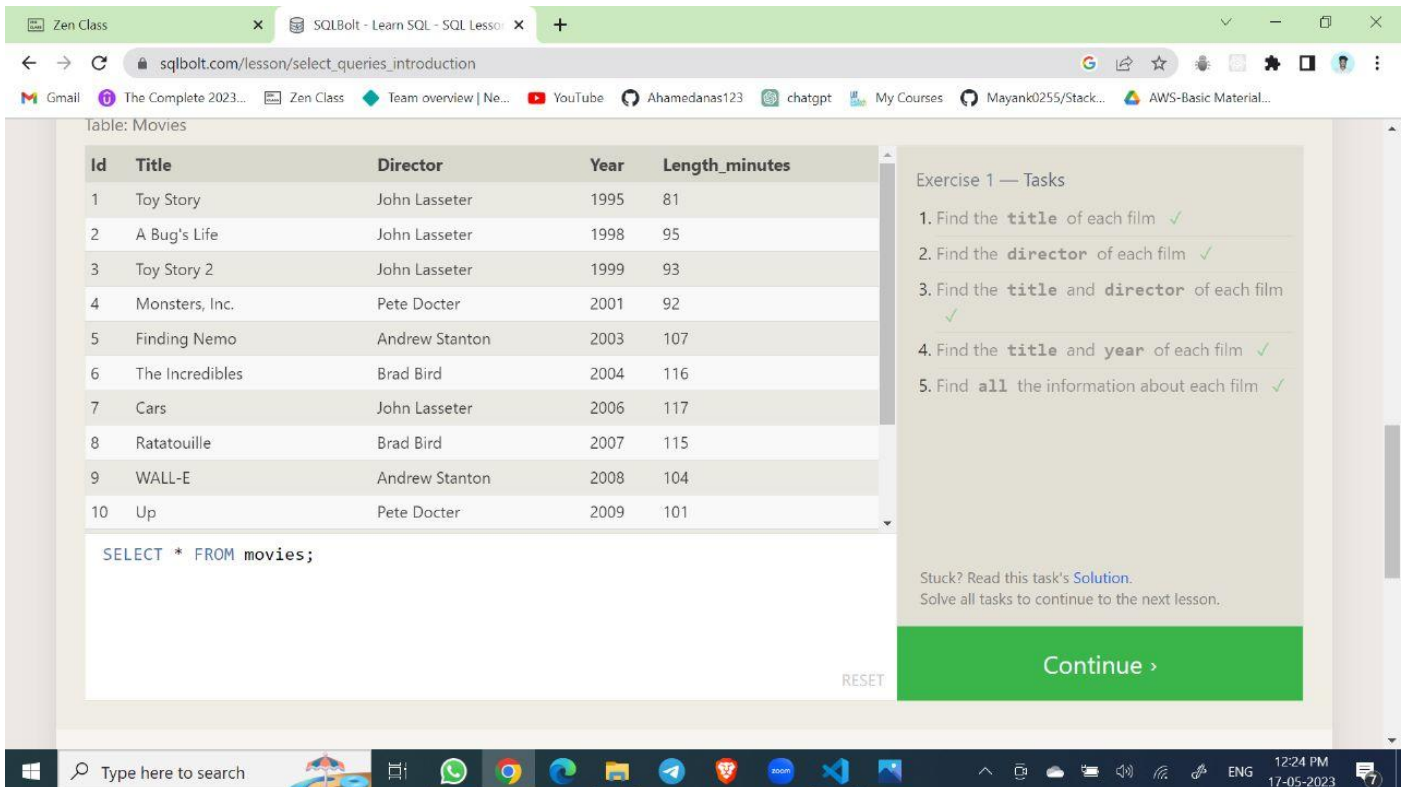


SQLBOLT

SQL Lesson 1: SELECT queries 10



The screenshot shows the SQLBolt website interface. At the top, there's a browser tab for 'SQLBolt - Learn SQL - SQL Lesson 1'. The URL is 'sqlbolt.com/lesson/select_queries_introduction'. Below the browser, there's a table of movies with columns: Id, Title, Director, Year, and Length_minutes. The table contains 10 rows of data. To the right of the table, there's a section titled 'Exercise 1 — Tasks' with five tasks listed. Below the tasks, there's a 'Continue >' button. At the bottom of the page, there's a Windows taskbar with various icons and the system clock showing 12:24 PM on 17-05-2023.

