MUSIC LIBRARY MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

LALITH SURYA G [RA2211032010009]
SHABARINATHAN K R V[RA2211032010011]
AMAL R[RA2211032010015]

Under the Guidance of

Dr. V. NALLARASAN

(Assistant Professor, Department of Networking and Communications)

in partial fulfillment of the requirementsfor the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING
with specialization in (INTERNET OF THINGS)

DEPARTMENT OF NETWORKING AND COMMUNICATIONS
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE ANDTECHNOLOGY
KATTANKULATHUR- 603 203

MAY 2024

1

SRM

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

RA2211032010009, RA2211032010011, RA2211032010015 Certified to be the bonafide work done by **LALITH SURYA G,SHABARINATHAN KRV, AMAL R** of II year/IV sem B.Tech Degree Course in the Project Course – **21CSC205P Database Management Systems** in **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**, Kattankulathur for the academic year 2023-2024.

Faculty In Charge
DR.V.Nallarasan
Assistant Professor
NWC Department
SRMIST-KTR

HEAD OF THE DEPARTMENT
DR.Annapurani paniyappan.K
Professor &Head
NWC Department
SRMIST-KTR

ABSTRACT

This project proposes the development of a comprehensive Music Library Management System (MLMS) using Database Management System (DBMS) techniques. The MLMS aims to address the complexities associated with organizing, storing, retrieving, and managing vast collections of music tracks, albums, artists, genres, and other relevant metadata. Leveraging the power of modern database technologies, the system provides functionalities such as seamless search and retrieval, personalized recommendations, user access control, and robust data integrity mechanisms. Key features of the proposed MLMS include a user-friendly interface accessible via web or mobile platforms, support for various file formats, integration with online music databases for automatic metadata retrieval, and customizable tagging and categorization options. Additionally, the system incorporates advanced algorithms for analyzing user preferences and behavior to enhance recommendation accuracy and user experience. By implementing the Music Library Management System, users ranging from individual music enthusiasts to large-scale music libraries and streaming platforms can efficiently organize, explore, and enjoy their music collections while facilitating better decision-making and resource allocation for music-related businesses.

PROBLEM STATEMENT

The current landscape of digital music consumption presents challenges in effectively managing vast and diverse music libraries. Existing solutions often lack the robustness and flexibility required to handle the complexities associated with organizing, storing, and retrieving music collections. Users encounter difficulties in navigating and exploring their libraries, leading to inefficiencies in accessing desired music content. Additionally, traditional methods of music library management are often labor-intensive and prone to errors, hindering the seamless organization and maintenance of music collections.In light of these challenges, there is a pressing need for a comprehensive Music Library Management System (MLMS) that leverages the capabilities of Database Management System (DBMS) technologies. The MLMS should address key pain points such as inefficient search and retrieval mechanisms, limited support for metadata management, lack of personalized recommendation features, and inadequate scalability to accommodate growing music collections.Therefore, the problem at hand is to design and develop an MLMS that offers a user-friendly interface, robust database architecture, advanced search and recommendation algorithms, and seamless integration with existing music databases. The system should provide users with efficient tools for organizing, exploring, and enjoying their music collections while catering to the diverse needs of individual users, music enthusiasts, and businesses in the music industry.

TABLE OF CONTENTS

CHAPTER 1

1.1 PROBLEM UNDERSTANDING

- **Current Challenges in Music Library Management:** The problem statement highlights that the current methods of managing digital music libraries face several challenges. These challenges might include difficulties in organizing large collections of music, retrieving specific songs or albums, and maintaining accurate metadata (information about the music, such as artist, album, genre, etc.).
- **Inefficiencies in Navigation and Exploration:** Users may find it hard to navigate through their music libraries and explore the content they have. This could be due to limitations in the user interface or search capabilities of existing systems.
- **Labor-Intensive and Error-Prone Methods:** Traditional methods of managing music libraries, such as manual organization or using rudimentary software tools, can be time-consuming and prone to mistakes. This could lead to inaccuracies in metadata or difficulty in keeping the library organized.
- **Need for a Comprehensive Solution:** Given these challenges, there's a need for a comprehensive Music Library Management System (MLMS) that can address these issues effectively. Such a system should utilize Database Management System (DBMS) technologies to provide robustness, scalability, and flexibility in managing music collections.
- **Key Features Required:** The MLMS should offer a user-friendly interface, advanced search and recommendation capabilities, and seamless integration with existing music databases. It should cater to the needs of both individual users and businesses in the music industry.

1.2 Identification of Entity and Relationships

In the proposed database schema, several entities are identified to represent various components of a music library management system. These entities include User, Song, Artist, Album, Playlist, Genre, License, RecordLabel, Subscription, Publisher, and SongRating. Each entity is characterized by unique attributes that capture relevant information for effective management of music-related data.

The User entity represents individuals who interact with the system, identified by attributes such as UserID, Username, Password, and SubscriptionType. Users can create playlists, rate songs, and subscribe to different plans offered by the system.

Songs, represented by the Song entity, are fundamental units of the music library, featuring attributes like SongID, Title, Genre, ArtistID, AlbumID, ReleaseDate, Duration, Lyrics, and CoverArt. Songs are associated with specific artists, albums, and genres, facilitating organized categorization and retrieval of music content.

Artists, Albums, and Genres are entities that provide additional context to the songs in the library. The Artist entity includes attributes such as ArtistID, Name, and Bio, while the Album entity comprises AlbumID, Title, ArtistID, ReleaseDate, and CoverArt. Genres, represented by the Genre entity, are characterized by GenreID, Name, and Description, allowing for the classification of songs based on their musical styles.

Playlists enable users to organize and manage their favorite songs, with attributes like PlaylistID, UserID, Name, Description, and CreationDate. Users can create multiple playlists and add songs to them, fostering personalized music listening experiences.

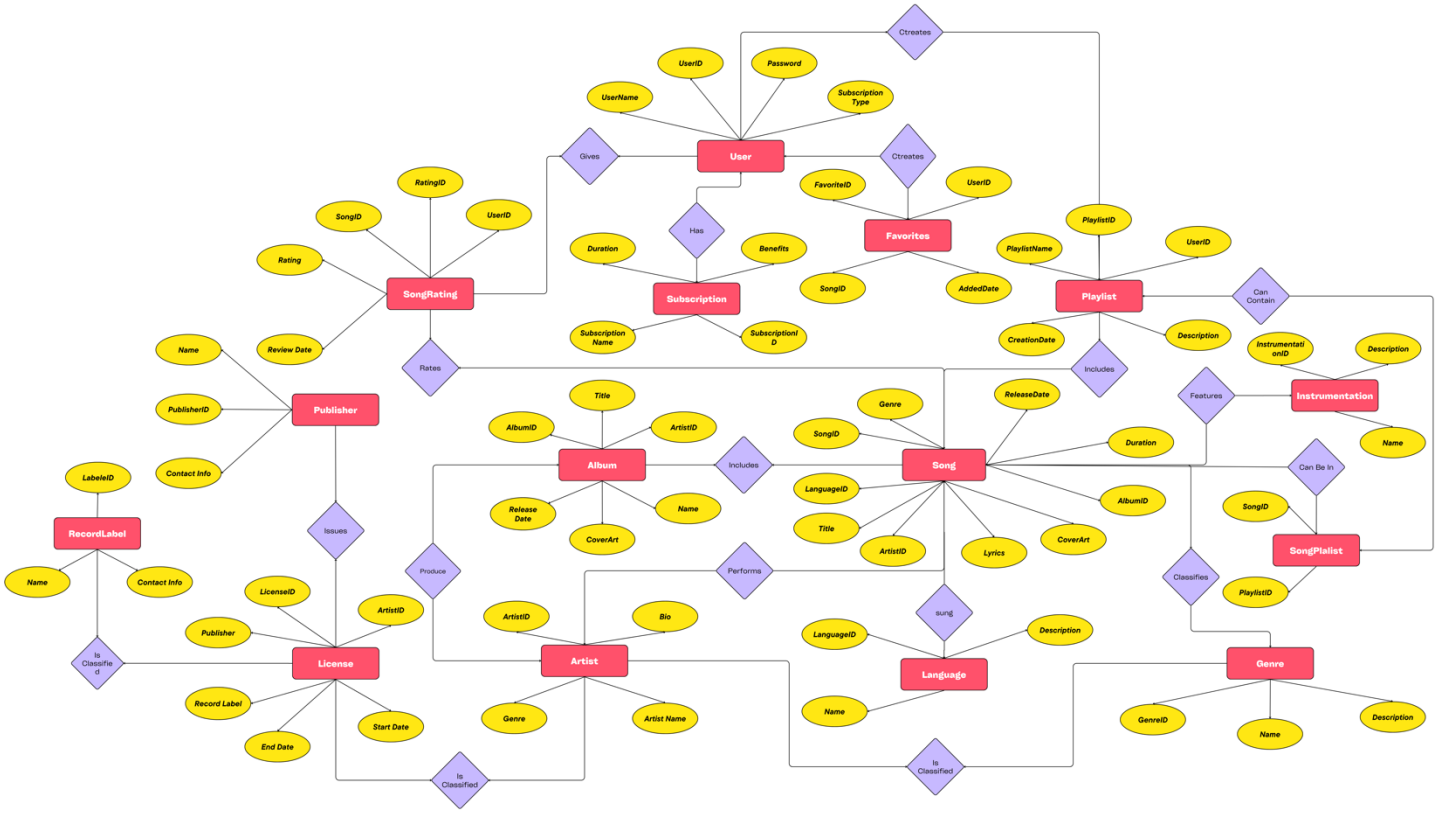
The Subscription entity captures information about different subscription plans offered by the system, including attributes like SubscriptionID, Name, Price, Duration, and Benefits. Users can subscribe to these plans to access premium features and services.

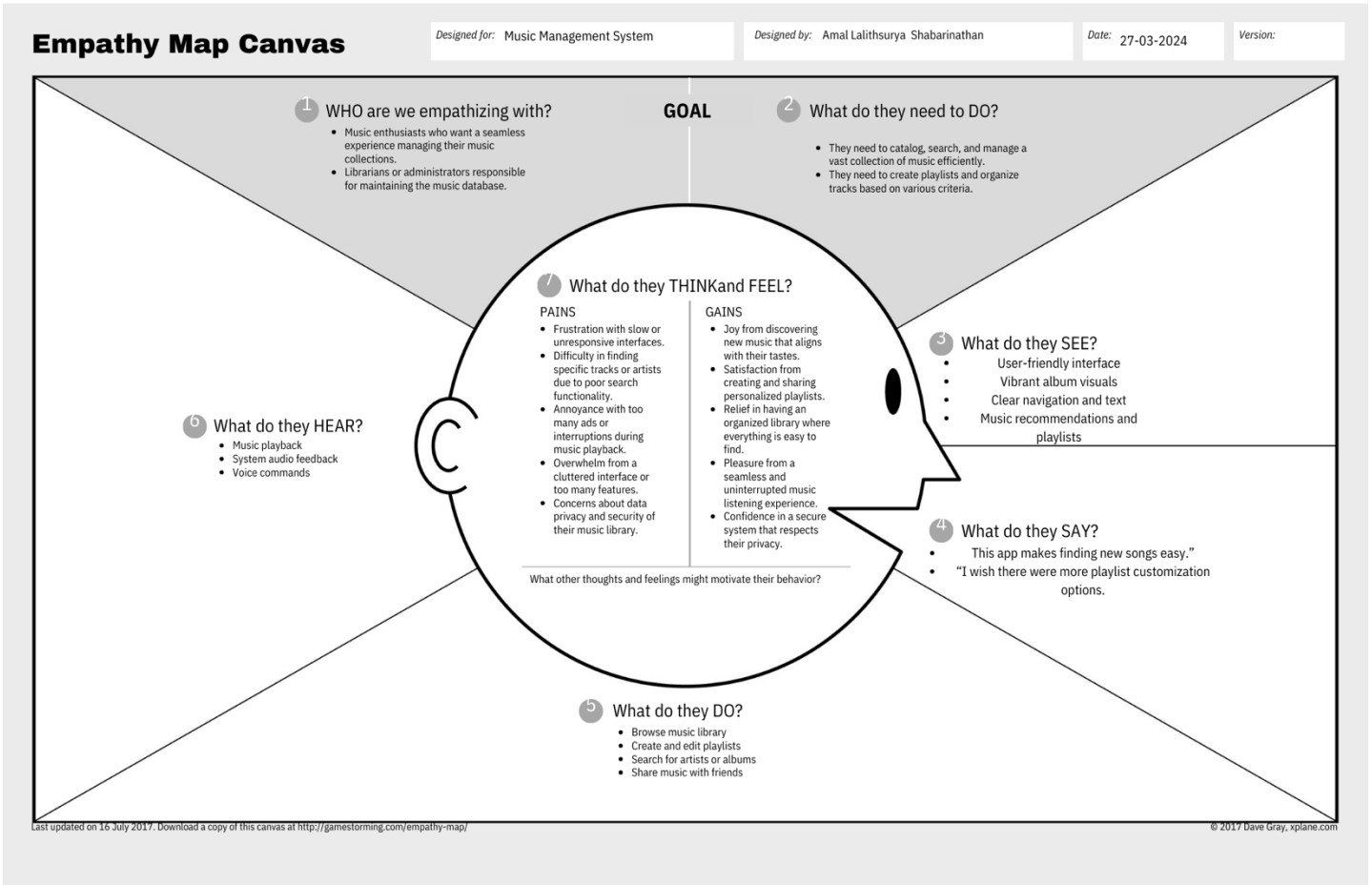
Entities such as License, RecordLabel, and Publisher are involved in managing the rights and distribution of music content. The License entity includes LicenseID, RecordLabelID, PublisherID, StartDate, and EndDate attributes, facilitating the licensing of music for commercial purposes.

The schema also defines relationships between entities to establish associations and dependencies. One-to-many relationships exist between entities such as User to Playlist, Artist to Song, Artist to Album, Album to Song, Genre to Song, Genre to Artist, Subscription to User, RecordLabel to License, and Publisher to License. Many-to-many relationships are established between Song and Playlist, as well as Song and User, enabling multiple songs to be associated with multiple playlists and users.

In summary, the proposed database schema provides a structured framework for managing music-related data, encompassing entities representing users, songs, artists, albums, genres, playlists, subscriptions, licenses, and publishers, along with defined relationships to facilitate efficient organization, retrieval, and utilization of music content within a comprehensive music library management system.

1.3 CONSTRUCTION OF DB USING ER MODEL FOR THE PROJECT





CHAPTER 2

2.1 DESIGN OF RELATIONAL SCHEMAS

2.1.1 Tables:

- **Artist:** Stores information about artists, including ID, name, optional biography, and genre (which can be linked to a separate Genre table).
- **Album:** Stores information about albums, including ID, title, artist (linked to Artist table), release date, cover art (stored as a BLOB), and foreign key referencing the Artist table.
- **Song:** Stores information about songs, including ID, title, genre (linked to Genre table), artist (linked to Artist table), album (linked to Album table), release date, duration, lyrics, cover art (stored as a BLOB), and foreign keys referencing Genre, Artist, and Album tables.
- **Playlist:** Stores information about playlists, including ID, user (linked to User table), name, description, creation date, and foreign key referencing the User table.
- **Genre (Optional):** Stores information about genres, including ID and name (with an optional description).
- **License (Optional):** Tracks licensing information, including ID, record label (linked to RecordLabel table), publisher (linked to Publisher table), artist (linked to Artist table), start and end dates of the license.
- **User:** Stores user information, including ID, username, password, and subscription type.
- **Subscription (Optional):** Defines different subscription plans, including ID, name, price, duration, and benefits.
- **Publisher (Optional):** Stores information about publishers, including ID and name (with an optional contact info).
- **RecordLabel (Optional):** Stores information about record labels, including ID and name (with an optional contact info).
- **SongRating (Optional):** Tracks user song ratings, including ID, song (linked to Song table), user (linked to User table), rating, and review date.
- **SongPlaylist:** This table creates a many-to-many relationship between Songs and Playlists. It has no primary key on its own, but a composite primary key made of SongID and PlaylistID. It references both Song and Playlist tables.

