In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df_tracks=pd.read_csv("tracks.csv")
```

In [3]:
```
df_tracks.head()
```

Out[3]:

	id	name	popularity	duration_ms	explicit	artists	
0	35iwgR4jXetI318WEWsa1Q	Carve	6	126903	0	['Uli']	['45tIt06Xo
1	021ht4sdgPcrDgSk7JTbKY	Capítulo 2.16 - Banquero Anarquista	0	98200	0	['Fernando Pessoa']	['14jtPCOoNZwc
2	07A5yehtSnoedViJAZkNnc	Vivo para Quererte - Remasterizado	0	181640	0	['Ignacio Corsini']	['5LiOoJbxVSAM
3	08FmqUhxtyLTn6pAh6bk45	El Prisionero - Remasterizado	0	176907	0	['Ignacio Corsini']	['5LiOoJbxVSAM
4	08y9GfoqCWfOGsKdwojr5e	Lady of the Evening	0	163080	0	['Dick Haymes']	['3BiJGZsyX9s

In [4]:
```
df_tracks.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 586672 entries, 0 to 586671
Data columns (total 20 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   id                586672 non-null  object 
 1   name              586601 non-null  object 
 2   popularity        586672 non-null  int64  
 3   duration_ms       586672 non-null  int64  
 4   explicit          586672 non-null  int64  
 5   artists           586672 non-null  object 
 6   id_artists        586672 non-null  object 
 7   release_date      586672 non-null  object 
 8   danceability      586672 non-null  float64
 9   energy            586672 non-null  float64
 10  key               586672 non-null  int64  
 11  loudness          586672 non-null  float64
 12  mode              586672 non-null  int64  
 13  speechiness       586672 non-null  float64
 14  acousticness      586672 non-null  float64
 15  instrumentalness  586672 non-null  float64
 16  liveness          586672 non-null  float64
 17  valence           586672 non-null  float64
 18  tempo             586672 non-null  float64
```

```
 19  time_signature    586672 non-null  int64
dtypes: float64(9), int64(6), object(5)
memory usage: 89.5+ MB
```

In [5]:
```
#nullvalues
pd.isnull(df_tracks).sum()
```

Out[5]:
```
id                  0
name               71
popularity          0
duration_ms         0
explicit            0
artists             0
id_artists          0
release_date        0
danceability        0
energy              0
key                 0
loudness            0
mode                0
speechiness         0
acousticness        0
instrumentalness    0
liveness            0
valence             0
tempo               0
time_signature      0
dtype: int64
```

In [6]:
```
df_tracks.describe().transpose()
```

Out[6]:

	count	mean	std	min	25%	50%	7
popularity	586672.0	27.570053	18.370642	0.0	13.0000	27.000000	41.00
duration_ms	586672.0	230051.167286	126526.087418	3344.0	175093.0000	214893.000000	263867.00
explicit	586672.0	0.044086	0.205286	0.0	0.0000	0.000000	0.00
danceability	586672.0	0.563594	0.166103	0.0	0.4530	0.577000	0.68
energy	586672.0	0.542036	0.251923	0.0	0.3430	0.549000	0.74
key	586672.0	5.221603	3.519423	0.0	2.0000	5.000000	8.00
loudness	586672.0	-10.206067	5.089328	-60.0	-12.8910	-9.243000	-6.48
mode	586672.0	0.658797	0.474114	0.0	0.0000	1.000000	1.00
speechiness	586672.0	0.104864	0.179893	0.0	0.0340	0.044300	0.07
acousticness	586672.0	0.449863	0.348837	0.0	0.0969	0.422000	0.78
instrumentalness	586672.0	0.113451	0.266868	0.0	0.0000	0.000024	0.00
liveness	586672.0	0.213935	0.184326	0.0	0.0983	0.139000	0.27
valence	586672.0	0.552292	0.257671	0.0	0.3460	0.564000	0.76
tempo	586672.0	118.464857	29.764108	0.0	95.6000	117.384000	136.32
time_signature	586672.0	3.873382	0.473162	0.0	4.0000	4.000000	4.00

In [7]:
```
most_popular=df_tracks.query('popularity>90',inplace=False).sort_values('popularity',as
most_popular.head()
```

Out[7]:

	id	name	popularity	duration_ms	explicit	artists	
93802	4iJyoBOLtHqaGxP12qzhQI	Peaches (feat. Daniel Caesar & Giveon)	100	198082	1	['Justin Bieber', 'Daniel Caesar', 'Giveon']	['1uNFoZAHB '20wkVL
93803	7lPN2DXiMsVn7XUKtOW1CS	drivers license	99	242014	1	['Olivia Rodrigo']	['1McMsnEElTI
93804	3Ofmpyhv5UAQ70mENzB277	Astronaut In The Ocean	98	132780	0	['Masked Wolf']	['1uU7g3DNSb
92810	5QO79kh1waicV47BqGRL3g	Save Your Tears	97	215627	1	['The Weeknd']	['1Xyo4u8uXC1
92811	6tDDoYIxWvMLTdKpjFkc1B	telepatía	97	160191	0	['Kali Uchis']	['1U1el3k54V\

In [8]:
```
#max monthly listeners
df_listeners=df_tracks["artists"].value_counts()
df_listeners[:5]
```

Out[8]:
```
['Die drei ???']         3856
['TKKG Retro-Archiv']    2006
['Benjamin Blümchen']    1503
['Bibi Blocksberg']      1472
['Lata Mangeshkar']      1373
Name: artists, dtype: int64
```

In [9]:
```
#drop unwanted columns
df_tracks1=df_tracks.drop(["key","mode","explicit"],axis=1)
```

In [10]:
```
df_tracks1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 586672 entries, 0 to 586671
Data columns (total 17 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   id                586672 non-null  object
 1   name              586601 non-null  object
 2   popularity        586672 non-null  int64
 3   duration_ms       586672 non-null  int64
 4   artists           586672 non-null  object
 5   id_artists        586672 non-null  object
 6   release_date      586672 non-null  object
```

```
 7   danceability      586672 non-null  float64
 8   energy            586672 non-null  float64
 9   loudness          586672 non-null  float64
 10  speechiness       586672 non-null  float64
 11  acousticness      586672 non-null  float64
 12  instrumentalness  586672 non-null  float64
 13  liveness          586672 non-null  float64
 14  valence           586672 non-null  float64
 15  tempo             586672 non-null  float64
 16  time_signature    586672 non-null  int64
dtypes: float64(9), int64(3), object(5)
memory usage: 76.1+ MB
```

In [11]:

```
#apply correlation
corr_df=df_tracks1.corr(method="pearson")
```

Correlation is an indication about the changes between two variables

The interpretation of the Pearson's correlation coefficient is as follows:-

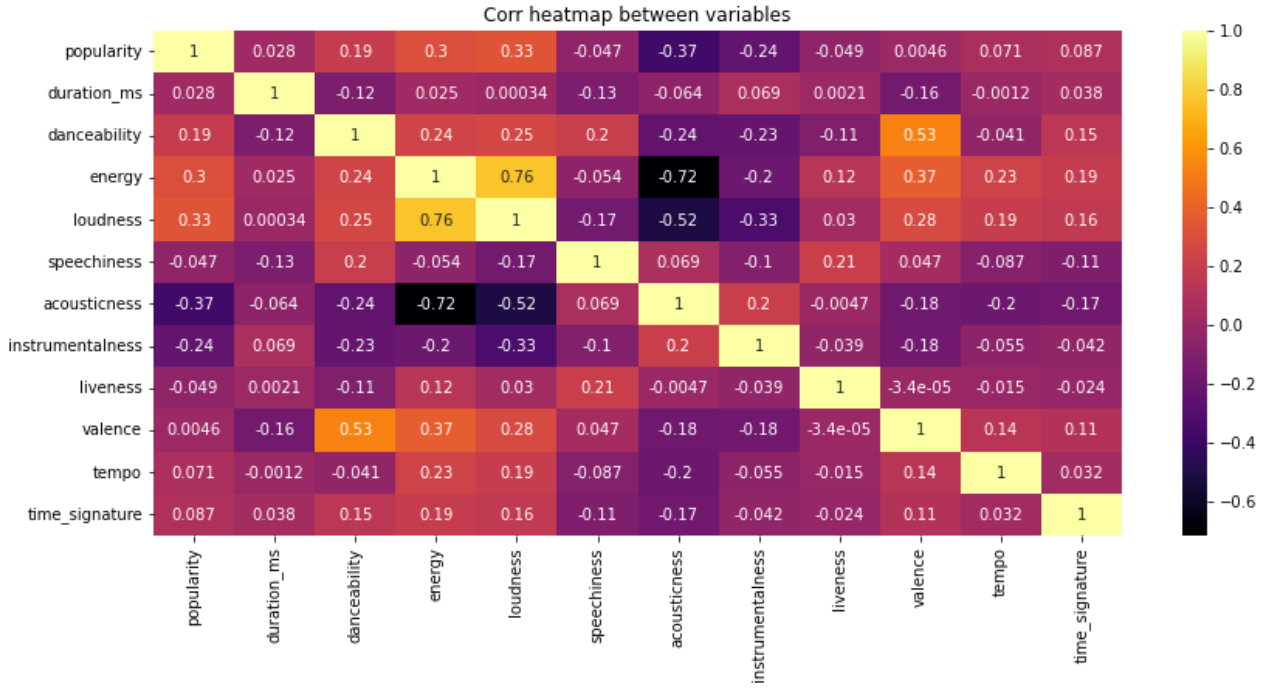
A correlation coefficient of 1 means there is a positive increase of a fixed proportion of others, for every positive increase in one variable. Like, the size of the shoe goes up in perfect correlation with foot length. If the correlation coefficient is 0, it indicates that there is no relationship between the variables. A correlation coefficient of -1 means there is a negative decrease of a fixed proportion, for every positive increase in one variable. Like, the amount of water in a tank will decrease in a perfect correlation with the flow of a water tap.

In [12]:

```
plt.figure(figsize=(14,6))
heatmap=sns.heatmap(corr_df,annot=True,cmap="inferno")
heatmap.set_title("Corr heatmap between variables")
heatmap.set_xticklabels(heatmap.get_xticklabels(),rotation=90)
```

Out[12]:

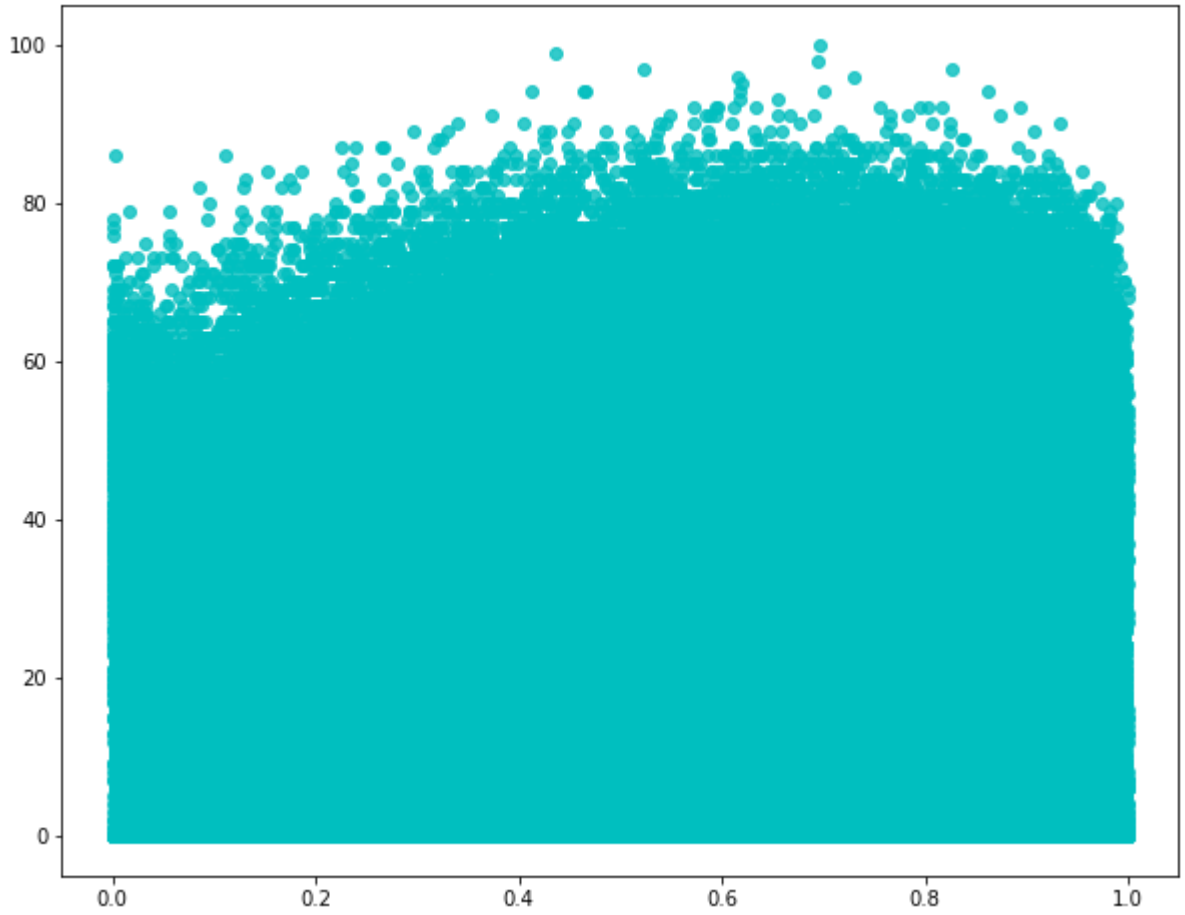
```
[Text(0.5, 0, 'popularity'),
 Text(1.5, 0, 'duration_ms'),
 Text(2.5, 0, 'danceability'),
 Text(3.5, 0, 'energy'),
 Text(4.5, 0, 'loudness'),
 Text(5.5, 0, 'speechiness'),
 Text(6.5, 0, 'acousticness'),
 Text(7.5, 0, 'instrumentalness'),
 Text(8.5, 0, 'liveness'),
