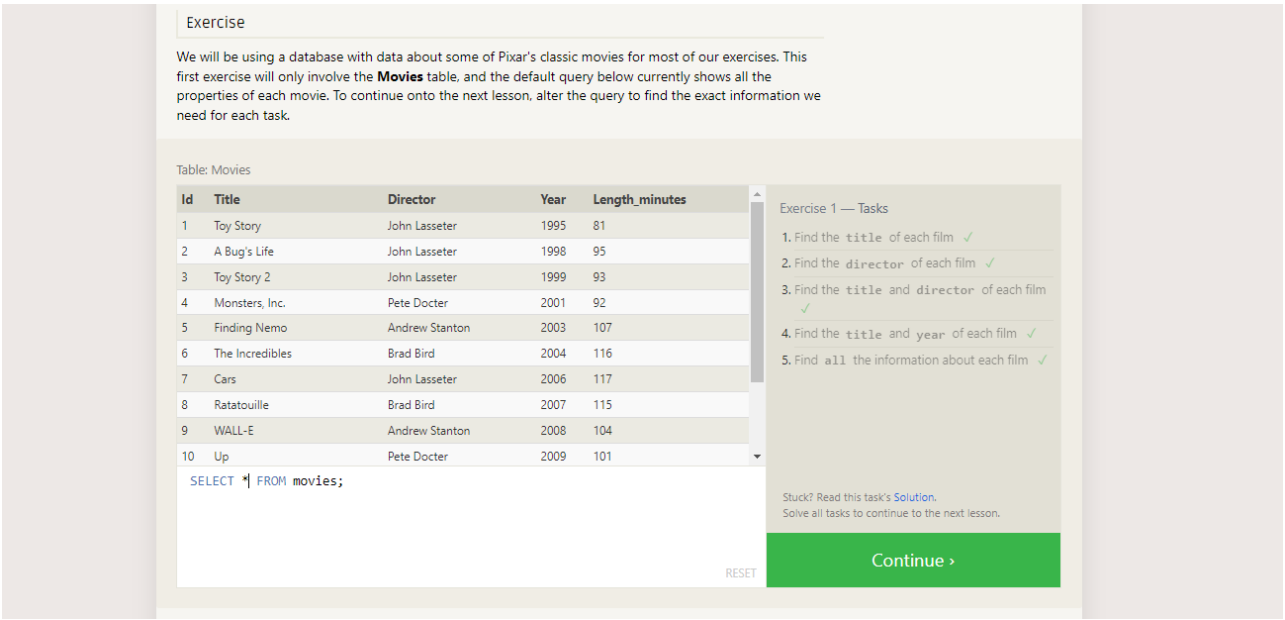
SQL Lesson 1: SELECT queries 101



Exercise

We will be using a database with data about some of Pixar's classic movies for most of our exercises. This first exercise will only involve the **Movies** table, and the default query below currently shows all the properties of each movie. To continue onto the next lesson, alter the query to find the exact information we need for each task.

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;
```

RESET

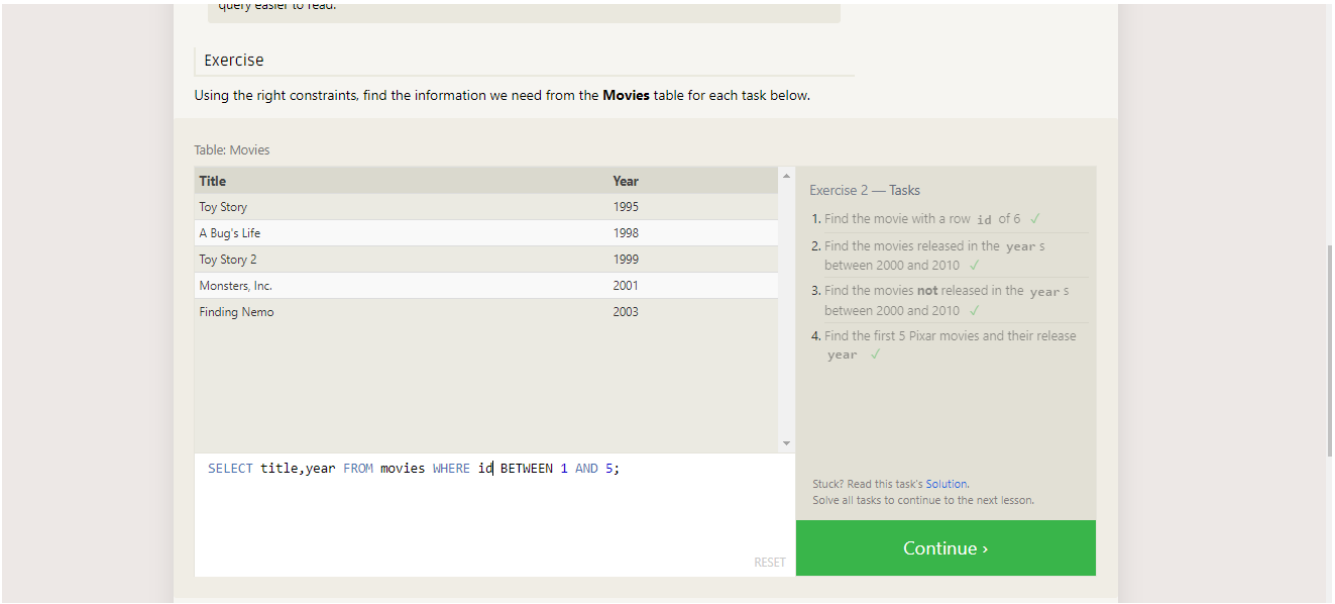
Exercise 1 — Tasks

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.

Continue ›

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



query easier to read.

Exercise

Using the right constraints, find the information we need from the **Movies** table for each task below.

Table: Movies

Title	Year
Toy Story	1995
A Bug's Life	1998
Toy Story 2	1999
Monsters, Inc.	2001
Finding Nemo	2003

