ENTREGABLE 1 - EXTRAER DATOS DE TOP 10 CANCIONES POR POPULARIDAD DE API DE SPOTIFY E INSERTAR LA DATA EN UNA TABLA DE AWS REDSHIFT.

In [1]:
```
!pip install spotipy
!pip install wheel
!pip install pandas
!pip install psycopg2
```

```
Requirement already satisfied: spotipy in c:\users\cnieto1\anaconda3\envs\dhdsblend20
21\lib\site-packages (2.23.0)
Requirement already satisfied: redis>=3.5.3 in c:\users\cnieto1\anaconda3\envs\dhdsbl
end2021\lib\site-packages (from spotipy) (5.0.0)
Requirement already satisfied: six>=1.15.0 in c:\users\cnieto1\anaconda3\envs\dhdsble
nd2021\lib\site-packages (from spotipy) (1.16.0)
Requirement already satisfied: urllib3>=1.26.0 in c:\users\cnieto1\anaconda3\envs\dhd
sblend2021\lib\site-packages (from spotipy) (1.26.9)
Requirement already satisfied: requests>=2.25.0 in c:\users\cnieto1\anaconda3\envs\dh
dsblend2021\lib\site-packages (from spotipy) (2.27.1)
Requirement already satisfied: async-timeout>=4.0.2 in c:\users\cnieto1\anaconda3\env
s\dhdsblend2021\lib\site-packages (from redis>=3.5.3->spotipy) (4.0.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\cnieto1\anaconda3\envs\dhdsbl
end2021\lib\site-packages (from requests>=2.25.0->spotipy) (3.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\cnieto1\anaconda3\envs
\dhdsblend2021\lib\site-packages (from requests>=2.25.0->spotipy) (2022.6.15)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\cnieto1\anaconda
3\envs\dhdsblend2021\lib\site-packages (from requests>=2.25.0->spotipy) (2.0.4)
Requirement already satisfied: wheel in c:\users\cnieto1\anaconda3\envs\dhdsblend2021
\lib\site-packages (0.37.1)
Requirement already satisfied: pandas in c:\users\cnieto1\anaconda3\envs\dhdsblend202
1\lib\site-packages (1.1.5)
Requirement already satisfied: pytz>=2017.2 in c:\users\cnieto1\anaconda3\envs\dhdsbl
end2021\lib\site-packages (from pandas) (2021.3)
Requirement already satisfied: numpy>=1.15.4 in c:\users\cnieto1\anaconda3\envs\dhdsb
lend2021\lib\site-packages (from pandas) (1.19.2)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\cnieto1\anaconda3\e
nvs\dhdsblend2021\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\cnieto1\anaconda3\envs\dhdsblend2
021\lib\site-packages (from python-dateutil>=2.7.3->pandas) (1.16.0)
Requirement already satisfied: psycopg2 in c:\users\cnieto1\anaconda3\envs\dhdsblend2
021\lib\site-packages (2.9.7)
```

Pasos para obtener mis credenciales de la API de Spotify, y el uso de la API:

Para obtener credenciales y usar la API de Spotify, necesitas registrarte como desarrollador en la plataforma de Spotify y crear una aplicación. Aquí están los pasos generales que debes seguir:

a) Crea una cuenta de Spotify: Si no tienes una cuenta de Spotify, crea una en el sitio web de Spotify.

b) Inicia sesión en el Panel de Desarrolladores de Spotify: Visita el Panel de Desarrolladores de Spotify en https://developer.spotify.com/dashboard/login.

c) Crea una nueva aplicación: Una vez que hayas iniciado sesión, puedes crear una nueva aplicación. Proporciona un nombre y una descripción para tu aplicación.

d) Configura los ajustes de la aplicación:

Especifica la descripción y los detalles de la aplicación. Indica si tu aplicación es comercial o personal. Proporciona la dirección de correo electrónico de contacto. Selecciona las capacidades que tu aplicación usará (por ejemplo, acceder a datos de usuario, controlar dispositivos, etc.).

e) Aceptar los términos y condiciones: Asegúrate de leer y aceptar los términos y condiciones de la plataforma de desarrolladores de Spotify.

f) Obtén las credenciales de API:

Después de crear la aplicación, se te proporcionarán las credenciales de API, que generalmente incluyen un ID de cliente (client ID) y una clave secreta (client secret). Estas credenciales serán necesarias para autenticarte y realizar solicitudes a la API de Spotify. g) Configura redireccionamientos de URI: En la configuración de la aplicación, es posible que necesites especificar los redireccionamientos de URI permitidos, que se utilizan en el flujo de autenticación. Esto depende del tipo de autenticación que utilices.

h) Utiliza las credenciales en tu aplicación: Ahora puedes usar las credenciales de API en tu aplicación para autenticarte y realizar solicitudes a la API de Spotify.

