Analyzing Spotify Data To Improve Music Discovery

Sanjukta Baruah
School of Engg. and Applied Sciences
Data Science
Buffalo, United States
sbaruah@buffalo.edu

Priyadarshini Raghavendra
School of Engg. and Applied Sciences
Data Science
Buffalo, United States
praghave@buffalo.edu

Sujay Shrivastava
School of Engg. and Applied Sciences
Data Science
Buffalo, United States
sujayshr@buffalo.edu

***Abstract*—The initial dataset underwent a normalization process, evolving into a schema with diverse relations and relation types, unleashing powerful capabilities for insightful data analysis and a deeper understanding of the complex musical universe within Spotify's expansive database**

I. Problem Statement

The music industry stands at the cusp of transformation, with streaming platforms like Spotify fundamentally altering the way people engage with and consume music. In this dynamic landscape, understanding the nuanced intricacies of user behavior, preferences, and industry trends is pivotal. Spotify, as one of the most prominent music streaming services, accumulates vast datasets encompassing user interactions, playlists, music attributes, and more. The question at hand is how to unlock the full potential of this data goldmine. To effectively address this challenge, we propose the creation of a comprehensive and normalized database, underpinned by a well-structured Entity-Relationship (E/R) diagram. The database will serve as the bedrock for a holistic analysis, uncovering insights, trends, and correlations across a multitude of dimensions.

II. Addressing Efficiency: Elevating Management and Performance

The problem we aim to solve is to efficiently manage and query a Spotify dataset containing a vast amount of user-related information, music content, and their relationships. We start from ground zero, recognizing the dataset's complexity, along with redundant or unnecessary attributes, which can lead to suboptimal database performance and increased storage requirements. Our goal is to design and maintain a normalized Spotify dataset to address these challenges and enhance data management, query efficiency, and overall database performance.

To solve this problem, we plan to:

1. Normalize the Database: We will restructure the dataset to eliminate data redundancy and ensure that each piece of information is stored only once. This normalization process will involve organizing the data into separate tables and establishing relationships between them to minimize data duplication.

2. Improve Data Integrity: By adhering to normalization principles, we can enhance data integrity and reduce the risk of anomalies such as update, insert, or delete anomalies. This will result in more reliable and accurate data.

3. Optimize Query Performance: Normalized data can significantly improve query performance. With appropriate indexing and efficient querying strategies, we can reduce query execution times and enhance the user experience for Spotify users and data analysts.

4. Minimize Storage Requirements: By eliminating redundant data, we can reduce storage requirements and potentially lower infrastructure costs, making the dataset more cost-effective to maintain.

5. Enhance Data Management: The normalized dataset will provide a clear and organized structure that simplifies data management, making it easier to maintain and expand in the future.

In summary, our approach involves restructuring the Spotify dataset into a normalized form, which will lead to better data organization, improved query performance, reduced storage costs, and enhanced data management capabilities. This, in turn, will empower Spotify and its stakeholders to make informed decisions, provide more personalized recommendations, and create a richer and more diverse musical landscape.

III. Key Queries and Future Scope

The project's core questions, intertwined with the entities and attributes within the database, are as follows:

A. User Behavior and Preferences:

1) What are the most popular genres in different regions, and how does this relate to demographics?
2) Does track duration affect user engagement, and are there regional differences in listening habits?
3) Are user playlist titles linked to regional demographics, and which playlists are preferred in different areas?
4) Which tracks have the largest following, and how is this distributed geographically?

B. Exploring User Listening Patterns:

1) User Behavior Analysis: Can the Spotify dataset reveal trends in account creation dates, and are there specific

timeframes when user sign-ups spike? Additionally, does the dataset show how the conversion rate from free to premium subscription has evolved over the past year?
2) Time-to-Subscription & Churn: Based on the data, what is the average time it takes for Spotify users to upgrade to a premium subscription after signing up for a free account? Can we identify users who have been free subscribers for an extended period, and how many of them eventually churn from the platform?
3) Marketing Impact & Demographics: Does the Spotify dataset provide insights into the impact of specific marketing campaigns or promotions on the initiation of premium subscriptions? Moreover, can we discern any correlations between user demographics (age, location) and their subscription behavior within the dataset?
4) User Engagement & Acquisition Channels: Within the Spotify dataset, can we find evidence that correlates user engagement metrics (such as daily song plays) with their likelihood to upgrade to premium services? Additionally, are there discernible patterns regarding which acquisition channels (e.g., social media, referrals) have the highest conversion rates for premium subscriptions?
5) Long-Term User Value: Based on the data, can we determine whether users who subscribe to premium within a certain timeframe have a higher long-term value to Spotify, and can we quantify this value to inform strategic decisions for user retention and monetization?

