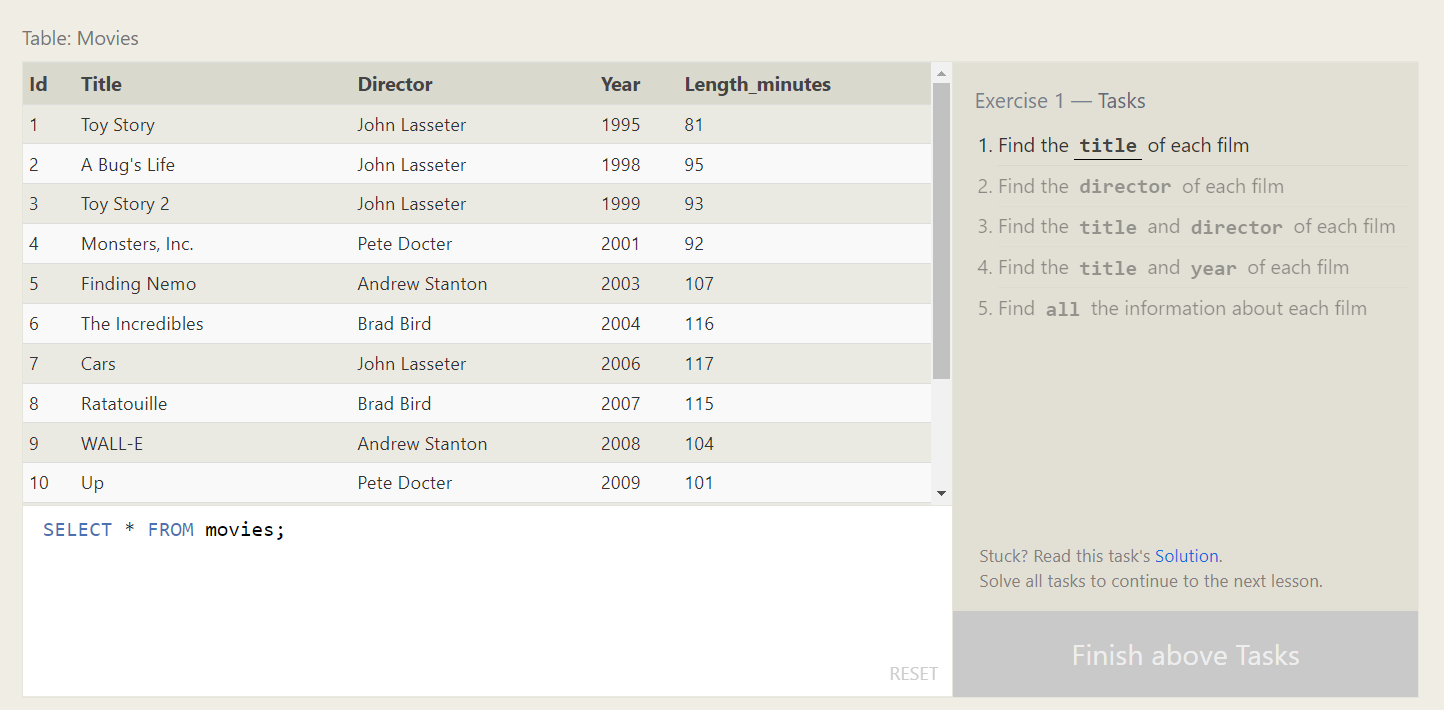
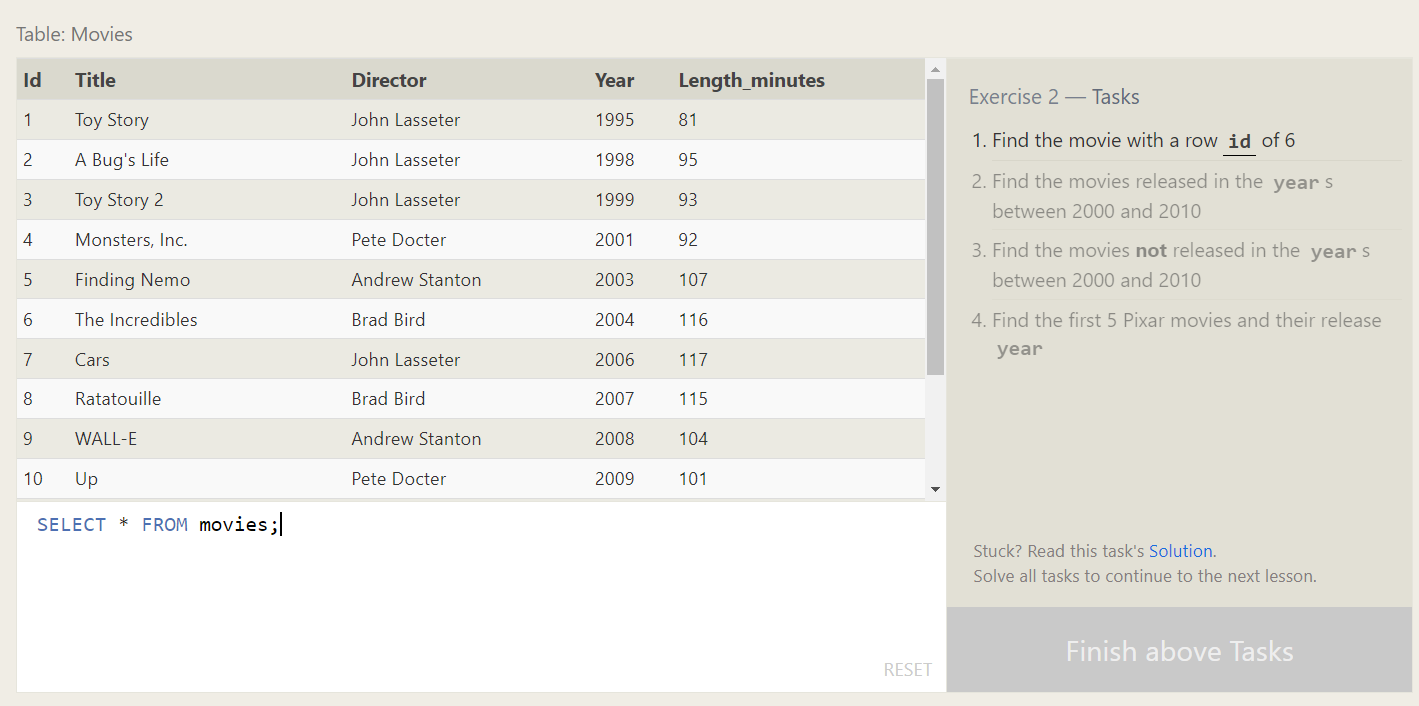
**SQL Lesson 1: SELECT queries 101**



