```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

#read csv file
df=pd.read_csv("C:/Users/ajayv/Downloads/spotify/spotify_tracks.csv")
df.head()
```

```
                 track_id                                  track_name
\
0  2r0ROhr7pRN4MXDMT1fEmd                  Leo Das Entry (From "Leo")

1  4I38e6Dg52a2o2a8i5Q5PW                                 AAO KILLELLE

2  59NoiRhnom3lTeRFaBzOev      Mayakiriye Sirikiriye - Orchestral EDM

3  5uUqRQd385pvLxC8JX3tXn    Scene Ah Scene Ah - Experimental EDM Mix

4  1KaBRg2xgNeCljmyxBH1mo  Gundellonaa X I Am A Disco Dancer - Mashup


                                         artist_name  year  popularity
\
0                                Anirudh Ravichander  2024          59

1  Anirudh Ravichander, Pravin Mani, Vaishali Sri...  2024          47

2           Anirudh Ravichander, Anivee, Alvin Bruno  2024          35

3  Anirudh Ravichander, Bharath Sankar, Kabilan, ...  2024          24

4  Anirudh Ravichander, Benny Dayal, Leon James, ...  2024          22


                                         artwork_url  \
0  https://i.scdn.co/image/ab67616d0000b273ce9c65...
1  https://i.scdn.co/image/ab67616d0000b273be1b03...
2  https://i.scdn.co/image/ab67616d0000b27334a1dd...
3  https://i.scdn.co/image/ab67616d0000b27332e623...
4  https://i.scdn.co/image/ab67616d0000b2735a59b6...

                                   album_name  acousticness
danceability  \
0                  Leo Das Entry (From "Leo")        0.0241
0.753
1                                AAO KILLELLE        0.0851
0.780
2      Mayakiriye Sirikiriye (Orchestral EDM)        0.0311
0.457
3    Scene Ah Scene Ah (Experimental EDM Mix)        0.2270
0.718
```

```
4  Gundellonaa X I Am a Disco Dancer (Mashup)        0.0153
0.689

   duration_ms  ...   key  liveness  loudness  mode  speechiness
tempo  \
0      97297.0  ...   8.0    0.1000    -5.994   0.0       0.1030
110.997
1     207369.0  ...  10.0    0.0951    -5.674   0.0       0.0952
164.995
2      82551.0  ...   2.0    0.0831    -8.937   0.0       0.1530
169.996
3     115831.0  ...   7.0    0.1240   -11.104   1.0       0.4450
169.996
4     129621.0  ...   7.0    0.3450    -9.637   1.0       0.1580
128.961

   time_signature  valence
track_url  \
0             4.0    0.459
https://open.spotify.com/track/2r0ROhr7pRN4MXD...
1             3.0    0.821
https://open.spotify.com/track/4I38e6Dg52a2o2a...
2             4.0    0.598
https://open.spotify.com/track/59NoiRhnom3lTeR...
3             4.0    0.362
https://open.spotify.com/track/5uUqRQd385pvLxC...
4             4.0    0.593
https://open.spotify.com/track/1KaBRg2xgNeCljm...

   language
0     Tamil
1     Tamil
2     Tamil
3     Tamil
4     Tamil

[5 rows x 22 columns]
```

Checking null value

```
df.isnull().sum()

track_id            0
track_name          0
artist_name         0
year                0
popularity          0
artwork_url         0
```

```
album_name          0
acousticness        0
danceability        0
duration_ms         0
energy              0
instrumentalness    0
key                 0
liveness            0
loudness            0
mode                0
speechiness         0
tempo               0
time_signature      0
valence             0
track_url           0
language            0
dtype: int64
```

Columns name and their data types

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 62317 entries, 0 to 62316
Data columns (total 22 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   track_id          62317 non-null  object 
 1   track_name        62317 non-null  object 
 2   artist_name       62317 non-null  object 
 3   year              62317 non-null  int64  
 4   popularity        62317 non-null  int64  
 5   artwork_url       62317 non-null  object 
 6   album_name        62317 non-null  object 
 7   acousticness      62317 non-null  float64
 8   danceability      62317 non-null  float64
 9   duration_ms       62317 non-null  float64
 10  energy            62317 non-null  float64
 11  instrumentalness  62317 non-null  float64
 12  key               62317 non-null  float64
 13  liveness          62317 non-null  float64
 14  loudness          62317 non-null  float64
 15  mode              62317 non-null  float64
 16  speechiness       62317 non-null  float64
 17  tempo             62317 non-null  float64
 18  time_signature    62317 non-null  float64
 19  valence           62317 non-null  float64
 20  track_url         62317 non-null  object 
 21  language          62317 non-null  object
dtypes: float64(13), int64(2), object(7)
memory usage: 10.5+ MB
```

statistical summary

```
df.describe()
```

```
               year    popularity  acousticness  danceability
duration_ms  \
count  62317.000000  62317.000000  62317.000000  62317.000000
6.231700e+04
mean    2014.425935     15.358361      0.362292      0.596807
2.425270e+05
std        9.645113     18.626908      0.314609      0.186209
1.129999e+05
min     1971.000000      0.000000     -1.000000     -1.000000
5.000000e+03
25%     2011.000000      0.000000      0.067100      0.497000
1.921600e+05
50%     2017.000000      7.000000      0.286000      0.631000
2.362670e+05
75%     2022.000000     26.000000      0.632000      0.730000
2.862400e+05
max     2024.000000     93.000000      0.996000      0.986000
4.581483e+06

             energy  instrumentalness           key      liveness  \
count  62317.000000      62317.000000  62317.000000  62317.000000
mean       0.602496          0.146215      5.101658      0.194143
std        0.246144          0.307804      3.553469      0.172030
min       -1.000000         -1.000000     -1.000000     -1.000000
25%        0.440000          0.000000      2.000000      0.093200
50%        0.639000          0.000025      5.000000      0.125000
75%        0.803000          0.015200      8.000000      0.243000
max        1.000000          0.999000     11.000000      0.998000

              loudness          mode   speechiness         tempo  \
count   62317.000000  62317.000000  62317.000000  62317.000000
mean      -65.103433      0.586052      0.087722    117.931247
std      2369.051478      0.493682      0.115150     28.509459
min   -100000.000000     -1.000000     -1.000000     -1.000000
25%       -10.727000      0.000000      0.036700     95.942000
50%        -7.506000      1.000000      0.048900    117.991000
75%        -5.456000      1.000000      0.089100    135.081000
max         1.233000      1.000000      0.959000    239.970000

