**SQL Lesson 1: SELECT queries 101**

<https://sqlbolt.com/lesson/select_queries_introduction>

Exercise 1 — Tasks

1. Find the title of each film
   1. SELECT title FROM movies;
2. Find the director of each film
   1. SELECT director FROM movies;
3. Find the title and director of each film
   1. SELECT title, director FROM movies;
4. Find the title and year of each film
   1. SELECT title, year FROM movies;
5. Find all the information about each film
   1. SELECT \* FROM movies;

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

<https://sqlbolt.com/lesson/select_queries_with_constraints>

Exercise 2 — Tasks

1. Find the movie with a row id of 6
   1. SELECT \* FROM movies

where id = 6;

1. Find the movies released in the years between 2000 and 2010
   1. SELECT \* FROM movies

where year BETWEEN 2000 and 2010;

1. Find the movies not released in the years between 2000 and 2010
   1. SELECT \* FROM movies

where year NOT BETWEEN 2000 and 2010;

1. Find the first 5 Pixar movies and their release year
   1. SELECT \* FROM movies

ORDER BY year

LIMIT 5;

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

<https://sqlbolt.com/lesson/select_queries_with_constraints_pt_2>

Exercise 3 — Tasks

1. Find all the Toy Story movies
   1. SELECT \* FROM movies

WHERE Title LIKE '%Toy Story%';

1. Find all the movies directed by John Lasseter
   1. SELECT \* FROM movies

WHERE Director = 'John Lasseter';

1. Find all the movies (and director) not directed by John Lasseter
   1. SELECT \* FROM movies

WHERE Director NOT LIKE 'John Lasseter';

1. Find all the WALL-\* movies
   1. SELECT \* FROM movies

WHERE TITLE LIKE '%Wall%';

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

<https://sqlbolt.com/lesson/filtering_sorting_query_results>

Exercise 4 — Tasks

1. List all directors of Pixar movies (alphabetically), without duplicates
   1. SELECT Distinct(DIRECTOR) FROM movies

ORDER BY DIRECTOR ASC;

1. List the last four Pixar movies released (ordered from most recent to least)
   1. SELECT Distinct(Title) FROM movies

ORDER BY YEAR DESC

LIMIT 4;

1. List the first five Pixar movies sorted alphabetically
   1. SELECT Distinct(Title) FROM movies

ORDER BY TITLE ASC

LIMIT 5;

1. List the next five Pixar movies sorted alphabetically
   1. SELECT Distinct(Title) FROM movies

ORDER BY TITLE ASC

LIMIT 5 OFFSET 5;

**SQL Review: Simple SELECT Queries**

<https://sqlbolt.com/lesson/select_queries_review>

Review 1 — Tasks

1. List all the Canadian cities and their populations
   1. SELECT \* FROM north\_american\_cities

WHERE COUNTRY = 'Canada';

1. Order all the cities in the United States by their latitude from north to south
   1. SELECT \* FROM north\_american\_cities

WHERE COUNTRY = 'United States'

ORDER BY LATITUDE DESC;

1. List all the cities west of Chicago, ordered from west to east
   1. SELECT \* FROM north\_american\_cities

WHERE LONGITUDE < -87.629798

ORDER BY LONGITUDE ASC;

1. List the two largest cities in Mexico (by population)
   1. SELECT \* FROM north\_american\_cities

WHERE COUNTRY = 'Mexico'

ORDER BY POPULATION DESC

LIMIT 2;

1. List the third and fourth largest cities (by population) in the United States and their population
   1. SELECT \* FROM north\_american\_cities

WHERE COUNTRY = 'United States'

ORDER BY POPULATION DESC

LIMIT 2 OFFSET 2;

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

<https://sqlbolt.com/lesson/select_queries_with_joins>

Exercise 6 — Tasks

1. Find the domestic and international sales for each movie
   1. SELECT \* FROM movies

LEFT JOIN Boxoffice

on movies.id = boxoffice.movie\_id;

1. Show the sales numbers for each movie that did better internationally rather than domestically
   1. SELECT \* FROM movies

LEFT JOIN Boxoffice

on movies.id = boxoffice.movie\_id

WHERE International\_sales > Domestic\_sales;

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

SELECT \* FROM movies

JOIN Boxoffice

on movies.id = boxoffice.movie\_id

ORDER BY RATING DESC;

**SQL Lesson 7: OUTER JOINs**

<https://sqlbolt.com/lesson/select_queries_with_outer_joins>

Exercise 7 — Tasks

1. Find the list of all buildings that have employees
   1. SELECT DISTINCT building FROM employees;
2. Find the list of all buildings and their capacity
   1. SELECT \* FROM buildings;
3. List all buildings and the distinct employee roles in each building (including empty buildings)
   1. SELECT distinct building\_name, Role FROM Buildings

