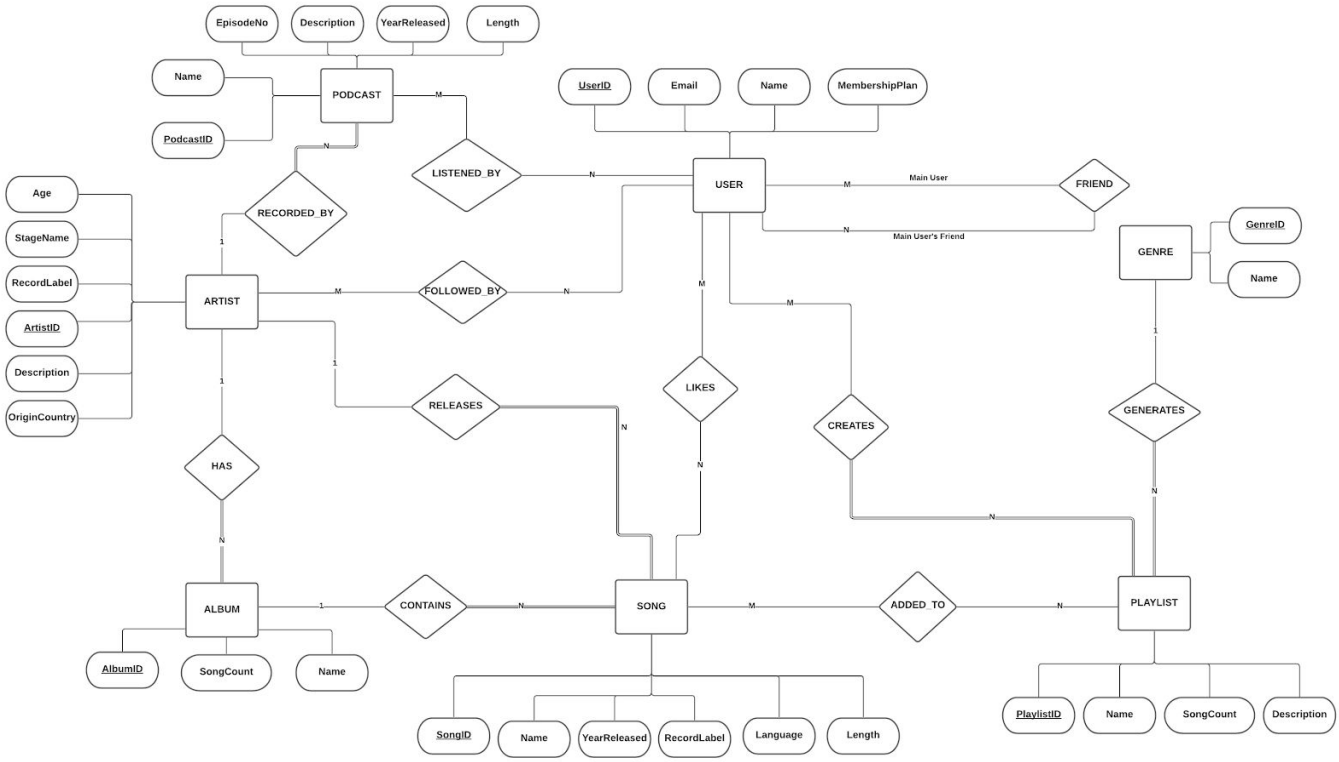
Final Project Proposal

GroupProject_team5 Members:
Joaquin Luna, Anirudh Prasad, Madhav Rajkondawar, Srikar Duggempudi, Jay Brar

Design Documentation

Entity-Relationship Diagram

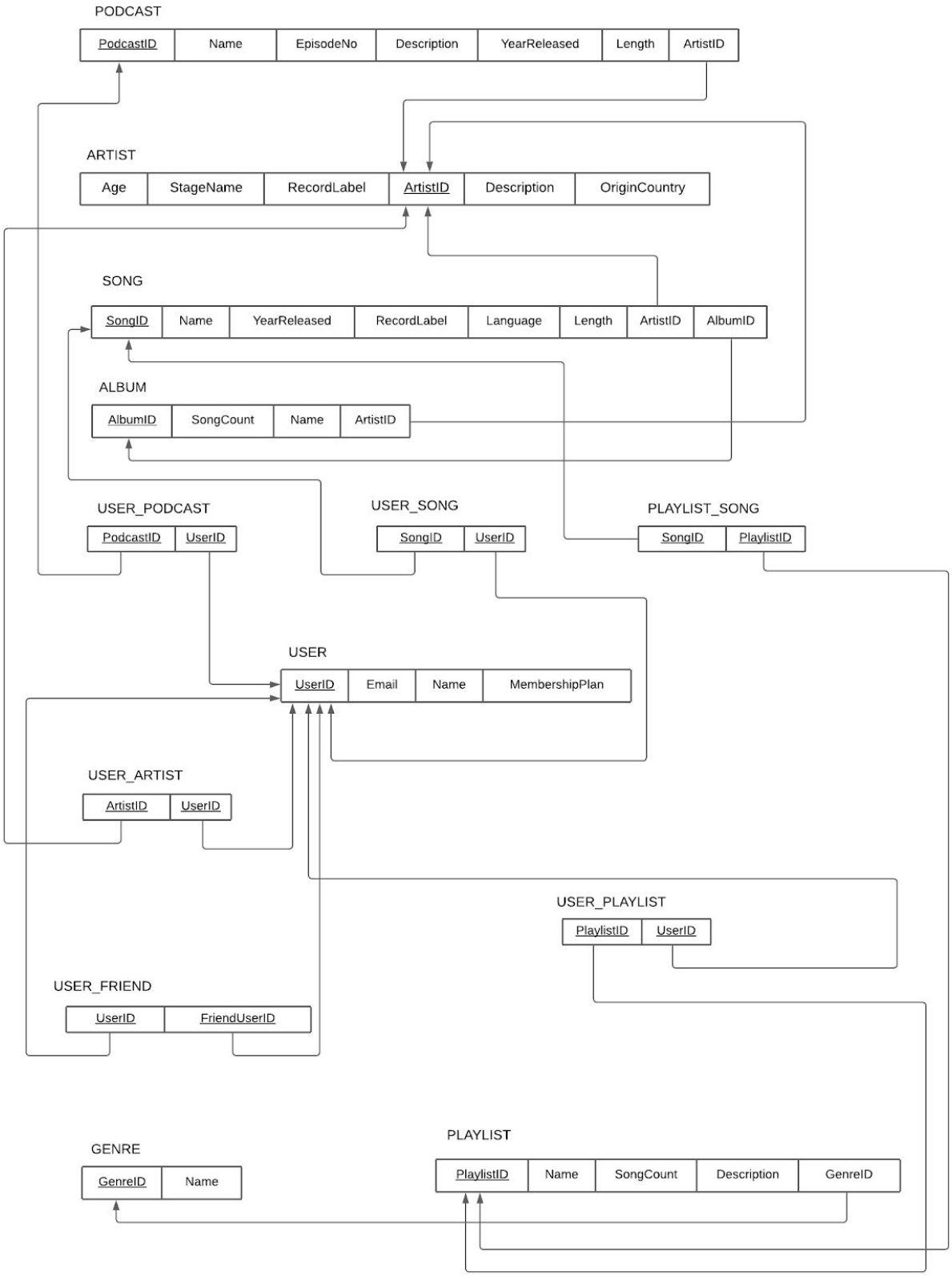


(**NOTE**: The PNG file of this diagram is provided in the zip file)

Assumptions:

1. **Every song must belong to an album**
2. **A single song can be treated as an album**
3. **A genre generates a recommended set of playlist(s)**

Relational Data Model



(**NOTE**: The PNG file of this diagram is provided in the zip file)

SQL Statements

SQL statements used to create all database entities are included in the "MusicLibraryDatabase.sql" file.

SQL Statement	**Purpose**
SELECT name FROM SONG WHERE ArtistID = (SELECT ArtistID FROM ARTIST WHERE StageName = 'Drake');	List all the songs released by a certain artist
SELECT Name FROM ALBUM WHERE ArtistID = (SELECT ArtistID FROM ARTIST WHERE StageName = 'Jay-Z');	List all the albums released by a certain artist
Select StageName FROM ARTIST WHERE (Age < 35);	List names of Artists whose age is less than 35
SELECT StageName FROM ARTIST WHERE OriginCountry = 'Indonesia';	List names of Artists from Indonesia
SELECT StageName, COUNT(*) FROM ALBUM, ARTIST WHERE ALBUM.ArtistID = ARTIST.ArtistID GROUP BY StageName;	List the number of albums released by each artist
Select StageName, MIN(Age) FROM ARTIST;	List the name and age of the youngest artist
SELECT AlbumID, Name, Length FROM SONG ORDER BY AlbumID;	List all the songs in all the albums
SELECT Name, Length FROM SONG WHERE Length LIKE '%00:02%';	List all the songs which have a length of more than 2 but less than 3 minutes
SELECT u.UserID, s.Name FROM USER as u, USER_SONG AS us_s, SONG as s WHERE us_s.UserID = u.UserID AND us_s.SongID = s.SongID AND u.UserID = 20000;	List all the songs listened to by a user with UserID 20000
SELECT AlbumID, Name FROM SONG WHERE AlbumID = (SELECT AlbumID FROM ALBUM WHERE AlbumID = 30000);	List all the songs in an album with albumID 30000
SELECT pl.PlaylistID, s.Name FROM PLAYLIST AS pl, SONG AS s, PLAYLIST_SONG AS pl_s WHERE pl_s.PlaylistID = pl.PlaylistID AND pl_s.SongID = s.SongID AND pl.PlaylistID = 30000;	List all the songs in the playlist with PlaylistID 30000

