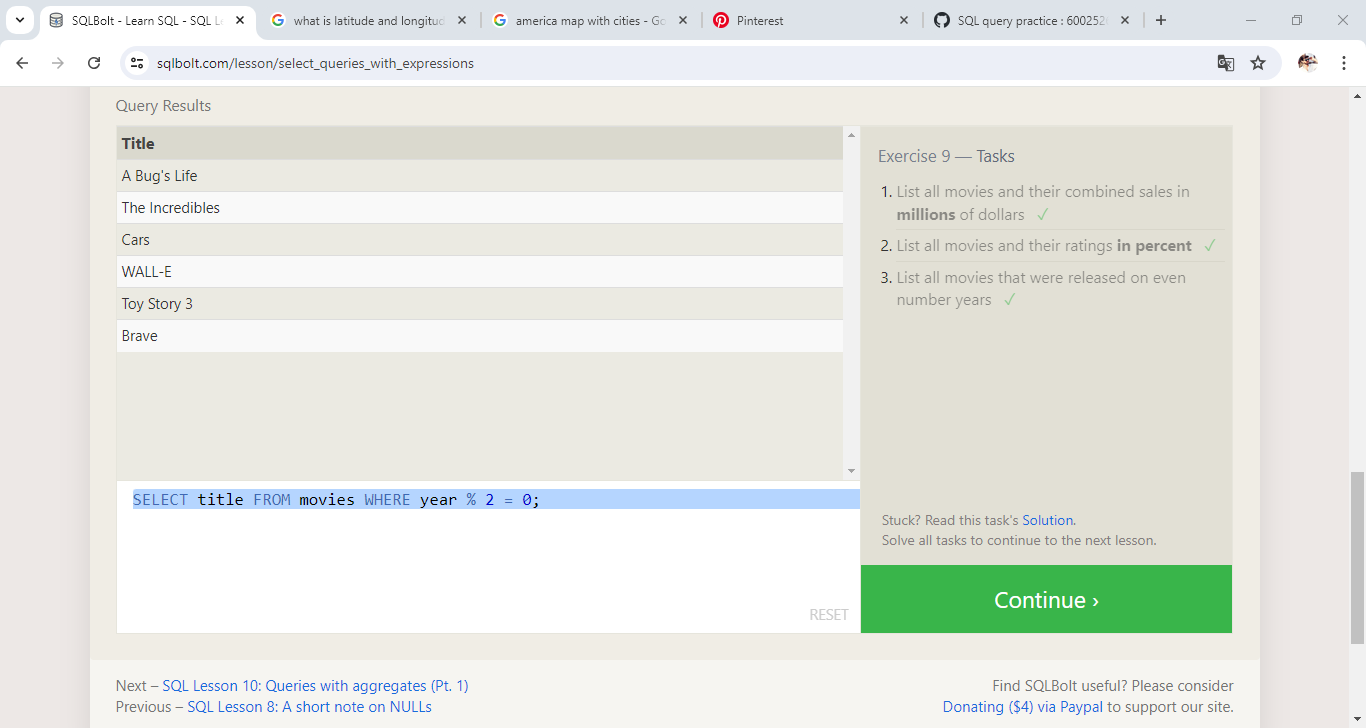
**FROM movies**

**INNER JOIN boxoffice**

**ON movies.id = boxoffice.movie\_id;**

3. List all movies that were released on even number years

Ans: **SELECT title FROM movies WHERE year % 2 = 0;**



**10. QUERIES WITH AGGREGATES**

1. Find the longest time that an employee has been at the studio

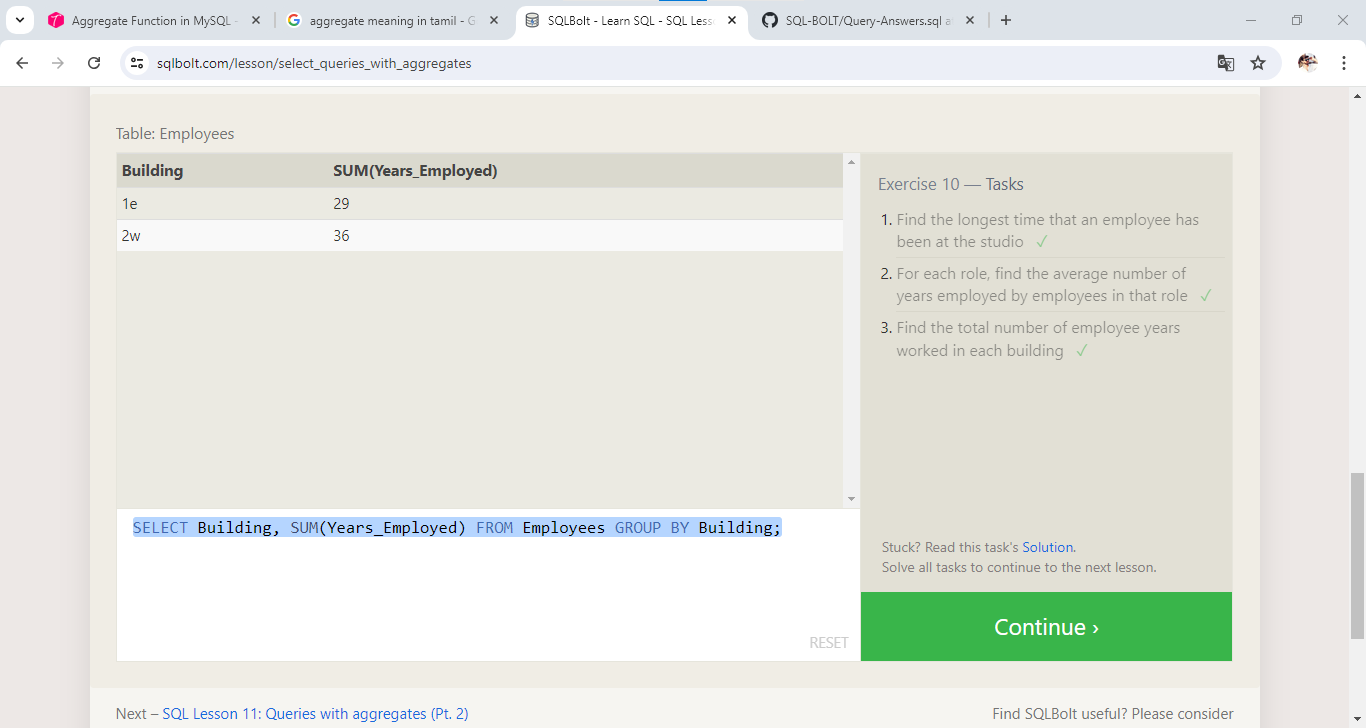
Ans: **SELECT MAX(Years\_employed) FROM Employees;**

2. For each role, find the average number of years employed by employees in that role

