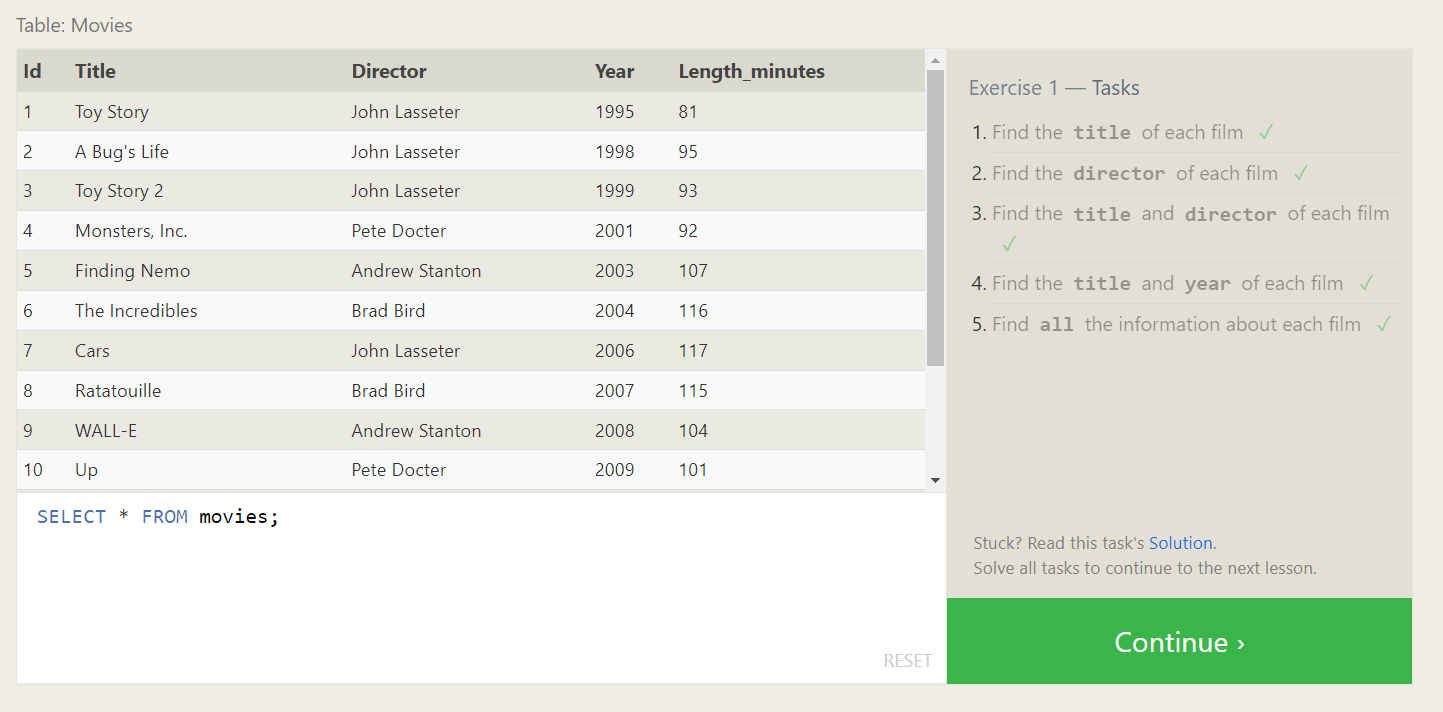
# MYSQL Day-1 Task

**SQL Lesson 1: SELECT queries**



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**



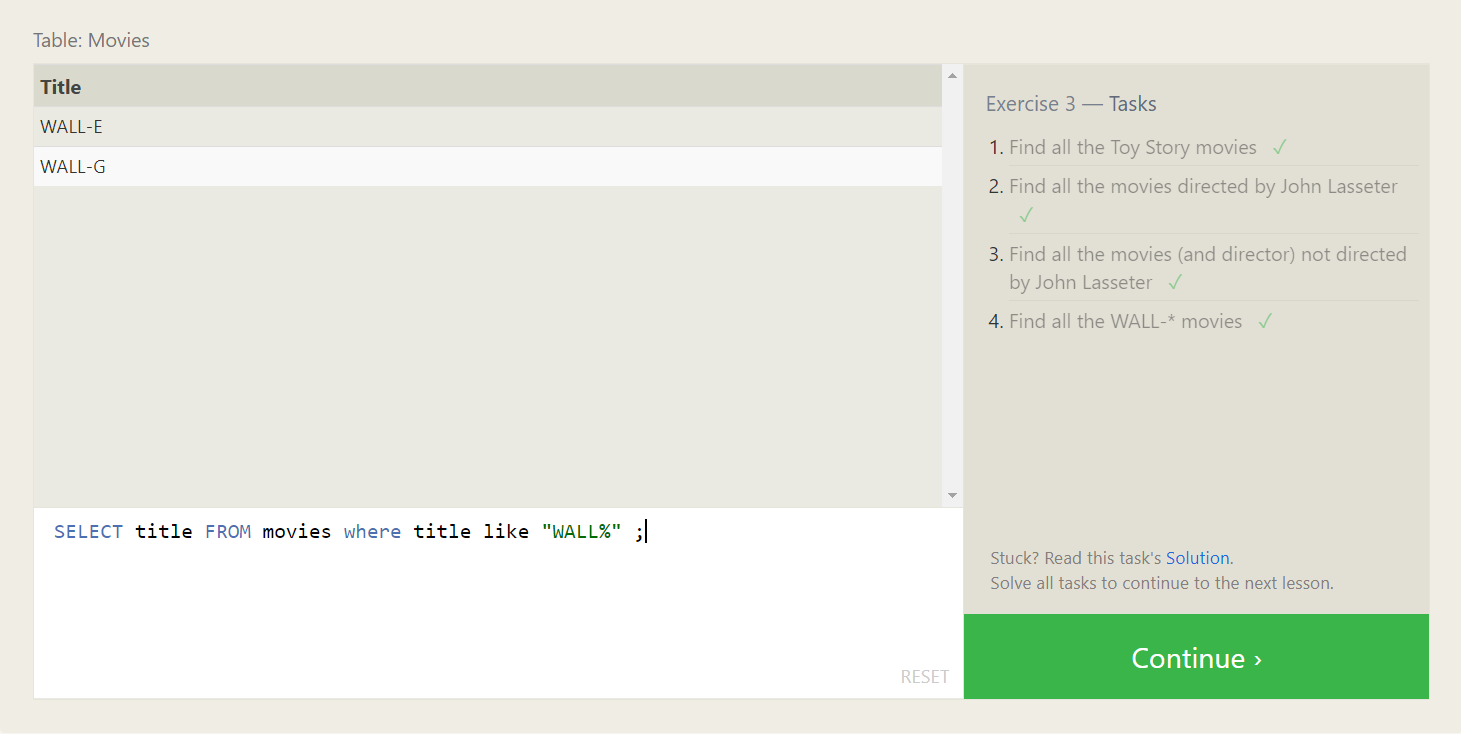
1. select title from movies where id=6

2. select title from movies where year between 2000 and 2010

3. select title from movies where year not between 2000 and 2010

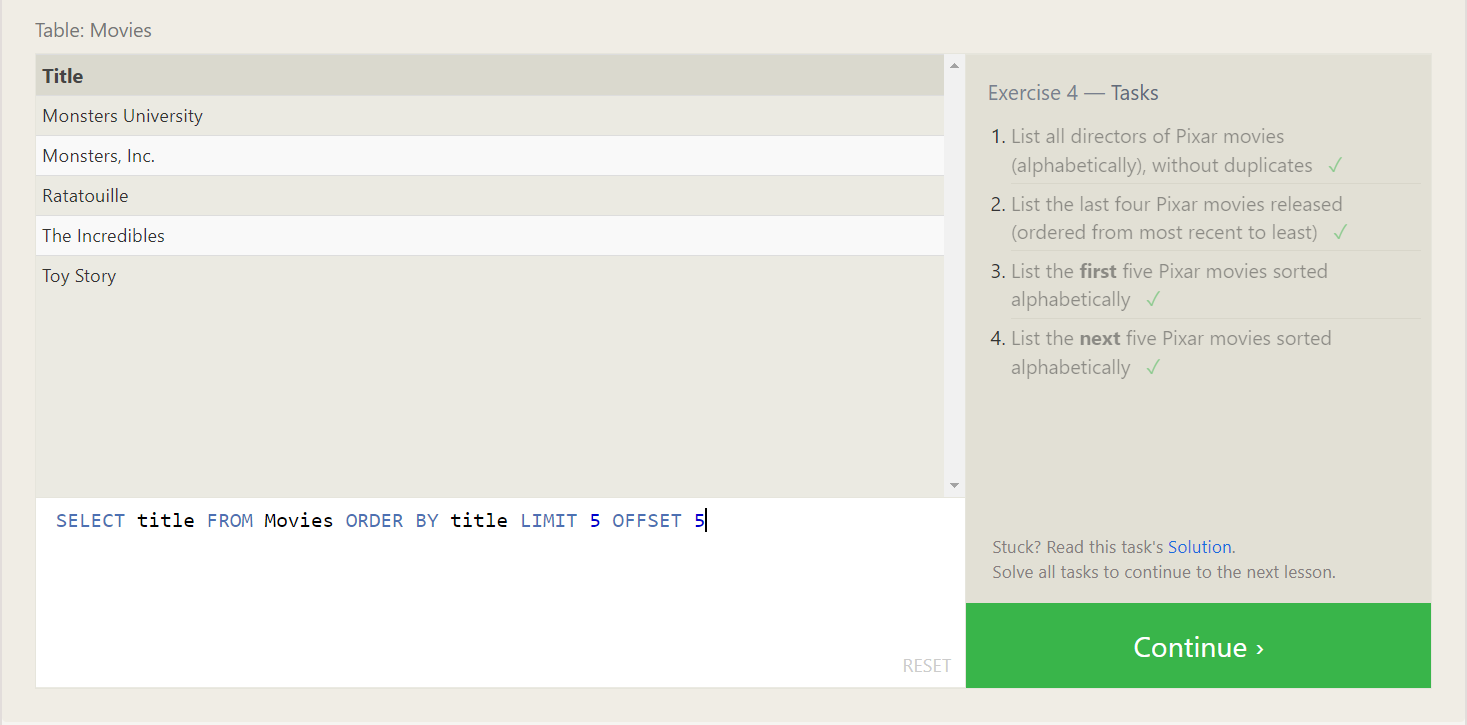
4. select title, year from movies where id between 1 and 5

**SQL Lesson 3: Queries with constraints**



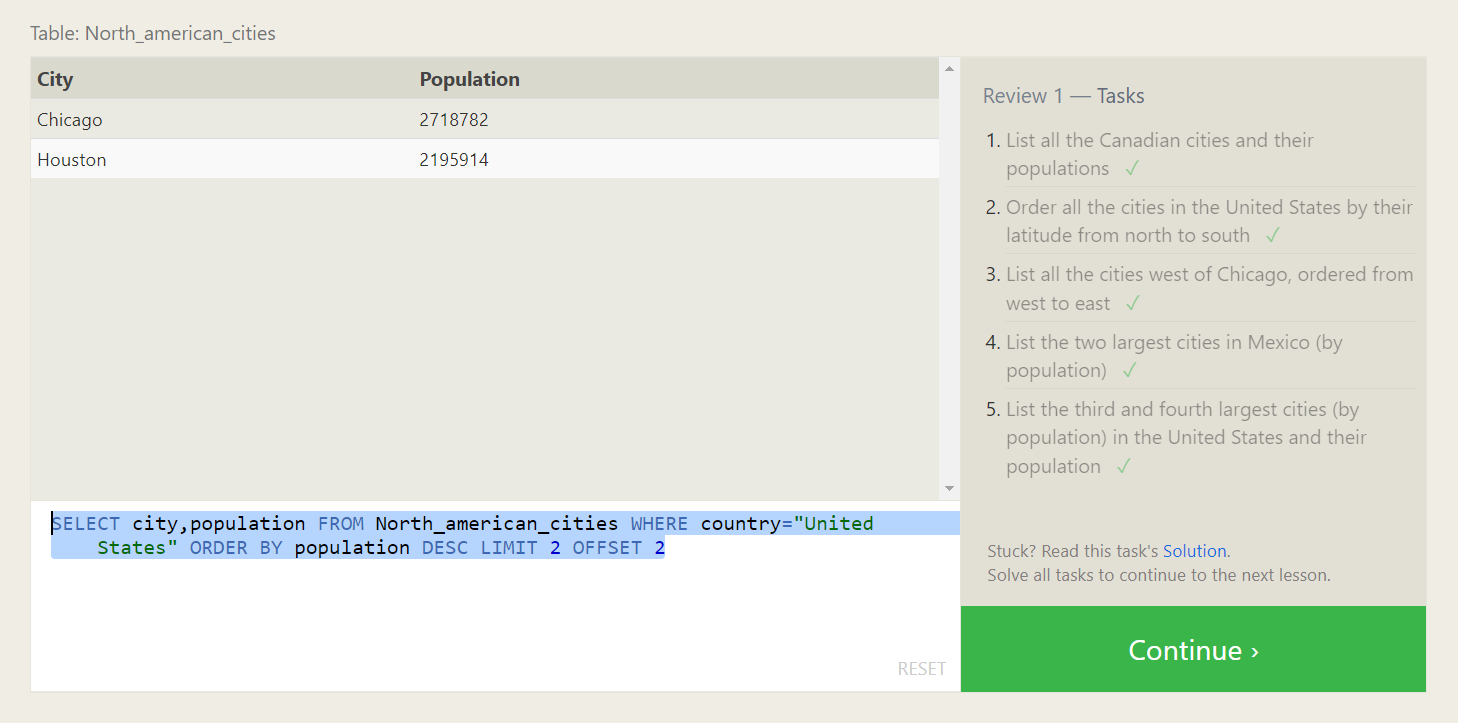
1. SELECT title FROM movies where title like "Toy%"
2. SELECT title FROM movies where director="John Lasseter"
3. SELECT title FROM movies where director != "John Lasseter"
4. SELECT title FROM movies where title like "WALL%"

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



1. SELECT distinct director FROM movies order by director
2. SELECT title FROM movies order by year desc limit 4
3. SELECT title FROM Movies order by title limit 5
4. SELECT title FROM Movies ORDER BY title LIMIT 5 OFFSET 5

