Music Streaming Application –Database Design Project Summary

This project presents the design and implementation of a relational database for a music streaming application. It was developed as part of an academic exercise to demonstrate practical database modeling, normalization to Third Normal Form (3NF), and application-level CRUD operations using Python and MySQL.

Application Description

The Music Streaming App is a simplified but robust platform that allows users to create accounts, build playlists, upload profile images, and listen to songs uploaded by artists. Key features include:

- User Management
- Song Management
- Playlist Management
- Profile Image Storage

Database Design Commentary

Entity Integrity

- All tables are equipped with Primary Keys (PK)
- Composite primary keys are used where needed, such as the PlaylistSong table

Referential Integrity

- Foreign Keys (FK) are used extensively
- Referential integrity is enforced to prevent orphan records

Data Integrity

- Appropriate data types and constraints are implemented
- Input validation is handled in Python

Normalization

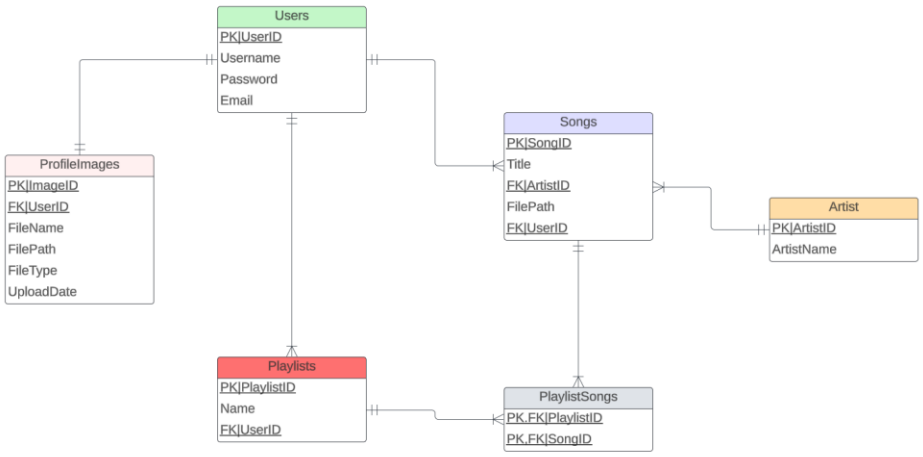
- The database is normalized to Third Normal Form (3NF)

Higher Normal Forms

- BCNF is satisfied since all determinants are candidate keys

- 4NF and 5NF are considered satisfied due to lack of multi-valued and join dependencies

Entity-Relationship Diagram (ERD)



Implementation and Testing

CRUD Operations

Implemented in Python using pymysql. Includes:

- insert_user() – With password hashing (bcrypt)
- insert_playlist() – Playlist creation
- insert_song() – Song entry and metadata
- insert_profile_image() – Store file name and path
- link_song_to_playlist() – Manage many-to-many song/playlist relations

Security

- bcrypt is used for hashing passwords
- File paths for profile images are stored safely

Testing and Validation

- Table creation and constraint validation
- Record insertion and retrieval
- Retrieval of songs by artist

- Playlist creation and retrieval by user ID
- Update and delete operations

Project Files

- schema.sql – SQL script for table creation
- crud_operations.py – Python functions for CRUD with validation
- queries.sql – Search queries and aggregation
- test_cases.md – Documented test scenarios
- erd.png – ERD diagram
- schema_screenshot.png – Completed schema view

SQL Schema Screenshot

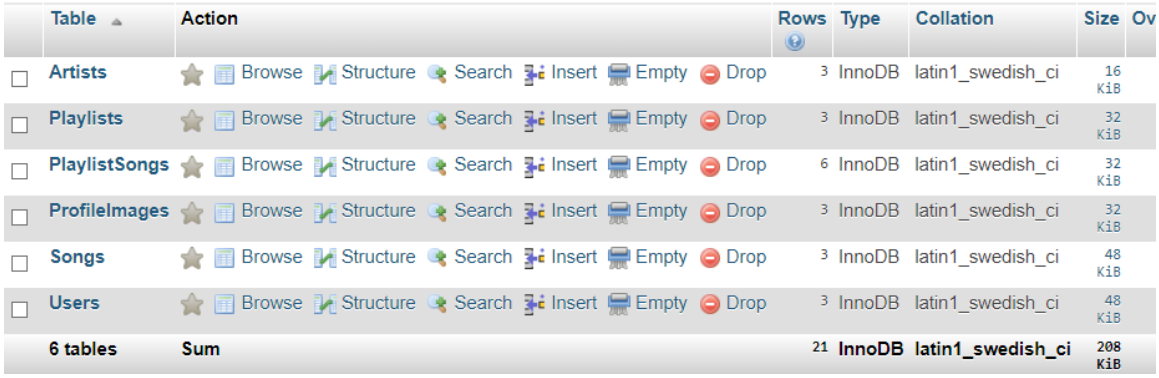


	Table	Action	Rows	Type	Collation	Size	Ov
☐	Artists	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	16 KiB	
☐	Playlists	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	32 KiB	
☐	PlaylistSongs	Browse Structure Search Insert Empty Drop	6	InnoDB	latin1_swedish_ci	32 KiB	
☐	ProfileImages	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	32 KiB	
☐	Songs	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	48 KiB	
☐	Users	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	48 KiB	
	6 tables	Sum	21	InnoDB	latin1_swedish_ci	208 KiB	

Conclusion

The project delivers a well-structured and secure database system for a music streaming app. It demonstrates proper schema design, normalization, and practical CRUD operations. With user-centric features and modular code, the system supports scalability, performance &clarity.

Prepared by: Nsikanabasi B. Umoh

Date: November 2024