Kevin Bowers, Noah Brothers, Rebecca Shyu

1) ***Problem Statement*** (revised from Phase 1)

In a world, where access to music has become easier and much less expensive, the consumer is demanding newer, more innovative, and more efficient ways to stream music and learn more about their favorite artists. This application expands upon the features implemented by several of the most popular music streaming services, such as Apple Music and Spotify. In addition to allowing the user to search for music and add songs to playlists, the application also allows them to search for tours that are happening, and learn more about the artists and the record labels that have signed them. Ultimately, this application aims to enhance the overall experience of music listeners in an ever-increasingly more digital and gratifying age.

2) ***Conceptual Database Design*** (revised from Phase 1)

5 entity sets:

- Album
    - Album ID
    - Name
    - Release date
    - Artist Name
- Artist (Subclass: Solo/Band)
    - Name (Assume no artist/band has same name)
    - Date started in music
    - Age (derived)
    - Social Media Handles (Instagram, Twitter, Facebook)
- Upcoming Tour (Weak set/relationship)
    - Name
    - Dates
        - Length (Derived)
    - Location
- Record Label
    - Name (Assume trademarked)
    - Address
    - Phone Number
    - CEO
- Song (Weak set/relationship)
    - Name (Partial Key) along with album ID
    - Length
    - Genre
- Playlist
    - Name (Assume each playlist has a different name)
    - Length (derived)
    - Date created
    - Creator Name

- Assumptions:
  No artist/band has the same name
  Assume every label name is trademarked (i.e. no two labels have the same name)
  Assume playlists are uniquely named
  Assume every song is in an album (no singles)
  Assume every artist has at least one album
  Assume every artist belongs to a label
  Assume an album only has one artist
  Assume an artist only has one tour at a time

3) ***Logical Database Design*** (revised from Phase 2)

We originally were deducted 5 points, but Professor Lin gave us those points back so we did not revise anything.

Album

album_id	name	release_date	album_artist_name

Artist

name	date	solo_flag	band_flag	label_name

ArtistSocials

name	insta	twitter	facebook

UpcomingTour

tour_name	artist

TourStop

tour_name	date	location

RecordLabel

label_name	address	phone_num	ceo

Song

song_name	album_id	length	genre

Playlist

playlist_name	date_created	creator_name

PlaylistSong

playlist_name	song_name	album_id

Tables of Attribute Types

Album

Attribute	Type	Description
album_ID	Integer	Unique Album ID
name	Variable length string	Album Name
release_date	Date	Release Date of Album
album_artist_name	Variable length string	Name of artist of album

Artist

Attribute	Type	Description
name	Variable length string	Artist name
date	Date	Date of first release
solo_flag	boolean	flag to say if artist is a solo person
band_flag	boolean	Flag to say is artist is a band
label_name	Variable length string	Name of label of artist

ArtistSocials

Attribute	Type	Description
name	Variable length string	Artist Name
insta	Variable length string	Social Media handle (instagram)
twitter	Variable length string	Social Media handle (twitter)
facebook	Variable length string	Social Media handle (facebook)

UpcomingTour

Attribute	Type	Description
tour_name	Variable length string	Unique Tour Name
artist	Variable length string	Name of artist performing tour

TourStop

Attribute	Type	Description
tour_name	Variable length string	Unique Tour Name
date	Date	Date of stop
location	Variable length string	Location of stop

RecordLabel

Attribute	Type	Description
label_name	Variable length string	Unique name of label
address	Variable length string	Address of record label
phone_num	10 character string	Main number for record label
ceo	Variable length string	Name of label's CEO

Song

Attribute	Type	Description
song_name	Variable length string	Name of song
album_id	Integer	Unique Album ID
length	Integer	Length of song in seconds
genre	Variable length string	Genre of song

Playlist

Attribute	Type	Description
playlist_name	Variable length string	Unique name of playlist
date_created	Date	Date playlist was created
creator_name	Variable length string	Name of creator of playlist

PlaylistSong

Attribute	Type	Description
playlist_name	Variable length string	Unique name of playlist
song_name	Variable length string	Name of song
album_id	Integer	Unique Album ID

4) ***Application Program Design*** (revised from Phase 2)

Search_for_song

//This function searches for a song. It accesses the "Song" table.
Input: song name.
Steps:
(1) Retrieve "song_name" that matches the user's input
(2) Display all "song_name"s with album_id

Add_song_to_playlist

//This function adds a song to an existing playlist. It accesses the "Song", "Album", and "Playlist, and "PlaylistSong" tables.
Input: name of song, name of album, and name of target playlist.
Steps:
(1) Retrieve album ID based on name of album
(2) Join "Song" and "Album" tables based on albumid
(3) Check if playlist exists in "Playlist" table

(4) If it exists, find playlist based on user input for name of target playlist
(5) Insert “song_name” and “album_id” into “PlaylistSong”

Find_tour_length

//This function finds the total length of an artist’s tour
Input: Tour name
Steps:
(1) Find upcoming tour based on tour name (user input)
(2) Join upcoming tour and tour stop tables based on tourname
(3) Get the first date from the joined tables
(4) Get the last date from the joined tables
(5) Calculate the difference in dates

