Class 32 Task

**Complete SQL Bolt, and submit the screenshots**

**-- CH1 - Select Queries 101**

a) Find the title of each film

SELECT Title

FROM Movies;

b) Find the director of each film

SELECT Director

FROM Movies;

c) Find the title and director of each film

SELECT Title, Director

FROM Movies;

d) Find the title and year of each film

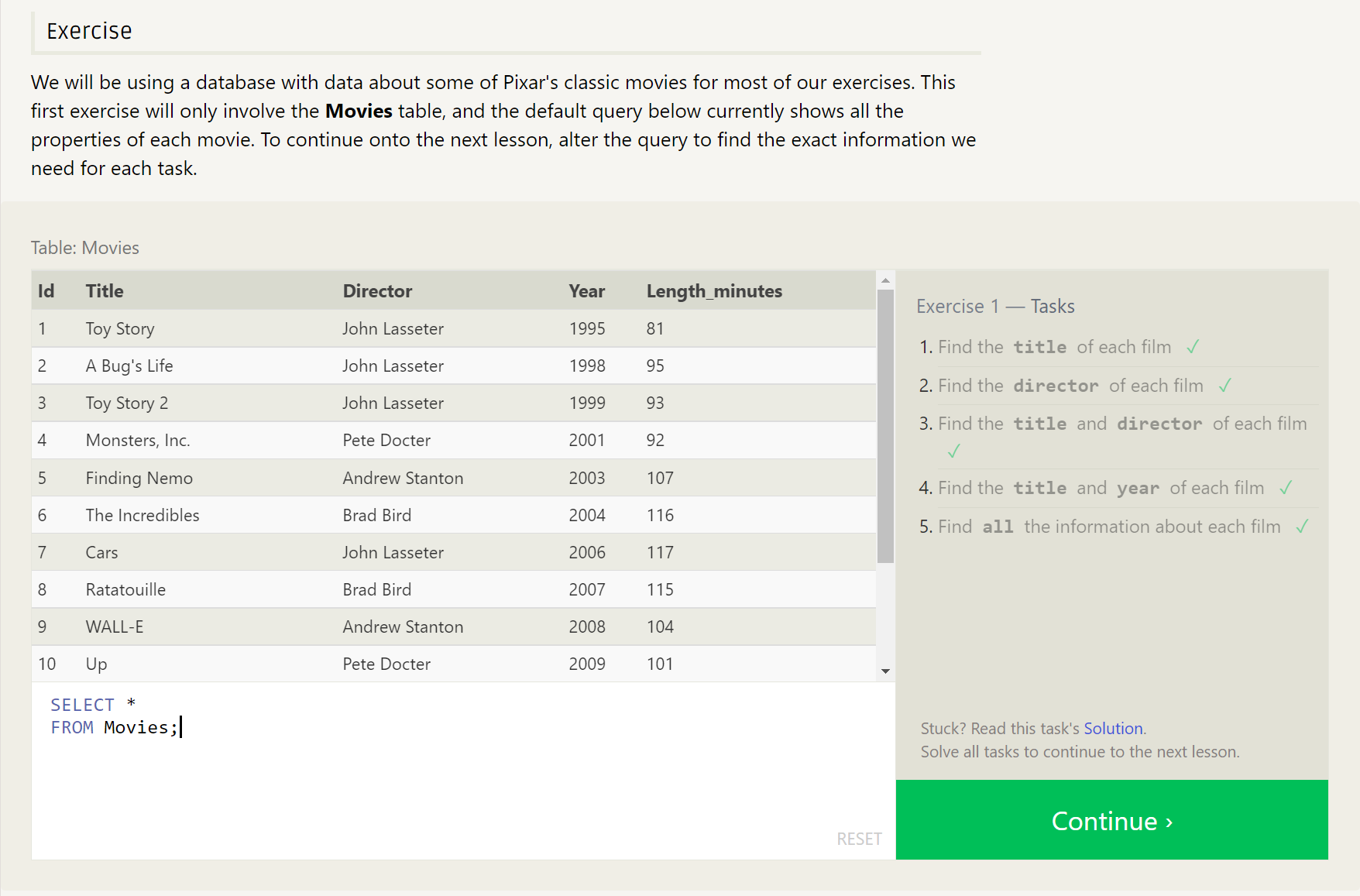
SELECT Title, Director

FROM Movies;

e) Find all the information about each film

SELECT \*

FROM Movies;



**-- CH2 - Queries with constraints (Pt. 1)**

1. Find the movie with a row id of 6

SELECT \*

FROM Movies

WHERE Id = 6;

1. Find the movies released in the years between 2000 and 2010

SELECT \*

FROM Movies

WHERE Year BETWEEN 2000 AND 2010;

1. Find the movies not released in the years between 2000 and 2010

SELECT \*

FROM Movies

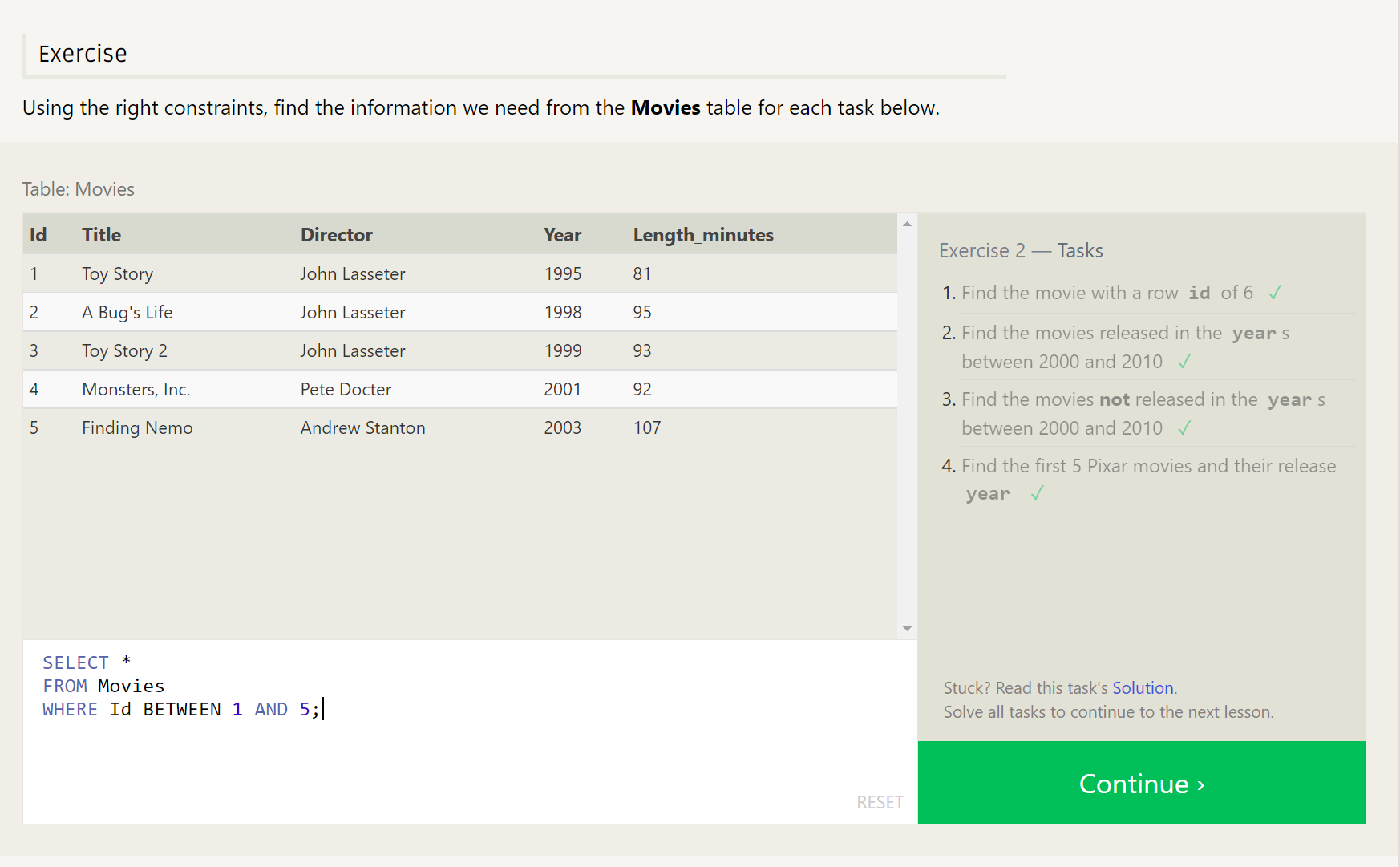
WHERE Year NOT BETWEEN 2000 AND 2010;

1. Find the first 5 Pixar movies and their release year

SELECT \*

FROM Movies

WHERE Id BETWEEN 1 AND 5;



**-- CH3 - Queries with constraints (Pt. 2)**

1. Find all the Toy Story movies

SELECT \*

FROM Movies

WHERE Title LIKE "%Toy Story%";

1. Find all the movies directed by John Lasseter

SELECT \*

FROM Movies

WHERE Director = "John Lasseter";

