**“MUSIC STORE DATA ANALYSIS”**

* **AIM OF PROJECT:**

The aim of a music store data analysis project is to create a structured and organized system for storing, managing, and retrieving information related to music using MYSQL.

* **INTRODUCTION:**

In an era dominated by the consumption and sharing of digital music, the need for a robust and organized music database is paramount. The Music Database Project aims to address this need by

providing a comprehensive system for managing and accessing information about artists, albums, songs, and genres.

* **OBJECTIVE:**

**Efficient Data Management:** Organize music-related data in a structured and accessible format.

**User-Friendly Interface**: Intuitive navigation and a user-friendly interface for easy interaction.

**Personalization**: Enable users to mark favorites, and personalize their music experience.

**Data Analysis**: Provide insights through data analysis, such as popular genres, most-played songs, and artist rankings.

|  |
| --- |
| ARTISTS |
| *ArtistsID(PK)*  *ArtistsName* |

|  |
| --- |
| GENRES |
| *GenersID (PK)*  *GenerName* |

|  |
| --- |
| ALBUMS |
| *AlbumID(PK)*  *AlbumTitle*  *RealeaseYear*  *ArtistsID(FK)*  *GenersID(FK)* |

|  |
| --- |
| USERS |
| UserID(PK)  UserName |

|  |
| --- |
| USERFAVORITES |
| *UserID(FK)*  *UserName(FK)* |

|  |
| --- |
| SONGS |
| *SongID(PK)*  *SongTitle*  *TrackNumber*  *Duration*  *AlbumID(FK)* |