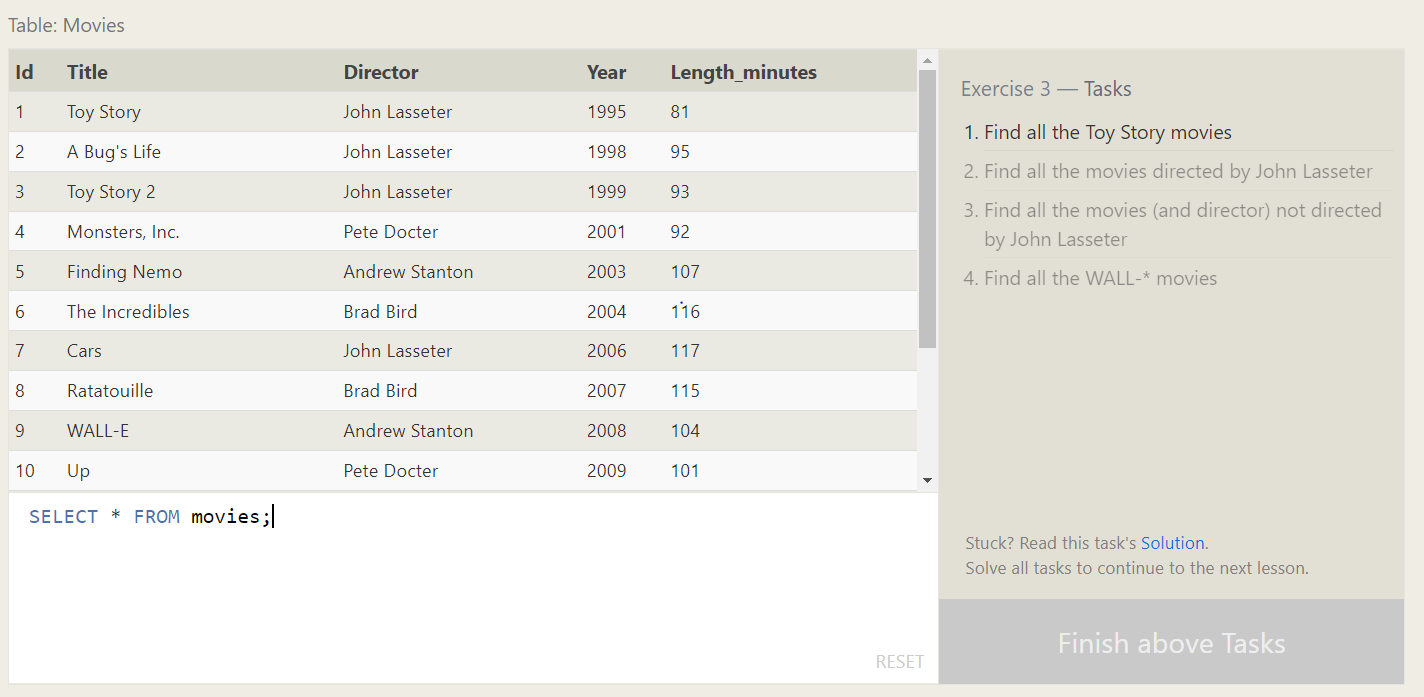
1. SELECT Title FROM movies;
2. SELECT director FROM movies;
3. SELECT title,director FROM movies;
4. SELECT title,year FROM movies;
5. SELECT \* FROM movies;

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



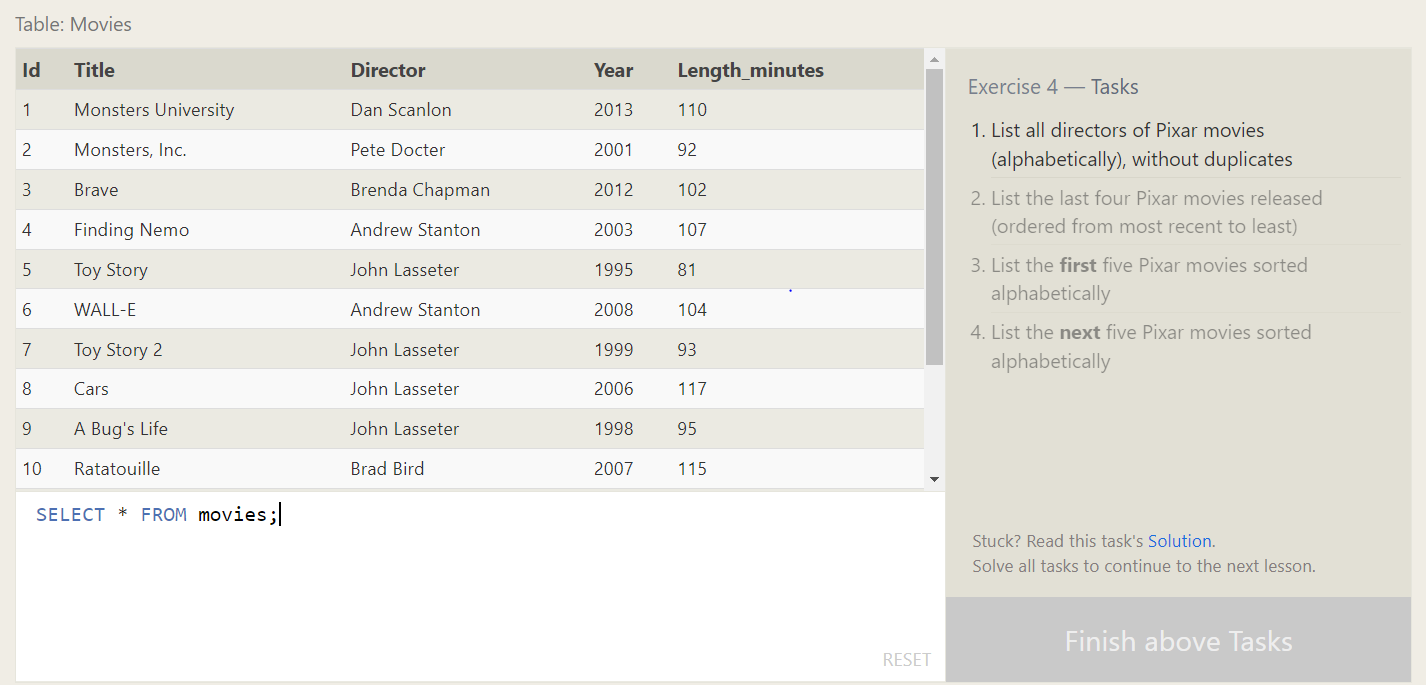
1. SELECT \* FROM Movies WHERE Id = 6;
2. SELECT \* FROM Movies where year between 2000 and 2010;
3. SELECT \* FROM Movies where year not between 2000 and 2010;
4. SELECT Id, Title, Year FROM Movies ORDER BY Id LIMIT 5;

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



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**



1. SELECT DISTINCT Director

FROM movies

ORDER BY Director;

2. SELECT Title, Director, Year

FROM movies

ORDER BY Year DESC

LIMIT 4;

3. SELECT Title, Director, Year

FROM movies

ORDER BY Title

LIMIT 5;

