

# MUSIC LIBRARY MANAGEMENT

A MINI-PROJECT BY:

Nithilan M                      230701216

Pranav Ram S                      230701234

Prasanna Kumar M                      230701237

*in partial fulfillment of the award of the degree*

*OF*

***BACHELOR OF ENGINEERING***

*IN*

COMPUTER SCIENCE AND ENGINEERING



RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2024

## BONAFIDE CERTIFICATE

Certified that this project “MUSIC LIBRARY MANAGEMENT” is the bonafide work of “NITHILAN M,PRANAV RAM S,PRASANNA KUMAR M” who carried out the project work under my supervision.

Submitted for the practical examination held on \_\_\_\_\_

SIGNATURE

SIGNATURE

Ms.ASWANA LAL

ASSISTANT PROFESSOR,  
Computer Science and Engineering,  
Rajalakshmi Engineering College  
(Autonomous),  
Thandalam,  
Chennai-602105

INTERNAL EXAMINER

EXTERNAL EXAMINER

## ABSTRACT

The Music Library Management System developed in Java with MySQL is an efficient and robust software solution for organizing, managing, and accessing music collections digitally.

This system offers an intuitive user interface and comprehensive functionalities to store, retrieve, and categorize music tracks seamlessly.

By integrating Java for the application logic and MySQL for the database, it ensures scalability, reliability, and performance.

The project simplifies the management of music libraries for individual users or organizations, enabling efficient search, playback, and updates to the database.

Its secure and user-friendly design provides a centralized platform to enhance the overall music management experience.

# TABLE OF CONTENTS

## 1. INTRODUCTION

- 1.1 INTRODUCTION
- 1.2 IMPLEMENTATION
- 1.3 SCOPE OF THE PROJECT
- 1.4 WEBSITE FEATURES

## 2. SYSTEM SPECIFICATION

- 2.1 HARDWARE SPECIFICATION
- 2.2 SOFTWARE SPECIFICATION

## 3. SAMPLE CODE

CONSISTS OF

LOGIN PAGE DESIGN  
DASHBOARD DESIGN  
LOGIN PAGE BACKEND  
REMOVE SONGS IN  
DASHBOARD  
UPDATE SONGS IN  
DASHBOARD

## 4. SNAPSHOTS

- 4.1 ADDITION OF SONGS
- 4.2 AFTER ADDITION
- 4.3 SELECTION OF SONG WHICH IS  
NEEDED TO BE DELETED
- 4.4 INTERFACE AFTER DELETION
- 4.5 INTERFACE IN MYSQL AFTER  
ADDITION AND DELETION OF SONG

## 5. CONCLUSION

## 6. REFERENCES

# INTRODUCTION

## INTRODUCTION

The Music Library Management System is a step forward in integrating technology into everyday tasks, making it simpler to organize, explore, and enjoy music libraries effectively.

### 1.1 IMPLEMENTATION

The MUSIC LIBRARY MANAGEMENT project discussed here is implemented using concepts of JAVA SWING AND MYSQL

### 1.2 SCOPE OF THE PROJECT

The Music Library Management System has wide applicability, serving both individual and organizational needs. This project lays the groundwork for an advanced music management tool that can be extended to support future developments in multimedia organization.

### 1.3 WEBSITE FEATURES

- Dashboard showing possible actions.
- Guide to choosing Songs .
- Option to Save Songs on website.

