Maruthi Vemuri msv2130, Si Kai Lee sl3950

Description

The project is to replicate a database of a music streaming website e.g. Spotify, Pandora and Tidal while excluding some of the less salient and complicated features such as users . We have the following entities: Artist, Album, Song, Genre, Playlist, Label and Concert and the following relationships: contribute_to, comprises_of, has_signed, releases, contains and performs_at. Our most interesting finding is that we initially thought that we could aggregate artists and albums and the TA agreed with us but we realised the aggregated entity could not model all the relations accurately We also found that we did not need to include weak entities and the at most relation.

We hope to scrape data from databases such as Discogs, Gracenote and MusicBrainz to populate all fields of the database apart from concert information. We can then obtain concert information from sites such as Songkick which maintains a list of concerts for each artist and genre information.

The users would access the database from a website that would be similar in features to Spotify and they will be able to search for all single entities listed above and a combination of artist and song, artist and album, artist and label, artist and concert and album and label. If time permits, we would implement a simple recommendation system for the website. Users would be able to add data to all fields of the website but deletion of artists, songs, albums and genre would be carried out by moderators. Lastly, playlists would be freely created and maintained by the users.

Both of us will not drop the course as it will serve our Systems requirement, therefore, there is no contingency plan.

Primary Keys

1. Key for attribute Song is s_id.
2. Key for attribute Genre is g_id.
3. Key for attribute Labels is l_id.
4. Key for attribute Artist is a_id.
5. Key for attribute Album is al_id.
6. Key for attribute Concert is c_id.
7. Key for attribute Playlist is p_id.

Constraints (Key: **Bold** means at least once, Underline exactly one)

1. Each artist must contribute to at least one song -> (**artist**, album, **song**)
2. Each album comprises of at least one song -> (album, **song**)
3. Each playlist contains at least one song -> (playlist, **song**)
4. Each concert must have at least one performing artist -> (concert, **artist**)
5. Each song belongs to at least one genre -> (song, **genre**)
6. Each artist must have signed on to only one label -> (artist, label)
7. Every label has to have released at least one album -> (label, **album**)

ER-> SQL

```
CREATE TABLE Artist{
a_id int,
a_name text,
a_year date,
a_country text,
PRIMARY KEY(a_id)
}

CREATE TABLE Album{
al_id int,
al_name text,
al_dur time,
al_#_songs int,
PRIMARY KEY(al_id)
}

CREATE TABLE Song{
s_id int,
s_name text,
s_no int,
s_dur time,
s_compose text,
PRIMARY KEY(s_id)
}
```

```
CREATE TABLE Genre{
g_id int,
g_name text,
PRIMARY KEY(g_id)
}

CREATE TABLE Label{
l_id int,
l_name text,
PRIMARY KEY(l_id)
}

CREATE TABLE Playlist{
p_id int,
p_name text,
p_time date,
p_#_songs int,
p_dur time,
p_user text,
PRIMARY KEY(p_id)
}

CREATE TABLE Concert{
c_time time,
c_name text,
c_date date,
c_loc text,
PRIMARY KEY(c_loc, c_date, c_time)
}

CREATE TABLE performs_at{
a_id int,
c_time time,
c_date date,
c_loc text,
PRIMARY KEY(a_id, c_loc, c_date, c_time),
FOREIGN KEY(a_id) REFERENCES Artist,
FOREIGN KEY(c_loc, c_date, c_time) REFERENCES Concert
}
```

```
CREATE TABLE contributes_to{
a_id int NOT NULL,
s_id int,
al_id int,
PRIMARY KEY(s_id),
FOREIGN KEY(a_id) REFERENCES Artist,
FOREIGN KEY(s_id) REFERENCES Song,
FOREIGN KEY(al_id) REFERENCES Album
}

CREATE TABLE belongs_to{
s_id int,
g_id int NOT NULL,
PRIMARY KEY(s_id),
FOREIGN KEY(g_id) REFERENCES Genre,
FOREIGN KEY(s_id) REFERENCES Song
}

CREATE TABLE release{
al_id int,
l_id int,
r_year date,
PRIMARY KEY(l_id,al_id),
FOREIGN KEY(al_id) REFERENCES Album,
FOREIGN KEY(l_id) REFERENCES Label
}

CREATE TABLE has_signed{
l_id int NOT NULL,
a_id int,
PRIMARY KEY(a_id),
FOREIGN KEY(l_id) REFERENCES Labels
}

CREATE TABLE contains{
p_id int,
s_id int,
PRIMARY KEY(p_id, s_id),
FOREIGN KEY(p_id) REFERENCES Playlists,
FOREIGN KEY(s_id) REFERENCES Songs
}
```