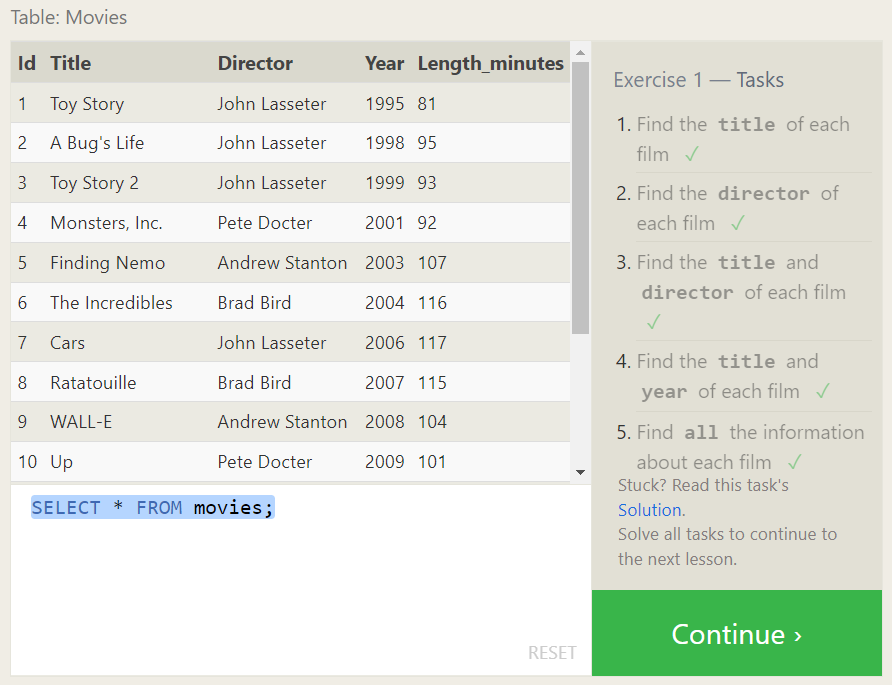
Exercise 1 — Tasks



1, Find the **title** of each film

SELECT title FROM movies;

1. Find the **director** of each film

SELECT director FROM movies;

1. Find the **title** and **director** of each film

SELECT Title ,director FROM movies;

1. Find the **title** and **year** of each film

SELECT Title ,Year FROM movies;

5. Find **all** the information about each film

SELECT \* FROM movies;

Exercise 2 — Tasks



1.Find the movie with a row **id** of 6

SELECT \* FROM movies where id =6 ;

2. Find the movies released in the **year**s between 2000 and 2010

SELECT \* FROM movies where year between 2000 and 2010 ;

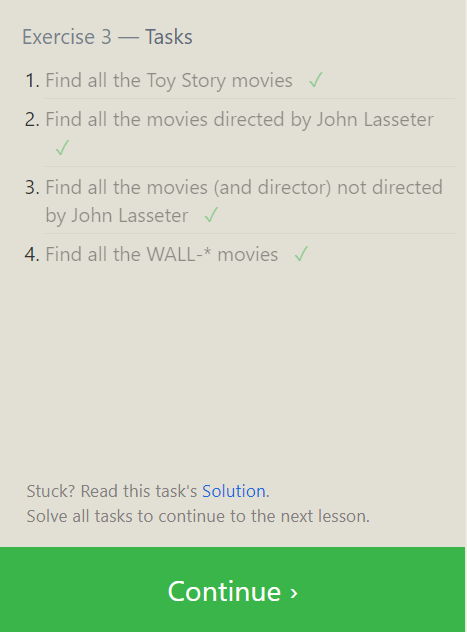
3. Find the movies **not** released in the **year**s between 2000 and 2010

SELECT \* FROM movies where year not between 2000 and 2010 ;

4 Find the first 5 Pixar movies and their release **year**

SELECT \* FROM movies limit 5;

Exercise 3 — Tasks



1. Find all the Toy Story movies

SELECT \* from movies where Title like 'Toy Story%'

2. Find all the movies directed by John Lassete

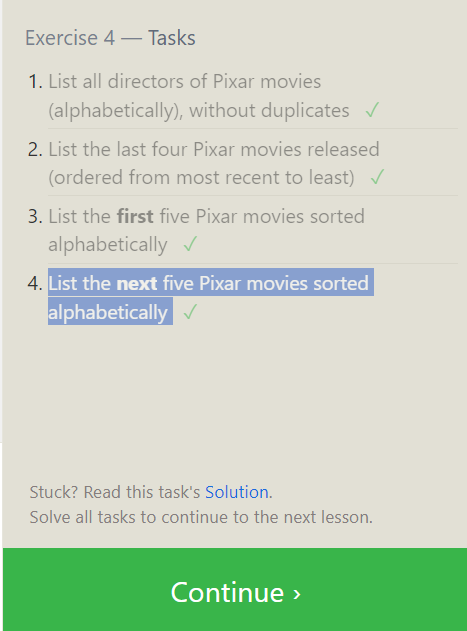
SELECT \* from movies where director like 'John%' ;

3.Find all the movies (and director) not directed by John Lassete

SELECT \* from movies where director not like 'John%' ;

4.Find all the WALL-\* movies

SELECT \* from movies where title like 'WALL-%' ;



Exercise 4 — Tasks

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)

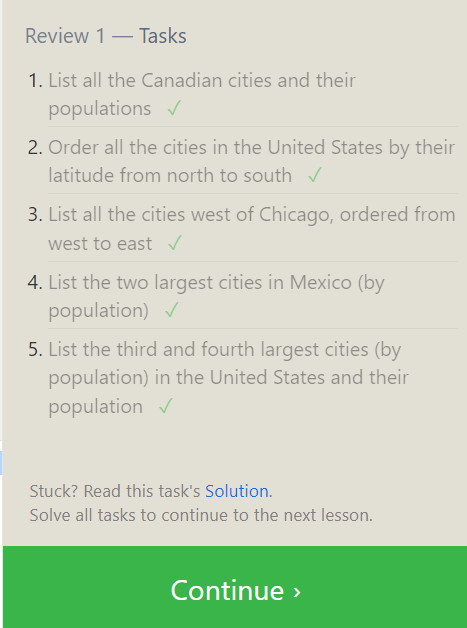
SELECT \* FROM movies order by year desc limit 4 ;

3.List the **first** five Pixar movies sorted alphabetically

SELECT \* FROM movies order by title asc limit 5 ;

4. List the **next** five Pixar movies sorted alphabetically

SELECT \* FROM movies order by title asc limit 5 offset 5 ;



Review 1 — Tasks

1.List all the Canadian cities and their populations

SELECT city,population FROM north\_american\_cities where country ='Canada'

2. 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;

3. List all the cities west of Chicago, ordered from west to east

SELECT \* FROM north\_american\_cities where longitude <-87.629798 order by longitude;

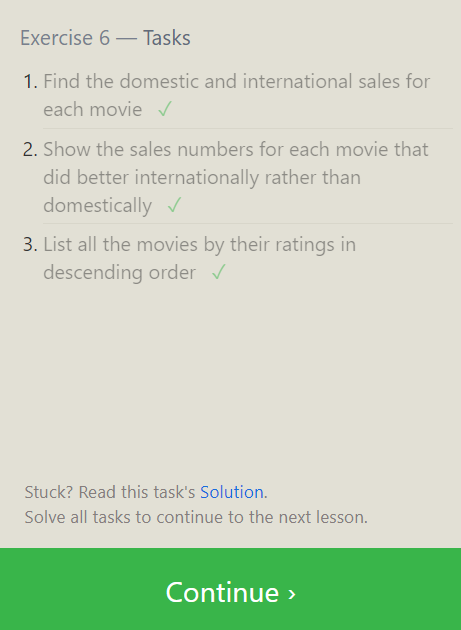
4. List the two largest cities in Mexico (by population)

SELECT \* FROM north\_american\_cities where country ='Mexico'order by population desc limit 2 ;

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

Exercise 6 — Tasks

SELECT \* FROM north\_american\_cities where country ='United States' order by population desc limit 2 offset 2 ;



1. Find the domestic and international sales for each movie ✓

SELECT \* FROM movies m join boxoffice b on M.Id= B.movie\_id;

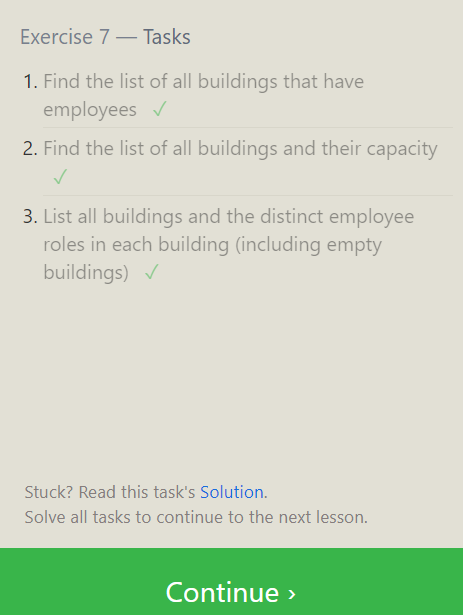
2 .Show the sales numbers for each movie that did better internationally rather than domestically

