PGDM- Big Data Analytics

Big Data Management & Analytics

Project Report

ER Diagram – Music Streaming Database

Submitted to: Prof. Amarnath Mitra

Submitted By: Group 6

Krishnendu Adhikary – 055022

Mohit Agarwal – 055024

Introduction

The purpose of this report is to document the **database schema and entity-relationship diagram (ERD)** for a **music streaming system**. This system is designed to efficiently manage various components, including **users, playlists, tracks, albums, artists, and music labels**, ensuring seamless data organization, retrieval, and management.

With the growing demand for digital music platforms, a **well-structured and normalized database** is essential to maintain **data integrity, minimize redundancy, and optimize query performance**. This report provides a detailed overview of the **database schema**, outlining the structure and attributes of each entity. Additionally, it explains the **relationships between tables**, ensuring that data dependencies are properly managed.

To uphold **data consistency and eliminate anomalies**, the database is designed to comply with the **First Normal Form (1NF) and Second Normal Form (2NF)**. This ensures that all tables have a **unique identifier (primary key), no repeating groups, and that partial dependencies are eliminated**. The report also highlights **primary and foreign key constraints**, which define how entities are linked and maintain referential integrity.

By implementing a **structured and normalized approach**, this database enhances **efficiency, scalability, and ease of access**. The following sections delve into the **database schema, entity relationships, and ER diagram**, providing a comprehensive understanding of the system's design and functionality.

Data Description

The dataset represents a **Music Streaming System**, designed to manage users, playlists, tracks, albums, artists, music labels, and contracts efficiently. It is structured to ensure data integrity and normalization while supporting seamless organization and retrieval.

The User table stores details such as UserID (primary key), Name, Email, and SubscriptionType, categorizing users into Premium, Free, and Standard subscriptions. The MusicLabel table captures information about record labels, including LabelID, Name, EstablishedYear, ContactInfo, and Headquarters. The Artist table maintains details of musicians, identified by ArtistID, with attributes like StageName, Birthdate, Genre, and LabelID (foreign key linking to MusicLabel).

The Contract table records agreements between artists and music labels, containing ContractID, StartDate, EndDate, ContractTerms, ArtistID, and LabelID. Albums are stored in the Album table, uniquely identified by AlbumID, and include Title, ReleaseDate, Genre, LabelID, and ArtistID. Each album contains multiple tracks stored in the Track table, which includes TrackID, Title, Duration, and AlbumID.

Playlists are user-generated collections of tracks, stored in the Playlist table with PlaylistID, Name, CreatedBy, CreationDate, and UserID. The PlaylistTrack table acts as a **junction table**, mapping many-to-many relationships between playlists and tracks, with composite keys PlaylistID and TrackID.

This structured dataset ensures **data consistency, referential integrity, and efficient query execution**, supporting a well-optimized **music streaming platform** with a focus on personalization and seamless content organization.

Schema for the Chosen Database

The chosen database is a **relational database** designed for a **music streaming platform**, ensuring **efficient data organization, integrity, and normalization**. It consists of multiple entities such as **Users, Playlists, Tracks, Albums, Artists, Music Labels, and Contracts**, with relationships structured to **eliminate redundancy and optimize retrieval**.

The User table manages subscriber details, including UserID (primary key), Name, Email, and SubscriptionType (Premium, Free, Standard). Playlists created by users are stored in the Playlist table, which includes PlaylistID, Name, CreatedBy, and CreationDate, ensuring seamless user engagement.

Music content is categorized into **Albums and Tracks**, where the Album table records AlbumID, Title, ReleaseDate, Genre, LabelID, and ArtistID. The Track table stores TrackID, Title, Duration, and AlbumID, maintaining a **one-to-many** relationship between albums and tracks.