Es importante tener en cuenta que la API de Spotify puede requerir diferentes tipos de autenticación según tus necesidades. Puedes encontrar más información sobre cómo autenticarte y cómo realizar solicitudes específicas en la documentación oficial de la API de Spotify.

Recuerda también que es fundamental cumplir con las políticas de uso y términos de servicio de Spotify al desarrollar aplicaciones que interactúen con su plataforma.

Explicación del siguiente código de Python:

El fragmento results['tracks']['total'] se utiliza para extraer el valor de la clave 'total' dentro del diccionario 'tracks' en el diccionario results. En este contexto:

results: Es el objeto que contiene la respuesta de la búsqueda a la API de Spotify. 'tracks': Es una clave dentro del objeto results que apunta a un diccionario que contiene información relacionada con las canciones. 'total': Es una clave dentro del diccionario 'tracks' que contiene el número total de canciones encontradas en la búsqueda. Entonces, results['tracks']['total'] está accediendo al valor de 'total' que representa la cantidad total de canciones encontradas en la búsqueda.

Este valor es importante para determinar cuántas iteraciones del bucle while se deben realizar para recopilar todas las canciones que cumplen el criterio de búsqueda. En cada iteración, el valor de offset se incrementa en el número de resultados por página (generalmente 50), y la búsqueda se realiza nuevamente. Si offset sigue siendo menor que total, hay más resultados disponibles y se necesita otra iteración.

Explicación del fragmento "for idx, track in enumerate(results['tracks']['items']):" del script.

Esta línea de código está creando un bucle for que itera a través de la lista de canciones ('items') en los resultados de la búsqueda de la API de Spotify. Cada elemento en esta lista representa una canción individual que cumple con el criterio de búsqueda.

Voy a desglosar la línea para que sea más comprensible:

enumerate(results['tracks']['items']): enumerate() es una función incorporada en Python que devuelve un objeto iterable que produce pares de valores (índice, elemento). En este caso, results['tracks']['items'] es la lista de canciones obtenida de la respuesta de la API.

for idx, track in enumerate(...): Esta línea inicia un bucle for que itera sobre cada elemento en la lista de canciones. idx representa el índice del elemento en la lista y track representa los datos de la canción actual.

En resumen, el bucle for está siendo utilizado para recorrer cada canción en la lista de canciones obtenida de los resultados de la búsqueda. Durante cada iteración del bucle, el valor de idx representa el índice de la canción actual en la lista, y el valor de track contiene los datos de la canción (como su nombre, álbum, artistas, etc.) que se están procesando en esa iteración.

In [2]:
```
#RECORDAR: La variable "results" en la API tiene la siguiente forma:

#results = {
#    'tracks': {
#        'items': [
#            # Lista de pistas (cada pista es un diccionario)
#            {
#                'id': '...',
#                'name': '...',
#                'artists': [...],
#                'album': {...},
#                'duration_ms': ...,
#                'popularity': ...
#                # Otros campos específicos de la pista
#            },
#            # Más pistas...
#        ]
#    }
#}
```

IMPORTANTE: Aunque no es garantía, la API de Spotify suele devolver los resultados de manera que las canciones más populares o relevantes (basadas en su algoritmo) se encuentren en los

primeros lugares. Esto puede explicar por qué los resultados que obtengo pueden parecer estar en algún tipo de orden.

En resumen, aunque no obtengo resultados completamente aleatorios, los resultados de la API de Spotify se basarán en criterios de relevancia y popularidad, y la API decidirá el orden en que se devuelven los resultados en función de esos criterios.

