Harmony (Music)

MIDI Appreciation Committee (Group 10)
Student 1 ID: Jesse Pingitore - jjp9217@rit.edu
Student 2 ID: Satvik Sharma - sss7060@rit.edu
Student 3 ID: Justin Swistak - jts7382@rit.edu
Student 4 ID: Ishan Shah - is4761@rit.edu

March 20, 2022

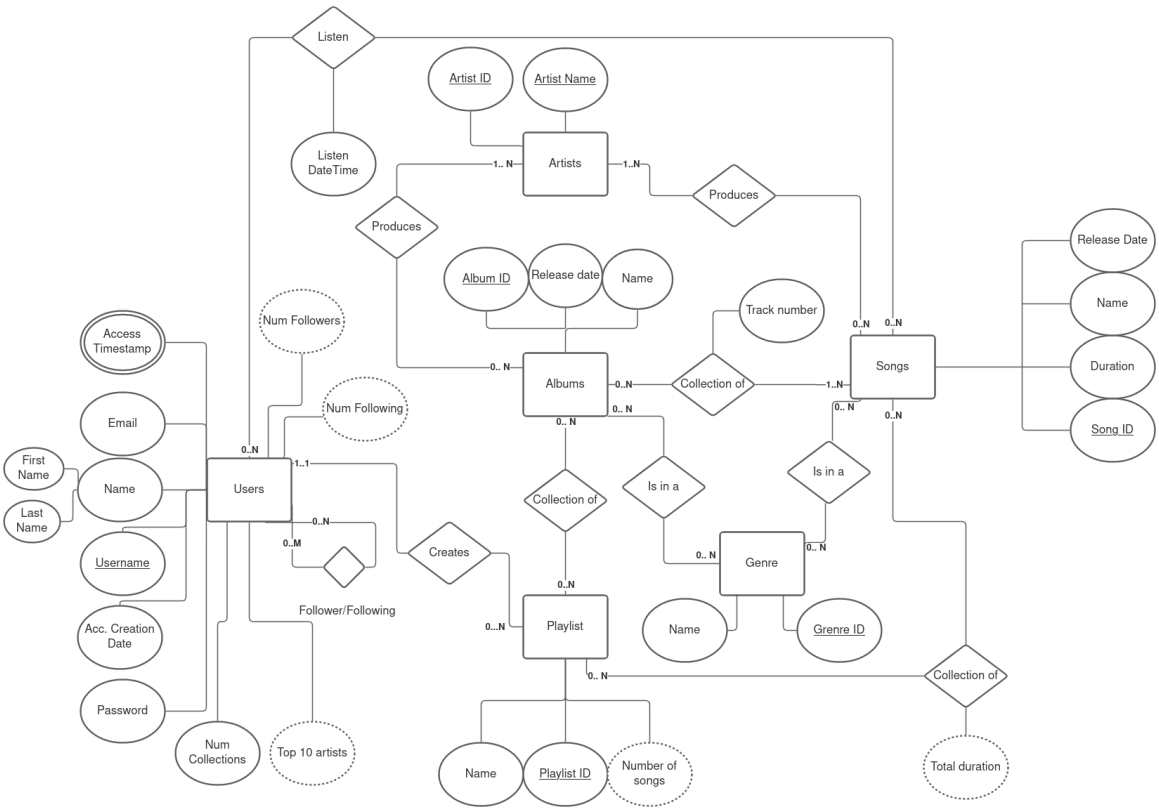
1 Introduction

Our project will be in the Music Domain, with a backup domain of movies. The application will be CLI-based, and will be essentially like a music information repository ala My Anime List. The language that will form the front-end for our application is Python, chosen due to its fast start-up time and small amount of required boilerplate code. The project will essentially host user profiles and song/album profiles, with users able to friend other users and create playlists. The completed project will be able to recommend songs to the user based on their song play history (user-submitted, this is not a music player) and song tags.

2 Design

2.1 Conceptual Model

When creating our diagram we first looked at the main entities we would need to represent our data. This was Users, Artists, Albums, Songs, and Playlists. We then discussed which attribute types these entities should have, based on how to best represent their real-life properties while also considering DB-implementable state. From there, we developed the relationships between


Listen
Artist ID
Artist Name
Listen DateTime
Artists
1.. N
1..N
Produces
Produces
Album ID
Release date
Name
Track number
Num Followers
Access Timestamp
Albums
0.. N
0..N
Collection of
Songs
1..N
Release Date
Name
Duration
Song ID
Email
Num Following
First Name
Name
Last Name
Users
0..N
1..1
Collection of
Is in a
Is in a
Username
0..N
0..M
Creates
Genre
Grenre ID
Acc. Creation Date
Follower/Following
Playlist
0...N
Name
Collection of
Password
Num Collections
Top 10 artists
Name
Playlist ID
Number of songs
Total duration

the entities. Artists make albums and songs, album contains songs, playlists contain songs and albums, users play songs, users create playlists, and users follow other users. We determined the multiplicity of those relations based on the required existence of the participants on both sides, for an album to exist there must exist at least one artist who created it, but an arbitrary amount of artists could have contributed to it, and an artist can have contributed to anywhere between 0 and many albums. Though albums and songs are at some point existentially linked to artists, they exist substantially independently to merit a proper relationship, rather than a weak relationship.