Artists and record labels play a crucial role in the music industry. The Artist table captures ArtistID, StageName, Birthdate, and Genre, linking them to a **music label** via LabelID. The MusicLabel table includes LabelID, Name, EstablishedYear, and ContactInfo, defining the **one-to-many** relationship between labels and artists. Additionally, contractual agreements between artists and labels are recorded in the Contract table, storing ContractID, StartDate, EndDate, and ContractTerms.

To support **many-to-many** relationships, the PlaylistTrack table serves as a **junction table**, mapping tracks to multiple playlists. This structure ensures a **highly scalable, well-optimized database**, facilitating seamless user interaction, efficient content management, and personalized music recommendations.

Entities and Attributes

Below are the key entities along with their attributes:

- **User**
  - UserID (PK)
  - Name
  - Email
  - SubscriptionType
- **Playlist**
  - PlaylistID (PK)
  - Name
  - CreatedBy
  - CreationDate
  - UserID (FK)
- **PlaylistTrack** (Bridge Table)
  - PlaylistID (FK)
  - TrackID (FK)
- **Track**
  - TrackID (PK)
  - Title
  - Duration
  - AlbumID (FK)
- **Album**
  - AlbumID (PK)
  - Title
  - ReleaseDate
  - Genre
  - LabelID (FK)
  - ArtistID (FK)

- **Artist**
    - ArtistID (PK)
    - Name
    - StageName
    - Birthdate
    - Genre
    - ContactInfo
- **Music Label**
    - LabelID (PK)
    - Name
    - EstablishedYear
    - ContactInfo
    - Headquarters
- **Contract**
    - ContractID (PK)
    - StartDate
    - EndDate
    - ContractTerms
    - ArtistID (FK)
    - LabelID (FK)

Relationship Between Tables

The relationships between the entities are described as follows:

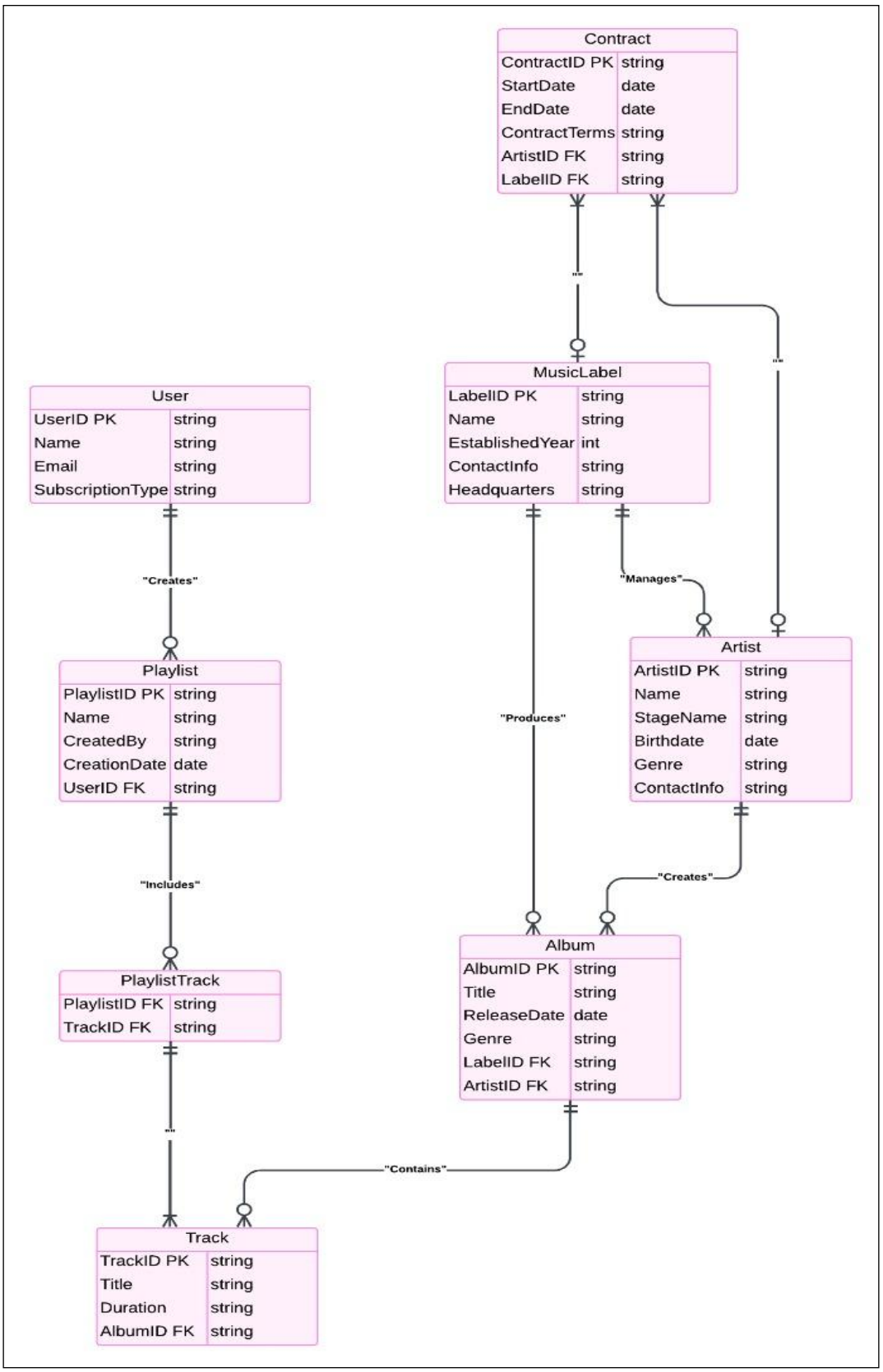
- A **User** can create multiple **Playlists** (One-to-Many).
- A **Playlist** can contain multiple **Tracks**, and a **Track** can belong to multiple **Playlists** (Many-to-Many via PlaylistTrack).
- An **Album** contains multiple **Tracks** (One-to-Many).
- An **Artist** creates multiple **Albums** (One-to-Many).
- A **Music Label** manages multiple **Artists** (One-to-Many).
- A **Contract** is established between an **Artist** and a **Music Label** (Many-to-Many).