C. Music Consumption Trends:

1) How does music consumption change over time, and are there seasonal trends in track popularity?
2) Do music preferences vary with time, demographics, and geography?

D. Artist and Album Insights:

1) Which artists have the largest music catalog, and how does their popularity vary by region?
2) Do album attributes like title and release date affect user engagement across regions?

E. Economic and Marketing Analysis:

1) How do marketing campaigns affect user behavior and subscriptions, and is there a correlation with transaction data?
2) Can we identify the impact of music releases on user activity and subscription transactions?

F. Podcast Analysis:

1) Podcast genres favored by users and regional topic preferences.
2) Effect of the number of podcast episodes on user engagement, and its correlation with regional demographics.
3) Influence of podcast hosts on popularity among users in various regions.

IV. Understanding the Impact of Streaming Music

The world of music has transformed with the advent of streaming services like Spotify. These services have brought music to our fingertips and have generated a wealth of data. This data is crucial because the music industry is highly competitive. Knowing how people interact with music is essential for success. Our project, which involves organizing and analyzing Spotify's data, can make a significant difference. It can help the music industry create better strategies to cater to the preferences of users. This means more effective marketing, personalized song suggestions, and discovering new talent. Additionally, we aim to understand how different aspects of music affect how much people like it. This knowledge can lead to more exciting and varied music. Moreover, the correlation between attributes will help organizations gain profit by enabling them to tailor their music content and promotions to align more precisely with the preferences of their target audience, ultimately driving revenue and profitability. In short, our project is a big step in changing the music industry through data, and this is vital for the industry's success and growth.

V. Project contribution and importance

Our project's significance lies in its potential to transform the music industry. By thoroughly analyzing Spotify's dataset, we offer essential insights into user behavior and industry trends. Data normalization empowers stakeholders to optimize their offerings for evolving consumer expectations, resulting in improved marketing, personalized recommendations, and the discovery of emerging talent. Our understanding of the interplay between music attributes and user engagement drives innovation in music production, creating a more diverse and engaging musical landscape. This project is positioned to advance the music industry toward a more user-centric and artist-friendly future. In conclusion, "Revolutionizing Music Analytics" represents a pivotal step in reshaping the music industry.

VI. Database Superiority Over Excel

A database is necessary over an Excel file because of its efficiency in handling large datasets, scalability for future growth, and the ability to execute complex queries. Unlike Excel, a database can accommodate vast amounts of data more effectively, and it can grow with Spotify's expanding catalog. Complex queries, vital for in-depth analysis, are often impractical in Excel but can be executed seamlessly in a database. This shift to a database is essential for a robust, adaptable, and analytically powerful foundation.

VII. Target Users

A. Music Analysts and Researchers

These professionals will use the database for in-depth analyses on user behavior, preferences, and industry trends. For example, they might study the impact of track duration on user engagement.

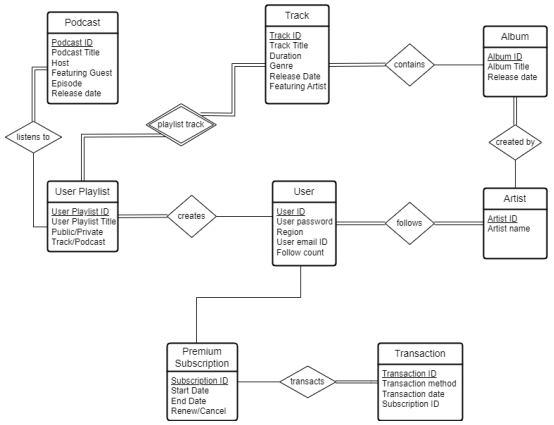

Fig. 1. Unnormalised E/R

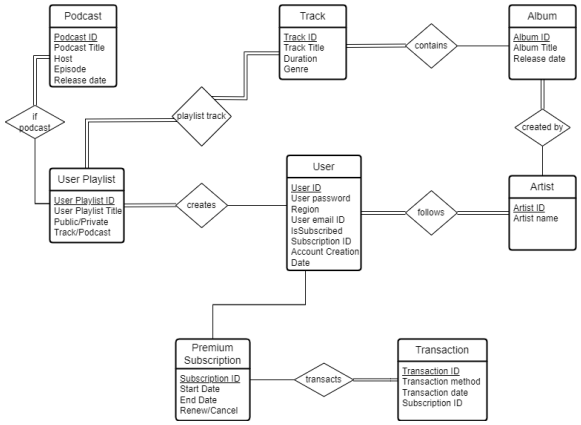

Fig. 2. Normalised E/R

B. Marketing Teams

Marketing teams within the music industry can harness the database to refine their strategies. For instance, they might use the data to optimize marketing campaigns by understanding how user demographics and geographic location affect music preferences on the platform.

C. Playlist Curators

Playlist curators aim to craft engaging playlists for users. They'll depend on the database to identify trends in music consumption and to personalize recommendations. In a real-life scenario, a playlist curator might use the data to tailor playlists based on regional preferences and user behavior.

D. Artists and Labels

Artists and record labels are keen to understand the impact of their music. They'll use the database to gauge which tracks have the most fervent following and how it's distributed across regions. This knowledge can guide their promotional efforts and touring plans.

E. Database Administrators

Database administrators play a crucial role in maintaining and optimizing the database. They ensure its reliability, security, and performance. In the scenario, a database administrator might perform routine maintenance, address any technical issues, and ensure the data remains accessible and secure for all users.

VIII. E/R Diagram

A. Transforming an Unnormalized E/R into a Normalized E/R:

In a strategic effort to streamline and enhance the database structure, several significant modifications have been made to the Spotify dataset, optimizing data organization and relationships. The process of converting unnormalized data into normalized data is driven by the following key observations:

- *Account Creation Date:* Added to the User table to capture the date of user registration.
- *IsSubscribed and SubscriptionID:* These attributes in the User table were revised to simplify the relationship between users and their premium subscriptions. Now, SubscriptionID is a foreign key referencing the Premium Subscription table, ensuring that a user's subscription information is linked more directly.
- *Track/Podcast Attribute:* Added to the User Playlist table to distinguish between playlists and podcasts, making it easier to manage and categorize content.
- *Follow Count:* Follower count in the User table is removed, as it would make more sense to query to get follower count using user IDs.
- *Featuring Guest, Featuring Artist:* Both Featuring Guest and Featuring Artist details will already be integrated into the title, eliminating the need for separate columns. We've opted for a restricted level of atomicity, concentrating solely on the primary host for a podcast and the main artist for a track.

B. Entities and Attributes Description:

This E/R diagram illustrates the structure of a database designed for Spotify, a popular music streaming service. Below, you'll find a comprehensive breakdown of the various entities and their associated attributes, each with detailed descriptions:

C. Database Schema for Spotify

User Table:

- UserID (`integer, primary key`) – UserID is unique to every user.
- User E-mail (`varchar`) – Email ID is considered as demographics of the user.
- Password (`varchar`) – Password will be stored in the database using MD5 encryption.
- Region (`varchar`) – Records from where the application is being used.
- Account Creation Date (`date`) – Date of user's registration on Spotify.
- IsSubscribed (`Boolean`) – Stores information on whether the user has a monthly premium subscription.

- SubscriptionID (integer, foreign key) – References Premium subscription table, if IsSubscribed is True, else will be NULL.

Premium Subscription Table:

- Subscription ID (integer, primary key) – Each user will have a unique subscription ID, if IsSubscribed attribute of User table is True.
- Start Date (date) – Start date of subscription. Default value will be 01/01/2018.
- End Date (date) – End date of subscription. Attribute will be updated when the user cancels the subscription. Default value will be 01/01/2099.
- Renew/Cancel (Boolean) – Stores information on renewal or cancellation of the subscription.

Transaction Table:

- Transaction ID (integer, primary key) – Every transaction will have a unique transaction ID.
- Transaction method (varchar) – Every transaction can have a different method of payment.
- Transaction date (date) – Stores the date of the transaction made. If the user renews the subscription, the transaction date will keep updating with the same date every month.
- Subscription ID (integer, foreign key) – References Premium subscription table.

User Playlist Table:

- User Playlist ID (integer, primary key) – Every playlist will have a unique ID.
- User Playlist Title (varchar) – Title of the playlist.
- Public/Private (Boolean) – Whether the playlist created by the user is marked private or public. This will decide the granting of access of other users to a particular playlist.
- Track/Podcast (Boolean) – In a playlist, this attribute is used to differentiate between a playlist and a podcast.

Podcast Table:

- Podcast ID (integer, primary key) – Every podcast will have a unique ID.
- Podcast title (varchar) – Title of the podcast.
- Host (varchar) – Considered demographics of podcasts.
- Episode (integer) – Every podcast can have multiple episodes.
- Release date (date) – Records the date of release of the podcast.

Track Table:

- TrackID (integer, primary key) – Every track has a unique ID.
- Track Title (varchar) – Every track has a title.
- Duration (datetime) – Records the duration of each track.
- Genre (varchar) – Indicates the genre of each track. Genres are repetitive values; there exist a limited number of genres.

Album Table:

- Album ID (integer, primary key) – Every album has a unique ID.
- Album Title (varchar) – Every album will have a title.
- Release Date (date) – Records the date of release of an album.

Artist Table:

- Artist ID (integer, primary key) – Every artist will have a unique ID.
- Artist name (varchar) – Name of the artist.

D. Justification for Final E/R Being in BCNF

BCNF, or Boyce-Codd Normal Form, is a rule for organizing data in a relational database:

1) The database must be in 3NF, which means data is organized with minimal redundancy and no indirect relationships.
2) For every non-trivial rule that connects data in the database, the part on the left side of the rule must be able to uniquely identify records. This ensures data is structured to avoid problems like redundancy.

We are listing the functional dependencies for each Entity Table below.

User Table:

- UserID → User E-mail
- UserID → Password
- UserID → Region
- UserID → Account Creation Date
- UserID → IsSubscribed
- IsSubscribed → SubscriptionID

All functional dependencies are on the primary key (UserID). There are no partial or transitive dependencies. This table satisfies the requirements for both 3NF and BCNF.

Premium Subscription Table:

- SubscriptionID → Start Date
- SubscriptionID → End Date
- SubscriptionID → Renew/Cancel

All functional dependencies are on the primary key (SubscriptionID). There are no partial or transitive dependencies. This table satisfies the requirements for both 3NF and BCNF.

Transaction Table:

- Transaction ID → Transaction Method
- Transaction ID → Transaction Date
- Subscription ID → Transaction ID

All functional dependencies are on the primary key (Transaction ID and Subscription ID). There are no partial or transitive dependencies. This table satisfies the requirements for both 3NF and BCNF.

User Playlist Table:

- User Playlist ID → User Playlist Title

- User Playlist ID → Public/Private
- User Playlist ID → Track/Podcast

All functional dependencies are on the primary key (User Playlist ID). There are no partial or transitive dependencies. This table satisfies the requirements for both 3NF and BCNF.

Podcast Table:

- Podcast ID → Podcast Title
- Podcast ID → Host
- Podcast ID → Episode
- Podcast ID → Release Date

All functional dependencies are on the primary key (Podcast ID). There are no partial or transitive dependencies. This table satisfies the requirements for both 3NF and BCNF.

Track Table:

- TrackID → Track Title
- TrackID → Duration
- TrackID → Genre

All functional dependencies are on the primary key (TrackID). There are no partial or transitive dependencies. This table satisfies the requirements for both 3NF and BCNF.

Album Table:

- Album ID → Album Title
- Album ID → Release Date

All functional dependencies are on the primary key (Album ID). There are no partial or transitive dependencies. This table satisfies the requirements for both 3NF and BCNF.

Artist Table:

- Artist ID → Artist Name

All functional dependencies are on the primary key (Artist ID). There are no partial or transitive dependencies. This table satisfies the requirements for both 3NF and BCNF.

Follows Table (Relationship between Users and Artists):

- (UserID, ArtistID) → Follower Count

All functional dependencies are on the combined primary key (UserID, ArtistID). There are no partial or transitive dependencies. This table satisfies the requirements for both 3NF and BCNF.

REFERENCES

[1] https://medium.com/towards-data-engineering/design-the-database-for-a-system-like-spotify-95ffd1fb5927
[2] https://www.geeksforgeeks.org/python-pandas-working-with-dates-and-times/.
[3] https://www.geeksforgeeks.org/python-faker-library/
[4] https://www.geeksforgeeks.org/types-of-keys-in-relational-model-candidate-super-primary-alternate-and-foreign/
[5] https://www.holistics.io/blog/top-5-free-database-diagram-design-tools/
[6] https://app.diagrams.net/