DATABASE MANAGEMENT SYSTEM - CSA0593

ASSIGNMENT 5

R. YASHWANTH VARMA

192311392

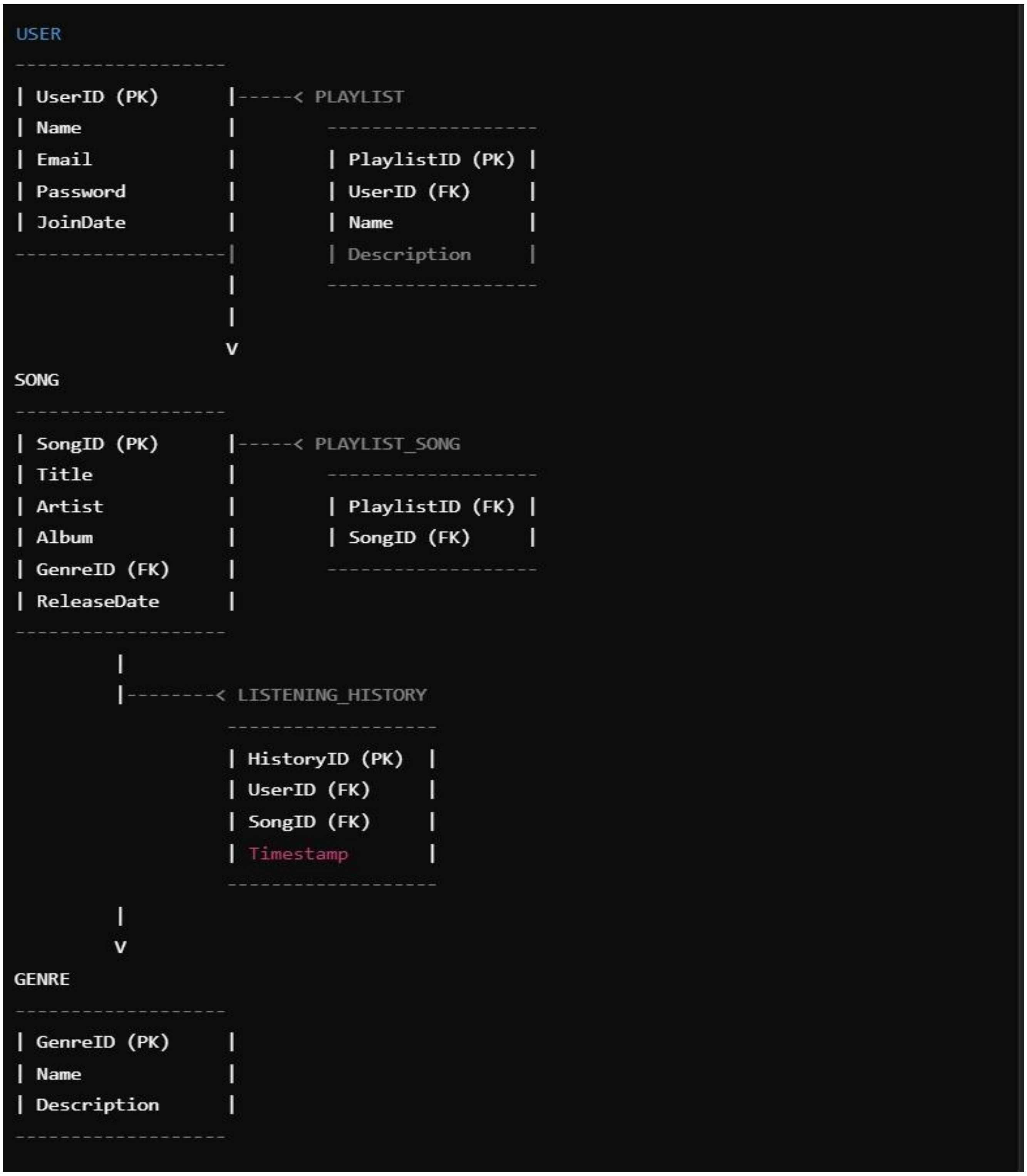
QUESTION:

Build a database schema to support a music streaming platform, including users, playlists, and listening history.

