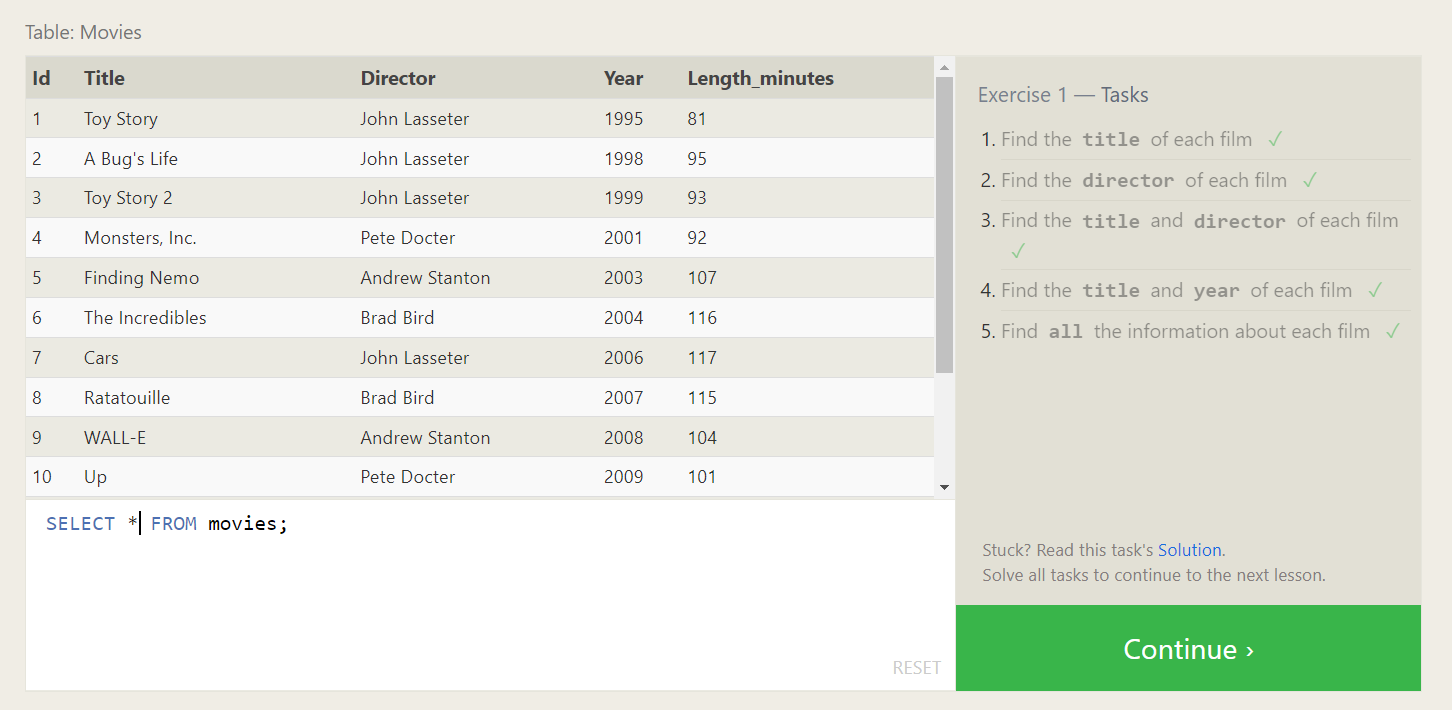
**MySQL TASK**

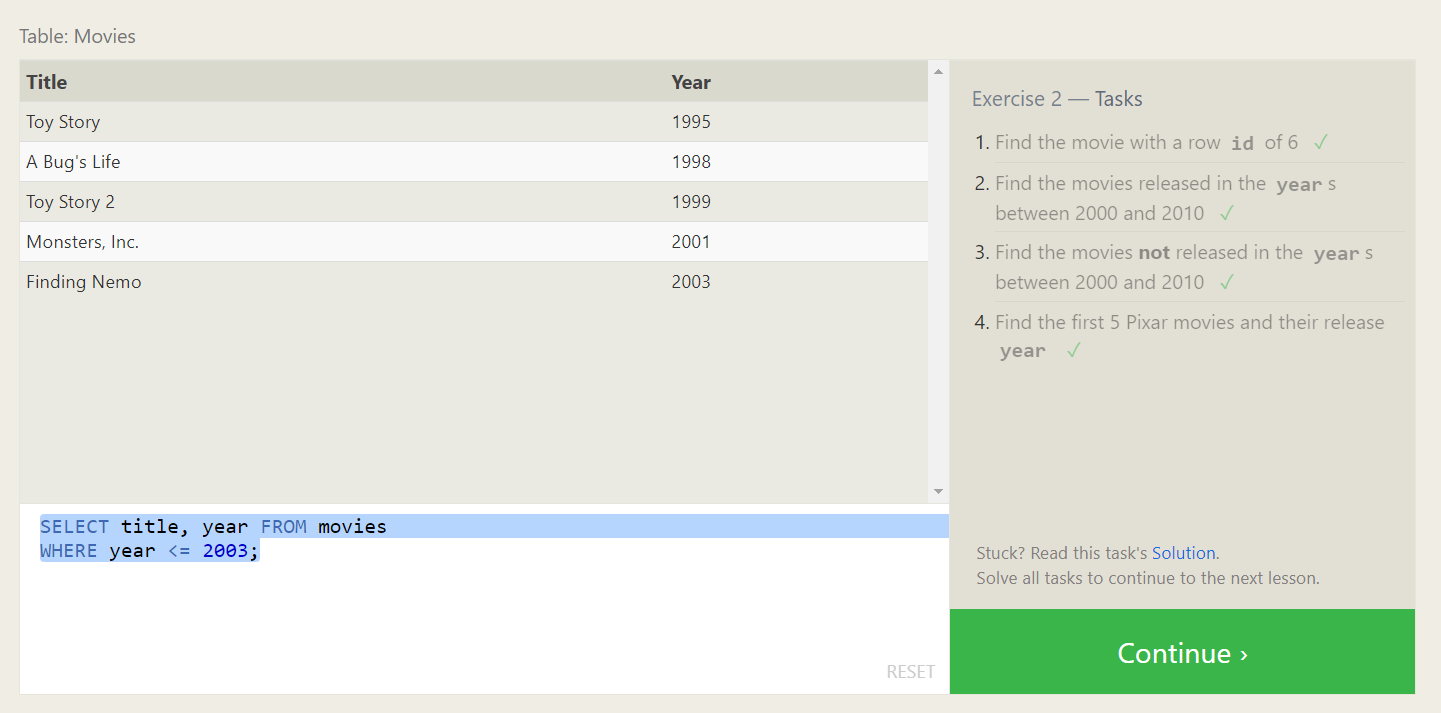
**SQL Lesson 1: SELECT queries 101**

****

**Solutions :**

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**

****

**Solutions :**

1. **SELECT \* FROM movies**

**WHERE Id = 6;**

1. **SELECT \* FROM movies**

**WHERE year BETWEEN 2000 AND 2010;**

1. **SELECT \* FROM movies**

**WHERE Year NOT BETWEEN 2000 AND 2010;**

1. **SELECT title, year FROM movies**

**WHERE year <= 2003;**

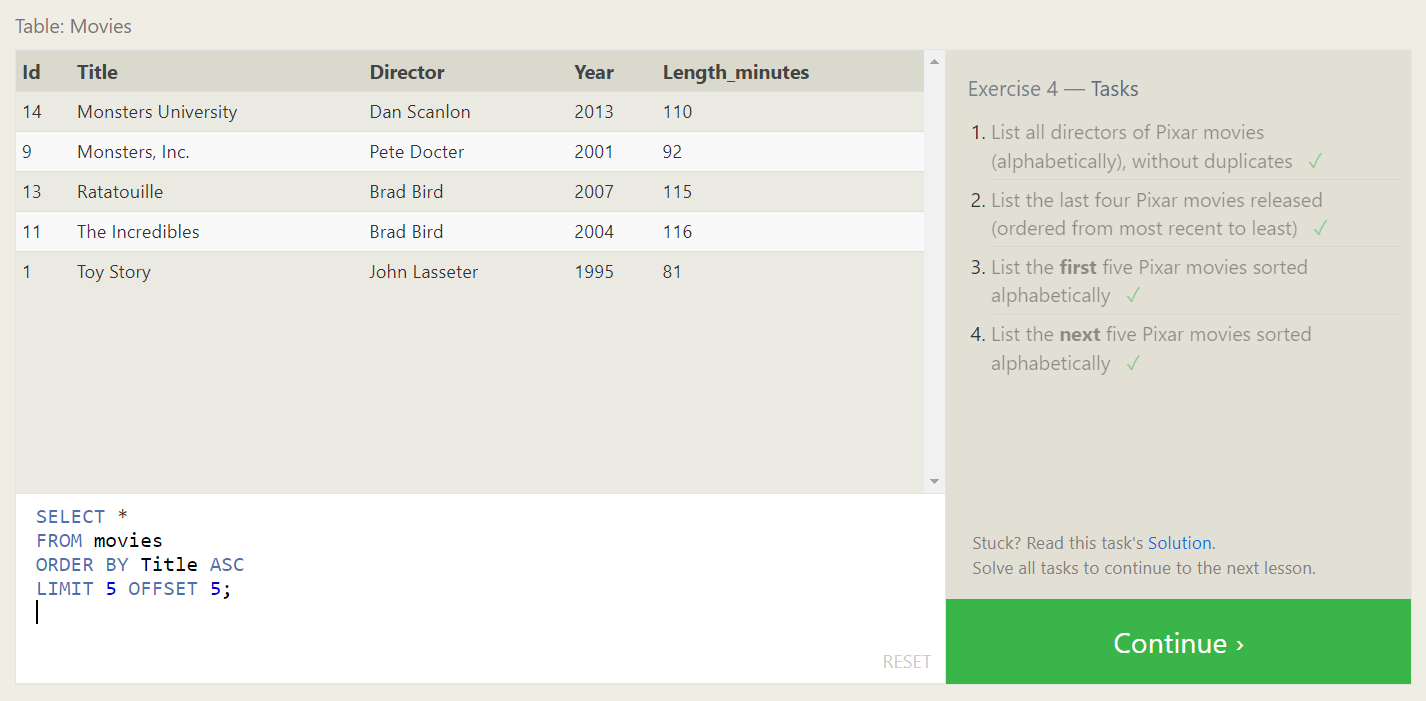
**SQL Lesson 3: Queries with constraints**

****

**Solutions :**

1. **SELECT \* FROM movies WHERE Title LIKE 'Toy Story%';**
2. **SELECT \* FROM movies WHERE Director = 'John Lasseter';**
3. **SELECT \* FROM movies WHERE Director != 'John Lasseter';**
4. **SELECT \* FROM movies WHERE Title LIKE 'WALL-%';**