Left JOIN Employees

on Buildings.Building\_name = Employees.Building;

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

<https://sqlbolt.com/lesson/select_queries_with_nulls>

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

WHERE Building is Null;

1. Find the names of the buildings that hold no employees
   1. SELECT distinct building\_name

FROM Buildings

LEFT JOIN employees

on Buildings.Building\_name = Employees.Building

WHERE role is NULL;

**SQL Lesson 9: Queries with expressions**

<https://sqlbolt.com/lesson/select_queries_with_expressions>

Exercise 9 — Tasks

1. List all movies and their combined sales in millions of dollars
   1. SELECT Title, (Domestic\_sales + International\_sales) / 1000000 FROM movies

Left Join Boxoffice

on Movies.ID = Boxoffice.Movie\_id;

1. List all movies and their ratings in percent
   1. SELECT Title, Rating \*10 FROM movies

Left Join Boxoffice

on Movies.ID = Boxoffice.Movie\_id;

1. List all movies that were released on even number years
   1. SELECT Title, Year FROM movies

WHERE Year % 2= 0;

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

<https://sqlbolt.com/lesson/select_queries_with_aggregates>

Exercise 10 — Tasks

1. Find the longest time that an employee has been at the studio
   1. SELECT max(Years\_employed) FROM employees;
2. For each role, find the average number of years employed by employees in that role
   1. SELECT role, avg(Years\_employed) FROM employees

GROUP BY role;

1. Find the total number of employee years worked in each building
   1. SELECT Building, sum(Years\_employed) FROM employees

GROUP BY Building;

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

<https://sqlbolt.com/lesson/select_queries_with_aggregates_pt_2>

Exercise 11 — Tasks

1. Find the number of Artists in the studio (without a HAVING clause)
   1. SELECT Role, count(role) FROM employees

WHERE role = "Artist";

1. Find the number of Employees of each role in the studio
   1. SELECT Role, count(role) AS Employee\_Count FROM employees

GROUP BY role;

1. Find the total number of years employed by all Engineers
   1. SELECT Role, sum(Years\_employed) AS Total\_Years FROM employees

WHERE role = "Engineer";

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

<https://sqlbolt.com/lesson/select_queries_order_of_execution>

Exercise 12 — Tasks

1. Find the number of movies each director has directed
   1. SELECT Director, COUNT(title) AS Movies\_Directed FROM movies

GROUP BY Director;

1. Find the total domestic and international sales that can be attributed to each director
   1. SELECT Director, sum(Domestic\_Sales + International\_Sales) as TOTAL\_SALES FROM movies

LEFT JOIN Boxoffice

on Movies.ID = Boxoffice.Movie\_ID

Group by Director;

**SQL Lesson 13: Inserting rows**

<https://sqlbolt.com/lesson/inserting_rows>

Exercise 13 — Tasks

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

VALUES (4, "Toy Story 4", "John Lasseter", 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.
   1. INSERT INTO Boxoffice

VALUES (4, 8.7, 340000000, 270000000);

**SQL Lesson 14: Updating rows**

<https://sqlbolt.com/lesson/updating_rows>

Exercise 14 — Tasks

1. The director for A Bug's Life is incorrect, it was actually directed by John Lasseter
   1. Update Movies

Set Director = "John Lasseter"

where Title = "A Bug's Life";

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

Set Year = 1999

where id = 3;

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

Set Title = "Toy Story 3",

Director = "Lee Unkrich"

Where id = 11;

**SQL Lesson 15: Deleting rows**

Exercise 15 — Tasks

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

where year < 2005;

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

where Director = "Andrew Stanton";

**SQL Lesson 16: Creating tables**

<https://sqlbolt.com/lesson/creating_tables>

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

This table has no constraints.

* 1. CREATE TABLE database(

Name TEXT,

Version INTEGER,

Download\_count INTEGER

);

**SQL Lesson 17: Altering tables**

<https://sqlbolt.com/lesson/altering_tables>

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.
   1. Alter Table movies

ADD Aspect\_ratio FLOAT;

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.
   1. Alter Table movies

ADD Language TEXT

DEFAULT English;

**SQL Lesson 18: Dropping tables**

<https://sqlbolt.com/lesson/dropping_tables>

Exercise 18 — Tasks

1. We've sadly reached the end of our lessons, lets clean up by removing the Movies table
   1. DROP TABLE IF EXISTS MOVIES;
2. And drop the BoxOffice table as well
   1. DROP TABLE IF EXISTS BoxOffice;