- Define tables for users, songs, playlists, listening history, and genres.
- Write queries to recommend songs based on user listening patterns and genre preferences.
- Implement procedures to update playlist suggestions and identify trending songs.
- Describe optimization strategies to support quick access to frequently-streamed songs and playlists.

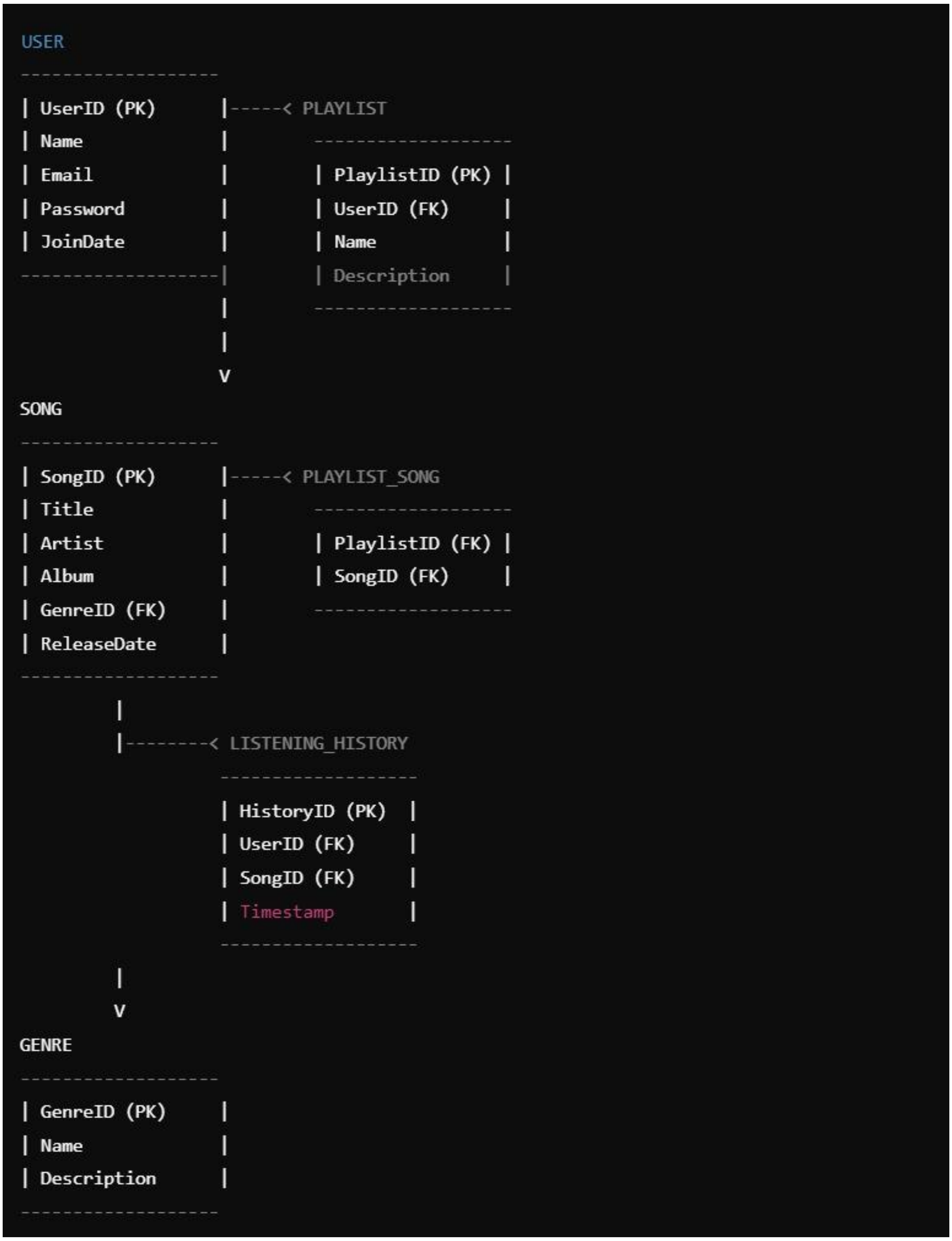
ANSWER:

CONCEPTUAL ER DIAGRAM:



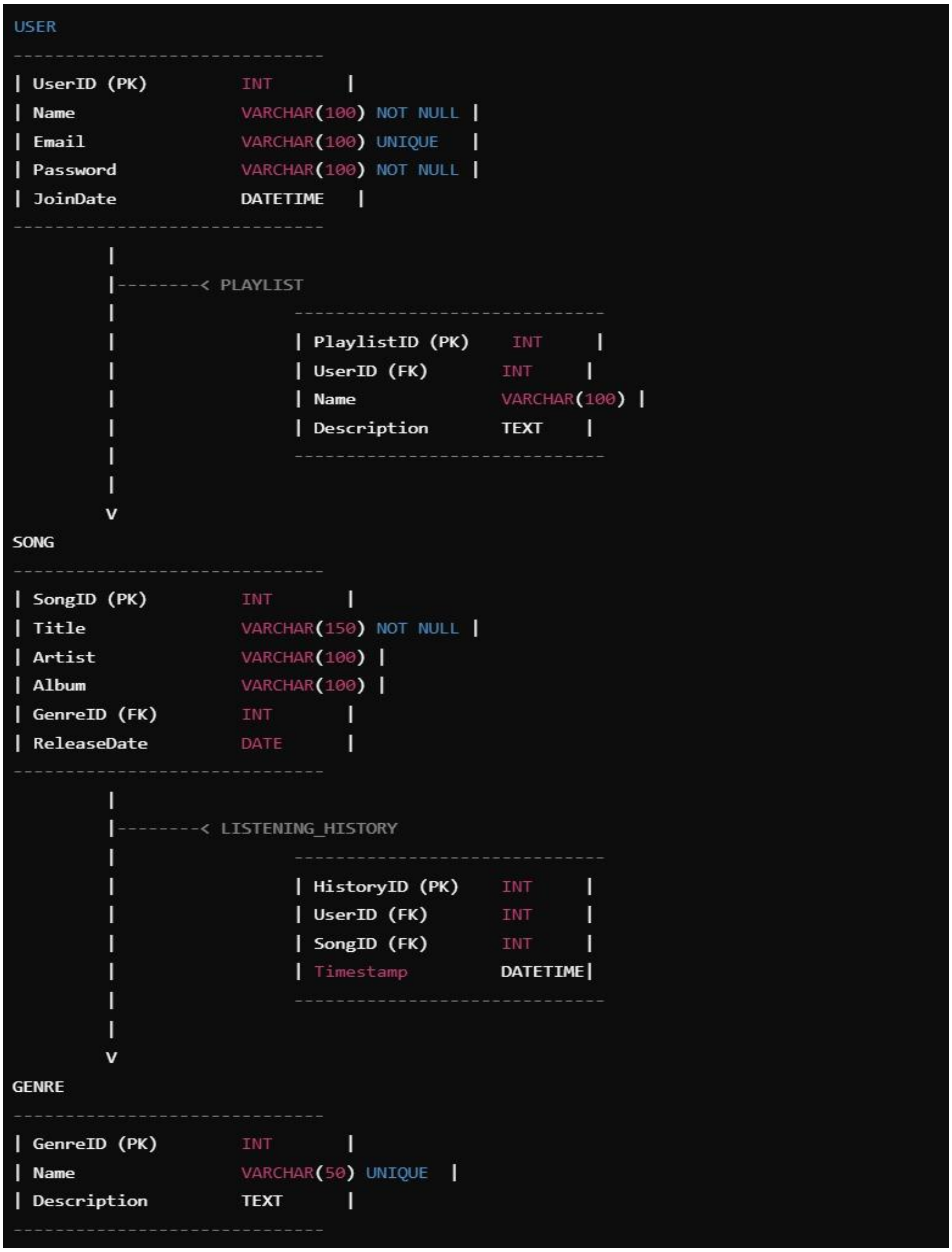
```
USER
-------------------
| UserID (PK)       |-----< PLAYLIST
| Name              |         -------------------
| Email             |         | PlaylistID (PK) |
| Password          |         | UserID (FK)     |
| JoinDate          |         | Name            |
-------------------|          | Description     |
                    |         -------------------
                    |
                    v
SONG
-------------------
| SongID (PK)       |-----< PLAYLIST_SONG
| Title             |         -------------------
| Artist            |         | PlaylistID (FK) |
| Album             |         | SongID (FK)     |
| GenreID (FK)      |         -------------------
| ReleaseDate       |
-------------------
          |
          |--------< LISTENING_HISTORY
                    -------------------
                    | HistoryID (PK)  |
                    | UserID (FK)     |
                    | SongID (FK)     |
                    | Timestamp       |
                    -------------------
          |
          v
GENRE
-------------------
| GenreID (PK)      |
| Name              |
| Description       |
-------------------
```

