MUSIC DATABASE MANAGEMENT

Submitted by:
Name : Ankitha C
SRN : PES1UG20CS626
Section : K

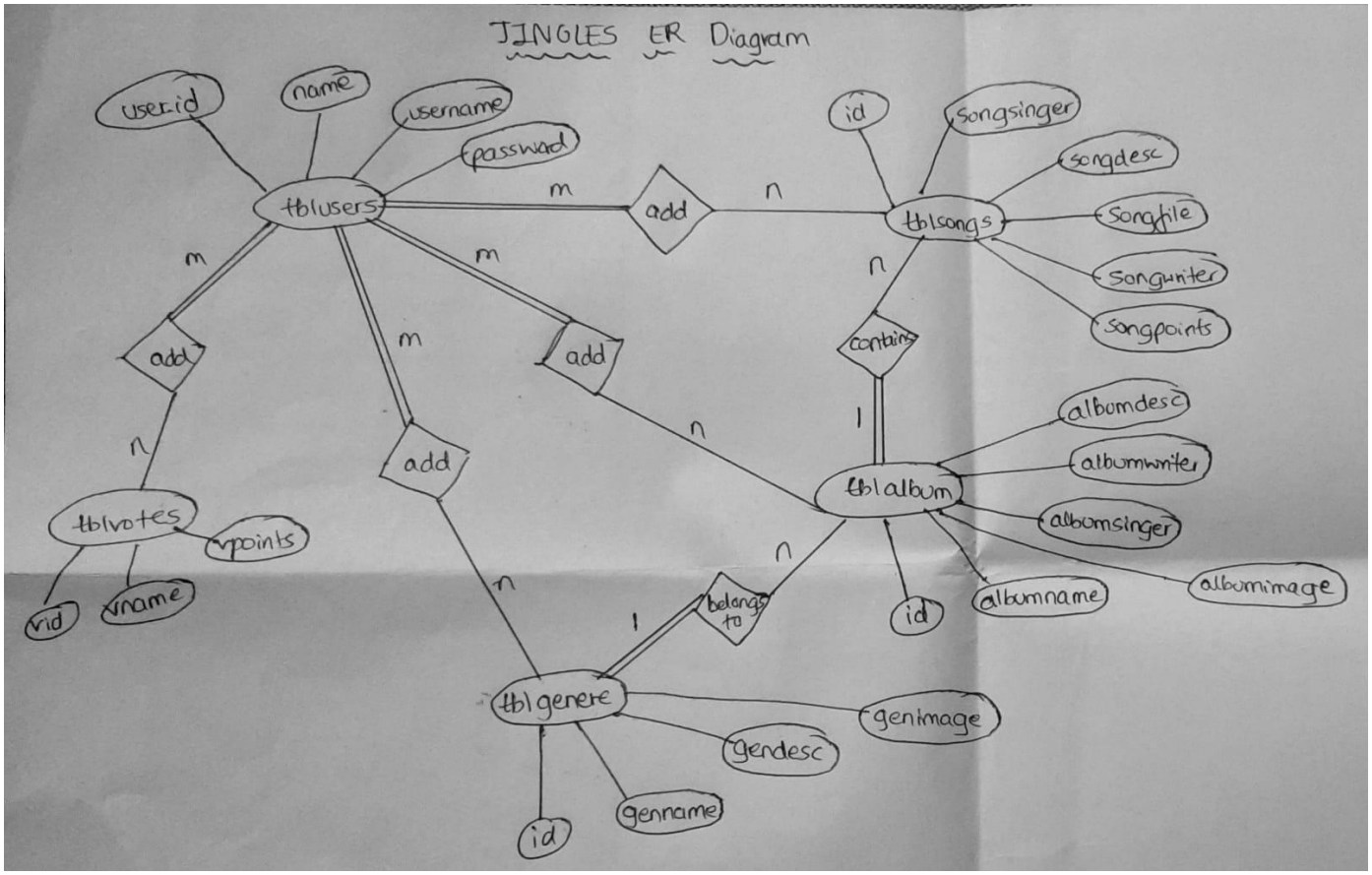
ABSTRACT

JINGLES music database website has a user side and an admin side, where a user can easily see the available albums and play the music. Furthermore, the admin is crucial to the management of this system. All of the primary tasks in this project must be completed by the user from the admin side.