       time_signature       valence
```

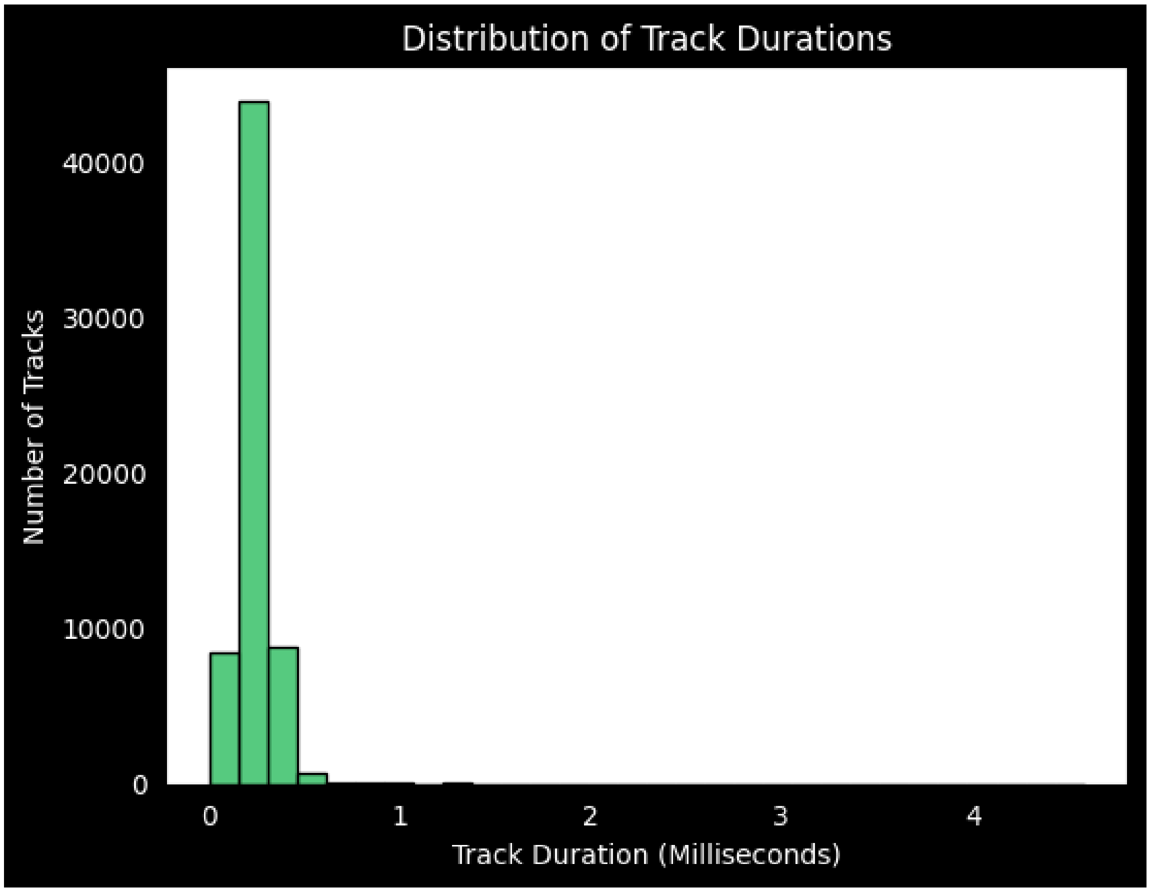
```
count    62317.000000  62317.000000
mean         3.857086      0.495226
std          0.502660      0.264787
min         -1.000000     -1.000000
25%          4.000000      0.292000
50%          4.000000      0.507000
75%          4.000000      0.710000
max          5.000000      0.995000
```

What is the most popular track based on the 'popularity' score?

```
most_popular_track = df.loc[df['popularity'].idxmax()]
print(f"Most Popular Track: {most_popular_track['track_name']} by
{most_popular_track['artist_name']}")

Most Popular Track: Big Dawgs by Hanumankind, Kalmi
```

What is the average duration of all tracks?

```
average_duration = df['duration_ms'].mean()
print(f"Average Track Duration: {average_duration:.2f} milliseconds")

# Spotify Green colour
spotify_green = '#1DB954'

# Set the background color
plt.figure(facecolor='black')

sns.histplot(data=df, x='duration_ms',bins=30, color=spotify_green)
plt.xlabel('Track Duration (Milliseconds)', color='white')
plt.ylabel('Number of Tracks', color='white')
plt.title('Distribution of Track Durations', color='white')
plt.xticks(color='white')
plt.yticks(color='white')
plt.show()

Average Track Duration: 242527.04 milliseconds
```

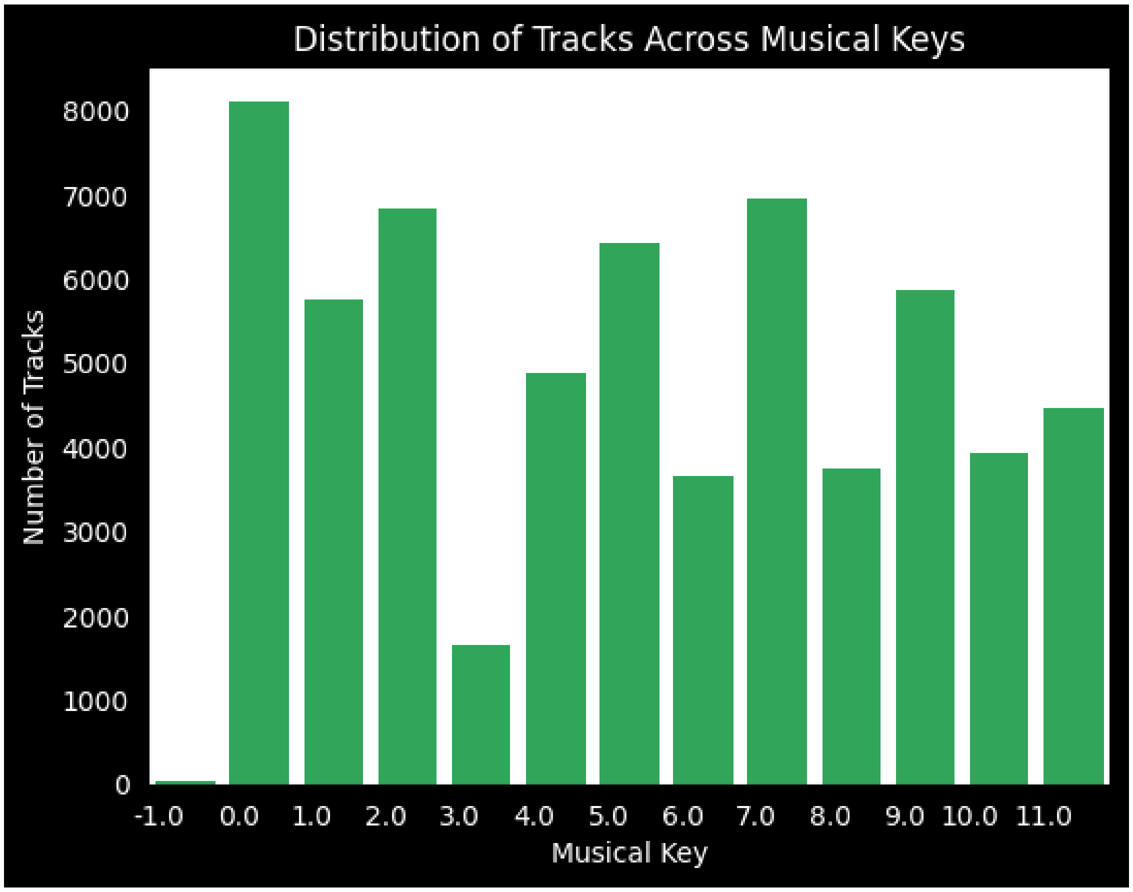


Which artist has the highest number of tracks?

