Most Streamed Spotify Songs 2024

Name :ADEBAYO ANTHONY AYOWOLE | dataset from Kaggle: https://www.kaggle.com/datasets/nelgiriyewithar streamed-spotify-songs-2024 (https://www.kaggle.com/datasets/nelgiriyewitha streamed-spotify-songs-2024)

Description

This dataset offers a detailed collection of the top-streamed songs on Spotify in 2024. It provides in-depth information on each track's features, popularity, and availability across various music platforms, making it a valuable tool for music analysts, enthusiasts, and industry professionals. The dataset includes details such as the track name, artist, release date, ISRC, streaming statistics, and presence on platforms like YouTube, TikTok, and others.

In [93]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
#load the dataset
```

In [94]:
```
spotify_data = pd.read_csv("spotify_data.csv", encoding = 'ISO-8859-1')

#the reason for adding (encoding = 'ISO-8859-1') in loading the dataset i
#UTF-8: which is default and most common. so I added
#ISO-8859-1 (latin1):
#Common fallback, especially for Western European languages.
#Windows-1252: Similar to ISO-8859-1, used in many Windows systems.
#If the file reads successfully after specifying the correct encoding, yo
#visualization steps.
```

In [95]:
```
1 spotify_data
```

Out[95]:

	Track	Album Name	Artist	Release Date	ISRC	All Time Rank	Track Score	Sp Str
0	MILLION DOLLAR BABY	Million Dollar Baby - Single	Tommy Richman	4/26/2024	QM24S2402528	1	725.4	390,47
1	Not Like Us	Not Like Us	Kendrick Lamar	5/4/2024	USUG12400910	2	545.9	323,70
2	i like the way you kiss me	I like the way you kiss me	Artemas	3/19/2024	QZJ842400387	3	538.4	601,30
3	Flowers	Flowers - Single	Miley Cyrus	1/12/2023	USSM12209777	4	444.9	2,031,28
4	Houdini	Houdini	Eminem	5/31/2024	USUG12403398	5	423.3	107,03
...	...	...	...	...	...	...	...	

Top 10 Artist By Total Spotify Stream

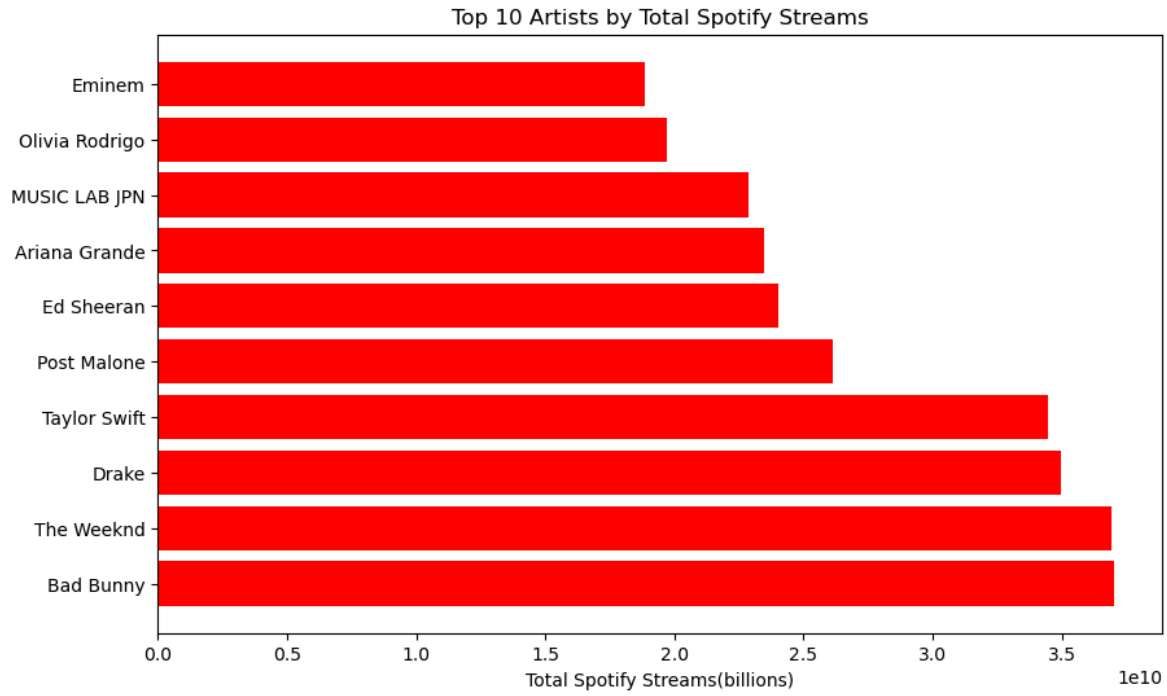
In [98]:
```
1 #spotify_data['Spotify Streams']=spotify_data['Spotify Streams'].str.repl