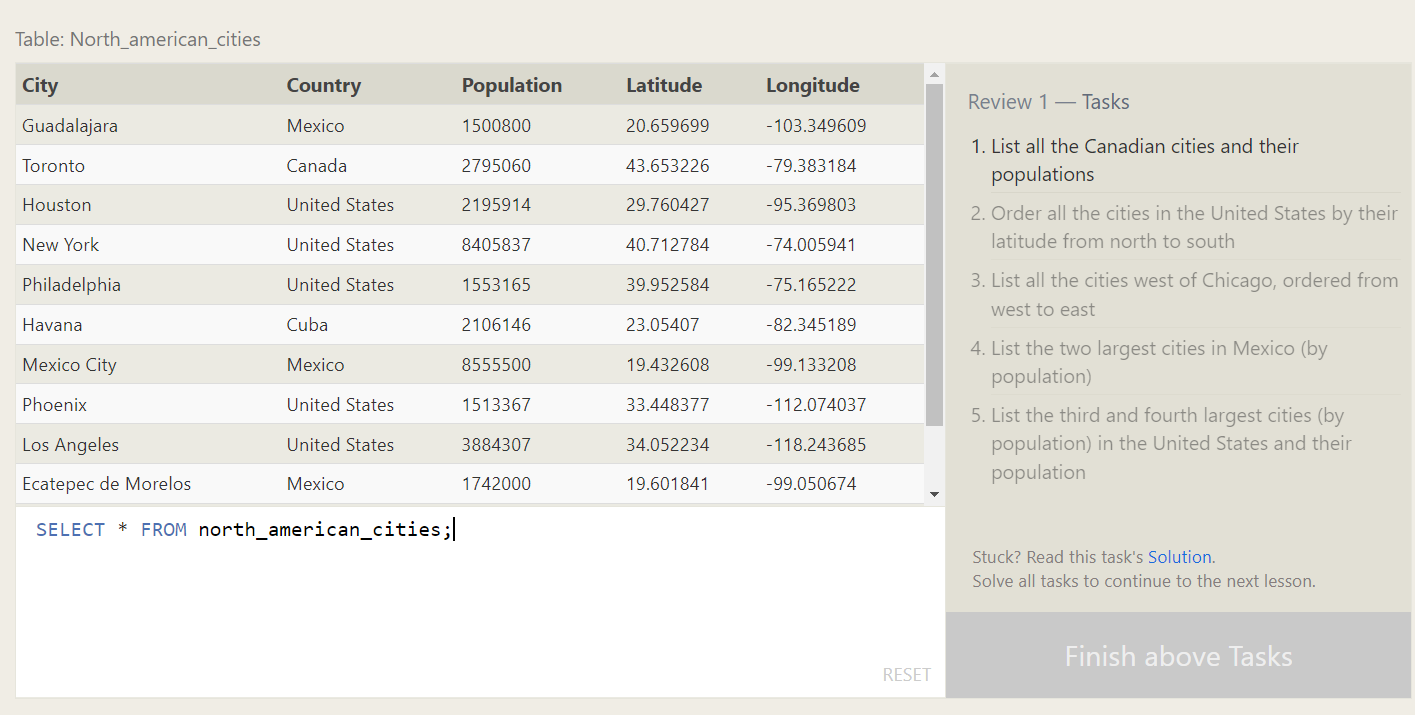
4. SELECT Title, Director, Year

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, Latitude

FROM North\_american\_cities

WHERE Country = 'United States'

ORDER BY Latitude DESC;

4. SELECT city, population FROM north\_american\_cities

WHERE country LIKE "Mexico"

ORDER BY population DESC

LIMIT 2;

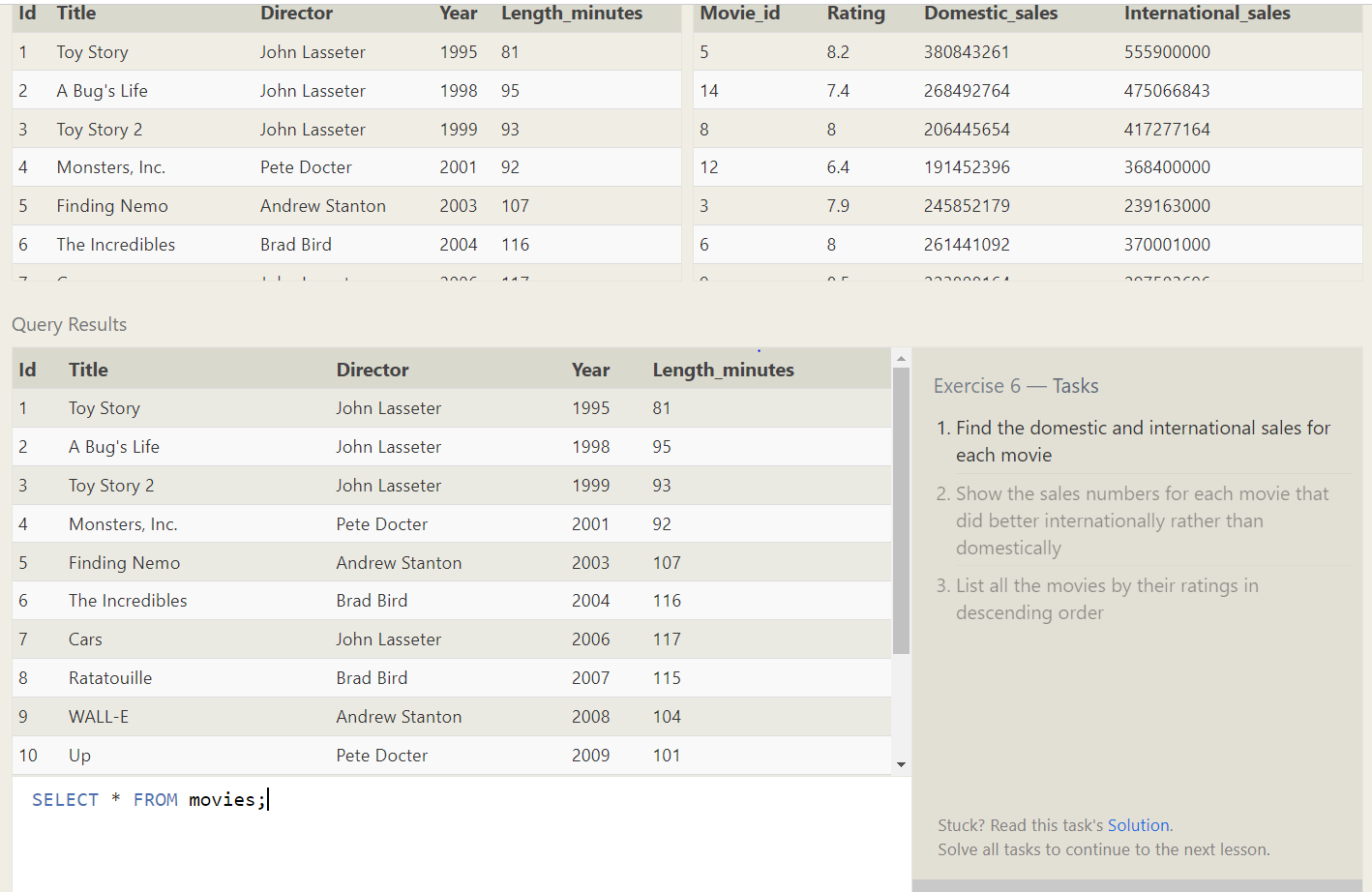
5. SELECT city, population FROM north\_american\_cities