```
top_artist = df['artist_name'].value_counts().idxmax()
print(f"Artist with the most tracks: {top_artist}")

Artist with the most tracks: Shankar Mahadevan
```

What is the distribution of tracks across different musical keys?

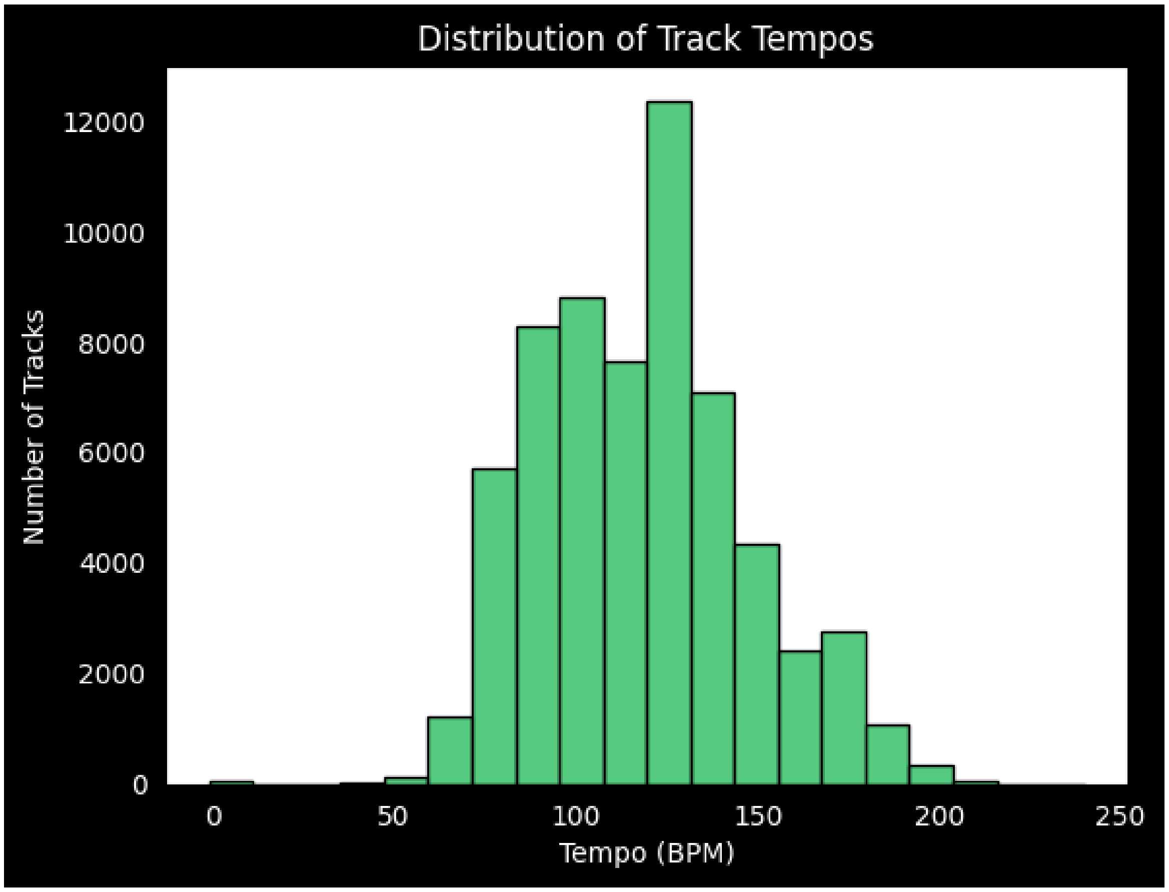
```
key_counts = df['key'].value_counts()

plt.figure(facecolor='black')
sns.barplot(x=key_counts.index, y=key_counts.values, color='#1DB954')
plt.xlabel('Musical Key', color='white')
plt.ylabel('Number of Tracks', color='white')
plt.title('Distribution of Tracks Across Musical Keys', color='white')
plt.xticks(rotation=0, ha='right', color='white')
plt.yticks(color='white')
plt.show()
```



What is the average tempo of all tracks?

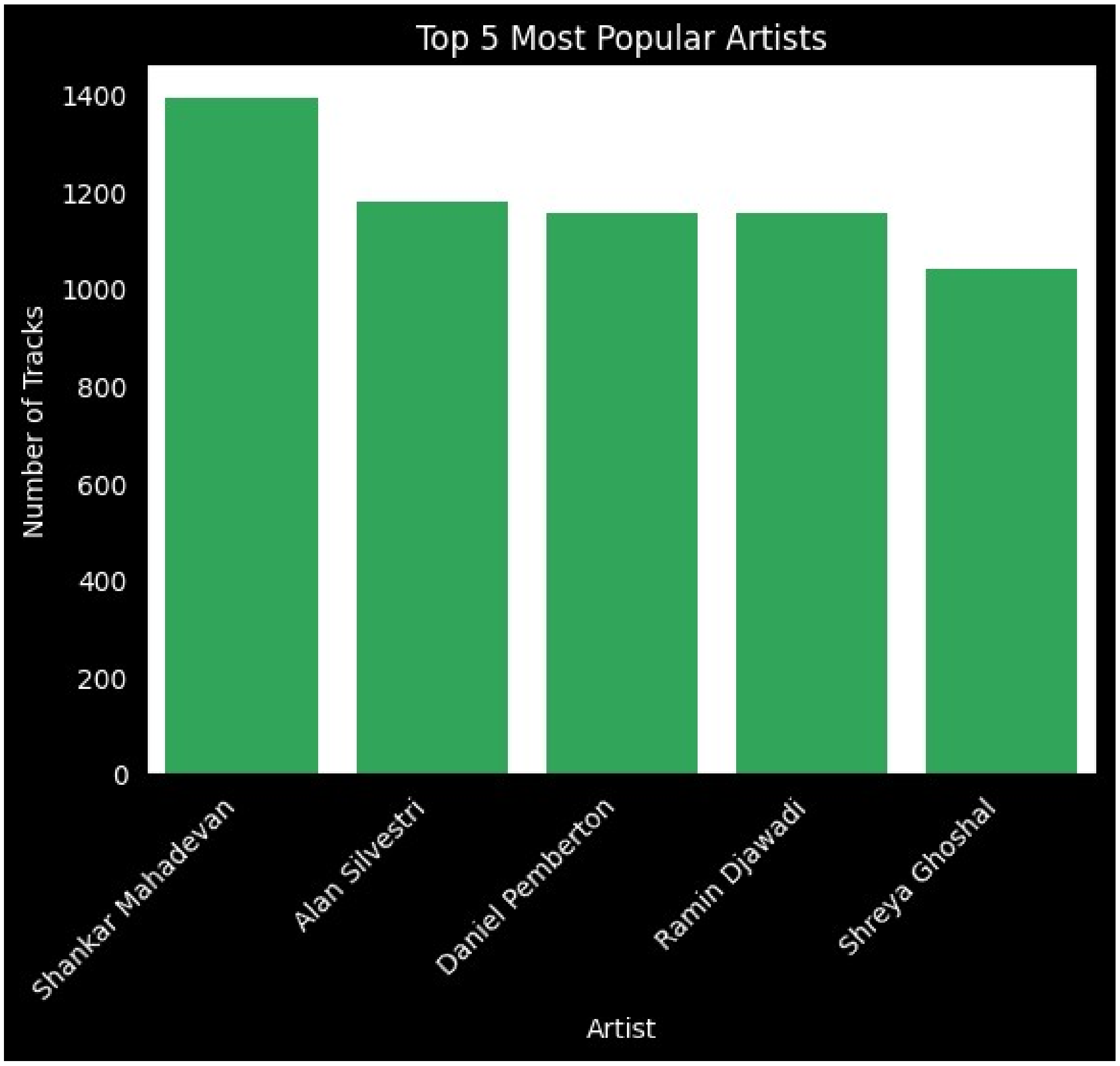
```
average_tempo = df['tempo'].mean()
print(f"Average Tempo: {average_tempo:.2f} BPM")