**SQL Review: Simple SELECT Queries**

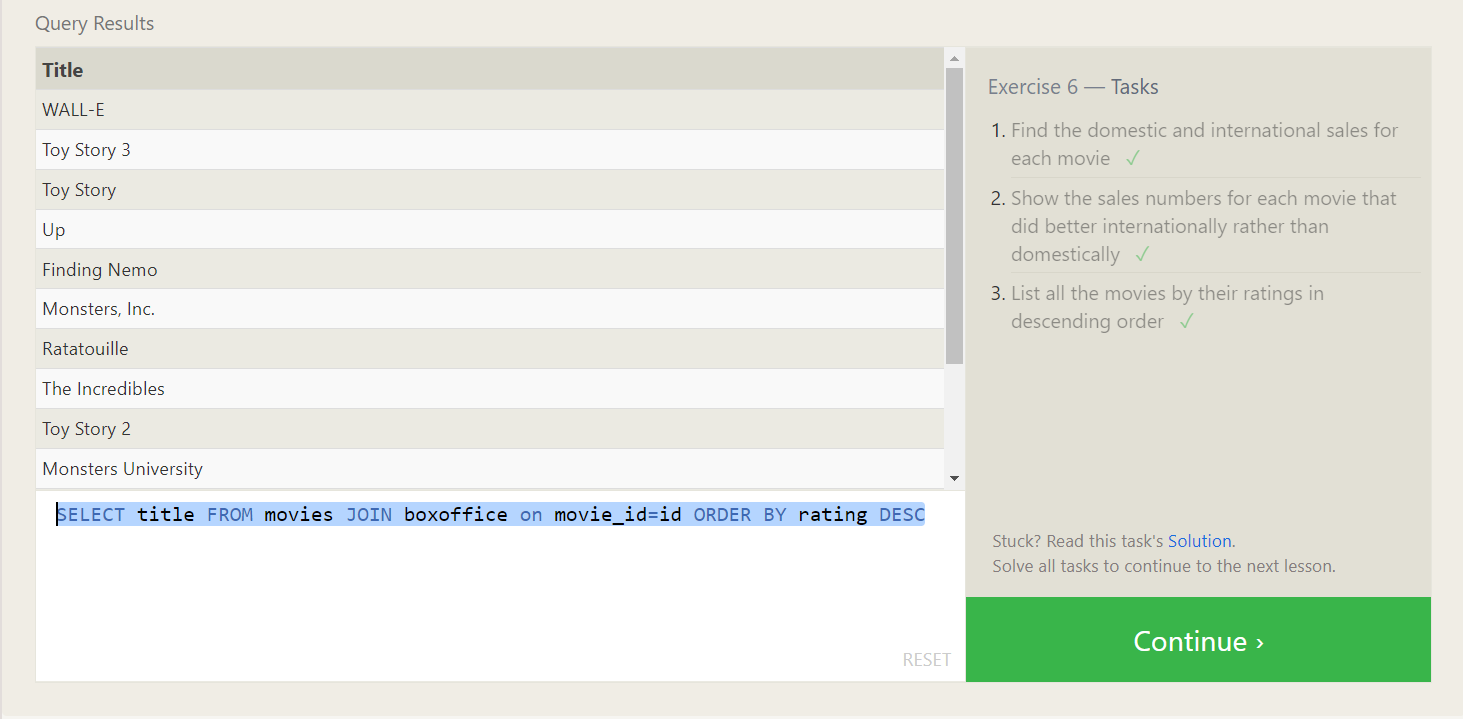


1. SELECT city, population FROM north\_american\_cities where country = "Canada"
2. SELECT CITY FROM north\_american\_cities where country = "United States" order by latitude desc
3. select city from North\_american\_cities where longitude<

(select Longitude from North\_american\_cities where city = "Chicago" ) order by longitude asc

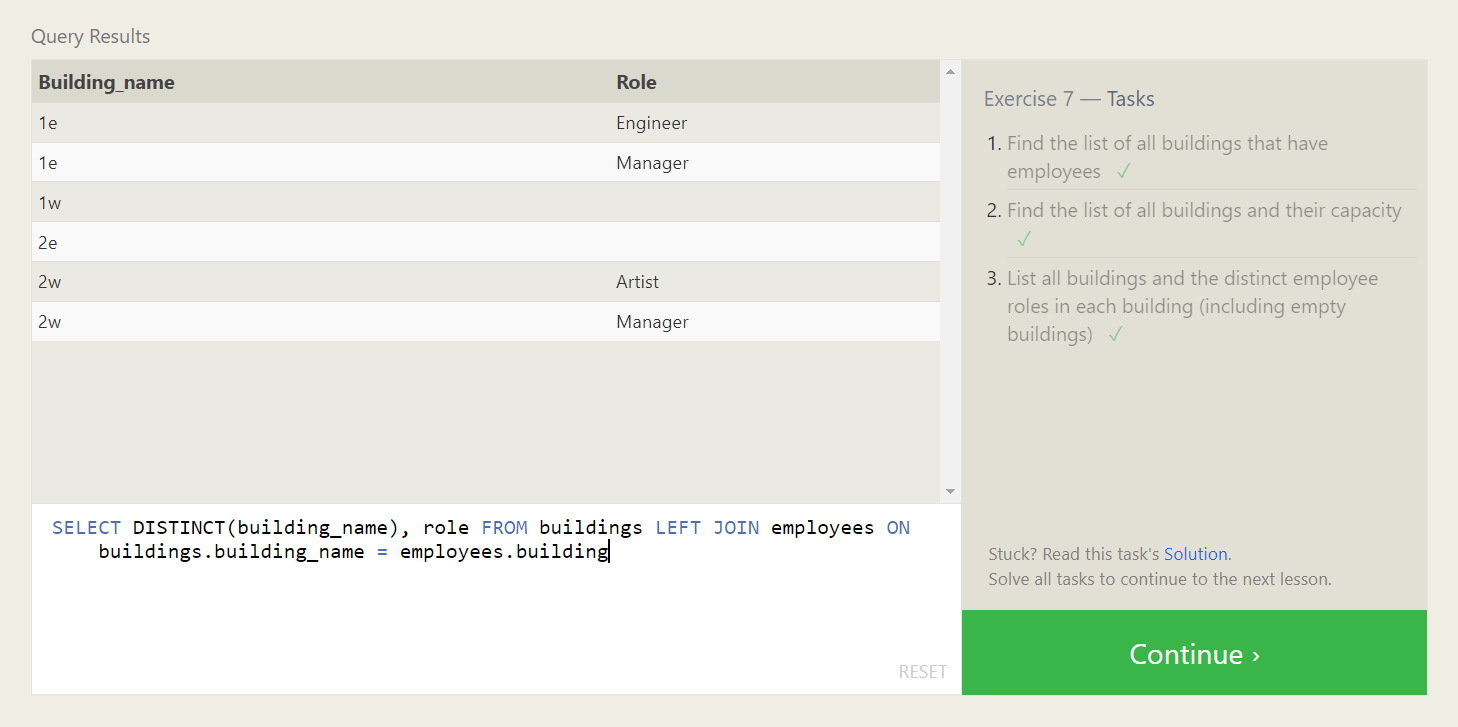
1. SELECT city FROM North\_american\_cities WHERE country="Mexico" ORDER BY population DESC limit 2
2. SELECT city,population FROM North\_american\_cities WHERE country="United States" ORDER BY population DESC LIMIT 2 OFFSET 2

