ANP-C8415

StudentID: AF0471570

Name: Akula Sindhu

Database Design for

Music Library Management System

MUSIC LIBRARY MANAGEMENT SYSTEM

A Music Library Management System is a software application designed to organize, store, and manage digital music collections efficiently. It allows users to add, update, delete, and categorize songs based on metadata such as title, artist, album, genre, and release year. The system often includes features like playlist creation, search functionality, sorting, and filtering. It can be used by individuals, DJs, or music libraries to maintain their music databases and enhance accessibility. Advanced systems may also support audio playback, cloud synchronization, and integration with streaming services.

The Music Library Management System is a digital solution designed to efficiently organize and manage a collection of music files. It enables users to store, retrieve, and play music while maintaining detailed metadata such as title, artist, album, genre, duration, and release year. The system allows for the easy addition, deletion, and editing of songs, ensuring that the music library remains up to date and well-structured. Users can create and manage playlists, sort music by various attributes, and use advanced search and filter options to quickly find desired tracks.

It often includes a built-in audio player with features like play, pause, skip, and repeat. Some systems also support importing music from folders or exporting playlists. For enhanced usability, user authentication can be included to manage access rights and personalize user experiences. The application can be built as a desktop, web, or mobile platform depending on user needs.

Technologies commonly used include HTML, CSS, JavaScript for the frontend, and Python (Flask/Django), Node.js, or PHP for the backend. Databases like MySQL, MongoDB, or PostgreSQL are used to store music data. Optional features may include cloud storage integration, rating systems, user reviews, and multi-device synchronization. This system is particularly useful for individuals, DJs, libraries, and institutions looking to manage music collections efficiently and enjoyably.

Entities:

- Artists
- Albums
- Songs
- Users
- Playlists
- PlaylistSongs

Entity-Relationships:

The image depicts an Entity-Relationship (ER) diagram for a Music Library Management System. There are six main entities: **Users, Artists, Albums, Songs, Playlists, and Playlist Songs**.
Each user has a username and password, and can create multiple playlists.
Artists have unique IDs and names and can be linked to multiple songs and albums.
Albums contain songs and are associated with both artists and users.
Each song has a song ID, title, and duration and can belong to multiple playlists.
The Songs and Playlists relationship is many-to-many, managed via the Playlist Songs entity.

Users can create multiple playlists (many-to-many relationship shown with 'm'). The ER diagram reflects relationships such as artist-song, album-artist, and user-playlist clearly. This structure ensures efficient management of music data and relationships between entities.

