SVKM's NMIMS MPSTME (Shirpur Campus)

Project

Course – B. Tech CS II Year

Group Members	Name	Roll no
	Vipul Mashruwala	B231
	Urvaang Naik	B236

Music Management Database System

Abstract

A database is the single most useful environment in which to store data and an ideal tool to manage and manipulate that data. The benefits of a well-structured database are infinite, with increased efficiency and time-saving benefits. Our team's interest is centred around this area. At the very start, we create a database on Music management system. We use MySQL for this purpose. We determine attributes and entities and figure out relationships among entities. Then we draw the entity-relationship diagram, convert it to a relational model (relational tables) and normalize the tables. We implement the design, create tables and insert values inside the tables using MySQL. We execute sample queries on the system and verify that our system contains all required information making retrieval of the information fast and efficient.

Database Design

The Entity-Relationship Model

The entity-relationship (E-R) model was developed to facilitate database design by allowing specification of an enterprise schema that represents the overall logical structure of a database.

Mapping Cardinality: Mapping cardinalities express the number of entities to which another entity can be associated via a relationship set. Cardinality can be-

- One-to-one
- One-to-many
- Many-to-one
- Many-to-many

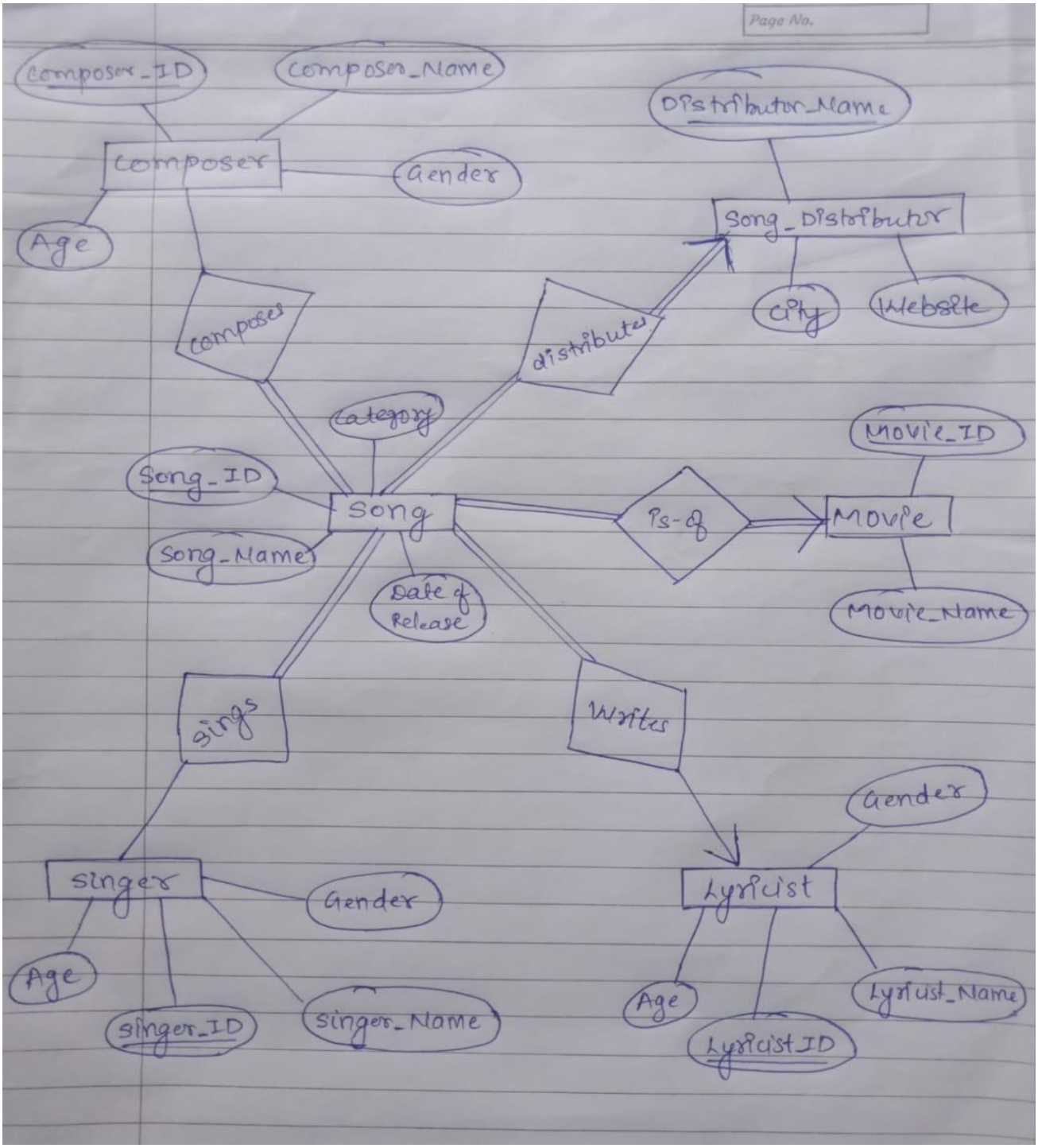
Participation Constraint: The participation constraint specifies the number of instances of an entity can participate in a relationship set. There are two types of participation constraints-