```
SELECT title,year FROM movies WHERE id BETWEEN 1 AND 5;
```

RESET

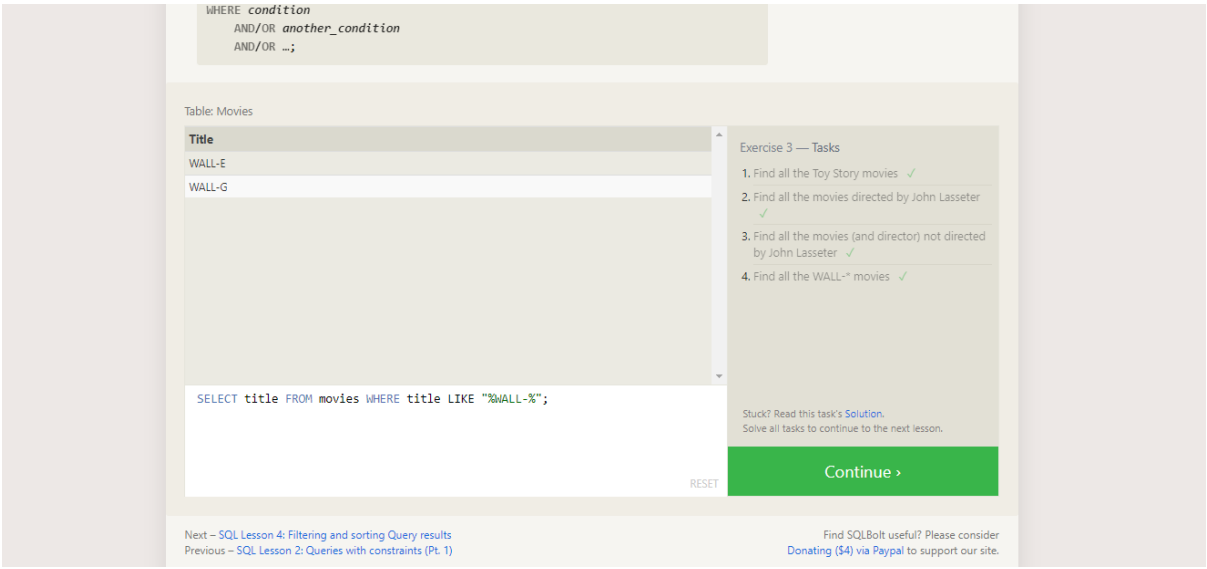
Exercise 2 — Tasks

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.

Continue ›

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



```
WHERE condition
    AND/OR another_condition
    AND/OR …;
```

Table: Movies

Title
WALL-E
WALL-G

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 ✓

```
SELECT title FROM movies WHERE title LIKE "%WALL-%";
```

RESET

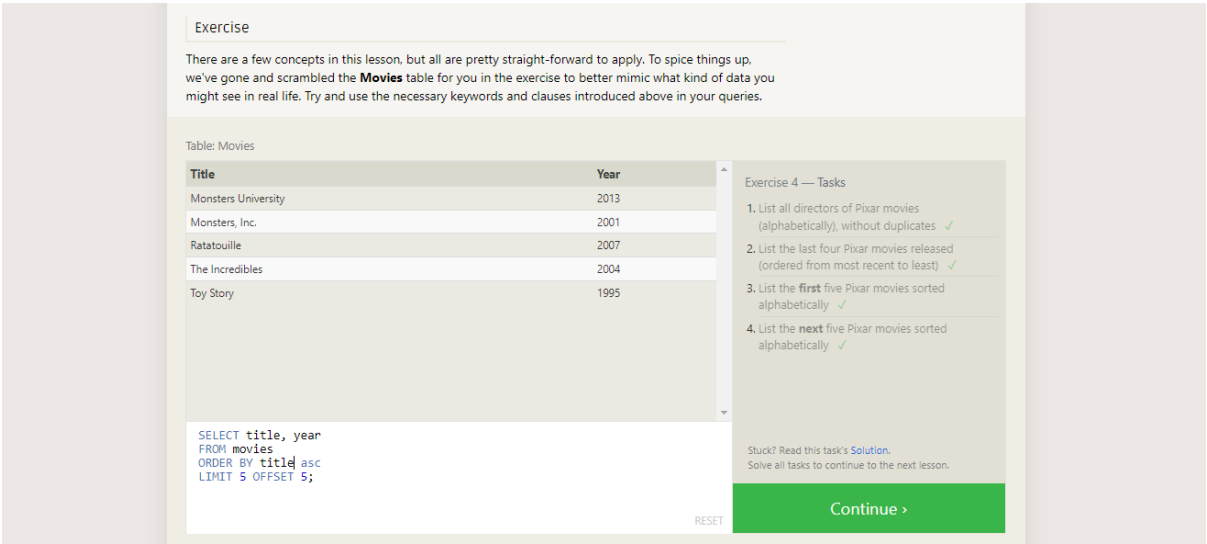
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

Next – SQL Lesson 4: Filtering and sorting Query results
Previous – SQL Lesson 2: Queries with constraints (Pt. 1)

Find SQLBolt useful? Please consider Donating ($4) via Paypal to support our site.

SQL Lesson 4: Filtering and sorting Query results



Exercise

There are a few concepts in this lesson, but all are pretty straight-forward to apply. To spice things up, we've gone and scrambled the **Movies** table for you in the exercise to better mimic what kind of data you might see in real life. Try and use the necessary keywords and clauses introduced above in your queries.

Table: Movies

Title	Year
Monsters University	2013
Monsters, Inc.	2001
Ratatouille	2007
The Incredibles	2004
Toy Story	1995

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 ✓

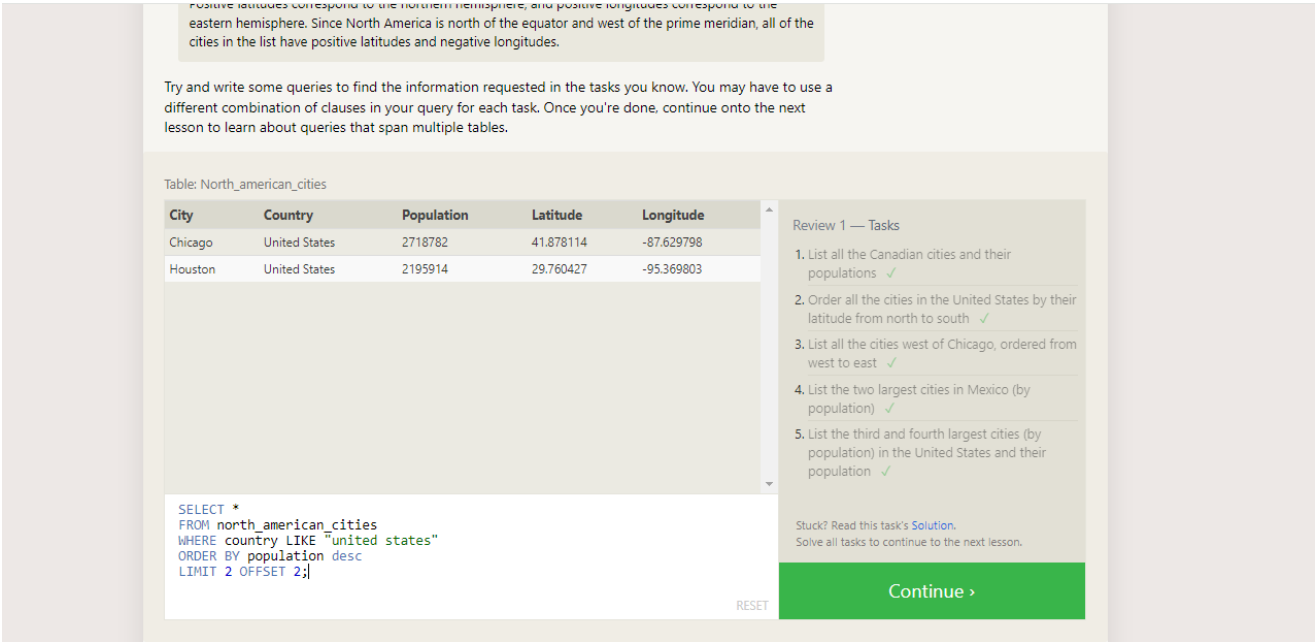
```
SELECT title, year
FROM movies
ORDER BY title asc
LIMIT 5 OFFSET 5;
```

RESET

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

Continue ›

SQL Review: Simple SELECT Queries



eastern hemisphere. Since North America is north of the equator and west of the prime meridian, all of the cities in the list have positive latitudes and negative longitudes.

Try and write some queries to find the information requested in the tasks you know. You may have to use a different combination of clauses in your query for each task. Once you're done, continue onto the next lesson to learn about queries that span multiple tables.

Table: North_american_cities

City	Country	Population	Latitude	Longitude
Chicago	United States	2718782	41.878114	-87.629798
Houston	United States	2195914	29.760427	-95.369803