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

SELECT \*

FROM Movies

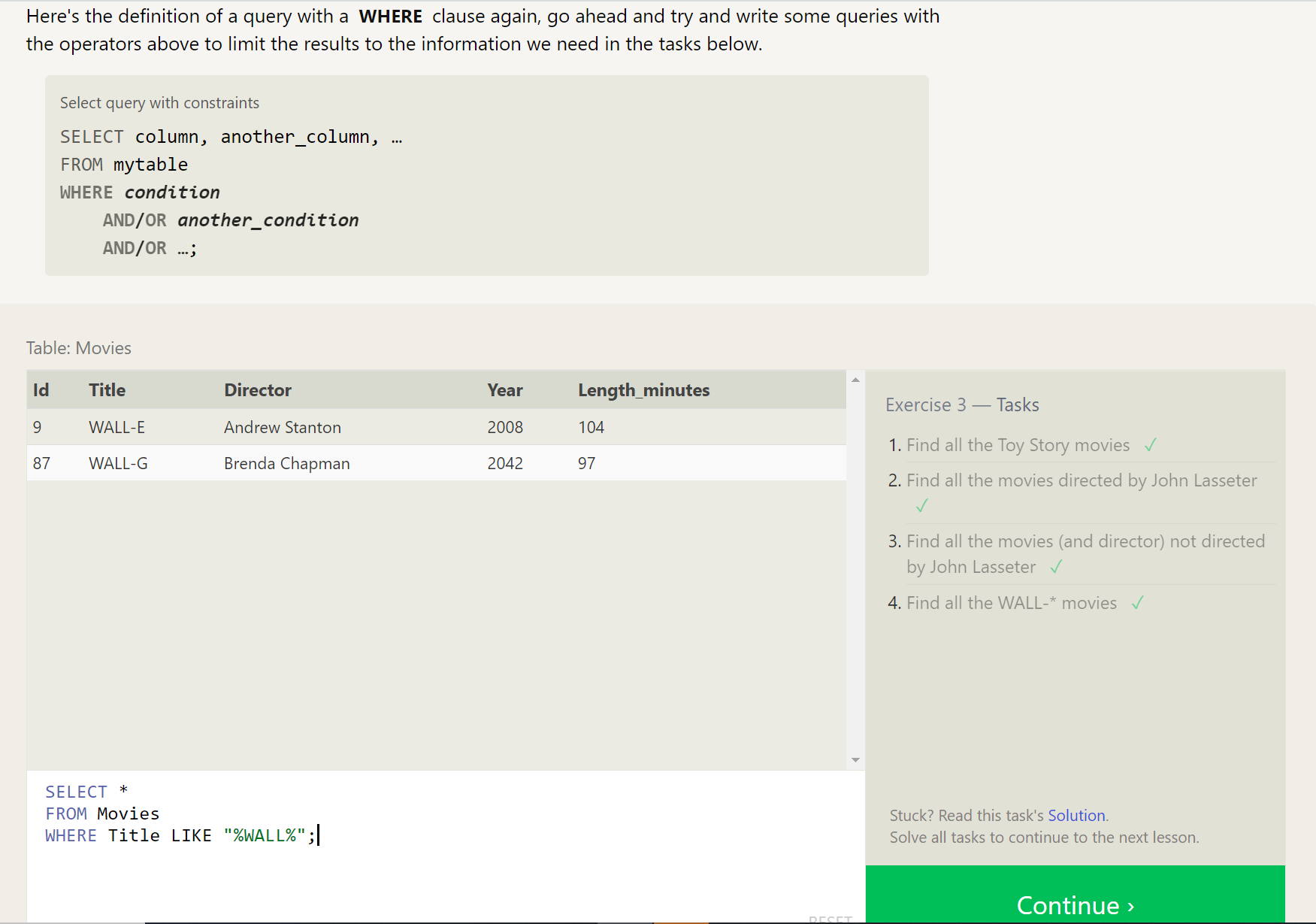
WHERE Director != "John Lasseter";

1. Find all the WALL-\* movies

SELECT \*

FROM Movies

WHERE Title LIKE "%WALL%";



**-- CH4 - Filtering and sorting Query results**

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

SELECT DISTINCT Director

FROM Movies

ORDER BY Director;

1. List the last four Pixar movies released (ordered from most recent to least)

SELECT \*

FROM Movies

ORDER BY Year DESC

LIMIT 4;

1. List the first five Pixar movies sorted alphabetically

SELECT \*

FROM Movies

ORDER BY Title ASC

LIMIT 5;

1. List the next five Pixar movies sorted alphabetically

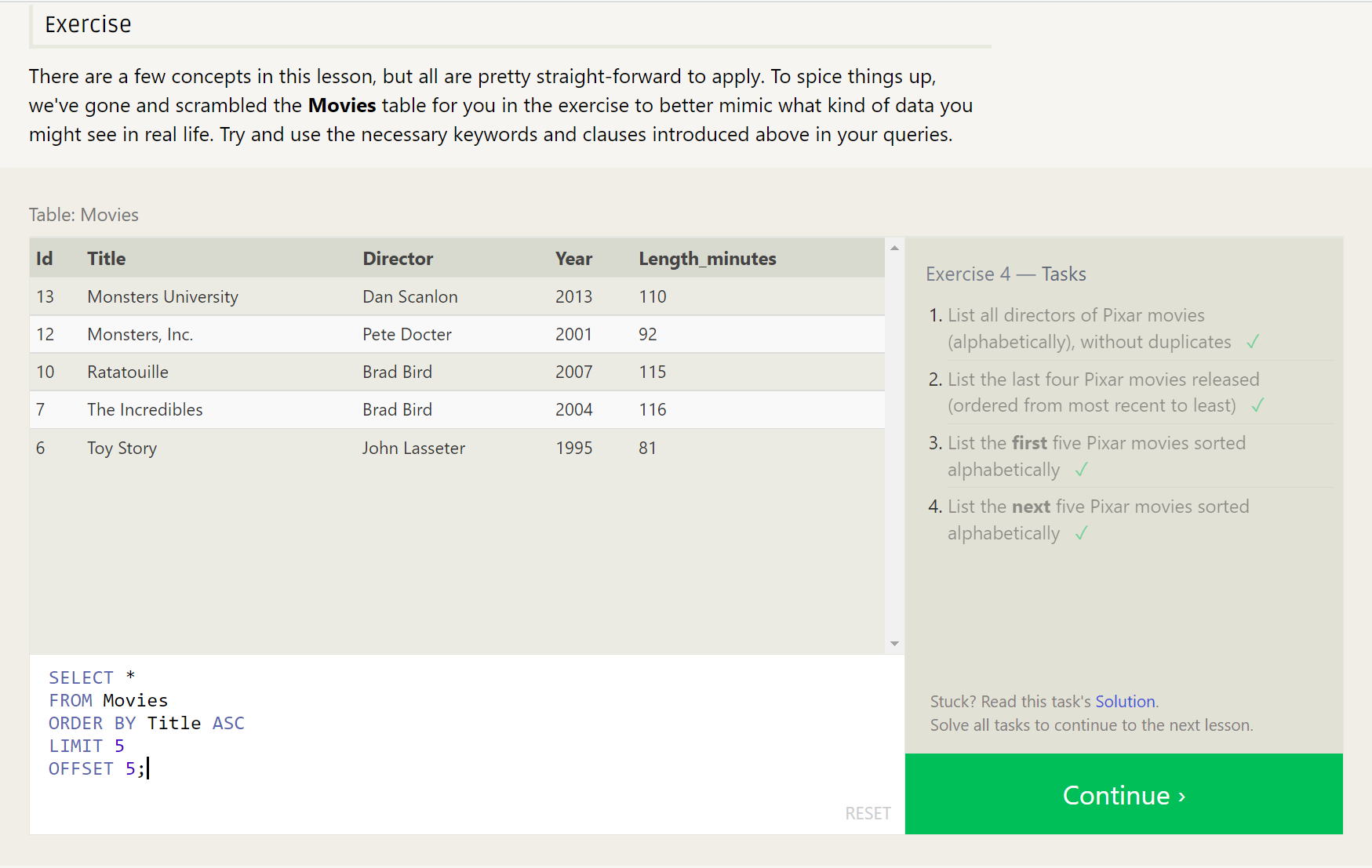
SELECT \*

FROM Movies

ORDER BY Title ASC

LIMIT 5

OFFSET 5;



**-- CH5 - Review Simple SELECT Queries**

a) List all the Canadian cities and their populations

SELECT \* FROM North\_american\_cities

WHERE Country LIKE "Canada";

b) Order all the cities in the United States by their latitude from north to south

SELECT \* FROM North\_american\_cities

WHERE Country = "United States"

ORDER BY Latitude DESC;

c) List all the cities west of Chicago, ordered from west to east

SELECT \* FROM North\_american\_cities

WHERE Longitude < -87.69

ORDER BY Longitude ASC;

d) List the two largest cities in Mexico (by population)

SELECT \* FROM North\_american\_cities

WHERE Country LIKE "Mexico"

ORDER BY Population DESC

LIMIT 2;