plt.figure(facecolor='black')
sns.histplot(data=df, x='tempo', bins=20, color='#1DB954')
plt.xlabel('Tempo (BPM)', color='white')
plt.ylabel('Number of Tracks', color='white')
plt.title('Distribution of Track Tempos', color='white')
plt.xticks(color='white')
plt.yticks(color='white')
plt.show()
```

```
Average Tempo: 117.93 BPM
```



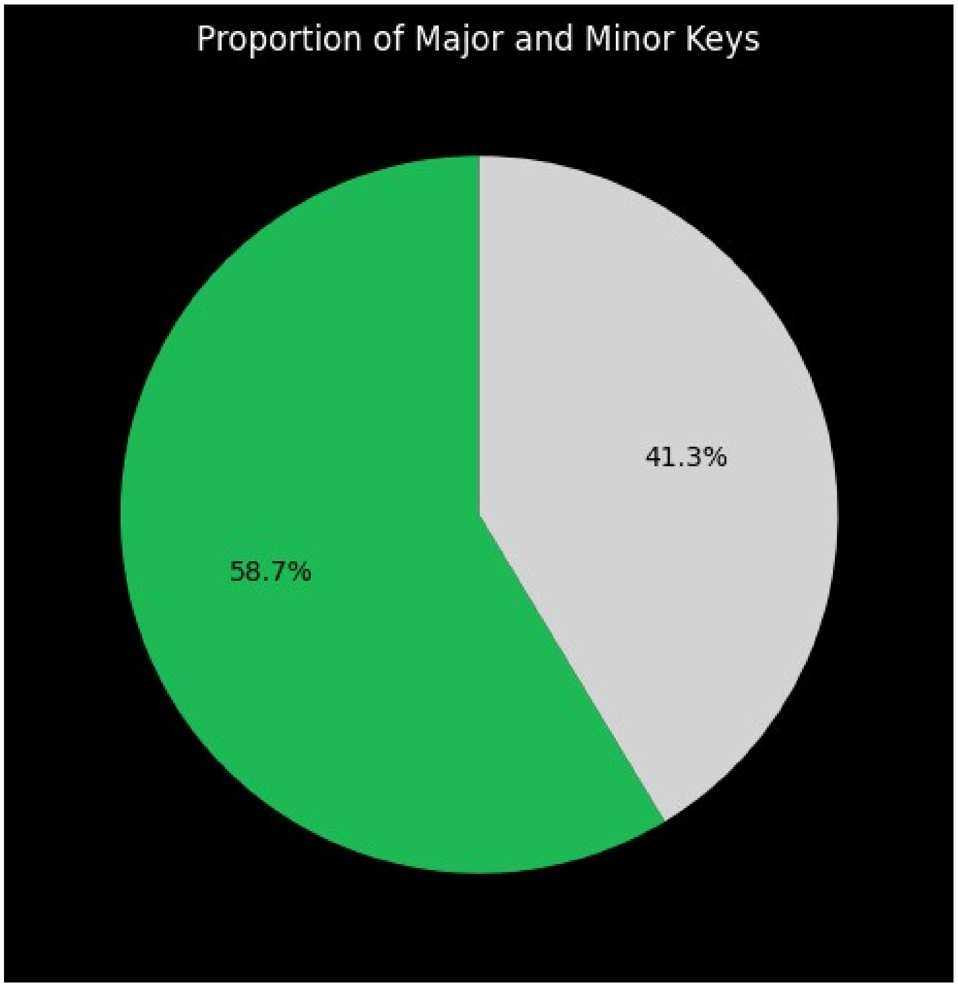
Find the top 5 most popular artists.

```
top_artists = df['artist_name'].value_counts().head(5)

plt.figure(facecolor='black')
sns.barplot(x=top_artists.index, y=top_artists.values,
color='#1DB954')
plt.xlabel('Artist', color='white')
plt.ylabel('Number of Tracks', color='white')
plt.title('Top 5 Most Popular Artists', color='white')
plt.xticks(rotation=45, ha='right', color='white')
plt.yticks(color='white')
plt.show()
```



What is the percentage of tracks in the major key?

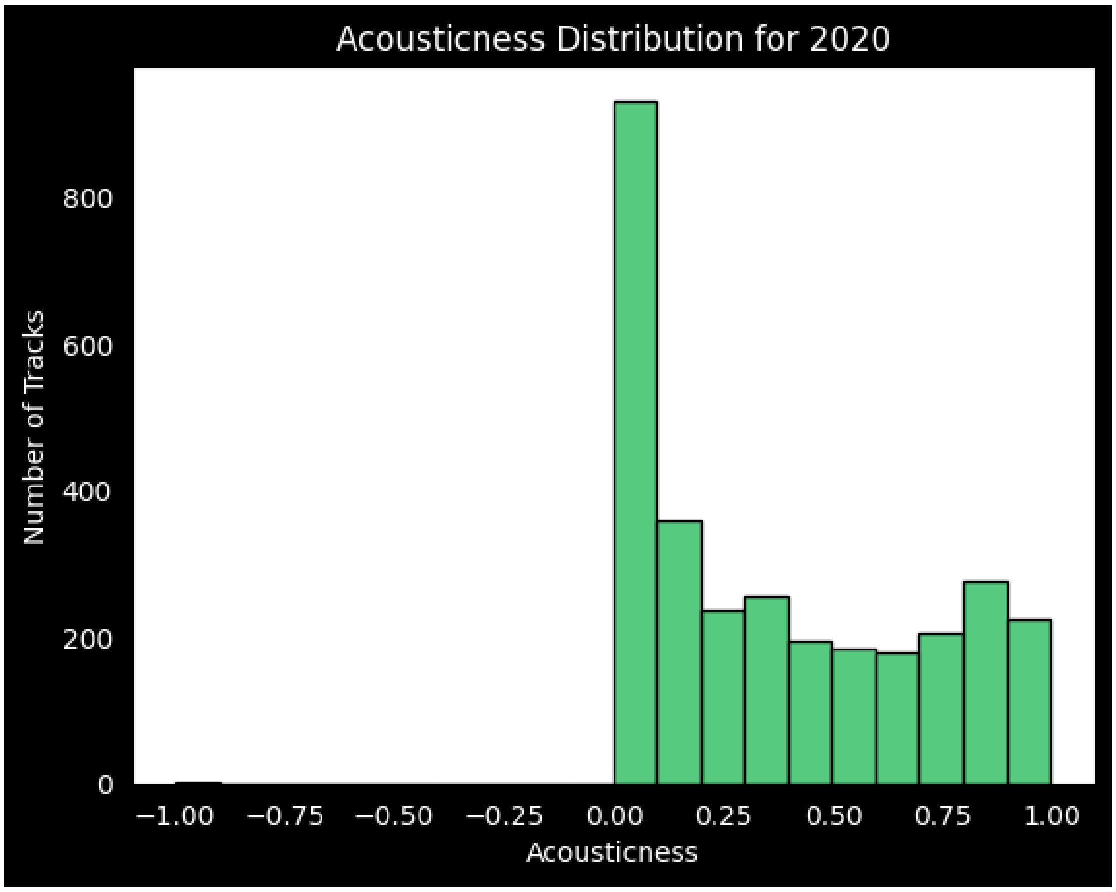
```
major_tracks = df[df['mode'] == 1]
percentage_major = (len(major_tracks) / len(df)) * 100