- Total participation
- Partial participation

E-R diagram: It can express the overall logical structure of a database graphically. A diagram consists of some major components—

Rectangles: represent entity set.
Ellipses: represent attributes.
Diamonds: represent relationships.
Lines: which link attributes to entity sets and entity sets to relationship sets.

ER-Diagram:



Our E-R diagram represents the Music Management system. It has eight entity sets. They are -

a) Song: (Attributes- Song_ID, Song_Name, Category, Movie_ID(fk), Lyricist_ID(fk), date_of_release, Distributor_Name(fk)).
b) Singer: (Attributes- Singer_ID, Singer_Name, Gender, Age).
c) Sings: (Attributes- Song_ID(fk), Singer_ID(fk)).
d) Movie: (Attributes- Movie_ID, Movie_Name).
e) Composer: (Attributes- Composer_ID, Composer_Name, Gender, Age).
f) Composes: (Attributes- Song_ID(fk), Composer_ID(fk)).
g) Lyricist: (Attributes- Lyricist_ID, Lyricist_Name, Gender, Age).
h) Song_Distributor: (Attributes- Distributor_Name, City, Website).

Abbreviations of all attributes are given in relational schema. Some notes about entity sets, their attributes and cardinalities among them –

- **Song -** A song is a musical composition intended to be performed by the human voice. Entity Song is associated/related with other entities like Singer, Composer, Lyricist, Song_Distributor.
- **Singer -** Who sings song. Singer_ID, Singer_Name, Gender and Age are included in the database.
- **Sings –** Relationship Table between Song and Singer. Song_ID and Singer_ID are the attributes which are foreign key in the table.
- **Movie –** Movie_ID and Movie_Name information will be stored in the database.
- **Composer –** Who composes song. Composer_ID, Composer_Name, Gender and Age are included in the database.
- **Composes -** Relationship Table between Song and Singer. Song_ID and Composer_ID are the attributes which are foreign key in the table.
- **Lyricist-** Who writes lyrics. Lyricist_ID, Lyricist_Name, Gender and Age are included in the database.
- **Song_Distributor -** Who distributes song. Distributor Name, City and Distributor's Website information will be stored in database.

Cardinality:

- **Song & Movie- (Relationship- (is of), Many to one)** One Song can be of one Movie while one Movie can have many songs.
- **Song & Singer (Relationship-(sings), Many to Many)** One Singer can sing many songs while One Song can be sung by many Singers.
- **Song & Composer- (Relationship-(composes), Many to Many)** One composer can compose many songs while One song can be composed by many Composers.
- **Song & Lyricist - (Relationship-(writes), Many to one).** One Song can be written by one lyricist only while One Lyricist can write many songs.

- **Song & Song_Distributor- (Relationship-(distributes), Many to One)** One Song can be distributed by one Distributor only while One Distributor can distribute many songs.

Participation Constraints:

- **Song & Movie (Total, Total)-** From Song Table, Total Participation is there and from Movie Table, there is Total Participation.
- **Song & Singer (Total, Partial)** -From Song Table, Total Participation is there and from Singer Table, there is Partial Participation.
- **Song & Composer Total, Partial)** -From Song Table, Total Participation is there and from Composer Table, there is Partial Participation.
- **Song & Lyricist Total, Partial)** -From Song Table, Total Participation is there and from Lyricist Table, there is Partial Participation.
- **Song & Song_Distributor (Total, Total)-** From Song Table, Total Participation is there and from Song_Distribution Table, there is Total Participation.