SELECT \* FROM movies m join boxoffice b on M.Id= B.movie\_id where b.International\_sales > Domestic\_sales;

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

SELECT \* FROM MOVIES m join boxoffice b on m.id = b.movie\_id order by b.rating desc ;

Exercise 7 — Tasks



1. Find the list of all buildings that have employees

SELECT distinct building FROM employees;

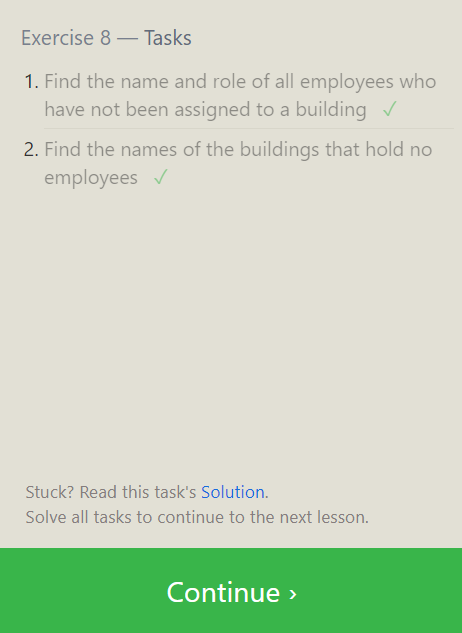
1. Find the list of all buildings and their capacity

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;

Exercise 8 — Tasks



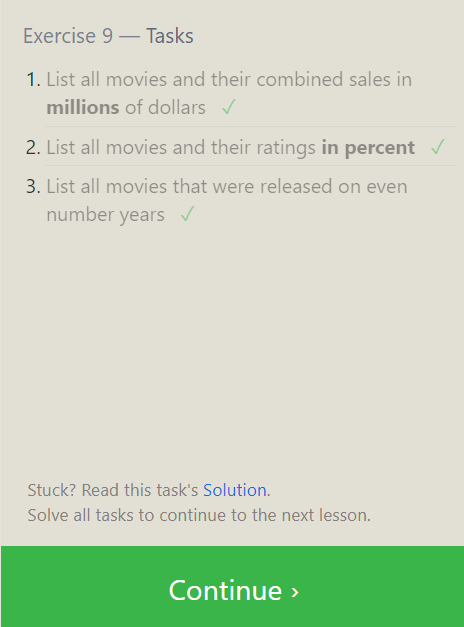
1. Find the name and role of all employees who have not been assigned to a building

SELECT \* FROM employees 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 ;

Exercise 9 — Tasks



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

SELECT title, (domestic\_sales + international\_sales) / 1000000 AS gross\_sales\_millionsFROM movies JOIN boxoffice ON movies.id = boxoffice.movie\_id;

1. List all movies and their ratings **in percent**

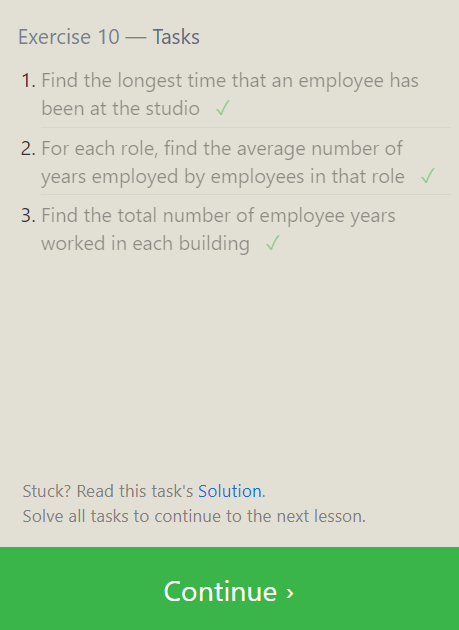
SELECT title, rating \* 10 AS rating\_percent

FROM movies JOIN boxoffice ON movies.id = boxoffice.movie\_id;

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

SELECT \* from movies where year%2=0;

