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
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 - 1 Tasks

1. Find the **“title”** of each film

   SELECT title FROM movies;

2. Find the **“director”** of each film

   SELECT director FROM movies;

3. Find the **“title”** and **“director”** of each film

   SELECT title, director FROM movies;

4. Find the **“title”** and **“year”** of each film

   SELECT title, year FROM movies;

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

   SELECT * FROM movies;

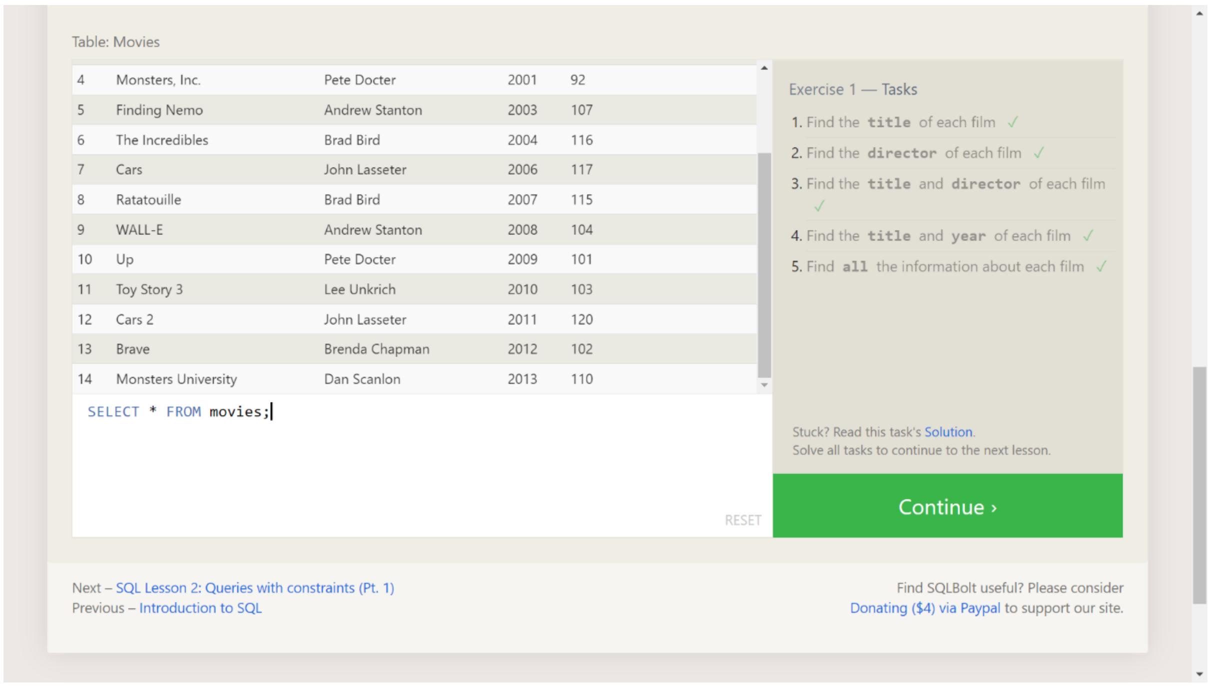


Table: Movies

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

```
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.

Next – SQL Lesson 2: Queries with constraints (Pt. 1)
Previous – Introduction to SQL

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

Exercise - 2 Tasks

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

   SELECT * FROM movies WHERE id = 6;

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

   SELECT * FROM movies WHERE year BETWEEN 2000 AND 2010;

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

   SELECT * FROM movies WHERE year NOT BETWEEN 2000 AND 2010;

4. Find the first 5 Pixar movies and their realease **year**

   SELECT title, year FROM movies LIMIT 5;

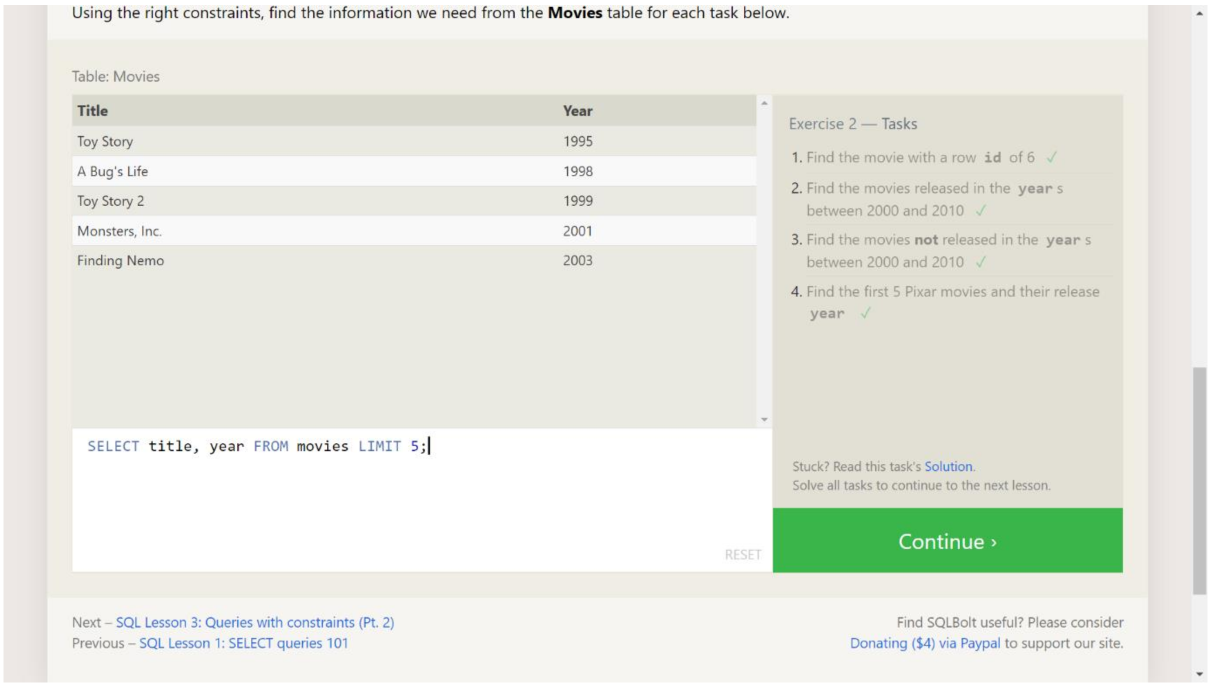

Using the right constraints, find the information we need from the **Movies** table for each task below.

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 LIMIT 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.

Next – SQL Lesson 3: Queries with constraints (Pt. 2)
Previous – SQL Lesson 1: SELECT queries 101

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

<u>Exercise - 3 Tasks</u>

1. Find all the Toy Story movies

   SELECT * FROM movies WHERE title LIKE '%Toy%';

2. Find all the movies directed by Jhon Lasseter

   SELECT * FROM movies WHERE director LIKE '%John Lasseter%';

