MUSIC STORE

PROJECT BY - RASIKA SANJAY JADHAV

GUIDED BY- MR. SAMEER WARSOLKAR SIR

ABSTRACT:

This project involves the creation of a comprehensive SQL database for a music store. The database includes various interconnected tables such as employee, customer, invoice, invoice line, playlist, playlist track, track, artist, album, and genre. The goal is to manage and query data related to employees, customers, sales, music tracks, playlists, and artists efficiently.

It is intended to capture information about employees involved in the process, customers who purchase music, artists whose works are being distributed, playlists that include specific tracks, albums containing multiple tracks, individual tracks, the genre of each track, the relationship between tracks and playlists, invoices generated for sales, and line items within those invoices.

The database should ensure data integrity, scalability, and security while providing an efficient way to query and update information.

RASIKA SANJAY JADHAV

Rasiakajadhav1211@gmail.com

Batch ID: KL302/DS/04-06 pm

MUSIC STORE

Project for SQL Module

OBJECTIVES:

The primary objective of this project is to design and implement a relational database for a music store that can handle various aspects such as

- Employee Information: Details about store staff.
- Customer Profiles: Data on customers, including purchase history and preferences.
- Artist Information: Details about musicians and bands.
- Playlist Data: Information about curated song collections.
- Album and Track Details: Metadata for albums and individual songs.
- Genre Classification: Categorization of music by genre.

FUNCTIONALITY:

1. Invoice line: Stores itemized details for sales transactions.
2. Invoice: Represents a complete sales transaction.
3. Playlist track: Defines the relationship between playlists and tracks.
4. Track: Represents a single musical composition.
5. Album: Represents a collection of tracks.
6. Playlist: Represents a curated collection of tracks.
7. Genre: Categorizes music by style.
8. Artist: Stores information about musicians.
9. Customer: Stores information about customers.
10. Employee: Stores information about store employees.

Database name – MUSIC STORE

Here are the table names corresponding to the schemas:

- Invoice line
- Invoice
- Playlist track
- Track
- Album
- Playlist
- Genre
- Artist
- Customer
- Employee

How these tables/entities are related to each other is shown pictorially on next page through

ER diagram, i.e., Entity Relationship Diagram

ER-DIAGRAM (ENTITY RELATION – DIAGRAM) FOR MUSIC STORE:

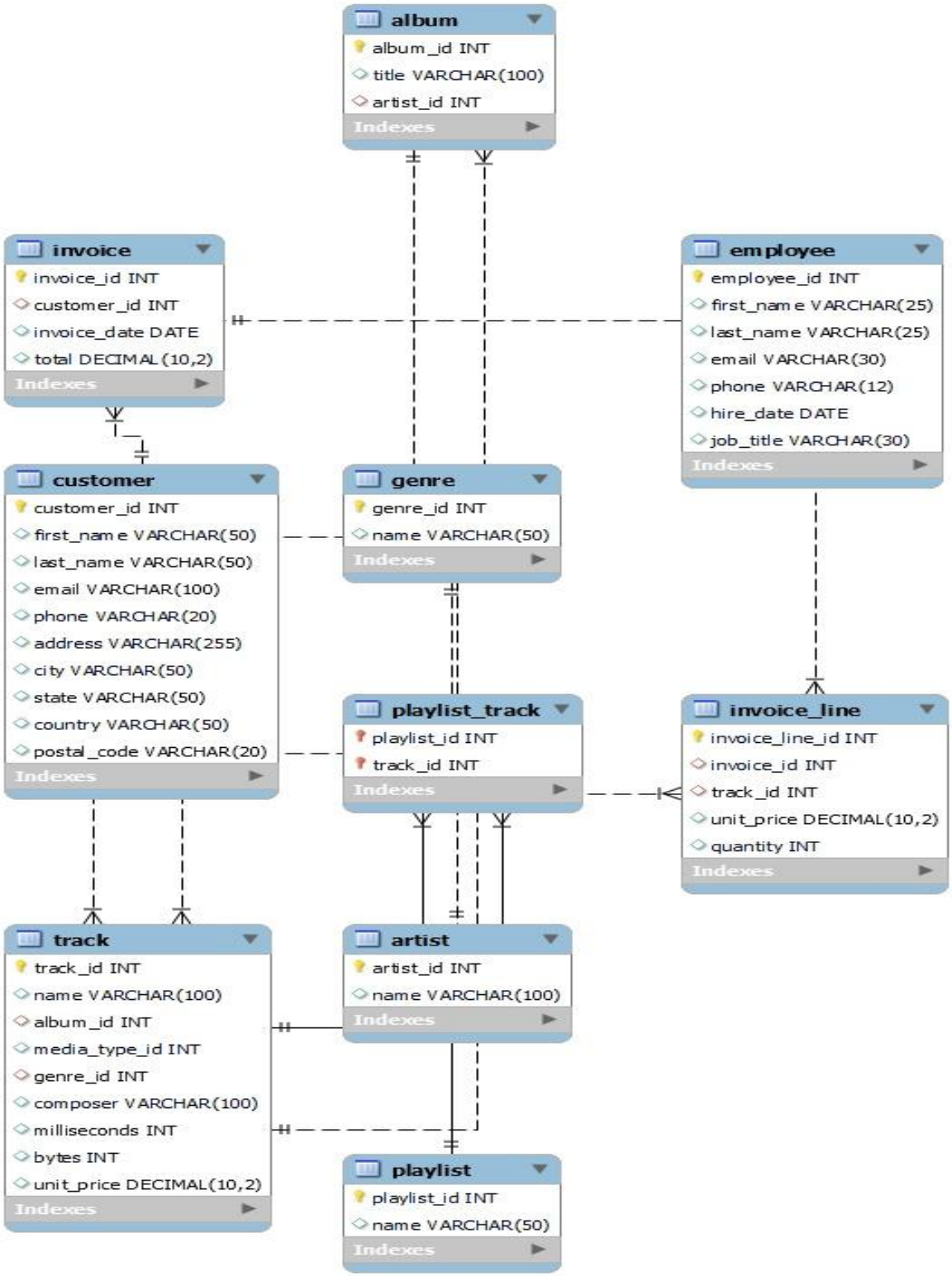

TABLE DESCRIPTIONS:

1.Employee:

Field	Type	Null	Key	Default	Extra
employee_id	int	NO	PRI	NULL	auto_increment
first_name	varchar(25)	YES		NULL	
last_name	varchar(25)	YES		NULL	
email	varchar(30)	YES		NULL	
phone	varchar(12)	YES		NULL	
hire_date	date	YES		NULL	
job_title	varchar(30)	YES		NULL	

2.customer

Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	auto_increment
first_name	varchar(20)	YES		NULL	
last_name	varchar(25)	YES		NULL	
email	varchar(20)	YES		NULL	
phone	varchar(12)	YES		NULL	
address	varchar(60)	YES		NULL	
city	varchar(15)	YES		NULL	
state	varchar(20)	YES		NULL	
country	varchar(30)	YES		NULL	
postal_code	varchar(30)	YES		NULL	

3.Artist

Field	Type	Null	Key	Default	Extra
artist_id	int	NO	PRI	NULL	
name	varchar(30)	YES		NULL	

4.Playlist

Field	Type	Null	Key	Default	Extra
playlist_id	int	NO	PRI	NULL	auto_increment
name	varchar(30)	YES		NULL	

5.Genre

Field	Type	Null	Key	Default	Extra
genre_id	int	NO	PRI	NULL	
name	varchar(60)	YES		NULL	

6.Album

Field	Type	Null	Key	Default	Extra
album_id	int	NO	PRI	NULL	
title	varchar(90)	YES		NULL	
artist_id	int	YES	MUL	NULL	

7.Track

Field	Type	Null	Key	Default	Extra
track_id	int	NO	PRI	NULL	
name	varchar(90)	YES		NULL	
album_id	int	YES	MUL	NULL	
media_type_id	int	YES		NULL	
genre_id	int	YES	MUL	NULL	
composer	varchar(90)	YES		NULL	
milliseconds	int	YES		NULL	
bytes	int	YES		NULL	
unit_price	decimal(10,2)	YES		NULL	

8.Playlist Track

Field	Type	Null	Key	Default	Extra
playlist_id	int	NO	PRI	NULL	
track_id	int	NO	PRI	NULL	

9.Invoice

Field	Type	Null	Key	Default	Extra
invoice_id	int	NO	PRI	NULL	auto_increment
customer_id	int	YES	MUL	NULL	
invoice_date	date	YES		NULL	
total	decimal(10,2)	YES		NULL	

10.Invoice line

Field	Type	Null	Key	Default	Extra
invoice_line_id	int	NO	PRI	NULL	auto_increment
invoice_id	int	YES	MUL	NULL	
track_id	int	YES	MUL	NULL	
unit_price	decimal(10,2)	YES		NULL	
quantity	int	YES		NULL	

COMMANDS:

>CREATE SCHEMA MUSIC;

> CREATE DATABASE MUSIC STORE;

>USE MUSIC STORE;

--TO CREATE TABLE EMPLOYEE

```
>CREATE TABLE EMPLOYEE
  CREATE TABLE employee (
    employee id INT PRIMARY KEY AUTO INCREMENT,
    first name VARCHAR (25),
    last name VARCHAR (25),
    email VARCHAR (30),
    phone VARCHAR (12),
    hire date DATE,
    job title varchar (30)
);
```

--TO CREATE TABLE CUSTOMER

```
CREATE TABLE customer (
    customer id INT PRIMARY KEY AUTO INCREMENT,
    first name VARCHAR (20),
    last name VARCHAR (25),
    email VARCHAR (20),
    phone VARCHAR (12),
    address VARCHAR 60),
    city VARCHAR (15),
```

```
    state VARCHAR (20),
    country VARCHAR (30),
    postal code VARCHAR (30)
);
```

--TO CREATE TABLE ARTIST

```
CREATE TABLE artist (
    artist id INT PRIMARY KEY,
    name VARCHAR (30)
);
```

--TO CREATE TABLE GENRE

```
CREATE TABLE genre (
    genre id INT PRIMARY KEY,
    name VARCHAR (60)
);
```

--TO CREATE TABLE PLAYLIST

```
CREATE TABLE playlist (
    playlist id INT PRIMARY KEY   AUTO INCREMENT,
    name VARCHAR (30)
);
```

--TO CREATE TABLE ALBUM

```
CREATE TABLE album (
    album id INT PRIMARY KEY,
    title VARCHAR (90),
    artist id INT,
    FOREIGN KEY (artist id) REFERENCES artist (artist id)
);
```

--TO CREATE TABLE TRACK

```
CREATE TABLE track (
   track id INT PRIMARY KEY,
   name VARCHAR (90),
   album id INT,
   media type id INT,
   genre id INT,
   composer VARCHAR (90),
   milliseconds INT,
   bytes INT,
   unit price DECIMAL (10, 2),
   FOREIGN KEY (album id) REFERENCES album (album id),
   FOREIGN KEY (genre id) REFERENCES genre (genre id)
);
```

--TO CREATE TABLE PLAYLIST TRACK

```
CREATE TABLE playlist track (
   playlist id INT,
   track id INT,
   PRIMARY KEY (playlist id, track id),
   FOREIGN KEY (playlist id) REFERENCES playlist (playlist id),
   FOREIGN KEY (track id) REFERENCES track (track id)
);
```

--TO CREATE TABLE INVOICE

```
CREATE TABLE invoice (
   invoice id INT PRIMARY KEY AUTO INCREMENT,
   customer id INT,
   invoice date DATE,
   total DECIMAL (10, 2),
   FOREIGN KEY (customer id) REFERENCES customer (customer
id)
```

```
);
```

--TO CREATE TABLE INVOICE LINE

```
CREATE TABLE invoice line (
   invoice line id INT PRIMARY KEY AUTO INCREMENT,
   invoice id INT,
   track id INT,
   unit price DECIMAL (10, 2),
   quantity INT,
   FOREIGN KEY (invoice id) REFERENCES invoice (invoice id),
   FOREIGN KEY (track id) REFERENCES track (track id)
);
```

--TO INSERT VALUES INTO EMPLOYEE

```
INSERT INTO employee (first name, last name, email, phone, hire
date, job title) VALUES

('Arti', 'Sharma', 'arti.sharma@example.com', '9876543210',
'2022-01-15', 'Manager'),
('Priya', 'Singh', 'priya.singh@example.com', '9876543211',
'2021-02-20', 'Sales Associate'),
('Mehul', 'Verma', 'mehull.verma@example.com', '9876543212',
'2020-03-25', 'Developer'),
('Sneha', 'Patel', 'sneha.patel@example.com', '9876543213',
'2019-04-30', 'HR'),
('Vikram', 'Rao', 'vikram.rao@example.com', '9876543214',
'2018-05-10', 'Accountant'),
('Anjali', 'Nair', 'anjali.nair@example.com', '9876543215', '2017-
06-15', 'Marketing Manager'),
('Karan', 'Kumar', 'karan.kumar@example.com', '9876543216',
'2016-07-20', 'Sales Manager'),
('Kavita', 'Joshi', 'kavita.joshi@example.com', '9876543217',
'2015-08-25', 'Developer'),
```

('Suresh', 'Gupta', 'suresh.gupta@example.com', '9876543218', '2014-09-30', 'Support Engineer'),
('Meena', 'Desai', 'meena.desai@example.com', '9876543219', '2013-10-05', 'HR'),
('Yash', 'Mehta', 'yash.mehta@example.com', '9876543220', '2012-11-10', 'Manager'),
('Pooja', 'Reddy', 'pooja.reddy@example.com', '9876543221', '2011-12-15', 'Sales Associate'),
('Nikhil', 'Chopra', 'nikhil.chopra@example.com', '9876543222', '2010-01-20', 'Developer'),
('Monika', 'Shah', 'monika.shah@example.com', '9876543223', '2009-02-25', 'HR'),
('Manish', 'Bose', 'manish.bose@example.com', '9876543224', '2008-03-30', 'Accountant'),
('Rasika', 'Kapoor', 'rasika.kapoor@example.com', '9876543225', '2007-04-05', 'Marketing Manager'),
('Sanjay', 'Mishra', 'sanjay.mishra@example.com', '9876543226', '2006-05-10', 'Sales Manager'),
('Divya', 'Iyer', 'divya.iyer@example.com', '9876543227', '2005-06-15', 'Developer'),
('Rajesh', 'Agarwal', 'rajesh.agarwal@example.com', '9876543228', '2004-07-20', 'Support Engineer'),
('Swati', 'Bhatia', 'swati.bhatia@example.com', '9876543229', '2003-08-25', 'HR');

--TO INSERT VALUES INTO CUSTOMER

INSERT INTO customer (first name, last name, email, phone, address, city, state, country, postal code) VALUES
('Aarav', 'Sharma', 'aarav.sharma@ex.com', '9876543310', '123 MG Road', 'Mumbai', 'Maharashtra', 'India', '400001'),
('Ananya', 'Verma', 'ananya.verma@ex.com', '9876543311', '456 Park Street', 'Delhi', 'Delhi', 'India', '110001'),

('Rohan', 'Patel', 'rohan.patel@ex.com', '9876543312', '789 Lake View', 'Bangalore', 'Karnataka', 'India', '560001'),
('Isha', 'Reddy', 'isha.reddy@ex.com', '9876543313', '101 Green Avenue', 'Hyderabad', 'Telangana', 'India', '500001'),
('Karan', 'Gupta', 'karan.gupta@ex.com', '9876543314', '202 Blue Street', 'Chennai', 'Tamil Nadu', 'India', '600001'),
('Sneha', 'Nair', 'sneha.nair@ex.com', '9876543315', '303 Red Road', 'Kochi', 'Kerala', 'India', '682001'),
('Vikram', 'Singh', 'vikram.singh@ex.com', '9876543316', '404 Yellow Lane', 'Jaipur', 'Rajasthan', 'India', '302001'),
('Pooja', 'Mehta', 'pooja.mehta@ex.com', '9876543317', '505 White Street', 'Ahmedabad', 'Gujarat', 'India', '380001'),
('Arjun', 'Bose', 'arjun.bose@ex.com', '9876543318', '606 Black Road', 'Kolkata', 'West Bengal', 'India', '700001'),
('Neha', 'Kapoor', 'neha.kapoor@ex.com', '9876543319', '707 Orange Avenue', 'Pune', 'Maharashtra', 'India', '411001'),
('Suresh', 'Chopra', 'suresh.chopra@ex.com', '9876543320', '808 Pink Street', 'Lucknow', 'Uttar Pradesh', 'India', '226001'),
('Divya', 'Agarwal', 'divya.agarwal@ex.com', '9876543321', '909 Purple Lane', 'Bhopal', 'Madhya Pradesh', 'India', '462001'),
('Rajesh', 'Joshi', 'rajesh.joshi@ex.com', '9876543322', '1010 Brown Road', 'Indore', 'Madhya Pradesh', 'India', '452001'),
('Meena', 'Desai', 'meena.desai@ex.com', '9876543323', '1111 Grey Avenue', 'Surat', 'Gujarat', 'India', '395001'),
('Amit', 'Kumar', 'amit.kumar@ex.com', '9876543324', '1212 Silver Street', 'Patna', 'Bihar', 'India', '800001'),
('Priya', 'Rao', 'priya.rao@ex.com', '9876543325', '1313 Gold Lane', 'Nagpur', 'Maharashtra', 'India', '440001'),
('Nikhil', 'Shah', 'nikhil.shah@ex.com', '9876543326', '1414 Bronze Road', 'Chandigarh', 'Chandigarh', 'India', '160001'),
('Ritu', 'Iyer', 'ritu.iyer@ex.com', '9876543327', '1515 Copper Avenue', 'Thirupuram', 'Kerala', 'India', '695001'),

('Manish', 'Bhatia', 'manish.bhatia@ex.com', '9876543328', '1616 Platinum Street', 'Vadodara', 'Gujarat', 'India', '390001'), ('Swati', 'Mishra', 'swati.mishra@ex.com', '9876543329', '1717 Diamond Lane', 'Kanpur', 'Uttar Pradesh', 'India','208001');

--TO INSERT VALUES INTO ARTIST

INSERT INTO artist (artist id, name) VALUES
(1, 'Arijit Singh'),
(2, 'Shreya Ghoshal'),
(3, 'Sonu Nigam'),
(4, 'Lata Mangeshkar'),
(5, 'Kishore Kumar'),
(6, 'Asha Bhosle'),
(7, 'Mohammed Rafi'),
(8, 'Udit Narayan'),
(9, 'Alka Yagnik'),
(10, 'Kumar Sanu'),
(11, 'Sunidhi Chauhan'),
(12, 'Neha Kakkar'),
(13, 'Armaan Malik'),
(14, 'Badshah'),
(15, 'Guru Randhawa'),
(16, 'Mika Singh'),
(17, 'Shankar Mahadevan'),
(18, 'Shaan'),
(19, 'Ankit Tiwari'),
(20, 'Jubin Nautiyal');

--TO INSERT VALUES INTO GENRE

INSERT INTO genre (genre id, name) VALUES
(1, 'Romantic'),
(2, 'Action'),

(3, 'Drama'),
(4, 'Comedy'),
(5, 'Thriller'),
(6, 'Horror'),
(7, 'Musical'),
(8, 'Adventure'),
(9, 'Fantasy'),
(10, 'Mystery'),
(11, 'Sci-Fi'),
(12, 'Family'),
(13, 'Biography'),
(14, 'Historical'),
(15, 'Crime'),
(16, 'War'),
(17, 'Sports'),
(18, 'Animation'),
(19, 'Documentary'),
(20, 'Western');

--TO INSERT VALUES INTO PLAYLIST

INSERT INTO playlist (playlist id, name) VALUES
(1, 'Romantic Hits'),
(2, 'Party Anthems'),
(3, 'Golden Era Classics'),
(4, 'Soulful Sufi Songs'),
(5, 'Dance Floor Grooves'),
(6, 'Love Ballads'),
(7, 'Retro Rewind'),
(8, 'Blockbuster Soundtracks'),
(9, 'Heartbreak Hits'),
(10, 'Rainy Day Melodies'),
(11, 'Item Song Extravaganza'),
(12, 'Wedding Playlist'),

(13, 'Road Trip Beats'),
(14, 'Feel-Good Bollywood'),
(15, 'Unplugged Gems'),
(16, 'Foot-Tapping Numbers'),
(17, 'Evergreen Duets'),
(18, 'Sentimental Favourites'),
(19, 'Celebratory Songs'),
(20,'Bolywood Remixes');

--TO INSERT VALUES INTO ALBUM

INSERT INTO album (album id, title, artist id) VALUES
(1, 'Aashiqui 2', 1),
(2,'Dil Se', 2),
(3, 'Kal Ho Naa Ho', 3),
(4, 'Rab Ne Bana Di Jodi', 4),
(5, 'Slumdog Millionaire', 5),
(6, 'Once Upon a Time in Mumbai', 6),
(7, 'Kabhi Khushi Kabhi Gham', 7),
(8, 'Dilwale Dulhania Le Jayenge', 8),
(9, 'Kuch Kuch Hota Hai', 9),
(10, 'Hum Aapke Hain Koun', 10),
(11, 'Dhoom 2', 11),
(12, 'Bajrangi Bhaijaan', 12),
(13, 'Ae Dil Hai Mushkil', 13),
(14, 'Badshah', 14),
(15, 'High Rated Gabru', 15),
(16, 'Mika Singh Hits', 16),
(17, 'Rock On!!', 17),
(18,'Tanha Dil', 18),
(19, 'Aashiqui 2', 19),
(20, 'Kabir Singh', 20);

--TO INSERT VALUES INTO TRACK

INSERT INTO track (track id, name, album id, media type id, genre id, composer, milliseconds, bytes, unit price) VALUES
(1, 'Tum Hi Ho', 1, 1, 1, 'Mithoon', 250000, 5000000, 15.00),
(2, 'Chaiya Chaiya', 2, 1, 2, 'A.R. Rahman', 300000, 6000000, 20.00),
(3, 'Kal Ho Naa Ho', 3, 1, 1, 'Shankar-Ehsaan-Loy', 280000, 5500000, 18.00),
(4, 'Tujh Mein Rab Dikhta Hai', 4, 1, 3, 'Salim-Sulaiman', 270000, 5200000, 17.00),
(5, 'Jai Ho', 5, 1, 2, 'A.R. Rahman', 320000, 6500000, 22.00),
(6, 'Pee Loon', 6, 1, 1, 'Pritam', 260000, 5100000, 16.00),
(7, 'Kabira', 7, 1, 3, 'Pritam', 290000, 5700000, 19.00),
(8, 'Gallan Goodiyan', 8, 1, 2, 'Shankar-Ehsaan-Loy', 310000, 6300000, 21.00),
(9, 'Tum Mile', 9, 1, 1, 'Pritam', 275000, 5400000, 18.50),
(10, 'Badtameez Dil', 10, 1, 2, 'Pritam', 295000, 5800000, 19.50),
(11, 'Tera Ban Jaunga', 11, 1, 1, 'Akhil Sachdeva', 265000, 5150000, 16.50),
(12,'Zaalima', 12, 1, 3, 'JAM8', 285000, 5550000, 18.50),
(13,'Dil Dhadakne Do', 13, 1, 2, 'Shankar-Ehsaan-Loy', 305000, 6200000, 20.50),
(14,'Raabta', 14, 1, 1, 'Pritam', 270000, 5250000, 17.50),
(15,'Ae Dil Hai Mushkil', 15, 1, 3, 'Pritam', 300000, 6000000, 20.00),
(16,'Ghungroo', 16, 1, 2, 'Vishal-Shekhar', 310000, 6300000, 21.00),
(17,'Bekhayali', 17, 1, 1, 'Sachet-Parampara', 280000, 5600000, 18.00),
(18, 'Malang', 18, 1, 3, 'Ved Sharma', 290000, 5800000, 19.00),
(19, 'Tera Yaar Hoon Main', 19, 1, 1, 'Rochak Kohli', 275000, 5500000, 18.50),

(20, 'Nashe Se Chad Gayi', 20,1,2, 'Vishal Shekhar', 295000, 590000, 19.50);

--TO INSERT VALUES INTO PLAYLIST TRACK

INSERT INTO playlist track (playlist id, track id) VALUES
(1, 1),
(1, 2),
(2, 3),
(2, 4),
(3, 5),
(3, 6),
(4, 7),
(4, 8),
(5, 9),
(5, 10),
(6, 11),
(6, 12),
(7, 13),
(7, 14),
(8, 15),
(8, 16),
(9, 17),
(9, 18),
(10, 10),
(10, 20);

--TO INSERT VALUES INTO INVOICE

INSERT INTO invoice (invoice date, total) VALUES
('2023-01-15', 1500.00),
('2023-01-20', 2000.00),
('2023-02-10', 2500.00),
('2023-02-15', 3000.00),
('2023-03-05', 3500.00),

('2023-03-10', 4000.00),
('2023-03-20', 4500.00),
('2023-04-01', 5000.00),
('2023-04-10', 5500.00),
('2023-04-20', 6000.00),
('2023-05-01', 6500.00),
('2023-05-10', 7000.00),
('2023-05-20', 7500.00),
('2023-06-01', 8000.00),
('2023-06-10', 8500.00),
('2023-06-20', 9000.00),
('2023-07-01', 9500.00),
('2023-07-10', 10000.00),
('2023-07-20', 10500.00),
('2023-08-01', 11000.00);

--TO INSERT VALUES INTO INVOICE LINE

INSERT INTO invoice line (invoice id, track id, unit price, quantity) VALUES

(1, 1, 15.00, 1),
(1, 2, 20.00, 2),
(2, 3, 25.00, 1),
(2, 4, 30.00, 3),
(3, 5, 35.00, 2),
(3, 6, 40.00, 1),
(4, 7, 45.00, 3),
(4, 8, 50.00, 2),
(5, 9, 55.00, 1),
(5, 10, 60.00, 3),

(6, 11, 65.00, 2),
(6, 12, 70.00, 1),
(7, 13, 75.00, 3),
(7, 14, 80.00, 2),
(8, 15, 85.00, 1),
(8, 16, 90.00, 3),
(9, 17, 95.00, 2),
(9, 18, 100.00, 1),
(10, 19, 105.00, 3),
(10, 20, 110.00, 2);

NORMAL QUERIES:

-- Find the Most Expensive Track

> SELECT name, unit_price FROM track ORDER BY unit_price DESC LIMIT 1;

```
-- normal Queris
 -- Find the Most Expensive Track
   SELECT name, unit_price FROM track ORDER BY unit_price DESC LIMIT 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

name	unit_price
Jai Ho	22.00

Output

Action Output

#	Time	Action	Message
1	22:12:16	SELECT name, unit_price FROM track ORDER BY unit_price DESC LIMIT 1	1 row(s) returned

-- List All Employees Hired After a Specific Date

>SELECT first_name, last_name, hire_date FROM employee WHERE hire_date > '2020-01-01';

```
SELECT first_name, last_name, hire_date FROM employee WHERE hire_date > '2020-01-01';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	hire_date
Arti	Sharma	2022-01-15
Priya	Singh	2021-02-20
Mehul	Verma	2020-03-25

employee 2

Output

Action Output

#	Time	Action	Message	Durati
1	22:32:05	SELECT first_name, last_name, hire_date FROM employee WHERE hire_date > '2020-01-01' LIMIT 0, 1000	3 row(s) returned	0.000

-- Calculate the Average Total of All Invoices

>SELECT AVG(total) AS average_invoice_total FROM Invoice;

```
335
336   -- Calculate the Average Total of All Invoices
337 •  SELECT   AVG(total) AS average_invoice_total FROM Invoice;
338
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

average_invoice_total
6250.000000

Result 3

Output

Action Output

#	Time	Action	Message
1	22:32:45	SELECT AVG(total) AS average_invoice_total FROM Invoice LIMIT 0, 1000	1 row(s) returned

-- Find Customers from a Specific City

> SELECT first_name, last_name, city, email FROM customer WHERE city ='MUMBAI';

```
338
339   -- Find Customers from a Specific City
340 •   SELECT first_name, last_name, city, email FROM customer WHERE city ='MUMBAI';
341
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	city	email
Aarav	Sharma	Mumbai	aarav.sharma@ex.com

customer 5

Output

Action Output

#	Time	Action	Message	Duration
1	22:33:43	SELECT first_name, last_name, city, email FROM customer WHERE city ='MUMBAI' LIMIT 0, 1000	1 row(s) returned	0.000 se

```
-- Count the Number of Invoices for Each Customer
> SELECT customer_id, COUNT(*) AS NumberofInvoices
FROM invoice
group by customer_id;
```

```
-- Count the Number of Invoices for Each Customer
SELECT customer_id, COUNT(*) AS NumberofInvoices
FROM invoice
group by customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	NumberofInvoices
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1

Result 9

Output

Action Output

#	Time	Action	Message
1	22:48:05	SELECT customer_id, COUNT(*) AS NumberofInvoices FROM invoice group by customer_id LIMIT 0, 1000	20 row(s) returned

SUB QUERIES:

```
-- Find Customers with Invoices Above Average Total+
  > SELECT first_name, last_name FROM customer
   WHERE customer_id IN (SELECT customer_id FROM invoice
   WHERE total > (SELECT AVG(total) FROM invoice));
```

```
-- subqueries
 -- Find Customers with Invoices Above Average Total+
  SELECT first_name, last_name FROM customer
  WHERE customer_id IN (SELECT customer_id FROM invoice
  WHERE total > (SELECT AVG(total) FROM invoice));
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name
Suresh	Chopra
Divya	Agarwal
Rajesh	Joshi
Meena	Desai
Amit	Kumar
Priya	Rao
Nikhil	Shah
Ritu	Iyer
Manish	Bhatia
Swati	Mishra

customer 10

Output

Action Output

#	Time	Action	Message
1	22:49:10	SELECT first_name, last_name FROM customer WHERE customer_id IN (SELECT customer_id FROM i...	10 row(s) returned

```
--  List Albums with Tracks Longer Than the Average Track Length
  > SELECT title
   FROM album
   WHERE album_id IN (
     SELECT album_id
     FROM track
     WHERE milliseconds > (SELECT AVG(milliseconds) FROM track)
     );
```

```sql
-- List Albums with Tracks Longer Than the Average Track Length
SELECT title
FROM album
WHERE album_id IN (
    SELECT album_id
    FROM track
    WHERE milliseconds > (SELECT AVG(milliseconds) FROM track)
    );
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

title
Dil Se
Slumdog Millionaire
Kabhi Khushi Kabhie Gham
Dilwale Dulhania Le Jayenge
Hum Aapke Hain Koun
Ae Dil Hai Mushkil
High Rated Gabru
Mika Singh Hits
Tanha Dil
Kabir Singh

album 12

Output

Action Output

#	Time	Action	Message
1	22:50:48	SELECT title FROM album WHERE album_id IN (SELECT album_id FROM track WHER...	10 row(s) returned

-- List Customers with No Invoices

>SELECT first_name, last_name FROM customer WHERE customer_id NOT IN (SELECT customer_id FROM invoice);

```
-- List Customers with No Invoices
SELECT first_name, last_name FROM customer WHERE customer_id NOT IN (SELECT customer_id FROM invoice);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name

customer 13

Output

Action Output

#	Time	Action	Message	Duration /
1	22:52:08	SELECT first_name, last_name FROM customer WHERE customer_id NOT IN (SELECT customer_id FROM...	0 row(s) returned	0.000 sec

-- Find Most Expensive Track in Each Genre

> SELECT name, genre_id, unit_price FROM track

WHERE unit_price = (SELECT MAX (unit_price) FROM track AS t

WHERE t.genre_id = track.genre_id);

```
-- Find Most Expensive Track in Each Genre
SELECT name, genre_id, unit_price FROM track
WHERE unit_price = (SELECT MAX(unit_price) FROM track AS t
 WHERE t.genre_id = track.genre_id);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	genre_id	unit_price
Jai Ho	2	22.00
Tum Mile	1	18.50
Ae Dil Hai Mushkil	3	20.00
Tera Yaar Hoon Main	1	18.50

track 14

Output

Action Output

#	Time	Action	Message
1	22:52:26	SELECT name, genre_id, unit_price FROM track WHERE unit_price = (SELECT MAX(unit_price) FROM trac...	4 row(s) returned

JOINS:

```
-- List All Tracks and Their Playlists
> SELECT t.name AS track_name, p.name AS playlist_name
FROM track t
RIGHT JOIN playlist_track pt ON t.track_id = pt.track_id
RIGHT JOIN playlist p ON pt.playlist_id = p.playlist_id;
```

```
-- -+joins
-- List All Tracks and Their Playlists
SELECT t.name AS track_name, p.name AS playlist_name
FROM track t
RIGHT JOIN playlist_track pt ON t.track_id = pt.track_id
RIGHT JOIN playlist p ON pt.playlist_id = p.playlist_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

track_name	playlist_name
Tum Hi Ho	Romantic Hits
Chaiyya Chaiyya	Romantic Hits
Kal Ho Naa Ho	Party Anthems
Tujh Mein Rab Dikhta Hai	Party Anthems
Jai Ho	Golden Era Classics
Pee Loon	Golden Era Classics
Kabira	Soulful Sufi Songs
Gallan Goodiyan	Soulful Sufi Songs
Tum Mile	Dance Floor Grooves
Badtameez Dil	Dance Floor Grooves
Tera Ban Jaunga	Love Ballads
Zaalima	Love Ballads
Dil Dhadakne Do	Retro Rewind
Raabta	Retro Rewind
Ae Dil Hai Mushkil	Blockbuster Soundtr...

Result 15

Output

Action Output

#	Time	Action	Message
1	22:52:54	SELECT t.name AS track_name, p.name AS playlist_name FROM track t RIGHT JOIN playlist_track pt ...	30 row(s) returned

-- List all tracks in the ‘Romantic’ genre

>SELECT t.name FROM track t

JOIN genre g ON t.genre_id = g.genre_id

WHERE g.name ='Romantic';

```
-- List all tracks in the ‘Romantic’ genre
SELECT t.name FROM track t
JOIN genre g ON t.genre_id = g.genre_id
WHERE g.name ='Romantic';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name
Tum Hi Ho
Kal Ho Naa Ho
Pee Loon
Tum Mile
Tera Ban Jaunga
Raabta
Bekhayali
Tera Yaar Hoon Main

Result 16

Output

Action Output

#	Time	Action	Message
1	22:53:27	SELECT t.name FROM track t JOIN genre g ON t.genre_id = g.genre_id WHERE g.name ='Romantic' LIMI...	8 row(s) returned

```
-- List all invoices with customer details
>SELECT i.invoice_id, c.first_name, c.last_name, i.total
FROM invoice i
JOIN customer c ON i.customer_id = c.customer_id;
```

```
394  -- List all invoices with customer details
395 • SELECT i.invoice_id, c.first_name, c.last_name, i.total
396   FROM invoice i
397   JOIN customer c ON i.customer_id = c.customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

invoice_id	first_name	last_name	total
1	Aarav	Sharma	1500.00
2	Ananya	Verma	2000.00
3	Rohan	Patel	2500.00
4	Isha	Reddy	3000.00
5	Karan	Gupta	3500.00
6	Sneha	Nair	4000.00
7	Vikram	Singh	4500.00
8	Pooja	Mehta	5000.00
9	Arjun	Bose	5500.00
10	Neha	Kapoor	6000.00
11	Suresh	Chopra	6500.00
12	Divya	Agarwal	7000.00
13	Rajesh	Joshi	7500.00
14	Meena	Desai	8000.00
15	Amit	Kumar	8500.00
16	Priya	Rao	9000.00
17	Nikhil	Shah	9500.00

Result 17

Output

Action Output

#	Time	Action	Message
1	22:53:53	SELECT i.invoice_id, c.first_name, c.last_name, i.total FROM invoice i JOIN customer c ON i.customer_id ...	20 row(s) returned

-- List All Customers Who Live in the Same state

>SELECT c1.customer_id, c1.first_name, c1.last_name, c1.state

FROM customer c1

JOIN customer c2 ON c1.state = c2.state AND c1.customer_id <> c2.customer_id

ORDER BY c1.state, c1.customer_id;

```
-- List All Customers Who Live in the Same state
SELECT c1.customer_id, c1.first_name, c1.last_name, c1.state
FROM customer c1
JOIN customer c2 ON c1.state = c2.state AND c1.customer_id <> c2.customer_id
ORDER BY c1.state, c1.customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	first_name	last_name	state
8	Pooja	Mehta	Gujarat
8	Pooja	Mehta	Gujarat
14	Meena	Desai	Gujarat
14	Meena	Desai	Gujarat
19	Manish	Bhatia	Gujarat
19	Manish	Bhatia	Gujarat
6	Sneha	Nair	Kerala
18	Ritu	Iyer	Kerala
12	Divya	Agarwal	Madhya Pradesh
13	Rajesh	Joshi	Madhya Pradesh
1	Aarav	Sharma	Maharashtra
1	Aarav	Sharma	Maharashtra
10	Neha	Kapoor	Maharashtra
10	Neha	Kapoor	Maharashtra
16	Priya	Rao	Maharashtra
16	Priya	Rao	Maharashtra
11	Suresh	Chopra	Uttar Pradesh
20	Swati	Mishra	Uttar Pradesh

Result 18

Output

Action Output

#	Time	Action	Message
1	22:54:45	SELECT c1.customer_id, c1.first_name, c1.last_name, c1.state FROM customer c1 JOIN customer c2 ON c...	18 row(s) returned

```
-- List All Employees and Their Invoices (Including Employees with No
>Invoices and Invoices with No Employees)

SELECT e.first_name, e.last_name, i.invoice_id, i.total

FROM employee e

join invoice i ON e.employee_id = i.customer_id;
```

Limit to 1000 rows

```
-- List All Employees and Their Invoices (Including Employees with No Invoices and Invoices with No Employees)
SELECT e.first_name, e.last_name, i.invoice_id, i.total
FROM employee e
 join invoice i ON e.employee_id = i.customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	invoice_id	total
Arti	Sharma	1	1500.00
Priya	Singh	2	2000.00
Mehul	Verma	3	2500.00
Sneha	Patel	4	3000.00
Vikram	Rao	5	3500.00
Anjali	Nair	6	4000.00
Karan	Kumar	7	4500.00
Kavita	Joshi	8	5000.00
Suresh	Gupta	9	5500.00
Meena	Desai	10	6000.00
Yash	Mehta	11	6500.00
Pooja	Reddy	12	7000.00
Nikhil	Chopra	13	7500.00
Monika	Shah	14	8000.00
Manish	Bose	15	8500.00
Rasika	Kapoor	16	9000.00
Sanjay	Mishra	17	9500.00
Divya	Iyer	18	10000.00
Rajesh	Agarwal	19	10500.00
Swati	Bhatia	20	11000.00

Result 21

Output

Action Output

#	Time	Action	Message

CONCLUSION:

This project successfully demonstrates the creation and management of a comprehensive music store database using SQL. By designing a well-structured schema and implementing various queries, we have shown how to efficiently handle and retrieve data related to employees, customers, sales, and music inventory. The database provides a solid foundation for managing a music store's operations and can be further enhanced with additional features and analytics for deeper insights.