```
SELECT *
FROM north_american_cities
WHERE country LIKE "united states"
ORDER BY population desc
LIMIT 2 OFFSET 2;
```

RESET

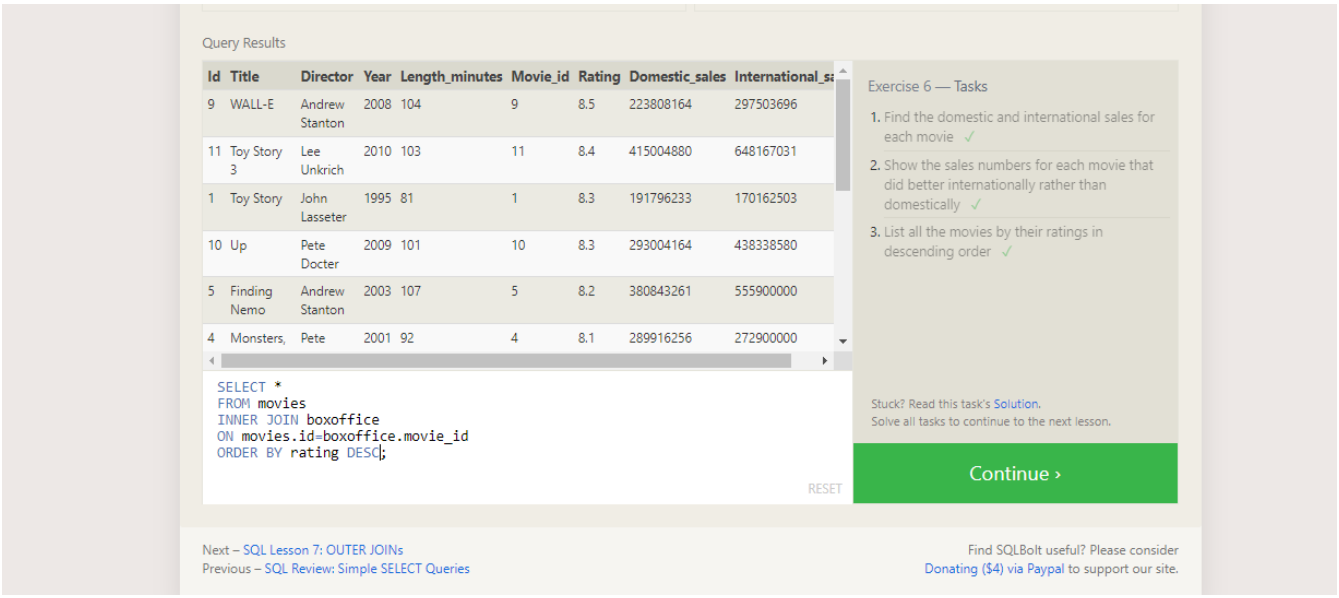
Review 1 — Tasks

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.

Continue ›

SQL Lesson 6: Multi-table queries with JOINs



Query Results

Id	Title	Director	Year	Length_minutes	Movie_id	Rating	Domestic_sales	International_sa
9	WALL-E	Andrew Stanton	2008	104	9	8.5	223808164	297503696
11	Toy Story 3	Lee Unkrich	2010	103	11	8.4	415004880	648167031
1	Toy Story	John Lasseter	1995	81	1	8.3	191796233	170162503
10	Up	Pete Docter	2009	101	10	8.3	293004164	438338580
5	Finding Nemo	Andrew Stanton	2003	107	5	8.2	380843261	555900000
4	Monsters,	Pete	2001	92	4	8.1	289916256	272900000

