SQL PROJECT

SPOTIFY DATABASE MANAGEMENT REPOSITORY

NAME: NIKESH VAREKAR

COURSE: MASTER IN DATA SCIENCE AND ANALYTICS WITH AI

ABOUT DATASET:

The Spotify dataset can be understood as a structured collection of data that reflects the various elements and interactions within the Spotify music streaming service. Below is a detailed breakdown of what a typical Spotify dataset might contain. Users Table, Artists Table, Songs Table, Playlists Table, StreamingHistory Table, Album table, playlistsongs table.

AVAILABLE TABLES:

Users Table

To store information about users on the platform.

Artists Table

To store information about music artists.

Albums Table

To store information about music albums.

Songs Table

To store information about individual songs.

Playlists Table

To store information about user-created playlists.

PlaylistSongs Table

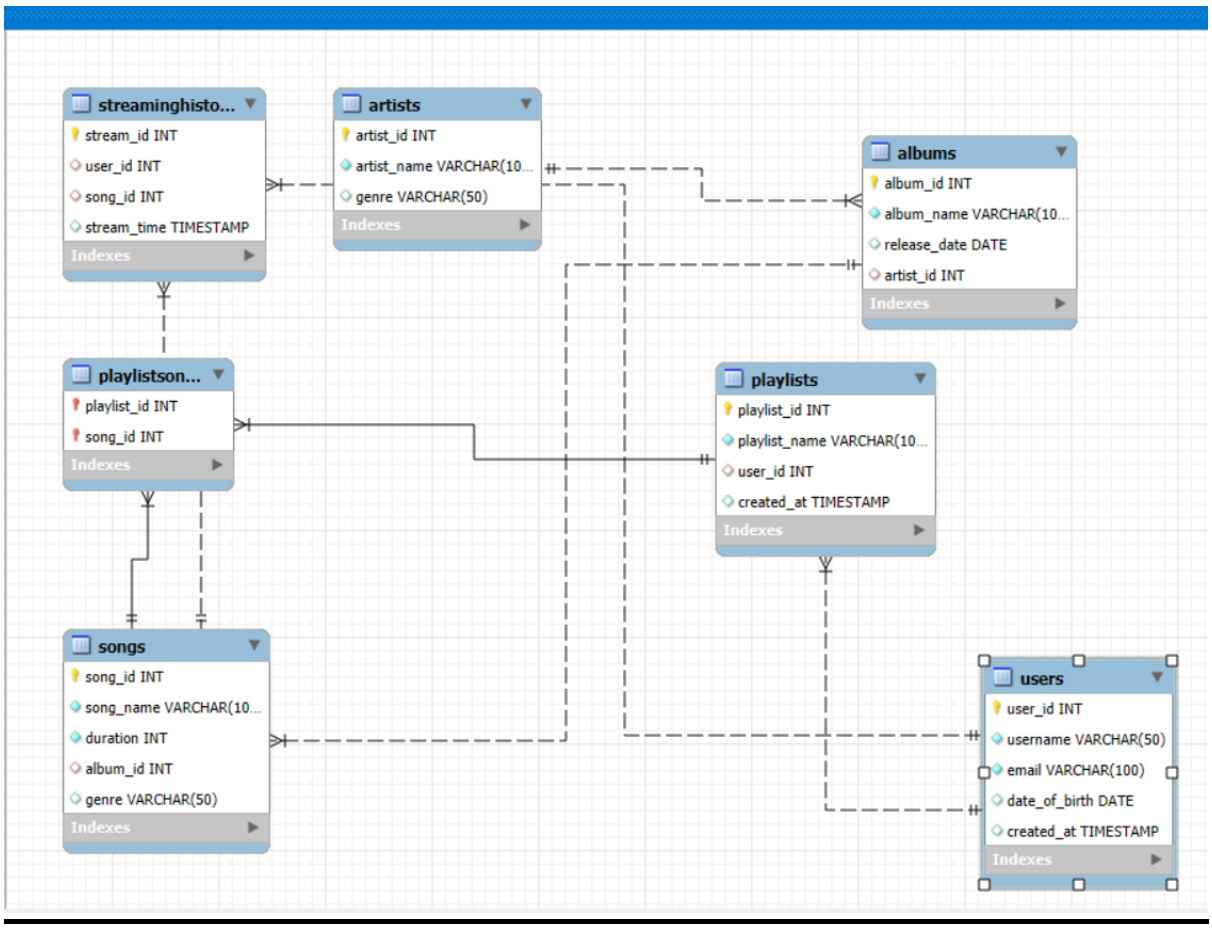
To represent the many-to-many relationship between playlists and songs.

StreamingHistory Table

To log user streaming activities, such as when a user plays a song.

DATASET FROM: SAMPLE DATASET FROM CHATGPT

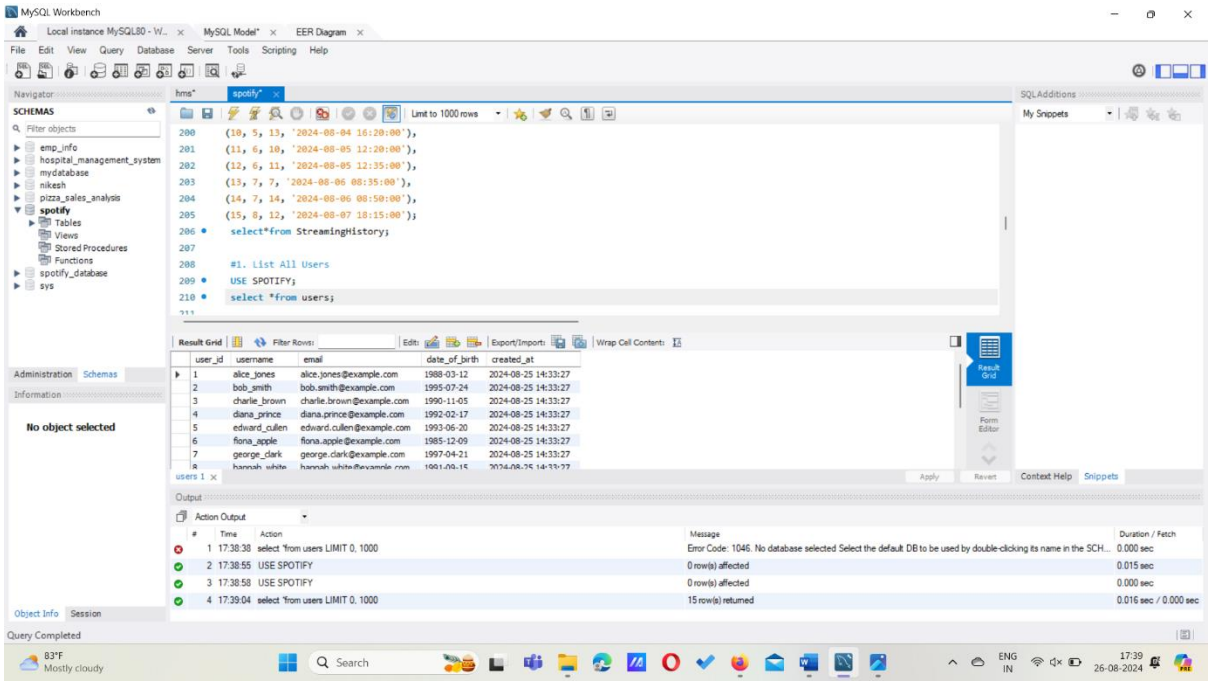
ER DIAGRAM:


streaminghisto...
stream_id INT
user_id INT
song_id INT
stream_time TIMESTAMP
Indexes
artists
artist_id INT
artist_name VARCHAR(10...
genre VARCHAR(50)
Indexes
albums
album_id INT
album_name VARCHAR(10...
release_date DATE
artist_id INT
Indexes
playlistson...
playlist_id INT
song_id INT
Indexes
playlists
playlist_id INT
playlist_name VARCHAR(10...
user_id INT
created_at TIMESTAMP
Indexes
songs
song_id INT
song_name VARCHAR(10...
duration INT
album_id INT
genre VARCHAR(50)
Indexes
users
user_id INT
username VARCHAR(50)
email VARCHAR(100)
date_of_birth DATE
created_at TIMESTAMP
Indexes

QUERIES:

#1. List All Users

select *from users;

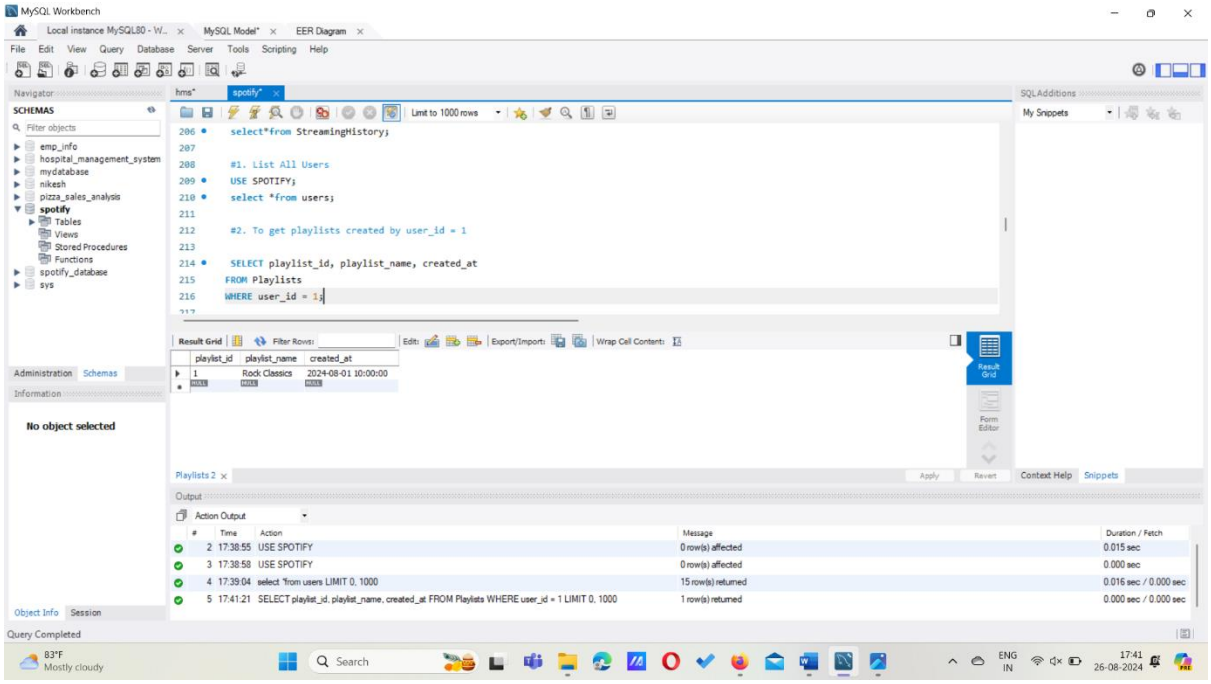


#2. To get playlists created by user_id = 1

SELECT playlist_id, playlist_name, created_at

FROM Playlists

WHERE user_id = 1;

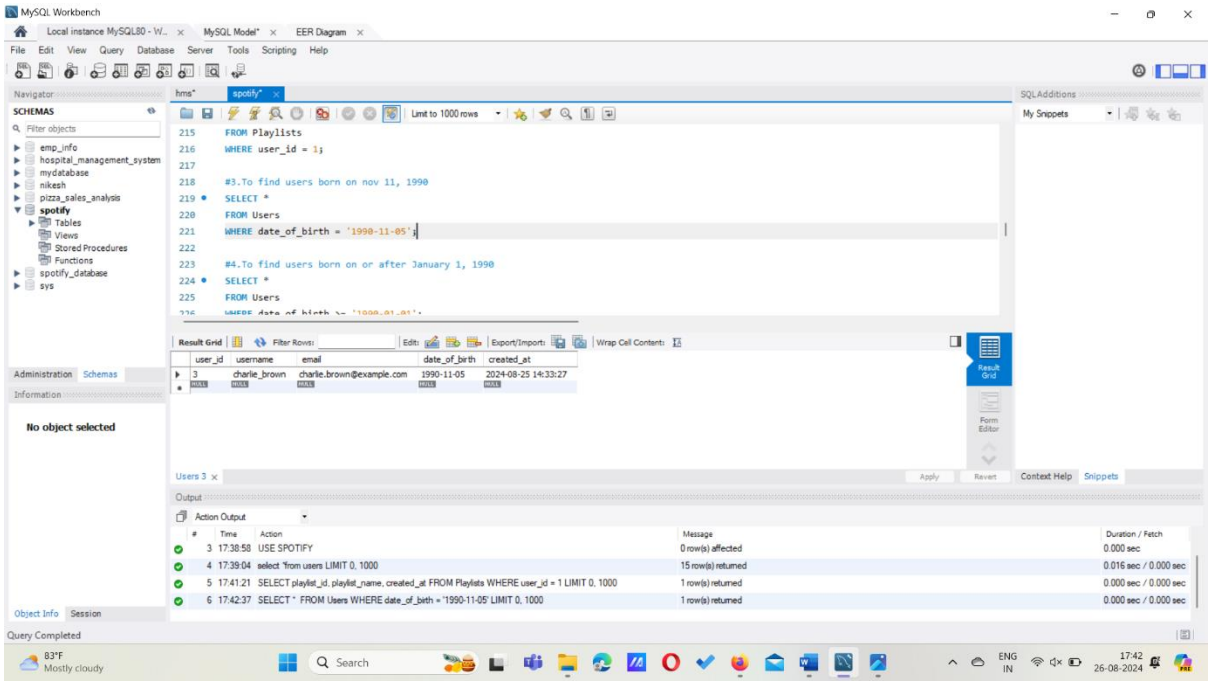


#3.To find users born on nov 11, 1990

SELECT *

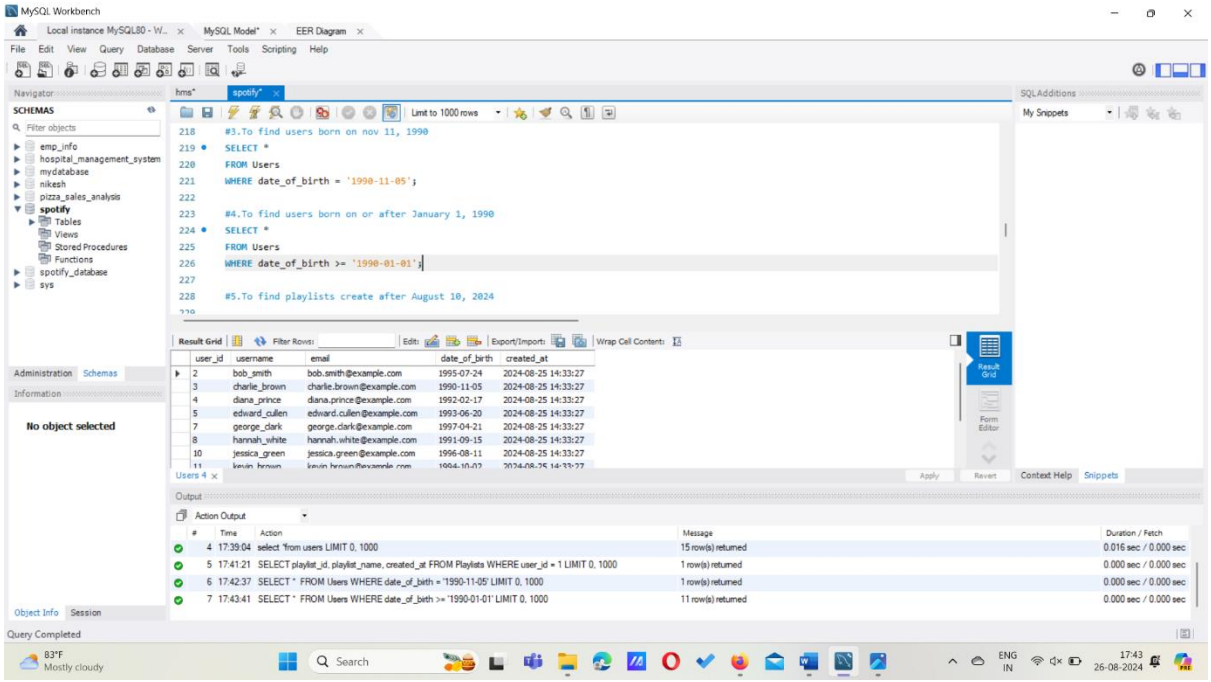
FROM Users

WHERE date_of_birth = '1990-11-05';



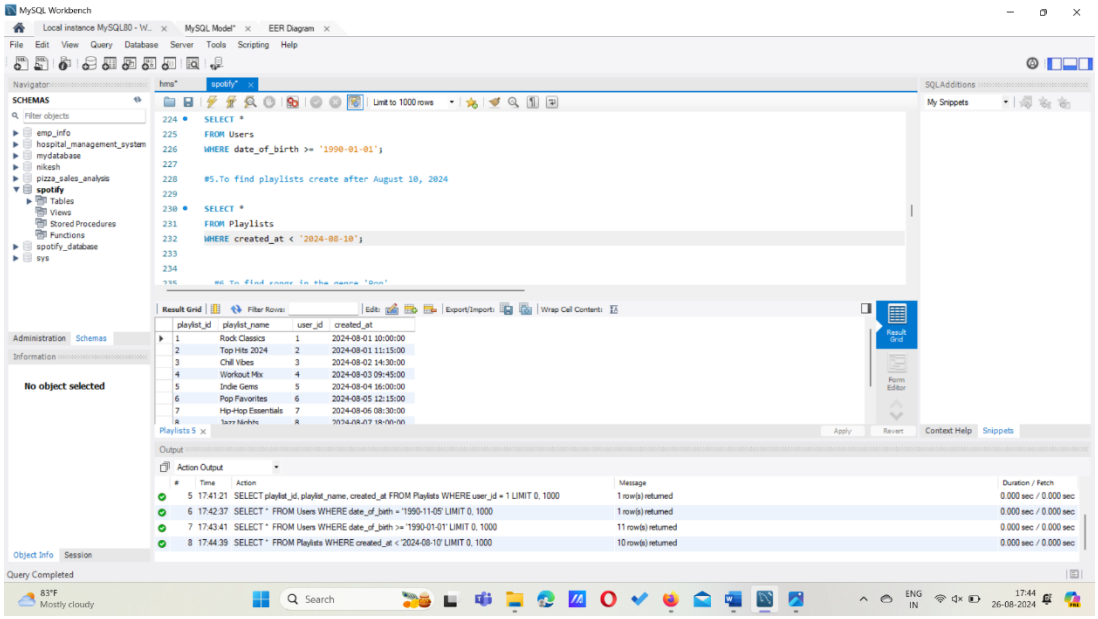
#4.To find users born on or after January 1, 1990

```
SELECT *
FROM Users
WHERE date_of_birth >= '1990-01-01';
```



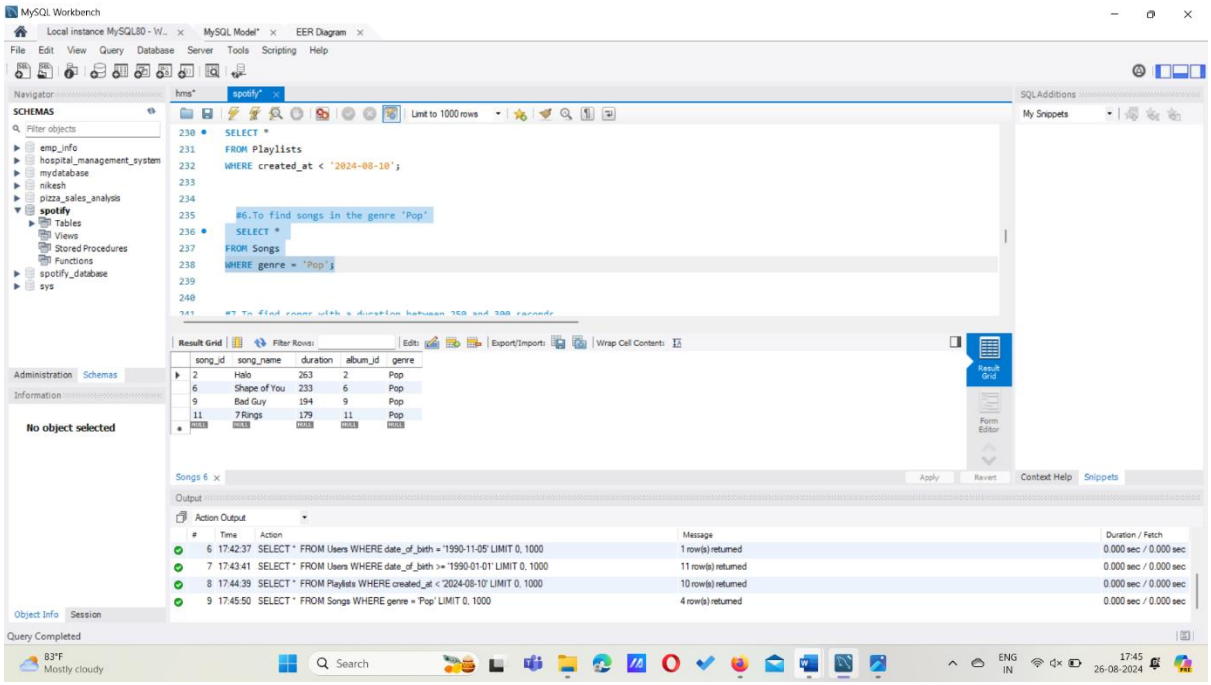
#5.To find playlists create after August 10, 2024

```
SELECT *
FROM Playlists
WHERE created_at < '2024-08-10';
```



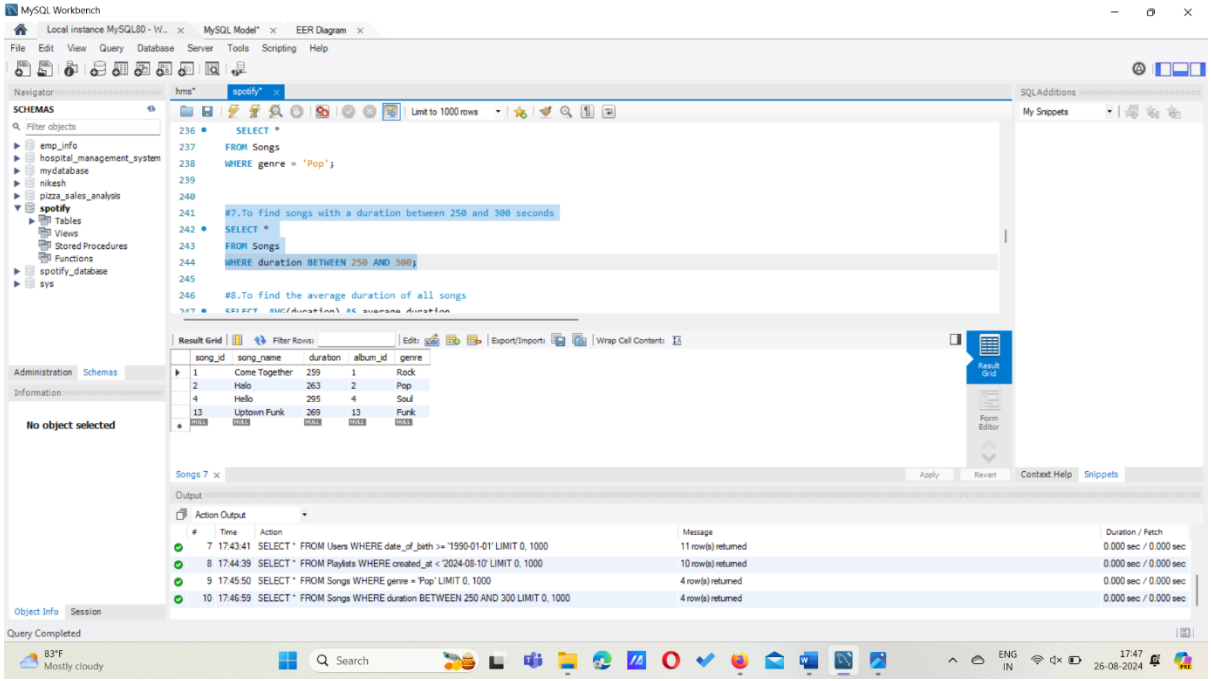
#6.To find songs in the genre 'Pop'

```
 SELECT *
FROM Songs
WHERE genre = 'Pop';
```



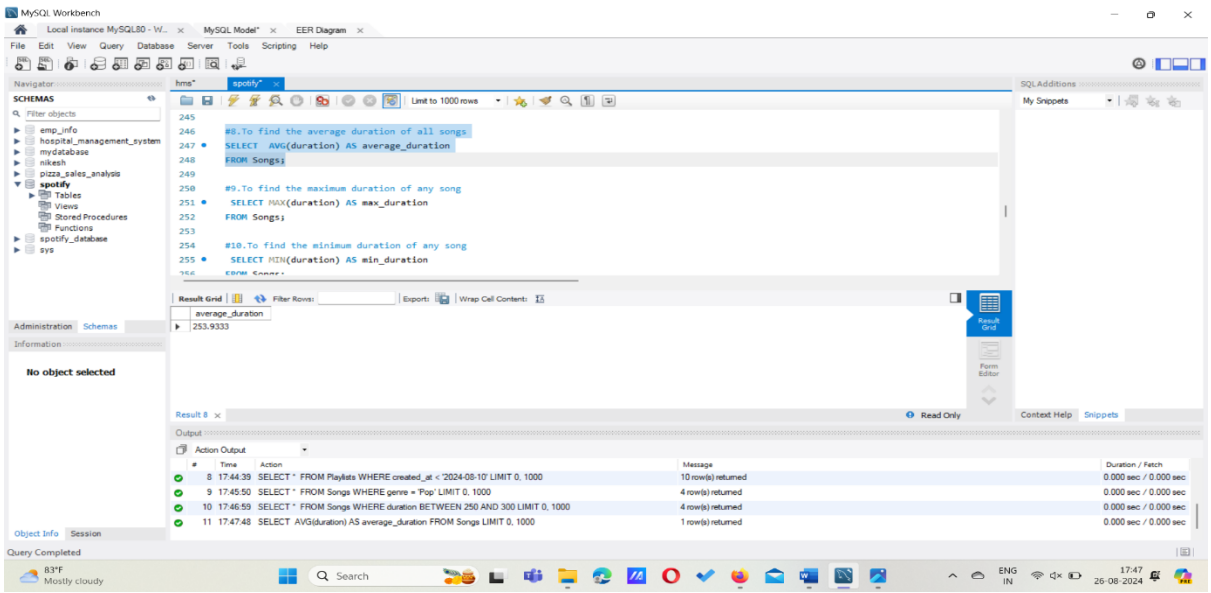
#7.To find songs with a duration between 250 and 300 seconds

```
SELECT *
FROM Songs
WHERE duration BETWEEN 250 AND 300;
```



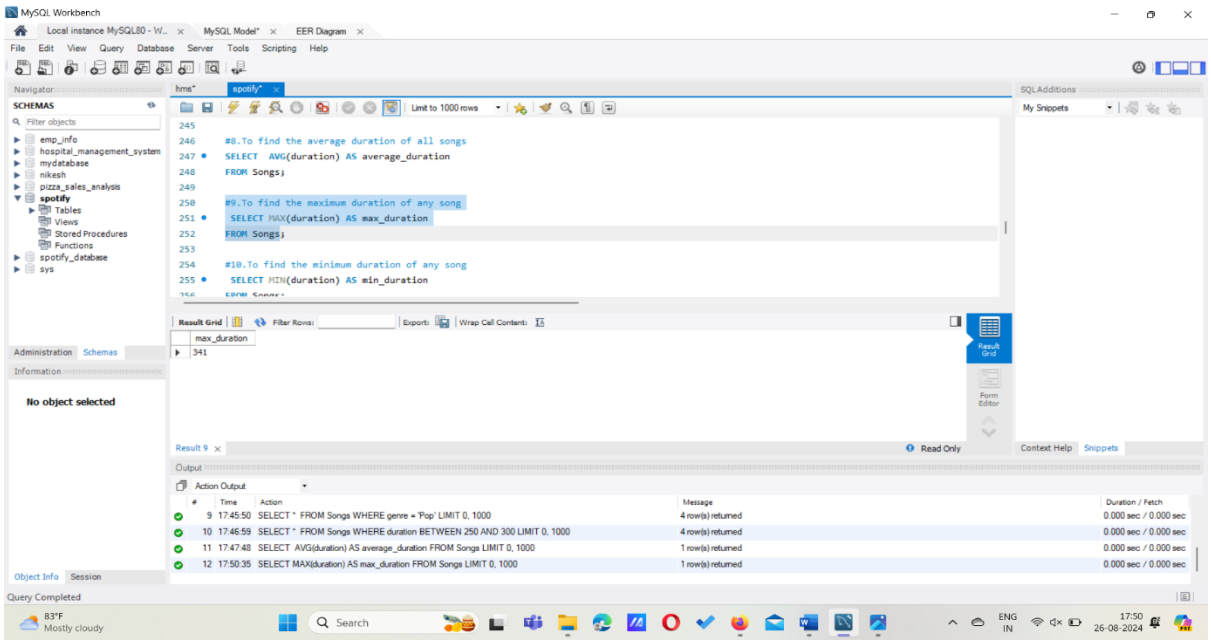
#8.To find the average duration of all songs

```
SELECT  AVG(duration) AS average_duration
FROM Songs;
```



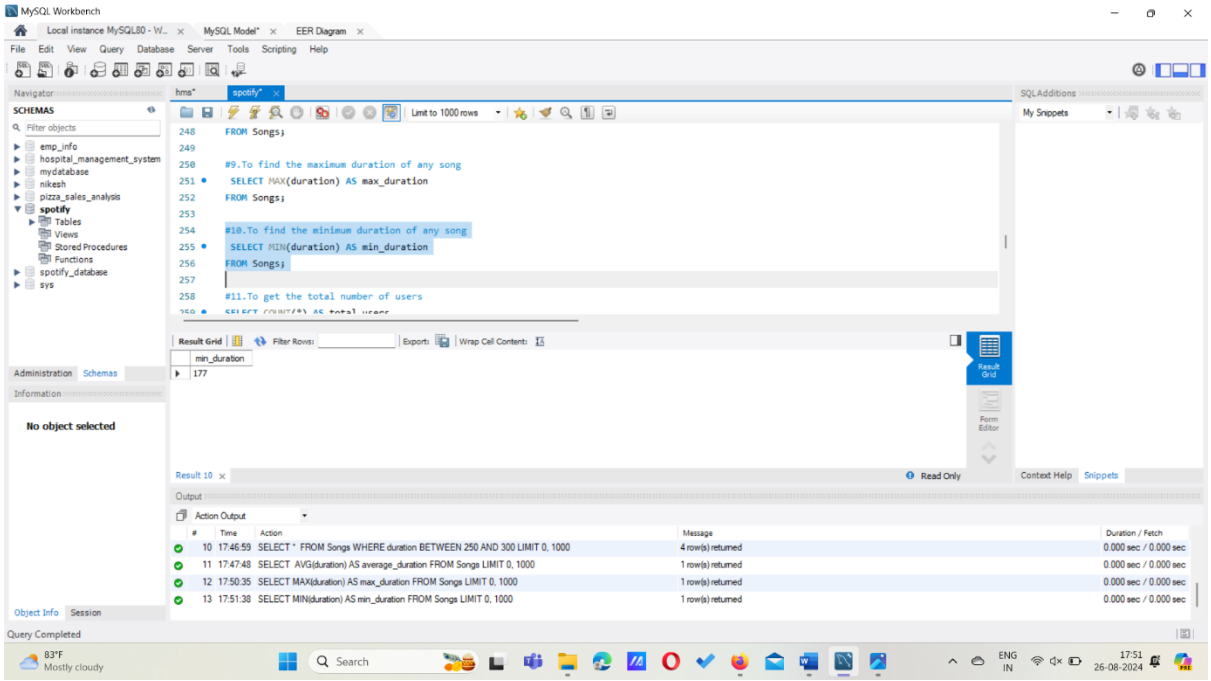
#9.To find the maximum duration of any song

```
SELECT MAX(duration) AS max_duration
FROM Songs
```



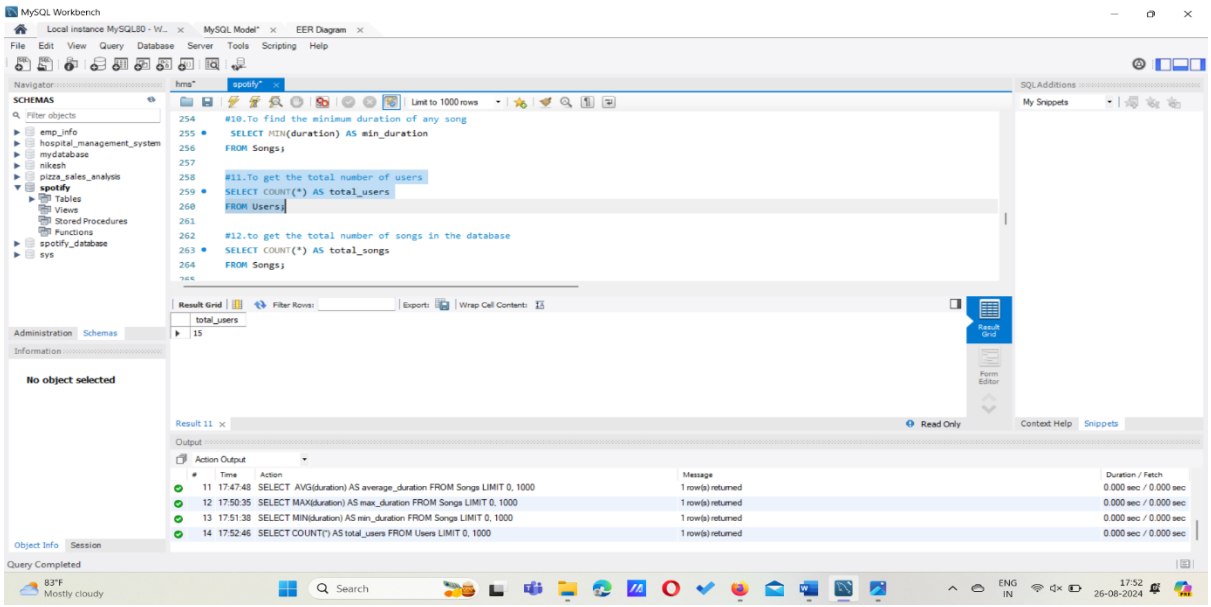
#10.To find the minimum duration of any song

```
SELECT MIN(duration) AS min_duration
FROM Songs;
```



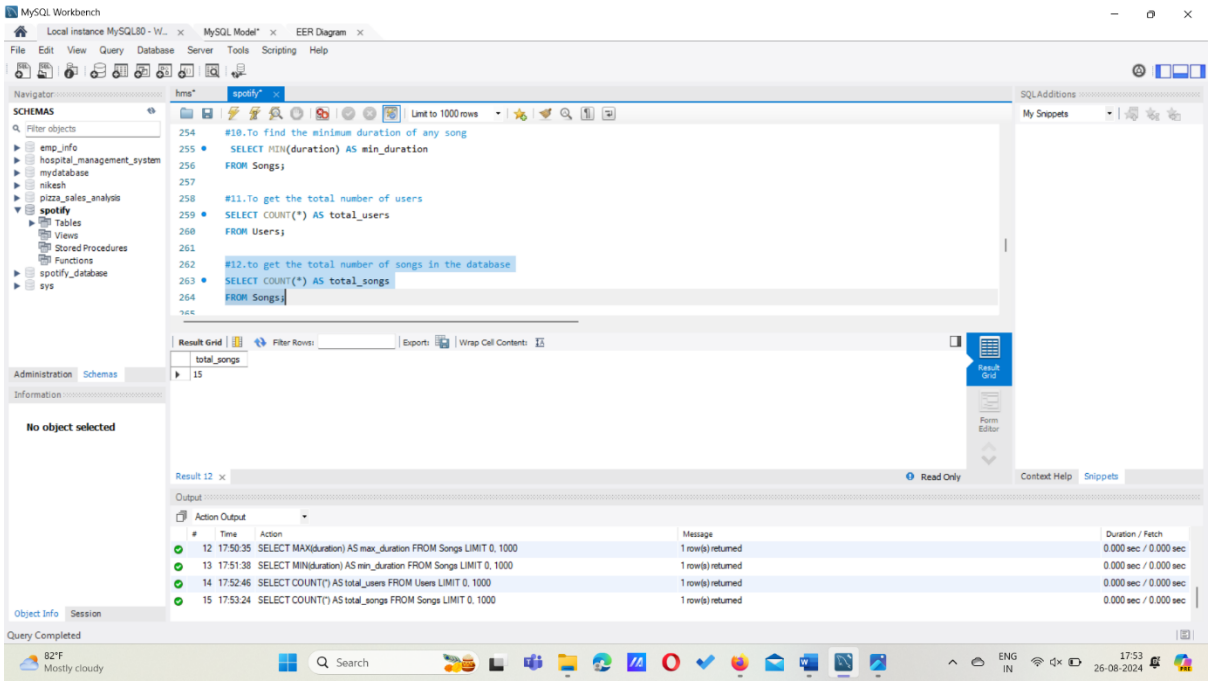
#11.To get the total number of users

```
SELECT COUNT(*) AS total_users
FROM Users;
```



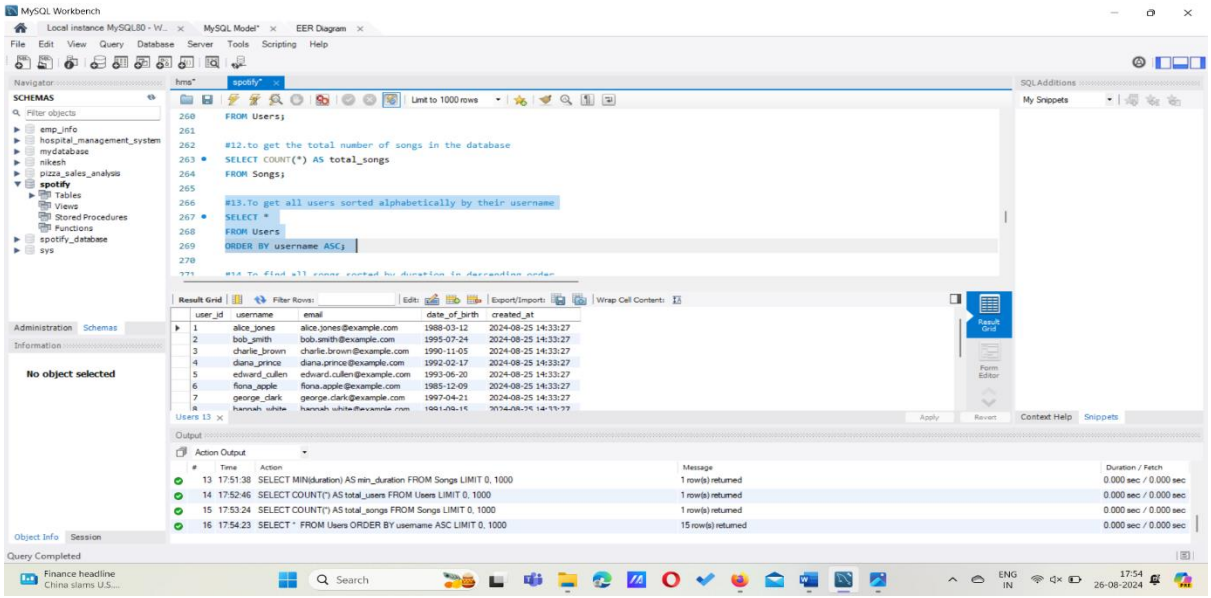
#12.to get the total number of songs in the database

```
SELECT COUNT(*) AS total_songs
FROM Songs;
```



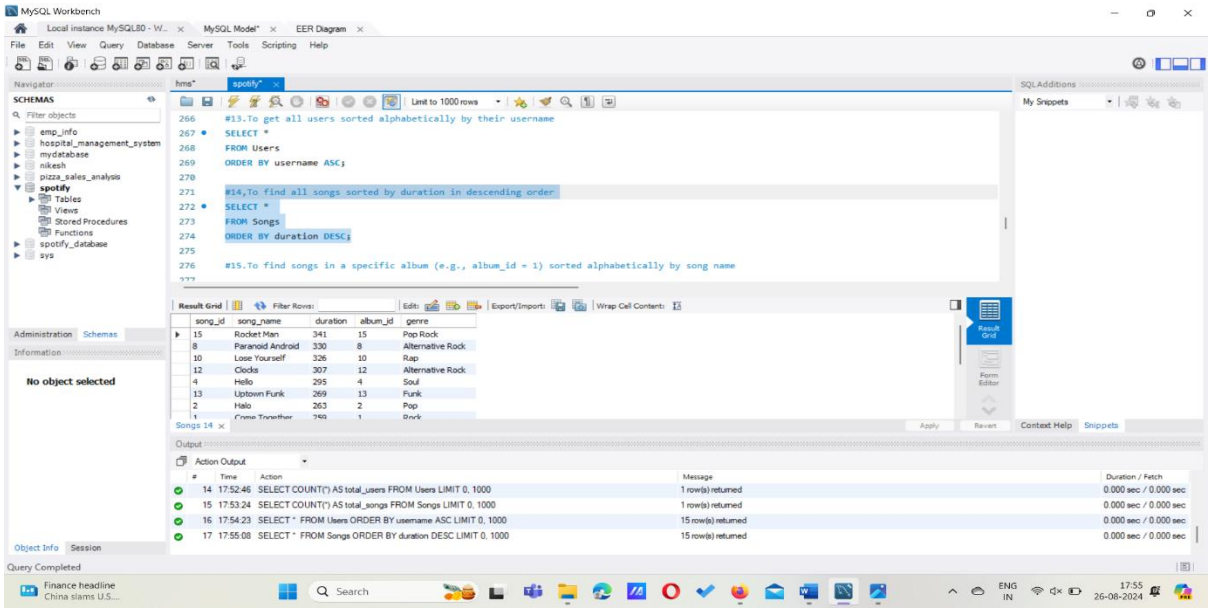
#13.To get all users sorted alphabetically by their username

```
SELECT *
FROM Users
ORDER BY username ASC;
```



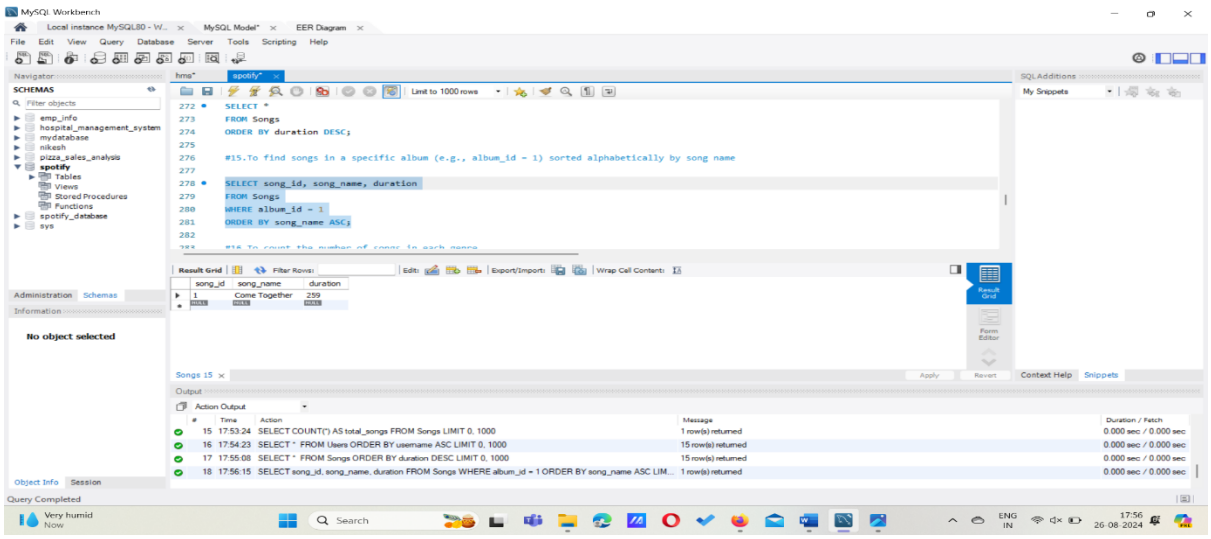
#14,To find all songs sorted by duration in descending order

```
SELECT *
FROM Songs
ORDER BY duration DESC;
```



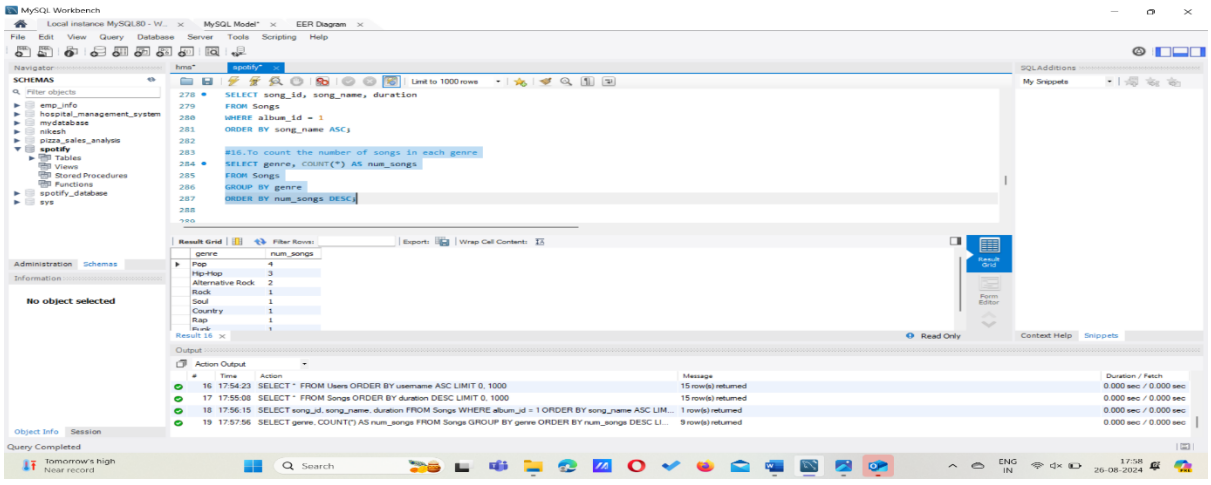
#15.To find songs in a specific album (e.g., album_id = 1) sorted alphabetically by song name

```
SELECT song_id, song_name, duration
FROM Songs
WHERE album_id = 1
ORDER BY song_name ASC;
```



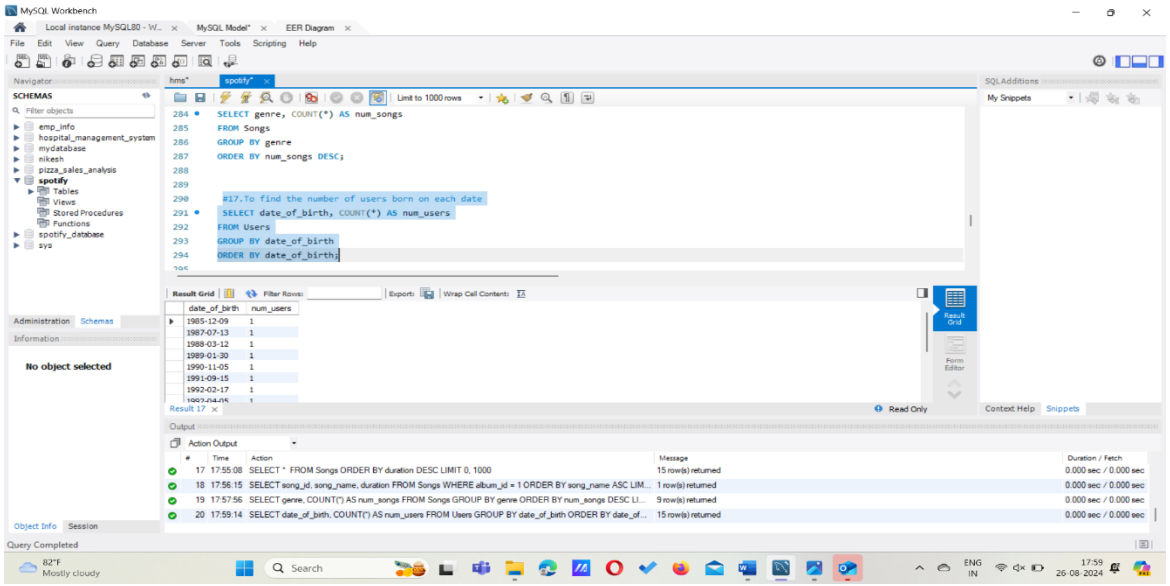
#16.To count the number of songs in each genre

```
SELECT genre, COUNT(*) AS num_songs
FROM Songs
GROUP BY genre
ORDER BY num_songs DESC;
```



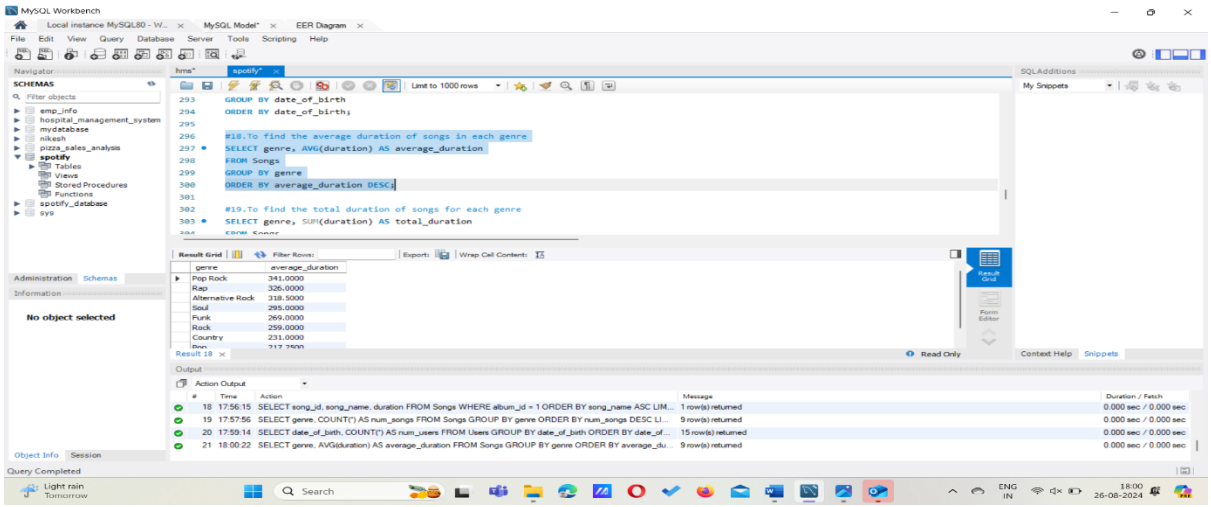
#17.To find the number of users born on each date

```
SELECT date_of_birth, COUNT(*) AS num_users
FROM Users
GROUP BY date_of_birth
ORDER BY date_of_birth;
```



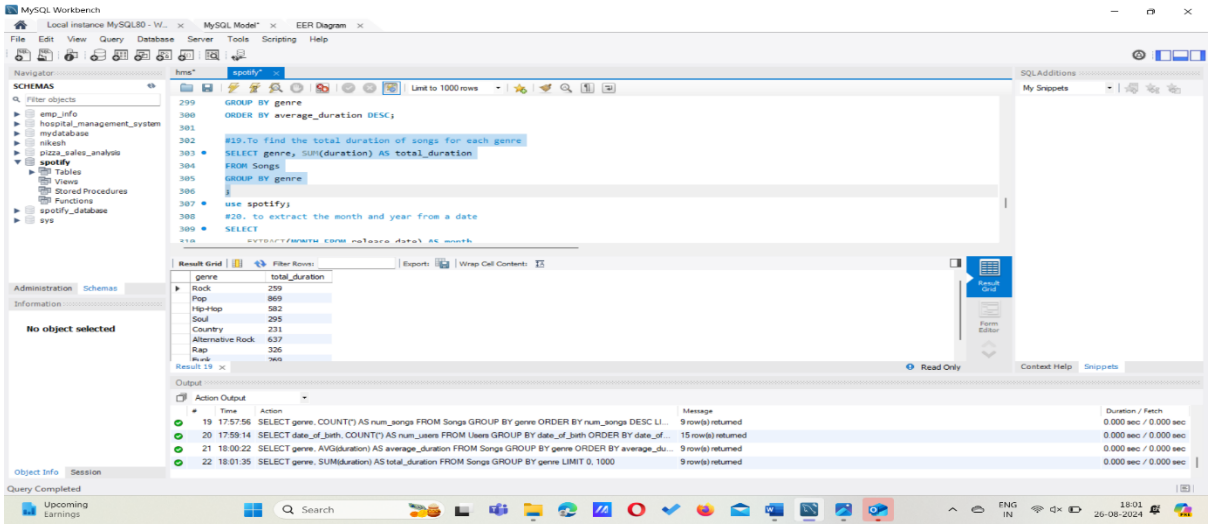
#18.To find the average duration of songs in each genre

```
SELECT genre, AVG(duration) AS average_duration
FROM Songs
GROUP BY genre
ORDER BY average_duration DESC;
```



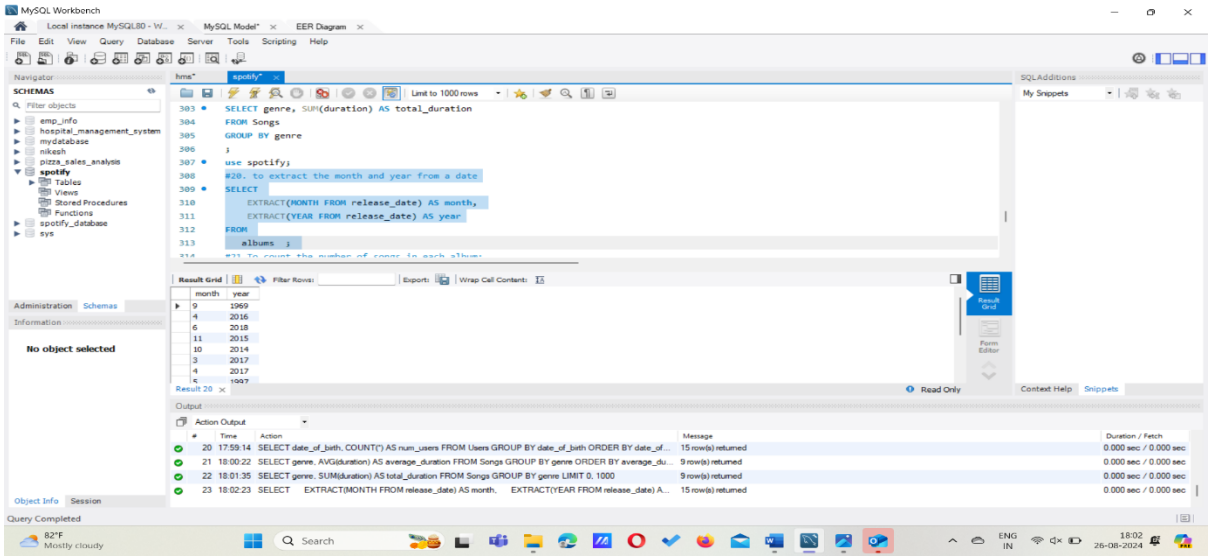
#19.To find the total duration of songs for each genre

```
SELECT genre, SUM(duration) AS total_duration
FROM Songs
GROUP BY genre;
```



#20. to extract the month and year from a date

```
SELECT
    EXTRACT(MONTH FROM release_date) AS month,
    EXTRACT(YEAR FROM release_date) AS year
FROM ALBUMS;
```



#21.To count the number of songs in each album:

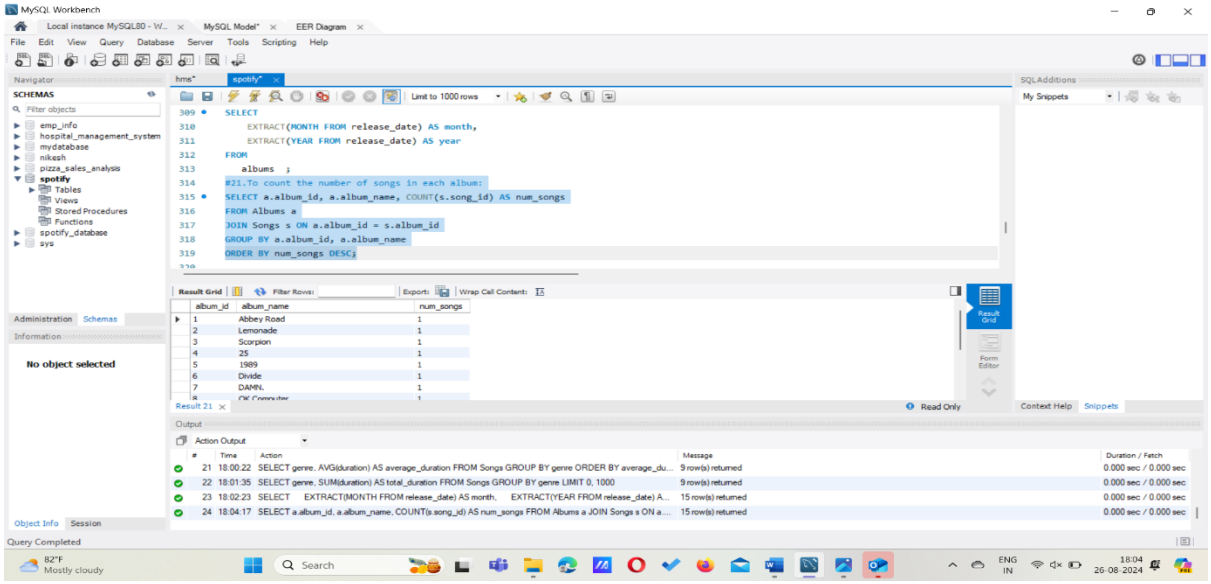
SELECT a.album_id, a.album_name, COUNT(s.song_id) AS num_songs

FROM Albums a

JOIN Songs s ON a.album_id = s.album_id

GROUP BY a.album_id, a.album_name

ORDER BY num_songs DESC;



#22.To find the number of streams per song, grouped by date of streaming

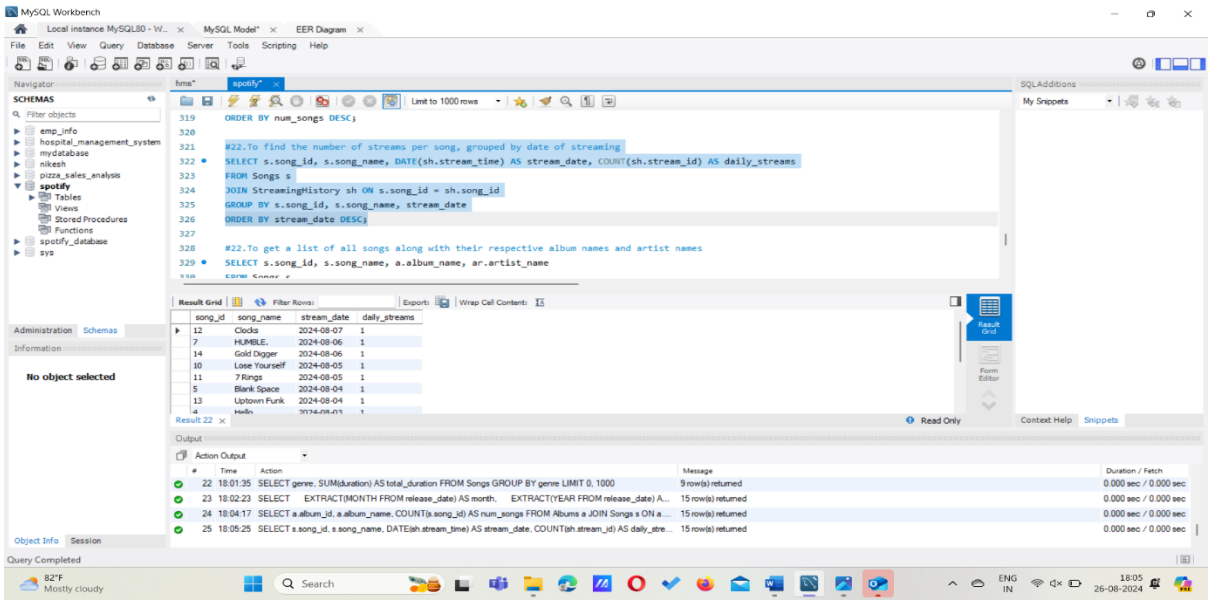
SELECT s.song_id, s.song_name, DATE(sh.stream_time) AS stream_date, COUNT(sh.stream_id) AS daily_streams

FROM Songs s

JOIN StreamingHistory sh ON s.song_id = sh.song_id

GROUP BY s.song_id, s.song_name, stream_date

ORDER BY stream_date DESC;



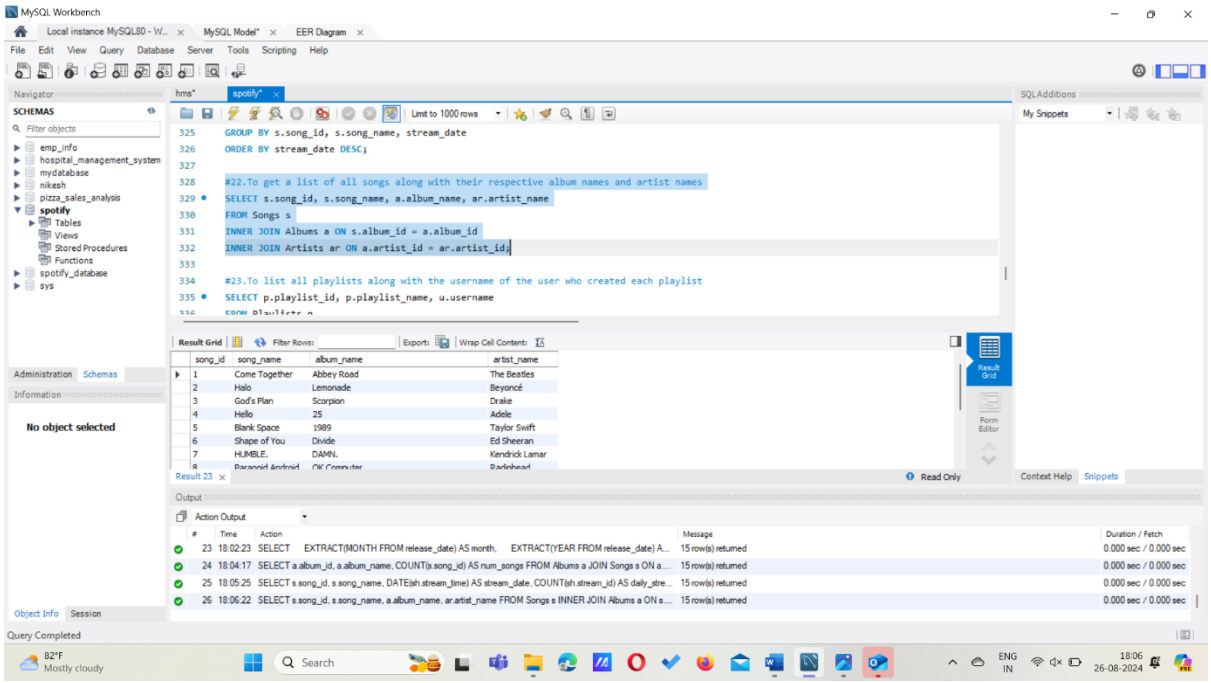
#22.To get a list of all songs along with their respective album names and artist names

SELECT s.song_id, s.song_name, a.album_name, ar.artist_name

FROM Songs s

INNER JOIN Albums a ON s.album_id = a.album_id

INNER JOIN Artists ar ON a.artist_id = ar.artist_id;

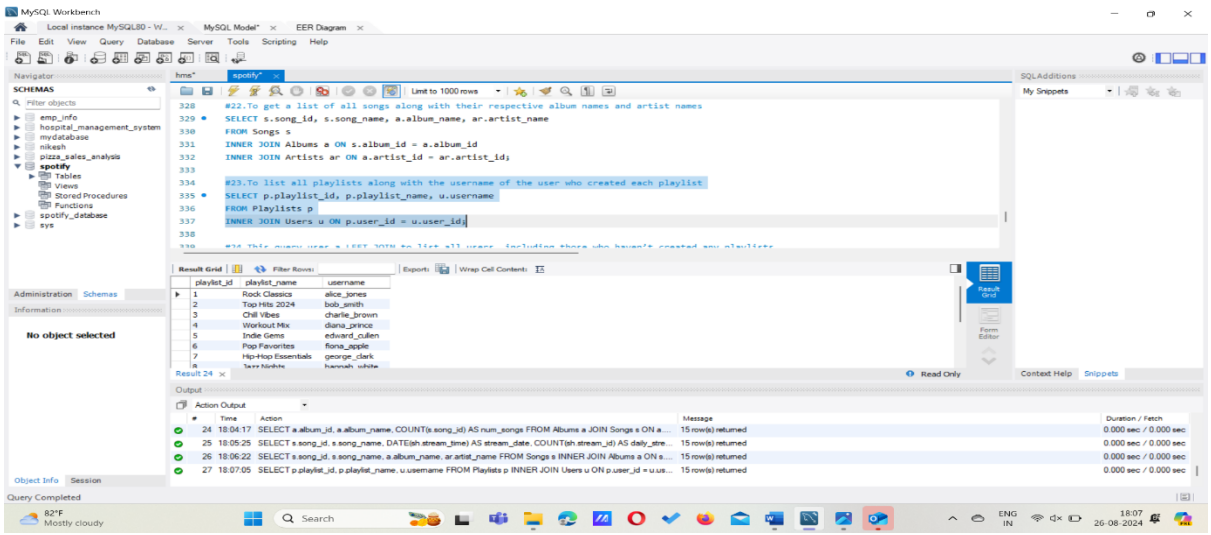


#23.To list all playlists along with the username of the user who created each playlist

SELECT p.playlist_id, p.playlist_name, u.username

FROM Playlists p

INNER JOIN Users u ON p.user_id = u.user_id;

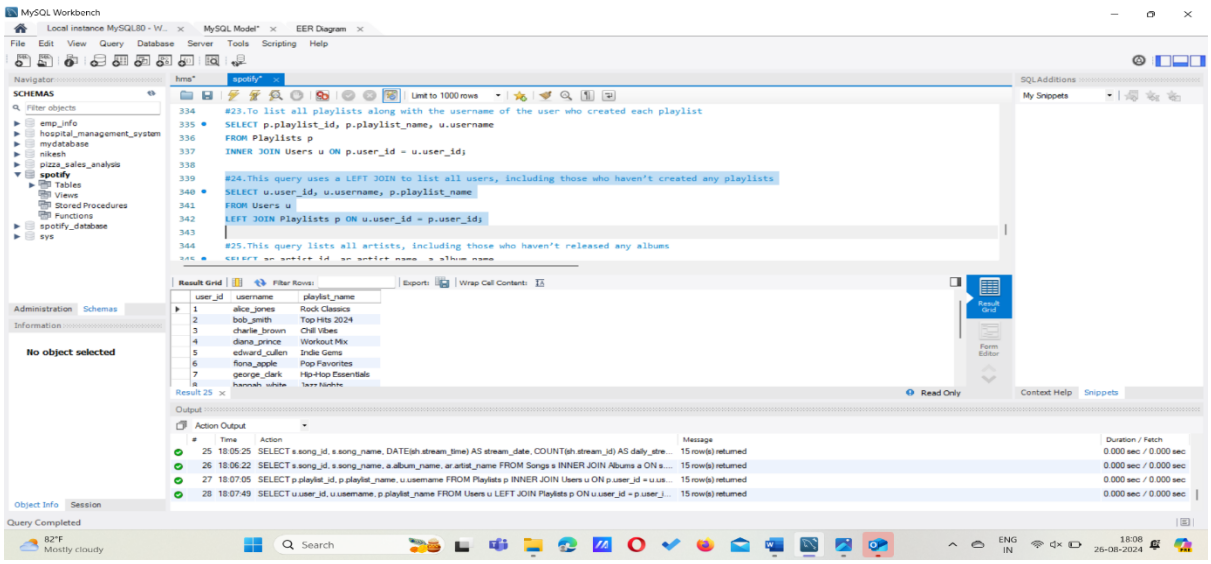


#24.This query uses a LEFT JOIN to list all users, including those who haven't created any playlists