Por este motivo, para asegurarme de obtener las Top 10 canciones por día, lo que voy a hacer es, una vez extraídos los 1000 registros que me devuelve la API de Spotify por día, voy a ordenar el dataset resultante "df" por "Popularidad" desc, y de ese dataset me voy a quedar con los 10 primeros registros (obteniendo el dataset "top_10_songs" que se verá más adelante):

In [3]:
```
# Guardo mi contraseña "client_secret" de la API de Spotify en un archivo .txt por cue
with open("C:/Users/cnieto1/Desktop/Curso Data Engineering - Coderhouse/Clases/Entrega
    pwd= f.read()
```

In [4]:
```
#Chequeo si la variable pwd tomó bien el archivo txt:
import os

file_path = "C:/Users/cnieto1/Desktop/Curso Data Engineering - Coderhouse/Clases/Entre
if os.path.exists(file_path):
    with open(file_path, 'r') as f:
        pwd = f.read()
else:
    print("El archivo no existe en la ruta especificada.")
```

In [5]:
```
#Ingreso mis credenciales de la API de Spotify:
import pandas as pd
import spotipy
from spotipy.oauth2 import SpotifyClientCredentials
import datetime
import time

client_id = 'dbe61651ee31461681339d9d1780f672'
client_secret = pwd

sp = spotipy.Spotify(auth_manager=SpotifyClientCredentials(client_id, client_secret))

#Consulto los datos de la API de Spotify:

d = []
#current_date = datetime.datetime.now().strftime('%Y-%m-%d')
total = 1
offset = 0

while offset < total:
    results = sp.search(q="year:2023", type='track', offset=offset, limit=50)
    total = results['tracks']['total']                          # Actualiza a la variabl
    offset += 50                                                # Aumenta el offset en 5
    for idx, track in enumerate(results['tracks']['items']):
        artist_id = track['artists'][0]['id']
        track_genre = sp.artist(artist_id)['genres']  # Obtener los géneros del artist
```

```
        track_genre = ', '.join(track_genre)  # Convertir la lista de géneros a una co
        d.append(
            {
                'id': track['id'],
                'Track': track['name'],
                'Album': track['album']['name'],
                'Artist': track['artists'][0]['name'],
                'Release Date': track['album']['release_date'],
                'Track Number': track['track_number'],
                'Popularity': track['popularity'],
                'album_cont': track['album']['total_tracks'],
                'Duration': track['duration_ms'],
                'Audio Preview URL': track['preview_url'],
                'Album URL': track['album']['external_urls']['spotify'],
                #'Execution Date': current_date,
                'Genre': track_genre  # Agrega el género al diccionario
            }
        )
        #time.sleep(0.5)  # Espera 0.5 segundos para respetar los límites de la API

df = pd.DataFrame(d)
```

In [6]:

```
# ACTIVAR ESTE SCRIPT EN CASO DE QUE SE PREFIERA USAR LA ALTERNATIVA DE TRABAJAR CON 5
# Script de Alternativa para extraer 50 registros de canciones de la API de Spotify (p
# Preferí no usar este Script, ya que con el Script de arriba obtengo 1000 registros e

#Consultando los datos:
# import pandas as pd
# results = sp.search(q='year:2023', type='track', limit=50) # utilizo type='Track' pa
# data = {'Id': [],'Artista': [], 'Cancion': [],'Duracion_ms': [], 'Genero': [],'Album
# for track in results['tracks']['items']:
#     id = track['id']
#     artist_name = track['artists'][0]['name'] # Si una pista tiene varios artistas,
#     artist_id = track['artists'][0]['id']
#     track_name = track['name']
#     duration_ms = track['duration_ms']
#     track_id = track['id']
#     album_group = track['album']['name']
#     album_img = track['album']['images'][0]['url'] #imagen de album
#     album_cont = track['album']['total_tracks']
#     track_genre = sp.artist(artist_id)['genres']
#     track_popularity = track['popularity']
#     track_year = track['album']['release_date']
#     #Quitar las comillas
#     track_name = track_name.replace("'", "")
#     album_group = album_group.replace("'", "")
#     #Separar el género por coma
#     track_genre = ', '.join(track_genre)

#     data['Id'].append(id)
#     data['Artista'].append(artist_name)
#     data['Cancion'].append(track_name)
#     data['Duracion_ms'].append(duration_ms)
#     data['Album'].append(album_group)
#     data['Album_img'].append(album_img)
#     data['Total_canciones_album'].append(album_cont)
#     data['Genero'].append(track_genre)
#     data['Popularidad'].append(track_popularity)
#     data['fecha_lanzamiento'].append(track_year)
```

```
# df = pd.DataFrame(data)
# #Evitar que haya canciones duplicadas
# df.drop_duplicates(subset=['Artista', 'Cancion','Album'], keep='first', inplace=True
# #Reemplazar valores nulos o vacios en el campo Género por Desconocido
# df['Genero'].fillna('Desconocido', inplace=True)
# df.loc[df['Genero'] == '', 'Genero'] = 'Desconocido'
# #Evitar que se cargue una canción con duración 0 ms
# df = df[df['Duracion_ms'] != 0]
# #Verificar que la fecha se muestre en formato fecha
# df['fecha_lanzamiento'] = pd.to_datetime(df['fecha_lanzamiento'], format='%Y-%m-%d')
# display(df)
```