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 movies.id = boxoffice.movie\_id;

2. SELECT title, domestic\_sales, international\_sales

FROM movies

JOIN boxoffice

ON movies.id = boxoffice.movie\_id where International\_sales>Domestic\_sales;

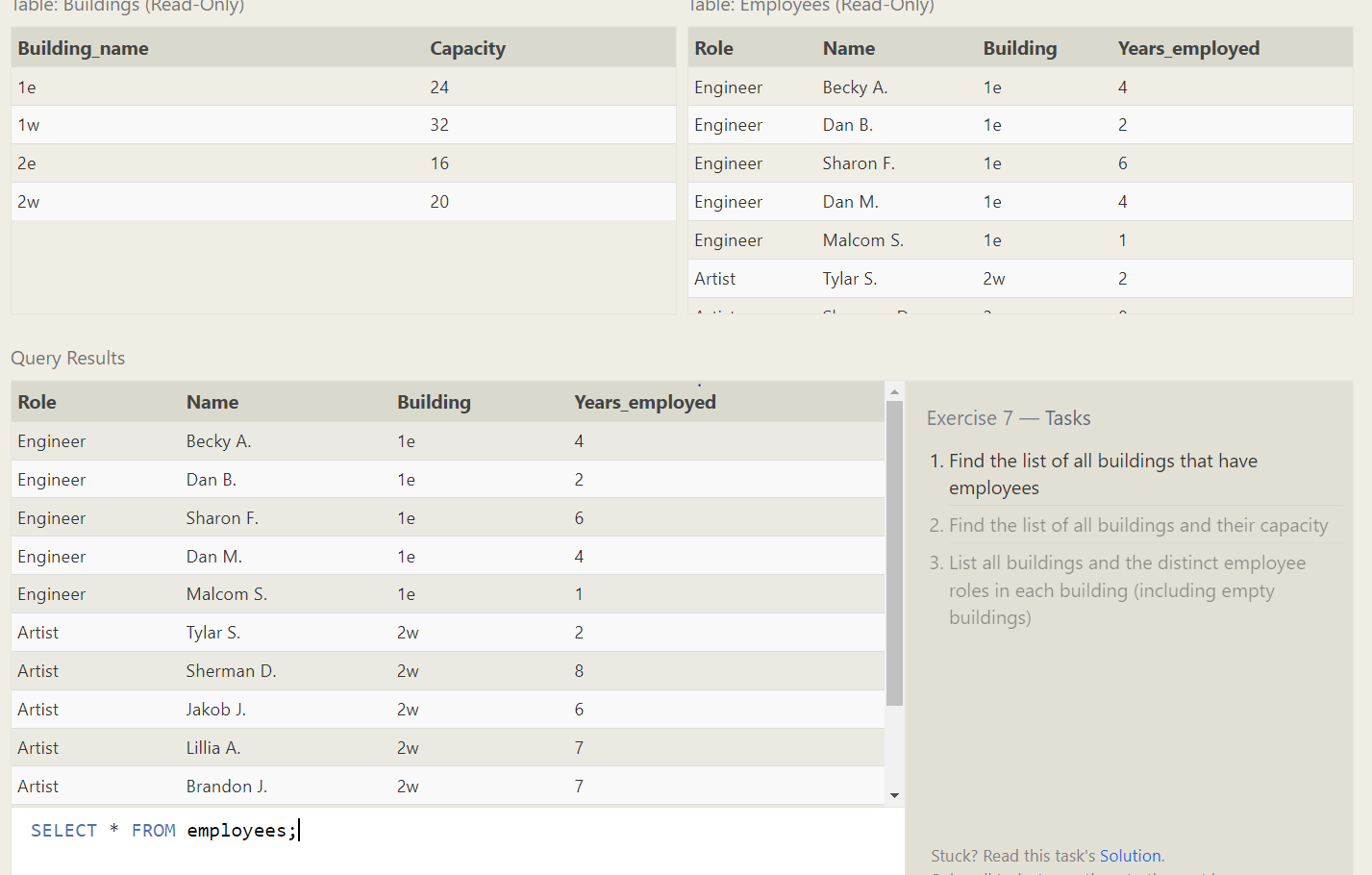
3. SELECT title, domestic\_sales, international\_sales

FROM movies

JOIN boxoffice

ON movies.id = boxoffice.movie\_id where International\_sales>Domestic\_sales;

**SQL Lesson 7: OUTER JOINs**



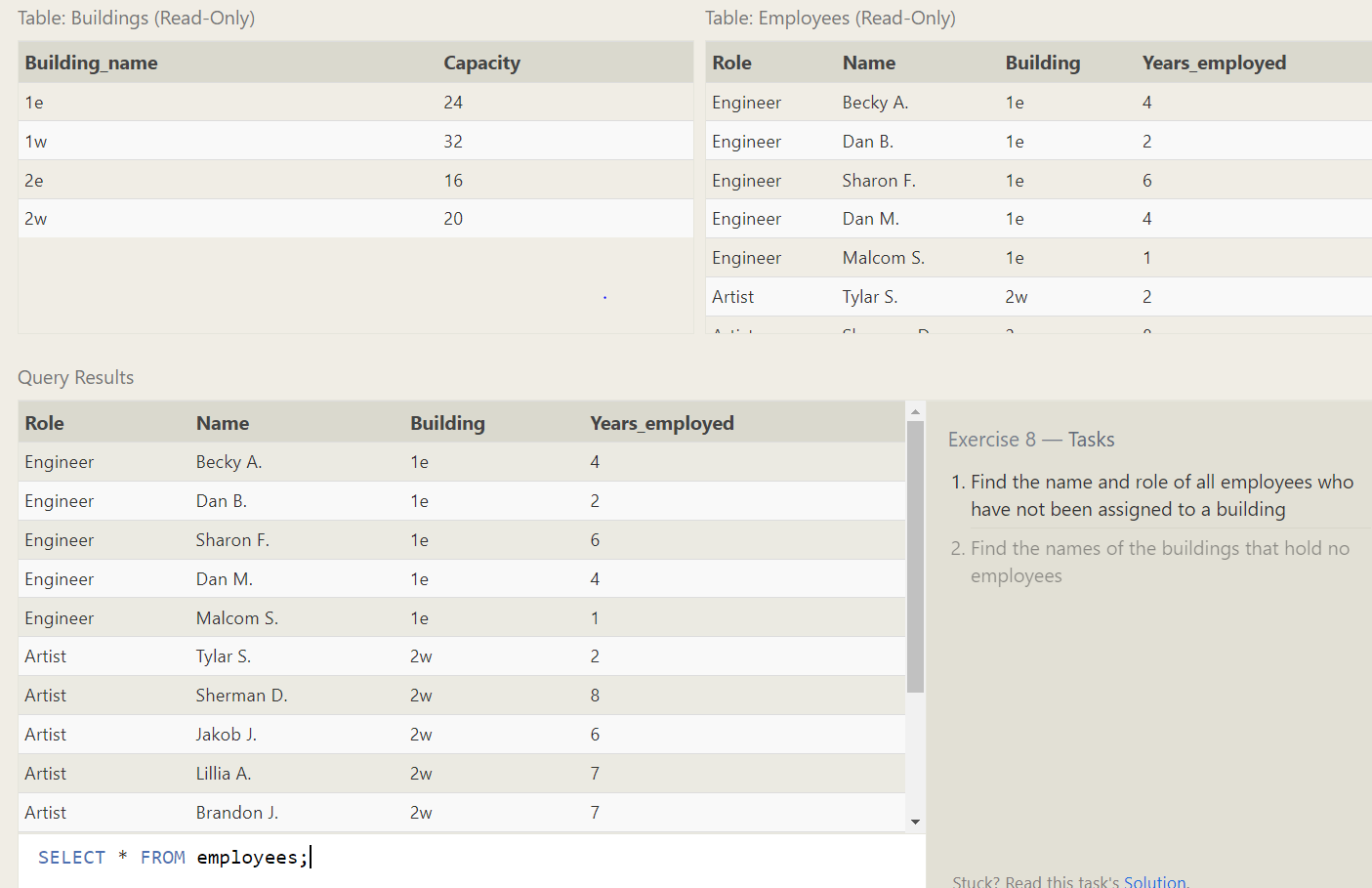
1. SELECT DISTINCT building FROM employees;
2. SELECT Building\_name,capacity FROM Buildings;
3. SELECT DISTINCT building\_name, role