2 artist_streams = spotify_data.groupby('Artist')['Spotify Streams'].sum().
3 top10_artists = artist_streams.sort_values(by='Spotify Streams',ascending
```

In [99]:
```
1 top10_artists
```

Out[99]:

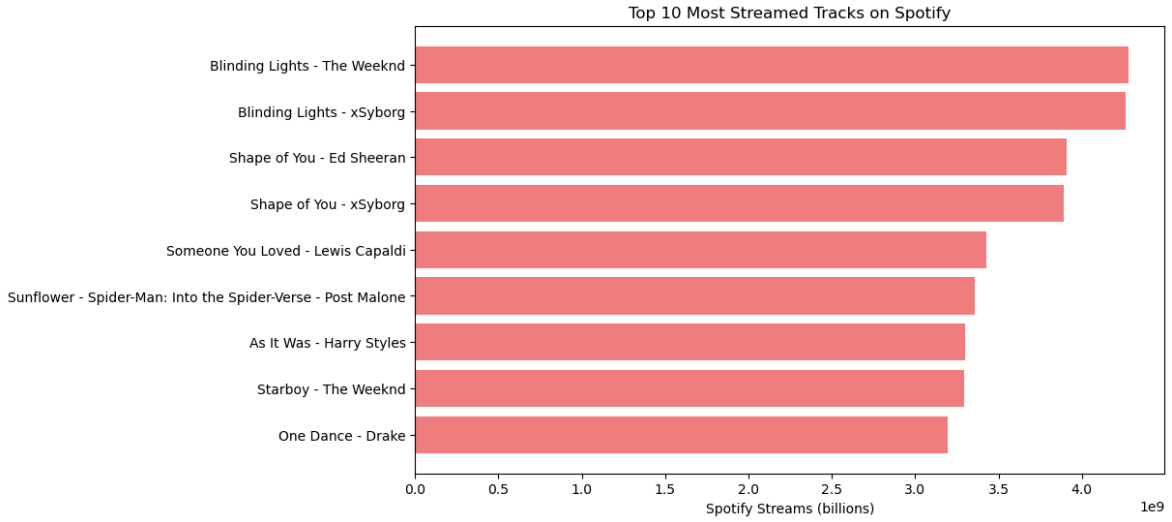
	Artist	Spotify Streams
183	Bad Bunny	3.705483e+10
1728	The Weeknd	3.694854e+10
517	Drake	3.496216e+10
1690	Taylor Swift	3.447077e+10
1428	Post Malone	2.613747e+10
545	Ed Sheeran	2.401490e+10
129	Ariana Grande	2.346499e+10
1147	MUSIC LAB JPN	2.286669e+10
1355	Olivia Rodrigo	1.972922e+10
576	Eminem	1.887888e+10

In [100]:
```
# plot the top 10 artists
plt.figure(figsize=(10,6))
plt.barh(top10_artists['Artist'], top10_artists['Spotify Streams'], color
plt.xlabel('Total Spotify Streams(billions)')
plt.title('Top 10 Artists by Total Spotify Streams')
plt.show()
```



2. Top 10 Most Streamed Tracks on Spotify

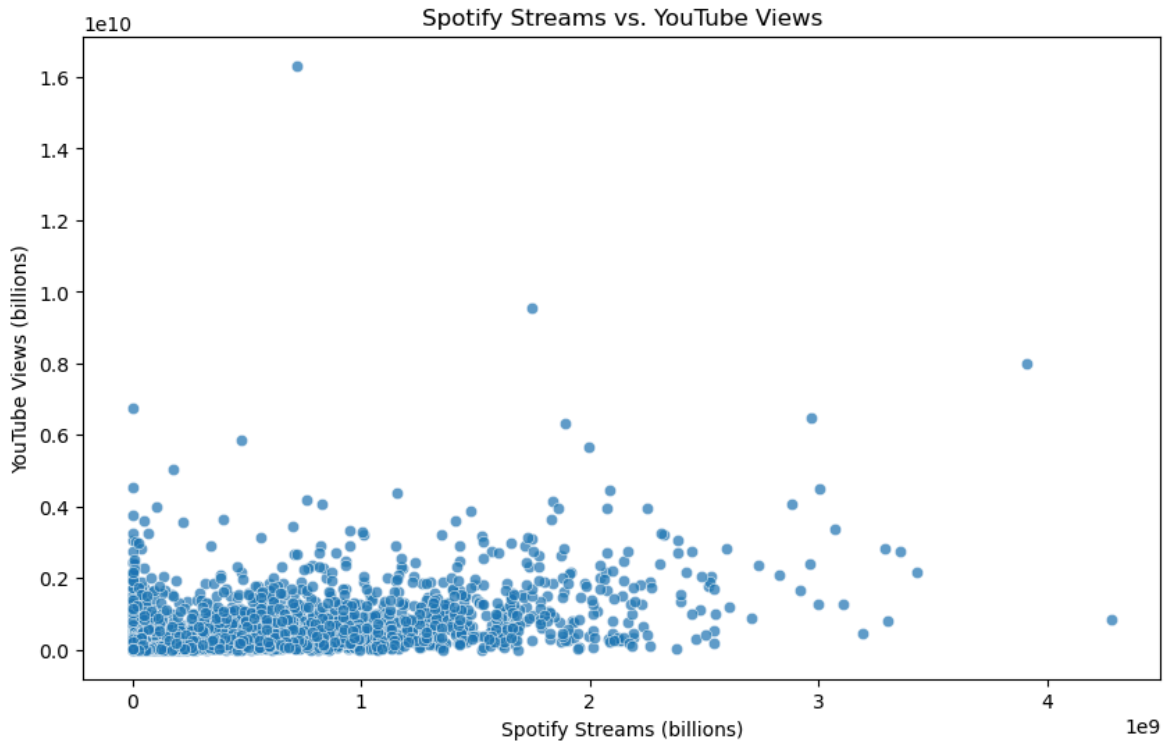
In [59]:
```
# Sort the data by "Spotify Streams" to find the top 10 most streamed tra
top_tracks = spotify_data[['Track', 'Artist', 'Spotify Streams']].sort_va