2.2 Reduction to tables

First, we created a table for each of the entities with their non-derived and single-valued attributes. We then created a separate table to match the multi-value attributes with the primary key from the entity they belong to. This was primarily a concern for the Songs, Albums, and Artists tables. Having separate tables to represent the relations between them (Artists to Songs, Artists to Albums) allows us to match artists to their works more flexibly than having a literal artist field in Song and Album. The same goes for matching Playlists to Songs, and Playlists to Albums.

Users(Username, Acc_Creation_Date, Password, First_Name, Last_Name, email)
Access_Timestamps(Timestamp_ID, Username, DateTime)
Listens(SongID, Username, DateTime)
Following(FollowerUsername, FollowingUsername)
Artists(Artist_ID, Artist_Name)
ArtistAlbumProduction(Artist_ID, Album_ID)
ArtistSongProduction(Artist_ID, Song_ID)
Songs(Song_ID, Release Date, Name, Duration)
Albums (Album_ID, Release_Date, Name)
Genre (Genre ID, Name)
Album Genre (Album ID, genre ID)
Song Genre (Song ID, genre ID)
Playlists(Playlist_ID, Username, Name)
Song_In_Album(Album_ID,Song_ID, track number)
CollectionAlbums(playlist_ID, Album_ID)
CollectionSongs(playlist_ID, Song_ID))

2.3 Data Requirements/Constraints

Use this section to list all the data domains and constraints that cannot be captured in your EER diagram but must be enforced by the database system. For example, there may be attribute types with a restricted domain, you must list those attribute types here and their domains. Similarly, attribute types with restrictions like uniqueness or required must be also listed here.

2.4 Sample instance data

Use this section to include sample of entities for every entity type in your EER diagram. Include also sample of relationships for every relationship type. For example, assume you have an entity type *Course* in your EER diagram with the attribute types *ID* and *name*. A sample of a *Course* entity can be *CSCI320, Principles of Data Management*.

Include 5 samples for every entity type and relationship type.

3 Implementation

Use this section to describe the overall implementation of your database. Include samples of SQL statements to create the tables (DDL statements) and a description of the ETL process, including examples of the SQL insert statements used to populate each table initially.

Include also sample of the SQL insert statements used in your application program to insert new data in the database. Finally, add an appendix of all the SQL statements created in your application during Phase 4 and a description of the indexes created to boost the performance of your application.

4 Data Analysis

4.1 Hypothesis

Use this section to state the objectives of your data analysis; what are the observations you are expecting to find. Note that your final observations may end up differing from your proposal, that is also a valid result.

4.2 Data Preprocessing

Use this section to describe the preprocessing steps you have performed to prepare the data for the analytics. Preprocessing steps may include: data cleaning (e.g., filling missing values, fixing outliers), formatting the data (e.g., resolving issues like inconsistent abbreviations, multiples date format in the data), combining or splitting fields, add new information (data enrichness).

Explain how the data was extracted from the database for the analysis; if you used complex queries or views, or both.

4.3 Data Analytics & Visualization

Use this section to explain the process/techniques used to analyze the data, use data visualization to present the results, and explain them.

4.4 Conclusions

Use this section to explain the conclusions drawn from your data analysis.

5 Lessons Learned

Use this section to describe the issues you faced during the project and how you overcame them. Also, describe what you learned during this effort; this section, like the others, plays a critical component in determining your final grade.

The next subsection is meant to provide you with some help in dealing with figures, tables and references, as these are sometimes hard for folks new to LaTeX. Your figures and tables may be distributed all over your paper (not just here), as appropriate for your paper.

Please delete the following subsection before you make any submissions!

Table 1: Feelings about Issues

Flavor	Percentage	Comments
Issue 1	10%	Loved it a lot
Issue 2	20%	Disliked it immensely
Issue 3	30%	Didn't care one bit
Issue 4	40%	Duh?

5.1 Tables, Figures, and Citations/References

Tables, figures, and references in technical documents need to be presented correctly. As many students are not familiar with using these objects, here is a quick guide extracted from the ACM style guide.

First, note that figures in the report must be original, that is, created by the student: please do not cut-and-paste figures from any other paper or report you have read or website. Second, if you do need to include figures, they should be handled as demonstrated here. State that Figure 1 is a simple illustration used in the ACM Style sample document. Never refer to the figure below (or above) because figures may be placed by LaTeX at any appropriate location that can change when you recompile your source *.tex* file. Incidentally, in proper technical writing (for reasons beyond the scope of this discussion), table captions are above the table and figure captions are below the figure. So the truly junk information about flavors is shown in Table 1.

Figure 1: A sample black & white graphic (JPG).

6 Resources

Include in this section the resources you have used in your project beyond the normal code development such as data sets or data analytic tools (i.e. Weka, R).