SELECT u.user_id, u.username, p.playlist_name

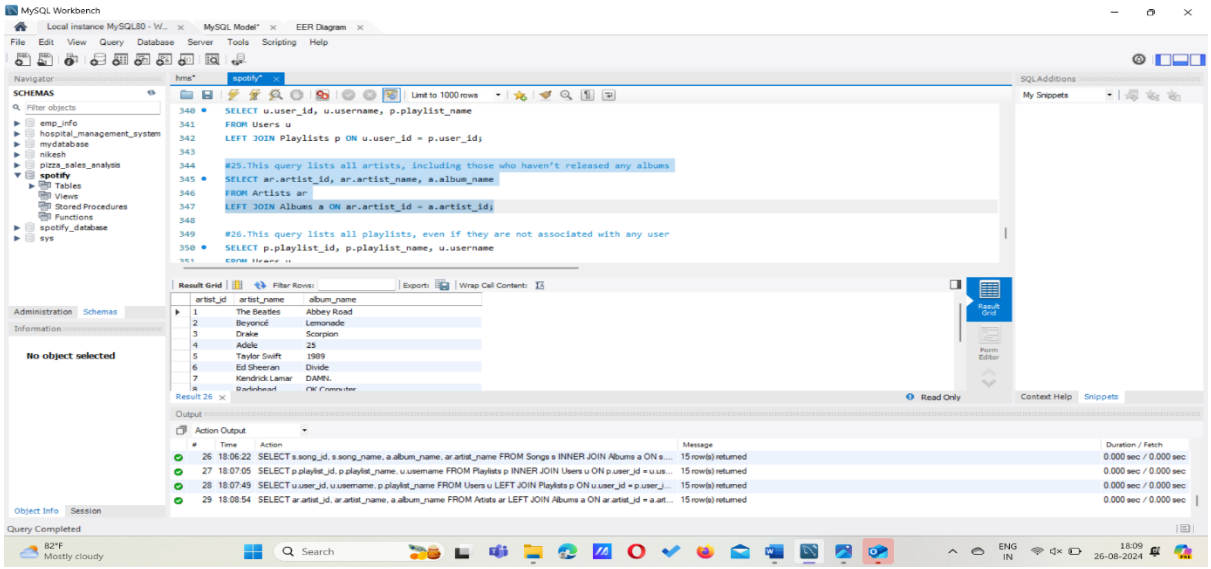
FROM Users u

LEFT JOIN Playlists p ON u.user_id = p.user_id;



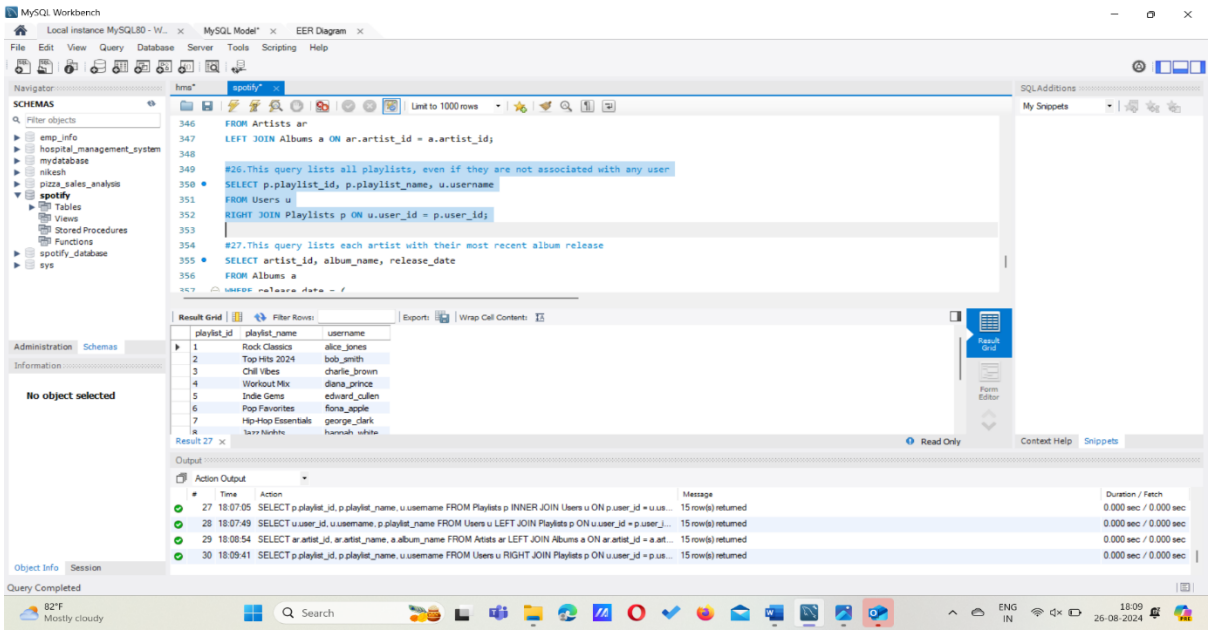
#25.This query lists all artists, including those who haven’t released any albums

```
SELECT ar.artist_id, ar.artist_name, a.album_name
FROM Artists ar
LEFT JOIN Albums a ON ar.artist_id = a.artist_id;
```



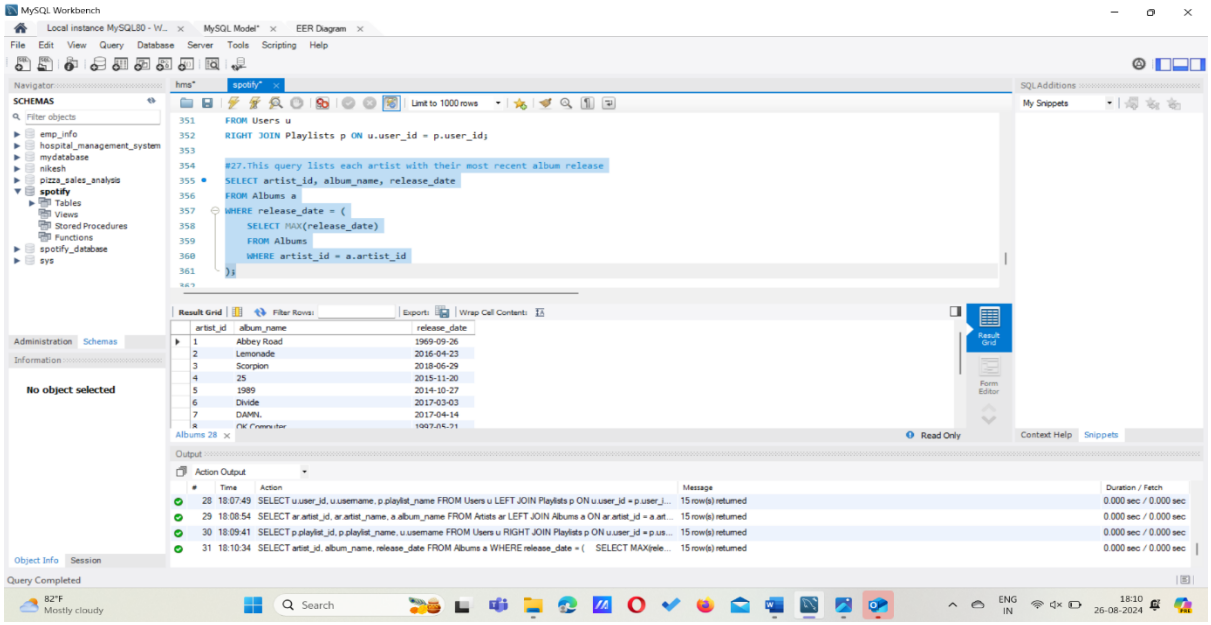
#26.This query lists all playlists, even if they are not associated with any user

```
SELECT p.playlist_id, p.playlist_name, u.username
FROM Users u
RIGHT JOIN Playlists p ON u.user_id = p.user_id;
```



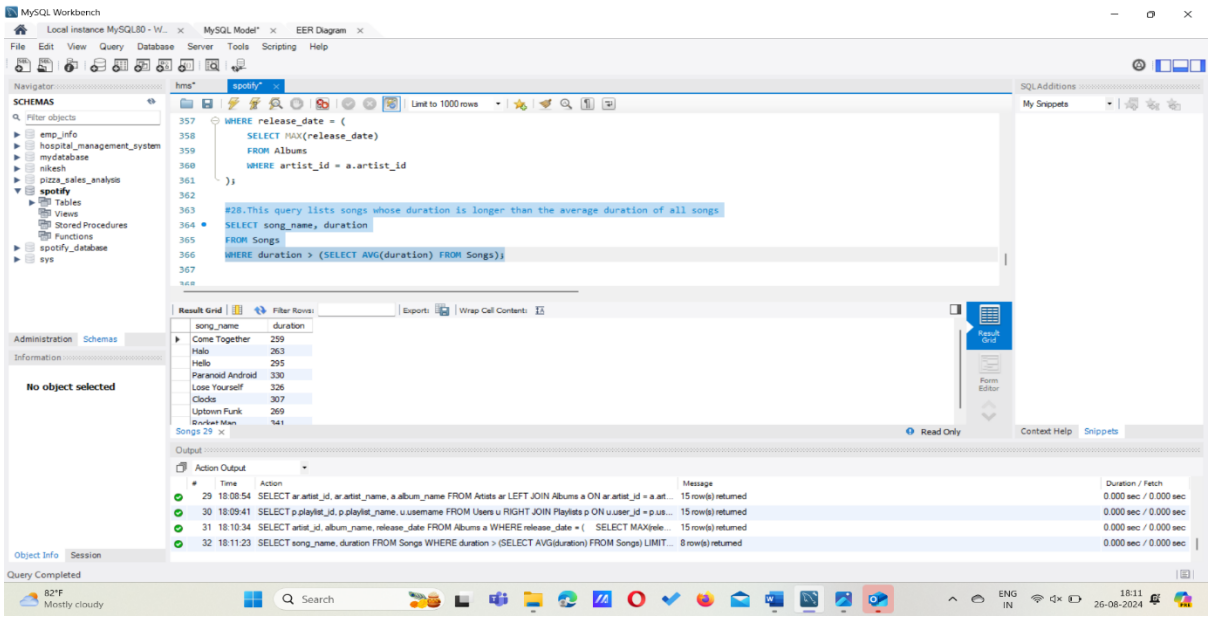
#27.This query lists each artist with their most recent album release

```
SELECT artist_id, album_name, release_date
FROM Albums a
WHERE release_date = (
    SELECT MAX(release_date)
    FROM Albums
    WHERE artist_id = a.artist_id
);
```



#28.This query lists songs whose duration is longer than the average duration of all songs

```
SELECT song_name, duration
FROM Songs
WHERE duration > (SELECT AVG(duration) FROM Songs);
```



CONCLUSION:

The Spotify Database Management System is a structured solution that efficiently organizes and manages vast data related to users, artists, albums, songs, playlists, and streaming activities on the platform. The Spotify Database Management System is a crucial component that enhances the platform's operations, user engagement, and business intelligence, ensuring a seamless and personalized experience for its global users.