SELECT Name FROM PODCAST ORDER BY Name;	List all of the podcasts and sort them in alphabetical order
SELECT P.Name FROM USER_PODCAST UP, PODCAST P WHERE UP.UserID = 40000 AND UP.PodcastID = P.PodcastID;	List all podcasts listened to by user with UserID 40000
SELECT Name FROM PODCAST WHERE Length < "01:00:00";	List all the podcasts which have a length shorter than 1 hour
SELECT F.Name FROM USER_FRIEND UF, USER F WHERE UF.UserID = 10001 AND UF.FriendUserID = F.UserID;	List all friends of user with UserID 10001
SELECT StageName FROM ARTIST WHERE OriginCountry LIKE "%U.S.A%";	List all artists who are from U.S.A
SELECT StageName FROM ARTIST WHERE OriginCountry = ‘Canada’;	List all the artists who are from Canada
SELECT Name FROM SONG WHERE Language = ‘Hindi’;	List all Hindi songs
SELECT USER.Name FROM USER, USER_PLAYLIST, PLAYLIST, GENRE WHERE USER.UserID = USER_PLAYLIST.UserID AND USER_PLAYLIST.PlaylistID = PLAYLIST.PlaylistID AND PLAYLIST.GenreID = GENRE.GenreID AND GENRE.Name = ‘Rap’;	List all user names that listen to Rap Genre
SELECT Name FROM ALBUM WHERE Name LIKE ‘%2%’;	List all albums with the number 2 in the Album title
SELECT Name FROM PODCAST WHERE YearReleased = 2020;	List all podcasts from the year 2020
SELECT Name From PLAYLIST WHERE SongCount = 3;	List all the playlists that have 3 total songs
SELECT AVG(SongCount) FROM ALBUM;	List the average number of songs the albums have
SELECT COUNT(*) FROM SONG WHERE YearReleased < 2020;	List the number of songs that were released before 2020
SELECT Count(*) FROM USER WHERE MembershipPlan = 'Premium';	List the number of users with a Premium subscription

Normal Form

Each of the relations in the Music Streaming Library database is in the Boyce-Codd normal form. Information regarding how each relation satisfies the first, second, third, and Boyce-Codd normal forms is included below.

1NF - All attributes are single atomic values and don't contain any multi-level nested tuples. No multivalued, composite and their combinations, or nested relations.

2NF - Relation is in 1NF and all non-key attributes are fully functionally dependent on the primary/secondary keys.

3NF - Relation is in 2NF and all non-key attributes are not fully functionally dependent on any other non-key attributes within the same relation. Every non-key attribute is directly dependent on the key.

BCNF - Relation is in 3NF and all non-key attributes are not fully functionally dependent on the key attributes within the same relation.

Project Evaluation

Effort Spent:

We spent much of the last weeks of the quarter meeting together when our schedules lined up to work on the final project. Meeting regularly enabled us to work quickly and efficiently which helped us make good progress towards the completion of the project. Together, we were able to finish the project early due to the amount of work we were able to spread out amongst the previous several weeks.

What went right:

We were able to clearly define the application area of the database and what kind of data this database will hold. In addition, our clearly defined schedule for completing deliverables was followed strictly till the end. We started with the ER diagram which laid the foundation for our core database design. This was followed by the relation schema to clearly map the several relations. Since we were able to easily refer back to the schema, the transition to creating the database in SQL was an easy task to accomplish.

What went wrong:

We weren't able to spend as much time meeting together as we could have due to conflicting schedules from other classes. Oftentimes, we would have to delay meetings or reschedule for another time due to interruptions coming up and disrupting the meeting schedule we had planned, though we were still able to finish our tasks by the deadlines we set.

How would we do it differently if we were to do it again?

If we had to do this project again, we would have liked to have more data in our database so that we got to perform more powerful queries to attain interesting information. We would have spent more time on our proposal and have a clearer plan going into our final project. Our initial plan had functions that were too basic and did not really address any interesting data points. Once we realized that, we made changes to our plan to ensure that the SQL queries sought out information that would be useful.