Find_artist_socials

//This function returns the social media handles for a given artist
Input: Artist name
Steps:
(1) Check to make sure artist with given name exists in the Artist table
(2) Get social media handles based on ArtistName

Search_tour_info

//This function searches for tour dates and locations based on the name of a tour
Input: Tour name
Steps:
(1) Match the input to a tour name from the UpcomingTour table
(2) Join the UpcomingTour and TourStop tables based on tourname
(3) Get the date/location tuples from the joined tables

Find_date_by_location

//This function finds tours and tour dates given a specific location
Input: Location and/or tour name
(1) Get dates from the TourStop table with the given location and/or name of the tour

Get_all_songs by same album

//This function gets all songs that are on the same album
Input: Album name
Steps:
(1) Get the albumid from the Album table given the user’s input
(2) Select all songs where the albumid is the output of step 1

Get_all_artists from record label

//This function gets all artists who are connected with a given record label
Input: Name of record label
Steps:

(1) Select all artists whose label is the given label name

Add_song

//This function adds a given song to the database from an external source.

Input: Song name, Album ID, Length of song, genre

(1) User specifies all four attributes in Song relation.
(2) Function checks to make sure none of the values are NULL, and that the Album ID specified exists in its primary key location in Album.
(3) Function inserts input as a tuple into Song, using INSERT INTO clause.

Delete_song

//This function deletes a given song from the database.

Input: Song name, Album ID

(1) User inputs the song name and album ID.
(2) The function searches the Song table for a tuple with the given values for SongName and AlbumID.
(3) If a match is not found, a message is output to the user saying that the action cannot be completed.
(4) If a match is found, the entire tuple is deleted using DELETE clause.

Modify_Tour

//This function can modify information about a tour stop.

Input: Attribute to be updated in TourStop (Tour name, date, and/or location of tour)

(1) The user is prompted to select which fields they would like to modify, and for which tour.
(2) The user is prompted to input values for the attributes in TourStop that they selected.
(3) The function checks to make sure the input values match the specified domain constraints. If not, an error message is output and the user is prompted to enter values again.
(4) If all input values have valid domain, modifies the tuple using UPDATE.

Find_total_time

//This function finds the total time length of a given playlist.

Input: Name of the playlist

(1) User inputs the name of the playlist they would like to find total time for.
(2) Function makes sure the playlist with the given name exists in the database. If not, rejects the operation.
(3) Function joins the Song, Playlist, and PlaylistSong relations, using join conditions SongName=SongName, PlaylistName=PlaylistName, and AlbumID=AlbumID. Then the function sums the lengths of the terms using SUM.
(4) The output is displayed.

Create_playlist

//This function creates an empty playlist in Playlist.

(1) The user is asked to specify a playlist name, the date they are creating it and their name.

(2) The function checks to make sure that none of the values are NULL and that they match the domain constraints. If not, an error message is output and the operation is rejected.
(3) The function inserts the new playlist into the Playlist relation using INSERT INTO clause.

5) ***User Manual***: This is for end-users who may not have database knowledge. Describe precisely how to use your system step by step with screenshots of your system interface and sample outputs.

-- straight-up SQL

To maintain the database, values can be inserted, modified, and deleted from the database using a variety of SQL commands and functions. To insert a value into a given table, one can use the INSERT INTO SQL command, with the VALUES being those specified. This should be done in the create_and_insert.sql file. An example of inserting the values for Lizzo as an artist is shown in the screenshots

To find the date of a tour given the location, the user will first input a location. Then, the query to find the date of a tour is executed. The tour name and tour stop date for all tours with the given location are then retrieved.

To get all of the songs on a given album, the user will need to know the album ID. They will input this. The query will then execute, and the output returned will be all of the songs on the album with this album ID.

To get all artists who are signed to a record label, the user will need to know the name of the record label. They will input this record label name, the query will execute, and it will return all artists who are signed to this record label.

To search for a given song, the user only needs to know the name of the song. They will input this, and the query will execute. The information that will be output will include the song name, its album ID in the database, its length, and the genre. If the user wants to search for a song of a given genre, they just need to input the given genre and all songs in that genre will be returned as the output. To search by album, the user will need to know the album ID. They will input this, and it will be output alongside the song names on the album and the genre.

To insert a song into a given playlist, the user will need to know the name of the playlist they are trying to insert into, the name of the song, and its album ID. The user will input all 3 of these values, and the database will first search for all songs with the given name. Then, it will search for songs of that name whose album ID is the one input. If found, the song will be inserted into the playlist.

To find the length in days between two tour dates, both tour dates need to be input by the user. The datediff function is then used to find the difference in days between these two dates. The output is this difference.

To obtain all of the social media accounts of a given artist, the user will need to input the artist's name. If the artist with an input name exists in the database, the database will return their Instagram, Twitter, and Facebook handles in that order. Otherwise, nothing will be returned.

To delete a song from the database, the user just needs to input the name of a song that already exists. The database will then delete all songs with the given name.

To create an empty playlist, the INSERT INTO command must be used. The user needs only provide the name of a new playlist, as well as their name, and the time it was created will automatically be generated using the now() function.