PES UNIVERSITY

Department of CSE

UE22CS343BB3 - DATABASE TECHNOLOGIES

Jan – May 2025

DBT ASSIGNMENT #1

Name: Mohammed Hassan

SRN: PES2UG22CS317

Github link: Assignment-1 queries and read files

(a) Database Preparation:

The database preparation is completed. Topic chosen is **Music Database (for a user).**

Step 1: The table construction quarry

```
-- Active: 1731390564563@@127.0.0.1@3306@dbt25 a1 pes2ug22cs317 mohammedhassan
-- Creation of the database.
CREATE DATABASE `DBT25 A1 PES2UG22CS317 MohammedHassan` ;

-- Domain selected music. [Music Streaming Service]

-- Users Table
CREATE TABLE Users (
    user_id INT PRIMARY KEY,
    username VARCHAR(50) NOT NULL UNIQUE,
    email VARCHAR(100) NOT NULL UNIQUE,
    password_hash VARCHAR(255) NOT NULL,
    subscription_type ENUM('free', 'premium', 'family') DEFAULT 'free',
    date_joined DATE
);

-- Artists Table
CREATE TABLE Artists (
```

```sql
    artist_id INT PRIMARY KEY,
    artist_name VARCHAR(100) NOT NULL,
    bio TEXT,
    monthly_listeners INT DEFAULT 0,
    verified BOOLEAN DEFAULT FALSE
);

-- Albums Table
CREATE TABLE Albums (
    album_id INT PRIMARY KEY,
    album_name VARCHAR(100) NOT NULL,
    artist_id INT,
    release_date DATE,
    album_type ENUM('single', 'EP', 'album'),
    total_tracks INT,
    FOREIGN KEY (artist_id) REFERENCES Artists(artist_id)
);

-- Songs Table
CREATE TABLE Songs (
    song_id INT PRIMARY KEY,
    song_name VARCHAR(100) NOT NULL,
    album_id INT,
    duration INT,  -- Duration in seconds
    track_number INT,
    explicit BOOLEAN DEFAULT FALSE,
    play_count INT DEFAULT 0,
    FOREIGN KEY (album_id) REFERENCES Albums(album_id)
);

-- Playlists Table
CREATE TABLE Playlists (
    playlist_id INT PRIMARY KEY,
    playlist_name VARCHAR(100) NOT NULL,
    user_id INT,
    created_date DATETIME,
    is_public BOOLEAN DEFAULT TRUE,
    description TEXT,
    FOREIGN KEY (user_id) REFERENCES Users(user_id)
);

-- PlaylistSongs Table (Junction table)
CREATE TABLE PlaylistSongs (
    playlist_id INT,
    song_id INT,
    date_added DATETIME,
    PRIMARY KEY (playlist_id, song_id),
    FOREIGN KEY (playlist_id) REFERENCES Playlists(playlist_id),
    FOREIGN KEY (song_id) REFERENCES Songs(song_id)
);

-- UserLibrary Table (Liked/Saved content)
CREATE TABLE UserLibrary (
    user_id INT,
    song_id INT,
    date_added DATETIME,
    PRIMARY KEY (user_id, song_id),
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (song_id) REFERENCES Songs(song_id)
);
```

Step 2: Adding of the base values in the tables.

Adding the initial values to the tables and keeping one user as email 'mohammedhassan@pes.edu' and username as 'PES2UG22CS317'

```sql
USE `DBT25 A1 PES2UG22CS317 MohammedHassan`;

-- Insert base users (you + 10 others)
INSERT INTO Users (user_id, username, email, password_hash, subscription_type,
date_joined) VALUES
(1, 'PES2UG22CS317', 'mohammedhassan@pes.edu', SHA2('pass123', 256),
'premium', '2024-01-01'),
(2, 'john_doe', 'john@example.com', SHA2('pass456', 256), 'free', '2024-01-
02'),
(3, 'emma_smith', 'emma@example.com', SHA2('pass789', 256), 'premium', '2024-
01-03'),
(4, 'alex_brown', 'alex@example.com', SHA2('pass101', 256), 'family', '2024-
01-04'),
(5, 'sarah_wilson', 'sarah@example.com', SHA2('pass102', 256), 'free', '2024-
01-05'),
(6, 'mike_davis', 'mike@example.com', SHA2('pass103', 256), 'premium', '2024-
01-06'),
(7, 'lisa_miller', 'lisa@example.com', SHA2('pass104', 256), 'family', '2024-
01-07'),
(8, 'david_jones', 'david@example.com', SHA2('pass105', 256), 'free', '2024-
01-08'),
(9, 'anna_white', 'anna@example.com', SHA2('pass106', 256), 'premium', '2024-
01-09'),
(10, 'james_taylor', 'james@example.com', SHA2('pass107', 256), 'free', '2024-
01-10'),
(11, 'maria_garcia', 'maria@example.com', SHA2('pass108', 256), 'premium',
'2024-01-11');

-- Insert one base artist
INSERT INTO Artists (artist_id, artist_name, bio, monthly_listeners, verified)
```

```sql
VALUES (1, 'Base Artist', 'First artist in the system', 1000000, true);

-- Insert base albums with varied release dates
INSERT INTO Albums (album_id, album_name, artist_id, release_date, album_type,
total_tracks)
VALUES
(1, 'First Album', 1, '2024-01-01', 'album', 1),
(2, 'Second Album', 1, '2023-06-15', 'album', 1),
(3, 'Third Album', 1, '2022-12-25', 'album', 1);

-- Insert one base song
INSERT INTO Songs (song_id, song_name, album_id, duration, track_number,
explicit, play_count)
VALUES (1, 'First Song', 1, 180, 1, false, 0);

INSERT INTO Songs (song_id, song_name, album_id, duration, track_number,
explicit, play_count) VALUES
(1001, 'First Song', 1, 180, 1, false, 0),
(1002, 'Love Story', 1, 240, 2, false, 1000),
(1003, 'Endless Love', 2, 195, 1, false, 500),
(1004, 'Love Me Like You Do', 2, 235, 2, false, 2000),
(1005, 'True Love', 3, 210, 1, false, 1500);


-- Insert one base playlist
INSERT INTO Playlists (playlist_id, playlist_name, user_id, created_date,
is_public, description)
VALUES (1, 'My First Playlist', 1, NOW(), true, 'Base playlist');
```

Step 3: A Python code for loading 10,000+ rows of data into in least 2 of these tables those are `playlistsongs` and `userlibrary`

```python
import random
from datetime import datetime, timedelta
import mysql.connector

def connect_to_db():
    return mysql.connector.connect(
        host="localhost",
        user="root",
        password="dbms",
        database="DBT25 A1 PES2UG22CS317 MohammedHassan"
    )

def generate_artists(cursor, num_artists=100):
    print("Generating artists...")
    for i in range(2, num_artists + 1):
        cursor.execute("""
            INSERT INTO Artists (artist_id, artist_name, bio,
monthly_listeners, verified)
            VALUES (%s, %s, %s, %s, %s)
        """, (i, f'Artist {i}', f'Bio for artist {i}',
              random.randint(1000, 1000000), random.choice([True, False])))
    return range(1, num_artists + 1)

def generate_albums(cursor, artist_ids, num_albums=200):
    print("Generating albums...")
    start_date = datetime(2020, 1, 1)
    for i in range(2, num_albums + 1):
        release_date = start_date + timedelta(days=random.randint(0, 1000))
        cursor.execute("""
            INSERT INTO Albums (album_id, album_name, artist_id, release_date,
album_type, total_tracks)
            VALUES (%s, %s, %s, %s, %s, %s)
        """, (i, f'Album {i}', random.choice(artist_ids), release_date,
              random.choice(['single', 'EP', 'album']), random.randint(1,
20)))
    return range(1, num_albums + 1)

def generate_songs(cursor, album_ids, num_songs=1000):
    print("Generating songs...")
    for i in range(2, num_songs + 1):
        cursor.execute("""
            INSERT INTO Songs (song_id, song_name, album_id, duration,
track_number, explicit, play_count)
            VALUES (%s, %s, %s, %s, %s, %s, %s)
        """, (i, f'Song {i}', random.choice(album_ids),
              random.randint(120, 400), random.randint(1, 20),
              random.choice([True, False]), random.randint(0, 1000000)))
    return range(1, num_songs + 1)

def generate_playlists(cursor, num_playlists=100):
    print("Generating playlists...")
    # Get all user IDs
    cursor.execute("SELECT user_id FROM Users")
    user_ids = [row[0] for row in cursor.fetchall()]

    for i in range(2, num_playlists + 1):
        created_date = datetime.now() - timedelta(days=random.randint(0, 365))
        # Simple random assignment without bias
        user_id = random.choice(user_ids)
        cursor.execute("""
            INSERT INTO Playlists (playlist_id, playlist_name, user_id,
created_date, is_public, description)
            VALUES (%s, %s, %s, %s, %s, %s)
        """, (i, f'Playlist {i}', user_id, created_date,
```

```python
            random.choice([True, False]), f'Description for playlist {i}'))
    return range(1, num_playlists + 1)

def generate_playlist_songs(cursor, playlist_ids, song_ids,
num_records=10000):
    print("Generating playlist songs...")
    records = set()
    while len(records) < num_records:
        records.add((random.choice(playlist_ids), random.choice(song_ids)))

    for playlist_id, song_id in records:
        try:
            cursor.execute("""
                INSERT INTO PlaylistSongs (playlist_id, song_id, date_added)
                VALUES (%s, %s, %s)
            """, (playlist_id, song_id, datetime.now() -
timedelta(days=random.randint(0, 365))))
        except mysql.connector.IntegrityError:
            continue

def generate_user_library(cursor, song_ids, num_records=10000):
    print("Generating user library...")
    # Get all user IDs
    cursor.execute("SELECT user_id FROM Users")
    user_ids = [row[0] for row in cursor.fetchall()]

    records = set()
    target_records = num_records // len(user_ids)  # Equal records per user

    # Give each user equal number of songs
    for user_id in user_ids:
        user_records = 0
        while user_records < target_records:
            record = (user_id, random.choice(song_ids))
            if record not in records:
                records.add(record)
                user_records += 1

    # Insert the records
    for user_id, song_id in records:
        try:
            cursor.execute("""
                INSERT INTO UserLibrary (user_id, song_id, date_added)
                VALUES (%s, %s, %s)
            """, (user_id, song_id, datetime.now() -
timedelta(days=random.randint(0, 365))))
        except mysql.connector.IntegrityError:
            continue
```

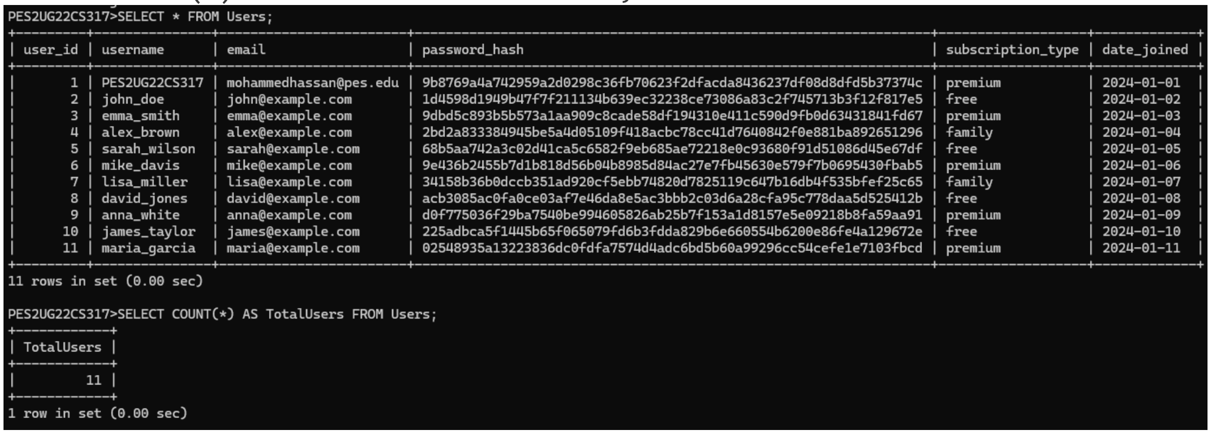
```python
def main():
    conn = connect_to_db()
    cursor = conn.cursor()

    try:
        print("Starting data generation...")

        artist_ids = generate_artists(cursor)
        conn.commit()

        album_ids = generate_albums(cursor, artist_ids)
        conn.commit()

        song_ids = generate_songs(cursor, album_ids)
        conn.commit()

        playlist_ids = generate_playlists(cursor)
        conn.commit()

        generate_playlist_songs(cursor, playlist_ids, song_ids)
        conn.commit()

        generate_user_library(cursor, song_ids)
        conn.commit()

        # Verify counts
        cursor.execute("SELECT COUNT(*) FROM PlaylistSongs")
        print(f"PlaylistSongs count: {cursor.fetchone()[0]}")
        cursor.execute("SELECT COUNT(*) FROM UserLibrary")
        print(f"UserLibrary count: {cursor.fetchone()[0]}")

        # Add user count verification
        cursor.execute("SELECT COUNT(*) FROM Users")
        print(f"Total Users: {cursor.fetchone()[0]}")

        cursor.execute("""
            SELECT u.username, COUNT(ul.song_id) as song_count
            FROM Users u
            LEFT JOIN UserLibrary ul ON u.user_id = ul.user_id
            GROUP BY u.user_id, u.username
        """)
        print("\nSongs per user:")
        for row in cursor.fetchall():
            print(f"{row[0]}: {row[1]} songs")

    except Exception as e:
        print(f"Error: {e}")
```

```python
            conn.rollback()
    finally:
        cursor.close()
        conn.close()
        print("Data generation complete!")

if __name__ == "__main__":
    main()
```

(b) Queries Creation and Performance Measurement:

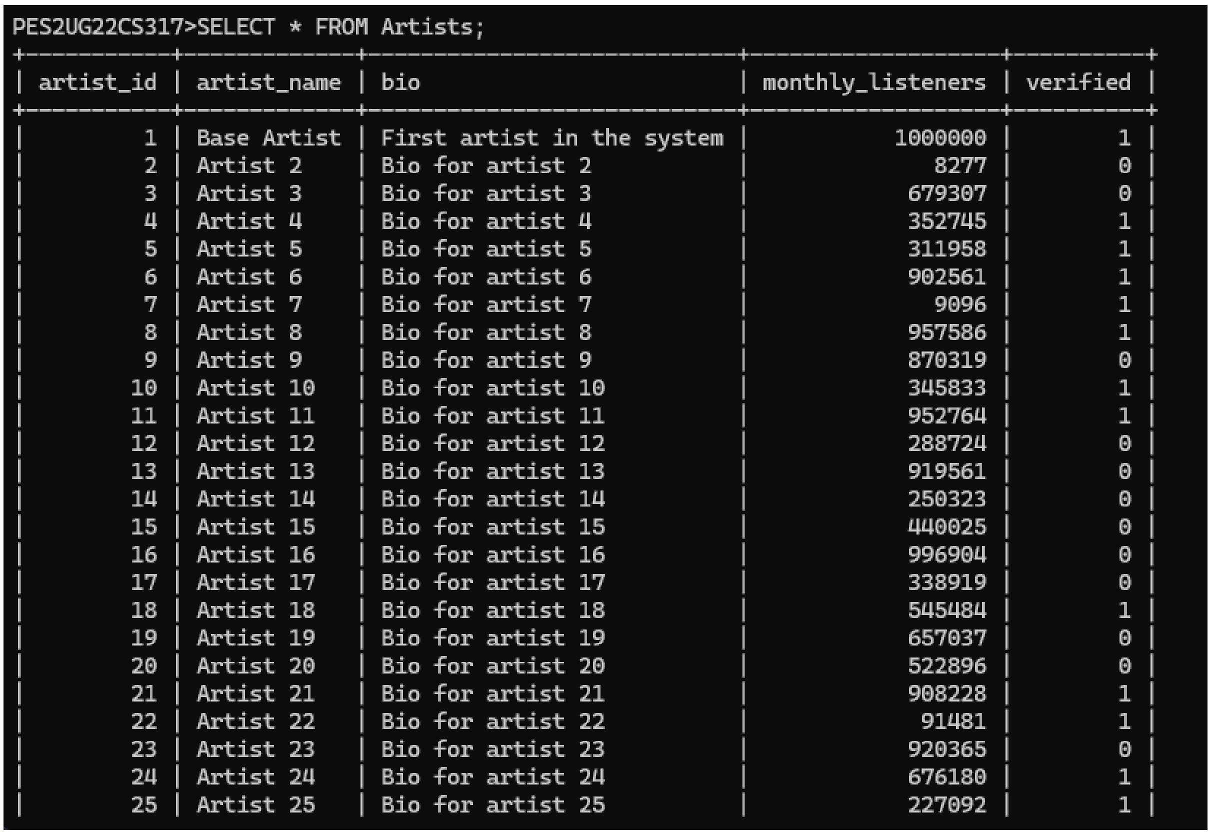
1. Execute "SELECT *" queries on all tables to display data and count the rows:

```sql
-- Select all data and count rows from the Users table
SELECT * FROM Users;
SELECT COUNT(*) AS TotalUsers FROM Users;
```

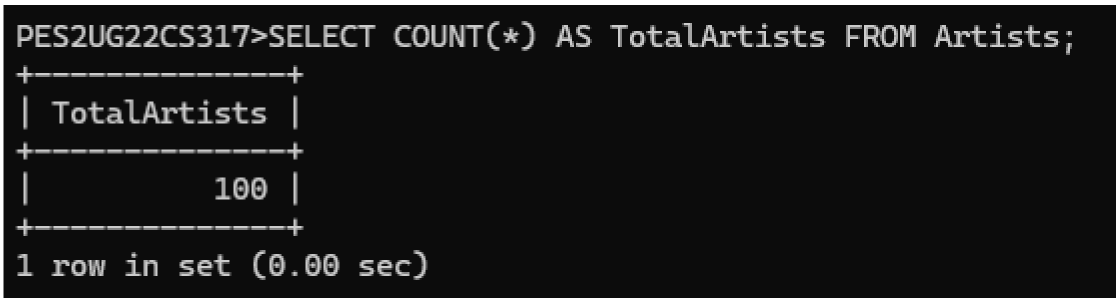


```
PES2UG22CS317>SELECT * FROM Users;
+---------+--------------+------------------------+------------------------------------------------------------------+-------------------+-------------+
| user_id | username     | email                  | password_hash                                                    | subscription_type | date_joined |
+---------+--------------+------------------------+------------------------------------------------------------------+-------------------+-------------+
|       1 | PES2UG22CS317 | mohammedhassan@pes.edu | 9b8769a4a742959a2d0298c36fb70623f2dfacda8436237df08d8dfd5b37374c | premium           | 2024-01-01  |
|       2 | john_doe     | john@example.com       | 1d4598d1949b47f7f211134b639ec32238ce73086a83c2f745713b3f12f817e5 | free              | 2024-01-02  |
|       3 | emma_smith   | emma@example.com       | 9dbd5c893b5b573a1aa909c8cade58df194310e411c590d9fb0d63431841fd67 | premium           | 2024-01-03  |
|       4 | alex_brown   | alex@example.com       | 2bd2a833384945be5a4d05109f418acbc78cc41d7640842f0e881ba892651296 | family            | 2024-01-04  |
|       5 | sarah_wilson | sarah@example.com      | 68b5aa742a3c02d41ca5c6582f9eb685ae72218e0c93680f91d51086d45e67df | free              | 2024-01-05  |
|       6 | mike_davis   | mike@example.com       | 9e436b2455b7d1b818d56b04b8985d84ac27e7fb45630e579f7b0695430fbab5 | premium           | 2024-01-06  |
|       7 | lisa_miller  | lisa@example.com       | 34158b36b0dccb351ad920cf5ebb74820d7825119c647b16db4f535bfef25c65 | family            | 2024-01-07  |
|       8 | david_jones  | david@example.com      | acb3085ac0fa0ce03af7e46da8e5ac3bbb2c03d6a28cfa95c778daa5d525412b | free              | 2024-01-08  |
|       9 | anna_white   | anna@example.com       | d0f775036f29ba7540be994605826ab25b7f153a1d8157e5e09218b8fa59aa91 | premium           | 2024-01-09  |
|      10 | james_taylor | james@example.com      | 225adbca5f1445b65f065079fd6b3fdda829b6e660554b6200e86fe4a129672e | free              | 2024-01-10  |
|      11 | maria_garcia | maria@example.com      | 02548935a13223836dc0fdfa7574d4adc6bd5b60a99296cc54cefe1e7103fbcd | premium           | 2024-01-11  |
+---------+--------------+------------------------+------------------------------------------------------------------+-------------------+-------------+
11 rows in set (0.00 sec)

PES2UG22CS317>SELECT COUNT(*) AS TotalUsers FROM Users;
+------------+
| TotalUsers |
+------------+
|         11 |
+------------+
1 row in set (0.00 sec)
```

```sql
-- Select all data and count rows from the Artists table
SELECT * FROM Artists ;
SELECT COUNT(*) AS TotalArtists FROM Artists;
```