3. Find all the movies (and director) not directed by Jhon Lasseter

   SELECT * FROM movies WHERE director NOT LIKE '%John Lasseter%';

4. Find all the WALL-* movies

   SELECT * FROM movies where title Like '%Wall%';

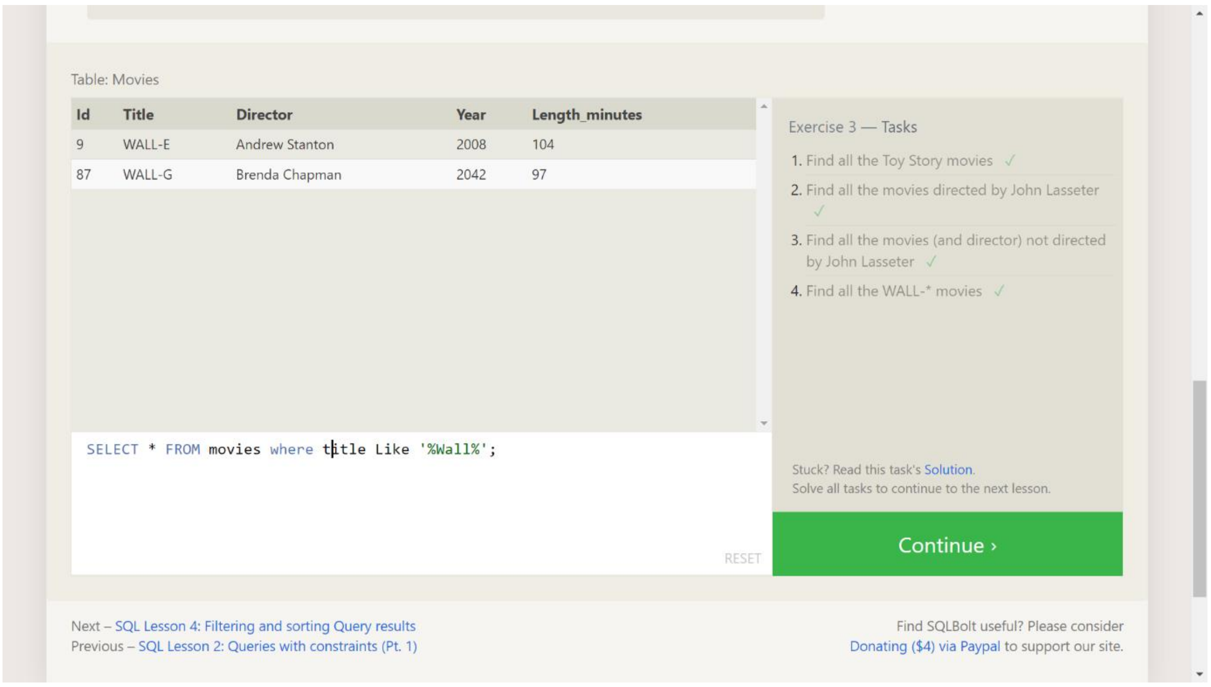


Table: Movies

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.

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.

<u>Exercise - 4 Tasks</u>

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

   SELECT DISTINCT director FROM movies ORDER BY director;

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

   SELECT title,year FROM movies ORDER BY year DESC LIMIT 4;

3. List the first five Pixar movies sorted alphabetically

   SELECT title FROM movies ORDER BY title LIMIT 5;

4. List the next five Pixar movies sorted alphabetically

   SELECT title FROM movies ORDER BY title LIMIT 5 OFFSET 5;

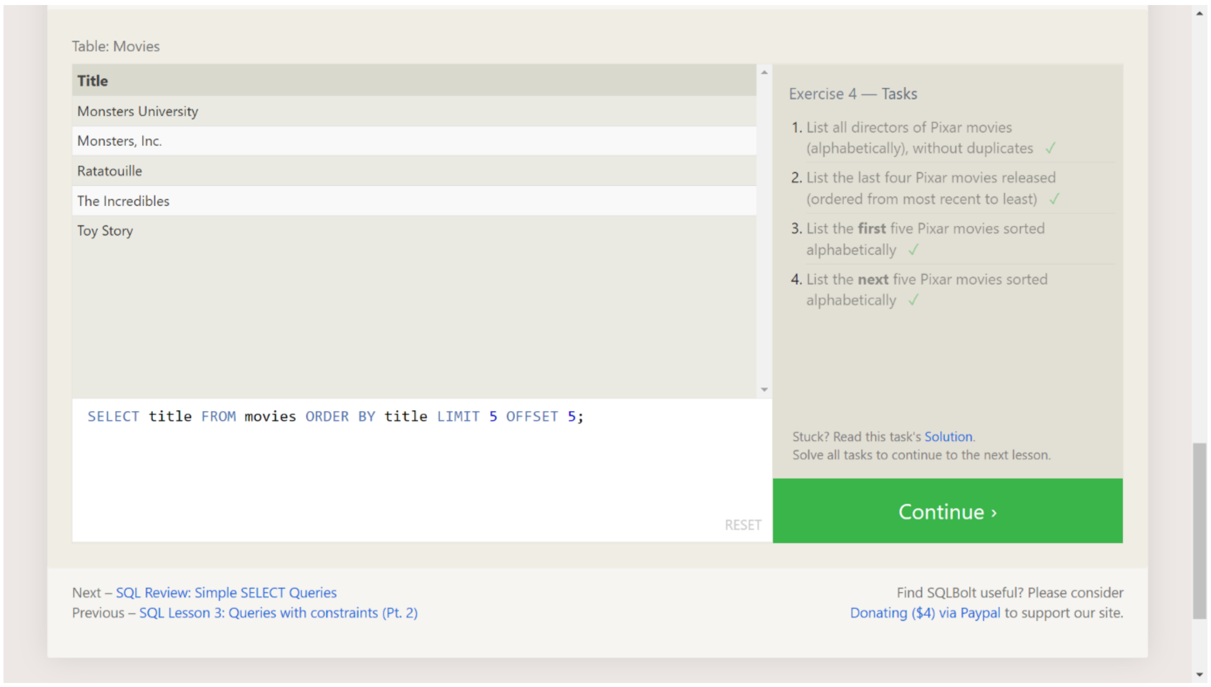


Table: Movies

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.

Next – SQL Review: Simple SELECT Queries
Previous – SQL Lesson 3: Queries with constraints (Pt. 2)

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

Table: North_american_cities

City	Country	Population	Latitude	Longitude
Guadalajara	Mexico	1500800	20.659699	-103.349609
Toronto	Canada	2795060	43.653226	-79.383184
Houston	United States	2195914	29.760427	-95.369803
New York	United States	8405837	40.712784	-74.005941
Philadelphia	United States	1553165	39.952584	-75.165222
Havana	Cuba	2106146	23.05407	-82.345189
Mexico City	Mexico	8555500	19.432608	-99.133208
Phoenix	United States	1513367	33.448377	-112.074037
Los Angeles	United States	3884307	34.052234	-118.243685
Ecatepec de Morelos	Mexico	1742000	19.601841	-99.050674
Montreal	Canada	1717767	45.501689	-73.567256
Chicago	United States	2718782	41.878114	-87.629798

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 city, latitude 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 city,longitude FROM North_american_cities WHERE longitude < -87.629798 ORDER BY longitude;

