```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mour
```

```
import requests
import base64
import pandas as pd
import concurrent.futures
import sqlite3
from time import sleep
```

## Charger les données

```
artist_data = pd.read_csv('/content/drive/MyDrive/Spotify/artist_data.csv')
spotify_data = pd.read_csv('/content/drive/MyDrive/Spotify/spotify_data.csv')
tag_artist_data = pd.read_csv('/content/drive/MyDrive/Spotify/tag_artist_data.csv')
tag_genre_data = pd.read_csv('/content/drive/MyDrive/Spotify/tag_genre_data.csv')

print("Initial Data:")
print("Artist data:", len(artist_data))
print("Spotify data:", len(spotify_data))
print("Tag artist data:", len(tag_artist_data))
print("Tag genre data:", len(tag_genre_data))
```

```
Initial Data:
Artist data: 10000
Spotify data: 10000
Tag artist data: 27301
Tag genre data: 11
```

## Nettoyer et fusionner les données

```
artist_data = artist_data.dropna().drop_duplicates()
spotify_data = spotify_data.dropna().drop_duplicates()
tag_artist_data = tag_artist_data.dropna().drop_duplicates()
tag_genre_data = tag_genre_data.dropna().drop_duplicates()

print("After Cleaning:")
print("Artist data:", len(artist_data))
print("Spotify data:", len(spotify_data))
print("Tag artist data:", len(tag_artist_data))
print("Tag genre data:", len(tag_genre_data))
```

```
After Cleaning:
Artist data: 9998
Spotify data: 10000
Tag artist data: 27301
Tag genre data: 11
```

```
merged_data = artist_data.merge(spotify_data, on='user_id', how='inner')
merged_data = merged_data.merge(tag_artist_data, on='user_id', how='inner')
merged_data = merged_data.merge(tag_genre_data, on='tag_id', how='inner')

print("After Merging:")
print("Merged data:", len(merged_data))
```

```
After Merging:
Merged data: 27297
```

```
print(merged_data)
```

```
       user_id     artist_name              spotify_id  tag_id   genre
0        24824     tatum quinn  3s0DwmaExsRr8KGfE8RkhH     601  reggae
1       303724  jason kerrison  7iLGqGUSoPQtj8OH61HFwZ     601  reggae
2       451943   milo x kahefa  70o5L3YGkE7lxVWPro3fkV     601  reggae
3       169814  gerson marques  7vdWVGhRjWiT8VHuS9D9a5     601  reggae
4       373197         engeezo  3EhLaFxXQijaAcWcoEGWJC     601  reggae
...        ...             ...                     ...     ...     ...
27292    18644            rain  5TRVLPPCKjLrdOK6z9gzBO     884   disco
27293    18584           abran  40u6S69aTCQsz28xU6EdcS     884   disco
27294    18678  jon mark doyle  00I5hPsAQZOVvYwz76C15G     884   disco
27295    18743   saint bernard  3eLjjCHq4mf9RExxvNzzQB     884   disco
27296    18618        katcross  5wKycvCDB0qE5vvw8CE13M     884   disco

[27297 rows x 5 columns]
```

## Identifiants API Spotify

```
client_id = '97cffbac5df344e68a4046a957d6cc99'
client_secret = '6d06d829812c405a8e69c48f1aa696d4'

credentials = f"{client_id}:{client_secret}"
encoded_credentials = base64.b64encode(credentials.encode())

auth_url = 'https://accounts.spotify.com/api/token'
auth_headers = {
    'Authorization': f'Basic {encoded_credentials.decode()}'
}
auth_data = {
    'grant_type': 'client_credentials'
}

auth_response = requests.post(auth_url, headers=auth_headers, data=auth_data)
access_token = auth_response.json().get('access_token')
print("Access Token:", access_token)
```

```
Access Token: BQDV7K3-NKfRLDZu0yXbnHHAMl0xivno4WHJyLQ7CLJu5h2ZcjNZ8tAcs7OxQZro_PuW5vku_E
```

Intégration des scores de popularité des artistes Spotify

Le code récupère les scores de popularité des artistes à partir de l'API Spotify et les intègre dans un ensemble de données fusionné. Il gère les limites de taux de l'API avec une logique de réessai et refusionne les données pour garantir l'exactitude des scores de popularité dans le résultat final.

```
import time
import requests

def get_artist_popularity(spotify_id):
    url = f'https://api.spotify.com/v1/artists/{spotify_id}'
    headers = {
        'Authorization': f'Bearer {access_token}'
    }
    response = requests.get(url, headers=headers)
    if response.status_code == 200:
        return response.json().get('popularity')
    else:
        return None

def fetch_popularity_scores(spotify_ids):
    with concurrent.futures.ThreadPoolExecutor(max_workers=10) as executor:
        popularity_scores = list(executor.map(get_artist_popularity, spotify_ids)
    return popularity_scores

spotify_data['popularity'] = fetch_popularity_scores(spotify_data['spotify_id'])
```

```
merged_data = artist_data.merge(spotify_data, on='user_id', how='inner')
merged_data = merged_data.merge(tag_artist_data, on='user_id', how='inner')
merged_data = merged_data.merge(tag_genre_data, on='tag_id', how='inner')

print(merged_data.head())
```

```
   user_id     artist_name              spotify_id  popularity  tag_id   genre
0    24824     tatum quinn  3s0DwmaExsRr8KGfE8RkhH         NaN     601  reggae
1   303724  jason kerrison  7iLGqGUSoPQtj8OH61HFwZ         NaN     601  reggae
2   451943   milo x kahefa  70o5L3YGkE7lxVWPro3fkV         NaN     601  reggae
3   169814  gerson marques  7vdWVGhRjWiT8VHuS9D9a5         NaN     601  reggae
4   373197         engeezo  3EhLaFxXQijaAcWcoEGWJC         NaN     601  reggae
```

```
print(spotify_data[spotify_data['popularity'].isna()])
```

## Requête SQL

La requête SQL agrège les données pour répertorier l'ID Spotify de chaque artiste, l'ID utilisateur, le genre, le nombre total de genres attribués et le nombre d'artistes par genre. Les résultats sont enregistrés dans un fichier CSV pour analyse.

```
conn = sqlite3.connect(':memory:')
cursor = conn.cursor()

merged_data.to_sql('merged_data', conn, index=False, if_exists='replace')

sql_query = """
SELECT
    md.artist_name,
    md.spotify_id,
    md.user_id,
    md.genre,
    COUNT(DISTINCT md.tag_id) AS total_genres,
    COUNT(DISTINCT a.user_id) AS total_artists
FROM
    merged_data md
JOIN
    merged_data a ON md.genre = a.genre
GROUP BY
    md.artist_name, md.spotify_id, md.user_id, md.genre
"""

results = pd.read_sql_query(sql_query, conn)
print(results)
```

```
results.to_csv('/content/drive/MyDrive/Spotify/artist_genre_summary.csv', index=False)
```