2.1.2 **Relationships:**

The code defines various relationships between tables:

- **One-to-Many:**
  - User to Playlist (one user can have many playlists)
  - Artist to Song (one artist can have many songs)
  - Artist to Album (one artist can have many albums)
  - Album to Song (one album can have many songs)
  - Genre to Song (one genre can have many songs) (optional)
  - Genre to Artist (one genre can have many artists) (optional)
  - Subscription to User (one subscription applies to one user) (optional)
  - RecordLabel to License (one record label can have many licenses) (optional)
  - Publisher to License (one publisher can have many licenses) (optional)
- **Many-to-Many:**
  - Song to Playlist (a song can be in many playlists, and a playlist can contain many songs)
- **Many-to-One:** (This can be seen as the reverse of one-to-many relationships)
  - Playlist to User (one playlist belongs to one user)
  - Song to Artist (one song belongs to one artist)
  - Song to Album (one song belongs to one album)
  - Song to Genre (one song belongs to one genre) (optional)
  - Artist to Genre (one artist can belong to one genre) (optional)
  - User to Subscription (one user has one subscription) (optional)
  - License to RecordLabel (one license belongs to one record label) (optional)
  - License to Publisher (one license belongs to one publisher) (optional)

2.2 CREATION OF DATABASE TABLES FOR THE PROJECT

1. user(UserID,Username,Password,SubscriptionType)

UserId	UserName	Password	SubscriptionType

2. song(SongID,Title,Genre,ArtistID,AlbumID,ReleaseDate,Duration,Lyrics,CoverArt)

SongID	Title	Genre	ArtistID	AlbumID	ReleaseDate	Duration	Lyrics	CoverArt

3. artist(ArtistID,Name,Bio,Genre)

ArtistID	Name	Bio	Genre

4. album(AlbumID,Title,ArtistID,ReleaseData,CoverArt)

AlbumID	Title	ArtistID	ReleaseData	CoverArt

5. playlist(PlayListID,UserID,Name,Description,CreationDate)

PlayListID	UserID	Name	Description	CreationDate

6. genre(GenreID,Name,Description)

GenreID	Name	Description

7. license(LicenseID,RecordLabel,Publisher,ArtistID,StartDate,EndDate)

LicenseID	,RecordLabel	,Publisher,	ArtistID	StartDate	EndDate

8. recordlablel(RecordLable,Name,ContactInfo)

RecordLable	Name	ContactInfo

9. subscription(SubscriptionID,Name,Price,Duration,Benefits)

SubsricptionID	Name	Price	Duration	Benefits

10. publisher(PublisherID,Name,ContactInfo)

PublisherID	Name	ContactInfo

11. songrating(RatingID,SongID,UserID,Rating,ReviewDate)

RatingID	SongID	USerID	Rating	ReviewDate

12. songplaylist(SongID,PlaylistID)

SongID	PlaylistID

13. instrumentation(InstrumentationID,Description,Name)

InstrumentationID	Name	Description

CARDINALITY MAPPING:

1. song includes playlist (m to n)

song(SongID,Title,Genre,ArtistID,AlbumID,ReleaseDate,Duration,Lyrics,CoverArt)

SongID	Title	Genre	ArtistID	AlbumID	ReleaseDate	Duration	Lyrics	CoverArt

playlist(PlayListID,UserID,Name,Description,CreationDate)

PlayListID	UserID	Name	Description	CreationDate

Includes(SongID,PlayListID)

SongID	PlayListID

user creates playlist (1 to m)

UserId	UserName	Password	SubscriptionType

PlayListCreates(UserID,PlayListID,UserID,Name,Description,CreationDate)

PlayListID	UserID	Name	Description	CreationDate

2. Artist performs song

artist(ArtistID,Name,Bio,Genre)

ArtistID	Name	Bio	Genre

Perfomsong(ArtistID,SongID,Title,Genre,ArtistID,AlbumID,ReleaseDate,Duration,Lyrics, CoverArt)

SongID	Title	Genre	ArtistID	AlbumID	ReleaseDate	Duration	Lyrics	CoverArt

3. Artist produce Album

artist(ArtistID,Name,Bio,Genre)

ArtistID	Name	Bio	Genre

Produce Album(ArtistID,AlbumID,Title,ArtistID,ReleaseData,CoverArt)

AlbumID	Title	ArtistID	ReleaseData	CoverArt

4. Album includes Song

album(AlbumID,Title,ArtistID,ReleaseData,CoverArt)

AlbumID	Title	ArtistID	ReleaseData	CoverArt

Includesong(AlbumID,SongID,Title,Genre,ArtistID,AlbumID,ReleaseDate,Duration,Lyrics, CoverArt)

SongID	Title	Genre	ArtistID	AlbumID	ReleaseDate	Duration	Lyrics	CoverArt

5. Genre classifies Song

genre(GenreID,Name,Description)

GenreID	Name	Description

classifiessong(GenreID,SongID,Title,Genre,ArtistID,AlbumID,ReleaseDate,Duration,Lyrics, CoverArt)

SongID	GenreID	Title	Genre	ArtistID	AlbumID	ReleaseDate	Duration	Lyrics	CoverArt

6. Genre is classified Artist

genre(GenreID,Name,Description)

GenreID	Name	Description

ClassifiedArtist(GenreID,ArtistID,Name,Bio,Genre)

ArtistID	GenreID	Name	Bio	Genre

7. User has Subsricption

subscription(SubscriptionID,Name,Price,Duration,Benefits)

SubsricptionID	Name	Price	Duration	Benefits

User has(SubsricptionID,UserID,Username,Password,SubscriptionType)

UserID	SubsricptionID	UserName	Password	SubscriptionType

8. RecordLable is classified License

recordlablel(RecordLable,Name,ContactInfo)

RecordLable	Name	ContactInfo

License isclassified(RecordLableID,LicenseID,RecordLabel,Publisher,ArtistID,StartDate, EndDate)

RecordLable	LicenseID	RecordLabel	,Publisher,	ArtistID	StartDate	EndDate

9. Publisher Issues License

publisher(PublisherID,Name,ContactInfo)

PublisherID	Name	ContactInfo

Issues License (PublisherID,LicenseID,RecordLabel,Publisher,ArtistID,StartDate, EndDate)

LicenseID	PublisherID	RecordLabel	,Publisher,	ArtistID	StartDate	EndDate

CHAPTER 3

3.1 COMPLEX QUERIES BASED ON THE CONCEPTS OF CONSTRAINTS, SETS, JOINS, VIEWS, TRIGGERS AND CURSORS.

1. Constraints:

```
CREATE TABLE song_playlist (
   SongID INT,
   PlaylistID INT PRIMARY KEY,
   FOREIGN KEY (SongID) REFERENCES song(SongID),
   FOREIGN KEY (PlaylistID) REFERENCES playlist(PlaylistID)
);

CREATE TABLE songrating (
   RatingID INT PRIMARY KEY,
   SongID INT,
   UserID INT,
   Rating DECIMAL(3,2) DEFAULT NULL,
   ReviewDate DATE NOT NULL,
   FOREIGN KEY (SongID) REFERENCES song(SongID),
   FOREIGN KEY (UserID) REFERENCES USER1(UserID)
);

CREATE TABLE license (
   LicenseID INT PRIMARY KEY,
   RecordLabel VARCHAR(255) NOT NULL,
   Publisher INT NOT NULL,
   ArtistID INT,
   StartDate DATE,
   EndDate DATE,
   FOREIGN KEY (ArtistID) REFERENCES artist(ArtistID),
   FOREIGN KEY (Publisher) REFERENCES publisher(PublisherID)
);
```

2. SETS

```
-- Union of users with Mega Fan and Basic subscriptions.
SELECT * FROM User WHERE SubscriptionType = 'mega fan'
UNION
SELECT * FROM User WHERE SubscriptionType = 'basic';
```

```
-- Intersection of songs in Rock and Pop genres
SELECT * FROM Song WHERE GenreID = (
   SELECT GenreID FROM Genre WHERE Name = 'Rock'
)
INTERSECT
SELECT * FROM Song WHERE GenreID = (
   SELECT GenreID FROM Genre WHERE Name = 'Pop'
);

-- Difference of playlists owned by User A and not owned by User B
SELECT * FROM Playlist WHERE UserID = ' 84259553'
EXCEPT
SELECT * FROM Playlist WHERE UserID = ' 86465466';
```

3. **JOINS**

```
SELECT song.SongID,artist.ArtistID,song.Title
From song
INNER JOIN artist ON Song.ArtistID=artist.ArtistID
WHERE artist.ArtistID=('23015');

SELECT song.SongID,artist.ArtistID,song.Title
From song
INNER JOIN artist ON Song.ArtistID=artist.ArtistID;
```

4. **VIEWS:**

```
CREATE VIEW `alaipayuthey` AS
SELECT    `song`.`Title` AS `Title`, `song`.`Duration` AS `Duration`
FROM   `song`
WHERE  (`song`.`AlbumID` = '374')

CREATE VIEW `deivathirumagan` AS
SELECT  `song`.`Title` AS `Title`, `song`.`Duration` AS `Duration`
FROM  `song`
WHERE  (`song`.`AlbumID` = '902')
```

5. **TRIGGERS:**

---Trigger: validate_publisher_exists

```
DELIMITER //
CREATE TRIGGER validate_publisher_exists BEFORE INSERT ON license
FOR EACH ROW
BEGIN
DECLARE publisher_exists INT;

SELECT COUNT(*) INTO publisher_exists
FROM publisher
WHERE publisher.PublisherID = NEW.Publisher;

IF publisher_exists = 0 THEN
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Publisher does not exist.';
END IF;
END; //
DELIMITER;
```

---Trigger: enforce_password_length

```
DELIMITER //
CREATE TRIGGER enforce_password_length BEFORE INSERT ON USER1
FOR EACH ROW
BEGIN
DECLARE password_length INT;

SET password_length = LENGTH(NEW.Pass);

IF password_length < 8 THEN
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Password must be at least 8 characters long.';
END IF;
END; //
DELIMITER ;
```

---Trigger: enforce_genre_id

```
DELIMITER //
CREATE TRIGGER enforce_genre_id BEFORE INSERT ON song
FOR EACH ROW
BEGIN
IF NEW.GenreID IS NULL THEN
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Genre cannot be null.';
END IF;
END; //
DELIMITER ;
```

6. CURSORS

---Cursor for song table:

```
CREATE TEMPORARY TABLE song_details (
  SongTitle VARCHAR(255),
  ArtistName VARCHAR(255),
  GenreName VARCHAR(255)
);
INSERT INTO song_details (SongTitle, ArtistName, GenreName)
SELECT song.Title, artist.Name, genre.GenreName
FROM song
INNER JOIN artist ON song.ArtistID = artist.ArtistID
INNER JOIN genre ON song.GenreID = genre.GenreID;

DECLARE song_cursor CURSOR FOR
SELECT SongTitle, ArtistName, GenreName
FROM song_details;

DECLARE
  song_title VARCHAR(255);
  artist_name VARCHAR(255);
  genre_name VARCHAR(255);

OPEN song_cursor;

LOOP
  FETCH NEXT FROM song_cursor INTO song_title, artist_name, genre_name;

  IF @@FETCH_STATUS = 0 THEN  -- Check if there are more rows
    -- Process the retrieved data (e.g., print)
    SELECT CONCAT(song_title, ' by ', artist_name, ' (', genre_name, ')') AS song_info;
  ELSE
    CLOSE song_cursor;  -- Close the cursor if no more rows
    DEALLOCATE song_cursor;  -- Deallocate resources
    DROP TEMPORARY TABLE song_details;  -- Drop the temporary table
    EXIT LOOP;  -- Exit the loop
```

```
  END IF;
END LOOP;
```

---List all artists and their genres

```
DECLARE artist_cursor CURSOR FOR
SELECT a.name AS artist_name, g.name AS genre_name
FROM Artists a
JOIN Genres g ON a.genre_id = g.genre_id;

OPEN artist_cursor;

FETCH NEXT FROM artist_cursor INTO artist_name, genre_name;

WHILE @@FETCH_STATUS = 0
BEGIN
  -- Process each artist and genre combination (e.g., print to console)
  PRINT artist_name + ' - ' + genre_name;
  FETCH NEXT FROM artist_cursor INTO artist_name, genre_name;
END;

CLOSE artist_cursor;
DEALLOCATE artist_cursor;
```

---Find all tracks by a specific artist

```
DECLARE artist_id INT;

DECLARE track_cursor CURSOR FOR

SELECT t.title, t.duration

FROM Tracks t

JOIN Albums alb ON t.album_id = alb.album_id

JOIN Artists a ON alb.artist_id = a.artist_id

WHERE a.name = 'The Beatles'; -- Replace 'The Beatles' with desired artist name

SET artist_id = (SELECT artist_id FROM Artists WHERE name = 'The Beatles');
```

```
OPEN track_cursor;

FETCH NEXT FROM track_cursor INTO track_title, track_duration;

WHILE @@FETCH_STATUS = 0

BEGIN

 -- Process each track (e.g., print title and duration)

 PRINT track_title + ' (' + CONVERT(VARCHAR(5), track_duration / 60) + ':' + RIGHT('0' +
CONVERT(VARCHAR(2), track_duration % 60), 2) + ')';

 FETCH NEXT FROM track_cursor INTO track_title, track_duration;

END;

CLOSE track_cursor;

DEALLOCATE track_cursor;
```

---Update the genre of all tracks in a specific album:

```
DECLARE album_id INT;

DECLARE update_cursor CURSOR FOR

SELECT track_id

FROM Tracks

WHERE album_id = :target_album_id; -- Replace :target_album_id with actual album ID

SET album_id = :target_album_id; -- Replace with actual album ID

OPEN update_cursor;

FETCH NEXT FROM update_cursor INTO track_id;

WHILE @@FETCH_STATUS = 0

BEGIN

 UPDATE Tracks SET genre_id = :new_genre_id WHERE track_id = track_id;

 FETCH NEXT FROM update_cursor INTO track_id;

END;

CLOSE update_cursor;
```

```
DEALLOCATE update_cursor;
```

CHAPTER 4

ANALYZING THE PITFALLS, IDENTIFYING THE DEPENDENCIES, AND APPLYING NORMALIZATIONS

4.1 ANALYZING THE PITFALLS:

1. **Data redundancy:**

- **Genre information:** If the genre information is stored directly in the `Tracks` table instead of a separate `Genres` table, it can lead to redundancy if the same genre applies to multiple tracks.
- **Artist information in playlists (optional):** Including artist information directly in the `Playlists` table might be redundant if it's already stored in the `Artists` table and linked through the `Albums` table.

2. **Data inconsistency:**

- **Incorrect or missing genre IDs:** Inconsistent genre IDs (foreign keys) between `Artists` and `Genres` tables can lead to orphaned data or retrieval issues.
- **Missing or incorrect album IDs:** Inconsistent album IDs (foreign keys) between `Tracks` and `Albums` tables can lead to orphaned tracks or retrieval issues.