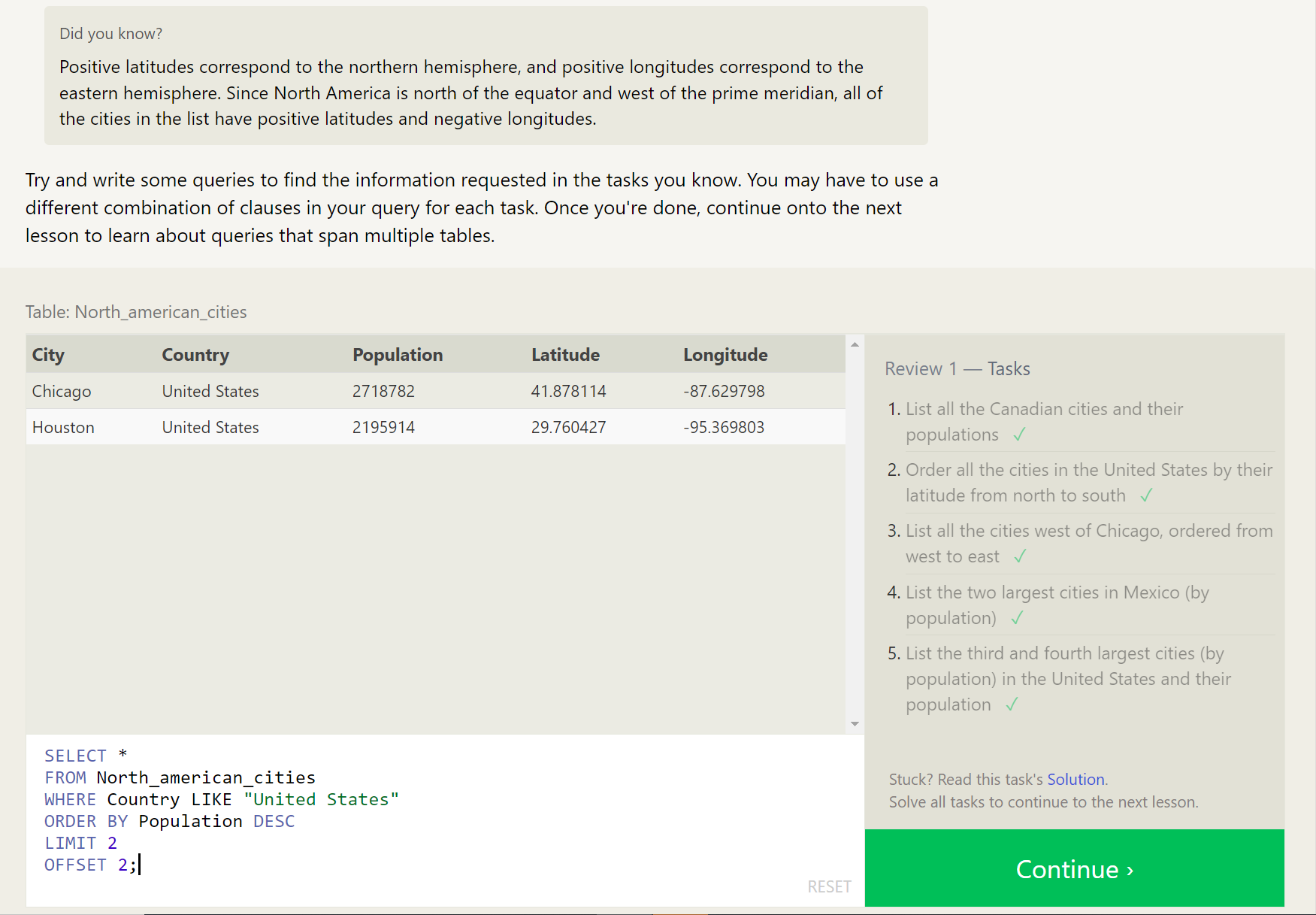
1. List the third and fourth largest cities (by population) in the United States and their population

SELECT \* FROM North\_american\_cities

WHERE Country LIKE "United States"

ORDER BY Population DESC

LIMIT 2 OFFSET 2;



**-- CH6 - Multi-table queries with JOINs**

1. Find the domestic and international sales for each movie

SELECT Title, International\_sales, Domestic\_sales

FROM Movies JOIN Boxoffice

ON Id=Movie\_id;

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

SELECT Title, International\_sales, Domestic\_sales

FROM Movies JOIN Boxoffice

ON Id=Movie\_id

WHERE International\_sales > Domestic\_sales;

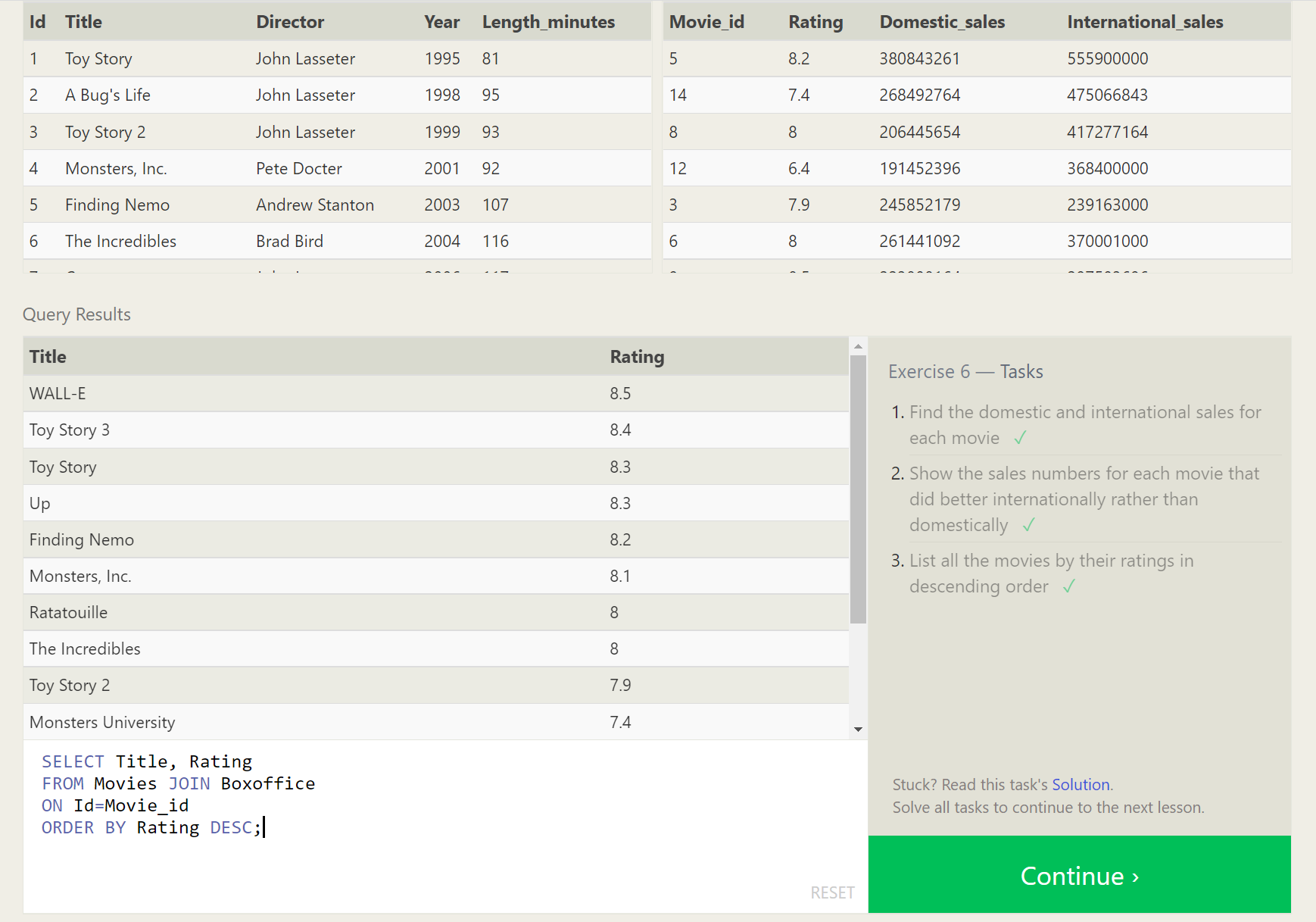
1. List all the movies by their ratings in descending order

SELECT Title, Rating

FROM Movies JOIN Boxoffice

ON Id=Movie\_id

ORDER BY Rating DESC;



**-- CH7 - OUTER JOIN**

1. Find the list of all buildings that have employees

SELECT DISTINCT Building

FROM Employees

LEFT JOIN Buildings ON Building=Building\_name

WHERE Years\_employed NOT NULL;