 Text(9.5, 0, 'valence'),
 Text(10.5, 0, 'tempo'),
 Text(11.5, 0, 'time_signature')]
```



In [13]:
```
plt.figure(figsize=(10,8))
sns.regplot(data=df_tracks1,y="popularity",x="energy",color="c").set(title="Loudness vs
```

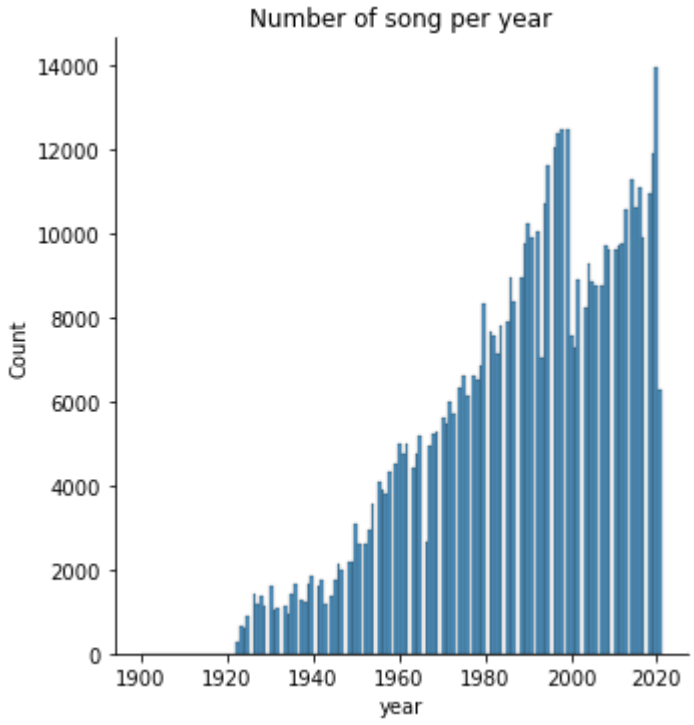
```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_9104/167242586.py in <module>
      1 plt.figure(figsize=(10,8))
----> 2 sns.regplot(data=df_tracks1,y="popularity",x="energy",color="c").set(title="Lou
dness vs Energy corr")