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 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 >](#)

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)

The screenshot shows a web browser window with the URL `sqlbolt.com/lesson/select_queries_with_constraints`. The page displays a table named 'Movies' with two columns: 'Title' and 'Year'. The table contains the following data:

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

Below the table, there is a text input field containing the SQL query:

```
SELECT title, year FROM movies
WHERE year <= 2003;
```

To the right of the table, there is a section titled 'Exercise 2 — Tasks' with four 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` ✓

Below the tasks, there is a green button labeled 'Continue >'. At the bottom of the page, there is a Windows taskbar with various application icons and a system clock showing 12:24 PM on 17-05-2023.

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 title, year FROM movies WHERE year <=2003;

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.

Continue >

1.SELECT * FROM movies where title like '%toy%'

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-_"

SQL Lesson 4: Filtering and sorting Query results

The screenshot shows the SQLBolt lesson interface for 'Lesson 4: Filtering and sorting Query results'. On the left, a table named 'Movies' is displayed with the following data:

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

Below the table, the SQL query is shown:

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

On the right, 'Exercise 4 — Tasks' are listed:

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 ✓

Below the tasks, there is a link to the solution: 'Stuck? Read this task's [Solution](#). Solve all tasks to continue to the next lesson.' A green 'Continue >' button is at the bottom right of the exercise section.

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;

SQL Lesson5 Review: Simple SELECT Queries

The screenshot shows the SQLBolt website interface. At the top, there's a browser tab for 'SQLBolt - Learn SQL - SQL Review'. The URL is 'sqlbolt.com/lesson/select_queries_review'. Below the browser tabs, there's a table titled 'Table: North_american_cities' with two columns: 'City' and 'Population'. The table contains two rows: Chicago with population 2718782 and Houston with population 2195914. Below the table, there's a SQL query editor with the following code:

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

To the right of the table, there's a 'Review 1 — Tasks' section with five tasks, each marked with a green checkmark:

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 ✓

Below the tasks, there's a link to 'Solution' and a note: 'Solve all tasks to continue to the next lesson.' At the bottom right, there's a green 'Continue >' button. The bottom of the screenshot shows a Windows taskbar with various icons and the system clock showing 12:28 PM on 17-05-2023.

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, Latitude FROM north_american_cities WHERE Longitude < -87.629798 ORDER BY Longitude;

3. SELECT city, population FROM north_american_cities WHERE country LIKE "Mexico" ORDER BY population DESC LIMIT 2;

4. 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

The screenshot shows the SQLBolt website interface. On the left, a table titled 'Query Results' displays movie titles and their ratings, sorted in descending order. The table has two columns: 'Title' and 'Rating'. The data rows are as follows:

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

Below the table, the SQL query used to generate the results is shown:

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

On the right side of the interface, 'Exercise 6 — Tasks' are listed:

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 ✓

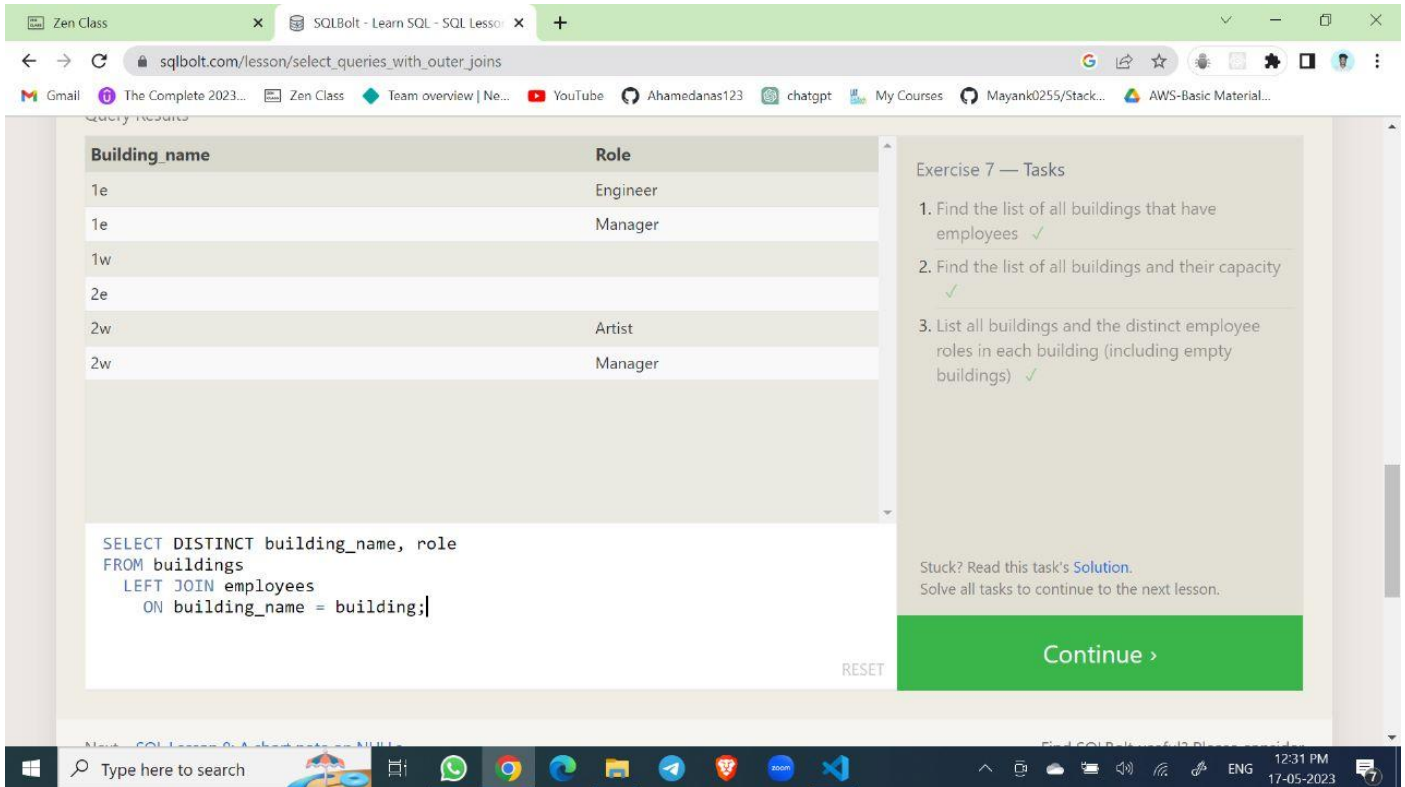
At the bottom of the exercise section, there is a link to the 'Solution' and a green 'Continue >' button.

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 INNER JOIN Boxoffice ON Id = Movie_id WHERE Domestic_sales < International_sales;`

3. `SELECT title, rating FROM movies JOIN boxoffice ON movies.id = boxoffice.movie_id ORDER BY rating DESC;`

SQL Lesson 7: OUTER JOINS



The screenshot shows the SQLBolt website interface. At the top, there's a browser tab for 'SQLBolt - Learn SQL - SQL Lesson 7'. The address bar shows 'sqlbolt.com/lesson/select_queries_with_outer_joins'. Below the browser, there's a table of query results and a sidebar with exercise tasks.

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.

Continue >

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;

SQL Lesson 8: A short note on NULLs

The screenshot shows a web browser window with the URL `sqlbolt.com/lesson/select_queries_with_nulls`. The page displays a SQL query and its results. The query is:

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

The query results show a table with the header **Building_name** and two rows: `1w` and `2e`.

On the right side of the page, there is a section titled "Exercise 8 — Tasks" with two tasks:

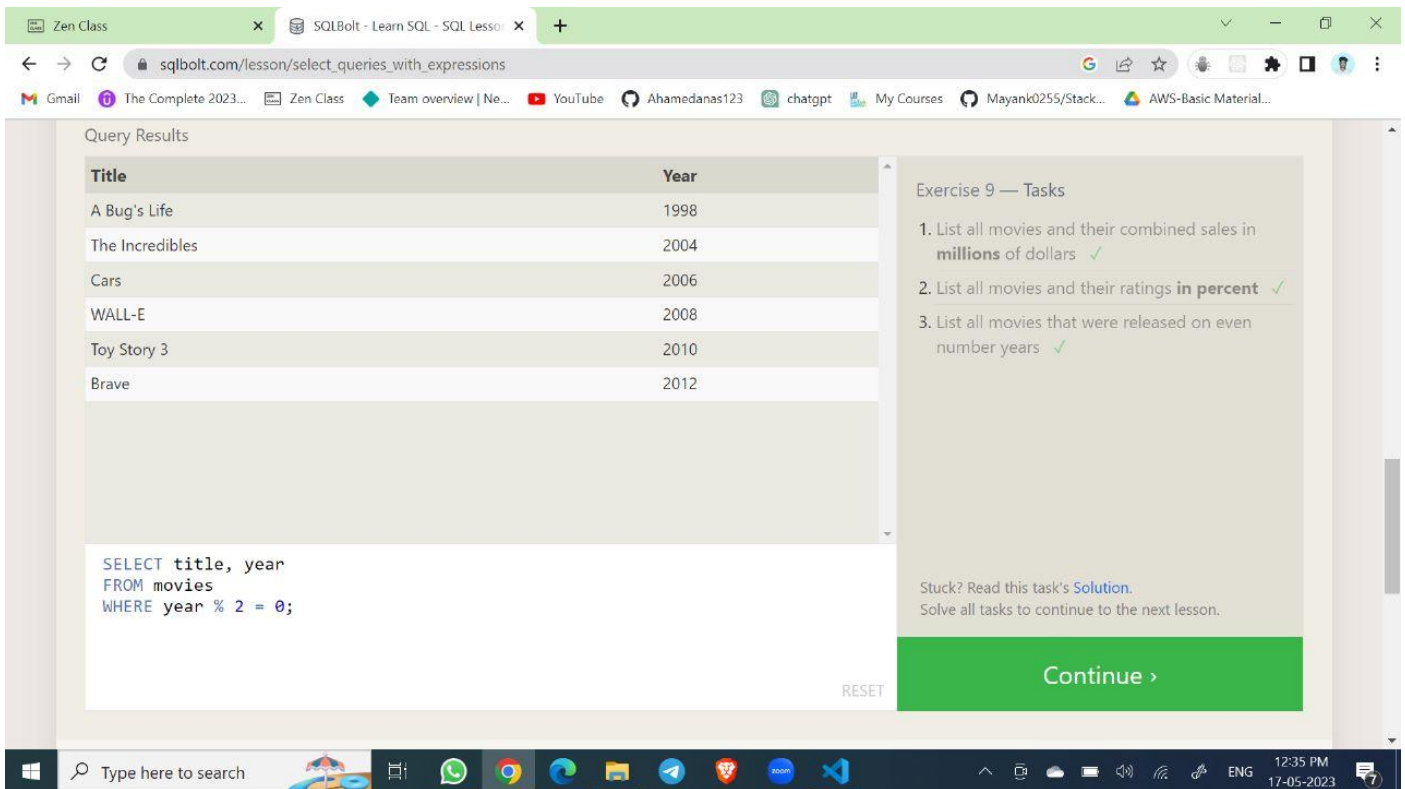
1. Find the name and role of all employees who have not been assigned to a building ✓
2. Find the names of the buildings that hold no employees ✓

Below the tasks, there is a link to the solution: "Stuck? Read this task's [Solution](#)." and a green button labeled "Continue >".

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;

SQL Lesson 9: Queries with expressions



The screenshot shows the SQLBolt website interface. At the top, there's a browser tab for 'sqlbolt.com/lesson/select_queries_with_expressions'. Below the browser, a table titled 'Query Results' displays movie data. The table has two columns: 'Title' and 'Year'. The data rows are: A Bug's Life (1998), The Incredibles (2004), Cars (2006), WALL-E (2008), Toy Story 3 (2010), and Brave (2012). Below the table, a SQL query is shown:

```
SELECT title, year
FROM movies
WHERE year % 2 = 0;
```

 To the right of the table, there's a section titled 'Exercise 9 — Tasks' with three tasks: 1. List all movies and their combined sales in millions of dollars (checked), 2. List all movies and their ratings in percent (checked), and 3. List all movies that were released on even number years (checked). Below the tasks, there's a 'Continue >' button. The bottom of the screenshot shows a Windows taskbar with various application icons and a system clock showing 12:35 PM on 17-05-2023.

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.