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

****

**Solutions :**

1. **SELECT DISTINCT Director**

**FROM movies**

**ORDER BY Director ASC;**

1. **SELECT \***

**FROM movies**

**ORDER BY Year DESC**

**LIMIT 4;**

1. **SELECT \***

**FROM movies**

**ORDER BY Title ASC**

**LIMIT 5;**

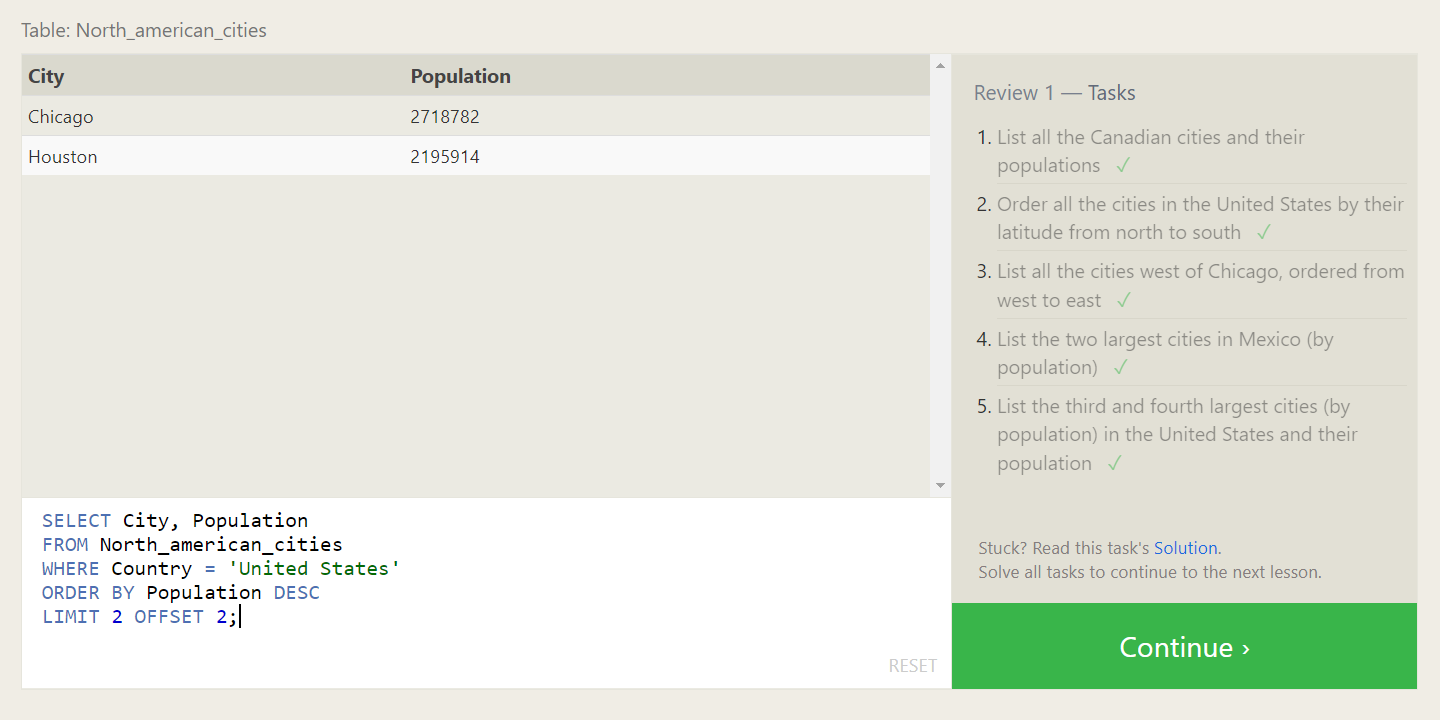
1. **SELECT \***

**FROM movies**

**ORDER BY Title ASC**

**LIMIT 5 OFFSET 5;**

**SQL Review: Simple SELECT Queries**

****

**Solutions :**

1. **SELECT City, Population**

**FROM North\_american\_cities**

**WHERE Country = 'Canada';**

1. **SELECT City, Latitude**

**FROM North\_american\_cities**

**WHERE Country = 'United States'**

**ORDER BY Latitude DESC;**

1. **SELECT City, Longitude**

**FROM North\_american\_cities**

**WHERE Longitude < (SELECT Longitude FROM North\_american\_cities WHERE City = 'Chicago')**

**ORDER BY Longitude ASC;**

1. **SELECT City, Population**

