DBMS - Mini Project

RECORD LABEL DATABASE

Submitted By:

Name:VINESH S

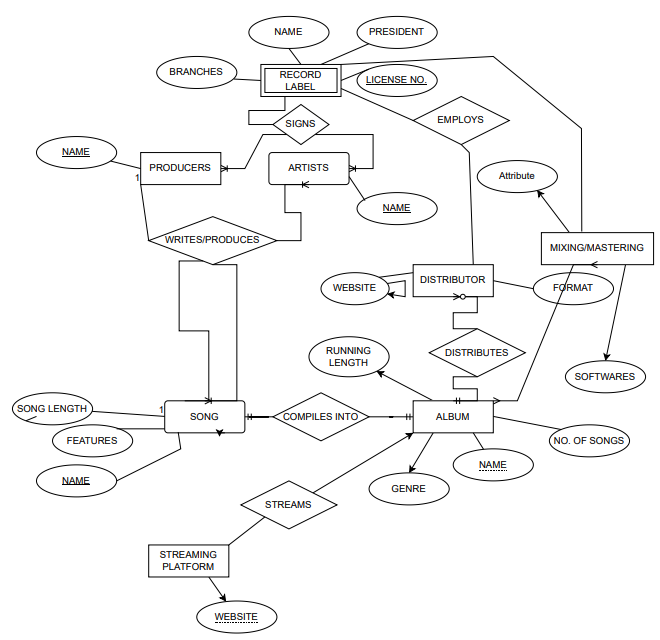
SRN:PES1UG20CS505

V Semester Section I

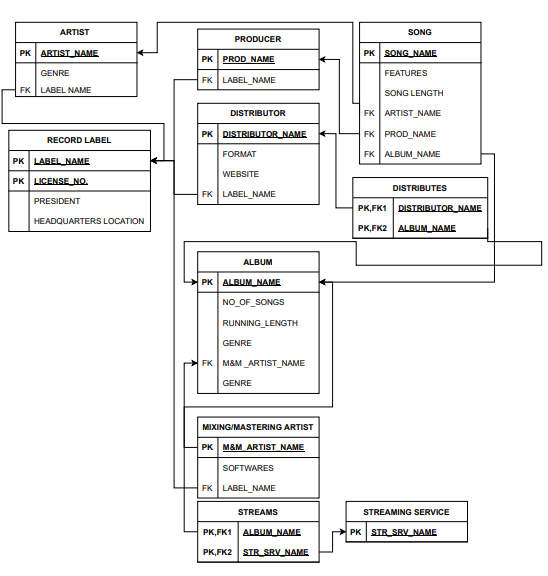
# Short Description and Scope of the Project:

This database application aims on maintaining the records of one of the most fundamental foundations of today’s music industry, the record labels. A record label provides support to various artist by helping them distribute their music, market it, give them a bigger audience by signing them under themselves. There are various nuances in making a record such as composing, producing mixing and mastering to deliver a final product. This final product has to be distributed in forms of streams and physical vinyl records by distributors. This application is a sample of such a database in action.

# ER Diagram



# Relational Schema



# DDL statements - Building the database

CREATE TABLE `dbms\_project`.`artist` (

`Artist\_name` CHAR NOT NULL,

`Genre` VARCHAR(45) NULL,

PRIMARY KEY (`Artist\_name`))

ALTER TABLE `dbms\_project`.`artist`

ADD COLUMN `label\_name` VARCHAR(45) NULL AFTER `Genre`;

CREATE TABLE `dbms\_project`.`record\_label` (

`Label\_name` CHAR NULL,

`license\_no` INT NOT NULL,

`President` VARCHAR(45) NULL,

`Headquarters/location` VARCHAR(45) NULL,

PRIMARY KEY (`Label\_name`, `license\_no`));

ALTER TABLE `dbms\_project`.`record\_label`

CHANGE COLUMN `Label\_name` `Label\_name` VARCHAR(45) NOT NULL ;

CREATE TABLE `dbms\_project`.`producer` (

`prod\_name` VARCHAR(45) NULL,

`label\_name` VARCHAR(45) NULL,

PRIMARY KEY (`prod\_name`));

CREATE TABLE `dbms\_project`.`song` (

`song\_name` VARCHAR(45) NULL,

`feature` VARCHAR(45) NULL,

`song\_length` INT NULL,

`artist\_name` VARCHAR(45) NULL,

`prod\_name` VARCHAR(45) NULL,

PRIMARY KEY (`song\_name`));

ALTER TABLE `dbms\_project`.`song`

ADD COLUMN `album\_name` VARCHAR(45) NULL AFTER `prod\_name`;

CREATE TABLE `dbms\_project`.`album` (

`album\_name` VARCHAR(45) NULL,

`no\_of\_songs` INT NULL,

`running\_length` INT NULL,

`album\_genre` VARCHAR(45) NULL,

`m&m\_artist\_name` VARCHAR(45) NULL,

PRIMARY KEY (`album\_name`));

CREATE TABLE `dbms\_project`.`distributor` (

`distributor\_name` VARCHAR(45) NOT NULL,

`format` VARCHAR(45) NULL,

`website` VARCHAR(45) NULL,

`label\_name` VARCHAR(45) NULL,

PRIMARY KEY (`distributor\_name`));

CREATE TABLE `dbms\_project`.`mixing&mastering\_artist` (

`m&m\_artist\_name` VARCHAR(45) NOT NULL,

`softwares` VARCHAR(45) NULL,

`label\_name` VARCHAR(45) NULL,

PRIMARY KEY (`m&m\_artist\_name`));

