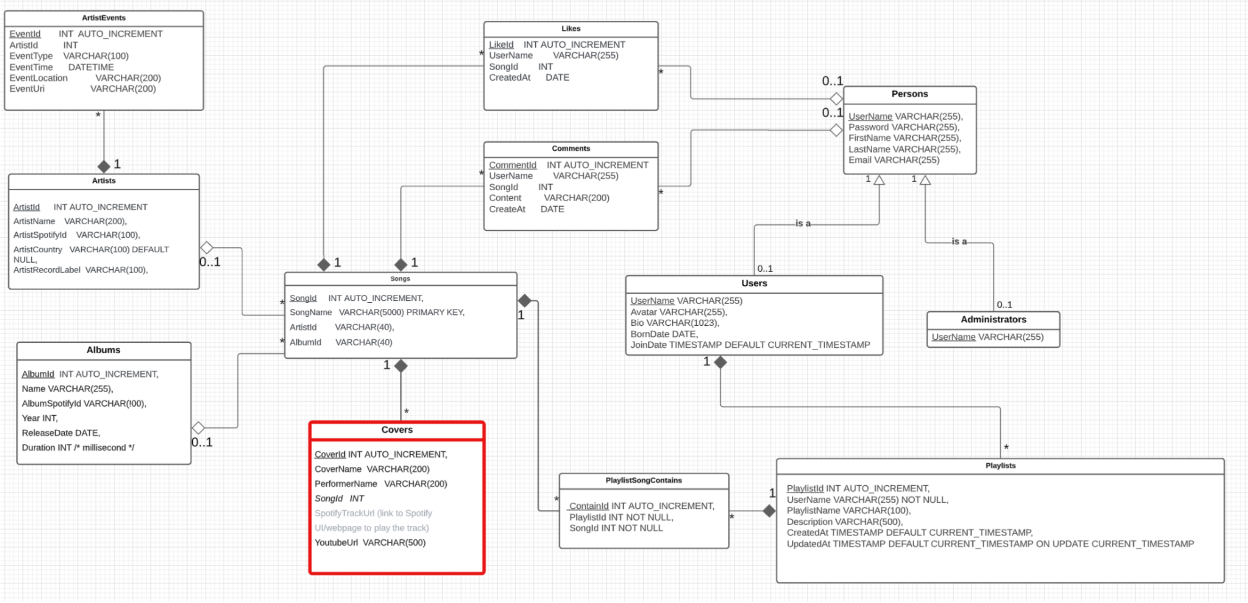
Group 3 PM3: Business Insights

Updated UML:

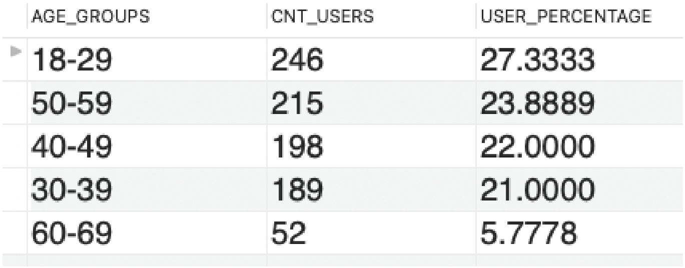
A new class “Cover” is added. It represents the cover performances of a song.


ArtistEvents
EventId INT AUTO_INCREMENT
ArtistId INT
EventType VARCHAR(100)
EventTime DATETIME
EventLocation VARCHAR(200)
EventUri VARCHAR(200)
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)
Comments
CommentId INT AUTO_INCREMENT
UserName VARCHAR(255)
SongId INT
Content VARCHAR(200)
CreateAt DATE
Artists
ArtistId INT AUTO_INCREMENT
ArtistName VARCHAR(200),
ArtistSpotifyId VARCHAR(100),
ArtistCountry VARCHAR(100) DEFAULT NULL,
ArtistRecordLabel VARCHAR(100),
Songs
SongId INT AUTO_INCREMENT,
SongName VARCHAR(5000) PRIMARY KEY,
ArtistId VARCHAR(40),
AlbumId VARCHAR(40)
Users
UserName VARCHAR(255)
Avatar VARCHAR(255),
Bio VARCHAR(1023),
BornDate DATE,
JoinDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP
Administrators
UserName VARCHAR(255)
Albums
AlbumId INT AUTO_INCREMENT,
Name VARCHAR(255),
AlbumSpotifyId VARCHAR(!00),
Year INT,
ReleaseDate DATE,
Duration INT /* millisecond */
Covers
CoverId INT AUTO_INCREMENT,
CoverName VARCHAR(200)
PerformerName VARCHAR(200)
SongId INT
SpotifyTrackUrl (link to Spotify UI/webpage to play the track)
YoutubeUrl VARCHAR(500)
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
is a
is a
0..1
1
*

Business Insight Questions:

1. Which age group uses MusicCraze the most?

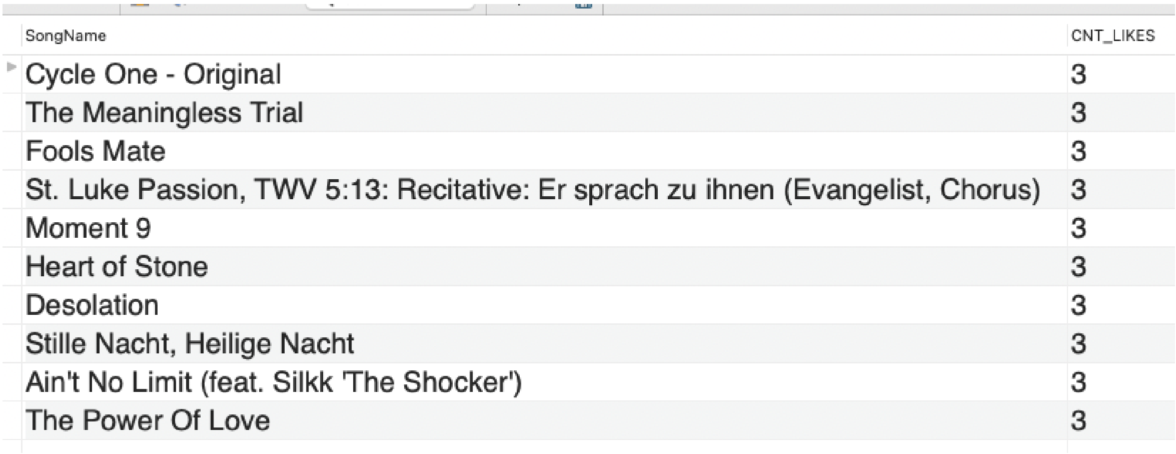
```
SELECT RESULTS.AGE_GROUPS, COUNT(*) AS CNT_USERS, 100 * COUNT(*)/(SELECT COUNT(*)
FROM Users) AS USER_PERCENTAGE
FROM (
  SELECT
    CASE
      WHEN TIMESTAMPDIFF(YEAR, BornDate, CURDATE()) BETWEEN 18 AND 29 THEN '18-29'
      WHEN TIMESTAMPDIFF(YEAR, BornDate, CURDATE()) BETWEEN 30 AND 39 THEN '30-39'
      WHEN TIMESTAMPDIFF(YEAR, BornDate, CURDATE()) BETWEEN 40 AND 49 THEN '40-49'
      WHEN TIMESTAMPDIFF(YEAR, BornDate, CURDATE()) BETWEEN 50 AND 59 THEN '50-59'
      WHEN TIMESTAMPDIFF(YEAR, BornDate, CURDATE()) BETWEEN 60 AND 69 THEN '60-69'
      ELSE 'Others'
    END AS AGE_GROUPS
  FROM Users
) AS RESULTS
GROUP BY RESULTS.AGE_GROUPS
ORDER BY CNT_USERS DESC;
```



AGE_GROUPS	CNT_USERS	USER_PERCENTAGE
18-29	246	27.3333
50-59	215	23.8889
40-49	198	22.0000
30-39	189	21.0000
60-69	52	5.7778

2. What top 10 songs have the most likes among users joined in 2020?

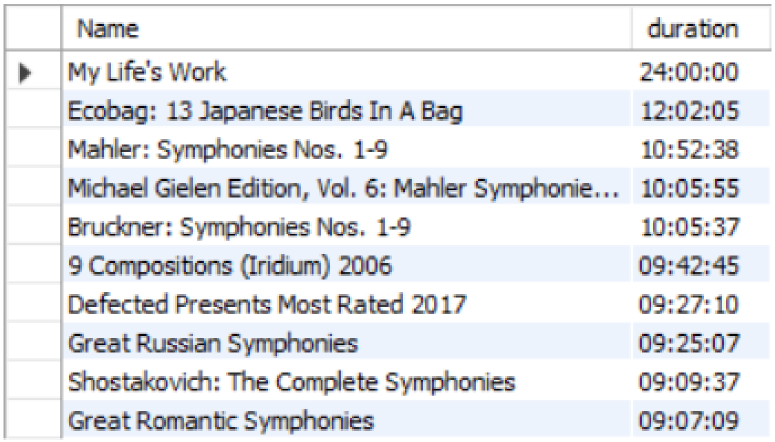
```
SELECT Songs.SongName, RESULTS.CNT_LIKES
FROM Songs INNER JOIN (
   SELECT Likes.SongId, COUNT(*) AS CNT_LIKES
   FROM Users INNER JOIN Likes ON Users.UserName = Likes.UserName
   WHERE YEAR(Users.JoinDate) = 2020
   GROUP BY Likes.SongId
   ORDER BY CNT_LIKES DESC) AS RESULTS
ON Songs.SongId = RESULTS.SongId
LIMIT 10;
```



SongName	CNT_LIKES
Cycle One - Original	3
The Meaningless Trial	3
Fools Mate	3
St. Luke Passion, TWV 5:13: Recitative: Er sprach zu ihnen (Evangelist, Chorus)	3
Moment 9	3
Heart of Stone	3
Desolation	3
Stille Nacht, Heilige Nacht	3
Ain't No Limit (feat. Silkk 'The Shocker')	3
The Power Of Love	3

3. Top 10 albums with longest duration?

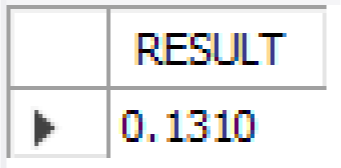
```
SELECT albums.Name, sec_to_time(floor(duration/1000)) AS duration FROM albums
          GROUP BY albums.AlbumId
   ORDER BY albums.duration DESC LIMIT 10;
```



	Name	duration
▶	My Life's Work	24:00:00
	Ecobag: 13 Japanese Birds In A Bag	12:02:05
	Mahler: Symphonies Nos. 1-9	10:52:38
	Michael Gielen Edition, Vol. 6: Mahler Symphonie...	10:05:55
	Bruckner: Symphonies Nos. 1-9	10:05:37
	9 Compositions (Iridium) 2006	09:42:45
	Defected Presents Most Rated 2017	09:27:10
	Great Russian Symphonies	09:25:07
	Shostakovich: The Complete Symphonies	09:09:37
	Great Romantic Symphonies	09:07:09

4. Percentage of users having their own playlist?

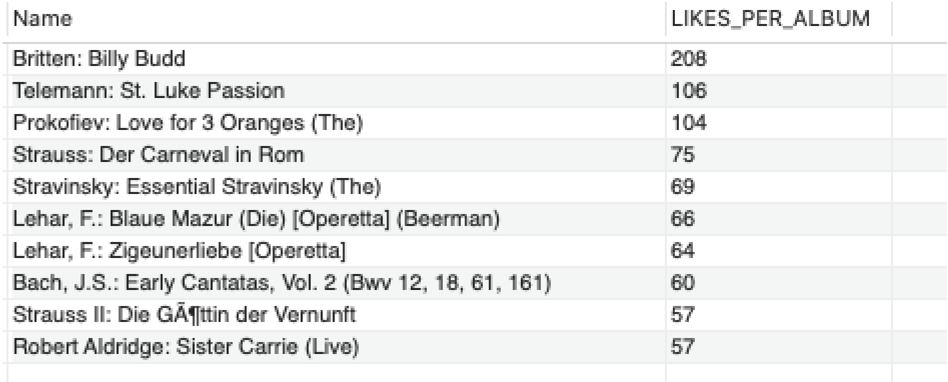
```
SELECT (PLAYLIST_COUNT / USER_COUNT) AS RESULT FROM
        (SELECT COUNT(*) AS USER_COUNT FROM persons) AS A
        CROSS JOIN
        (SELECT COUNT(*) AS PLAYLIST_COUNT FROM playlists) AS B;
```