LOGICAL E.R DIAGRAM:



```
USER
-------------------
| UserID (PK)       |-----< PLAYLIST
| Name              |        -------------------
| Email             |        | PlaylistID (PK) |
| Password          |        | UserID (FK)     |
| JoinDate          |        | Name            |
-------------------|        | Description     |
                    |        -------------------
                    |
                    V
SONG
-------------------
| SongID (PK)       |-----< PLAYLIST_SONG
| Title             |        -------------------
| Artist            |        | PlaylistID (FK) |
| Album             |        | SongID (FK)     |
| GenreID (FK)      |        -------------------
| ReleaseDate       |
-------------------
          |
          |--------< LISTENING_HISTORY
                    -------------------
                    | HistoryID (PK)  |
                    | UserID (FK)     |
                    | SongID (FK)     |
                    | Timestamp       |
                    -------------------
          |
          V
GENRE
-------------------
| GenreID (PK)      |
| Name              |
| Description       |
-------------------
```

PHYSICAL ER DIAGRAM:



```
USER
------------------------------
| UserID (PK)          INT          |
| Name                 VARCHAR(100) NOT NULL |
| Email                VARCHAR(100) UNIQUE   |
| Password             VARCHAR(100) NOT NULL |
| JoinDate             DATETIME     |
------------------------------
         |
         |--------< PLAYLIST
         |
         |                  ------------------------------
         |                  | PlaylistID (PK)    INT      |
         |                  | UserID (FK)        INT      |
         |                  | Name               VARCHAR(100) |
         |                  | Description        TEXT     |
         |                  ------------------------------
         |
         |
         v
SONG
------------------------------
| SongID (PK)          INT          |
| Title                VARCHAR(150) NOT NULL |
| Artist               VARCHAR(100) |
| Album                VARCHAR(100) |
| GenreID (FK)         INT          |
| ReleaseDate          DATE         |
------------------------------
         |
         |--------< LISTENING_HISTORY
         |
         |                  ------------------------------
         |                  | HistoryID (PK)     INT      |
         |                  | UserID (FK)        INT      |
         |                  | SongID (FK)        INT      |
         |                  | Timestamp          DATETIME|
         |                  ------------------------------
         |
         |
         v
GENRE
------------------------------
| GenreID (PK)         INT          |
| Name                 VARCHAR(50) UNIQUE   |
| Description          TEXT         |
------------------------------
```

MYSQL STATEMENTS:

```
mysql
CREATE DATABASE MusicStreaming;

USE MusicStreaming;

CREATE TABLE Users (
  UserID INT AUTO_INCREMENT PRIMARY KEY,
  Username VARCHAR(50),
  Email VARCHAR(100)
);

CREATE TABLE Songs (
  SongID INT AUTO_INCREMENT PRIMARY KEY,
  SongTitle VARCHAR(100),
  Artist VARCHAR(50),
  GenreID INT,
  FOREIGN KEY (GenreID) REFERENCES Genres(GenreID)
);

CREATE TABLE Playlists (
  PlaylistID INT AUTO_INCREMENT PRIMARY KEY,
  UserID INT,
  PlaylistName VARCHAR(100),
  FOREIGN KEY (UserID) REFERENCES Users(UserID)
);
```

```
CREATE TABLE PlaylistSongs (
  PlaylistID INT,
  SongID INT,
  PRIMARY KEY (PlaylistID, SongID),
  FOREIGN KEY (PlaylistID) REFERENCES Playlists(PlaylistID),
  FOREIGN KEY (SongID) REFERENCES Songs(SongID)
);

CREATE TABLE ListeningHistory (
  HistoryID INT AUTO_INCREMENT PRIMARY KEY,
  UserID INT,
  SongID INT,
  ListenDate DATE,
  FOREIGN KEY (UserID) REFERENCES Users(UserID),
  FOREIGN KEY (SongID) REFERENCES Songs(SongID)
);

CREATE TABLE Genres (
  GenreID INT AUTO_INCREMENT PRIMARY KEY,
  GenreName VARCHAR(50)
);
```