Ans**: SELECT Role, AVG(Years\_Employed) FROM Employees GROUP BY Role;**

3. Find the total number of employee years worked in each building

Ans: **SELECT Building, SUM(Years\_Employed) FROM Employees GROUP BY Building;**

****

**11. Queries with aggregates**

1. Find the number of Artists in the studio (without a HAVING clause)

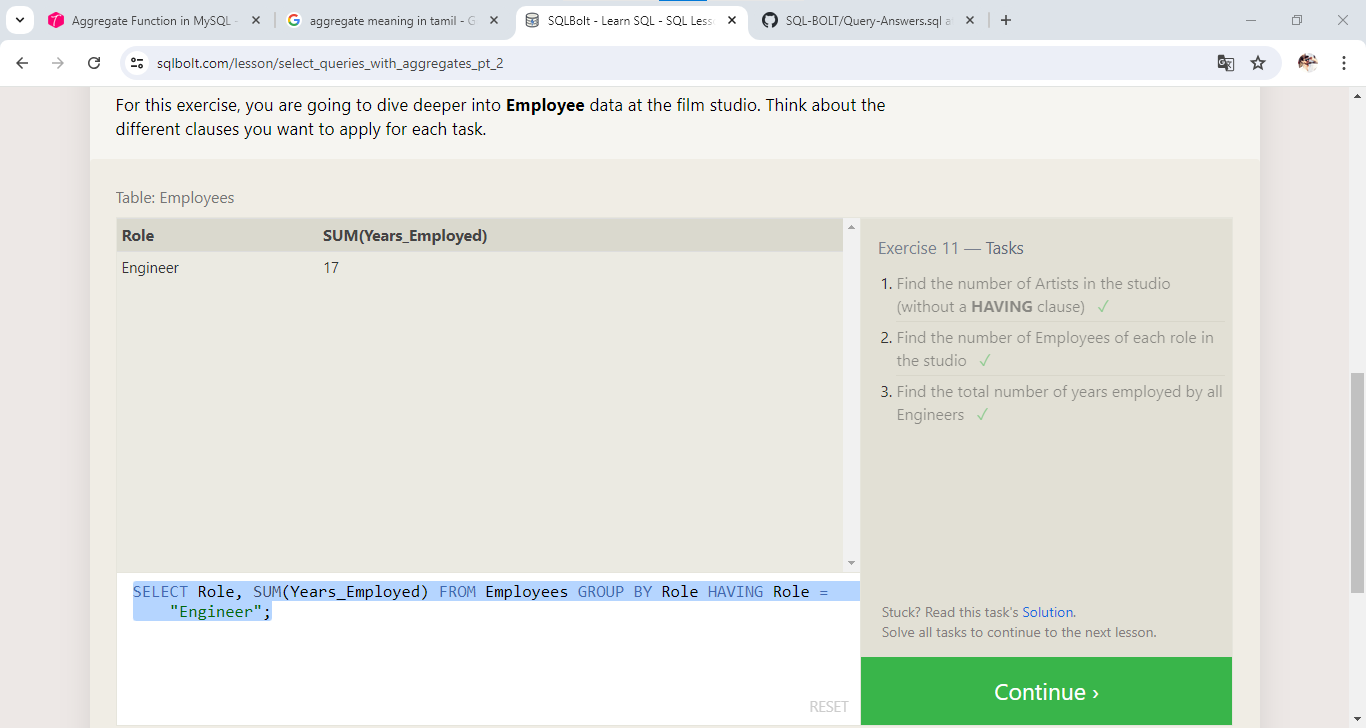
Ans: **SELECT Role, COUNT(\*) AS Number\_of\_Artists FROM Employees WHERE Role = "Artist";**