Primary and Foreign Key Table

Table	Primary Key	Foreign Key(s)
User	UserID	-
Playlist	PlaylistID	UserID
PlaylistTrack	-	PlaylistID, TrackID
Track	TrackID	AlbumID
Album	AlbumID	LabelID, ArtistID
Artist	ArtistID	-
MusicLabel	LabelID	-
Contract	ContractID	ArtistID, LabelID

ER Diagram

The ER diagram represents the relationships and attributes of the system, ensuring normalization and eliminating redundancy.



Conclusion

This database schema is designed to efficiently manage a **music streaming system** by implementing a **well-structured relational database** with normalized tables and well-defined relationships. By ensuring **data integrity, scalability, and optimal performance**, the schema effectively organizes and maintains critical information related to **users, playlists, tracks, albums, artists, music labels, and contracts** while eliminating redundancy.

A key aspect of this design is the **bridge table (PlaylistTrack)**, which facilitates **many-to-many** relationships between playlists and tracks, ensuring that the database adheres to **Second Normal Form (2NF) and Third Normal Form (3NF)**. By eliminating **partial and transitive dependencies**, the schema ensures **data consistency and referential integrity** across all entities. The use of **primary keys** uniquely identifies each record, while **foreign keys** establish and maintain logical connections between tables, preventing anomalies and duplication.

Additionally, the schema is **highly scalable**, allowing the system to handle a growing number of users, tracks, and playlists without compromising efficiency. Its structured approach supports seamless **content management, efficient data retrieval, and system reliability**, making it a strong foundation for a modern music streaming platform.

Furthermore, this design is adaptable for **future enhancements**, including **advanced analytics, personalized recommendations, and AI-driven music discovery**, ensuring long-term viability and continuous improvement of the system.