**FROM North\_american\_cities**

**WHERE Country = 'Mexico'**

**ORDER BY Population DESC**

**LIMIT 2;**

1. **SELECT City, Population**

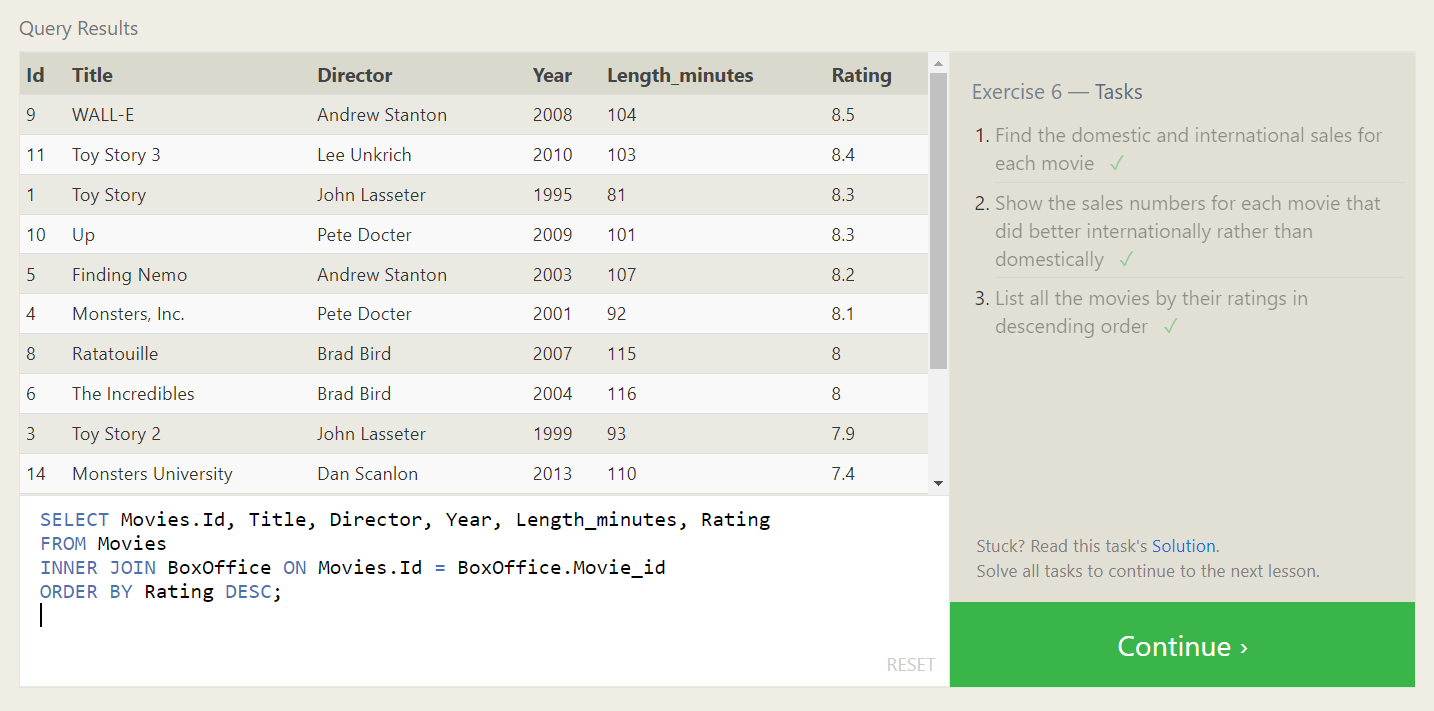
**FROM North\_american\_cities**

**WHERE Country = 'United States'**

**ORDER BY Population DESC**

**LIMIT 2 OFFSET 2;**

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

****

**Solutions :**

1. **SELECT Movies.Id, Title, Director, Year, Length\_minutes, Domestic\_sales, International\_sales**

**FROM Movies**

**INNER JOIN BoxOffice ON Movies.Id = BoxOffice.Movie\_id;**

1. **SELECT Movies.Id, Title, Director, Year, Length\_minutes, Domestic\_sales, International\_sales**

**FROM Movies**

**INNER JOIN BoxOffice ON Movies.Id = BoxOffice.Movie\_id**

**WHERE International\_sales > Domestic\_sales;**

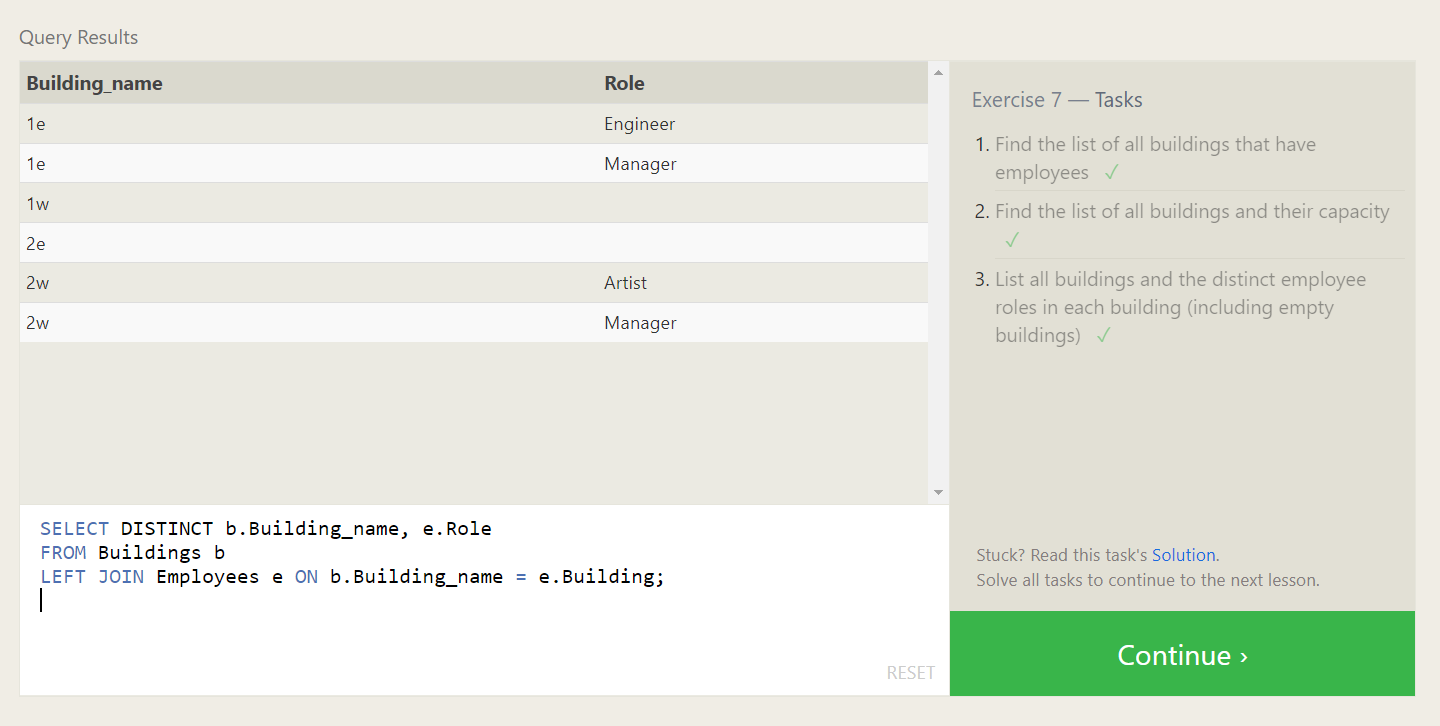
1. **SELECT Movies.Id, Title, Director, Year, Length\_minutes, Rating**