Relational Schemas

Song

Attribute Name	Description	Type
Song_ID	ID of the Song	Int
Song_Name	Name of the Song	varchar
Category	Category of the Song	Varchar
Movie_ID(fk)	ID of the Movie	Int
Lyricist_ID(fk)	ID of the Lyricist	Int
Date_of_release	Date at which distributor releases the song	date
Distributor_Name(fk)	Name of the Distributor	varchar

Singer

Attribute Name	Description	Type
Singer_ID	ID of the Singer	Int
Singer_Name	Name of the Singer	varchar
Gender	Gender of the Singer	varchar
Age	Age of the Singer	Int

Sings

Attribute Name	Description	Type
Song_ID(fk)	ID of the Song	Int
Singer_ID(fk)	ID of the Singer	Int

Movie

Attribute Name	Description	Type
Movie_ID	ID of the Movie	Int
Movie_Name	Name of the Movie	varchar

Composer

Attribute Name	Description	Type
Composer_ID	ID of the Composer	Int
Composer_Name	Name of the Composer	varchar
Gender	Gender of the Composer	varchar
Age	Age of the Composer	Int

Composes

Attribute Name	Description	Type
Song_ID(fk)	ID of the Song	Int
Composer_ID(fk)	ID of the Composer	Int

Lyricist

Attribute Name	Description	Type
Lyricist_ID	ID of the Lyricist	Int
Lyricist_Name	Name of the Lyricist	varchar
Gender	Gender of the Lyricist	varchar
Age	Age of the Lyricist	Int

Song_Distributor

Attribute Name	Description	Type
Distributor_Name	Name of the Song Distributor	Int
City	City of the Distributor	varchar
Website	Song Distributor Website	varchar

