Nick Weinert and David Passaro
CS 340 Project Step 2
4/14/19

URL: http://flip2.engr.oregonstate.edu:27934

Peer Review Feedback

During the peer review sessions, we received limited feedback throughout the term. Our initial HTML design was messy, and it was suggested that our html design could be more concise and clearer while limiting the options for insert and update. It was also suggested that we implement more straightforward filtering. We implemented this feedback. It was also suggested that we create a many-to-many relationship table between artists and genres. We decided not to implement this feedback and recreated the process through a filter functionality instead.

Upgrades to the Draft version

Our upgrades were inspired by the peer review session.

Song Database

Project Outline

We will be creating a database modeling the music industry. We will model songs, artists, labels and genres. They will be interconnected via logical relationships representing the single or multiple artists which created the song and the various genres in which they are encompassed. Since there are so many songs, genres, artist, etc in the world it will be possible to make a robust database with all the information regarding a specific song.

Verbal Database Outline

The entities include:

- Song- The song is the basic entity and will connect with all other entities. Its attributes include ○ id: This number is automatically assigned to each song as it is entered. This will be the auto-incrementing primary key.

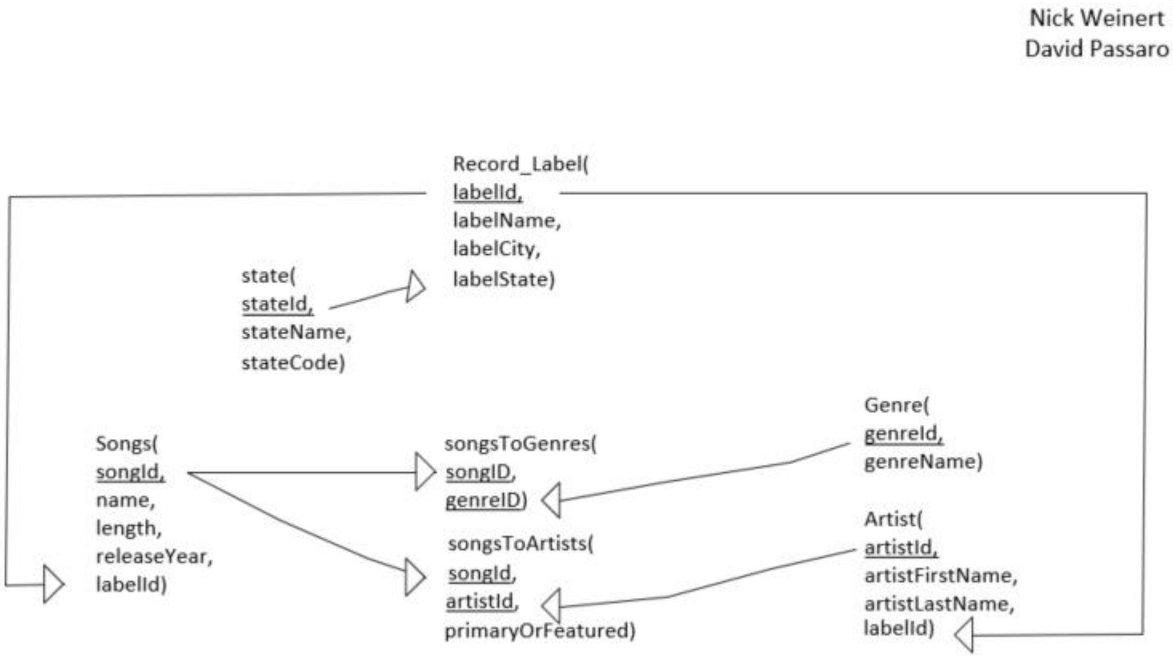
    - labelId: This is the unique identifier for which label the song belongs to. This references the labelId field from the Record_Label table and may be blank.
    - name: The song name will be a maximum of 50 characters and is without default and cannot be blank.
    - length: The song length is a time attribute representing total playtime of the song and is without default and cannot be blank. It uses the time() data type.
    - releaseYear: Release year as an year attribute, year song was released and is without default and can be blank. It uses the int() data type due to Maria DB's restriction on year data type values.