	RESULT
▶	0.1310

5. What are the Top 10 liked albums (defined by # likes received by songs)?

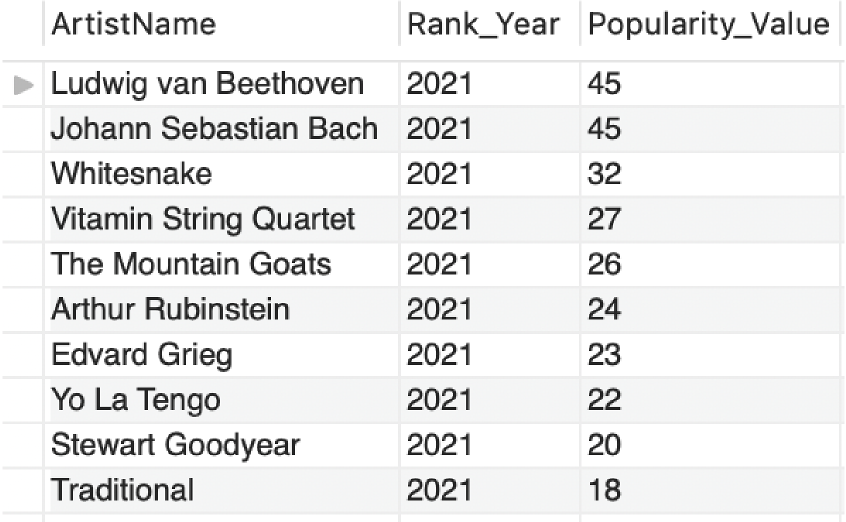
```
SELECT Albums.Name, SUM(LIKES_PER_SONG.CNT) AS LIKES_PER_ALBUM
  FROM Albums
  INNER JOIN
  (
    SELECT Songs.SongId, Songs.AlbumId, COUNT(*) AS CNT
      FROM Songs
          INNER JOIN Likes
          On Songs.SongId = Likes.SongId
      GROUP BY Songs.SongId
      ORDER BY CNT DESC
    ) AS LIKES_PER_SONG
  ON Albums.AlbumId = LIKES_PER_SONG.AlbumId
  GROUP BY Albums.AlbumId, Albums.Name
  ORDER BY LIKES_PER_ALBUM DESC
LIMIT 10;
```



Name	LIKES_PER_ALBUM
Britten: Billy Budd	208
Telemann: St. Luke Passion	106
Prokofiev: Love for 3 Oranges (The)	104
Strauss: Der Carneval in Rom	75
Stravinsky: Essential Stravinsky (The)	69
Lehar, F.: Blaue Mazur (Die) [Operetta] (Beerman)	66
Lehar, F.: Zigeunerliebe [Operetta]	64
Bach, J.S.: Early Cantatas, Vol. 2 (Bwv 12, 18, 61, 161)	60
Strauss II: Die GÃ¶ttin der Vernunft	57
Robert Aldridge: Sister Carrie (Live)	57

6. What are the Top 10 popular artists in 2021?

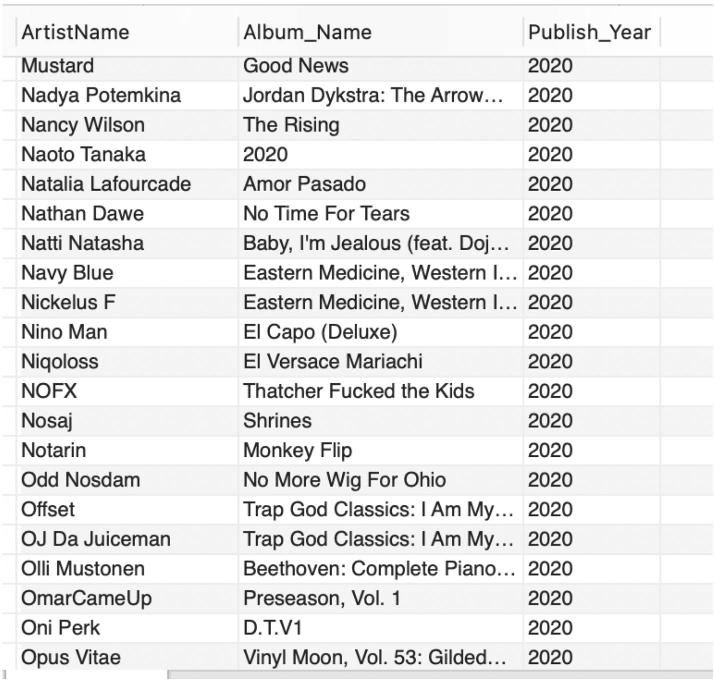
```
SELECT Artists.ArtistName, SONG_LIKES.Rank_Year, SUM(SONG_LIKES.CNT_LIKES) AS
Popularity_Value
FROM Artists INNER JOIN (
SELECT Songs.SongId, Songs.ArtistId, YEAR(Likes.CreatedAt) AS Rank_Year, COUNT(*) AS
CNT_LIKES
        FROM Songs INNER JOIN Likes ON Songs.SongId = Likes.SongId
        WHERE YEAR(Likes.CreatedAt) = 2021
        GROUP BY Songs.SongId, Likes.CreatedAt
        ORDER BY CNT_LIKES DESC) AS SONG_LIKES
ON Artists.ArtistId = SONG_LIKES.ArtistId
GROUP BY Artists.ArtistName, SONG_LIKES.Rank_Year
ORDER BY Popularity_Value DESC
LIMIT 10;
```



ArtistName	Rank_Year	Popularity_Value
Ludwig van Beethoven	2021	45
Johann Sebastian Bach	2021	45
Whitesnake	2021	32
Vitamin String Quartet	2021	27
The Mountain Goats	2021	26
Arthur Rubinstein	2021	24
Edvard Grieg	2021	23
Yo La Tengo	2021	22
Stewart Goodyear	2021	20
Traditional	2021	18