## SYSTEM SPECIFICATIONS

### 2.1 HARDWARE SPECIFICATIONS:

PROCESSOR	:	Intel i5
MEMORY SIZE	:	4GB(Minimum)
HARD DISK	:	500 GB of free space

### 2.2 SOFTWARE SPECIFICATIONS:

PROGRAMMING LANGUAGE :	Java, MySQL
FRONT-END	: Java
BACK-END	: MySQL
OPERATING SYSTEM	: Windows 10

## SAMPLE CODE

```
import javax.swing.*.*;

import javax.swing.table.DefaultTableModel;

import java.awt.*.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.*.*;

public class MusicLibrary extends JFrame {

    private JTextField titleField, artistField, albumField, genreField;

    private JTable songTable;

    private DefaultTableModel tableModel;

    private Connection connection;

    public MusicLibrary() {

        connectToDatabase();

        setTitle("Music Library Management System");

        setSize(600, 400);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setLocationRelativeTo(null);

        JPanel inputPanel = new JPanel(new GridLayout(5, 2));

        inputPanel.add(new JLabel("Title:"));

        titleField = new JTextField();

        inputPanel.add(titleField);

        inputPanel.add(new JLabel("Artist:"));

        artistField = new JTextField();

        inputPanel.add(artistField);

        inputPanel.add(new JLabel("Album:"));

        albumField = new JTextField();

        inputPanel.add(albumField);

        inputPanel.add(new JLabel("Genre:"));

        genreField = new JTextField();
```

```

inputPanel.add(genreField);

JButton addButton = new JButton("Add Song");
JButton deleteButton = new JButton("Delete Song");
inputPanel.add(addButton);
inputPanel.add(deleteButton);

tableModel = new DefaultTableModel(new String[]{"ID", "Title", "Artist", "Album", "Genre"}, 0);
songTable = new JTable(tableModel);

loadSongs();

add(new JScrollPane(songTable), BorderLayout.CENTER);
add(inputPanel, BorderLayout.SOUTH);

addButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        addSong();
    }
});

deleteButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        deleteSong();
    }
});
}

private void connectToDatabase() {
    try {
        connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/musiclibrarydb",
"root", "2006");

        System.out.println("Database connected successfully.");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```



```

    }
}

private void loadSongs() {
    try {
        Statement statement = connection.createStatement();
        ResultSet rs = statement.executeQuery("SELECT * FROM Songs");

        tableModel.setRowCount(0); // Clear existing rows
        while (rs.next()) {
            int id = rs.getInt("id");
            String title = rs.getString("title");
            String artist = rs.getString("artist");
            String album = rs.getString("album");
            String genre = rs.getString("genre");
            tableModel.addRow(new Object[]{id, title, artist, album, genre});
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

private void addSong() {
    String title = titleField.getText();
    String artist = artistField.getText();
    String album = albumField.getText();
    String genre = genreField.getText();

    try {
        PreparedStatement ps = connection.prepareStatement("INSERT INTO Songs (title, artist, album, genre) VALUES (?, ?, ?, ?)");

        ps.setString(1, title);
        ps.setString(2, artist);

```

```

        ps.setString(3, album);

        ps.setString(4, genre);

        ps.executeUpdate();

        loadSongs(); // Refresh table

        clearFields();

    } catch (SQLException e) {
        e.printStackTrace();
    }
}

private void deleteSong() {
    int selectedRow = songTable.getSelectedRow();

    if (selectedRow == -1) {
        JOptionPane.showMessageDialog(this, "Please select a song to delete.");

        return;
    }

    int songId = (int) tableModel.getValueAt(selectedRow, 0);

    try {
        PreparedStatement ps = connection.prepareStatement("DELETE FROM Songs WHERE id = ?");

        ps.setInt(1, songId);

        ps.executeUpdate();

        loadSongs(); // Refresh table

    } catch (SQLException e) {
        e.printStackTrace();
    }
}

private void clearFields() {
    titleField.setText("");

    artistField.setText("");

```

```
        albumField.setText("");
        genreField.setText("");
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            new MusicLibrary().setVisible(true);
        });
    }
}
```

## DATABASE QUERY

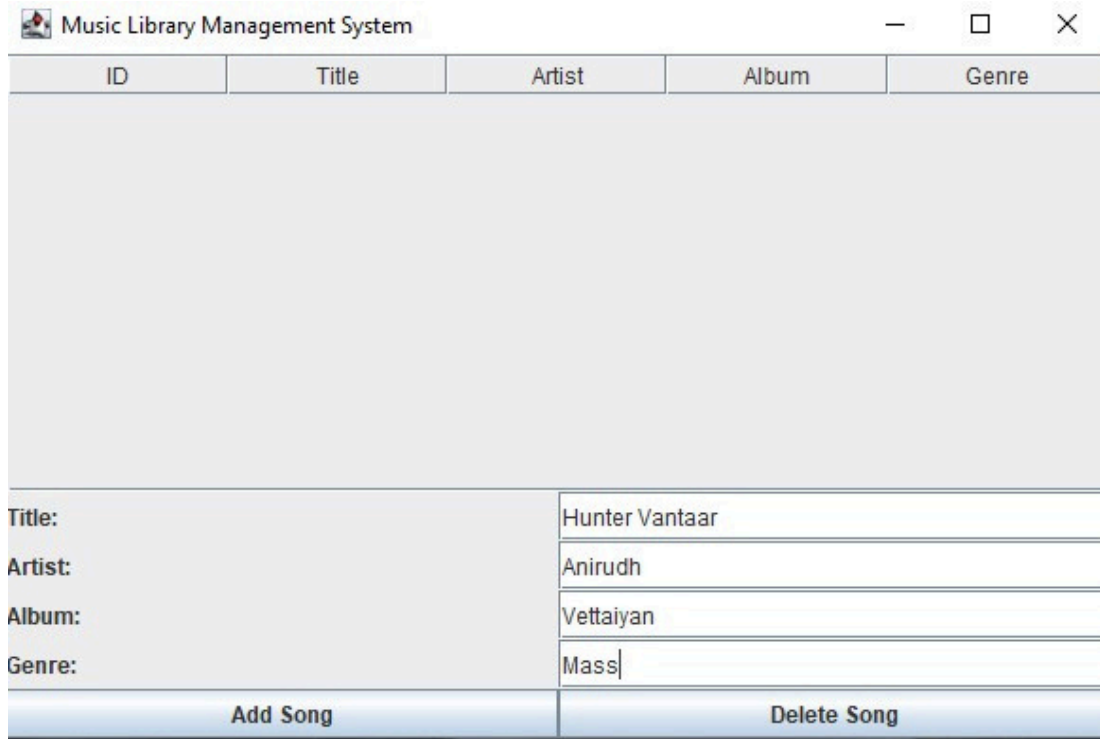
```
CREATE DATABASE musiclibrarydb;
```