CREATE TABLE `dbms\_project`.`streaming\_service` (

`str\_srv\_name` VARCHAR(45) NOT NULL,

PRIMARY KEY (`str\_srv\_name`));

CREATE TABLE `dbms\_project`.`distributes` (

`distributor\_name` VARCHAR(45) NOT NULL,

`album\_name` VARCHAR(45) NULL);

CREATE TABLE `dbms\_project`.`streams` (

`album\_name` VARCHAR(45) NOT NULL,

`str\_srv\_name` VARCHAR(45) NOT NULL,

PRIMARY KEY (`album\_name`, `str\_srv\_name`));

# POPULATING TABLES:

INSERT INTO `dbms\_project`.`album` (`album\_name`, `no\_of\_songs`, `running\_length`, `album\_genre`, `m&m\_artist\_name`) VALUES ('malibu', '12', '56', 'r&b', 'sam');

INSERT INTO `dbms\_project`.`album` (`album\_name`, `no\_of\_songs`, `running\_length`, `album\_genre`, `m&m\_artist\_name`) VALUES ('freenats', '13', '106', 'r&b', 'bill');

INSERT INTO `dbms\_project`.`album` (`album\_name`, `no\_of\_songs`, `running\_length`, `album\_genre`, `m&m\_artist\_name`) VALUES ('innebloom', '7', '34', 'electronic', 'nick');

INSERT INTO `dbms\_project`.`album` (`album\_name`, `no\_of\_songs`, `running\_length`, `album\_genre`, `m&m\_artist\_name`) VALUES ('oasis', '32', '143', 'rock', 'gur');

INSERT INTO `dbms\_project`.`album` (`album\_name`, `no\_of\_songs`, `running\_length`, `album\_genre`, `m&m\_artist\_name`) VALUES ('ventura ', '10', '45', 'r&b', 'sam');

INSERT INTO `dbms\_project`.`album` (`album\_name`, `no\_of\_songs`, `running\_length`, `album\_genre`, `m&m\_artist\_name`) VALUES ('compton', '20', '67', 'hip-hop', 'dre');

INSERT INTO `dbms\_project`.`album` (`album\_name`, `no\_of\_songs`, `running\_length`, `album\_genre`, `m&m\_artist\_name`) VALUES ('detox', '11', '55', 'hip-hop', 'dre');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`) VALUES ('bam', 'gunn', '346', 'paak');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('shiiii', 'roc', '311', 'gibbs', 'alchemist', 'oasis');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('cool', 'jay', '241', 'gibbs', 'hb', 'oasis');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('hugo', '256', 'paak', 'doc', 'innebloom');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('ny', 'fredd', '275', 'west', 'doc', 'innebloom');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('louis', 'butch', '211', 'kdot', 'mb', 'detox');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('polo', 'kim', '123', 'kdot', 'hb', 'ye');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('taste', 'krs', '234', 'sir', 'alchemist', 'ye');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('gold', 'vinesh', '121', 'paak', 'mb', 'mailbu');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('hope', '', '345', 'sir', 'doc', 'malibu');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('flashy', 'masterp', '322', 'west', 'doc', 'ye');

INSERT INTO `dbms\_project`.`song` (`song\_name`, `feature`, `song\_length`, `artist\_name`, `prod\_name`, `album\_name`) VALUES ('mac', '', '420', 'west', 'alchemist', 'innebloom');

UPDATE `dbms\_project`.`song` SET `prod\_name` = 'alchemist', `album\_name` = 'ye' WHERE (`song\_name` = 'bam');

INSERT INTO `dbms\_project`.`producer` (`prod\_name`, `label\_name`) VALUES ('alchemist', 'warner');

INSERT INTO `dbms\_project`.`producer` (`prod\_name`, `label\_name`) VALUES ('hb', 'warner');

INSERT INTO `dbms\_project`.`producer` (`prod\_name`, `label\_name`) VALUES ('mb', 'griselda');

INSERT INTO `dbms\_project`.`producer` (`prod\_name`, `label\_name`) VALUES ('doc', 'aftermath');

INSERT INTO `dbms\_project`.`producer` (`prod\_name`, `label\_name`) VALUES ('kanye', 'griselda');

INSERT INTO `dbms\_project`.`producer` (`prod\_name`, `label\_name`) VALUES ('pharrell', 'ugm');

INSERT INTO `dbms\_project`.`record\_label` (`Label\_name`, `license\_no`, `President`, `Headquarters/location`) VALUES ('ugm', '34556', 'tony', 'new jersey');

INSERT INTO `dbms\_project`.`record\_label` (`Label\_name`, `license\_no`, `President`, `Headquarters/location`) VALUES ('griselda', '24886', 'paulie', 'new york');

INSERT INTO `dbms\_project`.`record\_label` (`Label\_name`, `license\_no`, `President`, `Headquarters/location`) VALUES ('aftermath', '34609', 'chrissy', 'berlin');

INSERT INTO `dbms\_project`.`record\_label` (`Label\_name`, `license\_no`, `President`, `Headquarters/location`) VALUES ('warner', '88923', 'junior', 'los angeles');

INSERT INTO `dbms\_project`.`record\_label` (`Label\_name`, `license\_no`, `President`, `Headquarters/location`) VALUES ('massappeal', '45629', 'bobby', 'bronx');