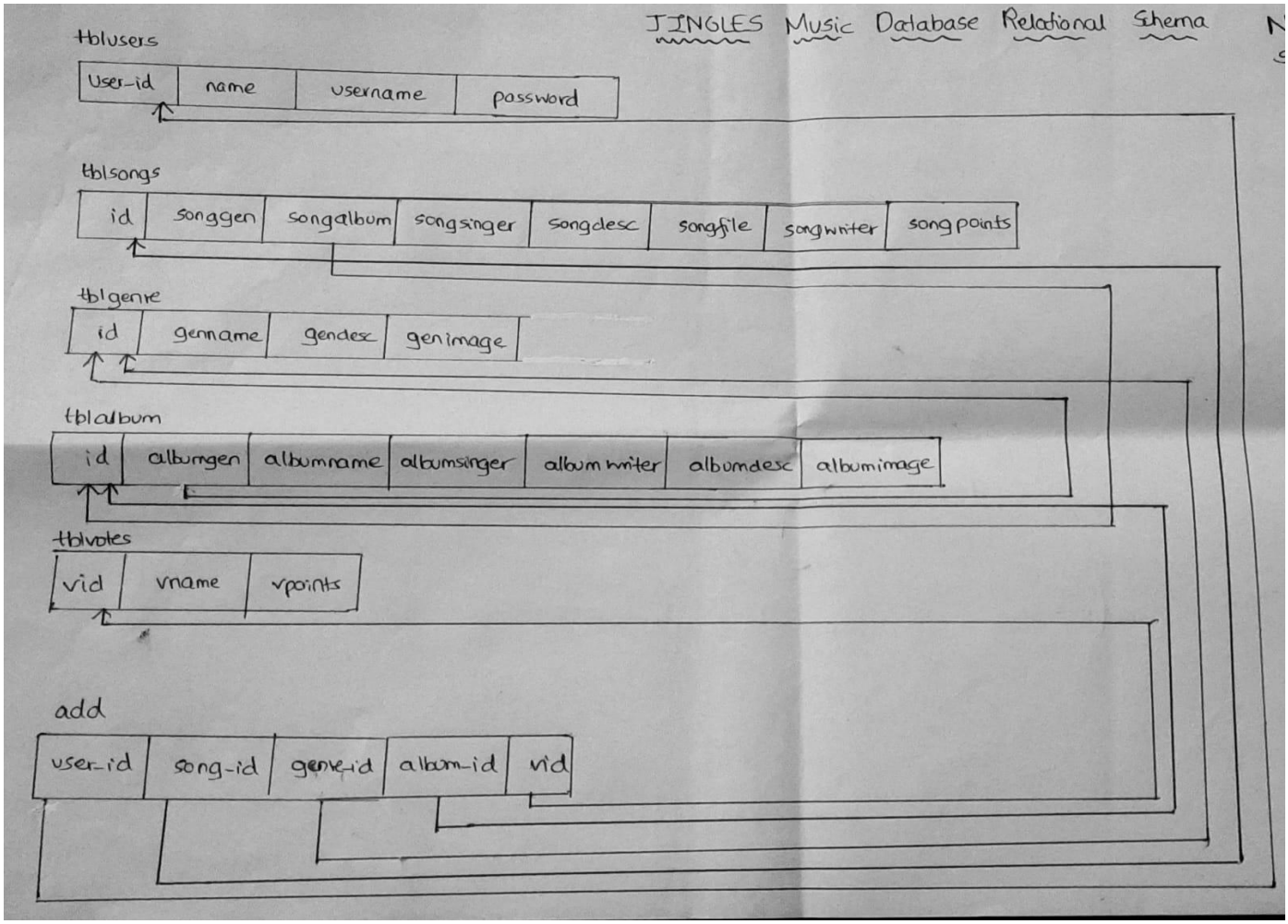
The user can view all of the most recent releases, the top 10 songs with rankings, news, and highlighted music. They can also vote for and listen to their favorite songs. The customers have the option to view every album and select any one to listen to its tracks.

The admin has total access to the system from the admin panel. Each music record can be managed by him or her. The administrator must choose a genre, name, performer, writer, description, and cover photo before adding albums. Admins have the ability to add album genres. Additionally, he or she can just add songs to the album recordings that already exist.

ER DIAGRAM


JINGLES ER Diagram
userid
name
username
password
tblusers
m
add
n
id
songsinger
songdesc
songfile
songwriter
songpoints
tblsongs
m
m
n
add
add
add
contains
n
n
1
albumdesc
albumwriter
albumsinger
albumimage
albumname
id
tblalbum
tblvotes
vpoints
vid
vname
n
n
1
belongs to
tblgenere
genimage
gendesc
genname
id

RELATIONAL SCHEMA



DDL Statements

```
CREATE TABLE 'tblalbum' (
  'id' int(100) NOT NULL,
  'albumgen' int(100) DEFAULT NULL,
  'albumname' varchar(60) DEFAULT NULL,
  'albumsinger' varchar(100) DEFAULT NULL,
  'albumwriter' varchar(100) DEFAULT NULL,
  'albumdesc' varchar(250) DEFAULT NULL,
  'albumimage' varchar(30) DEFAULT NULL
);

ALTER TABLE 'tblalbum'
```

```
  ADD PRIMARY KEY ('id');



CREATE TABLE 'tblgenre' (
  'id' int(10) NOT NULL,
  'genname' varchar(50) DEFAULT NULL,
  'gENDesc' varchar(250) DEFAULT NULL,
  'genimage' varchar(30) DEFAULT NULL
);

ALTER TABLE 'tblgenre'
  ADD PRIMARY KEY ('id');

CREATE TABLE 'tblsongs' (
  'id' int(100) NOT NULL,
  'songgen' varchar(10) DEFAULT NULL,
  'songalbum' varchar(50) DEFAULT NULL,
  'songsinger' varchar(100) DEFAULT NULL,
  'songdesc' varchar(250) DEFAULT NULL,
  'songfile' varchar(50) DEFAULT NULL,
  'songwriter' varchar(100) NOT NULL,
  'songpoints' int(100) NOT NULL
)

ALTER TABLE 'tblsongs'
  ADD PRIMARY KEY ('id');

CREATE TABLE 'tblusers' (
  'user_id' int(100) NOT NULL,
  'name' varchar(60) NOT NULL,
  'username' varchar(30) NOT NULL,
  'pASsword' varchar(30) NOT NULL
)

ALTER TABLE 'tblusers'
  ADD PRIMARY KEY ('user_id');

CREATE TABLE 'tblvotes' (
  'vid' int(10) NOT NULL,
  'vname' varchar(50) NOT NULL,
  'vpoints' int(10) NOT NULL DEFAULT 0
)
```

```
ALTER TABLE 'tblvotes'
  ADD PRIMARY KEY ('vid');
```

