MUSICIFY - MUSIC LIBRARY SYSTEM

A COURSE PROJECT REPORT

By

Ansab Aalim (RA2111027010030)

Shwetha Anand (RA2111027010045)

Kshitij Rastogi (RA2111027010051)

Under the guidance of **Dr. D Hemavathi**

In partial fulfilment for the Course

18CSC303J-Database Management Systems

In

School of Computing

FACULTY OF ENGINEERING AND TECHNOLOGY SRM INSTITUTE OF

SCIENCE AND TECHNOLOGY

Kattankulathur, Chengalpattu District

APRIL 2024.

Acknowledgement

We would like to express our gratitude to our Professor, Dr. D Hemavathi who gave us the golden opportunity to do this project on the topic **"<u>MUSICIFY</u>"** which also helped us in doing a lot of research and we came to know about so many new things we are thankful to her.

We are also thankful to all the other faculty, teaching, and non-teaching staff members of our department for their kind co-operation and help.

Lastly, we would also like to thank our friends who helped us a lot in finishing this project within the limited time.

Ansab Aalim	**(RA2111027010030)**
Shwetha Anand	**(RA2111027010045)**
Kshitij Rastogi	**(RA2111027010051)**

SRM INSTITUTE OF SCIENCE &TECHNOLOGY

COLLEGE OF ENGINEERING &TECHNOLOGY

S.R.M. NAGAR, KATTANKULATHUR – 603203

BONAFIDE CERTIFICATE

Certified that this project report “MUSICIFY” is the Bonafide work of Ansab Aalim (RA2111027010030), Shwetha Anand (RA2111027010045), Kshitij Rastogi (RA2111027010051) of III Year/VI Semester of B.Tech, CSE with Specialization in Big Data Analytics who carried out the mini project work under my supervision for the course 18CSC303J- Database Management Systems in Data Science and Business Systems department, School of Computing, SRM Institute of Science and Technology during the academic year 2023-2024 (Even Semester).

Signature of Head of the Department	Signature of Faculty In charge
Dr. Lakshmi M	Dr. D Hemavathi
Head of the Department	Associate Professor
Data Science and Business Systems	Data Science and Business Systems
School of Computing	School of Computing

Index

CONTENTS

1.INTRODUCTION

The "Music Library Database" is designed to efficiently manage and organize various aspects of a music library, catering to the needs of users, playlists, albums, tracks, artists, and their releases. The database schema is composed of several interconnected tables, each serving a specific purpose in storing and retrieving information related to music assets and their associated metadata.

- **User Table:**

The "User" table stores information about users who interact with the music library system. It includes fields such as username, first name, last name, and email. The username serves as the primary key, uniquely identifying each user.

- **Playlist Table:**

The "Playlist" table represents playlists created within the music library system. It contains details like playlist ID, playlist name, type, and creator's username. The creator's username field establishes a foreign key relationship with the "User" table, linking each playlist to its creator.

- **Album Table:**

The "Album" table holds information about albums available in the music library. It stores attributes such as album ID, year of release, title, genre, and whether it is a music album or a podcast. The album ID serves as the primary key, uniquely identifying each album.

- **Track Table:**

The "Track" table stores details about individual tracks or songs. It includes fields like track ID, track name, duration, and the album ID it belongs to. The album ID field establishes a foreign key relationship with the "Album" table, linking each track to its corresponding album.

- **Artist Table:**

The "Artist" table contains information about music artists. It includes fields such as artist ID, country of origin, and artist name. The artist ID serves as the primary key, uniquely identifying each artist.

- **Releases Table:**

The "Releases" table establishes a many-to-many relationship between artists and albums, indicating which artists have released which albums. It includes foreign keys referencing the artist ID from the "Artist" table and the album ID from the "Album" table, forming a composite primary key.

- **PlaylistTrack Table:**

The "PlaylistTrack" table associates tracks with playlists, defining the contents of each playlist. It includes foreign keys referencing the playlist ID from the "Playlist" table and the track ID from the "Track" table, forming a composite primary key.

1.2 Advantages of MySQL:

- **Efficient Data Organization:** The database structure efficiently organizes music-related data into separate tables, allowing for easy retrieval and management of information.
- **Data Integrity:** By enforcing foreign key constraints and defining appropriate relationships between tables, the database ensures data integrity, minimizing errors and inconsistencies in the stored information.
- **Scalability:** The modular design of the database allows for easy scalability as the music library grows. New albums, tracks, artists, and users can be seamlessly incorporated into the database without disrupting existing functionality.

2.1 About the Project:

The "Musicify" project aims to develop a comprehensive software solution for managing and organizing music collections. It caters to individuals, music enthusiasts, and organizations looking for an efficient way to store, categorize, and enjoy music content. Built on a robust database backend, the project offers a user-friendly interface and a wide range of features to meet the diverse needs of users.

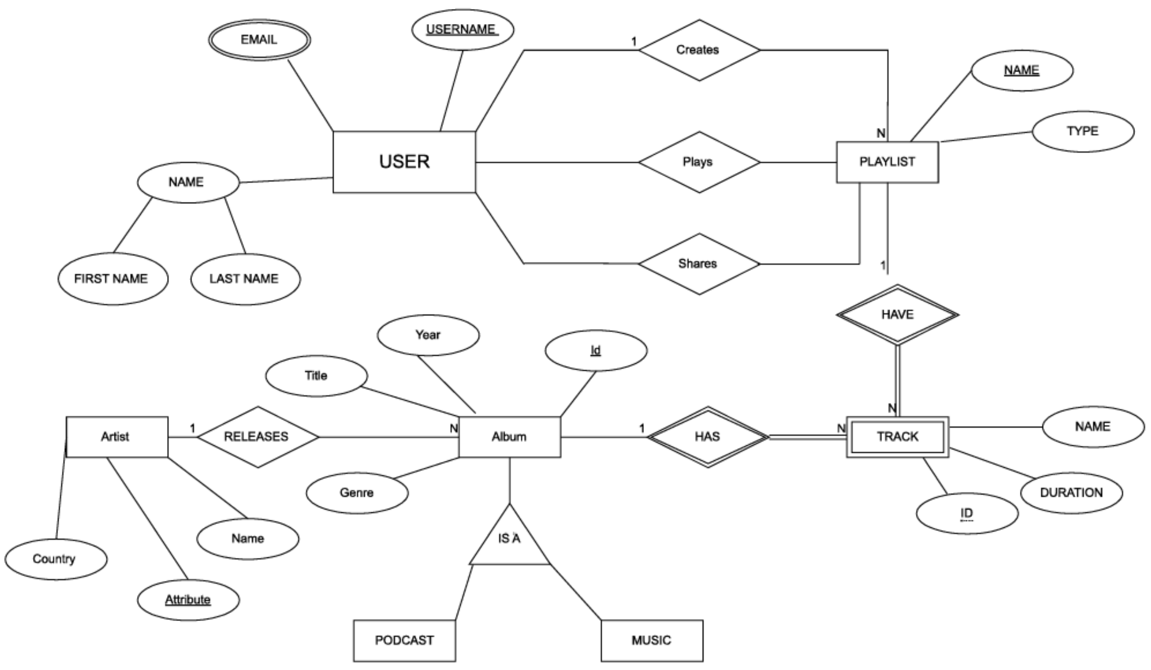
2.1.2 Main features are:

- **User Management**
- **Playlist Creation**
- **Music Catalog**
- **Search and Filtering**

2.1.3 Objectives:

- **Efficient Music Management:** The primary objective is to provide users with a centralized platform for efficiently managing their music collections, including creating playlists, organizing tracks, and discovering new music.
- **Enhanced User Experience:** The project aims to deliver a seamless and enjoyable user experience through intuitive interfaces, fast performance, and personalized recommendations.
- **Scalability and Reliability**: Ensuring the scalability and reliability of the system to accommodate growing user bases and handle increased traffic while maintaining high performance and uptime.
- **Data Security and Privacy**: Implementing robust security measures to protect user data, prevent unauthorized access, and adhere to privacy regulations and best practices.

ER Diagram


EMAIL
USERNAME
USER
NAME
FIRST NAME
LAST NAME
Creates
Plays
Shares
1
N
PLAYLIST
NAME
TYPE
1
HAVE
N
TRACK
NAME
DURATION
ID
HAS
1
N
Album
Year
Title
Id
Genre
IS A
PODCAST
MUSIC
RELEASES
1
N
Artist
Country
Attribute
Name

• MUSICIFY:

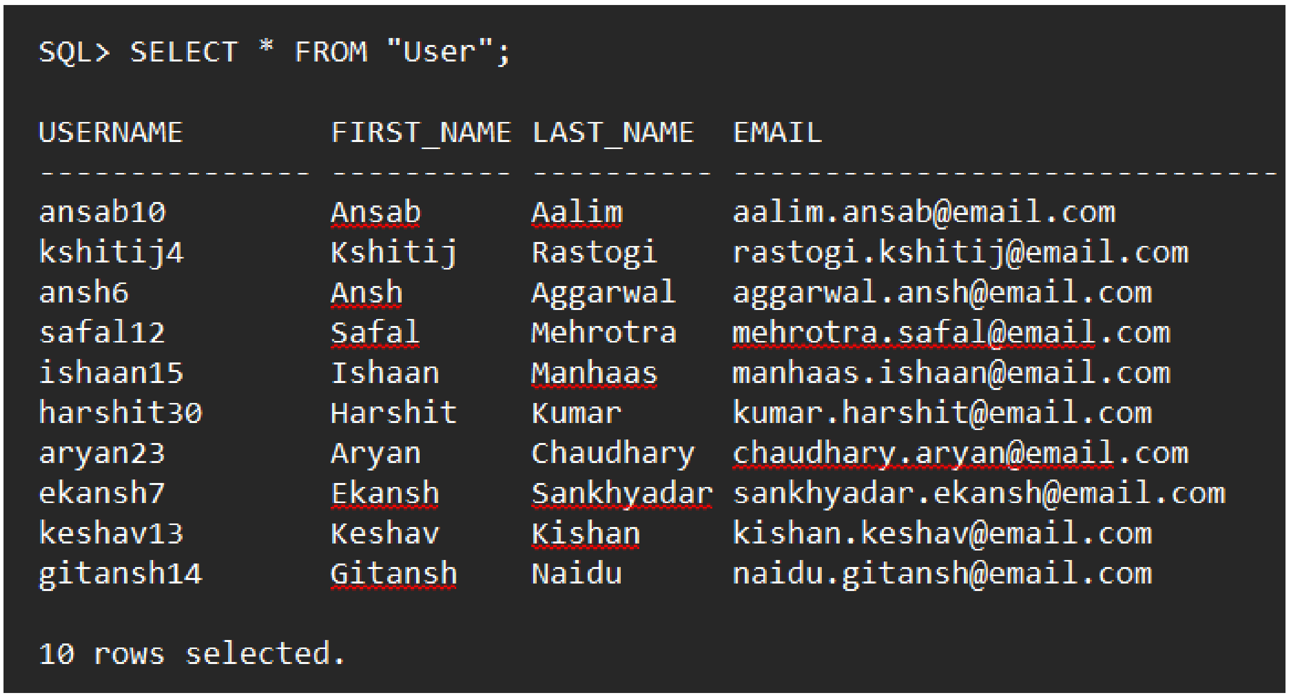
The proposed work consists of 7 tables that are interconnected. The team members work on tables and keep updating them by implementing queries.

The structure and function of each table are described below:

1. User table:

It has information about board members of the whole website. Details consist of Name, Email. Username here is the Primary key of the table.

- Username
- First_name
- Last_name
- Email

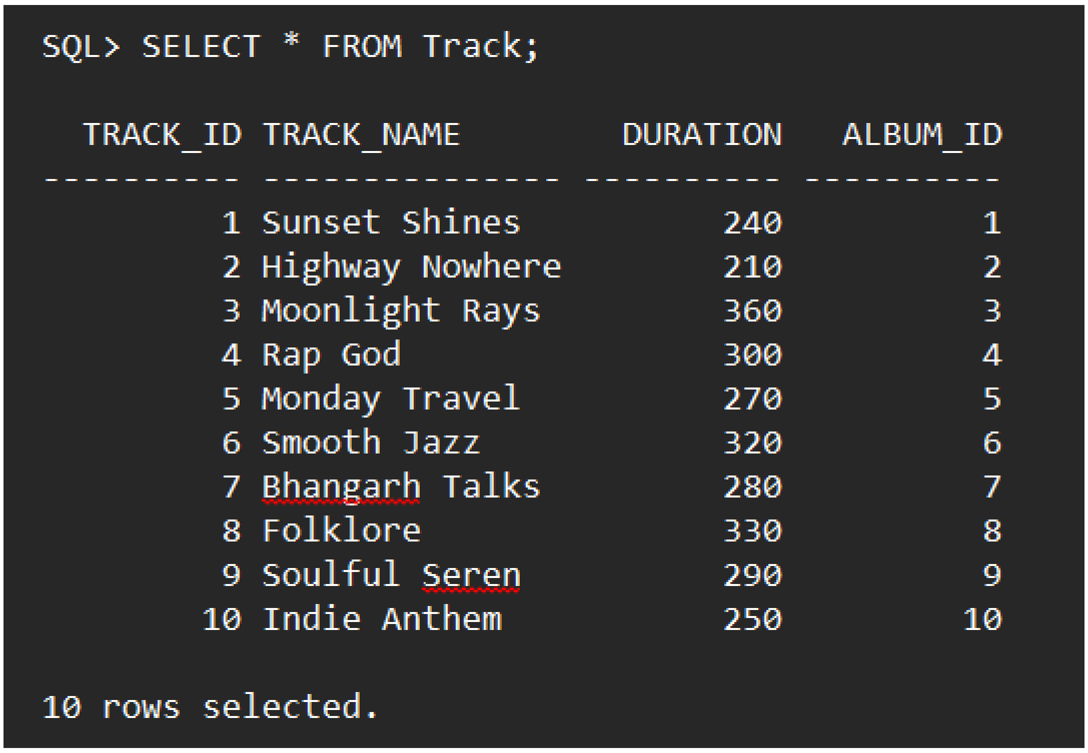


```
SQL> SELECT * FROM "User";

USERNAME        FIRST_NAME LAST_NAME  EMAIL
--------------- ---------- ---------- ------------------------------
ansab10         Ansab      Aalim      aalim.ansab@email.com
kshitij4        Kshitij    Rastogi    rastogi.kshitij@email.com
ansh6           Ansh       Aggarwal   aggarwal.ansh@email.com
safal12         Safal      Mehrotra   mehrotra.safal@email.com
ishaan15        Ishaan     Manhaas    manhaas.ishaan@email.com
harshit30       Harshit    Kumar      kumar.harshit@email.com
aryan23         Aryan      Chaudhary  chaudhary.aryan@email.com
ekansh7         Ekansh     Sankhyadar sankhyadar.ekansh@email.com
keshav13        Keshav     Kishan     kishan.keshav@email.com
gitansh14       Gitansh    Naidu      naidu.gitansh@email.com

10 rows selected.
```

2. Track Table:

This table is for vendors who contribute to a single branch. Their track id, track name duration, album_id are present in this table.

- Track_id
- Track_Name
- Duration
- Album_id

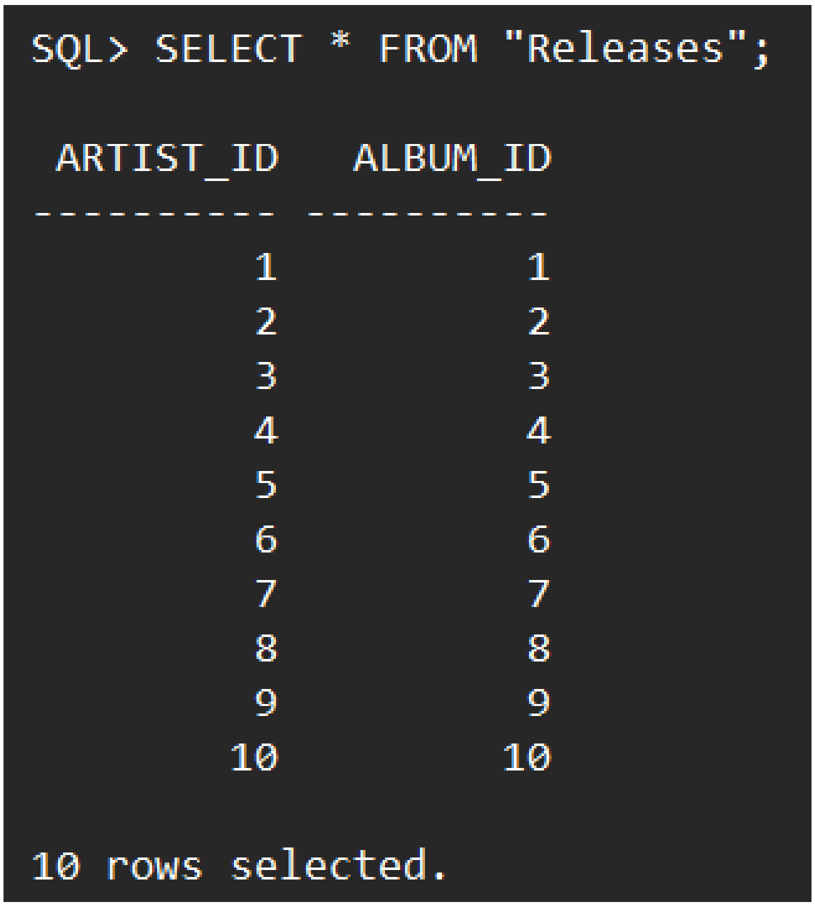


```
SQL> SELECT * FROM Track;

  TRACK_ID TRACK_NAME        DURATION   ALBUM_ID
---------- --------------- ---------- ----------
         1 Sunset Shines          240          1
         2 Highway Nowhere        210          2
         3 Moonlight Rays         360          3
         4 Rap God                300          4
         5 Monday Travel          270          5
         6 Smooth Jazz            320          6
         7 Bhangarh Talks         280          7
         8 Folklore               330          8
         9 Soulful Seren          290          9
        10 Indie Anthem           250         10

10 rows selected.
```

3. Releases Table:

The table consists of details of artist id and album id.

- Artist_id
- Album_id

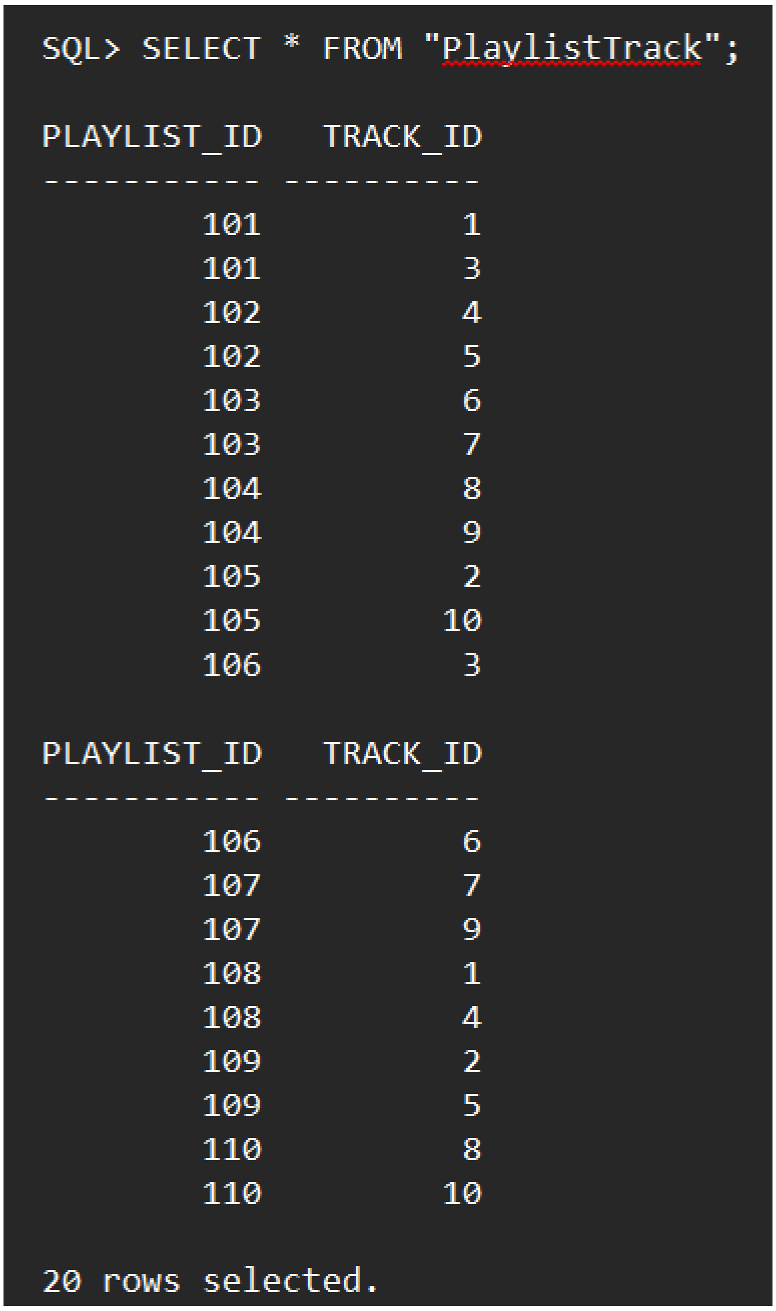


```
SQL> SELECT * FROM "Releases";

 ARTIST_ID   ALBUM_ID
---------- ----------
         1          1
         2          2
         3          3
         4          4
         5          5
         6          6
         7          7
         8          8
         9          9
        10         10

10 rows selected.
```

4. **Playlist Track Table:**

It is responsible for visualising trail data. It contains playlist_id, track_id.