plt.figure(figsize=(6, 6), facecolor='black')
plt.pie([len(major_tracks), len(df) - len(major_tracks)],
        labels=['Major', 'Minor'],
        autopct='%1.1f%%',
        startangle=90,
        colors=['#1DB954', 'lightgray'])
plt.title('Proportion of Major and Minor Keys', color='white')
plt.show()
```



Find the average acousticness of tracks released in 2020

```
# Filter for tracks released in 2020
tracks_2020 = df[df['year'] == 2020]

# Calculate the average acousticness for 2020
average_acousticness_2020 = tracks_2020['acousticness'].mean()

# Print the average acousticness
print(f"Average Acousticness for 2020:
{average_acousticness_2020:.2f}")

# Create a histogram of acousticness for 2020
plt.figure(facecolor='black')
sns.histplot(data=tracks_2020, x='acousticness', bins=20,
color='#1DB954')
plt.xlabel('Acousticness', color='white')
plt.ylabel('Number of Tracks', color='white')
plt.title('Acousticness Distribution for 2020', color='white')
plt.xticks(color='white')
plt.yticks(color='white')
plt.show()
```

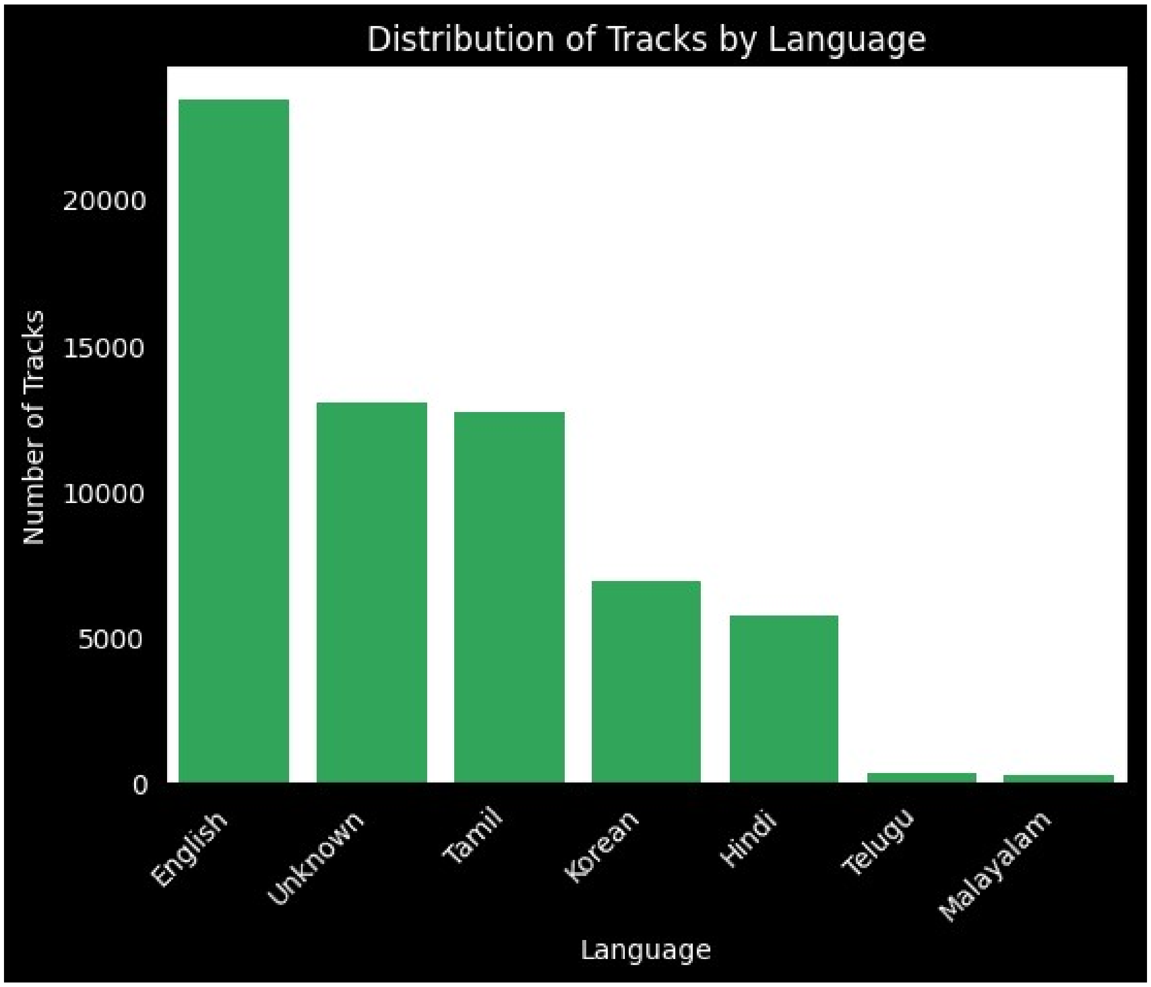
```
Average Acousticness for 2020: 0.37
```



Which language has the highest number of tracks?

