In [1]:
```
pip install plotly
```

```
Requirement already satisfied: plotly in c:\users\sanka\anaconda3\lib\site-packages (5.21.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\sanka\anaconda3\lib\site-packages (from plotly)
(8.2.3)
Requirement already satisfied: packaging in c:\users\sanka\anaconda3\lib\site-packages (from plotly) (20.
4)
Requirement already satisfied: six in c:\users\sanka\anaconda3\lib\site-packages (from packaging->plotly)
(1.15.0)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\sanka\anaconda3\lib\site-packages (from packag
ing->plotly) (2.4.7)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
import pandas as pd
import numpy as np
from scipy import stats, special

import seaborn as sns
from seaborn import pairplot, heatmap
import matplotlib.pyplot as plt
from matplotlib import pyplot as plt # for plotting
%matplotlib inline
sns.set_style("whitegrid")

import warnings # http://blog.johnmuellerbooks.com/2015/11/30/warnings-in-python-and-anaconda/
warnings.filterwarnings("ignore")
import plotly.express as px
import plotly.graph_objects as go
from plotly.offline import plot, iplot, init_notebook_mode
init_notebook_mode(connected=True)
```

In [3]:
```
dataframe = pd.read_csv('SpotifyFeatures.csv')
```

In [4]:
```
print(dataframe)
```

```
        genre              artist_name                       track_name  \
0       Movie           Henri Salvador      C'est beau de faire un Show
1       Movie        Martin & les fées  Perdu d'avance (par Gad Elmaleh)
2       Movie          Joseph Williams   Don't Let Me Be Lonely Tonight
3       Movie           Henri Salvador   Dis-moi Monsieur Gordon Cooper
4       Movie             Fabien Nataf                         Ouverture
...       ...                      ...                              ...
232720   Soul                    Slave                     Son Of Slide
232721   Soul  Jr Thomas & The Volcanos                     Burning Fire
232722   Soul             Muddy Waters   (I'm Your) Hoochie Coochie Man
232723   Soul                  R.LUM.R                    With My Words
232724   Soul           Mint Condition   You Don't Have To Hurt No More

                      track_id  popularity  acousticness  danceability  \
0       0BRjO6ga9RKCKjfDqeFgWV           0       0.61100         0.389
1       0BjC1NfoEOOusryehmNudP           1       0.24600         0.590
2       0CoSDzoNIKCRs124s9uTVy           3       0.95200         0.663
3       0Gc6TVm52BwZD07Ki6tIvf           0       0.70300         0.240
4       0IuslXpMROHdEPvSl1fTQK           4       0.95000         0.331
...                        ...         ...           ...           ...
232720  2XGLdVl7lGeq8ksM6Al7jT          39       0.00384         0.687
232721  1qWZdkBl4UVPj9lK6HuuFM          38       0.03290         0.785
232722  2ziWXUmQLrXTiYjCg2fZ2t          47       0.90100         0.517
232723  6EFsue2YbIG4Qkq8Zr9Rir          44       0.26200         0.745
232724  34XO9RwPMKjbvRry54QzWn          35       0.09730         0.758

        duration_ms  energy  instrumentalness key  liveness  loudness   mode  \
0             99373   0.910          0.000000  C#    0.3460    -1.828  Major
1            137373   0.737          0.000000  F#    0.1510    -5.559  Minor
2            170267   0.131          0.000000   C    0.1030   -13.879  Minor
3            152427   0.326          0.000000  C#    0.0985   -12.178  Major
4             82625   0.225          0.123000   F    0.2020   -21.150  Major
...             ...     ...               ...  ..       ...       ...    ...
232720       326240   0.714          0.544000   D    0.0845   -10.626  Major
232721       282447   0.683          0.000880   E    0.2370    -6.944  Minor
232722       166960   0.419          0.000000   D    0.0945    -8.282  Major
232723       222442   0.704          0.000000   A    0.3330    -7.137  Major
232724       323027   0.470          0.000049  G#    0.0836    -6.708  Minor

        speechiness    tempo time_signature  valence
0            0.0525  166.969            4/4    0.814
1            0.0868  174.003            4/4    0.816
2            0.0362   99.488            5/4    0.368
3            0.0395  171.758            4/4    0.227
4            0.0456  140.576            4/4    0.390
...             ...      ...            ...      ...
232720       0.0316  115.542            4/4    0.962
232721       0.0337  113.830            4/4    0.969
232722       0.1480   84.135            4/4    0.813
232723       0.1460  100.031            4/4    0.489
232724       0.0287  113.897            4/4    0.479

[232725 rows x 18 columns]
```