3. **Scalability limitations:**

- **Single table for playlists:** If playlists are simply a list of track IDs in the `Playlists` table, managing large playlists might become inefficient. A separate `Playlist_Tracks` table (many-to-many relationship) offers better scalability for extensive playlists.

4. **Normalization issues:**

- **Denormalization for performance:** While some denormalization (copying data) might be done strategically for faster queries (e.g., storing total playlist duration in the `Playlists` table), excessive denormalization can lead to data inconsistency issues during updates.

4.2 IDENTIFYING THE DEPENDENCIES:

1. **Foreign Keys:**

- `Artists.genre_id` depends on the primary key of the `Genres` table.
- `Albums.artist_id` depends on the primary key of the `Artists` table.
- `Tracks.album_id` depends on the primary key of the `Albums` table.

- `Playlist_Tracks.playlist_id` (optional) depends on the primary key of the `Playlists` table.
- `Playlist_Tracks.track_id` (optional) depends on the primary key of the `Tracks` table. (primary key combination in this case)

2. **Data Integrity:**

- Adding a genre in the `Genres` table should be reflected when assigning genres to artists in the `Artists` table.
- Adding an artist in the `Artists` table should be reflected when assigning artists to albums in the `Albums` table.
- Adding an album in the `Albums` table should be reflected when adding tracks to that album in the `Tracks` table.

3. **Data retrieval:**

- Retrieving information about a track requires joining the `Tracks` table with the `Albums` table and potentially the `Artists` and `Genres` tables (depending on the desired information).
- Retrieving all tracks in a playlist (optional) requires joining the `Playlists` table with the `Playlist_Tracks` table (many-to-many) and then the `Tracks` table to get track details.

4.3 APPLYING NORMALIZATIONS:

Album (2NF)

table1: album_id (primary key), title

table2: release_id (primary key), album_id (foreign key), release_date

Artist (1NF)

table1: artist_id (primary key), name, bio (consider splitting into a separate table if bio gets very large)

Genre (1NF)

table1: genre_id (primary key), name

ClassifiedArtist (already in 1NF - consider 2NF if needed)

table1: artist_id (primary key), genre_id (foreign key)
table 2: artist_id (primary key),name,bio

Song (3NF)

table1: song_id (primary key), title, duration

table2: song_id (primary key), artist_id (foreign key), release_date

Playlist (already in 2NF)

table1: playlist_id (primary key), user_id (foreign key), name

table2: playlist_id (foreign key), description, creation_date

IncludesSong (already in 2NF)

table1: playlist_id (foreign key), song_id (foreign key) (primary key combination)

SongRating (3NF)

table1: rating_id (primary key), user_id (foreign key)

table2: song_id (foreign key), rating_id (foreign key), rating, review_date

User (3NF)

table1: user_id (primary key), username, subscription_type

table2: user_id (primary key), password_hash

Language

table1: language_id (primary key), language_name

table2: song_id (foreign key), language_id (foreign key)

Subscription

table1: subscription_id (primary key), subscription_type, description (optional details about the plan)

RecordLabel (already in 2NF)

table1: label_id (primary key), name

Publisher (already in 2NF)

table1: publisher_id (primary key), name

License (already in 2NF)

table1: licenseID,recordlable,artistid,startdate ,end date

table2: recordlableid,publisher

CHAPTER 5

IMPLEMENTATION OF CONCURRENCY CONTROL AND RECOVERY MECHANISMS

Concurrency control in databases is a crucial aspect of ensuring data integrity and consistency in multiuser environments. It refers to the mechanisms and techniques used to manage and coordinate simultaneous access to data by multiple transactions. Without proper concurrency control, transactions could interfere with each other, leading to data corruption or inconsistencies.

One common approach to concurrency control is locking, where transactions acquire locks on data items to prevent other transactions from modifying them concurrently. There are different types of locks, such as read locks and write locks, which control the level of access permitted to a data item. Locking helps to maintain data integrity by ensuring that transactions can only access or modify data in a controlled manner.

Another approach to concurrency control is timestamping, where transactions are assigned unique timestamps based on their start times. Transactions are then ordered based on their timestamps, and conflicts are resolved by giving priority to transactions with earlier timestamps. Timestamping helps to ensure that transactions are executed in a serializable order, preventing conflicts and maintaining consistency.

Optimistic concurrency control is a different approach that assumes conflicts are rare. Transactions are allowed to proceed without acquiring locks, and conflicts are only checked for at the end of the transaction. If a conflict is detected, the transaction is rolled back and re- executed. This approach is less restrictive than locking but requires careful handling of conflicts to avoid unnecessary rollbacks.

Concurrency control is a complex area of database management, and different techniques may be more suitable depending on the specific requirements of the application. The goal of concurrency control is to ensure that transactions can execute concurrently without compromising data integrity, consistency, or performance.

The four important aspects of database management are:

1. Atomicity: Ensures that transactions are all or nothing, meaning either all operations within the transaction are successfully completed, or none of them are.
2. Consistency: Guarantees that the database remains in a consistent state before and after the execution of a transaction.
3. Isolation: Ensures that concurrent transactions do not interfere with each other, maintaining data integrity and preventing anomalies.
4. Durability: Ensures that once a transaction is committed, its effects are permanently saved and persisted, even in the event of a system failure.

Recovery Mechanism:

Recovery mechanisms in database management systems (DBMS) are essential for ensuring data durability and system reliability in the event of failures. These mechanisms are designed to restore the database to a consistent state after a failure and to minimize the impact of the failure on ongoing transactions. There are two main types of recovery mechanisms: undo recovery and redo recovery.

Undo recovery, also known as rollback, is the process of undoing the effects of transactions that were not completed at the time of failure. This ensures that any changes made by these transactions are not permanently applied to the database. Undo recovery typically involves using transaction logs to identify and reverse the changes made by incomplete transactions.

Redo recovery, on the other hand, is the process of reapplying the effects of transactions that were completed but not yet permanently stored in the database at the time of failure. This ensures that any changes made by these transactions are not lost. Redo recovery involves using transaction logs to identify and reapply the changes made by these transactions.

In addition to undo and redo recovery, DBMSs also use mechanisms such as checkpointing and logging to ensure data consistency and durability. Checkpointing involves periodically writing the current state of the database to disk, along with information about transactions that have been committed but not yet written to disk. This allows the system to recover to a consistent state after a failure by restoring the database to the last checkpoint and replaying the logs.

Logging is the process of recording all changes made to the database in a log file. This allows the system to reconstruct the state of the database at any point in time and to recover from failures by replaying the log from the last checkpoint. Logging also provides a way to ensure

the atomicity and durability of transactions, as changes are only considered committed once they have been recorded in the log.

Overall, recovery mechanisms are critical for maintaining data integrity and system reliability in database management systems. They ensure that data is not lost or corrupted in the event of

```
-- Set isolation level
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

-- Begin transaction
BEGIN;

-- Locking mechanism
SELECT * FROM Songs WHERE SongID = 1111 FOR UPDATE;

-- Perform operations within the transaction
UPDATE Songs SET Title = 'New Title' WHERE SongID = 1111;

-- Commit transaction
COMMIT;

-- Write-ahead logging (WAL)
-- PostgreSQL handles WAL internally, ensuring durability of transactions

-- Checkpointing
CHECKPOINT;

-- Restore from backup
pg_basebackup -U root - /Users/shabarinathankrv/Downloads/includesong.csv -Ft -Xs -z -P -v -h
localhost:3306
```

CHAPTER 6

CODE FOR THE PROJECT:

QUERIES:

```
CREATE TABLE album (
   AlbumID INT PRIMARY KEY,
   Title VARCHAR(255) NOT NULL,
   ReleaseDate DATE,
);
```

```
INSERT INTO album (AlbumID, Title, ReleaseDate) VALUES
(106, 'Kattradhu Tamil', '2019'),
(145, 'Aruvi', '2016'),
(287, 'Aranmanai 2', '2016'),
(374, 'Alaipayuthey', '2000'),
(389, 'Paiyaa', '2010'),
(405, '96', '2018'),
(439, 'Ratsasan', '2018'),
(469, '7G Rainbow Colony', '2004'),
(473, 'Varuvādzhi... Part 2', '2001'),
(481, 'Minnale (Theatrical Version)', '2001'),
(499, 'Vaaranam Aayiram', '2008'),
(557, 'Kaithi', '2019'),
(565, 'Kanaa', '2018'),
(641, 'O Kadhal Kanmani', '2015'),
(651, 'Irumbu Thirai', '2018'),
(691, 'Naalaiya Varum', '2006'),
(694, 'Comali', '2019'),
(785, 'Nerungi Vaa Muthuvam', '2017'),
(823, 'Nenjathe Atle', '2002'),
(824, 'Kaththi', '2014'),
(897, 'Remo', '2016'),
(902, 'Deiva Thirumagan', '2011'),
(964, 'Master', '2021');
```

```
-- Artist table
CREATE TABLE artist (

   ArtistID INT PRIMARY KEY,
   Name VARCHAR(255) NOT NULL,
   Bio VARCHAR(1000),
   );
```

```
INSERT INTO Artists (ArtistID, Name, Bio) VALUES
(19003, 'Sam C.S.', 'A composer known for electronic and rock influences, receiving National Film Awards and Filmfare Awards South.'),
(23015, 'G.V. Prakash Kumar', 'Injects folk and rock elements into his compositions, a National Film Award winner.'),
(36973, 'A.R. Rahman', 'A versatile maestro known for fusing Indian classical, electronic, and world music, winning numerous prestigious awards.'),
(40043, 'Yuvan Shankar Raja', 'A composer who blends electronic and hip hop influences, recognized with National Film Awards and Filmfare honors.'),
(46381, 'C.Sathya', 'Creates scores with electronic and orchestral elements, a National Film Award and Filmfare Awards South winner.'),
(61909, 'Ghibran', 'Blends electronic and folk music, a National Film Award and Filmfare Award South winner.'),
(62244, 'Hiphop Tamizha', 'A hip hop group known for Tamil lyrics and social commentary, winning Filmfare Awards South.'),
(74305, 'Anirudh Ravichander', 'A composer integrating electronic and hip hop, bagging Filmfare Awards South and Filmfare honors.'),
(86350, 'Dhibu Ninan Thomas', 'An artist utilizing electronic and indie rock, recipient of Filmfare Awards South.'),
(91515, 'Harris Jayaraj', 'Creates catchy melodies and orchestral arrangements, decorated with Filmfare Awards.'),
(92025, 'Santhosh Narayanan', 'Merges electronic and hip hop for impactful scores, earning National Film Awards and Filmfare accolades.'),
(96829, 'Govind Vasantha', 'A composer weaving folk and classical sounds, recognized with National Film Awards and a Filmfare Award South.'),
(98429, 'Sean Roldan', 'A composer weaving folk and classical sounds, recognized with National Film Awards and a Filmfare Award South.');


-- Playlist table
CREATE TABLE playlist (
    PlaylistID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Description VARCHAR(1000),
    CreationDate DATE NOT NULL,
);

INSERT INTO Playlists (playlistID, Name, Description, CreationDate) VALUES
(5501, 'Guitar Hero Anthems', 'Shred like a rockstar with this playlist featuring epic guitar solos and iconic rock anthems, perfect for air guitar enthusiasts.', '2003-05-17'),
(5502, 'Chill Vibes Mix', 'Unwind with a collection of laid-back tunes perfect for relaxing evenings and mellow moments.', '2006-09-29'),
(5503, 'Road Trip Anthems', 'Hit the road with this upbeat playlist featuring an eclectic mix of songs to keep you energized on your journey.', '2009-04-08'),
(5504, 'Acoustic Serenade', 'Immerse yourself in the soothing sounds of acoustic melodies, perfect for peaceful moments and introspection.', '2004-11-05'),
(5505, 'Feel-Good Classics', 'Rediscover timeless classics that never fail to uplift your spirits and put a smile on your face.', '2007-06-22'),
(5506, 'Late Night Jams', 'Dive into a collection of smooth tunes and soulful beats, ideal for unwinding during the late hours.', '2005-08-12'),
(5507, 'Sunday Morning Melodies', 'Start your day on a tranquil note with this selection of gentle melodies and soothing harmonies.', '2002-03-18'),
```

(5508, 'Workout Beats', 'Get pumped up and motivated with this high-energy playlist filled with tracks to fuel your workout sessions.', '2010-10-31'),
(5509, 'Rainy Day Relaxation', 'Embrace the cozy vibes of a rainy day with this playlist designed to help you unwind and find peace.', '2008-07-09'),
(5510, 'Summer Breeze Playlist', 'Capture the essence of summer with this playlist featuring sunny tunes and feel-good vibes.', '2001-12-20'),
(5511, 'Retro Rewind', 'Take a trip down memory lane with this nostalgic playlist showcasing hits from decades past.', '2006-02-14'),
(5512, 'Dance Party Essentials', 'Turn up the volume and hit the dance floor with this collection of infectious beats and party anthems.', '2004-05-06'),
(5513, 'Study Focus Soundtrack', 'Stay focused and productive with this curated selection of instrumental tracks perfect for studying and concentration.', '2019-09-03'),
(5514, 'Indie Gems Collection', 'Discover hidden musical treasures and indie favorites in this eclectic compilation of indie tracks.', '2023-04-27'),
(5516, 'Jazz Lounge Favorites', 'Settle into the smooth ambiance of a jazz lounge with this playlist featuring classic jazz standards and contemporary favorites.', '2017-08-10'),
(5517, 'R&B Slow Jams', 'Set the mood with this sultry collection of R&B slow jams, perfect for romantic evenings and late-night vibes.', '2005-01-13'),
(5518, 'Rock Legends Playlist', 'Rock out to iconic anthems and legendary tracks from the greatest rock bands of all time.', '2021-06-15'),
(5519, 'Country Roads Playlist', 'Hit the open road with this playlist celebrating the heart and soul of country music, perfect for scenic drives and adventures.', '2018-11-08'),
(5520, 'EDM Energy Boost', 'Get ready to party with this high-octane playlist featuring pulsating beats and electrifying EDM tracks.', '2016-03-11'),
(5521, 'Soulful Grooves', 'Immerse yourself in the soulful rhythms and smooth melodies of this playlist, guaranteed to lift your spirits and move your soul.', '2012-07-30'),
(5522, '90s Nostalgia Hits', 'Relive the magic of the 90s with this throwback playlist featuring all your favorite hits from the decade.', '2014-10-19'),
(5523, 'Classical Masterpieces', 'Experience the beauty and grandeur of classical music with this curated selection of timeless masterpieces by renowned composers.', '2000-12-12'),
(5524, 'Pop Perfection Playlist', 'Indulge in the catchy hooks and infectious melodies of this playlist featuring the best in pop music.', '2002-11-11');

```
-- Genre table
CREATE TABLE genre (
   GenreID INT PRIMARY KEY,
   Name VARCHAR(255) NOT NULL,
   Description TEXT(1000)
);
```

INSERT INTO Genres (GenreID, Name, Description) VALUES
(1311, 'POP', 'Popular music, characterized by catchy melodies, simple harmonies, and danceable rhythms. Examples include artists like Michael Jackson, Taylor Swift, and BTS.'),
(1523, 'ROCK', 'A broad genre with many subgenres, typically featuring electric guitars, drums, and bass. Examples include The Beatles, Led Zeppelin, and Nirvana.'),
(2145, 'HIP HOP', 'Characterized by a strong rhythmic beat, rapping (spoken-word lyrics), and sampling of other music sources. Examples include artists like Jay-Z, Kendrick Lamar, and Nicki Minaj.'),
(2209, 'ELECTRONIC', 'Created with synthesizers, drum machines, and other electronic instruments. Examples include subgenres like techno, house, and dubstep.'),
(2939, 'COUNTRY', 'Often features acoustic guitars, fiddles, and themes of rural life, heartbreak, and storytelling. Examples include artists like Johnny Cash, Dolly Parton, and Garth Brooks.'),
(3208, 'R&B', 'Characterized by soulful vocals, syncopated rhythms, and influences from blues and jazz. Examples include artists like Aretha Franklin, Marvin Gaye, and Beyoncé.'),