```
language_counts = df['language'].value_counts()

# Find the language with the highest number of tracks
most_common_language = language_counts.idxmax()
print(f"Most common language: {most_common_language}")

# Create a bar chart
plt.figure(facecolor='black')
sns.barplot(x=language_counts.index, y=language_counts.values,
color='#1DB954')
plt.xlabel('Language', color='white')
plt.ylabel('Number of Tracks', color='white')
plt.title('Distribution of Tracks by Language', color='white')
plt.xticks(rotation=45, ha='right', color='white')
plt.yticks(color='white')
plt.show()
```

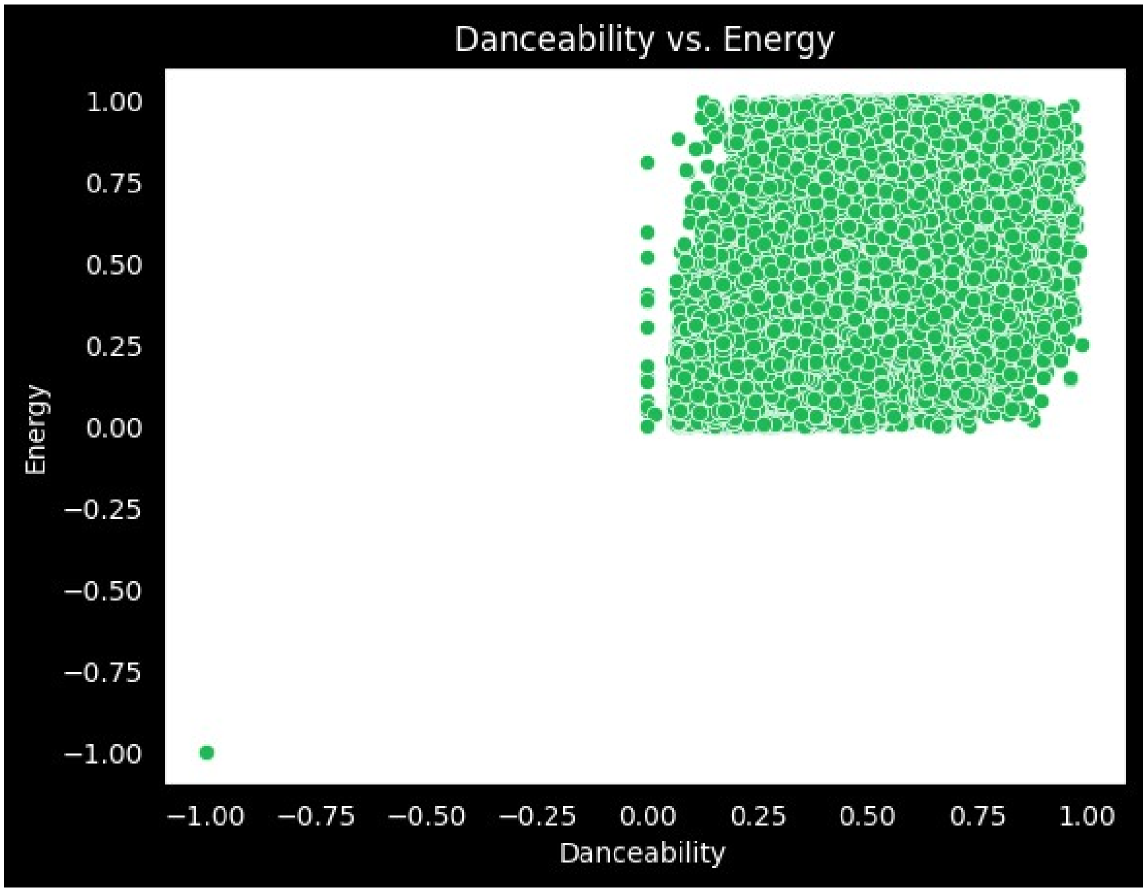
```
Most common language: English
```



What is the correlation between danceability and energy?

```
correlation = df['danceability'].corr(df['energy'])
print(f"Correlation between danceability and energy:
{correlation:.2f}")
```

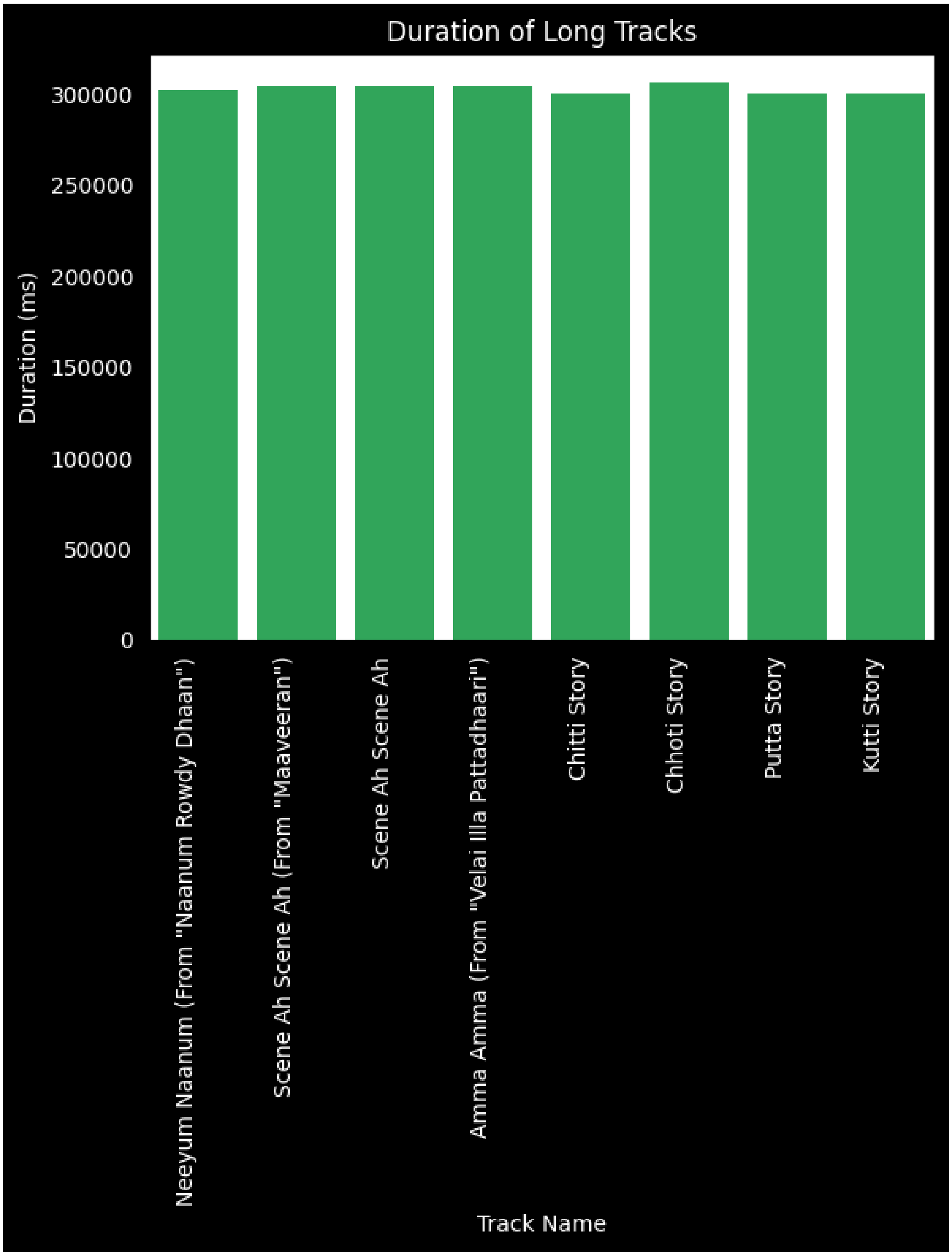
```
# Create a scatter plot
plt.figure(facecolor='black')
sns.scatterplot(x='danceability', y='energy', data=df,
color='#1DB954')
plt.xlabel('Danceability', color='white')
plt.ylabel('Energy', color='white')
plt.title('Danceability vs. Energy', color='white')
plt.xticks(color='white')
plt.yticks(color='white')
plt.show()