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



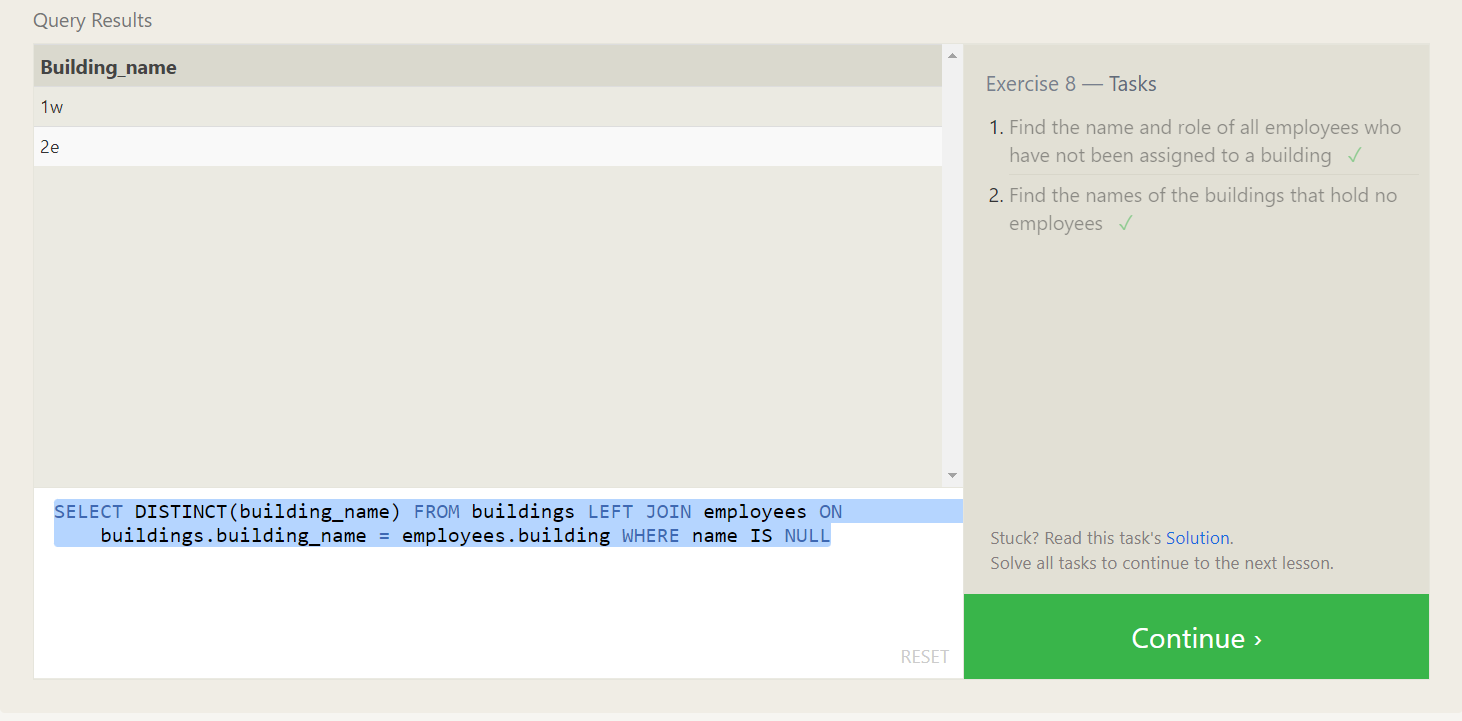
1. SELECT title, domestic\_sales, international\_sales FROM Boxoffice JOIN Movies on id=movie\_id
2. SELECT title, Domestic\_sales,International\_sales FROM Boxoffice join Movies on id=movie\_id WHERE Domestic\_sales<International\_sales
3. SELECT title FROM movies JOIN boxoffice on movie\_id=id ORDER BY rating DESC

**SQL Lesson 7: OUTER JOINs**



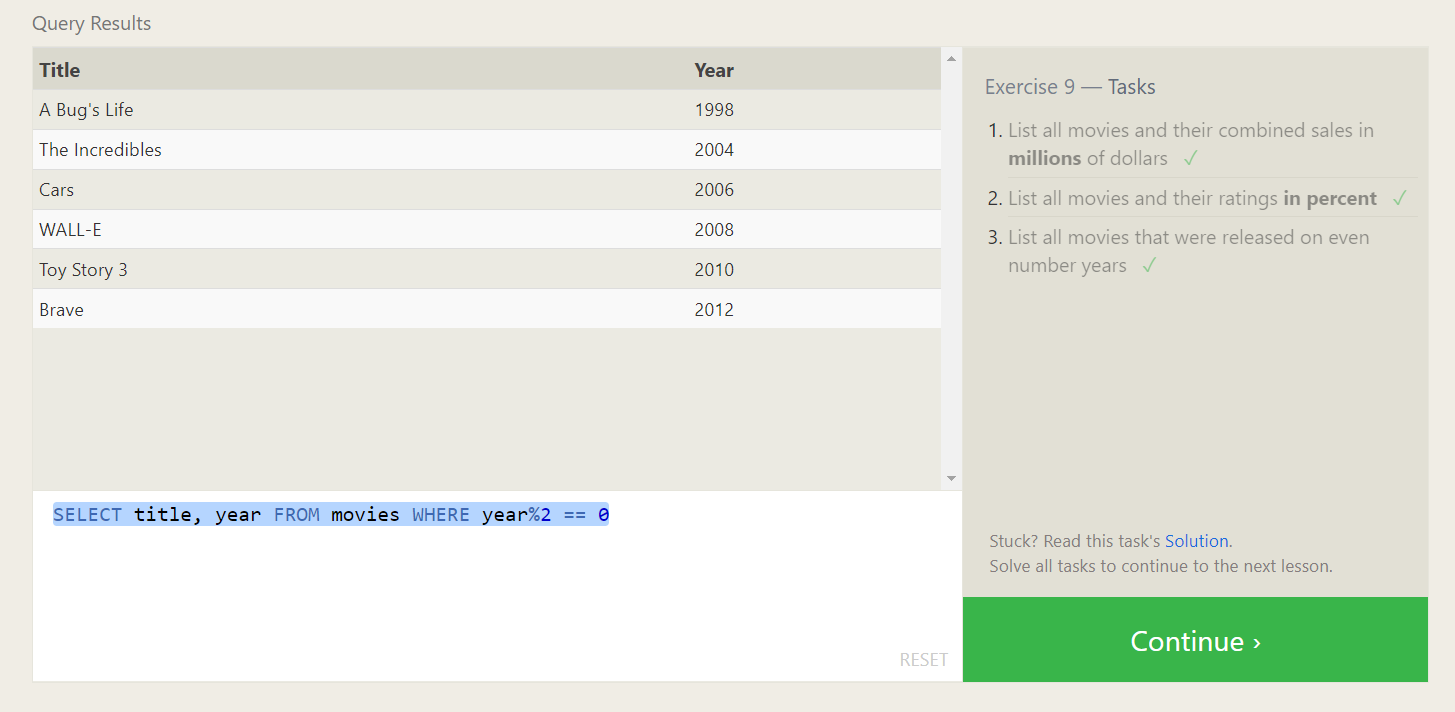
1. SELECT DISTINCT building FROM employees
2. SELECT building\_name, capacity FROM buildings
3. SELECT DISTINCT(building\_name), role FROM buildings LEFT JOIN employees ON buildings.building\_name = employees.building

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



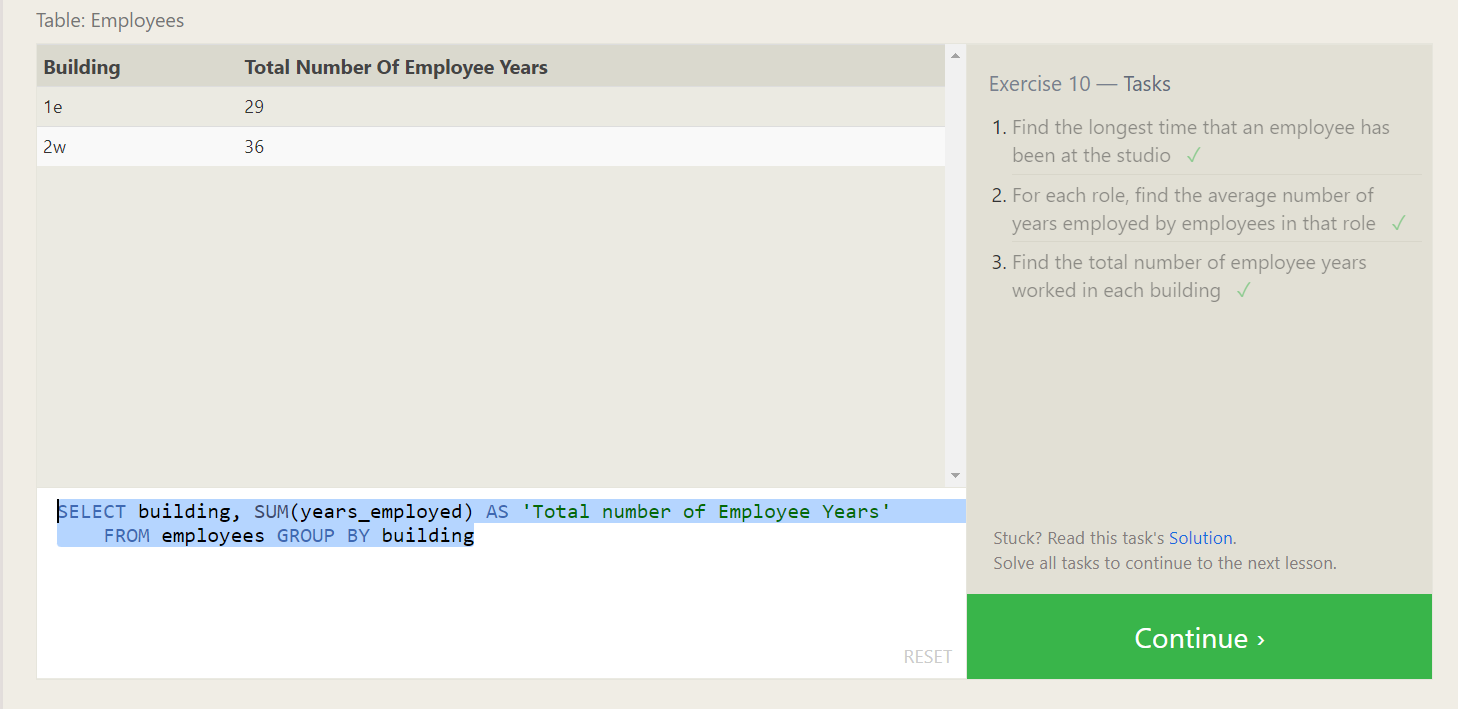
1. SELECT name, role FROM employees WHERE building IS NULL
2. SELECT DISTINCT(building\_name) FROM buildings LEFT JOIN employees ON buildings.building\_name = employees.building WHERE name IS NULL