(3249, 'JAZZ', 'Emphasizes improvisation, syncopation, and complex harmonies. Examples include artists like Miles Davis, John Coltrane, and Ella Fitzgerald.'),
(3269, 'CLASSICAL', 'A broad genre encompassing a wide range of musical styles and periods, from the Baroque era to the present day. Examples include composers like Beethoven, Mozart, and Tchaikovsky.'),
(3408, 'WORLD', 'Music that incorporates elements from different cultures and regions around the world. This can include genres like reggae, flamenco, and bossa nova.'),
(4064, 'METAL', 'Characterized by heavy distortion, fast tempos, and powerful vocals. Examples include subgenres like heavy metal, thrash metal, and death metal.'),
(4203, 'FOLK', 'Traditional music of a particular community or people, often passed down orally from generation to generation. It typically features acoustic instruments, simple melodies, and lyrics that tell stories about everyday life, history, or folklore. Examples include artists like Bob Dylan, Joan Baez, and Simon & Garfunkel.'),
(4999, 'BLUES', 'A genre with roots in African American communities of the Southern United States. It's characterized by a "blue note" (a slightly flattened third or seventh scale degree), call and response vocals, and themes of hardship, love, and resilience. Examples include artists like B.B. King, Robert Johnson, and Muddy Waters.'),
(6100, 'REGGAE', 'Developed in Jamaica in the late 1960s, reggae is characterized by a laid-back tempo, a distinctive offbeat rhythm section (often played on the guitar with a "skank" pattern), and lyrics that often deal with social and political issues. Examples include artists like Bob Marley, Peter Tosh, and Bunny Wailer.'),
(6456, 'FUNK', 'A genre that emerged from African American music in the 1960s. It's characterized by prominent bass lines, syncopated rhythms, and the use of electric instruments like keyboards and guitars. Funk music often emphasizes groove and improvisation. Examples include artists like James Brown, Parliament-Funkadelic, and Sly and the Family Stone.'),
(7664, 'SOUL', 'A genre that blends elements of gospel, R&B, and blues. Soul music typically features powerful vocals, emotional lyrics, and a strong focus on rhythm and melody. Examples include artists like Aretha Franklin, Ray Charles, and Marvin Gaye.'),
(7665, 'GOSPEL', 'Music with religious content, often associated with Christian worship. Gospel music typically features uplifting lyrics, call and response vocals, and a strong rhythmic pulse. Examples include artists like Mahalia Jackson, The Staple Singers, and Kirk Franklin.'),
(7908, 'SOUNDTRACK', 'Instrumental music or songs composed, produced, or compiled specifically as part of a motion picture or television program. Soundtracks can encompass a wide range of genres depending on the film or show. Examples include iconic scores from composers like John Williams, Hans Zimmer, and Ennio Morricone.'),
(8463, 'AMBIENT', 'Electronic music that is intended to create an atmosphere or mood rather than to be a focus in itself. Ambient music can be calming, relaxing, or even unsettling, depending on the artist and the specific piece. Examples include artists like Brian Eno, Aphex Twin, and Boards of Canada.'),
(8691, 'EXPERIMENTAL', 'Music that pushes boundaries and challenges traditional forms. Experimental music

```
-- License table
CREATE TABLE license (
    LicenseID INT PRIMARY KEY,
    RecordLabel VARCHAR(255) NOT NULL,
    Publisher INT NOT NULL,
    ArtistID INT,
    StartDate DATE,
    EndDate DATE,
    FOREIGN KEY (ArtistID) REFERENCES artist(ArtistID),
    FOREIGN KEY (Publisher) REFERENCES publisher(PublisherID)
);

INSERT INTO Licenses (LicenseID, RecordLabel, Publisher, ArtistID, StartDate, EndDate) VALUES
(11001, 'Saregama', 'Saregama', 19003, 1947, 2030),
(11002, 'Kizhakku Pathippagam', 'Kizhakku Pathippagam', 23015, 1988, 2040),
(11003, 'International Music Publishers', 'International Music Publishers', 36973, 1985, 2026),
```

```
(11004, 'Lahari Music', 'Lahari Music', 40043, 1978, 2078),
(11005, 'Think Music India', 'Think Music India', 46381, 1996, 2065),
(11006, 'Sony Music South', 'Sony Music South', 61909, 1998, 2034),
(11007, 'Muzik 247', 'Muzik 247', 62244, 1990, 2030),
(11008, 'Divo', 'Divo', 74305, 1980, 2025),
(11009, 'Aditya Music', 'Aditya Music', 19003, 1987, 2026),
(11010, 'Vel Records', 'Vel Records', 91515, 1967, 2040),
(11011, 'Five Star Audio', 'Five Star Audio', 92025, 1956, 2054),
(11012, 'Star Music India', 'Star Music India', 96829, 1978, 2034),
(11013, 'Millennium Audios', 'Millennium Audios', 61909, 1998, 2028),
(11014, 'T-Series', 'T-Series', 91515, 1982, 2029),
(11015, 'Vandana publications', 'Vandana publications', 36973, 1979, 2030),
(11016, 'Tips Industries', 'Tips Industries', 61909, 1964, 2031),
(11017, 'Antony Music', 'Antony Music', 40043, 1947, 2025),
(11018, 'Woods Music', 'Woods Music', 46381, 1999, 2032),
(11019, 'Believe Digital:', 'Believe Digital:', 19003, 1991, 2026),
(11020, 'WM Music', 'WM Music', 74305, 1982, 2034);

-- Publisher table
CREATE TABLE publisher (
    PublisherID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    ContactInfo VARCHAR(255) UNIQUE
);

INSERT INTO Publishers (PublisherID, Name, ContactInfo) VALUES
(1001, 'Saregama', '9236255504'),
(1002, 'Kizhakku Pathippagam', '7759581463'),
(1003, 'International Music Publishers', '7578119141'),
(1004, 'Lahari Music', '9424906608'),
(1005, 'Think Music India', '8383962967'),
(1006, 'Sony Music South', '8185447108'),
(1007, 'Muzik 247', '7502992053'),
(1008, 'Divo', '9394380013'),
(1009, 'Aditya Music', '8738620618'),
(1010, 'Vel Records', '9966260826'),
(1011, 'Five Star Audio', '8668771601'),
(1012, 'Star Music India', '8946519798'),
(1013, 'Millennium Audios', '7220575841'),
(1014, 'T-Series', '7560824360'),
(1015, 'Vandana publications', '9602290732'),
(1016, 'Tips Industries', '8754505091'),
(1017, 'Antony Music', '8013285551'),
(1018, 'Woods Music', '9635654092'),
(1019, 'Believe Digital:', '7701227522'),
(1020, 'WM Music', '9818127190');


-- RecordLabel table
CREATE TABLE recordlabel (
    RecordLabelID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    ContactInfo VARCHAR(255) UNIQUE
);
```

```
INSERT INTO RecordLabels (RecordLabelID, Name, ContactInfo) VALUES
(2001, 'Saregama', '9236255504'),
(2002, 'Kizhakku Pathippagam', '7759581463'),
(2003, 'International Music Publishers', '7578119141'),
(2004, 'Lahari Music', '9424906608'),
(2005, 'Think Music India', '8383962967'),
(2006, 'Sony Music South', '8185447108'),
(2007, 'Muzik 247', '7502992053'),
(2008, 'Divo', '9394380013'),
(2009, 'Aditya Music', '8738620618'),
(2010, 'Vel Records', '9966260826'),
(2011, 'Five Star Audio', '8668771601'),
(2012, 'Star Music India', '8946519798'),
(2013, 'Millennium Audios', '7220575841'),
(2014, 'T-Series', '7560824360'),
(2015, 'Vandana publications', '9602290732');


--song table
CREATE TABLE song (
    SongID INT PRIMARY KEY,
    Title VARCHAR(255) NOT NULL,
    ReleaseDate DATE,
    Duration TIME NOT NULL,
);

INSERT INTO Songs (SongID, Title, ReleaseDate, Duration) VALUES
(1111, 'Jagada Thom', '2011', '0:03:37'),
(1112, 'Aariro', '2011', '0:04:11'),
(1113, 'Kadha Solla Poraen', '2011', '0:03:27'),
(1114, 'Life Is Beautiful', '2011', '0:04:53'),
(1115, 'Pa Pa Pappa', '2011', '0:03:22'),
(1116, 'Vennilave Vennilave', '2011', '0:04:58'),
(1117, 'Vizhigalil Oru Vaanavil', '2011', '0:02:56'),
(1121, 'Yethi Yethi', '2008', '0:03:17'),
(1122, 'Oh Shanthi Shanthi', '2008', '0:03:23'),
(1123, 'Nenjukkul Peidhidum', '2008', '0:02:47'),
(1124, 'Mundhinam', '2008', '0:05:34'),
(1125, 'Ava Enna', '2008', '0:04:58'),
(1126, 'Annul Maelae', '2008', '0:04:34'),
(1127, 'Adiyae Kolluthey', '2008', '0:04:12'),
(1131, 'Yedho Ondru Ennai', '2010', '0:03:34'),
(1132, 'Thuli Thuli', '2010', '0:04:43'),
(1133, 'Suthudhe Suthudhe', '2010', '0:04:56'),
(1134, 'Poongatre Poongatre', '2010', '0:05:20'),
(1135, 'En Kadhal Solla', '2010', '0:04:54'),
(1136, 'Adada Mazhaida', '2010', '0:04:27'),
(1141, 'Venmathiye', '2001', '0:05:27'),
(1142, 'Vaseegara', '2001', '0:04:59'),
(1143, 'Vaerene Ivanyaro', '2001', '0:05:25'),
(1144, 'Poopol Poopol', '2001', '0:01:57'),
(1145, 'Ore Nyabagam', '2001', '0:01:54'),
(1146, 'Ooh Mama', '2001', '0:04:38'),
```

```
(1147, 'Nenjai Poopol', '2001', '0:01:01'),
(1148, 'Maddy Maddy', '2001', '0:01:15'),
(1149, 'Azhagiya Theeye', '2001', '0:05:55'),
(1151, 'Party song', '2017', '0:03:23'),
(1152, 'Liberty Song', '2017', '0:05:23'),
(1161, 'Yaro Yarodi', '2000', '0:06:05'),
(1162, 'Snehidhane', '2000', '0:05:07'),
(1163, 'September Madham', '2000', '0:05:57'),
(1164, 'Pachchai Nirame', '2000', '0:04:35'),
(1165, 'Kadhal Sadugudu', '2000', '0:05:56'),
(1166, 'Evano Oruvan', '2000', '0:03:44'),
(1167, 'Alai Payuthey', '2000', '0:05:46'),
(1171, 'Hey Piriyame Piriyame', '2018', '0:03:51'),
(1172, 'Kannamma Kanvizhi', '2018', '0:03:26'),
(1181, 'Walking Through The Rainbow Theme Music', '2004', '0:03:22'),
(1182, 'Ninaithu Ninaithu Parthen', '2004', '0:04:37'),
(1183, 'Naam Vayathukku', '2004', '0:05:07'),
(1184, 'Kan Pesum Varthaigal', '2004', '0:05:50'),
(1185, 'Idhu Porkkalama', '2004', '0:03:09'),
(1191, 'Othaiyadi Pathayila', '2018', '0:02:45'),
(1192, 'Vaayadi Petha Pulla', '2018', '0:04:04'),
(1193, 'Oonjala Oonjala', '2018', '0:04:31'),
(2001, 'Paisa Note', '2019', '0:04:31'),
(2002, 'Yaara Comali', '2019', '0:03:30'),
(2003, 'Hi Sonna Pothum', '2019', '0:03:45'),
(2004, 'Nanba Nanba', '2019', '0:04:43');

-- SongRating table
CREATE TABLE songrating (
    RatingID INT PRIMARY KEY,
    SongID INT,
    UserID INT,
    Rating DECIMAL(3,2) DEFAULT NULL,
    ReviewDate DATE NOT NULL,
    FOREIGN KEY (SongID) REFERENCES song(SongID),
    FOREIGN KEY (UserID) REFERENCES USER1(UserID)
);

INSERT INTO Ratings (RatingID, SongID, UserID, Rating, ReviewDate) VALUES
(301, 1111, 18983208, 4.8, '13/01/05'),
(302, 1111, 37437849, 4.7, '09/02/05'),
(303, 1111, 68989430, 4.2, '18/09/10'),
(304, 1112, 88196267, 4.4, '12/10/18'),
(305, 1112, 62145320, 4.6, '22/01/19'),
(306, 1112, 86246535, 3.7, '31/03/17'),
(307, 1113, 58192554, 4.9, '24/06/20'),
(308, 1113, 99802145, 5, '19/10/23'),
(309, 1113, 71920293, 5, '14/11/19'),
(310, 1114, 66203440, 4.9, '11/11/02'),
(311, 1114, 44001355, 3.9, '30/04/89'),
(312, 1114, 88196267, 4.4, '30/09/03'),
(313, 1115, 14661113, 4.2, '25/12/18'),
(314, 1115, 86465466, 4.1, '23/10/23'),
(315, 1115, 68989430, 3.5, '15/10/23'),
(316, 1116, 73249313, 2.5, '08/08/04'),
```

```
(317, 1117, 98526216, 2.6, '07/07/04'),
(318, 1117, 32262495, 1, '06/06/06'),
(319, 1121, 66203440, 1.1, '21/07/12'),
(320, 1121, 18418073, 1.9, '23/06/04'),
(321, 1121, 18418073, 3.6, '28/05/23'),
(322, 1116, 32262495, 2.9, '17/05/05'),
(323, 1116, 88196267, 5, '02/03/13'),
(324, 1117, 71920293, 5, '15/09/17'),
(3011, 1111, 58192554, 4.5, '01/01/01'),
(3021, 1111, 71920293, 4.6, '12/03/12'),
(3031, 1111, 13361601, 5, '13/01/22'),
(3041, 1112, 33407476, 4, '18/04/15'),
(3051, 1112, 81443060, 5, '29/05/17'),
(3061, 1112, 44001355, 4.1, '04/07/16'),
(3071, 1113, 99802145, 3.5, '21/09/14'),
(3081, 1113, 46691274, 3.6, '15/08/10'),
(3091, 1113, 39357475, 4.9, '31/10/15'),
(3101, 1114, 54314743, 5, '05/12/13'),
(3111, 1114, 39357475, 5, '08/02/10'),
(3121, 1114, 39357475, 3.7, '17/03/05'),
(3131, 1115, 79472726, 3.9, '17/03/05'),
(3141, 1115, 71920293, 4.2, '23/04/18'),
(3151, 1115, 73249313, 4.3, '30/05/15'),
(3161, 1116, 54314743, 4.3, '06/06/04'),
(3171, 1116, 13361601, 4.5, '16/07/09'),
(3181, 1116, 46691274, 4.8, '18/08/16'),
(3191, 1117, 44001355, 4.1, '24/09/12'),
(3201, 1117, 33407476, 5, '02/10/15'),
(3211, 1117, 86246535, 4.1, '07/12/01'),
(3221, 1121, 81443060, 5, '14/02/09'),
(3231, 1121, 62145320, 5, '20/03/16'),
(3241, 1121, 13361601, 4.8, '09/04/12');


-- Subscription table
CREATE TABLE subscription (
    SubscriptionID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Price DECIMAL(10, 2) DEFAULT 0,
    Duration INT,
    Benefits  VARCHAR(1000)
);

INSERT INTO Subscriptions (SubscriptionID, Name, Price, Duration, Benefits) VALUES
(1, 'Basic', 0, 'Life Long', 'Listen Music'),
(2, 'Fan', 79, '1 Month', 'Listen music without ads'),
(3, 'Mega Fan', 99, '1 Month', 'Listen Music Without ads,You Can Skip The songs,Can add ur playlist
suggestion to all the user'),
(4, 'Mega Fan Annual', 999, '1 Year', 'Listen Music Without ads,You Can Skip The songs,Can add ur playlist
suggestion to all the user');
```

```
--user table
CREATE TABLE USER1 (
   UserID INT PRIMARY KEY,
   Username VARCHAR(255) NOT NULL,
   Password VARCHAR(255) NOT NULL,
   SubscriptionType VARCHAR(255) Default ‘basic’
);

INSERT INTO Users (UserID, Username, Password, SubscriptionType) VALUES
(13361601, 'Leo', 'Alpha1234', 'basic'),
(14661113, 'Mia', 'Password567', 'basic'),
(18418073, 'Isabella', 'Secure789', 'fan'),
(18983208, 'Felix', '1234Beta567', 'fan'),
(29173027, 'Amelia', 'ABCdef890', 'mega fan'),
(32262495, 'Daniel', '9876Gamma', 'mega fan'),
(33407476, 'William', 'Delta123', 'mega fan annual'),
(37437849, 'Jack', '789Alpha', 'mega fan'),
(39357475, 'Noah', 'Epsilon456', 'mega fan'),
(44001355, 'Mason', 'Beta5678', 'fan'),
(46691274, 'Grace', '123Gamma456', 'fan'),
(50380543, 'Grace', 'Delta7890', 'basic'),
(54314743, 'Olivia', '890Epsilon', 'mega fan annual'),
(58192554, 'Sophia', 'Zeta1234', 'mega fan'),
(62145320, 'Benjamin', '5678Eta', 'fan'),
(66203440, 'Charlotte', 'Theta901', 'mega fan'),
(68989430, 'Evelyn', '234Iota', 'basic'),
(70201199, 'Daniel', 'Kappa5678', 'basic'),
(71920293, 'Owen', '9012Lambda', 'mega fan'),
(73249313, 'Felix', '345Mu', 'basic'),
(78604411, 'Amelia', 'Nu678', 'basic'),
(79472726, 'Lucas', '1234Xi', 'mega fan annual'),
(81443060, 'Lily', '567Omicron', 'mega fan annual'),
(84259553, 'Evelyn', '234Rho', 'mega fan annual'),
(86246535, 'Henry', 'Pi890', 'mega fan'),
(86465466, 'Penelope', 'Sigma567', 'fan'),
(88196267, 'Charlotte', '890Tau', 'basic'),
(98526216, 'Katherine', 'Chi789', 'basic'),
(99802145, 'Madison', 'Upsilon123', 'fan');

CREATE TABLE album_n1(
   AlbumID INT PRIMARY KEY,
   Title VARCHAR(255) NOT NULL,
);
INSERT INTO album_n1 (AlbumID, Title)
VALUES (106, 'Kattradhu Tamil'),
      (145, 'Aruvi'),
      (287, 'Aranmanai 2' ),
      (374, 'Alaipayuthey'),
      (389, 'Paiyaa'),
      (405, '96'),
      (439, 'Ratsasan'),
      (469, '7G Rainbow colony'),
      (473, 'Varuvádzhi... Part 2'),
```

```
    (481, 'Minnale (Theatrical Version)'),
    (499, 'Vaaranam Aayiram'),
    (557, 'Kaithi'),
    (565, 'Kanaa'),
    (641, 'O Kadhal Kanmani'),
    (651, 'Irumbu Thirai'),
    (691, 'Naalaiya Varum'),
    (694, 'Comali'),
    (785, 'Nerungi Vaa Muthuvam'),
    (823, 'Nenjathe Atli'),
    (824, 'Kaththi'),
    (897, 'Remo'),
    (902, 'Deiva Thirum'),
    (964, 'Master');

CREATE TABLE album_n2 (
   AlbumID INT PRIMARY KEY,
   ReleaseDate DATE,
);

INSERT INTO album_n2 (AlbumID,ReleaseDate)
VALUES (106, '2019'),
    (145,  '2016'),
    (287,  '2016'),
    (374, '2000'),
    (389, '2010'),
    (405, '2018'),
    (439,  '2018'),
    (469,  '2004'),
    (473, '2001'),
    (481, '2001'),
    (499,  '2008'),
    (557, '2019'),
    (565, '2018'),
    (641,  '2015'),
    (651, '2018'),
    (691,  '2006'),
    (694, '2019'),
    (785, '2017'),
    (823, '2002'),
    (824,  '2014'),
    (897,  '2016'),
    (902,  '2011'),
    (964, '2021');

CREATE TABLE classifiedartist (

   ArtistID INT PRIMARY KEY,
   Name VARCHAR(255) NOT NULL,
   GenreID INT
   Bio VARCHAR(1000),
   FOREIGN KEY(GenreID) REFERENCES genre(GenreID)
   );
```

```
INSERT INTO Artists (ArtistID, Name, Bio, GenreID) VALUES
(19003, 'Sam C.S.', 'A composer known for electronic and rock influences, receiving National Film Awards and Filmfare Awards South.', 2209),
(23015, 'G.V. Prakash Kumar', 'Injects folk and rock elements into his compositions, a National Film Award winner.', 1523),
(36973, 'A.R. Rahman', 'A versatile maestro known for fusing Indian classical, electronic, and world music, winning numerous prestigious awards.', 3408),
(40043, 'Yuvan Shankar Raja', 'A composer who blends electronic and hip hop influences, recognized with National Film Awards and Filmfare honors.', 2145),
(46381, 'C.Sathya', 'Creates scores with electronic and orchestral elements, a National Film Award and Filmfare Awards South winner.', 2209),
(61909, 'Ghibran', 'Blends electronic and folk music, a National Film Award and Filmfare Award South winner.', 4203),
(62244, 'Hiphop Tamizha', 'A hip hop group known for Tamil lyrics and social commentary, winning Filmfare Awards South.', 2145),
(74305, 'Anirudh Ravichander', 'A composer integrating electronic and hip hop, bagging Filmfare Awards South and Filmfare honors.', 2145),
(86350, 'Dhibu Ninan Thomas', 'An artist utilizing electronic and indie rock, recipient of Filmfare Awards South.', 1523),
(91515, 'Harris Jayaraj', 'Creates catchy melodies and orchestral arrangements, decorated with Filmfare Awards.', 9132),
(92025, 'Santhosh Narayanan', 'Merges electronic and hip hop for impactful scores, earning National Film Awards and Filmfare accolades.', 2145),
(96829, 'Govind Vasantha', 'A composer weaving folk and classical sounds, recognized with National Film Awards and a Filmfare Award South.', 4203),
(98429, 'Sean Roldan', 'A composer weaving folk and classical sounds, recognized with National Film Awards and a Filmfare Award South.', 9132);


CREATE TABLE classifiedartist_n1(

   ArtistID INT PRIMARY KEY,
   Name VARCHAR(255) NOT NULL,
   Bio VARCHAR(1000)
   );

INSERT INTO Artists (ArtistID, Name, Bio) VALUES
(19003, 'Sam C.S.', 'A composer known for electronic and rock influences, receiving National Film Awards and Filmfare Awards South.'),
(23015, 'G.V. Prakash Kumar', 'Injects folk and rock elements into his compositions, a National Film Award winner.'),
(36973, 'A.R. Rahman', 'A versatile maestro known for fusing Indian classical, electronic, and world music, winning numerous prestigious awards.'),
(40043, 'Yuvan Shankar Raja', 'A composer who blends electronic and hip hop influences, recognized with National Film Awards and Filmfare honors.'),
(46381, 'C.Sathya', 'Creates scores with electronic and orchestral elements, a National Film Award and Filmfare Awards South winner.'),
(61909, 'Ghibran', 'Blends electronic and folk music, a National Film Award and Filmfare Award South winner.'),
(62244, 'Hiphop Tamizha', 'A hip hop group known for Tamil lyrics and social commentary, winning Filmfare Awards South.'),
(74305, 'Anirudh Ravichander', 'A composer integrating electronic and hip hop, bagging Filmfare Awards South and Filmfare honors.'),
```

```
(86350, 'Dhibu Ninan Thomas', 'An artist utilizing electronic and indie rock, recipient of Filmfare Awards South.'),
(91515, 'Harris Jayaraj', 'Creates catchy melodies and orchestral arrangements, decorated with Filmfare Awards.'),
(92025, 'Santhosh Narayanan', 'Merges electronic and hip hop for impactful scores, earning National Film Awards and Filmfare accolades.'),
(96829, 'Govind Vasantha', 'A composer weaving folk and classical sounds, recognized with National Film Awards and a Filmfare Award South.'),
(98429, 'Sean Roldan', 'A composer weaving folk and classical sounds, recognized with National Film Awards and a Filmfare Award South.');

CREATE TABLE classifiedartist_n2(

   ArtistID INT PRIMARY KEY,
   GenreID INT,
   FOREIGN KEY(GenreID) REFERENCES genre(GenreID)
   );

INSERT INTO Artists (ArtistID, GenreID) VALUES
(19003, 2209),
(23015, 1523),
(36973, '36973'),
(40043, 2145),
(46381, 2209),
('61909', '4203' )
(62244, 2145),
(74305, 2145),
(86350, 1523),
(91515, 9132),
(92025, 2145),
('96829', '4203'),
(98429, 9132);

CREATE TABLE classifiessong (
   SongID INT PRIMARY KEY,
   Title VARCHAR(255) NOT NULL,
   GenreID INT,
   ReleaseDate DATE,
   Duration TIME NOT NULL,
   FOREIGN KEY(GenreID) REFERENCES genre(GenreID)
                              );
INSERT INTO Songs (SongID, Title, GenreID, ReleaseDate, Duration) VALUES
(1111, 'Jagada Thom', 3249, 2011, '0:03:37'),
(1112, 'Aariro', 3249, 2011, '0:04:11'),
(1113, 'Kadha Solla Poraen', 3249, 2011, '0:03:27'),
(1114, 'Life Is Beautiful', 3249, 2011, '0:04:53'),
(1115, 'Pa Pa Pappa', 3249, 2011, '0:03:22'),
(1116, 'Vennilave Vennilave', 3249, 2011, '0:04:58'),
(1117, 'Vizhigalil Oru Vaanavil', 3249, 2011, '0:02:56'),
```

