Spotify Data Analysis

Importing all the required libraries

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

Loading the tracks.csv file as df_tracks dataframe

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

Out[2]:

	id	name	popularity	duration_ms	explicit	artists	
0	35iwgR4jXetI318WEWsa1Q	Carve	6	126903	0	['Uli']	['45tIt06XoI
1	021ht4sdgPcrDgSk7JTbKY	Capítulo 2.16 - Banquero Anarquista	0	98200	0	['Fernando Pessoa']	['14jtPCOoNZwq
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 [3]:
```
# Checking for null values

pd.isnull(df_tracks).sum()
```

Out[3]:
```
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 [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
```

5 Least popular songs on Spotify

In [5]:
```
sorted_df = df_tracks.sort_values("popularity", ascending = True).head(5)
sorted_df
```

Out[5]:

	id	name	popularity	duration_ms	explicit	artists	
546130	181rTRhCcggZPwP2TUcVqm	Newspaper Reports On Abner, 20 February 1935	0	896575	0	['Norris Goff', 'Chester Lauck', 'Carlton Bric...	['3WCwCPDMp '7vk8U
546222	0yOCz3V5KMm8l1T8EFc60i	恋は水の上で	0	188440	0	['Hibari Misora']	['1m5pMY5bl
546221	0y48Hhwe52099UqYjegRCO	私の誕生日	0	173467	0	['Hibari Misora']	['1m5pMY5bl
546220	0xCmgtf9ka07hkZg3D6PaV	エル・チョクロ (EL CHOCLO)	0	205280	0	['Hibari Misora']	['1m5pMY5bl
546219	0tBXS3VuCPX7KWUFH2nros	恋は不思議なもの	0	185733	0	['Hibari Misora']	['1m5pMY5bl

Now, lets see some descriptive statistics for the numerical variables in the dataset in hand

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

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

5 Most popular songs on Spotify

In [7]:
```
most_popular = df_tracks.query("popularity > 90", inplace = False).sort_values(by = "pop
most_popular[ : 5]
```

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('20wkVLu
93803	7lPN2DXiMsVn7XUKtOW1CS	drivers license	99	242014	1	['Olivia Rodrigo']	['1McMsnEElTh
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']	['1U1el3k54Vv

Setting release_date column as index and changing it's dtype to datetime

In [8]:
```
df_tracks.set_index("release_date", inplace = True)
df_tracks.index = pd.to_datetime(df_tracks.index)
df_tracks.head()
```

Out[8]:

	id	name	popularity	duration_ms	explicit	artists	
release_date							
1922-02-22	35iwgR4jXetI318WEWsa1Q	Carve	6	126903	0	['Uli']	
1922-06-01	021ht4sdgPcrDgSk7JTbKY	Capítulo 2.16 - Banquero Anarquista	0	98200	0	['Fernando Pessoa']	['14jt
1922-03-21	07A5yehtSnoedViJAZkNnc	Vivo para Quererte - Remasterizado	0	181640	0	['Ignacio Corsini']	['5LiC
1922-03-21	08FmqUhxtyLTn6pAh6bk45	El Prisionero - Remasterizado	0	176907	0	['Ignacio Corsini']	['5LiC
1922-01-01	08y9GfoqCWfOGsKdwojr5e	Lady of the Evening	0	163080	0	['Dick Haymes']	['3

Changing the duration from ms to seconds and storing it in column named duration and dropping the duration_ms column