1. Find the list of all buildings and their capacity

SELECT \*

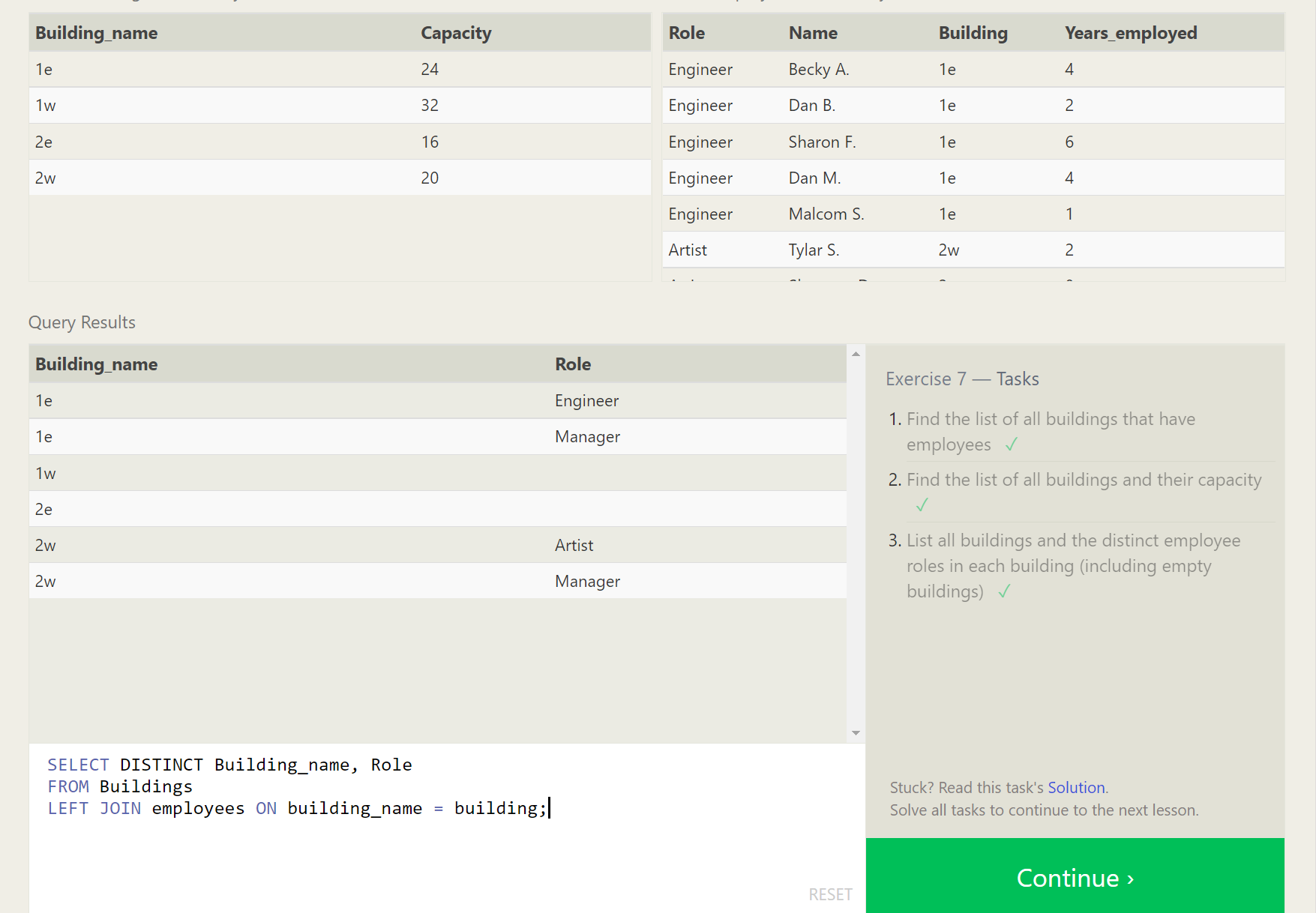
FROM Buildings;

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

SELECT DISTINCT Building\_name, Role

FROM Buildings

LEFT JOIN employees ON building\_name = building;



**-- CH8 - A short note on NULLs**

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

SELECT \*

FROM Employees

LEFT JOIN Buildings

ON Building\_name = Building

WHERE Building IS NULL;

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

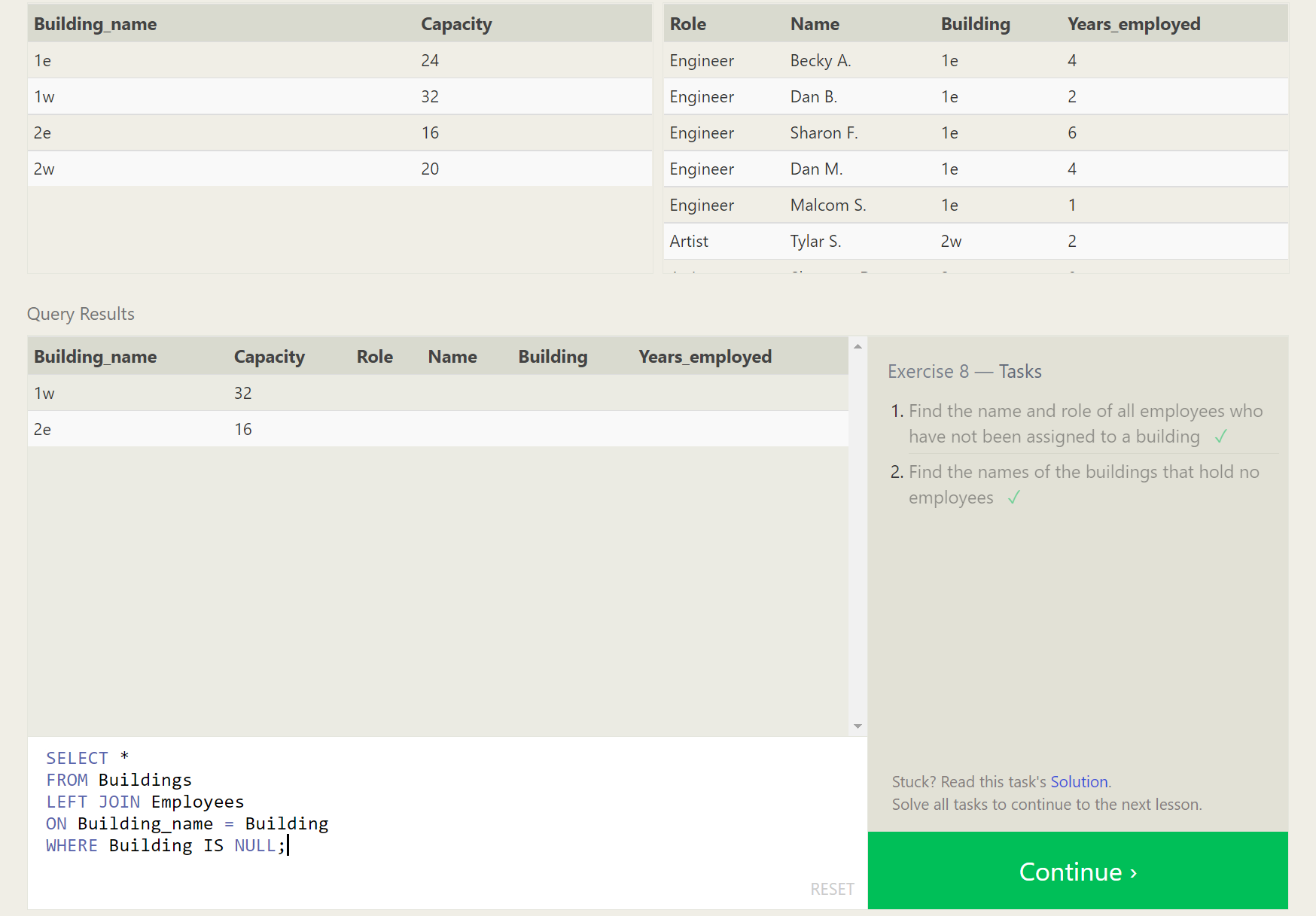
SELECT \*

FROM Buildings

LEFT JOIN Employees

ON Building\_name = Building

WHERE Building IS NULL;



**-- CH9 - Queries with expressions**

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

SELECT Title, (Domestic\_sales + International\_sales)/1000000 AS Total\_Sales\_Millions

FROM Movies

LEFT JOIN Boxoffice ON Id=Movie\_Id;

1. List all movies and their ratings in percent

SELECT Title, Rating\*10 as Percent

FROM Movies

LEFT JOIN Boxoffice ON Id=Movie\_Id;

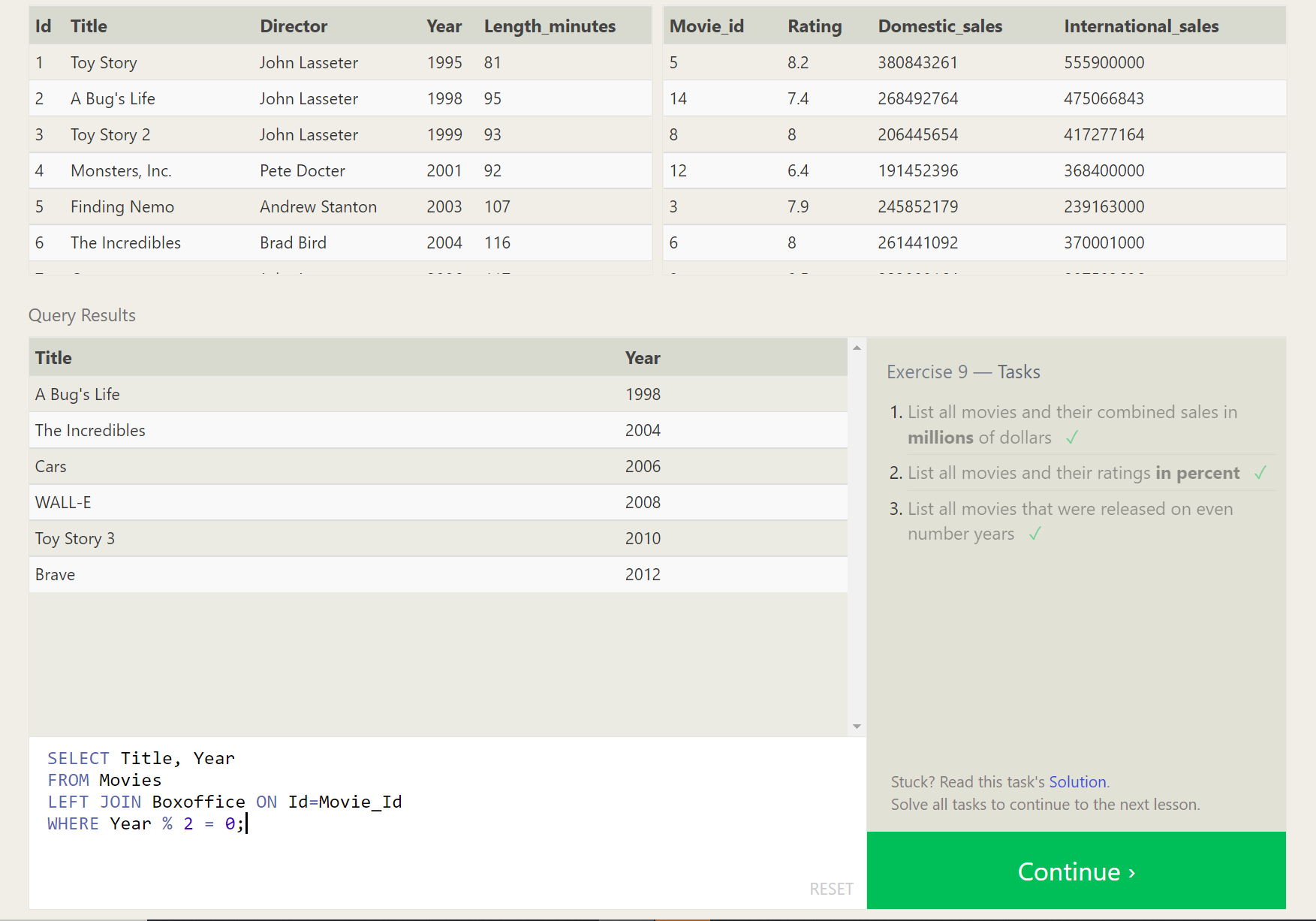
1. List all movies that were released on even number years