```
SELECT *
FROM movies
INNER JOIN boxoffice
ON movies.id=boxoffice.movie_id
ORDER BY rating DESC;
```

RESET

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 ✓

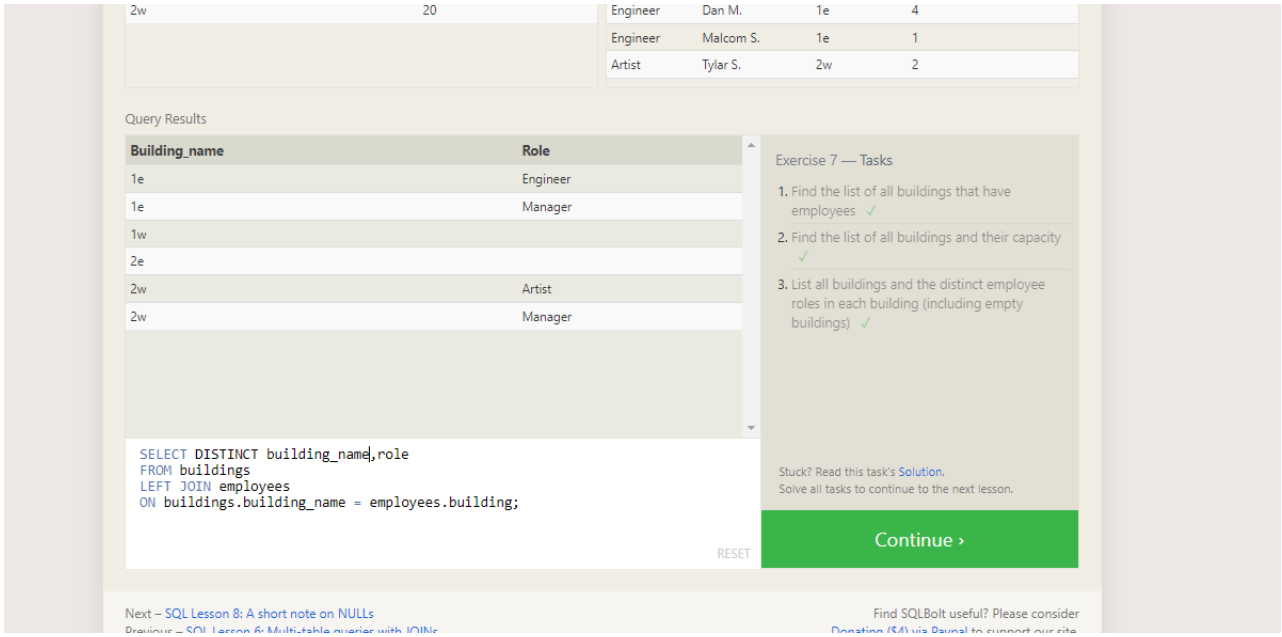
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

Next – SQL Lesson 7: OUTER JOINs
Previous – SQL Review: Simple SELECT Queries

Find SQLBolt useful? Please consider Donating ($4) via Paypal to support our site.

SQL Lesson 7: OUTER JOINs



2w	20

Engineer	Dan M.	1e	4
Engineer	Malcom S.	1e	1
Artist	Tylar S.	2w	2

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 buildings.building_name = employees.building;
```

RESET

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) ✓

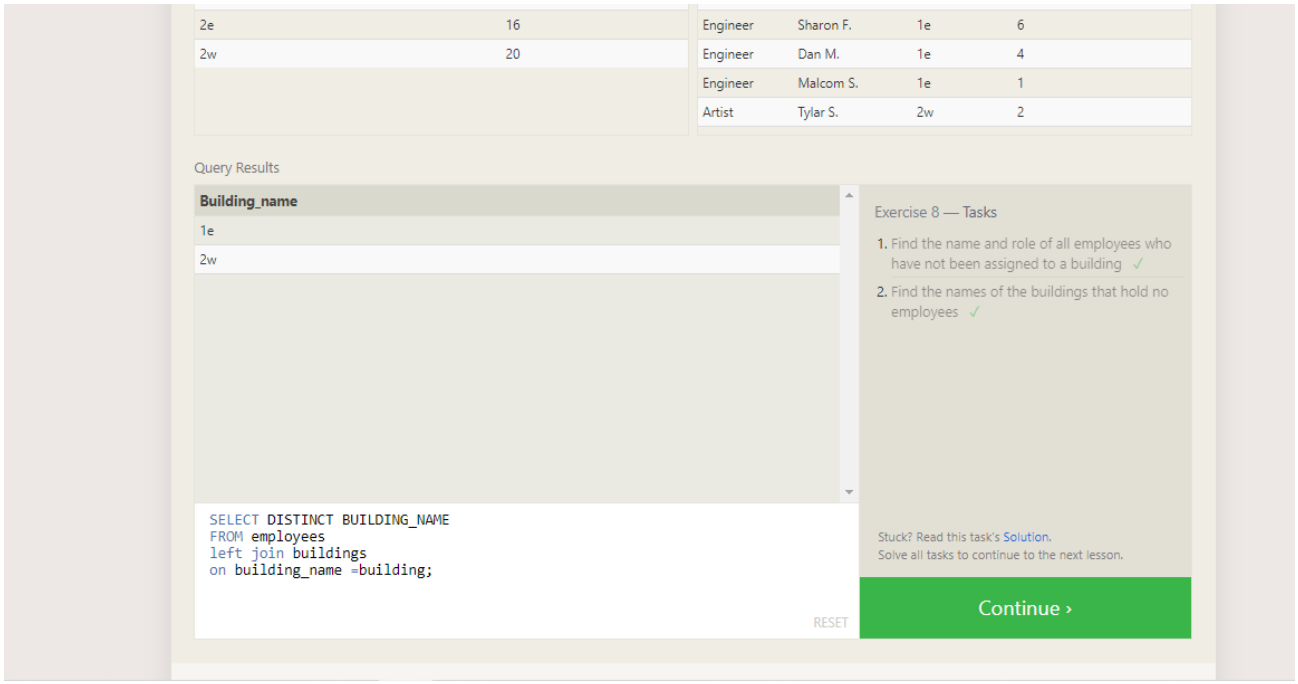
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

Next – SQL Lesson 8: A short note on NULLs

Find SQLBolt useful? Please consider

SQL Lesson 8: A short note on NULLs



2e	16
2w	20

Engineer	Sharon F.	1e	6
Engineer	Dan M.	1e	4
Engineer	Malcom S.	1e	1
Artist	Tylar S.	2w	2