7. Which Artists published new Album in 2020?

```
SELECT Artists.ArtistName, ALBUM_PUBLISH.Album_Name, ALBUM_PUBLISH.Publish_Year
FROM Artists INNER JOIN (
SELECT Albums.Name AS Album_Name, Albums.YEAR AS Publish_Year, Songs.ArtistId
        FROM Albums INNER JOIN Songs ON Songs.AlbumId = Albums.AlbumId
        WHERE Albums.YEAR = 2020
   GROUP BY Albums.Name, Songs.ArtistId) AS ALBUM_PUBLISH
ON Artists.ArtistId = ALBUM_PUBLISH.ArtistId
GROUP BY Artists.ArtistName, ALBUM_PUBLISH.Album_Name;
```

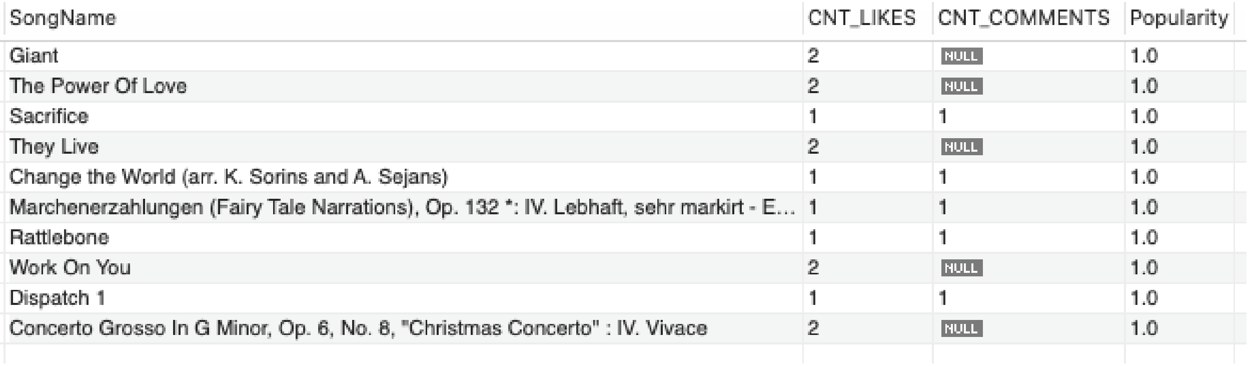


ArtistName	Album_Name	Publish_Year
Mustard	Good News	2020
Nadya Potemkina	Jordan Dykstra: The Arrow…	2020
Nancy Wilson	The Rising	2020
Naoto Tanaka	2020	2020
Natalia Lafourcade	Amor Pasado	2020
Nathan Dawe	No Time For Tears	2020
Natti Natasha	Baby, I'm Jealous (feat. Doj…	2020
Navy Blue	Eastern Medicine, Western I…	2020
Nickelus F	Eastern Medicine, Western I…	2020
Nino Man	El Capo (Deluxe)	2020
Niqoloss	El Versace Mariachi	2020
NOFX	Thatcher Fucked the Kids	2020
Nosaj	Shrines	2020
Notarin	Monkey Flip	2020
Odd Nosdam	No More Wig For Ohio	2020
Offset	Trap God Classics: I Am My…	2020
OJ Da Juiceman	Trap God Classics: I Am My…	2020
Olli Mustonen	Beethoven: Complete Piano…	2020
OmarCameUp	Preseason, Vol. 1	2020
Oni Perk	D.T.V1	2020
Opus Vitae	Vinyl Moon, Vol. 53: Gilded…	2020

8. What are the top 10 popular songs this month?(Combine their number of likes and comments with weight 1:1)

```
SELECT Songs.SongName, LikesResult.CNT_LIKES, CommentsResult.CNT_COMMENTS,
        (IFNULL(LikesResult.CNT_LIKES, 0) * 0.5 +
        IFNULL(CommentsResult.CNT_COMMENTS, 0) * 0.5) AS Popularity
FROM Songs
LEFT JOIN(
        SELECT Likes.SongId, COUNT(*) AS CNT_LIKES
        FROM Users INNER JOIN Likes
        ON Users.UserName = Likes.UserName
        WHERE MONTH(Likes.CreatedAt) = MONTH(CURRENT_DATE())
        GROUP BY Likes.SongId) AS LikesResult
ON Songs.SongId = LikesResult.SongId
```

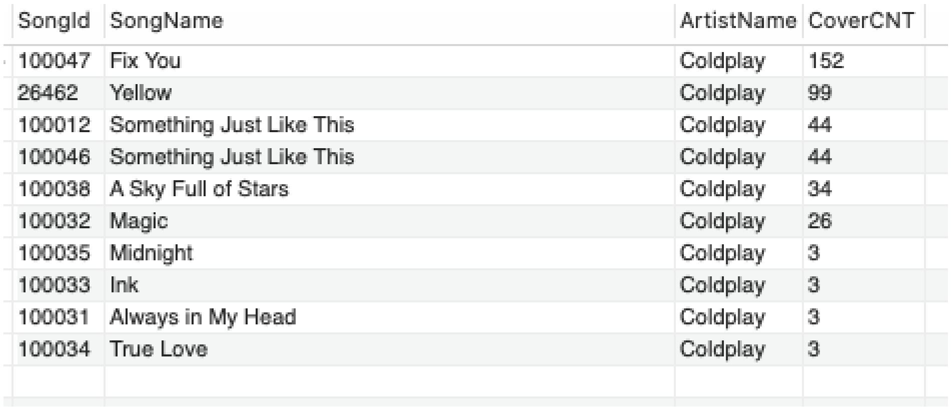
```
LEFT JOIN(
        SELECT Comments.SongId, COUNT(*) AS CNT_COMMENTS
        FROM Users INNER JOIN Comments
        ON Users.UserName = Comments.UserName
        WHERE MONTH(Comments.CreatedAt) = MONTH(CURRENT_DATE())
        GROUP BY Comments.SongId) AS CommentsResult
ON Songs.SongId = CommentsResult.SongId
ORDER BY Popularity DESC
LIMIT 10;
```



