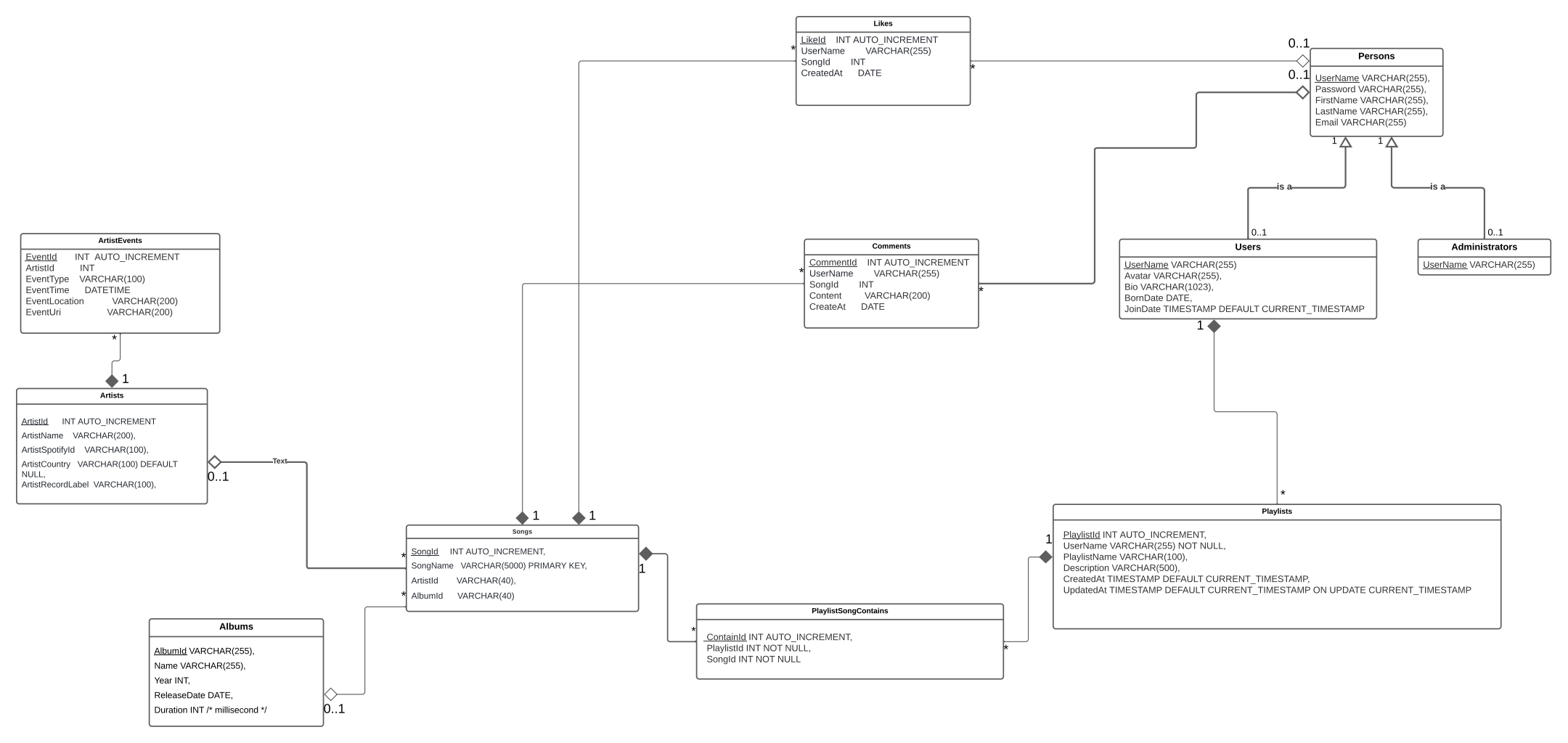
Likes
LikeId INT AUTO_INCREMENT
UserName VARCHAR(255)
SongId INT
CreatedAt DATE
Persons
UserName VARCHAR(255),
Password VARCHAR(255),
FirstName VARCHAR(255),
LastName VARCHAR(255),
Email VARCHAR(255)
is a
is a
Users
UserName VARCHAR(255)
Avatar VARCHAR(255),
Bio VARCHAR(1023),
BornDate DATE,
JoinDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP
Administrators
UserName VARCHAR(255)
Comments
CommentId INT AUTO_INCREMENT
UserName VARCHAR(255)
SongId INT
Content VARCHAR(200)
CreateAt DATE
ArtistEvents
EventId INT AUTO_INCREMENT
ArtistId INT
EventType VARCHAR(100)
EventTime DATETIME
EventLocation VARCHAR(200)
EventUri VARCHAR(200)
Artists
ArtistId INT AUTO_INCREMENT
ArtistName VARCHAR(200),
ArtistSpotifyId VARCHAR(100),
ArtistCountry VARCHAR(100) DEFAULT NULL,
ArtistRecordLabel VARCHAR(100),
Text
Songs
SongId INT AUTO_INCREMENT,
SongName VARCHAR(5000) PRIMARY KEY,
ArtistId VARCHAR(40),
AlbumId VARCHAR(40)
Albums
AlbumId VARCHAR(255),
Name VARCHAR(255),
Year INT,
ReleaseDate DATE,
Duration INT /* millisecond */
PlaylistSongContains
ContainId INT AUTO_INCREMENT,
PlaylistId INT NOT NULL,
SongId INT NOT NULL
Playlists
PlaylistId INT AUTO_INCREMENT,
UserName VARCHAR(255) NOT NULL,
PlaylistName VARCHAR(100),
Description VARCHAR(500),
CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
UpdatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP

SQL script:

Notes: The UPDATE statements at the bottom are used to build the foreign-key references from Songs to Artists, and from Songs to Albums. Currently our UPDATE statements cannot run due to a "Lock Wait Timeout Exceeded Error", and we haven't fixed it yet. However, all the other CREATE and LOAD Statements are working well and satisfy the homework requirements.