DML Statements

```
INSERT INTO `tblalbum`
(`id`,`albumgen`,`albumname`,`albumsinger`,`albumwriter`,`albumdesc`,`albumimage`)VALUES
(117, 30, 'Flute', 'Suhan', 'Suhan', 'Instrumental Flute', 'flute.jpeg'),
(118, 30, 'Piano', 'Anu', 'Anu', 'Instrumental Piano', 'piano.jpeg'),
(120, 32, 'Indian', 'Anvitha', 'Anvitha', 'Indian Patriotic', 'patriotic.jpeg'),
(121, 29, 'Carnatic', 'Ram', 'Ram', 'Carnatic ClASsical', 'clASsical.jpeg'),
(122, 29, 'Hindustani', 'Tom', 'Tom', 'Hindustani ClASsical', 'clASsical.jpeg'),
(123, 32, 'National', 'Nation', 'Nation', 'National Patriotic', 'patriotic.jpeg'),
(124, 33, 'Kannada Melody', 'Kannada', 'Kannada', 'Kannada Melody', 'Melody.png'),
(125, 33, 'Hindi Melody', 'Hindi', 'Hindi', 'Hindi Melody', 'Melody.png'),
(126, 28, 'South India', 'Kannada', 'Kannada', 'Southern Popular', 'popular.jpeg'),
(127, 28, 'English Popular', 'English', 'English', 'English Popular', 'popular.jpeg'),
(128, 28, 'North India', 'Hindi', 'Hindi', 'Northern Popular', 'popular.jpeg'),
```

```
INSERT INTO `tblgenre` (`id`, `genname`, `gENDesc`, `genimage`) VALUES
(33, 'Melody', 'Melody Music', 'Melody.png'),
(32, 'Patriotic', 'Patriotic Music', 'patriotic.jpeg'),
(28, 'Popular', 'Popular Music', 'popular.jpeg'),
(29, 'ClASsical', 'ClASsical Music', 'clASsical.jpeg'),
(30, 'Instrument', 'Instrumental Music', 'instrumental.jpeg');
```

```
INSERT INTO `tblsongs` (`id`, `songgen`, `songalbum`, `songsinger`, `songdesc`, `songfile`,
`songwriter`, `songpoints`) VALUES
(59, 'Instrument', '118', 'Anu', 'piano 4', 'piano4.mp3', 'Anu', 0),
(57, 'Instrument', '118', 'Anu', 'piano 2', 'piano2.mp3', 'Anu', 0),
(56, 'Instrument', '118', 'Anu', 'piano1', 'piano1.mp3', 'Anu', 0),
(68, 'Patriotic', '120', 'Anvitha', 'Indian 1', 'Indian1.mp3', 'Anvitha', 0),
(63, 'Instrument', '117', 'Suhan', 'Flute 3', 'flute3.mp3', 'Suhan', 1),
(60, 'Instrument', '118', 'Anu', 'piano 5', 'piano5.mp3', 'Anu', 0),
(58, 'Instrument', '118', 'Anu', 'piano 3', 'piano3.mp3', 'Anu', 6),
(61, 'Instrument', '117', 'Suhan', 'Flute 1', 'flute1.mp3', 'Suhan', 29),
(62, 'Instrument', '117', 'Suhan', 'Flute 2', 'flute2.mp3', 'Suhan', 2),
(71, 'Patriotic', '120', 'Anvitha', 'Indian 4', 'Indian4.mp3', 'Anvitha', 1),
(70, 'Patriotic', '120', 'Anvitha', 'Indian 3', 'Indian3.mp3', 'Anvitha', 0),
(69, 'Patriotic', '120', 'Anvitha', 'Indian 2', 'indian2.mp3', 'Anvitha', 1),
```

(72, 'ClASsical', '121', 'Ram', 'Carnatic1', 'carnatic1.mp3', 'Ram', 1),
(73, 'ClASsical', '122', 'Tom', 'hindustani 1', 'hindustani1.mp3', 'Tom', 1),
(74, 'Melody', '124', 'Kannada', 'Kannada Melody1', 'KannadaMelody1.mp3', 'Kannada', 3),
(75, 'Melody', '124', 'Kannada', 'Kannada Melody2', 'KannadaMelody2.mp3', 'Kannada', 1),
(76, 'Melody', '125', 'Hindi', 'Hindi Melody1', 'HindiMelody1.mp3', 'Hindi', 0),
(77, 'Melody', '125', 'Hindi', 'Hindi Melody2', 'Hindi Melody2.mp3', 'Hindi', 1),
(78, 'Melody', '125', 'Hindi', 'Hindi Melody3', 'Hindi Melody3.mp3', 'Hindi', 1),
(79, 'Popular', '126', 'Kannada', 'South India1', 'South India1.mp3', 'Kannada', 2),
(80, 'Popular', '126', 'Kannada', 'South India3', 'South India3.mp3', 'Kannada', 0),
(81, 'Popular', '127', 'English', 'English Popular3', 'English Popular3.mp3', 'English', 0),
(82, 'Popular', '127', 'English', 'English Popular1', 'English Popular1.mp3', 'English', 1),
(83, 'Popular', '127', 'English', 'English Popular2', 'English Popular2.mp3', 'English', 1),
(84, 'Popular', '128', 'Hindi', 'North India1', 'North India1.mp3', 'Hindi', 1),
(85, 'Popular', '128', 'Hindi', 'North India2', 'North India2.mp3', 'Hindi', 2);