SongName	CNT_LIKES	CNT_COMMENTS	Popularity
Giant	2	NULL	1.0
The Power Of Love	2	NULL	1.0
Sacrifice	1	1	1.0
They Live	2	NULL	1.0
Change the World (arr. K. Sorins and A. Sejans)	1	1	1.0
Marchenerzahlungen (Fairy Tale Narrations), Op. 132 *: IV. Lebhaft, sehr markirt - E…	1	1	1.0
Rattlebone	1	1	1.0
Work On You	2	NULL	1.0
Dispatch 1	1	1	1.0
Concerto Grosso In G Minor, Op. 6, No. 8, "Christmas Concerto" : IV. Vivace	2	NULL	1.0

9. What are the 10 most covered songs by "Coldplay"?

```
SELECT TargetArtistSongs.SongId, SongName, ArtistName, COUNT(*) AS CoverCNT
FROM
    (
    SELECT SongId, SongName, TargetArtist.ArtistName
    FROM Songs
        INNER JOIN
        (SELECT ArtistId, ArtistName FROM Artists
         WHERE ArtistName = 'Coldplay') AS TargetArtist
        ON Songs.ArtistId = TargetArtist.ArtistId
    ) AS TargetArtistSongs
    INNER JOIN  Covers
    ON TargetArtistSongs.SongId = Covers.SongId
GROUP BY TargetArtistSongs.SongId, SongName, ArtistName
ORDER BY CoverCNT DESC
LIMIT 10;
```

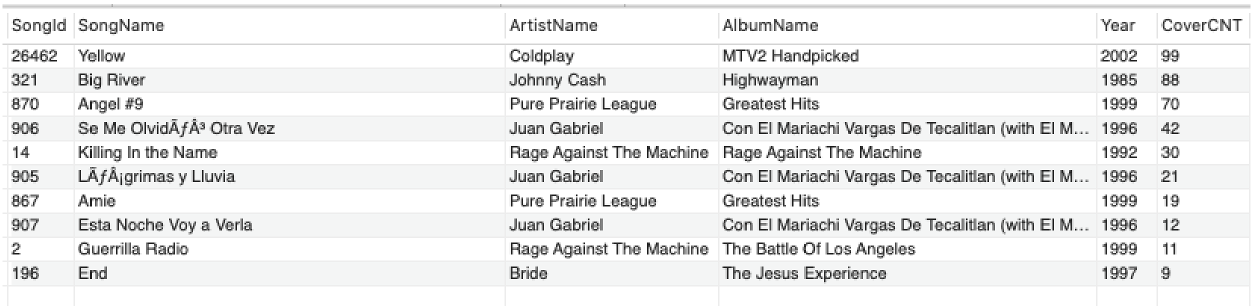


SongId	SongName	ArtistName	CoverCNT
100047	Fix You	Coldplay	152
26462	Yellow	Coldplay	99
100012	Something Just Like This	Coldplay	44
100046	Something Just Like This	Coldplay	44
100038	A Sky Full of Stars	Coldplay	34
100032	Magic	Coldplay	26
100035	Midnight	Coldplay	3
100033	Ink	Coldplay	3
100031	Always in My Head	Coldplay	3
100034	True Love	Coldplay	3

Result Annotation:
Songs with the same title can be in different albums of the same artist, and therefore they have different SongId. Meanwhile, our Covers Table includes the covers information for each song with a unique SongId in Songs Table. Therefore we have two "Something Just Like This" in our result.

10. What are the 10 most covered songs released between 1990 to 2021? (Show the ArtistName as well)

```
SELECT TargetYearSongs.SongId, SongName, ArtistName, AlbumName, Year, COUNT(*) AS CoverCNT
FROM
    (
      SELECT SongId, SongName, ArtistName, TargetYearAlbums.Name AS AlbumName, Year
      FROM
        (SELECT AlbumId, Name, Year FROM Albums
         WHERE Year <= 2021 and Year >= 1980) AS TargetYearAlbums
        INNER JOIN Songs
        ON Songs.AlbumId = TargetYearAlbums.AlbumId
        INNER JOIN Artists
        ON Songs.ArtistId = Artists.ArtistId
    ) AS TargetYearSongs
    INNER JOIN Covers
    ON Covers.SongId = TargetYearSongs.SongId
GROUP BY TargetYearSongs.SongId, SongName, ArtistName, AlbumName, Year
ORDER BY CoverCNT DESC
LIMIT 10;
```



SongId	SongName	ArtistName	AlbumName	Year	CoverCNT
26462	Yellow	Coldplay	MTV2 Handpicked	2002	99
321	Big River	Johnny Cash	Highwayman	1985	88
870	Angel #9	Pure Prairie League	Greatest Hits	1999	70
906	Se Me OlvidÃƒÂ³ Otra Vez	Juan Gabriel	Con El Mariachi Vargas De Tecalitlan (with El M…	1996	42
14	Killing In the Name	Rage Against The Machine	Rage Against The Machine	1992	30
905	LÃƒÂ¡grimas y Lluvia	Juan Gabriel	Con El Mariachi Vargas De Tecalitlan (with El M…	1996	21
867	Amie	Pure Prairie League	Greatest Hits	1999	19
907	Esta Noche Voy a Verla	Juan Gabriel	Con El Mariachi Vargas De Tecalitlan (with El M…	1996	12
2	Guerrilla Radio	Rage Against The Machine	The Battle Of Los Angeles	1999	11
196	End	Bride	The Jesus Experience	1997	9

Result Annotation:

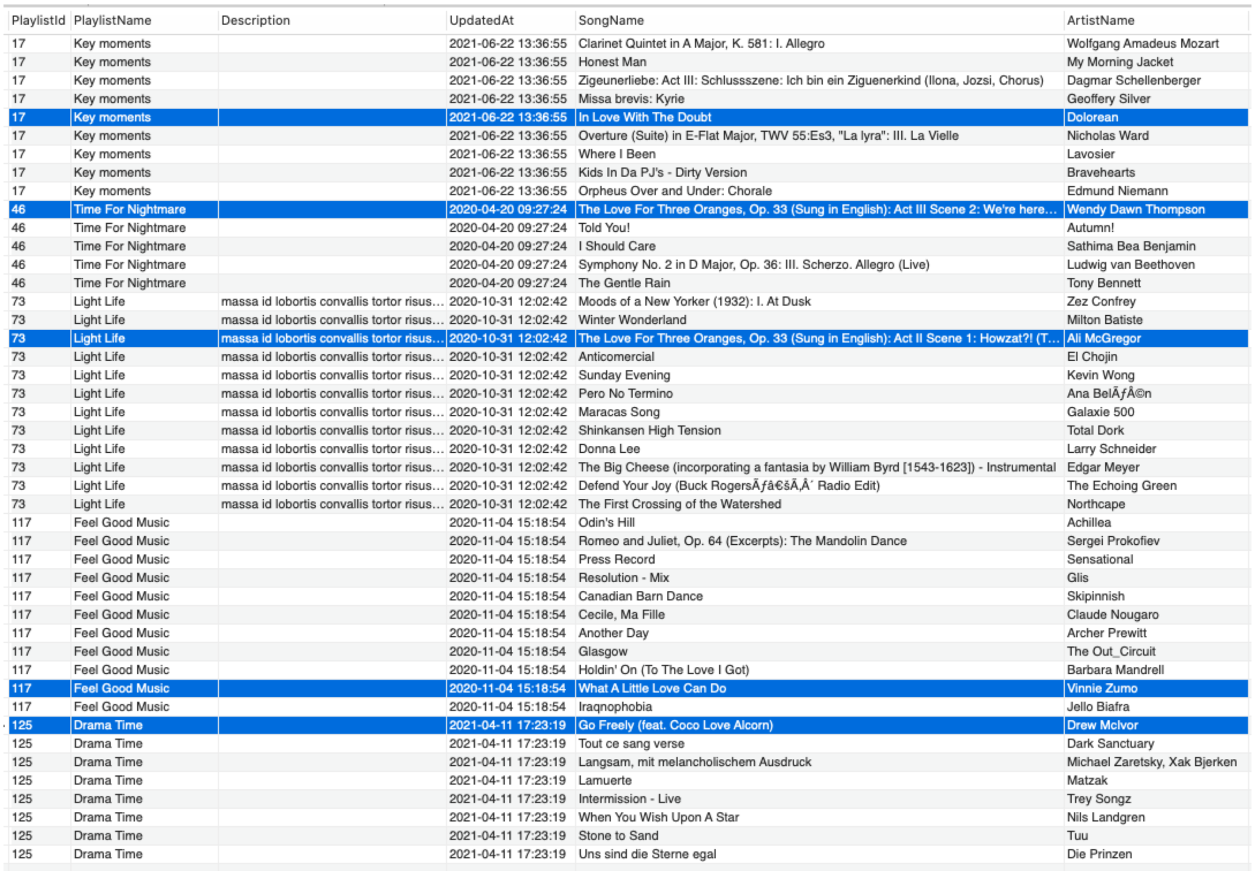
1) Currently, our data in “Covers” TABLE only include the cover performances of the first 1000 songs and artist Coldplay’s songs in “Songs” TABLE.
2) The first 1000 songs in “Songs” TABLE are all released around 1990s
3) In “Songs” TABLE, the “AlbumId” column is still NULL for all of Coldplay’s albums(except for the song “Yellow” with SongId 26462). Thus, the intermediate table TargetYearAlbums won’t include Coldplay’s albums.
4) Due to the above reasons: The “year” of songs in the result list mostly fall into the 1990s. And it contains only one song of ‘Coldplay’: ‘Yellow’.

11. Find all playlists that contain the song with a title of "xxx" by artist "yyy". Show the playlist's name, description, updated date, and all the other songs that are added into these playlists.

(The below SQL statements show the results for the song with a title that contains “love” and by artists whose names contain letter ‘o’.)

```
SELECT TargetPlaylistsInfo.PlaylistId AS PlaylistId, PlaylistName, Description, UpdatedAt,
Songs.SongName, ArtistName
FROM
(
  SELECT Playlists.PlaylistId AS PlaylistId, PlaylistName, Description, UpdatedAt
  FROM
    (SELECT PlaylistId, Songs.SongId, SongName, ArtistName
      FROM PlaylistSongContains
        INNER JOIN Songs
          ON PlaylistSongContains.SongId = Songs.SongId
        INNER JOIN Artists
          ON Artists.ArtistId = Songs.ArtistId
      WHERE SongName LIKE '% love %' AND ArtistName LIKE '%o%') AS TargetPlaylists
    INNER JOIN Playlists
      ON TargetPlaylists.PlaylistId = Playlists.PlaylistId
) AS TargetPlaylistsInfo
INNER JOIN PlaylistSongContains
  ON TargetPlaylistsInfo.PlaylistId = PlaylistSongContains.PlaylistId
```

```
INNER JOIN Songs
  ON PlaylistSongContains.SongId = Songs.SongId
INNER JOIN Artists
  ON Songs.ArtistId = Artists.ArtistId
ORDER BY PlaylistId;
```