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

   SELECT city,population FROM North_american_cities WHERE country LIKE "Mexico" ORDER BY Population DESC LIMIT 2;

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

   SELECT city,population FROM North_american_cities WHERE country LIKE "United States" ORDER BY population DESC LIMIT 2 OFFSET 2;

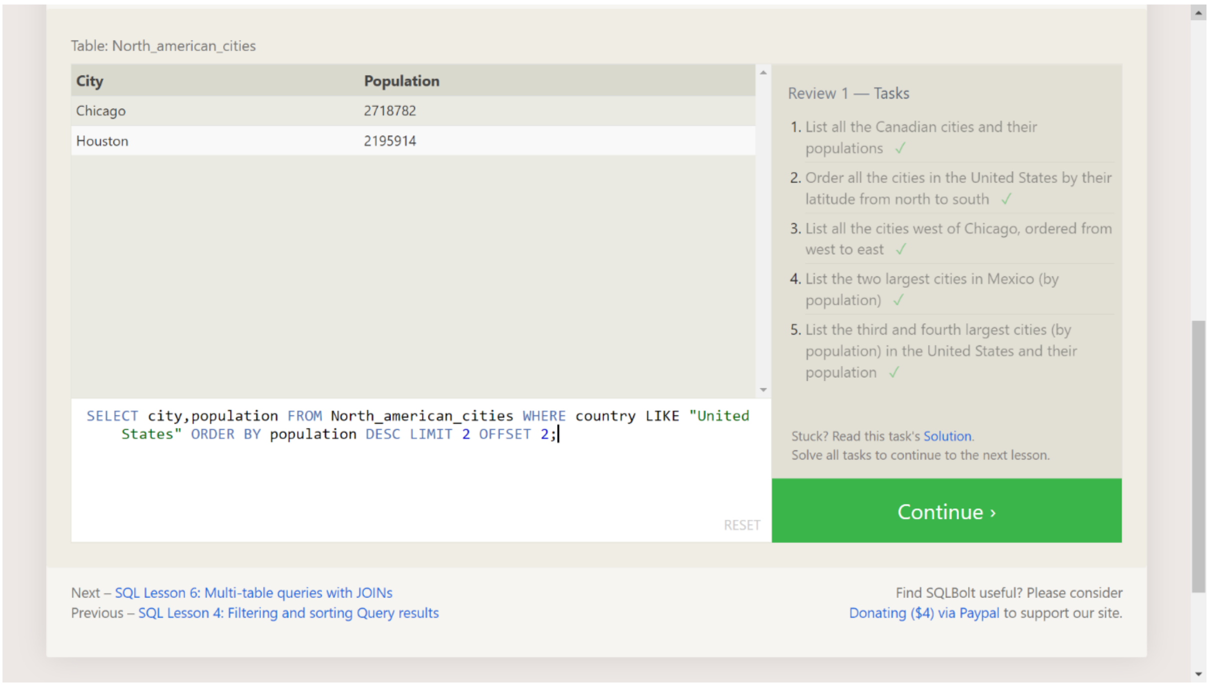


Table: North_american_cities

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.

Next – SQL Lesson 6: Multi-table queries with JOINs
Previous – SQL Lesson 4: Filtering and sorting Query results

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

Table: Movies (Read-Only)

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
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

Table: Boxoffice (Read-Only)

Movie_id	Rating	Domestic_sales	International_sales
5	8.2	380843261	555900000
14	7.4	268492764	475066843
8	8	206445654	417277164
12	6.4	191452396	368400000
3	7.9	245852179	239163000
6	8	261441092	370001000
9	8.5	223808164	297503696
11	8.4	415004880	648167031
1	8.3	191796233	170162503
7	7.2	244082982	217900167
10	8.3	293004164	438338580
4	8.1	289916256	272900000
2	7.2	162798565	200600000
13	7.2	237283207	301700000

Excercise - 6 Tasks

1. Find the domestic and international sales for each movie

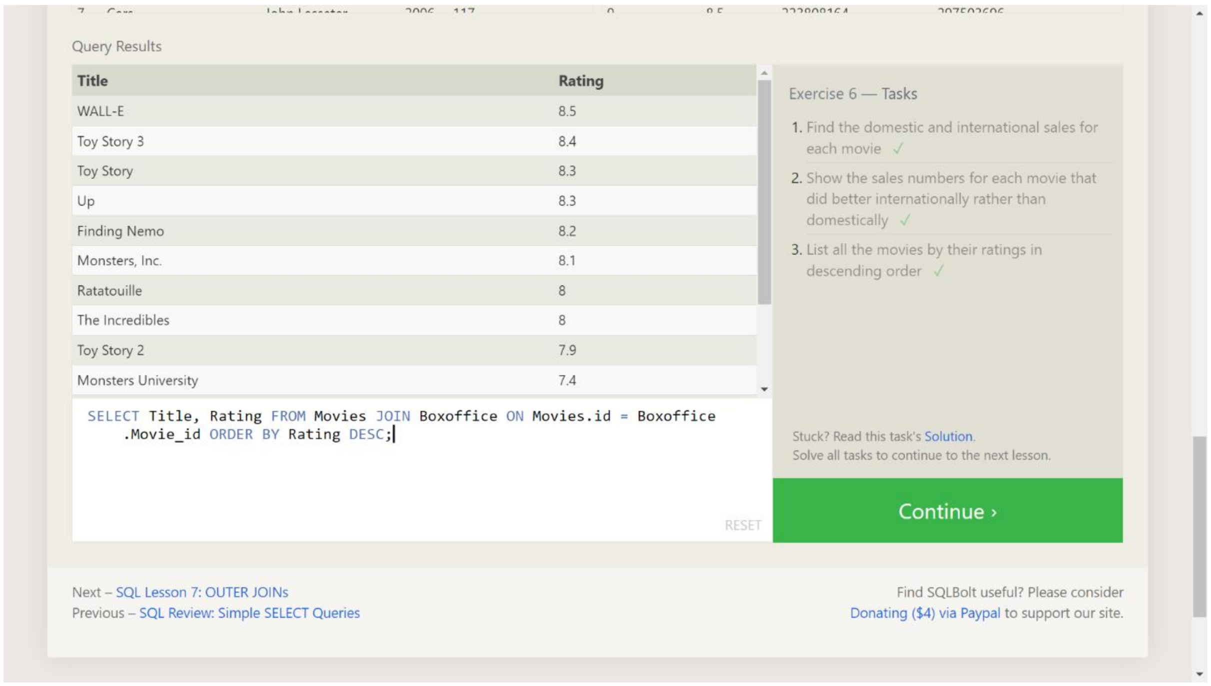
   SELECT Title, Domestic_sales, International_sales FROM Movies JOIN Boxoffice ON Movies.id = Boxoffice.movie_id;

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

   SELECT Title, Domestic_sales, International_sales FROM Movies JOIN Boxoffice ON Movies.id = Boxoffice.movie_id WHERE International_sales > Domestic_sales;

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

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



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