INSERT INTO `tblusers` (`user_id`, `name`, `username`, `pASsword`) VALUES
(15, 'Ramya', 'Ramya', 'Ramyaabc'),
(3, 'Ankitha', 'Ankitha', 'Ankithaabc'),
(4, 'Suman', 'Suman', 'Suman@pASs');

INSERT INTO `tblvotes` (`vid`, `vname`, `vpoints`) VALUES
(7, 'Melody', 5),
(6, 'Patriotic', 0),
(5, 'Popular', 1),
(8, 'ClASsical', 2),
(9, 'Instrument', 3);

TOOLS USED

FRONTEND: HTML, CSS
BACKEND : PHP, JAVASCRIPT
DATABASE : MYSQL

REGULAR JOIN

1. Display number of albums in each genre in descending order

```
SELECT COUNT(tblalbum.id) AS no_of_albums, tblgenre.genname
FROM tblgenre JOIN tblalbum
WHERE tblgenre.id = tblalbum.albumgen
GROUP BY genname
HAVING COUNT(tblalbum.id) > 1
ORDER BY COUNT(tblalbum.id)DESC;
```

```
MariaDB [dbmis]> SELECT COUNT(tblalbum.id) as no_of_albums,tblgenre.genname
    -> FROM tblgenre JOIN tblalbum
    -> WHERE tblgenre.id=tblalbum.albumgen
    -> GROUP BY genname
    -> HAVING COUNT(tblalbum.id) > 1
    -> ORDER BY COUNT(tblalbum.id)DESC;
+--------------+------------+
| no_of_albums | genname    |
+--------------+------------+
|            3 | Popular    |
|            2 | Instrument |
|            2 | Patriotic  |
|            2 | Classical  |
|            2 | Melody     |
+--------------+------------+
5 rows in set (0.001 sec)

MariaDB [dbmis]>
MariaDB [dbmis]>
```

2. Write a JOIN query to display the song files with corresponding genre and album.

```
SELECT tblsongs.songfile, tblgenre.genname, tblalbum.albumname
FROM ((tblgenre INNER JOIN tblalbum ON tblgenre.id = tblalbum.albumgen)
INNER JOIN tblsongs ON tblsongs.songalbum = tblalbum.id)
WHERE songfile like "E%";
```

```
MariaDB [dbmis]> SELECT tblsongs.songfile,tblgenre.genname,tblalbum.albumname FROM ((tblgenre INNER JOIN tblalbum ON tbl
genre.id = tblalbum.albumgen)
    -> INNER JOIN tblsongs ON tblsongs.songalbum =tblalbum.id) where songfile like "E%";
+----------------------+---------+-----------------+
| songfile             | genname | albumname       |
+----------------------+---------+-----------------+
| English Popular3.mp3 | Popular | English Popular |
| English Popular1.mp3 | Popular | English Popular |
| English Popular2.mp3 | Popular | English Popular |
+----------------------+---------+-----------------+
3 rows in set (0.001 sec)
```

3. Write a JOIN query to display album details with their corresponding genre.

```
SELECT tblalbum.albumname, tblalbum.albumimage, tblalbum.albumsinger, tblgenre.genname,
tblalbum.albumwriter
FROM  tblgenre INNER JOIN tblalbum
ON tblalbum.albumgen = tblgenre.id;
```

```
MariaDB [dbmis]> SELECT tblalbum.albumname,tblalbum.albumimage,tblalbum.albumsinger,tblgenre.genname,
    -> tblalbum.albumwriter
    -> FROM
    -> tblgenre INNER JOIN tblalbum ON tblalbum.albumgen = tblgenre.id;
+-----------------+----------------+-------------+------------+-------------+
| albumname       | albumimage     | albumsinger | genname    | albumwriter |
+-----------------+----------------+-------------+------------+-------------+
| Flute           | flute.jpeg     | Suhan       | Instrument | Suhan       |
| Piano           | piano.jpeg     | Anu         | Instrument | Anu         |
| Indian          | patriotic.jpeg | Anvitha     | Patriotic  | Anvitha     |
| Carnatic        | classical.jpeg | Ram         | Classical  | Ram         |
| Hindustani      | classical.jpeg | Tom         | Classical  | Tom         |
| National        | patriotic.jpeg | Nation      | Patriotic  | Nation      |
| Kannada Melody  | Melody.png     | Kannada     | Melody     | Kannada     |
| Hindi Melody    | Melody.png     | Hindi       | Melody     | Hindi       |
| South India     | popular.jpeg   | Kannada     | Popular    | Kannada     |
| English Popular | popular.jpeg   | English     | Popular    | English     |
| North India     | popular.jpeg   | Hindi       | Popular    | Hindi       |
+-----------------+----------------+-------------+------------+-------------+
11 rows in set (0.001 sec)

MariaDB [dbmis]> S
```

