Code to create DB :

|  |
| --- |
| -- Create the URL table  CREATE TABLE URL (  Track VARCHAR(255) PRIMARY KEY,  Spotify\_url VARCHAR(255),  Youtube\_url VARCHAR(255)  );  -- Create the Statistic table  CREATE TABLE Statistic (  Track VARCHAR(255) PRIMARY KEY,  Danceability FLOAT,  Energy FLOAT,  Key INT,  Loudness FLOAT,  Speechiness FLOAT,  Acousticness FLOAT,  Instrumentalness FLOAT,  Liveness FLOAT,  Valence FLOAT,  Tempo FLOAT,  Duration\_ms INT  );  -- Create the Song table  CREATE TABLE Song (  Track VARCHAR(255) PRIMARY KEY,  Artist VARCHAR(255),  Title VARCHAR(255),  Likes INT,  Comments INT,  Description TEXT,  Licensed BOOLEAN,  official\_video BOOLEAN,  Stream VARCHAR(255),  FOREIGN KEY (Track) REFERENCES URL(Track)  );  -- Create the User table  CREATE TABLE Users (  Username VARCHAR(255) PRIMARY KEY,  Email VARCHAR(255) UNIQUE,  Password VARCHAR(255),  Phonenumber VARCHAR(255),  Address TEXT,  Last\_10\_songs\_played TEXT[] -- This can be an array of track IDs  );  -- Create the Playlist table  CREATE TABLE Playlist (  Username VARCHAR(255),  Playlist\_Number INT,  Song\_name VARCHAR(255),  FOREIGN KEY (Username) REFERENCES Users(Username),  PRIMARY KEY (Username, Playlist\_Number) -- Composite primary key for Username and Playlist\_Number  ); |

Inserting some data :

|  |
| --- |
| Temporary Table :  CREATE TEMPORARY TABLE temp\_music\_data (  "Sr No" SERIAL PRIMARY KEY,  Artist VARCHAR(255),  Url\_spotify VARCHAR(255),  Track VARCHAR(255),  Album VARCHAR(255),  Album\_type VARCHAR(255),  Uri VARCHAR(255),  Danceability FLOAT,  Energy FLOAT,  Key INT,  Loudness FLOAT,  Speechiness FLOAT,  Acousticness FLOAT,  Instrumentalness FLOAT,  Liveness FLOAT,  Valence FLOAT,  Tempo FLOAT,  Duration\_ms INT,  Url\_youtube VARCHAR(255),  Title VARCHAR(255),  Channel VARCHAR(255),  Views INT,  Likes INT,  Comments INT,  Description TEXT,  Licensed BOOLEAN,  official\_video BOOLEAN,  Stream VARCHAR(255)  ); |

|  |
| --- |
| Code to upload data in DB using python :  import psycopg2  from psycopg2 import sql  import csv  # Database connection parameters  dbname = "Metamusic"  user = "postgres"  password = "root"  host = "localhost"  port = "5432"  # CSV file path  csv\_file\_path = r"C:\Users\ajink\Downloads\Spotify\_Youtube.csv"  # SQL query to create the temporary table  table\_name = "temp\_music\_data"  try:  # Connect to the database  conn = psycopg2.connect(  dbname=dbname,  user=user,  password=password,  host=host,  port=port  )  # Create a cursor  cur = conn.cursor()  # Open the CSV file and read its contents  with open(csv\_file\_path, 'r', newline='', encoding='utf-8') as csv\_file:  csv\_reader = csv.DictReader(csv\_file)  for row in csv\_reader:  # Skip the row if any value in the row is empty  if not all(row.values()):  continue  # Insert each non-empty row into the database table  columns = ', '.join(row.keys())  values = ', '.join([f"%({column})s" for column in row.keys()])  insert\_query = f"INSERT INTO {table\_name} ({columns}) VALUES ({values})"  cur.execute(insert\_query, row)  # Commit the transaction  conn.commit()  print("Data loaded successfully!")  except psycopg2.Error as e:  print("Error:", e)  finally:  # Close the cursor and connection  if cur:  cur.close()  if conn:  conn.close() |

|  |
| --- |
| Actually Inserting data :  DELETE FROM Song;  DELETE FROM Statistic;  DELETE FROM URL;  INSERT INTO URL (Track, Spotify\_url, Youtube\_url)  SELECT Track, Url\_spotify, Url\_youtube  FROM temp\_music\_data  ON CONFLICT (Track) DO NOTHING;  INSERT INTO Statistic (Track, Danceability, Energy, Key, Loudness, Speechiness, Acousticness, Instrumentalness, Liveness, Valence, Tempo, Duration\_ms)  SELECT Track, Danceability, Energy, Key, Loudness, Speechiness, Acousticness, Instrumentalness, Liveness, Valence, Tempo, Duration\_ms  FROM temp\_music\_data  ON CONFLICT (Track) DO NOTHING;  INSERT INTO Song (Track, Artist, Title, Likes, Comments, Description, Licensed, official\_video, Stream)  SELECT Track, Artist, Title, Likes, Comments, Description, Licensed, official\_video, Stream  FROM temp\_music\_data  ON CONFLICT (Track) DO NOTHING; |
| ---------Voila ------our DB is setup and ready to launch |

Some Query Used to create the dashboard or demo website for meta music :

|  |
| --- |
| SELECT Track, Title, Likes  FROM Song  ORDER BY Likes DESC  LIMIT 10; |

|  |
| --- |
| SELECT Track, Title, Stream  FROM Song  ORDER BY Stream DESC  LIMIT 10; |
| SELECT Artist,  SUM(Likes) AS Total\_Likes,  SUM(CAST(Stream AS INTEGER)) AS Total\_Streams,  SUM(Likes) + SUM(CAST(Stream AS INTEGER)) AS Combined\_Score  FROM Song  GROUP BY Artist  ORDER BY Combined\_Score DESC  LIMIT 10; |
| SELECT Track, Title, Comments  FROM Song  ORDER BY Comments DESC  LIMIT 10; |