INSERT INTO `dbms\_project`.`artist` (`Artist\_name`, `Genre`, `label\_name`) VALUES ('paak', 'r&b', 'warner');

INSERT INTO `dbms\_project`.`artist` (`Artist\_name`, `Genre`, `label\_name`) VALUES ('gibbs', 'hip-hop', 'ugm');

INSERT INTO `dbms\_project`.`artist` (`Artist\_name`, `Genre`, `label\_name`) VALUES ('kdot', 'hip-hop', 'ugm');

INSERT INTO `dbms\_project`.`artist` (`Artist\_name`, `Genre`, `label\_name`) VALUES ('west', 'rock', 'aftermath');

INSERT INTO `dbms\_project`.`artist` (`Artist\_name`, `Genre`, `label\_name`) VALUES ('sir', 'house', 'griselda');

INSERT INTO `dbms\_project`.`artist` (`Artist\_name`, `Genre`, `label\_name`) VALUES ('nas', 'coke-rap', 'griselda');

INSERT INTO `dbms\_project`.`streaming\_service` (`str\_srv\_name`) VALUES ('spotify');

INSERT INTO `dbms\_project`.`streaming\_service` (`str\_srv\_name`) VALUES ('apple music');

INSERT INTO `dbms\_project`.`streaming\_service` (`str\_srv\_name`) VALUES ('pandora');

INSERT INTO `dbms\_project`.`streaming\_service` (`str\_srv\_name`) VALUES ('bandcamp');

INSERT INTO `dbms\_project`.`streaming\_service` (`str\_srv\_name`) VALUES ('soundcloud');

INSERT INTO `dbms\_project`.`distributor` (`distributor\_name`, `format`, `website`, `label\_name`) VALUES ('complex', 'flac', 'complex.com', 'ugm');

INSERT INTO `dbms\_project`.`distributor` (`distributor\_name`, `format`, `website`, `label\_name`) VALUES ('npr', 'alac', 'nprmusic.com', 'ugm');

INSERT INTO `dbms\_project`.`distributor` (`distributor\_name`, `format`, `website`, `label\_name`) VALUES ('sopranos', 'alac', 'sopranos.to', 'griselda');

INSERT INTO `dbms\_project`.`distributor` (`distributor\_name`, `format`, `website`, `label\_name`) VALUES ('dimeo', 'wav', 'dimeo.com', 'warner');

INSERT INTO `dbms\_project`.`distributor` (`distributor\_name`, `format`, `website`, `label\_name`) VALUES ('shady', 'wav', 'shadyrecords.to', 'griselda');

INSERT INTO `dbms\_project`.`distributor` (`distributor\_name`, `format`, `website`, `label\_name`) VALUES ('massappeal', 'wav', 'ma.com', 'griselda');

INSERT INTO `dbms\_project`.`distributor` (`distributor\_name`, `format`, `website`, `label\_name`) VALUES ('stonesthrow', 'flac', 'stones.com', 'warner');

INSERT INTO `dbms\_project`.`mixing&mastering\_artist` (`m&m\_artist\_name`, `softwares`, `label\_name`) VALUES ('sam', 'fruityloops', 'griselda');

INSERT INTO `dbms\_project`.`mixing&mastering\_artist` (`m&m\_artist\_name`, `softwares`, `label\_name`) VALUES ('nick', 'cubase', 'griselda');

INSERT INTO `dbms\_project`.`mixing&mastering\_artist` (`m&m\_artist\_name`, `softwares`, `label\_name`) VALUES ('gur', 'cubase', 'ugm');

INSERT INTO `dbms\_project`.`mixing&mastering\_artist` (`m&m\_artist\_name`, `softwares`, `label\_name`) VALUES ('dre', 'yamaha', 'ugm');

INSERT INTO `dbms\_project`.`mixing&mastering\_artist` (`m&m\_artist\_name`, `softwares`, `label\_name`) VALUES ('bill', 'steinberg', 'warner');

INSERT INTO `dbms\_project`.`distributes` (`distributor\_name`, `album\_name`) VALUES ('complex', 'malibu');

INSERT INTO `dbms\_project`.`distributes` (`distributor\_name`, `album\_name`) VALUES ('sopranos', 'detox');

INSERT INTO `dbms\_project`.`distributes` (`distributor\_name`, `album\_name`) VALUES ('npr', 'ye');

INSERT INTO `dbms\_project`.`distributes` (`distributor\_name`, `album\_name`) VALUES ('npr', 'ventura');

INSERT INTO `dbms\_project`.`distributes` (`distributor\_name`, `album\_name`) VALUES ('complex', 'oasis');

# FOREIGN KEY

ALTER TABLE `dbms\_project`.`artist`

CHANGE COLUMN `Artist\_name` `Artist\_name` CHAR(45) NOT NULL ,

ADD INDEX `label\_name\_idx` (`label\_name` ASC) VISIBLE;

alter table producer add foreign key(label\_name) references record\_label(label\_name);

alter table artist add foreign key(label\_name) references record\_label(label\_name);

alter table song add constraint fk1 FOREIGN KEY(prod\_name) REFERENCES producer(prod\_name);

alter table album add foreign key(mnm\_artist\_name) references mixingnmastering\_artist(mnm\_artist\_name);

alter table mixingnmastering\_artist add foreign key(label\_name) references record\_label(label\_name);

alter table distributor add foreign key(label\_name) references record\_label(label\_name) ON DELETE CASCADE;

alter table distributes add foreign key(distributor\_name) references distributor(distributor\_name) ON DELETE CASCADE;

alter table distributes add foreign key(album\_name) references album(album\_name);

alter table streams add foreign key(str\_srv\_name) references streaming\_service(str\_srv\_name);

alter table streams add foreign key(album\_name) references album(album\_name);