- playlist_id
- Track_id

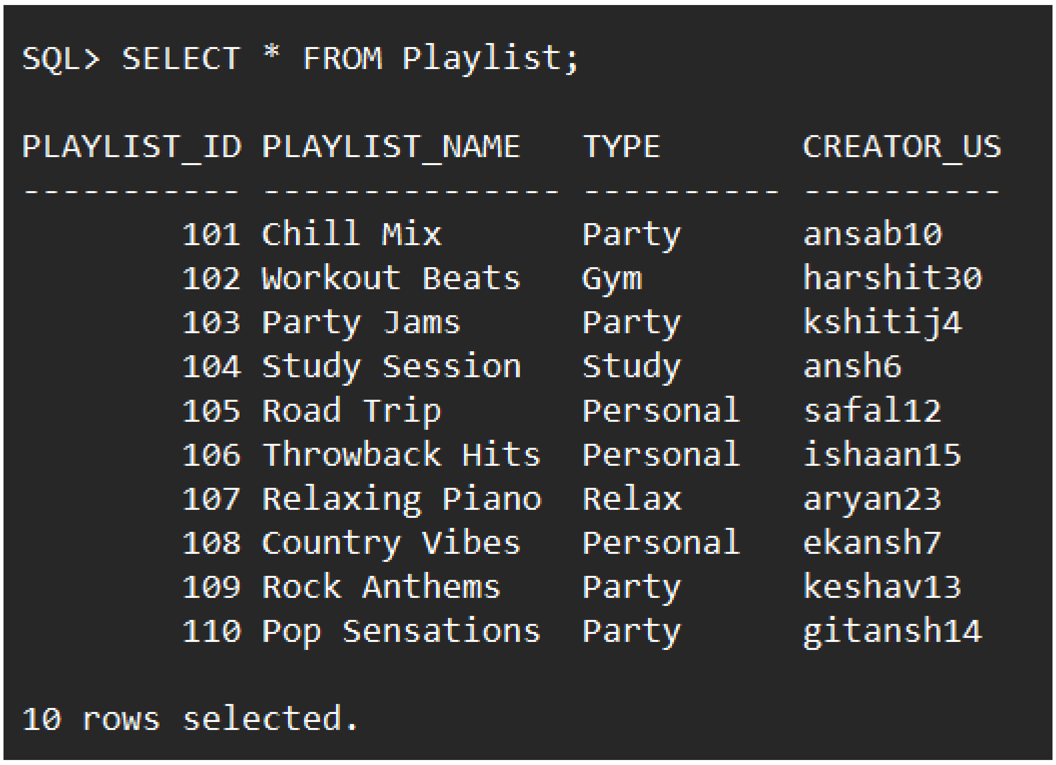


```
SQL> SELECT * FROM "PlaylistTrack";

PLAYLIST_ID   TRACK_ID
----------- ----------
        101          1
        101          3
        102          4
        102          5
        103          6
        103          7
        104          8
        104          9
        105          2
        105         10
        106          3

PLAYLIST_ID   TRACK_ID
----------- ----------
        106          6
        107          7
        107          9
        108          1
        108          4
        109          2
        109          5
        110          8
        110         10

20 rows selected.
```

5. **Playlist Table:**

This table has the database of all the shops which are present in a website. It contains the playlist id, playlist name, type.

- playlist_id
- playlist_name
- type

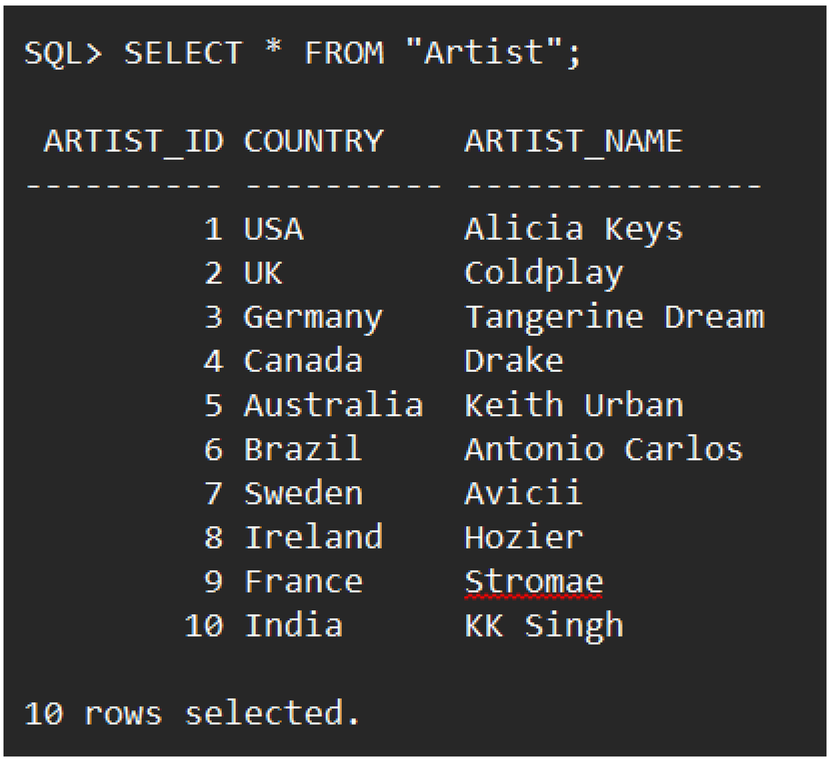


```
SQL> SELECT * FROM Playlist;

PLAYLIST_ID PLAYLIST_NAME   TYPE       CREATOR_US
----------- --------------- ---------- ----------
        101 Chill Mix       Party      ansab10
        102 Workout Beats   Gym        harshit30
        103 Party Jams      Party      kshitij4
        104 Study Session   Study      ansh6
        105 Road Trip       Personal   safal12
        106 Throwback Hits  Personal   ishaan15
        107 Relaxing Piano  Relax      aryan23
        108 Country Vibes   Personal   ekansh7
        109 Rock Anthems    Party      keshav13
        110 Pop Sensations  Party      gitansh14

10 rows selected.
```

6. **Artist Table:**

This table has the details of artist considered. The details include, artist_is, country, artist_name.

- Artist_id
- country
- Artist_name

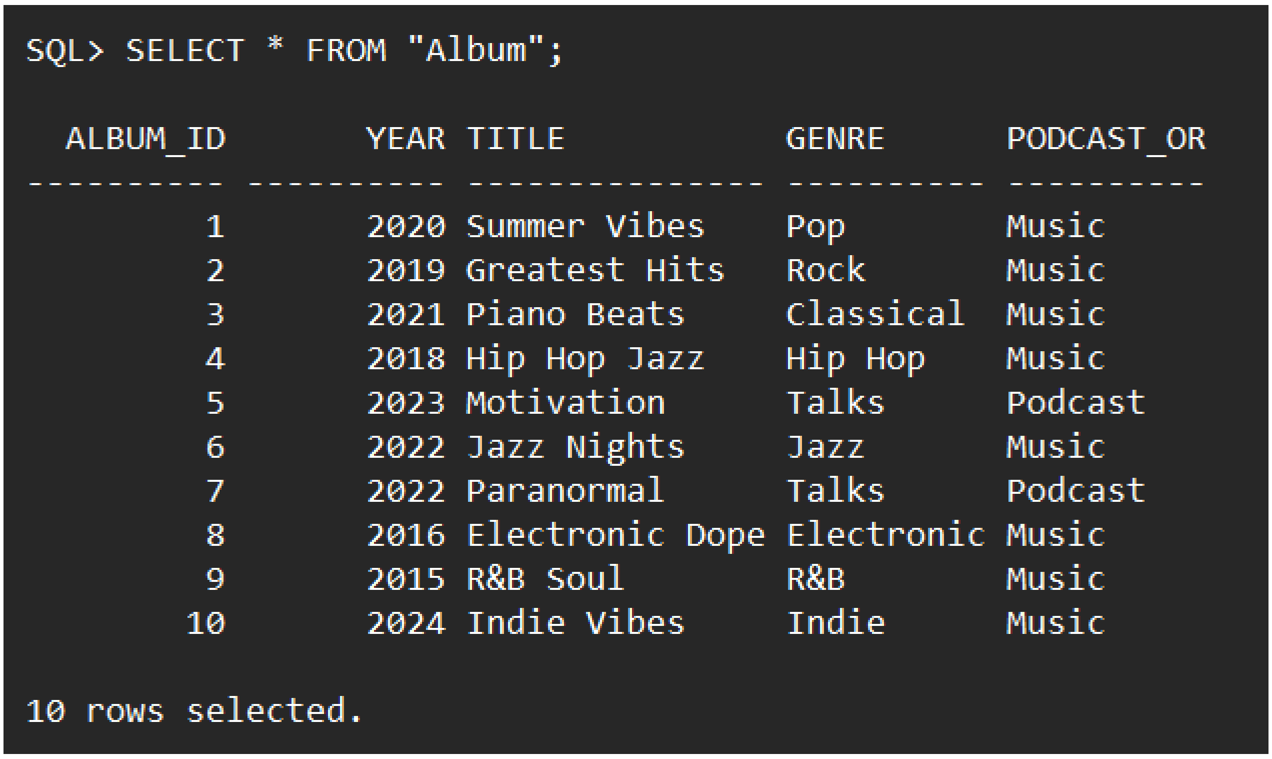


```
SQL> SELECT * FROM "Artist";

 ARTIST_ID COUNTRY    ARTIST_NAME
---------- ---------- ---------------
         1 USA        Alicia Keys
         2 UK         Coldplay
         3 Germany    Tangerine Dream
         4 Canada     Drake
         5 Australia  Keith Urban
         6 Brazil     Antonio Carlos
         7 Sweden     Avicii
         8 Ireland    Hozier
         9 France     Stromae
        10 India      KK Singh

10 rows selected.
```

7. **Album Table:**

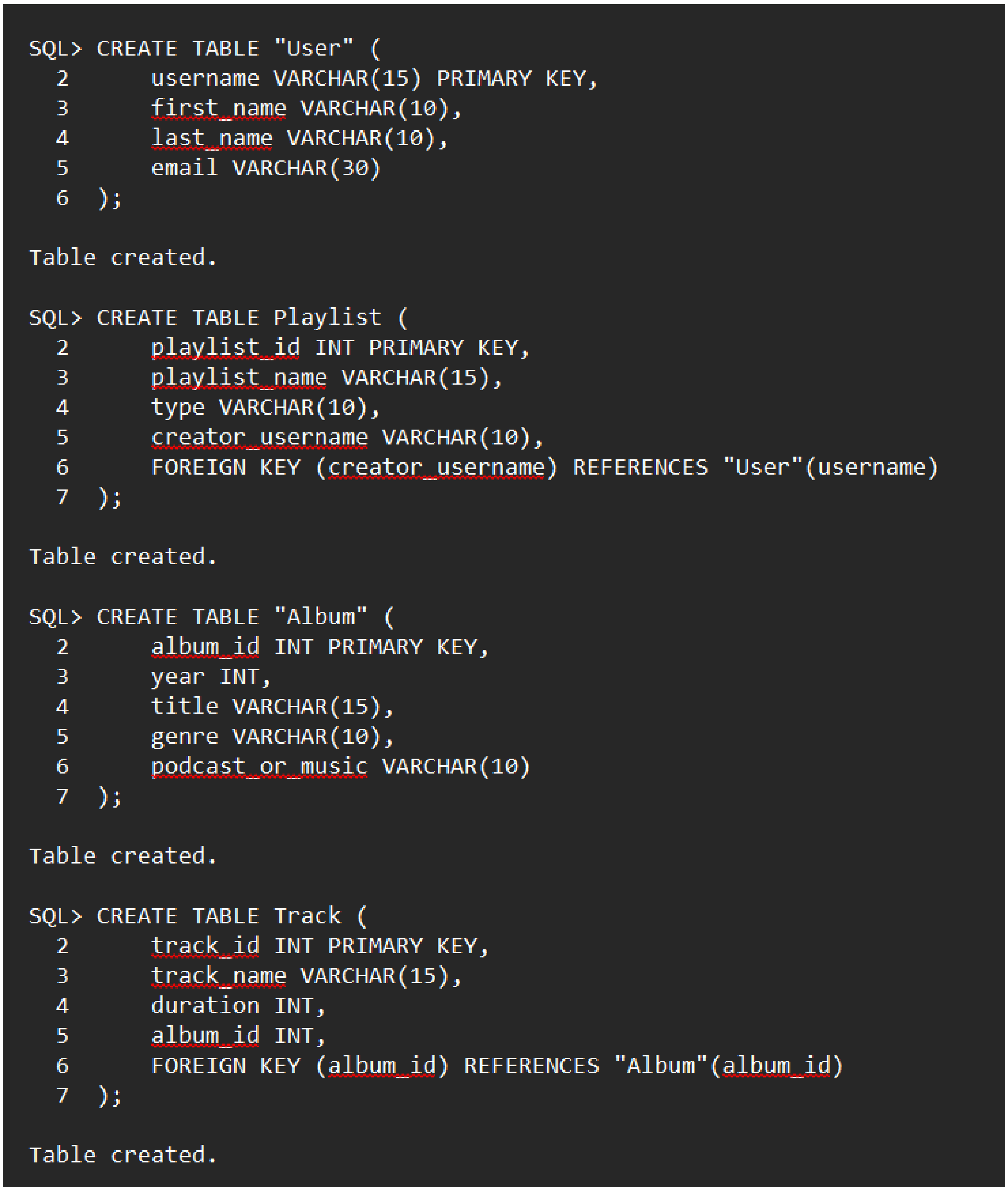
The master table has details of all branches of the details belongs to our project.

- Album id
- Year
- Title

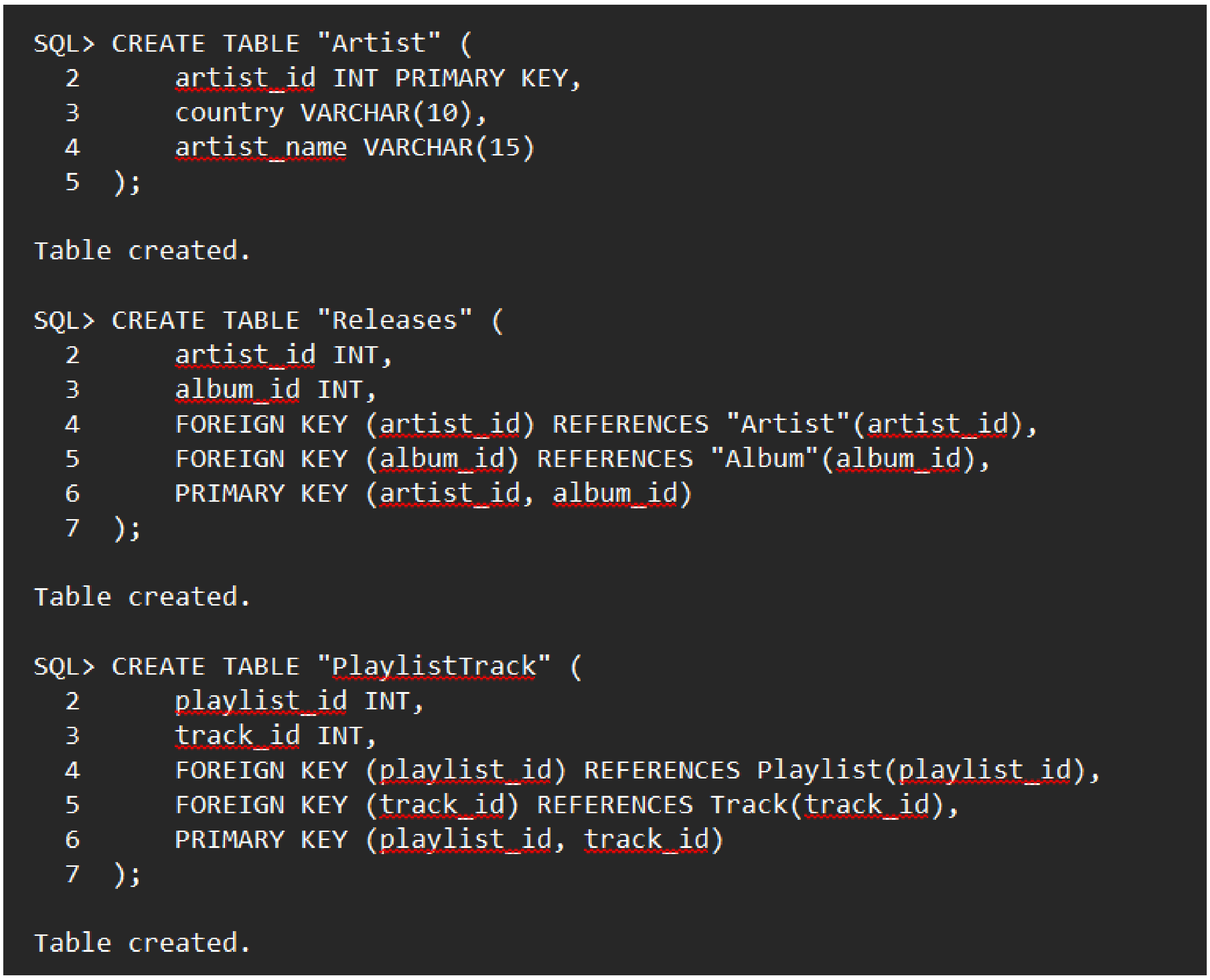


```
SQL> SELECT * FROM "Album";

  ALBUM_ID       YEAR TITLE           GENRE      PODCAST_OR
---------- ---------- --------------- ---------- ----------
         1       2020 Summer Vibes    Pop        Music
         2       2019 Greatest Hits   Rock       Music
         3       2021 Piano Beats     Classical  Music
         4       2018 Hip Hop Jazz    Hip Hop    Music
         5       2023 Motivation      Talks      Podcast
         6       2022 Jazz Nights     Jazz       Music
         7       2022 Paranormal      Talks      Podcast
         8       2016 Electronic Dope Electronic Music
         9       2015 R&B Soul        R&B        Music
        10       2024 Indie Vibes     Indie      Music

10 rows selected.
```

- **<u>TABLES:</u>**

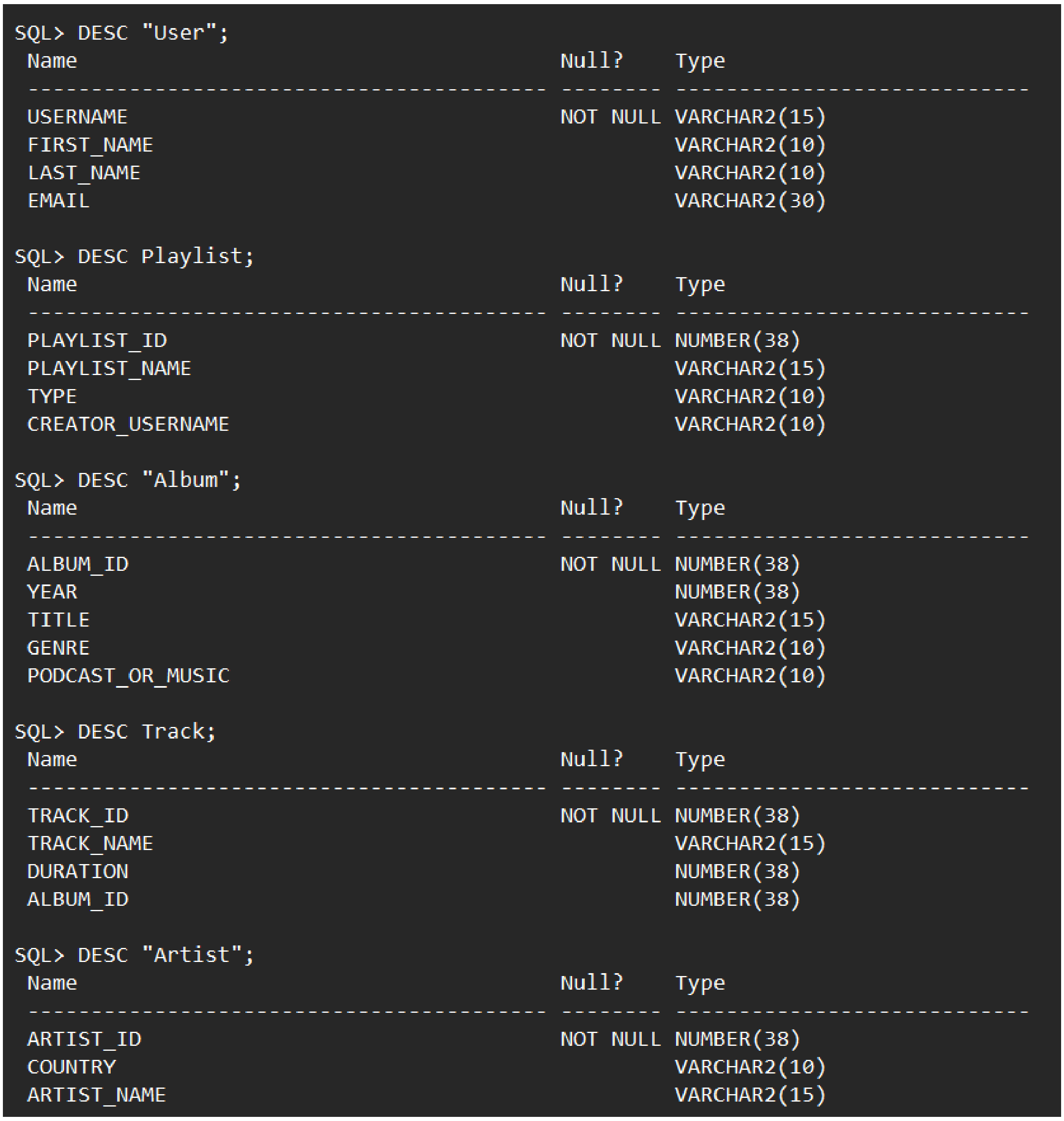


```
SQL> CREATE TABLE "User" (
  2      username VARCHAR(15) PRIMARY KEY,
  3      first_name VARCHAR(10),
  4      last_name VARCHAR(10),
  5      email VARCHAR(30)
  6  );

Table created.

SQL> CREATE TABLE Playlist (
  2      playlist_id INT PRIMARY KEY,
  3      playlist_name VARCHAR(15),
  4      type VARCHAR(10),
  5      creator_username VARCHAR(10),
  6      FOREIGN KEY (creator_username) REFERENCES "User"(username)
  7  );

Table created.

SQL> CREATE TABLE "Album" (
  2      album_id INT PRIMARY KEY,
  3      year INT,
  4      title VARCHAR(15),
  5      genre VARCHAR(10),
  6      podcast_or_music VARCHAR(10)
  7  );

Table created.

SQL> CREATE TABLE Track (
  2      track_id INT PRIMARY KEY,
  3      track_name VARCHAR(15),
  4      duration INT,
  5      album_id INT,
  6      FOREIGN KEY (album_id) REFERENCES "Album"(album_id)
  7  );

Table created.
```



```
SQL> CREATE TABLE "Artist" (
  2      artist_id INT PRIMARY KEY,
  3      country VARCHAR(10),
  4      artist_name VARCHAR(15)
  5  );

Table created.

SQL> CREATE TABLE "Releases" (
  2      artist_id INT,
  3      album_id INT,
  4      FOREIGN KEY (artist_id) REFERENCES "Artist"(artist_id),
  5      FOREIGN KEY (album_id) REFERENCES "Album"(album_id),
  6      PRIMARY KEY (artist_id, album_id)
  7  );

Table created.

SQL> CREATE TABLE "PlaylistTrack" (
  2      playlist_id INT,
  3      track_id INT,
  4      FOREIGN KEY (playlist_id) REFERENCES Playlist(playlist_id),
  5      FOREIGN KEY (track_id) REFERENCES Track(track_id),
  6      PRIMARY KEY (playlist_id, track_id)
  7  );

Table created.
```

- **DESCRIPTION:**

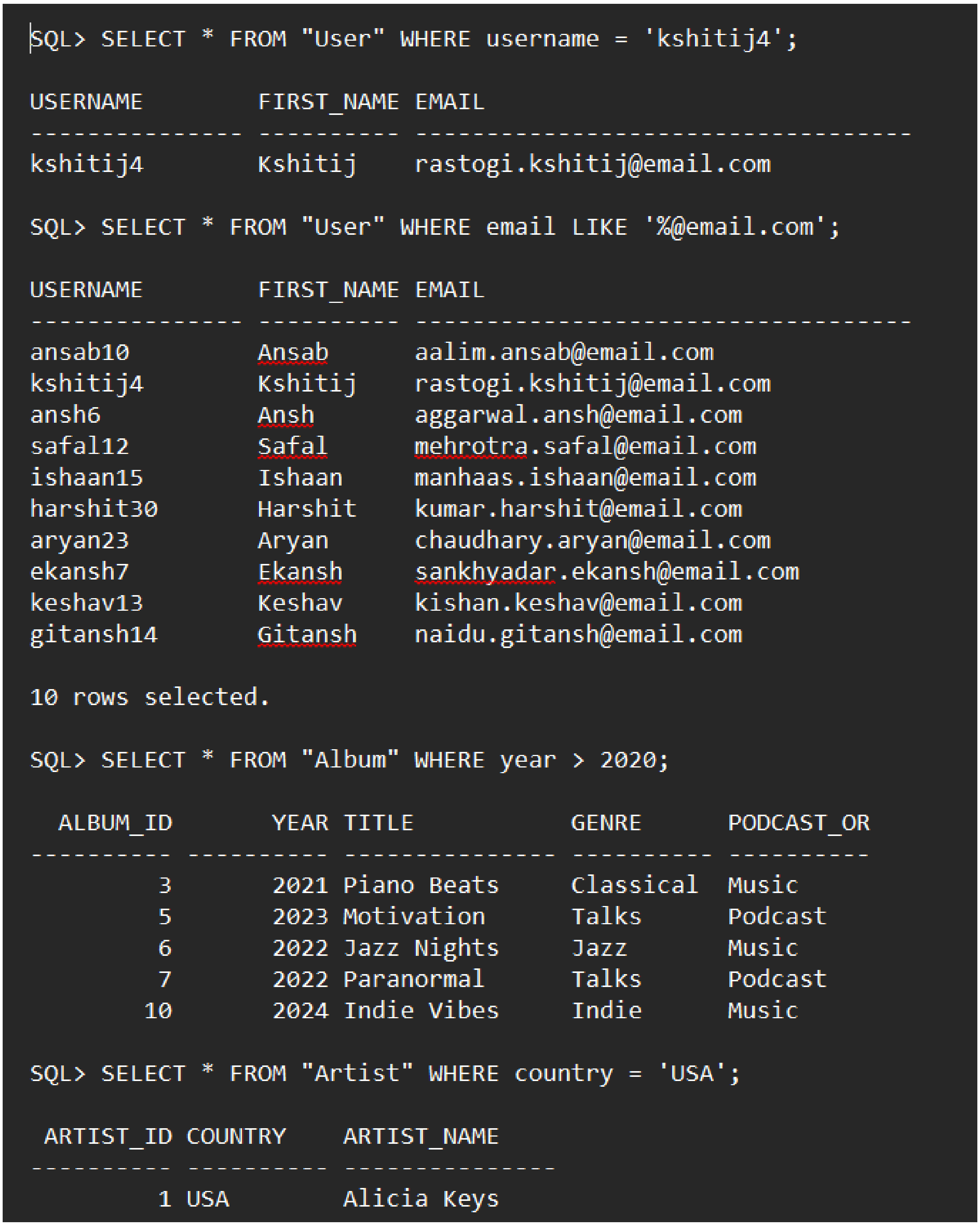


```
SQL> DESC "User";
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 USERNAME                                  NOT NULL VARCHAR2(15)
 FIRST_NAME                                         VARCHAR2(10)
 LAST_NAME                                          VARCHAR2(10)
 EMAIL                                              VARCHAR2(30)

SQL> DESC Playlist;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PLAYLIST_ID                               NOT NULL NUMBER(38)
 PLAYLIST_NAME                                      VARCHAR2(15)
 TYPE                                               VARCHAR2(10)
 CREATOR_USERNAME                                   VARCHAR2(10)

SQL> DESC "Album";
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ALBUM_ID                                  NOT NULL NUMBER(38)
 YEAR                                               NUMBER(38)
 TITLE                                              VARCHAR2(15)
 GENRE                                              VARCHAR2(10)
 PODCAST_OR_MUSIC                                   VARCHAR2(10)

SQL> DESC Track;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TRACK_ID                                  NOT NULL NUMBER(38)
 TRACK_NAME                                         VARCHAR2(15)
 DURATION                                           NUMBER(38)
 ALBUM_ID                                           NUMBER(38)

SQL> DESC "Artist";
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ARTIST_ID                                 NOT NULL NUMBER(38)
 COUNTRY                                            VARCHAR2(10)
 ARTIST_NAME                                        VARCHAR2(15)
```

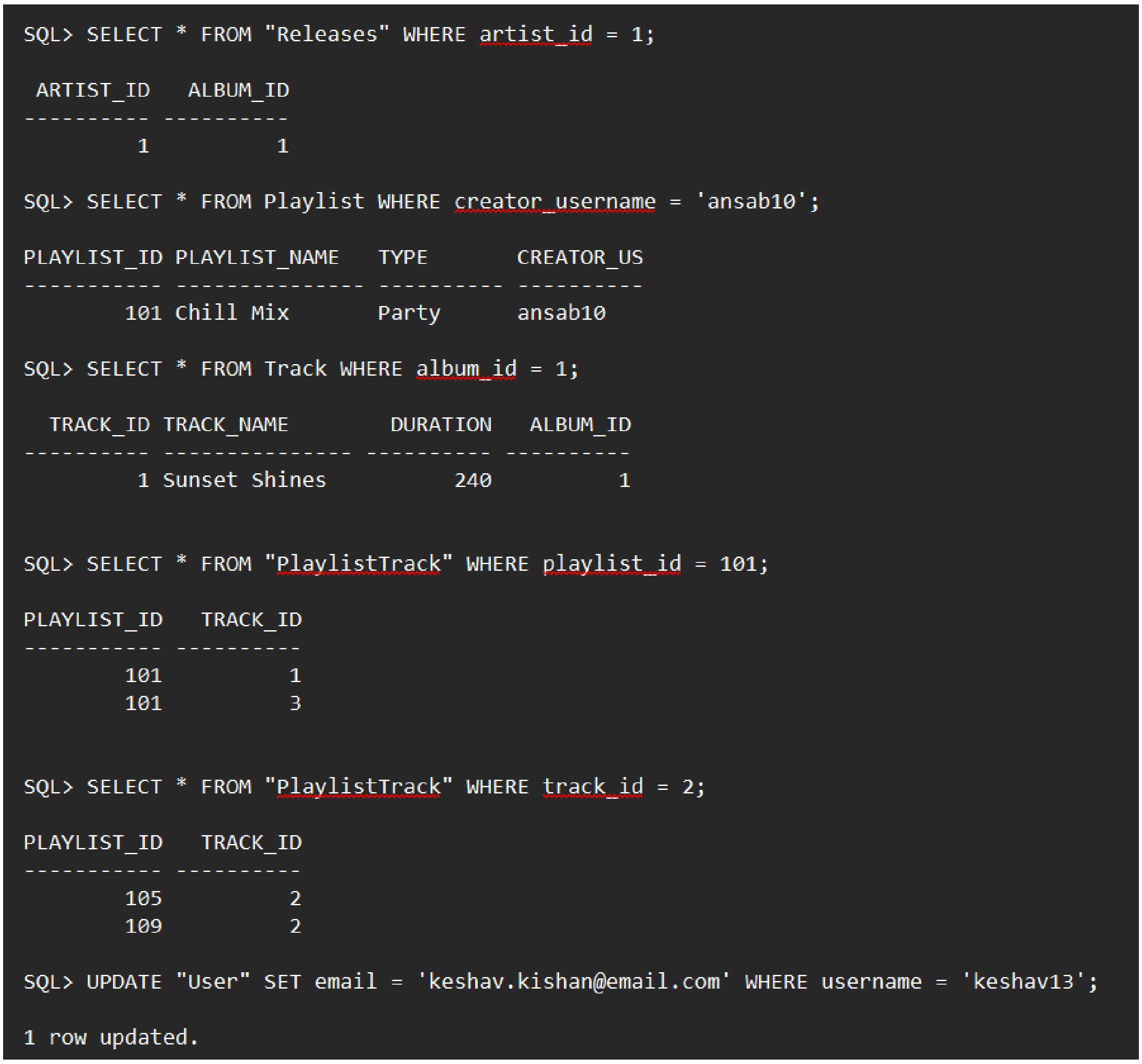
```
SQL> DESC "Releases";
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ARTIST_ID                                 NOT NULL NUMBER(38)
 ALBUM_ID                                  NOT NULL NUMBER(38)

SQL> DESC "PlaylistTrack";
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PLAYLIST_ID                               NOT NULL NUMBER(38)
 TRACK_ID                                  NOT NULL NUMBER(38)
```

- **<u>DML COMMANDS:</u>**

```
SQL> SELECT * FROM "PlaylistTrack" WHERE track_id = 2;

PLAYLIST_ID   TRACK_ID
----------- ----------
        105          2
        109          2

SQL> UPDATE "User" SET email = 'keshav.kishan@email.com' WHERE username = 'keshav13';

1 row updated.

SQL> UPDATE "Album" SET genre = 'Rock' WHERE album_id = 1;

1 row updated.

SQL> UPDATE "Artist" SET country = 'UK' WHERE artist_id = 1;

1 row updated.

SQL> UPDATE Playlist SET playlist_name = 'LoFi Beats' WHERE playlist_id = 104;

1 row updated.

SQL> UPDATE Track SET duration = 300 WHERE track_id = 1;

1 row updated.
```

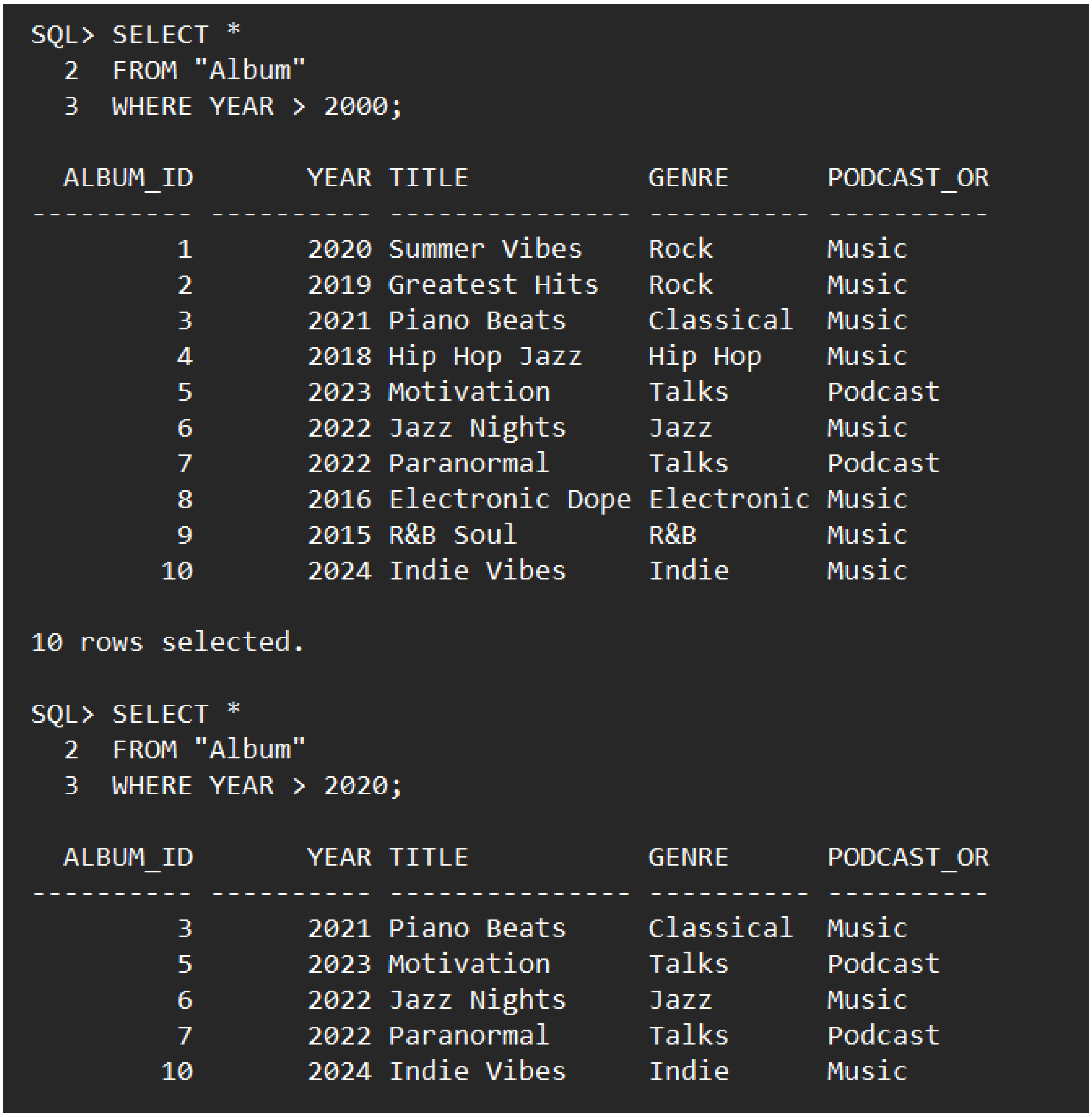


```
SQL> SELECT * FROM "User" WHERE username = 'kshitij4';

USERNAME        FIRST_NAME EMAIL
--------------- ---------- -----------------------------------
kshitij4        Kshitij    rastogi.kshitij@email.com

SQL> SELECT * FROM "User" WHERE email LIKE '%@email.com';

USERNAME        FIRST_NAME EMAIL
--------------- ---------- -----------------------------------
ansab10         Ansab      aalim.ansab@email.com
kshitij4        Kshitij    rastogi.kshitij@email.com
ansh6           Ansh       aggarwal.ansh@email.com
safal12         Safal      mehrotra.safal@email.com
ishaan15        Ishaan     manhaas.ishaan@email.com
harshit30       Harshit    kumar.harshit@email.com
aryan23         Aryan      chaudhary.aryan@email.com
ekansh7         Ekansh     sankhyadar.ekansh@email.com
keshav13        Keshav     kishan.keshav@email.com
gitansh14       Gitansh    naidu.gitansh@email.com

10 rows selected.

SQL> SELECT * FROM "Album" WHERE year > 2020;

  ALBUM_ID       YEAR TITLE           GENRE      PODCAST_OR
---------- ---------- --------------- ---------- ----------
         3       2021 Piano Beats     Classical  Music
         5       2023 Motivation      Talks      Podcast
         6       2022 Jazz Nights     Jazz       Music
         7       2022 Paranormal      Talks      Podcast
        10       2024 Indie Vibes     Indie      Music

SQL> SELECT * FROM "Artist" WHERE country = 'USA';

 ARTIST_ID COUNTRY    ARTIST_NAME
---------- ---------- ---------------
         1 USA        Alicia Keys
```

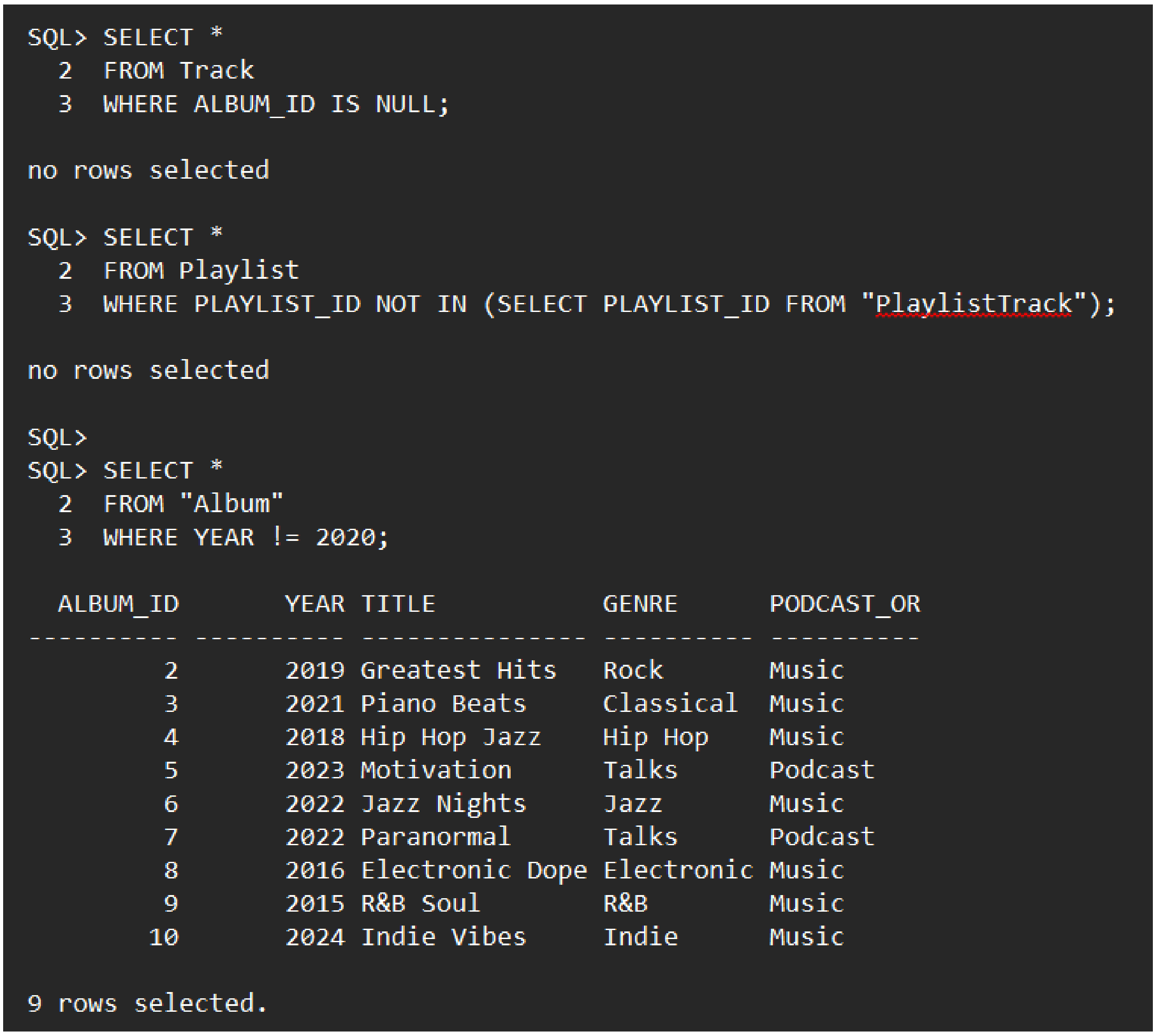


```
SQL> SELECT * FROM "Releases" WHERE artist_id = 1;

 ARTIST_ID   ALBUM_ID
---------- ----------
         1          1

SQL> SELECT * FROM Playlist WHERE creator_username = 'ansab10';

PLAYLIST_ID PLAYLIST_NAME   TYPE       CREATOR_US
----------- --------------- ---------- ----------
        101 Chill Mix       Party      ansab10

SQL> SELECT * FROM Track WHERE album_id = 1;

  TRACK_ID TRACK_NAME        DURATION   ALBUM_ID
---------- --------------- ---------- ----------
         1 Sunset Shines          240          1


SQL> SELECT * FROM "PlaylistTrack" WHERE playlist_id = 101;

PLAYLIST_ID   TRACK_ID
----------- ----------
        101          1
        101          3


SQL> SELECT * FROM "PlaylistTrack" WHERE track_id = 2;

PLAYLIST_ID   TRACK_ID
----------- ----------
        105          2
        109          2

SQL> UPDATE "User" SET email = 'keshav.kishan@email.com' WHERE username = 'keshav13';

1 row updated.
```