Source Code:

```
mysql> create database Music_Management_System;
Query OK, 1 row affected (0.05 sec)

mysql> use Music_Management_System;
Database changed
mysql> create table Movie(Movie_ID int primary key,Movie_Name varchar(35) not null);
Query OK, 0 rows affected (0.20 sec)

mysql> create table Singer(Singer_ID int primary key,Singer_Name varchar(35) not
null,Gender varchar(10),Age int);
Query OK, 0 rows affected (0.04 sec)

mysql> create table Composer(Composer_ID int primary key,Composer_Name varchar(35)
not null,Gender varchar(10),Age int);
Query OK, 0 rows affected (0.10 sec)

mysql> create table Lyricist(Lyricist_ID int primary key,Lyricist_Name varchar(35) not
null,Gender varchar(10),Age int);
Query OK, 0 rows affected (0.05 sec)

mysql> create table Song_Distributor(Distributor_Name varchar(40) primary key,City
varchar(50),Website varchar(50));
Query OK, 0 rows affected (0.11 sec)

mysql> create table Song(Song_ID int primary key,Song_Name varchar(50) not
null,Category varchar(20),Movie_ID int,Lyricist_ID int,Date_of_release
date,Distributor_Name varchar(40),foreign key(Movie_ID) references
Movie(Movie_ID),foreign key(Lyricist_ID) references Lyricist(Lyricist_ID),foreign
key(Distributor_Name) references Song_Distributor(Distributor_Name));
Query OK, 0 rows affected (0.08 sec)

mysql> create table Sings(Song_ID int,Singer_ID int,foreign key(Song_ID) references
Song(Song_ID),foreign key(Singer_ID) references Singer(Singer_ID));
Query OK, 0 rows affected (0.10 sec)

mysql> create table Composes(Song_ID int,Composer_ID int,foreign key(Song_ID)
references Song(Song_ID),foreign key(Composer_ID) references
Composer(Composer_ID));
Query OK, 0 rows affected (0.06 sec)

mysql> insert into Movie value(1,"Dangal");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Movie value(2,"Scam 1992");
```

```
Query OK, 1 row affected (0.01 sec)

mysql> insert into Movie value(3,"Tanhaji");
Query OK, 1 row affected (0.00 sec)

mysql> insert into Movie value(4,"Malang");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Movie value(5,"Sadak 2");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Movie value(6,"Dil Bechara");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Singer value(101,"Arjit Singh","Male",33);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Singer value(102,"Shreya Ghoshal","Female",37);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Singer value(103,"Sonu Nigam","Male",47);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Singer value(104,"Armaan Malik","Male",25);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Singer value(105,"Neha Kakkar","Female",32);
Query OK, 1 row affected (0.00 sec)

mysql> insert into Singer value(106,"Vishal Dadlani","Male",47);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Singer value(107,"Shekhar Ravjiani","Male",42);
Query OK, 1 row affected (0.00 sec)

mysql> insert into Composer value(111,"A.R. Rahman","Male",54);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Composer value(222,"Falguni Pathak","Female",50);
Query OK, 1 row affected (0.01 sec)
mysql> insert into Composer value(104,"Armaan Malik","Male",25);
Query OK, 1 row affected (0.00 sec)

mysql> insert into Composer value(107,"Shekhar Ravjiani","Male",42);
Query OK, 1 row affected (0.01 sec)
```

```
mysql> insert into Composer value(106,"Vishal Dadlani","Male",47);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Lyricist value(11,"Gulzar","Male",86);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Lyricist value(12,"Irshad Kamil",49);

mysql> insert into Lyricist value(12,"Irshad Kamil","Male", 49);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Lyricist value(13,"Javed Akhtar","Male", 76);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Lyricist value(14,"Mithoon","Male",36);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Lyricist value(15,"Anvita Dutt","Female",49);
Query OK, 1 row affected (0.01 sec)

mysql> insert into Song_Distributor value("Saregama
India","Kolkata","www.saregama.com");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Song_Distributor value("T-Series","New Delhi","www.tseries.com");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Song_Distributor value("Sony Music","New
Delhi","www.sonymusic.com");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Song_Distributor value("Tips Industries","Mumbai","www.tips.in");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Song values(501,"Dangal","Title Song",1,11,'2017-7-21',"T-Series");
Query OK, 1 row affected (0.06 sec)

mysql> insert into
Song(Song_ID,Song_Name,Category,Movie_ID,Date_of_release,Distributor_Name)
values(502,"Scam","BG TUNE",'2020-10-17',"Sony Music");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Song values(503,"Ghamand kar","Theme Song",3,13,'2020-2-20',"T-
Series");
Query OK, 1 row affected (0.00 sec)
```

```
mysql> insert into Song value(504,"Tinak Tinak","Patriotic",3,12,'2020-2-20',"T-Series");
Query OK, 1 row affected (0.05 sec)

mysql> insert into Song value(505,"Malang","Title Song",4,15,'2018-4-26',"Tips
Industries");
Query OK, 1 row affected (0.01 sec)

mysql> insert into Song value(506,"Ghar Chalen","Romantic Song",4,13,'2018-4-28',"Tips
Industries");
Query OK, 1 row affected (0.04 sec)

mysql> insert into Song value(507,"Tum se hi","Romantic Song",5,12,'2016-9-
13',"Saregama India");
Query OK, 1 row affected (0.00 sec)

mysql> insert into Song value(508,"Shukriya","Sad Song",5,12,'2016-9-13',"Saregama
India");
Query OK, 1 row affected (0.05 sec)

mysql> insert into Song value(509,"FriendZone","Pop Song",6,15,'2019-5-9',"Sony
Music");
Query OK, 1 row affected (0.04 sec)

mysql> insert into Song value(510,"Dil Bechara","Title Song",6,15,'2019-5-12',"Sony
Music");
Query OK, 1 row affected (0.01 sec)

mysql> insert into sings value(501,101);
Query OK, 1 row affected (0.05 sec)

mysql> insert into sings value(502,107);
Query OK, 1 row affected (0.01 sec)

mysql> insert into sings value(503,104);
Query OK, 1 row affected (0.00 sec)

mysql> insert into sings value(503,106);
Query OK, 1 row affected (0.01 sec)

mysql> insert into sings value(504,101);
Query OK, 1 row affected (0.03 sec)

mysql> insert into sings value(504,107);
Query OK, 1 row affected (0.01 sec)

mysql> insert into sings value(505,104);
```

```
Query OK, 1 row affected (0.01 sec)

mysql> insert into sings value(506,101);
Query OK, 1 row affected (0.01 sec)

mysql> insert into sings value(507,106);
Query OK, 1 row affected (0.03 sec)

mysql> insert into sings value(507,102);
Query OK, 1 row affected (0.00 sec)

mysql> insert into sings value(508,105);
Query OK, 1 row affected (0.01 sec)

mysql> insert into sings value(509,106);
Query OK, 1 row affected (0.03 sec)

mysql> insert into sings value(510,101);
Query OK, 1 row affected (0.01 sec)

mysql> insert into sings value(510,102);
Query OK, 1 row affected (0.01 sec)

mysql> insert into sings(Singer_ID) value(103);
Query OK, 1 row affected (0.01 sec)

mysql> insert into composes values(501,106);
Query OK, 1 row affected (0.06 sec)

mysql> insert into composes values(502,104);
Query OK, 1 row affected (0.01 sec)

mysql> insert into composes values(503,111);
Query OK, 1 row affected (0.01 sec)

mysql> insert into composes values(504,104);
Query OK, 1 row affected (0.01 sec)

mysql> insert into composes values(504,106);
Query OK, 1 row affected (0.01 sec)

mysql> insert into composes values(505,111);
Query OK, 1 row affected (0.02 sec)

mysql> insert into composes values(506,106);
Query OK, 1 row affected (0.00 sec)
```

```
mysql> insert into composes values(507,222);
Query OK, 1 row affected (0.01 sec)

mysql> insert into composes values(507,222);
Query OK, 1 row affected (0.01 sec)

mysql> insert into composes values(508,111);
Query OK, 1 row affected (0.01 sec)

mysql> insert into composes values(508,222);
Query OK, 1 row affected (0.01 sec)

mysql> insert into composes values(509,104);
Query OK, 1 row affected (0.00 sec)

mysql> insert into composes values(510,111);
Query OK, 1 row affected (0.00 sec)

mysql> insert into composes(Composer_ID) values(107);
Query OK, 1 row affected (0.01 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> notee;
```