Query Results

Building_name
1e
2w

```
SELECT DISTINCT BUILDING_NAME
FROM employees
left join buildings
on building_name =building;
```

RESET

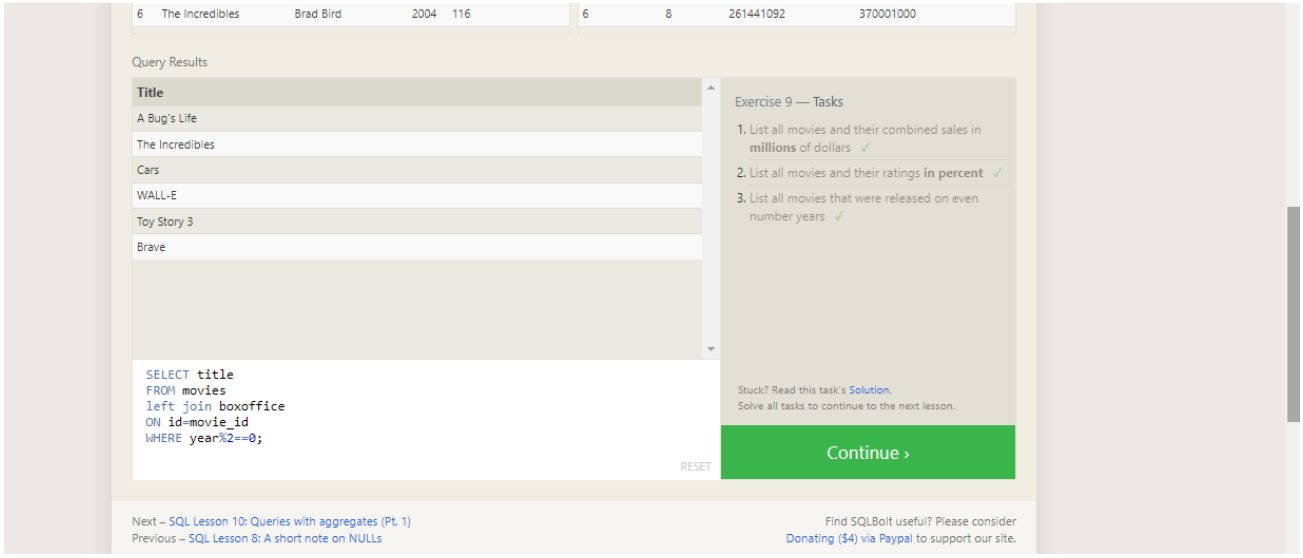
Exercise 8 — Tasks

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.

Continue ›

SQL Lesson 9: Queries with expressions



6 The Incredibles Brad Bird 2004 116

6 8 261441092 370001000

Query Results

Title
A Bug's Life
The Incredibles
Cars
WALL-E
Toy Story 3
Brave

```
SELECT title
FROM movies
left join boxoffice
ON id=movie_id
WHERE year%2==0;
```

RESET

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 ✓

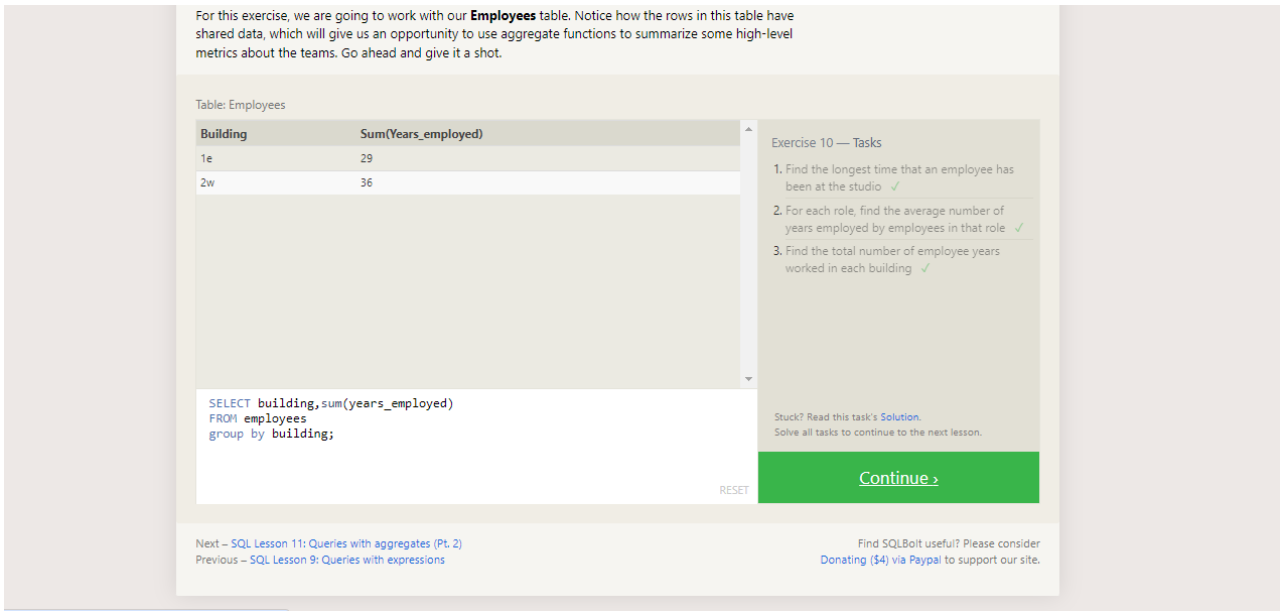
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

Next – SQL Lesson 10: Queries with aggregates (Pt. 1)
Previous – SQL Lesson 8: A short note on NULLs

Find SQLBolt useful? Please consider
Donating ($4) via Paypal to support our site.

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



For this exercise, we are going to work with our **Employees** table. Notice how the rows in this table have shared data, which will give us an opportunity to use aggregate functions to summarize some high-level metrics about the teams. Go ahead and give it a shot.