Exercise 10 — Tasks



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

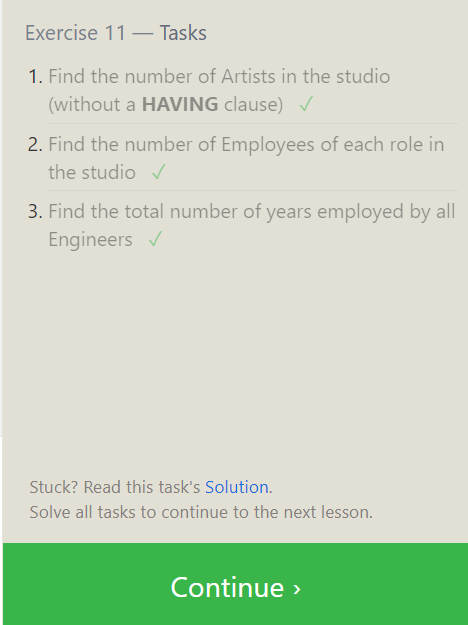
SELECT role, avg(years\_employed) FROM employees group by role;

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

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

FROM employeesGROUP BY building;

Exercise 11 — Tasks



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

SELECT count(role) FROM employees where role='Artist';

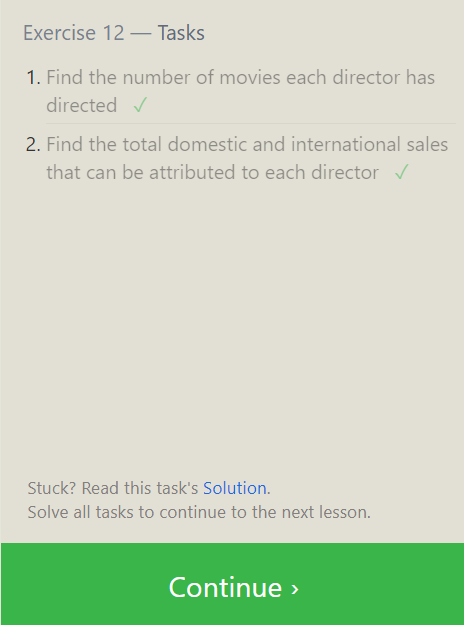
1. Find the number of Employees of each role in the studio

SELECT role,count(role) FROM employees group by role;

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

SELECT sum(years\_employed) from Employees where role ='Engineer' ;

Exercise 12 — Tasks



1. Find the number of movies each director has directed

SELECT director, COUNT(id) as Num\_movies\_directed FROM movies

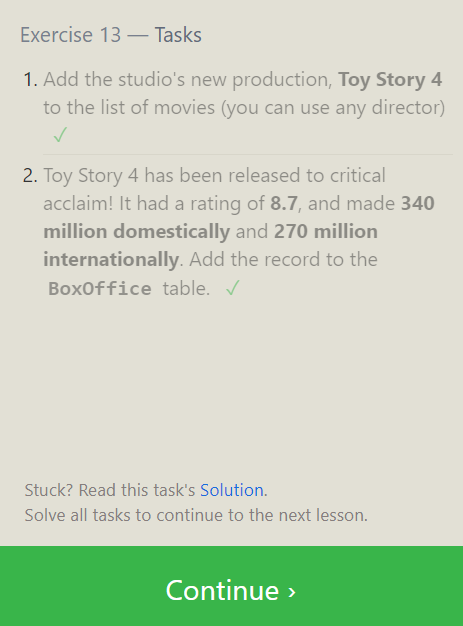
GROUP BY director;

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

SELECT director, SUM(domestic\_sales + international\_sales) as Cumulative\_sales\_from\_all\_movies FROM movies INNER JOIN boxoffice

ON movies.id = boxoffice.movie\_id GROUP BY director;

Exercise 13 — Tasks



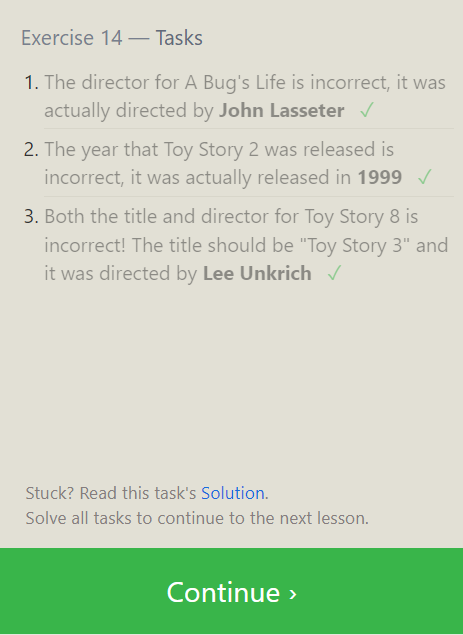
1. Add the studio's new production, **Toy Story 4** to the list of movies (you can use any director)

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

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);