FROM buildings

LEFT JOIN employees

ON building\_name = building;

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



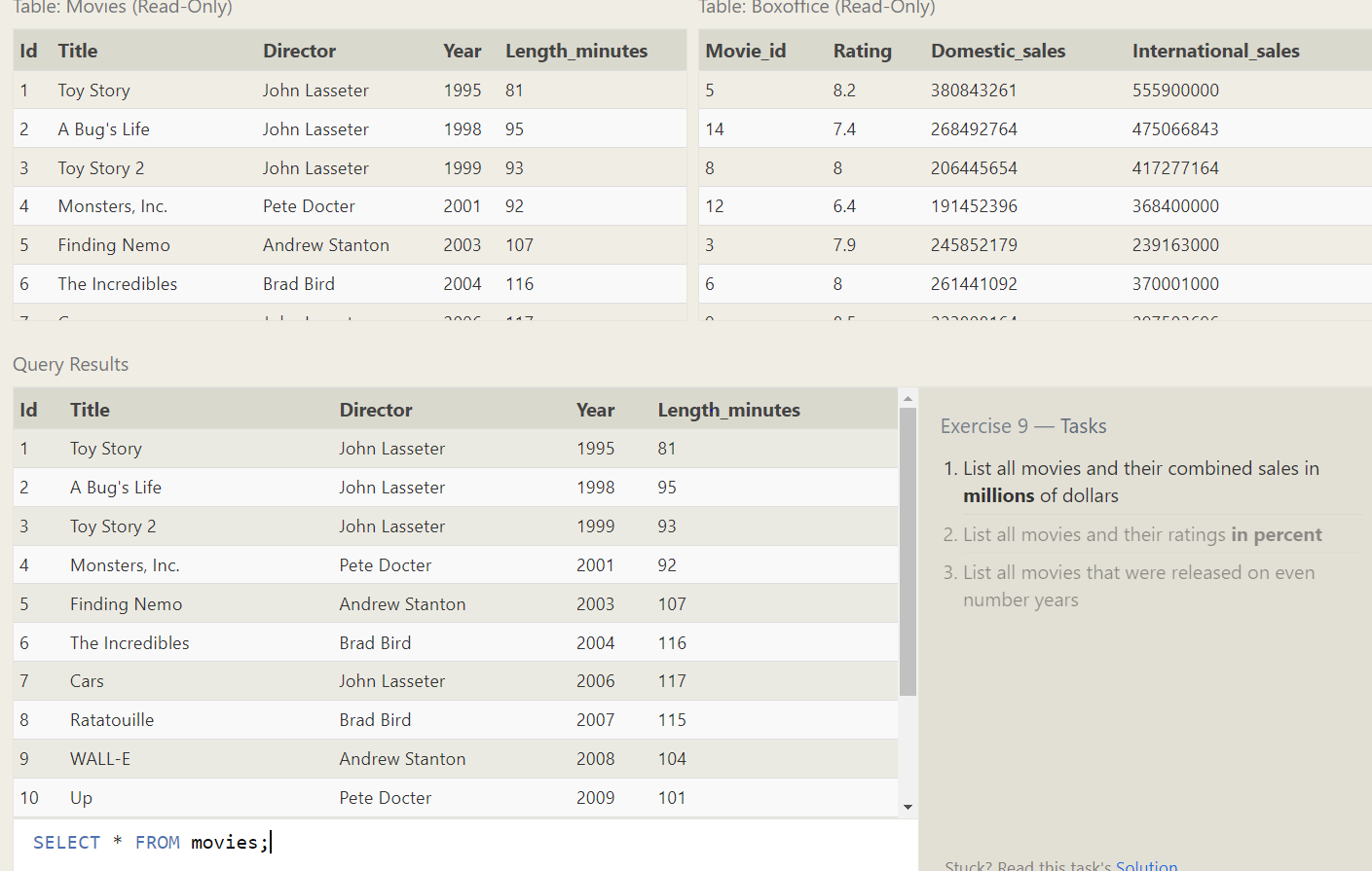
1. SELECT \* FROM employees where building is null;
2. SELECT DISTINCT building\_name

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 id = movie\_id;