Table: Employees

Building	Sum(Years_employed)
1e	29
2w	36

```
SELECT building,sum(years_employed)
FROM employees
group by building;
```

RESET

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 ✓

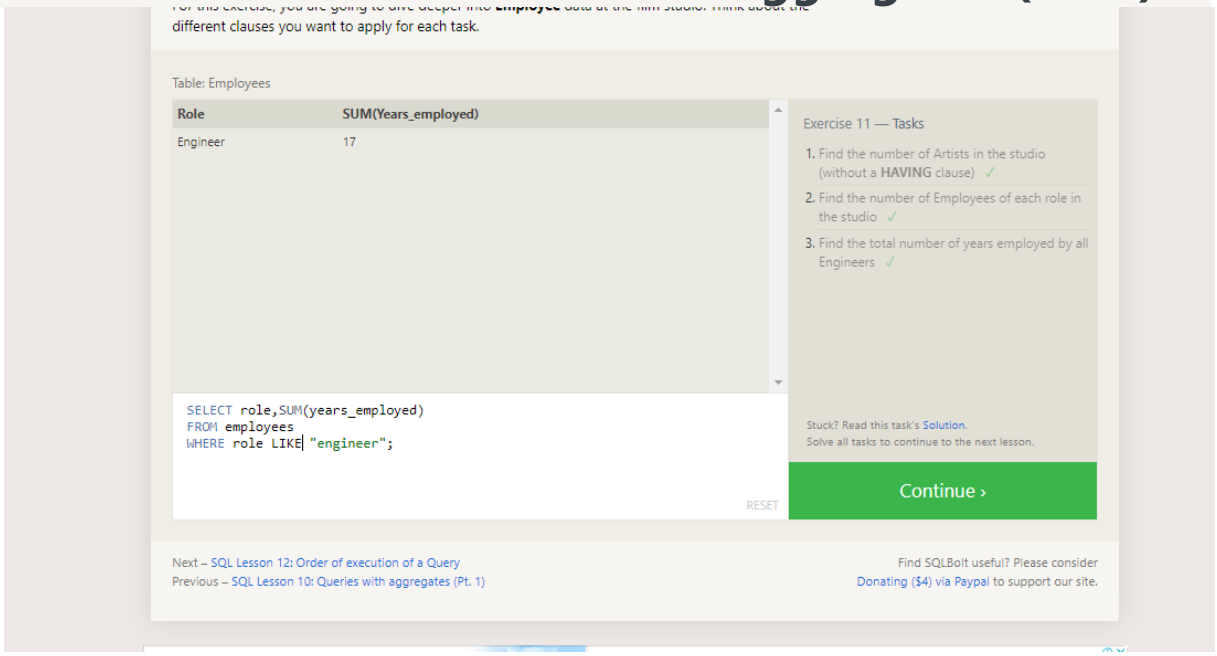
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

Next – SQL Lesson 11: Queries with aggregates (Pt. 2)
Previous – SQL Lesson 9: Queries with expressions

Find SQLBolt useful? Please consider
Donating ($4) via Paypal to support our site.

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



different clauses you want to apply for each task.