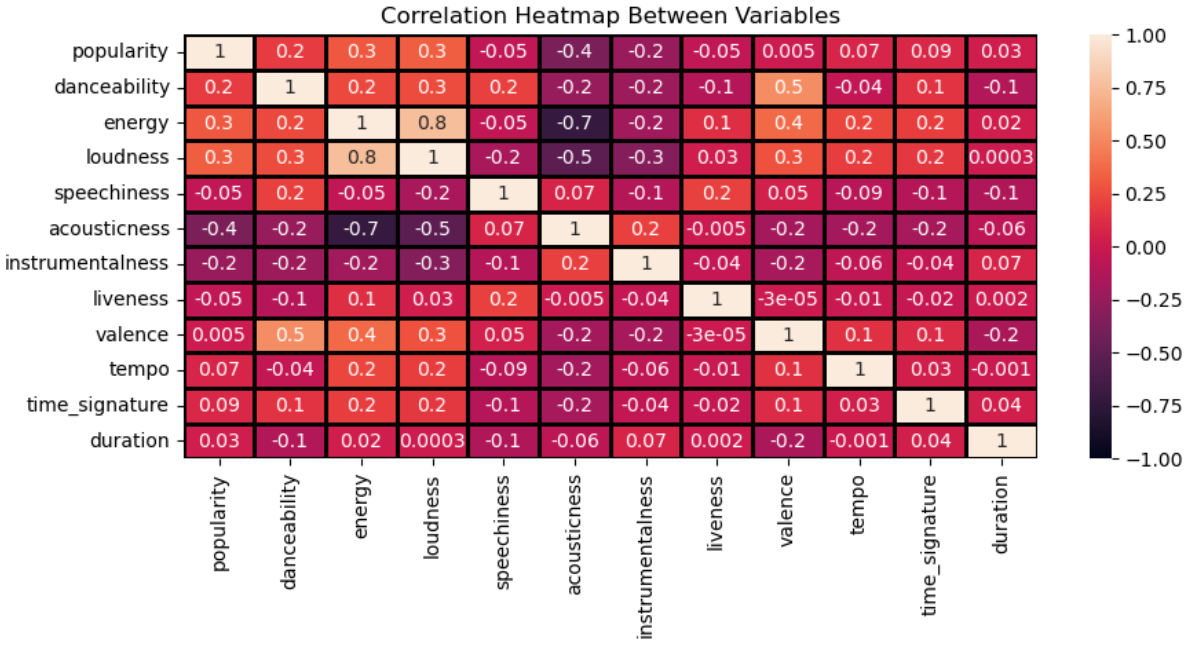
In [9]:
```
df_tracks["duration"] = df_tracks["duration_ms"].apply(lambda t : round(t / 1000))
df_tracks.drop("duration_ms", inplace =True, axis = 1)
df_tracks.duration.head()
```

Out[9]:
```
release_date
1922-02-22    127
1922-06-01     98
1922-03-21    182
1922-03-21    177
1922-01-01    163
Name: duration, dtype: int64
```

Correlation map between variables

We also drop three unwanted rows named key, mode and explicit

In [10]:
```
corr_df = df_tracks.drop(["key", "mode", "explicit"], axis=1).corr(method="pearson", num
plt.figure(figsize = (10,4))
heatmap = sns.heatmap(corr_df, annot = True, fmt = ".1g", vmin = -1, vmax = 1,
                      center = 0, cmap = "rocket", linewidths = 1, linecolor = "Black")
heatmap.set_title("Correlation Heatmap Between Variables")
heatmap.set_xticklabels(heatmap.get_xticklabels(), rotation = 90)
plt.show()
```