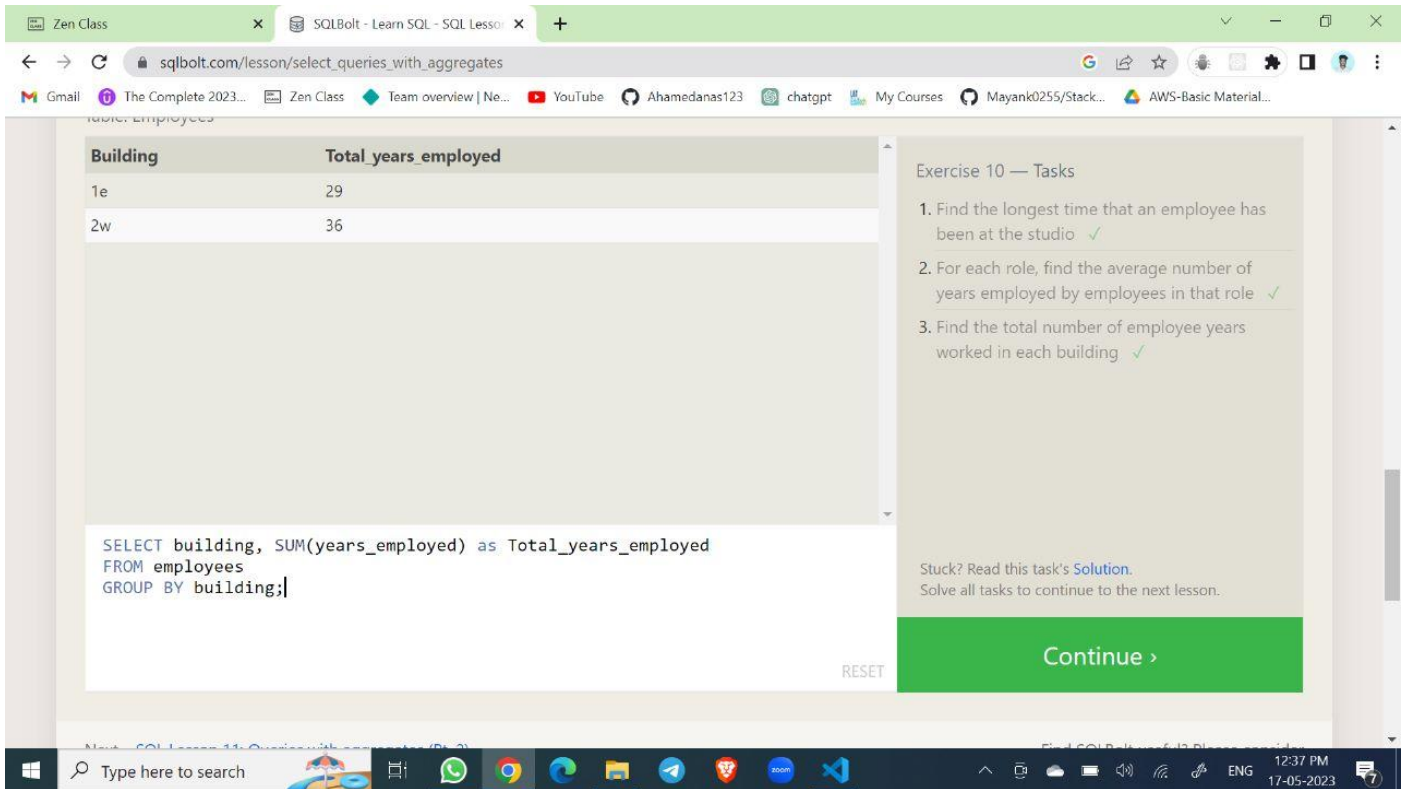
[Continue >](#)

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;`

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



The screenshot shows the SQLBolt website interface. At the top, there's a browser tab for 'SQLBolt - Learn SQL - SQL Lesson 10'. The address bar shows 'sqlbolt.com/lesson/select_queries_with_aggregates'. Below the browser, there's a table with the following data:

Building	Total_years_employed
1e	29
2w	36

Below the table, there's a SQL query editor with the following code:

```
SELECT building, SUM(years_employed) as Total_years_employed
FROM employees
GROUP BY building;
```

To the right of the query editor, there's a section titled 'Exercise 10 — Tasks' with three 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 ✓

Below the tasks, there's a link 'Stuck? Read this task's Solution.' and a button 'Continue >'. At the bottom of the page, there's a Windows taskbar with various icons and the system clock showing '12:37 PM 17-05-2023'.

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)

The screenshot shows the SQLBolt interface for Lesson 11. On the left, a table displays the result of a query:

Role	SUM(Years_employed)
Engineer	17

Below the table, the SQL query is shown:

```
SELECT role, SUM(years_employed)
FROM employees
GROUP BY role
HAVING role = "Engineer";
```

On the right, the 'Exercise 11 — Tasks' section lists three 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 ✓

At the bottom right, there is a green 'Continue >' button. The bottom of the screen shows a Windows taskbar with various application icons and the system clock indicating 12:39 PM on 17-05-2023.

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";`

SQL Lesson 12: Order of execution of a Query

The screenshot shows a web browser window with the URL `sqlbolt.com/lesson/select_queries_order_of_execution`. The page displays the results of a SQL query and a list of tasks for Exercise 12.

