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

Reading CSV into variable dataset

In [ ]:
```
df = pd.read_csv('/Users/fizz/Desktop/spotifynewclean.csv')
df.head()
```

Out[ ]:

	acousticness	artists	danceability	energy	explicit	instrumentalness	key	liveness	loudness	mode	name	popularity	spe
0	0.00146	The Weeknd	0.514	0.730	0	0.0001	1	0.0897	-5.934	1	Blinding Lights	100	
1	0.24700	DaBaby, Roddy Ricch	0.746	0.690	1	0.0000	11	0.1010	-7.956	1	ROCKSTAR (feat. Roddy Ricch)	99	
2	0.73100	Powfu, beabadoobee	0.726	0.431	0	0.0000	8	0.6960	-8.765	0	death bed (feat. beabadoobee)	97	
3	0.23300	THE SCOTTS, Travis Scott, Kid Cudi	0.716	0.537	1	0.0000	0	0.1570	-7.648	0	THE SCOTTS	96	
4	0.10400	Roddy Ricch	0.896	0.586	1	0.0000	10	0.7900	-6.687	0	The Box	95	

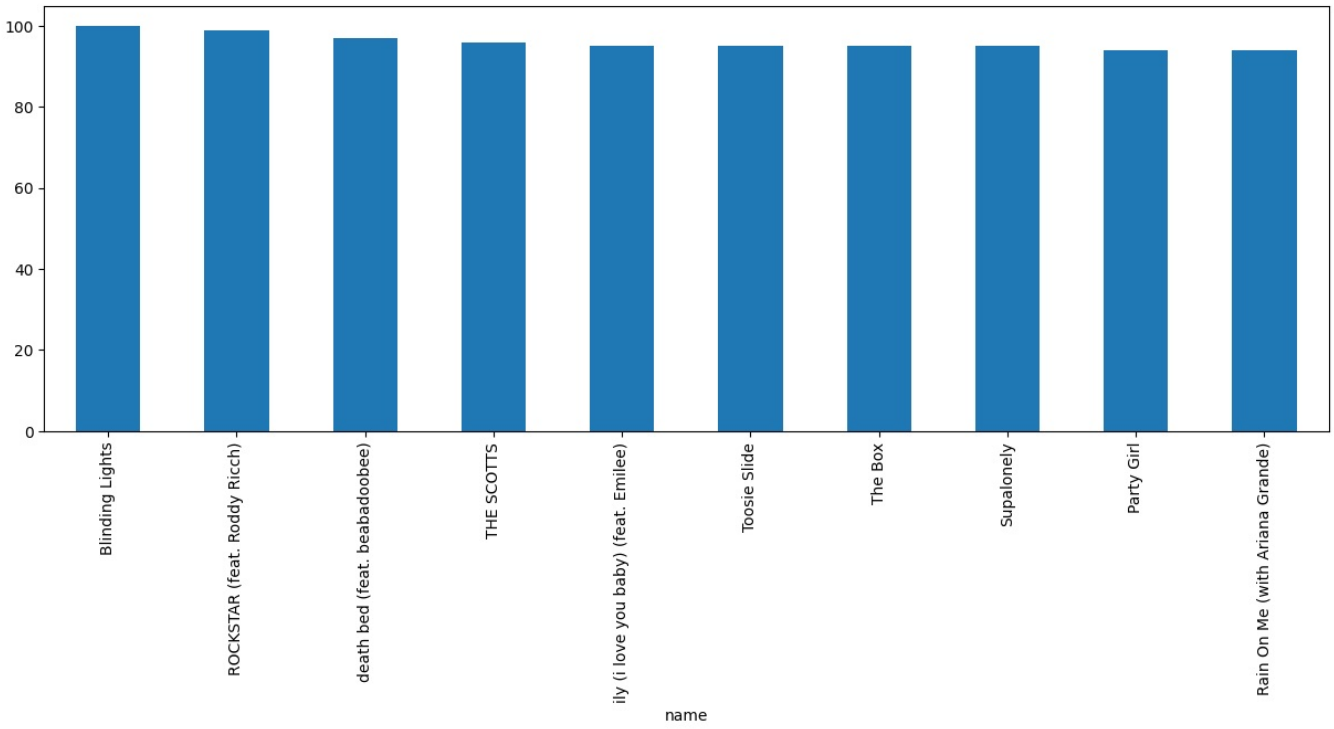
To know the datatpe of all the attributes in the dataset

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 130432 entries, 0 to 130431
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   acousticness      130432 non-null  float64
 1   artists           130432 non-null  object 