**SQL Lesson 9: Queries with expressions**



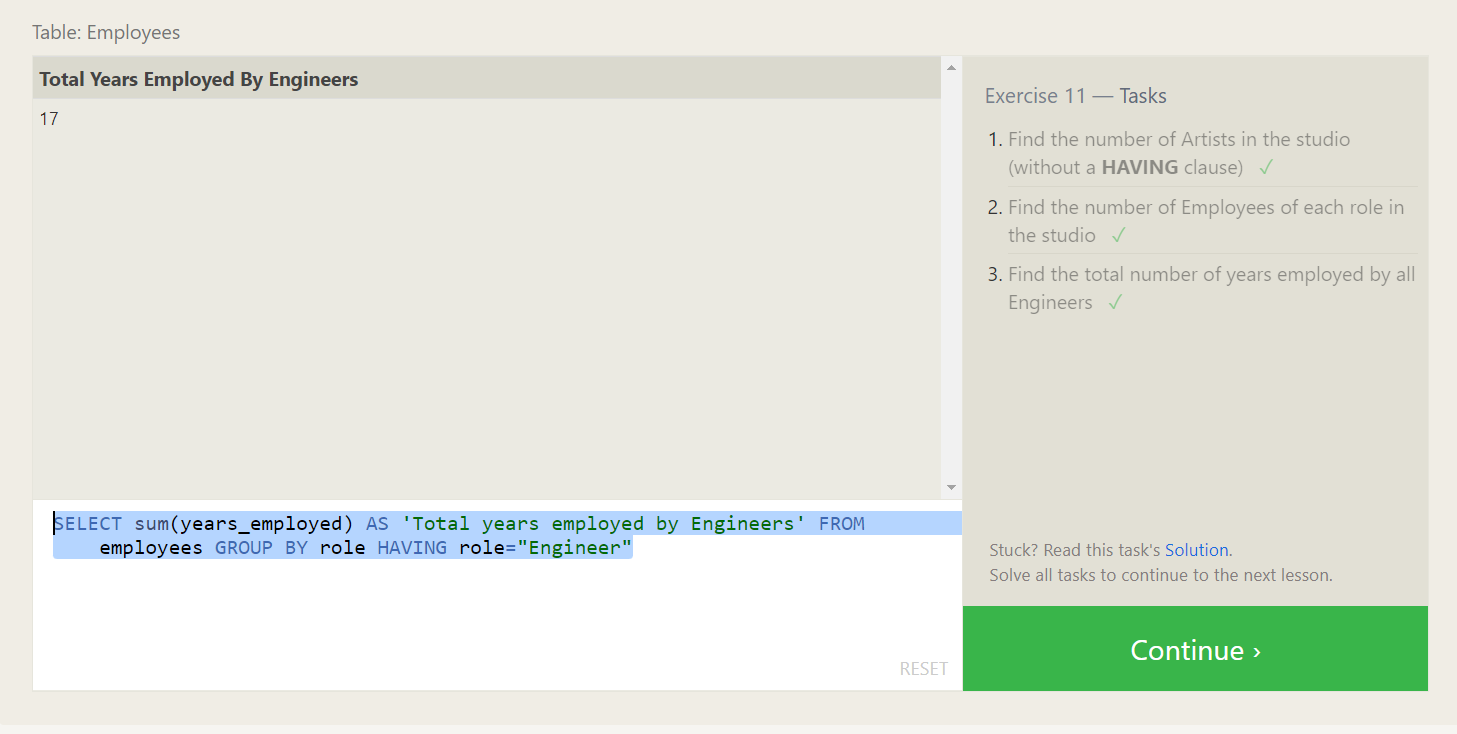
1. SELECT title , (domestic\_sales+international\_sales)/1000000 AS combined\_sales FROM movies LEFT JOIN boxoffice ON id=movie\_id
2. SELECT title, round((rating/10),2)\*100 as rating FROM movies JOIN boxoffice ON movie\_id = id
3. SELECT title, year FROM movies WHERE year%2 == 0

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



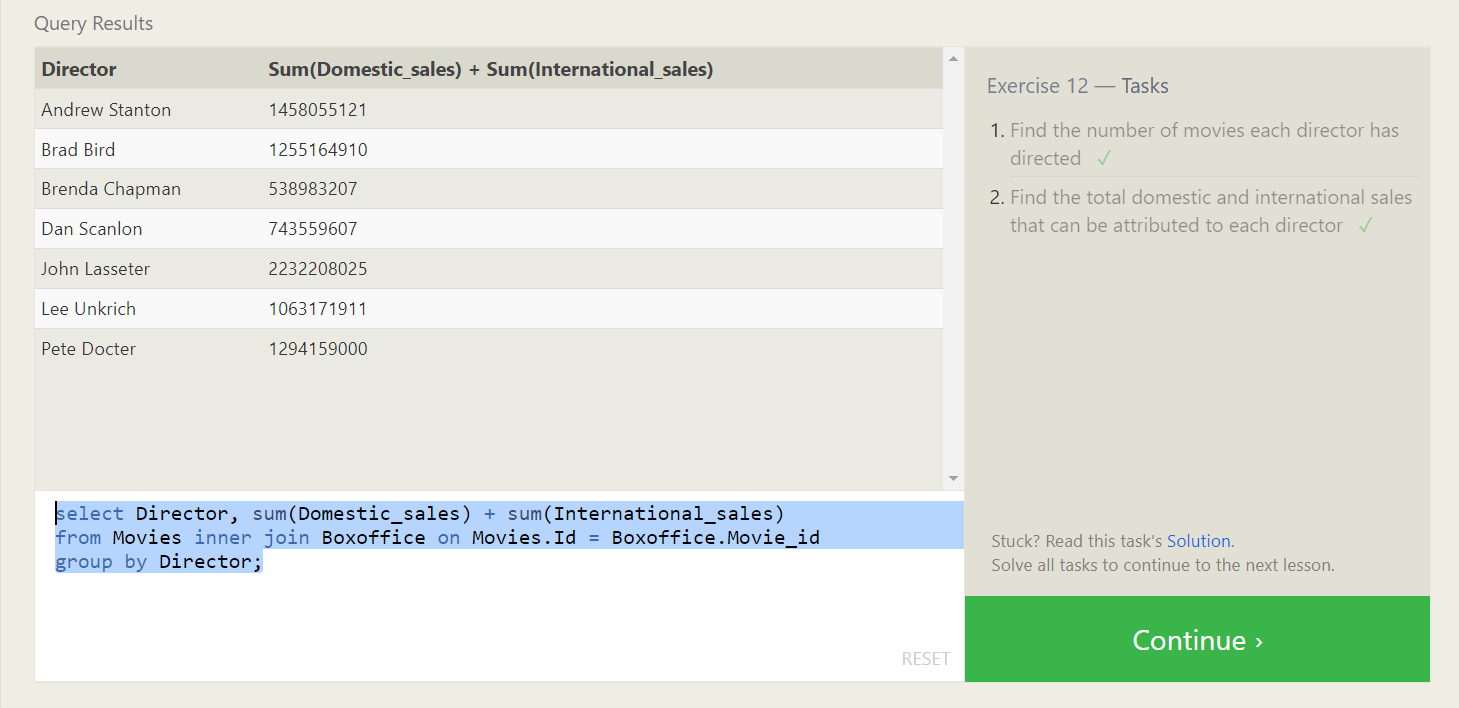
1. SELECT sum(years\_employed) AS longest\_time, name FROM employees GROUP BY name ORDER BY longest\_time DESC limit 1
2. SELECT role, avg(years\_employed) FROM employees GROUP BY role
3. SELECT building, SUM(years\_employed) AS 'Total number of Employee Years' FROM employees GROUP BY building