**FROM Movies**

**INNER JOIN BoxOffice ON Movies.Id = BoxOffice.Movie\_id**

**ORDER BY Rating DESC;**

**SQL Lesson 7: OUTER JOINs**

****

**Solutions :**

1. **SELECT DISTINCT e.Building**

**FROM Employees e**

**LEFT JOIN Buildings b ON e.Building = b.Building\_name**

**WHERE e.Building IS NOT NULL;**

1. **SELECT b.Building\_name, b.Capacity**

**FROM Buildings b**

**LEFT JOIN Employees e ON b.Building\_name = e.Building**

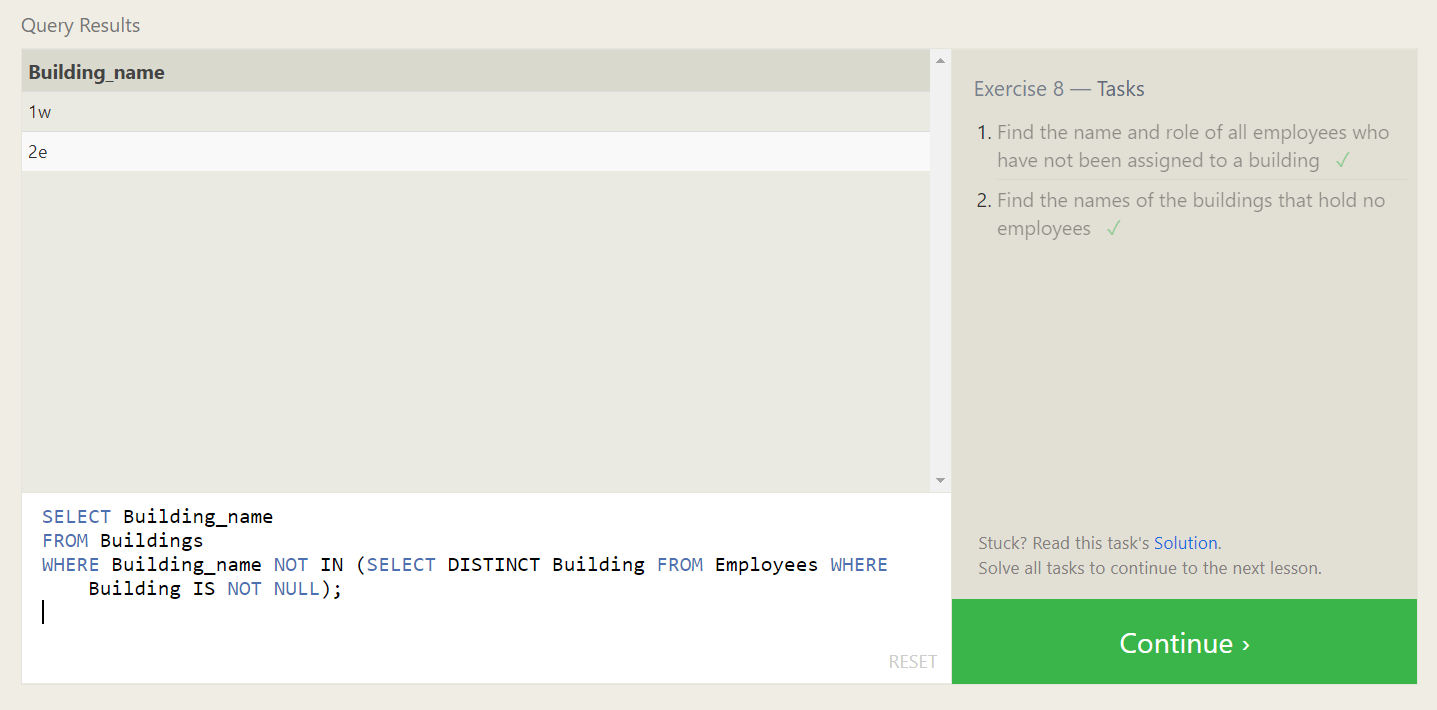
**GROUP BY b.Building\_name, b.Capacity;**