Table: Employees

Role	SUM(Years_employed)
Engineer	17

```
SELECT role,SUM(years_employed)
FROM employees
WHERE role LIKE "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 ✓

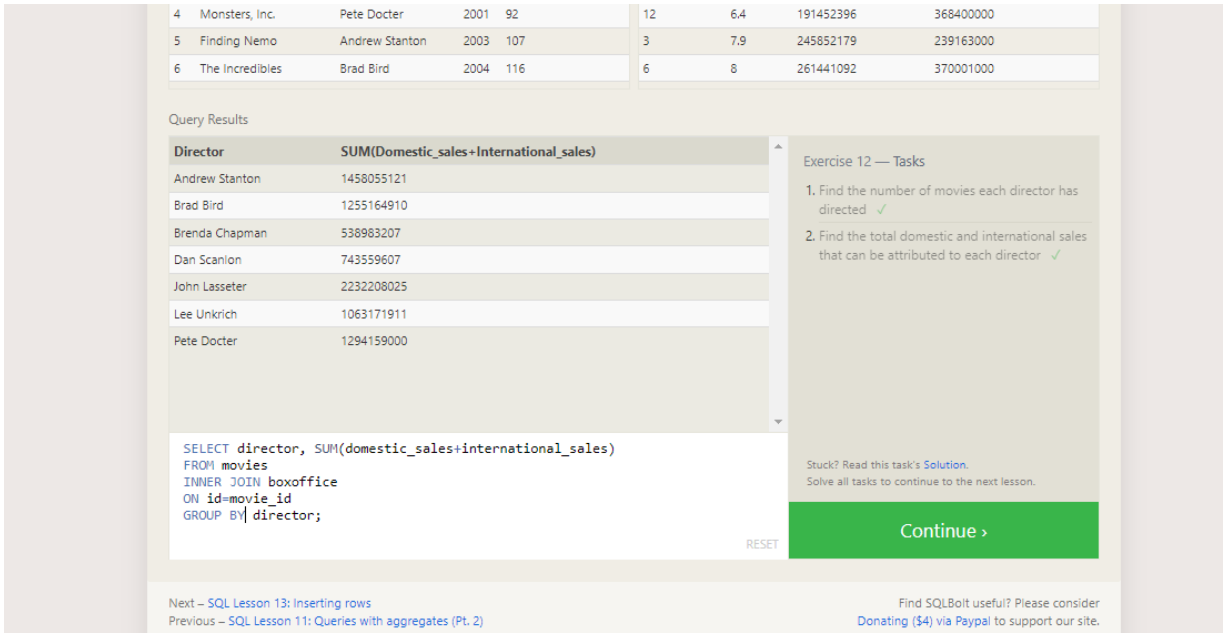
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

Next – SQL Lesson 12: Order of execution of a Query
Previous – SQL Lesson 10: Queries with aggregates (Pt. 1)

Find SQLBolt useful? Please consider Donating ($4) via Paypal to support our site.

SQL Lesson 12: Order of execution of a Query



4	Monsters, Inc.	Pete Docter	2001	92	12	6.4	191452396	368400000
5	Finding Nemo	Andrew Stanton	2003	107	3	7.9	245852179	239163000
6	The Incredibles	Brad Bird	2004	116	6	8	261441092	370001000

Query Results

Director	SUM(Domestic_sales+International_sales)
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)
FROM movies
INNER JOIN boxoffice
ON id=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 ✓

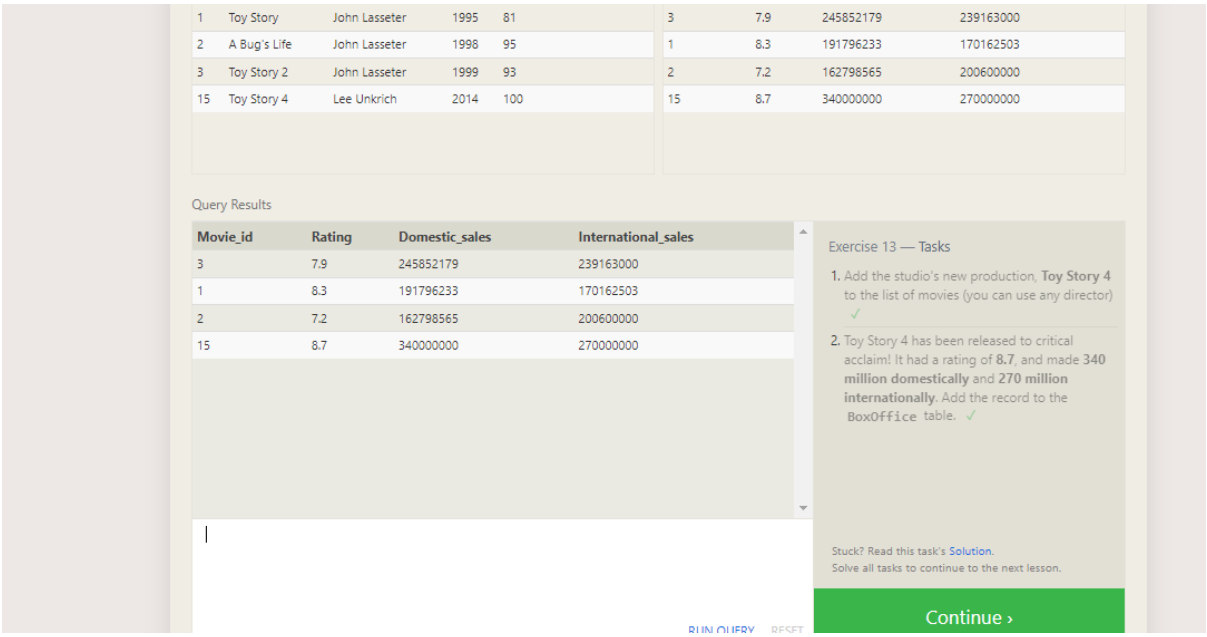
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

Next – SQL Lesson 13: Inserting rows
Previous – SQL Lesson 11: Queries with aggregates (Pt. 2)

Find SQLBolt useful? Please consider Donating ($4) via Paypal to support our site.

SQL Lesson 13: Inserting rows



1	Toy Story	John Lasseter	1995	81	3	7.9	245852179	239163000
2	A Bug's Life	John Lasseter	1998	95	1	8.3	191796233	170162503
3	Toy Story 2	John Lasseter	1999	93	2	7.2	162798565	200600000
15	Toy Story 4	Lee Unkrich	2014	100	15	8.7	340000000	270000000

Query Results

Movie_id	Rating	Domestic_sales	International_sales
3	7.9	245852179	239163000
1	8.3	191796233	170162503
2	7.2	162798565	200600000
15	8.7	340000000	270000000

Exercise 13 — Tasks

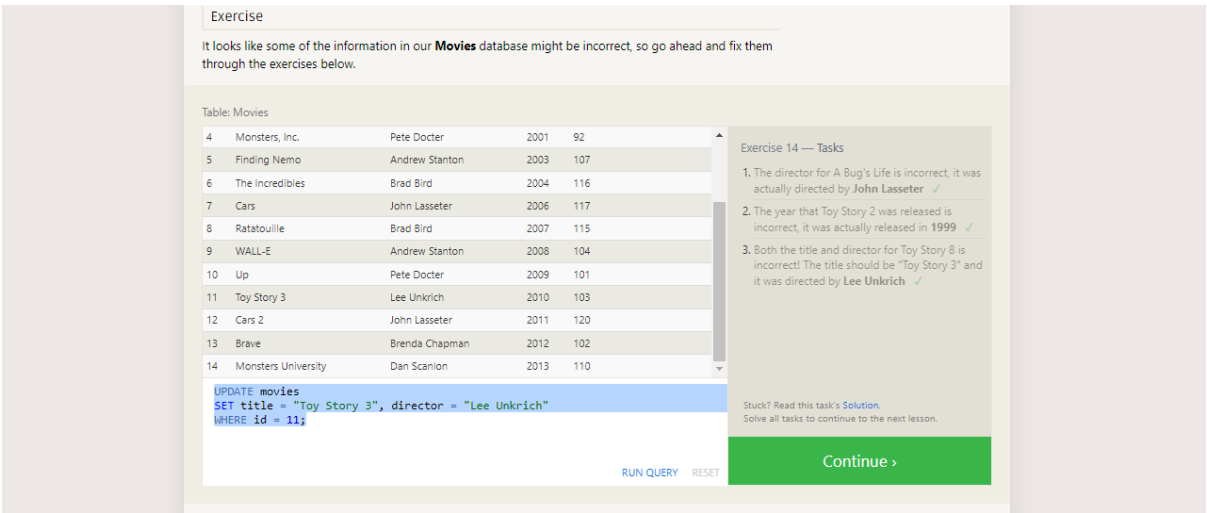
1. Add the studio's new production, **Toy Story 4** to the list of movies (you can use any director) ✓
2. Toy Story 4 has been released to critical acclaim! It had a rating of **8.7**, and made **340 million domestically** and **270 million internationally**. Add the record to the `BoxOffice` table. ✓

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

Continue ›

RUN QUERY RESET

SQL Lesson 14: Updating rows



Exercise

It looks like some of the information in our **Movies** database might be incorrect, so go ahead and fix them through the exercises below.

Table: Movies

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
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