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



1. SELECT COUNT(role) FROM employees WHERE role="Artist"
2. SELECT role, COUNT(role) AS 'Number of Employees' FROM employees GROUP BY role
3. SELECT sum(years\_employed) AS 'Total years employed by Engineers' FROM employees GROUP BY role HAVING role="Engineer"

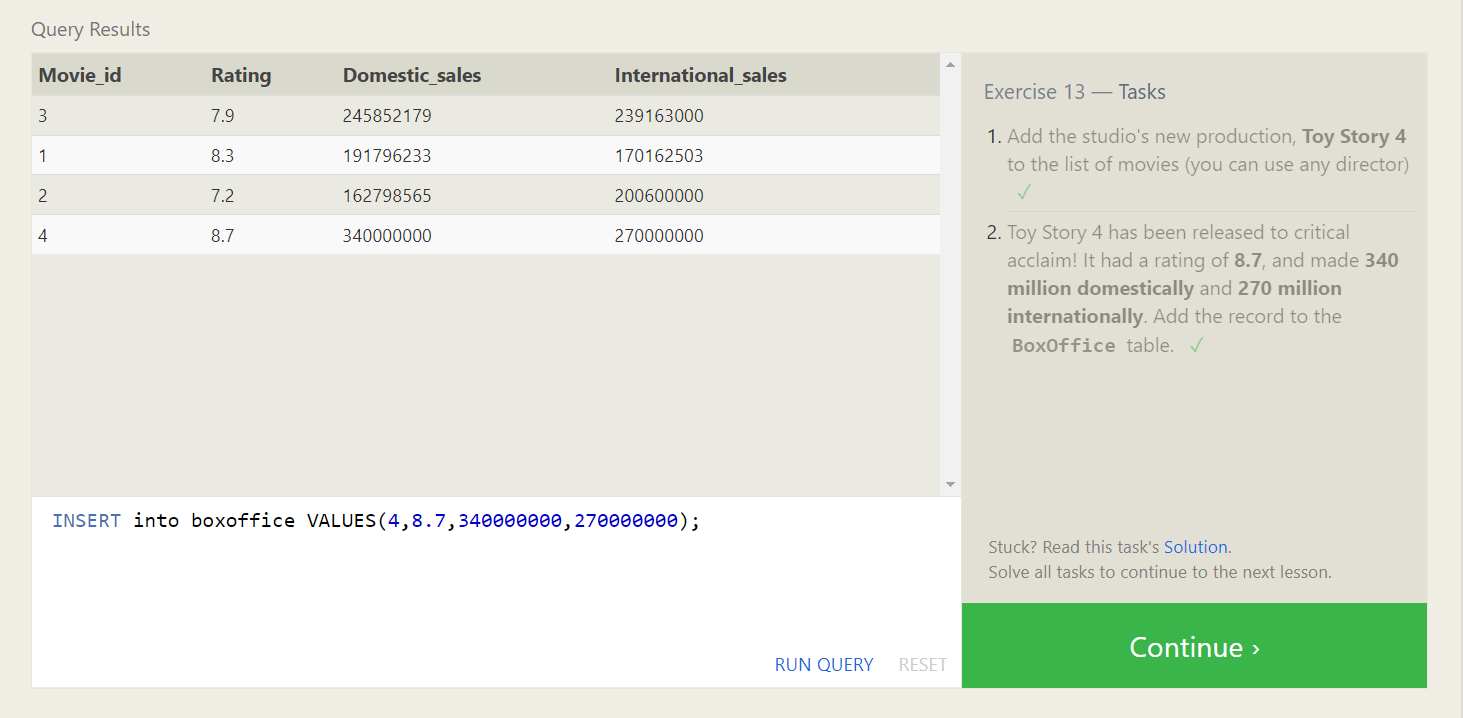
**SQL Lesson 12: Order of execution of a Query**



1. SELECT director, COUNT(title) AS 'Number of Movies' FROM movies GROUP BY director
2. select Director, sum(Domestic\_sales) + sum(International\_sales)

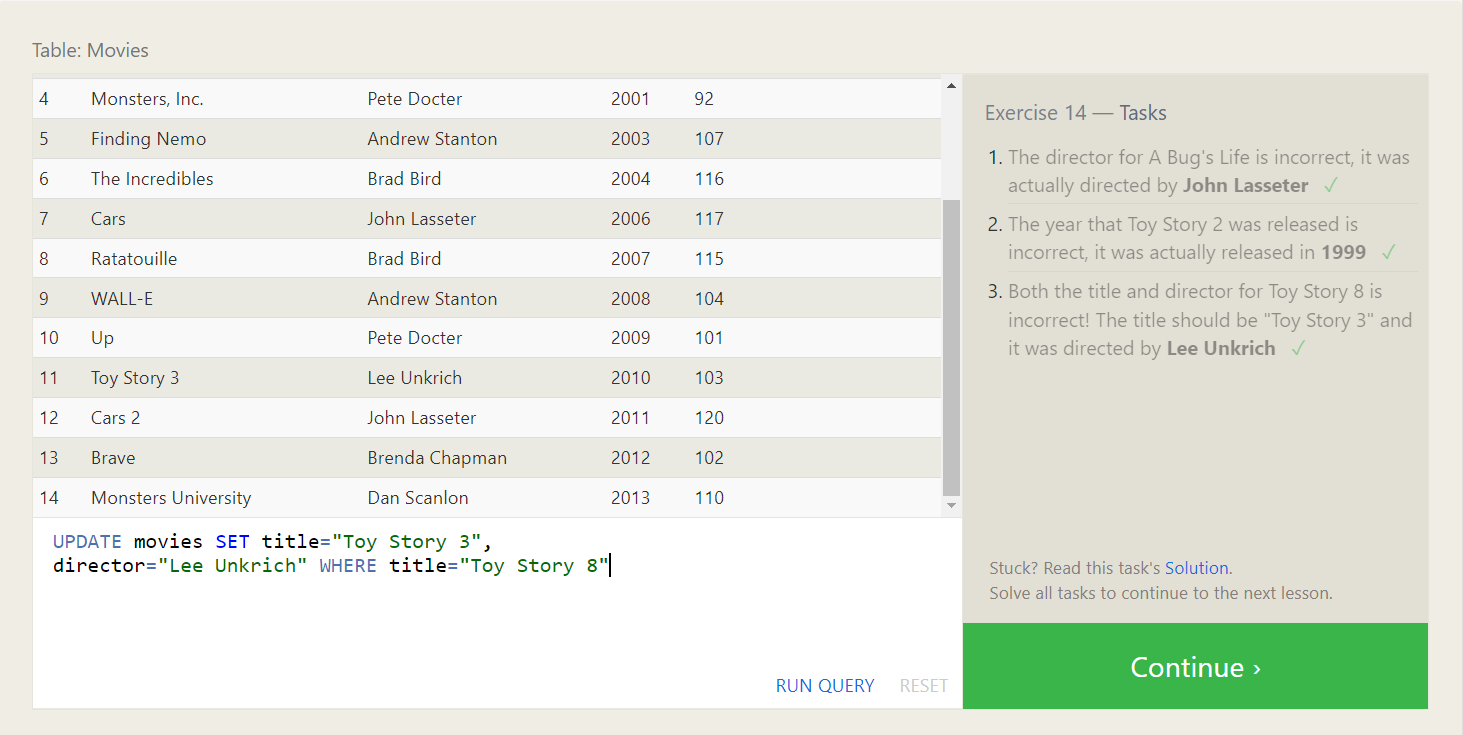
from Movies inner join Boxoffice on Movies.Id = Boxoffice.Movie\_id group by Director

