Project 1: Music Database

Harshil Desai & Kaveri Krishnaraj

The music database is designed to store the details of a music collection and live performance schedule of bands. The details include individual bands sorted by the genre of music they make, details about the band members, the instruments played by them, album collection, recording company that produces them and songs that are part of the album. The live performance schedule of the band at different cities is also stored.

Entity Sets

Album (A_ID, name, no. of tracks, release_year)
Recording Company(R_ID, name, founder),
Tracks(T_ID, title, length),
Band(B_ID, name, hometown, homepage),
Musician(M_ID, name, DOB, place of birth, active years, associated acts),
Membership(joined date, end date),
Place of live performance (P_ID, location, venue, ph. no),
Instrument(I_ID, name)

Business Rules

A band can make multiple albums. Each band can have 1, or multiple musicians and each musician's membership duration must be recorded. Each band contains some musician. An album could contain multiple songs and each album contains some song. A recording company could produce multiple albums and exactly one recording company must produce each album. Each band can perform live at 0,1, or multiple places and the date must be recorded. Each place is hosted by some band. A musician can play 0,1, or multiple instruments. Each instrument is played by some musician

Translating ER Diagram to a Relational Database

As a Band can have multiple musicians and their membership year needs to recorded , we combined the membership entity with the relationship member_of. So the Musician_Member_of_Band table contains the band_id and album_id as the foreign key which would

reference the Band and album table respectively. We also called the join_year as not null since very member of the band should have a membership year. The participation constraint from the band to musician can not be represented in the table.
The band_live_performance table has band_id as foreign key which references the band table. The participation constraint from the live_venue to band can not be represented in the table. A live_venue tuple can not be present without a name, so the name field is declared as not null.
Instrument_Played_by_Musician table has musician_id as foreign key which references musician table. The participation constraint from the instrument to musician cannot be represented in the table. Each instrument present in the table needs to be unique.
As a band can make multiple albums, the album table has band_id as foreign key, which references the band table.
As an album contains multiple tracks and a band can make multiple songs, the track table has album_id and band_id as a foreign key which references album table and band table respectively. The participation constraint from album to track table cannot be represented.
As only one recording company produces an album, the foreign key album_id is declared as not null and it references the album table.
The participation constraint from record company to album can not be represented.

Data acquisition approach

We obtained details regarding the various bands , albums and muscians from websites like spotify, itunes library, Wikipedia and IMDB. We also used the band's homepage to retrieve data.
We used insert statements with data to populate the relational schema.

User Interface

The user can interact with the database using a web application. The user would be able to choose a genre from a drop down where various genres would be listed. This action would lead to another page where the user would be able to pick a band from the drop down which would list bands belonging to the genre chosen. This action would lead the user to another page where the information about the band would be displayed. Here the user can view information about the band members and the live schedule of the band. The user can further pick a band member’s name from the drop down which would display more information of the band member. The user can also pick an album from the album drop down list. This action which would lead the user to another page displaying information about the album and the track list.