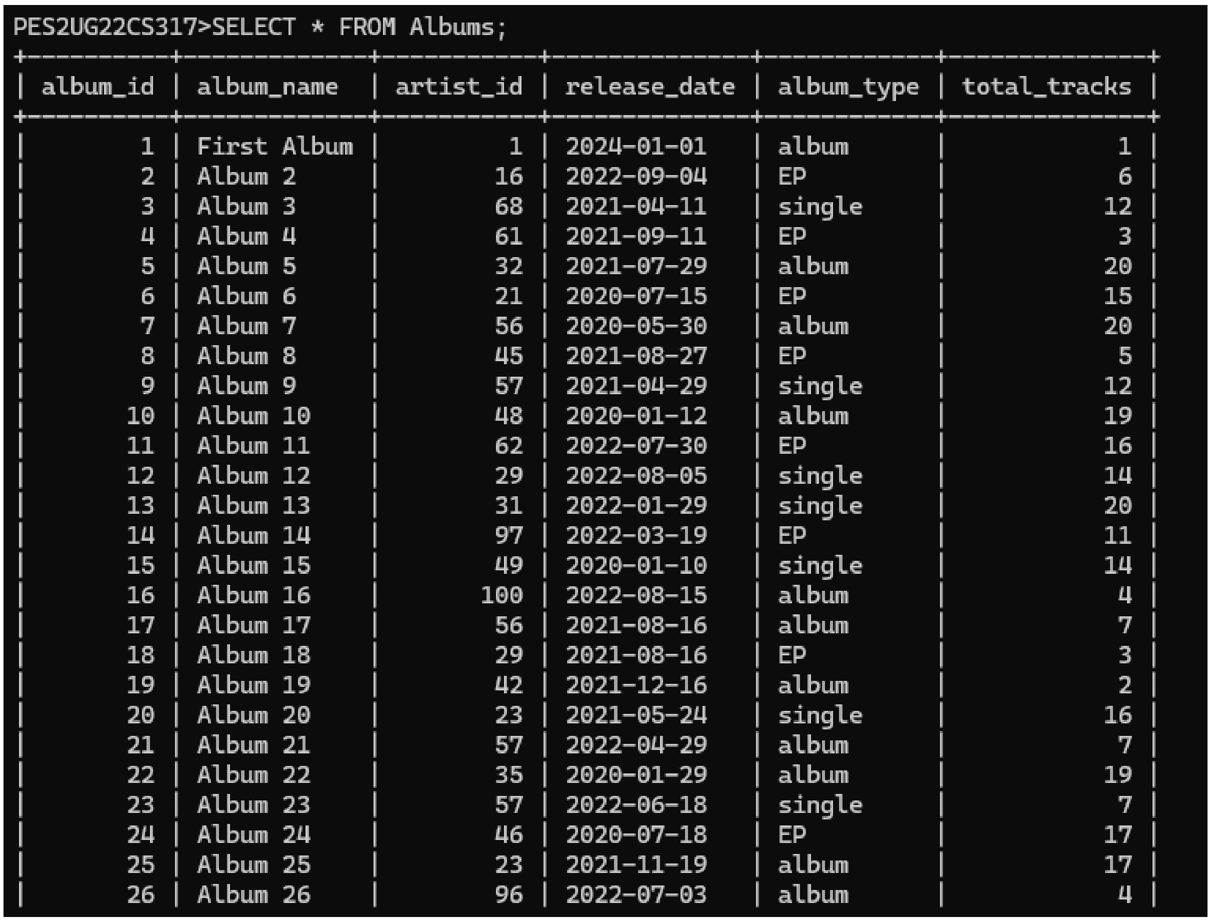


```
PES2UG22CS317>SELECT * FROM Artists;
+-----------+-------------+---------------------------+-------------------+----------+
| artist_id | artist_name | bio                       | monthly_listeners | verified |
+-----------+-------------+---------------------------+-------------------+----------+
|         1 | Base Artist | First artist in the system |           1000000 |        1 |
|         2 | Artist 2    | Bio for artist 2          |              8277 |        0 |
|         3 | Artist 3    | Bio for artist 3          |            679307 |        0 |
|         4 | Artist 4    | Bio for artist 4          |            352745 |        1 |
|         5 | Artist 5    | Bio for artist 5          |            311958 |        1 |
|         6 | Artist 6    | Bio for artist 6          |            902561 |        1 |
|         7 | Artist 7    | Bio for artist 7          |              9096 |        1 |
|         8 | Artist 8    | Bio for artist 8          |            957586 |        1 |
|         9 | Artist 9    | Bio for artist 9          |            870319 |        0 |
|        10 | Artist 10   | Bio for artist 10         |            345833 |        1 |
|        11 | Artist 11   | Bio for artist 11         |            952764 |        1 |
|        12 | Artist 12   | Bio for artist 12         |            288724 |        0 |
|        13 | Artist 13   | Bio for artist 13         |            919561 |        0 |
|        14 | Artist 14   | Bio for artist 14         |            250323 |        0 |
|        15 | Artist 15   | Bio for artist 15         |            440025 |        0 |
|        16 | Artist 16   | Bio for artist 16         |            996904 |        0 |
|        17 | Artist 17   | Bio for artist 17         |            338919 |        0 |
|        18 | Artist 18   | Bio for artist 18         |            545484 |        1 |
|        19 | Artist 19   | Bio for artist 19         |            657037 |        0 |
|        20 | Artist 20   | Bio for artist 20         |            522896 |        0 |
|        21 | Artist 21   | Bio for artist 21         |            908228 |        1 |
|        22 | Artist 22   | Bio for artist 22         |             91481 |        1 |
|        23 | Artist 23   | Bio for artist 23         |            920365 |        0 |
|        24 | Artist 24   | Bio for artist 24         |            676180 |        1 |
|        25 | Artist 25   | Bio for artist 25         |            227092 |        1 |
```

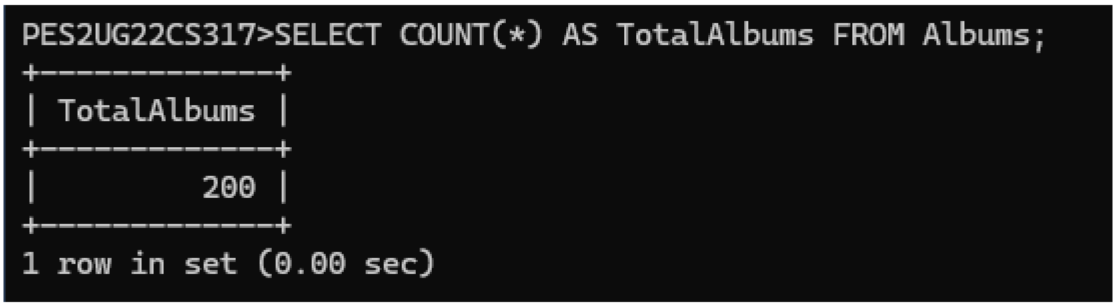


```
PES2UG22CS317>SELECT COUNT(*) AS TotalArtists FROM Artists;
+--------------+
| TotalArtists |
+--------------+
|          100 |
+--------------+
1 row in set (0.00 sec)
```

```sql
-- Select all data and count rows from the Albums table
SELECT * FROM Albums;
SELECT COUNT(*) AS TotalAlbums FROM Albums;
```

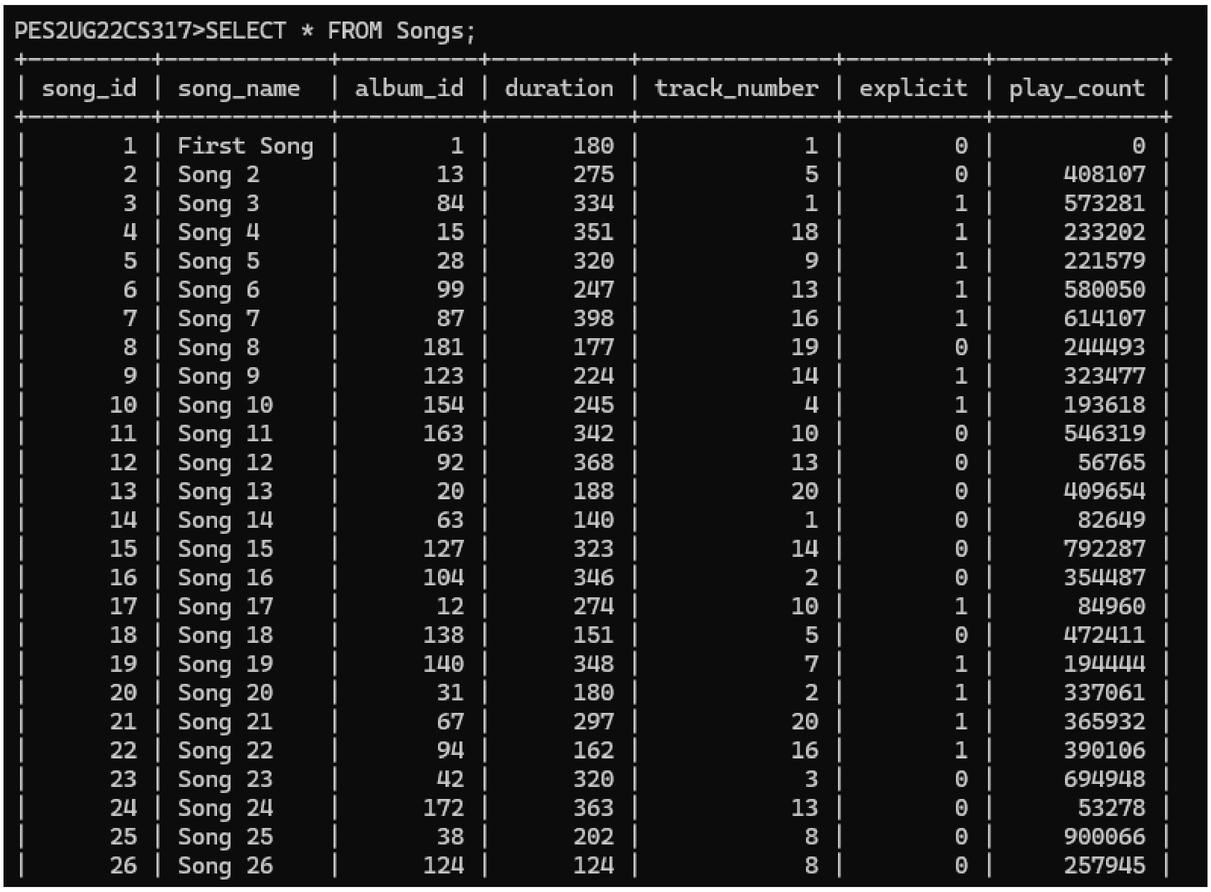


```
PES2UG22CS317>SELECT * FROM Albums;
+----------+-------------+-----------+--------------+------------+--------------+
| album_id | album_name  | artist_id | release_date | album_type | total_tracks |
+----------+-------------+-----------+--------------+------------+--------------+
|        1 | First Album |         1 | 2024-01-01   | album      |            1 |
|        2 | Album 2     |        16 | 2022-09-04   | EP         |            6 |
|        3 | Album 3     |        68 | 2021-04-11   | single     |           12 |
|        4 | Album 4     |        61 | 2021-09-11   | EP         |            3 |
|        5 | Album 5     |        32 | 2021-07-29   | album      |           20 |
|        6 | Album 6     |        21 | 2020-07-15   | EP         |           15 |
|        7 | Album 7     |        56 | 2020-05-30   | album      |           20 |
|        8 | Album 8     |        45 | 2021-08-27   | EP         |            5 |
|        9 | Album 9     |        57 | 2021-04-29   | single     |           12 |
|       10 | Album 10    |        48 | 2020-01-12   | album      |           19 |
|       11 | Album 11    |        62 | 2022-07-30   | EP         |           16 |
|       12 | Album 12    |        29 | 2022-08-05   | single     |           14 |
|       13 | Album 13    |        31 | 2022-01-29   | single     |           20 |
|       14 | Album 14    |        97 | 2022-03-19   | EP         |           11 |
|       15 | Album 15    |        49 | 2020-01-10   | single     |           14 |
|       16 | Album 16    |       100 | 2022-08-15   | album      |            4 |
|       17 | Album 17    |        56 | 2021-08-16   | album      |            7 |
|       18 | Album 18    |        29 | 2021-08-16   | EP         |            3 |
|       19 | Album 19    |        42 | 2021-12-16   | album      |            2 |
|       20 | Album 20    |        23 | 2021-05-24   | single     |           16 |
|       21 | Album 21    |        57 | 2022-04-29   | album      |            7 |
|       22 | Album 22    |        35 | 2020-01-29   | album      |           19 |
|       23 | Album 23    |        57 | 2022-06-18   | single     |            7 |
|       24 | Album 24    |        46 | 2020-07-18   | EP         |           17 |
|       25 | Album 25    |        23 | 2021-11-19   | album      |           17 |
|       26 | Album 26    |        96 | 2022-07-03   | album      |            4 |
```

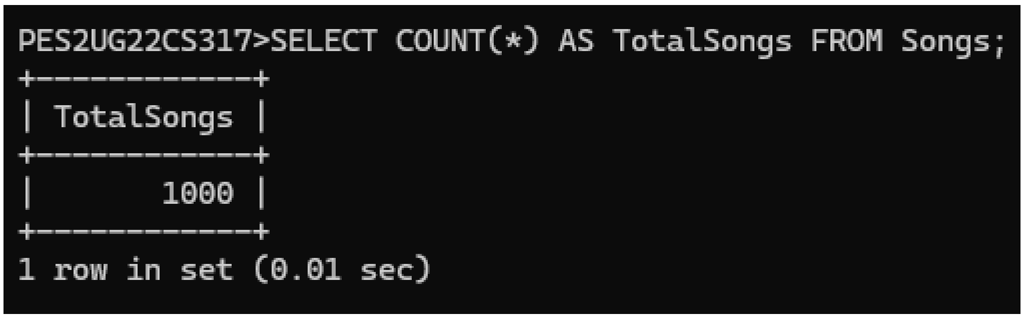


```
PES2UG22CS317>SELECT COUNT(*) AS TotalAlbums FROM Albums;
+-------------+
| TotalAlbums |
+-------------+
|         200 |
+-------------+
1 row in set (0.00 sec)
```

```
-- Select all data and count rows from the Songs table
SELECT * FROM Songs;
SELECT COUNT(*) AS TotalSongs FROM Songs;
```

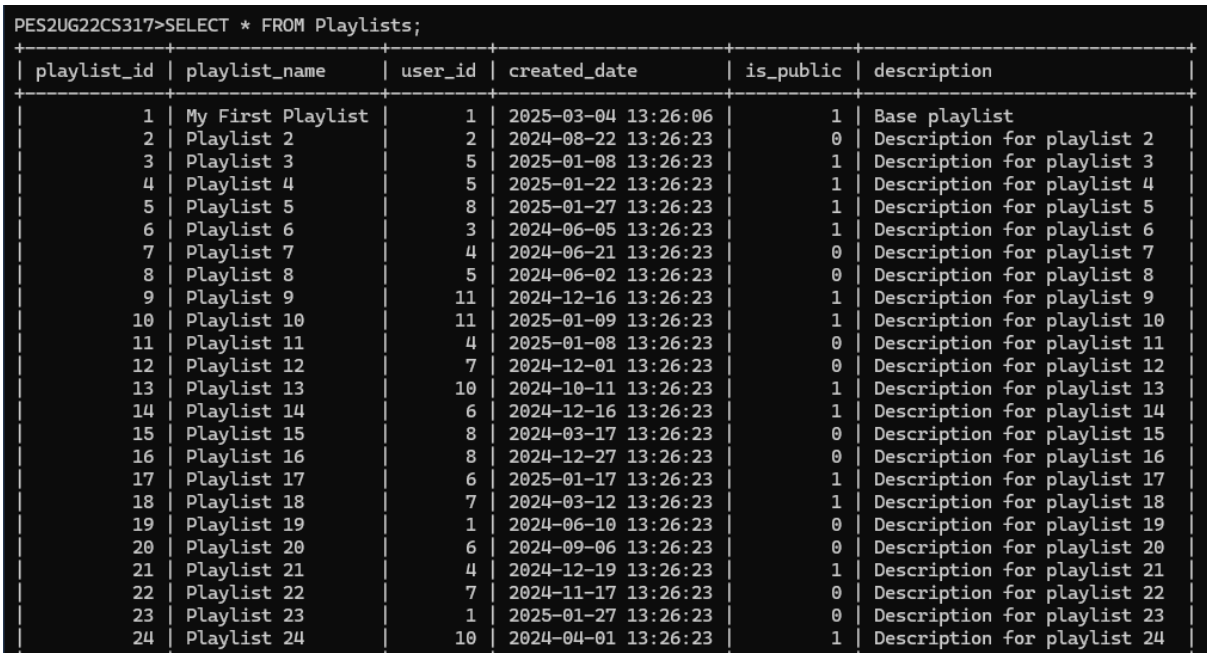


```
PES2UG22CS317>SELECT * FROM Songs;
+---------+------------+----------+----------+--------------+----------+------------+
| song_id | song_name  | album_id | duration | track_number | explicit | play_count |
+---------+------------+----------+----------+--------------+----------+------------+
|       1 | First Song |        1 |      180 |            1 |        0 |          0 |
|       2 | Song 2     |       13 |      275 |            5 |        0 |     408107 |
|       3 | Song 3     |       84 |      334 |            1 |        1 |     573281 |
|       4 | Song 4     |       15 |      351 |           18 |        1 |     233202 |
|       5 | Song 5     |       28 |      320 |            9 |        1 |     221579 |
|       6 | Song 6     |       99 |      247 |           13 |        1 |     580050 |
|       7 | Song 7     |       87 |      398 |           16 |        1 |     614107 |
|       8 | Song 8     |      181 |      177 |           19 |        0 |     244493 |
|       9 | Song 9     |      123 |      224 |           14 |        1 |     323477 |
|      10 | Song 10    |      154 |      245 |            4 |        1 |     193618 |
|      11 | Song 11    |      163 |      342 |           10 |        0 |     546319 |
|      12 | Song 12    |       92 |      368 |           13 |        0 |      56765 |
|      13 | Song 13    |       20 |      188 |           20 |        0 |     409654 |
|      14 | Song 14    |       63 |      140 |            1 |        0 |      82649 |
|      15 | Song 15    |      127 |      323 |           14 |        0 |     792287 |
|      16 | Song 16    |      104 |      346 |            2 |        0 |     354487 |
|      17 | Song 17    |       12 |      274 |           10 |        1 |      84960 |
|      18 | Song 18    |      138 |      151 |            5 |        0 |     472411 |
|      19 | Song 19    |      140 |      348 |            7 |        1 |     194444 |
|      20 | Song 20    |       31 |      180 |            2 |        1 |     337061 |
|      21 | Song 21    |       67 |      297 |           20 |        1 |     365932 |
|      22 | Song 22    |       94 |      162 |           16 |        1 |     390106 |
|      23 | Song 23    |       42 |      320 |            3 |        0 |     694948 |
|      24 | Song 24    |      172 |      363 |           13 |        0 |      53278 |
|      25 | Song 25    |       38 |      202 |            8 |        0 |     900066 |
|      26 | Song 26    |      124 |      124 |            8 |        0 |     257945 |
```