1. **SELECT DISTINCT b.Building\_name, e.Role**

**FROM Buildings b**

**LEFT JOIN Employees e ON b.Building\_name = e.Building;**

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

****

**Solutions :**

1. **SELECT Name, Role**

**FROM Employees**

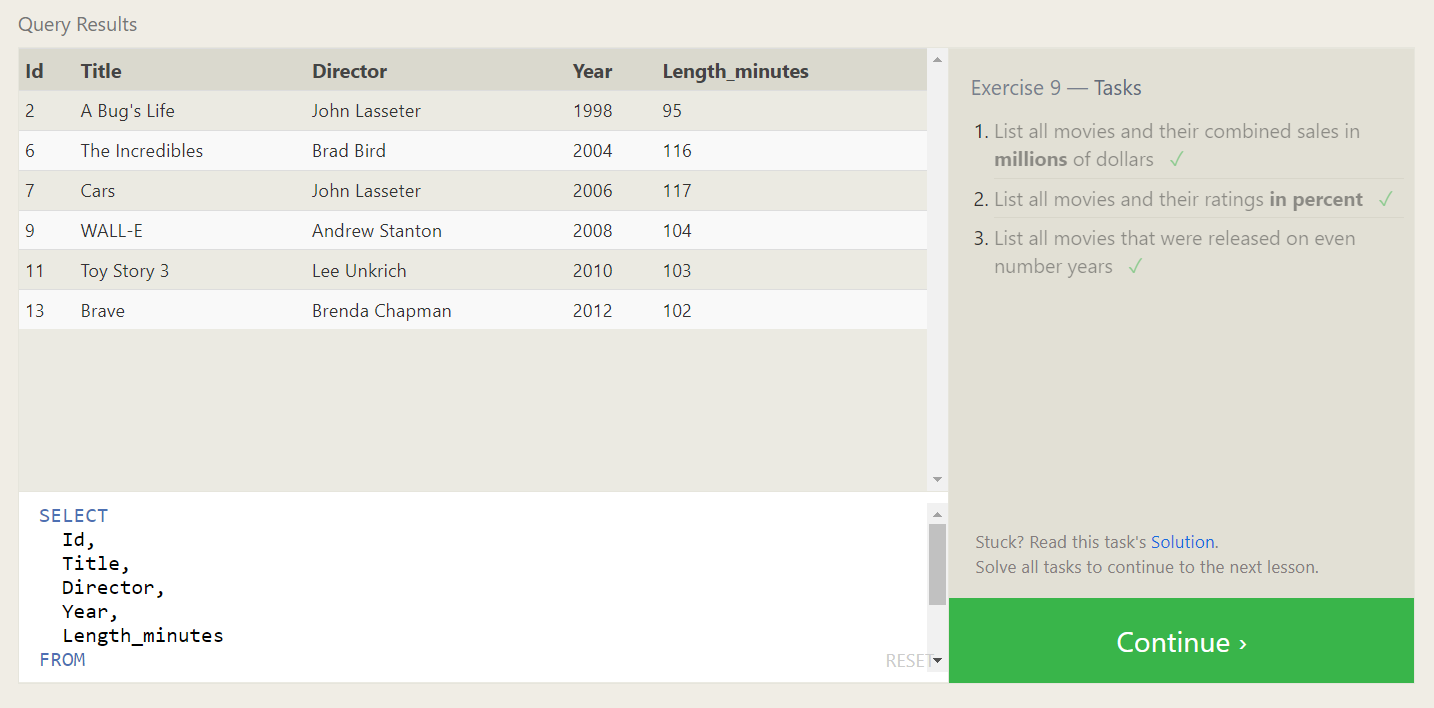
**WHERE Building IS NULL;**

1. **SELECT Building\_name**

**FROM Buildings**

**WHERE Building\_name NOT IN (SELECT DISTINCT Building FROM Employees WHERE Building IS NOT NULL);**

**SQL Lesson 9: Queries with expressions**

****

**Solutions :**

1. **SELECT**

**m.Id,**

**m.Title,**

**m.Director,**

**m.Year,**

**m.Length\_minutes,**

**(b.Domestic\_sales + b.International\_sales) / 1000000 AS combined\_sales\_millions**

**FROM**

**Movies m**

**JOIN**

**Boxoffice b ON m.Id = b.Movie\_id;**

1. **SELECT**

**m.Id,**

**m.Title,**

**m.Director,**

**m.Year,**

**m.Length\_minutes,**

**b.Rating \* 10 AS rating\_percent**

**FROM**

**Movies m**

**JOIN**

**Boxoffice b ON m.Id = b.Movie\_id;**

1. **SELECT**

**Id,**

**Title,**

**Director,**

**Year,**

**Length\_minutes**

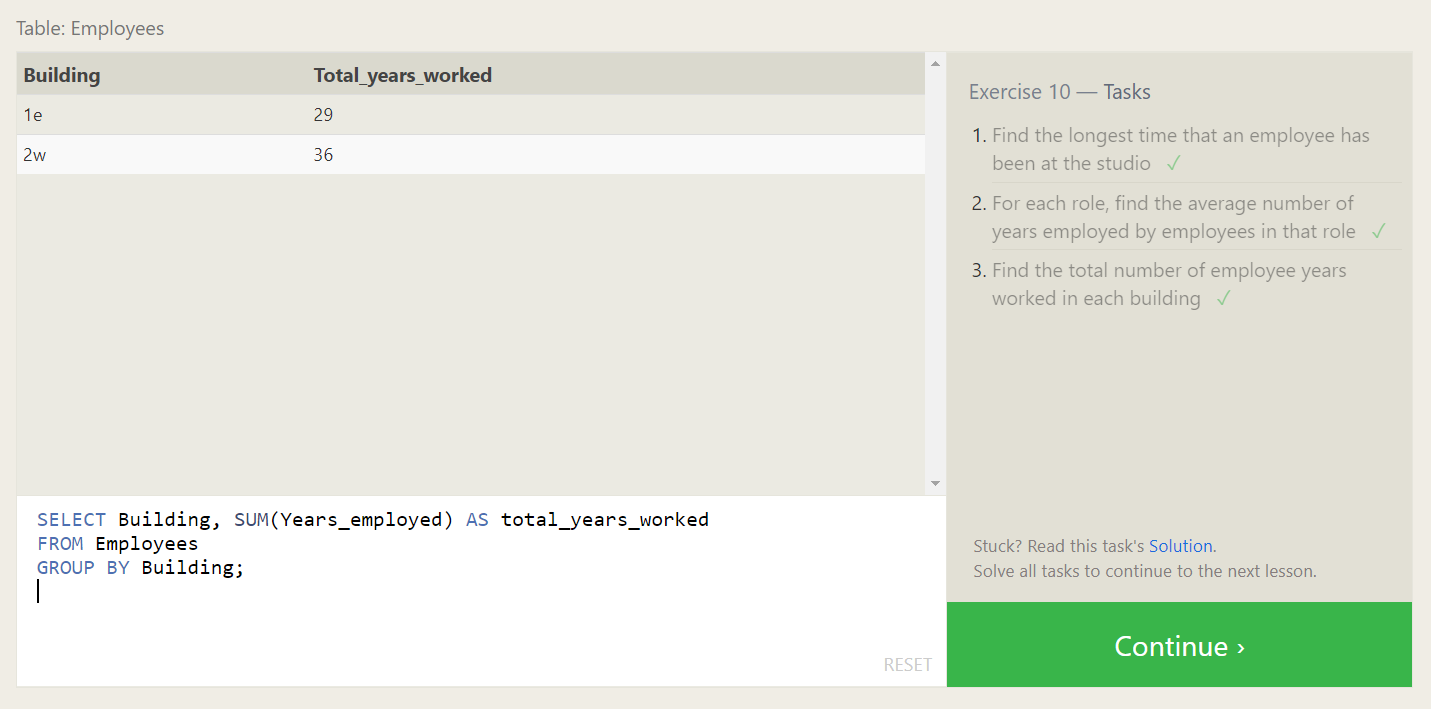
**FROM**

**Movies**

**WHERE**

**Year % 2 = 0;**

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