```
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.

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.

Table: Buildings (Read-Only)

Building_name	Capacity
1e	24
1w	32
2e	16
2w	20

Table: Employees (Read-Only)

Role	Name	Building	Years_employed
Engineer	Becky A.	1e	4
Engineer	Dan B.	1e	2
Engineer	Sharon F.	1e	6
Engineer	Dan M.	1e	4
Engineer	Malcom S.	1e	1
Artist	Tylar S.	2w	2
Artist	Sherman D.	2w	8
Artist	Jakob J.	2w	6
Artist	Lillia A.	2w	7
Artist	Brandon J.	2w	7
Manager	Scott K.	1e	9
Manager	Shirlee M.	1e	3
Manager	Daria O.	2w	6

Excercise - 7 Tasks

1. Find the list of all buildings that have employees

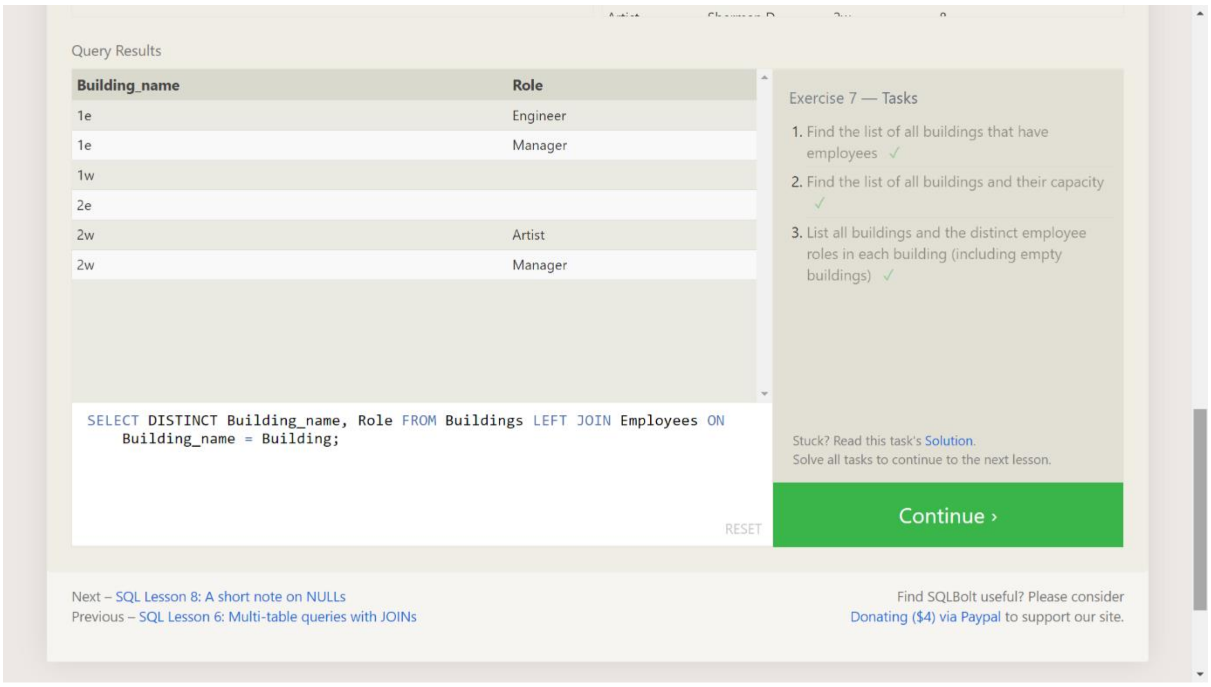
   SELECT DISTINCT Building FROM Employees;

2. Find the list of all buildings and their capacity

   SELECT * FROM Buildings;

3. 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;


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.

Next – SQL Lesson 8: A short note on NULLs
Previous – SQL Lesson 6: Multi-table queries with JOINs

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

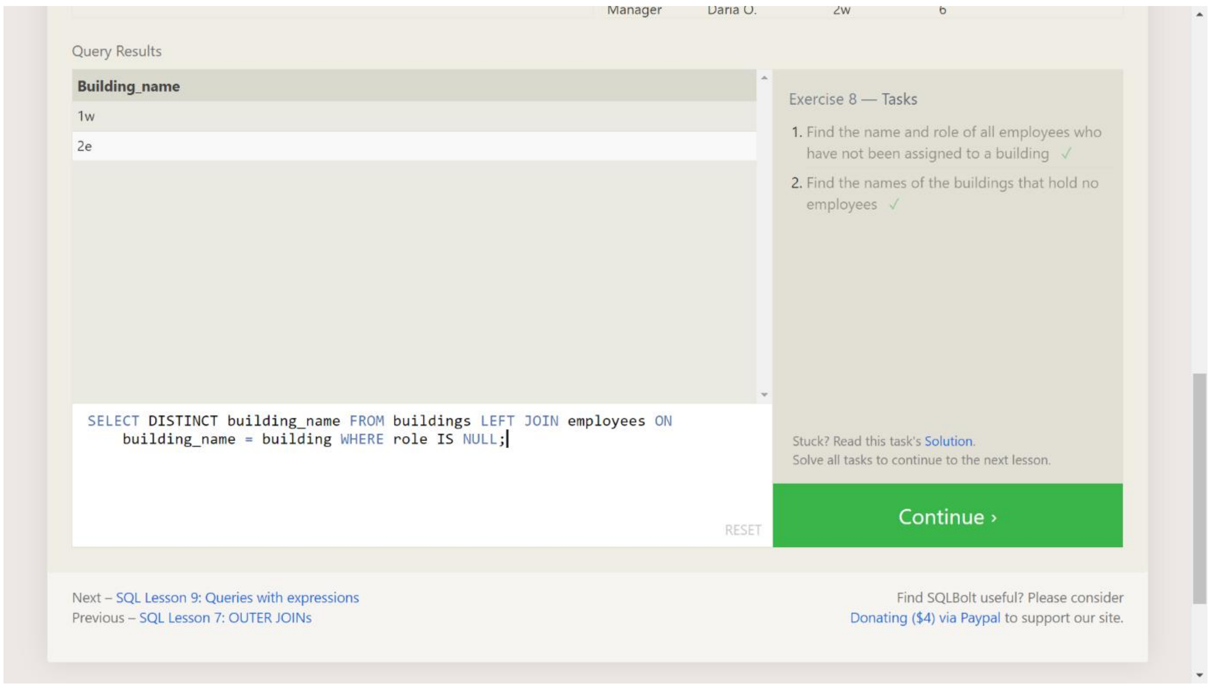
Excercise - 8 Tasks

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

   SELECT Name, Role FROM Employees WHERE Building IS NULL;

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

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


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.

Next – SQL Lesson 9: Queries with expressions
Previous – SQL Lesson 7: OUTER JOINs

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

Excercise - 9 Tasks

1. Find the list of all buildings that have employees

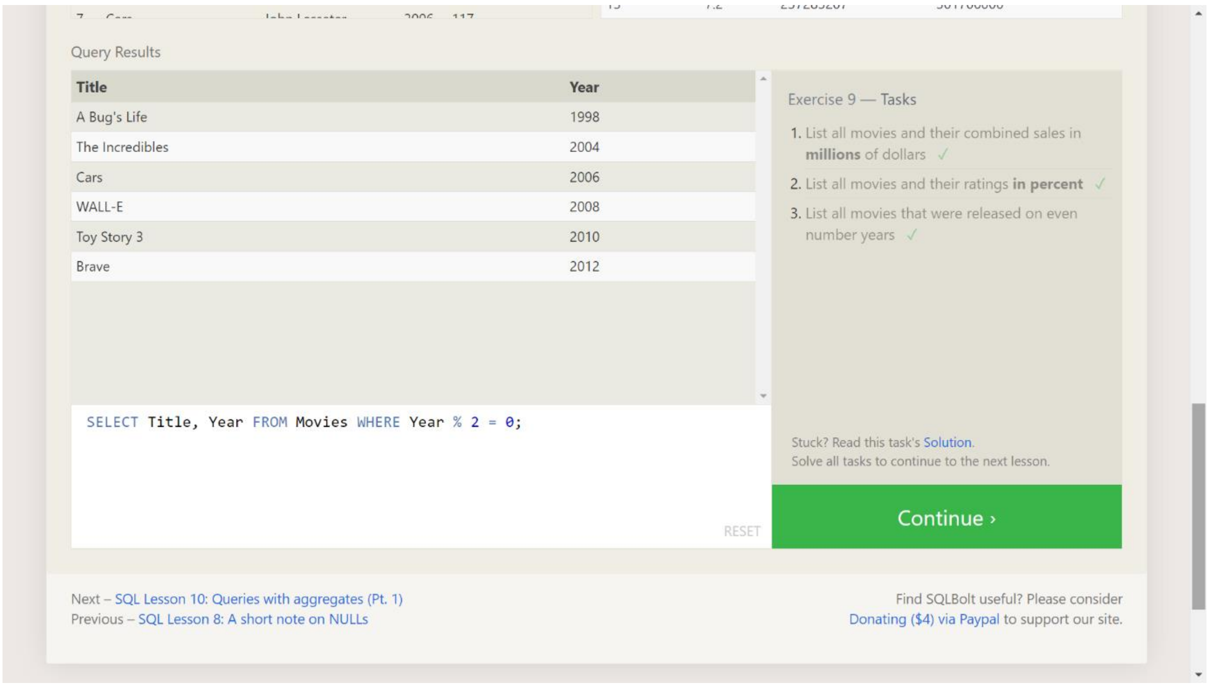
   SELECT title, (domestic_sales + international_sales) / 1000000 AS Gross_sales_millions FROM Movies JOIN Boxoffice ON movies.id = Boxoffice.Movie_id;

2. Find the list of all buildings and their capacity

   SELECT Title, Rating * 10 AS rating_percent FROM Movies JOIN Boxoffice ON Movies.id = Boxoffice.Movie_id;

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

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



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.