- **OPERATERS:**

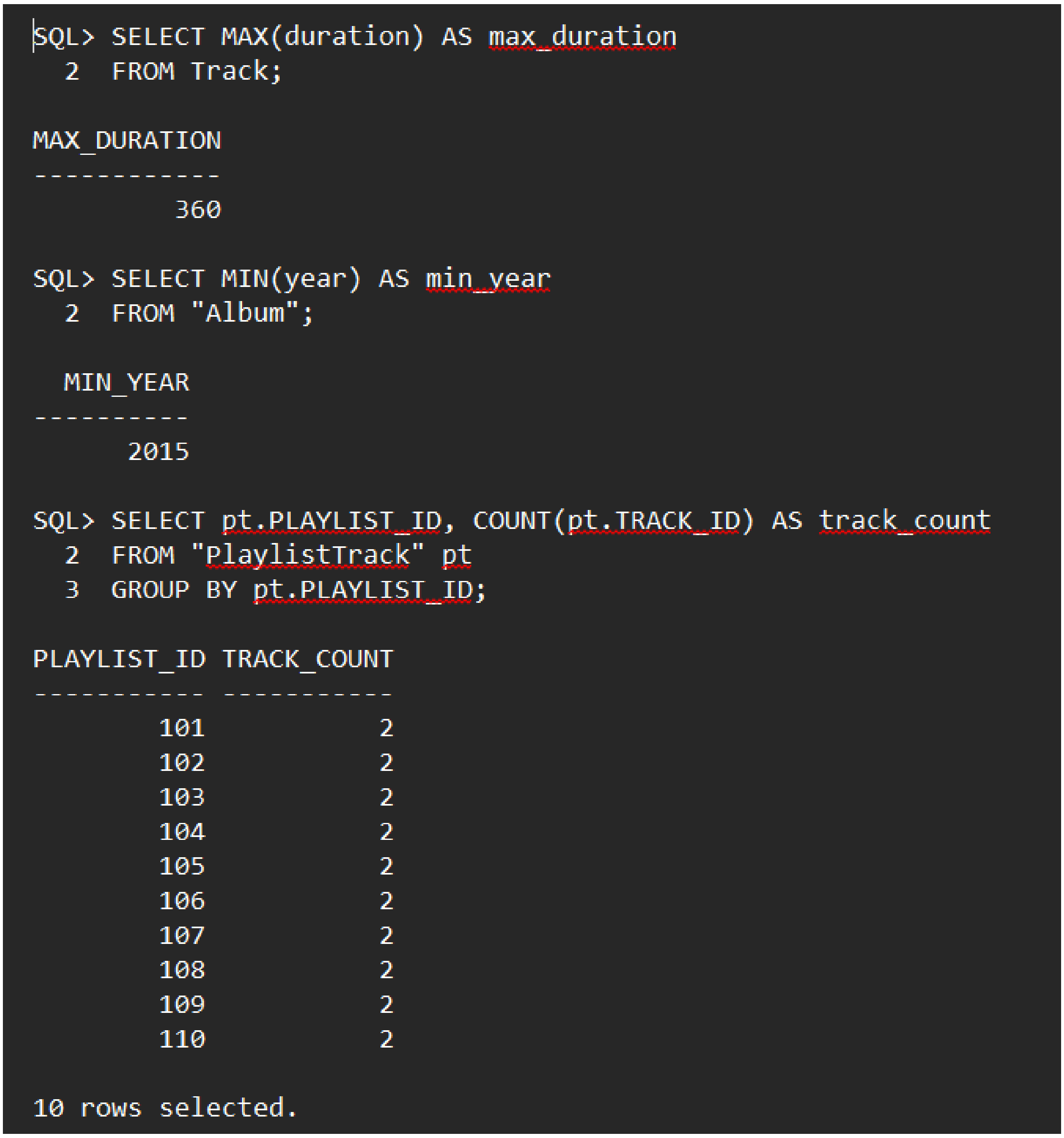


```
SQL> SELECT *
  2  FROM "Album"
  3  WHERE YEAR > 2000;

  ALBUM_ID       YEAR TITLE           GENRE      PODCAST_OR
---------- ---------- --------------- ---------- ----------
         1       2020 Summer Vibes    Rock       Music
         2       2019 Greatest Hits   Rock       Music
         3       2021 Piano Beats     Classical  Music
         4       2018 Hip Hop Jazz    Hip Hop    Music
         5       2023 Motivation      Talks      Podcast
         6       2022 Jazz Nights     Jazz       Music
         7       2022 Paranormal      Talks      Podcast
         8       2016 Electronic Dope Electronic Music
         9       2015 R&B Soul        R&B        Music
        10       2024 Indie Vibes     Indie      Music

10 rows selected.

SQL> SELECT *
  2  FROM "Album"
  3  WHERE YEAR > 2020;

  ALBUM_ID       YEAR TITLE           GENRE      PODCAST_OR
---------- ---------- --------------- ---------- ----------
         3       2021 Piano Beats     Classical  Music
         5       2023 Motivation      Talks      Podcast
         6       2022 Jazz Nights     Jazz       Music
         7       2022 Paranormal      Talks      Podcast
        10       2024 Indie Vibes     Indie      Music
```

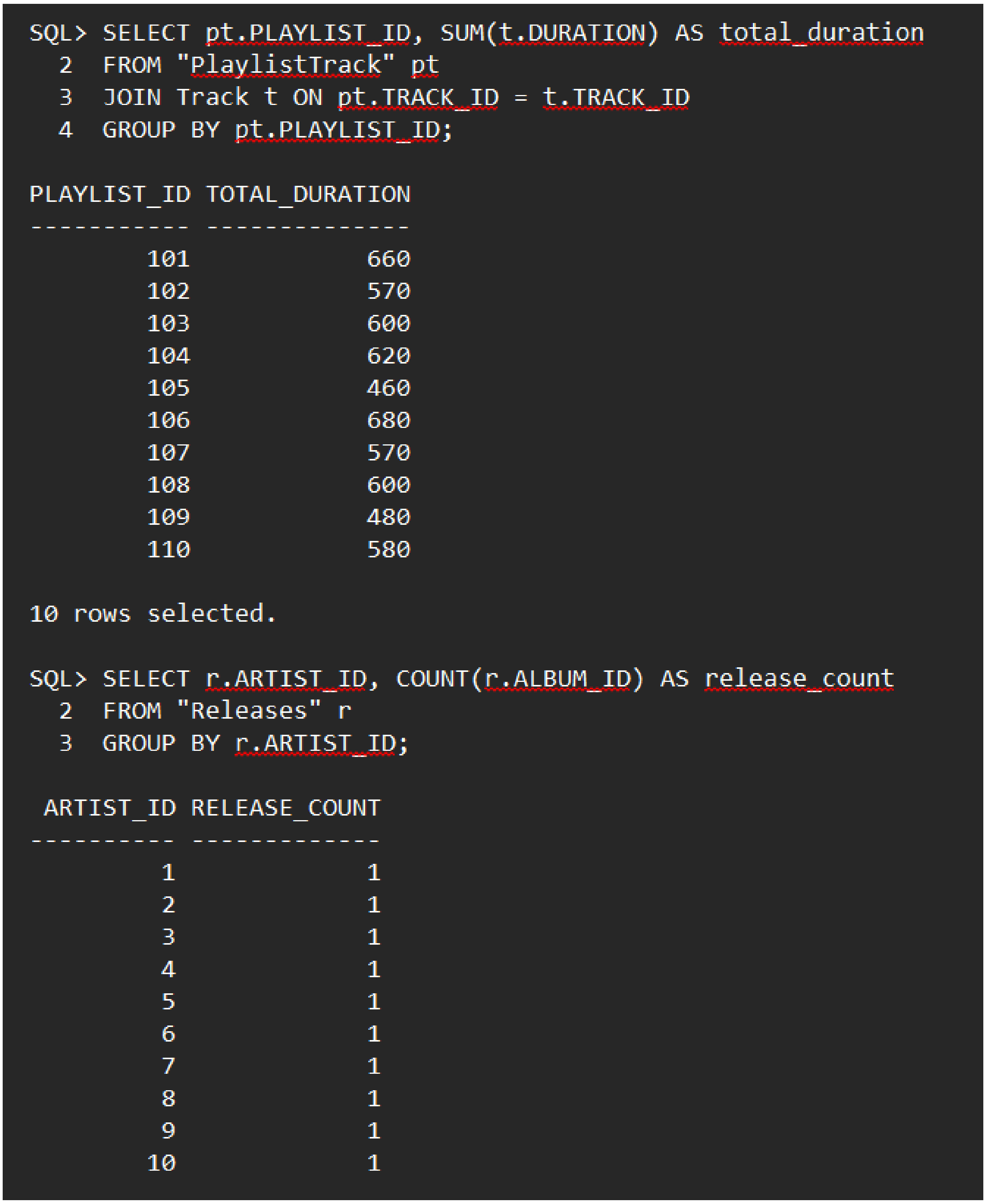


```
SQL> SELECT *
  2  FROM Track
  3  WHERE ALBUM_ID IS NULL;

no rows selected

SQL> SELECT *
  2  FROM Playlist
  3  WHERE PLAYLIST_ID NOT IN (SELECT PLAYLIST_ID FROM "PlaylistTrack");

no rows selected

SQL>
SQL> SELECT *
  2  FROM "Album"
  3  WHERE YEAR != 2020;

  ALBUM_ID       YEAR TITLE           GENRE      PODCAST_OR
---------- ---------- --------------- ---------- ----------
         2       2019 Greatest Hits   Rock       Music
         3       2021 Piano Beats     Classical  Music
         4       2018 Hip Hop Jazz    Hip Hop    Music
         5       2023 Motivation      Talks      Podcast
         6       2022 Jazz Nights     Jazz       Music
         7       2022 Paranormal      Talks      Podcast
         8       2016 Electronic Dope Electronic Music
         9       2015 R&B Soul        R&B        Music
        10       2024 Indie Vibes     Indie      Music

9 rows selected.
```

- **NUMERIC OPERATIONS:**

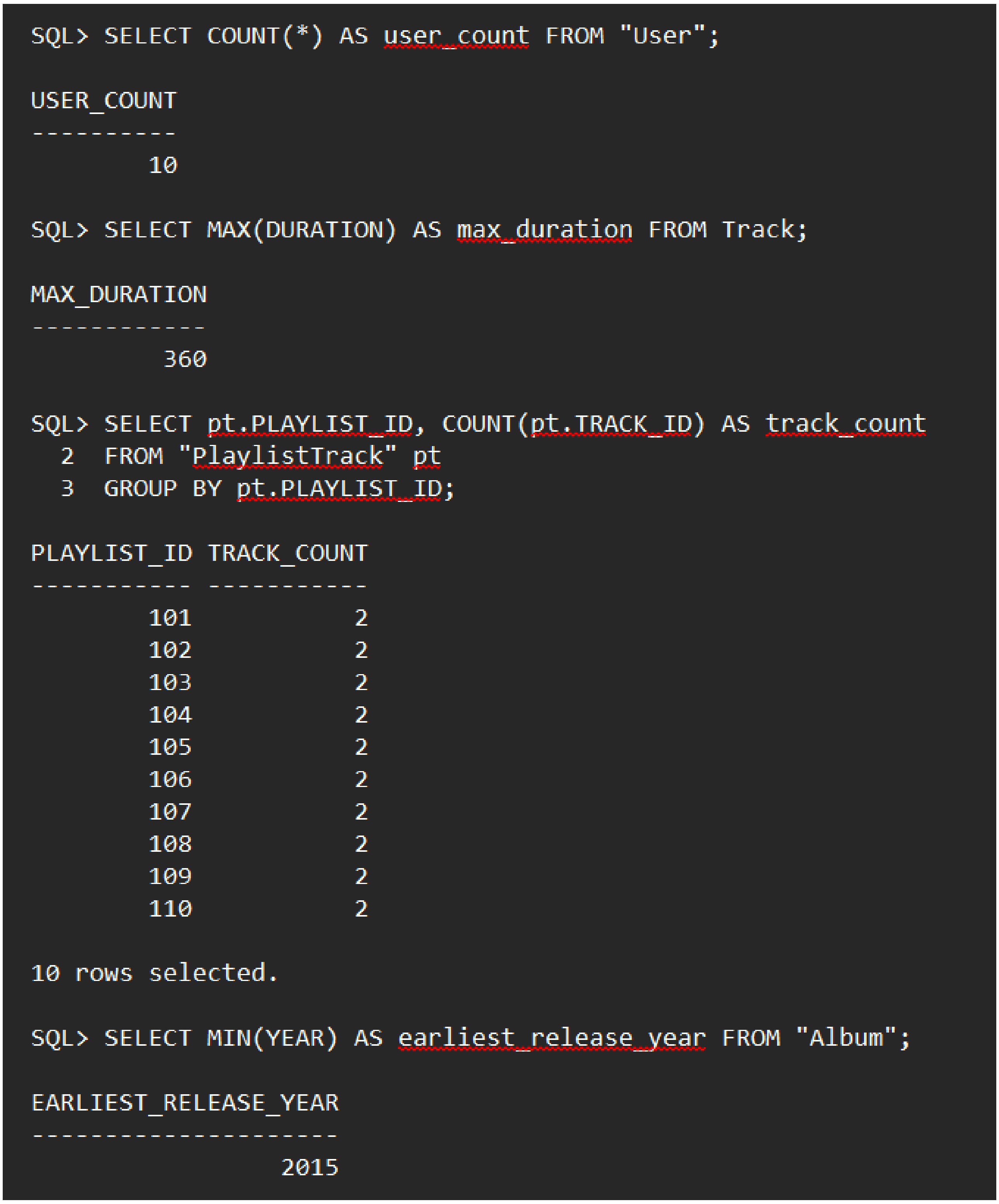


```
SQL> SELECT MAX(duration) AS max_duration
  2  FROM Track;

MAX_DURATION
------------
         360

SQL> SELECT MIN(year) AS min_year
  2  FROM "Album";

  MIN_YEAR
----------
      2015

SQL> SELECT pt.PLAYLIST_ID, COUNT(pt.TRACK_ID) AS track_count
  2  FROM "PlaylistTrack" pt
  3  GROUP BY pt.PLAYLIST_ID;

PLAYLIST_ID TRACK_COUNT
----------- -----------
        101           2
        102           2
        103           2
        104           2
        105           2
        106           2
        107           2
        108           2
        109           2
        110           2

10 rows selected.
```

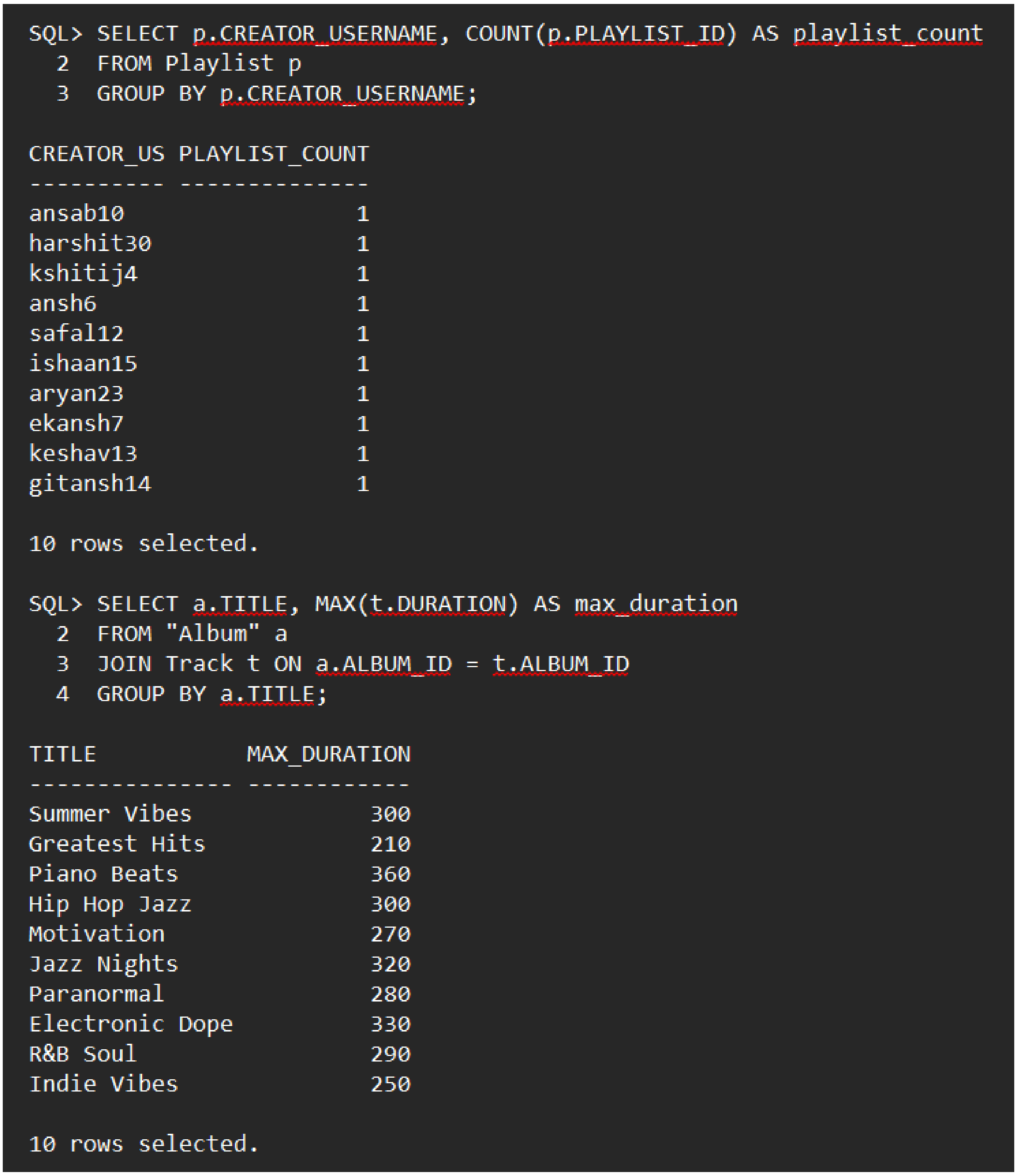


```
SQL> SELECT pt.PLAYLIST_ID, SUM(t.DURATION) AS total_duration
     FROM "PlaylistTrack" pt
     JOIN Track t ON pt.TRACK_ID = t.TRACK_ID
     GROUP BY pt.PLAYLIST_ID;

PLAYLIST_ID TOTAL_DURATION
----------- --------------
        101            660
        102            570
        103            600
        104            620
        105            460
        106            680
        107            570
        108            600
        109            480
        110            580

10 rows selected.

SQL> SELECT r.ARTIST_ID, COUNT(r.ALBUM_ID) AS release_count
     FROM "Releases" r
     GROUP BY r.ARTIST_ID;

 ARTIST_ID RELEASE_COUNT
---------- -------------
         1             1
         2             1
         3             1
         4             1
         5             1
         6             1
         7             1
         8             1
         9             1
        10             1
```