Correlation between danceability and energy: 0.48
```



Find all tracks with a duration greater than 5 minutes.

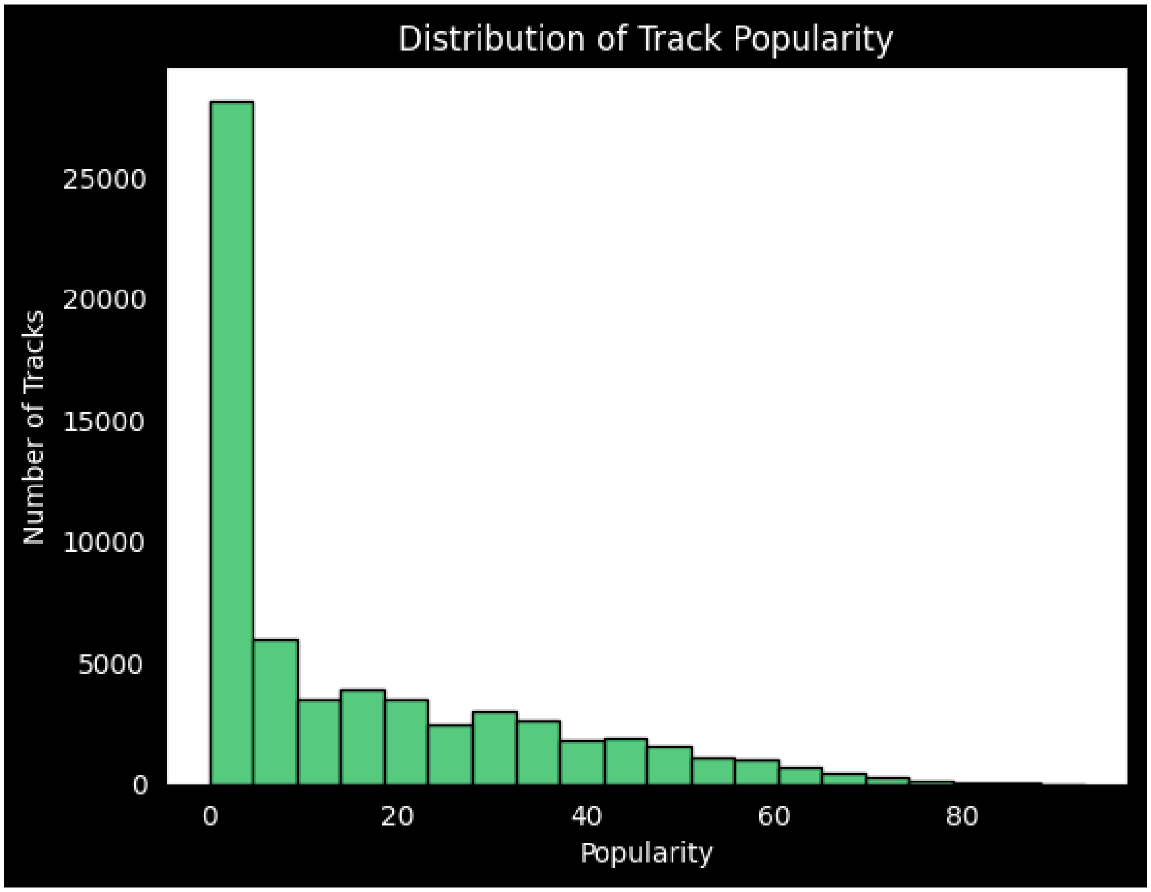
```
long_tracks = df[df['duration_ms'] > 300000]  # 300000 milliseconds =
5 minutes

plt.figure(facecolor='black')
```

```
sns.barplot(x='track_name', y='duration_ms', data=long_tracks[:10],
color='#1DB954')  # Plot only the top 10 long tracks
plt.xlabel('Track Name', color='white')
plt.ylabel('Duration (ms)', color='white')
plt.title('Duration of Long Tracks', color='white')
plt.xticks(rotation=90, ha='right', color='white')
plt.yticks(color='white')
plt.show()
```


Duration of Long Tracks
Duration (ms)
300000
250000
200000
150000
100000
50000
0
Neeyum Naanum (From "Naanum Rowdy Dhaan")
Scene Ah Scene Ah (From "Maaveeran")
Scene Ah Scene Ah
Amma Amma (From "Velai Illa Pattadhaari")
Chitti Story
Chhoti Story
Putta Story
Kutti Story
Track Name

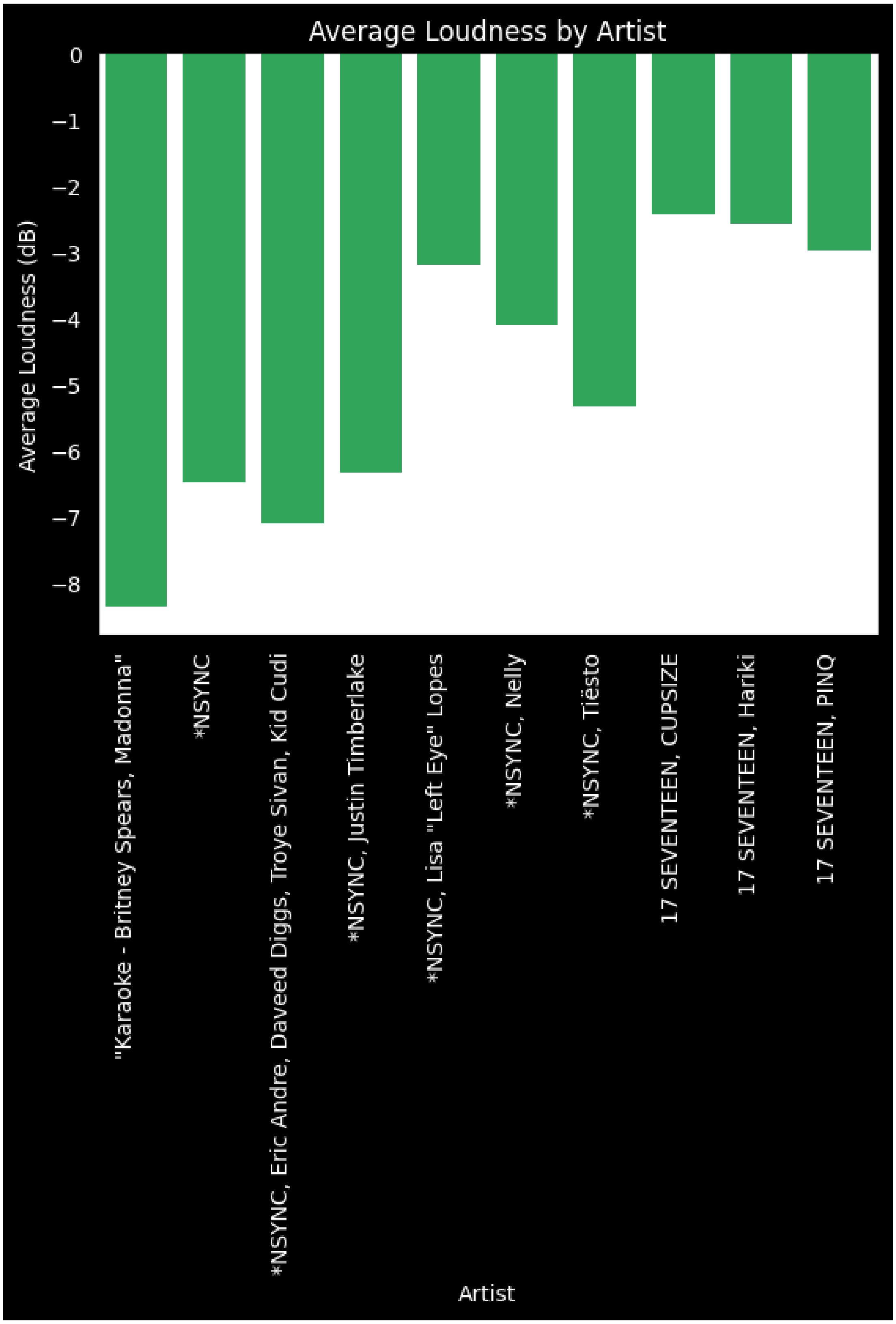
Create a histogram of track popularity.