~\Anaconda3\lib\site-packages\seaborn\_decorators.py in inner_f(*args, **kwargs)
     44             )
     45         kwargs.update({k: arg for k, arg in zip(sig.parameters, args)})
---> 46         return f(**kwargs)
     47     return inner_f
     48

~\Anaconda3\lib\site-packages\seaborn\regression.py in regplot(x, y, data, x_estimator,
 x_bins, x_ci, scatter, fit_reg, ci, n_boot, units, seed, order, logistic, lowess, robus
t, logx, x_partial, y_partial, truncate, dropna, x_jitter, y_jitter, label, color, marke
r, scatter_kws, line_kws, ax)
    861     scatter_kws["marker"] = marker
    862     line_kws = {} if line_kws is None else copy.copy(line_kws)
--> 863     plotter.plot(ax, scatter_kws, line_kws)
    864     return ax
    865

~\Anaconda3\lib\site-packages\seaborn\regression.py in plot(self, ax, scatter_kws, line_
kws)
    368
    369         if self.fit_reg:
--> 370             self.lineplot(ax, line_kws)
    371
    372         # Label the axes

~\Anaconda3\lib\site-packages\seaborn\regression.py in lineplot(self, ax, kws)
```

```
    411         """Draw the model."""
    412         # Fit the regression model
--> 413         grid, yhat, err_bands = self.fit_regression(ax)
    414         edges = grid[0], grid[-1]
    415

~\Anaconda3\lib\site-packages\seaborn\regression.py in fit_regression(self, ax, x_range,
grid)
    219             yhat, yhat_boots = self.fit_logx(grid)
    220         else:
--> 221             yhat, yhat_boots = self.fit_fast(grid)
    222
    223         # Compute the confidence interval at each grid point

~\Anaconda3\lib\site-packages\seaborn\regression.py in fit_fast(self, grid)
    240             return yhat, None
    241
--> 242         beta_boots = algo.bootstrap(X, y,
    243                                     func=reg_func,
    244                                     n_boot=self.n_boot,

~\Anaconda3\lib\site-packages\seaborn\algorithms.py in bootstrap(*args, **kwargs)
     82     for i in range(int(n_boot)):
     83         resampler = integers(0, n, n, dtype=np.intp)  # intp is indexing dtype
---> 84         sample = [a.take(resampler, axis=0) for a in args]
     85         boot_dist.append(f(*sample, **func_kwargs))
     86     return np.array(boot_dist)

~\Anaconda3\lib\site-packages\seaborn\algorithms.py in <listcomp>(.0)
     82     for i in range(int(n_boot)):
     83         resampler = integers(0, n, n, dtype=np.intp)  # intp is indexing dtype
---> 84         sample = [a.take(resampler, axis=0) for a in args]
     85         boot_dist.append(f(*sample, **func_kwargs))
     86     return np.array(boot_dist)

KeyboardInterrupt:
```



In [18]:
```
df_tracks["release_date"]=pd.to_datetime(df_tracks["release_date"])
df_tracks["year"]=df_tracks["release_date"].dt.year