Exercise 14 — Tasks



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

**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;

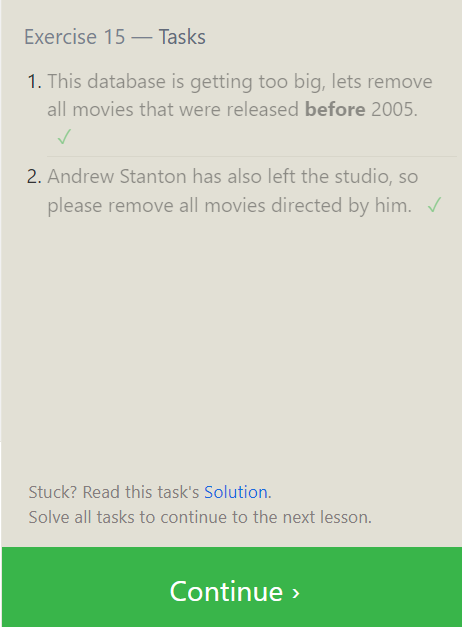
1. Both the title and director for Toy Story 8 is incorrect! The title should be "Toy Story 3" and it was directed by **Lee Unkrich**

UPDATE movies

SET title = "Toy Story 3", director = "Lee Unkrich"

WHERE id = 11;

Exercise 15 — Task



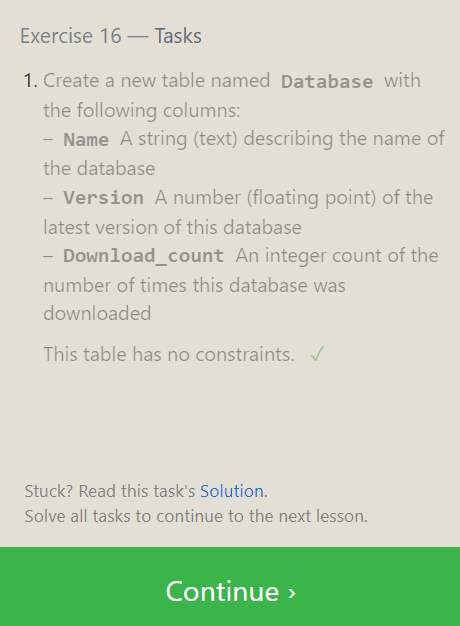
1. This database is getting too big, lets remove all movies that were released **before** 2005.

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’ ;

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

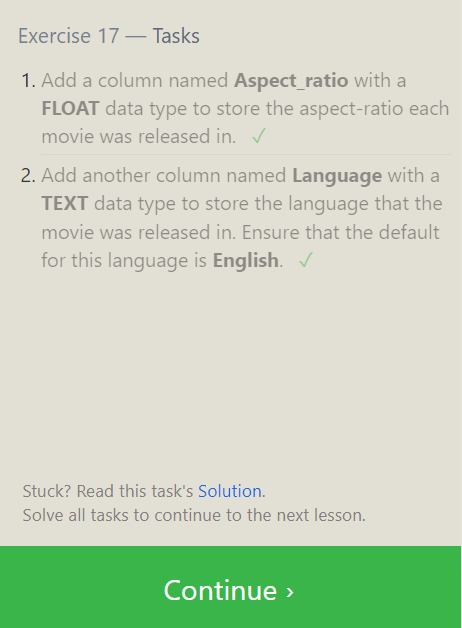
This table has no constraints.

CREATE TABLE Database (

Name TEXT,

Version FLOAT,

Download\_count INTEGER);

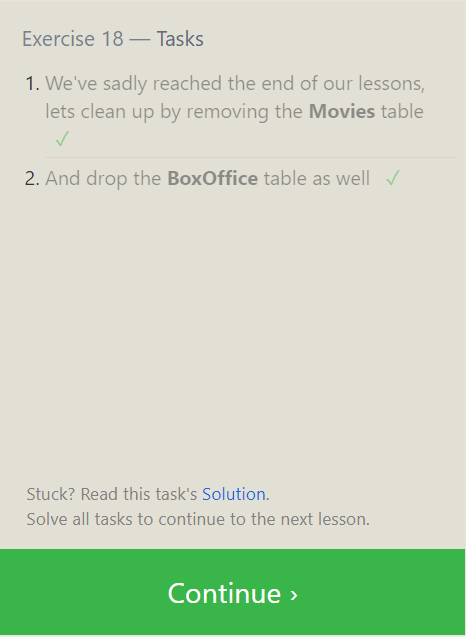


1. Add a column named **Aspect\_ratio** with a **FLOAT** data type to store the aspect-ratio each movie was released in.

ALTER TABLE Movies ADD COLUMN Aspect\_ratio FLOAT DEFAULT 2.39;

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";



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;