SELECT Title, Year

FROM Movies

LEFT JOIN Boxoffice ON Id=Movie\_Id

WHERE Year % 2 = 0;



**-- CH10 - Queries with aggregates (Pt. 1)**

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

SELECT MAX(Years\_employed)

FROM Employees;

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

SELECT Role, AVG(Years\_Employed)

FROM Employees

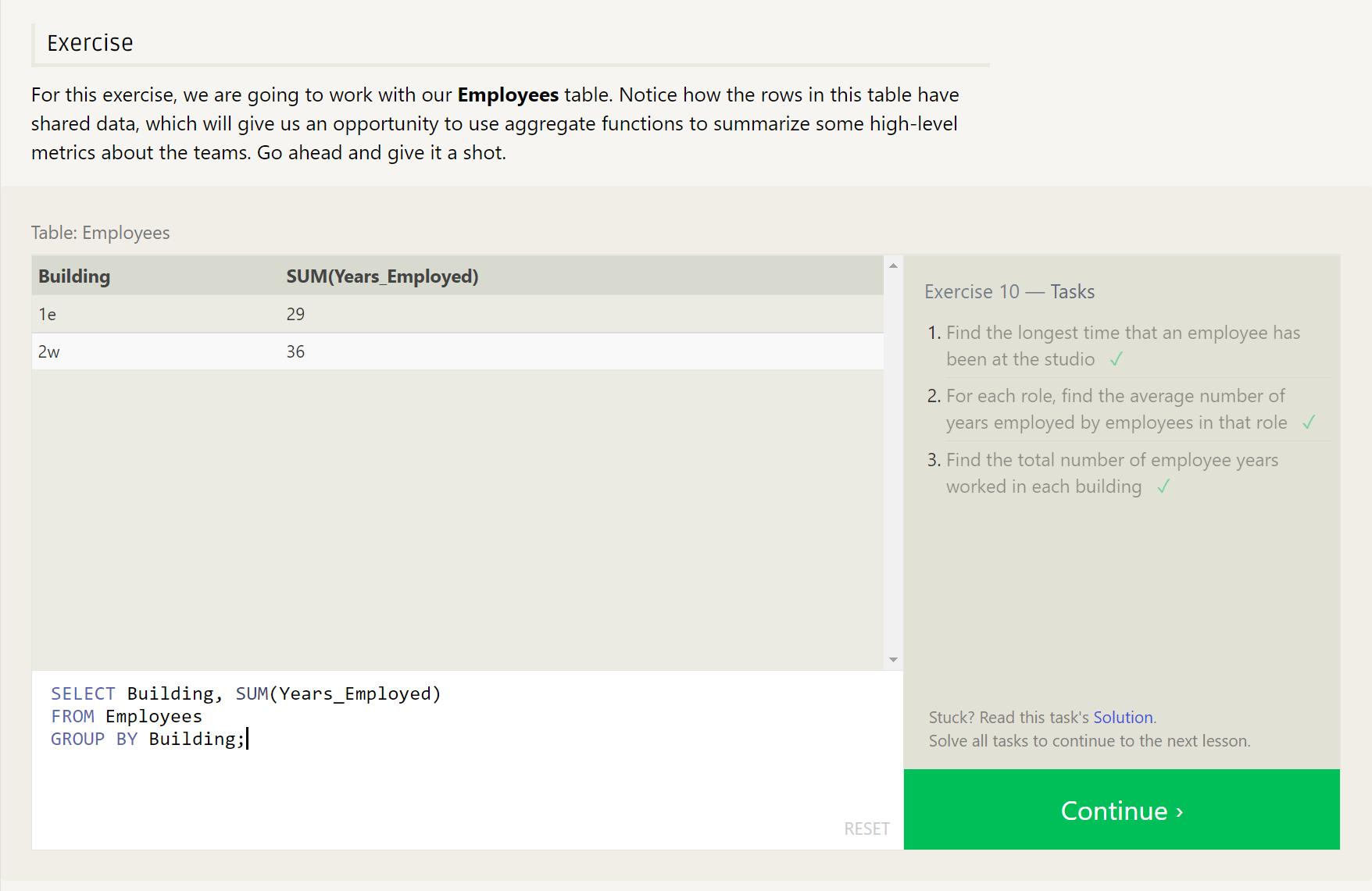
GROUP BY Role;

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

SELECT Building, SUM(Years\_Employed)

FROM Employees

GROUP BY Building;



**-- CH11 - Queries with aggregates (Pt. 2)**

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

SELECT Role, COUNT(\*) AS Number\_of\_Artists

FROM Employees

WHERE Role = "Artist";

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

SELECT Role, COUNT(\*)

FROM Employees

GROUP BY Role;

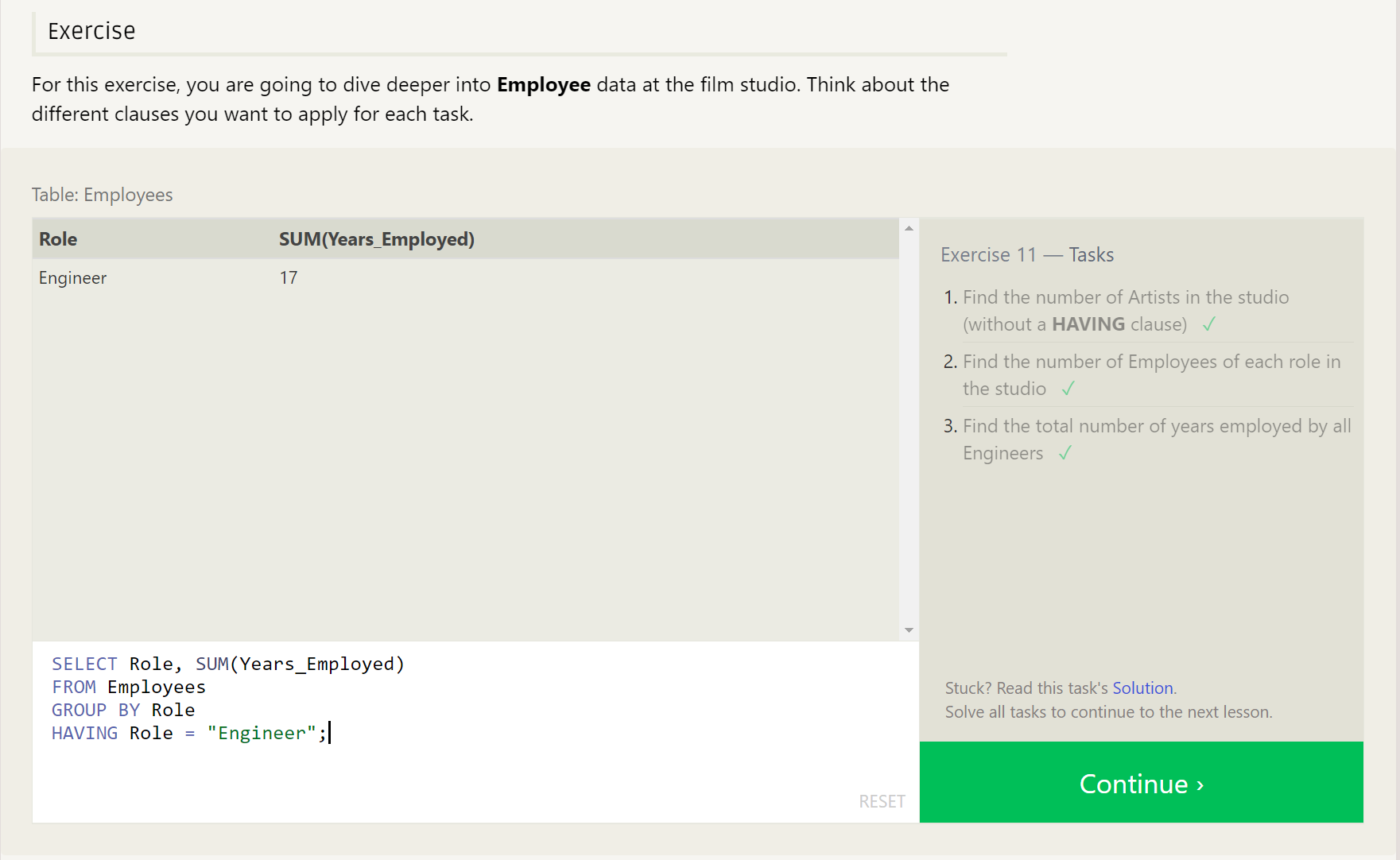
1. Find the total number of years employed by all Engineers

SELECT Role, SUM(Years\_Employed)

FROM Employees

GROUP BY Role

HAVING Role = "Engineer";



**-- CH12 - Order of execution of a Query**

1. Find the number of movies each director has directed

SELECT \*, COUNT(Title)