* **Structure of Tables**

**1.ARTISTS**

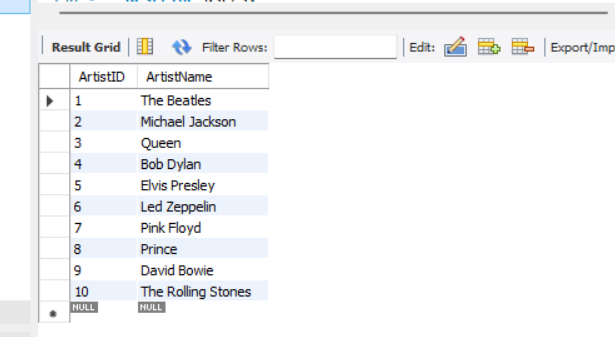
*CREATE TABLE Artists (*

*ArtistID INT PRIMARY KEY,*

*ArtistName VARCHAR(255) NOT NULL*

*);*

*Select \* from Artists;*



**2. GENRES**

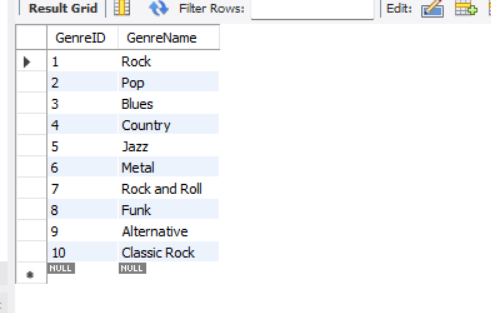
*CREATE TABLE Genres (*

*GenreID INT PRIMARY KEY,*

*GenreName VARCHAR(255) NOT NULL*

*);*

*Select \* from Genres;*



**3. ALBUMS**

*CREATE TABLE Albums (*

*AlbumID INT PRIMARY KEY,*

*AlbumTitle VARCHAR(255) NOT NULL,*

*ReleaseYear INT,*

*ArtistID INT,*

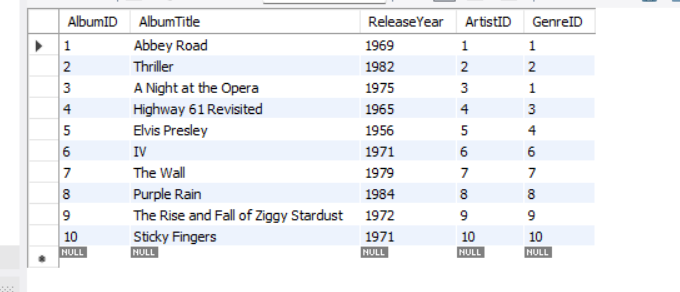
*GenreID INT,*

*FOREIGN KEY (ArtistID) REFERENCES Artists(ArtistID),*

*FOREIGN KEY (GenreID) REFERENCES Genres(GenreID)*

*);*

*Select \* from Albums;*



***4.SONGS***

*CREATE TABLE Songs (*

*SongID INT PRIMARY KEY,*

*SongTitle VARCHAR(255) NOT NULL,*

*TrackNumber INT,*

*AlbumID INT,*

*Duration INT,*

*FOREIGN KEY (AlbumID) REFERENCES Albums(AlbumID)*

*);*

*Select \* from Songs;*

**

***5. Users***

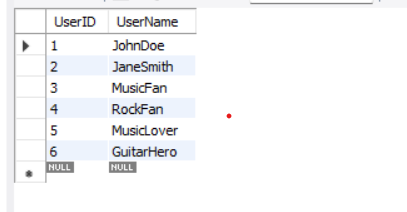
*CREATE TABLE Users (*

*UserID INT PRIMARY KEY,*

*UserName VARCHAR(255) NOT NULL*

*);*

*Select \* from Users;*

**

***6.USERFAVORITES***

*CREATE TABLE UserFavorites (*

*UserID INT,*

*SongID INT,*

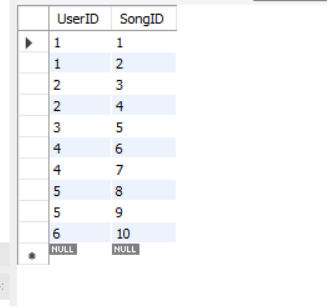
*PRIMARY KEY (UserID, SongID),*

*FOREIGN KEY (UserID) REFERENCES Users(UserID),*

*FOREIGN KEY (SongID) REFERENCES Songs(SongID)*

*);*

*Select \* From Userfavorites;*

**

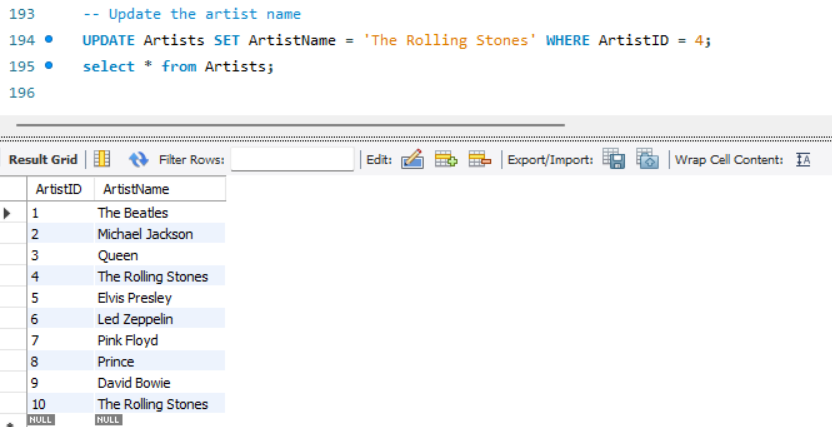
* **QUERIES**

**DML(Data Manipulation language)**

#Update the artist name

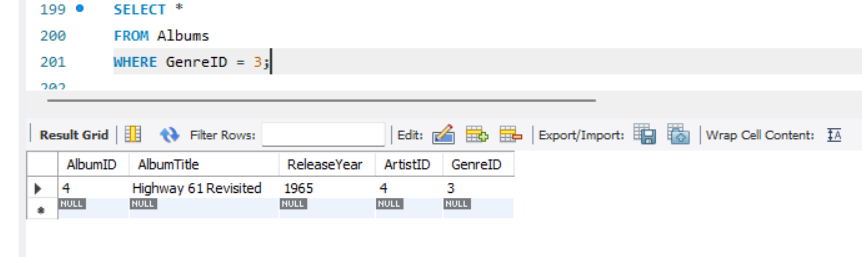
*UPDATE Artists SET ArtistName = 'The Rolling Stones' WHERE ArtistID = 4;*