```
USE musiclibrarydb;
```

```
CREATE TABLE Songs (  
    id INT PRIMARY KEY AUTO_INCREMENT,  
    title VARCHAR(255) NOT NULL,  
    artist VARCHAR(255),  
    album VARCHAR(255),  
    genre VARCHAR(100)  
);
```

## SNAPSHOTS

### 4.1 ADDITION OF SONGS



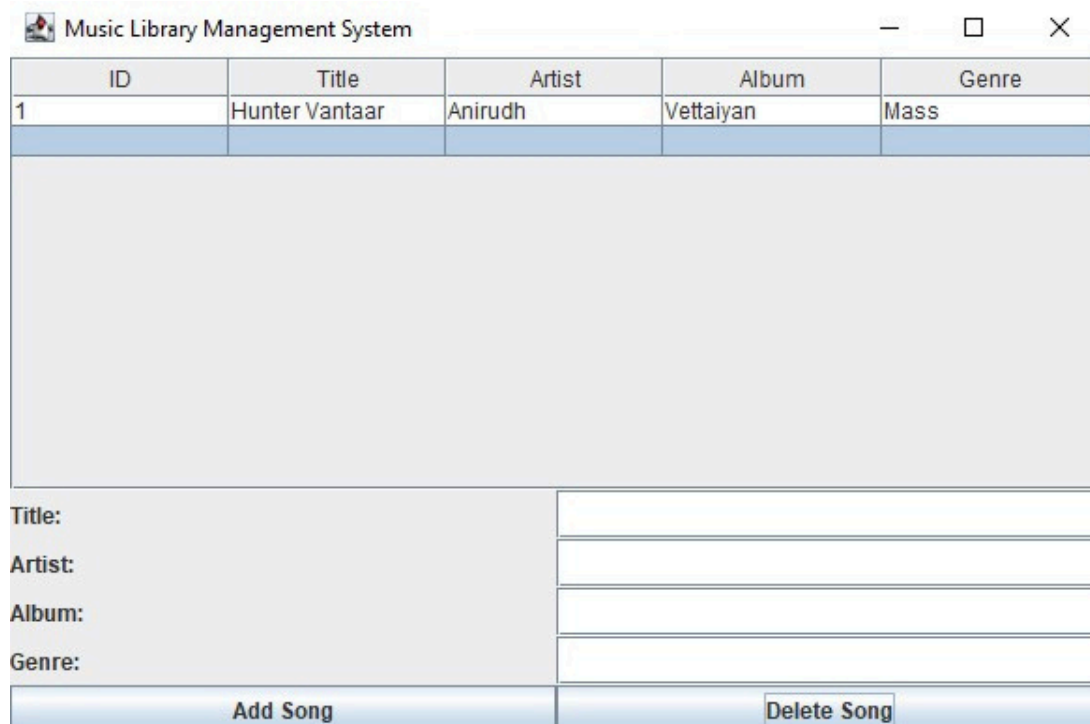
The screenshot shows a window titled "Music Library Management System" with standard window controls (minimize, maximize, close). Below the title bar is a table with five columns: ID, Title, Artist, Album, and Genre. The table is currently empty. Below the table is a form with four input fields labeled "Title:", "Artist:", "Album:", and "Genre:". The "Title:" field contains the text "Hunter Vantaar", "Artist:" contains "Anirudh", "Album:" contains "Vettaiyan", and "Genre:" contains "Mass". At the bottom of the form are two buttons: "Add Song" and "Delete Song".

ID	Title	Artist	Album	Genre
----	-------	--------	-------	-------

Title: Hunter Vantaar  
Artist: Anirudh  
Album: Vettaiyan  
Genre: Mass

Add Song Delete Song