(1121, 'Yethi Yethi', 7908, 2008, '0:03:17'),
(1122, 'Oh Shanthi Shanthi', 7908, 2008, '0:03:23'),
(1123, 'Nenjukkul Peidhidum', 7908, 2008, '0:02:47'),
(1124, 'Mundhinam', 7908, 2008, '0:05:34'),
(1125, 'Ava Enna', 7908, 2008, '0:04:58'),
(1126, 'Annul Maelae', 7908, 2008, '0:04:34'),
(1127, 'Adiyae Kolluthey', 7908, 2008, '0:04:12'),
(1131, 'Yedho Ondru Ennai', 9132, 2010, '0:03:34'),
(1132, 'Thuli Thuli', 9132, 2010, '0:04:43'),
(1133, 'Suthudhe Suthudhe', 9132, 2010, '0:04:56'),
(1134, 'Poongatre Poongatre', 9132, 2010, '0:05:20'),
(1135, 'En Kadhal Solla', 9132, 2010, '0:04:54'),
(1136, 'Adada Mazhaida', 9132, 2010, '0:04:27'),
(1141, 'Venmathiye', 2209, 2001, '0:05:27'),
(1142, 'Vaseegara', 2209, 2001, '0:04:59'),
(1143, 'Vaerene Ivanyaro', 2209, 2001, '0:05:25'),
(1144, 'Poopol Poopol', 2209, 2001, '0:01:57'),
(1145, 'Ore Nyabagam', 2209, 2001, '0:01:54'),
(1146, 'Ooh Mama', 2209, 2001, '0:04:38'),
(1147, 'Nenjai Poopol', 2209, 2001, '0:01:01'),
(1148, 'Maddy Maddy', 2209, 2001, '0:01:15'),
(1149, 'Azhagiya Theeye', 2209, 2001, '0:05:55'),
(1151, 'Party song', 1523, 2017, '0:03:23'),
(1152, 'Liberty Song', 1523, 2017, '0:05:23'),
(1161, 'Yaro Yarodi', 9132, 2000, '0:06:05'),
(1162, 'Snehidhane', 9132, 2000, '0:05:07'),
(1163, 'September Madham', 9132, 2000, '0:05:57'),
(1164, 'Pachchai Nirame', 9132, 2000, '0:04:35'),
(1165, 'Kadhal Sadugudu', 9132, 2000, '0:05:56'),
(1166, 'Evano Oruvan', 9132, 2000, '0:03:44'),
(1167, 'Alai Payuthey', 9132, 2000, '0:05:46'),
(1171, 'Hey Piriyame Piriyame', 3249, 2018, '0:03:51'),
(1172, 'Kannamma Kanvizhi', 3249, 2018, '0:03:26'),
(1181, 'Walking Through The Rainbow Theme Music', 7908, 2004, '0:03:22'),
(1182, 'Ninaithu Ninaithu Parthen', 7908, 2004, '0:04:37');

```
CREATE TABLE classifiessong _n1(
   SongID INT PRIMARY KEY,
   Title VARCHAR(255) NOT NULL,
   ReleaseDate DATE,
   Duration TIME NOT NULL,
   );

INSERT INTO Songs (SongID, Title, ReleaseDate, Duration) VALUES
(1111, 'Jagada Thom', '2011', '0:03:37'),
(1112, 'Aariro', '2011', '0:04:11'),
(1113, 'Kadha Solla Poraen', '2011', '0:03:27'),
(1114, 'Life Is Beautiful', '2011', '0:04:53'),
(1115, 'Pa Pa Pappa', '2011', '0:03:22'),
(1116, 'Vennilave Vennilave', '2011', '0:04:58'),
(1117, 'Vizhigalil Oru Vaanavil', '2011', '0:02:56'),
(1121, 'Yethi Yethi', '2008', '0:03:17'),
(1122, 'Oh Shanthi Shanthi', '2008', '0:03:23'),
(1123, 'Nenjukkul Peidhidum', '2008', '0:02:47'),
(1124, 'Mundhinam', '2008', '0:05:34'),
(1125, 'Ava Enna', '2008', '0:04:58'),
(1126, 'Annul Maelae', '2008', '0:04:34'),
(1127, 'Adiyae Kolluthey', '2008', '0:04:12'),
(1131, 'Yedho Ondru Ennai', '2010', '0:03:34'),
(1132, 'Thuli Thuli', '2010', '0:04:43'),
(1133, 'Suthudhe Suthudhe', '2010', '0:04:56'),
(1134, 'Poongatre Poongatre', '2010', '0:05:20'),
(1135, 'En Kadhal Solla', '2010', '0:04:54'),
(1136, 'Adada Mazhaida', '2010', '0:04:27'),
(1141, 'Venmathiye', '2001', '0:05:27'),
(1142, 'Vaseegara', '2001', '0:04:59'),
(1143, 'Vaerene Ivanyaro', '2001', '0:05:25'),
(1144, 'Poopol Poopol', '2001', '0:01:57'),
(1145, 'Ore Nyabagam', '2001', '0:01:54'),
(1146, 'Ooh Mama', '2001', '0:04:38'),
(1147, 'Nenjai Poopol', '2001', '0:01:01'),
(1148, 'Maddy Maddy', '2001', '0:01:15'),
(1149, 'Azhagiya Theeye', '2001', '0:05:55'),
(1151, 'Party song', '2017', '0:03:23'),
(1152, 'Liberty Song', '2017', '0:05:23'),
(1161, 'Yaro Yarodi', '2000', '0:06:05'),
(1162, 'Snehidhane', '2000', '0:05:07'),
(1163, 'September Madham', '2000', '0:05:57'),
(1164, 'Pachchai Nirame', '2000', '0:04:35'),
(1165, 'Kadhal Sadugudu', '2000', '0:05:56'),
(1166, 'Evano Oruvan', '2000', '0:03:44'),
(1167, 'Alai Payuthey', '2000', '0:05:46'),
(1171, 'Hey Piriyame Piriyame', '2018', '0:03:51'),
(1172, 'Kannamma Kanvizhi', '2018', '0:03:26'),
(1181, 'Walking Through The Rainbow Theme Music', '2004', '0:03:22'),
(1182, 'Ninaithu Ninaithu Parthen', '2004', '0:04:37'),
(1183, 'Naam Vayathukku', '2004', '0:05:07'),
(1184, 'Kan Pesum Varthaigal', '2004', '0:05:50'),
(1185, 'Idhu Porkkalama', '2004', '0:03:09'),
(1191, 'Othaiyadi Pathayila', '2018', '0:02:45'),
```

```
(1192, 'Vaayadi Petha Pulla', '2018', '0:04:04'),
(1193, 'Oonjala Oonjala', '2018', '0:04:31'),
(2001, 'Paisa Note', '2019', '0:04:31'),
(2002, 'Yaara Comali', '2019', '0:03:30'),
(2003, 'Hi Sonna Pothum', '2019', '0:03:45'),
(2004, 'Nanba Nanba', '2019', '0:04:43');

CREATE TABLE classifiessong_n2(
   SongID INT PRIMARY KEY,
   GenreID INT,
   FOREIGN KEY(GenreID) REFERENCES genre(GenreID)
);

INSERT INTO Songs (SongID, GenreID) VALUES
(1111, 3249),
(1112, 3249),
(1113, 3249),
(1114, 3249),
(1115, 3249),
(1116, 3249),
(1117, 3249),
(1121, 7908),
(1122, 7908),
(1123, 7908),
(1124, 7908),
(1125, 7908),
(1126, 7908),
(1127, 7908),
(1131, 9132),
(1132, 9132),
(1133, 9132),
(1134, 9132),
(1135, 9132),
(1136, 9132),
(1141, 2209),
(1142, 2209),
(1143, 2209),
(1144, 2209),
(1145, 2209),
(1146, 2209),
(1147, 2209),
(1148, 2209),
(1149, 2209),
(1151, 1523),
(1152, 1523),
(1161, 9132),
(1162, 9132),
(1163, 9132),
(1164, 9132),
(1165, 9132),
(1166, 9132),
(1167, 9132),
(1171, 3249),
(1172, 3249),
(1181, 7908),
```

```
(1182, 7908),
(1183, 7908),
(1184, 7908),
(1185, 7908),
(1191, 1523),
(1192, 1523),
(1193, 1523),
(2001, 2145),
(2002, 2145),
(2003, 2145),
(2004, 2145);

CREATE TABLE createsplaylist (
   PlaylistID INT PRIMARY KEY,
   UserID INT,
   Name VARCHAR(255) NOT NULL,
   Description VARCHAR(1000),
   CreationDate DATE NOT NULL,
   FOREIGN KEY (UserID) REFERENCES USER1(UserID)
);
INSERT INTO Playlists (PlaylistID, UserID, Name, Description, CreationDate)
VALUES
(5501, 29173027, 'Guitar Hero Anthems', 'Shred like a rockstar with this playlist featuring epic guitar solos and iconic rock anthems, perfect for air guitar enthusiasts.', '2003-05-17'),
(5502, 13361601, 'Chill Vibes Mix', 'Unwind with a collection of laid-back tunes perfect for relaxing evenings and mellow moments.', '2006-09-29'),
(5503, 70201199, 'Road Trip Anthems', 'Hit the road with this upbeat playlist featuring an eclectic mix of songs to keep you energized on your journey.', '2009-04-08'),
(5504, 37437849, 'Acoustic Serenade', 'Immerse yourself in the soothing sounds of acoustic melodies, perfect for peaceful moments and introspection.', '2004-11-05'),
(5505, 81443060, 'Feel-Good Classics', 'Rediscover timeless classics that never fail to uplift your spirits and put a smile on your face.', '2007-06-22'),
(5506, 79472726, 'Late Night Jams', 'Dive into a collection of smooth tunes and soulful beats, ideal for unwinding during the late hours.', '2005-08-12'),
(5507, 84259553, 'Sunday Morning Melodies', 'Start your day on a tranquil note with this selection of gentle melodies and soothing harmonies.', '2002-03-18'),
(5508, 88196267, 'Workout Beats', 'Get pumped up and motivated with this high-energy playlist filled with tracks to fuel your workout sessions.', '2010-10-31'),
(5509, 54314743, 'Rainy Day Relaxation', 'Embrace the cozy vibes of a rainy day with this playlist designed to help you unwind and find peace.', '2008-07-09'),
(5510, 73249313, 'Summer Breeze Playlist', 'Capture the essence of summer with this playlist featuring sunny tunes and feel-good vibes.', '2001-12-20'),
(5511, 99802145, 'Retro Rewind', 'Take a trip down memory lane with this nostalgic playlist showcasing hits from decades past.', '2006-02-14'),
(5512, 86465466, 'Dance Party Essentials', 'Turn up the volume and hit the dance floor with this collection of infectious beats and party anthems.', '2004-05-06'),
(5513, 44001355, 'Study Focus Soundtrack', 'Stay focused and productive with this curated selection of instrumental tracks perfect for studying and concentration.', '2019-09-03'),
(5514, 99802145, 'Indie Gems Collection', 'Discover hidden musical treasures and indie favorites in this eclectic compilation of indie tracks.', '2023-04-27'),
(5516, 86246535, 'Jazz Lounge Favorites', 'Settle into the smooth ambiance of a jazz lounge with this playlist featuring classic jazz standards and contemporary favorites.', '2017-08-10'),
(5517, 81443060, 'R&B Slow Jams', 'Set the mood with this sultry collection of R&B slow jams, perfect for romantic evenings and late-night vibes.', '2005-01-13'),
```

