1) SQL Lesson 1 SELECT queries 101:

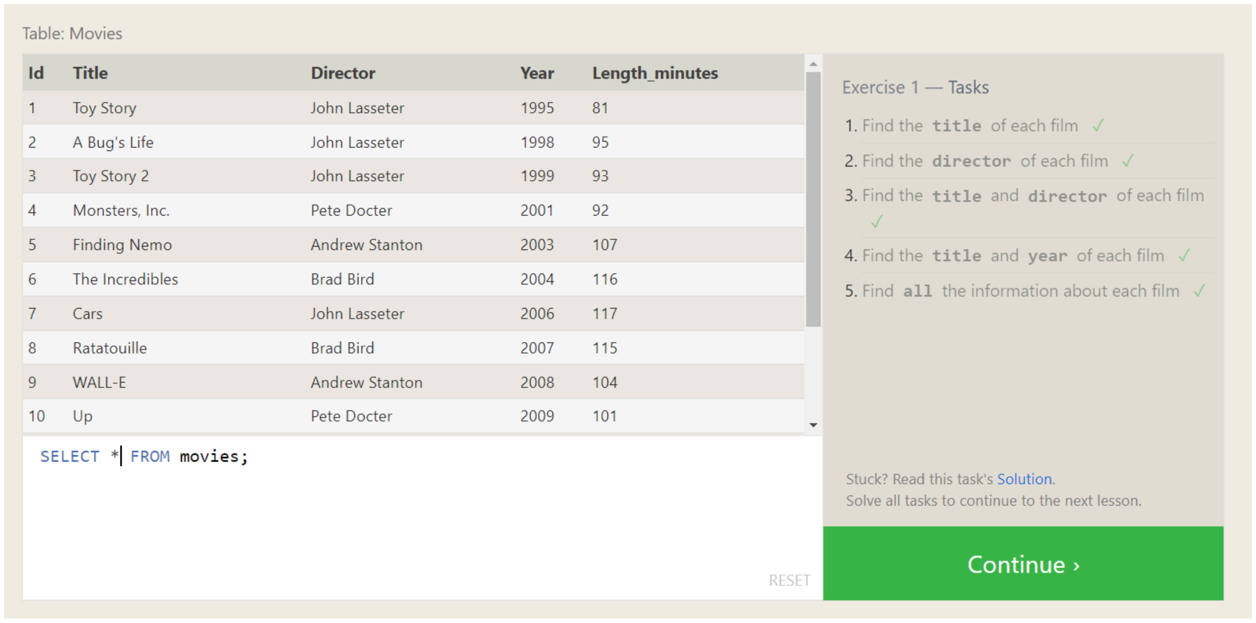


Table: Movies

Id	Title	Director	Year	Length_minutes
1	Toy Story	John Lasseter	1995	81
2	A Bug's Life	John Lasseter	1998	95
3	Toy Story 2	John Lasseter	1999	93
4	Monsters, Inc.	Pete Docter	2001	92
5	Finding Nemo	Andrew Stanton	2003	107
6	The Incredibles	Brad Bird	2004	116
7	Cars	John Lasseter	2006	117
8	Ratatouille	Brad Bird	2007	115
9	WALL-E	Andrew Stanton	2008	104
10	Up	Pete Docter	2009	101

```
SELECT * FROM movies;
```

Exercise 1 — Tasks

1. Find the `title` of each film ✓
2. Find the `director` of each film ✓
3. Find the `title` and `director` of each film ✓
4. Find the `title` and `year` of each film ✓
5. Find `all` the information about each film ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Answer:

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;

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

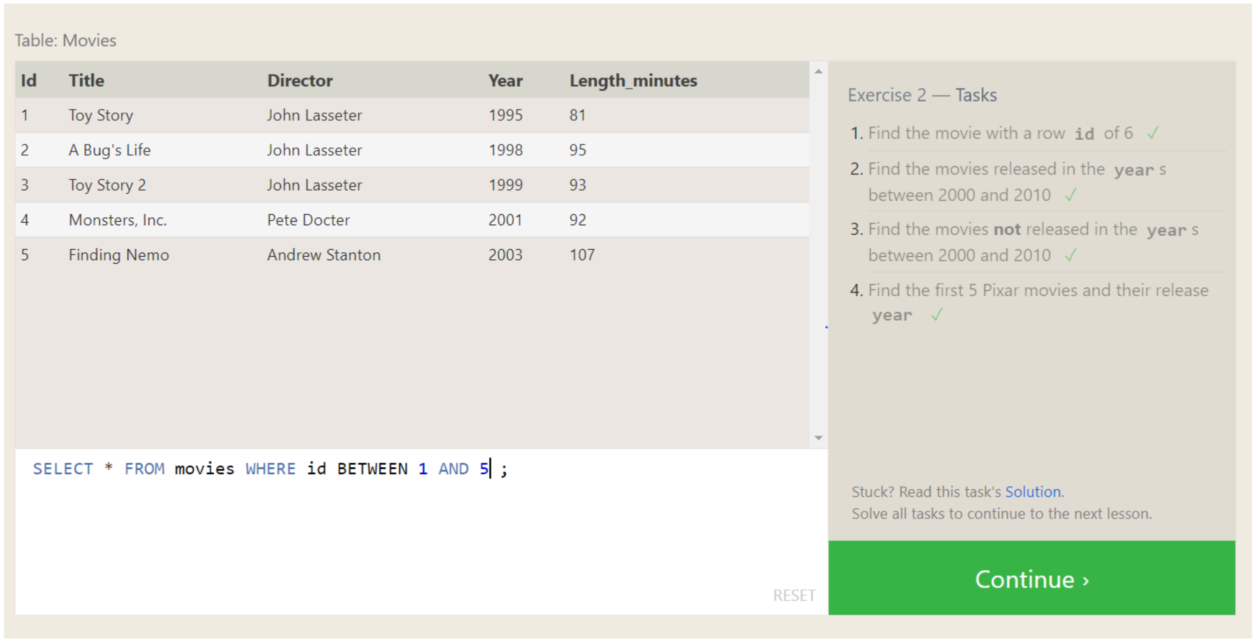


Table: Movies

Id	Title	Director	Year	Length_minutes
1	Toy Story	John Lasseter	1995	81
2	A Bug's Life	John Lasseter	1998	95
3	Toy Story 2	John Lasseter	1999	93
4	Monsters, Inc.	Pete Docter	2001	92
5	Finding Nemo	Andrew Stanton	2003	107

```
SELECT * FROM movies WHERE id BETWEEN 1 AND 5 ;
```

Exercise 2 — Tasks

1. Find the movie with a row `id` of 6 ✓
2. Find the movies released in the `year`s between 2000 and 2010 ✓
3. Find the movies **not** released in the `year`s between 2000 and 2010 ✓
4. Find the first 5 Pixar movies and their release `year` ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Answer:

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 * FROM movies WHERE id BETWEEN 1 AND 5;