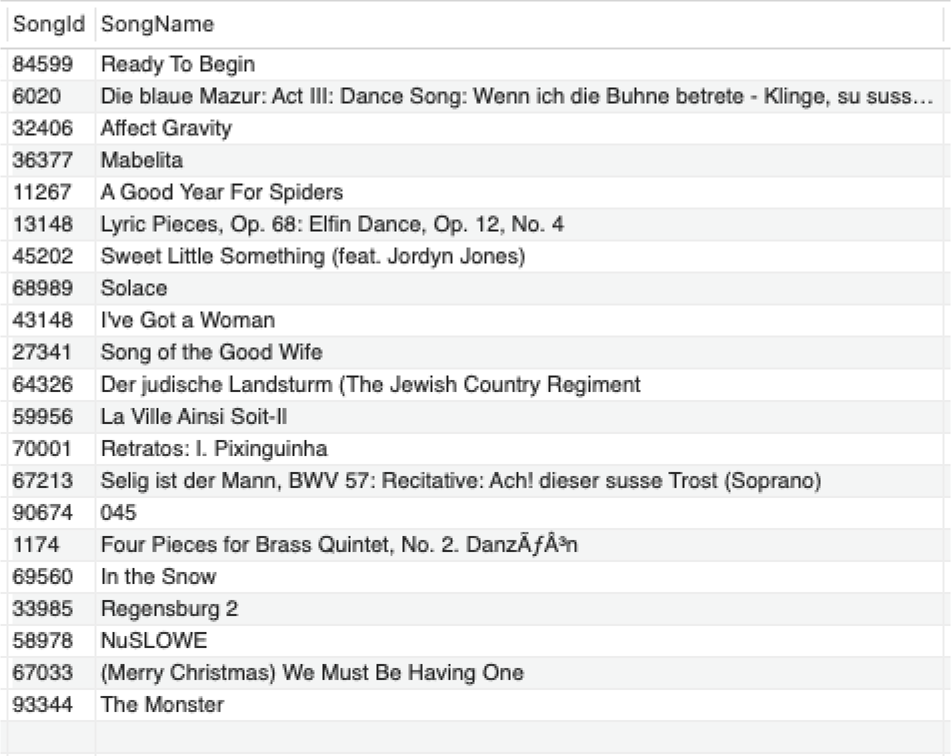


PlaylistId	PlaylistName	Description	UpdatedAt	SongName	ArtistName
17	Key moments		2021-06-22 13:36:55	Clarinet Quintet in A Major, K. 581: I. Allegro	Wolfgang Amadeus Mozart
17	Key moments		2021-06-22 13:36:55	Honest Man	My Morning Jacket
17	Key moments		2021-06-22 13:36:55	Zigeunerliebe: Act III: Schlussszene: Ich bin ein Ziguenerkind (Ilona, Jozsi, Chorus)	Dagmar Schellenberger
17	Key moments		2021-06-22 13:36:55	Missa brevis: Kyrie	Geoffery Silver
17	Key moments		2021-06-22 13:36:55	In Love With The Doubt	Dolorean
17	Key moments		2021-06-22 13:36:55	Overture (Suite) in E-Flat Major, TWV 55:Es3, "La lyra": III. La Vielle	Nicholas Ward
17	Key moments		2021-06-22 13:36:55	Where I Been	Lavosier
17	Key moments		2021-06-22 13:36:55	Kids In Da PJ's - Dirty Version	Bravehearts
17	Key moments		2021-06-22 13:36:55	Orpheus Over and Under: Chorale	Edmund Niemann
46	Time For Nightmare		2020-04-20 09:27:24	The Love For Three Oranges, Op. 33 (Sung in English): Act III Scene 2: We're here…	Wendy Dawn Thompson
46	Time For Nightmare		2020-04-20 09:27:24	Told You!	Autumn!
46	Time For Nightmare		2020-04-20 09:27:24	I Should Care	Sathima Bea Benjamin
46	Time For Nightmare		2020-04-20 09:27:24	Symphony No. 2 in D Major, Op. 36: III. Scherzo. Allegro (Live)	Ludwig van Beethoven
46	Time For Nightmare		2020-04-20 09:27:24	The Gentle Rain	Tony Bennett
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	Moods of a New Yorker (1932): I. At Dusk	Zez Confrey
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	Winter Wonderland	Milton Batiste
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	The Love For Three Oranges, Op. 33 (Sung in English): Act II Scene 1: Howzat?! (T…	Ali McGregor
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	Anticomercial	El Chojin
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	Sunday Evening	Kevin Wong
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	Pero No Termino	Ana BelÃƒÂ©n
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	Maracas Song	Galaxie 500
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	Shinkansen High Tension	Total Dork
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	Donna Lee	Larry Schneider
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	The Big Cheese (incorporating a fantasia by William Byrd [1543-1623]) - Instrumental	Edgar Meyer
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	Defend Your Joy (Buck RogersÃƒâ€šÃ‚´ Radio Edit)	The Echoing Green
73	Light Life	massa id lobortis convallis tortor risus…	2020-10-31 12:02:42	The First Crossing of the Watershed	Northcape
117	Feel Good Music		2020-11-04 15:18:54	Odin's Hill	Achillea
117	Feel Good Music		2020-11-04 15:18:54	Romeo and Juliet, Op. 64 (Excerpts): The Mandolin Dance	Sergei Prokofiev
117	Feel Good Music		2020-11-04 15:18:54	Press Record	Sensational
117	Feel Good Music		2020-11-04 15:18:54	Resolution - Mix	Glis
117	Feel Good Music		2020-11-04 15:18:54	Canadian Barn Dance	Skipinnish
117	Feel Good Music		2020-11-04 15:18:54	Cecile, Ma Fille	Claude Nougaro
117	Feel Good Music		2020-11-04 15:18:54	Another Day	Archer Prewitt
117	Feel Good Music		2020-11-04 15:18:54	Glasgow	The Out_Circuit
117	Feel Good Music		2020-11-04 15:18:54	Holdin' On (To The Love I Got)	Barbara Mandrell
117	Feel Good Music		2020-11-04 15:18:54	What A Little Love Can Do	Vinnie Zumo
117	Feel Good Music		2020-11-04 15:18:54	Iraqnophobia	Jello Biafra
125	Drama Time		2021-04-11 17:23:19	Go Freely (feat. Coco Love Alcorn)	Drew McIvor
125	Drama Time		2021-04-11 17:23:19	Tout ce sang verse	Dark Sanctuary
125	Drama Time		2021-04-11 17:23:19	Langsam, mit melancholischem Ausdruck	Michael Zaretsky, Xak Bjerken
125	Drama Time		2021-04-11 17:23:19	Lamuerte	Matzak
125	Drama Time		2021-04-11 17:23:19	Intermission - Live	Trey Songz
125	Drama Time		2021-04-11 17:23:19	When You Wish Upon A Star	Nils Landgren
125	Drama Time		2021-04-11 17:23:19	Stone to Sand	Tuu
125	Drama Time		2021-04-11 17:23:19	Uns sind die Sterne egal	Die Prinzen