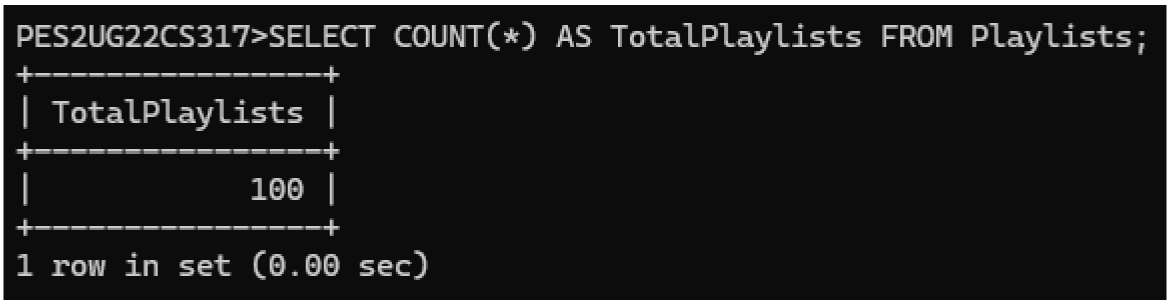


```
PES2UG22CS317>SELECT COUNT(*) AS TotalSongs FROM Songs;
+------------+
| TotalSongs |
+------------+
|       1000 |
+------------+
1 row in set (0.01 sec)
```

```
-- Select all data and count rows from the Playlists table
SELECT * FROM Playlists;
SELECT COUNT(*) AS TotalPlaylists FROM Playlists;
```

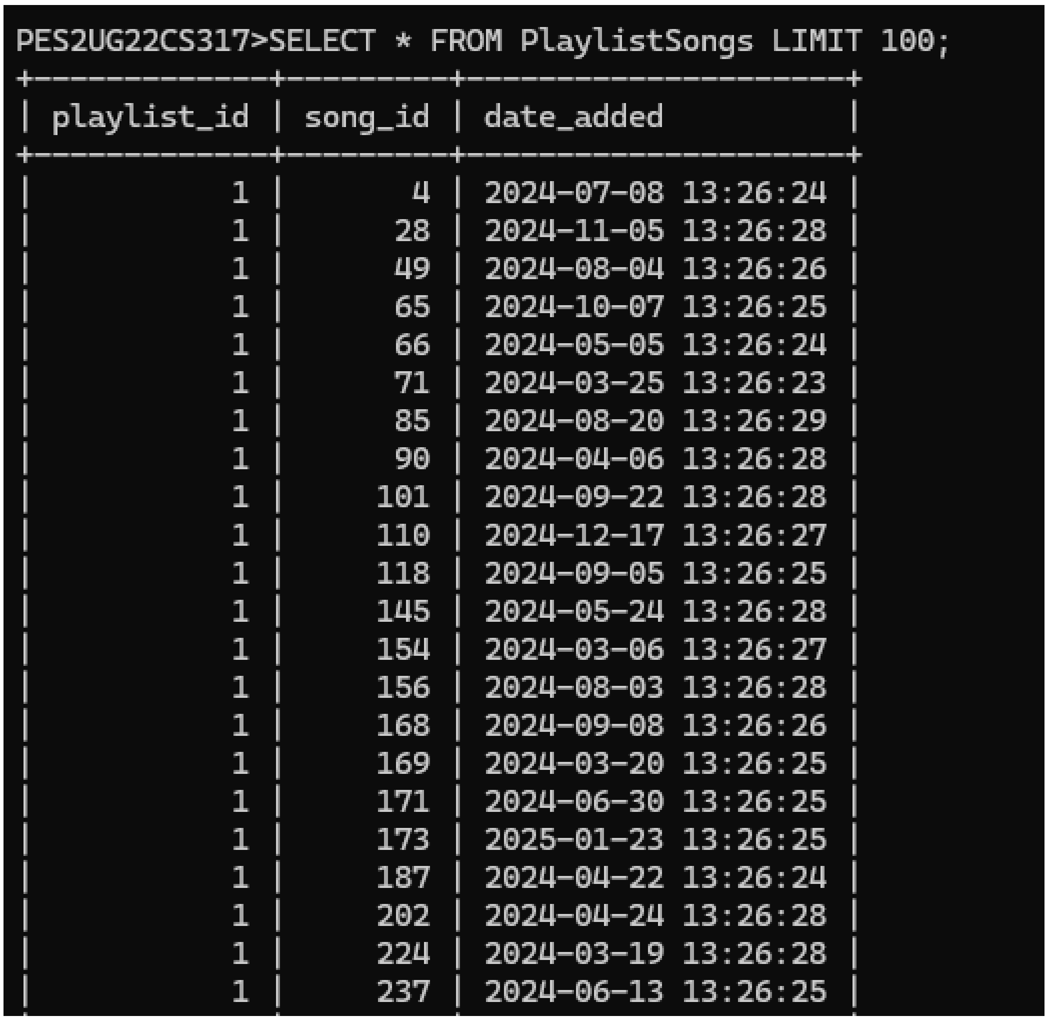


```
PES2UG22CS317>SELECT * FROM Playlists;
+-------------+------------------+---------+---------------------+-----------+---------------------------+
| playlist_id | playlist_name    | user_id | created_date        | is_public | description               |
+-------------+------------------+---------+---------------------+-----------+---------------------------+
|           1 | My First Playlist |       1 | 2025-03-04 13:26:06 |         1 | Base playlist             |
|           2 | Playlist 2       |       2 | 2024-08-22 13:26:23 |         0 | Description for playlist 2  |
|           3 | Playlist 3       |       5 | 2025-01-08 13:26:23 |         1 | Description for playlist 3  |
|           4 | Playlist 4       |       5 | 2025-01-22 13:26:23 |         1 | Description for playlist 4  |
|           5 | Playlist 5       |       8 | 2025-01-27 13:26:23 |         1 | Description for playlist 5  |
|           6 | Playlist 6       |       3 | 2024-06-05 13:26:23 |         1 | Description for playlist 6  |
|           7 | Playlist 7       |       4 | 2024-06-21 13:26:23 |         0 | Description for playlist 7  |
|           8 | Playlist 8       |       5 | 2024-06-02 13:26:23 |         0 | Description for playlist 8  |
|           9 | Playlist 9       |      11 | 2024-12-16 13:26:23 |         1 | Description for playlist 9  |
|          10 | Playlist 10      |      11 | 2025-01-09 13:26:23 |         1 | Description for playlist 10 |
|          11 | Playlist 11      |       4 | 2025-01-08 13:26:23 |         0 | Description for playlist 11 |
|          12 | Playlist 12      |       7 | 2024-12-01 13:26:23 |         0 | Description for playlist 12 |
|          13 | Playlist 13      |      10 | 2024-10-11 13:26:23 |         1 | Description for playlist 13 |
|          14 | Playlist 14      |       6 | 2024-12-16 13:26:23 |         1 | Description for playlist 14 |
|          15 | Playlist 15      |       8 | 2024-03-17 13:26:23 |         0 | Description for playlist 15 |
|          16 | Playlist 16      |       8 | 2024-12-27 13:26:23 |         0 | Description for playlist 16 |
|          17 | Playlist 17      |       6 | 2025-01-17 13:26:23 |         1 | Description for playlist 17 |
|          18 | Playlist 18      |       7 | 2024-03-12 13:26:23 |         1 | Description for playlist 18 |
|          19 | Playlist 19      |       1 | 2024-06-10 13:26:23 |         0 | Description for playlist 19 |
|          20 | Playlist 20      |       6 | 2024-09-06 13:26:23 |         0 | Description for playlist 20 |
|          21 | Playlist 21      |       4 | 2024-12-19 13:26:23 |         1 | Description for playlist 21 |
|          22 | Playlist 22      |       7 | 2024-11-17 13:26:23 |         0 | Description for playlist 22 |
|          23 | Playlist 23      |       1 | 2025-01-27 13:26:23 |         0 | Description for playlist 23 |
|          24 | Playlist 24      |      10 | 2024-04-01 13:26:23 |         1 | Description for playlist 24 |
```

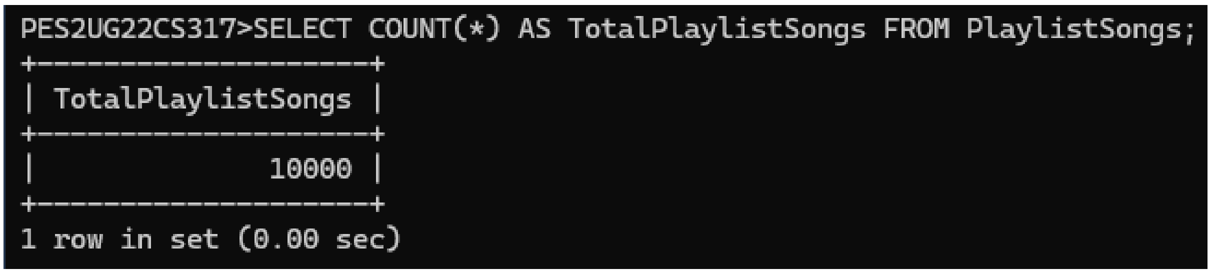


```
PES2UG22CS317>SELECT COUNT(*) AS TotalPlaylists FROM Playlists;
+----------------+
| TotalPlaylists |
+----------------+
|            100 |
+----------------+
1 row in set (0.00 sec)
```

```
-- Select all data and count rows from the PlaylistSongs table
SELECT * FROM PlaylistSongs;
SELECT COUNT(*) AS TotalPlaylistSongs FROM PlaylistSongs;


-- I have used limit 100 as there is large amount of data that couldn't be
displayed in a single command
```

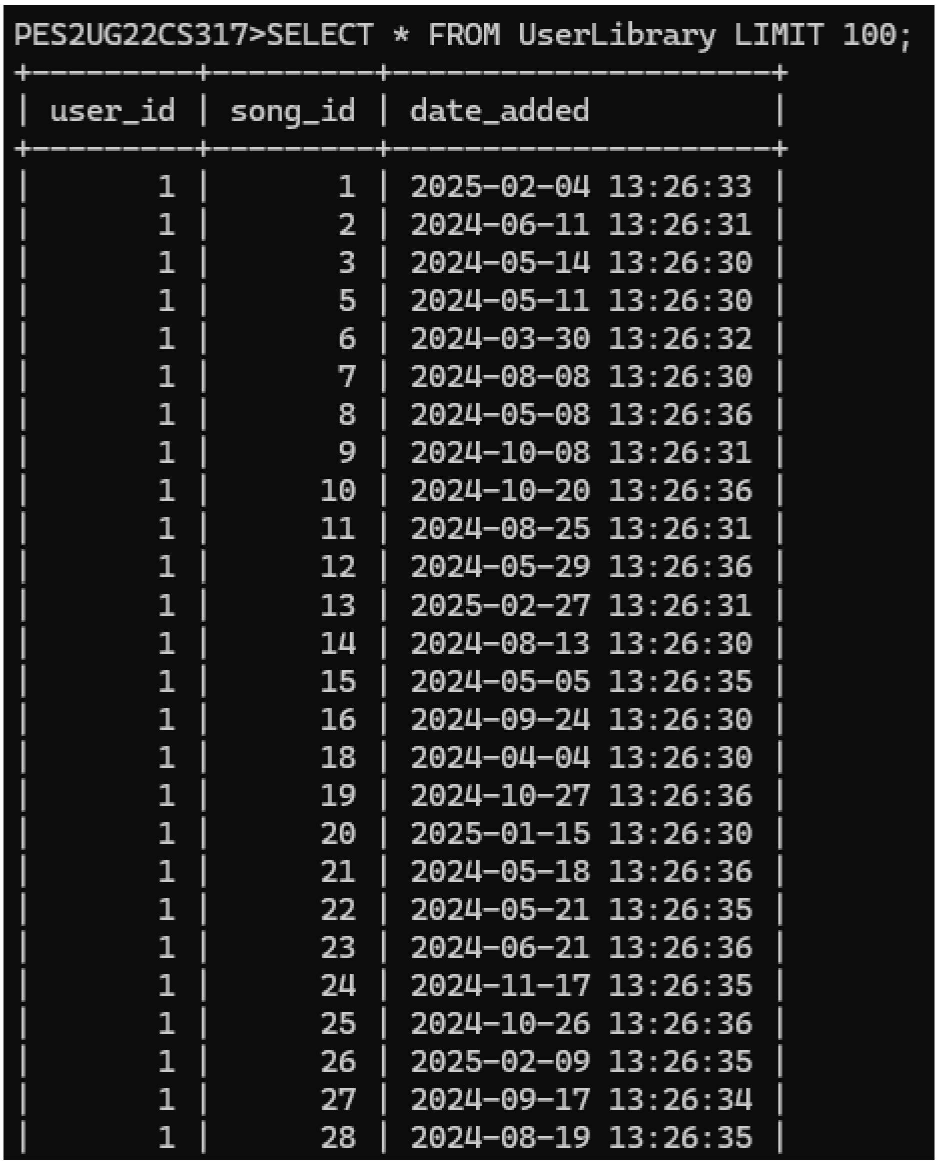


```
PES2UG22CS317>SELECT * FROM PlaylistSongs LIMIT 100;
+-------------+---------+---------------------+
| playlist_id | song_id | date_added          |
+-------------+---------+---------------------+
|           1 |       4 | 2024-07-08 13:26:24 |
|           1 |      28 | 2024-11-05 13:26:28 |
|           1 |      49 | 2024-08-04 13:26:26 |
|           1 |      65 | 2024-10-07 13:26:25 |
|           1 |      66 | 2024-05-05 13:26:24 |
|           1 |      71 | 2024-03-25 13:26:23 |
|           1 |      85 | 2024-08-20 13:26:29 |
|           1 |      90 | 2024-04-06 13:26:28 |
|           1 |     101 | 2024-09-22 13:26:28 |
|           1 |     110 | 2024-12-17 13:26:27 |
|           1 |     118 | 2024-09-05 13:26:25 |
|           1 |     145 | 2024-05-24 13:26:28 |
|           1 |     154 | 2024-03-06 13:26:27 |
|           1 |     156 | 2024-08-03 13:26:28 |
|           1 |     168 | 2024-09-08 13:26:26 |
|           1 |     169 | 2024-03-20 13:26:25 |
|           1 |     171 | 2024-06-30 13:26:25 |
|           1 |     173 | 2025-01-23 13:26:25 |
|           1 |     187 | 2024-04-22 13:26:24 |
|           1 |     202 | 2024-04-24 13:26:28 |
|           1 |     224 | 2024-03-19 13:26:28 |
|           1 |     237 | 2024-06-13 13:26:25 |
```

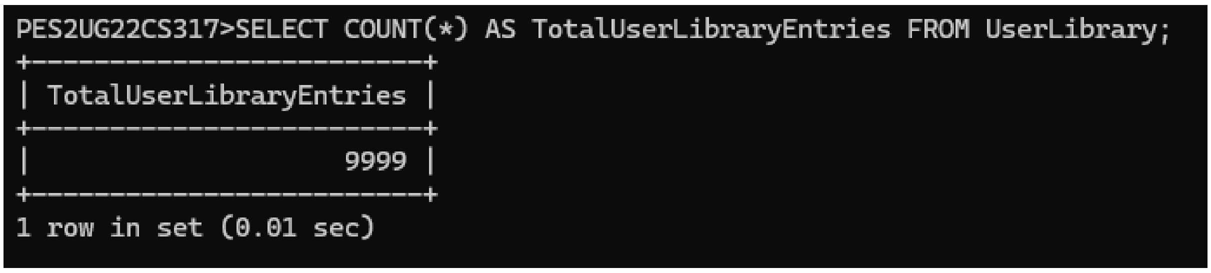


```
PES2UG22CS317>SELECT COUNT(*) AS TotalPlaylistSongs FROM PlaylistSongs;
+--------------------+
| TotalPlaylistSongs |
+--------------------+
|              10000 |
+--------------------+
1 row in set (0.00 sec)
```

```sql
-- Select all data and count rows from the UserLibrary table
SELECT * FROM UserLibrary;
SELECT COUNT(*) AS TotalUserLibraryEntries FROM UserLibrary;

-- I have used limit 100 as there is large amount of data that couldn’t be
displayed in a single command
```



```
PES2UG22CS317>SELECT * FROM UserLibrary LIMIT 100;
+---------+---------+---------------------+
| user_id | song_id | date_added          |
+---------+---------+---------------------+
|       1 |       1 | 2025-02-04 13:26:33 |
|       1 |       2 | 2024-06-11 13:26:31 |
|       1 |       3 | 2024-05-14 13:26:30 |
|       1 |       5 | 2024-05-11 13:26:30 |
|       1 |       6 | 2024-03-30 13:26:32 |
|       1 |       7 | 2024-08-08 13:26:30 |
|       1 |       8 | 2024-05-08 13:26:36 |
|       1 |       9 | 2024-10-08 13:26:31 |
|       1 |      10 | 2024-10-20 13:26:36 |
|       1 |      11 | 2024-08-25 13:26:31 |
|       1 |      12 | 2024-05-29 13:26:36 |
|       1 |      13 | 2025-02-27 13:26:31 |
|       1 |      14 | 2024-08-13 13:26:30 |
|       1 |      15 | 2024-05-05 13:26:35 |
|       1 |      16 | 2024-09-24 13:26:30 |
|       1 |      18 | 2024-04-04 13:26:30 |
|       1 |      19 | 2024-10-27 13:26:36 |
|       1 |      20 | 2025-01-15 13:26:30 |
|       1 |      21 | 2024-05-18 13:26:36 |
|       1 |      22 | 2024-05-21 13:26:35 |
|       1 |      23 | 2024-06-21 13:26:36 |
|       1 |      24 | 2024-11-17 13:26:35 |
|       1 |      25 | 2024-10-26 13:26:36 |
|       1 |      26 | 2025-02-09 13:26:35 |
|       1 |      27 | 2024-09-17 13:26:34 |
|       1 |      28 | 2024-08-19 13:26:35 |
```



```
PES2UG22CS317>SELECT COUNT(*) AS TotalUserLibraryEntries FROM UserLibrary;
+-------------------------+
| TotalUserLibraryEntries |
+-------------------------+
|                    9999 |
+-------------------------+
1 row in set (0.01 sec)
```

2. Points to note:

Craft a variety of queries to exercise both index scans and table scans.

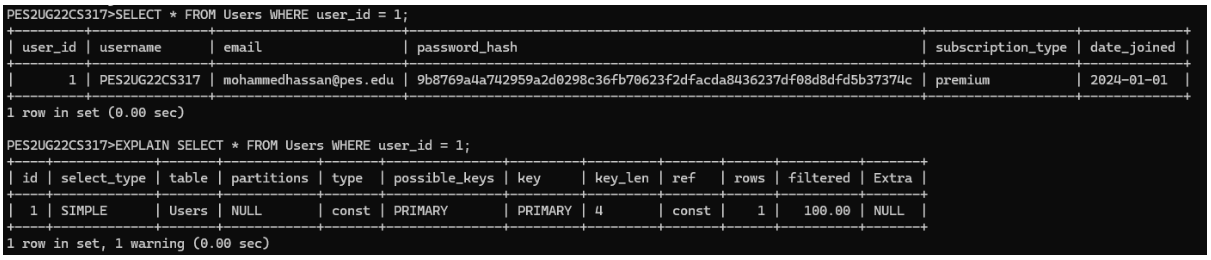
Also include queries with multi-table joins involving 3 tables; including both "SELECT *" and conditional "SELECT" queries with a subset of columns.