2. Find the number of Employees of each role in the studio

Ans: **SELECT Role, COUNT(\*) FROM Employees GROUP BY Role;**

3. Find the total number of years employed by all Engineers

Ans: **SELECT Role, SUM(Years\_Employed) FROM Employees GROUP BY Role HAVING Role = "Engineer";**

****

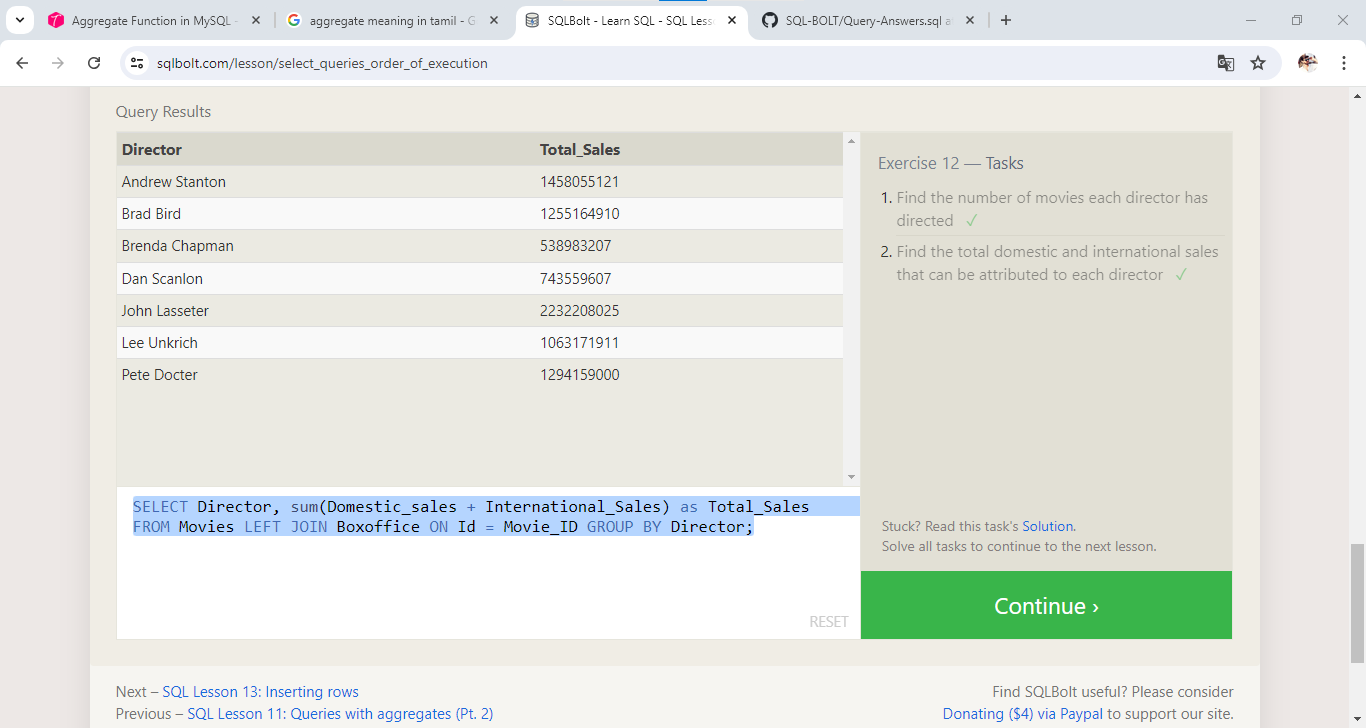
**12. ORDER OF EXECUTION OF A QUERY**

1. Find the number of movies each director has directed

Ans: **SELECT \*, COUNT(Title) FROM Movies GROUP BY Director;**

2. Find the total domestic and international sales that can be attributed to each director

Ans: **SELECT Director, sum(Domestic\_sales + International\_Sales) as Total\_Sales FROM Movies LEFT JOIN Boxoffice ON Id = Movie\_ID GROUP BY Director;**

****