****

**Solutions :**

1. **SELECT MAX(Years\_employed) AS longest\_time**

**FROM Employees;**

1. **SELECT Role, AVG(Years\_employed) AS avg\_years\_employed**

**FROM Employees**

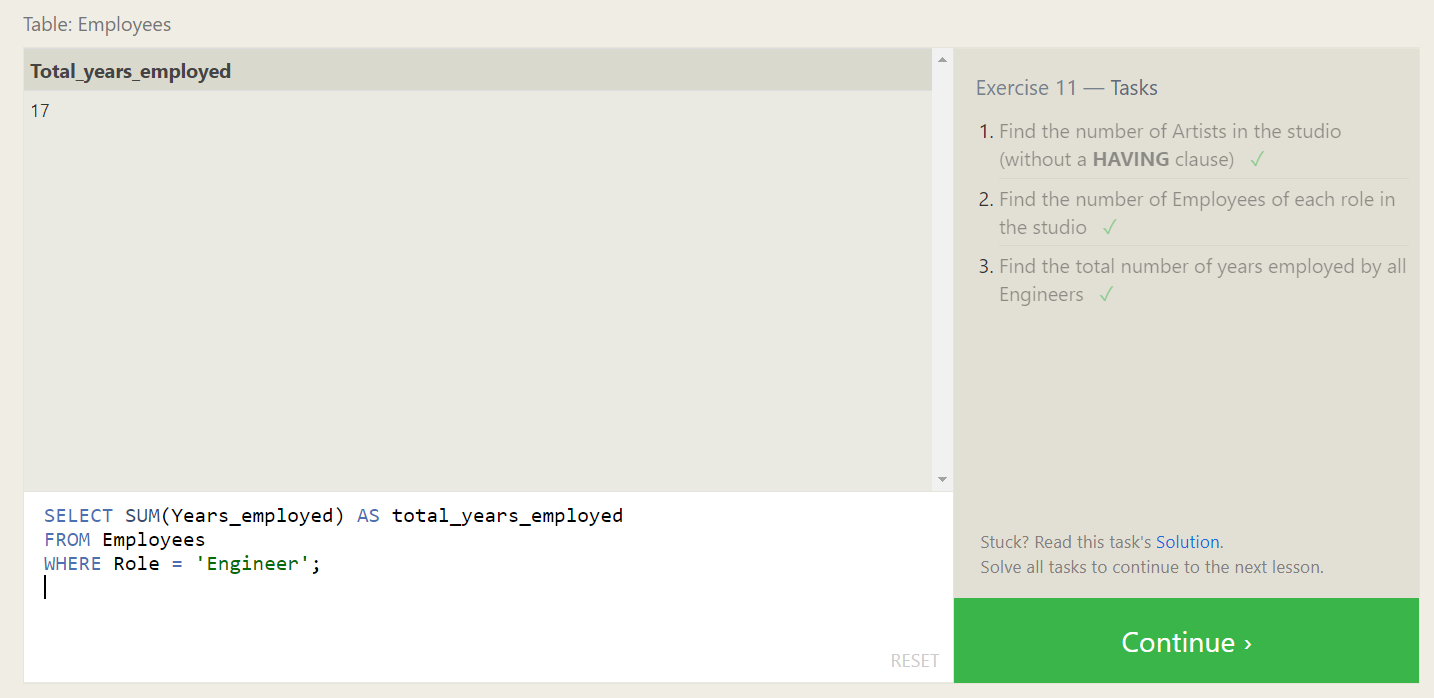
**GROUP BY Role;**

1. **SELECT Building, SUM(Years\_employed) AS total\_years\_worked**

**FROM Employees**

**GROUP BY Building;**

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

****

**Solutions :**

1. **SELECT COUNT(\*) AS num\_artists**

**FROM Employees**

**WHERE Role = 'Artist';**

1. **SELECT Role, COUNT(\*) AS num\_employees**

**FROM Employees**

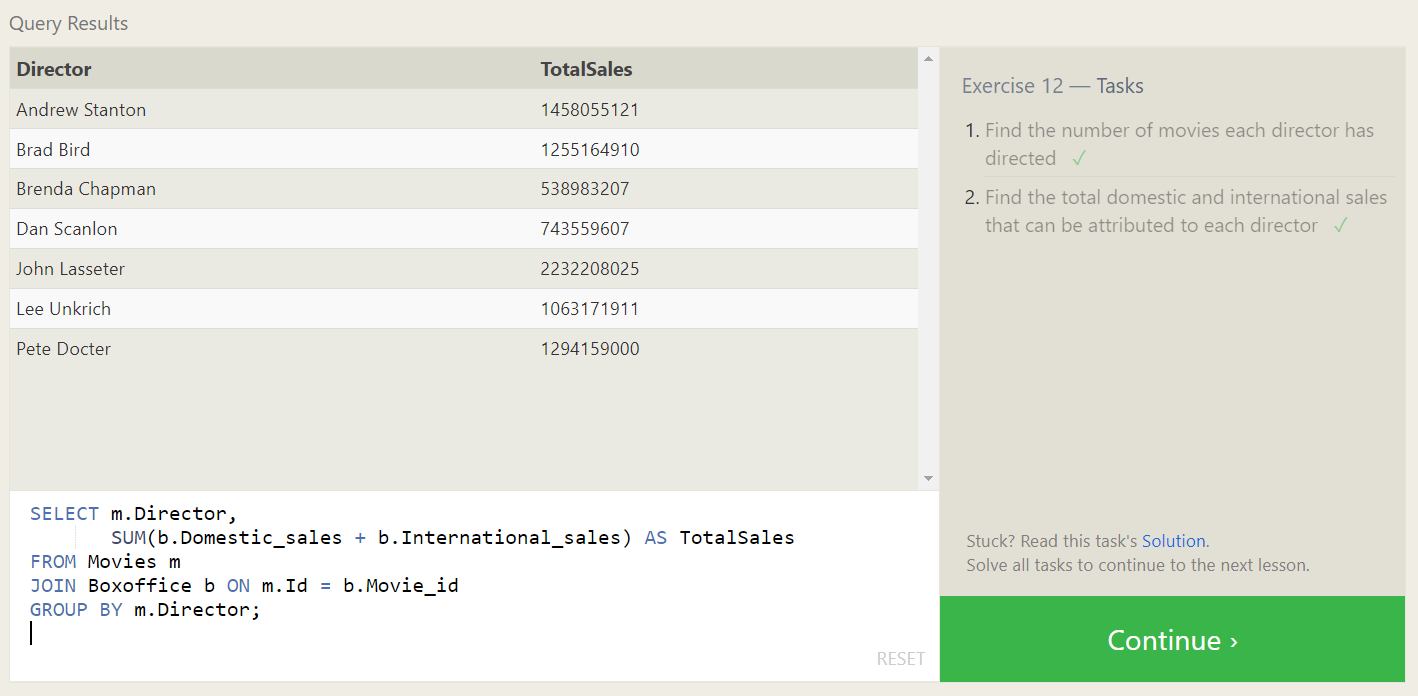
**GROUP BY Role;**

1. **SELECT SUM(Years\_employed) AS total\_years\_employed**

**FROM Employees**

**WHERE Role = 'Engineer';**

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

****

**Solutions :**

1. **SELECT Director, COUNT(\*) AS NumMoviesDirected**

**FROM Movies**

**GROUP BY Director;**

1. **SELECT m.Director,**

**SUM(b.Domestic\_sales + b.International\_sales) AS TotalSales**

**FROM Movies m**

**JOIN Boxoffice b ON m.Id = b.Movie\_id**

**GROUP BY m.Director;**

**SQL Lesson 13: Inserting rows**

****

**Solutions :**

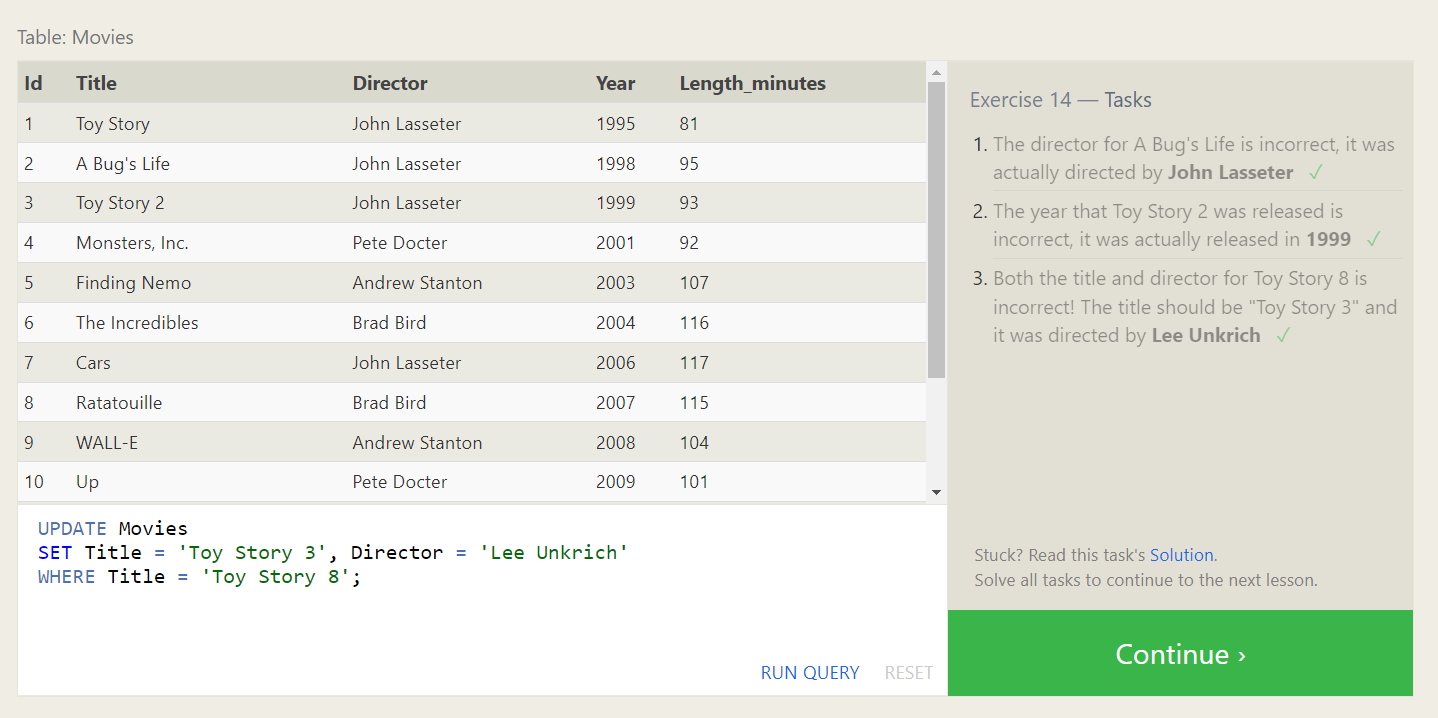
1. **INSERT INTO Movies (Title, Director, Year, Length\_minutes)**