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.

Excercise - 10 Tasks

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

   SELECT MAX(years_employed) as Max_years_employed FROM employees;

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

   SELECT Role, AVG(years_employed) as Average_years_employed FROM Employees GROUP BY Role;

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

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

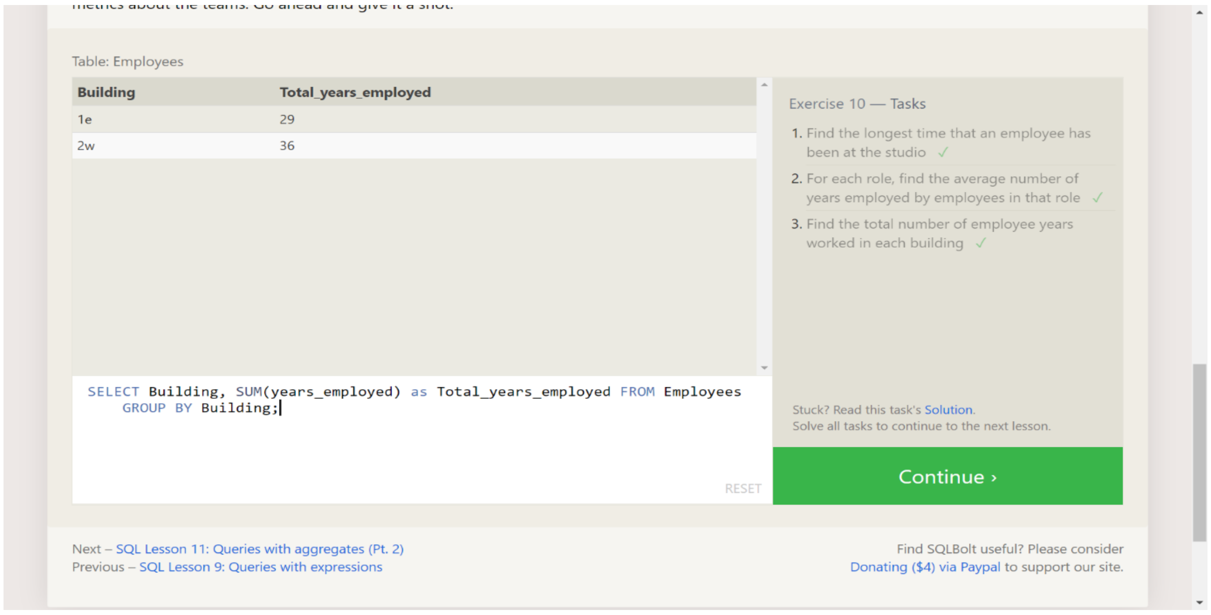

metrics about the teams. Go ahead and give it a shot.

Table: Employees

Building	Total_years_employed
1e	29
2w	36

```
SELECT Building, SUM(years_employed) as Total_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.

<u>**Excercise - 11 Tasks**</u>

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";

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

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

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

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

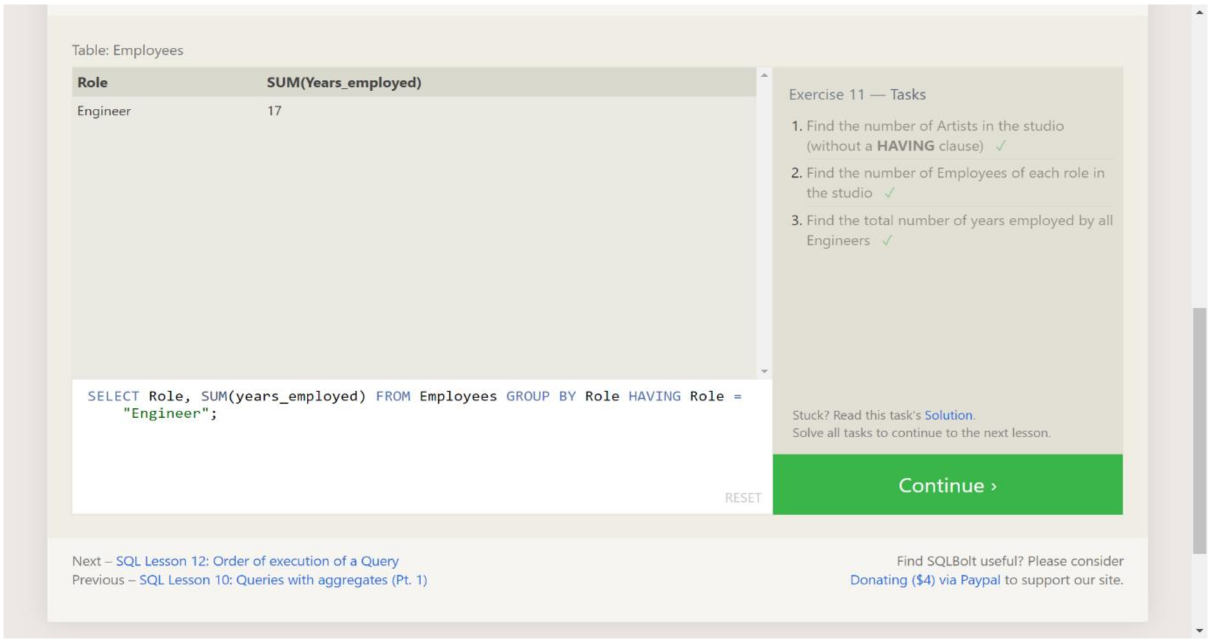


Table: Employees

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 ›

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.

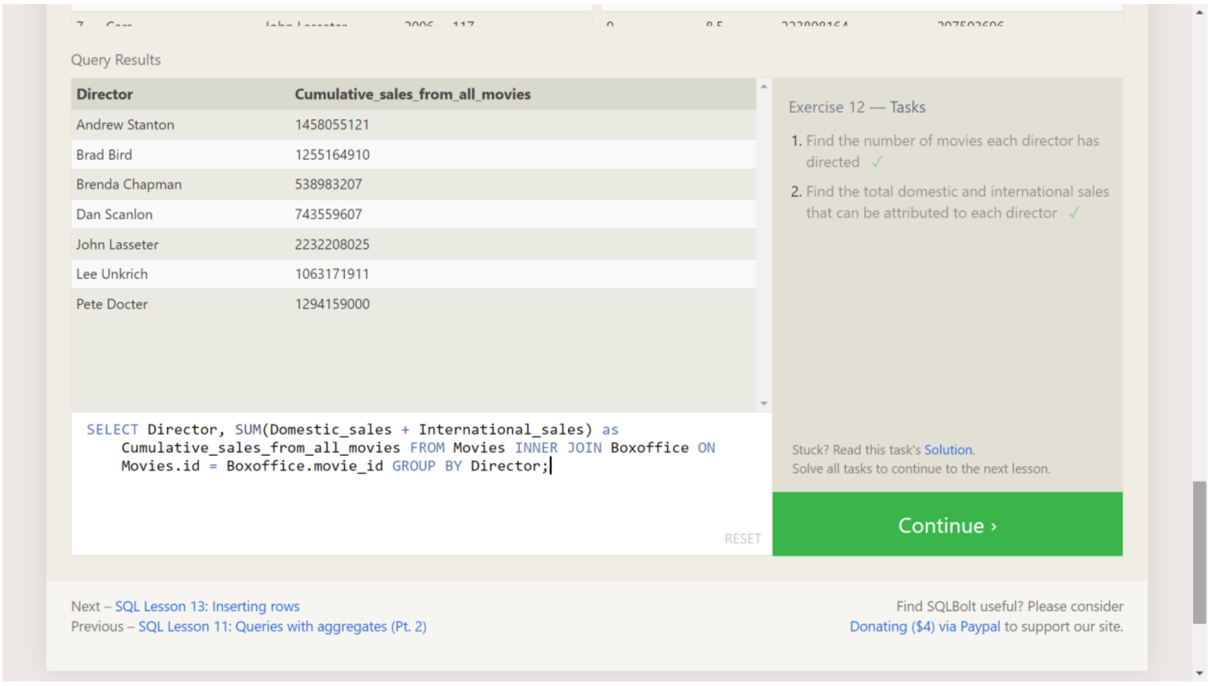
Excercise - 12 Tasks

1. Find the number of movies each director has directed

   SELECT Director, COUNT(id) as Num_movies_directed FROM Movies GROUP BY Director;

2. 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;



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;
```