(5518, 37437849, 'Rock Legends Playlist', 'Rock out to iconic anthems and legendary tracks from the greatest rock bands of all time.', '2021-06-15'),
(5519, 37437849, 'Country Roads Playlist', 'Hit the open road with this playlist celebrating the heart and soul of country music, perfect for scenic drives and adventures.', '2018-11-08'),
(5520, 70201199, 'EDM Energy Boost', 'Get ready to party with this high-octane playlist featuring pulsating beats and electrifying EDM tracks.', '2016-03-11'),
(5521, 86465466, 'Soulful Grooves', 'Immerse yourself in the soulful rhythms and smooth melodies of this playlist, guaranteed to lift your spirits and move your soul.', '2012-07-30'),
(5522, 98526216, '90s Nostalgia Hits', 'Relive the magic of the 90s with this throwback playlist featuring all your favorite hits from the decade.', '2014-10-19'),
(5523, 99802145, 'Classical Masterpieces', 'Experience the beauty and grandeur of classical music with this curated selection of timeless masterpieces by renowned composers.', '2000-12-12'),
(5524, 29173027, 'Pop Perfection Playlist', 'Indulge in the catchy hooks and infectious melodies of this playlist featuring the best in pop music.', '2002-11-11');

```
CREATE TABLE createsplaylist_n1 (
   PlaylistID INT PRIMARY KEY,
   Name VARCHAR(255) NOT NULL,
   Description VARCHAR(1000),
   CreationDate DATE NOT NULL
);
```

INSERT INTO Playlists (PlaylistID, Name, Description, CreationDate)
VALUES
(5501, 'Guitar Hero Anthems', 'Shred like a rockstar with this playlist featuring epic guitar solos and iconic rock anthems, perfect for air guitar enthusiasts.', '2003-05-17'),
(5502, 'Chill Vibes Mix', 'Unwind with a collection of laid-back tunes perfect for relaxing evenings and mellow moments.', '2006-09-29'),
(5503, 'Road Trip Anthems', 'Hit the road with this upbeat playlist featuring an eclectic mix of songs to keep you energized on your journey.', '2009-04-08'),
(5504, 'Acoustic Serenade', 'Immerse yourself in the soothing sounds of acoustic melodies, perfect for peaceful moments and introspection.', '2004-11-05'),
(5505, 'Feel-Good Classics', 'Rediscover timeless classics that never fail to uplift your spirits and put a smile on your face.', '2007-06-22'),
(5506, 'Late Night Jams', 'Dive into a collection of smooth tunes and soulful beats, ideal for unwinding during the late hours.', '2005-08-12'),
(5507, 'Sunday Morning Melodies', 'Start your day on a tranquil note with this selection of gentle melodies and soothing harmonies.', '2002-03-18'),
(5508, 'Workout Beats', 'Get pumped up and motivated with this high-energy playlist filled with tracks to fuel your workout sessions.', '2010-10-31'),
(5509, 'Rainy Day Relaxation', 'Embrace the cozy vibes of a rainy day with this playlist designed to help you unwind and find peace.', '2008-07-09'),
(5510, 'Summer Breeze Playlist', 'Capture the essence of summer with this playlist featuring sunny tunes and feel-good vibes.', '2001-12-20'),
(5511, 'Retro Rewind', 'Take a trip down memory lane with this nostalgic playlist showcasing hits from decades past.', '2006-02-14'),
(5512, 'Dance Party Essentials', 'Turn up the volume and hit the dance floor with this collection of infectious beats and party anthems.', '2004-05-06'),
(5513, 'Study Focus Soundtrack', 'Stay focused and productive with this curated selection of instrumental tracks perfect for studying and concentration.', '2019-09-03'),
(5514, 'Indie Gems Collection', 'Discover hidden musical treasures and indie favorites in this eclectic compilation of indie tracks.', '2023-04-27'),