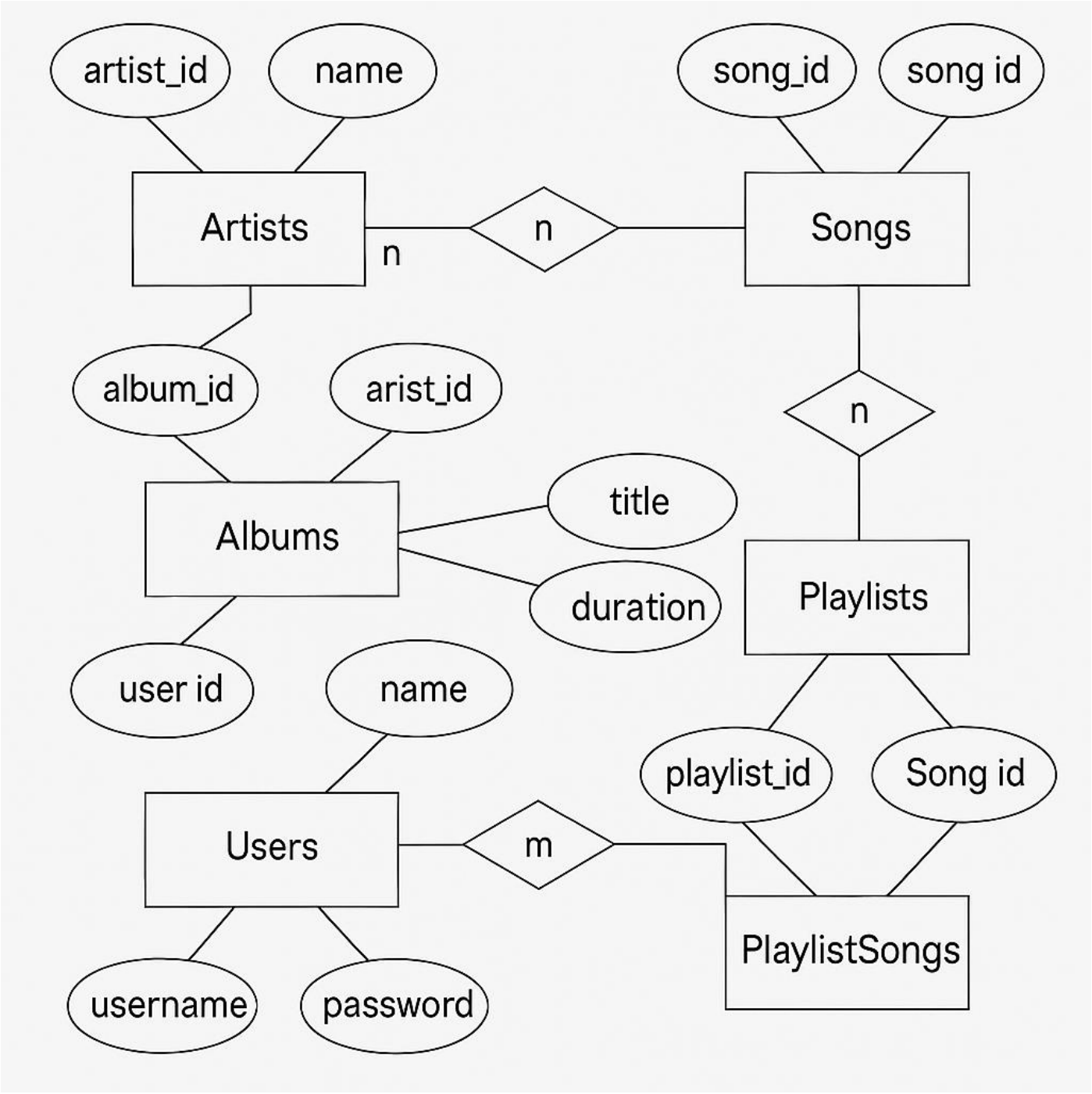
1. **Many-to-Many Relationships**: Songs ↔ Artists, Songs ↔ Playlists (via Playlist Songs).
2. **Users**: Can create and manage multiple playlists, secured with login credentials.
3. **Albums**: Linked to both Artists and Users; each has a title and duration.
4. **Playlist Songs**: A junction table to manage many-to-many song-playlist associations.
5. **Normalization**: The data is normalized into logical entities with unique IDs and clear attributes.

Attributes are the fields of entities that refer to the characteristics or properties of entities within a database.

1. **Artists**
   - artist_id (PK): Unique ID for each artist
   - name: Name of the artist
   - genre: Music genre
2. **Albums**
   - album_id (PK): Unique ID for each album
   - artist_id (FK): Reference to the artist who created the album
   - title: Album name
   - release_year: Year album was release
3. **Songs**
   - song_id (PK): Unique ID for each song
   - album_id (FK): Reference to the album it belongs to
   - title: Title of the song
   - duration: Length of the song (time)
   - `play_count`: Number of times the song was played (default = 0)
4. **Users**
   - user_id (PK): Unique ID for each user
   - username: Name chosen by the user
   - password: Password for user login
5. **Playlists**
   - playlist_id (PK): Unique ID for each playlist
   - user_id (FK): Reference to the user who created the playlist
   - name: Playlist name

6. **Playlist Songs**
    - playlist_id (FK, PK): Reference to playlist
    - song_id (FK, PK): Reference to song
    - Composite PK (playlist_id, song_id): Ensures a song can appear only once per playlist

ENTITY RELATIONSHIP DIAGRAM - MUSIC LIBRARY MANAGEMENT SYSTEM



ENTITY RELATIONSHIP DIAGRAM - MUSIC LIBRARY MANAGEMENT SYSTEM