# Plot the top 10 most streamed tracks
plt.figure(figsize=(10, 6))
plt.barh(top_tracks['Track'] + ' - ' + top_tracks['Artist'], top_tracks['
plt.xlabel('Spotify Streams (billions)')
plt.title('Top 10 Most Streamed Tracks on Spotify')
plt.gca().invert_yaxis()
plt.show()
```



3. Comparison Between Spotify Streams and YouTube Views

In [60]:
```
# Convert the 'YouTube Views' column to strings (in case it contains non-
spotify_data['YouTube Views'] = spotify_data['YouTube Views'].astype(str)
```

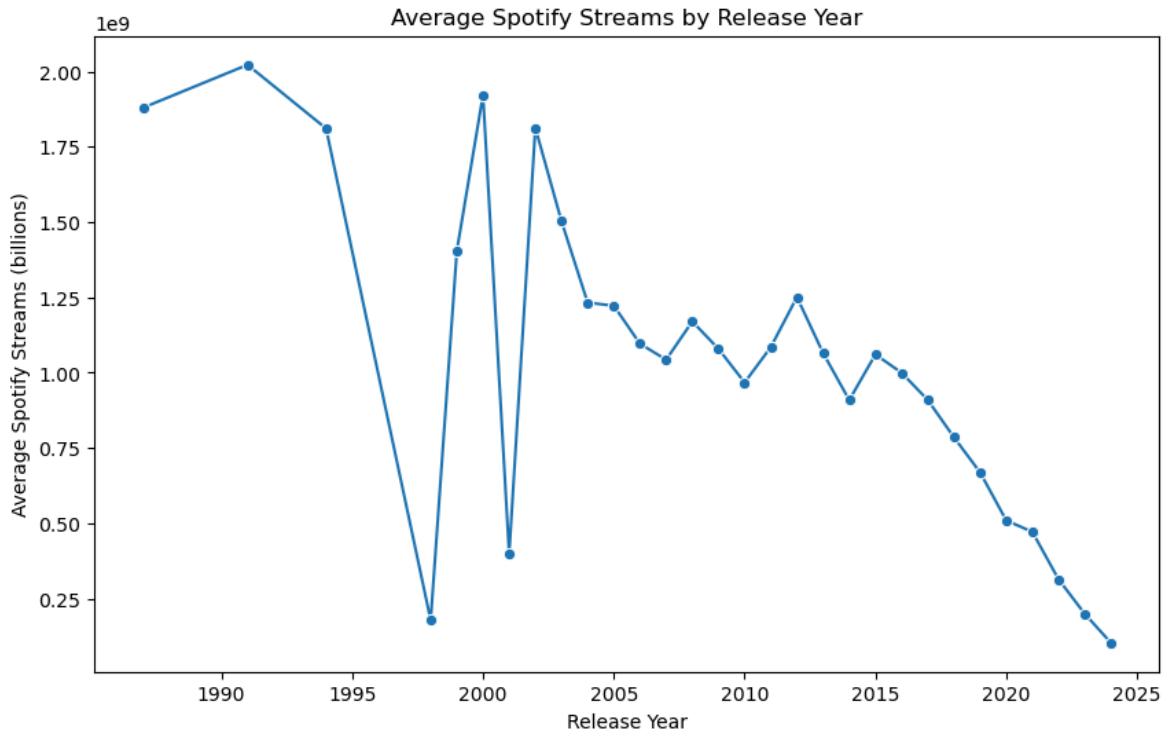
In [61]:
```
# Clean YouTube Views
spotify_data['YouTube Views'] = spotify_data['YouTube Views'].str.replace