4. Perform a JOIN operation to display the song files with the album they are belonging to.

```
SELECT tblgenre.genname, tblvotes.vpoints
FROM tblgenre
CROSS JOIN tblvotes
WHERE (tblgenre.genname = tblvotes.vpoints AND count(tblvotes.vpoints >=1));
```

```
MariaDB [dbmis]> SELECT tblsongs.songfile, tblalbum.albumname
    -> FROM tblsongs
    -> LEFT JOIN tblalbum ON tblsongs.songalbum = tblalbum.id
    -> ORDER BY tblsongs.songpoints;
+----------------------+-----------------+
| songfile             | albumname       |
+----------------------+-----------------+
| piano4.mp3           | Piano           |
| South India3.mp3     | South India     |
| English Popular3.mp3 | English Popular |
| piano5.mp3           | Piano           |
| HindiMelody1.mp3     | Hindi Melody    |
| Indian1.mp3          | Indian          |
| piano1.mp3           | Piano           |
| piano2.mp3           | Piano           |
| Indian3.mp3          | Indian          |
| Hindi Melody2.mp3    | Hindi Melody    |
| Hindi Melody3.mp3    | Hindi Melody    |
| English Popular1.mp3 | English Popular |
| English Popular2.mp3 | English Popular |
| North India1.mp3     | North India     |
| KannadaMelody2.mp3   | Kannada Melody  |
| Indian4.mp3          | Indian          |
| hindustani1.mp3      | Hindustani      |
| carnatic1.mp3        | Carnatic        |
| indian2.mp3          | Indian          |
| flute3.mp3           | Flute           |
| flute2.mp3           | Flute           |
| North India2.mp3     | North India     |
| South India1.mp3     | South India     |
| KannadaMelody1.mp3   | Kannada Melody  |
| piano3.mp3           | Piano           |
| flute1.mp3           | Flute           |
+----------------------+-----------------+
26 rows in set (0.009 sec)

MariaDB [dbmis]>
```

CORRELATED QUERIES

1. Retrieve those genres which have at leASt 1 vote.

```
SELECT genname
FROM tblgenre
WHERE EXISTS
(SELECT vpoints FROM tblvotes WHERE tblgenre.genname = tblvotes.vname AND vpoints
>=1);
```

```
MariaDB [dbmis]> SELECT genname
    -> FROM tblgenre
    -> WHERE EXISTS
    -> (SELECT vpoints FROM tblvotes WHERE tblgenre.genname = tblvotes.vname AND vpoints >=1);
+------------+
| genname    |
+------------+
| Melody     |
| Popular    |
| Classical  |
| Instrument |
+------------+
4 rows in set (0.013 sec)

MariaDB [dbmis]>
```

2. Retrieve song description whose album is “Flute”

```
SELECT songdesc
FROM tblsongs
WHERE songalbum=(
SELECT id
FROM tblalbum
WHERE albumname="Flute");
```

```
MariaDB [dbmis]> SELECT songdesc
    -> FROM tblsongs
    -> WHERE songalbum=(
    -> SELECT id
    -> FROM tblalbum
    -> WHERE albumname="Flute");
+----------+
| songdesc |
+----------+
| Flute 3  |
| Flute 1  |
| Flute 2  |
+----------+
3 rows in set (0.002 sec)
```

NESTED QUERIES

1. Write a query to display album names with songpoints>=2.

```
SELECT albumname
FROM tblalbum
WHERE id = ANY
( SELECT songalbum
FROM tblsongs
WHERE songpoints > 2);
```

```
MariaDB [dbmis]> SELECT albumname
    -> FROM tblalbum
    -> WHERE id = ANY
    -> ( SELECT songalbum
    -> FROM tblsongs
    -> WHERE songpoints > 2);
+----------------+
| albumname      |
+----------------+
| Piano          |
| Flute          |
| Kannada Melody |
+----------------+
3 rows in set (0.001 sec)

MariaDB [dbmis]>
```

2. Write a query to display the genre WHERE albumwriter=”kannada”.

```
SELECT genname
FROM tblgenre WHERE id= ANY
( SELECT albumgen FROM tblalbum WHERE albumwriter="kannada");
```

```
MariaDB [dbmis]> SELECT genname
    -> FROM tblgenre WHERE id= ANY
    -> ( SELECT albumgen FROM tblalbum WHERE albumwriter="kannada");
+---------+
| genname |
+---------+
| Melody  |
| Popular |
+---------+
2 rows in set (0.003 sec)

MariaDB [dbmis]>
```

AGGREGATE FUNCTIONS

1. Write a query to display number of songs in each album

```
SELECT COUNT(tblsongs.id) AS no_of_songs, albumname
FROM tblalbum JOIN tblsongs
WHERE tblsongs.songalbum = tblalbum.id
GROUP BY tblalbum.albumname;
```

```
MariaDB [dbmis]> SELECT COUNT(tblsongs.id) AS no_of_songs, albumname
    -> FROM tblalbum JOIN tblsongs
    -> WHERE tblsongs.songalbum = tblalbum.id
    -> GROUP BY tblalbum.albumname;
+-------------+-----------------+
| no_of_songs | albumname       |
+-------------+-----------------+
|           1 | Carnatic        |
|           3 | English Popular |
|           3 | Flute           |
|           3 | Hindi Melody    |
|           1 | Hindustani      |
|           4 | Indian          |
|           2 | Kannada Melody  |
|           2 | North India     |
|           5 | Piano           |
|           2 | South India     |
+-------------+-----------------+
10 rows in set (0.003 sec)

MariaDB [dbmis]> _
```

2. Write a query to display sum of songpoints for each genre

```
SELECT SUM(songpoints) AS points_for_each_genre,genname
FROM tblsongs JOIN tblgenre
WHERE tblsongs.songgen = tblgenre.genname
GROUP BY tblgenre.genname;
```

```
MariaDB [dbmis]> SELECT SUM(songpoints) AS points_for_each_genre,genname
    -> FROM tblsongs JOIN tblgenre
    -> WHERE tblsongs.songgen = tblgenre.genname
    -> GROUP BY tblgenre.genname;
+----------------------+------------+
| points_for_each_genre | genname    |
+----------------------+------------+
|                    2 | Classical  |
|                   38 | Instrument |
|                    6 | Melody     |
|                    2 | Patriotic  |
|                    7 | Popular    |
+----------------------+------------+
5 rows in set (0.004 sec)

MariaDB [dbmis]>
```

3. Find the song with highest votes

SELECT songfile, songgen, MAX(songpoints)
AS Highest_Rating
FROM tblsongs;

```
MariaDB [dbmis]> SELECT songfile, songgen, MAX(songpoints)
    -> AS Highest_Rating
    -> FROM tblsongs;
+------------+------------+----------------+
| songfile   | songgen    | Highest_Rating |
+------------+------------+----------------+
| piano4.mp3 | Instrument |             29 |
+------------+------------+----------------+
1 row in set (0.001 sec)

MariaDB [dbmis]>
```

SET OPERATIONS

1. Retrieve genres whose albumwriter is not “Hindi”

SELECT genname
FROM tblgenre

```
EXCEPT
SELECT albumname
FROM tblalbum
WHERE albumwriter = "Hindi";
```

```
MariaDB [dbmis]> SELECT genname
    -> FROM tblgenre
    ->
    -> EXCEPT
    -> SELECT albumname
    -> FROM tblalbum
    -> WHERE albumwriter = "Hindi";
+------------+
| genname    |
+------------+
| Melody     |
| Patriotic  |
| Popular    |
| Classical  |
| Instrument |
+------------+
5 rows in set (0.002 sec)

MariaDB [dbmis]>
```

2. Display singers who have sung at least one song in an album

```
SELECT songsinger AS sungby
FROM tblsongs
INTERSECT
SELECT albumsinger FROM tblalbum;
```

```
MariaDB [dbmis]> SELECT songsinger AS sungby
    -> FROM tblsongs INTERSECT
    -> SELECT albumsinger FROM tblalbum;
+---------+
| sungby  |
+---------+
| Anu     |
| Anvitha |
| Suhan   |
| Ram     |
| Tom     |
| Kannada |
| Hindi   |
| English |
+---------+
8 rows in set (0.001 sec)

MariaDB [dbmis]>
```

VIEW

1. Create a view to display 7 latest albums released

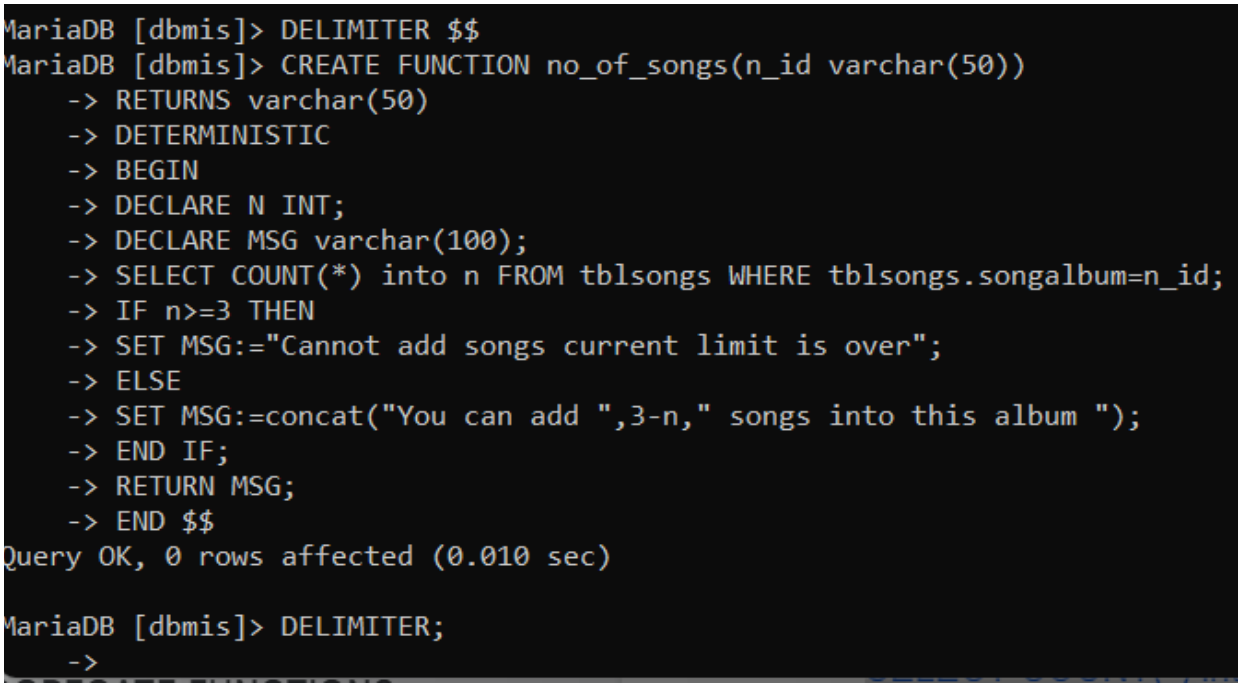
```
CREATE VIEW latest_release AS
SELECT * FROM tblalbum
ORDER BY id DESC LIMIT 7;
SELECT * FROM latest_release;
```

```
MariaDB [dbmis]> CREATE VIEW latest_release AS
    -> SELECT * FROM tblalbum
    -> ORDER BY id DESC LIMIT 7;
Query OK, 0 rows affected (0.008 sec)

MariaDB [dbmis]> SELECT * from latest_release;
+-----+----------+-----------------+-------------+-------------+----------------------+----------------+
| id  | albumgen | albumname       | albumsinger | albumwriter | albumdesc            | albumimage     |
+-----+----------+-----------------+-------------+-------------+----------------------+----------------+
| 128 |       28 | North India     | Hindi       | Hindi       | Northern Popular     | popular.jpeg   |
| 127 |       28 | English Popular | English     | English     | English Popular      | popular.jpeg   |
| 126 |       28 | South India     | Kannada     | Kannada     | Southern Popular     | popular.jpeg   |
| 125 |       33 | Hindi Melody    | Hindi       | Hindi       | Hindi Melody         | Melody.png     |
| 124 |       33 | Kannada Melody  | Kannada     | Kannada     | Kannada Melody       | Melody.png     |
| 123 |       32 | National        | Nation      | Nation      | National Patriotic   | patriotic.jpeg |
| 122 |       29 | Hindustani      | Tom         | Tom         | Hindustani Classical | classical.jpeg |
+-----+----------+-----------------+-------------+-------------+----------------------+----------------+
7 rows in set (0.002 sec)

MariaDB [dbmis]>
```

FUNCTION

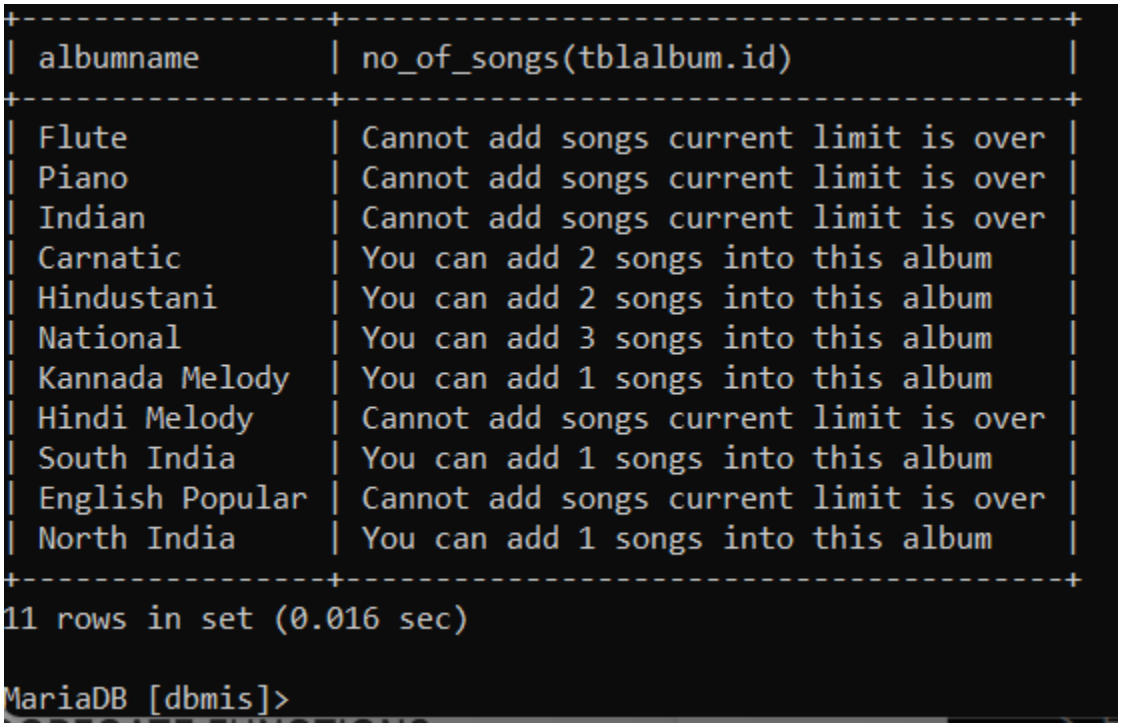
1. Write a function to display the number of songs that can be added into an album without crossing over 3 songs in each album. If the limit is crossed, display a warning message.

```
DELIMITER $$
CREATE FUNCTION no_of_songs(n_id varchar(50))
RETURNS varchar(50)
DETERMINISTIC
BEGIN
DECLARE N INT;
DECLARE MSG varchar(100);
SELECT COUNT(*) into n FROM tblsongs WHERE tblsongs.songalbum=n_id;
IF n>=3 THEN
SET MSG:="Cannot add songs current limit is over";
ELSE
SET MSG:=concat("You can add ",3-n," songs into this album ");
END IF;
RETURN MSG;
END $$
DELIMITER;
```



```
MariaDB [dbmis]> DELIMITER $$
MariaDB [dbmis]> CREATE FUNCTION no_of_songs(n_id varchar(50))
    -> RETURNS varchar(50)
    -> DETERMINISTIC
    -> BEGIN
    -> DECLARE N INT;
    -> DECLARE MSG varchar(100);
    -> SELECT COUNT(*) into n FROM tblsongs WHERE tblsongs.songalbum=n_id;
    -> IF n>=3 THEN
    -> SET MSG:="Cannot add songs current limit is over";
    -> ELSE
    -> SET MSG:=concat("You can add ",3-n," songs into this album ");
    -> END IF;
    -> RETURN MSG;
    -> END $$
Query OK, 0 rows affected (0.010 sec)

MariaDB [dbmis]> DELIMITER;
    ->
```

```
SELECT albumname,no_of_songs(tblalbum.id) FROM tblalbum;
```