FROM Movies

GROUP BY Director;

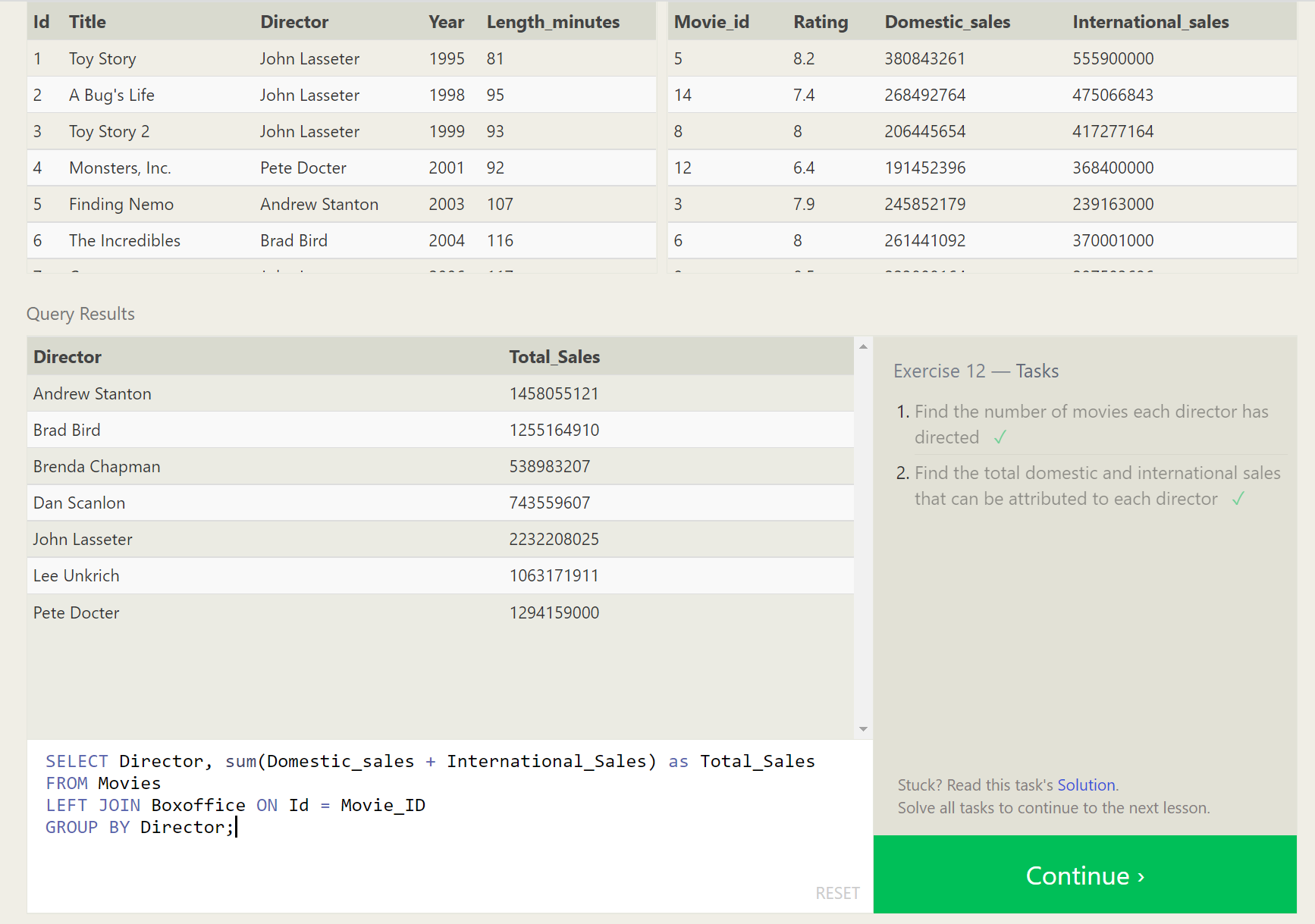
1. Find the total domestic and international sales that can be attributed to each director

SELECT Director, SUM(Domestic\_sales + International\_Sales) as Total\_Sales

FROM Movies

LEFT JOIN Boxoffice ON Id = Movie\_ID

GROUP BY Director;



**-- CH13 - Inserting rows**

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

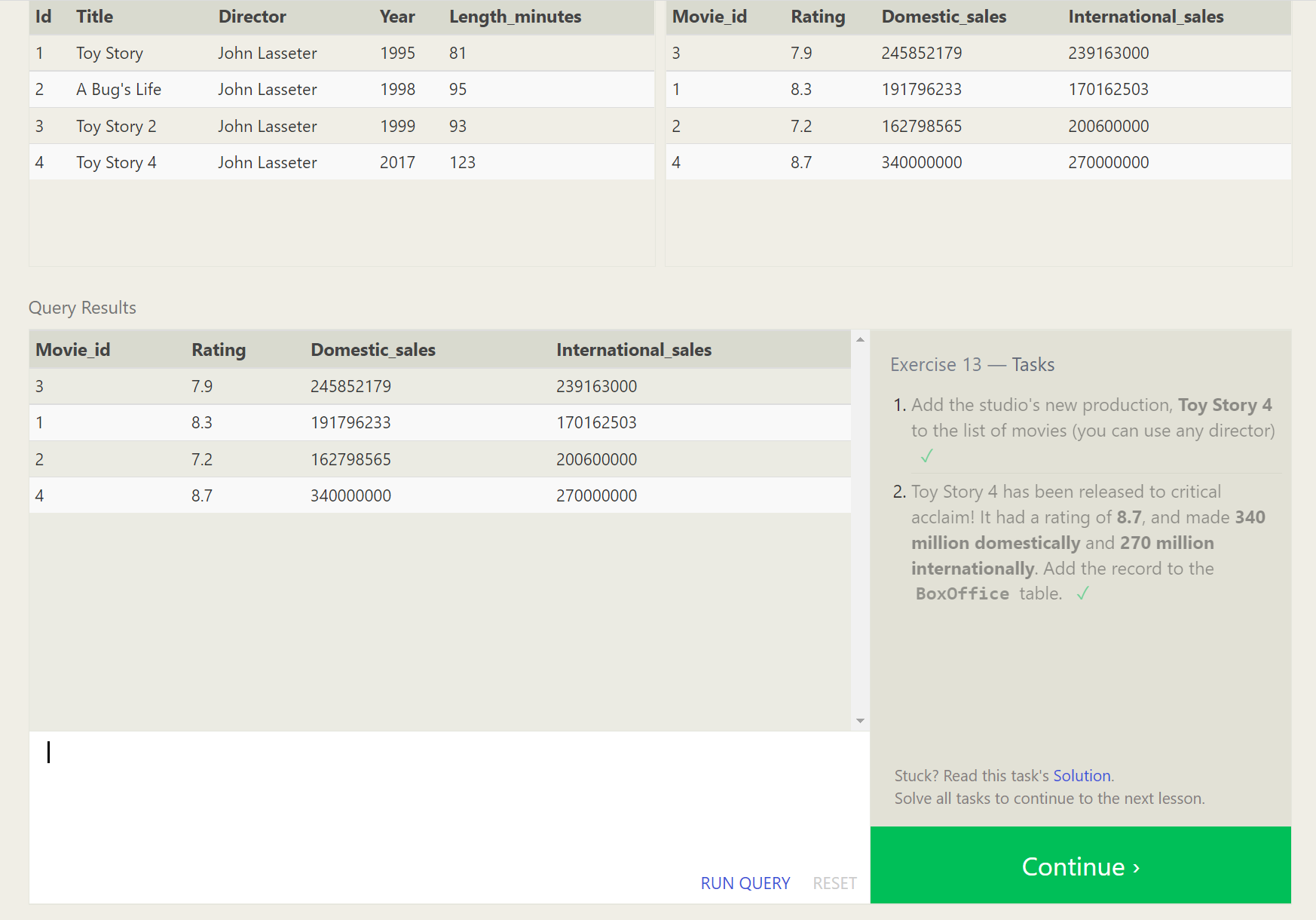
INSERT INTO Movies,

VALUES (4, "Toy Story 4", "John Lasseter", 2017, 123);

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 BoxOffice table.