- Genre: The musical genre in which a song belongs. This is linked to the song through a many-to-many relationship table.
    - id: This number is automatically assigned to each genre. This will be the autoincrementing unique identifier.
    - name: The genre name will be a maximum of 50 characters and is without default and cannot be blank.
- Artist: The artist or artists which wrote or participated in the song. There may be one or more artists per song. This is linked to the song through a many-to-many relationship table.
    - id: This number is automatically assigned to each artist. This will be the autoincrementing unique identifier.
    - labelId: This is the unique identifier for which label the artist belongs to. This references the labelId field from the Record_Label table and may be blank.
    - firstName: The artist first name will be a maximum of 50 characters and is without default and cannot be blank. This field will also be used as the band name if the artist is a band and not a singular person.
    - lastName: The artist last name will be a maximum of 50 characters and is without default and can be blank if the artist only has one common name or is a band (ie. Madonna).
- Record_Label: The label which produced the song. A song may or may not have come from a record label so there may not be entries for every song.
    - labelId: This number is automatically assigned to each label. This will be the autoincrementing unique identifier.
    - labelName: The labelName is the name of the record label, it will be a maximum of 50 characters and is without default and can be blank.
    - labelCity: The labelCity is the city in which the record label is headquartered. It will be a maximum of 50 characters and may be blank.
    - labelState: The labelState is the state in which the record label is headquartered. It will be a foreign key reference to the state table.
- state: The state table produces to possible states that a user can choose from. It is simply for input validation purposes. No adding or removing can be done to the state table through the web interface
    - stateId: This number is automatically assigned to each state. This will be the autoincrementing unique identifier.
    - stateName: The stateName is the name of the record state, it will be a maximum of 128 characters and is without default and is not blank.
    - stateCode: The stateCode is the two-letter unique and accepted state code.

The relationships include:

- **Songs are from artists:** many songs may be from many artists-> many to many relationship. One attribute of this relationship is whether the artist is the primary or featured artist on the song. This attribute will be stored as a simple integer.
- **Songs have genres:** many songs may be from many genres -> many to many relationship
- **Songs are from record labels:** Many songs may be from only zero to one labels-> one to many relationship
- **Artists work for labels:** Many artists come from zero to one labels-> one to many relationship
- **Labels are in states:** Each label has an associated state value

Additionally, Artists perform songs inside of genres. Although this relationship could be viewed as a many-to-many relationship, we have decided to implement this relationship with JOINS instead of creating a new table.



Nick Weinert
David Passaro

Record_Label(
labelId,
labelName,
labelCity,
labelState)

state(
stateId,
stateName,
stateCode)

Songs(
songId,
name,
length,
releaseYear,
labelId)