*select \* from Artists;*

****

**SELECT with WHERE clause:**

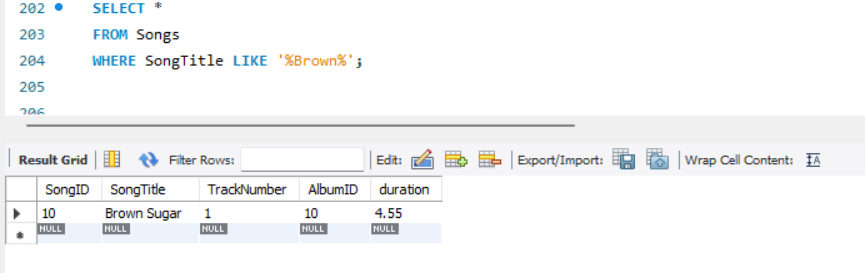
#Retrieve albums of a specific genre:

*SELECT \* FROM Albums WHERE GenreID = 3;*

**LIKE operator:**

#Retrieve all songs with titles containing the word 'Brown'.

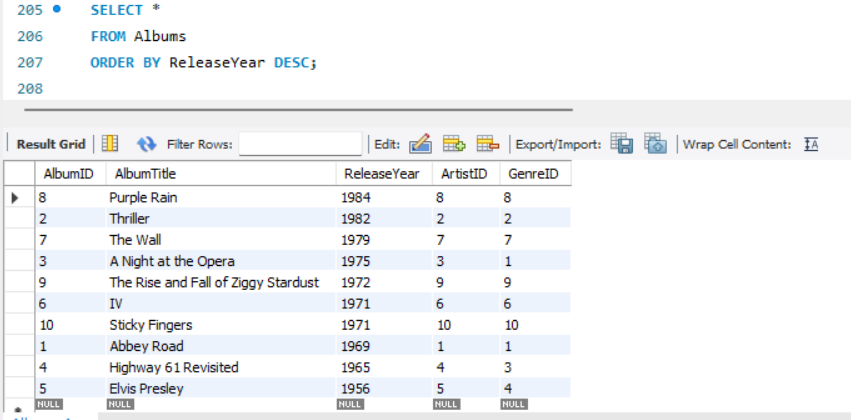
*SELECT \* FROM Songs WHERE SongTitle LIKE '%Brown%';*



**ORDER BY clause:**

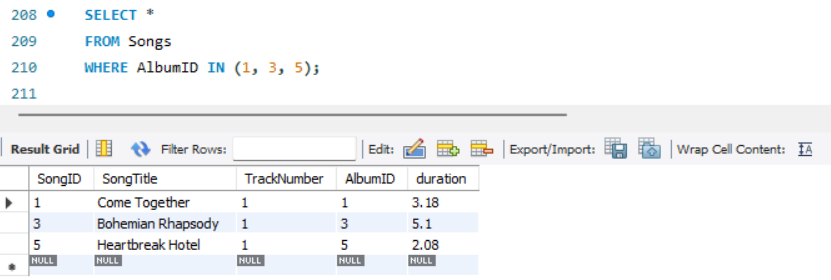
#Retrieve all albums, ordered by release year in descending order