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.

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.

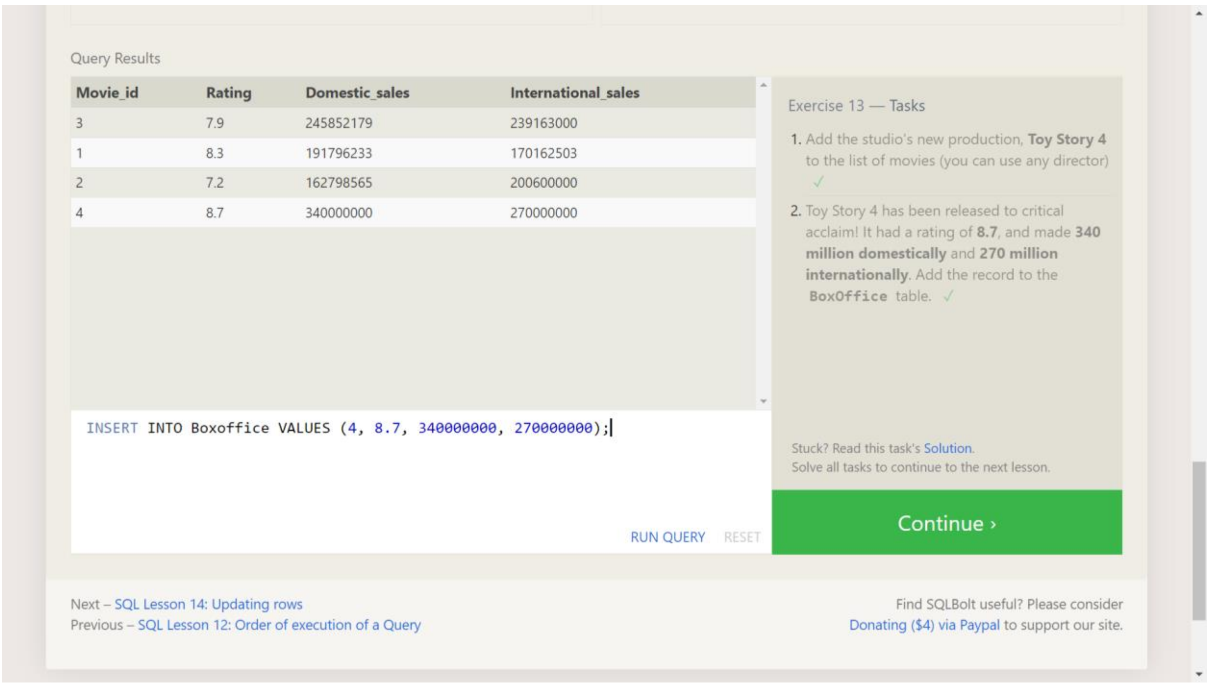
Excercise - 13 Tasks

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

   SELECT Director, COUNT(id) as Num_movies_directed FROM Movies GROUP BY 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.

   INSERT INTO Boxoffice VALUES (4, 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
4	8.7	340000000	270000000

```
INSERT INTO Boxoffice VALUES (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.

Next – SQL Lesson 14: Updating rows
Previous – SQL Lesson 12: Order of execution of a Query

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

<u>Excercise - 14 Tasks</u>

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;

2. The year that Toy Story 2 was released is incorrect, it was actually released in 1999

   UPDATE Movies SET Year = 1999 WHERE Id = 3;

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

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

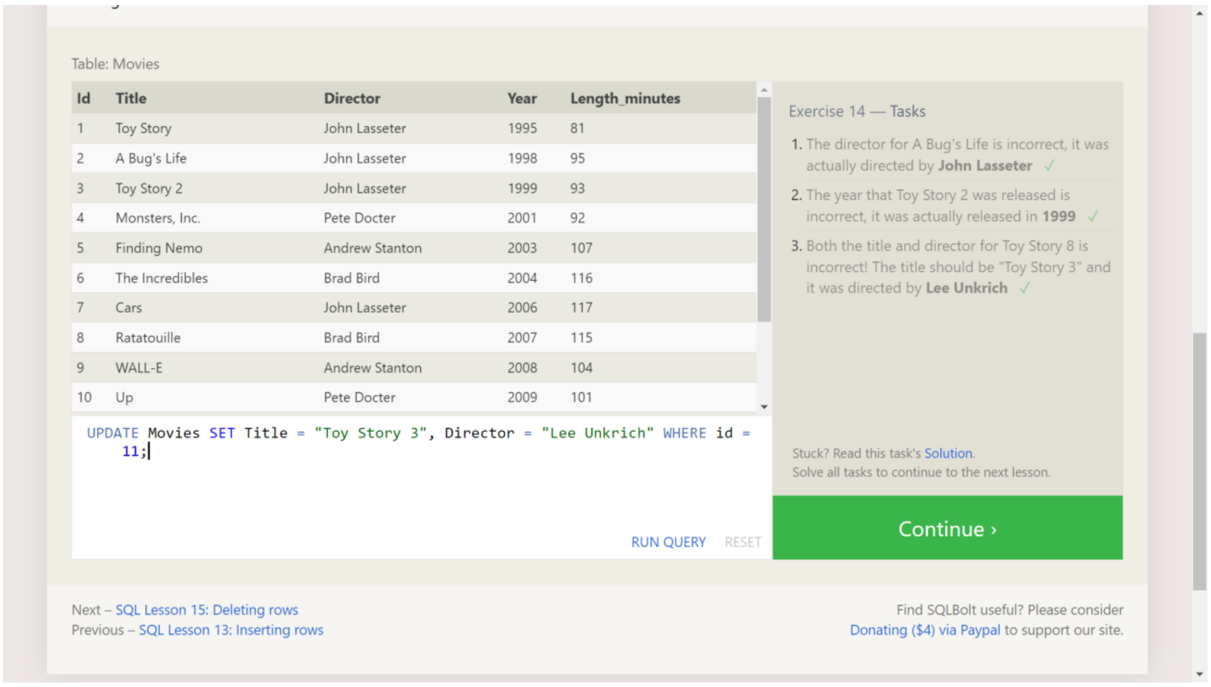


Table: Movies

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

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

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.

Next – SQL Lesson 15: Deleting rows
Previous – SQL Lesson 13: Inserting rows

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

<u>Excercise - 15 Tasks</u>

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

   DELETE FROM Movies where Year < 2005;

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

   DELETE FROM Movies where Director = "Andrew Stanton";

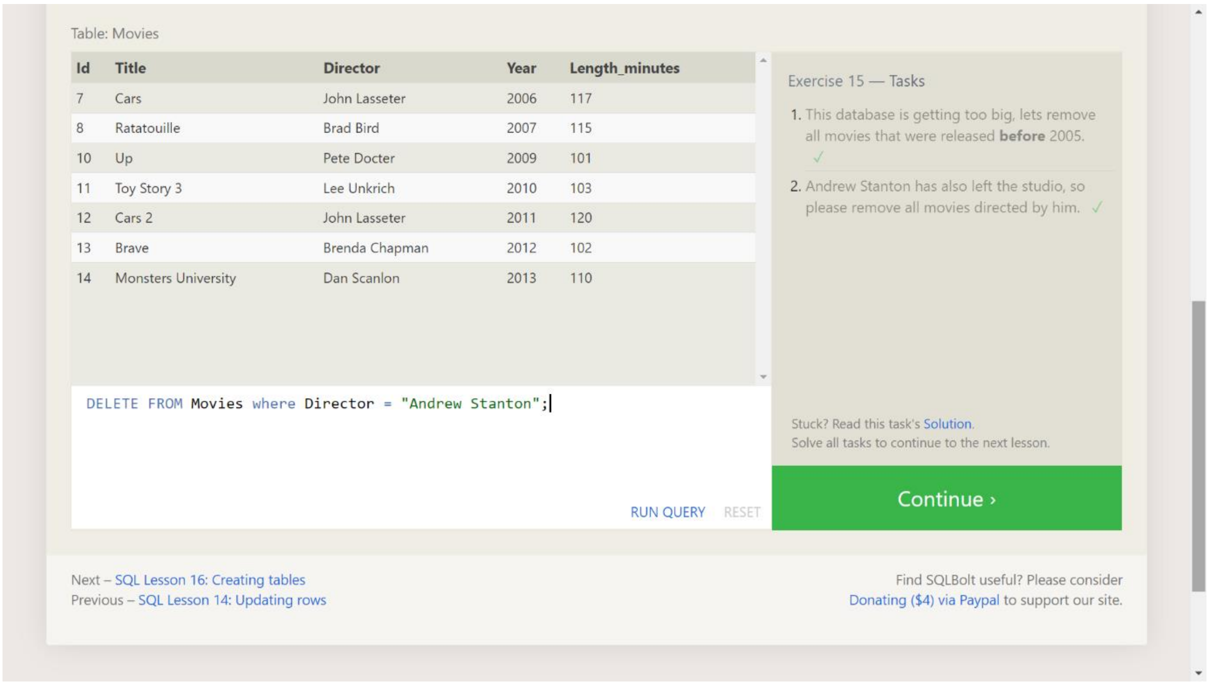


Table: Movies

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