# Plot a scatter plot to compare Spotify Streams and YouTube Views
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Spotify Streams', y='YouTube Views', data=spotify_data
plt.xlabel('Spotify Streams (billions)')
plt.ylabel('YouTube Views (billions)')
plt.title('Spotify Streams vs. YouTube Views')
plt.show()
```



4. Correlation Between Release Year and Streaming Numbers

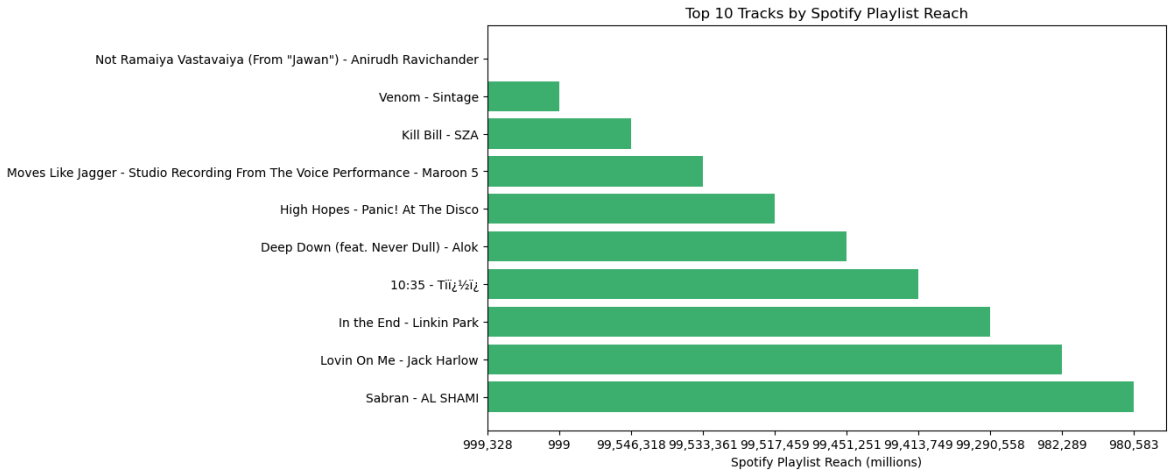
In [69]:
```
# Clean the 'Spotify Streams' column: remove commas and convert to numeri
#spotify_data['Spotify Streams'] = spotify_data['Spotify Streams'].str.re
#spotify_data['Spotify Streams'] = spotify_data['Spotify Streams'].str.re
```

In [70]:
```
# Extract year from the release date
spotify_data['Release Year'] = pd.to_datetime(spotify_data['Release Date'