```
CREATE SCHEMA IF NOT EXISTS MusiCraze;
USE MusiCraze;

DROP TABLE IF EXISTS PlaylistSongContains;
DROP TABLE IF EXISTS Playlists;
DROP TABLE IF EXISTS Comments;
DROP TABLE IF EXISTS Likes;
DROP TABLE IF EXISTS Songs;
DROP TABLE IF EXISTS ArtistEvents;
DROP TABLE IF EXISTS Administrators;
DROP TABLE IF EXISTS Users;
DROP TABLE IF EXISTS Albums;
DROP TABLE IF EXISTS Artists;
DROP TABLE IF EXISTS Persons;

CREATE TABLE Persons (
   UserName VARCHAR(255),
   `Password` VARCHAR(255),
   FirstName VARCHAR(255),
   LastName VARCHAR(255),
   Email VARCHAR(255),
   CONSTRAINT pk_Persons_UserName PRIMARY KEY (UserName)
);


CREATE TABLE Artists (
   ArtistId  INT AUTO_INCREMENT,
   ArtistName  VARCHAR(200),
   ArtistSpotifyId  VARCHAR(100),
   ArtistCountry  VARCHAR(100) DEFAULT NULL,
   ArtistRecordLabel  VARCHAR(100),
   CONSTRAINT pk_Artists_ArtistId PRIMARY KEY (ArtistId)
);


CREATE TABLE Albums (
```

```
  Name VARCHAR(5000),
  AlbumId VARCHAR(100),
  AlbumSpotifyId VARCHAR(100),
  Year INT,
  ReleaseDate DATE,
  Duration INT, /* millisecond */

  CONSTRAINT Pk_albums_album_id
    PRIMARY KEY (AlbumId)
);

CREATE TABLE Users (
    UserName VARCHAR(255),
    Avatar VARCHAR(255),
    Bio VARCHAR(1023),
    BornDate DATE,
    JoinDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    CONSTRAINT pk_Users_UserName PRIMARY KEY (UserName),
    CONSTRAINT fk_Users_UserName FOREIGN KEY (UserName)
       REFERENCES Persons(UserName)
       ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE Administrators (
    UserName VARCHAR(255),
    CONSTRAINT pk_Administrators_UserName PRIMARY KEY (UserName),
    CONSTRAINT fk_Administrators_UserName FOREIGN KEY (UserName)
       REFERENCES Persons(UserName)
       ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE ArtistEvents (
          EventId  INT AUTO_INCREMENT,
    ArtistId  INT,
    EventType  VARCHAR(100),
    EventTime   DATETIME,
    EventLocation  VARCHAR(200),
    EventUri   VARCHAR(200),
    CONSTRAINT pk_ArtistEvents_EventId PRIMARY KEY (EventId),
          CONSTRAINT fk_ArtistEvents_ArtistId FOREIGN KEY (ArtistId)
                REFERENCES Artists(ArtistId)
                ON UPDATE CASCADE ON DELETE CASCADE
```

```
);


# use ArtistSpotifyId & AlbumSpotifyId to bridge ArtistId & AlbumId
CREATE TABLE Songs (
        SongId  INT AUTO_INCREMENT,
        SongName  VARCHAR(5000) NOT NULL,
   SpotifyId VARCHAR(100),
        ArtistSpotifyId  VARCHAR(100),
   ArtistId  INT,
   AlbumSpotifyId VARCHAR(100),
   AlbumId INT,
 CONSTRAINT pk_Songs_SongId PRIMARY KEY (SongId),
 CONSTRAINT fk_Songs_ArtistId FOREIGN KEY (ArtistId)
        REFERENCES Artists(ArtistId)
        ON UPDATE CASCADE ON DELETE SET NULL
);


CREATE TABLE Likes(
   LikeId INT NOT NULL AUTO_INCREMENT,
   UserName VARCHAR(255),
   SongId INT NOT NULL,
   CreatedAt DATE,
   CONSTRAINT pk_Likes_LikeId PRIMARY KEY(LikeId),
   CONSTRAINT fk_Likes_UserName FOREIGN KEY(UserName)
                REFERENCES Persons(UserName)
                ON UPDATE CASCADE ON DELETE SET NULL,
   CONSTRAINT fk_Likes_SongId FOREIGN KEY(SongId)
                REFERENCES Songs(SongId)
                ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE Comments(
   CommentId INT NOT NULL AUTO_INCREMENT,
   UserName VARCHAR(255),
   SongId INT NOT NULL,
   Content VARCHAR(200),
   CreatedAt DATE,
   CONSTRAINT pk_Comments_CommentId PRIMARY KEY(CommentId),
   CONSTRAINT fk_Comments_UserName FOREIGN KEY(UserName)
                REFERENCES Persons(UserName)
                ON UPDATE CASCADE ON DELETE SET NULL,
```

```
   CONSTRAINT kf_Comments_SongId FOREIGN KEY(SongId)
                  REFERENCES Songs(SongId)
                  ON UPDATE CASCADE ON DELETE CASCADE
);

CREATE TABLE Playlists(
  PlaylistId INT AUTO_INCREMENT,
  UserName VARCHAR(255) NOT NULL,
  PlaylistName VARCHAR(100),
  Description VARCHAR(500),
  # With a DEFAULT clause but no ON UPDATE CURRENT_TIMESTAMP clause,
  # the column has the given default value and is NOT automatically updated to the current
timestamp.
  CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

  # With both DEFAULT CURRENT_TIMESTAMP and ON UPDATE CURRENT_TIMESTAMP,
  # the column has the current timestamp for its default value and
  # is automatically updated to the current timestamp.
  # Reference: https://dev.mysql.com/doc/refman/8.0/en/timestamp-initialization.html
  UpdatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,

  CONSTRAINT pk_Playlists_PlaylistId
    PRIMARY KEY (PlaylistId),
  CONSTRAINT fk_Playlists_UserName
    FOREIGN KEY (UserName)
    REFERENCES Users(UserName)
    ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE PlaylistSongContains(
  ContainId INT AUTO_INCREMENT,
  PlaylistId INT NOT NULL,
  SongId INT NOT NULL,
  CONSTRAINT pk_PlayListSongContains_ContainId
    PRIMARY KEY (ContainId),
  CONSTRAINT fk_PlayListSongContains_PlaylistId
    FOREIGN KEY (PlaylistId)
    REFERENCES Playlists(PlaylistId)
    ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT fk_PlayListSongContains_SongId
    FOREIGN KEY (SongId)
    REFERENCES Songs(SongId)
    ON UPDATE CASCADE ON DELETE CASCADE
```

```
);

LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/persons.csv' INTO TABLE Persons
   FIELDS TERMINATED BY ','
   ENCLOSED BY '"'
   LINES TERMINATED BY '\n'
   IGNORE 1 LINES;


LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/artist.csv'
INTO TABLE Artists
          FIELDS TERMINATED BY ',' ENCLOSED BY '"'
          LINES TERMINATED BY '\n'
          IGNORE 1 LINES (ArtistName,ArtistSpotifyId);

LOAD DATA LOCAL
                    INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/album.csv'
                    INTO TABLE Albums FIELDS TERMINATED BY ',' ENCLOSED BY '"'
                    LINES TERMINATED BY '\n'
      IGNORE 1 LINES (Name, AlbumId, Year, ReleaseDate, Duration);

LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/users.csv'
INTO TABLE Users
   FIELDS TERMINATED BY ','
   ENCLOSED BY '"'
   LINES TERMINATED BY '\n'
   IGNORE 1 LINES;

LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/administrators.csv' INTO TABLE Administrators
   FIELDS TERMINATED BY ','
   ENCLOSED BY '"'
   LINES TERMINATED BY '\n'
   IGNORE 1 LINES;

###### ============ LOAD ARTIST EVENTS =========
```

```
LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/songs.csv'
INTO TABLE Songs
        FIELDS TERMINATED BY ',' ENCLOSED BY '"'
        LINES TERMINATED BY '\r\n'
        IGNORE 1 LINES (SongId,SongName,SpotifyId,ArtistSpotifyId,AlbumSpotifyId);


LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/playlists.cs
v' INTO TABLE Playlists
  FIELDS TERMINATED BY ',' ENCLOSED BY '"'
        LINES TERMINATED BY '\n'
        IGNORE 1 LINES (UserName, PlaylistName, Description, CreatedAt, UpdatedAt);


LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/playlistson
gcontains.csv' INTO TABLE PlaylistSongContains
  FIELDS TERMINATED BY ','    # Don't need ENCLOSED BY. CSV contains only numbers, and
therefore datas are not quoted.
        LINES TERMINATED BY '\n'
        IGNORE 1 LINES (PlaylistId, SongId);

UPDATE Songs s, Artists a
        SET s.ArtistId = a.ArtistId
    WHERE s.ArtistSpotifyId = a.ArtistSpotifyId;

UPDATE Songs s, Albums a
        SET s.AlbumId = a.AlbumId
    WHERE s.AlbumSpotifyId = a.AlbumSpotifyId;
```

Row counts of each table: