DDL STATEMENTS
Group 07

1.user

```
CREATE TABLE users(
user_name VARCHAR,
user_id SERIAL,
email VARCHAR,
password VARCHAR(8),
PRIMARY KEY(user_id)
);
```

2.creator

```
CREATE TABLE creator(
creator_name VARCHAR,
creator_id SERIAL,
user_id INT,
PRIMARY KEY(creator_id)
);
```

3.listener

```
CREATE TABLE listener(
listener_name VARCHAR,
listener_id SERIAL,
user_id INT,
subscription_id INT,
PRIMARY KEY(listener_id)
FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

4.create_song

```
CREATE TABLE create_song(
creator_id INT,
song_id INT,
role_type VARCHAR
PRIMARY KEY(song_id),
FOREIGN KEY(song_id) REFERENCES song(song_id),
FOREIGN KEY(creator_id) REFERENCES creator(creator_id)
);
```

5.singer

```
CREATE TABLE singer(
singer_id SERIAL,
singer_name VARCHAR,
PRIMARY KEY(singer_id)
);
```

6.composer

```
CREATE TABLE composer(
Composer_id  SERIAL,
composer_name VARCHAR
PRIMARY KEY(Composer_id)
);
```

7.playlist

```
CREATE TABLE playlist(
user_id INT,
playlist_name VARCHAR(50),
playlist_id SERIAL,
duration FLOAT,
PRIMARY KEY(playlist_id),
FOREIGN KEY(user_id) REFERENCES users(user_id)
        );
```

8.song_on_platform

```
CREATE TABLE song_on_platform(
song_id INT,
platform_id INT,
platform_link VARCHAR,
PRIMARY KEY(song_id,platform_id),
FOREIGN KEY(platform_id) REFERENCES platform(platform_id),
FOREIGN KEY(song_id) REFERENCES song(song_id)
);
```

9.platform

```
CREATE TABLE platform(
platform_id SERIAL,
platform_name VARCHAR,
PRIMARY KEY(platform_id)
);
```

10.song

```
CREATE TABLE song(
song_id INT,
SongName VARCHAR,
artist_name VARCHAR,
--lyrics VARCHAR(200),
album_name VARCHAR(20),
duration FLOAT,
release_date VARCHAR,
rating INT,
count_of_listener INT,
genre VARCHAR,
-- movie_id INT,
PRIMARY KEY(song_id),
-- FOREIGN KEY(movie_id) REFERENCES movie(movie_id)
);
```

11.subscription

```
CREATE TABLE subscription(
user_id INT,
price INT,
subscription_type VARCHAR(20),
PRIMARY KEY(user_id )
FOREIGN KEY(user_id) REFERENCES users(user_id)
);
```

12.recommendation

```
CREATE TABLE recommendation(
recommendation_rank INT,
song_id INT PRIMARY KEY,
FOREIGN KEY(song_id) REFERENCES song(song_id),
);
```

13.rates

```
CREATE TABLE rates(
song_id INT,
listener_id INT,
rating INT,
PRIMARY KEY(song_id,listener_id),
FOREIGN KEY(song_id) REFERENCES song(song_id),
FOREIGN KEY(listener_id) REFERENCES listener(listener_id)
);
```

14.movie

```
CREATE TABLE movie(
movie_name VARCHAR,
movie_id INT,
song_id INT,
PRIMARY KEY(movie_id,song_id),
FOREIGN KEY(song_id) REFERENCES song(song_id)
);
```

15. share

```
CREATE TABLE share(
user_id INT,
playlist_id INT,
Access_type VARCHAR(20),
PRIMARY KEY(user_id,playlist_id),
FOREIGN KEY(user_id) REFERENCES users(user_id),
FOREIGN KEY(playlist_id) REFERENCES playlist(playlist_id)
);
```

16. playlist_Song

```
CREATE TABLE playlist_song(
song_id INT,
playlist_id,
PRIMARY KEY(playlist_id, song_id),
FOREIGN KEY(song_id) REFERENCES song(song_id),
FOREIGN KEY(playlist_id) REFERENCES playlist(playlist_id)
);
```

(Not included)
owns

```
CREATE TABLE owns(
playlist_id INT,
user_id INT,
-- Ownership_type CHAR,
PRIMARY KEY(playlist_id, user_id),
FOREIGN KEY(user_id) REFERENCES users(user_id),
FOREIGN KEY(playlist_id) REFERENCES playlist(playlist_id)
);
```

https://www.kaggle.com/code/rohitsinghbani/music-recommendation/input