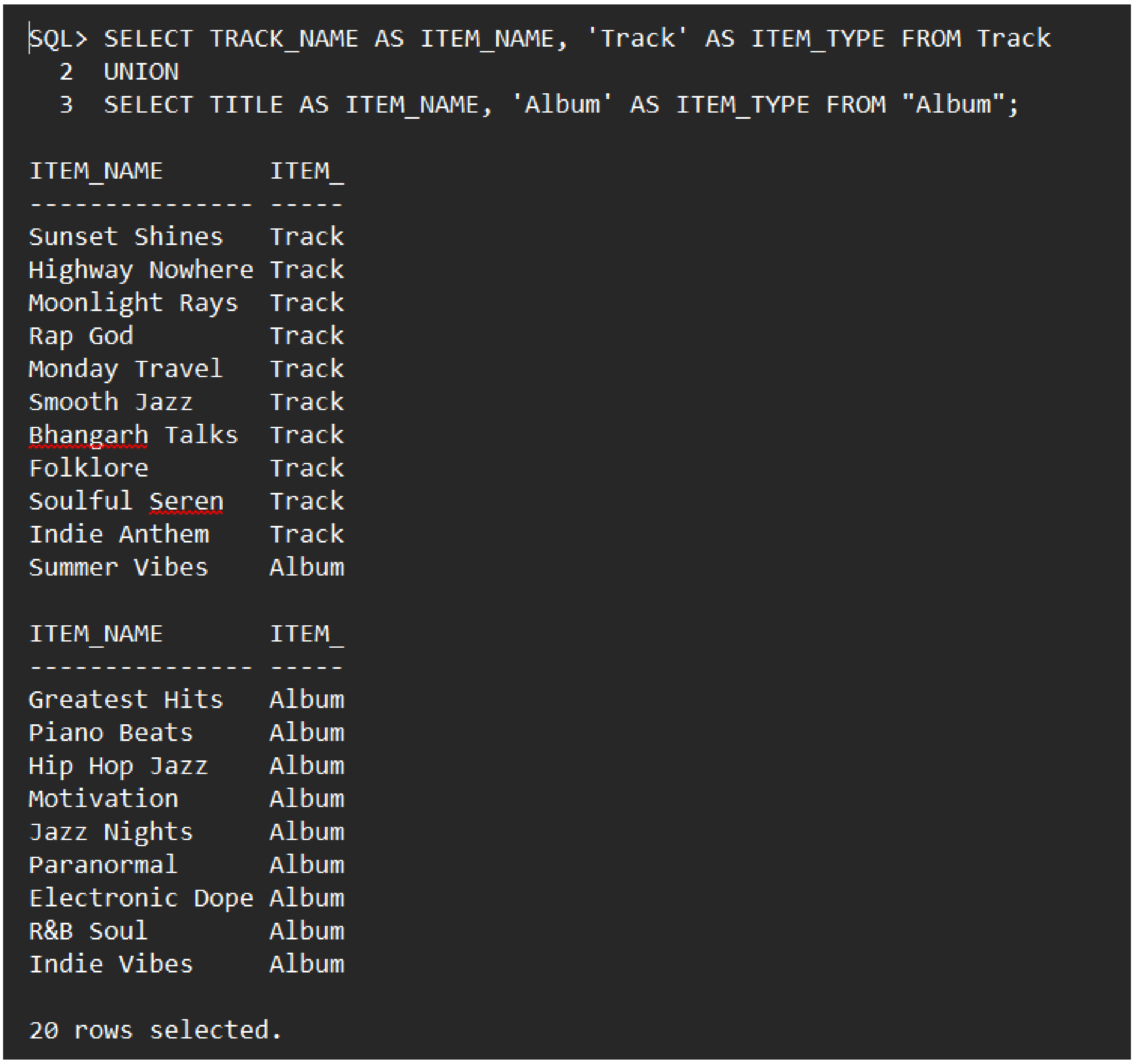
- **COUNT:**



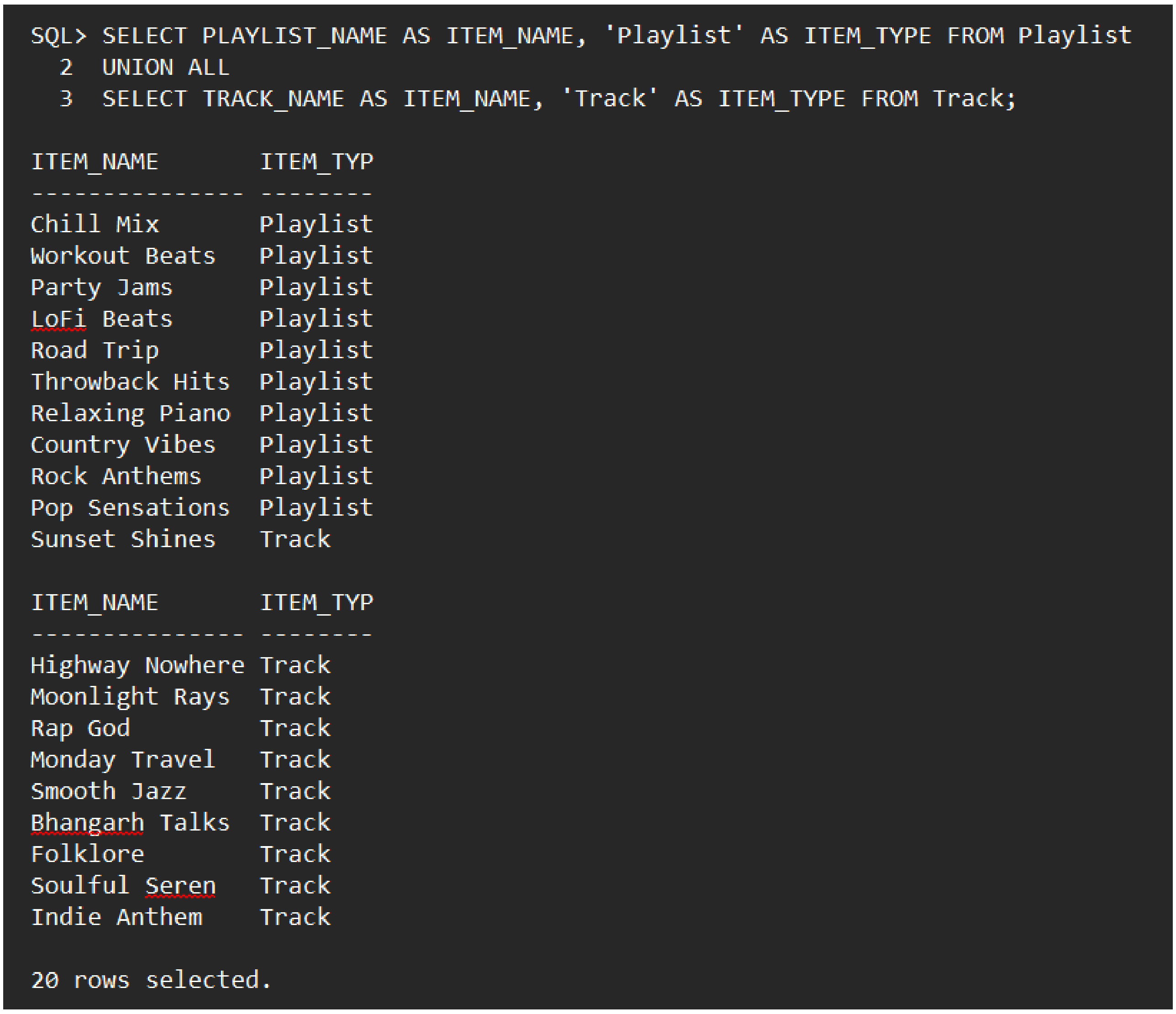
```
SQL> SELECT COUNT(*) AS user_count FROM "User";

USER_COUNT
----------
        10

SQL> SELECT MAX(DURATION) AS max_duration FROM Track;

MAX_DURATION
------------
         360

SQL> SELECT pt.PLAYLIST_ID, COUNT(pt.TRACK_ID) AS track_count
  2  FROM "PlaylistTrack" pt
  3  GROUP BY pt.PLAYLIST_ID;

PLAYLIST_ID TRACK_COUNT
----------- -----------
        101           2
        102           2
        103           2
        104           2
        105           2
        106           2
        107           2
        108           2
        109           2
        110           2

10 rows selected.

SQL> SELECT MIN(YEAR) AS earliest_release_year FROM "Album";

EARLIEST_RELEASE_YEAR
---------------------
                 2015
```


```
SQL> SELECT p.CREATOR_USERNAME, COUNT(p.PLAYLIST_ID) AS playlist_count
  2  FROM Playlist p
  3  GROUP BY p.CREATOR_USERNAME;

CREATOR_US PLAYLIST_COUNT
---------- --------------
ansab10                 1
harshit30               1
kshitij4                1
ansh6                   1
safal12                 1
ishaan15                1
aryan23                 1
ekansh7                 1
keshav13                1
gitansh14               1

10 rows selected.

SQL> SELECT a.TITLE, MAX(t.DURATION) AS max_duration
  2  FROM "Album" a
  3  JOIN Track t ON a.ALBUM_ID = t.ALBUM_ID
  4  GROUP BY a.TITLE;

TITLE           MAX_DURATION
--------------- ------------
Summer Vibes             300
Greatest Hits            210
Piano Beats              360
Hip Hop Jazz             300
Motivation               270
Jazz Nights              320
Paranormal               280
Electronic Dope          330
R&B Soul                 290
Indie Vibes              250

10 rows selected.
```

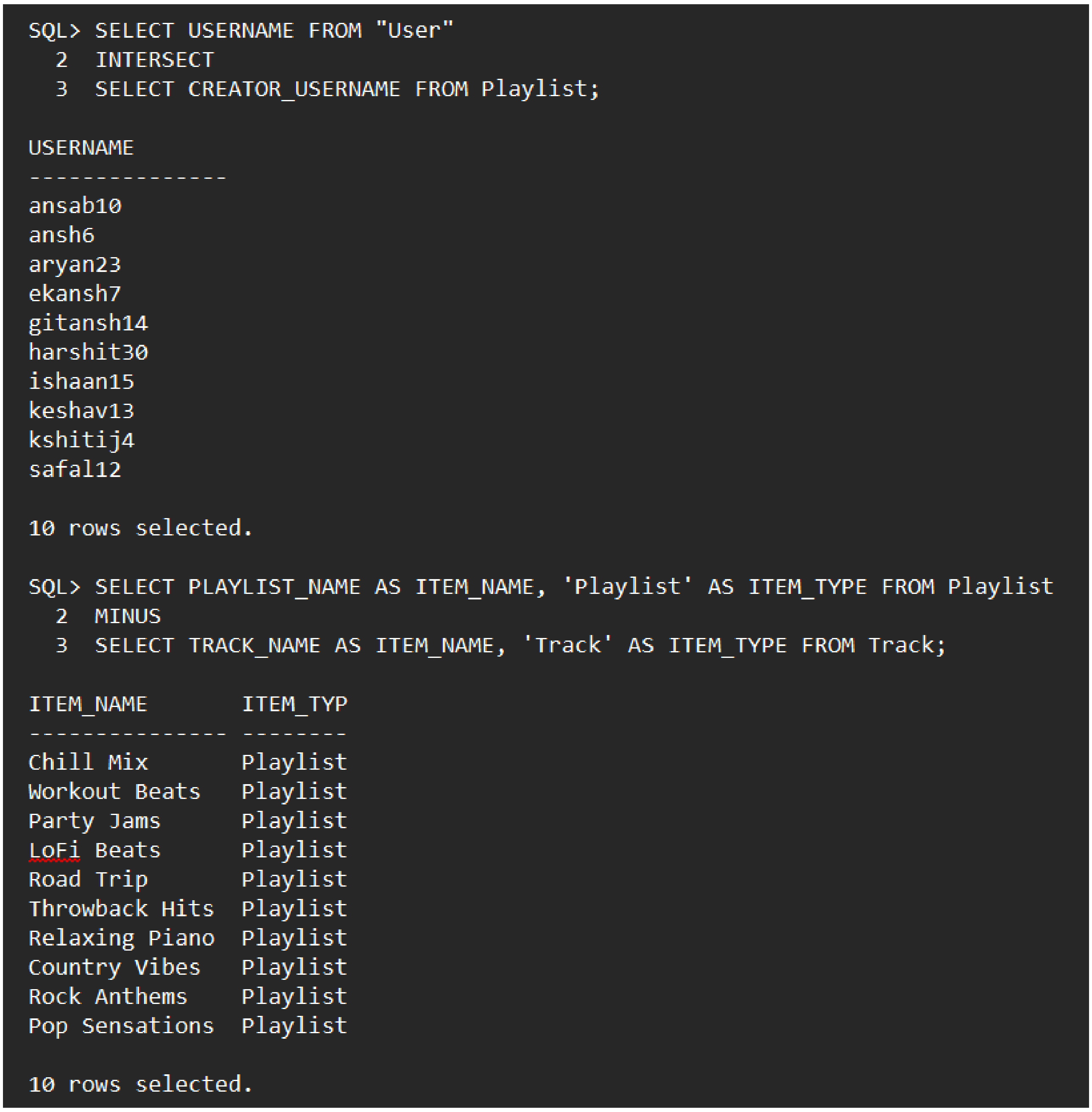
- **SET OPERATORS:**



```
SQL> SELECT TRACK_NAME AS ITEM_NAME, 'Track' AS ITEM_TYPE FROM Track
  2  UNION
  3  SELECT TITLE AS ITEM_NAME, 'Album' AS ITEM_TYPE FROM "Album";

ITEM_NAME       ITEM_
--------------- -----
Sunset Shines   Track
Highway Nowhere Track
Moonlight Rays  Track
Rap God         Track
Monday Travel   Track
Smooth Jazz     Track
Bhangarh Talks  Track
Folklore        Track
Soulful Seren   Track
Indie Anthem    Track
Summer Vibes    Album

ITEM_NAME       ITEM_
--------------- -----
Greatest Hits   Album
Piano Beats     Album
Hip Hop Jazz    Album
Motivation      Album
Jazz Nights     Album
Paranormal      Album
Electronic Dope Album
R&B Soul        Album
Indie Vibes     Album

20 rows selected.
```

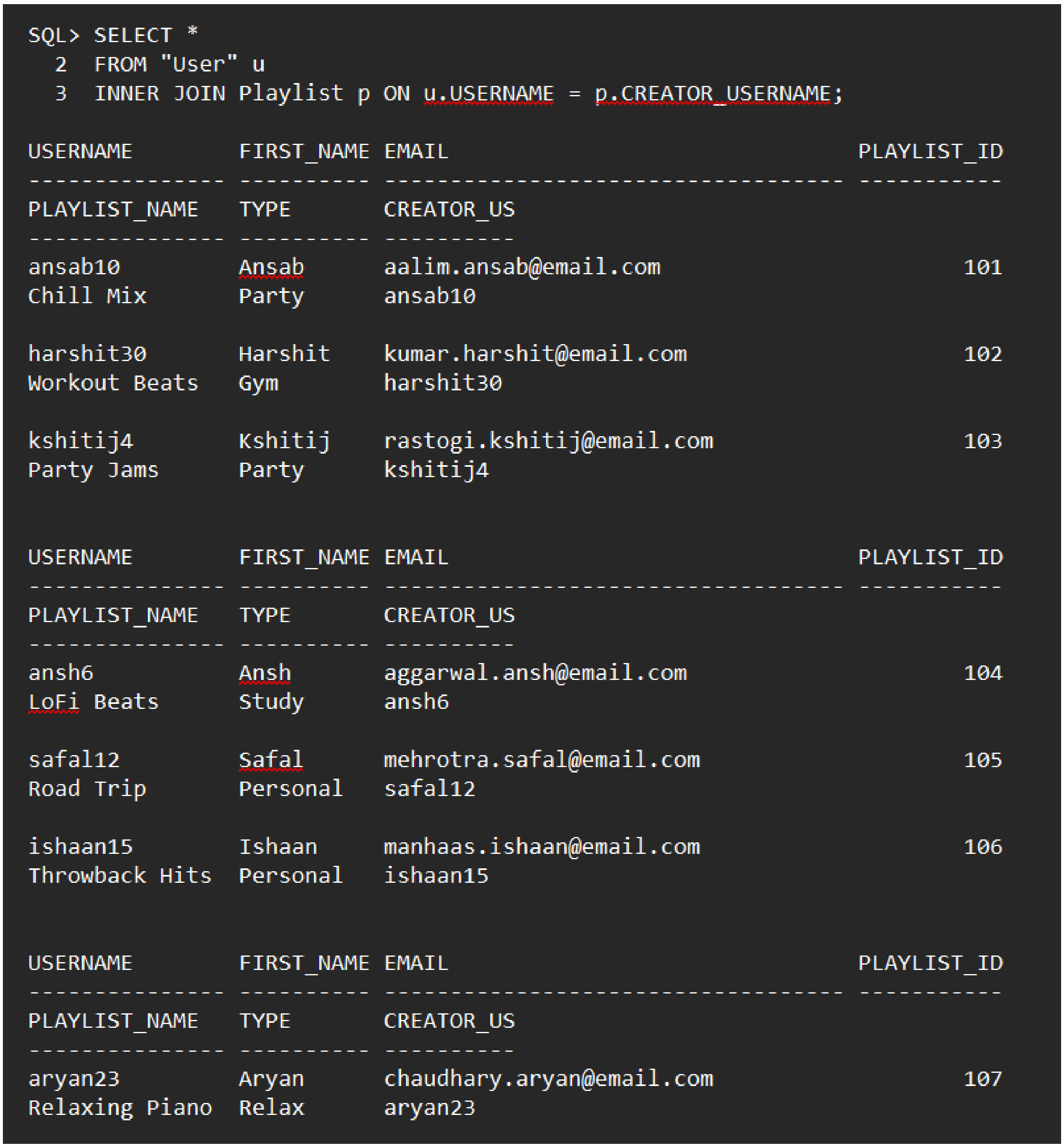


```
SQL> SELECT PLAYLIST_NAME AS ITEM_NAME, 'Playlist' AS ITEM_TYPE FROM Playlist
  2  UNION ALL
  3  SELECT TRACK_NAME AS ITEM_NAME, 'Track' AS ITEM_TYPE FROM Track;

ITEM_NAME       ITEM_TYP
--------------- --------
Chill Mix       Playlist
Workout Beats   Playlist
Party Jams      Playlist
LoFi Beats      Playlist
Road Trip       Playlist
Throwback Hits  Playlist
Relaxing Piano  Playlist
Country Vibes   Playlist
Rock Anthems    Playlist
Pop Sensations  Playlist
Sunset Shines   Track

ITEM_NAME       ITEM_TYP
--------------- --------
Highway Nowhere Track
Moonlight Rays  Track
Rap God         Track
Monday Travel   Track
Smooth Jazz     Track
Bhangarh Talks  Track
Folklore        Track
Soulful Seren   Track
Indie Anthem    Track

20 rows selected.
```

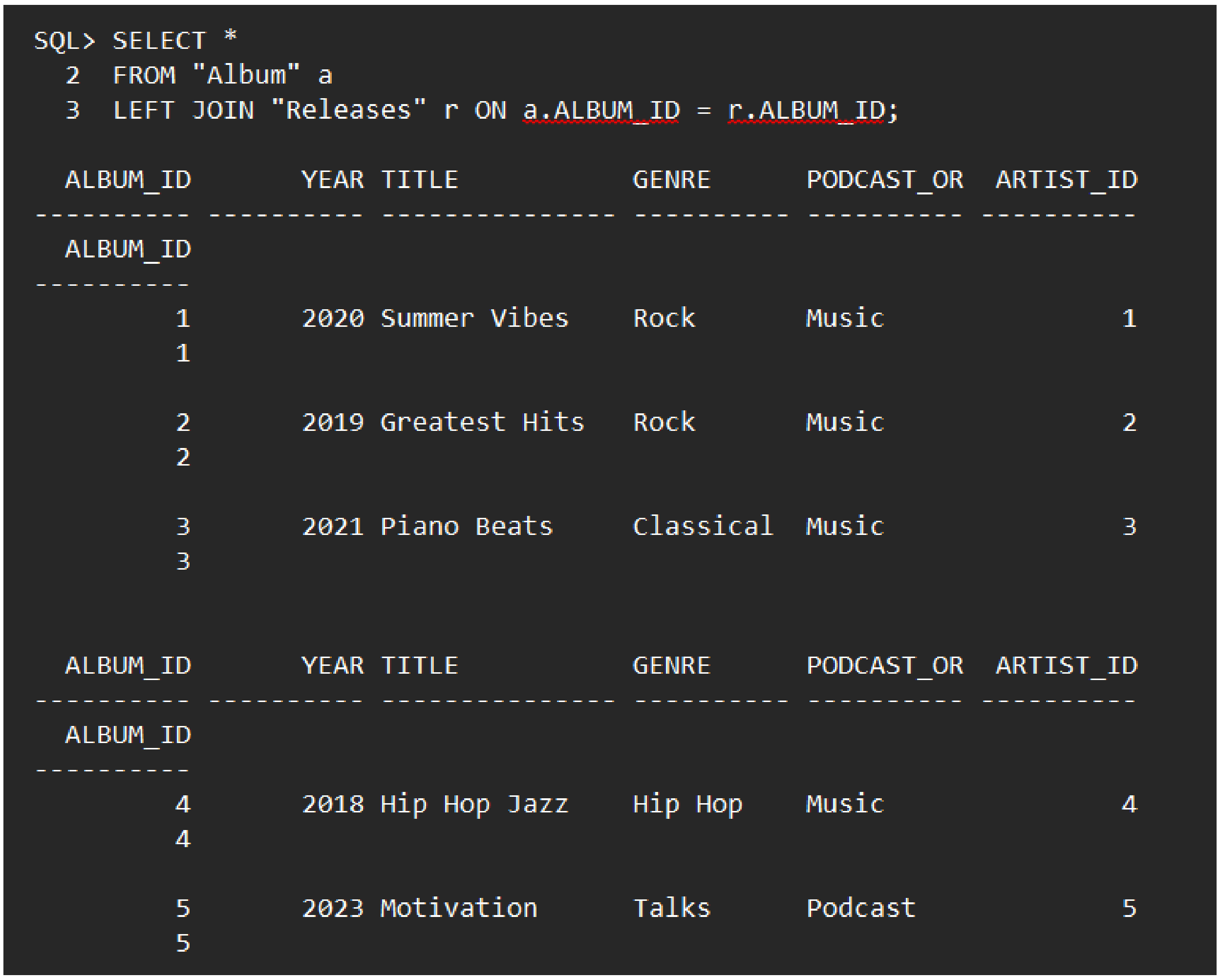


```
SQL> SELECT USERNAME FROM "User"
  2  INTERSECT
  3  SELECT CREATOR_USERNAME FROM Playlist;

USERNAME
---------------
ansab10
ansh6
aryan23
ekansh7
gitansh14
harshit30
ishaan15
keshav13
kshitij4
safal12

10 rows selected.

SQL> SELECT PLAYLIST_NAME AS ITEM_NAME, 'Playlist' AS ITEM_TYPE FROM Playlist
  2  MINUS
  3  SELECT TRACK_NAME AS ITEM_NAME, 'Track' AS ITEM_TYPE FROM Track;

ITEM_NAME       ITEM_TYP
--------------- --------
Chill Mix       Playlist
Workout Beats   Playlist
Party Jams      Playlist
LoFi Beats      Playlist
Road Trip       Playlist
Throwback Hits  Playlist
Relaxing Piano  Playlist
Country Vibes   Playlist
Rock Anthems    Playlist
Pop Sensations  Playlist

10 rows selected.
```