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

SQL Query:

```
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 ✓

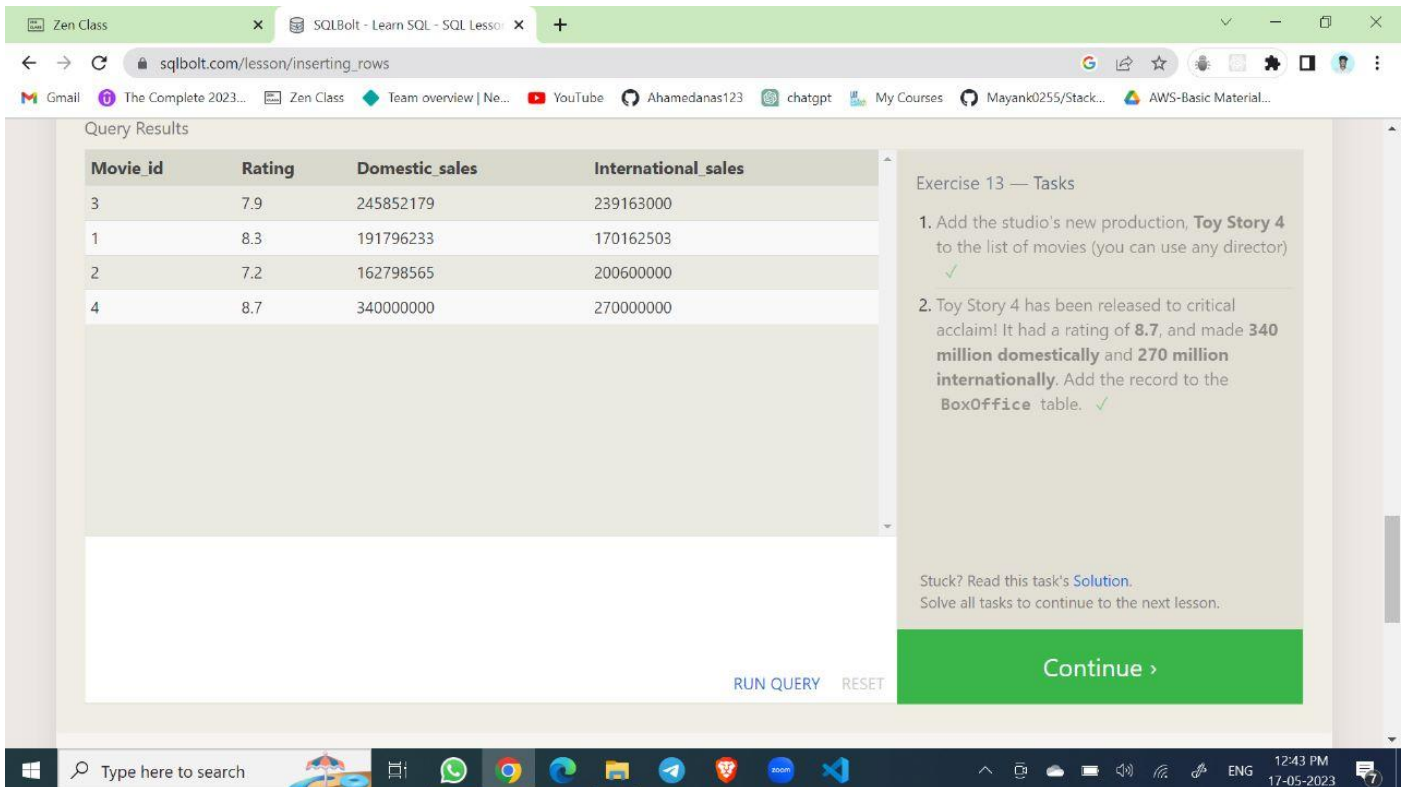
Stuck? Read this task's [Solution](#).
Solve all tasks to continue to the next lesson.

[Continue >](#)

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



The screenshot shows the SQLBolt website interface. On the left, a table titled 'Query Results' displays data for movies. The table has four columns: 'Movie_id', 'Rating', 'Domestic_sales', and 'International_sales'. The data rows are as follows:

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

Below the table, there is a 'RUN QUERY' button and a 'RESET' button. On the right side, there is a section titled 'Exercise 13 — Tasks' with two tasks:

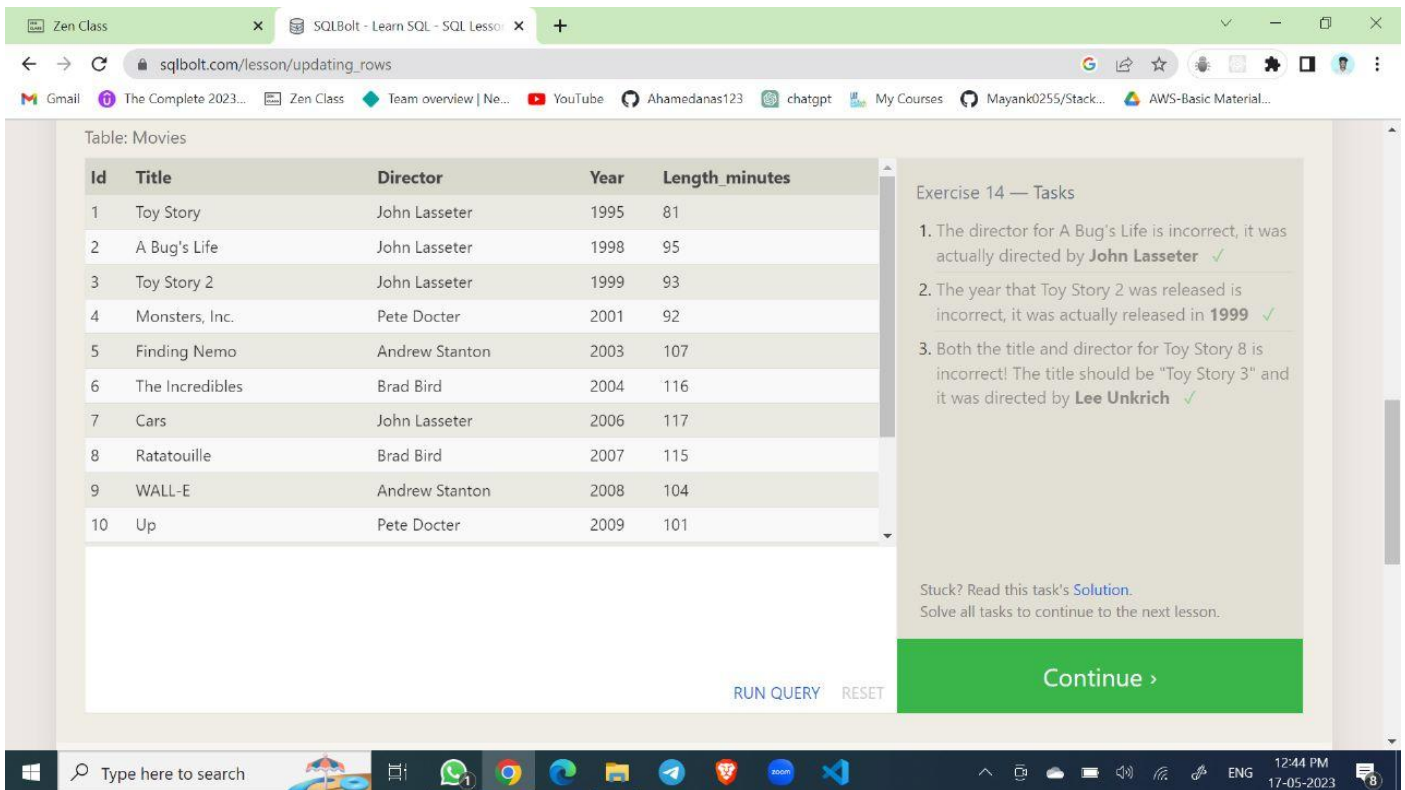
1. Add the studio's new production, **Toy Story 4** to the list of movies (you can use any director) ✓
2. Toy Story 4 has been released to critical acclaim! It had a rating of **8.7**, and made **340 million domestically** and **270 million internationally**. Add the record to the **BoxOffice** table.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

Below the tasks, there is a link to 'Solution' and a 'Continue >' button.

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



The screenshot shows a web browser window with the URL `sqlbolt.com/lesson/updating_rows`. The page displays a table named 'Movies' with the following data:

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

To the right of the table, there are three tasks for Exercise 14:

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** ✓

Below the tasks, there is a green button labeled 'Continue >'. At the bottom of the page, there is a Windows taskbar with various application icons and a system clock showing 12:44 PM on 17-05-2023.

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;

SQL Lesson 15: Deleting rows

The screenshot shows a web browser window with the URL `sqlbolt.com/lesson/deleting_rows`. The page displays a table named 'Movies' with the following data:

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

To the right of the table, there is a section titled 'Exercise 15 — Tasks' with two tasks:

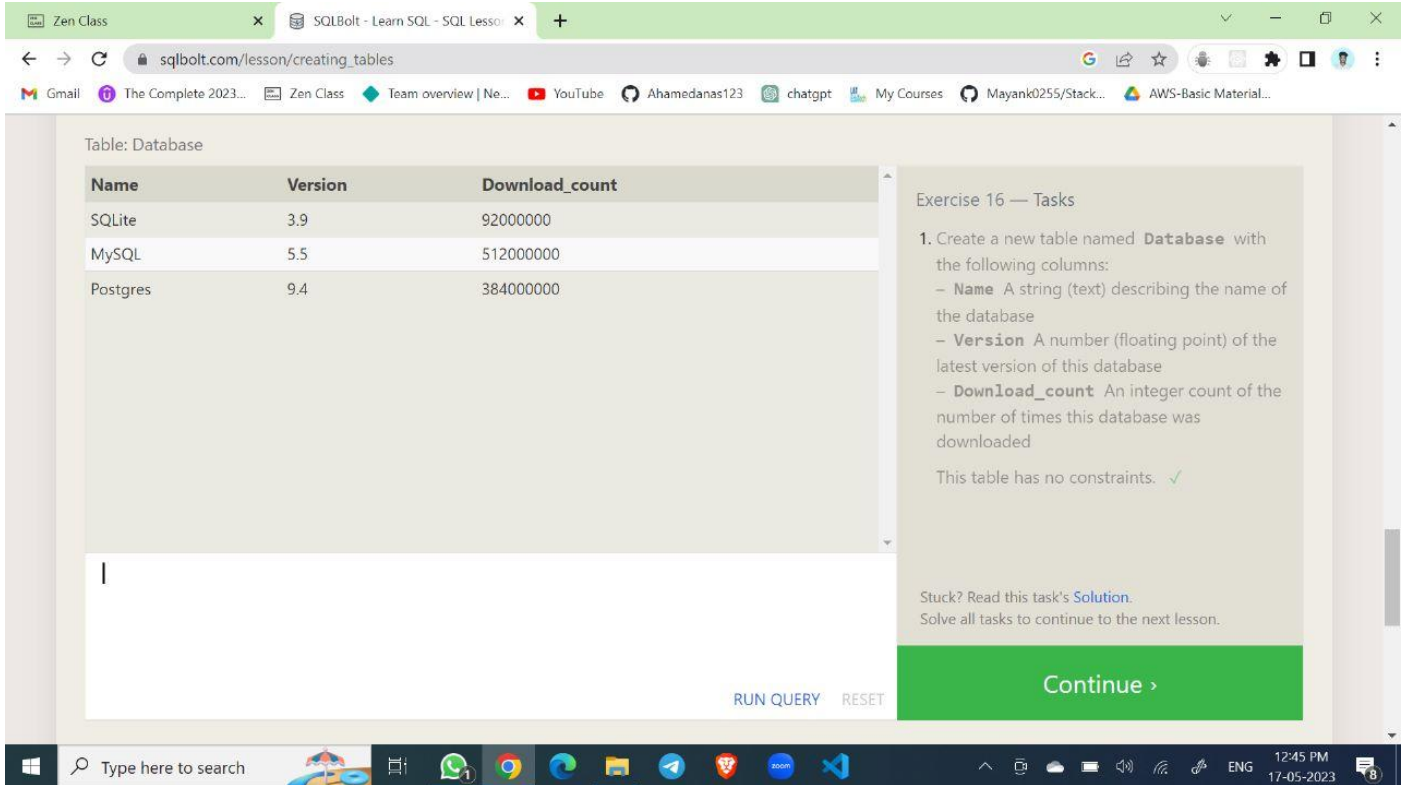
1. This database is getting too big, lets remove all movies that were released **before** 2005. ✓
2. Andrew Stanton has also left the studio, so please remove all movies directed by him. ✓

Below the tasks, there is a link to 'Solution' and a 'Continue >' button. At the bottom of the page, there is a 'RUN QUERY' button and a 'RESET' button.

1. DELETE FROM movies where year < 2005;

2. DELETE FROM movies where director = "Andrew Stanton";

SQL Lesson 16: Creating tables



The screenshot shows a web browser window with the URL `sqlbolt.com/lesson/creating_tables`. The page displays a table named "Table: Database" with the following data:

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

Below the table is a text input field with a cursor. To the right, the "Exercise 16 — Tasks" section contains the following instructions:

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.

At the bottom right, there is a green "Continue >" button. The browser's taskbar at the bottom shows various application icons and the system clock indicating 12:45 PM on 17-05-2023.

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

SQL Lesson 17: Altering tables

The screenshot shows a web browser window with the URL `sqlbolt.com/lesson/altering_tables`. The page displays a table named 'Table: Movies' with the following data:

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
	New column added	Pete Docter	2009	101	2.39	English

Below the table, the SQL command is shown:

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

Buttons for 'RUN QUERY' and 'RESET' are visible. To the right, 'Exercise 17 — Tasks' lists two 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**. ✓

A green 'Continue >' button is at the bottom right of the exercise section.

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

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

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.

[Continue >](#)


RUN QUERY RESET

1. DROP TABLE Movies;

2. DROP TABLE BoxOffice;

SQLBolt
Learn SQL with simple, interactive exercises.

SQL Lesson X: To infinity and beyond!



You've finished the tutorial!

Interactive Tutorial More Topics