Creating a sample data which is 10% of the original data

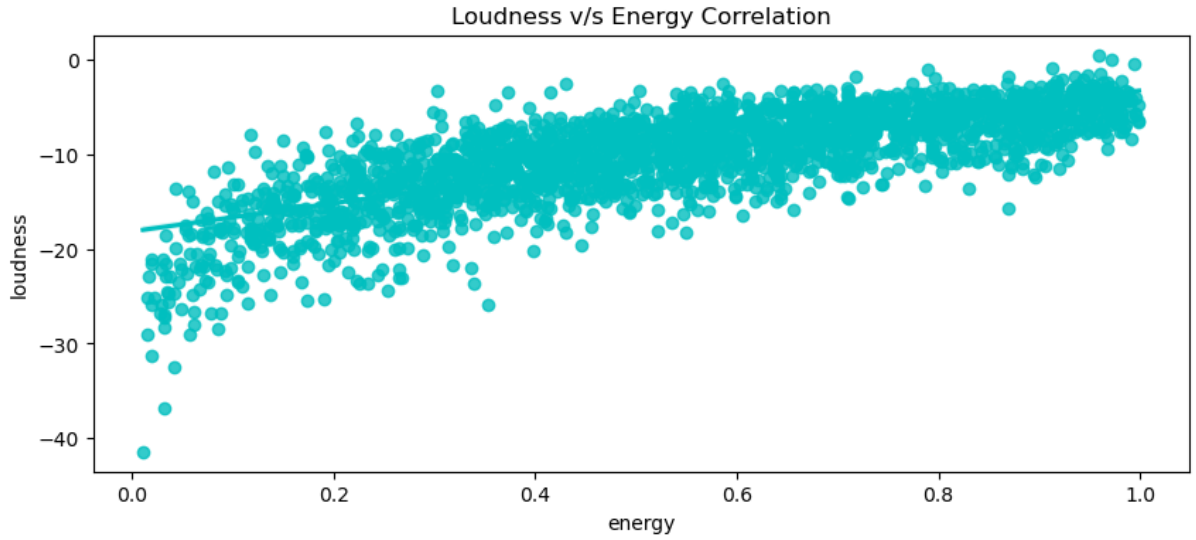
In [11]:
```
sample_df = df_tracks.sample(int(0.004 * len(df_tracks)))
```

In [12]:
```
print(len(sample_df))
```

```
2346
```

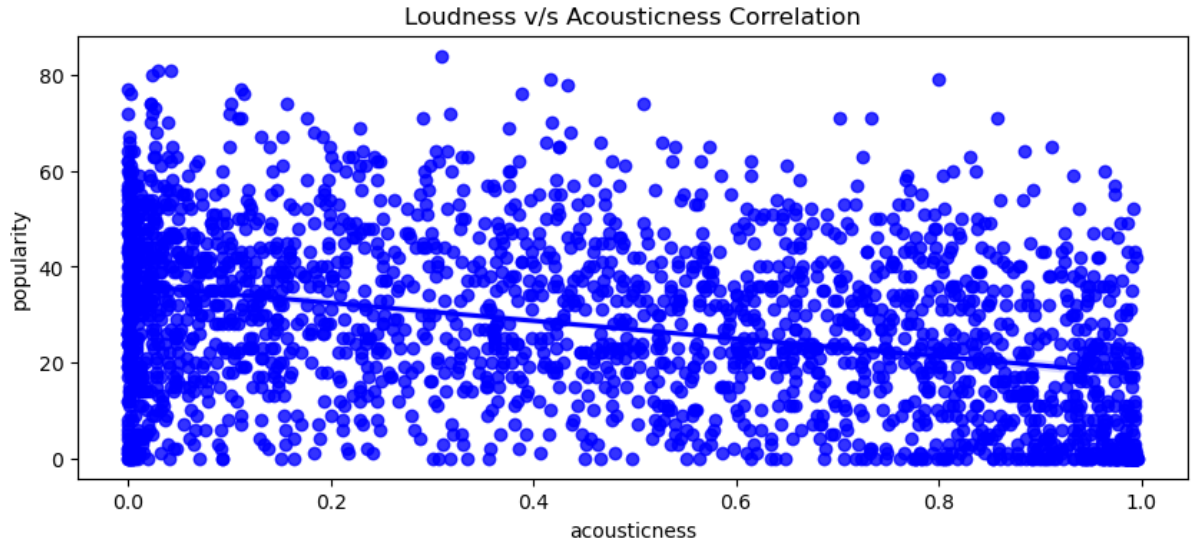
Regression plot between loudness and energy

In [13]:
```
plt.figure(figsize = (10,4))
sns.regplot(x = "energy", y = "loudness", data = sample_df, color = "c").set(title = "Lo
plt.show()
```



Regression plot between popularity and acousticness

In [14]:
```
plt.figure(figsize = (10,4))
sns.regplot(x = "acousticness", y = "popularity", data = sample_df, color = "b").set(tit
plt.show()
```



Extracting year from release_date column

In [15]:
```
df_tracks["dates"] = df_tracks.index.get_level_values("release_date")
df_tracks.dates = pd.to_datetime(df_tracks.dates)
years = df_tracks.dates.dt.year
```