 2   danceability      130432 non-null  float64
 3   energy            130432 non-null  float64
 4   explicit          130432 non-null  int64  
 5   instrumentalness  130432 non-null  float64
 6   key               130432 non-null  int64  
 7   liveness          130432 non-null  float64
 8   loudness          130432 non-null  float64
 9   mode              130432 non-null  int64  
 10  name              130432 non-null  object 
 11  popularity        130432 non-null  int64  
 12  speechiness       130432 non-null  float64
 13  tempo             130432 non-null  float64
 14  valence           130432 non-null  float64
 15  year              130432 non-null  int64  
 16  duration_min      130432 non-null  float64
 17  Song Decade       130432 non-null  object 
dtypes: float64(10), int64(5), object(3)
memory usage: 17.9+ MB
```

In [ ]:
```
top_ten_tracks = df.groupby("name")['popularity'].mean().sort_values(ascending=False).head(10)
top_ten_tracks.head(10)

top_ten_tracks.plot(kind = 'bar', figsize = (15,5))
```

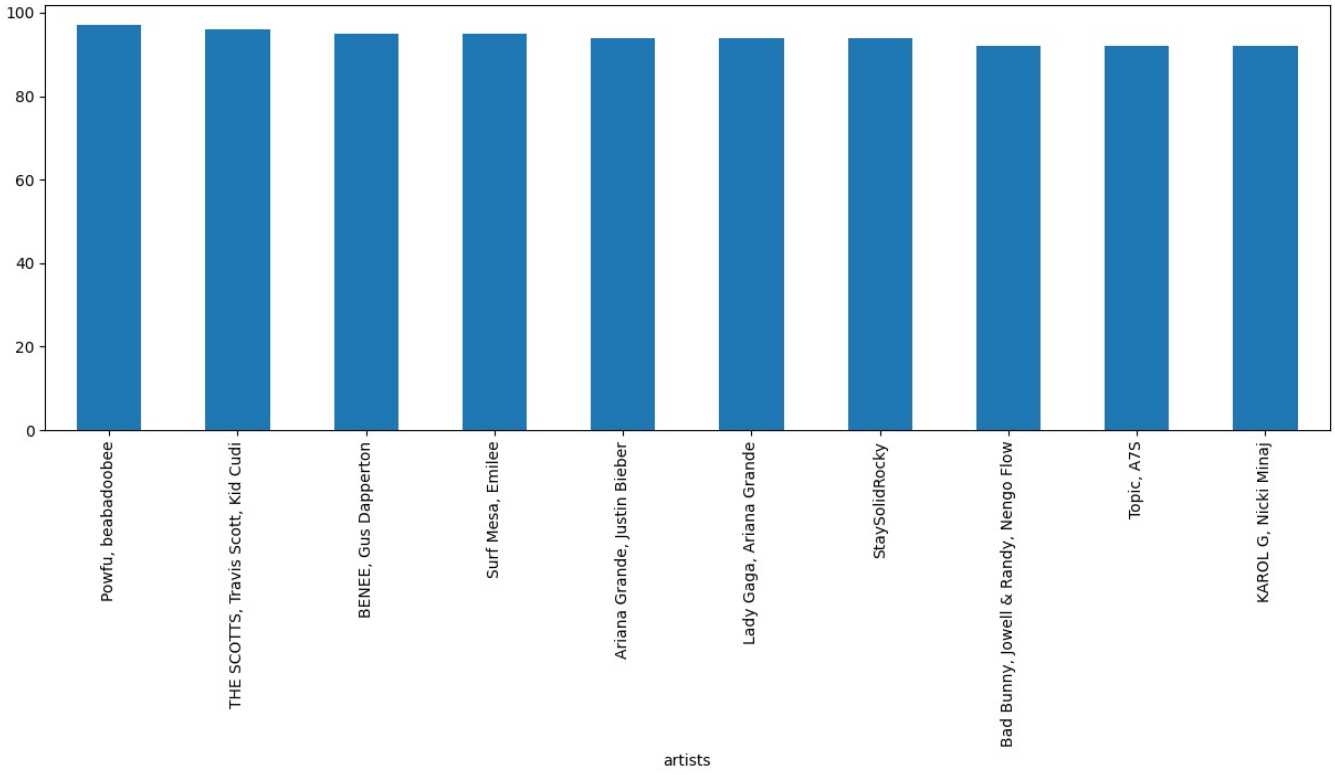
Out[ ]:
```
<AxesSubplot: xlabel='name'>
```



The above bargraph shows the top 10 songs based on popularity over the years (1920 to 2020)

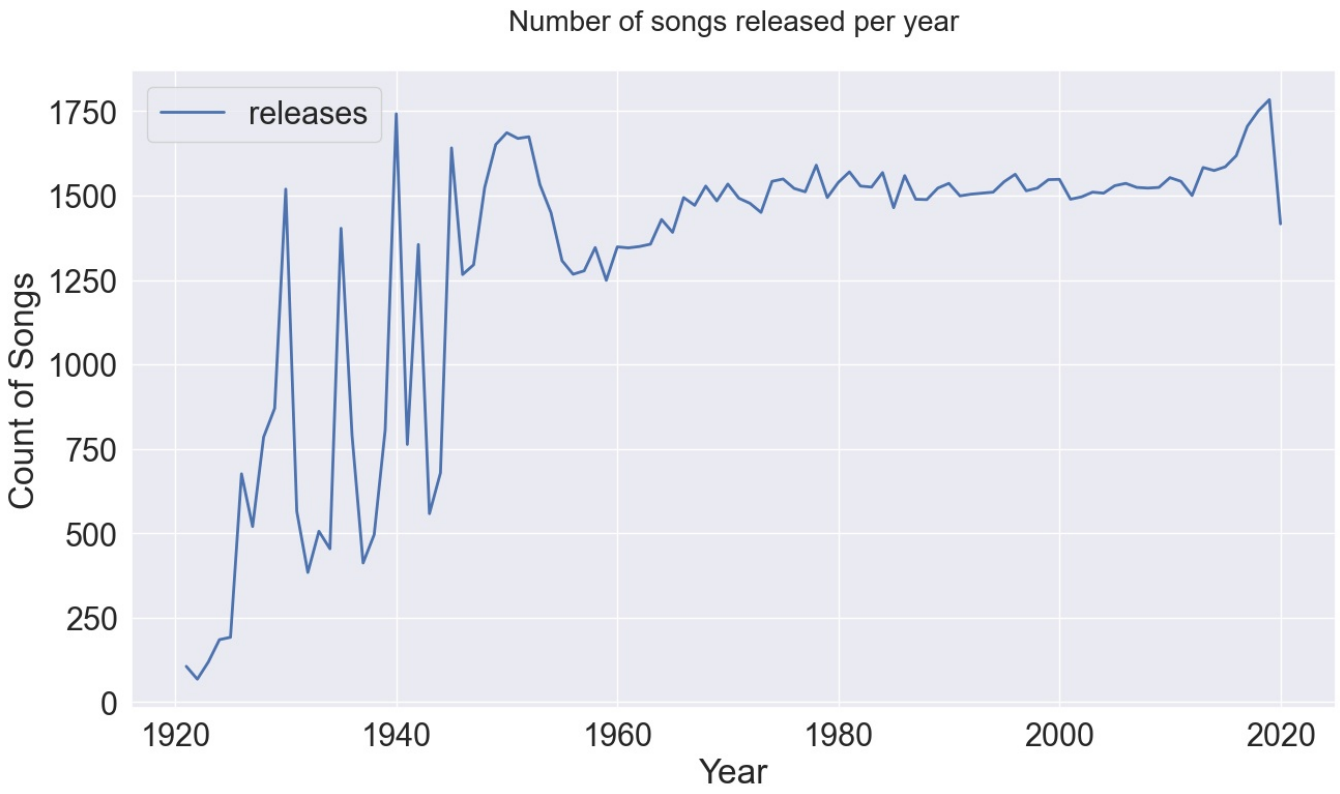
In [ ]:
```
top_ten_artists = df.groupby("artists")['popularity'].mean().sort_values(ascending=False).head(10)
top_ten_artists.head(10)

top_ten_artists.plot (kind = 'bar', figsize = (15,5))
```

Out[ ]:
```
<AxesSubplot: xlabel='artists'>
```



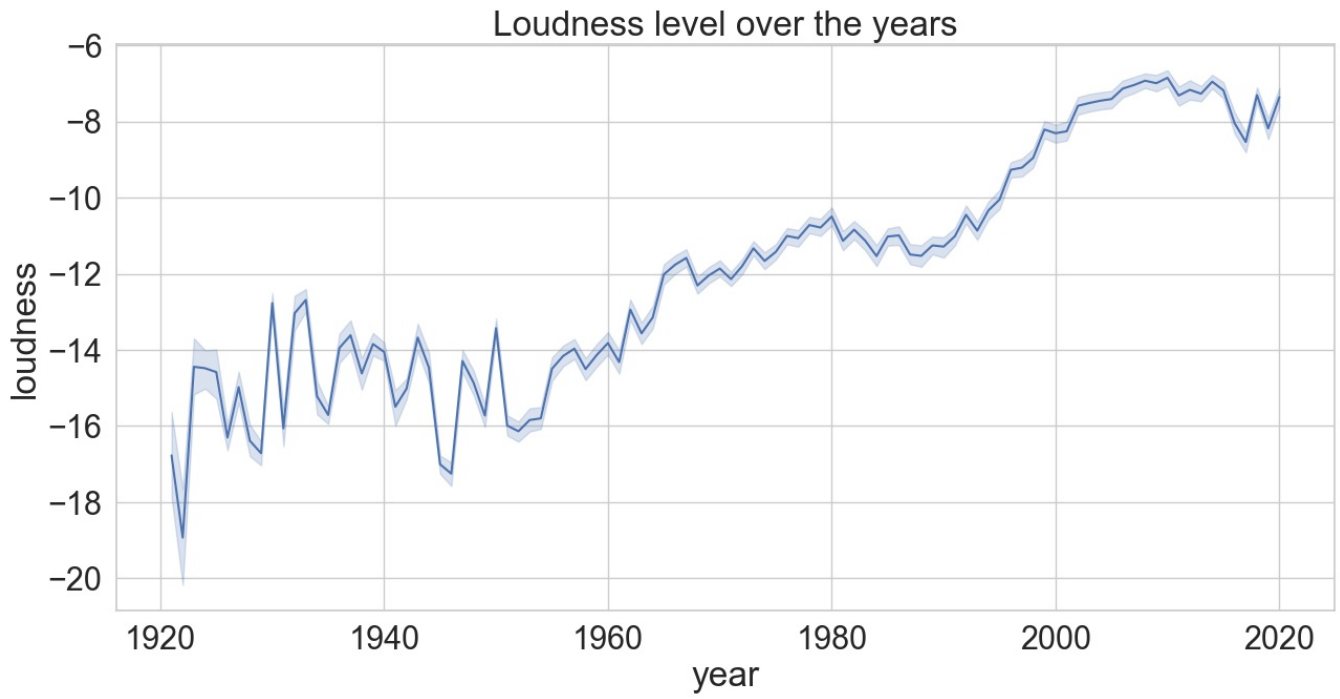
The above bargraph shows the top 10 artists based on popularity over the years (1920 to 2020)

In [ ]:
```
number_of_releases = pd.DataFrame(df['year'].value_counts())
number_of_releases.rename({'year': 'releases'}, axis=1, inplace=True)
number_of_releases = number_of_releases.sort_index()
sns.set(font_scale=2)

ax=number_of_releases.plot(kind='line',figsize=(15,8), linewidth=2)
plt.title("Number of songs released per year",y=1.05,fontsize=20)
plt.xlabel('Year')
plt.ylabel('Count of Songs')
ax.axes.get_xaxis().set_visible(True)
```



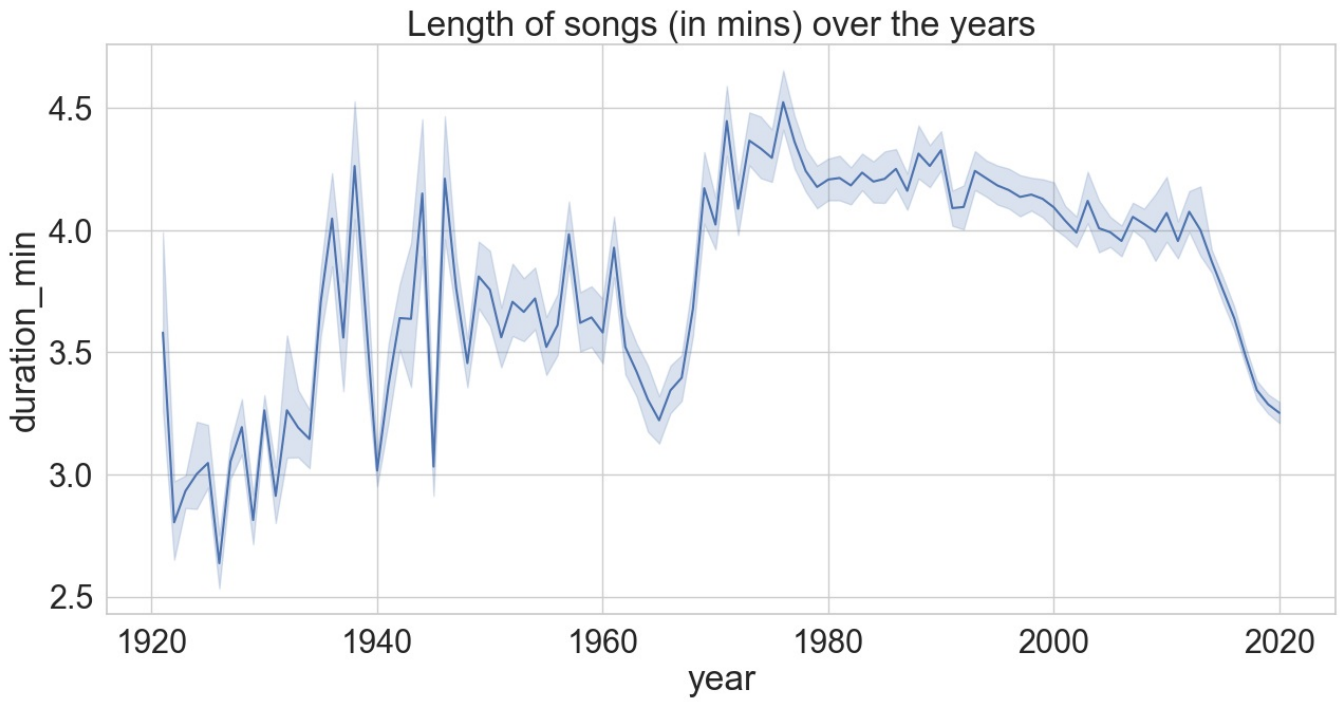
The above graph shows the number of songs released over the years (1029 to 2020)

In [ ]:
```
sns.set_style(style="whitegrid")
fig_dims=(15,7)
fig,ax=plt.subplots(figsize=fig_dims)
fig=sns.lineplot(x=df["year"],y=df["loudness"],ax=ax).set(title="Loudness level over the years")
```



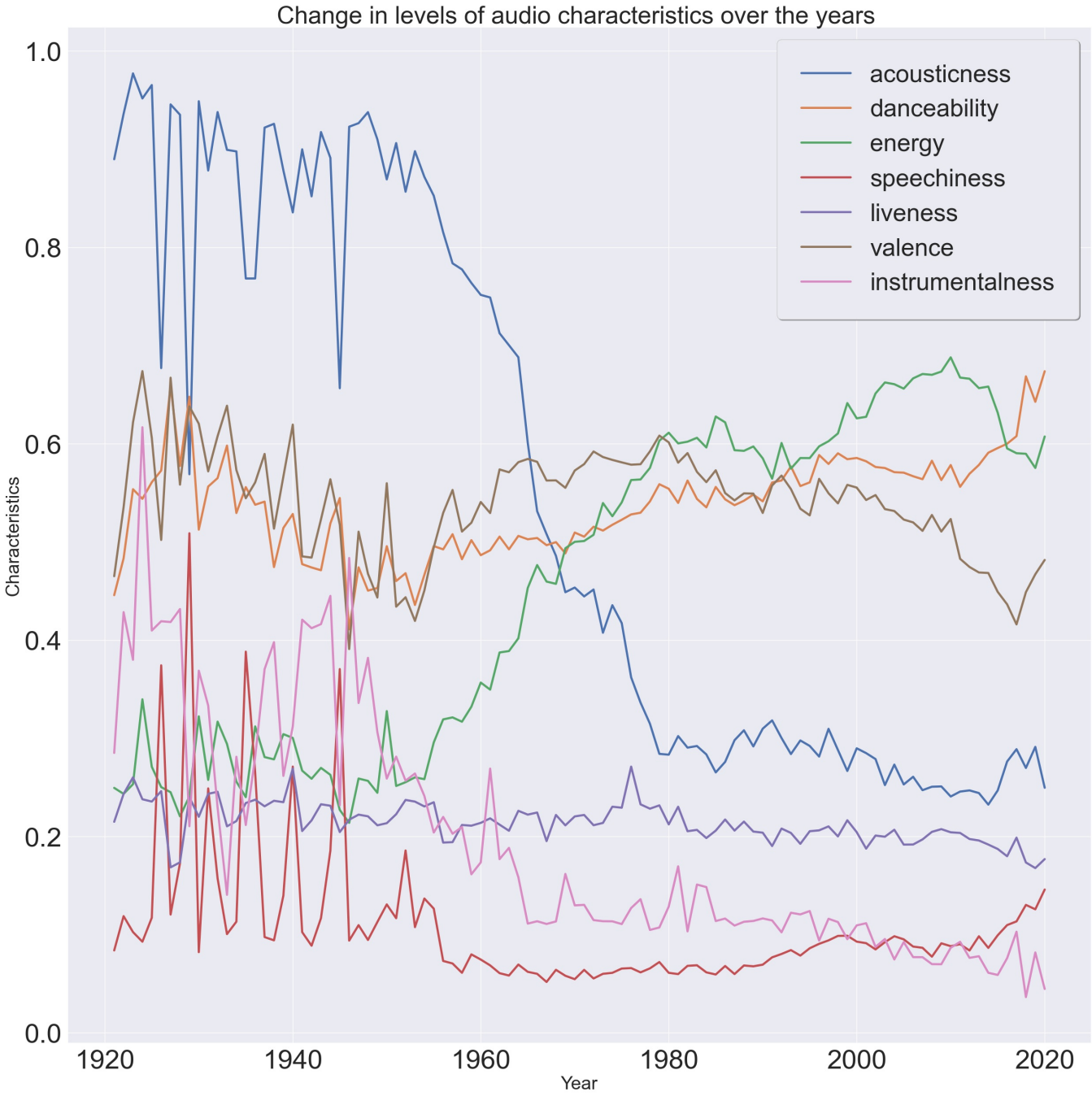
The above graph shows loudness level over the years. It can be noted that the loudness level has increased over the years.

In [ ]:
```
sns.set_style(style="whitegrid")
fig_dims=(15,7)
fig,ax=plt.subplots(figsize=fig_dims)
fig=sns.lineplot(x=df["year"],y=df["duration_min"],ax=ax).set(title="Length of songs (in mins) over the years")
```



The above graph shows length of songs (in mins) over the years (1920 to 2020). It can be noted that the lowest average length was between 1920 to 1940 and highest being 1970 to 1980.

In [ ]:
```
columns = ["acousticness","danceability","energy","speechiness","liveness","valence", "instrumentalness"]
plt.figure(figsize=(30,30))
sns.set(font_scale=4)
for c in columns:
    x = df.groupby('year')[c].mean()
    sns.lineplot(x,linewidth = 3.5,label=c)