Run Explain/Analyze Plans for above queries and document each of them

Index Scan Examples:

Quarry 1: Using primary key (index scan)

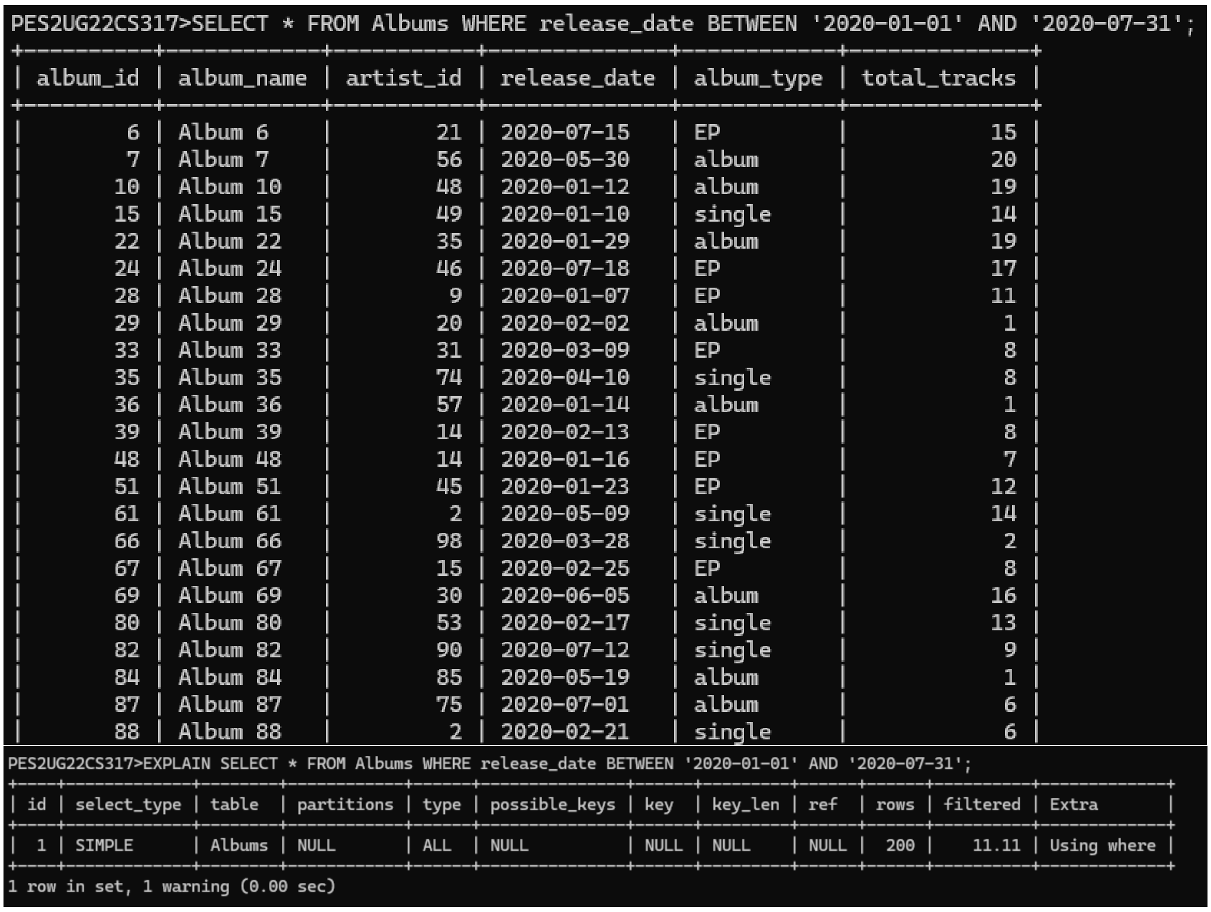
```
SELECT * FROM Users WHERE user_id = 1;

EXPLAIN SELECT * FROM Users WHERE user_id = 1;
```



```
PES2UG22CS317>SELECT * FROM Users WHERE user_id = 1;
+---------+---------------+----------------------+------------------------------------------------------------------+-------------------+-------------+
| user_id | username      | email                | password_hash                                                    | subscription_type | date_joined |
+---------+---------------+----------------------+------------------------------------------------------------------+-------------------+-------------+
|       1 | PES2UG22CS317 | mohammedhassan@pes.edu | 9b8769a4a742959a2d0298c36fb70623f2dfacda8436237df08d8dfd5b37374c | premium           | 2024-01-01  |
+---------+---------------+----------------------+------------------------------------------------------------------+-------------------+-------------+
1 row in set (0.00 sec)

PES2UG22CS317>EXPLAIN SELECT * FROM Users WHERE user_id = 1;
+----+-------------+-------+------------+-------+---------------+---------+---------+-------+------+----------+-------+
| id | select_type | table | partitions | type  | possible_keys | key     | key_len | ref   | rows | filtered | Extra |
+----+-------------+-------+------------+-------+---------------+---------+---------+-------+------+----------+-------+
|  1 | SIMPLE      | Users | NULL       | const | PRIMARY       | PRIMARY | 4       | const |    1 |   100.00 | NULL  |
+----+-------------+-------+------------+-------+---------------+---------+---------+-------+------+----------+-------+
1 row in set, 1 warning (0.00 sec)
```

Quarry 2: Range scan on date

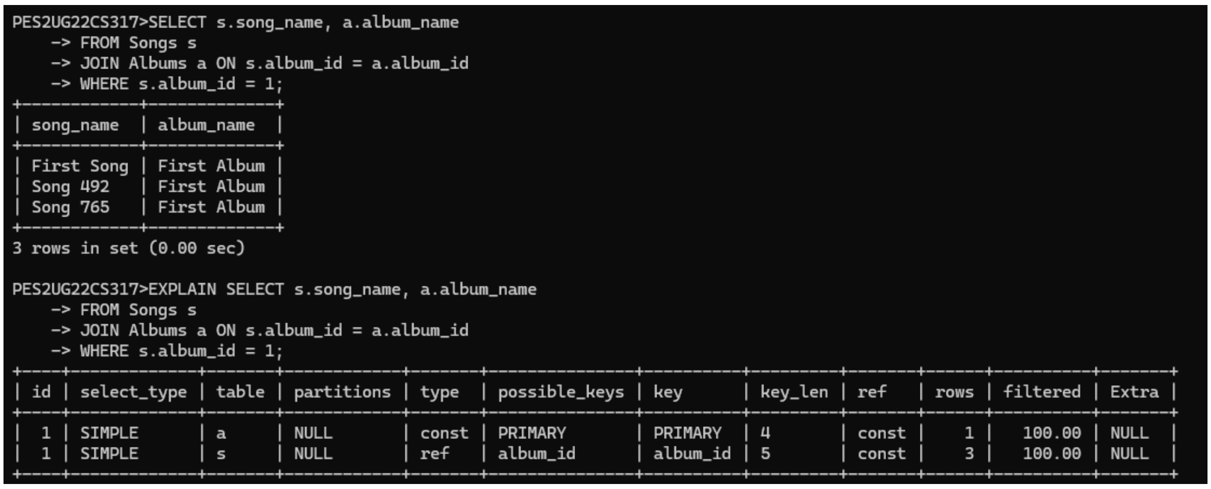
```
SELECT * FROM Albums WHERE release_date BETWEEN '2020-01-01' AND '2020-07-31';

EXPLAIN SELECT * FROM Albums WHERE release_date BETWEEN '2020-01-01' AND
'2020-07-31';
```



```
PES2UG22CS317>SELECT * FROM Albums WHERE release_date BETWEEN '2020-01-01' AND '2020-07-31';
+----------+------------+-----------+--------------+------------+--------------+
| album_id | album_name | artist_id | release_date | album_type | total_tracks |
+----------+------------+-----------+--------------+------------+--------------+
|        6 | Album 6    |        21 | 2020-07-15   | EP         |           15 |
|        7 | Album 7    |        56 | 2020-05-30   | album      |           20 |
|       10 | Album 10   |        48 | 2020-01-12   | album      |           19 |
|       15 | Album 15   |        49 | 2020-01-10   | single     |           14 |
|       22 | Album 22   |        35 | 2020-01-29   | album      |           19 |
|       24 | Album 24   |        46 | 2020-07-18   | EP         |           17 |
|       28 | Album 28   |         9 | 2020-01-07   | EP         |           11 |
|       29 | Album 29   |        20 | 2020-02-02   | album      |            1 |
|       33 | Album 33   |        31 | 2020-03-09   | EP         |            8 |
|       35 | Album 35   |        74 | 2020-04-10   | single     |            8 |
|       36 | Album 36   |        57 | 2020-01-14   | album      |            1 |
|       39 | Album 39   |        14 | 2020-02-13   | EP         |            8 |
|       48 | Album 48   |        14 | 2020-01-16   | EP         |            7 |
|       51 | Album 51   |        45 | 2020-01-23   | EP         |           12 |
|       61 | Album 61   |         2 | 2020-05-09   | single     |           14 |
|       66 | Album 66   |        98 | 2020-03-28   | single     |            2 |
|       67 | Album 67   |        15 | 2020-02-25   | EP         |            8 |
|       69 | Album 69   |        30 | 2020-06-05   | album      |           16 |
|       80 | Album 80   |        53 | 2020-02-17   | single     |           13 |
|       82 | Album 82   |        90 | 2020-07-12   | single     |            9 |
|       84 | Album 84   |        85 | 2020-05-19   | album      |            1 |
|       87 | Album 87   |        75 | 2020-07-01   | album      |            6 |
|       88 | Album 88   |         2 | 2020-02-21   | single     |            6 |
PES2UG22CS317>EXPLAIN SELECT * FROM Albums WHERE release_date BETWEEN '2020-01-01' AND '2020-07-31';
+----+-------------+--------+------------+------+---------------+------+---------+------+------+----------+-------------+
| id | select_type | table  | partitions | type | possible_keys | key  | key_len | ref  | rows | filtered | Extra       |
+----+-------------+--------+------------+------+---------------+------+---------+------+------+----------+-------------+
|  1 | SIMPLE      | Albums | NULL       | ALL  | NULL          | NULL | NULL    | NULL |  200 |    11.11 | Using where |
+----+-------------+--------+------------+------+---------------+------+---------+------+------+----------+-------------+
1 row in set, 1 warning (0.00 sec)
```

Quarry 3: Index scan on foreign key

```
SELECT s.song_name, a.album_name
FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
WHERE s.album_id = 1;

EXPLAIN SELECT s.song_name, a.album_name
FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
WHERE s.album_id = 1;
```

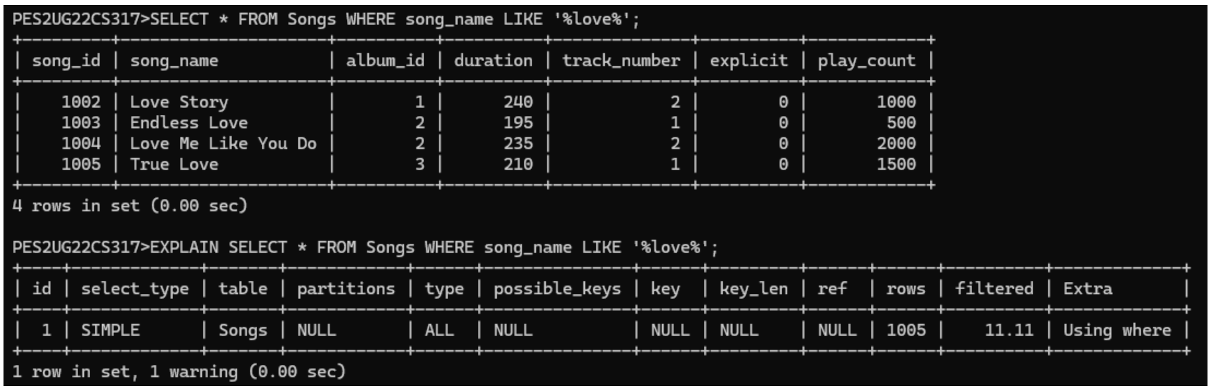


```
PES2UG22CS317>SELECT s.song_name, a.album_name
    -> FROM Songs s
    -> JOIN Albums a ON s.album_id = a.album_id
    -> WHERE s.album_id = 1;
+------------+-------------+
| song_name  | album_name  |
+------------+-------------+
| First Song | First Album |
| Song 492   | First Album |
| Song 765   | First Album |
+------------+-------------+
3 rows in set (0.00 sec)

PES2UG22CS317>EXPLAIN SELECT s.song_name, a.album_name
    -> FROM Songs s
    -> JOIN Albums a ON s.album_id = a.album_id
    -> WHERE s.album_id = 1;
+----+-------------+-------+------------+-------+---------------+----------+---------+-------+------+----------+-------+
| id | select_type | table | partitions | type  | possible_keys | key      | key_len | ref   | rows | filtered | Extra |
+----+-------------+-------+------------+-------+---------------+----------+---------+-------+------+----------+-------+
|  1 | SIMPLE      | a     | NULL       | const | PRIMARY       | PRIMARY  | 4       | const |    1 |   100.00 | NULL  |
|  1 | SIMPLE      | s     | NULL       | ref   | album_id      | album_id | 5       | const |    3 |   100.00 | NULL  |
+----+-------------+-------+------------+-------+---------------+----------+---------+-------+------+----------+-------+
```

Table Scan Examples

Quarry 4: Full table scan with LIKE

```
SELECT * FROM Songs WHERE LOWER(song_name) LIKE LOWER('%love%');
EXPLAIN SELECT * FROM Songs WHERE LOWER(song_name) LIKE LOWER('%love%');
```

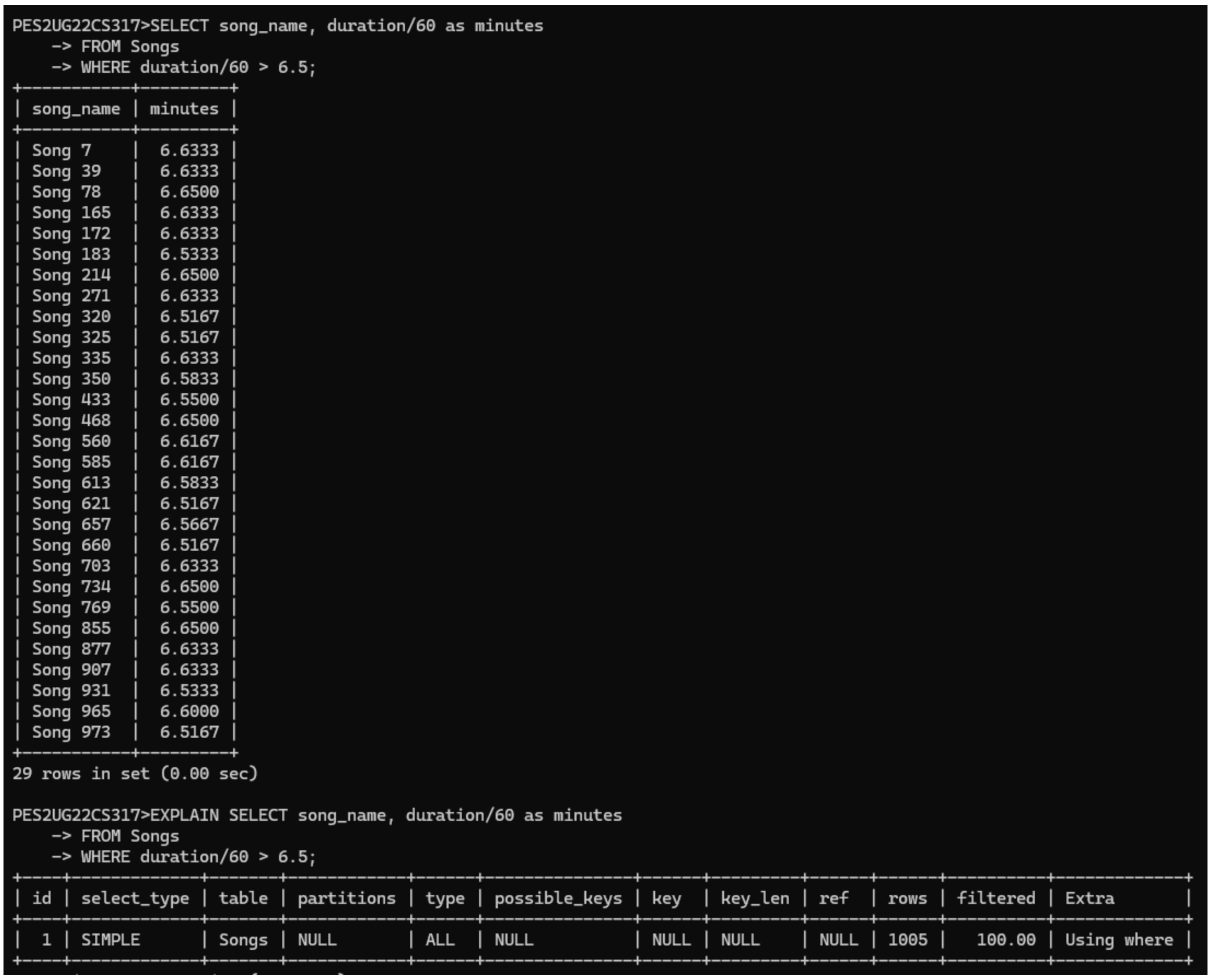


```
PES2UG22CS317>SELECT * FROM Songs WHERE song_name LIKE '%love%';
+---------+---------------------+----------+----------+--------------+----------+------------+
| song_id | song_name           | album_id | duration | track_number | explicit | play_count |
+---------+---------------------+----------+----------+--------------+----------+------------+
|    1002 | Love Story          |        1 |      240 |            2 |        0 |       1000 |
|    1003 | Endless Love        |        2 |      195 |            1 |        0 |        500 |
|    1004 | Love Me Like You Do |        2 |      235 |            2 |        0 |       2000 |
|    1005 | True Love           |        3 |      210 |            1 |        0 |       1500 |
+---------+---------------------+----------+----------+--------------+----------+------------+
4 rows in set (0.00 sec)

PES2UG22CS317>EXPLAIN SELECT * FROM Songs WHERE song_name LIKE '%love%';
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-------------+
| id | select_type | table | partitions | type | possible_keys | key  | key_len | ref  | rows | filtered | Extra       |
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-------------+
|  1 | SIMPLE      | Songs | NULL       | ALL  | NULL          | NULL | NULL    | NULL | 1005 |    11.11 | Using where |
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-------------+
1 row in set, 1 warning (0.00 sec)
```

Quarry 5: Table scan with calculation

```
SELECT song_name, duration/60 as minutes
FROM Songs
WHERE duration/60 > 6.5;

EXPLAIN SELECT song_name, duration/60 as minutes
```

```sql
FROM Songs
WHERE duration/60 > 6.5;
```