In [16]:
```
df_tracks.dates.head()
```

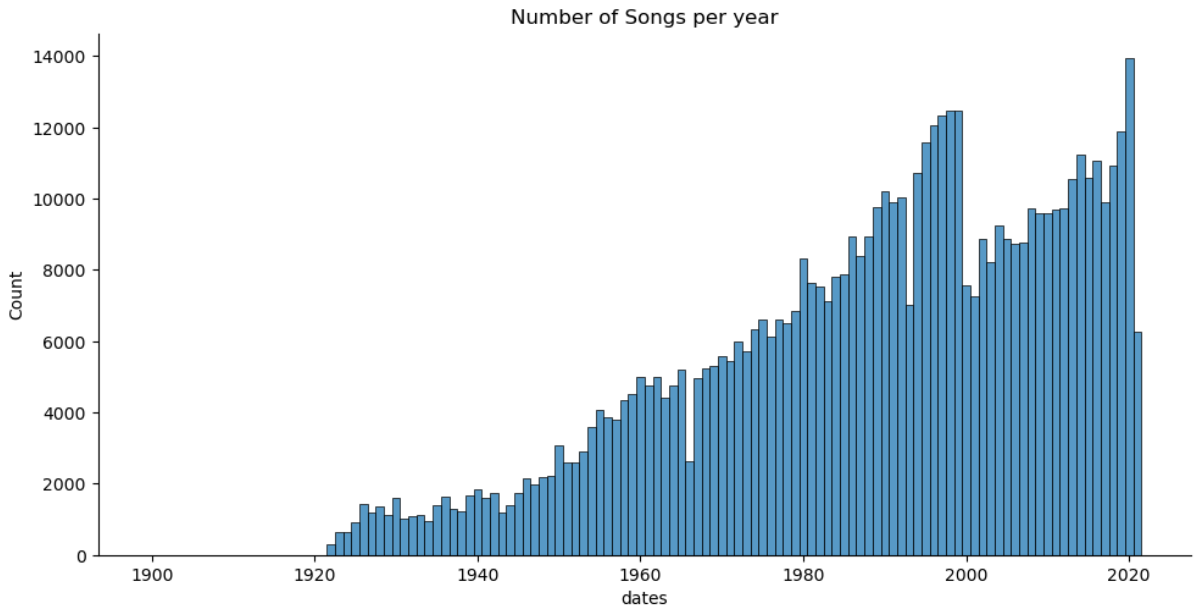
Out[16]:
```
release_date
1922-02-22   1922-02-22
1922-06-01   1922-06-01
1922-03-21   1922-03-21
1922-03-21   1922-03-21
1922-01-01   1922-01-01
Name: dates, dtype: datetime64[ns]
```

In [17]:
```
years.head()
```

Out[17]:
```
release_date
1922-02-22    1922
1922-06-01    1922
1922-03-21    1922
1922-03-21    1922
1922-01-01    1922
Name: dates, dtype: int64
```

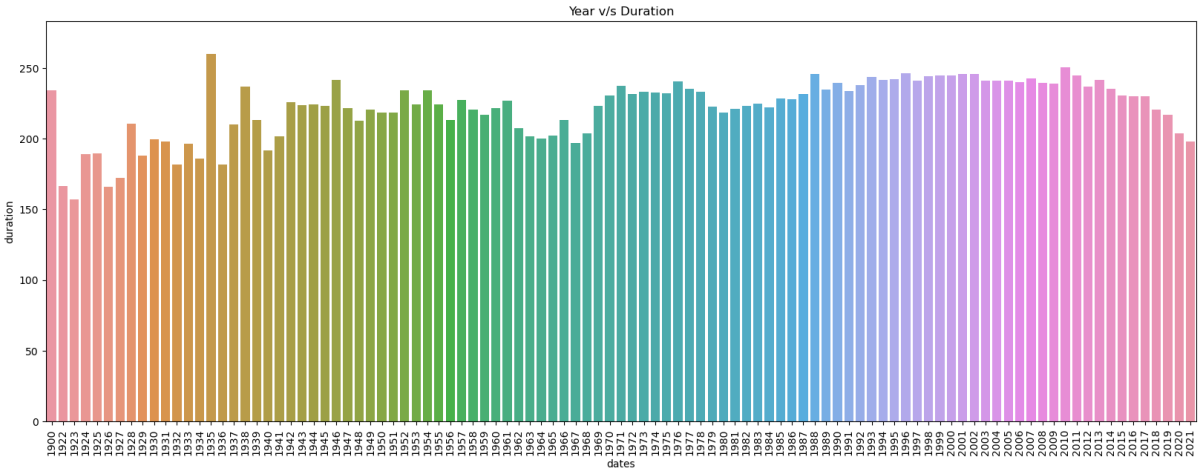
Distribution chart to show number of songs in each year