2. SELECT title, rating \* 10 AS rating\_percent

FROM movies

JOIN boxoffice

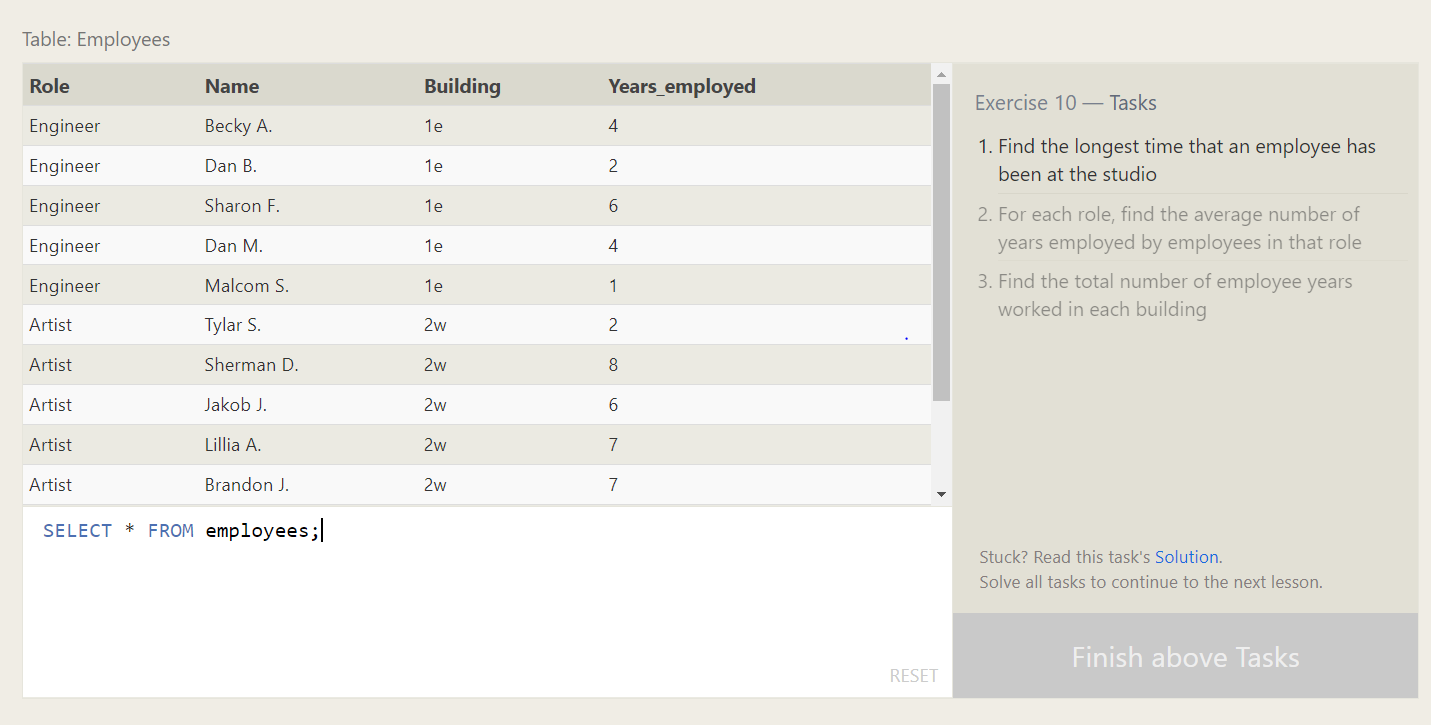
ON movies.id = boxoffice.movie\_id;

3. SELECT title, year

FROM movies

WHERE year % 2 = 0;

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



1. SELECT MAX(years\_employed) as Max\_years\_employed

FROM employees;

2. SELECT role, AVG(years\_employed) as Average\_years\_employed

FROM employees

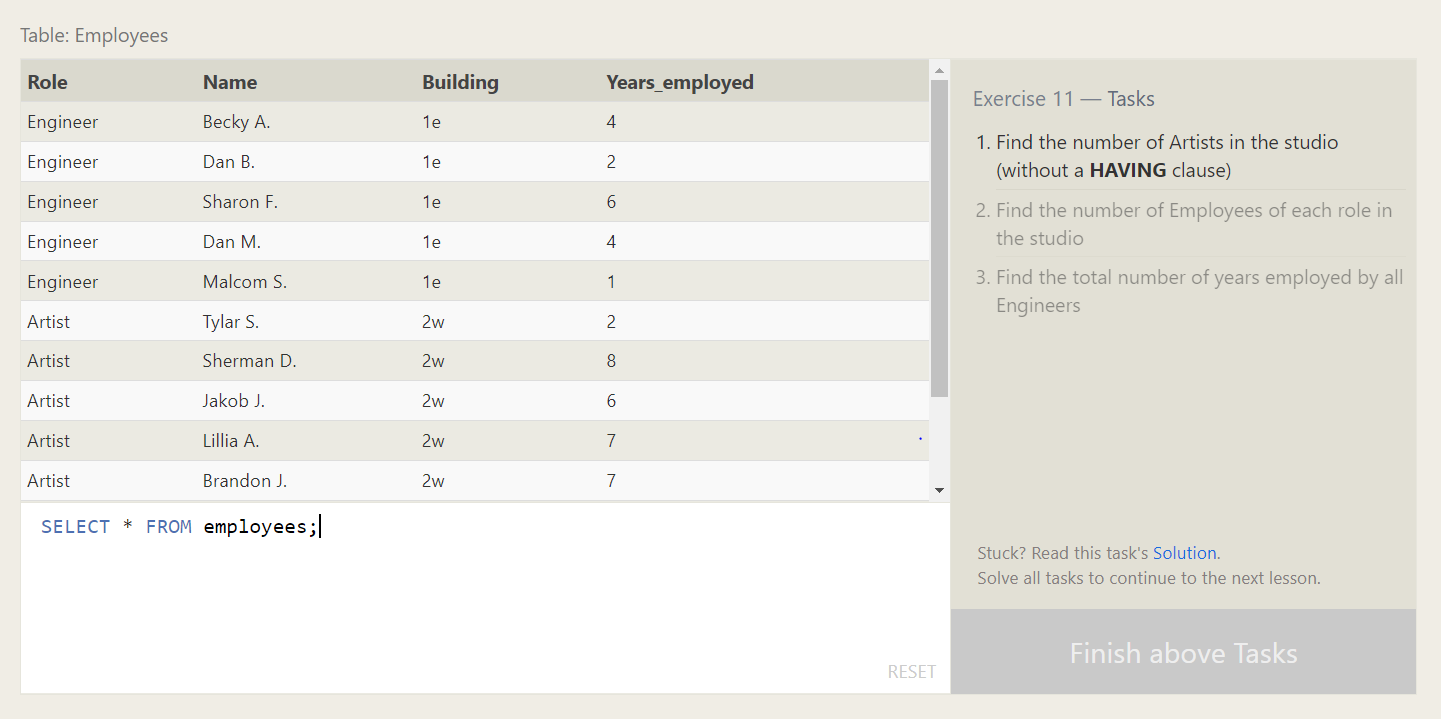
GROUP BY role;

3. SELECT building, SUM(years\_employed) as Total\_years\_employed

FROM employees

GROUP BY building;