*SELECT \* FROM Albums ORDER BY ReleaseYear DESC;*



**IN operator:**

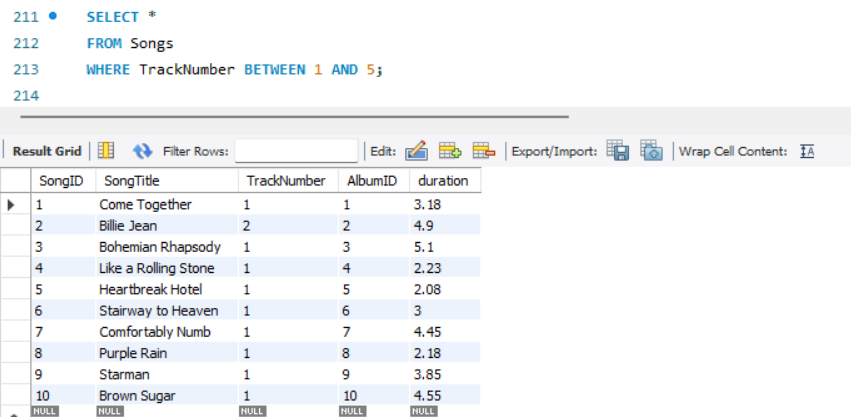
**#**Retrieve all songs from a specific list of albums.

*SELECT \* FROM Songs WHERE AlbumID IN (1, 3, 5);* 

**BETWEEN operator:**

#Retrieve all songs with a track number between 1 and 5.

*SELECT \* FROM Songs WHERE TrackNumber BETWEEN 1 AND 5;*

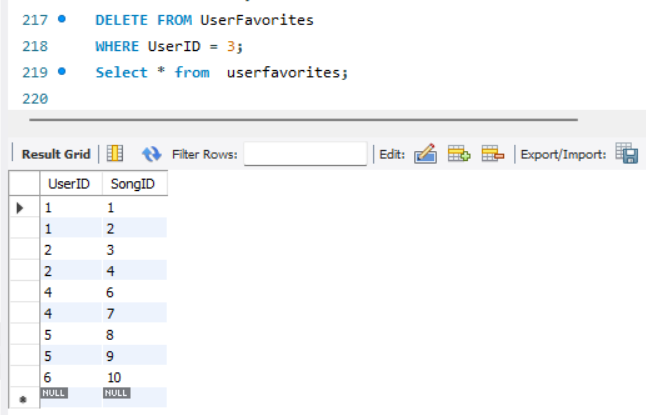


**DELETE with WHERE clause:**

#Delete all user favorites for a specific user.

*DELETE FROM UserFavorites WHERE UserID = 3;*

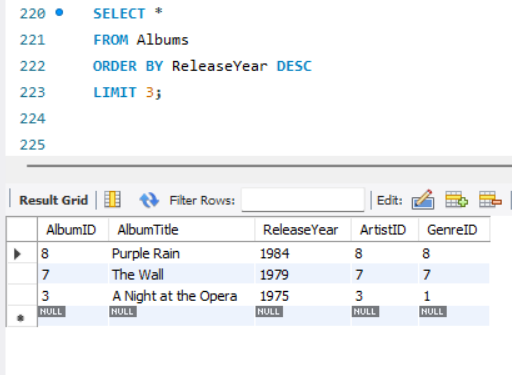
*Select \* from userfavorites;*



**LIMIT Clause:**

#Retrieve the top 3 albums ordered by release year:

*SELECT \* FROM Albums ORDER BY ReleaseYear DESC LIMIT 3;*



**Aggregate Function:**

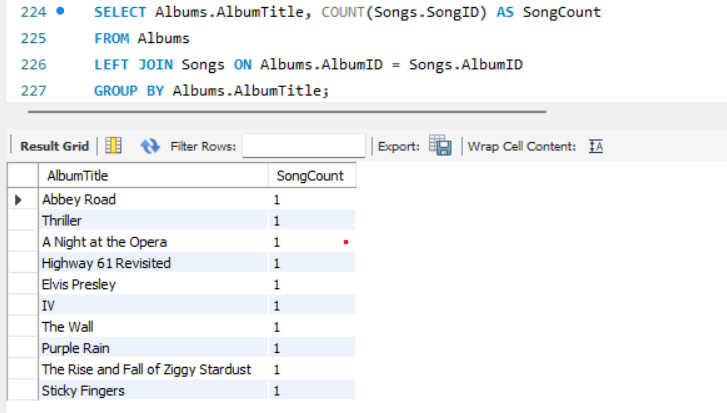
1.#Count the number of songs in each album:

*SELECT Albums.AlbumTitle, COUNT(Songs.SongID) AS SongCount*

*FROM Albums*

*LEFT JOIN Songs ON Albums.AlbumID = Songs.AlbumID*

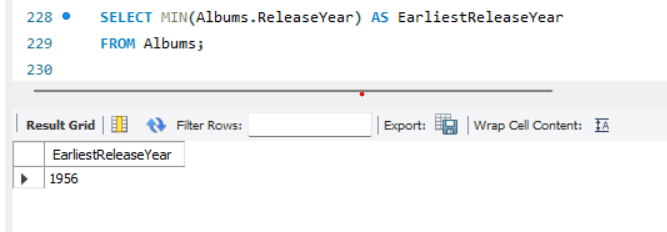
*GROUP BY Albums.AlbumTitle;*



2.#Find the earliest release year among all albums:

*SELECT MIN(Albums.ReleaseYear) AS EarliestReleaseYear*

*FROM Albums;*



**Group by with Having clause:**

#Find users who have more than one favorite songs

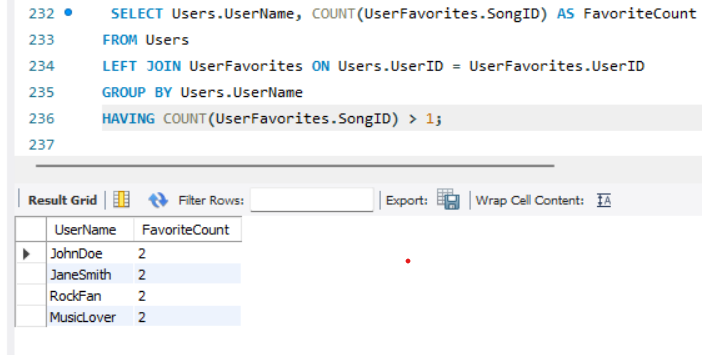
*SELECT Users.UserName, COUNT(UserFavorites.SongID) AS FavoriteCount*