plt.title('Change in levels of audio characteristics over the years ', fontsize=40)
plt.xlabel('Year',fontsize=30)
plt.ylabel('Characteristics',fontsize=30)
plt.legend(fancybox=True, framealpha=1, shadow=True, borderpad=1, prop={'size': 40}, loc = 'upper right')
plt.show()
```



The above graph shows the change in levels of the audio characteristics over the years.

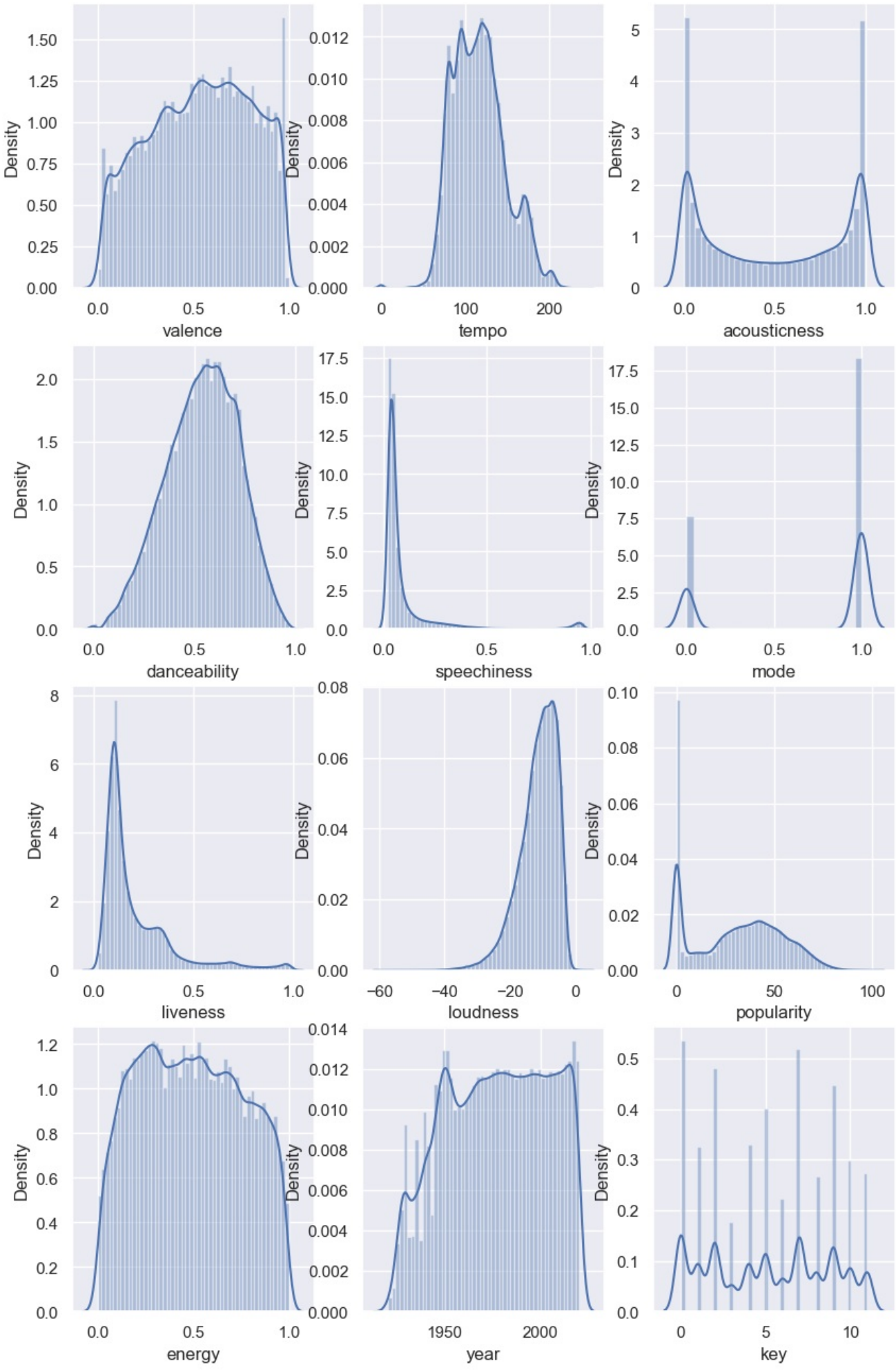
In [ ]:
```
fig,ax = plt.subplots(4,3,figsize=(10,16))
sns.set(font_scale = 1)
sns.distplot(df['valence'],ax=ax[0,0])
sns.distplot(df['tempo'],ax=ax[0,1])
sns.distplot(df['acousticness'],ax=ax[0,2])
sns.distplot(df['danceability'],ax=ax[1,0])
sns.distplot(df['speechiness'],ax=ax[1,1])
sns.distplot(df['mode'],ax=ax[1,2])
sns.distplot(df['liveness'],ax=ax[2,0])
sns.distplot(df['loudness'],ax=ax[2,1])
sns.distplot(df['popularity'],ax=ax[2,2])
sns.distplot(df['energy'],ax=ax[3,0])
sns.distplot(df['year'],ax=ax[3,1])
sns.distplot(df['key'],ax=ax[3,2])
```

```
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['valence'],ax=ax[0,0])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:4: UserWarning:
```

```
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['tempo'],ax=ax[0,1])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['acousticness'],ax=ax[0,2])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['danceability'],ax=ax[1,0])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:7: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['speechiness'],ax=ax[1,1])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:8: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['mode'],ax=ax[1,2])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:9: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['liveness'],ax=ax[2,0])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:10: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['loudness'],ax=ax[2,1])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:11: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
```

```
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['popularity'],ax=ax[2,2])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:12: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['energy'],ax=ax[3,0])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:13: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['year'],ax=ax[3,1])
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/1076098017.py:14: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['key'],ax=ax[3,2])
```

Out[ ]: <AxesSubplot: xlabel='key', ylabel='Density'>

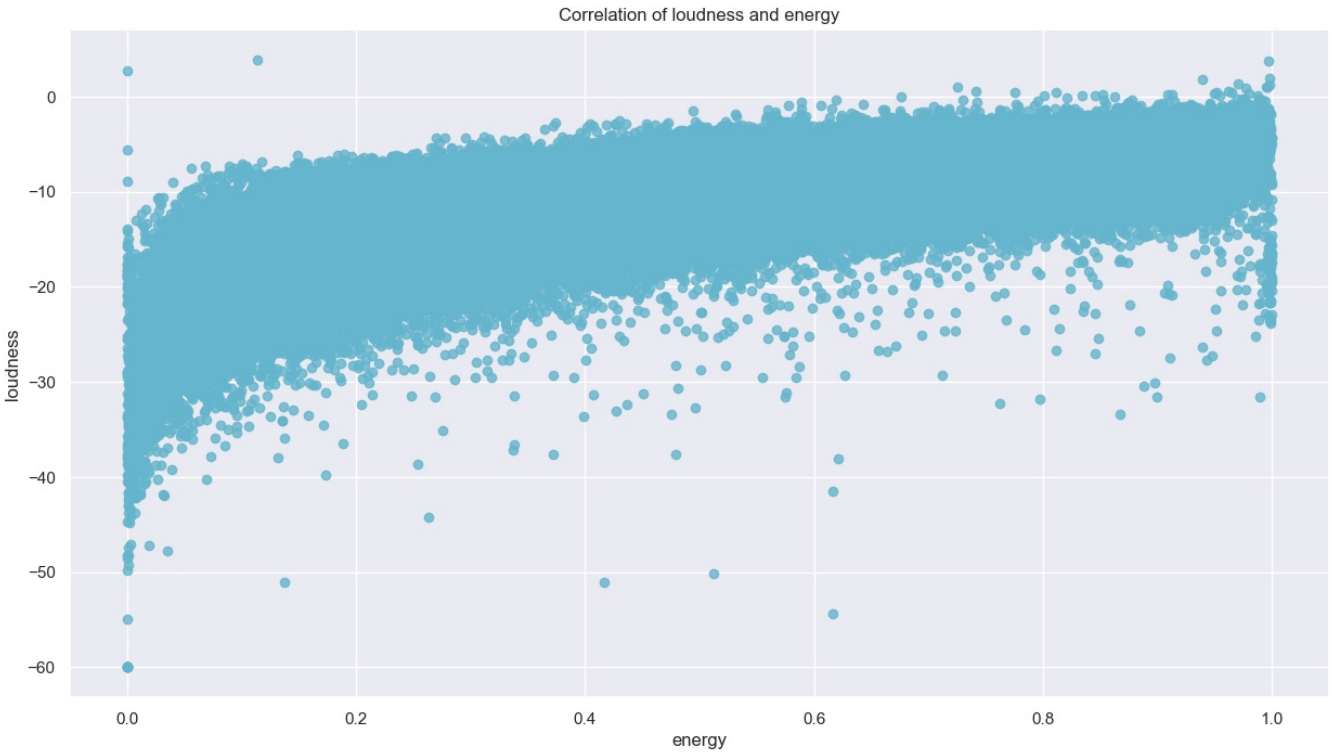


The above bar graph shows distribution of all the attributes, from which, it can be understood that,

1. Danceability and tempo are closer to normal distribution than others
2. Liveliness is right skewed
3. Loudness is left skewed

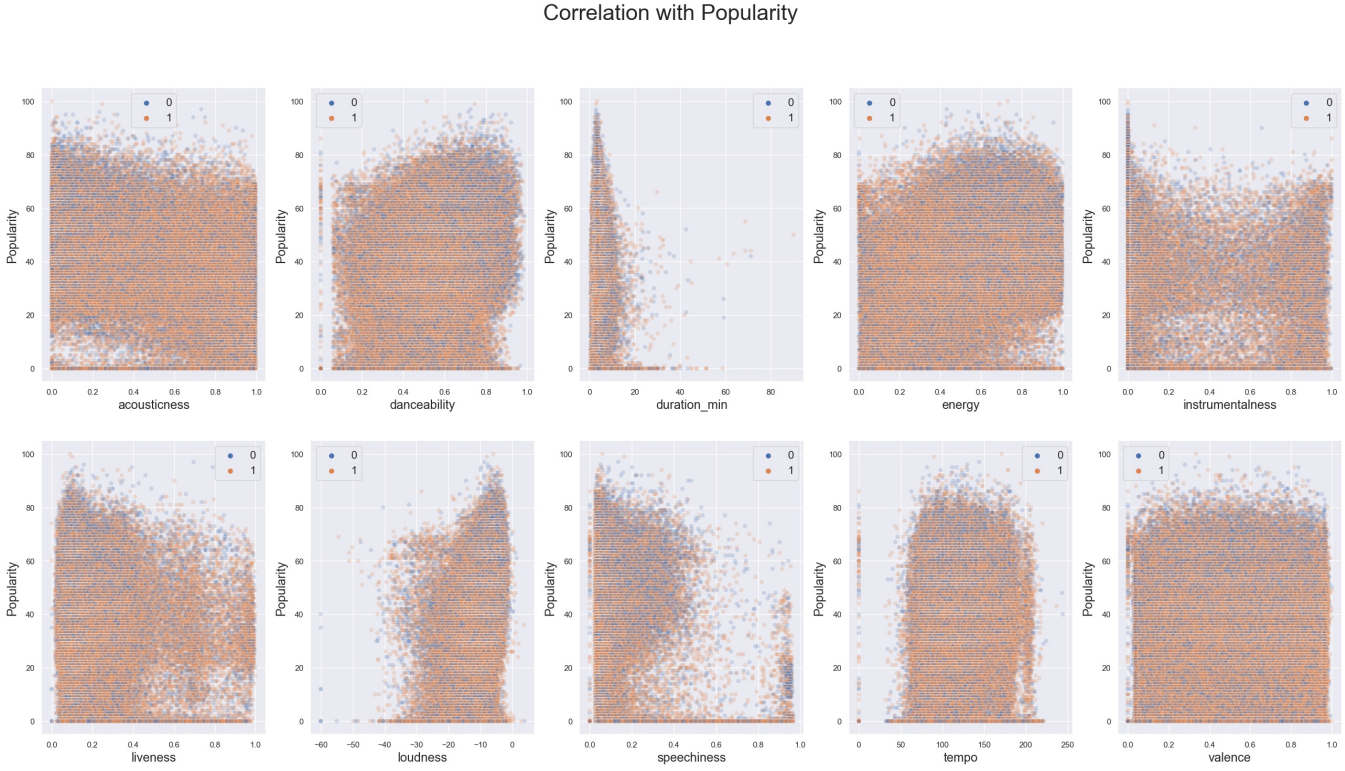
In [ ]:
```
plt.figure(figsize=(15,8))
sns.regplot(data = df,x='energy',y='loudness',color='c').set(title='Correlation of loudness and energy')
```

Out[ ]:
```
[Text(0.5, 1.0, 'Correlation of loudness and energy')]
```



The above graph shows that the energy and loudness of the songs on Spotify are positively and strongly correlated.