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

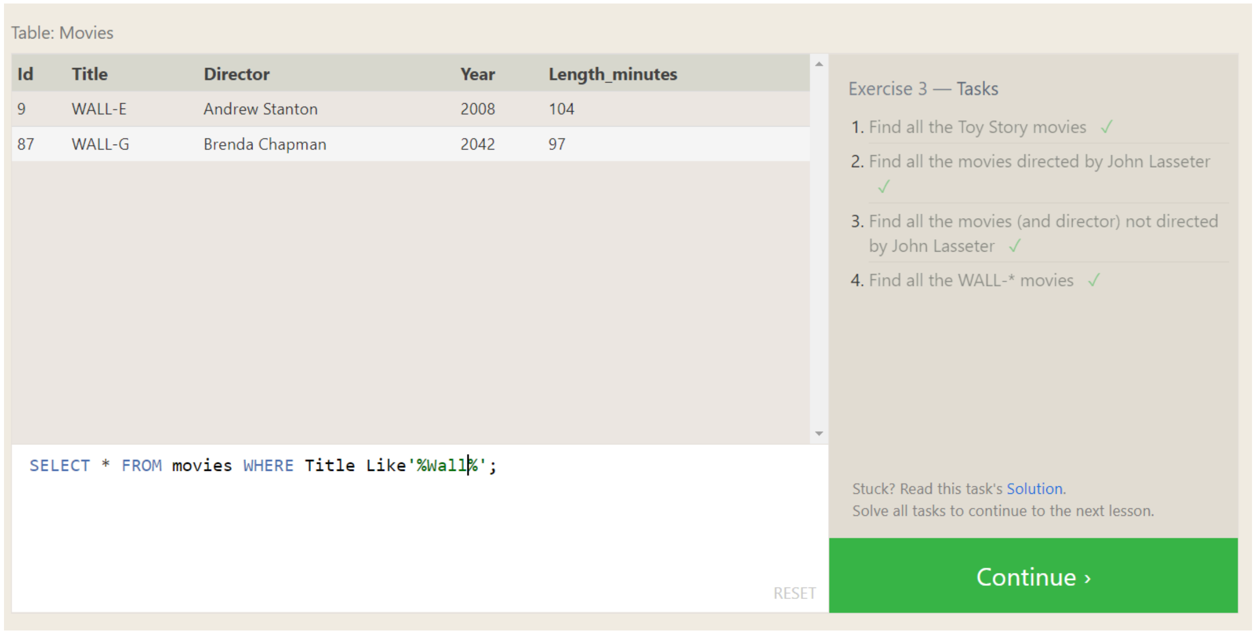


Table: Movies

Id	Title	Director	Year	Length_minutes
9	WALL-E	Andrew Stanton	2008	104
87	WALL-G	Brenda Chapman	2042	97

```
SELECT * FROM movies WHERE Title Like'%Wall%';
```

Exercise 3 — Tasks

1. Find all the Toy Story movies ✓
2. Find all the movies directed by John Lasseter ✓
3. Find all the movies (and director) not directed by John Lasseter ✓
4. Find all the WALL-* movies ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

<u>Answer:</u>

1. SELECT * FROM movies where Title Like '%Toy Story%';

2. SELECT * FROM movies where Director Like '%john Lasseter%';

3. SELECT * FROM movies where Director NOT Like '%john Lasseter%';

4. SELECT * FROM movies where Title Like '%Wall%';

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

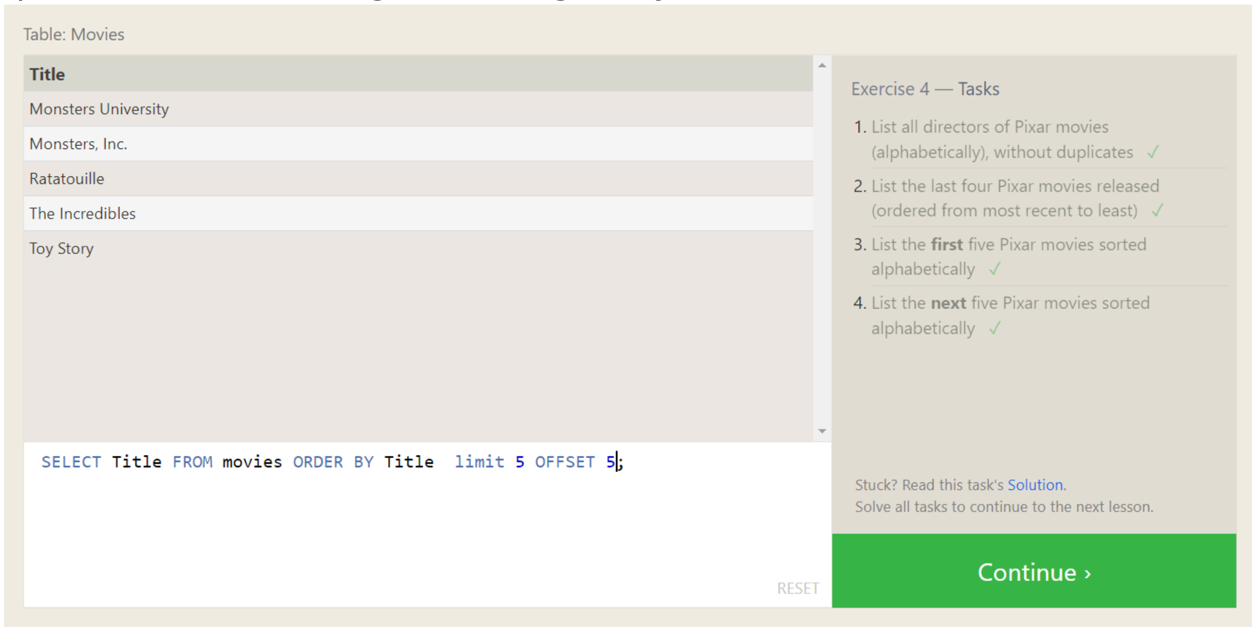


Table: Movies

Title
Monsters University
Monsters, Inc.
Ratatouille
The Incredibles
Toy Story

```
SELECT Title FROM movies ORDER BY Title  limit 5 OFFSET 5;
```

Exercise 4 — Tasks

1. List all directors of Pixar movies (alphabetically), without duplicates ✓
2. List the last four Pixar movies released (ordered from most recent to least) ✓
3. List the **first** five Pixar movies sorted alphabetically ✓
4. List the **next** five Pixar movies sorted alphabetically ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Answer:

1. SELECT distinct Director FROM movies ORDER BY Director ASC;

2. SELECT Title, Year FROM movies ORDER BY year DESC LIMIT 4;

3. SELECT Title FROM Movies ORDER BY Title ASC LIMIT 5;

4. SELECT Title FROM Movies ORDER BY Title ASC LIMIT 5 OFFSET 5;

5)SQL Review: Simple SELECT Queries:

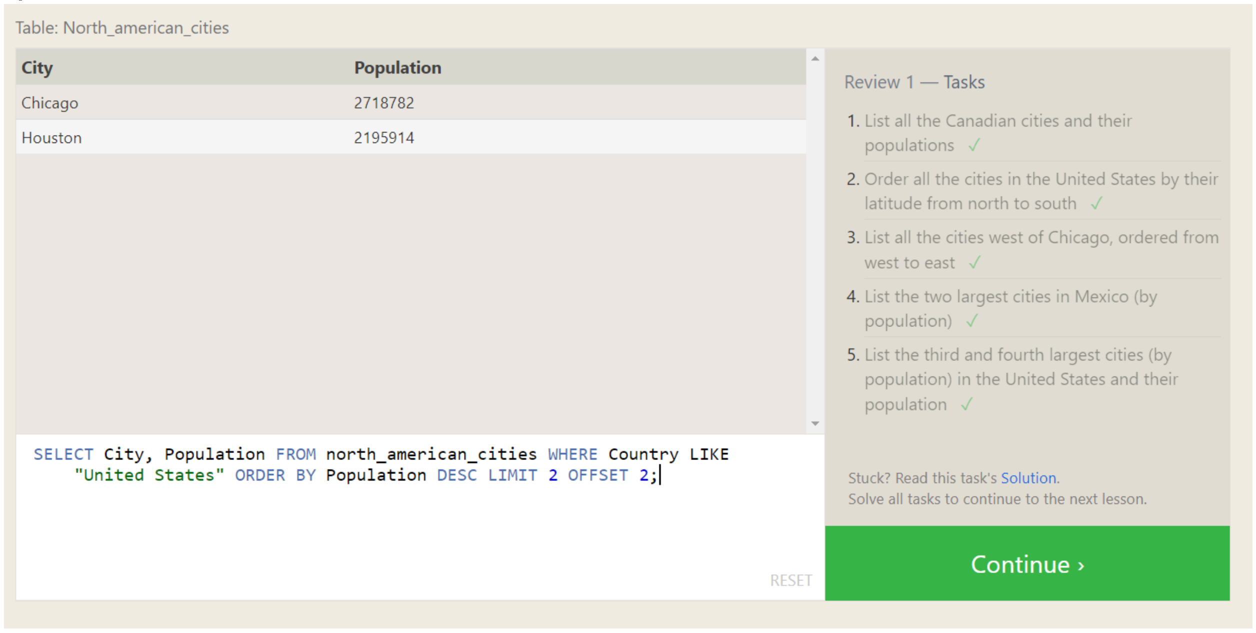


Table: North_american_cities

City	Population
Chicago	2718782
Houston	2195914

```
SELECT City, Population FROM north_american_cities WHERE Country LIKE
    "United States" ORDER BY Population DESC LIMIT 2 OFFSET 2;
```

Review 1 — Tasks

1. List all the Canadian cities and their populations ✓
2. Order all the cities in the United States by their latitude from north to south ✓
3. List all the cities west of Chicago, ordered from west to east ✓
4. List the two largest cities in Mexico (by population) ✓
5. List the third and fourth largest cities (by population) in the United States and their population ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Answer:

1. SELECT City, Population FROM north_american_cities WHERE Country = "Canada";

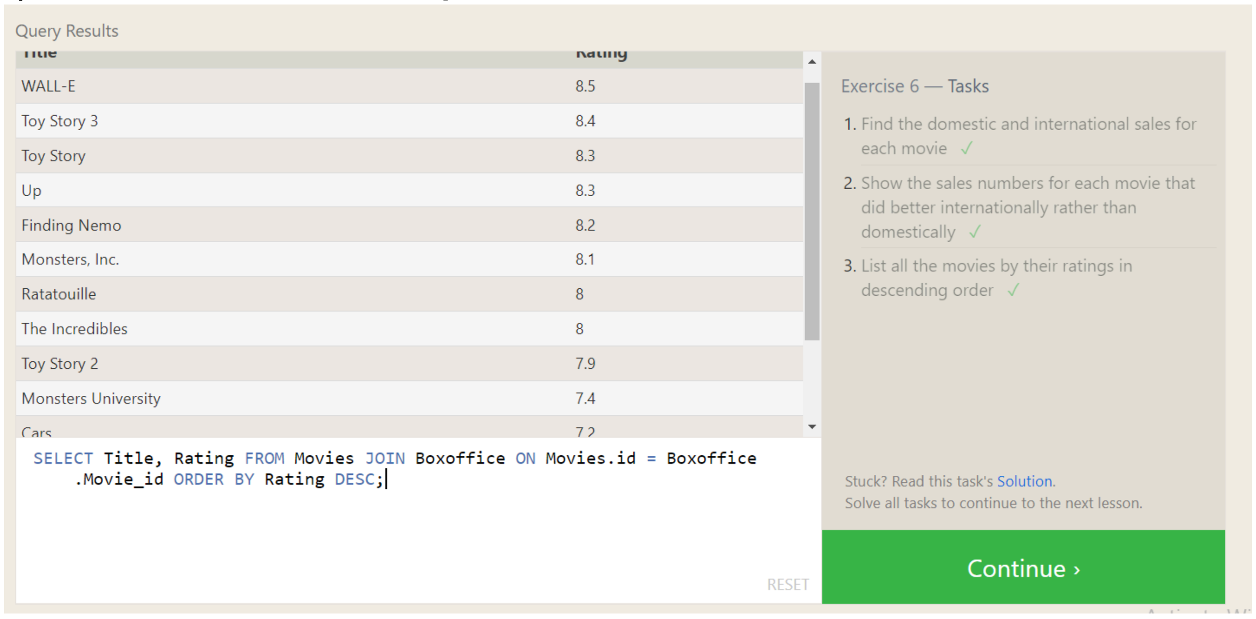
2. SELECT City, latitude FROM north_american_cities WHERE Country = "United States" ORDER BY latitude DESC;

3. SELECT City, longitude FROM north_american_cities WHERE longitude < -87.629798 ORDER BY longitude ASC;

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;

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


Query Results

Title	Rating
WALL-E	8.5
Toy Story 3	8.4
Toy Story	8.3
Up	8.3
Finding Nemo	8.2
Monsters, Inc.	8.1
Ratatouille	8
The Incredibles	8
Toy Story 2	7.9
Monsters University	7.4
Cars	7.2

```
SELECT Title, Rating FROM Movies JOIN Boxoffice ON Movies.id = Boxoffice
    .Movie_id ORDER BY Rating DESC;
```

Exercise 6 — Tasks

1. Find the domestic and international sales for each movie ✓
2. Show the sales numbers for each movie that did better internationally rather than domestically ✓
3. List all the movies by their ratings in descending order ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

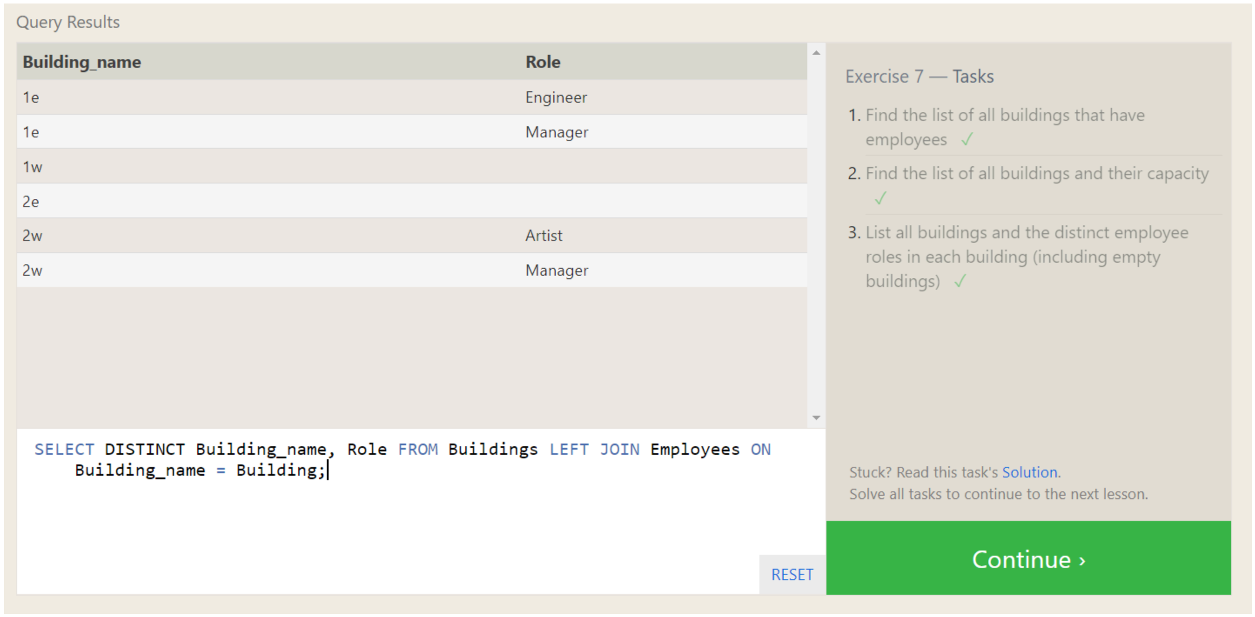
Answer:

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, Rating FROM Movies JOIN Boxoffice ON Movies.id = Boxoffice.Movie_id ORDER BY Rating DESC;

7)SQL Lesson 7: OUTER JOINs:


Query Results

Building_name	Role
1e	Engineer
1e	Manager
1w	
2e	
2w	Artist
2w	Manager

```
SELECT DISTINCT Building_name, Role FROM Buildings LEFT JOIN Employees ON
    Building_name = Building;
```

Exercise 7 — Tasks

1. Find the list of all buildings that have employees ✓
2. Find the list of all buildings and their capacity ✓
3. List all buildings and the distinct employee roles in each building (including empty buildings) ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

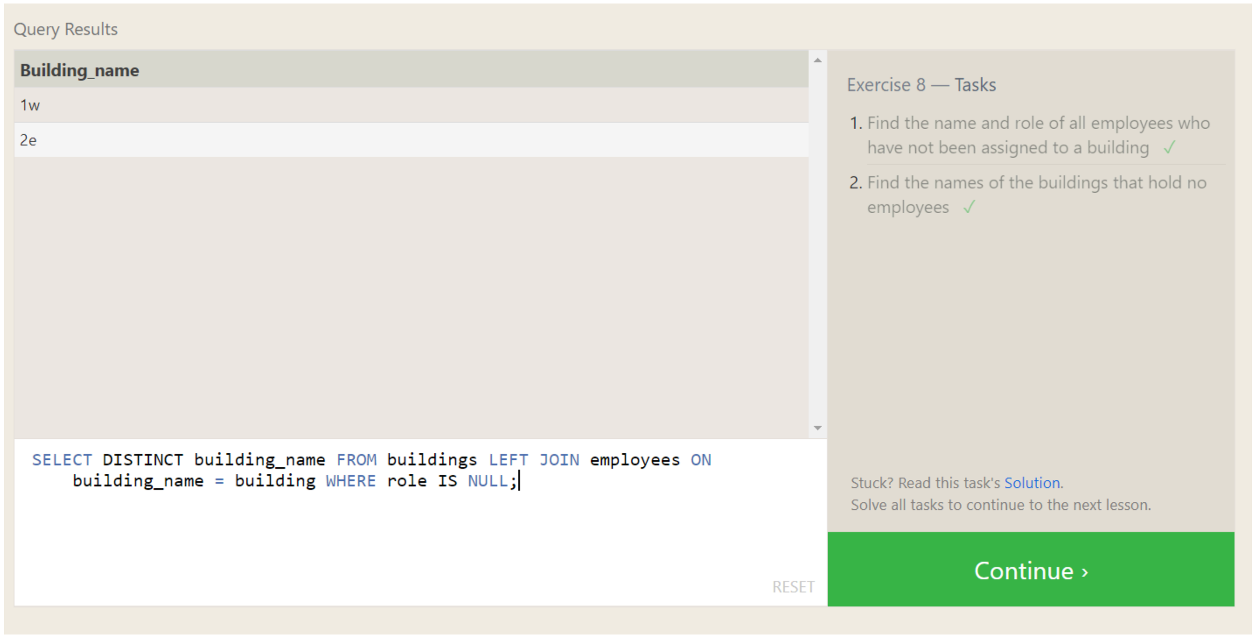
<u>Answer:</u>

1. SELECT DISTINCT Building FROM Employees;

2. SELECT * FROM Buildings;

3. SELECT DISTINCT Building_name, Role FROM Buildings LEFT JOIN Employees ON Building_name = Building;

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


Query Results

Building_name
1w
2e

```
SELECT DISTINCT building_name FROM buildings LEFT JOIN employees ON
    building_name = building WHERE role IS NULL;
```

Exercise 8 — Tasks

1. Find the name and role of all employees who have not been assigned to a building ✓
2. Find the names of the buildings that hold no employees ✓

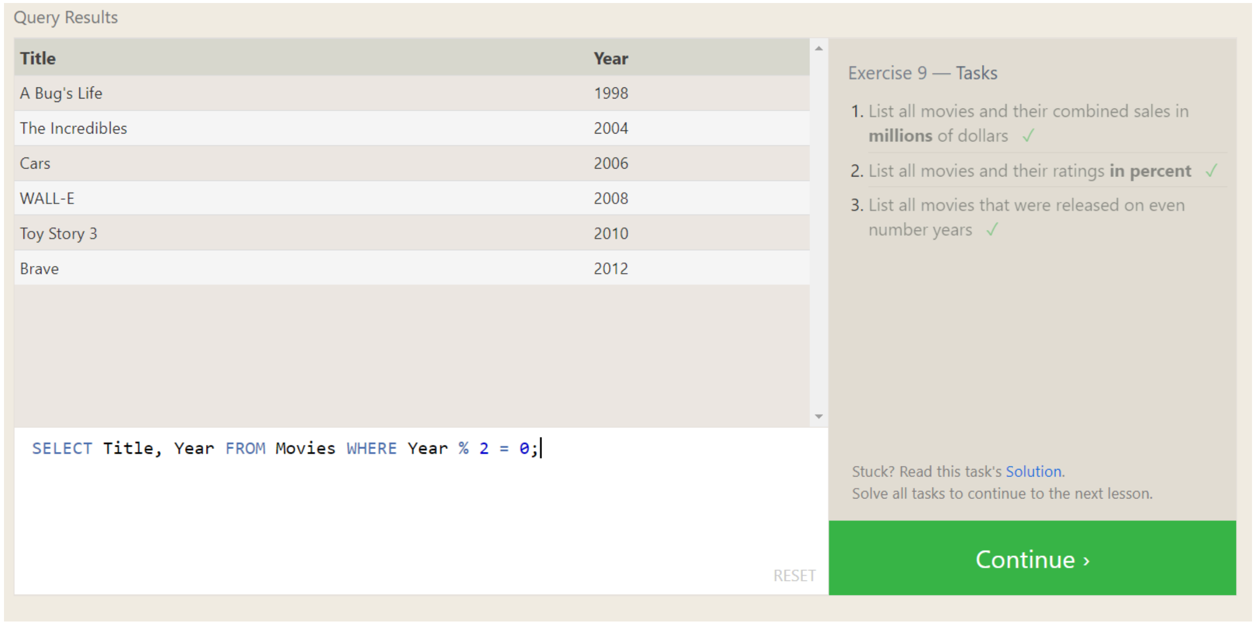
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Answer:

1. SELECT Name, Role FROM Employees WHERE Building IS NULL;

2. SELECT DISTINCT building_name FROM buildings LEFT JOIN employees ON building_name = building WHERE role IS NULL;

9)SQL Lesson 9: Queries with expressions:



Query Results

Title	Year
A Bug's Life	1998
The Incredibles	2004
Cars	2006
WALL-E	2008
Toy Story 3	2010
Brave	2012

```
SELECT Title, Year FROM Movies WHERE Year % 2 = 0;
```