```
DELETE FROM Movies where Director = "Andrew Stanton";
```

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.

Next – SQL Lesson 16: Creating tables
Previous – SQL Lesson 14: Updating rows

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

Excercise - 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);

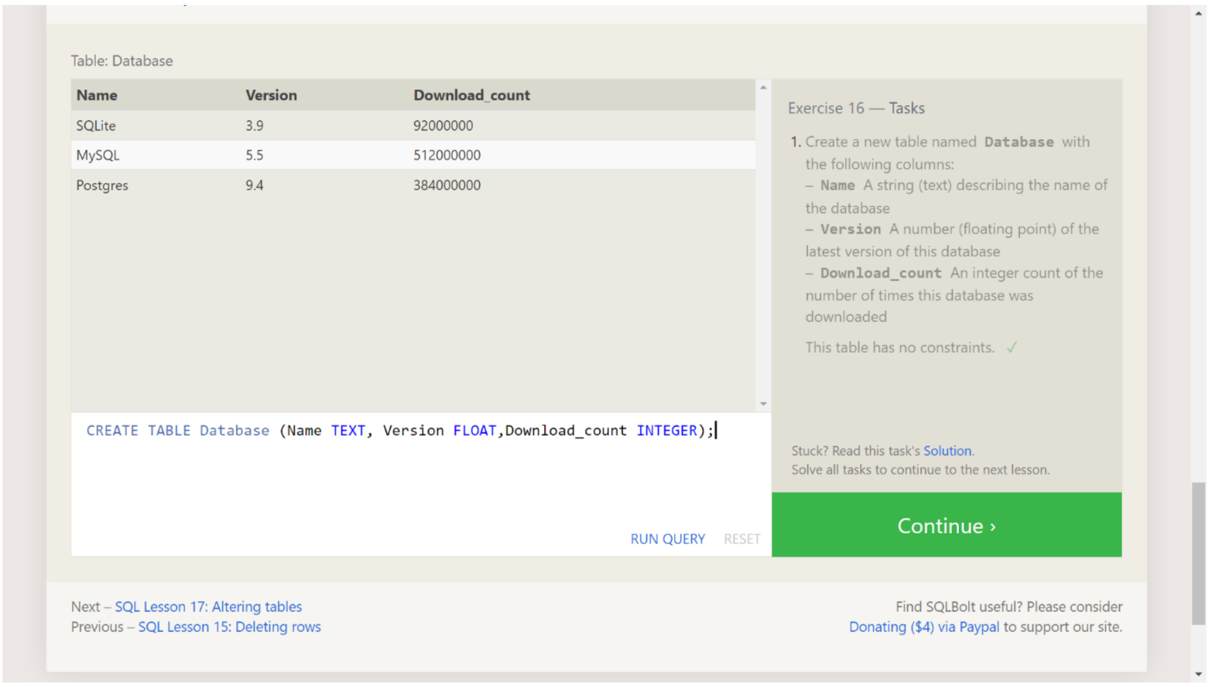


Table: Database

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

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

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.

Next – SQL Lesson 17: Altering tables
Previous – SQL Lesson 15: Deleting rows

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

Excercise - 17 Tasks

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;

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.

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

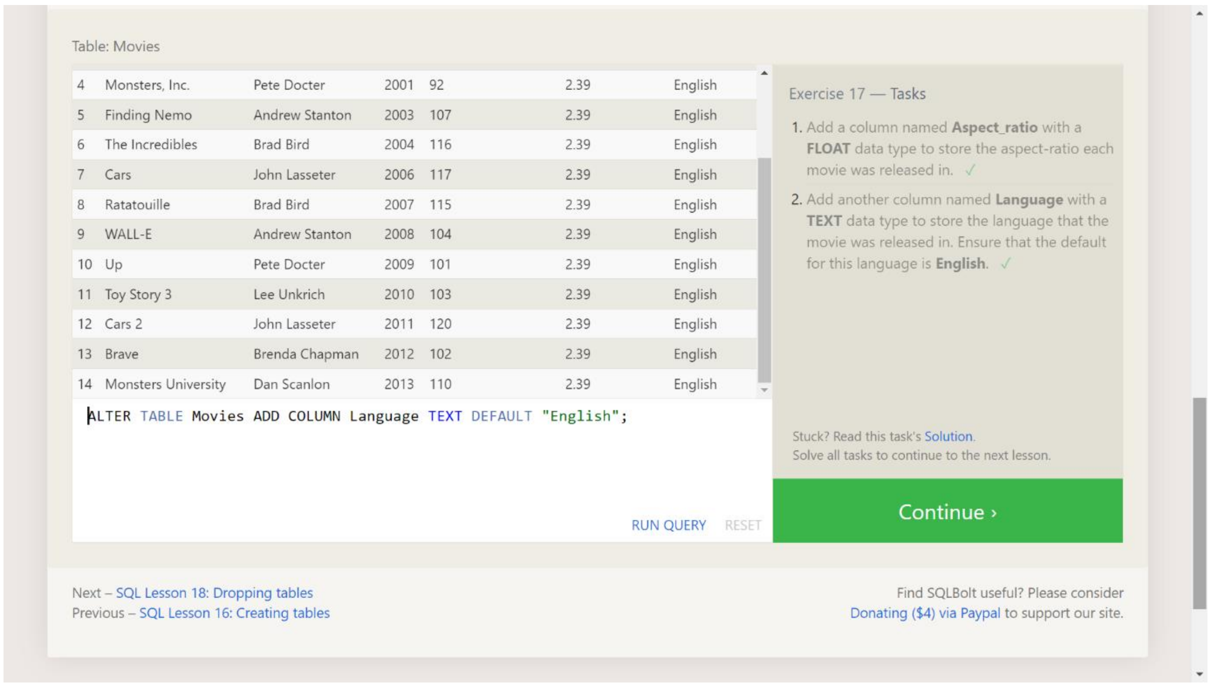


Table: Movies

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
11	Toy Story 3	Lee Unkrich	2010	103	2.39	English
12	Cars 2	John Lasseter	2011	120	2.39	English
13	Brave	Brenda Chapman	2012	102	2.39	English
14	Monsters University	Dan Scanlon	2013	110	2.39	English