Tables:

Song

Song_ID	Song_Name	Category	Movie_ID	Lyricist_ID	Date_of_release	Distributor_Name
501	Dangal	Title Song	1	11	2017-07-21	T-Series
502	Scam	BG TUNE	2	NULL	2020-10-17	Sony Music
503	Ghamand kar	Theme Song	3	13	2020-02-20	T-Series
504	Tinak Tinak	Patriotic	3	12	2020-02-20	T-Series
505	Malang	Title Song	4	15	2018-04-26	Tips Industries
506	Ghar Chalen	Romantic Song	4	13	2018-04-28	Tips Industries
507	Tum se hi	Romantic Song	5	12	2016-09-13	Saregama India
508	Shukriya	Sad Song	5	12	2016-09-13	Saregama India
509	FriendZone	Pop Song	6	15	2019-05-09	Sony Music
510	Dil Bechara	Title Song	6	15	2019-05-12	Sony Music

Singer

Singer_ID	Singer_Name	Gender	Age
101	Arjit Singh	Male	33
102	Shreya Ghoshal	Female	37
103	Sonu Nigam	Male	47
104	Armaan Malik	Male	25
105	Neha Kakkar	Female	32
106	Vishal Dadlani	Male	47
107	Shekhar Ravjiani	Male	42

Sings

Song_ID	Singer_ID
501	101
502	107
503	104
503	106
504	101
504	107
505	104
506	101
507	106
507	102
508	105
509	106
510	101
510	102
NULL	103

Movie

Movie_ID	Movie_Name
1	Dangal
2	Scam 1992
3	Tanhaji
4	Malang
5	Sadak 2
6	Dil Bechara

Composer

Composer_ID	Composer_Name	Gender	Age
104	Armaan Malik	Male	25
106	Vishal Dadlani	Male	47
107	Shekhar Ravjiani	Male	42
111	A.R. Rahman	Male	54
222	Falguni Pathak	Female	50

Composes

Song_ID	Composer_ID
501	106
502	104
503	111
504	104
504	106
505	111
506	106
508	111
508	222
509	104
510	111
507	222
NULL	107

Lyricist

Lyricist_ID	Lyricist_Name	Gender	Age
11	Gulzar	Male	86
12	Irshad Kamil	Male	49
13	Javed Akhtar	Male	76
14	Mithoon	Male	36
15	Anvita Dutt	Female	49

Song_Distributor

Distributor_Name	City	Website
Saregama India	Kolkata	www.saregama.com
Sony Music	New Delhi	www.sonymusic.com
T-Series	New Delhi	www.tseries.com
Tips Industries	Mumbai	www.tips.in

Queries:

1. **List Name of Song where Composer is Vishal Dadlani, Lyricist is Gulzar and Movie is Dangal.**

```
mysql> Select song_Name from Song natural join Composes where Composer_ID=(select Composer_ID from Composer wher
e Composer_Name="Vishal Dadlani") and Movie_ID=(select Movie_ID from Movie where Movie_Name="Dangal") and Lyrici
st_ID=(select Lyricist_ID from Lyricist where Lyricist_Name="Gulzar");
+-----------+
| song_Name |
+-----------+
| Dangal    |
+-----------+
```

2. **Display names of Singer who are not Composer.**

```
mysql> select singer_name from singer where singer_id not in(select composer_ID from composer);
+----------------+
| singer_name    |
+----------------+
| Arjit Singh    |
| Shreya Ghoshal |
| Sonu Nigam     |
| Neha Kakkar    |
+----------------+
```

3. **Display names of Composer who are not Singer.**

```
mysql> select composer_name from composer where composer_id not in(select singer_ID from singer);
+----------------+
| composer_name  |
+----------------+
| A.R. Rahman    |
| Falguni Pathak |
+----------------+
```

4. **List names of Lyricist who have written lyrics for movie "Malang".**

```
mysql> select Lyricist_Name from Song natural join Lyricist natural join Movie where movie_Name="Malang";
+--------------+
| Lyricist_Name |
+--------------+
| Anvita Dutt   |
| Javed Akhtar  |
+--------------+
```

5. **List names of singer who have not sung any song in any movie**

```
mysql> select singer_Name from Singer natural join sings  where song_id IS NULL;
+-------------+
| singer_Name |
+-------------+
| Sonu Nigam  |
+-------------+
```

6. **Display names of composers who are singers also.**

```
mysql> select composer_Name from composer,singer where singer_ID=composer_ID;
+------------------+
| composer_Name    |
+------------------+
| Armaan Malik     |
| Vishal Dadlani   |
| Shekhar Ravjiani |
+------------------+
```

7. **Display the names of Song where Singer associated with song is female.**

```
mysql> select Song_Name from Singer natural join Song natural join Sings where gender="Female";
+-------------+
| Song_Name   |
+-------------+
| Tum se hi   |
| Dil Bechara |
| Shukriya    |
+-------------+
```

8. **Display the names of Distributor who have distributed more than 2 Songs.**

```
mysql> select count(song_id),distributor_name from song group by distributor_name having count(song_id)>2;
+----------------+------------------+
| count(song_id) | distributor_name |
+----------------+------------------+
|              3 | Sony Music       |
|              3 | T-Series         |
+----------------+------------------+
```

9. **Display names of Songs where Singer is "Arjit Singh" and Distributor is "T-Series".**

```
mysql> select song_name from song natural join singer natural join sings where singer_name="Arjit Singh" and Dis
tributor_name="T-series";
+-------------+
| song_name   |
+-------------+
| Dangal      |
| Tinak Tinak |
+-------------+
```

10. **Display names of lyricist who have written lyrics for more than 1 song**.

```
mysql> select count(song_ID),lyricist_name from song natural join lyricist group by lyricist_name having count(s
ong_ID)>1;
+----------------+---------------+
| count(song_ID) | lyricist_name |
+----------------+---------------+
|              3 | Irshad Kamil  |
|              2 | Javed Akhtar  |
|              3 | Anvita Dutt   |
+----------------+---------------+
```

11. **Display names of Lyricist where Singer is Neha Kakkar.**

```
mysql> Select lyricist_name from song natural join lyricist where song_id in (select song_id from sings natural
join singer where singer_name="Neha Kakkar");
+---------------+
| lyricist_name |
+---------------+
| Irshad Kamil  |
+---------------+
```

12. List names of song released in last 2 years.

```
mysql> select song_name from song where (year(curdate()) - year(date_of_release)) <=2;
+-------------+
| song_name   |
+-------------+
| Scam        |
| Ghamand kar |
| Tinak Tinak |
| FriendZone  |
| Dil Bechara |
+-------------+
```

13. Display names of Song ending with a.

```
mysql> select Song_Name from Song where Song_name like '%a';
+--------------+
| Song_Name    |
+--------------+
| Shukriya     |
| Dil Bechara  |
+--------------+
```

14. Display names of Song where Category="Romantic Song".

```
mysql> select Song_Name from Song where Category="Romantic Song";
+--------------+
| Song_Name    |
+--------------+
| Ghar Chalen  |
| Tum se hi    |
+--------------+
```