In [7]:
```
df.head()
```

Out[7]:

	id	Track	Album	Artist	Release Date	Track Number	Popularity	album
0	6AQbmUe0Qwf5PZnt4HmTXv	Boy's a Liar Pt. 2	Boy's a liar Pt. 2	PinkPantheress	2023-02-03	1	91	
1	3dnP0JxCgygwQH9Gm7q7nb	Ella Baila Sola	Ella Baila Sola	Eslabon Armado	2023-03-16	1	92	
2	7bPp2NmpmyhLJ7zWazAXMu	TULUM	GÉNESIS	Peso Pluma	2023-06-29	15	93	
3	7mXuWTczZNxG5EDcjFEuJR	LADY GAGA	GÉNESIS	Peso Pluma	2023-06-29	10	94	
4	1odExI7RdWc4BT515LTAwj	Daylight	Daylight	David Kushner	2023-04-14	1	96	

In [8]:
```
#Obtenemos un Dataframe de 1000 registros y 11 columnas:
df.shape
```

Out[8]:
```
(1000, 12)
```

In [9]:
```
#Hago las siguientes Transformaciones al DataFrame "df":

#Evitar que haya canciones duplicadas:
df.drop_duplicates(subset=['Artist', 'Track','Album'], keep='first', inplace=True) # e
#Reemplazar valores nulos o vacios en el campo Género por Desconocido:
df['Genre'].fillna('Desconocido', inplace=True)
df.loc[df['Genre'] == '', 'Genre'] = 'Desconocido'  # En resumen, el código busca todo
#Evitar que se cargue una canción con duración 0 ms:
df = df[df['Duration'] != 0]
```

```
#Verificar que las fechas se muestren en formato fecha:
df['Release Date'] = pd.to_datetime(df['Release Date'], format='%Y-%m-%d')

#display(df)
df.head()
```

Out[9]:

	id	Track	Album	Artist	Release Date	Track Number	Popularity	album
0	6AQbmUe0Qwf5PZnt4HmTXv	Boy's a Liar Pt. 2	Boy's a liar Pt. 2	PinkPantheress	2023-02-03	1	91	
1	3dnP0JxCgygwQH9Gm7q7nb	Ella Baila Sola	Ella Baila Sola	Eslabon Armado	2023-03-16	1	92	
2	7bPp2NmpmyhLJ7zWazAXMu	TULUM	GÉNESIS	Peso Pluma	2023-06-29	15	93	
3	7mXuWTczZNxG5EDcjFEuJR	LADY GAGA	GÉNESIS	Peso Pluma	2023-06-29	10	94	
4	1odExI7RdWc4BT515LTAwj	Daylight	Daylight	David Kushner	2023-04-14	1	96	

In [10]:

```
# Me filtro a mi DataFrame "df" por las top 10 canciones con mayor popularidad de ese

# Ordeno el DataFrame por popularidad de manera descendente:
df_sorted = df.sort_values(by='Popularity', ascending=False)

# Selecciono las primeras 10 filas (las canciones con mayor popularidad):
top_10_songs = df_sorted.head(10)

top_10_songs.head()
```

Out[10]:

	id	Track	Album	Artist	Release Date	Track Number	Popularity	album_c
41	7x9aauaA9cu6tyfpHnqDLo	Seven (feat. Latto) (Explicit Ver.)	Seven (feat. Latto)	Jung Kook	2023-07-14	2	100	
38	7ABLbnD53cQK00mhcaOUVG	LALA	LA VIDA ES UNA	Myke Towers	2023-03-23	22	99	
17	6wf7Yu7cxBSPrRlWeSeK0Q	What Was I Made For? [From The Motion Picture ...	What Was I Made For? [From The Motion Picture ...	Billie Eilish	2023-07-13	1	98	
80	6XbtvPmIpyCbjuT0e8cQtp	Columbia	Columbia	Quevedo	2023-07-07	1	97	
6	1vYXt7VSjH9JIM5oRRo7vA	Dance The Night (From Barbie The Album)	Dance The Night (From Barbie The Album)	Dua Lipa	2023-05-25	1	97	

