

MySQL TASK

SQL Lesson 1: SELECT queries 101

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 ›

Solutions :

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)

Table: Movies

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

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 >

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

RESET

Solutions :

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

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 * FROM movies WHERE Title LIKE 'WALL-%';

RESET

Continue >

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

Solutions :

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-%';**
-

SQL Lesson 4: Filtering and sorting Query results

Table: Movies

Id	Title	Director	Year	Length_minutes
14	Monsters University	Dan Scanlon	2013	110
9	Monsters, Inc.	Pete Docter	2001	92
13	Ratatouille	Brad Bird	2007	115
11	The Incredibles	Brad Bird	2004	116
1	Toy Story	John Lasseter	1995	81

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.

Continue >

```
SELECT *  
FROM movies  
ORDER BY Title ASC  
LIMIT 5 OFFSET 5;  
|
```

RESET

Solutions :

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;**

SQL Review: Simple SELECT Queries

Table: North_american_cities

City	Population
Chicago	2718782
Houston	2195914

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 ›

```
SELECT City, Population
FROM North_american_cities
WHERE Country = 'United States'
ORDER BY Population DESC
LIMIT 2 OFFSET 2;
```

RESET

Solutions :

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, Longitude
FROM North_american_cities
WHERE Longitude < (SELECT Longitude FROM North_american_cities
WHERE City = 'Chicago')
ORDER BY Longitude ASC;**
4. **SELECT City, Population
FROM North_american_cities
WHERE Country = 'Mexico'
ORDER BY Population DESC
LIMIT 2;**
5. **SELECT City, Population
FROM North_american_cities
WHERE Country = 'United States'
ORDER BY Population DESC
LIMIT 2 OFFSET 2;**

SQL Lesson 6: Multi-table queries with JOINS

Query Results

Id	Title	Director	Year	Length_minutes	Rating
9	WALL-E	Andrew Stanton	2008	104	8.5
11	Toy Story 3	Lee Unkrich	2010	103	8.4
1	Toy Story	John Lasseter	1995	81	8.3
10	Up	Pete Docter	2009	101	8.3
5	Finding Nemo	Andrew Stanton	2003	107	8.2
4	Monsters, Inc.	Pete Docter	2001	92	8.1
8	Ratatouille	Brad Bird	2007	115	8
6	The Incredibles	Brad Bird	2004	116	8
3	Toy Story 2	John Lasseter	1999	93	7.9
14	Monsters University	Dan Scanlon	2013	110	7.4

```
SELECT Movies.Id, Title, Director, Year, Length_minutes, Rating
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 >

Solutions :

1. **SELECT Movies.Id, Title, Director, Year, Length_minutes, Domestic_sales, International_sales
FROM Movies
INNER JOIN BoxOffice ON Movies.Id = BoxOffice.Movie_id;**
2. **SELECT Movies.Id, Title, Director, Year, Length_minutes, Domestic_sales, International_sales
FROM Movies
INNER JOIN BoxOffice ON Movies.Id = BoxOffice.Movie_id
WHERE International_sales > Domestic_sales;**
3. **SELECT Movies.Id, Title, Director, Year, Length_minutes, Rating
FROM Movies
INNER JOIN BoxOffice ON Movies.Id = BoxOffice.Movie_id
ORDER BY Rating DESC;**

SQL Lesson 7: OUTER JOINS

Query Results

Building_name	Role
1e	Engineer
1e	Manager
1w	
2e	
2w	Artist
2w	Manager

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 >

```
SELECT DISTINCT b.Building_name, e.Role
FROM Buildings b
LEFT JOIN Employees e ON b.Building_name = e.Building;
|
```

RESET

Solutions :

1. **SELECT DISTINCT e.Building**
FROM Employees e
LEFT JOIN Buildings b ON e.Building = b.Building_name
WHERE e.Building IS NOT NULL;
 2. **SELECT b.Building_name, b.Capacity**
FROM Buildings b
LEFT JOIN Employees e ON b.Building_name = e.Building
GROUP BY b.Building_name, b.Capacity;
 3. **SELECT DISTINCT b.Building_name, e.Role**
FROM Buildings b
LEFT JOIN Employees e ON b.Building_name = e.Building;
-

SQL Lesson 8: A short note on NULLs

Query Results

Building_name
1w
2e

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 ›

