CSCI 466 - Final Project
MySqueal
Description / Relational Schema

Entities: Contributor, Song, Karaoke File, Queue, John Winans

Contributor: A contributor is for each person who has somehow worked on a song and has been properly credited for it. Whether they are the artist who recorded the music, the people/person who wrote the song, the people/person who produced it, sound engineers, mastered it, mixed it, etc. They are stored in this database solely for reasons about the song and its file. This is important as there can be multiple versions of a song.

Song: This entity represents the song's existence in the database to be used for the karaoke session. The song has an identifier to identify a song. This is important as we need to be able to differentiate between song versions: there could be a regular version, a remixed version, or a duetted version of the song to work between two people. It also has a genre, whether it is a pop song, a country song, or an R&B song. The song also has a title. It is also to note that it may be possible for songs to have the same title but still be different versions of that same song. Our identifier will help us in this distinction. Length will also be important since you would like to know how long you will be singing the listed song.

Karaoke File: A Karaoke file holds the song to be used for a karaoke session. The file has a title and an identifier to ensure the correct karaoke file is being chosen for the correct song.

Queue: The queue is the order at which a song is added. If someone wanted to sing a song first and someone chose something after, that second person would get added after them. This entity represents the 2 queues in our system: The normal queue and the priority queue. The normal queue will function like a regular line say at a register in a store: first come, first serve. The priority queue works more like a line at an amusement park: If you paid a certain amount, you get to move up in the line instead of waiting. The queue will have the price of a song along with the name of the user to input into the queue.

John Winans: John Winans

Relationships:

Contributor-to-Song (M-M)

The relationship between Contibutor to Song relationship represents the people who have made the song and created it having their credits tied to the song. This relationship is many-to-many because there could be as little as one person to have worked on a song or there could be multiple people who have contributed to the creation of a song.

Song-to-Karaoke File (1-1)

This relationship is for the Song and Karaoke File, since each song has a single karaoke file, is 1 to 1. A single song does not have multiple karaoke files as it would get too complicated to differentiate which to use.

Song-to-Queue (M-M)

The relationship between a Song and the Queue is many to many. This is because one or multiple songs can get added to one or multiple queues. For example: two songs are added to the Normal and Priority Queue. One gets added to the normal and the other gets added to the priority. We could also add two songs to either queue by themselves. Since the queue is designed to take multiple songs at once, this is required.

Relational Schema

Contributor(Name)
Song(SongID, Genre, Title, Length)
KaraokeFile(FileID, Title)
Queue(QueueID, SongPrice, UsersName)

To convert this, we will look at the relationships between the entities.

Contributor(Name)
Song(SongID, Genre, Title, Length)
Creates(SongID† ,Name†)

As this is a binary many-to-many relationship, we really create a new relation to represent the relationship between the two.

We will also do this with our Song-Queue relationship, since it has the same relationship.

Contributor(Name)
Song(SongID, Genre, Title, Length)
Queue(QueueID, SongPrice, UsersName)
Creates(SongID†, Name†)
AddsTo(SongID†, QueueID†)

Since Song and Karaoke have a one-to-one relationship, we give each entity a foreign key, finishing our relational scheme

Contributor(Name)
Song(SongID, Genre, Title, Length, FileID†)
Queue(QueueID, SongPrice, UsersName)

KaraokeFile(FileID, Title, SongID†)
Creates(SongID†, Name†)
AddsTo(SongID†, QueueID†)