12. What songs are liked by another specific user who have liked the same song with the base user?

(Assumption: Base user "sfoskenq5" knows the fact that another specific user ”akeinratlv” liked a same song with himself/herself. Now, the base user "sfoskenq5” wants to know what other songs are like by this user ”akeinratlv”. The SQL statement below solves this problem.)

```
SELECT Songs.SongId, Songs.SongName
FROM
  (
    SELECT OtherUser_Likes.SongId AS SongId
    FROM
      (
        SELECT Likes.SongId AS SongId
        FROM Likes
    WHERE UserName = 'akeinratlv'
   ) AS OtherUser_Likes
  WHERE OtherUser_Likes.SongId NOT IN
   (
    SELECT Likes.SongId AS SongId
    FROM Likes
    WHERE UserName = 'sfoskenq5'
   )
 ) AS Filtered_OtherUser_Likes
INNER JOIN Songs ON Filtered_OtherUser_Likes.SongId = Songs.SongId;
```



SongId	SongName
84599	Ready To Begin
6020	Die blaue Mazur: Act III: Dance Song: Wenn ich die Buhne betrete - Klinge, su suss…
32406	Affect Gravity
36377	Mabelita
11267	A Good Year For Spiders
13148	Lyric Pieces, Op. 68: Elfin Dance, Op. 12, No. 4
45202	Sweet Little Something (feat. Jordyn Jones)
68989	Solace
43148	I've Got a Woman
27341	Song of the Good Wife
64326	Der judische Landsturm (The Jewish Country Regiment
59956	La Ville Ainsi Soit-Il
70001	Retratos: I. Pixinguinha
67213	Selig ist der Mann, BWV 57: Recitative: Ach! dieser susse Trost (Soprano)
90674	045
1174	Four Pieces for Brass Quintet, No. 2. DanzÃƒÂ³n
69560	In the Snow
33985	Regensburg 2
58978	NuSLOWE
67033	(Merry Christmas) We Must Be Having One
93344	The Monster

Advanced feature: How to target other users (rank by the number of liked songs with the base user)?

Updated SQL Script for Creating tables and Loading data:

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

```
USE MusiCraze;

DROP TABLE IF EXISTS Covers;
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 (
  AlbumId INT AUTO_INCREMENT,
  Name VARCHAR(5000),
  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
);


#CREATE TABLE Songs(
#  SongId INT AUTO_INCREMENT,
#  SongName VARCHAR(100) NOT NULL,
#  SpotifyId VARCHAR(100),
#  ArtistId INT,
#  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,
#CONSTRAINT fk_Songs_AlbumId FOREIGN KEY (AlbumId)
#         REFERENCES Albums(AlbumId)
#         ON UPDATE CASCADE ON DELETE SET NULL
# );

# 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,
 CONSTRAINT fk_Songs_AlbumId FOREIGN KEY (AlbumId)
        REFERENCES Albums(AlbumId)
        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,
          CONSTRAINT uq_UserName_SongId UNIQUE(SongId, UserName)      # Updated:
Added uniqueness constraints
);
```

```
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

);


CREATE TABLE Covers (
  CoverId INT AUTO_INCREMENT,
  CoverName VARCHAR(200),
  PerformerName VARCHAR(200),
  YoutubeUrl VARCHAR(500),
  SongId INT,

  CONSTRAINT pk_Covers_CoverId
    PRIMARY KEY (CoverId),
  CONSTRAINT fk_Covers_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, AlbumSpotifyId, 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/administrat
ors.csv' INTO TABLE Administrators
   FIELDS TERMINATED BY ','
   ENCLOSED BY '"'
   LINES TERMINATED BY '\n'
   IGNORE 1 LINES;

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

# songsWithArtistAlbumId.csv constains matching foreign keys(ArtistId and AlbumId) for each
song.
LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/songsWith
ArtistAlbumId.csv' INTO TABLE Songs
  FIELDS TERMINATED BY ',' ENCLOSED BY '"'
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES (SongId,SongName,SpotifyId,ArtistSpotifyId,ArtistId, AlbumSpotifyId,
AlbumId);
```

```
#### ==== Insert Coldplay's(a specific artist's) Songs data ======

LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/coldplay.cs
v' INTO TABLE Songs
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES (SongName,SpotifyId,ArtistSpotifyId,AlbumSpotifyId);

# Updated ArtistId and AlbumId field:
UPDATE Songs s, Artists a
            SET s.ArtistId = a.ArtistId
    WHERE s.ArtistSpotifyId = a.ArtistSpotifyId and s.ArtistSpotifyId =
'4gzpq5DPGxSnKTe4SA8HAU';  # Coldplay's ArtistSpotifyId

UPDATE Songs s, Albums a
            SET s.AlbumId = a.AlbumId
    WHERE s.AlbumSpotifyId = a.AlbumSpotifyId and s.ArtistSpotifyId =
'4gzpq5DPGxSnKTe4SA8HAU';



LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/likes.csv'
INTO TABLE Likes
  FIELDS TERMINATED BY ',' ENCLOSED BY '"'
            LINES TERMINATED BY '\n'
            IGNORE 1 LINES (LikeId, UserName, SongId, CreatedAt);


LOAD DATA LOCAL INFILE
'/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/comments.
csv' INTO TABLE Comments
  FIELDS TERMINATED BY ',' ENCLOSED BY '"'
            LINES TERMINATED BY '\n'
            IGNORE 1 LINES (UserName, SongId, Content, CreatedAt);

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/playlistsongcontains.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);


LOAD DATA LOCAL INFILE '/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/Covers1-1000.csv' INTO TABLE Covers  # testData/result.csv
  FIELDS TERMINATED BY ',' ENCLOSED BY '"'
        LINES TERMINATED BY '\n'
        IGNORE 1 LINES (CoverName, PerformerName, YoutubeUrl, SongId);

LOAD DATA LOCAL INFILE '/Users/cindychen/Documents/NEU/Course_Material/cs5200/CS5200_GROUP/data/ColdPlayCovers_created_Mar10.csv' INTO TABLE Covers  # testData/result.csv
  FIELDS TERMINATED BY ',' ENCLOSED BY '"'
        LINES TERMINATED BY '\n'
        IGNORE 1 LINES (CoverName, PerformerName, YoutubeUrl, SongId);
```