In [18]:
```
sns.displot(years, discrete = True, aspect = 2, height = 5, kind = "hist").set(title = "
plt.show()
```



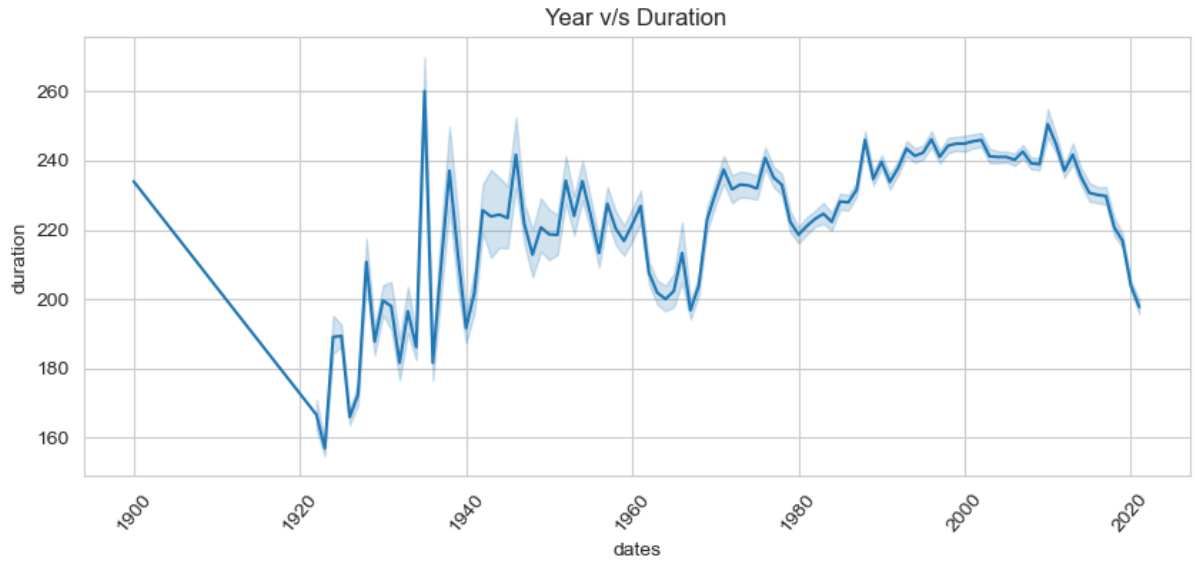
Distribution chart to show song length trend in each year

In [19]:
```
total_dr = df_tracks.duration
fig_dims = (20,7)
fig, ax = plt.subplots(figsize = fig_dims)
fig = sns.barplot(x = years, y = total_dr, ax = ax, errwidth = False).set(title = "Year
plt.xticks(rotation = 90)
plt.show()
```



Line plot to see the average duration of songs over the years

In [20]:
```
total_dr = df_tracks.duration
sns.set_style(style = "whitegrid")
fig_dims = (10,4)
fig, ax = plt.subplots(figsize = fig_dims)
fig = sns.lineplot(x = years, y =total_dr, ax = ax).set(title = "Year v/s Duration")
plt.xticks(rotation = 50)
plt.show()
```



