**MUSIC RECOMMENDATION USING SQL**

**Table of Contents**

1. **Introduction**
2. **Objectives**
3. **Database Design**

**-Entities**

**-Relationships**

**4. SQL Queries and Analysis**

**-User Profiling**

**-Song Recommendations**

**5. Results**

**6. Challenges Faced**

**7. Conclusion**

**8. Future Work**

**Introduction**

The Music Recommendation System using SQL is a project aimed at creating a personalized music recommendation engine. Using SQL, this system analyzes user interactions with a music platform to recommend songs, artists, or genres that align with the user’s preferences. This project showcases the power of SQL in handling large-scale data and extracting actionable insights for personalized content delivery.

**Objectives**

* To design and implement a relational database that stores user interaction data, song details, and artist information.
* To write efficient SQL queries to generate personalized music recommendations based on user history and preferences.
* To analyse music consumption patterns and provide insights into user behaviour.

**Database Design**

**Entities**

**Users:** Stores user profiles and preferences.

**Songs:** Contains details about songs, including title, artist, album, and genre.

**Artists:** Information about music artists.

**Albums:** Details of music albums.

**User-playlist**: Details of playlists if users.

**Recommendations: It provides recommended details**

* **One-to-Many:** A user can have multiple interactions with different songs.
* **Many-to-One:** A song can belong to only one artist.
* **Many-to-Many:** Users can like multiple songs, and songs can be liked by multiple users.

**SQL Queries and Analysis**

**User Profiling**

This query analyzes user interaction data to profile users based on their most listened-to genres, preferred artists, and other behavior.

**Song Recommendations**

A complex SQL query that suggests new songs to users by finding similarities between their listening habits and popular trends in the database.

**Data Insights**

* Most active users based on the number of interactions.
* Trending artists and genres over a given period.

**Results**

The system successfully provided personalized recommendations and insights. For example, users with a preference for pop music were recommended the latest trending pop songs and artists.

**Challenges Faced**

* Managing large datasets efficiently with SQL.
* Optimizing query performance to ensure quick recommendations.
* Handling ambiguous or sparse data in the user interactions.

**Conclusion**

This project demonstrated the effective use of SQL for building a recommendation system, highlighting SQL's power in data analysis and its ability to manage relational data efficiently.

**Future Work**

* Integrating real-time data processing for live recommendations.
* Extending the database to include more user demographics and interaction types.