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



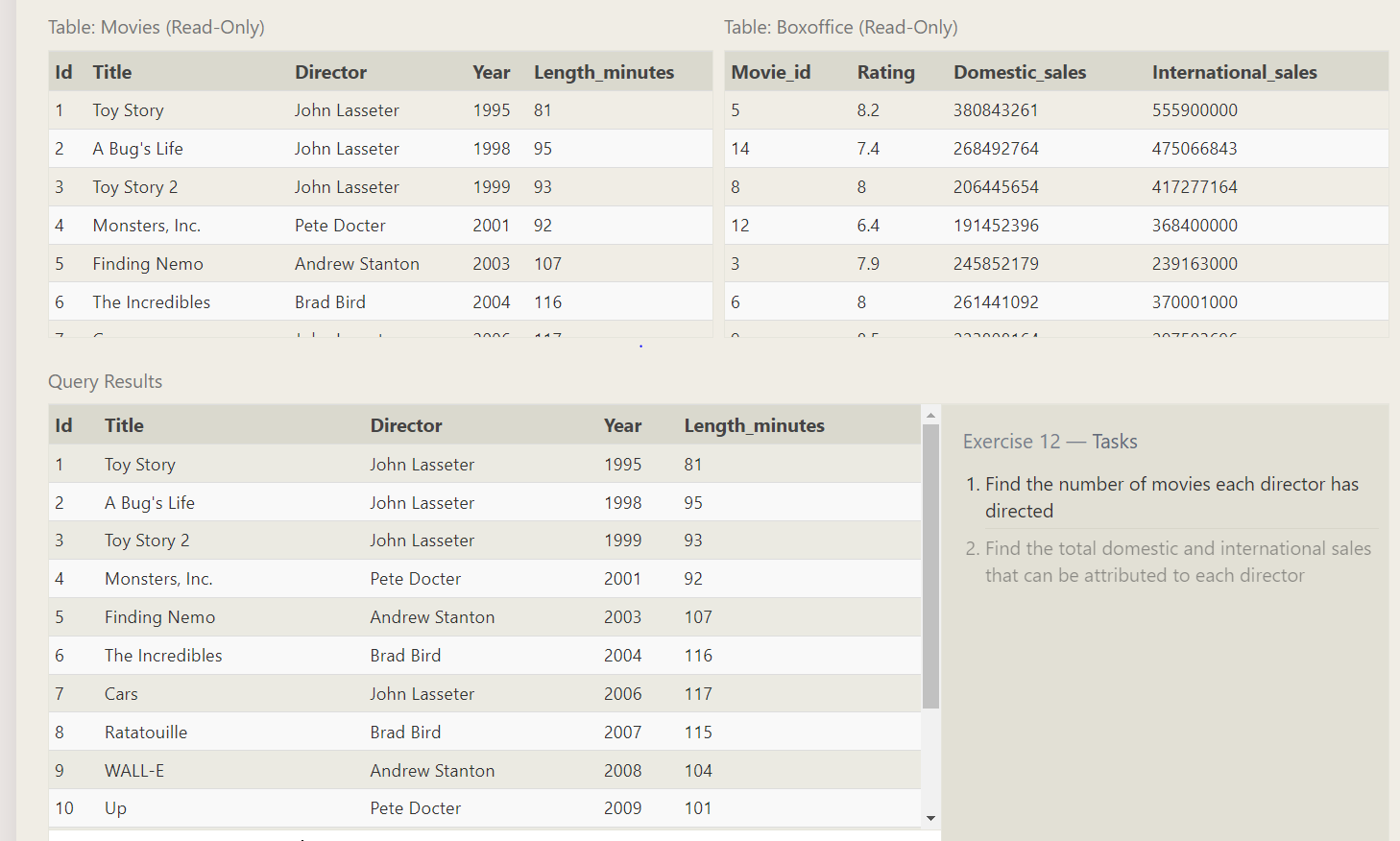
1. SELECT Role,count(\*) as sum FROM employees WHERE role = "Artist";;
2. SELECT Role,count(\*) as sum FROM employees WHERE role = "Artist";;
3. SELECT role, SUM(years\_employed)