Exercise 9 — Tasks

1. List all movies and their combined sales in **millions** of dollars ✓
2. List all movies and their ratings **in percent** ✓
3. List all movies that were released on even number years ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Answer:

1. SELECT Title, (domestic_sales + international_sales) / 1000000 AS Gross_sales_millions FROM Movies JOIN Boxoffice ON movies.id = Boxoffice.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;

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

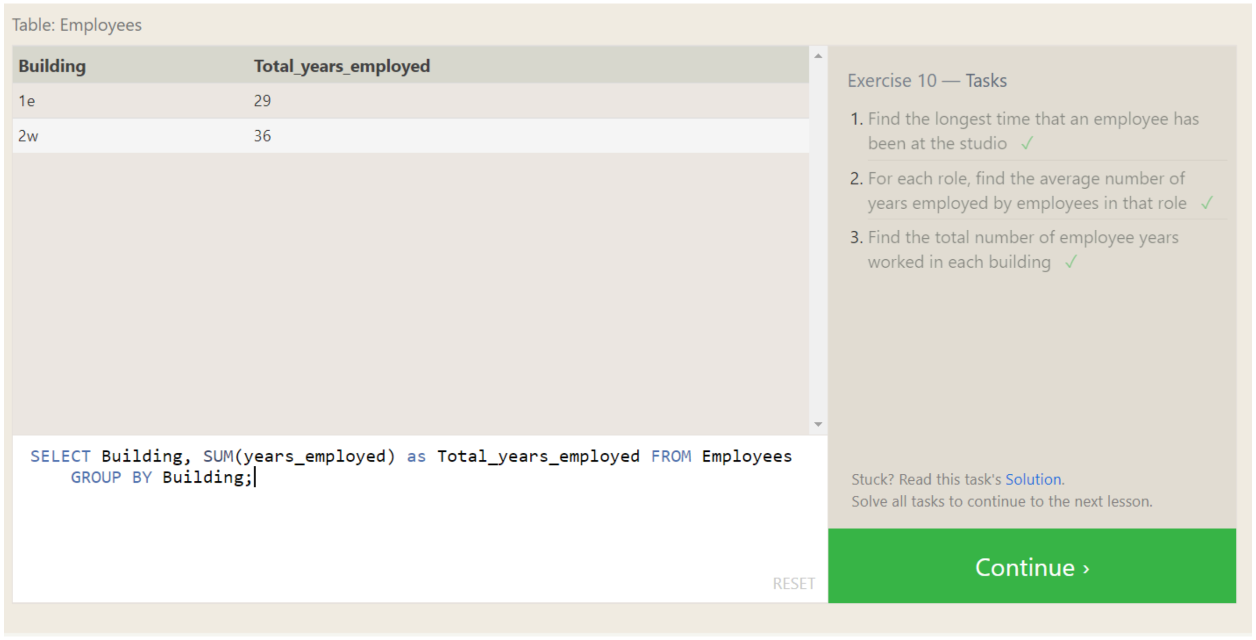


Table: Employees

Building	Total_years_employed
1e	29
2w	36

```
SELECT Building, SUM(years_employed) as Total_years_employed FROM Employees
    GROUP BY Building;
```

Exercise 10 — Tasks

1. Find the longest time that an employee has been at the studio ✓
2. For each role, find the average number of years employed by employees in that role ✓
3. Find the total number of employee years worked in each building ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

<u>Answer:</u>

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;

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

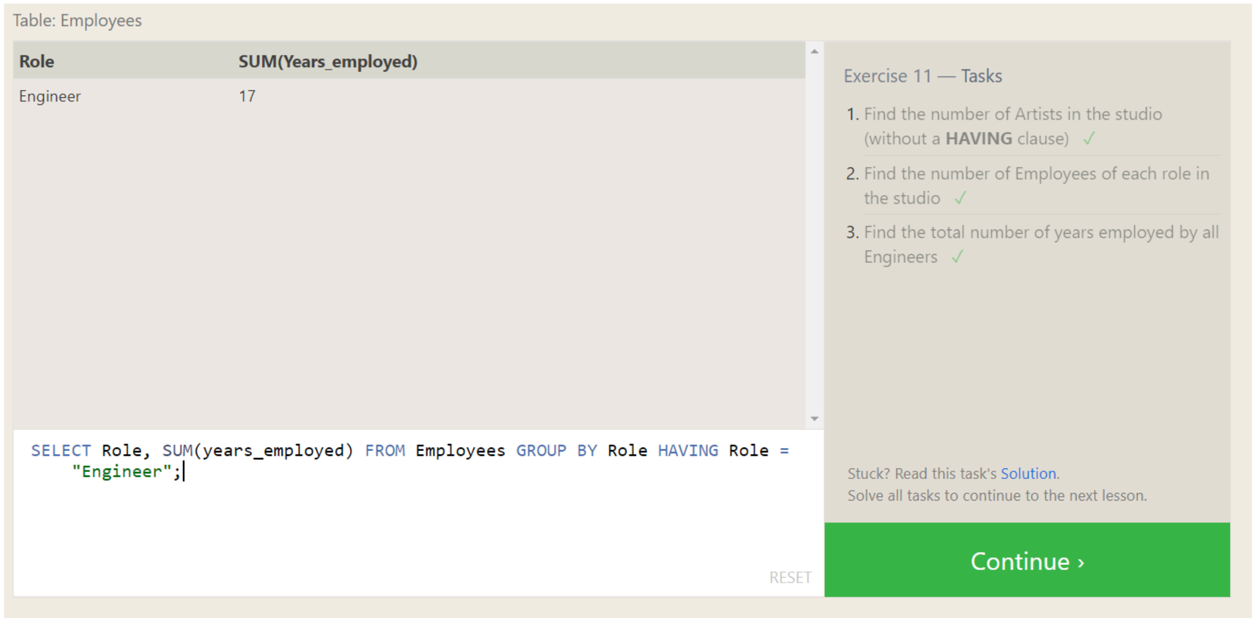


Table: Employees

Role	SUM(Years_employed)
Engineer	17

```
SELECT Role, SUM(years_employed) FROM Employees GROUP BY Role HAVING Role =
    "Engineer";
```

RESET

Exercise 11 — Tasks

1. Find the number of Artists in the studio (without a **HAVING** clause) ✓
2. Find the number of Employees of each role in the studio ✓
3. Find the total number of years employed by all Engineers ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