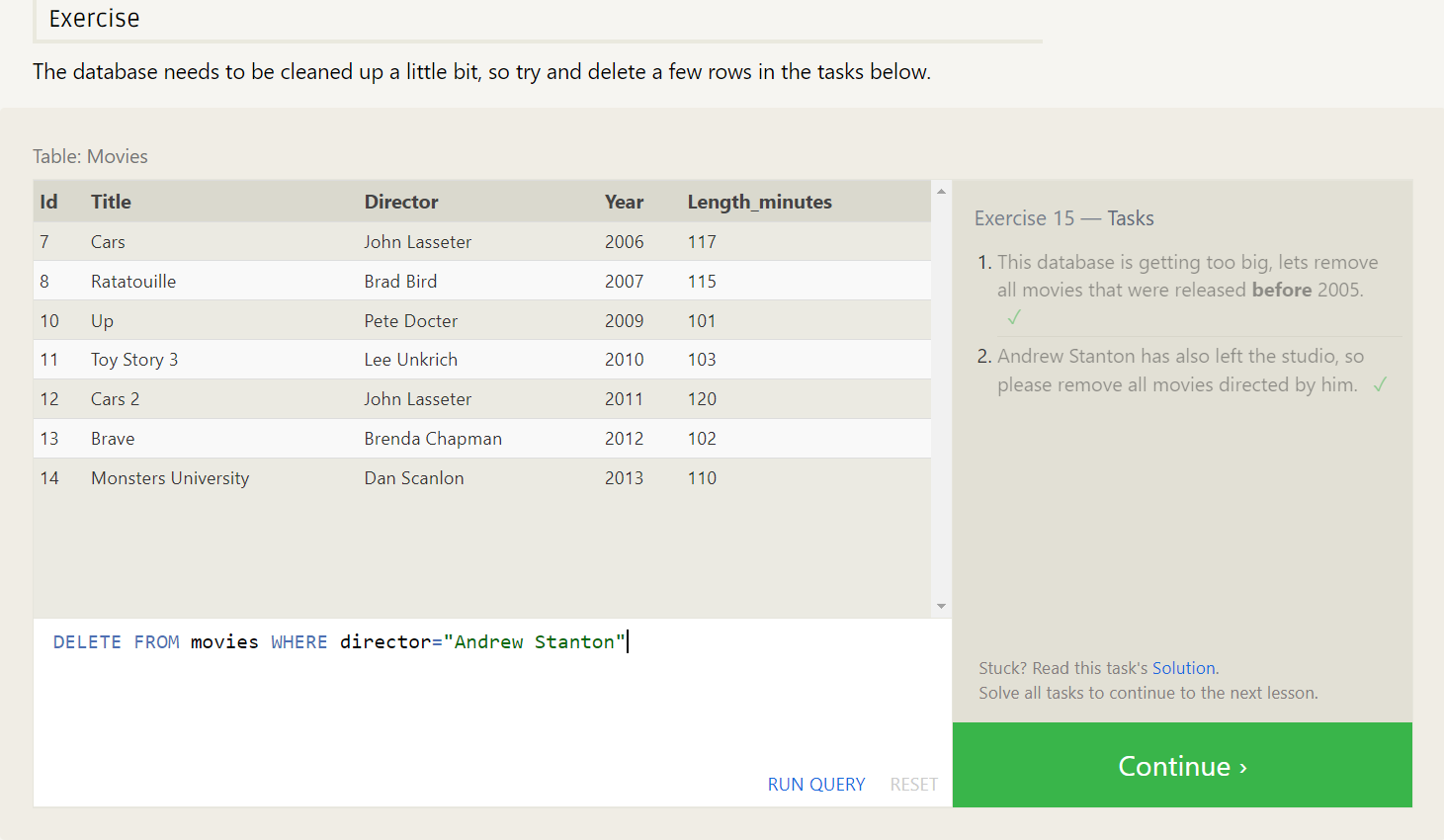
**SQL Lesson 13: Inserting rows**



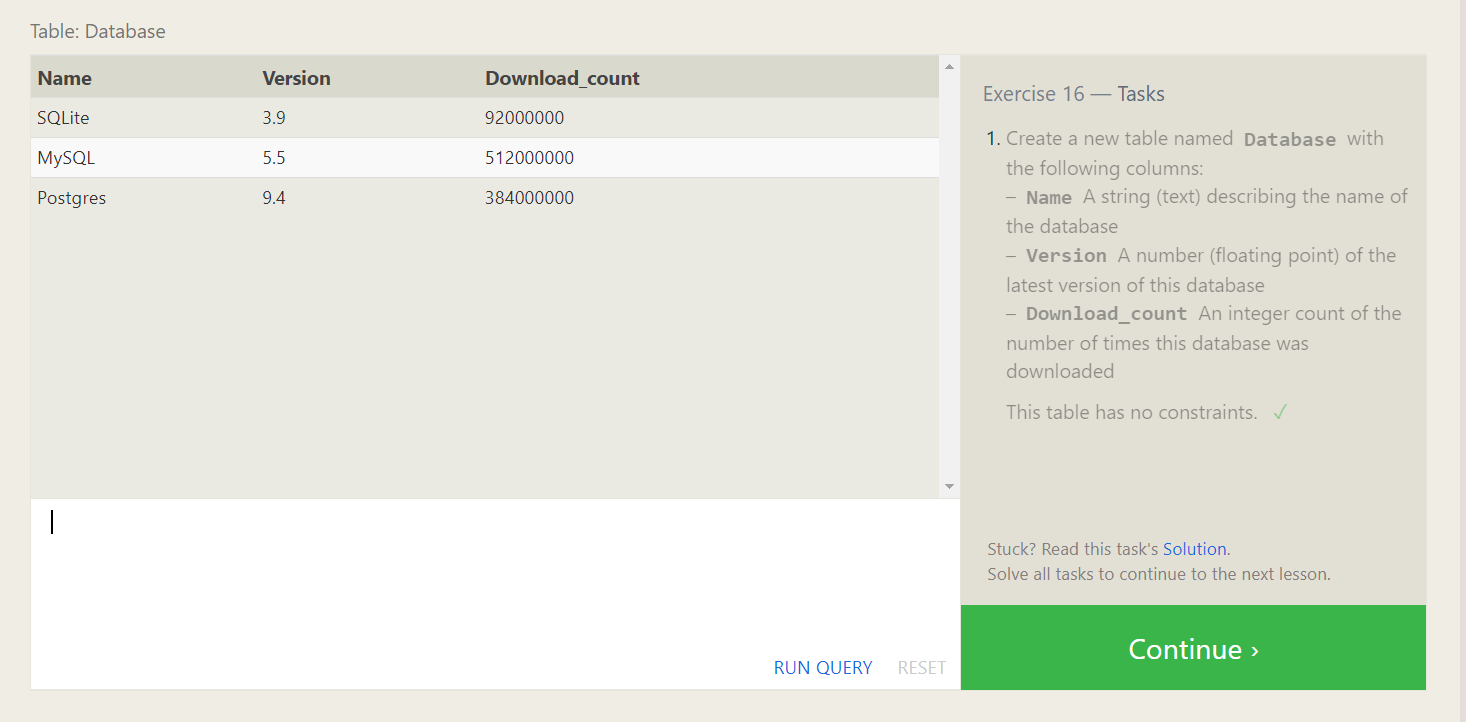
1. INSERT INTO movies(id,title,director, year, length\_minutes) VALUES(4, “Toy Story 4”, “Alex John”, 2023, 98);
2. INSERT into boxoffice VALUES(4,8.7,340000000,270000000);