Importing the genre data### Loading the tracks.csv file as df_tracks dataframe

In [21]:
```
df_genre = pd.read_csv("SpotifyFeatures.csv")
df_genre.head()
```

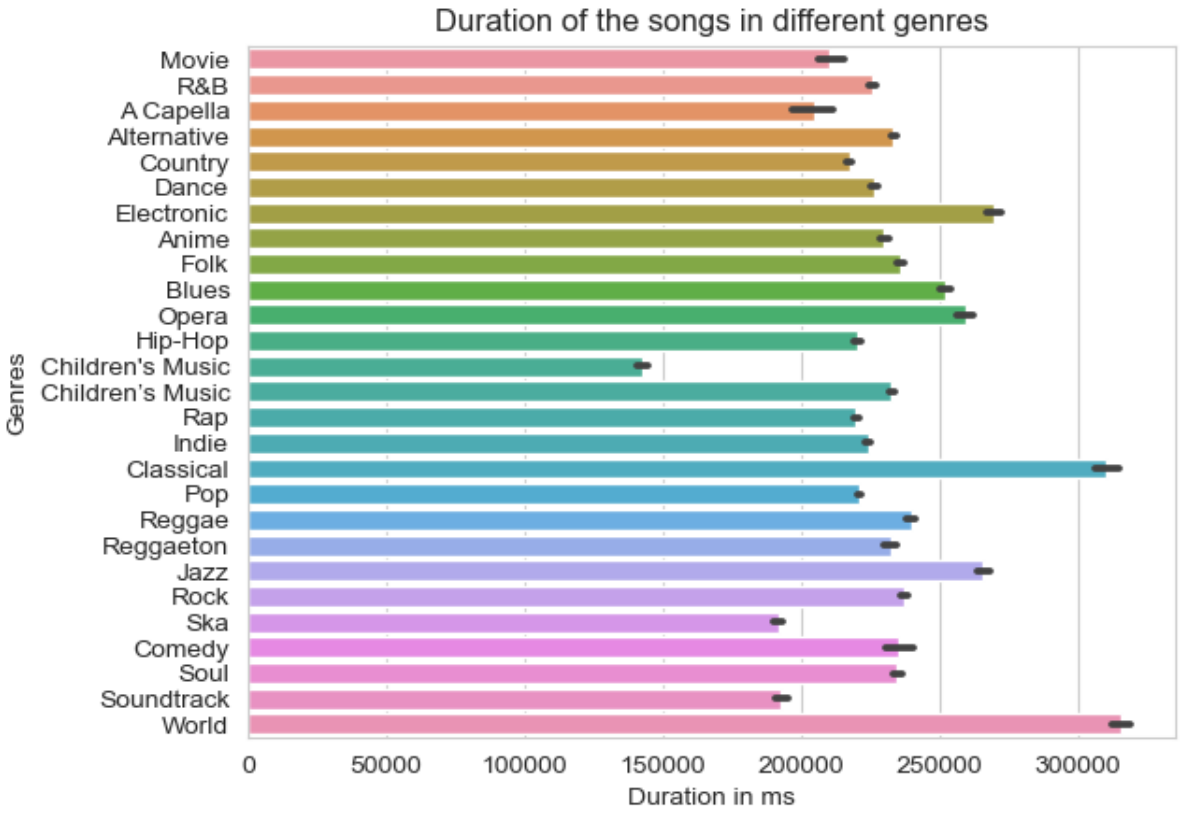
Out[21]:

	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	

Bar plot to show total duration of songs as per different genres

In [22]:
```
plt.title("Duration of the songs in different genres")
sns.color_palette("rocket", as_cmap = True)
sns.barplot(x = "duration_ms", y = "genre", data = df_genre)
plt.xlabel("Duration in ms")
plt.ylabel("Genres")
```

Out[22]:
```
Text(0, 0.5, 'Genres')
```



Top 5 genres by popularity

In [23]:
```
sns.set_style(style = "darkgrid")
plt.figure(figsize = (10,4))
famous = df_genre.sort_values(by = "popularity", ascending =  False).head(10)
sns.barplot(x ="popularity", y = "genre", data = famous).set(title = "Top 5 Genres by Pc
plt.show()
```