# Introduction to SQL Part 2

Today’s lab is all about writing SQL code to pull data from databases.

This is essential knowledge for all data-related careers including:

* Data analyst
* Database administrator
* Data scientist
* Data engineer

## Write SQL code to retrieve data from the Chinook database

### SELECT, FROM, WHERE, ORDER BY & UNION

The first set of questions are based on the tracks table from the Chinook database.

tracks

Table

Description automatically generated

1. Write a SQL query to get all columns from the tracks table.

SELECT \*

FROM Tracks;

1. Write a SQL query to select only the Name, Milliseconds, Bytes and AlbumId column from the tracks table.

SELECT Name, Milliseconds, Bytes, AlbumId

FROM Tracks;

1. Rewrite your SQL query from question 2 so that it only pulls rows with GenreId equal to 1.

SELECT Name, Milliseconds, Bytes, AlbumId

FROM Tracks

WHERE GenreId = 1;

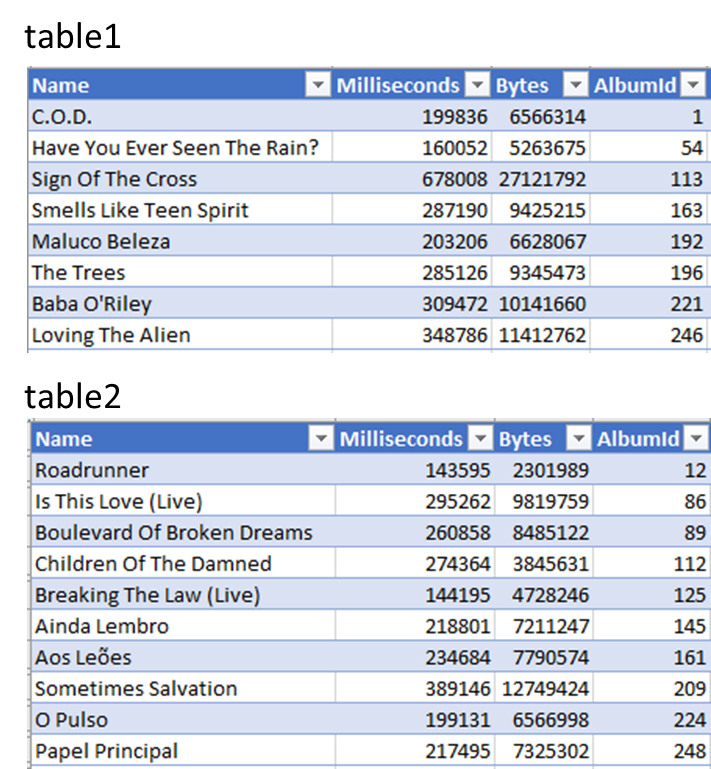
1. Rewrite your SQL query from question 2 again so that it only pulls rows with GenreID NOT equal to 1.

SELECT Name, Milliseconds, Bytes, AlbumId

FROM Tracks;

WHERE GenreId != 1;

1. Your queries from question 3 and question 4 produced identically-structured tables containing different data. Assume that the question 3 table is named “table1” and the question 4 table is named “table2”. Write a SQL query that stacks these two tables together to create one table.



SELECT \*

FROM Table1

UNION

SELECT \*

FROM Table2;

1. Rewrite your SQL query from question 2 again so that it only pulls rows that have GenreId equal to 1 OR AlbumId less than 100.

SELECT Name, Milliseconds, Bytes, AlbumId

FROM Tracks

WHERE GenreId = 1 OR AlbumId < 100;

1. Rewrite your SQL query from question 6 so that it also sorts in descending order according to Milliseconds.

SELECT Name, Milliseconds, Bytes, AlbumId

FROM Tracks

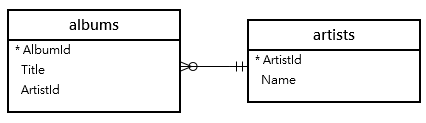
WHERE GenreId = 1 OR AlbumId < 100

ORDER BY Milliseconds DESC;

### INNER JOIN & LEFT JOIN

The next set of questions are based on multiple tables.

1. Write a SQL query to pull Title from the albums table and Name from the artists table using an inner join.



SELECT albums.Title,

artists.Name

FROM albums

INNER JOIN artists

ON albums.ArtistId = artists.ArtistId;

1. Rewrite your SQL query from question 8 so that the order of the columns is reversed with Name first and Title second.

SELECT artists.Name,

albums.Title

FROM artists

INNER JOIN albums

ON artists.ArtistId = albums.ArtistId;

1. Rewrite your SQL query from question 9 so that the resulting table is sorted according to Name.

SELECT artists.Name,

albums.Title

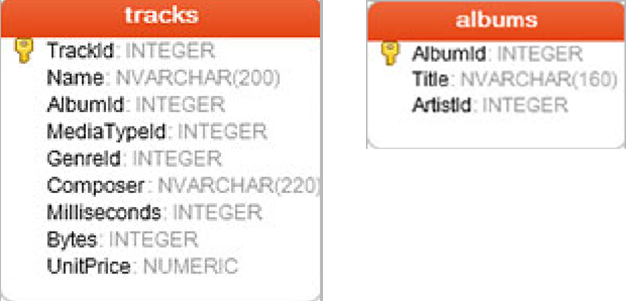
FROM artists

INNER JOIN albums

ON artists.ArtistId = albums.ArtistId

ORDER BY artists.Name;

1. Write a SQL query with a left join that pulls all the columns from the tracks table and adds Title from the albums table.



SELECT tracks.\*,

albums.Title

FROM tracks

LEFT JOIN albums

ON tracks.AlbumId = albums.AlbumId;

1. Rewrite your SQL query from question 11 so that the resulting table is sorted by Title.

SELECT tracks.\*,

albums.Title

FROM tracks

LEFT JOIN albums

ON tracks.AlbumId = albums.AlbumId

ORDER BY albums.Title;