Queries:

```
-- Recommend songs based on user listening patterns
SELECT
  Songs.SongTitle,
  Songs.Artist
```

```
FROM
  Songs
  JOIN ListeningHistory ON Songs.SongID = ListeningHistory.SongID
WHERE
  ListeningHistory.UserID = ?
  AND Songs.GenreID = (SELECT GenreID FROM Songs WHERE SongID =
(SELECT SongID FROM ListeningHistory WHERE UserID = ? ORDER BY
ListenDate DESC LIMIT 1));


-- Recommend songs based on genre preferences
SELECT
  Songs.SongTitle,
  Songs.Artist
FROM
  Songs
  JOIN Genres ON Songs.GenreID = Genres.GenreID
WHERE
  Genres.GenreName = ?;
```

Procedures:

```
DELIMITER //


CREATE PROCEDURE sp_UpdatePlaylistSuggestions(
  IN userID INT
)
BEGIN
  DECLARE finished INT DEFAULT 0;
  DECLARE songID INT;
  DECLARE playlistID INT;

  DECLARE curSongs CURSOR FOR
  SELECT SongID
  FROM ListeningHistory
  WHERE UserID = userID
  ORDER BY ListenDate DESC;

  DECLARE CONTINUE HANDLER FOR NOT FOUND SET finished = 1;

  OPEN curSongs;
    read_loop: LOOP
    FETCH curSongs INTO songID;
    IF finished = 1 THEN
      LEAVE read_loop;
    END IF;

    INSERT INTO PlaylistSongs (PlaylistID, SongID)
    VALUES ((SELECT PlaylistID FROM Playlists WHERE UserID = userID),
songID);
  END LOOP;

  CLOSE curSongs;
END //

DELIMITER;

DELIMITER //
```

```
CREATE PROCEDURE sp_IdentifyTrendingSongs(
  IN dateRange INT
)
BEGIN
  SELECT
    Songs.SongTitle,
    Songs.Artist,
    COUNT(*) AS ListenCount
  FROM
    Songs
    JOIN ListeningHistory ON Songs.SongID = ListeningHistory.SongID
  WHERE
    ListenDate >= DATE_SUB(CURDATE(), INTERVAL dateRange DAY)
  GROUP BY
    Songs.SongTitle, Songs.Artist
  ORDER BY
    ListenCount DESC;
END //
DELIMITER;

CREATE INDEX idx_listening_history_user_id ON ListeningHistory (UserID);
CREATE INDEX idx_listening_history_song_id ON ListeningHistory
(SongID);

CREATE INDEX idx_playlist_songs_playlist_id ON PlaylistSongs
(PlaylistID);
CREATE INDEX idx_playlist_songs_song_id ON PlaylistSongs (SongID);
```

```
PARTITION BY RANGE (YEAR(ListenDate)) (
  PARTITION p_2022 VALUES LESS THAN (2022),
  PARTITION p_2023 VALUES LESS THAN (2023),
  PARTITION p_2024 VALUES LESS THAN MAXVALUE
);
```

Conclusion:

This database schema supports a music streaming platform by storing users, songs, playlists, listening history, and genres. The queries recommend songs based on user listening patterns and genre preferences. The procedures update playlist suggestions and identify trending songs. Optimization strategies ensure quick access to frequently streamed songs and playlists.