*FROM Users*

*LEFT JOIN UserFavorites ON Users.UserID = UserFavorites.UserID*

*GROUP BY Users.UserName*

*HAVING COUNT(UserFavorites.SongID) > 1;*

****

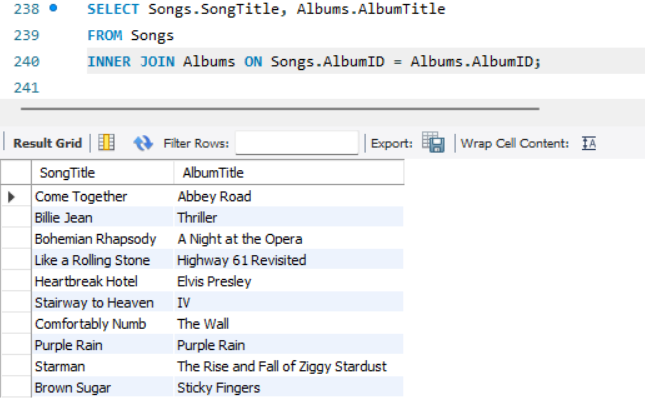
**JOINS:**

**1.Inner Join** - Retrieve songs with album information:

*SELECT Songs.SongTitle, Albums.AlbumTitle*

*FROM Songs*

*INNER JOIN Albums ON Songs.AlbumID = Albums.AlbumID;*

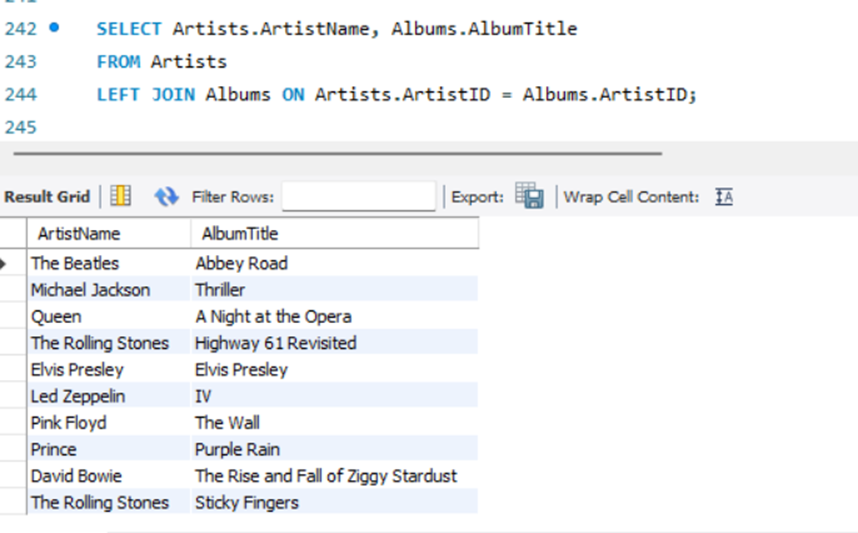
****

**2. Left Join** - Retrieve all artists and their albums.

*SELECT Artists.ArtistName, Albums.AlbumTitle*

*FROM Artists*

*LEFT JOIN Albums ON Artists.ArtistID = Albums.ArtistID;*

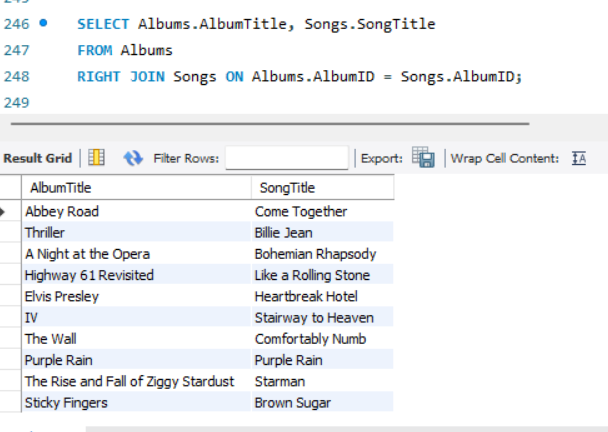
**

**3.Right join**-Retrieve all albums and their songs:

*SELECT Albums.AlbumTitle, Songs.SongTitle*

*FROM Albums*

*RIGHT JOIN Songs ON Albums.AlbumID = Songs.AlbumID;*

**

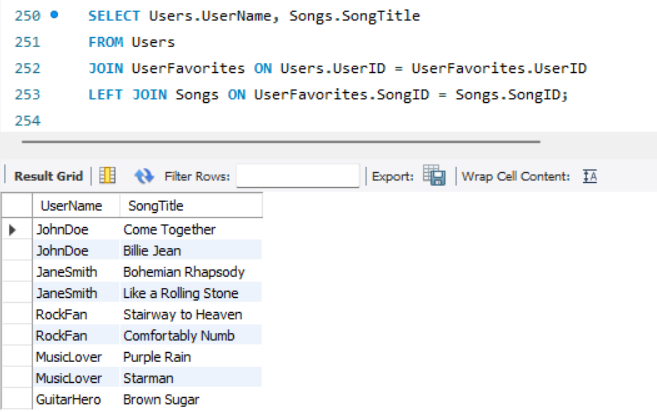
**4.Full Outer Join** - Retrieve all users and their favorite songs:

*SELECT Users.UserName, Songs.SongTitle*

*FROM Users*

*JOIN UserFavorites ON Users.UserID = UserFavorites.UserID*

*LEFT JOIN Songs ON UserFavorites.SongID = Songs.SongID;*

**

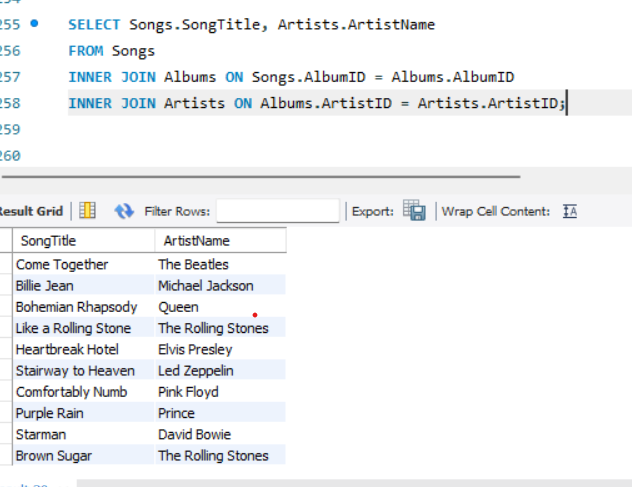
**4.Self Join** - Retrieve all songs with the artist name:

*SELECT Songs.SongTitle, Artists.ArtistName*

*FROM Songs*

*INNER JOIN Albums ON Songs.AlbumID = Albums.AlbumID*

*INNER JOIN Artists ON Albums.ArtistID = Artists.ArtistID*;



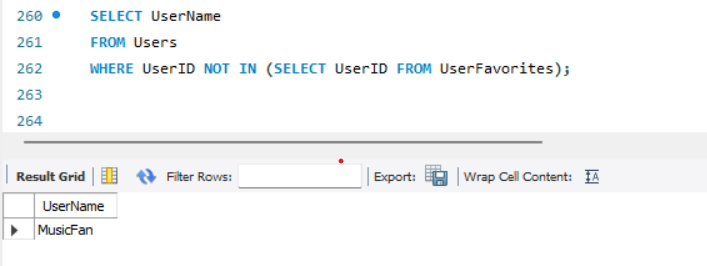
**SUBQUERIES:**

1. Find users who have not set any favorites:

*SELECT UserName*

*FROM Users*

*WHERE UserID NOT IN (SELECT UserID FROM UserFavorites);*

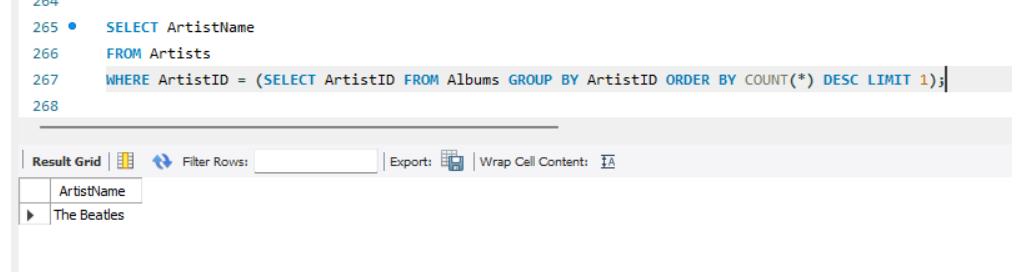


2. Retrieve the artist with the most albums:

*SELECT ArtistName*

*FROM Artists*

*WHERE ArtistID = (SELECT ArtistID FROM Albums GROUP BY ArtistID ORDER BY COUNT(\*) DESC LIMIT 1);*

**

3.Retrieve users who have favorites in a specific genre:

*SELECT UserName FROM Users*

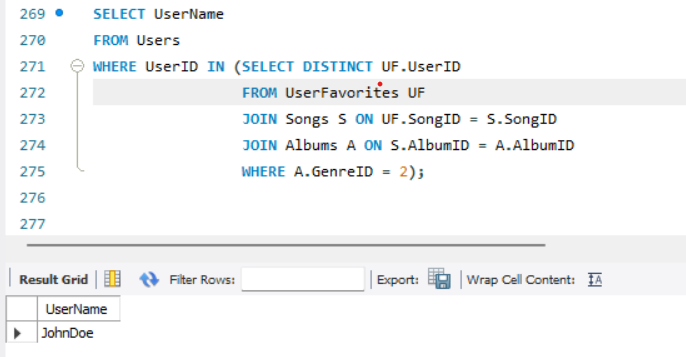
*WHERE UserID IN (SELECT DISTINCT UF.UserID*

*FROM UserFavorites UF*

*JOIN Songs S ON UF.SongID = S.SongID*

*JOIN Albums A ON S.AlbumID = A.AlbumID*

*WHERE A.GenreID = 2);*

**

--------------------------- THANK YOU ------------------------