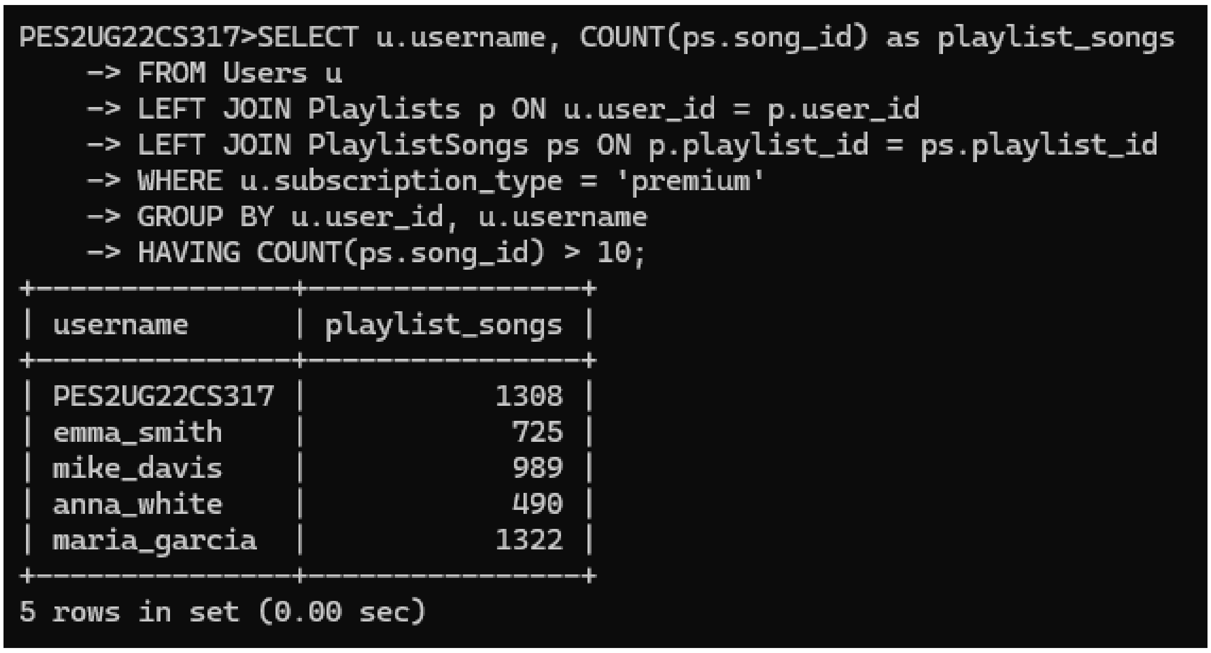


```
PES2UG22CS317>SELECT song_name, duration/60 as minutes
    -> FROM Songs
    -> WHERE duration/60 > 6.5;
+-----------+---------+
| song_name | minutes |
+-----------+---------+
| Song 7    |  6.6333 |
| Song 39   |  6.6333 |
| Song 78   |  6.6500 |
| Song 165  |  6.6333 |
| Song 172  |  6.6333 |
| Song 183  |  6.5333 |
| Song 214  |  6.6500 |
| Song 271  |  6.6333 |
| Song 320  |  6.5167 |
| Song 325  |  6.5167 |
| Song 335  |  6.6333 |
| Song 350  |  6.5833 |
| Song 433  |  6.5500 |
| Song 468  |  6.6500 |
| Song 560  |  6.6167 |
| Song 585  |  6.6167 |
| Song 613  |  6.5833 |
| Song 621  |  6.5167 |
| Song 657  |  6.5667 |
| Song 660  |  6.5167 |
| Song 703  |  6.6333 |
| Song 734  |  6.6500 |
| Song 769  |  6.5500 |
| Song 855  |  6.6500 |
| Song 877  |  6.6333 |
| Song 907  |  6.6333 |
| Song 931  |  6.5333 |
| Song 965  |  6.6000 |
| Song 973  |  6.5167 |
+-----------+---------+
29 rows in set (0.00 sec)

PES2UG22CS317>EXPLAIN SELECT song_name, duration/60 as minutes
    -> FROM Songs
    -> WHERE duration/60 > 6.5;
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-------------+
| id | select_type | table | partitions | type | possible_keys | key  | key_len | ref  | rows | filtered | Extra       |
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-------------+
|  1 | SIMPLE      | Songs | NULL       | ALL  | NULL          | NULL | NULL    | NULL | 1005 |   100.00 | Using where |
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-------------+
```

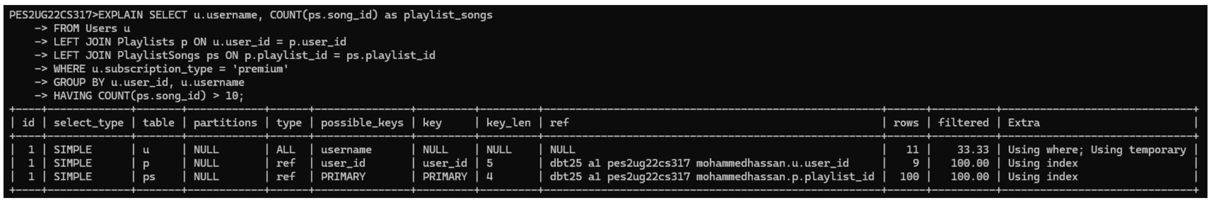
Mixed Scan Example

Quarry 6: Combination of index and table scan

```sql
SELECT u.username, COUNT(ps.song_id) as playlist_songs
FROM Users u
LEFT JOIN Playlists p ON u.user_id = p.user_id
LEFT JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
WHERE u.subscription_type = 'premium'
GROUP BY u.user_id, u.username
HAVING COUNT(ps.song_id) > 10;

EXPLAIN SELECT u.username, COUNT(ps.song_id) as playlist_songs
FROM Users u
LEFT JOIN Playlists p ON u.user_id = p.user_id
LEFT JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
WHERE u.subscription_type = 'premium'
GROUP BY u.user_id, u.username
HAVING COUNT(ps.song_id) > 10;
```



```
PES2UG22CS317>SELECT u.username, COUNT(ps.song_id) as playlist_songs
    -> FROM Users u
    -> LEFT JOIN Playlists p ON u.user_id = p.user_id
    -> LEFT JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
    -> WHERE u.subscription_type = 'premium'
    -> GROUP BY u.user_id, u.username
    -> HAVING COUNT(ps.song_id) > 10;
+----------------+----------------+
| username       | playlist_songs |
+----------------+----------------+
| PES2UG22CS317  |           1308 |
| emma_smith     |            725 |
| mike_davis     |            989 |
| anna_white     |            490 |
| maria_garcia   |           1322 |
+----------------+----------------+
5 rows in set (0.00 sec)
```

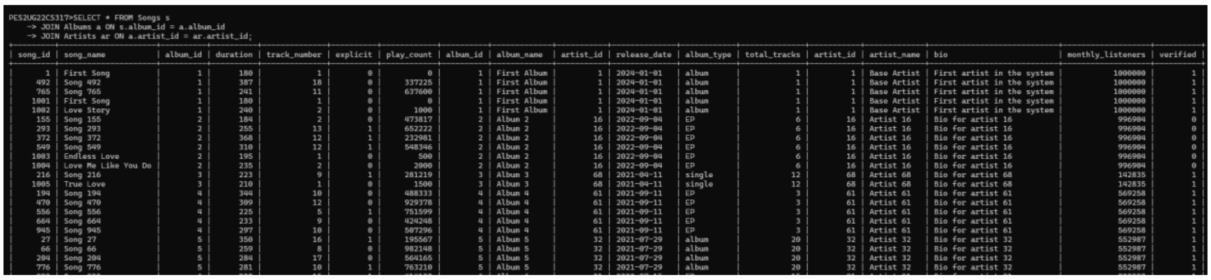


```
PES2UG22CS317>EXPLAIN SELECT u.username, COUNT(ps.song_id) as playlist_songs
    -> FROM Users u
    -> LEFT JOIN Playlists p ON u.user_id = p.user_id
    -> LEFT JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
    -> WHERE u.subscription_type = 'premium'
    -> GROUP BY u.user_id, u.username
    -> HAVING COUNT(ps.song_id) > 10;
+----+-------------+-------+------------+------+---------------+---------+---------+------------------------------------------------+------+----------+----------------------------------+
| id | select_type | table | partitions | type | possible_keys | key     | key_len | ref                                            | rows | filtered | Extra                            |
+----+-------------+-------+------------+------+---------------+---------+---------+------------------------------------------------+------+----------+----------------------------------+
|  1 | SIMPLE      | u     | NULL       | ALL  | username      | NULL    | NULL    | NULL                                           |   11 |    33.33 | Using where; Using temporary     |
|  1 | SIMPLE      | p     | NULL       | ref  | user_id       | user_id | 5       | dbt25 a1 pes2ug22cs317 mohammedhassan.u.user_id |    9 |   100.00 | Using index                      |
|  1 | SIMPLE      | ps    | NULL       | ref  | PRIMARY       | PRIMARY | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.p.playlist_id |  100 |   100.00 | Using index                  |
+----+-------------+-------+------------+------+---------------+---------+---------+------------------------------------------------+------+----------+----------------------------------+
```

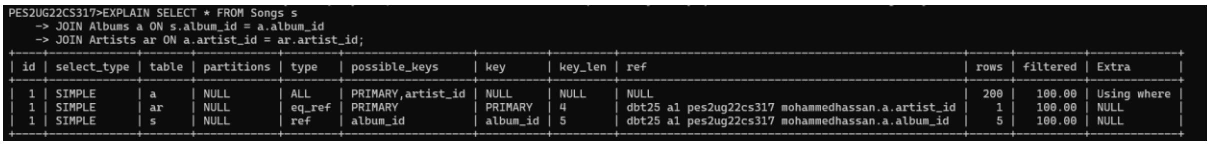
Multi-table join queries (3 tables)

Quarry 7: Get all songs with their album and artist details

```
SELECT * FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id;

EXPLAIN SELECT * FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id;
```



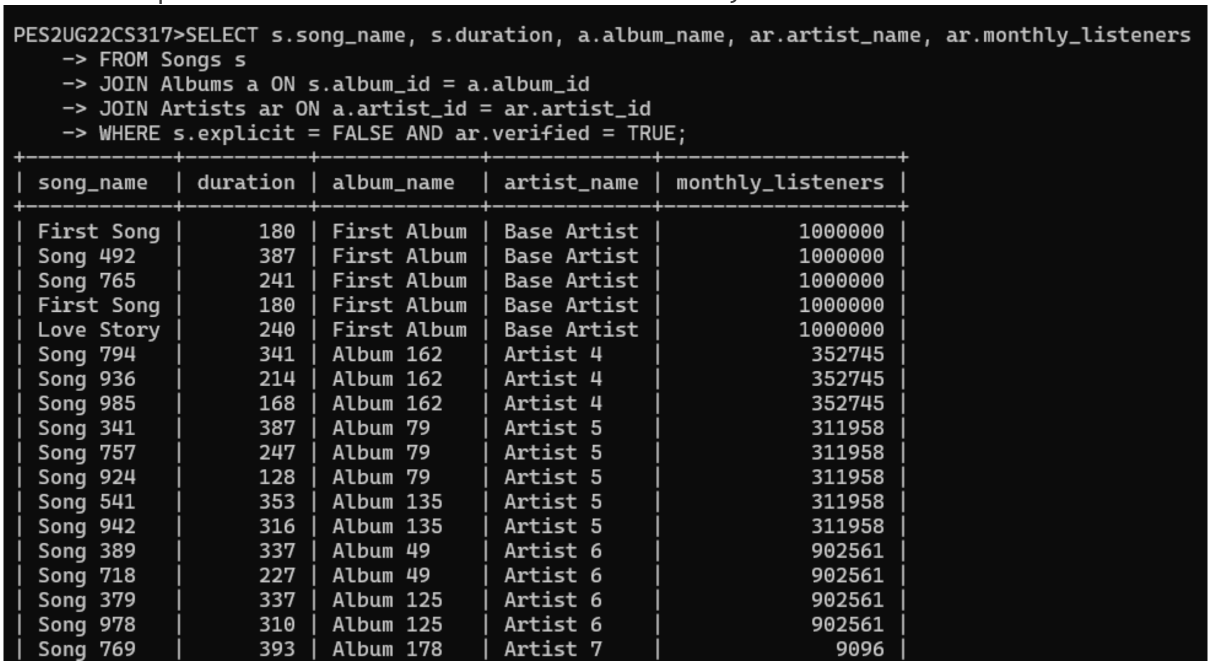
```
PES2UG22CS317>SELECT * FROM Songs s
    -> JOIN Albums a ON s.album_id = a.album_id
    -> JOIN Artists ar ON a.artist_id = ar.artist_id;
+---------+----------------------+----------+----------+--------------+----------+------------+----------+-------------+-----------+--------------+------------+--------------+-----------+-------------+-----------------------------+-------------------+----------+
| song_id | song_name            | album_id | duration | track_number | explicit | play_count | album_id | album_name  | artist_id | release_date | album_type | total_tracks | artist_id | artist_name | bio                         | monthly_listeners | verified |
+---------+----------------------+----------+----------+--------------+----------+------------+----------+-------------+-----------+--------------+------------+--------------+-----------+-------------+-----------------------------+-------------------+----------+
|       1 | First Song           |        1 |      180 |            1 |        0 |          0 |        1 | First Album |         1 | 2024-01-01   | album      |            1 |         1 | Base Artist | First artist in the system  |           1000000 |        1 |
|     492 | Song 492             |        1 |      387 |           18 |        0 |     337225 |        1 | First Album |         1 | 2024-01-01   | album      |            1 |         1 | Base Artist | First artist in the system  |           1000000 |        1 |
|     765 | Song 765             |        1 |      241 |           11 |        0 |     637600 |        1 | First Album |         1 | 2024-01-01   | album      |            1 |         1 | Base Artist | First artist in the system  |           1000000 |        1 |
|    1001 | First Song           |        1 |      180 |            1 |        0 |          0 |        1 | First Album |         1 | 2024-01-01   | album      |            1 |         1 | Base Artist | First artist in the system  |           1000000 |        1 |
|    1002 | Love Story           |        1 |      240 |            2 |        0 |       1000 |        1 | First Album |         1 | 2024-01-01   | album      |            1 |         1 | Base Artist | First artist in the system  |           1000000 |        1 |
|     155 | Song 155             |        2 |      184 |            2 |        0 |     473817 |        2 | Album 2     |        16 | 2022-09-04   | EP         |            6 |        16 | Artist 16   | Bio for artist 16           |            996984 |        0 |
|     293 | Song 293             |        2 |      255 |           13 |        1 |     652222 |        2 | Album 2     |        16 | 2022-09-04   | EP         |            6 |        16 | Artist 16   | Bio for artist 16           |            996984 |        0 |
|     372 | Song 372             |        2 |      368 |           12 |        1 |     232981 |        2 | Album 2     |        16 | 2022-09-04   | EP         |            6 |        16 | Artist 16   | Bio for artist 16           |            996984 |        0 |
|     549 | Song 549             |        2 |      310 |           12 |        1 |     548346 |        2 | Album 2     |        16 | 2022-09-04   | EP         |            6 |        16 | Artist 16   | Bio for artist 16           |            996984 |        0 |
|    1003 | Endless Love         |        2 |      195 |            1 |        0 |        500 |        2 | Album 2     |        16 | 2022-09-04   | EP         |            6 |        16 | Artist 16   | Bio for artist 16           |            996984 |        0 |
|    1004 | Love Me Like You Do  |        2 |      235 |            2 |        0 |       2000 |        2 | Album 2     |        16 | 2022-09-04   | EP         |            6 |        16 | Artist 16   | Bio for artist 16           |            996984 |        0 |
|     216 | Song 216             |        3 |      223 |            9 |        1 |     281219 |        3 | Album 3     |        68 | 2021-04-11   | single     |           12 |        68 | Artist 68   | Bio for artist 68           |            142835 |        1 |
|    1005 | True Love            |        3 |      210 |            1 |        0 |       1500 |        3 | Album 3     |        68 | 2021-04-11   | single     |           12 |        68 | Artist 68   | Bio for artist 68           |            142835 |        1 |
|     194 | Song 194             |        4 |      344 |           10 |        0 |     488333 |        4 | Album 4     |        61 | 2021-09-11   | EP         |            3 |        61 | Artist 61   | Bio for artist 61           |            569258 |        1 |
|     470 | Song 470             |        4 |      309 |           12 |        0 |     929378 |        4 | Album 4     |        61 | 2021-09-11   | EP         |            3 |        61 | Artist 61   | Bio for artist 61           |            569258 |        1 |
|     556 | Song 556             |        4 |      225 |            5 |        1 |     751599 |        4 | Album 4     |        61 | 2021-09-11   | EP         |            3 |        61 | Artist 61   | Bio for artist 61           |            569258 |        1 |
|     664 | Song 664             |        4 |      233 |            9 |        0 |     424248 |        4 | Album 4     |        61 | 2021-09-11   | EP         |            3 |        61 | Artist 61   | Bio for artist 61           |            569258 |        1 |
|     945 | Song 945             |        4 |      297 |           10 |        0 |     507296 |        4 | Album 4     |        61 | 2021-09-11   | EP         |            3 |        61 | Artist 61   | Bio for artist 61           |            569258 |        1 |
|      27 | Song 27              |        5 |      350 |           16 |        1 |     195567 |        5 | Album 5     |        32 | 2021-07-29   | album      |           20 |        32 | Artist 32   | Bio for artist 32           |            552987 |        1 |
|      66 | Song 66              |        5 |      259 |            8 |        0 |     982148 |        5 | Album 5     |        32 | 2021-07-29   | album      |           20 |        32 | Artist 32   | Bio for artist 32           |            552987 |        1 |
|     204 | Song 204             |        5 |      284 |           17 |        0 |     564165 |        5 | Album 5     |        32 | 2021-07-29   | album      |           20 |        32 | Artist 32   | Bio for artist 32           |            552987 |        1 |
|     776 | Song 776             |        5 |      281 |           10 |        1 |     763210 |        5 | Album 5     |        32 | 2021-07-29   | album      |           20 |        32 | Artist 32   | Bio for artist 32           |            552987 |        1 |
```



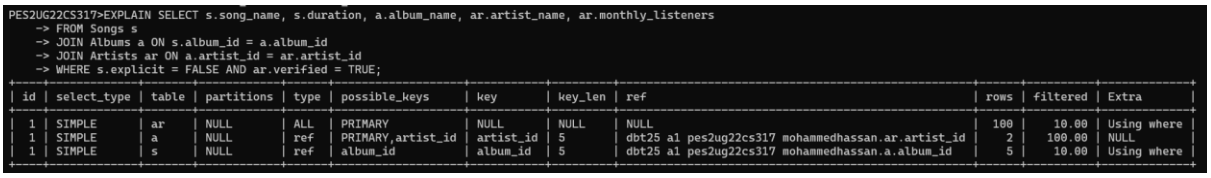
```
PES2UG22CS317>EXPLAIN SELECT * FROM Songs s
    -> JOIN Albums a ON s.album_id = a.album_id
    -> JOIN Artists ar ON a.artist_id = ar.artist_id;
+----+-------------+-------+------------+--------+-------------------+----------+---------+----------------------------------------------+------+----------+-------------+
| id | select_type | table | partitions | type   | possible_keys     | key      | key_len | ref                                          | rows | filtered | Extra       |
+----+-------------+-------+------------+--------+-------------------+----------+---------+----------------------------------------------+------+----------+-------------+
|  1 | SIMPLE      | a     | NULL       | ALL    | PRIMARY,artist_id | NULL     | NULL    | NULL                                         |  200 |   100.00 | Using where |
|  1 | SIMPLE      | ar    | NULL       | eq_ref | PRIMARY           | PRIMARY  | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.artist_id |    1 |   100.00 | NULL   |
|  1 | SIMPLE      | s     | NULL       | ref    | album_id          | album_id | 5       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.album_id  |    5 |   100.00 | NULL   |
+----+-------------+-------+------------+--------+-------------------+----------+---------+----------------------------------------------+------+----------+-------------+
```