**VALUES ('Toy Story 4', 'Any Director', 2023, 100);**

1. **INSERT INTO Boxoffice (Movie\_id, Rating, Domestic\_sales, International\_sales)**

**VALUES ((SELECT Id FROM Movies WHERE Title = 'Toy Story 4'), 8.7, 340000000, 270000000);**

**SQL Lesson 14: Updating rows**

****

**Solutions :**

1. **UPDATE Movies**

**SET Director = 'John Lasseter'**

**WHERE Title = 'A Bug''s Life';**

1. **UPDATE Movies**

**SET Year = 1999**

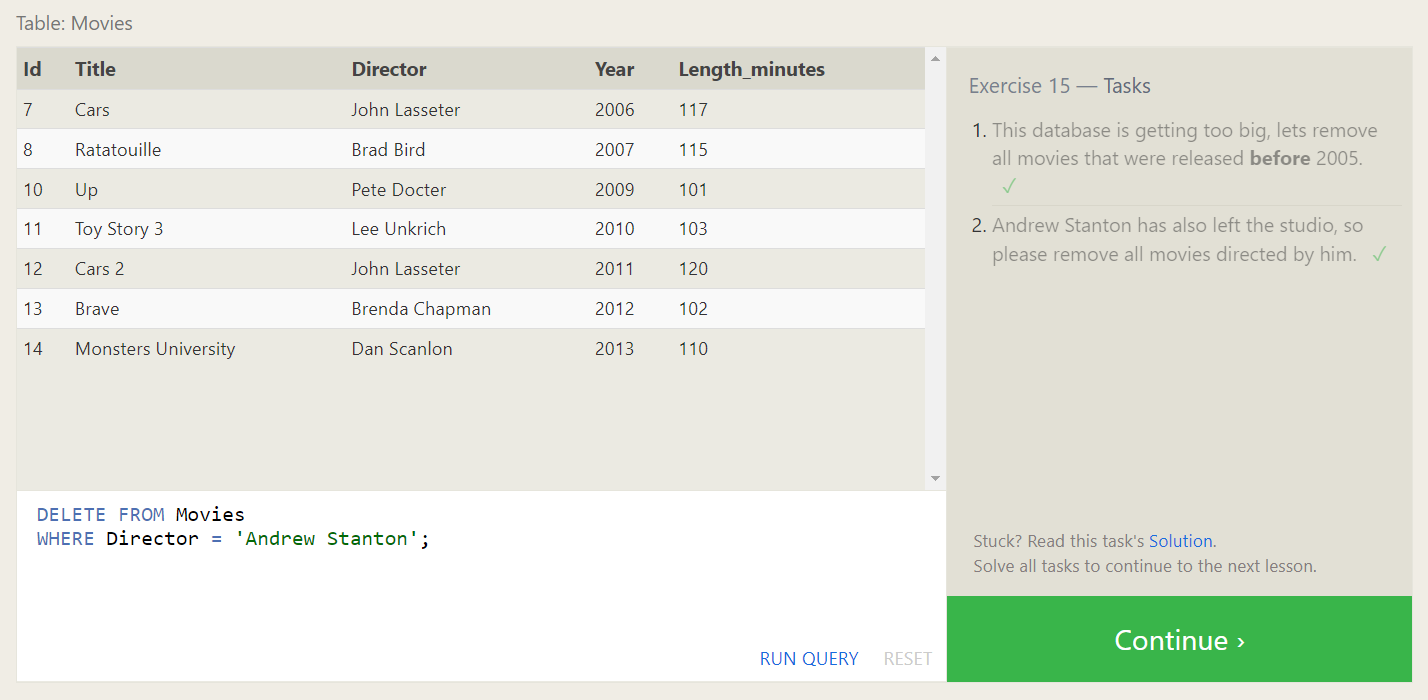
**WHERE Title = 'Toy Story 2';**

1. **UPDATE Movies**

**SET Title = 'Toy Story 3', Director = 'Lee Unkrich'**

**WHERE Title = 'Toy Story 8';**

**SQL Lesson 15: Deleting rows**

****

**Solutions :**

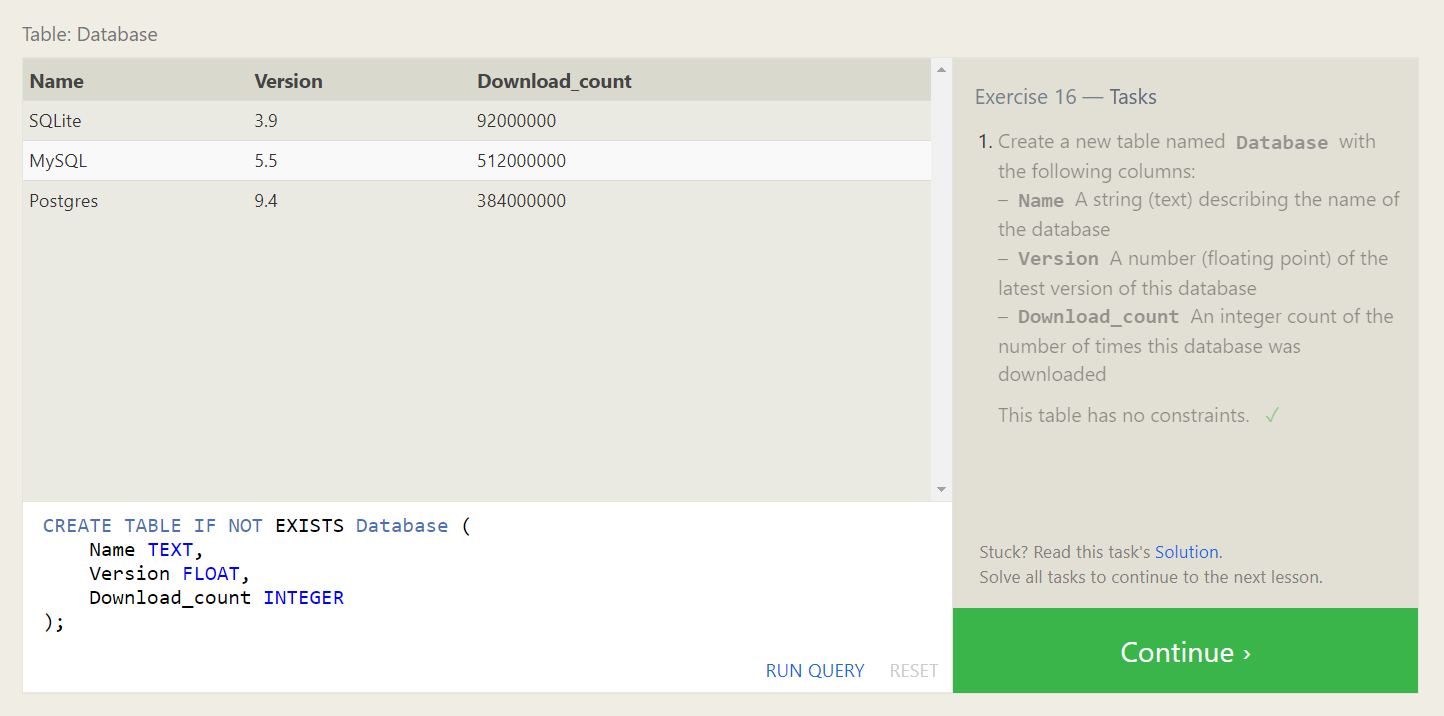