```
ALTER TABLE Movies ADD COLUMN Language TEXT DEFAULT "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.

Next – SQL Lesson 18: Dropping tables
Previous – SQL Lesson 16: Creating tables

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

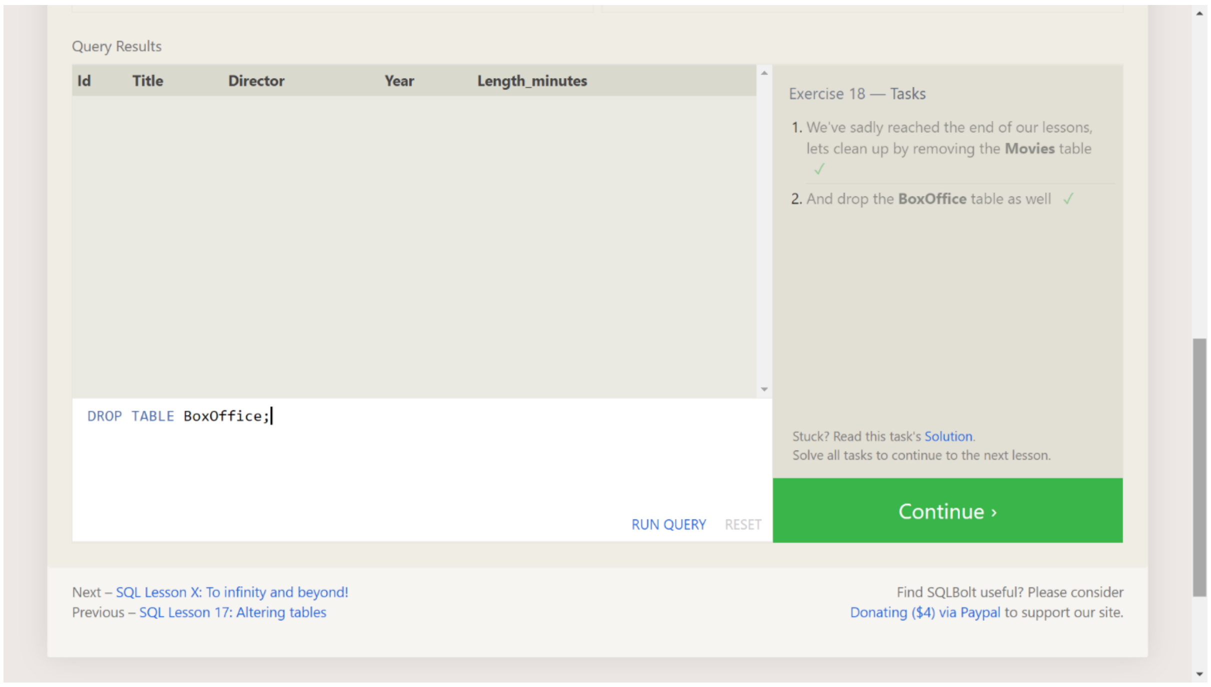
Excercise - 18 Tasks

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

   DROP TABLE Movies;

2. And drop the BoxOffice table as well.

   DROP TABLE BoxOffice;


Query Results

Id	Title	Director	Year	Length_minutes

```
DROP TABLE BoxOffice;
```

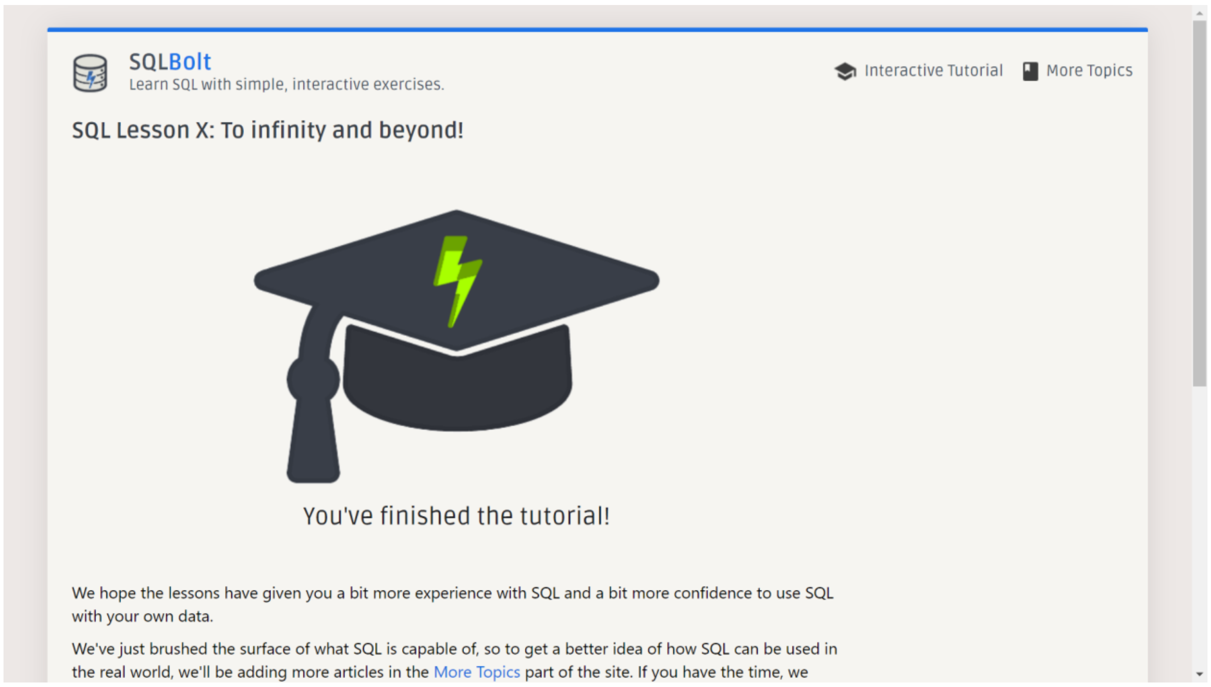
Exercise 18 — Tasks

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.

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.



SQLBolt
Learn SQL with simple, interactive exercises.

Interactive Tutorial
More Topics

SQL Lesson X: To infinity and beyond!

You've finished the tutorial!

We hope the lessons have given you a bit more experience with SQL and a bit more confidence to use SQL with your own data.

We've just brushed the surface of what SQL is capable of, so to get a better idea of how SQL can be used in the real world, we'll be adding more articles in the More Topics part of the site. If you have the time, we