# Calculate average streams by release year
yearly_streams = spotify_data.groupby('Release Year')['Spotify Streams'].

# Plot the correlation between release year and average streams
plt.figure(figsize=(10, 6))
sns.lineplot(x='Release Year', y='Spotify Streams', data=yearly_streams,
plt.xlabel('Release Year')
plt.ylabel('Average Spotify Streams (billions)')
plt.title('Average Spotify Streams by Release Year')
plt.show()

```



5. Tracks with the Highest Playlist Reach on Spotify

In [71]:
```
# Find the top 10 tracks by playlist reach
top_reach_tracks = spotify_data.sort_values(by='Spotify Playlist Reach',

# Plot the top 10 tracks by playlist reach
plt.figure(figsize=(10, 6))
plt.barh(top_reach_tracks['Track'] + ' - ' + top_reach_tracks['Artist'],
plt.xlabel('Spotify Playlist Reach (millions)')
plt.title('Top 10 Tracks by Spotify Playlist Reach')
plt.gca().invert_yaxis()  # Invert the y-axis to have the highest value o
plt.show()
```



6. Comparison of Explicit vs. Non-Explicit Tracks

In [74]:
```
# Replace 'Explicit' with the correct column name if different
average_streams_explicit = spotify_data.groupby('Explicit Track')['Spotif
```

Group by 'Explicit' and calculate the mean of 'Spotify Streams'

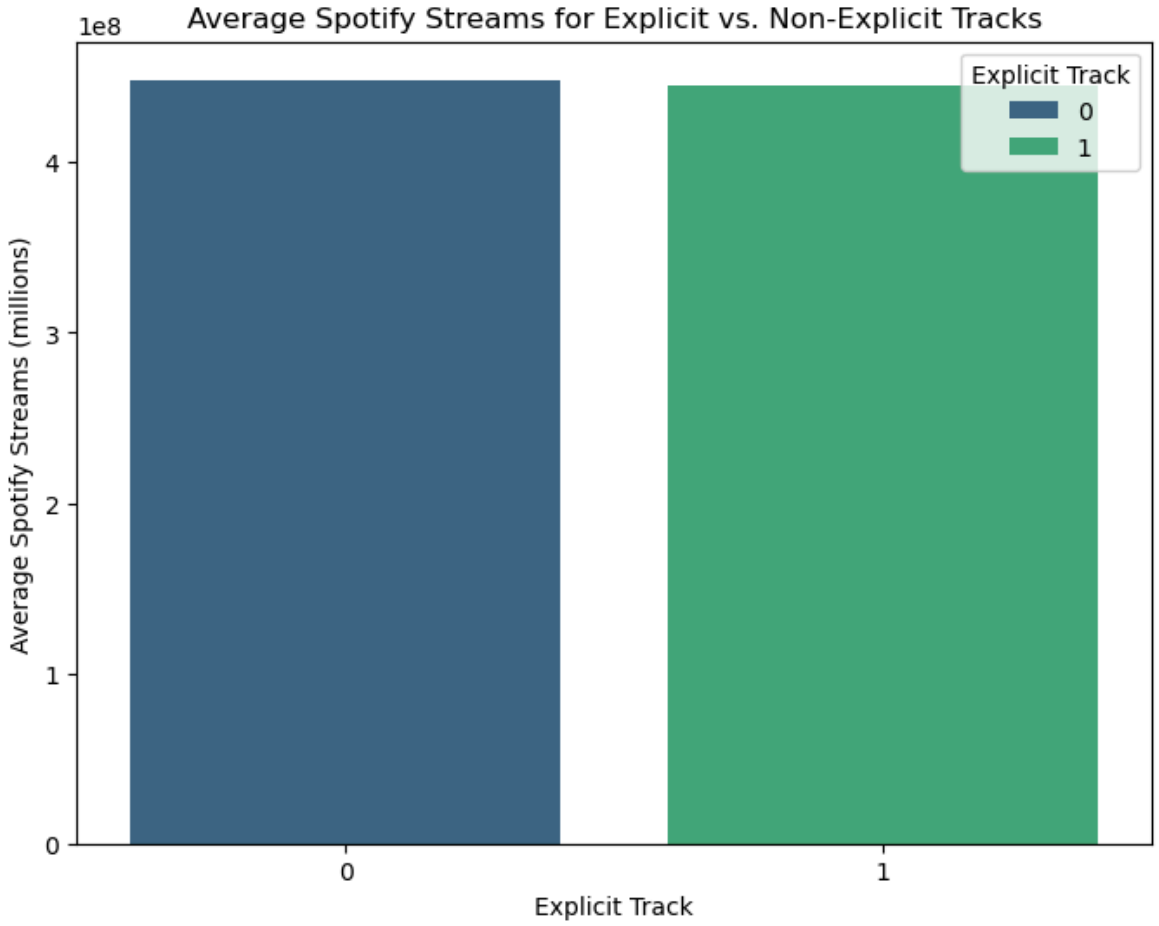
average_streams_explicit = spotify_data.groupby('Explicit Track')['Spotify Streams'].mean().reset_index()

In [80]:
```
average_streams_explicit
```

Out[80]:

	Explicit Track	Spotify Streams
0	0	4.485632e+08
1	1	4.453499e+08

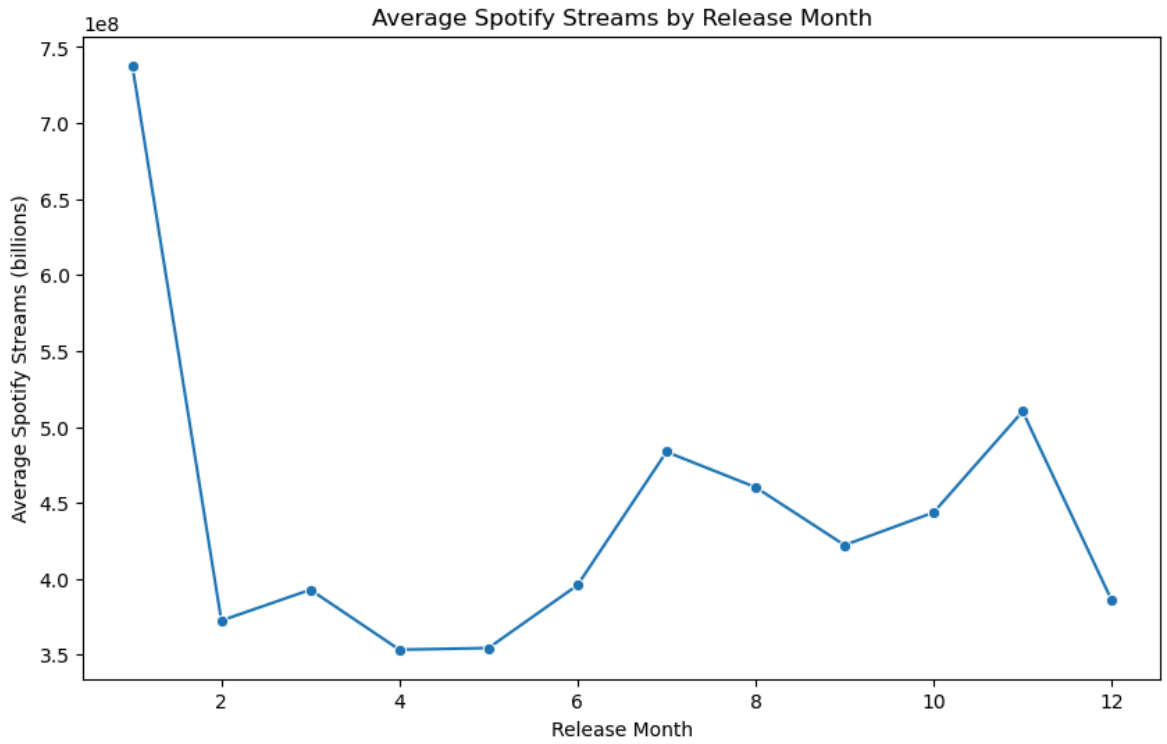
In [85]:
```