albumname	no_of_songs(tblalbum.id)
Flute	Cannot add songs current limit is over
Piano	Cannot add songs current limit is over
Indian	Cannot add songs current limit is over
Carnatic	You can add 2 songs into this album
Hindustani	You can add 2 songs into this album
National	You can add 3 songs into this album
Kannada Melody	You can add 1 songs into this album
Hindi Melody	Cannot add songs current limit is over
South India	You can add 1 songs into this album
English Popular	Cannot add songs current limit is over
North India	You can add 1 songs into this album

11 rows in set (0.016 sec)

MariaDB [dbmis]>

PROCEDURE

1. Write a procedure to update vote by one points in an specified genre

```
DELIMITER $$
CREATE PROCEDURE update_vote(IN name_v varchar(100))
BEGIN
DECLARE V varchar(100);
UPDATE tblvotes
SET vpoints = vpoints+1 WHERE vname = name_v ;
END $$
```

```
MariaDB [dbmis]> DELIMITER $$
MariaDB [dbmis]> CREATE PROCEDURE update_vote(IN name_v varchar(100))
    -> BEGIN
    -> DECLARE V varchar(100);
    -> UPDATE tblvotes
    -> SET vpoints = vpoints+1 WHERE vname = name_v ;
    -> END $$
Query OK, 0 rows affected (0.015 sec)

MariaDB [dbmis]>
```

select * FROM tblvotes;

```
MariaDB [dbmis]> select * FROM tblvotes;
+-----+------------+---------+
| vid | vname      | vpoints |
+-----+------------+---------+
|   7 | Melody     |       5 |
|   6 | Patriotic  |       0 |
|   5 | Popular    |       1 |
|   8 | Classical  |       2 |
|   9 | Instrument |       4 |
+-----+------------+---------+
5 rows in set (0.000 sec)

MariaDB [dbmis]>
```

CALL update_vote(“Instrument”)

```
MariaDB [dbmis]> CALL update_vote("Instrument");
Query OK, 1 row affected (0.001 sec)
```

select * FROM tblvotes;

```
MariaDB [dbmis]> select * FROM tblvotes;
+-----+------------+---------+
| vid | vname      | vpoints |
+-----+------------+---------+
|   7 | Melody     |       5 |
|   6 | Patriotic  |       0 |
|   5 | Popular    |       1 |
|   8 | Classical  |       2 |
|   9 | Instrument |       5 |
+-----+------------+---------+
5 rows in set (0.001 sec)

MariaDB [dbmis]>
```

TRIGGER

1. Write a trigger to display an error message on password length less than 8 for adding a new user.

```
DELIMITER $$
CREATE TRIGGER password_error
BEFORE INSERT
ON tblusers FOR EACH ROW
BEGIN
DECLARE ERROR_MSG varchar(100);
DECLARE val varchar(100);
SET ERROR_MSG =("Password length must be at least of length 8");
SET val = length(new.password);
IF (val < 8) THEN
signal sqlstate '45000'
SET message_text = ERROR_MSG;
END IF;
END $$
```

```
MariaDB [dbmis]> DELIMITER $$
MariaDB [dbmis]> CREATE TRIGGER password_error
    -> BEFORE INSERT
    -> ON tblusers FOR EACH ROW
    -> BEGIN
    -> DECLARE ERROR_MSG varchar(100);
    -> DECLARE val varchar(100);
    -> SET ERROR_MSG =("Password length must be at least of length 8");
    -> SET val = length(new.password);
    -> IF (val < 8) THEN
    -> signal sqlstate '45000'
    -> SET message_text = ERROR_MSG;
    -> END IF;
    -> END $$
Query OK, 0 rows affected (0.014 sec)

MariaDB [dbmis]>
```

INSERT INTO tblusers VALUES(11,”Anu”,”Anushree”,”ie”);

```
MariaDB [dbmis]> insert into tblusers values(11,"Anu","Anushree","ie");
ERROR 1644 (45000): Password length must be at least of length 8
MariaDB [dbmis]>
```