```
UPDATE movies
SET title = "Toy Story 3", director = "Lee Unkrich"
WHERE id = 11;
```

RUN QUERY RESET

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.

Continue ›

SQL Lesson 15: Deleting rows

The database needs to be cleaned up a little bit, so try and delete a few rows in the tasks below.

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.

RUN QUERY RESET

Continue ›

SQL Lesson 16: Creating tables

Exercise

In this exercise, you'll need to create a new table for us to insert some new rows into.

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

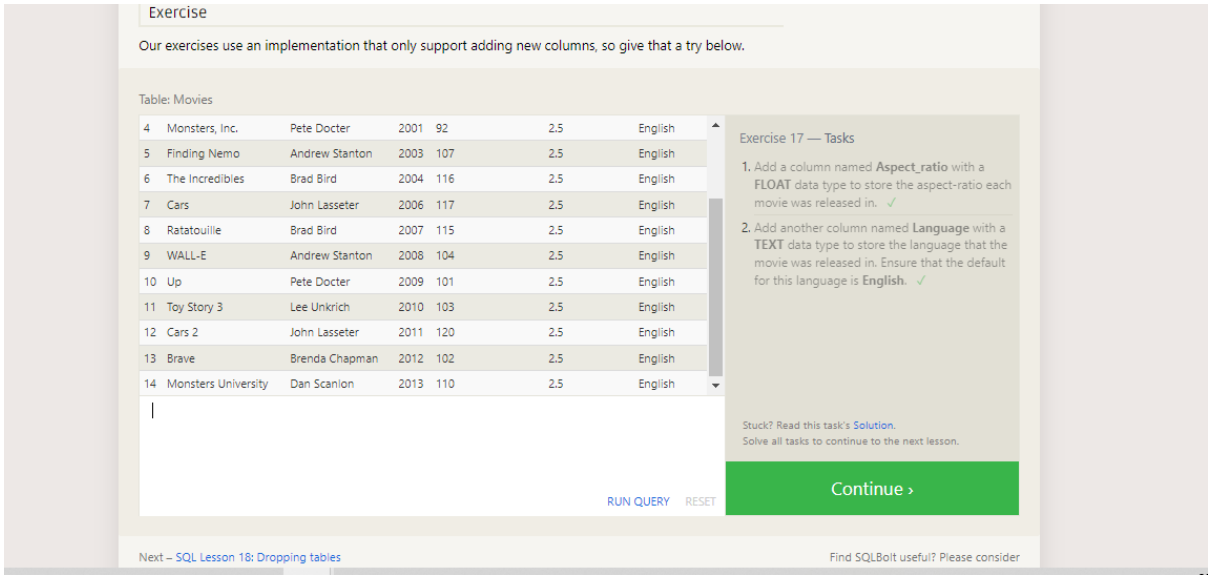
   This table has no constraints. ✓

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

RUN QUERY RESET

Continue ›

SQL Lesson 17: Altering tables



Exercise

Our exercises use an implementation that only support adding new columns, so give that a try below.

Table: Movies

4	Monsters, Inc.	Pete Docter	2001	92	2.5	English
5	Finding Nemo	Andrew Stanton	2003	107	2.5	English
6	The Incredibles	Brad Bird	2004	116	2.5	English
7	Cars	John Lasseter	2006	117	2.5	English
8	Ratatouille	Brad Bird	2007	115	2.5	English
9	WALL-E	Andrew Stanton	2008	104	2.5	English
10	Up	Pete Docter	2009	101	2.5	English
11	Toy Story 3	Lee Unkrich	2010	103	2.5	English
12	Cars 2	John Lasseter	2011	120	2.5	English
13	Brave	Brenda Chapman	2012	102	2.5	English
14	Monsters University	Dan Scanlon	2013	110	2.5	English

RUN QUERY RESET

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**. ✓

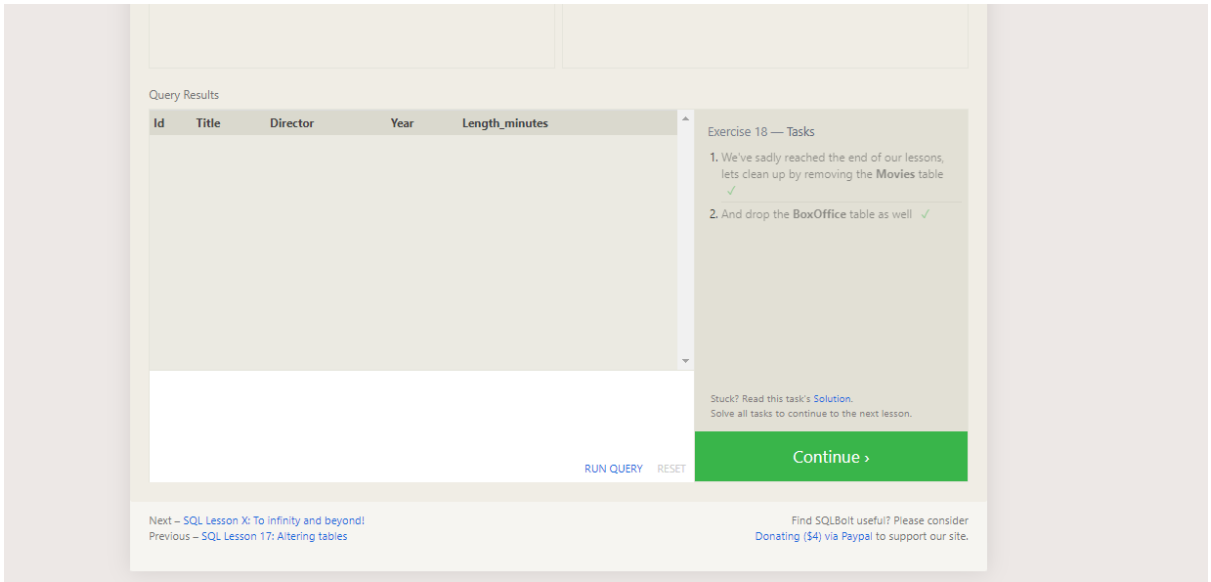
Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

Next – SQL Lesson 18: Dropping tables

Find SQLBolt useful? Please consider

SQL Lesson 18: Dropping tables



Query Results

Id	Title	Director	Year	Length_minutes

RUN QUERY RESET

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.

Continue ›

Next – SQL Lesson X: To infinity and beyond!
Previous – SQL Lesson 17: Altering tables

Find SQLBolt useful? Please consider
Donating ($4) via Paypal to support our site.