INSERT INTO Boxoffice

VALUES (4, 8.7, 340000000, 270000000);



**-- CH14 - Updating rows**

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

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

UPDATE Movies

SET Year = "1999"

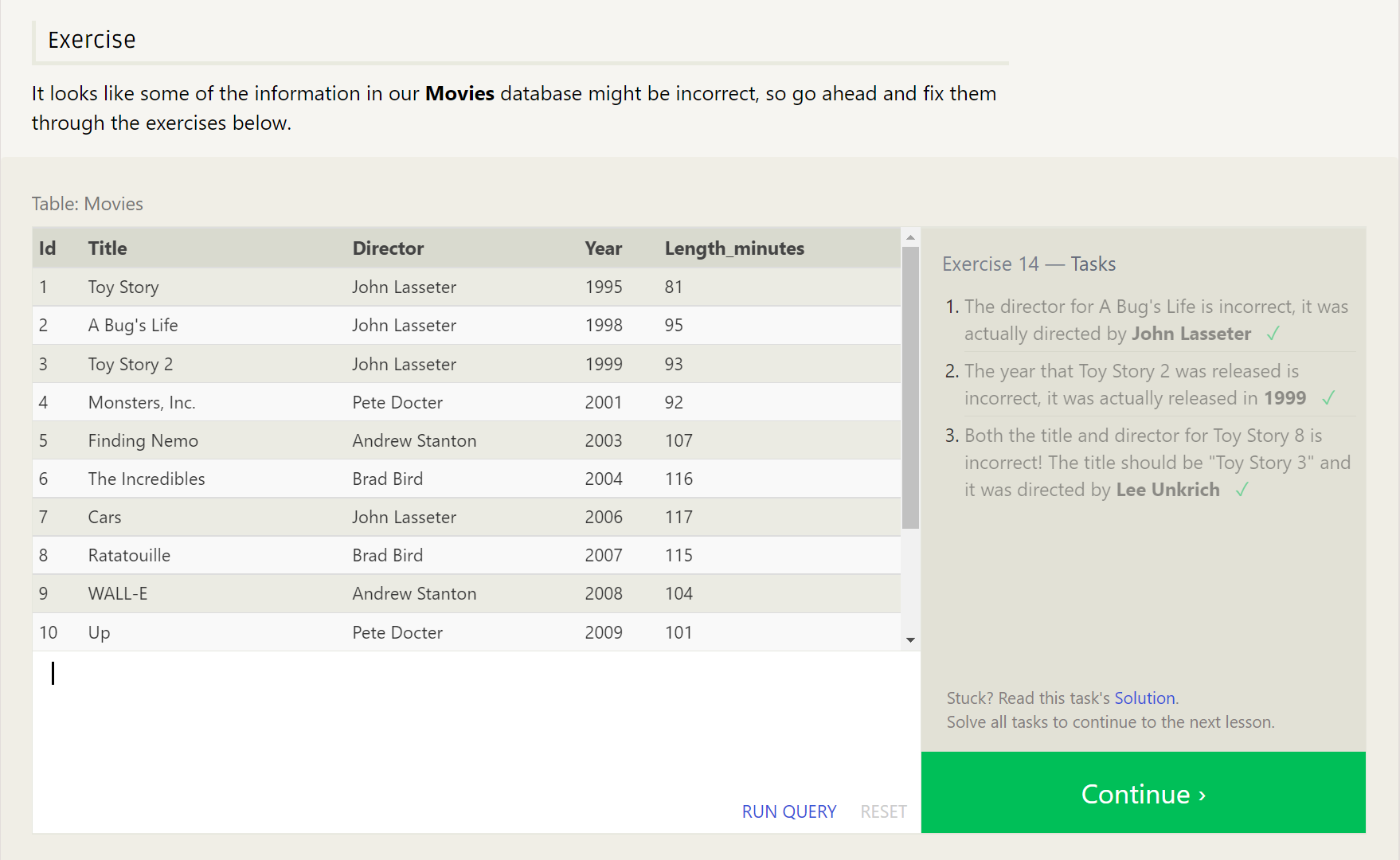
WHERE Id = 3;

1. Both the title and directory for Toy Story 8 is incorrect! The title should be "Toy Story 3" and it was directed by Lee Unkrich

UPDATE Movies

SET Title = "Toy Story 3", Director = "Lee Unkrich"

WHERE Id = 11;



**-- CH15 - Deleting rows**

1. This database is getting too big, lets remove all movies that were released before 2005.

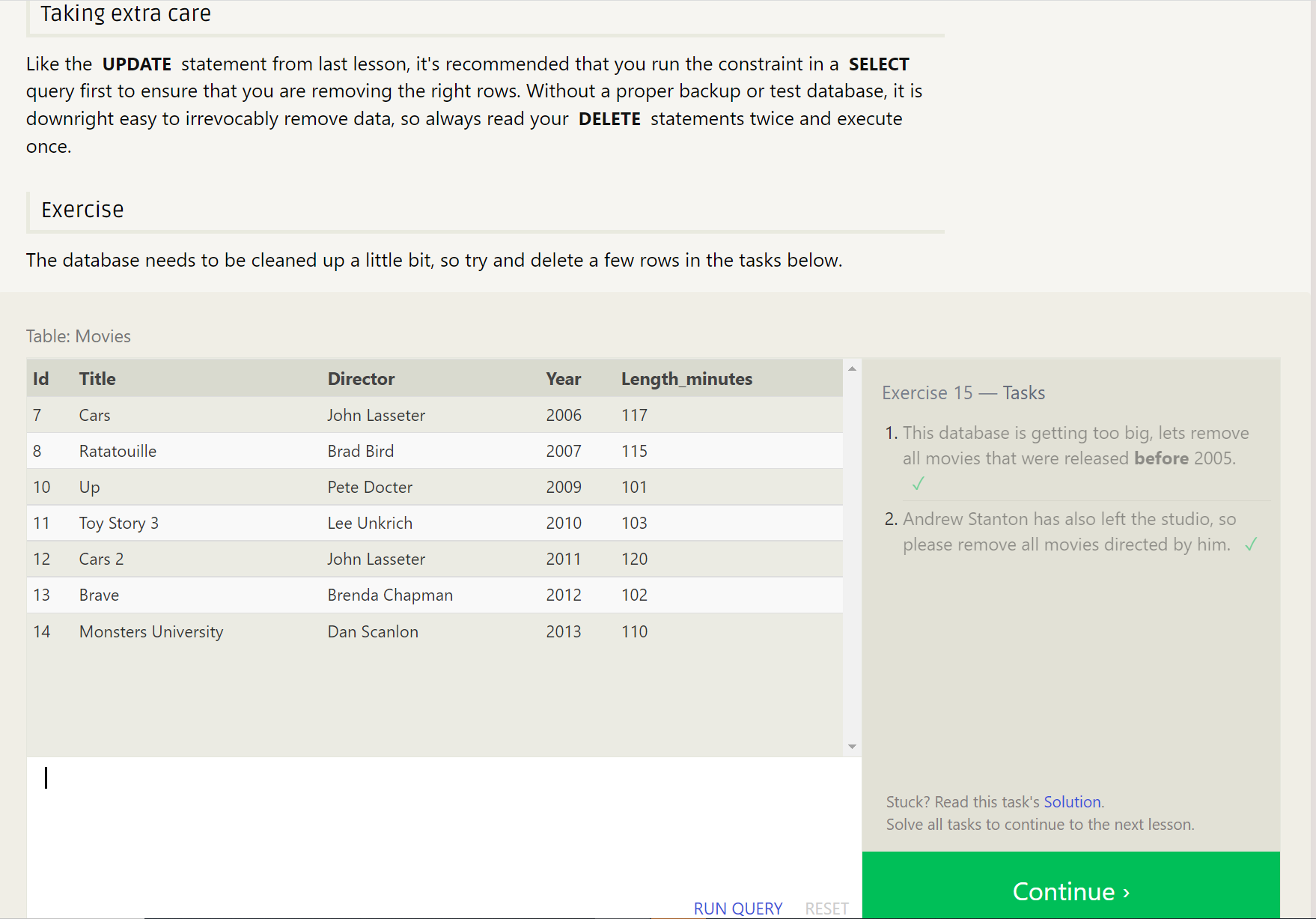
DELETE FROM Movies

WHERE Year < 2005;

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

DELETE FROM Movies

WHERE Director = "Andrew Stanton";



**-- CH16 - Creating Tables**

-- Create a new table named Database with the following columns:

-- 1. Name A string (text) describing the name of the database

-- 2. Version A number (floating point) of the latest version of this database

-- 3. Download\_count An integer count of the number of times this database was downloaded

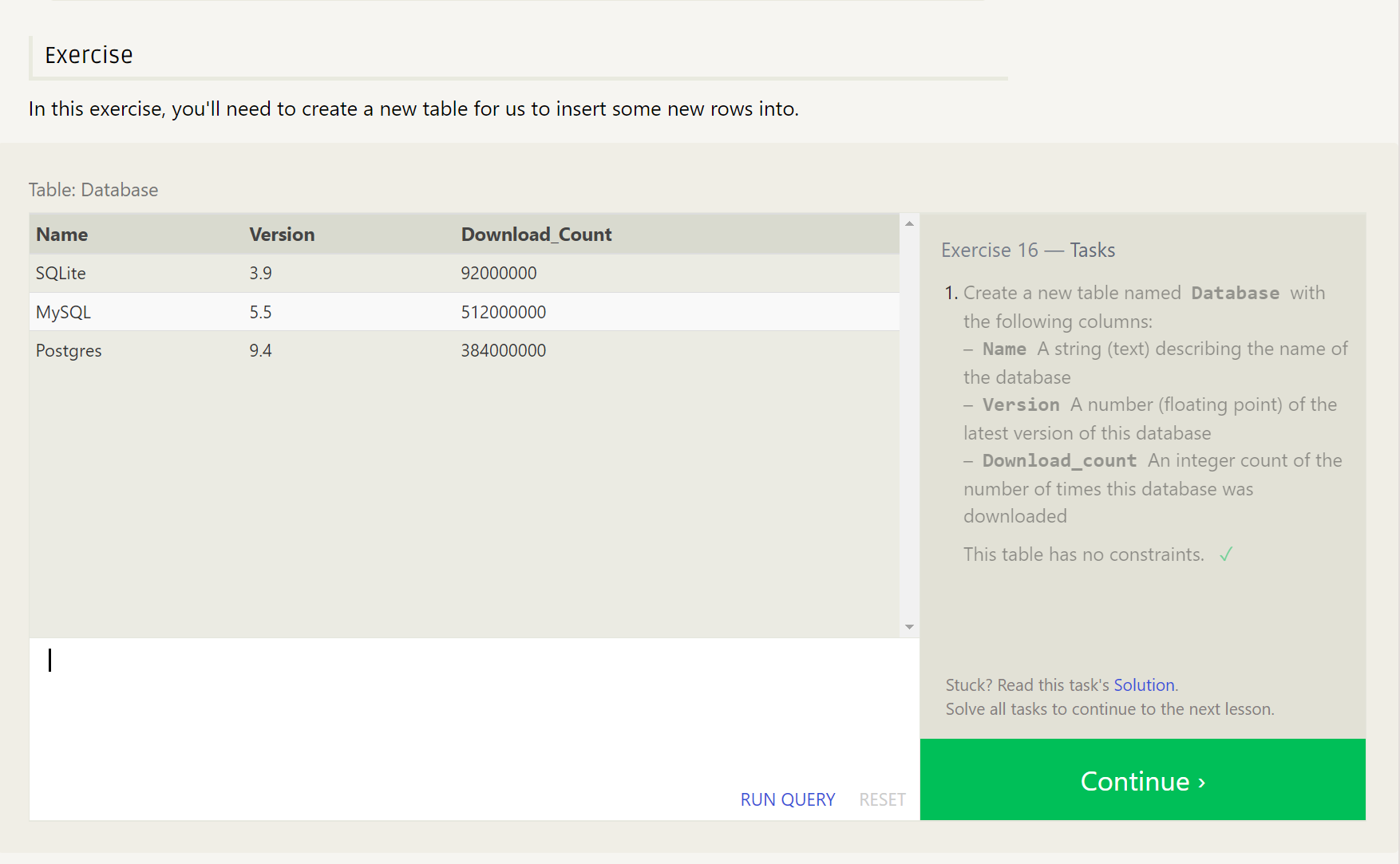
-- This table has no constraints.