# Plotting the average Spotify streams for explicit vs. non-explicit trac
plt.figure(figsize=(8, 6))
sns.barplot(x='Explicit Track',y='Spotify Streams',data=average_streams_e
dodge=False)
plt.xlabel('Explicit Track')
plt.ylabel('Average Spotify Streams (millions)')
plt.title('Average Spotify Streams for Explicit vs. Non-Explicit Tracks')
plt.show()
```



7.Seasonal Trends in Streaming

In [87]:
```
# Extract month from the release date
spotify_data['Release Month'] = pd.to_datetime(spotify_data['Release Date
```

In [88]:
```
# Calculate average streams by release month
monthly_streams = spotify_data.groupby('Release Month')['Spotify Streams'

# Plot seasonal trends in streaming
plt.figure(figsize=(10, 6))
sns.lineplot(x='Release Month', y='Spotify Streams', data=monthly_streams
plt.xlabel('Release Month')
plt.ylabel('Average Spotify Streams (billions)')
plt.title('Average Spotify Streams by Release Month')
plt.show()
```



8. Artist Consistency Over Years

In [89]:
```
# Calculate average streams by artist and release year
artist_yearly_streams = spotify_data.groupby(['Artist', 'Release Year'])[
```

In [90]:

```
# Plot artist consistency (example: top 5 artists with most years in data
top_artists_years = artist_yearly_streams['Artist'].value_counts().head(5
artist_consistency = artist_yearly_streams[artist_yearly_streams['Artist'

plt.figure(figsize=(12, 8))
sns.lineplot(x='Release Year', y='Spotify Streams', hue='Artist', data=ar
plt.xlabel('Release Year')
plt.ylabel('Average Spotify Streams (billions)')
plt.title('Artist Consistency Over Years')
plt.show()
```