In [11]:
```
#Obtenemos un Dataframe de 10 registros y 12 columnas:
top_10_songs.shape
```

Out[11]:
```
(10, 12)
```

In [12]:
```
# Construyo mi Dataframe con las columnas que considero más importantes y con el order
#data = {'Id': [],'Artista': [], 'Cancion': [],'Duracion_ms': [],'Album': [], 'Album_i
data = pd.DataFrame()

from datetime import datetime
current_date = datetime.now().strftime('%Y-%m-%d')

data['Id']= top_10_songs['id']
data['Artista']= top_10_songs['Artist']
data['Cancion']= top_10_songs['Track']
data['Duracion_ms']= top_10_songs['Duration']
data['Album']= top_10_songs['Album']
data['Album_img']= top_10_songs['Album URL']
data['Total_canciones_album']= top_10_songs['album_cont']
data['Audio_preview']= top_10_songs['Audio Preview URL']
data['Popularidad']= top_10_songs['Popularity']
```

```
data['Genero']= top_10_songs['Genre']
data['fecha_lanzamiento']= top_10_songs['Release Date']
data['Insert_date'] = current_date

data.head()
```

Out[12]:

	Id	Artista	Cancion	Duracion_ms	Album	
41	7x9aauaA9cu6tyfpHnqDLo	Jung Kook	Seven (feat. Latto) (Explicit Ver.)	184400	Seven (feat. Latto)	https://open.spotify.com/alb
38	7ABLbnD53cQK00mhcaOUVG	Myke Towers	LALA	197920	LA VIDA ES UNA	https://open.spotify.com/
17	6wf7Yu7cxBSPrRlWeSeK0Q	Billie Eilish	What Was I Made For? [From The Motion Picture ...	222369	What Was I Made For? [From The Motion Picture ...	https://open.spotify.com/
80	6XbtvPmIpyCbjuT0e8cQtp	Quevedo	Columbia	186000	Columbia	https://open.spotify.com/al
6	1vYXt7VSjH9JIM5oRRo7vA	Dua Lipa	Dance The Night (From Barbie The Album)	176579	Dance The Night (From Barbie The Album)	https://open.spotify.com/a

In [13]:
```
data.shape
```

Out[13]:
```
(10, 12)
```

In [14]:
```
display(data)
```

	Id	Artista	Cancion	Duracion_ms	Album	
41	7x9aauaA9cu6tyfpHnqDLo	Jung Kook	Seven (feat. Latto) (Explicit Ver.)	184400	Seven (feat. Latto)	https://open.spotify.com/a
38	7ABLbnD53cQK00mhcaOUVG	Myke Towers	LALA	197920	LA VIDA ES UNA	https://open.spotify.com
17	6wf7Yu7cxBSPrRlWeSeK0Q	Billie Eilish	What Was I Made For? [From The Motion Picture ...	222369	What Was I Made For? [From The Motion Picture ...	https://open.spotify.com
80	6XbtvPmIpyCbjuT0e8cQtp	Quevedo	Columbia	186000	Columbia	https://open.spotify.com/a
6	1vYXt7VSjH9JIM5oRRo7vA	Dua Lipa	Dance The Night (From Barbie The Album)	176579	Dance The Night (From Barbie The Album)	https://open.spotify.com/
12	3k79jB4aGmMDUQzEwa46Rz	Olivia Rodrigo	vampire	219724	vampire	https://open.spotify.com/a
66	2FDTHlrBguDzQkp7PVj16Q	Dave	Sprinter	229133	Sprinter	https://open.spotify.com/
22	2UW7JaomAMuX9pZrjVpHAU	Yng Lvcas	La Bebe - Remix	234352	La Bebe (Remix)	https://open.spotify.com/al
11	4rXLjWdF2ZZpXCVTfWcshS	Gunna	fukumean	125040	a Gift & a Curse	https://open.spotify.com/
27	7ro0hRteUMfnOioTFI5TG1	Bad Bunny	WHERE SHE GOES	231704	WHERE SHE GOES	https://open.spotify.com/a

In [15]:
```
# Guardo mi contraseña "pwd_redshift" de mi cuenta de Redshift en un archivo .txt por
with open("C:/Users/cnieto1/Desktop/Curso Data Engineering - Coderhouse/Clases/Entrega
    pwd_redshift= f.read()
```