1. **DELETE FROM Movies**

**WHERE Year < 2005;**

1. **DELETE FROM Movies**

**WHERE Director = 'Andrew Stanton';**

**SQL Lesson 16: Creating tables**

****

**Solutions :**

1. **CREATE TABLE IF NOT EXISTS Database (**

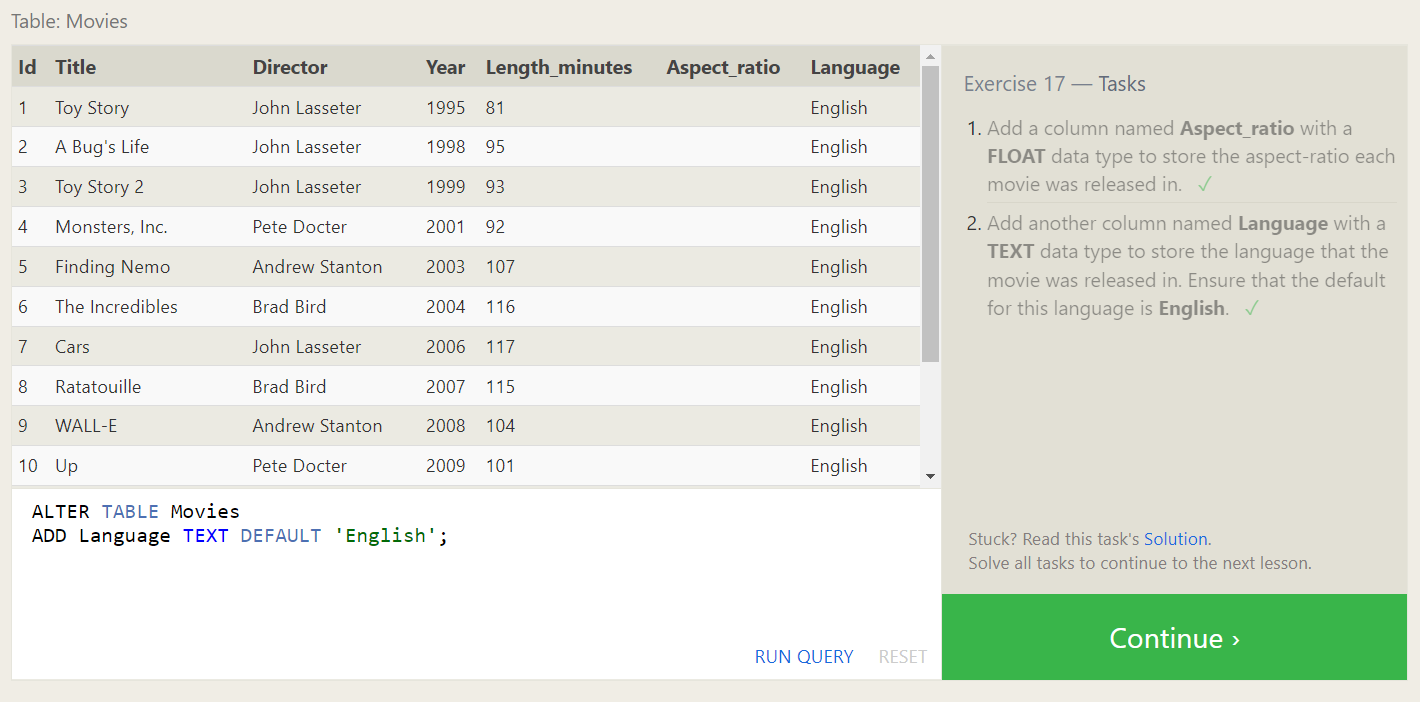
**Name TEXT,**

**Version FLOAT,**

**Download\_count INTEGER**

**);**

**SQL Lesson 17: Altering tables**

****

**Solutions :**

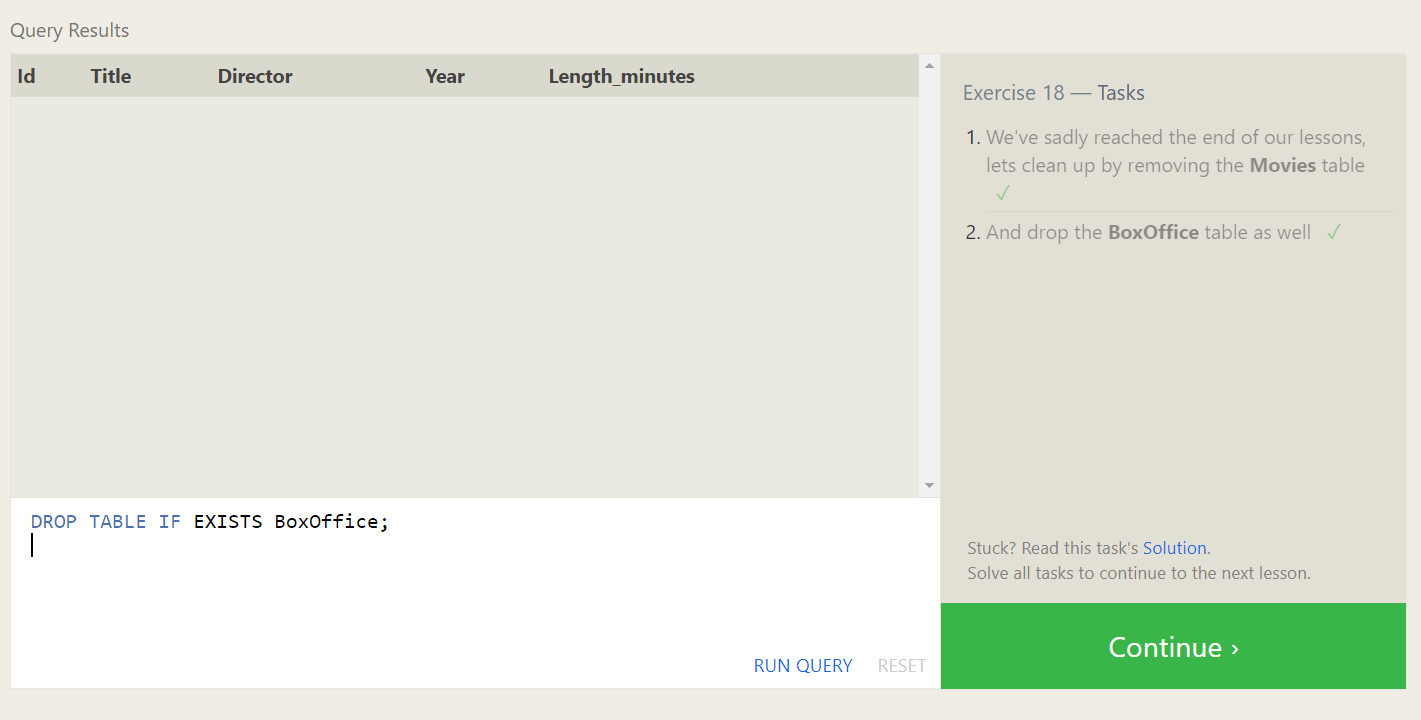
1. **ALTER TABLE Movies**

**ADD Aspect\_ratio FLOAT;**

1. **ALTER TABLE Movies**

**ADD Language TEXT DEFAULT 'English';**

**SQL Lesson 18: Dropping tables**

****

**Solutions :**

1. **DROP TABLE IF EXISTS Movies;**
2. **DROP TABLE IF EXISTS BoxOffice;**