(5516, 'Jazz Lounge Favorites', 'Settle into the smooth ambiance of a jazz lounge with this playlist featuring classic jazz standards and contemporary favorites.', '2017-08-10'),
(5517, 'R&B Slow Jams', 'Set the mood with this sultry collection of R&B slow jams, perfect for romantic evenings and late-night vibes.', '2005-01-13'),
(5518, 'Rock Legends Playlist', 'Rock out to iconic anthems and legendary tracks from the greatest rock bands of all time.', '2021-06-15'),
(5519, 'Country Roads Playlist', 'Hit the open road with this playlist celebrating the heart and soul of country music, perfect for scenic drives and adventures.', '2018-11-08'),
(5520, 'EDM Energy Boost', 'Get ready to party with this high-octane playlist featuring pulsating beats and electrifying EDM tracks.', '2016-03-11'),
(5521, 'Soulful Grooves', 'Immerse yourself in the soulful rhythms and smooth melodies of this playlist, guaranteed to lift your spirits and move your soul.', '2012-07-30'),
(5522, '90s Nostalgia Hits', 'Relive the magic of the 90s with this throwback playlist featuring all your favorite hits from the decade.', '2014-10-19'),
(5523, 'Classical Masterpieces', 'Experience the beauty and grandeur of classical music with this curated selection of timeless masterpieces by renowned composers.', '2000-12-12'),
(5524, 'Pop Perfection Playlist', 'Indulge in the catchy hooks and infectious melodies of this playlist featuring the best in pop music.', '2002-11-11');

```
CREATE TABLE createsplaylist_n2 (
   PlaylistID INT PRIMARY KEY,
   UserID INT,
   FOREIGN KEY (UserID) REFERENCES USER1(UserID)
);
INSERT INTO Playlists (PlaylistID, UserID)
VALUES
(5501, 29173027),
(5502, 13361601),
(5503, 70201199),
(5504, 37437849),
(5505, 81443060),
(5506, 79472726),
(5507, 84259553),
(5508, 88196267),
(5509, 54314743),
(5510, 73249313),
(5511, 99802145),
(5512, 86465466),
(5513, 44001355),
(5514, 99802145),
(5516, 86246535),
(5517, 81443060),
(5518, 37437849),
(5519, 37437849),
(5520, 70201199),
(5521, 86465466),
(5522, 98526216),
(5523, 99802145),
(5524, 29173027);
```

```
CREATE TABLE includesong (
    SongID INT PRIMARY KEY,
    Title VARCHAR(255) NOT NULL,
    ArtistID INT,
    ReleaseDate DATE,
    Duration TIME NOT NULL,
    FOREIGN KEY (ArtistID) REFERENCES artist(ArtistID)
);
INSERT INTO Songs (SongID, Title, ArtistID, ReleaseDate, Duration)
VALUES
(1111, 'Jagada Thom', 23015, 2011, '0:03:37'),
(1112, 'Aariro', 23015, 2011, '0:04:11'),
(1113, 'Kadha Solla Poraen', 23015, 2011, '0:03:27'),
(1114, 'Life Is Beautiful', 23015, 2011, '0:04:53'),
(1115, 'Pa Pa Pappa', 23015, 2011, '0:03:22'),
(1116, 'Vennilave Vennilave', 23015, 2011, '0:04:58'),
(1117, 'Vizhigalil Oru Vaanavil', 23015, 2011, '0:02:56'),
(1121, 'Yethi Yethi', 36973, 2008, '0:03:17'),
(1122, 'Oh Shanthi Shanthi', 36973, 2008, '0:03:23'),
(1123, 'Nenjukkul Peidhidum', 36973, 2008, '0:02:47'),
(1124, 'Mundhinam', 36973, 2008, '0:05:34'),
(1125, 'Ava Enna', 36973, 2008, '0:04:58'),
(1126, 'Annul Maelae', 36973, 2008, '0:04:34'),
(1127, 'Adiyae Kolluthey', 36973, 2008, '0:04:12'),
(1131, 'Yedho Ondru Ennai', 40043, 2010, '0:03:34'),
(1132, 'Thuli Thuli', 40043, 2010, '0:04:43'),
(1133, 'Suthudhe Suthudhe', 40043, 2010, '0:04:56'),
(1134, 'Poongatre Poongatre', 40043, 2010, '0:05:20'),
(1135, 'En Kadhal Solla', 40043, 2010, '0:04:54'),
(1136, 'Adada Mazhaida', 40043, 2010, '0:04:27'),
(1141, 'Venmathiye', 91515, 2001, '0:05:27'),
(1142, 'Vaseegara', 91515, 2001, '0:04:59'),
(1143, 'Vaerene Ivanyaro', 91515, 2001, '0:05:25'),
(1144, 'Poopol Poopol', 91515, 2001, '0:01:57'),
(1145, 'Ore Nyabagam', 91515, 2001, '0:01:54'),
(1146, 'Ooh Mama', 91515, 2001, '0:04:38'),
(1147, 'Nenjai Poopol', 91515, 2001, '0:01:01'),
(1148, 'Maddy Maddy', 91515, 2001, '0:01:15'),
(1149, 'Azhagiya Theeye', 91515, 2001, '0:05:55'),
(1151, 'Party song', 86350, 2017, '0:03:23'),
(1152, 'Liberty Song', 86350, 2017, '0:05:23'),
(1161, 'Yaro Yarodi', 36973, 2000, '0:06:05'),
(1162, 'Snehidhane', 36973, 2000, '0:05:07'),
(1163, 'September Madham', 36973, 2000, '0:05:57'),
(1164, 'Pachchai Nirame', 36973, 2000, '0:04:35'),
(1165, 'Kadhal Sadugudu', 36973, 2000, '0:05:56'),
(1166, 'Evano Oruvan', 36973, 2000, '0:03:44'),
(1167, 'Alai Payuthey', 36973, 2000, '0:05:46'),
(1171, 'Hey Piriyame Piriyame', 98429, 2018, '0:03:51'),
(1172, 'Kannamma Kanvizhi', 98429, 2018, '0:03:26'),
(1181, 'Walking Through The Rainbow Theme Music', 91515, 2004, '0:03:22'),
(1182, 'Ninaithu Ninaithu Parthen', 91515, 2004, '0:04:37'),
(1183, 'Naam Vayathukku', 91515, 2004, '0:05:07'),
(1184, 'Kan Pesum Varthaigal', 91515, 2004, '0:05:50'),
(1185, 'Idhu Porkkalama', 91515, 2004, '0:03:09'),
```

```
(1191, 'Othaiyadi Pathayila', 74305, 2018, '0:02:45'),
(1192, 'Vaayadi Petha Pulla', 74305, 2018, '0:04:04'),
(1193, 'Oonjala Oonjala', 74305, 2018, '0:04:31'),
(2001, 'Paisa Note', 62244, 2019, '0:04:31'),
(2002, 'Yaara Comali', 62244, 2019, '0:03:30'),
(2003, 'Hi Sonna Pothum', 62244, 2019, '0:03:45'),
(2004, 'Nanba Nanba', 62244, 2019, '0:04:43');

CREATE TABLE includesong_n1(
   SongID INT PRIMARY KEY,
   Title VARCHAR(255) NOT NULL,
   ReleaseDate DATE,
   Duration TIME NOT NULL,
);
\ INSERT INTO Songs (SongID, Title, ReleaseDate, Duration)
VALUES
(1111, 'Jagada Thom', 2011, '0:03:37'),
(1112, 'Aariro', 2011, '0:04:11'),
(1113, 'Kadha Solla Poraen', 2011, '0:03:27'),
(1114, 'Life Is Beautiful', 2011, '0:04:53'),
(1115, 'Pa Pa Pappa', 2011, '0:03:22'),
(1116, 'Vennilave Vennilave', 2011, '0:04:58'),
(1117, 'Vizhigalil Oru Vaanavil', 2011, '0:02:56'),
(1121, 'Yethi Yethi', 2008, '0:03:17'),
(1122, 'Oh Shanthi Shanthi', 2008, '0:03:23'),
(1123, 'Nenjukkul Peidhidum', 2008, '0:02:47'),
(1124, 'Mundhinam', 2008, '0:05:34'),
(1125, 'Ava Enna', 2008, '0:04:58'),
(1126, 'Annul Maelae', 2008, '0:04:34'),
(1127, 'Adiyae Kolluthey', 2008, '0:04:12'),
(1131, 'Yedho Ondru Ennai', 2010, '0:03:34'),
(1132, 'Thuli Thuli', 2010, '0:04:43'),
(1133, 'Suthudhe Suthudhe', 2010, '0:04:56'),
(1134, 'Poongatre Poongatre', 2010, '0:05:20'),
(1135, 'En Kadhal Solla', 2010, '0:04:54'),
(1136, 'Adada Mazhaida', 2010, '0:04:27'),
(1141, 'Venmathiye', 2001, '0:05:27'),
(1142, 'Vaseegara', 2001, '0:04:59'),
(1143, 'Vaerene Ivanyaro', 2001, '0:05:25'),
(1144, 'Poopol Poopol', 2001, '0:01:57'),
(1145, 'Ore Nyabagam', 2001, '0:01:54'),
(1146, 'Ooh Mama', 2001, '0:04:38'),
(1147, 'Nenjai Poopol', 2001, '0:01:01'),
(1148, 'Maddy Maddy', 2001, '0:01:15'),
(1149, 'Azhagiya Theeye', 2001, '0:05:55'),
(1151, 'Party song', 2017, '0:03:23'),
(1152, 'Liberty Song', 2017, '0:05:23'),
(1161, 'Yaro Yarodi', 2000, '0:06:05'),
(1162, 'Snehidhane', 2000, '0:05:07'),
(1163, 'September Madham', 2000, '0:05:57'),
(1164, 'Pachchai Nirame', 2000, '0:04:35'),
(1165, 'Kadhal Sadugudu', 2000, '0:05:56'),
(1166, 'Evano Oruvan', 2000, '0:03:44'),
(1167, 'Alai Payuthey', 2000, '0:05:46'),
(1171, 'Hey Piriyame Piriyame', 2018, '0:03:51'),
```

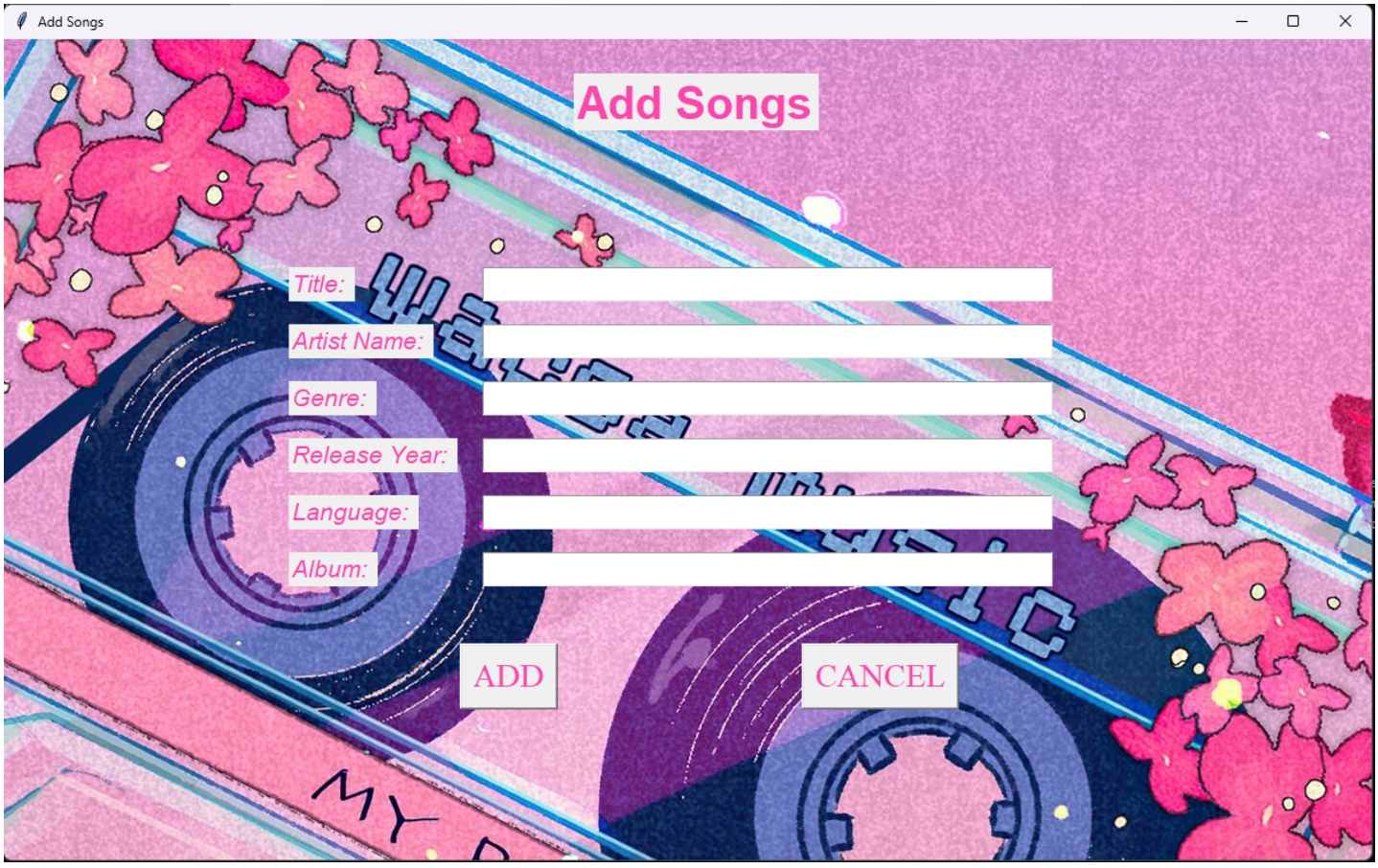
```
(1172, 'Kannamma Kanvizhi', 2018, '0:03:26'),
(1181, 'Walking Through The Rainbow Theme Music', 2004, '0:03:22'),
(1182, 'Ninaithu Ninaithu Parthen', 2004, '0:04:37'),
(1183, 'Naam Vayathukku', 2004, '0:05:07'),
(1184, 'Kan Pesum Varthaigal', 2004, '0:05:50'),
(1185, 'Idhu Porkkalama', 2004, '0:03:09'),
(1191, 'Othaiyadi Pathayila', 2018, '0:02:45'),
(1192, 'Vaayadi Petha Pulla', 2018, '0:04:04'),
(1193, 'Oonjala Oonjala', 2018, '0:04:31'),
(2001, 'Paisa Note', 2019, '0:04:31'),
(2002, 'Yaara Comali', 2019, '0:03:30'),
(2003, 'Hi Sonna Pothum', 2019, '0:03:45'),
(2004, 'Nanba Nanba', 2019, '0:04:43');

CREATE TABLE includesong_n2(
   SongID INT PRIMARY KEY,
   ArtistID INT
   FOREIGN KEY (ArtistID) REFERENCES artist(ArtistID)
);
INSERT INTO Songs (SongID, ArtistID)
VALUES
(1111, 23015),
(1112, 23015),
(1113, 23015),
(1114, 23015),
(1115, 23015),
(1116, 23015),
(1117, 23015),
(1121, 36973),
(1122, 36973),
(1123, 36973),
(1124, 36973),
(1125, 36973),
(1126, 36973),
(1127, 36973),
(1131, 40043),
(1132, 40043),
(1133, 40043),
(1134, 40043),
(1135, 40043),
(1136, 40043),
(1141, 91515),
(1142, 91515),
(1143, 91515),
(1144, 91515),
(1145, 91515),
(1146, 91515),
(1147, 91515),
(1148, 91515),
(1149, 91515),
(1151, 86350),
(1152, 86350),
(1161, 36973),
(1162, 36973),
(1163, 36973),
```

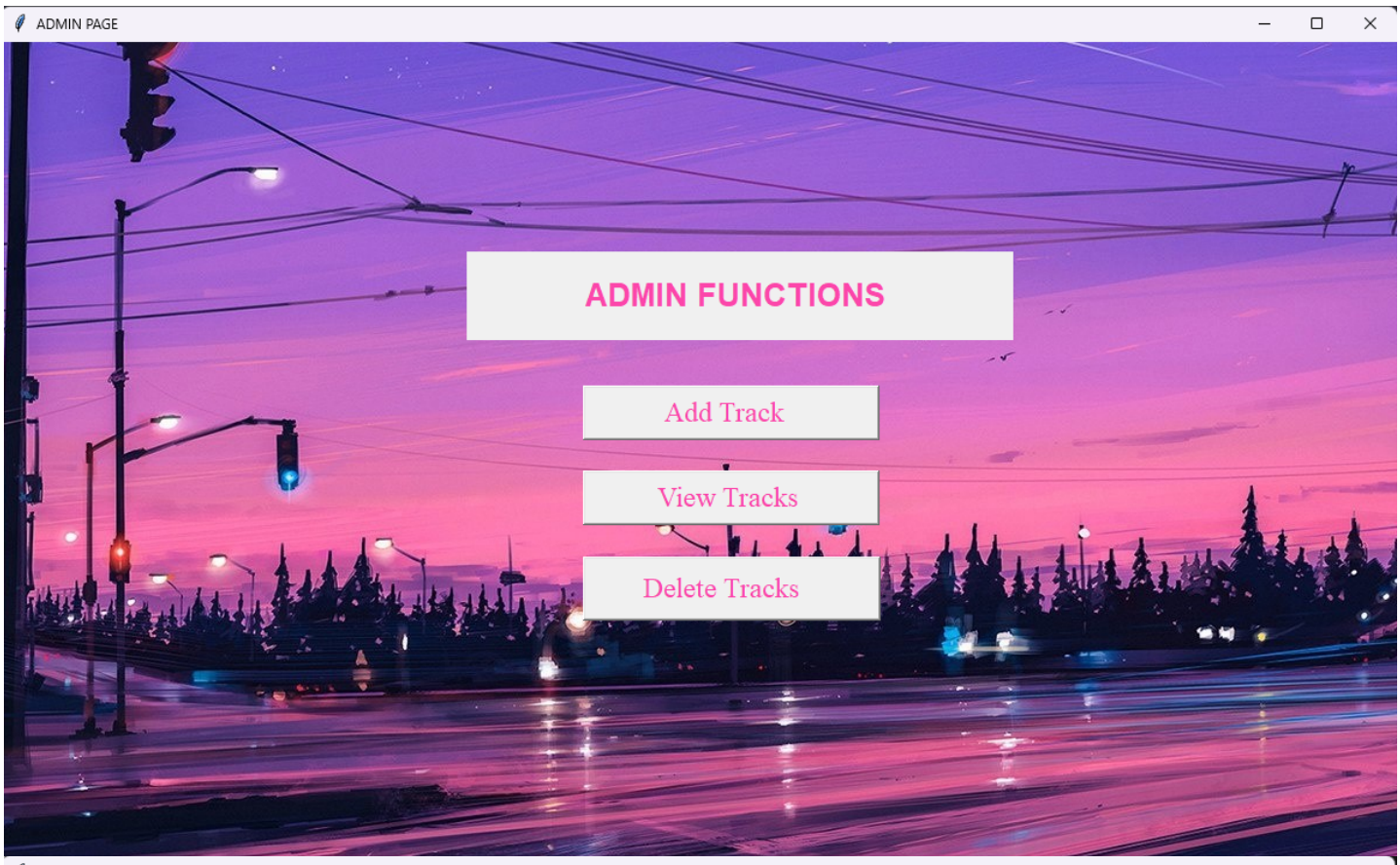
(1164, 36973),
(1165, 36973),
(1166, 36973),
(1167, 36973),
(1171, 98429),
(1172, 98429),
(1181, 91515),
(1182, 91515),
(1183, 91515),
(1184, 91515),
(1185, 91515),
(1191, 74305),
(1192, 74305),
(1193, 74305),
(2001, 62244),
(2002, 62244),
(2003, 62244),
(2004, 62244);