```
SELECT Building_name
FROM Buildings
WHERE Building_name NOT IN (SELECT DISTINCT Building FROM Employees WHERE
    Building IS NOT NULL);
|
```

RESET

Solutions :

1. **SELECT Name, Role
FROM Employees
WHERE Building IS NULL;**
 2. **SELECT Building_name
FROM Buildings
WHERE Building_name NOT IN (SELECT DISTINCT Building
FROM Employees WHERE Building IS NOT NULL);**
-

SQL Lesson 9: Queries with expressions

Query Results

Id	Title	Director	Year	Length_minutes
2	A Bug's Life	John Lasseter	1998	95
6	The Incredibles	Brad Bird	2004	116
7	Cars	John Lasseter	2006	117
9	WALL-E	Andrew Stanton	2008	104
11	Toy Story 3	Lee Unkrich	2010	103
13	Brave	Brenda Chapman	2012	102

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 >

```
SELECT
  Id,
  Title,
  Director,
  Year,
  Length_minutes
FROM
```

Solutions :

1.

```
SELECT
  m.Id,
  m.Title,
  m.Director,
  m.Year,
  m.Length_minutes,
  (b.Domestic_sales + b.International_sales) / 1000000 AS combined_sales_millions
FROM
  Movies m
JOIN
  Boxoffice b ON m.Id = b.Movie_id;
```
 2.

```
SELECT
  m.Id,
  m.Title,
  m.Director,
  m.Year,
  m.Length_minutes,
  b.Rating * 10 AS rating_percent
FROM
  Movies m
JOIN
  Boxoffice b ON m.Id = b.Movie_id;
```
 3.

```
SELECT
  Id,
  Title,
  Director,
  Year,
  Length_minutes
FROM
  Movies
WHERE
  Year % 2 = 0;
```
-

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

Table: Employees

Building	Total_years_worked
1e	29
2w	36

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 >

```
SELECT Building, SUM(Years_employed) AS total_years_worked
FROM Employees
GROUP BY Building;
```

RESET

Solutions :

1. **SELECT MAX(Years_employed) AS longest_time
FROM Employees;**
 2. **SELECT Role, AVG(Years_employed) AS avg_years_employed
FROM Employees
GROUP BY Role;**
 3. **SELECT Building, SUM(Years_employed) AS total_years_worked
FROM Employees
GROUP BY Building;**
-

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

Table: Employees

Total_years_employed
17

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 >

```
SELECT SUM(Years_employed) AS total_years_employed
FROM Employees
WHERE Role = 'Engineer';
```

RESET

Solutions :

1. **SELECT COUNT(*) AS num_artists
FROM Employees
WHERE Role = 'Artist';**
 2. **SELECT Role, COUNT(*) AS num_employees
FROM Employees
GROUP BY Role;**
 3. **SELECT SUM(Years_employed) AS total_years_employed
FROM Employees
WHERE Role = 'Engineer';**
-

SQL Lesson 12: Order of execution of a Query

Query Results

Director	TotalSales
Andrew Stanton	1458055121
Brad Bird	1255164910
Brenda Chapman	538983207
Dan Scanlon	743559607
John Lasseter	2232208025
Lee Unkrich	1063171911
Pete Docter	1294159000

```
SELECT m.Director,  
       SUM(b.Domestic_sales + b.International_sales) AS TotalSales  
FROM Movies m  
JOIN Boxoffice b ON m.Id = b.Movie_id  
GROUP BY m.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 >

Solutions :