CREATE TABLE Database (

Name TEXT,

Version FLOAT,

Download\_Count INTEGER);



**-- CH17 - Altering Tables**

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

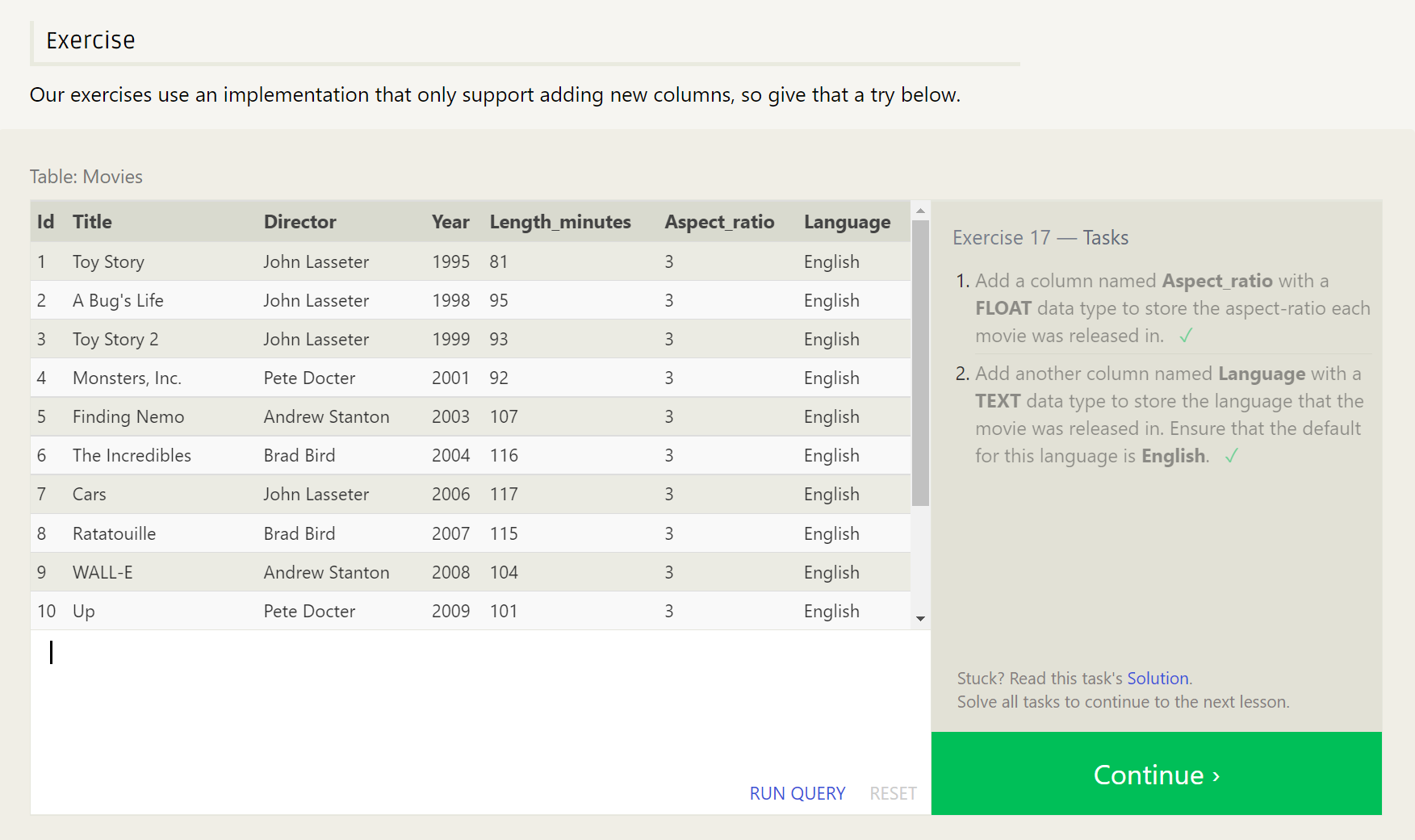
ALTER TABLE Movies

ADD COLUMN Aspect\_ratio FLOAT DEFAULT 3;

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.

ALTER TABLE Movies

ADD COLUMN Language TEXT DEFAULT "English";



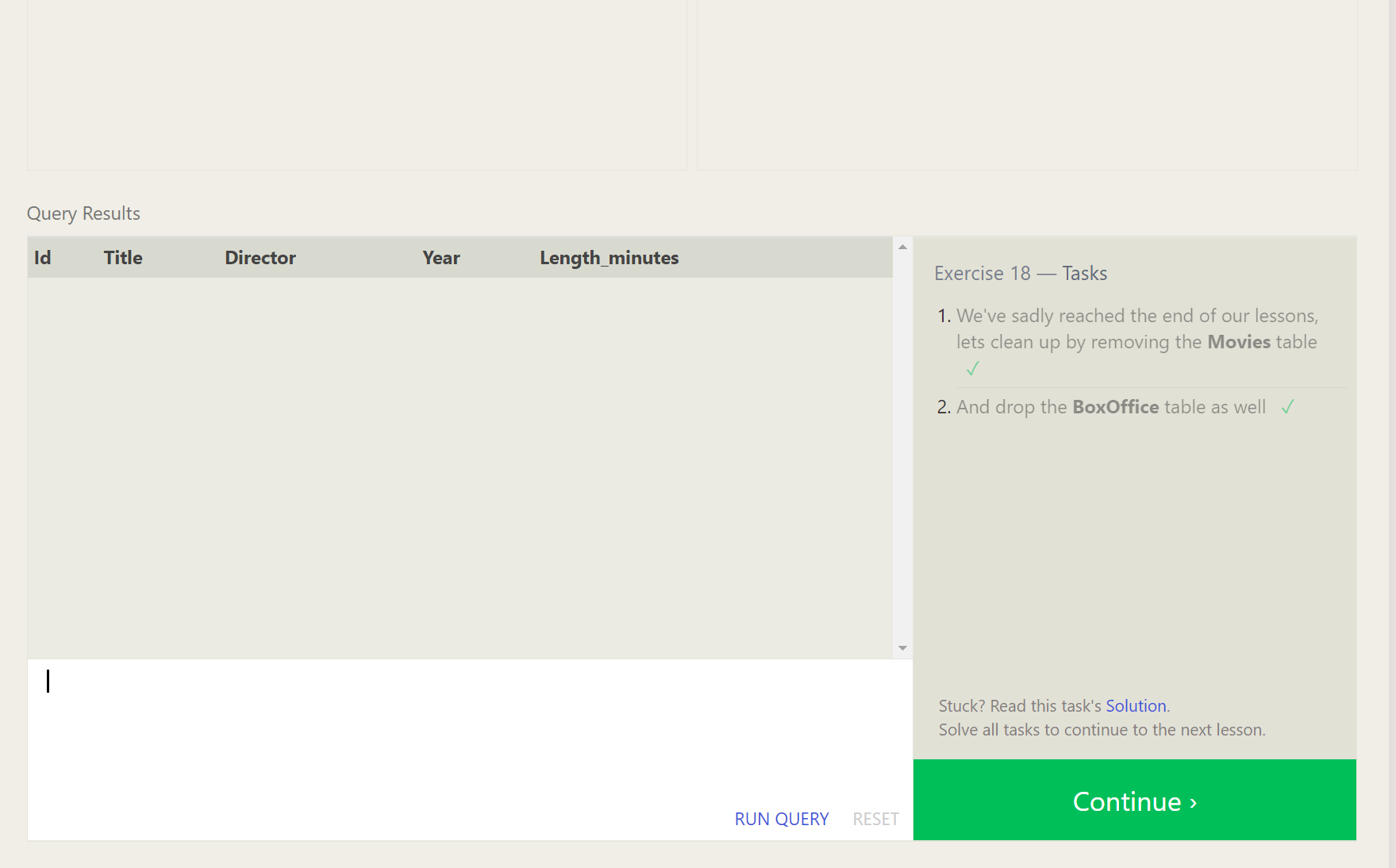
**-- CH18 - Dropping Tables**

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

DROP TABLE Movies;

1. And drop the BoxOffice table as well

DROP TABLE BoxOffice;



Finished…….