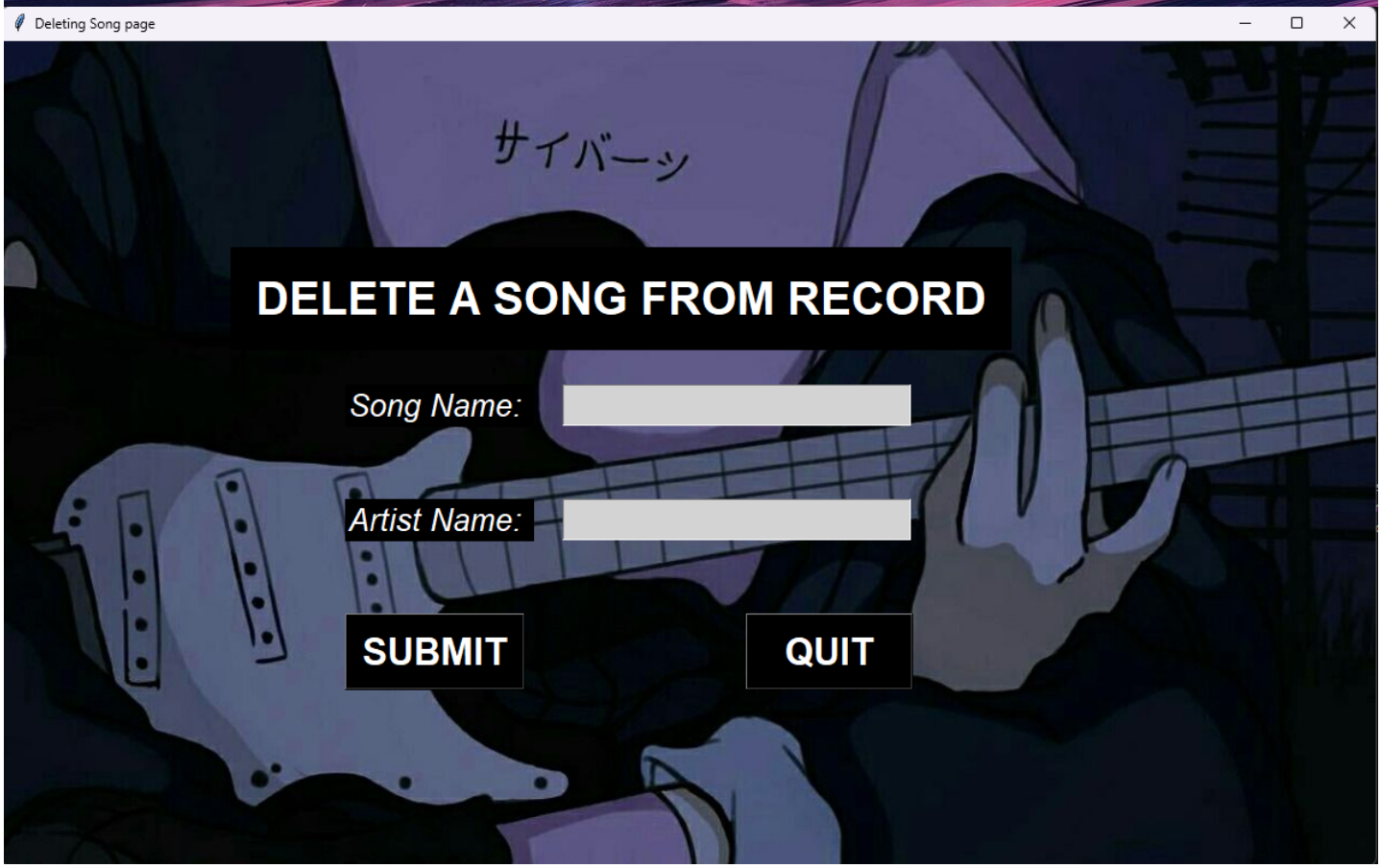
CHAPTER 7

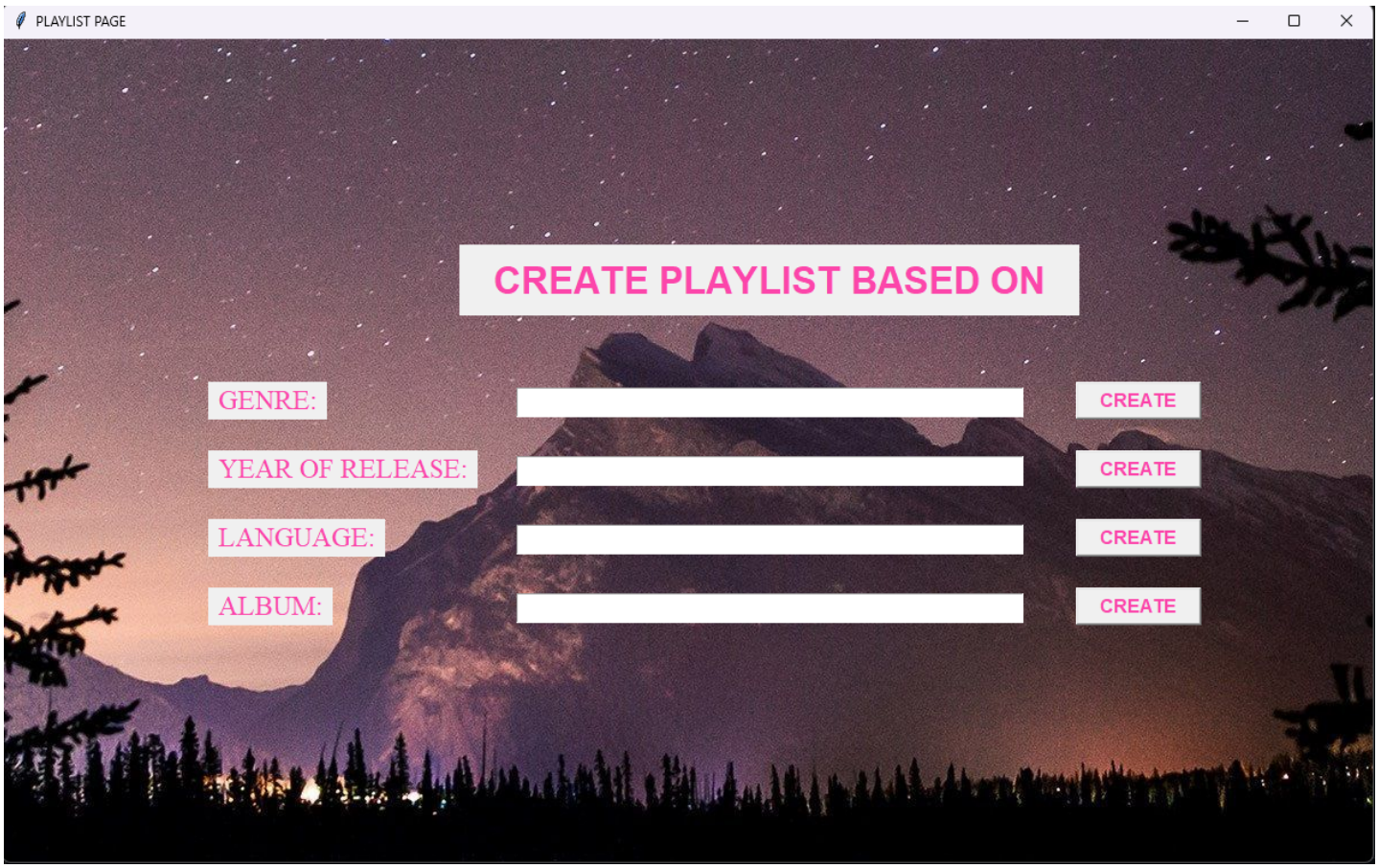
RESULT AND DISCUSSION

RESULTS:




ADMIN PAGE
ADMIN FUNCTIONS
Add Track
View Tracks
Delete Tracks

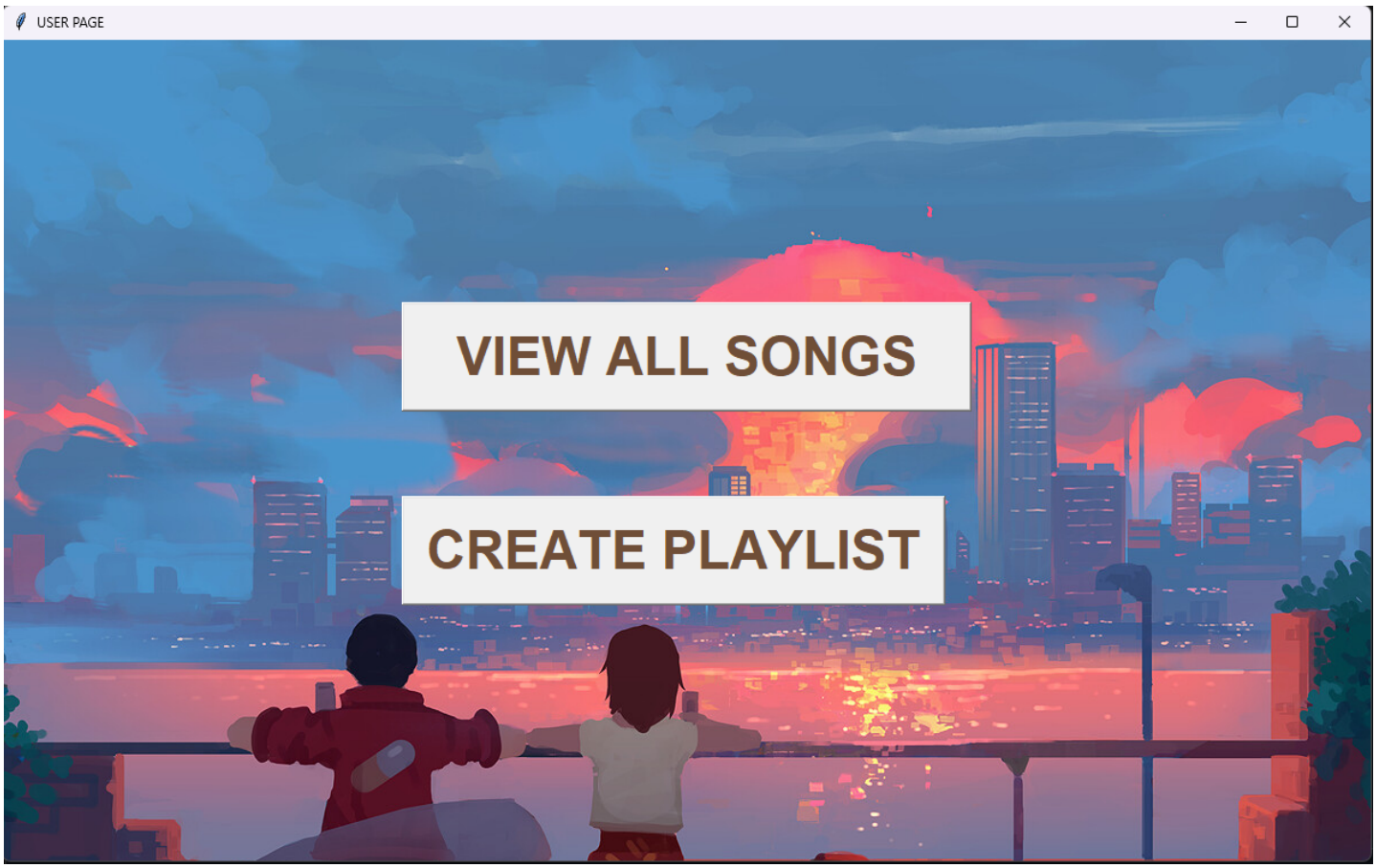

Deleting Song page
DELETE A SONG FROM RECORD
Song Name:
Artist Name:
SUBMIT
QUIT



View All Songs

Song Name	Artist Name	Genre	Release Year	Language
Attention	Charlie Puth	Pop Rock	2017	English
Channa Mereya	Arijit Singh	Hindustani Classical Music	2017	Hindi
Mere Dholna	shreya ghoshal	Hindustani Classical Music	2007	Hindi
Counting Stars	One Republic	Pop Rock	2013	English
Spread Love	Sushin Shyam	Indie Pop	2019	Malayalam

Close


USER PAGE
VIEW ALL SONGS
CREATE PLAYLIST

CHAPTER 8

CONCLUSION:

In conclusion, the project aimed to design and implement a robust music library management system (MLMS) using principles of database management. Through the process of database design, normalization, and schema construction, the project successfully organized music-related data into separate entities, including Artist, Genre, Album, and Song. Normalization techniques were applied to minimize redundancy, optimize data integrity, and establish efficient relationships between entities. Set operations and SQL queries were employed to manage and retrieve data effectively, while triggers were utilized to enforce data integrity constraints.

The resulting MLMS provides a structured and scalable platform for storing, organizing, and accessing music content. Users can easily navigate through the library, search for songs based on various criteria, and create personalized playlists. The system ensures data consistency, accuracy, and reliability through well-defined relationships and enforced constraints.

Moving forward, potential enhancements to the MLMS could include implementing user authentication and authorization mechanisms, incorporating advanced search functionalities, and expanding the database to include additional features such as user preferences, song ratings, and social sharing capabilities. Overall, the project demonstrates the importance of proper database design and management in building efficient and functional systems to meet the needs of music enthusiasts and industry professionals alike.

CHAPTER 9

FUTURE SCOPE:

User Authentication and Authorization: Implement secure user authentication mechanisms such as login credentials, password hashing, and session management. Additionally, incorporate role-based access control to ensure that users only have access to features and data appropriate for their roles.

Advanced Search Functionality: Enhance the search functionality to allow users to search for songs based on various criteria such as genre, artist, album, release year, and song duration. Implement filters, sorting options, and advanced search queries to improve the user experience.

User Preferences and Recommendations: Introduce features for users to customize their preferences, such as favorite genres, artists, or albums. Utilize machine learning algorithms to analyze user behavior and provide personalized song recommendations and playlists.

Social Sharing and Collaboration: Enable users to share their playlists, favorite songs, and music discoveries with friends and followers on social media platforms. Implement collaboration features to allow users to create collaborative playlists and share music experiences with others.

Song Ratings and Reviews: Allow users to rate and review songs, albums, and artists. Aggregate user ratings to generate popularity rankings and recommendations for trending music content.

Mobile Application Development: Develop a mobile application for the MLMS to provide users with access to their music library on the go. Ensure cross-platform compatibility and seamless synchronization of data between the web and mobile platforms.

Integration with Music Streaming Services: Explore partnerships and integrations with popular music streaming services to expand the MLMS's catalog of available songs and provide users with access to a wider range of music content.

Analytics and Reporting: Implement analytics tools to track user interactions, monitor system performance, and generate insights into music consumption patterns and trends. Use these insights to optimize the MLMS's features and offerings.

Internationalization and Localization: Support multiple languages and regional preferences to cater to a diverse user base. Adapt the MLMS's interface, content, and recommendations to suit the cultural and linguistic preferences of different regions.

Continuous Improvement and Maintenance: Regularly update and maintain the MLMS to address user feedback, fix bugs, and introduce new features and improvements. Stay up-to-date with industry trends and technological advancements to ensure the MLMS remains relevant and competitive in the market.

By embracing these future scopes, the MLMS can evolve into a comprehensive and innovative platform that offers an immersive music experience for users while staying ahead of the curve in the dynamic music industry landscape.

CHAPTER 10

10. ONLINE COURSE CERTIFICATE:

NPTEL Online Certification

(Funded by the MoE, Govt. of India)

This certificate is awarded to

SHABARINATHAN KRV

for successfully completing the course

Data Base Management System

with a consolidated score of **53** %

Online Assignments	21.46/25	Proctored Exam	31.5/75

Total number of candidates certified in this course: **6225**

Jan-Mar 2024

(8 week course)

Prof. Haimanti Banerji
Coordinator, NPTEL
IIT Kharagpur

Indian Institute of Technology Kharagpur

Roll No: NPTEL24CS21S544106885 To verify the certificate  No. of credits recommended: 2 or 3