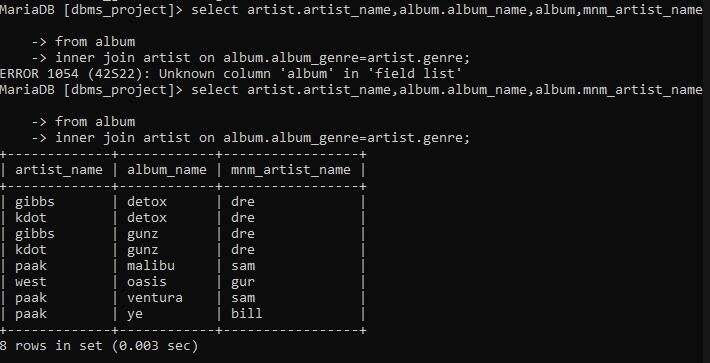
# Join Queries

1)Show the artist’s name, the album’s name and the mixing&mastering artist’s name where they have the same genre.

> select artist.artist\_name,album.album\_name,album.mnm\_artist\_name

-> from album

-> inner join artist on album.album\_genre=artist.genre;

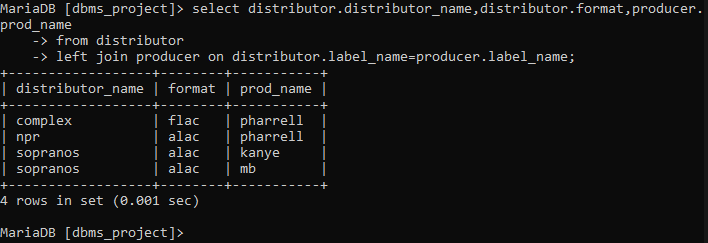


2)Show the distributor names of producers who are affiliated to the same music label and the format of music they exported in.

-> select distributor.distributor\_name,distributor.format,producer.prod\_name

-> from distributor

-> left join producer on distributor.label\_name=producer.label\_name;

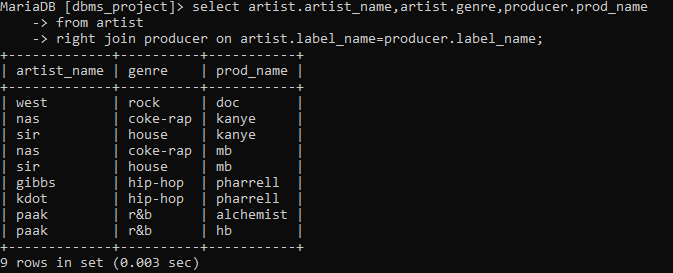


3)Showcase the artist-producer collaborations of the same label and the genre of the music they have worked on.

->select artist.artist\_name,artist.genre,producer.prod\_name

-> from artist

-> right join producer on artist.label\_name=producer.label\_name;

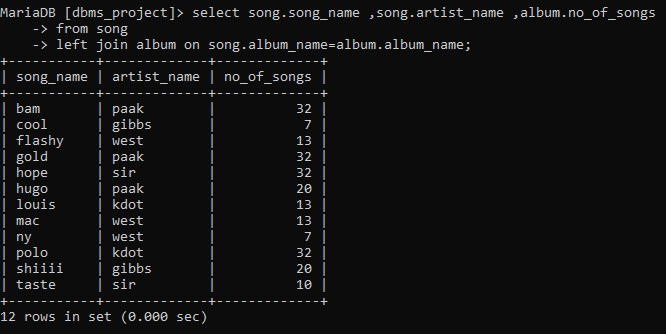


4)Show the songs and the album it is from, of an artist and the number of songs in the respective album.

->select song.song\_name ,song.artist\_name ,album.no\_of\_songs

-> from song

-> left join album on song.album\_name=album.album\_name;



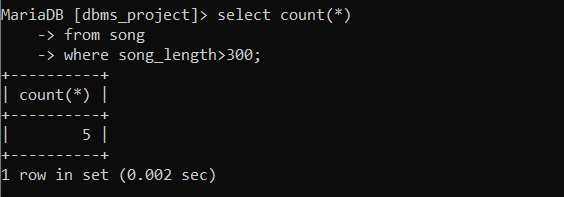
# AGGREGATE FUNCTIONS:

Show the number of songs with song length greater than 300 seconds

->select count(\*)

-> from song

-> where song\_length>300;

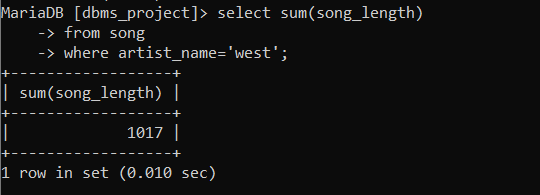


Show the total no\_of\_song’s length of the artist name west

MariaDB [dbms\_project]> select sum(song\_length)

-> from song

-> where artist\_name='west';

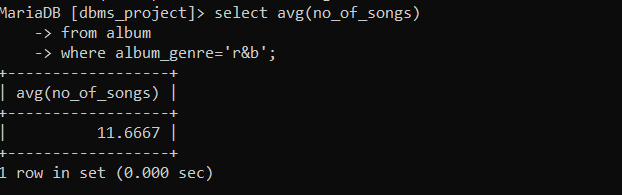


Show the average number of songs in an R&B genre album

->select avg(no\_of\_songs)

-> from album

-> where album\_genre='r&b';



There are 12 songs, on an average ,for the R&B genre.

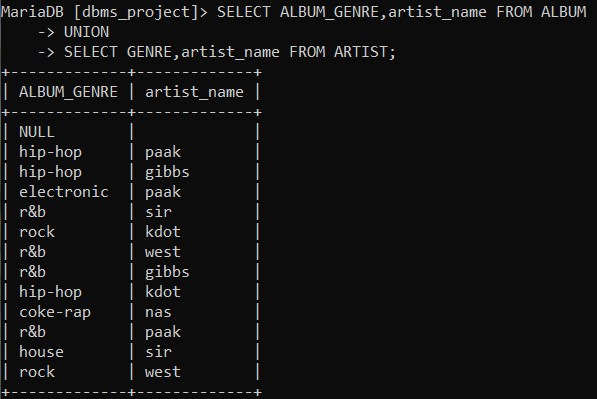
# SET OPERATIONS:

Show all the artist’s genres of the albums they have worked on

SELECT ALBUM\_GENRE,artist\_name FROM ALBUM

-> UNION

-> SELECT GENRE,artist\_name from artists;

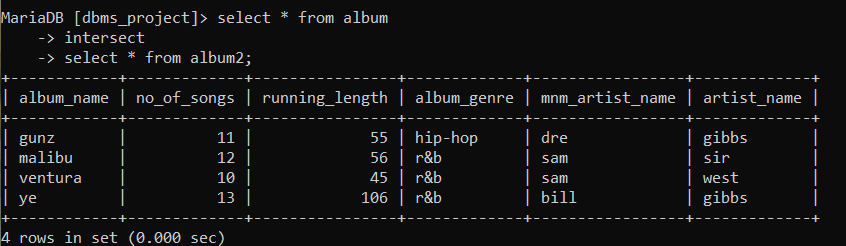


2) Show the common albums in Volume1(table name: album) and Volume2(Table name:album2)

select \* from album

-> intersect

-> select \* from album2;

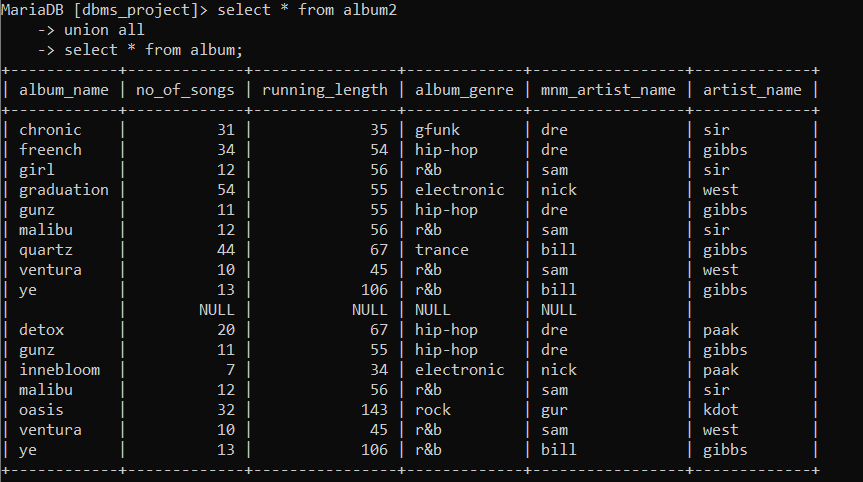


3)List out the albums that are in volume2 and volume1 wit the duplicates

MariaDB [dbms\_project]> select \* from album2

-> union all

-> select \* from album;



# STORED FUNCTION:

# Create a stored function for classifying a song as “short”,”medium” and “long”:

CREATE FUNCTION howlong(song\_length int)

-> returns varchar(20)

-> DETERMINISTIC

-> BEGIN

-> DECLARE thislong VARCHAR(20);

-> IF SONG\_LENGTH>300 THEN

-> SET thislong='long';

-> ELSEIF (song\_length<=300 AND song\_length>=200) THEN

-> SET thislong='medium';

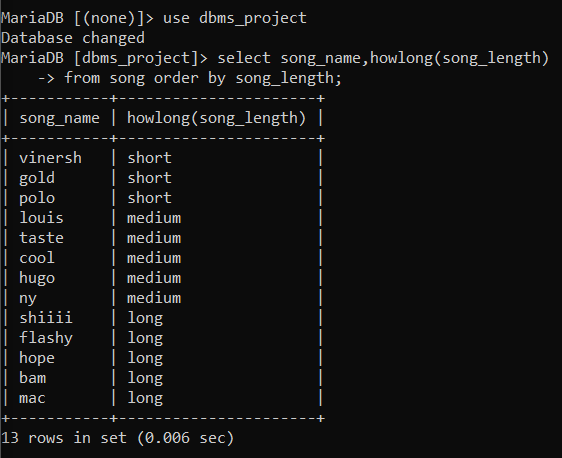
-> elseif song\_length<200 THEN

-> SET thislong='short';

-> END IF;

-> RETURN (thislong);

-> END$$



# STORED PROCEDURE:

Display the maximum number of songs in a volume of albums

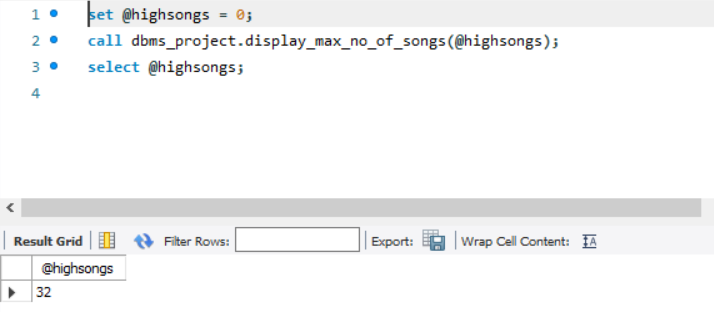
DELIMITER &&

MariaDB [dbms\_project]> CREATE PROCEDURE display\_max\_no\_of\_songs (OUT highsongs INT)

-> BEGIN

-> SELECT MAX(no\_of\_songs) INTO highsongs FROM album;

-> END &&



# TRIGGER

# Increment the number of songs field in album after a song has been added to the song table referencing to an album

DROP TRIGGER IF EXISTS `dbms\_project`.`song\_AFTER\_INSERT`;

DELIMITER $$

USE `dbms\_project`$$

CREATE DEFINER = CURRENT\_USER TRIGGER `dbms\_project`.`song\_AFTER\_INSERT` AFTER INSERT ON `song` FOR EACH ROW

BEGIN

update album

set no\_of\_songs=no\_of\_songs+1

where album\_name=NEW.album\_name;

END$$

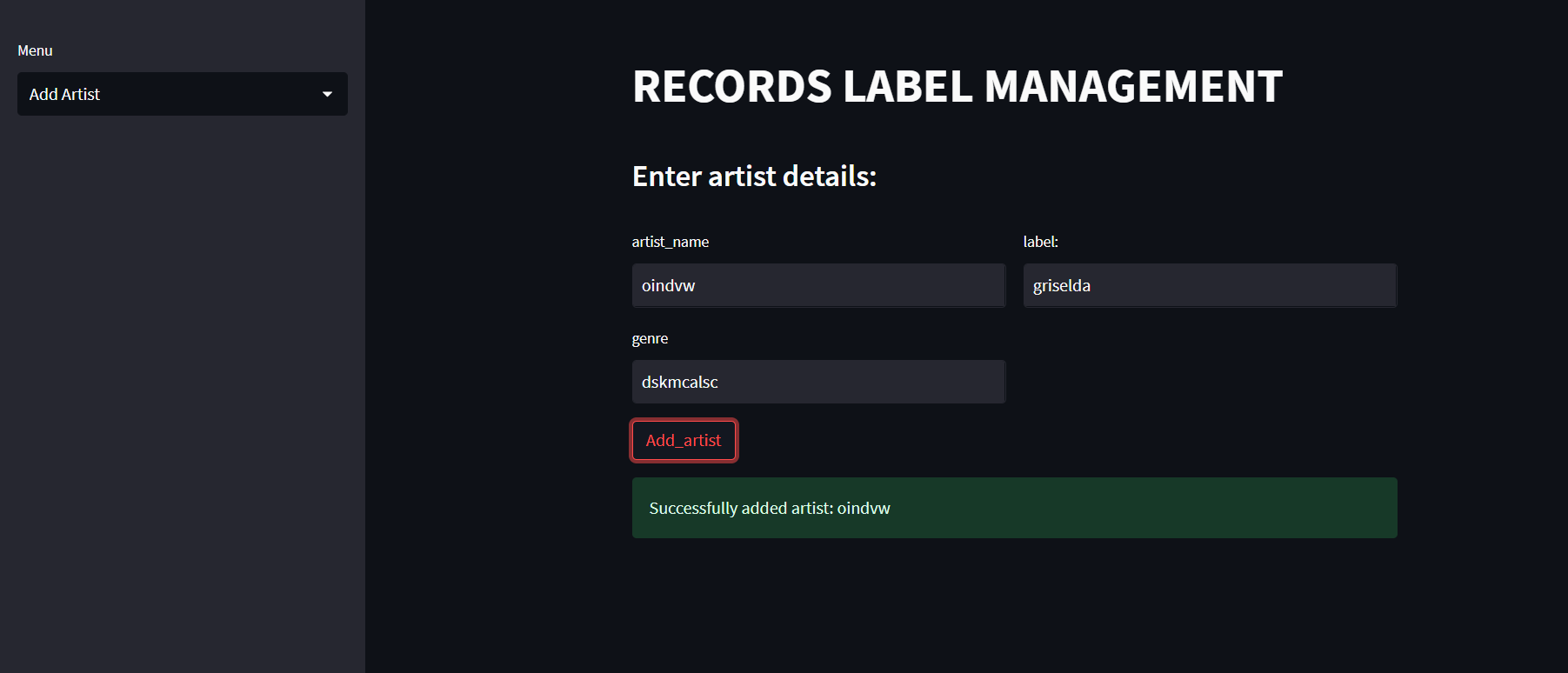
DELIMITER ;

# SIMPLE FRONTEND IMPLEMENTATION

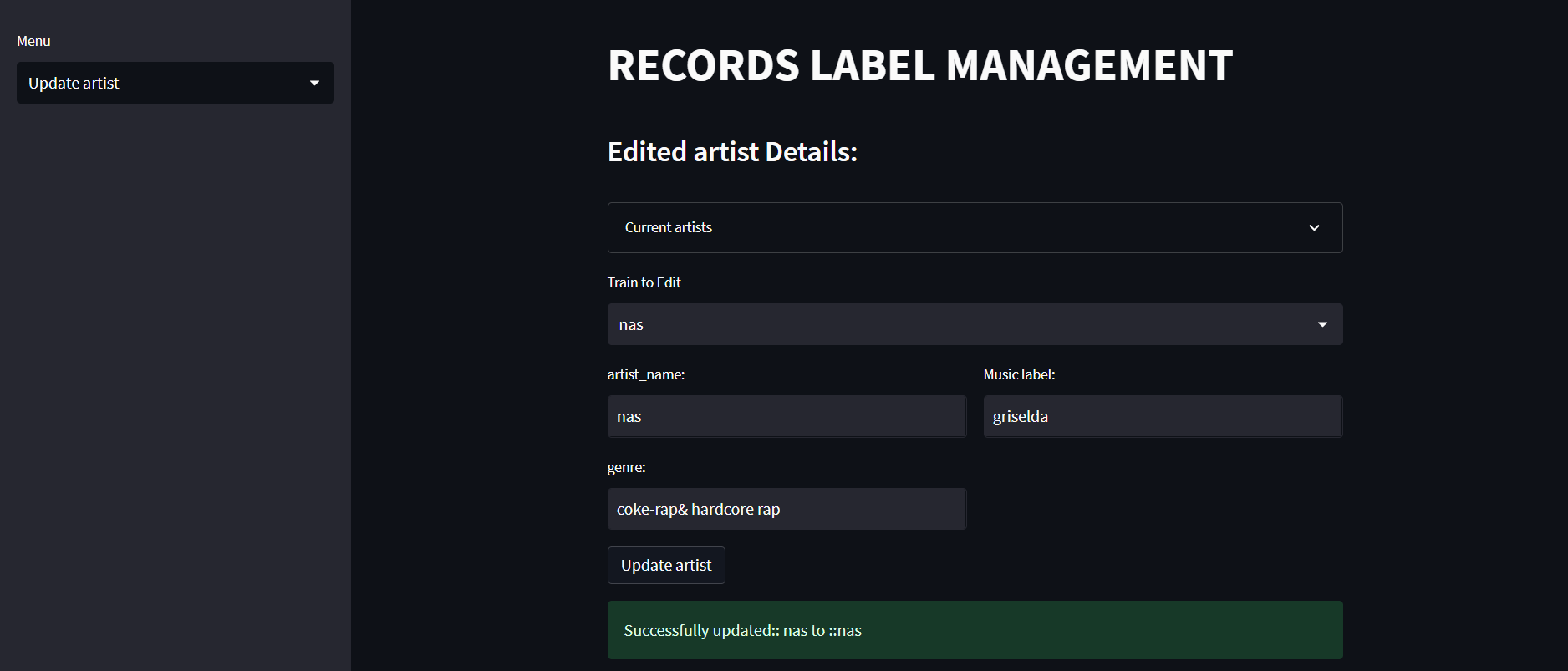
Streamlit has been used to display the various operations on the tables of the database

Provided below are the screenshots of the requirements of the project:

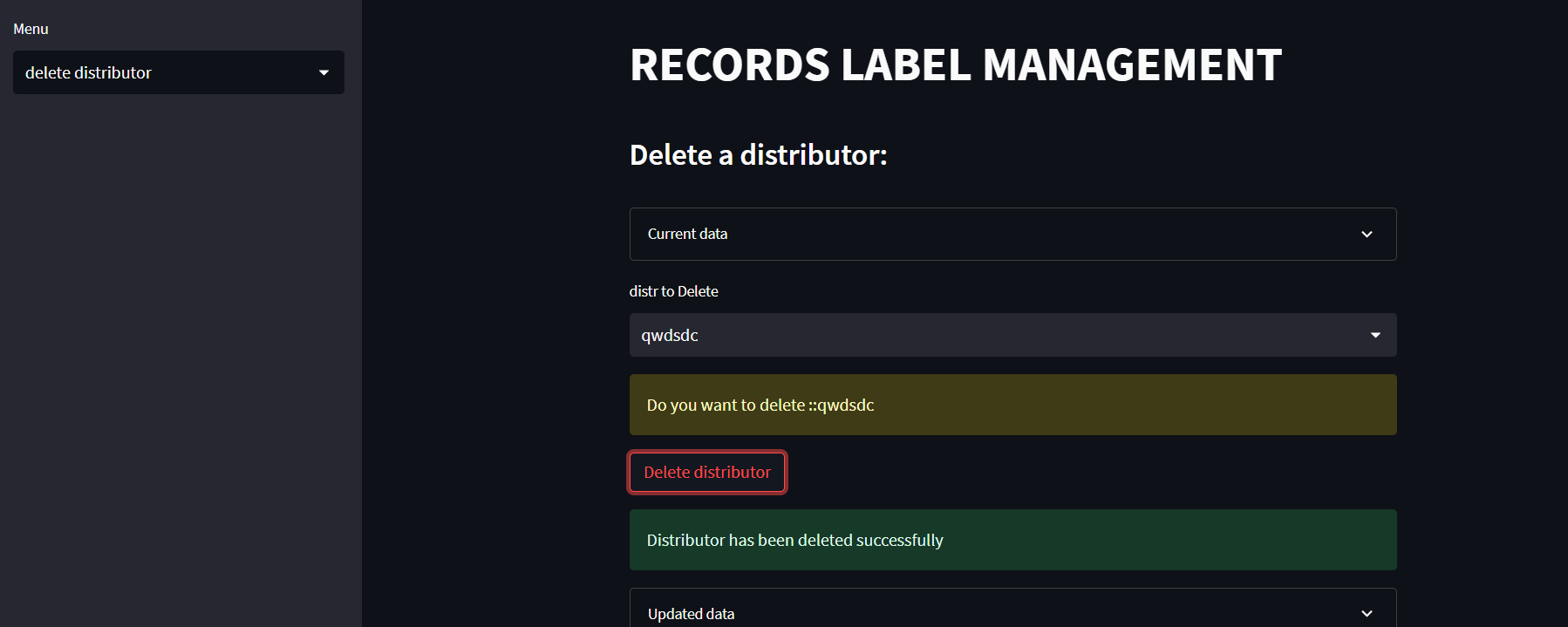
# Addition of a tuple:



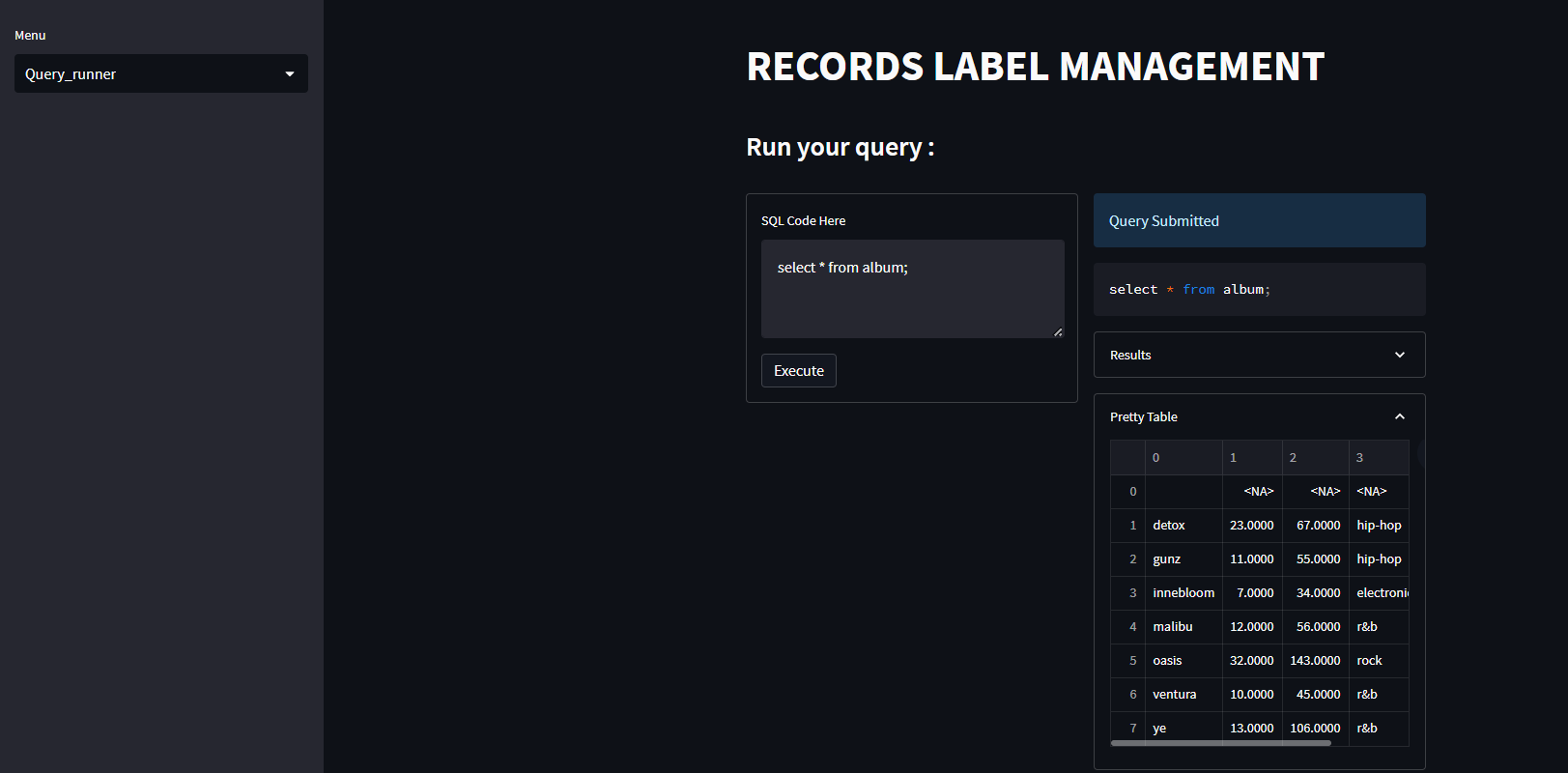
# Updation of a tuple :



# Deletion of a tuple:



# Query runner:



# MODIFICATION:

Create a stored procedure for showing the number of distributors an album has from the distributes table.

CREATE DEFINER=`root`@`localhost` PROCEDURE `distribute\_no`()

BEGIN

select album\_name, count(\*) as num\_distributors from distributes group by album\_name;

END