**SQL Lesson 14: Updating rows**



1. UPDATE movies SET director="John Lasseter" WHERE title="A Bug's Life"
2. UPDATE movies SET year=1999 WHERE title="Toy Story 2"
3. UPDATE movies SET title="Toy Story 3", director="Lee Unkrich" WHERE title="Toy Story 8"
4. **SQL Lesson 15: Deleting rows**
5. DELETE FROM movies WHERE year<2005
6. DELETE FROM movies WHERE director="Andrew Stanton"

**SQL Lesson 16: Creating tables**



CREATE TABLE Database(

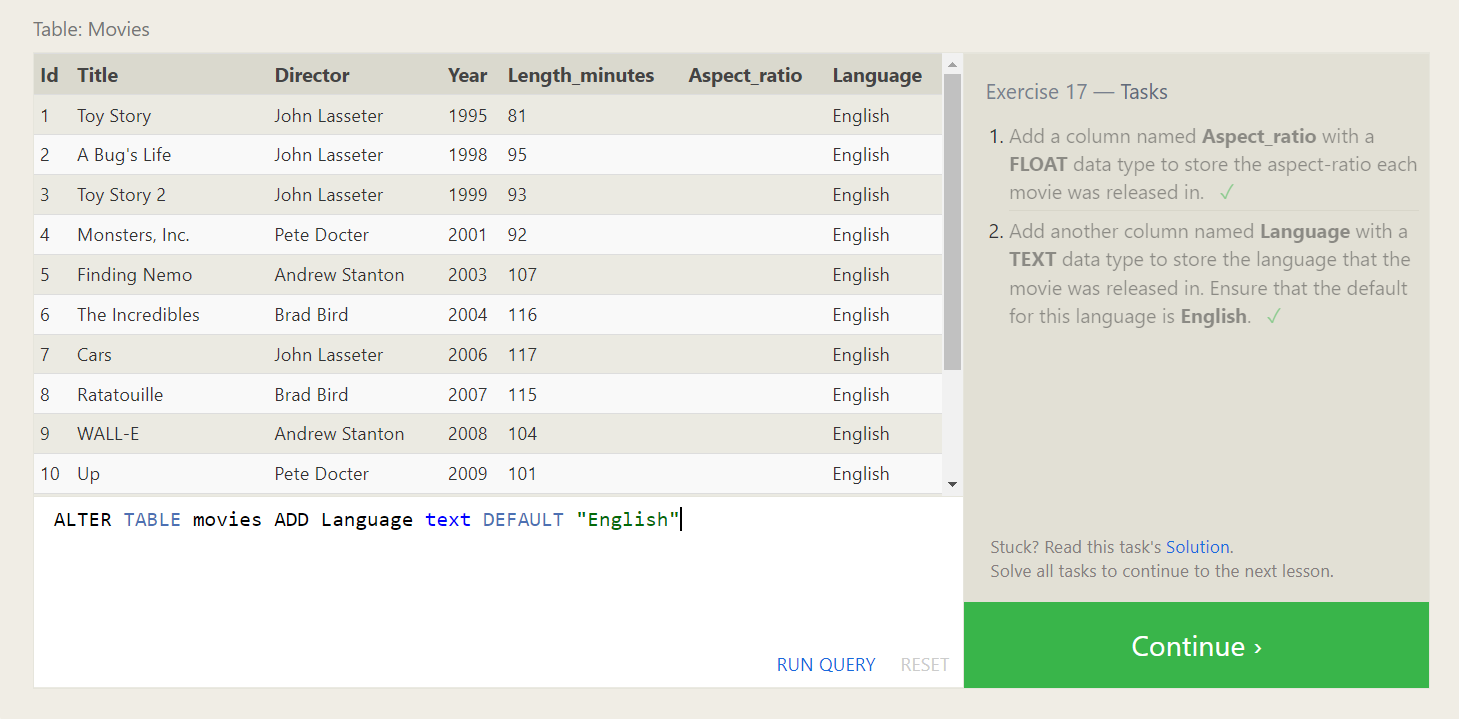
Name text,

Version float,

Download\_count int

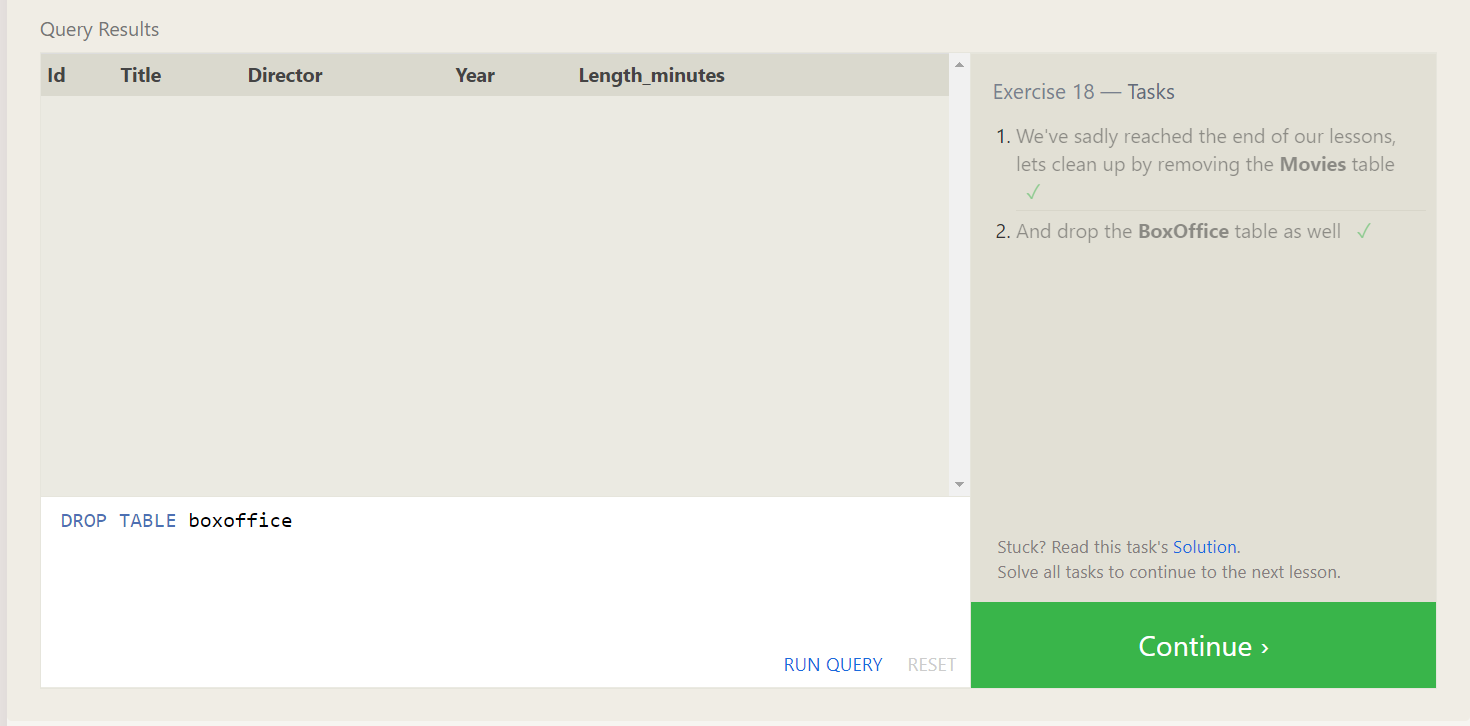
);

**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 IF EXISTS movies
2. DROP TABLE boxoffice