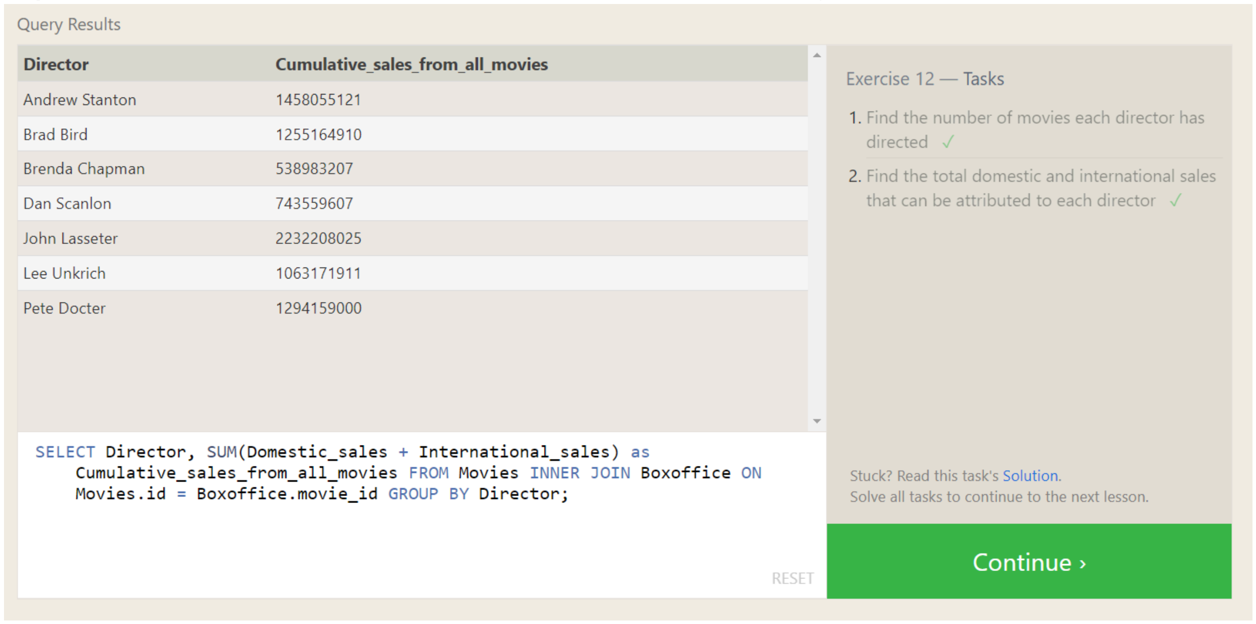
Answer:

1. SELECT Role, COUNT(*) as Number_of_artists FROM Employees WHERE Role = "Artist";

2. SELECT Role, COUNT(*)FROM Employees GROUP BY Role;

3. SELECT Role, SUM(years_employed) FROM Employees GROUP BY Role HAVING Role = "Engineer";

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



Query Results

Director	Cumulative_sales_from_all_movies
Andrew Stanton	1458055121
Brad Bird	1255164910
Brenda Chapman	538983207
Dan Scanlon	743559607
John Lasseter	2232208025
Lee Unkrich	1063171911
Pete Docter	1294159000

```
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;
```

RESET

Exercise 12 — Tasks

1. Find the number of movies each director has directed ✓
2. Find the total domestic and international sales that can be attributed to each director ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

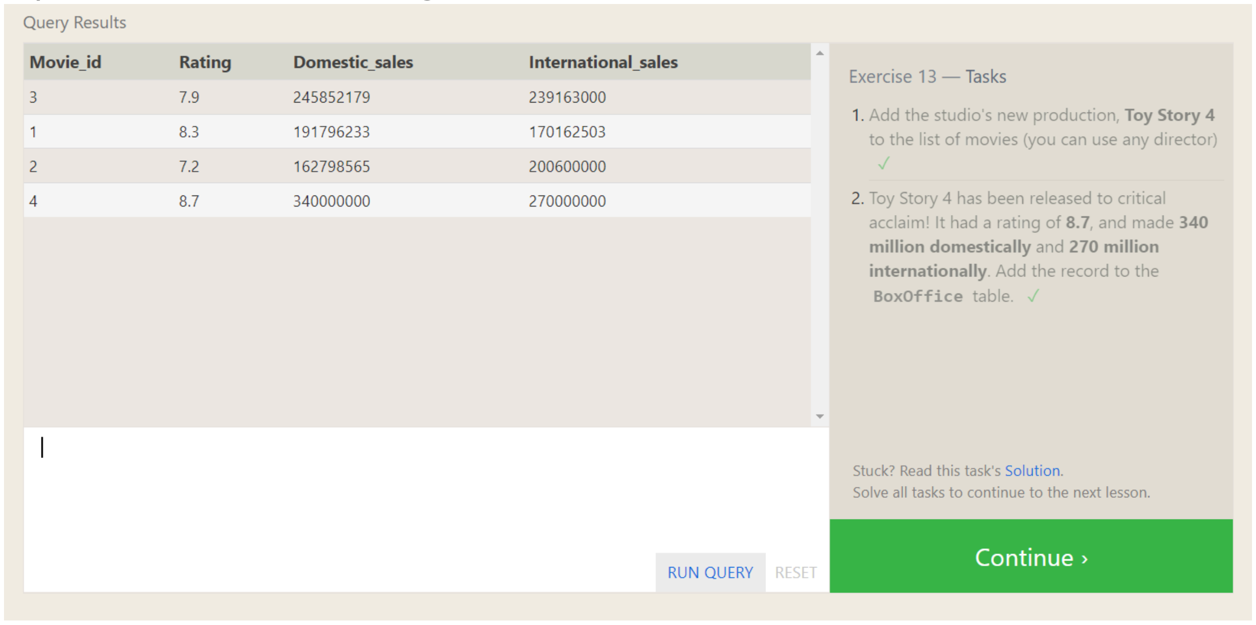
Continue ›

Answer:

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;

13) SQL Lesson 13: Inserting rows:



Query Results

Movie_id	Rating	Domestic_sales	International_sales
3	7.9	245852179	239163000
1	8.3	191796233	170162503
2	7.2	162798565	200600000
4	8.7	340000000	270000000

Exercise 13 — Tasks

1. Add the studio's new production, **Toy Story 4** to the list of movies (you can use any director) ✓
2. 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. ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

<u>Answer:</u>

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

2. INSERT INTO Boxoffice VALUES (4, 8.7, 340000000, 270000000);

14) SQL Lesson 14: Updating rows:

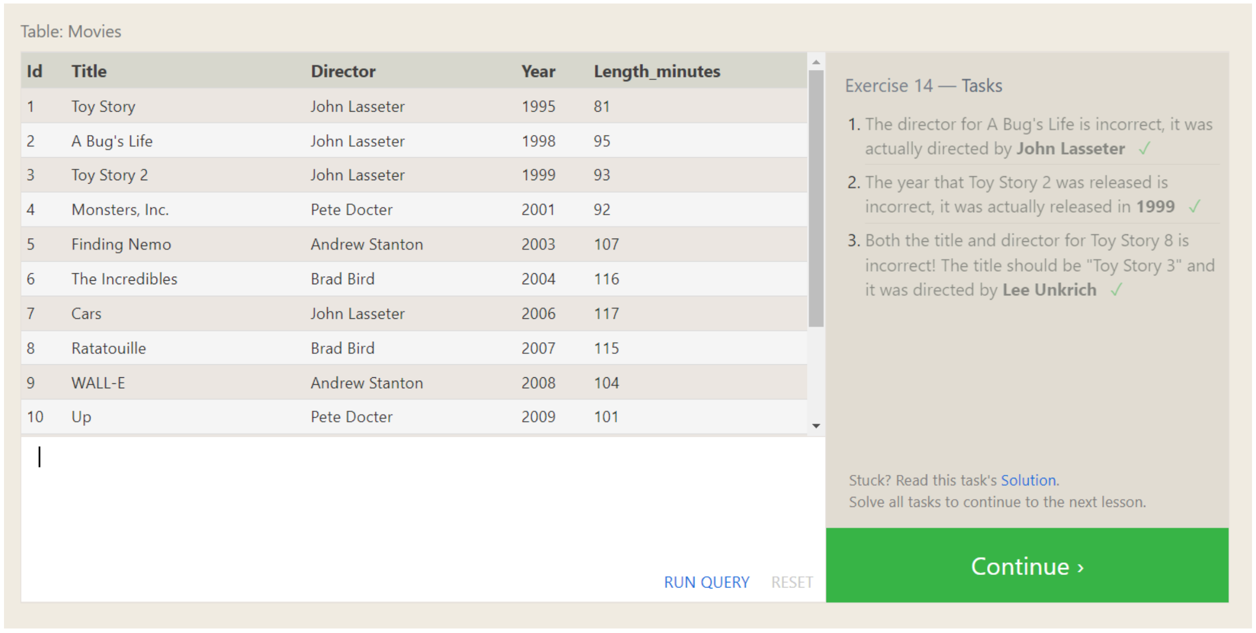


Table: Movies

Id	Title	Director	Year	Length_minutes
1	Toy Story	John Lasseter	1995	81
2	A Bug's Life	John Lasseter	1998	95
3	Toy Story 2	John Lasseter	1999	93
4	Monsters, Inc.	Pete Docter	2001	92
5	Finding Nemo	Andrew Stanton	2003	107
6	The Incredibles	Brad Bird	2004	116
7	Cars	John Lasseter	2006	117
8	Ratatouille	Brad Bird	2007	115
9	WALL-E	Andrew Stanton	2008	104
10	Up	Pete Docter	2009	101

Exercise 14 — Tasks

1. The director for A Bug's Life is incorrect, it was actually directed by **John Lasseter** ✓
2. The year that Toy Story 2 was released is incorrect, it was actually released in **1999** ✓
3. 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** ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Answer:

1. UPDATE Movies SET Director = "John Lasseter" WHERE id = 2;

2. UPDATE Movies SET Year = 1999 WHERE Id = 3;

3. UPDATE Movies SET Title = "Toy Story 3", Director = "Lee Unkrich" WHERE id = 11;

15) SQL Lesson 15: Deleting rows:

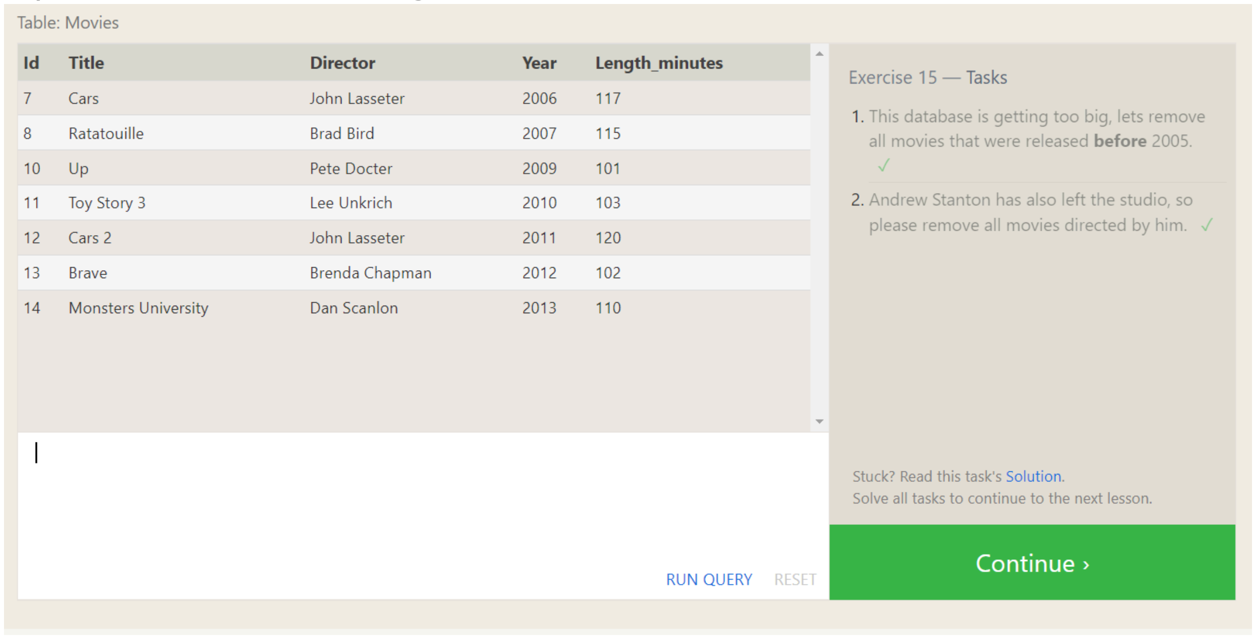


Table: Movies

Id	Title	Director	Year	Length_minutes
7	Cars	John Lasseter	2006	117
8	Ratatouille	Brad Bird	2007	115
10	Up	Pete Docter	2009	101
11	Toy Story 3	Lee Unkrich	2010	103
12	Cars 2	John Lasseter	2011	120
13	Brave	Brenda Chapman	2012	102
14	Monsters University	Dan Scanlon	2013	110

Exercise 15 — Tasks

1. This database is getting too big, lets remove all movies that were released **before** 2005. ✓
2. Andrew Stanton has also left the studio, so please remove all movies directed by him. ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Answer:

1. DELETE FROM Movies where Year < 2005;

2. DELETE FROM Movies where Director = "Andrew Stanton";

16) SQL Lesson 16: Creating tables:

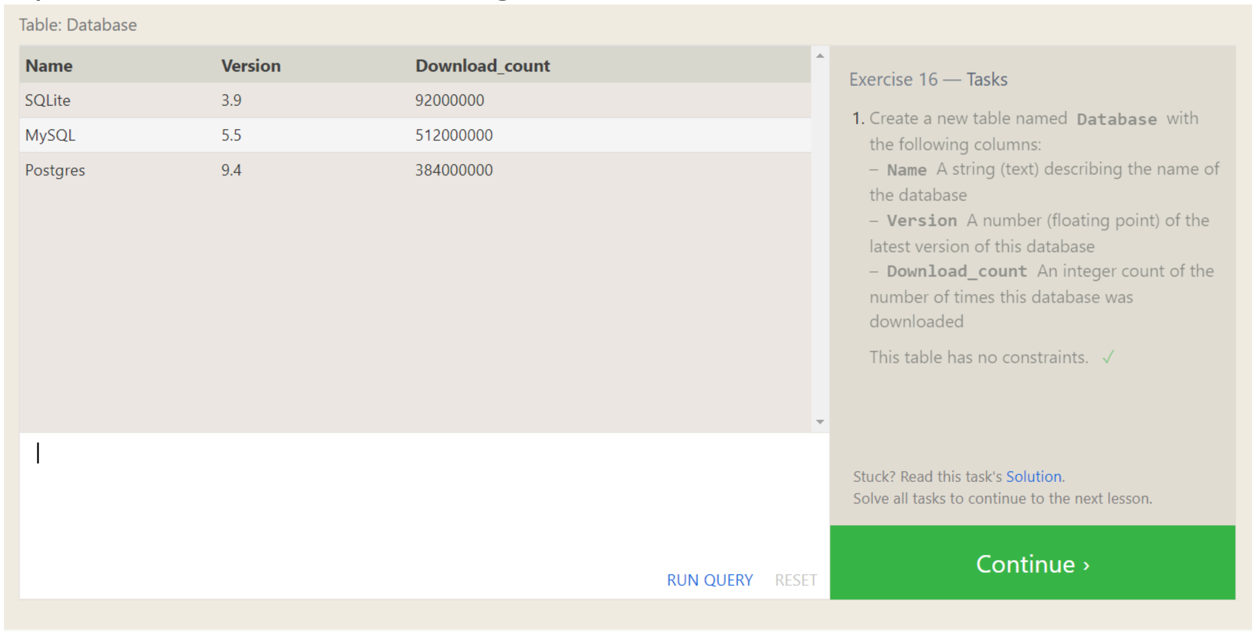


Table: Database

Name	Version	Download_count
SQLite	3.9	92000000
MySQL	5.5	512000000
Postgres	9.4	384000000