1. **SELECT Director, COUNT(*) AS NumMoviesDirected
FROM Movies
GROUP BY Director;**
 2. **SELECT m.Director,
SUM(b.Domestic_sales + b.International_sales) AS TotalSales
FROM Movies m
JOIN Boxoffice b ON m.Id = b.Movie_id
GROUP BY m.Director;**
-

SQL Lesson 13: Inserting rows

Query Results

Id	Title	Director	Year	Length_minutes
----	-------	----------	------	----------------

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 ›

```
INSERT INTO Boxoffice (Movie_id, Rating, Domestic_sales, International_sales)
VALUES ((SELECT Id FROM Movies WHERE Title = 'Toy Story 4'), 8.7, 340000000, 270000000);
```

RUN QUERY RESET

Solutions :

1. **INSERT INTO Movies (Title, Director, Year, Length_minutes)**
VALUES ('Toy Story 4', 'Any Director', 2023, 100);
 2. **INSERT INTO Boxoffice (Movie_id, Rating, Domestic_sales, International_sales)**
VALUES ((SELECT Id FROM Movies WHERE Title = 'Toy Story 4'), 8.7, 340000000, 270000000);
-

SQL Lesson 14: Updating rows

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 Title = 'Toy Story 8';
```

RUN QUERY RESET

Exercise 14 — Tasks

- The director for A Bug's Life is incorrect, it was actually directed by **John Lasseter** ✓
- The year that Toy Story 2 was released is incorrect, it was actually released in **1999** ✓
- 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 ›

Solutions :

- UPDATE Movies**
SET Director = 'John Lasseter'
WHERE Title = 'A Bug's Life';
 - UPDATE Movies**
SET Year = 1999
WHERE Title = 'Toy Story 2';
 - UPDATE Movies**
SET Title = 'Toy Story 3', Director = 'Lee Unkrich'
WHERE Title = 'Toy Story 8';
-

SQL Lesson 15: Deleting rows

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';
```

[RUN QUERY](#) [RESET](#)

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.

[Continue >](#)

Solutions :

1. **DELETE FROM Movies
WHERE Year < 2005;**
 2. **DELETE FROM Movies
WHERE Director = 'Andrew Stanton';**
-

SQL Lesson 16: Creating tables

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.

Continue >

```
CREATE TABLE IF NOT EXISTS Database (  
  Name TEXT,  
  Version FLOAT,  
  Download_count INTEGER  
);
```

RUN QUERY RESET

Solutions :

1. CREATE TABLE IF NOT EXISTS Database (
Name TEXT,
Version FLOAT,
Download_count INTEGER
);

SQL Lesson 17: Altering tables

Table: Movies

	Title	Director	Year	Length_minutes	Aspect_ratio	Language
1	Toy Story	John Lasseter	1995	81		English
2	A Bug's Life	John Lasseter	1998	95		English
3	Toy Story 2	John Lasseter	1999	93		English
4	Monsters, Inc.	Pete Docter	2001	92		English
5	Finding Nemo	Andrew Stanton	2003	107		English
6	The Incredibles	Brad Bird	2004	116		English
7	Cars	John Lasseter	2006	117		English
8	Ratatouille	Brad Bird	2007	115		English
9	WALL-E	Andrew Stanton	2008	104		English
10	Up	Pete Docter	2009	101		English

ALTER TABLE Movies

ADD Language TEXT DEFAULT 'English';

RUN QUERYRESET

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 >

Solutions :

1. **ALTER TABLE Movies ADD Aspect_ratio FLOAT;**
 2. **ALTER TABLE Movies ADD 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 >

```
DROP TABLE IF EXISTS BoxOffice;
```

RUN QUERY RESET

Solutions :

1. **DROP TABLE IF EXISTS Movies;**
 2. **DROP TABLE IF EXISTS BoxOffice;**
-



SQLBolt

Learn SQL with simple, interactive exercises.

SQL Lesson X: To infinity and beyond!



You've finished the tutorial!