***Task resource:*** [***https://sqlbolt.com/***](https://sqlbolt.com/)

**Exercise 1 — Tasks: SELECT**

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 ✓

**Answers*:***

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;

**Exercise 2 — Tasks: Queries with constraints**

1. Find the movie with a row id of 6 ✓
2. Find the movies released in the years between 2000 and 2010 ✓
3. Find the movies not released in the years between 2000 and 2010 ✓
4. Find the first 5 Pixar movies and their release year ✓

**Answers*:***

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 \* FROM movies LIMIT 5;

**Exercise 3 — Tasks: Queries with constraints**

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 ✓

**Answers**:

1. SELECT \* FROM movies WHERE title LIKE "Toy Story%";
2. SELECT \* FROM movies WHERE director="John Lasseter";
3. SELECT \* FROM movies WHERE director!="John Lasseter";
4. SELECT \* FROM movies WHERE title LIKE "WALL-%";

**Exercise 4 — Tasks: Filtering and sorting query results**

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 ✓

**Answers**:

1. SELECT DISTINCT director FROM movies ORDER BY director ASC;
2. SELECT \* FROM movies ORDER BY year DESC LIMIT 4;
3. SELECT \* FROM movies ORDER BY title ASC LIMIT 5;
4. SELECT \* FROM movies ORDER BY title ASC LIMIT 5 OFFSET 5;

**Review 1 — Tasks: Simple Select Queries**

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 ✓

**Answers**:

1. SELECT \* FROM north\_american\_cities WHERE country="Canada";
2. SELECT \* FROM north\_american\_cities WHERE country="United States" ORDER BY latitude DESC;
3. SELECT \* FROM north\_american\_cities WHERE longitude < -87.629798 ORDER BY longitude;
4. SELECT \* FROM north\_american\_cities WHERE country="Mexico" ORDER BY population DESC LIMIT 2;
5. SELECT \* FROM north\_american\_cities WHERE country="United States" ORDER BY population DESC LIMIT 2 OFFSET 2;

**Exercise 6 — Tasks:Multi-talble queries with JOIN’s**

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 ✓

**Answers**:

1. SELECT \* FROM movies INNER JOIN boxoffice ON movies.id = boxoffice.movie\_id;
2. SELECT \* FROM movies INNER JOIN boxoffice ON movies.id = boxoffice.movie\_id WHERE domestic\_sales < international\_sales GROUP BY international\_sales;
3. SELECT id, title, rating FROM movies INNER JOIN boxoffice ON movies.id = boxoffice.movie\_id ORDER BY rating DESC;

**Exercise 7 — Tasks: Outer JOIN’s**

1. Find the list of all buildings that have employees ✓
2. Find the list of all buildings and their capacity ✓List all buildings and the distinct employee roles in each building (including empty buildings) ✓
3. List all buildings and the distinct employee roles in each building (including empty buildings) ✓

**Answers**:

1. SELECT DISTINCT building FROM employees;
2. SELECT \* FROM buildings;
3. SELECT DISTINCT building\_name, role FROM buildings LEFT JOIN employees ON buildings.building\_name = employees.building;

**Exercise 8 — Tasks: NULL**

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 ✓

**Answers**:

1. SELECT name, role FROM employees WHERE building IS NULL;
2. SELECT building\_name FROM buildings LEFT JOIN employees ON buildings.building\_name = employees.building WHERE building IS NULL;

**Exercise 9 — Tasks: Queries with expressions**

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 ✓

**Answers**:

1. SELECT id, title, (domestic\_sales+international\_sales)/1000000 AS millions\_of\_dollars FROM movies INNER JOIN boxoffice ON movies.id = boxoffice.movie\_id;
2. SELECT id, title, rating \* 10 AS rating\_in\_percentage FROM movies INNER JOIN boxoffice ON movies.id = boxoffice.movie\_id;
3. SELECT id, title, year FROM movies WHERE year % 2 = 0;

**Exercise 10 — Tasks Queries with aggregates**

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 ✓

**Answers**:

1. SELECT role, name, MAX(years\_employed) AS longest\_work\_time FROM employees;
2. SELECT AVG(years\_employed) AS average\_time, role FROM employees GROUP BY role;
3. SELECT SUM(years\_employed), building FROM employees GROUP BY building;

**Exercise 11 — Tasks Queries with aggregates**

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 ✓

**Answers**:

1. SELECT COUNT(role) FROM employees WHERE role="Artist";
2. SELECT COUNT(role) AS number\_of\_employees, role FROM employees GROUP BY role;
3. SELECT SUM(years\_employed) AS total\_number\_of\_years FROM employees WHERE role="Engineer";

**Exercise 12 — Tasks: Order of execution of a query**

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 ✓

**Answers**:

1. SELECT COUNT(title) AS number\_of\_movies, director FROM movies GROUP BY director;
2. SELECT director, SUM(domestic\_sales + international\_sales) AS total\_sales FROM movies INNER JOIN boxoffice ON movies.id = boxoffice.movie\_id GROUP BY director ORDER BY total\_sales DESC;

**Exercise 13 — Tasks: Inserting rows**

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. ✓

**Answers**:

1. INSERT INTO movies (title, director, year, length\_minutes) VALUES ("Toy Story 4", "Kevin Spacey", 2025, 90);
2. INSERT INTO boxoffice VALUES (15, 8.7, 340000000, 270000000);

**Exercise 14 — Tasks: Updating Rows**

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 ✓

**Answers**:

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;

**Exercise 15 — Tasks: Deletion Rows**

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. ✓

Answer:

1. DELETE FROM movies WHERE year < 2005;
2. 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. ✓.

**Answers**:

1. CREATE TABLE Database (

Name TEXT,

Version FLOAT,

Download\_count INTEGER

);

**Exercise 17 — Tasks: Alter Table**

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. ✓

**Answers**:

1. ALTER TABLE movies ADD column Aspect\_ratio FLOAT;
2. ALTER TABLE movies ADD column Language TEXT DEFAULT English;

**Exercise 18 — Tasks: Drop Tables**

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 ✓

**Answers**:

1. DROP TABLE Movies;
2. DROP TABLE Boxoffice;