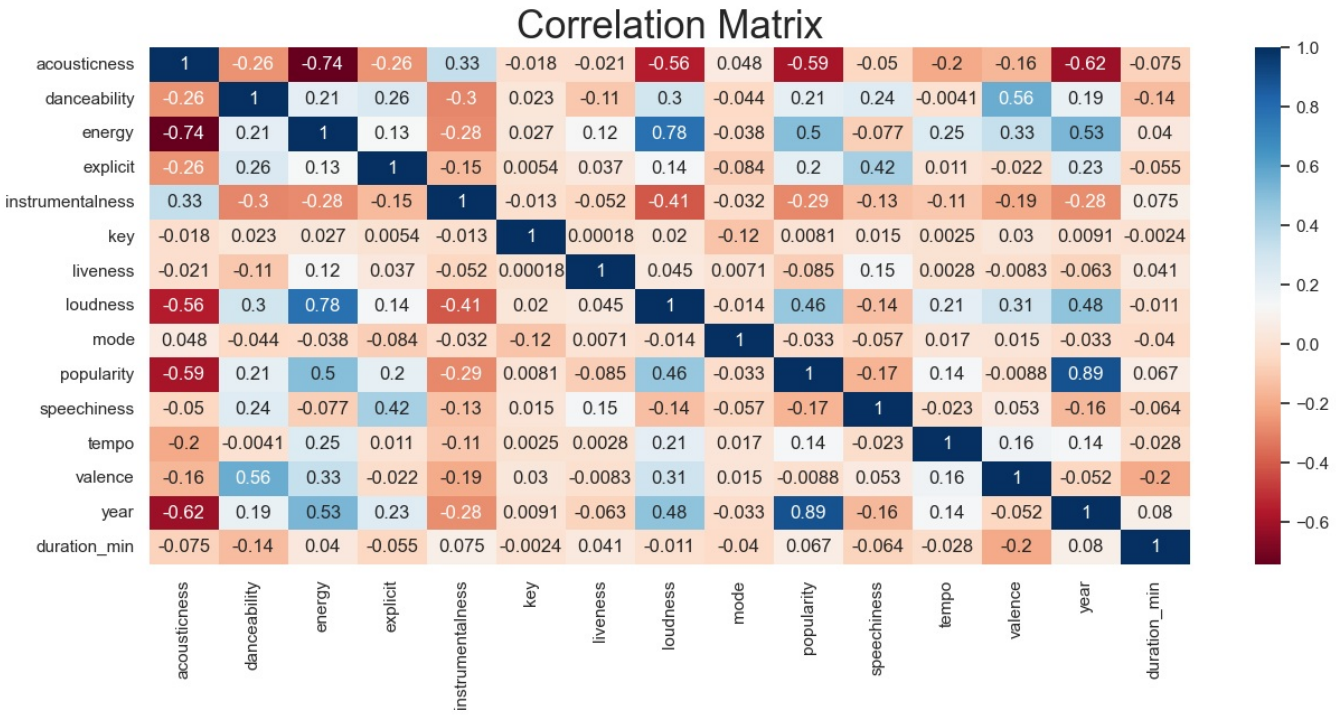
In [ ]:
```
plt.figure(figsize=(32,16))
num1 = 1

for col in ["acousticness","danceability","duration_min","energy","instrumentalness","liveness","loudness", "sp
    if num1<=10:
        ax = plt.subplot(2,5, num1)
        sns.scatterplot(x =col, y="popularity", data=df, hue='mode', legend = "full", alpha=0.2)
        plt.xlabel(col,fontsize = 17)
        plt.ylabel("Popularity",fontsize = 17)
        plt.legend(fontsize = 15)
    num1 +=1
plt.suptitle("Correlation with Popularity",fontsize = 30)
plt.show()
```



The above graph shows the correlation of all the audio attributes with that of popularity.

In [ ]:
```
plt.figure(figsize=(15,6))
plt.title('Correlation Matrix', fontsize=25)
sns.heatmap(df.corr(), cmap='RdBu', annot=True)
plt.show()
```

```
/var/folders/7h/jhk86n6n27x8cz6j32nxt0y80000gn/T/ipykernel_60973/3905058130.py:3: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  sns.heatmap(df.corr(), cmap='RdBu', annot=True)
```



Correlation matrix is a table which displays the correlation coefficients for different variables. The matrix depicts the correlation between all the possible pairs of values in a table. It is a powerful tool to summarize a large dataset and to identify and visualize patterns in the given data.

From the correlation matrix we can say:

1. High positive correlation between energy and loudness (+0.78) [which determines that with greater loudness in music, the energy increases]
2. High negative correlation between energy and acousticness (-0.74) [which determines that with greater acoustiness, energy is decreased]
3. Fairly high positive correlation between danceability and valance (+0.56) [Valence is sound positiveness, the more valence, the danceability of the music increases]
4. Failry high negative correlation between loudness and acousticness (-0.56) [which determines that with greater acoustiness, loudness is decreased]

Note: Correlation coefficient along the diagonal of the table are equal to 1 because each are perfectly correlated with itself.