Exercise 16 — Tasks

1. Create a new table named `Database` with the following columns:
   – `Name` A string (text) describing the name of the database
   – `Version` A number (floating point) of the latest version of this database
   – `Download_count` An integer count of the number of times this database was downloaded

   This table has no constraints. ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Answer:

1. CREATE TABLE Database (Name TEXT, Version FLOAT,Download_count INTEGER);

17) SQL Lesson 17: Altering tables:

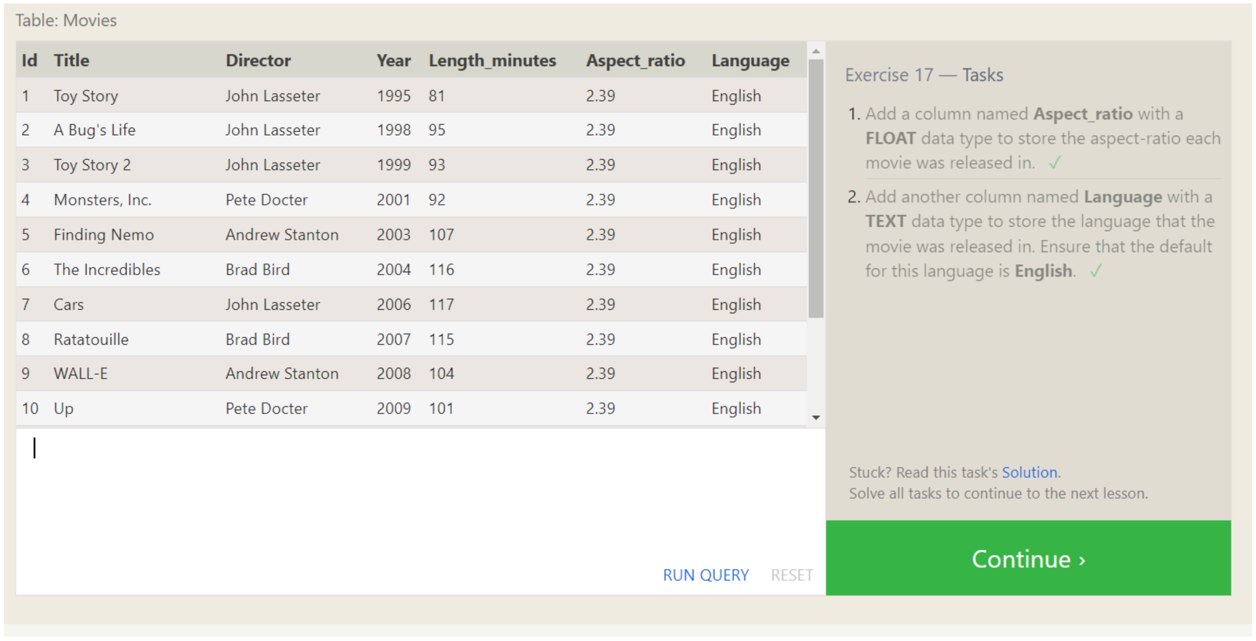


Table: Movies

Id	Title	Director	Year	Length_minutes	Aspect_ratio	Language
1	Toy Story	John Lasseter	1995	81	2.39	English
2	A Bug's Life	John Lasseter	1998	95	2.39	English
3	Toy Story 2	John Lasseter	1999	93	2.39	English
4	Monsters, Inc.	Pete Docter	2001	92	2.39	English
5	Finding Nemo	Andrew Stanton	2003	107	2.39	English
6	The Incredibles	Brad Bird	2004	116	2.39	English
7	Cars	John Lasseter	2006	117	2.39	English
8	Ratatouille	Brad Bird	2007	115	2.39	English
9	WALL-E	Andrew Stanton	2008	104	2.39	English
10	Up	Pete Docter	2009	101	2.39	English

Exercise 17 — Tasks

1. Add a column named **Aspect_ratio** with a **FLOAT** data type to store the aspect-ratio each movie was released in. ✓
2. 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**. ✓

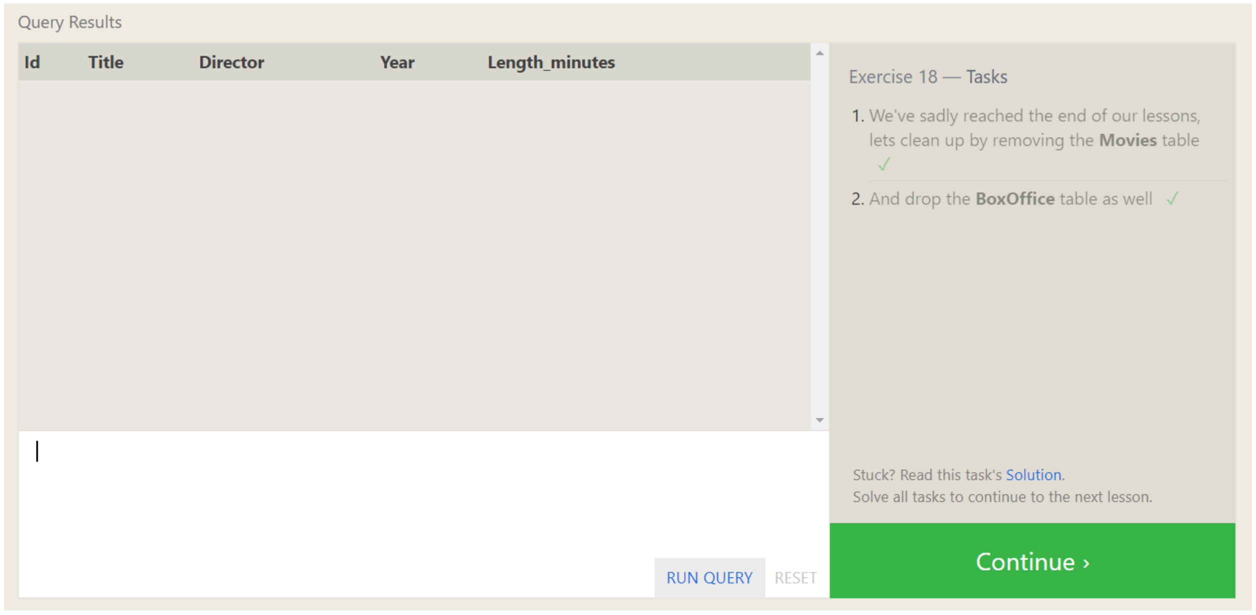
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

<u>Answer:</u>

1. ALTER TABLE Movies ADD COLUMN Aspect_ratio FLOAT DEFAULT 2.39;

2. ALTER TABLE Movies ADD COLUMN Language TEXT DEFAULT "English";

18) SQL Lesson 18: Dropping tables:



Query Results

Id	Title	Director	Year	Length_minutes

Exercise 18 — Tasks

1. We've sadly reached the end of our lessons, lets clean up by removing the **Movies** table ✓
2. And drop the **BoxOffice** table as well ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

<u>Answer:</u>

1. DROP TABLE Movies;

2. DROP TABLE BoxOffice;