In [16]:
```
# Creando la conexión a Redshift:
import psycopg2
```

```
url="data-engineer-cluster.cyhh5bfevlmn.us-east-1.redshift.amazonaws.com"
data_base="data-engineer-database"
user="christian_r_coderhouse"

try:
    conn = psycopg2.connect(
        host='data-engineer-cluster.cyhh5bfevlmn.us-east-1.redshift.amazonaws.com',
        dbname=data_base,
        user=user,
        password= pwd_redshift,
        port='5439'
    )
    print("Conectado a Redshift con éxito!")

except Exception as e:
    print("No es posible conectar a Redshift")
    print(e)
```

```
Conectado a Redshift con éxito!
```

In [17]:
```
# Código para hacer Drop Table de la tabla "canciones". USARLO SOLO en caso de que la
# RECORDAR: Antes de correr este código, correr primero el código anterior (Creando la

# Crear un cursor:
#cur = conn.cursor()

# Ejecutar la sentencia DROP TABLE:
#cur.execute("DROP TABLE IF EXISTS canciones")

# Hacer commit para aplicar los cambios:
#conn.commit()
```

In [18]:
```
#Crear la tabla si no existe:
with conn.cursor() as cur:
    cur.execute("""
        CREATE TABLE IF NOT EXISTS canciones
        (
        Id VARCHAR(50) primary key
        ,Artista VARCHAR(255)
        ,Cancion VARCHAR(255)
        ,Duracion_ms INTEGER
        ,Album VARCHAR(200)
        ,Album_img VARCHAR(300)
        ,Total_canciones_album INTEGER
        ,Audio_preview NVARCHAR(300)
        ,Popularidad INTEGER
        ,Genero VARCHAR(300)
        ,fecha_lanzamiento date
        ,Insert_date date

        )
    """)
    conn.commit()
```

In [19]:
```
# Comento este paso, para que cada vez que corra el script de la API, que me vaya inse
#Vaciar la tabla para evitar duplicados o inconsistencias:
#with conn.cursor() as cur:
#  cur.execute("Truncate table canciones")
```

```
#  count = cur.rowcount
# count
```

In [20]:
```
#consultando la tabla canciones:
cur = conn.cursor()
cur.execute("SELECT * FROM canciones")
results = cur.fetchall()
#results
```

In [21]:
```
#Insertando los datos en Redsfhift:
from psycopg2.extras import execute_values
with conn.cursor() as cur:
    execute_values(
        cur,
        '''
        INSERT INTO canciones (Id, Artista, Cancion, Duracion_ms, Album, Album_img, tc
        VALUES %s
        ''',
        [tuple(row) for row in data.values],
        page_size=len(data)
    )
    conn.commit()
```

In [22]:
```
# Veo cómo quedó la tabla en Redshift luego de hacer los Insert:
#consultando la tabla
cur = conn.cursor()
cur.execute("SELECT * FROM canciones")
results = cur.fetchall()
```

In [23]:
```
# Veo cómo quedó la tabla "canciones" en Redshift. Convierto "results" al DataFrame "d
column_names=['Id', 'Artista', 'Cancion', 'Duracion_ms', 'Album', 'Album_img', 'Total_
df_redshift = pd.DataFrame(results, columns=column_names)
df_redshift.head()
```

Out[23]:

	Id	Artista	Cancion	Duracion_ms	Album	
0	7x9aauaA9cu6tyfpHnqDLo	Jung Kook	Seven (feat. Latto) (Explicit Ver.)	184400	Seven (feat. Latto)	https://open.spotify.com/albu
1	7ABLbnD53cQK00mhcaOUVG	Myke Towers	LALA	197920	LA VIDA ES UNA	https://open.spotify.com/al
2	6wf7Yu7cxBSPrRlWeSeK0Q	Billie Eilish	What Was I Made For? [From The Motion Picture ...	222369	What Was I Made For? [From The Motion Picture ...	https://open.spotify.com/a
3	3k79jB4aGmMDUQzEwa46Rz	Olivia Rodrigo	vampire	219724	vampire	https://open.spotify.com/alb
4	6XbtvPmIpyCbjuT0e8cQtp	Quevedo	Columbia	186000	Columbia	https://open.spotify.com/alb

In [24]:
```
# Cierro tanto el cursor como la conexión a la base de datos:
cur.close()
conn.close()
```