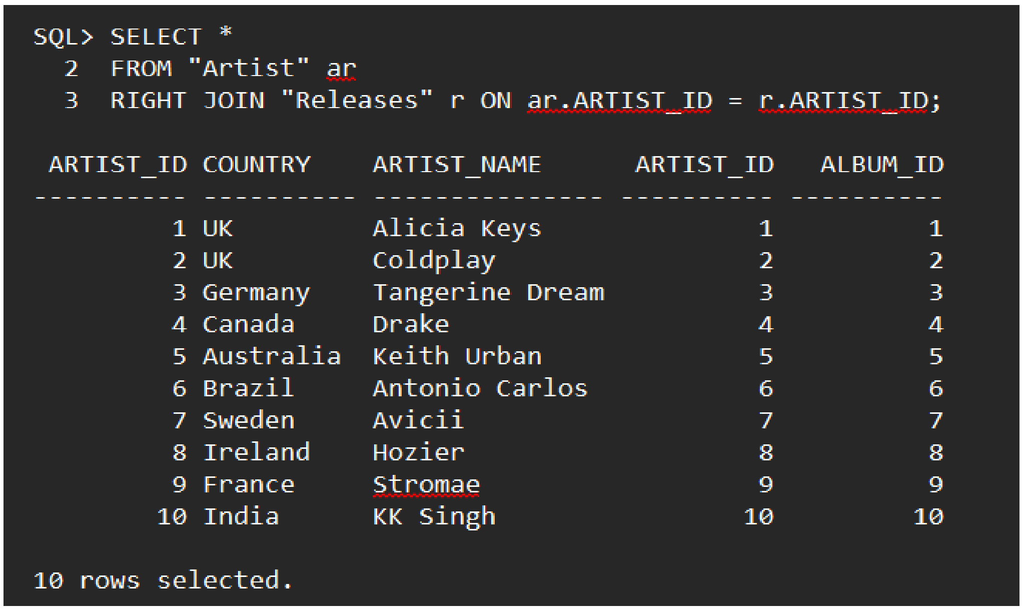
- **JOINS:**



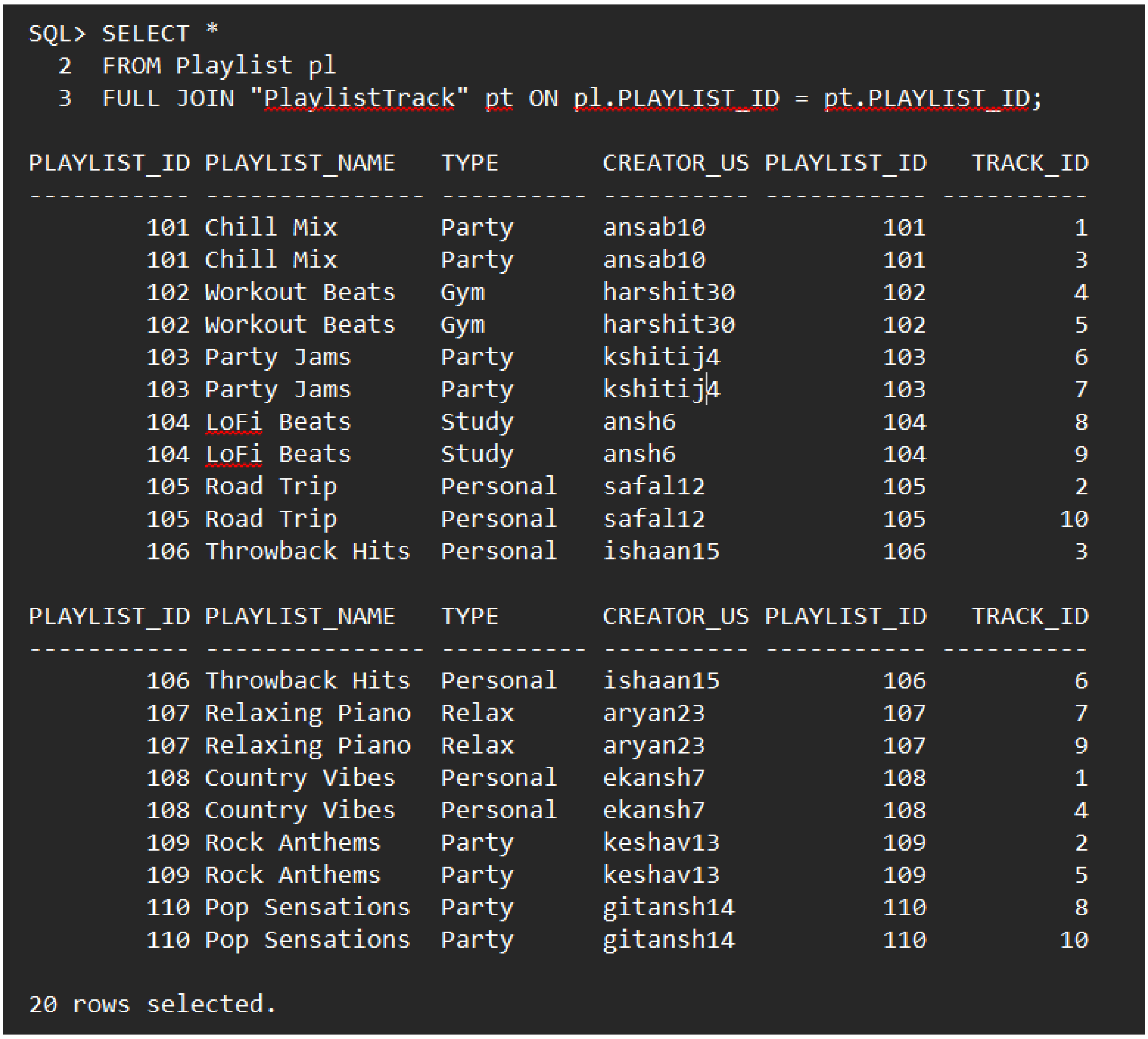
```
SQL> SELECT *
  2  FROM "User" u
  3  INNER JOIN Playlist p ON u.USERNAME = p.CREATOR_USERNAME;

USERNAME        FIRST_NAME EMAIL                                PLAYLIST_ID
--------------- ---------- ------------------------------------ -----------
PLAYLIST_NAME   TYPE       CREATOR_US
--------------- ---------- ----------
ansab10         Ansab      aalim.ansab@email.com                        101
Chill Mix       Party      ansab10

harshit30       Harshit    kumar.harshit@email.com                      102
Workout Beats   Gym        harshit30

kshitij4        Kshitij    rastogi.kshitij@email.com                    103
Party Jams      Party      kshitij4


USERNAME        FIRST_NAME EMAIL                                PLAYLIST_ID
--------------- ---------- ------------------------------------ -----------
PLAYLIST_NAME   TYPE       CREATOR_US
--------------- ---------- ----------
ansh6           Ansh       aggarwal.ansh@email.com                      104
LoFi Beats      Study      ansh6

safal12         Safal      mehrotra.safal@email.com                     105
Road Trip       Personal   safal12

ishaan15        Ishaan     manhaas.ishaan@email.com                     106
Throwback Hits  Personal   ishaan15


USERNAME        FIRST_NAME EMAIL                                PLAYLIST_ID
--------------- ---------- ------------------------------------ -----------
PLAYLIST_NAME   TYPE       CREATOR_US
--------------- ---------- ----------
aryan23         Aryan      chaudhary.aryan@email.com                    107
Relaxing Piano  Relax      aryan23
```



```
SQL> SELECT *
  2  FROM "Album" a
  3  LEFT JOIN "Releases" r ON a.ALBUM_ID = r.ALBUM_ID;

  ALBUM_ID       YEAR TITLE           GENRE      PODCAST_OR  ARTIST_ID
---------- ---------- --------------- ---------- ---------- ----------
  ALBUM_ID
----------
         1       2020 Summer Vibes    Rock       Music               1
         1

         2       2019 Greatest Hits   Rock       Music               2
         2

         3       2021 Piano Beats     Classical  Music               3
         3


  ALBUM_ID       YEAR TITLE           GENRE      PODCAST_OR  ARTIST_ID
---------- ---------- --------------- ---------- ---------- ----------
  ALBUM_ID
----------
         4       2018 Hip Hop Jazz    Hip Hop    Music               4
         4

         5       2023 Motivation      Talks      Podcast             5
         5
```

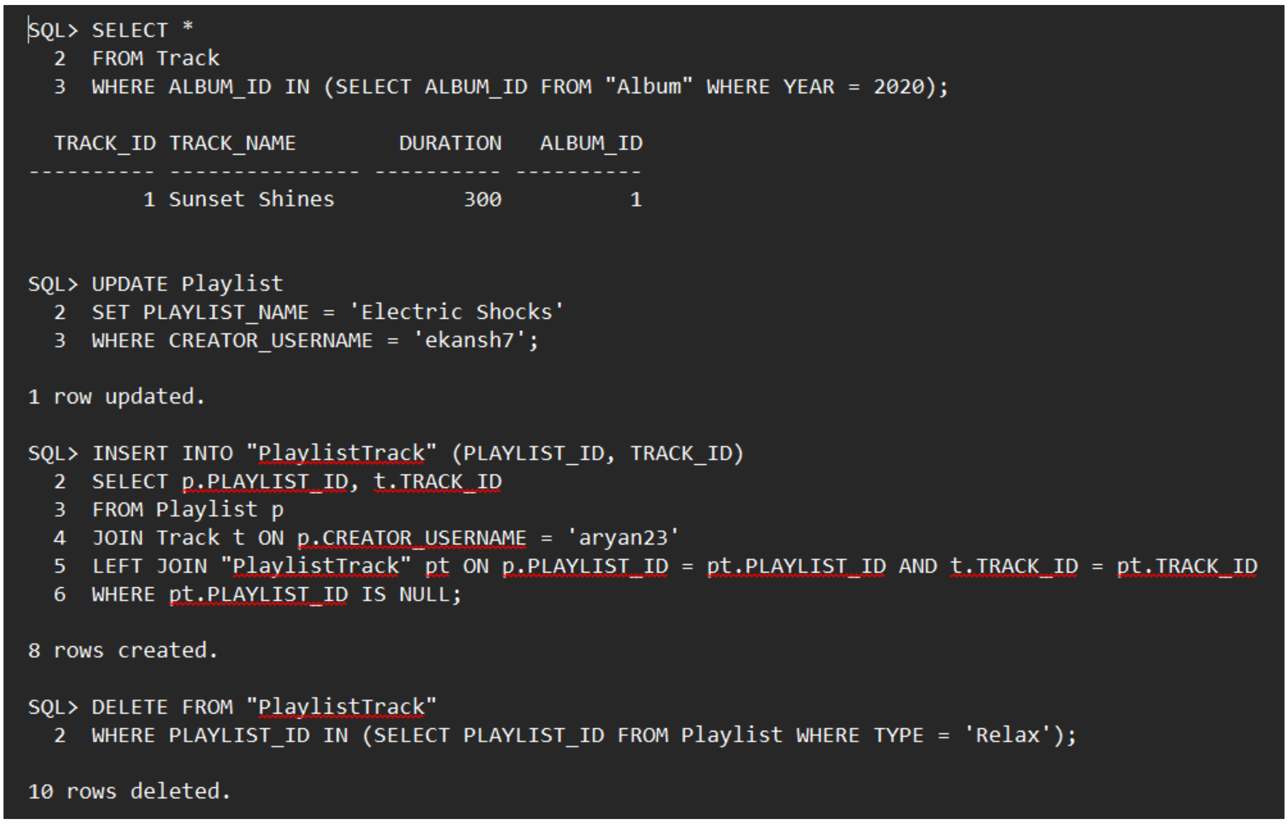


```
SQL> SELECT *
  2  FROM "Artist" ar
  3  RIGHT JOIN "Releases" r ON ar.ARTIST_ID = r.ARTIST_ID;

 ARTIST_ID COUNTRY    ARTIST_NAME      ARTIST_ID   ALBUM_ID
---------- ---------- --------------- ---------- ----------
         1 UK         Alicia Keys              1          1
         2 UK         Coldplay                 2          2
         3 Germany    Tangerine Dream          3          3
         4 Canada     Drake                    4          4
         5 Australia  Keith Urban              5          5
         6 Brazil     Antonio Carlos           6          6
         7 Sweden     Avicii                   7          7
         8 Ireland    Hozier                   8          8
         9 France     Stromae                  9          9
        10 India      KK Singh                10         10

10 rows selected.
```



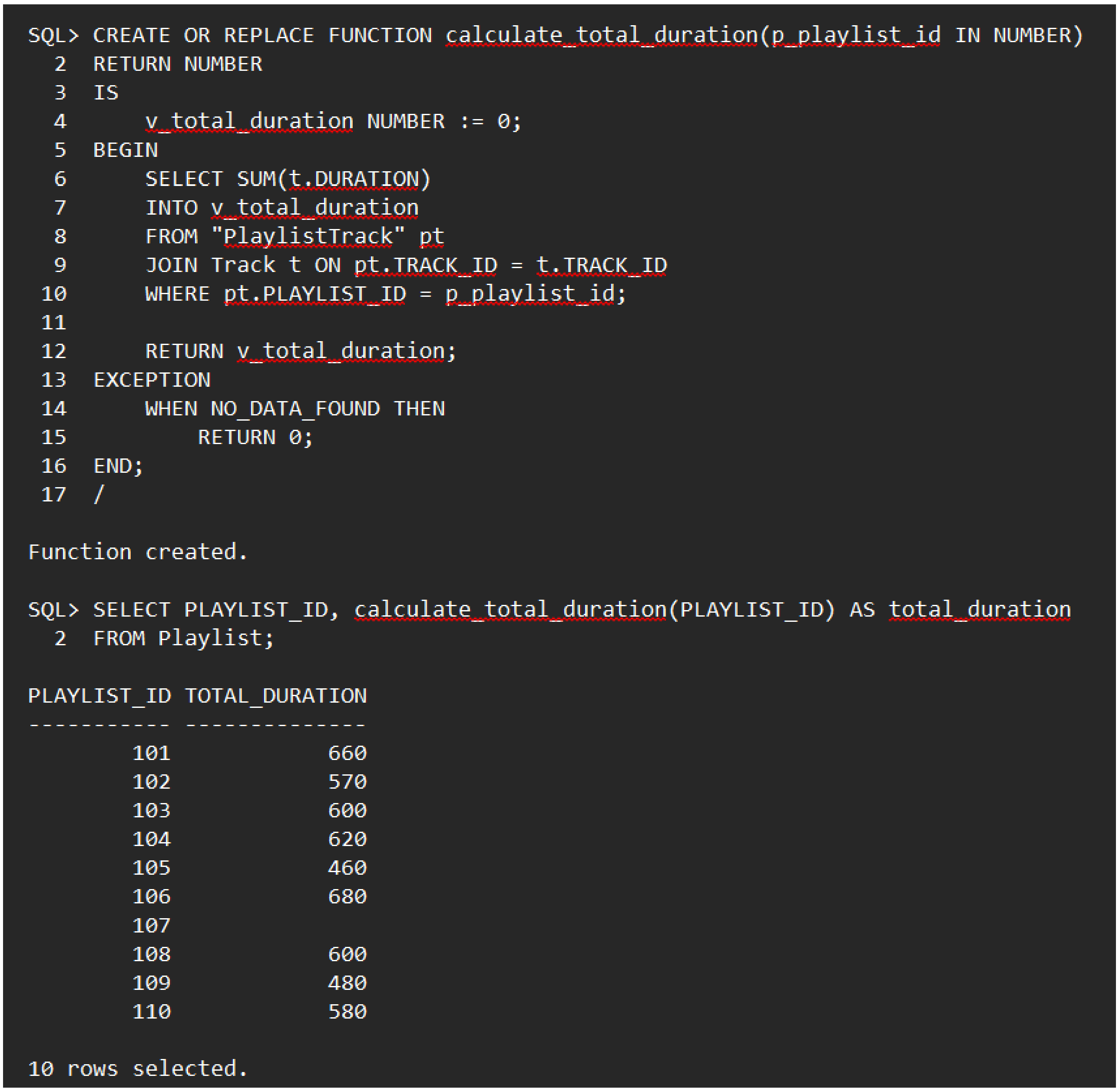
```
SQL> SELECT *
  2  FROM Playlist pl
  3  FULL JOIN "PlaylistTrack" pt ON pl.PLAYLIST_ID = pt.PLAYLIST_ID;

PLAYLIST_ID PLAYLIST_NAME   TYPE       CREATOR_US PLAYLIST_ID   TRACK_ID
----------- --------------- ---------- ---------- ----------- ----------
        101 Chill Mix       Party      ansab10            101          1
        101 Chill Mix       Party      ansab10            101          3
        102 Workout Beats   Gym        harshit30          102          4
        102 Workout Beats   Gym        harshit30          102          5
        103 Party Jams      Party      kshitij4           103          6
        103 Party Jams      Party      kshitij4           103          7
        104 LoFi Beats      Study      ansh6              104          8
        104 LoFi Beats      Study      ansh6              104          9
        105 Road Trip       Personal   safal12            105          2
        105 Road Trip       Personal   safal12            105         10
        106 Throwback Hits  Personal   ishaan15           106          3

PLAYLIST_ID PLAYLIST_NAME   TYPE       CREATOR_US PLAYLIST_ID   TRACK_ID
----------- --------------- ---------- ---------- ----------- ----------
        106 Throwback Hits  Personal   ishaan15           106          6
        107 Relaxing Piano  Relax      aryan23            107          7
        107 Relaxing Piano  Relax      aryan23            107          9
        108 Country Vibes   Personal   ekansh7            108          1
        108 Country Vibes   Personal   ekansh7            108          4
        109 Rock Anthems    Party      keshav13           109          2
        109 Rock Anthems    Party      keshav13           109          5
        110 Pop Sensations  Party      gitansh14          110          8
        110 Pop Sensations  Party      gitansh14          110         10

20 rows selected.
```

- **SUBQUERIES:**



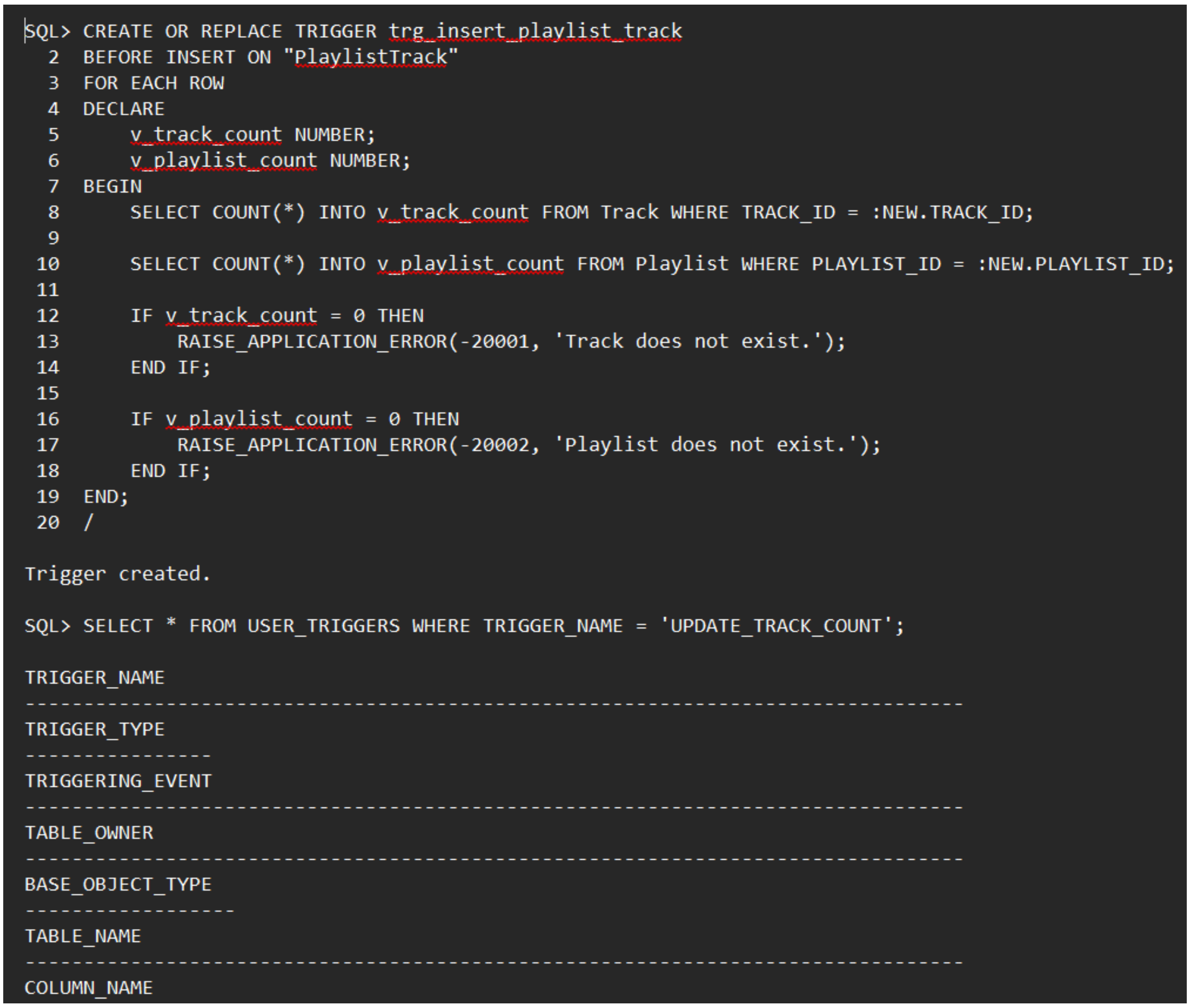
```
SQL> SELECT *
  2  FROM Track
  3  WHERE ALBUM_ID IN (SELECT ALBUM_ID FROM "Album" WHERE YEAR = 2020);

  TRACK_ID TRACK_NAME        DURATION   ALBUM_ID
---------- --------------- ---------- ----------
         1 Sunset Shines          300          1


SQL> UPDATE Playlist
  2  SET PLAYLIST_NAME = 'Electric Shocks'
  3  WHERE CREATOR_USERNAME = 'ekansh7';

1 row updated.

SQL> INSERT INTO "PlaylistTrack" (PLAYLIST_ID, TRACK_ID)
  2  SELECT p.PLAYLIST_ID, t.TRACK_ID
  3  FROM Playlist p
  4  JOIN Track t ON p.CREATOR_USERNAME = 'aryan23'
  5  LEFT JOIN "PlaylistTrack" pt ON p.PLAYLIST_ID = pt.PLAYLIST_ID AND t.TRACK_ID = pt.TRACK_ID
  6  WHERE pt.PLAYLIST_ID IS NULL;

8 rows created.

SQL> DELETE FROM "PlaylistTrack"
  2  WHERE PLAYLIST_ID IN (SELECT PLAYLIST_ID FROM Playlist WHERE TYPE = 'Relax');

10 rows deleted.
```

- **PL SQL:**



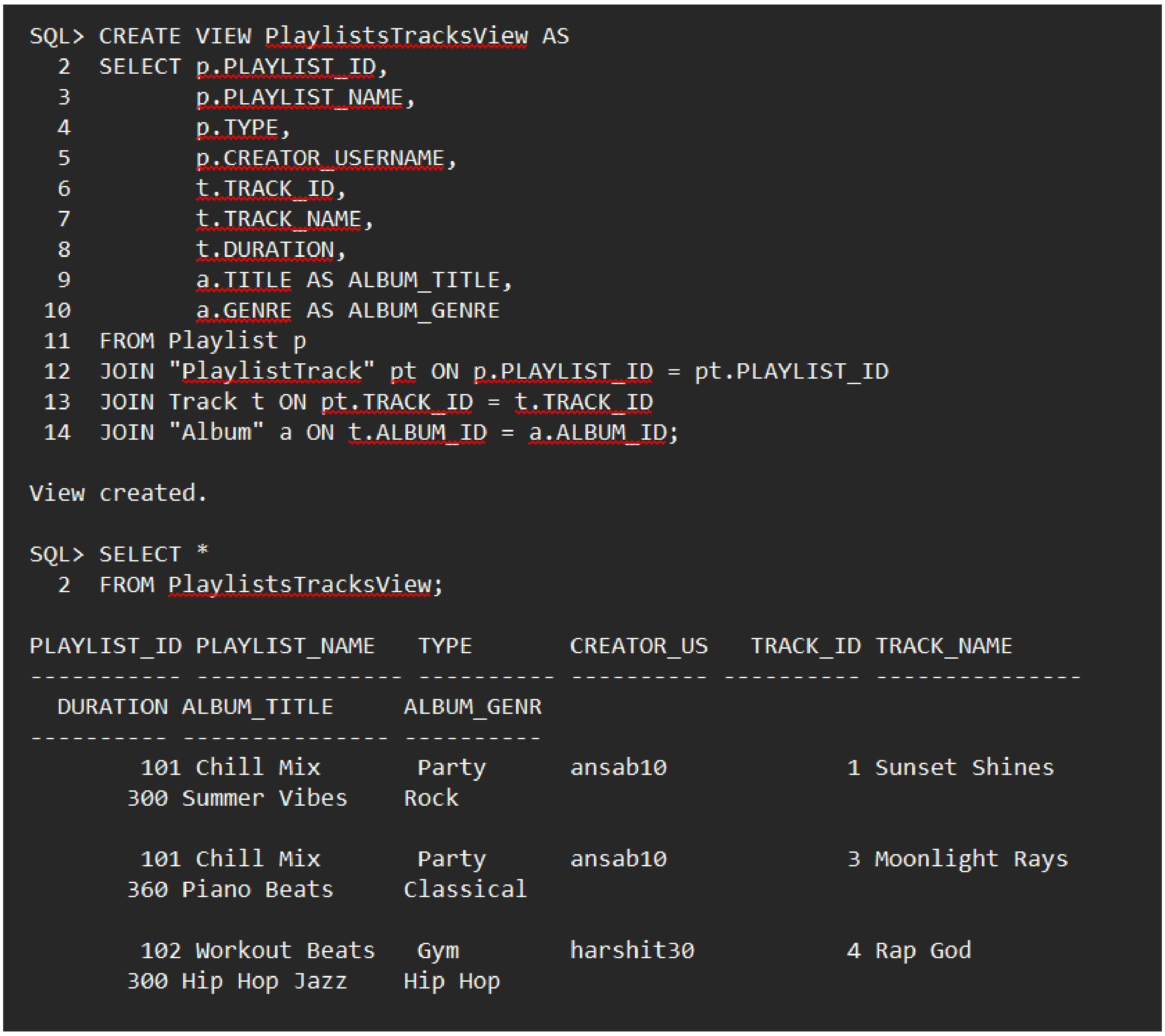
```
SQL> CREATE OR REPLACE FUNCTION calculate_total_duration(p_playlist_id IN NUMBER)
RETURN NUMBER
IS
    v_total_duration NUMBER := 0;
BEGIN
    SELECT SUM(t.DURATION)
    INTO v_total_duration
    FROM "PlaylistTrack" pt
    JOIN Track t ON pt.TRACK_ID = t.TRACK_ID
    WHERE pt.PLAYLIST_ID = p_playlist_id;

    RETURN v_total_duration;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN 0;
END;
/

Function created.

SQL> SELECT PLAYLIST_ID, calculate_total_duration(PLAYLIST_ID) AS total_duration
FROM Playlist;

PLAYLIST_ID TOTAL_DURATION
----------- --------------
        101            660
        102            570
        103            600
        104            620
        105            460
        106            680
        107
        108            600
        109            480
        110            580

10 rows selected.
```

- **<u>TRIGGERS:</u>**



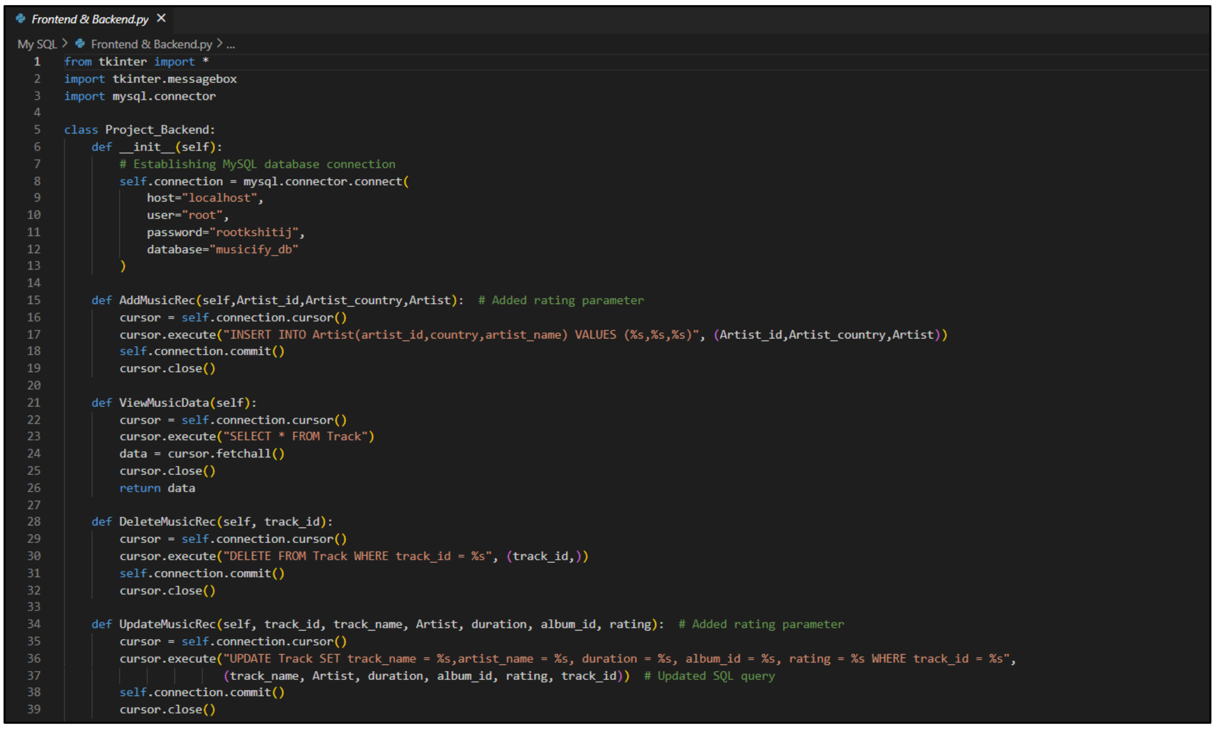
```
SQL> CREATE OR REPLACE TRIGGER trg_insert_playlist_track
BEFORE INSERT ON "PlaylistTrack"
FOR EACH ROW
DECLARE
    v_track_count NUMBER;
    v_playlist_count NUMBER;
BEGIN
    SELECT COUNT(*) INTO v_track_count FROM Track WHERE TRACK_ID = :NEW.TRACK_ID;

    SELECT COUNT(*) INTO v_playlist_count FROM Playlist WHERE PLAYLIST_ID = :NEW.PLAYLIST_ID;

    IF v_track_count = 0 THEN
        RAISE_APPLICATION_ERROR(-20001, 'Track does not exist.');
    END IF;

    IF v_playlist_count = 0 THEN
        RAISE_APPLICATION_ERROR(-20002, 'Playlist does not exist.');
    END IF;
END;
/

Trigger created.

SQL> SELECT * FROM USER_TRIGGERS WHERE TRIGGER_NAME = 'UPDATE_TRACK_COUNT';

TRIGGER_NAME
--------------------------------------------------------------------------------
TRIGGER_TYPE
----------------
TRIGGERING_EVENT
--------------------------------------------------------------------------------
TABLE_OWNER
--------------------------------------------------------------------------------
BASE_OBJECT_TYPE
------------------
TABLE_NAME
--------------------------------------------------------------------------------
COLUMN_NAME
```

- **VIEW:**



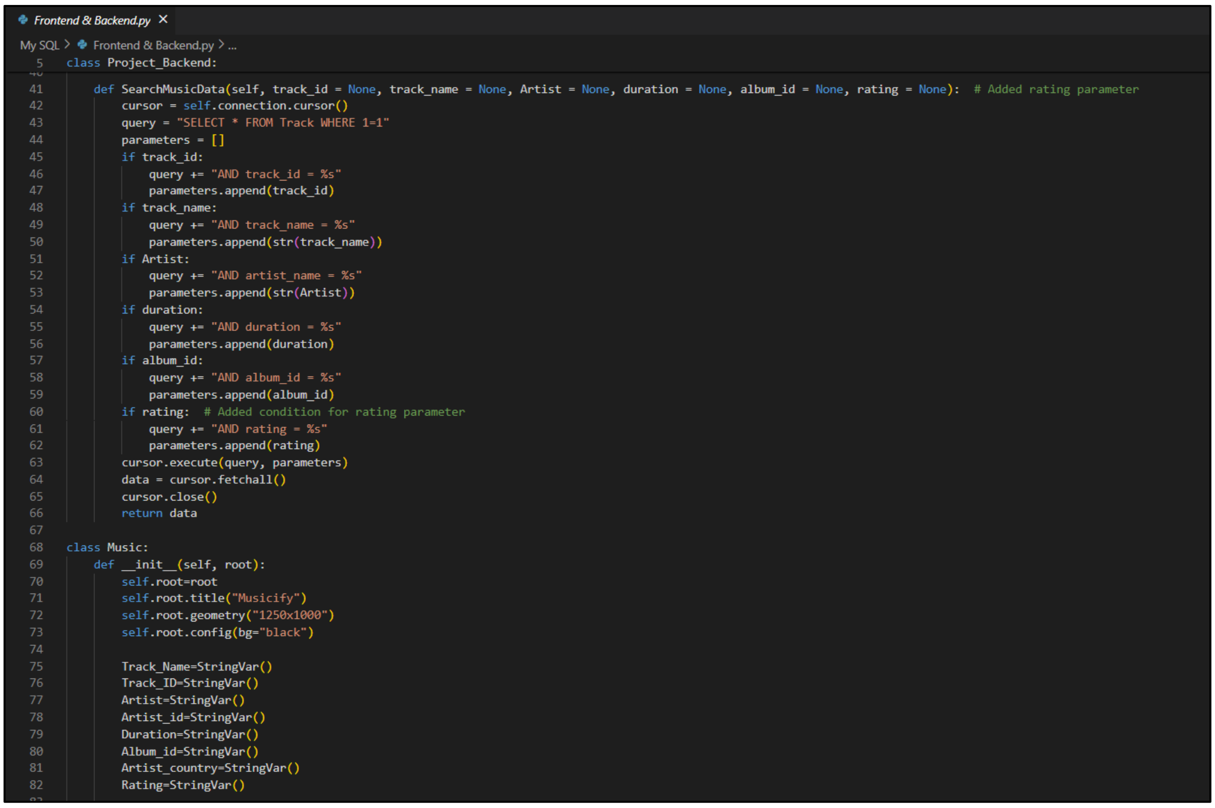
```
SQL> CREATE VIEW PlaylistsTracksView AS
  2  SELECT p.PLAYLIST_ID,
  3         p.PLAYLIST_NAME,
  4         p.TYPE,
  5         p.CREATOR_USERNAME,
  6         t.TRACK_ID,
  7         t.TRACK_NAME,
  8         t.DURATION,
  9         a.TITLE AS ALBUM_TITLE,
 10         a.GENRE AS ALBUM_GENRE
 11  FROM Playlist p
 12  JOIN "PlaylistTrack" pt ON p.PLAYLIST_ID = pt.PLAYLIST_ID
 13  JOIN Track t ON pt.TRACK_ID = t.TRACK_ID
 14  JOIN "Album" a ON t.ALBUM_ID = a.ALBUM_ID;

View created.

SQL> SELECT *
  2  FROM PlaylistsTracksView;

PLAYLIST_ID PLAYLIST_NAME   TYPE       CREATOR_US   TRACK_ID TRACK_NAME
----------- --------------- ---------- ---------- ---------- ---------------
  DURATION ALBUM_TITLE     ALBUM_GENR
---------- --------------- ----------
        101 Chill Mix       Party      ansab10             1 Sunset Shines
       300 Summer Vibes    Rock

        101 Chill Mix       Party      ansab10             3 Moonlight Rays
       360 Piano Beats     Classical

        102 Workout Beats   Gym        harshit30           4 Rap God
       300 Hip Hop Jazz    Hip Hop
```

Frontend & Backend Code:



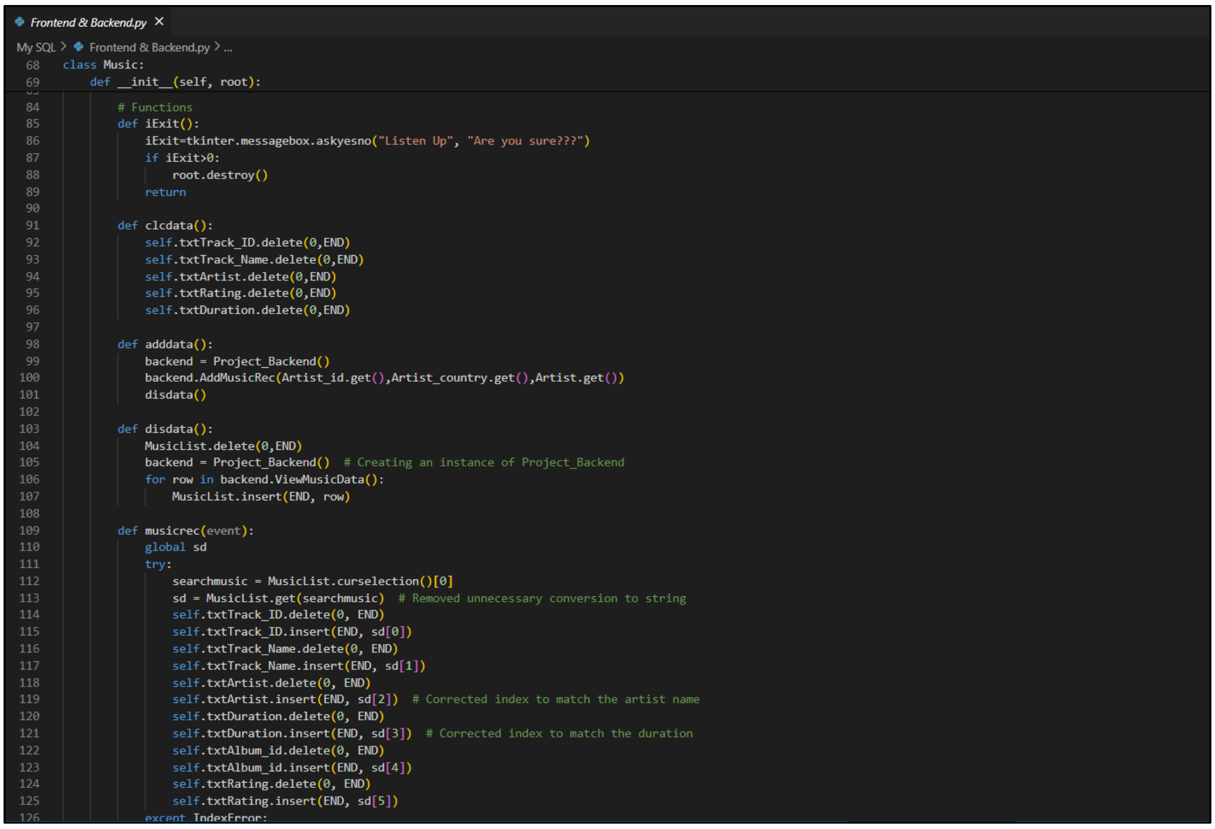
Frontend & Backend.py
My SQL > Frontend & Backend.py > ...

```python
from tkinter import *
import tkinter.messagebox
import mysql.connector

class Project_Backend:
    def __init__(self):
        # Establishing MySQL database connection
        self.connection = mysql.connector.connect(
            host="localhost",
            user="root",
            password="rootkshitij",
            database="musicify_db"
        )

    def AddMusicRec(self,Artist_id,Artist_country,Artist):  # Added rating parameter
        cursor = self.connection.cursor()
        cursor.execute("INSERT INTO Artist(artist_id,country,artist_name) VALUES (%s,%s,%s)", (Artist_id,Artist_country,Artist))
        self.connection.commit()
        cursor.close()

    def ViewMusicData(self):
        cursor = self.connection.cursor()
        cursor.execute("SELECT * FROM Track")
        data = cursor.fetchall()
        cursor.close()
        return data

    def DeleteMusicRec(self, track_id):
        cursor = self.connection.cursor()
        cursor.execute("DELETE FROM Track WHERE track_id = %s", (track_id,))
        self.connection.commit()
        cursor.close()

    def UpdateMusicRec(self, track_id, track_name, Artist, duration, album_id, rating):  # Added rating parameter
        cursor = self.connection.cursor()
        cursor.execute("UPDATE Track SET track_name = %s,artist_name = %s, duration = %s, album_id = %s, rating = %s WHERE track_id = %s",
                       (track_name, Artist, duration, album_id, rating, track_id))  # Updated SQL query
        self.connection.commit()
        cursor.close()
```

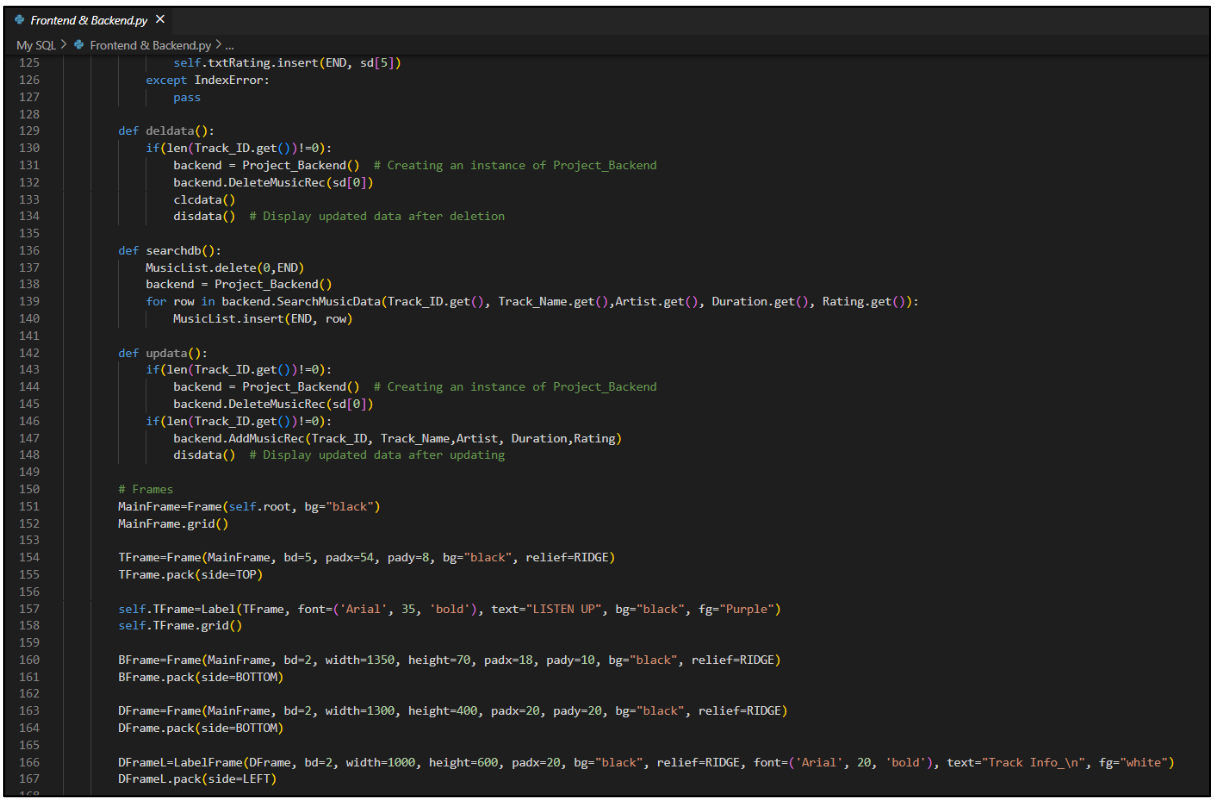


Frontend & Backend.py
My SQL > Frontend & Backend.py > ...

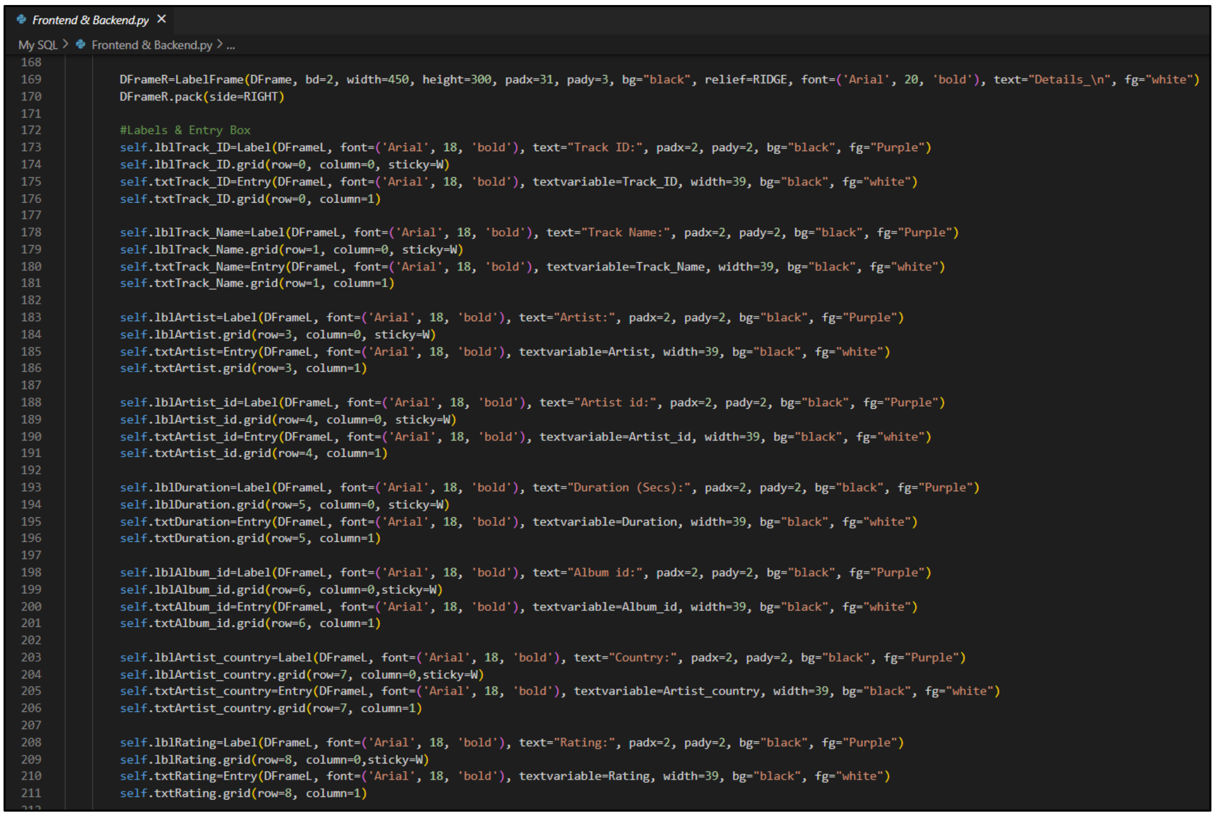
```python
class Project_Backend:
    def SearchMusicData(self, track_id = None, track_name = None, Artist = None, duration = None, album_id = None, rating = None):  # Added rating parameter
        cursor = self.connection.cursor()
        query = "SELECT * FROM Track WHERE 1=1"
        parameters = []
        if track_id:
            query += "AND track_id = %s"
            parameters.append(track_id)
        if track_name:
            query += "AND track_name = %s"
            parameters.append(str(track_name))
        if Artist:
            query += "AND artist_name = %s"
            parameters.append(str(Artist))
        if duration:
            query += "AND duration = %s"
            parameters.append(duration)
        if album_id:
            query += "AND album_id = %s"
            parameters.append(album_id)
        if rating:  # Added condition for rating parameter
            query += "AND rating = %s"
            parameters.append(rating)
        cursor.execute(query, parameters)
        data = cursor.fetchall()
        cursor.close()
        return data

class Music:
    def __init__(self, root):
        self.root=root
        self.root.title("Musicify")
        self.root.geometry("1250x1000")
        self.root.config(bg="black")

        Track_Name=StringVar()
        Track_ID=StringVar()
        Artist=StringVar()
        Artist_id=StringVar()
        Duration=StringVar()
        Album_id=StringVar()
        Artist_country=StringVar()
        Rating=StringVar()
```



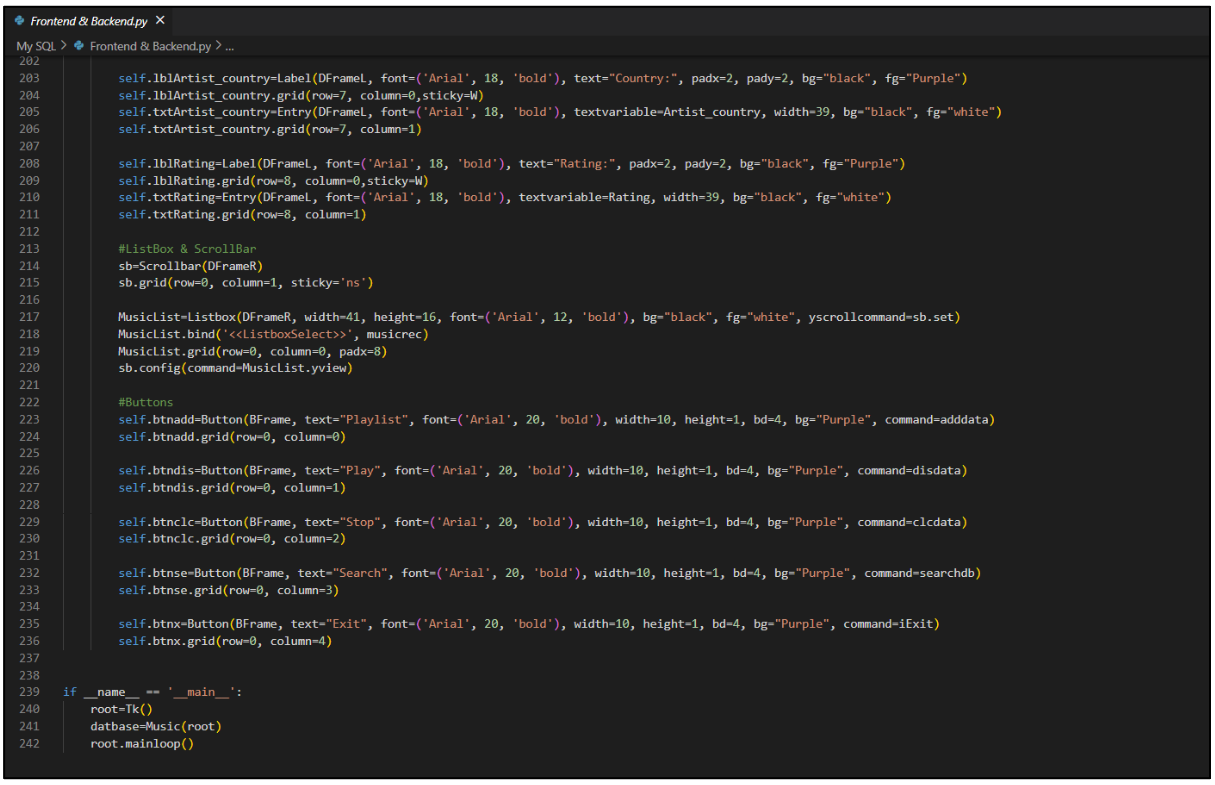
```python
class Music:
    def __init__(self, root):
        # Functions
        def iExit():
            iExit=tkinter.messagebox.askyesno("Listen Up", "Are you sure???")
            if iExit>0:
                root.destroy()
            return

        def clcdata():
            self.txtTrack_ID.delete(0,END)
            self.txtTrack_Name.delete(0,END)
            self.txtArtist.delete(0,END)
            self.txtRating.delete(0,END)
            self.txtDuration.delete(0,END)

        def adddata():
            backend = Project_Backend()
            backend.AddMusicRec(Artist_id.get(),Artist_country.get(),Artist.get())
            disdata()

        def disdata():
            MusicList.delete(0,END)
            backend = Project_Backend()  # Creating an instance of Project_Backend
            for row in backend.ViewMusicData():
                MusicList.insert(END, row)

        def musicrec(event):
            global sd
            try:
                searchmusic = MusicList.curselection()[0]
                sd = MusicList.get(searchmusic)  # Removed unnecessary conversion to string
                self.txtTrack_ID.delete(0, END)
                self.txtTrack_ID.insert(END, sd[0])
                self.txtTrack_Name.delete(0, END)
                self.txtTrack_Name.insert(END, sd[1])
                self.txtArtist.delete(0, END)
                self.txtArtist.insert(END, sd[2])  # Corrected index to match the artist name
                self.txtDuration.delete(0, END)
                self.txtDuration.insert(END, sd[3])  # Corrected index to match the duration
                self.txtAlbum_id.delete(0, END)
                self.txtAlbum_id.insert(END, sd[4])
                self.txtRating.delete(0, END)
                self.txtRating.insert(END, sd[5])
            except IndexError:
```



```python
                self.txtRating.insert(END, sd[5])
            except IndexError:
                pass

        def deldata():
            if(len(Track_ID.get())!=0):
                backend = Project_Backend()  # Creating an instance of Project_Backend
                backend.DeleteMusicRec(sd[0])
                clcdata()
                disdata()  # Display updated data after deletion

        def searchdb():
            MusicList.delete(0,END)
            backend = Project_Backend()
            for row in backend.SearchMusicData(Track_ID.get(), Track_Name.get(),Artist.get(), Duration.get(), Rating.get()):
                MusicList.insert(END, row)

        def updata():
            if(len(Track_ID.get())!=0):
                backend = Project_Backend()  # Creating an instance of Project_Backend
                backend.DeleteMusicRec(sd[0])
            if(len(Track_ID.get())!=0):
                backend.AddMusicRec(Track_ID, Track_Name,Artist, Duration,Rating)
                disdata()  # Display updated data after updating

        # Frames
        MainFrame=Frame(self.root, bg="black")
        MainFrame.grid()

        TFrame=Frame(MainFrame, bd=5, padx=54, pady=8, bg="black", relief=RIDGE)
        TFrame.pack(side=TOP)

        self.TFrame=Label(TFrame, font=('Arial', 35, 'bold'), text="LISTEN UP", bg="black", fg="Purple")
        self.TFrame.grid()

        BFrame=Frame(MainFrame, bd=2, width=1350, height=70, padx=18, pady=10, bg="black", relief=RIDGE)
        BFrame.pack(side=BOTTOM)

        DFrame=Frame(MainFrame, bd=2, width=1300, height=400, padx=20, pady=20, bg="black", relief=RIDGE)
        DFrame.pack(side=BOTTOM)

        DFrameL=LabelFrame(DFrame, bd=2, width=1000, height=600, padx=20, bg="black", relief=RIDGE, font=('Arial', 20, 'bold'), text="Track Info_\n", fg="white")
        DFrameL.pack(side=LEFT)
```

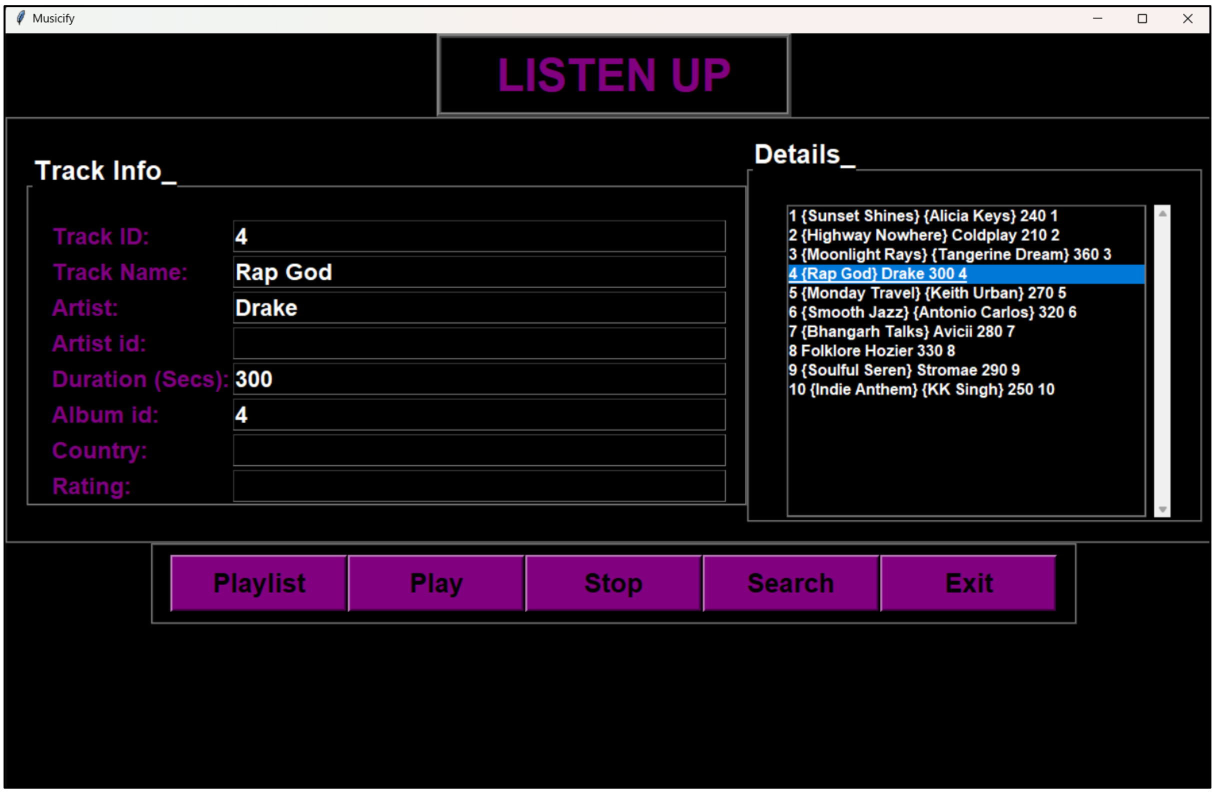


```python

        DFrameR=LabelFrame(DFrame, bd=2, width=450, height=300, padx=31, pady=3, bg="black", relief=RIDGE, font=('Arial', 20, 'bold'), text="Details_\n", fg="white")
        DFrameR.pack(side=RIGHT)

        #Labels & Entry Box
        self.lblTrack_ID=Label(DFrameL, font=('Arial', 18, 'bold'), text="Track ID:", padx=2, pady=2, bg="black", fg="Purple")
        self.lblTrack_ID.grid(row=0, column=0, sticky=W)
        self.txtTrack_ID=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Track_ID, width=39, bg="black", fg="white")
        self.txtTrack_ID.grid(row=0, column=1)

        self.lblTrack_Name=Label(DFrameL, font=('Arial', 18, 'bold'), text="Track Name:", padx=2, pady=2, bg="black", fg="Purple")
        self.lblTrack_Name.grid(row=1, column=0, sticky=W)
        self.txtTrack_Name=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Track_Name, width=39, bg="black", fg="white")
        self.txtTrack_Name.grid(row=1, column=1)

        self.lblArtist=Label(DFrameL, font=('Arial', 18, 'bold'), text="Artist:", padx=2, pady=2, bg="black", fg="Purple")
        self.lblArtist.grid(row=3, column=0, sticky=W)
        self.txtArtist=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Artist, width=39, bg="black", fg="white")
        self.txtArtist.grid(row=3, column=1)

        self.lblArtist_id=Label(DFrameL, font=('Arial', 18, 'bold'), text="Artist id:", padx=2, pady=2, bg="black", fg="Purple")
        self.lblArtist_id.grid(row=4, column=0, sticky=W)
        self.txtArtist_id=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Artist_id, width=39, bg="black", fg="white")
        self.txtArtist_id.grid(row=4, column=1)

        self.lblDuration=Label(DFrameL, font=('Arial', 18, 'bold'), text="Duration (Secs):", padx=2, pady=2, bg="black", fg="Purple")
        self.lblDuration.grid(row=5, column=0, sticky=W)
        self.txtDuration=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Duration, width=39, bg="black", fg="white")
        self.txtDuration.grid(row=5, column=1)

        self.lblAlbum_id=Label(DFrameL, font=('Arial', 18, 'bold'), text="Album id:", padx=2, pady=2, bg="black", fg="Purple")
        self.lblAlbum_id.grid(row=6, column=0,sticky=W)
        self.txtAlbum_id=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Album_id, width=39, bg="black", fg="white")
        self.txtAlbum_id.grid(row=6, column=1)

        self.lblArtist_country=Label(DFrameL, font=('Arial', 18, 'bold'), text="Country:", padx=2, pady=2, bg="black", fg="Purple")
        self.lblArtist_country.grid(row=7, column=0,sticky=W)
        self.txtArtist_country=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Artist_country, width=39, bg="black", fg="white")
        self.txtArtist_country.grid(row=7, column=1)

        self.lblRating=Label(DFrameL, font=('Arial', 18, 'bold'), text="Rating:", padx=2, pady=2, bg="black", fg="Purple")
        self.lblRating.grid(row=8, column=0,sticky=W)
        self.txtRating=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Rating, width=39, bg="black", fg="white")
        self.txtRating.grid(row=8, column=1)
```



```python
        self.lblArtist_country=Label(DFrameL, font=('Arial', 18, 'bold'), text="Country:", padx=2, pady=2, bg="black", fg="Purple")
        self.lblArtist_country.grid(row=7, column=0,sticky=W)
        self.txtArtist_country=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Artist_country, width=39, bg="black", fg="white")
        self.txtArtist_country.grid(row=7, column=1)

        self.lblRating=Label(DFrameL, font=('Arial', 18, 'bold'), text="Rating:", padx=2, pady=2, bg="black", fg="Purple")
        self.lblRating.grid(row=8, column=0,sticky=W)
        self.txtRating=Entry(DFrameL, font=('Arial', 18, 'bold'), textvariable=Rating, width=39, bg="black", fg="white")
        self.txtRating.grid(row=8, column=1)

        #ListBox & ScrollBar
        sb=Scrollbar(DFrameR)
        sb.grid(row=0, column=1, sticky='ns')

        MusicList=Listbox(DFrameR, width=41, height=16, font=('Arial', 12, 'bold'), bg="black", fg="white", yscrollcommand=sb.set)
        MusicList.bind('<<ListboxSelect>>', musicrec)
        MusicList.grid(row=0, column=0, padx=8)
        sb.config(command=MusicList.yview)

        #Buttons
        self.btnadd=Button(BFrame, text="Playlist", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="Purple", command=adddata)
        self.btnadd.grid(row=0, column=0)

        self.btndis=Button(BFrame, text="Play", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="Purple", command=disdata)
        self.btndis.grid(row=0, column=1)

        self.btnclc=Button(BFrame, text="Stop", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="Purple", command=clcdata)
        self.btnclc.grid(row=0, column=2)

        self.btnse=Button(BFrame, text="Search", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="Purple", command=searchdb)
        self.btnse.grid(row=0, column=3)

        self.btnx=Button(BFrame, text="Exit", font=('Arial', 20, 'bold'), width=10, height=1, bd=4, bg="Purple", command=iExit)
        self.btnx.grid(row=0, column=4)


if __name__ == '__main__':
    root=Tk()
    datbase=Music(root)
    root.mainloop()
```

Output:



Track Info_

Track ID:	4
Track Name:	Rap God
Artist:	Drake
Artist id:	
Duration (Secs):	300
Album id:	4
Country:	
Rating:	

Details_

```
1 {Sunset Shines} {Alicia Keys} 240 1
2 {Highway Nowhere} Coldplay 210 2
3 {Moonlight Rays} {Tangerine Dream} 360 3
4 {Rap God} Drake 300 4
5 {Monday Travel} {Keith Urban} 270 5
6 {Smooth Jazz} {Antonio Carlos} 320 6
7 {Bhangarh Talks} Avicii 280 7
8 Folklore Hozier 330 8
9 {Soulful Seren} Stromae 290 9
10 {Indie Anthem} {KK Singh} 250 10
```

CONCLUSION

The "Musicify" project represents a significant step forward in the realm of music management and enjoyment. Through meticulous planning, development, and implementation, we have created a robust software solution that caters to the diverse needs of music enthusiasts and organizations alike.

In conclusion, this project has successfully achieved its objectives of providing an efficient, user-friendly platform for organizing music collections, enhancing the overall listening experience, fostering social interaction, ensuring scalability and reliability, and prioritizing data security and privacy.

Moving forward, we remain committed to continuous improvement and innovation, striving to incorporate feedback from users, adapt to emerging technologies, and stay at the forefront of the music streaming landscape. With a strong foundation in place, we are confident that the "Music Library System" will continue to evolve and thrive, delivering unparalleled value to its users for years to come.