sns.displot(df_tracks["year"]).set(title="Number of song per year")
```

Out[18]: <seaborn.axisgrid.FacetGrid at 0x158ebc31ee0>



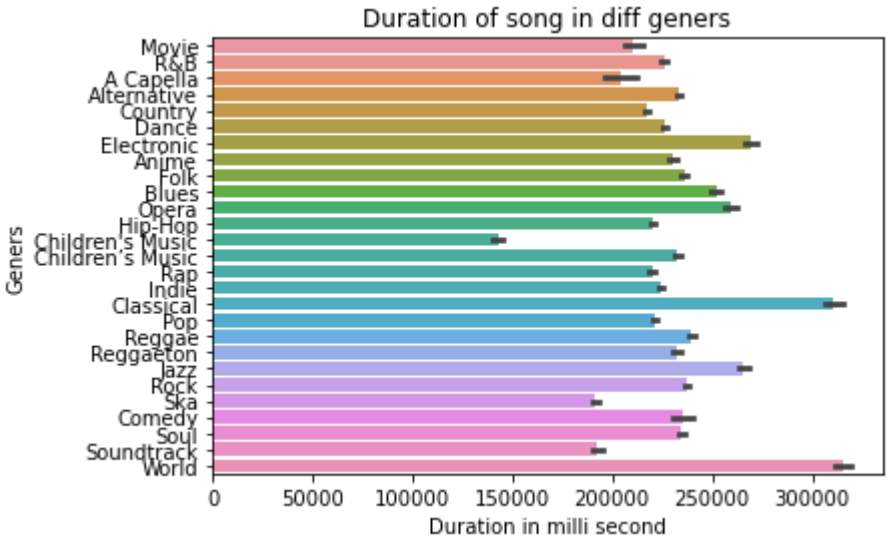
In [20]:
```
df=pd.read_csv("SpotifyFeatures.csv")
df.head()
```

Out[20]:

	genre	artist_name	track_name	track_id	popularity	acousticness	danceability	du
0	Movie	Henri Salvador	C'est beau de faire un Show	0BRjO6ga9RKCKjfDqeFgWV	0	0.611	0.389	
1	Movie	Martin & les fées	Perdu d'avance (par Gad Elmaleh)	0BjC1NfoEOOusryehmNudP	1	0.246	0.590	
2	Movie	Joseph Williams	Don't Let Me Be Lonely Tonight	0CoSDzoNIKCRs124s9uTVy	3	0.952	0.663	
3	Movie	Henri Salvador	Dis-moi Monsieur Gordon Cooper	0Gc6TVm52BwZD07Ki6tIvf	0	0.703	0.240	
4	Movie	Fabien Nataf	Ouverture	0IuslXpMROHdEPvSl1fTQK	4	0.950	0.331	

In [21]:
```
#finding out duration of different songs
plt.title("Duration of song in diff geners")
sns.color_palette("rocket",as_cmap=True)
sns.barplot(y=df["genre"],x=df["duration_ms"],data=df)
plt.xlabel("Duration in milli second")
plt.ylabel("Geners")
```

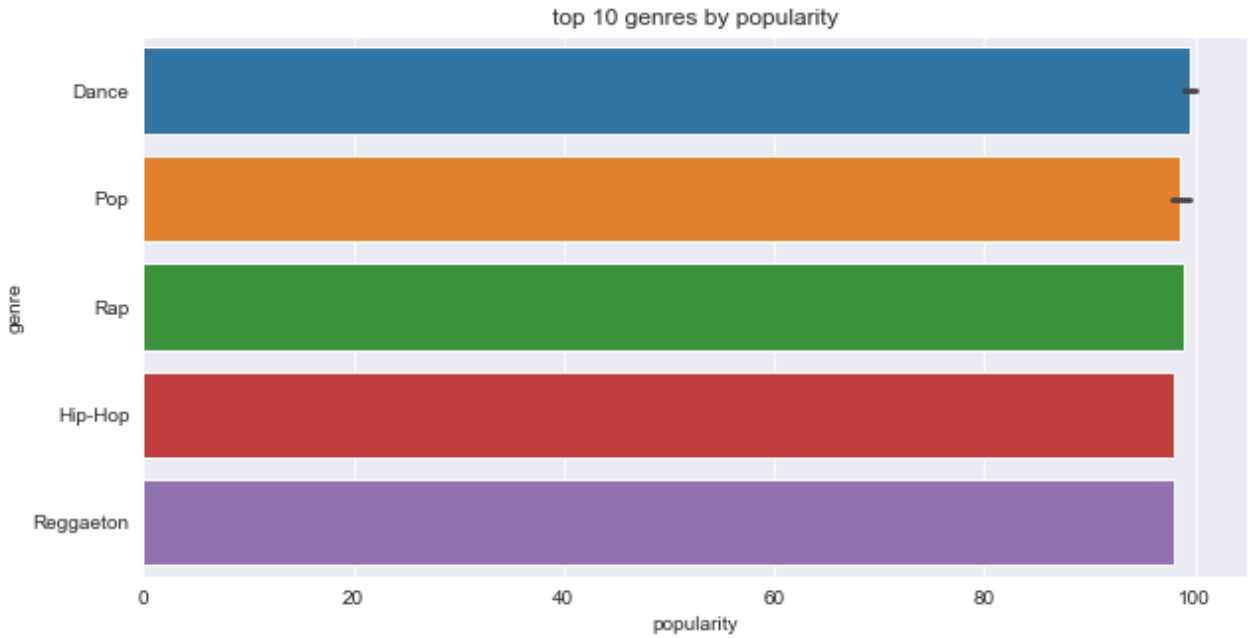
Out[21]:
```
Text(0, 0.5, 'Geners')
```



In [22]:
```
#top 10 genres by popularity
```

```
sns.set_style(style="darkgrid")
plt.figure(figsize=(10,5))
famous=df.sort_values("popularity",ascending=False).head(10)
sns.barplot(y="genre",x="popularity",data=famous).set(title="top 10 genres by popularit
```

Out[22]: [Text(0.5, 1.0, 'top 10 genres by popularity')]



In [ ]: