

Names: Maha Kanakala, Nikhila Sundar

## Database Schema

---

```
CREATE DATABASE mlk224_DB;
```

```
USE mlk224_DB;
```

```
CREATE TABLE artist(  
  `artist_name` VARCHAR(50) PRIMARY KEY NOT NULL UNIQUE  
);
```

```
CREATE TABLE album(  
  `album_name` VARCHAR(50) NOT NULL,  
  `release_date` DATETIME,  
  `artist_name` VARCHAR(50),  
  PRIMARY KEY (`album_name`, `artist_name`),  
  FOREIGN KEY (`artist_name`) REFERENCES artist(`artist_name`)  
);
```

```
CREATE TABLE song(  
  `song_title` VARCHAR(50) NOT NULL,  
  `artist_name` VARCHAR(50) NOT NULL,  
  `album_name` VARCHAR(50),  
  `song_creation_time` DATETIME,  
  PRIMARY KEY (`song_title`, `artist_name`),  
  FOREIGN KEY (`artist_name`) REFERENCES artist(`artist_name`),  
  FOREIGN KEY (`album_name`) REFERENCES album(`album_name`)  
);
```

```
CREATE TABLE user(  
  `username` VARCHAR(50) PRIMARY KEY NOT NULL UNIQUE  
);
```

```
CREATE TABLE playlist(  
  `playlist_title` VARCHAR(50) NOT NULL,  
  `time_created` DATETIME NOT NULL,  
  `username` VARCHAR(50),  
  PRIMARY KEY (`playlist_title`, `time_created`),  
  FOREIGN KEY (`username`) REFERENCES user(`username`)  
);
```

```
CREATE TABLE playlist_song(  
  `playlist_title` VARCHAR(50) NOT NULL,  
  `song_title` VARCHAR(50) NOT NULL,  
  PRIMARY KEY (`playlist_title`, `song_title`),  
  FOREIGN KEY (`playlist_title`) REFERENCES playlist(`playlist_title`),  
  FOREIGN KEY (`song_title`) REFERENCES song(`song_title`)
```

```
);
```

```
CREATE TABLE rating(  
  `username` VARCHAR(50) NOT NULL,  
  `song_title` VARCHAR(50) NOT NULL,  
  `album_name` VARCHAR(50),  
  `rating` TINYINT CHECK (rating >= 1 AND rating <= 5),  
  `rating_date` DATETIME,  
  PRIMARY KEY (`username`, `song_title`),  
  FOREIGN KEY (`username`) REFERENCES user(`username`),  
  FOREIGN KEY (`song_title`) REFERENCES song(`song_title`),  
  FOREIGN KEY (`album_name`) REFERENCES album(`album_name`)  
);
```

```
CREATE TABLE genre (  
  `song_title` VARCHAR(50) NOT NULL,  
  `genre` VARCHAR(10),  
  PRIMARY KEY (`song_title`, `genre`),  
  FOREIGN KEY (`song_title`) REFERENCES song(`song_title`)  
);
```

## Queries

---

1.

```
SELECT genre, count(*) as number_of_songs  
FROM genre  
GROUP BY genre  
ORDER BY number_of_songs desc  
LIMIT 3;
```

2.

```
SELECT artist_name  
FROM artist  
WHERE artist_name IN (  
  SELECT artist_name FROM song WHERE album_name IS NOT NULL  
) AND artist_name IN (  
  SELECT artist_name FROM song WHERE album_name IS NULL  
);
```

3.

```
SELECT album.album_name, AVG(rating.rating) AS average_user_rating
FROM rating
JOIN song ON song.song_title = rating.song_title
JOIN album ON album.album_name = rating.album_name
WHERE YEAR(rating.rating_date) BETWEEN 1990 AND 1999
GROUP BY album.album_name
ORDER BY average_user_rating DESC, album.album_name ASC
LIMIT 10;
```

4.

```
SELECT g.genre AS genre_name, COUNT(r.rating) AS number_of_song_ratings
FROM genre g
JOIN song s ON g.song_title = s.song_title
JOIN rating r ON s.song_title = r.song_title
WHERE YEAR(r.rating_date) BETWEEN 1991 AND 1995
GROUP BY g.genre
ORDER BY number_of_song_ratings DESC
LIMIT 3;
```

5.

```
SELECT p.username, p.playlist_title, AVG(avg_song_rating) AS average_song_rating
FROM playlist p
JOIN playlist_song ps ON p.playlist_title = ps.playlist_title
JOIN (
    SELECT song_title, AVG(rating) as avg_song_rating
    FROM rating
    GROUP BY song_title
) r ON ps.song_title = r.song_title
GROUP BY p.username, p.playlist_title
HAVING AVG(avg_song_rating) >= 4.0;
```

6.

```
SELECT username, COUNT(rating) AS number_of_ratings
FROM rating
GROUP BY username
ORDER BY number_of_ratings DESC
LIMIT 5;
```

7.

```
SELECT artist_name, COUNT(song_title) as number_of_songs
FROM song
where YEAR(song_creation_time) between 1990 and 2010
GROUP BY artist_name
ORDER BY number_of_songs DESC
LIMIT 10;
```

8.

```
SELECT song_title, COUNT(playlist_title) as number_of_playlists
FROM playlist_song
GROUP BY song_title
ORDER BY number_of_playlists DESC, song_title ASC
LIMIT 10;
```

9.

```
SELECT song.song_title, song.artist_name, COUNT(rating.rating) as number_of_ratings
FROM song
JOIN rating ON song.song_title = rating.song_title
WHERE song.album_name IS NULL
GROUP BY song_title
ORDER BY number_of_ratings DESC
LIMIT 20;
```

10.

```
SELECT artist_name
FROM artist
WHERE artist_name NOT IN (
    SELECT DISTINCT artist_name
    FROM album
    WHERE YEAR(release_date) > 1993
);
```