```
plt.figure(facecolor='black')
sns.histplot(data=df, x='popularity', bins=20, color='#1DB954')
plt.xlabel('Popularity', color='white')
plt.ylabel('Number of Tracks', color='white')
plt.title('Distribution of Track Popularity', color='white')
plt.xticks(color='white')
plt.yticks(color='white')
plt.show()
```



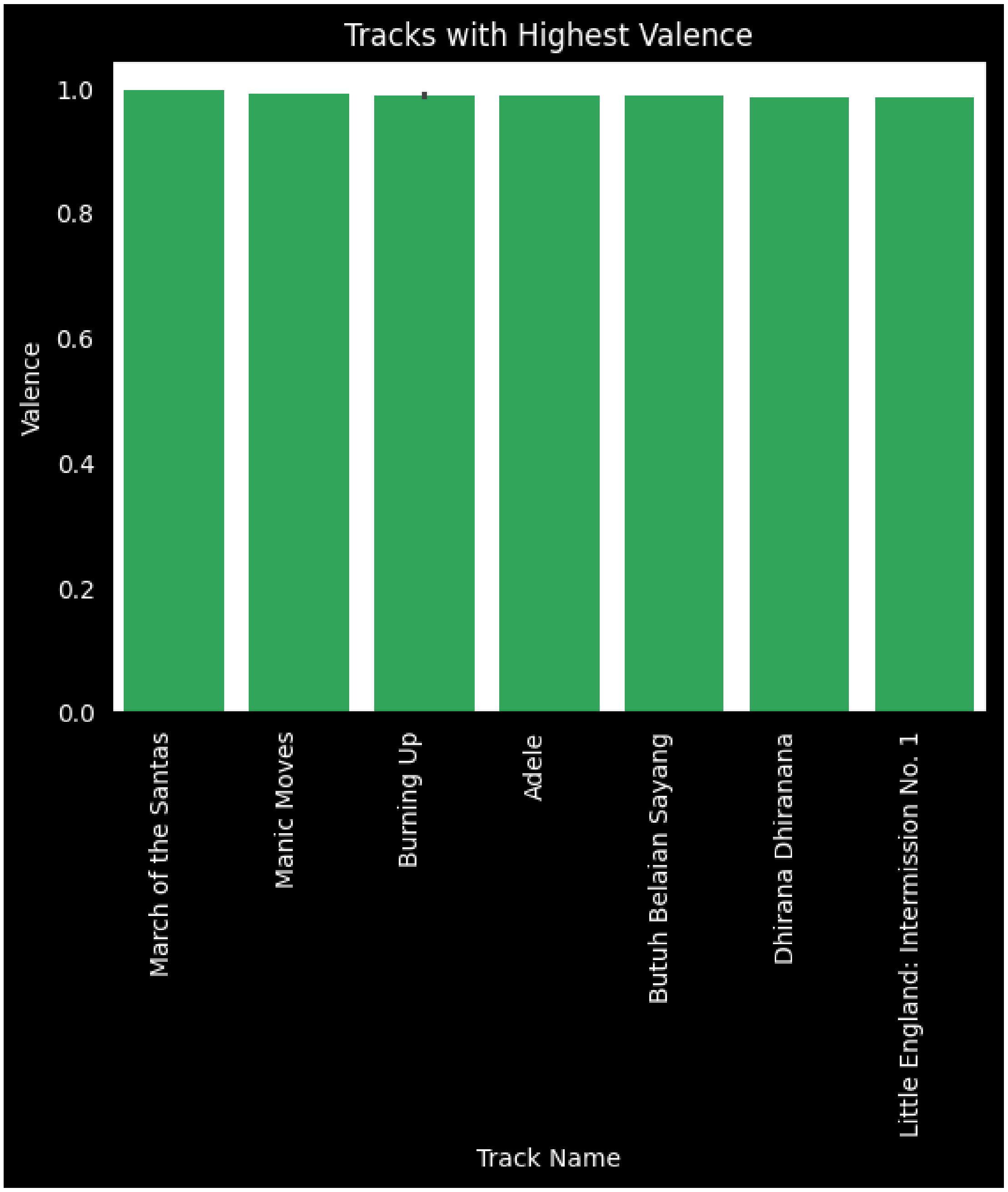
Find the average loudness of tracks by artist.

```
average_loudness_by_artist = df.groupby('artist_name')
['loudness'].mean()

plt.figure(facecolor='black')
sns.barplot(x=average_loudness_by_artist.index[:10],
            y=average_loudness_by_artist.values[:10],
            color='#1DB954')
plt.xlabel('Artist', color='white')
plt.ylabel('Average Loudness (dB)', color='white')
plt.title('Average Loudness by Artist', color='white')
plt.xticks(rotation=90, ha='right', color='white')
plt.yticks(color='white')
plt.show()
```


Average Loudness by Artist
Average Loudness (dB)
0
−1
−2
−3
−4
−5
−6
−7
−8
"Karaoke - Britney Spears, Madonna"
*NSYNC
*NSYNC, Eric Andre, Daveed Diggs, Troye Sivan, Kid Cudi
*NSYNC, Justin Timberlake
*NSYNC, Lisa "Left Eye" Lopes
*NSYNC, Nelly
*NSYNC, Tiësto
17 SEVENTEEN, CUPSIZE
17 SEVENTEEN, Hariki
17 SEVENTEEN, PINQ
Artist

Find the tracks with the highest valence.

```python
top_valence_tracks = df.nlargest(10, 'valence')

plt.figure(facecolor='black')
sns.barplot(x='track_name', y='valence', data=top_valence_tracks, 
color='#1DB954')
plt.xlabel('Track Name', color='white')
plt.ylabel('Valence', color='white')
plt.title('Tracks with Highest Valence', color='white')
plt.xticks(rotation=90, ha='right', color='white')
plt.yticks(color='white')
plt.show()
```


Tracks with Highest Valence
Valence
1.0
0.8
0.6
0.4
0.2
0.0
March of the Santas
Manic Moves
Burning Up
Adele
Butuh Belaian Sayang
Dhirana Dhiranana
Little England: Intermission No. 1
Track Name