Quarry 8: Select specific columns from songs, albums, and artists

```
SELECT s.song_name, s.duration, a.album_name, ar.artist_name,
ar.monthly_listeners
FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id
WHERE s.explicit = FALSE AND ar.verified = TRUE;

EXPLAIN SELECT s.song_name, s.duration, a.album_name, ar.artist_name,
ar.monthly_listeners
FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id
WHERE s.explicit = FALSE AND ar.verified = TRUE;
```



```
PES2UG22CS317>SELECT s.song_name, s.duration, a.album_name, ar.artist_name, ar.monthly_listeners
    -> FROM Songs s
    -> JOIN Albums a ON s.album_id = a.album_id
    -> JOIN Artists ar ON a.artist_id = ar.artist_id
    -> WHERE s.explicit = FALSE AND ar.verified = TRUE;
+------------+----------+-------------+-------------+-------------------+
| song_name  | duration | album_name  | artist_name | monthly_listeners |
+------------+----------+-------------+-------------+-------------------+
| First Song |      180 | First Album | Base Artist |           1000000 |
| Song 492   |      387 | First Album | Base Artist |           1000000 |
| Song 765   |      241 | First Album | Base Artist |           1000000 |
| First Song |      180 | First Album | Base Artist |           1000000 |
| Love Story |      240 | First Album | Base Artist |           1000000 |
| Song 794   |      341 | Album 162   | Artist 4    |            352745 |
| Song 936   |      214 | Album 162   | Artist 4    |            352745 |
| Song 985   |      168 | Album 162   | Artist 4    |            352745 |
| Song 341   |      387 | Album 79    | Artist 5    |            311958 |
| Song 757   |      247 | Album 79    | Artist 5    |            311958 |
| Song 924   |      128 | Album 79    | Artist 5    |            311958 |
| Song 541   |      353 | Album 135   | Artist 5    |            311958 |
| Song 942   |      316 | Album 135   | Artist 5    |            311958 |
| Song 389   |      337 | Album 49    | Artist 6    |            902561 |
| Song 718   |      227 | Album 49    | Artist 6    |            902561 |
| Song 379   |      337 | Album 125   | Artist 6    |            902561 |
| Song 978   |      310 | Album 125   | Artist 6    |            902561 |
| Song 769   |      393 | Album 178   | Artist 7    |              9096 |
```

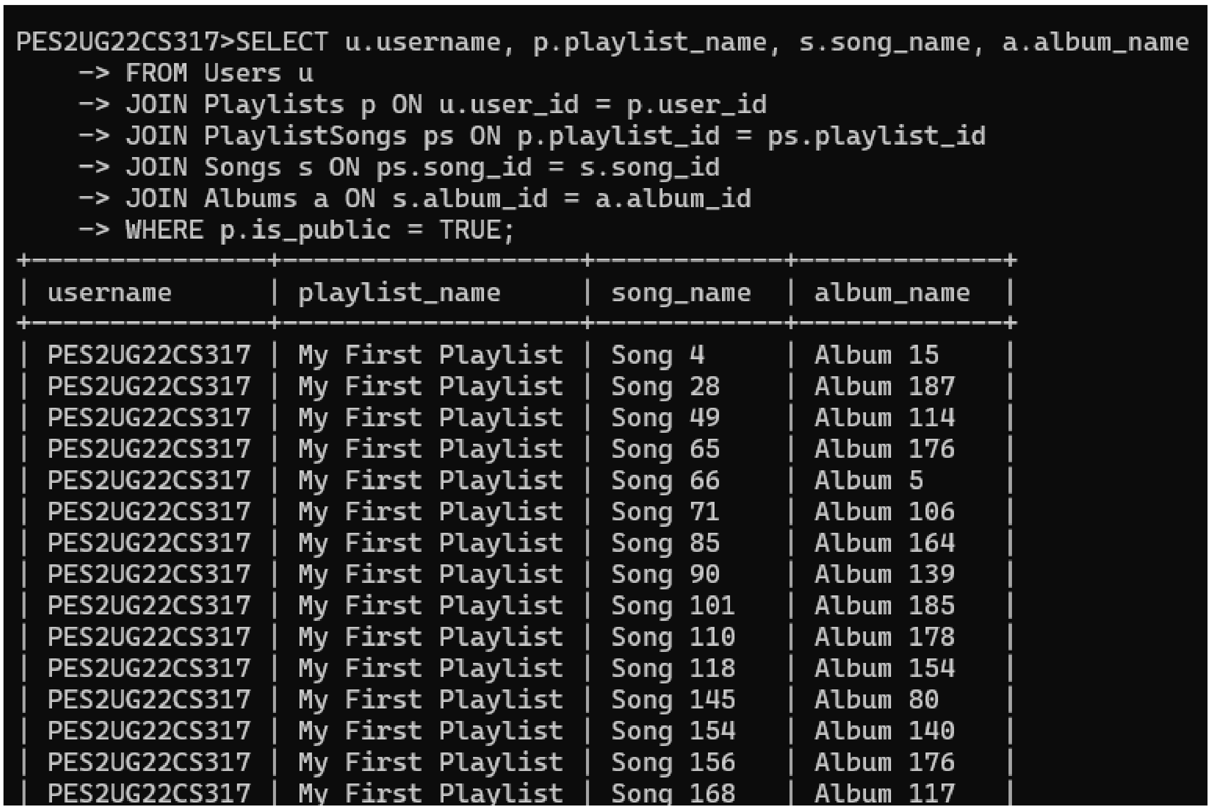


```
PES2UG22CS317>EXPLAIN SELECT s.song_name, s.duration, a.album_name, ar.artist_name, ar.monthly_listeners
    -> FROM Songs s
    -> JOIN Albums a ON s.album_id = a.album_id
    -> JOIN Artists ar ON a.artist_id = ar.artist_id
    -> WHERE s.explicit = FALSE AND ar.verified = TRUE;
+----+-------------+-------+------------+------+-------------------+-----------+---------+----------------------------------------------+------+----------+-------------+
| id | select_type | table | partitions | type | possible_keys     | key       | key_len | ref                                          | rows | filtered | Extra       |
+----+-------------+-------+------------+------+-------------------+-----------+---------+----------------------------------------------+------+----------+-------------+
|  1 | SIMPLE      | ar    | NULL       | ALL  | PRIMARY           | NULL      | NULL    | NULL                                         |  100 |    10.00 | Using where |
|  1 | SIMPLE      | a     | NULL       | ref  | PRIMARY,artist_id | artist_id | 5       | dbt25 a1 pes2ug22cs317 mohammedhassan.ar.artist_id |    2 |   100.00 | NULL        |
|  1 | SIMPLE      | s     | NULL       | ref  | album_id          | album_id  | 5       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.album_id  |    5 |    10.00 | Using where |
+----+-------------+-------+------------+------+-------------------+-----------+---------+----------------------------------------------+------+----------+-------------+
```

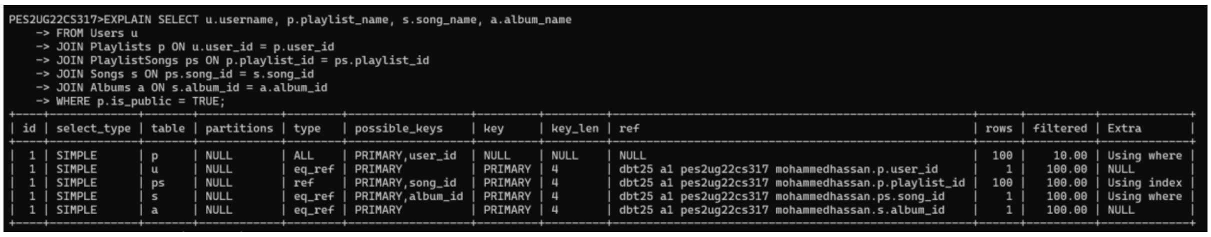
Quarry 9: Find user playlist details with song and album information

```
SELECT u.username, p.playlist_name, s.song_name, a.album_name
FROM Users u
JOIN Playlists p ON u.user_id = p.user_id
JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
JOIN Songs s ON ps.song_id = s.song_id
JOIN Albums a ON s.album_id = a.album_id
WHERE p.is_public = TRUE;
```

```sql
EXPLAIN SELECT u.username, p.playlist_name, s.song_name, a.album_name
FROM Users u
JOIN Playlists p ON u.user_id = p.user_id
JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
JOIN Songs s ON ps.song_id = s.song_id
JOIN Albums a ON s.album_id = a.album_id
WHERE p.is_public = TRUE;
```



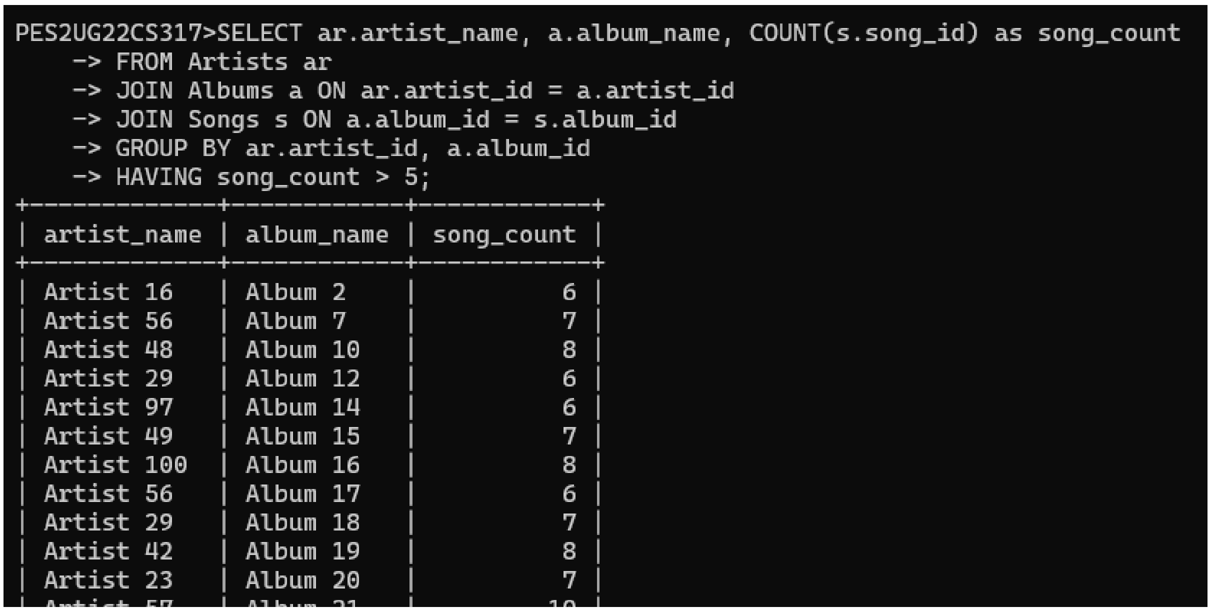
```
PES2UG22CS317>SELECT u.username, p.playlist_name, s.song_name, a.album_name
    -> FROM Users u
    -> JOIN Playlists p ON u.user_id = p.user_id
    -> JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
    -> JOIN Songs s ON ps.song_id = s.song_id
    -> JOIN Albums a ON s.album_id = a.album_id
    -> WHERE p.is_public = TRUE;
+---------------+-------------------+------------+-------------+
| username      | playlist_name     | song_name  | album_name  |
+---------------+-------------------+------------+-------------+
| PES2UG22CS317 | My First Playlist | Song 4     | Album 15    |
| PES2UG22CS317 | My First Playlist | Song 28    | Album 187   |
| PES2UG22CS317 | My First Playlist | Song 49    | Album 114   |
| PES2UG22CS317 | My First Playlist | Song 65    | Album 176   |
| PES2UG22CS317 | My First Playlist | Song 66    | Album 5     |
| PES2UG22CS317 | My First Playlist | Song 71    | Album 106   |
| PES2UG22CS317 | My First Playlist | Song 85    | Album 164   |
| PES2UG22CS317 | My First Playlist | Song 90    | Album 139   |
| PES2UG22CS317 | My First Playlist | Song 101   | Album 185   |
| PES2UG22CS317 | My First Playlist | Song 110   | Album 178   |
| PES2UG22CS317 | My First Playlist | Song 118   | Album 154   |
| PES2UG22CS317 | My First Playlist | Song 145   | Album 80    |
| PES2UG22CS317 | My First Playlist | Song 154   | Album 140   |
| PES2UG22CS317 | My First Playlist | Song 156   | Album 176   |
| PES2UG22CS317 | My First Playlist | Song 168   | Album 117   |
```



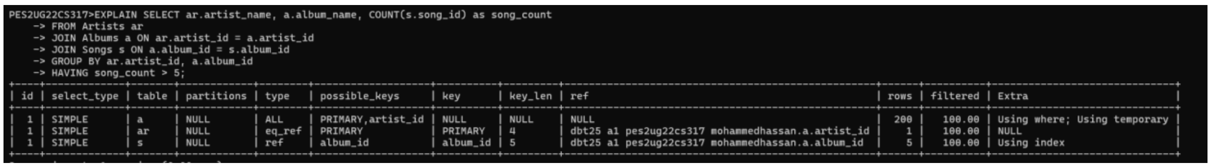
```
PES2UG22CS317>EXPLAIN SELECT u.username, p.playlist_name, s.song_name, a.album_name
    -> FROM Users u
    -> JOIN Playlists p ON u.user_id = p.user_id
    -> JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
    -> JOIN Songs s ON ps.song_id = s.song_id
    -> JOIN Albums a ON s.album_id = a.album_id
    -> WHERE p.is_public = TRUE;
+----+-------------+-------+------------+--------+-------------------+---------+---------+---------------------------------------------------+------+----------+-------------+
| id | select_type | table | partitions | type   | possible_keys     | key     | key_len | ref                                               | rows | filtered | Extra       |
+----+-------------+-------+------------+--------+-------------------+---------+---------+---------------------------------------------------+------+----------+-------------+
|  1 | SIMPLE      | p     | NULL       | ALL    | PRIMARY,user_id   | NULL    | NULL    | NULL                                              |  100 |    10.00 | Using where |
|  1 | SIMPLE      | u     | NULL       | eq_ref | PRIMARY           | PRIMARY | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.p.user_id     |    1 |   100.00 | NULL        |
|  1 | SIMPLE      | ps    | NULL       | ref    | PRIMARY,song_id   | PRIMARY | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.p.playlist_id |  100 |   100.00 | Using index |
|  1 | SIMPLE      | s     | NULL       | eq_ref | PRIMARY,album_id  | PRIMARY | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.ps.song_id    |    1 |   100.00 | Using where |
|  1 | SIMPLE      | a     | NULL       | eq_ref | PRIMARY           | PRIMARY | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.s.album_id    |    1 |   100.00 | NULL        |
+----+-------------+-------+------------+--------+-------------------+---------+---------+---------------------------------------------------+------+----------+-------------+
```

Quarry 10: Count songs per artist with album details

```sql
SELECT ar.artist_name, a.album_name, COUNT(s.song_id) as song_count
FROM Artists ar
JOIN Albums a ON ar.artist_id = a.artist_id
JOIN Songs s ON a.album_id = s.album_id
GROUP BY ar.artist_id, a.album_id
HAVING song_count > 5;

EXPLAIN SELECT ar.artist_name, a.album_name, COUNT(s.song_id) as song_count
FROM Artists ar
JOIN Albums a ON ar.artist_id = a.artist_id
JOIN Songs s ON a.album_id = s.album_id
GROUP BY ar.artist_id, a.album_id
HAVING song_count > 5;
```



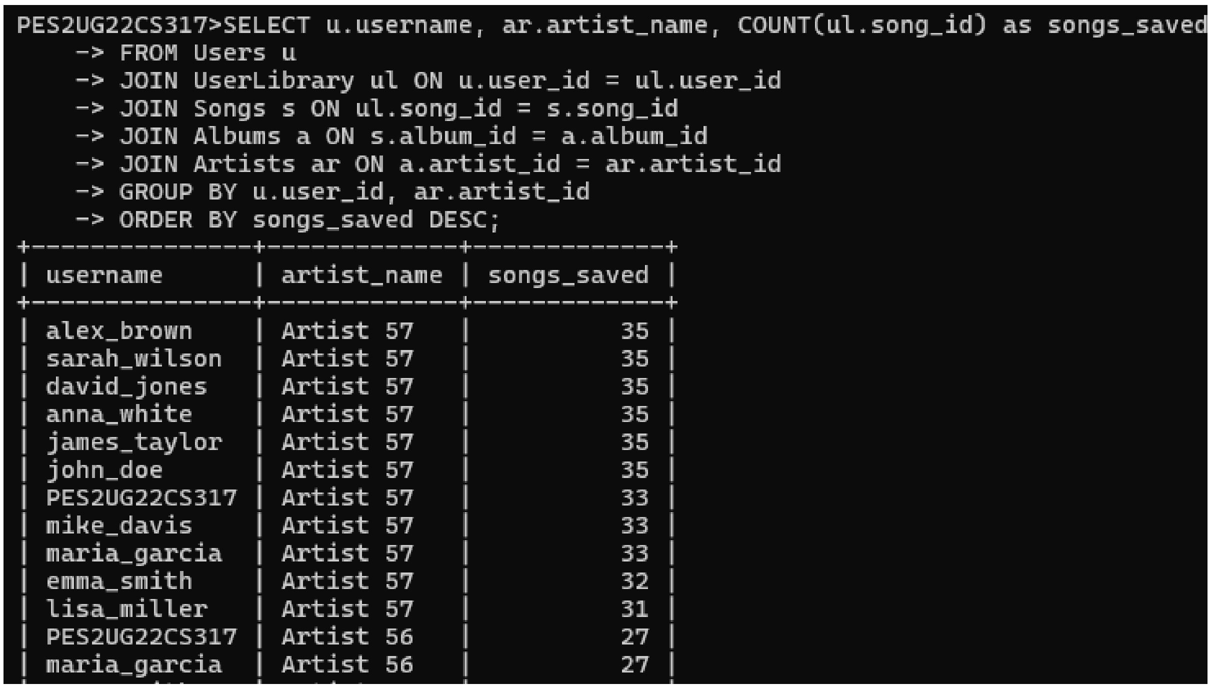
```
PES2UG22CS317>SELECT ar.artist_name, a.album_name, COUNT(s.song_id) as song_count
    -> FROM Artists ar
    -> JOIN Albums a ON ar.artist_id = a.artist_id
    -> JOIN Songs s ON a.album_id = s.album_id
    -> GROUP BY ar.artist_id, a.album_id
    -> HAVING song_count > 5;
+-------------+------------+------------+
| artist_name | album_name | song_count |
+-------------+------------+------------+
| Artist 16   | Album 2    |          6 |
| Artist 56   | Album 7    |          7 |
| Artist 48   | Album 10   |          8 |
| Artist 29   | Album 12   |          6 |
| Artist 97   | Album 14   |          6 |
| Artist 49   | Album 15   |          7 |
| Artist 100  | Album 16   |          8 |
| Artist 56   | Album 17   |          6 |
| Artist 29   | Album 18   |          7 |
| Artist 42   | Album 19   |          8 |
| Artist 23   | Album 20   |          7 |
```



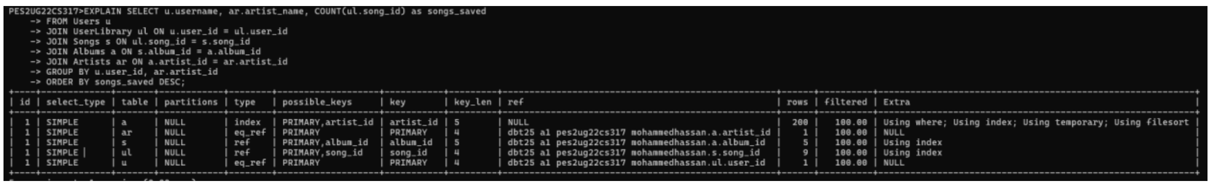
```
PES2UG22CS317>EXPLAIN SELECT ar.artist_name, a.album_name, COUNT(s.song_id) as song_count
    -> FROM Artists ar
    -> JOIN Albums a ON ar.artist_id = a.artist_id
    -> JOIN Songs s ON a.album_id = s.album_id
    -> GROUP BY ar.artist_id, a.album_id
    -> HAVING song_count > 5;
+----+-------------+-------+------------+--------+--------------------+----------+---------+-------------------------------------------+------+----------+------------------------------+
| id | select_type | table | partitions | type   | possible_keys      | key      | key_len | ref                                       | rows | filtered | Extra                        |
+----+-------------+-------+------------+--------+--------------------+----------+---------+-------------------------------------------+------+----------+------------------------------+
|  1 | SIMPLE      | a     | NULL       | ALL    | PRIMARY,artist_id  | NULL     | NULL    | NULL                                      |  200 |   100.00 | Using where; Using temporary |
|  1 | SIMPLE      | ar    | NULL       | eq_ref | PRIMARY            | PRIMARY  | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.artist_id |    1 |   100.00 | NULL                         |
|  1 | SIMPLE      | s     | NULL       | ref    | album_id           | album_id | 5       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.album_id  |    5 |   100.00 | Using index                  |
+----+-------------+-------+------------+--------+--------------------+----------+---------+-------------------------------------------+------+----------+------------------------------+
```

Quarry 11: User library analysis

```sql
SELECT u.username, ar.artist_name, COUNT(ul.song_id) as songs_saved
FROM Users u
JOIN UserLibrary ul ON u.user_id = ul.user_id
JOIN Songs s ON ul.song_id = s.song_id
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id
GROUP BY u.user_id, ar.artist_id
ORDER BY songs_saved DESC;


EXPLAIN SELECT u.username, ar.artist_name, COUNT(ul.song_id) as songs_saved
FROM Users u
JOIN UserLibrary ul ON u.user_id = ul.user_id
JOIN Songs s ON ul.song_id = s.song_id
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id
GROUP BY u.user_id, ar.artist_id
ORDER BY songs_saved DESC;
```



```
PES2UG22CS317>SELECT u.username, ar.artist_name, COUNT(ul.song_id) as songs_saved
    -> FROM Users u
    -> JOIN UserLibrary ul ON u.user_id = ul.user_id
    -> JOIN Songs s ON ul.song_id = s.song_id
    -> JOIN Albums a ON s.album_id = a.album_id
    -> JOIN Artists ar ON a.artist_id = ar.artist_id
    -> GROUP BY u.user_id, ar.artist_id
    -> ORDER BY songs_saved DESC;
+---------------+-------------+-------------+
| username      | artist_name | songs_saved |
+---------------+-------------+-------------+
| alex_brown    | Artist 57   |          35 |
| sarah_wilson  | Artist 57   |          35 |
| david_jones   | Artist 57   |          35 |
| anna_white    | Artist 57   |          35 |
| james_taylor  | Artist 57   |          35 |
| john_doe      | Artist 57   |          35 |
| PES2UG22CS317 | Artist 57   |          33 |
| mike_davis    | Artist 57   |          33 |
| maria_garcia  | Artist 57   |          33 |
| emma_smith    | Artist 57   |          32 |
| lisa_miller   | Artist 57   |          31 |
| PES2UG22CS317 | Artist 56   |          27 |
| maria_garcia  | Artist 56   |          27 |
```



```
PES2UG22CS317>EXPLAIN SELECT u.username, ar.artist_name, COUNT(ul.song_id) as songs_saved
    -> FROM Users u
    -> JOIN UserLibrary ul ON u.user_id = ul.user_id
    -> JOIN Songs s ON ul.song_id = s.song_id
    -> JOIN Albums a ON s.album_id = a.album_id
    -> JOIN Artists ar ON a.artist_id = ar.artist_id
    -> GROUP BY u.user_id, ar.artist_id
    -> ORDER BY songs_saved DESC;
+----+-------------+-------+------------+--------+-------------------+-----------+---------+----------------------------------------------------+------+----------+----------------------------------------------------------+
| id | select_type | table | partitions | type   | possible_keys     | key       | key_len | ref                                                | rows | filtered | Extra                                                    |
+----+-------------+-------+------------+--------+-------------------+-----------+---------+----------------------------------------------------+------+----------+----------------------------------------------------------+
|  1 | SIMPLE      | a     | NULL       | index  | PRIMARY,artist_id | artist_id | 5       | NULL                                               |  200 |   100.00 | Using where; Using index; Using temporary; Using filesort |
|  1 | SIMPLE      | ar    | NULL       | eq_ref | PRIMARY           | PRIMARY   | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.artist_id  |    1 |   100.00 | NULL                                                     |
|  1 | SIMPLE      | s     | NULL       | ref    | PRIMARY,album_id  | album_id  | 5       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.album_id   |    5 |   100.00 | Using index                                              |
|  1 | SIMPLE |     | ul    | NULL       | ref    | PRIMARY,song_id   | song_id   | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.s.song_id    |    9 |   100.00 | Using index                                              |
|  1 | SIMPLE      | u     | NULL       | eq_ref | PRIMARY           | PRIMARY   | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.ul.user_id   |    1 |   100.00 | NULL                                                     |
+----+-------------+-------+------------+--------+-------------------+-----------+---------+----------------------------------------------------+------+----------+----------------------------------------------------------+
```

(c) Indexing for Query Performance Improvement:

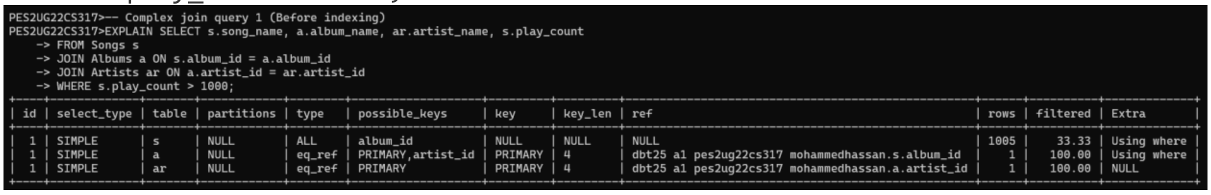
1. Create an optimal number of indexes on different tables within the selected mini-world database, focusing on larger tables for significant performance gains.

2. After index creation, run Explain/Analyze Plans on select queries and compare the results with the previous Explain Plans, particularly emphasizing the impact on multi-table joins involving 3 tables to demonstrate the effect of indexing on query performance.

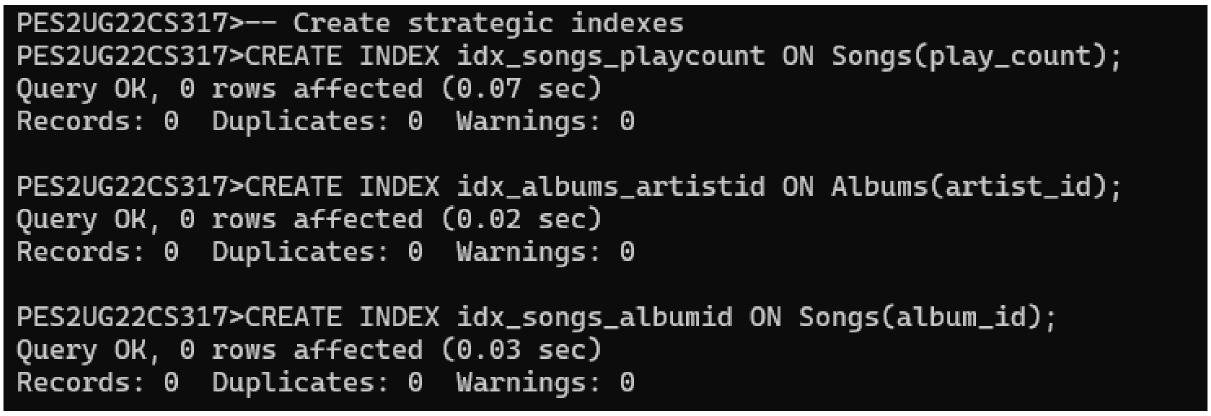
Example 1:

```
-- Complex join query 1 (Before indexing)
EXPLAIN SELECT s.song_name, a.album_name, ar.artist_name, s.play_count
FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id
WHERE s.play_count > 1000;

-- Create strategic indexes
CREATE INDEX idx_songs_playcount ON Songs(play_count);
CREATE INDEX idx_albums_artistid ON Albums(artist_id);
CREATE INDEX idx_songs_albumid ON Songs(album_id);
```

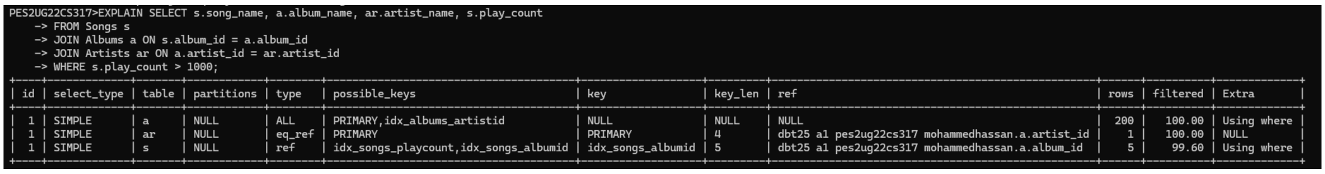
```sql
-- Compare execution plan after indexing
-- Complex join query 1 (After indexing)
EXPLAIN SELECT s.song_name, a.album_name, ar.artist_name, s.play_count
FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id
WHERE s.play_count > 1000;
```



```
PES2UG22CS317>-- Complex join query 1 (Before indexing)
PES2UG22CS317>EXPLAIN SELECT s.song_name, a.album_name, ar.artist_name, s.play_count
    -> FROM Songs s
    -> JOIN Albums a ON s.album_id = a.album_id
    -> JOIN Artists ar ON a.artist_id = ar.artist_id
    -> WHERE s.play_count > 1000;
+----+-------------+-------+------------+--------+--------------------+---------+---------+-------------------------------------------+------+----------+-------------+
| id | select_type | table | partitions | type   | possible_keys      | key     | key_len | ref                                       | rows | filtered | Extra       |
+----+-------------+-------+------------+--------+--------------------+---------+---------+-------------------------------------------+------+----------+-------------+
|  1 | SIMPLE      | s     | NULL       | ALL    | album_id           | NULL    | NULL    | NULL                                      | 1005 |    33.33 | Using where |
|  1 | SIMPLE      | a     | NULL       | eq_ref | PRIMARY,artist_id  | PRIMARY | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.s.album_id |    1 |   100.00 | Using where |
|  1 | SIMPLE      | ar    | NULL       | eq_ref | PRIMARY            | PRIMARY | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.artist_id |    1 |   100.00 | NULL        |
+----+-------------+-------+------------+--------+--------------------+---------+---------+-------------------------------------------+------+----------+-------------+
```



```
PES2UG22CS317>-- Create strategic indexes
PES2UG22CS317>CREATE INDEX idx_songs_playcount ON Songs(play_count);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

PES2UG22CS317>CREATE INDEX idx_albums_artistid ON Albums(artist_id);
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0

PES2UG22CS317>CREATE INDEX idx_songs_albumid ON Songs(album_id);
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

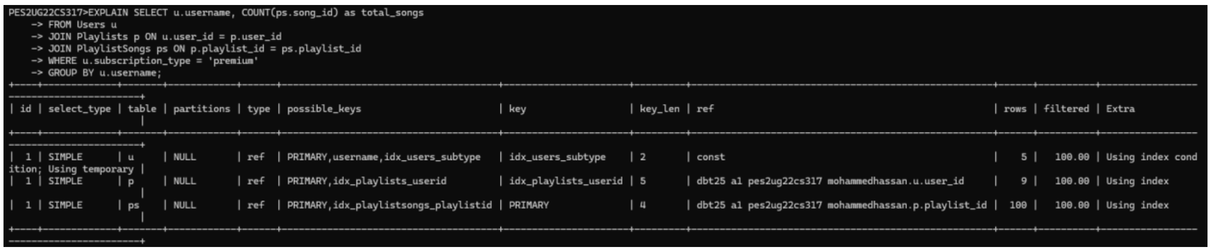


```
PES2UG22CS317>EXPLAIN SELECT s.song_name, a.album_name, ar.artist_name, s.play_count
    -> FROM Songs s
    -> JOIN Albums a ON s.album_id = a.album_id
    -> JOIN Artists ar ON a.artist_id = ar.artist_id
    -> WHERE s.play_count > 1000;
+----+-------------+-------+------------+--------+---------------------------------------+-------------------+---------+---------------------------------------------------+------+----------+-------------+
| id | select_type | table | partitions | type   | possible_keys                         | key               | key_len | ref                                               | rows | filtered | Extra       |
+----+-------------+-------+------------+--------+---------------------------------------+-------------------+---------+---------------------------------------------------+------+----------+-------------+
|  1 | SIMPLE      | a     | NULL       | ALL    | PRIMARY,idx_albums_artistid           | NULL              | NULL    | NULL                                              |  200 |   100.00 | Using where |
|  1 | SIMPLE      | ar    | NULL       | eq_ref | PRIMARY                               | PRIMARY           | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.artist_id |    1 |   100.00 | NULL        |
|  1 | SIMPLE      | s     | NULL       | ref    | idx_songs_playcount,idx_songs_albumid | idx_songs_albumid | 5       | dbt25 a1 pes2ug22cs317 mohammedhassan.a.album_id  |    5 |    99.60 | Using where |
+----+-------------+-------+------------+--------+---------------------------------------+-------------------+---------+---------------------------------------------------+------+----------+-------------+
```

Example 2:

```sql
-- Before indexing
EXPLAIN SELECT u.username, COUNT(ps.song_id) as total_songs
FROM Users u
JOIN Playlists p ON u.user_id = p.user_id
JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
WHERE u.subscription_type = 'premium'
GROUP BY u.username;

-- Create relevant indexes
CREATE INDEX idx_users_subtype ON Users(subscription_type);
CREATE INDEX idx_playlists_userid ON Playlists(user_id);
CREATE INDEX idx_playlistsongs_playlistid ON PlaylistSongs(playlist_id);

-- After indexing
EXPLAIN SELECT u.username, COUNT(ps.song_id) as total_songs
FROM Users u
JOIN Playlists p ON u.user_id = p.user_id
JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
```

```sql
WHERE u.subscription_type = 'premium'
GROUP BY u.username;
```



```
PES2UG22CS317>EXPLAIN SELECT u.username, COUNT(ps.song_id) as total_songs
    -> FROM Users u
    -> JOIN Playlists p ON u.user_id = p.user_id
    -> JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
    -> WHERE u.subscription_type = 'premium'
    -> GROUP BY u.username;
+----+-------------+-------+------------+------+-----------------------------------+----------------------+---------+--------------------------------------------------+------+----------+-----------------
-----------------------+
| id | select_type | table | partitions | type | possible_keys                     | key                  | key_len | ref                                              | rows | filtered | Extra
                       |
+----+-------------+-------+------------+------+-----------------------------------+----------------------+---------+--------------------------------------------------+------+----------+-----------------
-----------------------+
|  1 | SIMPLE      | u     | NULL       | ref  | PRIMARY,username,idx_users_subtype | idx_users_subtype   | 2       | const                                            |    5 |   100.00 | Using index cond
ition; Using temporary |
|  1 | SIMPLE      | p     | NULL       | ref  | PRIMARY,idx_playlists_userid      | idx_playlists_userid | 5       | dbt25 a1 pes2ug22cs317 mohammedhassan.u.user_id  |    9 |   100.00 | Using index
                       |
|  1 | SIMPLE      | ps    | NULL       | ref  | PRIMARY,idx_playlistsongs_playlistid | PRIMARY           | 4       | dbt25 a1 pes2ug22cs317 mohammedhassan.p.playlist_id |  100 |   100.00 | Using index
                       |
+----+-------------+-------+------------+------+-----------------------------------+----------------------+---------+--------------------------------------------------+------+----------+-----------------
-----------------------+
```

(d) Query Optimization with Varied Join Orders and Types

1Explore various optimization strategies by altering the join order of tables in multi-table join queries at least 2 times.

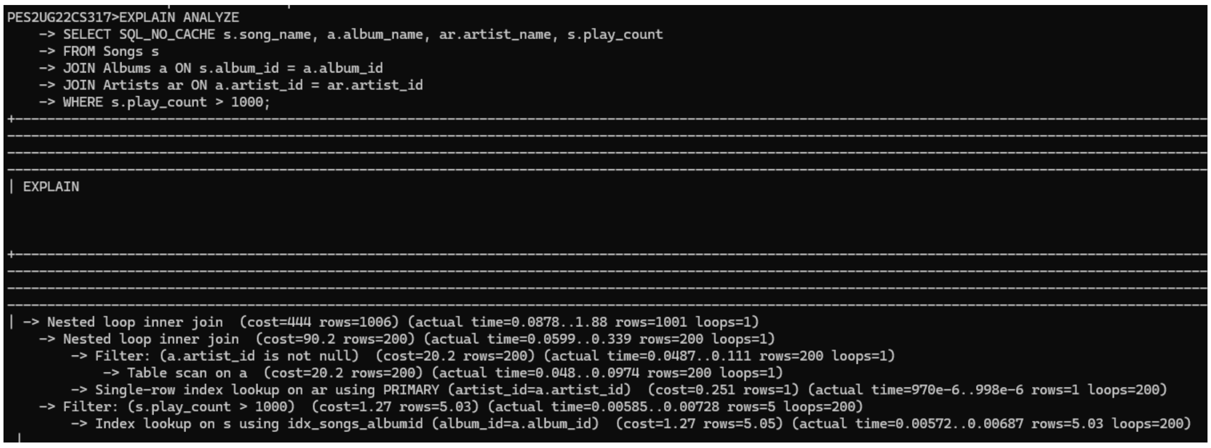
2 Incorporate a variety of join types such as outer joins, subqueries, etc., to diversify optimization approaches.
3 Analyze performance differences by comparing execution plans and actual execution performance.
4 Measure query execution time before and after optimization to quantify improvements accurately.

```sql
-- Enable profiling to measure execution time
SET profiling = 1;

-- Original Query 1: Basic join order (Songs -> Albums -> Artists)
SELECT SQL_NO_CACHE s.song_name, a.album_name, ar.artist_name, s.play_count
FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id
WHERE s.play_count > 1000;

-- Optimized Query 1: Changed join order (Artists -> Albums -> Songs)
SELECT SQL_NO_CACHE s.song_name, a.album_name, ar.artist_name, s.play_count
FROM Artists ar
JOIN Albums a ON ar.artist_id = a.artist_id
JOIN Songs s ON a.album_id = s.album_id
WHERE s.play_count > 1000;

-- Compare execution plans
EXPLAIN ANALYZE
SELECT SQL_NO_CACHE s.song_name, a.album_name, ar.artist_name, s.play_count
FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id
WHERE s.play_count > 1000;
```

```sql
EXPLAIN ANALYZE
SELECT SQL_NO_CACHE s.song_name, a.album_name, ar.artist_name, s.play_count
FROM Artists ar
JOIN Albums a ON ar.artist_id = a.artist_id
JOIN Songs s ON a.album_id = s.album_id
WHERE s.play_count > 1000;

-- Original Query 2: Simple joins
SELECT SQL_NO_CACHE u.username, p.playlist_name, COUNT(ps.song_id) as
song_count
FROM Users u
JOIN Playlists p ON u.user_id = p.user_id
JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
GROUP BY u.username, p.playlist_name;

-- Optimized Query 2: Using LEFT JOINs and subquery
SELECT SQL_NO_CACHE u.username, p.playlist_name,
    (SELECT COUNT(*)
     FROM PlaylistSongs ps
     WHERE ps.playlist_id = p.playlist_id) as song_count
FROM Users u
LEFT JOIN Playlists p ON u.user_id = p.user_id
WHERE u.subscription_type = 'premium';

-- Compare execution plans
EXPLAIN ANALYZE
SELECT SQL_NO_CACHE u.username, p.playlist_name, COUNT(ps.song_id) as
song_count
FROM Users u
JOIN Playlists p ON u.user_id = p.user_id
JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
GROUP BY u.username, p.playlist_name;

EXPLAIN ANALYZE
SELECT SQL_NO_CACHE u.username, p.playlist_name,
    (SELECT COUNT(*)
     FROM PlaylistSongs ps
     WHERE ps.playlist_id = p.playlist_id) as song_count
FROM Users u
LEFT JOIN Playlists p ON u.user_id = p.user_id
WHERE u.subscription_type = 'premium';

-- Show execution times
SHOW PROFILES;

-- Disable profiling
SET profiling = 0;
```

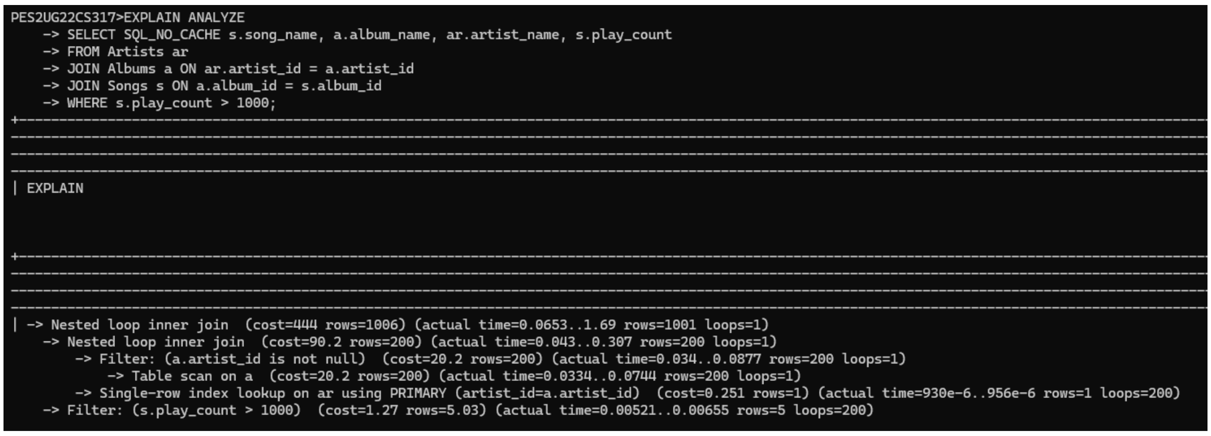
Compare execution plans for query 1:

Before optimization of query 1



```
PES2UG22CS317>EXPLAIN ANALYZE
    -> SELECT SQL_NO_CACHE s.song_name, a.album_name, ar.artist_name, s.play_count
    -> FROM Songs s
    -> JOIN Albums a ON s.album_id = a.album_id
    -> JOIN Artists ar ON a.artist_id = ar.artist_id
    -> WHERE s.play_count > 1000;
+------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
| EXPLAIN



+------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
| -> Nested loop inner join  (cost=444 rows=1006) (actual time=0.0878..1.88 rows=1001 loops=1)
    -> Nested loop inner join  (cost=90.2 rows=200) (actual time=0.0599..0.339 rows=200 loops=1)
        -> Filter: (a.artist_id is not null)  (cost=20.2 rows=200) (actual time=0.0487..0.111 rows=200 loops=1)
            -> Table scan on a  (cost=20.2 rows=200) (actual time=0.048..0.0974 rows=200 loops=1)
        -> Single-row index lookup on ar using PRIMARY (artist_id=a.artist_id)  (cost=0.251 rows=1) (actual time=970e-6..998e-6 rows=1 loops=200)
    -> Filter: (s.play_count > 1000)  (cost=1.27 rows=5.03) (actual time=0.00585..0.00728 rows=5 loops=200)
        -> Index lookup on s using idx_songs_albumid (album_id=a.album_id)  (cost=1.27 rows=5.05) (actual time=0.00572..0.00687 rows=5.03 loops=200)
 |
```

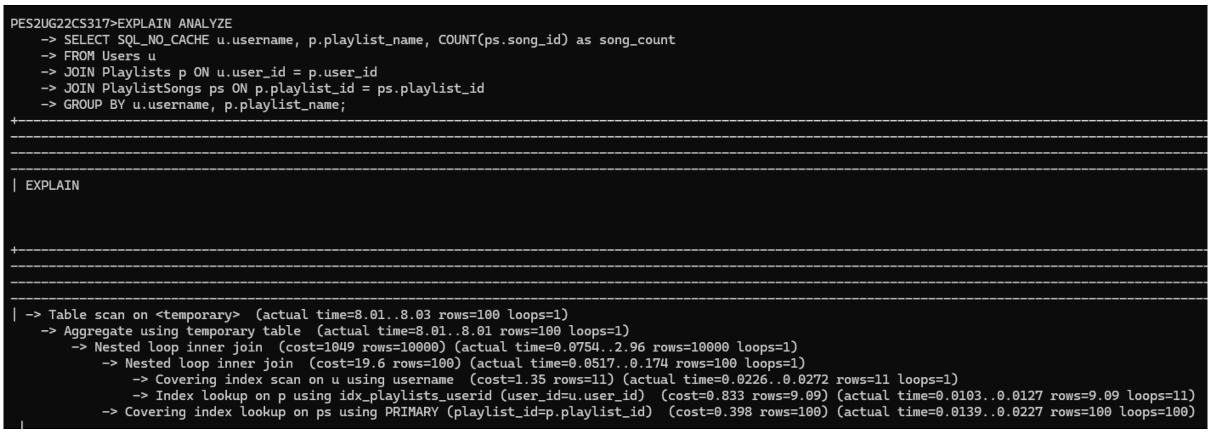
After optimization of query 1



```
PES2UG22CS317>EXPLAIN ANALYZE
    -> SELECT SQL_NO_CACHE s.song_name, a.album_name, ar.artist_name, s.play_count
    -> FROM Artists ar
    -> JOIN Albums a ON ar.artist_id = a.artist_id
    -> JOIN Songs s ON a.album_id = s.album_id
    -> WHERE s.play_count > 1000;
+------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
| EXPLAIN



+------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
| -> Nested loop inner join  (cost=444 rows=1006) (actual time=0.0653..1.69 rows=1001 loops=1)
    -> Nested loop inner join  (cost=90.2 rows=200) (actual time=0.043..0.307 rows=200 loops=1)
        -> Filter: (a.artist_id is not null)  (cost=20.2 rows=200) (actual time=0.034..0.0877 rows=200 loops=1)
            -> Table scan on a  (cost=20.2 rows=200) (actual time=0.0334..0.0744 rows=200 loops=1)
        -> Single-row index lookup on ar using PRIMARY (artist_id=a.artist_id)  (cost=0.251 rows=1) (actual time=930e-6..956e-6 rows=1 loops=200)
    -> Filter: (s.play_count > 1000)  (cost=1.27 rows=5.03) (actual time=0.00521..0.00655 rows=5 loops=200)
```

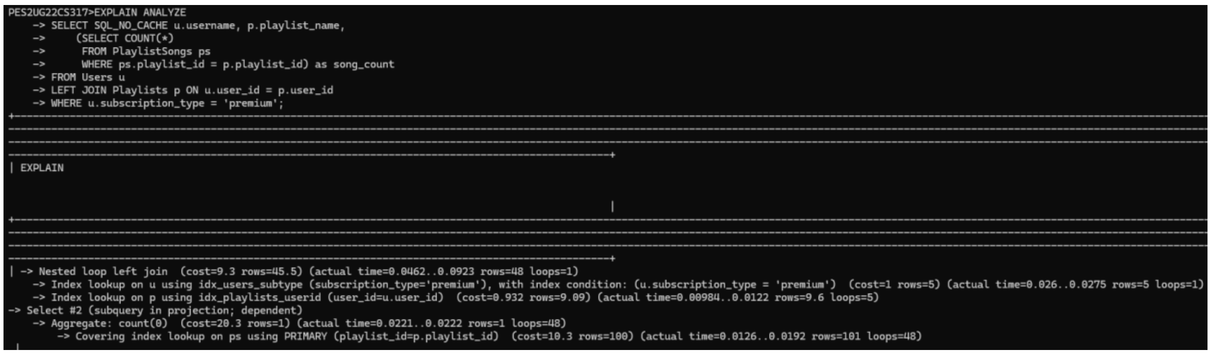
Compare execution plans for query 2:

Before optimization of query 2



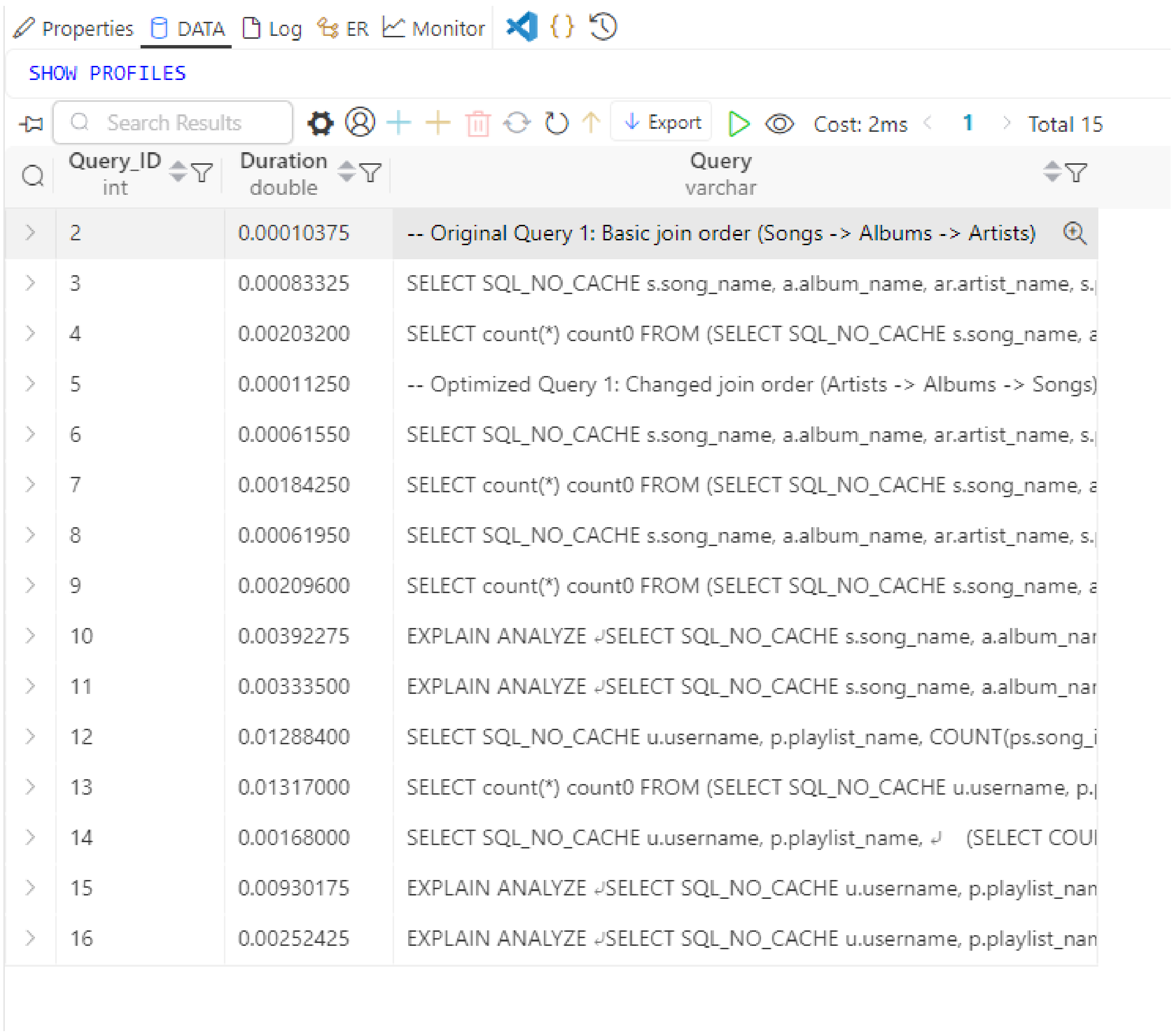
```
PES2UG22CS317>EXPLAIN ANALYZE
    -> SELECT SQL_NO_CACHE u.username, p.playlist_name, COUNT(ps.song_id) as song_count
    -> FROM Users u
    -> JOIN Playlists p ON u.user_id = p.user_id
    -> JOIN PlaylistSongs ps ON p.playlist_id = ps.playlist_id
    -> GROUP BY u.username, p.playlist_name;
+------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
| EXPLAIN



+------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------
| -> Table scan on <temporary>  (actual time=8.01..8.03 rows=100 loops=1)
    -> Aggregate using temporary table  (actual time=8.01..8.01 rows=100 loops=1)
        -> Nested loop inner join  (cost=1049 rows=10000) (actual time=0.0754..2.96 rows=10000 loops=1)
            -> Nested loop inner join  (cost=19.6 rows=100) (actual time=0.0517..0.174 rows=100 loops=1)
                -> Covering index scan on u using username  (cost=1.35 rows=11) (actual time=0.0226..0.0272 rows=11 loops=1)
                -> Index lookup on p using idx_playlists_userid (user_id=u.user_id)  (cost=0.833 rows=9.09) (actual time=0.0103..0.0127 rows=9.09 loops=11)
            -> Covering index lookup on ps using PRIMARY (playlist_id=p.playlist_id)  (cost=0.398 rows=100) (actual time=0.0139..0.0227 rows=100 loops=100)
 |
```

After optimization of query 2



```
PES2UG22CS317>EXPLAIN ANALYZE
    -> SELECT SQL_NO_CACHE u.username, p.playlist_name,
    ->     (SELECT COUNT(*)
    ->      FROM PlaylistSongs ps
    ->      WHERE ps.playlist_id = p.playlist_id) as song_count
    -> FROM Users u
    -> LEFT JOIN Playlists p ON u.user_id = p.user_id
    -> WHERE u.subscription_type = 'premium';
+----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------+
| EXPLAIN

                                                                                       |
+----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------+
| -> Nested loop left join  (cost=9.3 rows=45.5) (actual time=0.0462..0.0923 rows=48 loops=1)
    -> Index lookup on u using idx_users_subtype (subscription_type='premium'), with index condition: (u.subscription_type = 'premium')  (cost=1 rows=5) (actual time=0.026..0.0275 rows=5 loops=1)
    -> Index lookup on p using idx_playlists_userid (user_id=u.user_id)  (cost=0.932 rows=9.09) (actual time=0.00984..0.0122 rows=9.6 loops=5)
-> Select #2 (subquery in projection; dependent)
    -> Aggregate: count(0)  (cost=20.3 rows=1) (actual time=0.0221..0.0222 rows=1 loops=48)
        -> Covering index lookup on ps using PRIMARY (playlist_id=p.playlist_id)  (cost=10.3 rows=100) (actual time=0.0126..0.0192 rows=101 loops=48)
```

Show profiles with execution time:



```
SHOW PROFILES
```

Query_ID int	Duration double	Query varchar
2	0.00010375	-- Original Query 1: Basic join order (Songs -> Albums -> Artists)
3	0.00083325	SELECT SQL_NO_CACHE s.song_name, a.album_name, ar.artist_name, s.
4	0.00203200	SELECT count(*) count0 FROM (SELECT SQL_NO_CACHE s.song_name, a
5	0.00011250	-- Optimized Query 1: Changed join order (Artists -> Albums -> Songs)
6	0.00061550	SELECT SQL_NO_CACHE s.song_name, a.album_name, ar.artist_name, s.
7	0.00184250	SELECT count(*) count0 FROM (SELECT SQL_NO_CACHE s.song_name, a
8	0.00061950	SELECT SQL_NO_CACHE s.song_name, a.album_name, ar.artist_name, s.
9	0.00209600	SELECT count(*) count0 FROM (SELECT SQL_NO_CACHE s.song_name, a
10	0.00392275	EXPLAIN ANALYZE ↵SELECT SQL_NO_CACHE s.song_name, a.album_nar
11	0.00333500	EXPLAIN ANALYZE ↵SELECT SQL_NO_CACHE s.song_name, a.album_nar
12	0.01288400	SELECT SQL_NO_CACHE u.username, p.playlist_name, COUNT(ps.song_i
13	0.01317000	SELECT count(*) count0 FROM (SELECT SQL_NO_CACHE u.username, p.
14	0.00168000	SELECT SQL_NO_CACHE u.username, p.playlist_name, ↵ (SELECT COUI
15	0.00930175	EXPLAIN ANALYZE ↵SELECT SQL_NO_CACHE u.username, p.playlist_nan
16	0.00252425	EXPLAIN ANALYZE ↵SELECT SQL_NO_CACHE u.username, p.playlist_nan

(e) Query Analysis and Optimization:

Analyze and optimize a complex query within the mini-world database created in part (a).

Part 1: Query Analysis

1 Write a parse tree for a complex query, such as a 3-table join, by hand.

2 Formulate a relational algebra expression for the same query.

3 Create an initial query tree based on the relational algebra expression.

Part 2: Query Optimization

1 Optimize the initial query tree to enhance query performance.

2 Document the optimization steps taken to refine the query tree.

Complex Query Analysis and Optimization

Selected Query

```
SELECT s.song_name, a.album_name, ar.artist_name
FROM Songs s
JOIN Albums a ON s.album_id = a.album_id
JOIN Artists ar ON a.artist_id = ar.artist_id
WHERE s.play_count > 1000 AND ar.verified = TRUE;
```

Part 1: Query Analysis

Parse Tree

```
SELECT
├── PROJECTION (song_name, album_name, artist_name)
├── JOIN
│   ├── JOIN
│   │   ├── Songs (s)
│   │   ├── ON (s.album_id = a.album_id)
│   │   └── Albums (a)
│   ├── ON (a.artist_id = ar.artist_id)
│   └── Artists (ar)
└── WHERE
    └── AND
        ├── s.play_count > 1000
        └── ar.verified = TRUE
```

Relational Algebra Expression

```
π song_name, album_name, artist_name (
    σ play_count > 1000 ∧ verified = TRUE (
        Songs ⋈ album_id Albums ⋈ artist_id Artists
    )
)
```

Initial Query Tree

```
       PROJECT (song_name, album_name, artist_name)
                    |
                  JOIN
                /      \
               /        \
            JOIN      Artists
          /     \         |
       Songs  Albums   σ verified=TRUE
         |
σ play_count>1000
```

Part 2: Query Optimization

Optimization Steps

1. **Push Selection Operations Down**

```
       PROJECT (song_name, album_name, artist_name)
                    |
                  JOIN
                /      \
               /        \
            JOIN     σ verified=TRUE
          /     \         |
σ play_count>1000  Albums  Artists
         |
       Songs
```

2. **Reorder Joins Based on Table Sizes**
   - Artists (smallest) → Albums → Songs (largest)

```
       PROJECT (song_name, album_name, artist_name)
                    |
                  JOIN
                /      \
               /        \
            JOIN     σ play_count>1000
          /     \         |
σ verified=TRUE  Albums   Songs
         |
      Artists
```

3. **Add Index Support**
   - Create index on Songs(play_count)
   - Use existing indexes on album_id and artist_id

```sql
CREATE INDEX idx_songs_playcount ON Songs(play_count);
```

Final Optimized Query

```sql
SELECT s.song_name, a.album_name, ar.artist_name
FROM Artists ar
JOIN Albums a ON ar.artist_id = a.artist_id
JOIN Songs s ON a.album_id = s.album_id
WHERE ar.verified = TRUE
AND s.play_count > 1000;
```

Performance Impact

1. **Selection Push-Down**
   - Reduces intermediate result sizes
   - Filters data earlier in execution

2. **Join Reordering**
   - Starts with smallest table (Artists)
   - Minimizes intermediate result sizes
   - Reduces memory usage

3. **Index Usage**
   - Enables index scan instead of table scan
   - Improves join performance
   - Speeds up WHERE clause evaluation

Execution Plan Comparison

Before Optimization

```
-> Nested loop join
   -> Table scan on Songs
   -> Index lookup on Albums
   -> Index lookup on Artists
```

After Optimization

```
-> Nested loop join
   -> Index scan on Artists (verified=TRUE)
   -> Index lookup on Albums using artist_id
   -> Index lookup on Songs using album_id and play_count
```