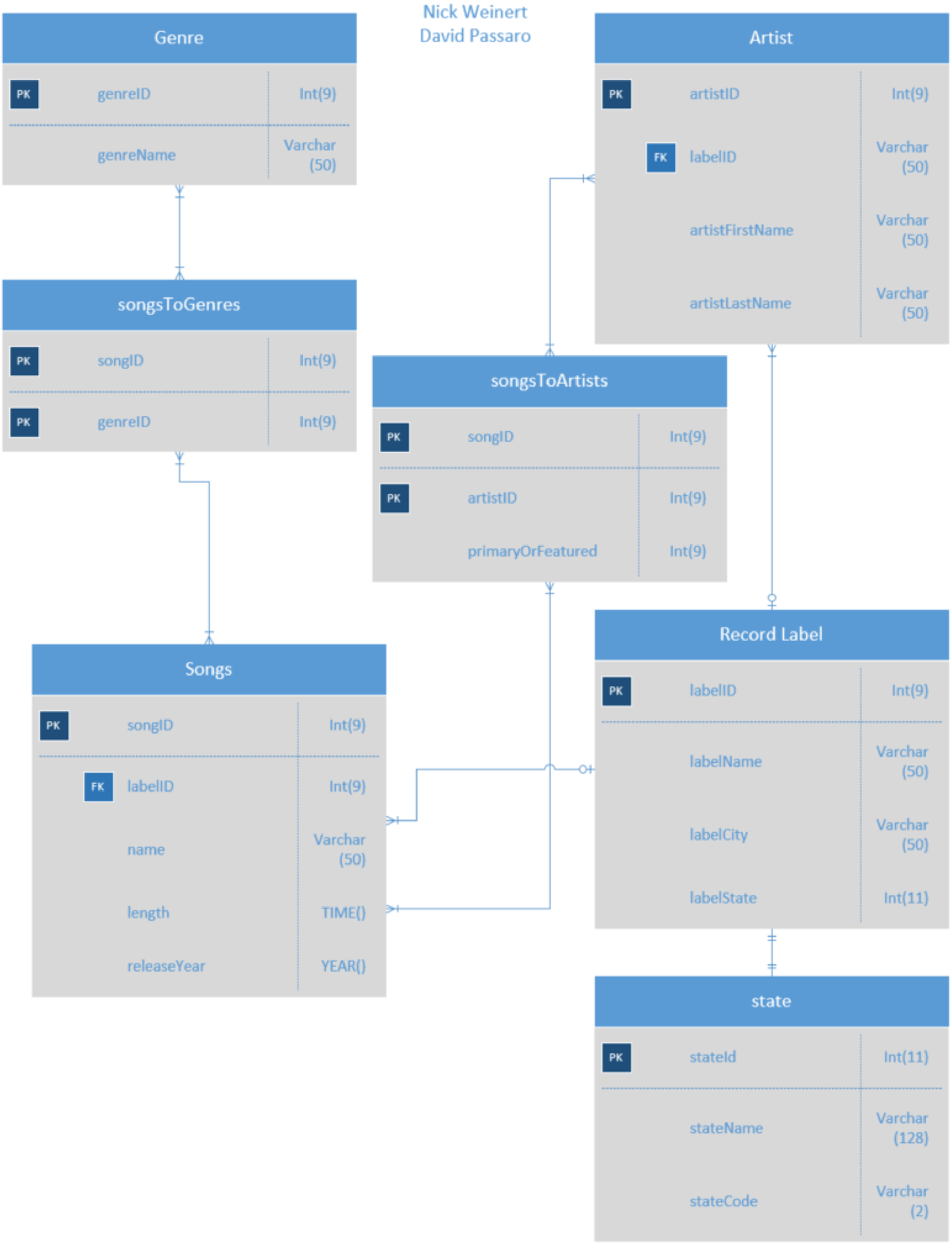
songsToGenres(
songID,
genreID)

songsToArtists(
songId,
artistId,
primaryOrFeatured)

Genre(
genreId,
genreName)

Artist(
artistId,
artistFirstName,
artistLastName,
labelId)

Nick Weinert and David Passaro
CS 340 Project and Database Outline
5/30/19



Nick Weinert
David Passaro

Genre

PK	genreID	Int(9)
	genreName	Varchar (50)

songsToGenres

PK	songID	Int(9)
PK	genreID	Int(9)

Songs

PK	songID	Int(9)
FK	labelID	Int(9)
	name	Varchar (50)
	length	TIME()
	releaseYear	YEAR()

Artist

PK	artistID	Int(9)
FK	labelID	Varchar (50)
	artistFirstName	Varchar (50)
	artistLastName	Varchar (50)

songsToArtists

PK	songID	Int(9)
PK	artistID	Int(9)
	primaryOrFeatured	Int(9)

Record Label

PK	labelID	Int(9)
	labelName	Varchar (50)
	labelCity	Varchar (50)
	labelState	Int(11)

state

PK	stateId	Int(11)
	stateName	Varchar (128)
	stateCode	Varchar (2)