### 4.2 AFTER ADDITION



The screenshot shows the same window as in 4.1, but now the table contains one row with the following data: ID: 1, Title: Hunter Vantaar, Artist: Anirudh, Album: Vettaiyan, Genre: Mass. The form below the table is empty, with the "Title:" field containing the text "Hunter Vantaar", "Artist:" containing "Anirudh", "Album:" containing "Vettaiyan", and "Genre:" containing "Mass". The "Add Song" and "Delete Song" buttons are still present at the bottom.

ID	Title	Artist	Album	Genre
1	Hunter Vantaar	Anirudh	Vettaiyan	Mass

Title: Hunter Vantaar  
Artist: Anirudh  
Album: Vettaiyan  
Genre: Mass


Add Song Delete Song

#### 4.3 SELECTION OF SONG WHICH IS NEEDED TO BE DELETED

Music Library Management System

ID	Title	Artist	Album	Genre
1	Hunter Vantaar	Anirudh	Vettaiyan	Mass

**Message** [X]

 Please select a song to delete.

**OK**

Title:

Artist:

Album:

Genre:

**Add Song** **Delete Song**

Music Library Management System

ID	Title	Artist	Album	Genre
1	Hunter Vantaar	Anirudh	Vettaiyan	Mass

Title:

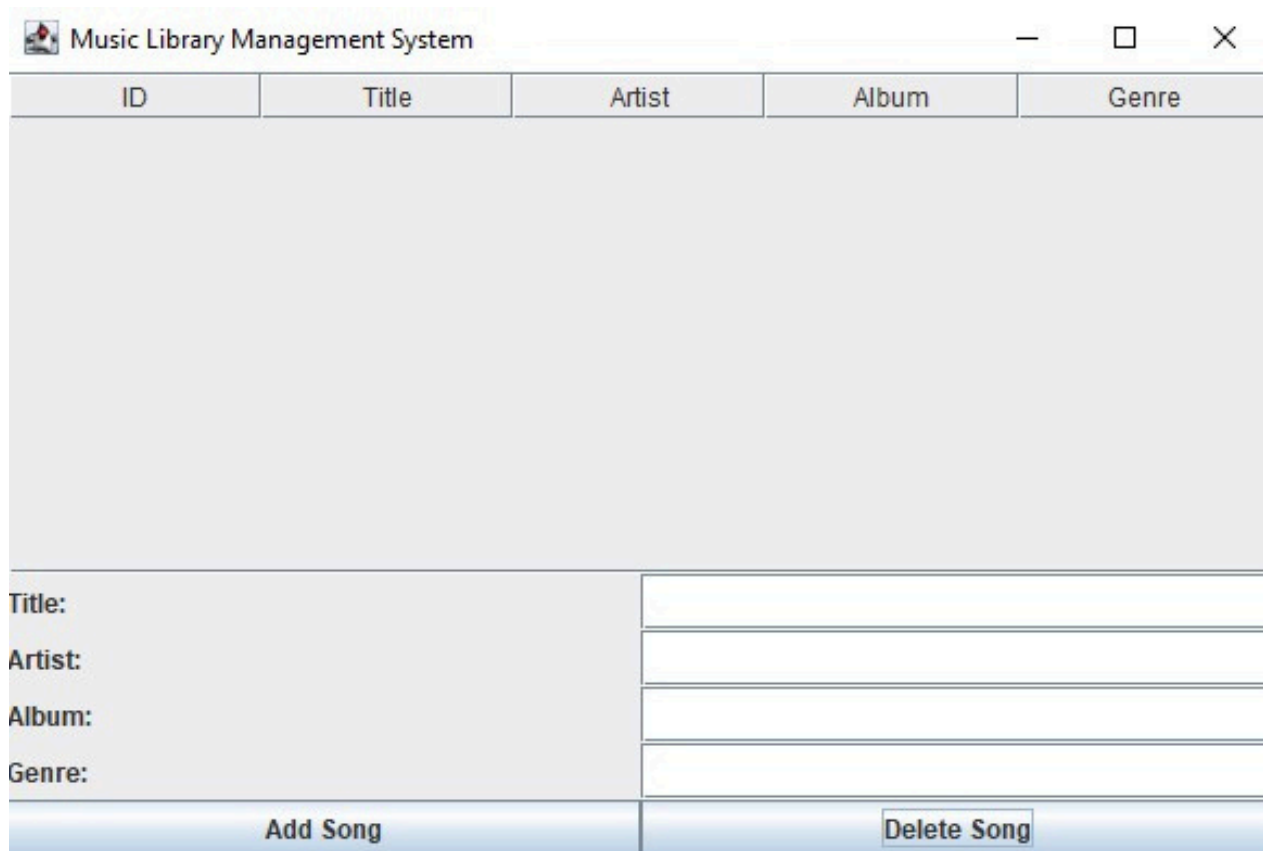
Artist:

Album:

Genre:

**Add Song** **Delete Song**

## 4.4 INTERFACE AFTER DELETION



The screenshot shows a window titled "Music Library Management System" with standard window controls (minimize, maximize, close). The main area contains a table with five columns: ID, Title, Artist, Album, and Genre. The table body is empty, indicating that all songs have been deleted. Below the table, there are four input fields labeled "Title:", "Artist:", "Album:", and "Genre:". At the bottom, there are two buttons: "Add Song" and "Delete Song".

ID	Title	Artist	Album	Genre
----	-------	--------	-------	-------

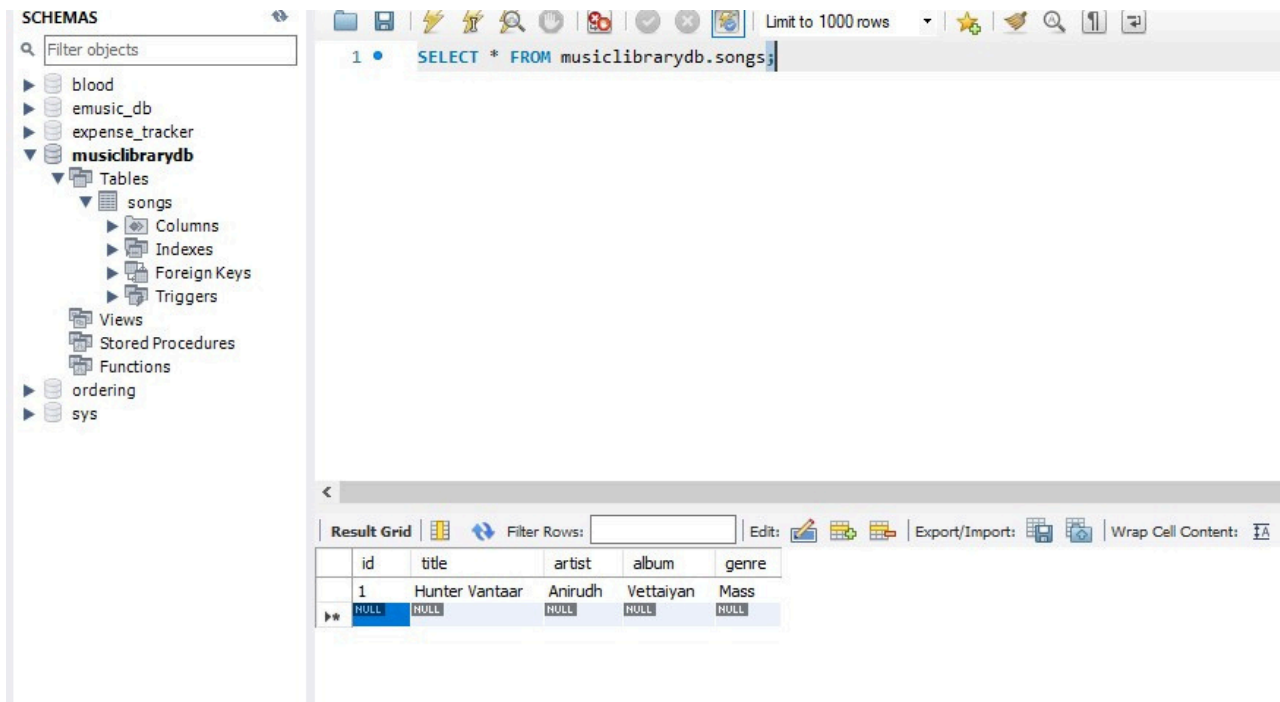
Title:

Artist:

Album:

Genre:

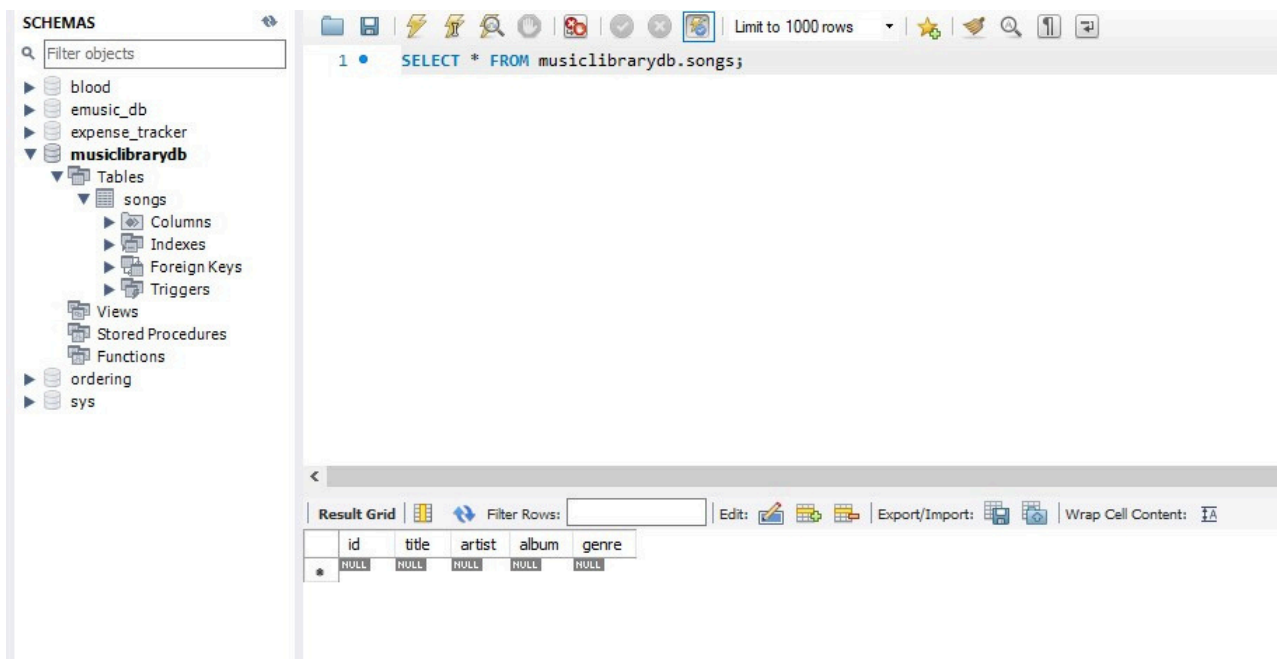
## 4.5 A) INTERFACE IN MY SQL AFTER ADDITION



The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays a tree view of databases, with 'musiclibrarydb' expanded to show its tables. The 'songs' table is selected. The main editor window shows the SQL query: `SELECT * FROM musiclibrarydb.songs;`. Below the query, the 'Result Grid' displays the data. The first row contains the song details, and the second row contains NULL values.

	id	title	artist	album	genre
1	1	Hunter Vantaar	Anirudh	Vettaiyan	Mass
»	NULL	NULL	NULL	NULL	NULL

## 4.5 B) INTERFACE IN MY SQL AFTER DELETION



The screenshot shows the MySQL Workbench interface after a deletion. The 'songs' table is still selected in the 'SCHEMAS' pane. The SQL query remains `SELECT * FROM musiclibrarydb.songs;`. The 'Result Grid' now shows only one row with NULL values, indicating that the data row has been removed.

	id	title	artist	album	genre
*	NULL	NULL	NULL	NULL	NULL



## CONCLUSION

The Music Library Management System is a comprehensive solution for organizing and managing digital music collections efficiently.

By leveraging Java's robust programming capabilities and MySQL's reliable database management, the system ensures a seamless user experience for personal and professional applications.

In conclusion, this project successfully demonstrates how technology can be utilized to modernize and streamline the management of music libraries, making it accessible and efficient for users across various domains.

## REFERENCES

1. <https://www.javatpoint.com/java-tutorial>
2. <https://www.w3schools.com/sql/>
3. <https://chatgpt.com/>
4. SQL | Codecademy