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

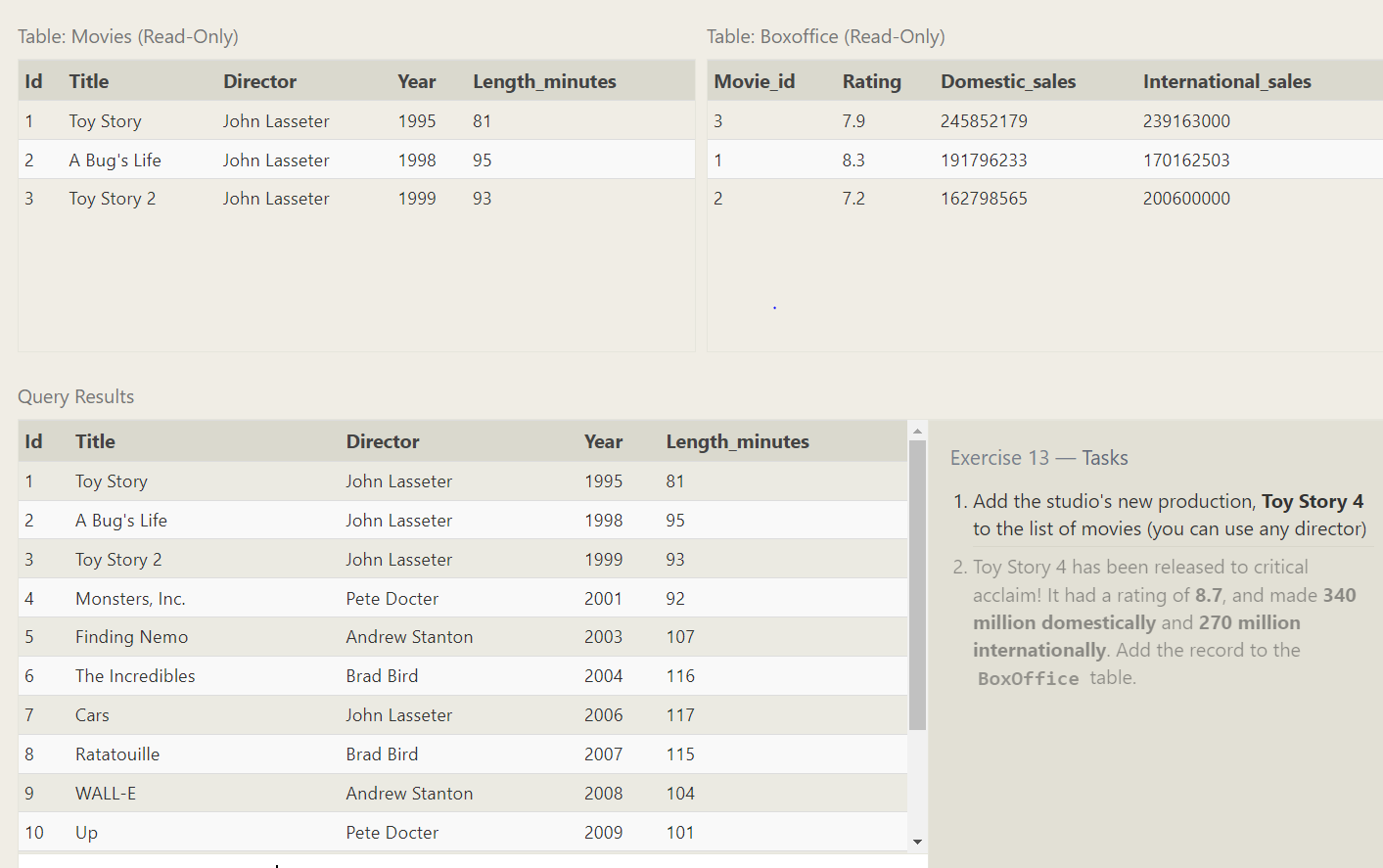
FROM movies

INNER JOIN boxoffice

ON movies.id = boxoffice.movie\_id

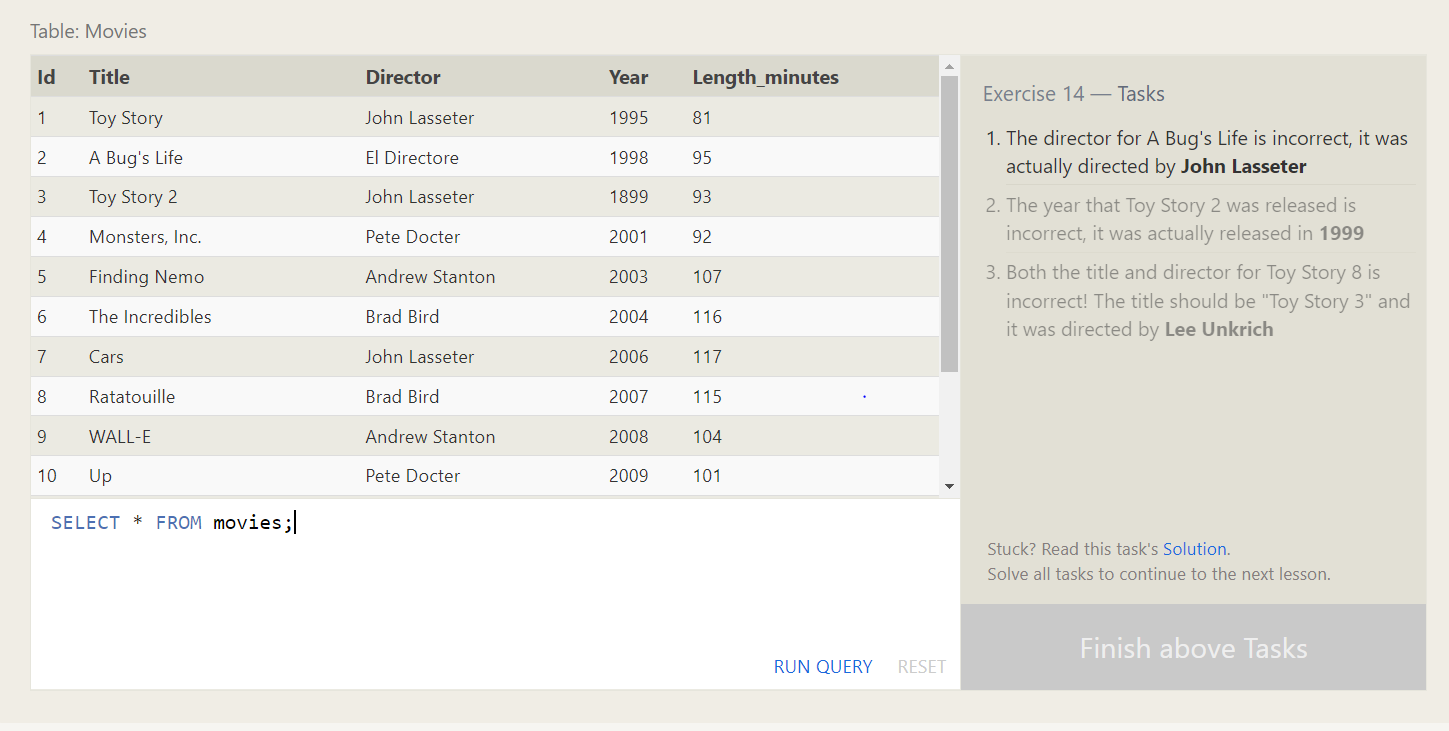
GROUP BY director;

**SQL Lesson 13: Inserting rows**



1. INSERT INTO movies VALUES (4, "Toy Story 4", "El Directore", 2015, 90);
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

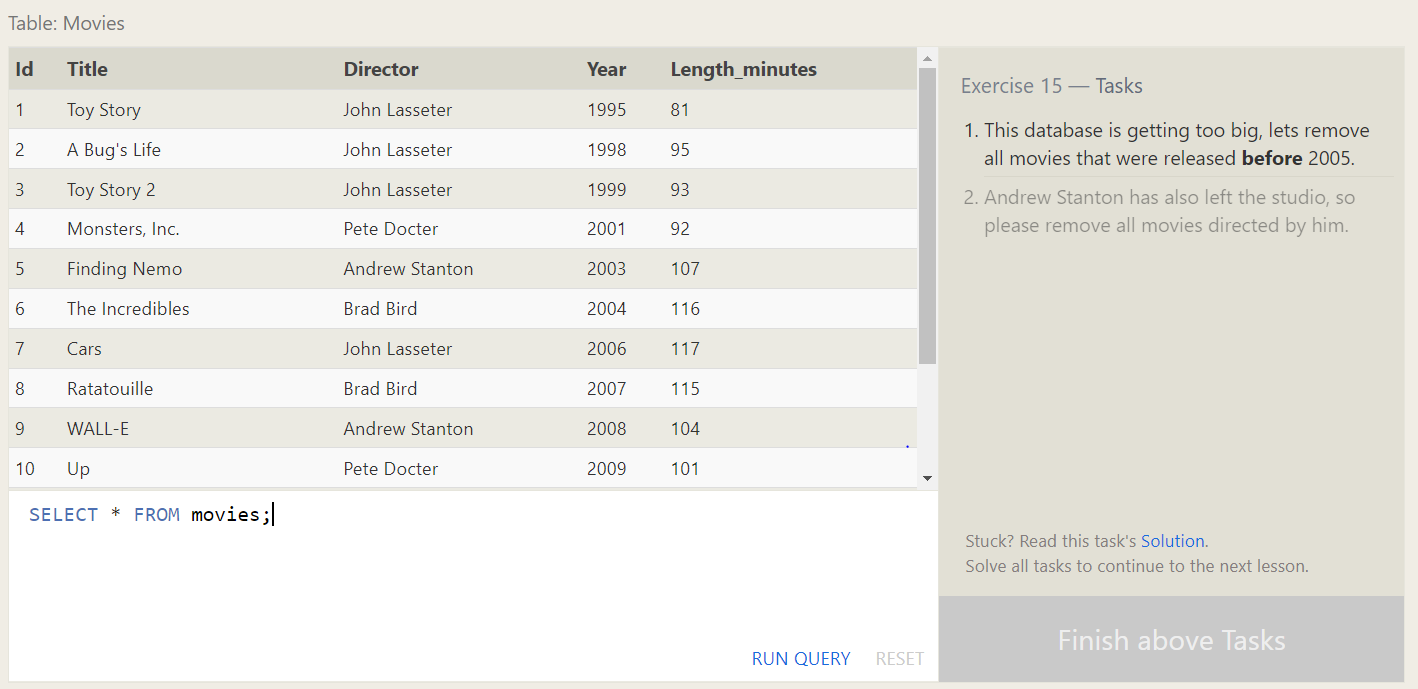
where title like 'Toy Story 2';

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**

create Table Database(

Name varchar(250),

version integer,

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;**