In [5]:
```
dataframe.columns
```

Out[5]:
```
Index(['genre', 'artist_name', 'track_name', 'track_id', 'popularity',
       'acousticness', 'danceability', 'duration_ms', 'energy',
       'instrumentalness', 'key', 'liveness', 'loudness', 'mode',
       'speechiness', 'tempo', 'time_signature', 'valence'],
      dtype='object')
```

In [6]:
```
dataframe.shape
```

Out[6]:
```
(232725, 18)
```

In [7]:
```
dataframe.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 232725 entries, 0 to 232724
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   genre             232725 non-null  object 
 1   artist_name       232725 non-null  object 
 2   track_name        232725 non-null  object 
 3   track_id          232725 non-null  object 
 4   popularity        232725 non-null  int64  
 5   acousticness      232725 non-null  float64
 6   danceability      232725 non-null  float64
 7   duration_ms       232725 non-null  int64  
 8   energy            232725 non-null  float64
 9   instrumentalness  232725 non-null  float64
 10  key               232725 non-null  object 
 11  liveness          232725 non-null  float64
 12  loudness          232725 non-null  float64
 13  mode              232725 non-null  object 
 14  speechiness       232725 non-null  float64
 15  tempo             232725 non-null  float64
 16  time_signature    232725 non-null  object 
 17  valence           232725 non-null  float64
dtypes: float64(9), int64(2), object(7)
memory usage: 32.0+ MB
```

In [8]:
```
dataframe.describe()
```

Out[8]:

| | popularity | acousticness | danceability | duration_ms | energy | instrumentalness | liveness | loudness |
|---|---|---|---|---|---|---|---|---|
| **count** | 232725.000000 | 232725.000000 | 232725.000000 | 2.327250e+05 | 232725.000000 | 232725.000000 | 232725.000000 | 232725.000000 |
| **mean** | 41.127502 | 0.368560 | 0.554364 | 2.351223e+05 | 0.570958 | 0.148301 | 0.215009 | -9.569885 |
| **std** | 18.189948 | 0.354768 | 0.185608 | 1.189359e+05 | 0.263456 | 0.302768 | 0.198273 | 5.998204 |
| **min** | 0.000000 | 0.000000 | 0.056900 | 1.538700e+04 | 0.000020 | 0.000000 | 0.009670 | -52.457000 |
| **25%** | 29.000000 | 0.037600 | 0.435000 | 1.828570e+05 | 0.385000 | 0.000000 | 0.097400 | -11.771000 |
| **50%** | 43.000000 | 0.232000 | 0.571000 | 2.204270e+05 | 0.605000 | 0.000044 | 0.128000 | -7.762000 |
| **75%** | 55.000000 | 0.722000 | 0.692000 | 2.657680e+05 | 0.787000 | 0.035800 | 0.264000 | -5.501000 |
| **max** | 100.000000 | 0.996000 | 0.989000 | 5.552917e+06 | 0.999000 | 0.999000 | 1.000000 | 3.744000 |

In [9]:
```
#creating subsets
df1 = dataframe[['track_id' , 'artist_name' , 'popularity' ]].loc[0:15]
df1
```

Out[9]:

| | track_id | artist_name | popularity |
|---|---|---|---|
| **0** | 0BRjO6ga9RKCKjfDqeFgWV | Henri Salvador | 0 |
| **1** | 0BjC1NfoEOOusryehmNudP | Martin & les fées | 1 |
| **2** | 0CoSDzoNIKCRs124s9uTVy | Joseph Williams | 3 |
| **3** | 0Gc6TVm52BwZD07Ki6tIvf | Henri Salvador | 0 |
| **4** | 0IuslXpMROHdEPvSl1fTQK | Fabien Nataf | 4 |
| **5** | 0Mf1jKa8eNAf1a4PwTbizj | Henri Salvador | 0 |
| **6** | 0NUiKYRd6jt1LKMYGkUdnZ | Martin & les fées | 2 |
| **7** | 0PbIF9YVD505GutwotpB5C | Laura Mayne | 15 |
| **8** | 0ST6uPfvaPpJLtQwhE6KfC | Chorus | 0 |
| **9** | 0VSqZ3KStsjcfERGdcWpFO | Le Club des Juniors | 10 |
| **10** | 0XKgegoxLclihK3Klpfo3N | Leopold Stokowski | 0 |
| **11** | 0hprxsuRM5vVCOfaM7l3gQ | Randy Newman | 2 |
| **12** | 0jF6HUm18fg6QQCLHhfhC0 | Idoles De La Musique | 4 |
| **13** | 0jlY0oRAp1T4mezDyEhOad | Chorus | 3 |
| **14** | 0pXwl2CRP5awxHsF9eET3L | Richard M. Sherman | 0 |
| **15** | 0uWUjxM7oDPKpb3T2y3oZm | Michel Roux | 0 |

In [10]:
```
#sort data
sortdata=dataframe.sort_values('popularity',ascending=False)
sortdata
```

Out[10]:

| | genre | artist_name | track_name | track_id | popularity | acousticness | danceability | duration_ms | energy | ins |
|---|---|---|---|---|---|---|---|---|---|---|
| **9027** | Dance | Ariana Grande | 7 rings | 14msK75pk3pA33pzPVNtBF | 100 | 0.578000 | 0.725 | 178640 | 0.321 | |
| **107804** | Pop | Ariana Grande | 7 rings | 14msK75pk3pA33pzPVNtBF | 100 | 0.578000 | 0.725 | 178640 | 0.321 | |
| **86951** | Rap | Post Malone | Wow. | 6MWtB6iiXyIwun0YzU6DFP | 99 | 0.163000 | 0.833 | 149520 | 0.539 | |
| **107803** | Pop | Post Malone | Wow. | 6MWtB6iiXyIwun0YzU6DFP | 99 | 0.163000 | 0.833 | 149520 | 0.539 | |
| **107802** | Pop | Ariana Grande | break up with your girlfriend, i'm bored | 4kV4N9D1iKVxx1KLvtTpjS | 99 | 0.042100 | 0.726 | 190440 | 0.554 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **195435** | Movie | Sally Dworsky | Inside Out | 5d5BLVzCOxhgN02r2XqPWw | 0 | 0.000305 | 0.456 | 241400 | 0.804 | |
| **195434** | Movie | Mike Douglas | September Song | 5VSpKjPu6bud6KHyyWQrJ1 | 0 | 0.839000 | 0.331 | 282720 | 0.221 | |
| **195433** | Movie | Keith David | The Christmas Story | 5JdfRifr7rjLNy9kegVu5z | 0 | 0.653000 | 0.604 | 207040 | 0.365 | |
| **195432** | Movie | Charlton Heston | Chorus: Come And Go With Me | 5IF0TxLi4Jzo0qEdxhZbUy | 0 | 0.972000 | 0.482 | 83667 | 0.312 | |
| **0** | Movie | Henri Salvador | C'est beau de faire un Show | 0BRjO6ga9RKCKjfDqeFgWV | 0 | 0.611000 | 0.389 | 99373 | 0.910 | |

232725 rows × 18 columns

In [11]:
```
trans=dataframe.transpose()
trans
```

Out[11]:

| | 0 | 1 | 2 | 3 | |
|---|---|---|---|---|---|
| **genre** | Movie | Movie | Movie | Movie | |
| **artist_name** | Henri Salvador | Martin & les fées | Joseph Williams | Henri Salvador | |
| **track_name** | C'est beau de faire un Show | Perdu d'avance (par Gad Elmaleh) | Don't Let Me Be Lonely Tonight | Dis-moi Monsieur Gordon Cooper | |
| **track_id** | 0BRjO6ga9RKCKjfDqeFgWV | 0BjC1NfoEOOusryehmNudP | 0CoSDzoNIKCRs124s9uTVy | 0Gc6TVm52BwZD07Ki6tIvf | 0IuslXpM |
| **popularity** | 0 | 1 | 3 | 0 | |
| **acousticness** | 0.611 | 0.246 | 0.952 | 0.703 | |
| **danceability** | 0.389 | 0.59 | 0.663 | 0.24 | |
| **duration_ms** | 99373 | 137373 | 170267 | 152427 | |
| **energy** | 0.91 | 0.737 | 0.131 | 0.326 | |
| **instrumentalness** | 0 | 0 | 0 | 0 | |
| **key** | C# | F# | C | C# | |
| **liveness** | 0.346 | 0.151 | 0.103 | 0.0985 | |
| **loudness** | -1.828 | -5.559 | -13.879 | -12.178 | |
| **mode** | Major | Minor | Minor | Major | |
| **speechiness** | 0.0525 | 0.0868 | 0.0362 | 0.0395 | |
| **tempo** | 166.969 | 174.003 | 99.488 | 171.758 | |
| **time_signature** | 4/4 | 4/4 | 5/4 | 4/4 | |
| **valence** | 0.814 | 0.816 | 0.368 | 0.227 | |

18 rows × 232725 columns

In [12]:
```
dataframe.isnull().sum() # checking missing values
```

Out[12]:
```
genre              0
artist_name        0
track_name         0
track_id           0
popularity         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
dtype: int64
```

In [13]:
```
dataframe['time_signature'].unique()
```

Out[13]:
```
array(['4/4', '5/4', '3/4', '1/4', '0/4'], dtype=object)
```

In [14]:
```
dataframe['popularity'].unique()
```

Out[14]:
```
array([  0,   1,   3,   4,   2,  15,  10,   8,   5,   6,   7,  11,  65,
        63,  62,  61,  68,  64,  66,  60,  69,  71,  76,  67,  70,  72,
        57,  59,  56,  28,  31,  74,  55,  53,   9,  13,  23,  12,  44,
        33,  25,  26,  24,  22,  20,  19,  18,  16,  17,  14,  83,  81,
        73,  78,  77,  75,  45,  42,  46,  54,  41,  52,  58,  51,  43,
        47,  48,  40,  50,  49,  39,  80,  37,  35,  21,  38,  36,  29,
        34,  32,  99, 100,  97,  92,  91,  95,  90,  93,  88,  87,  89,
        96,  86,  85,  84,  94,  82,  79,  27,  30,  98], dtype=int64)
```

In [15]:
```
dataframe.describe().style.background_gradient(cmap="Greens")
```

Out[15]:

| | popularity | acousticness | danceability | duration_ms | energy | instrumentalness | liveness | loudness |
|---|---|---|---|---|---|---|---|---|
| **count** | 232725.000000 | 232725.000000 | 232725.000000 | 232725.000000 | 232725.000000 | 232725.000000 | 232725.000000 | 232725.000000 |
| **mean** | 41.127502 | 0.368560 | 0.554364 | 235122.339306 | 0.570958 | 0.148301 | 0.215009 | -9.569885 |
| **std** | 18.189948 | 0.354768 | 0.185608 | 118935.909299 | 0.263456 | 0.302768 | 0.198273 | 5.998204 |
| **min** | 0.000000 | 0.000000 | 0.056900 | 15387.000000 | 0.000020 | 0.000000 | 0.009670 | -52.457000 |
| **25%** | 29.000000 | 0.037600 | 0.435000 | 182857.000000 | 0.385000 | 0.000000 | 0.097400 | -11.771000 |
| **50%** | 43.000000 | 0.232000 | 0.571000 | 220427.000000 | 0.605000 | 0.000044 | 0.128000 | -7.762000 |
| **75%** | 55.000000 | 0.722000 | 0.692000 | 265768.000000 | 0.787000 | 0.035800 | 0.264000 | -5.501000 |
| **max** | 100.000000 | 0.996000 | 0.989000 | 5552917.000000 | 0.999000 | 0.999000 | 1.000000 | 3.744000 |

In [16]:
```
dataframe.describe(include=["bool", "object"])
```

Out[16]:

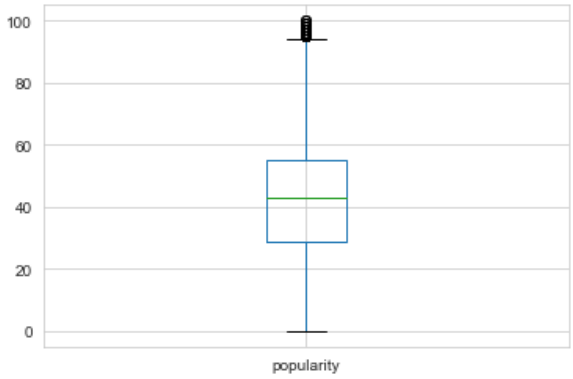
| | genre | artist_name | track_name | track_id | key | mode | time_signature |
|---|---|---|---|---|---|---|---|
| **count** | 232725 | 232725 | 232725 | 232725 | 232725 | 232725 | 232725 |
| **unique** | 27 | 14564 | 148615 | 176774 | 12 | 2 | 5 |
| **top** | Comedy | Giuseppe Verdi | Home | 6sVQNUvcVFTXvlk3ec0ngd | C | Major | 4/4 |
| **freq** | 9681 | 1394 | 100 | 8 | 27583 | 151744 | 200760 |

In [17]:
```
dataframe['duration_ms'] = (dataframe['duration_ms'] / 1000)
dataframe.rename({'duration_ms': 'duration_sec'}, axis=1, inplace=True)
dataframe.head()
```

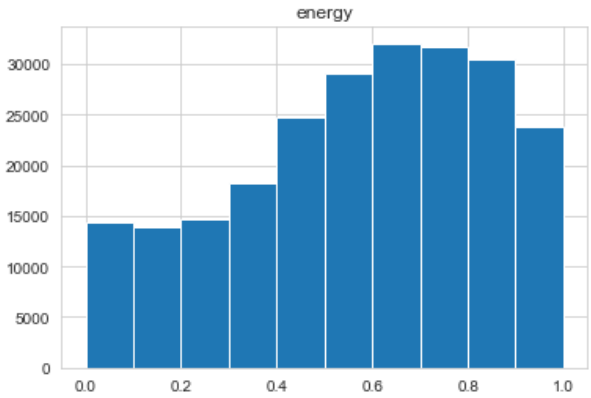
Out[17]:

| | genre | artist_name | track_name | track_id | popularity | acousticness | danceability | duration_sec | energy | instrume |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Movie | Henri Salvador | C'est beau de faire un Show | 0BRjO6ga9RKCKjfDqeFgWV | 0 | 0.611 | 0.389 | 99.373 | 0.910 | |
| **1** | Movie | Martin & les fées | Perdu d'avance (par Gad Elmaleh) | 0BjC1NfoEOOusryehmNudP | 1 | 0.246 | 0.590 | 137.373 | 0.737 | |
| **2** | Movie | Joseph Williams | Don't Let Me Be Lonely Tonight | 0CoSDzoNIKCRs124s9uTVy | 3 | 0.952 | 0.663 | 170.267 | 0.131 | |
| **3** | Movie | Henri Salvador | Dis-moi Monsieur Gordon Cooper | 0Gc6TVm52BwZD07Ki6tIvf | 0 | 0.703 | 0.240 | 152.427 | 0.326 | |
| **4** | Movie | Fabien Nataf | Ouverture | 0IusIXpMROHdEPvSl1fTQK | 4 | 0.950 | 0.331 | 82.625 | 0.225 | |

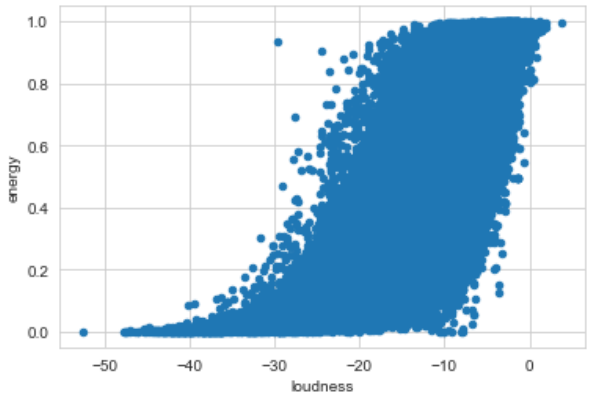
In [18]:
```
# Data Visulaization
dataframe.boxplot('popularity') #  Boxplot for numerical columns
plt.show()
```




In [19]:
```
# Histogram for numerical columns
dataframe.hist('energy')
plt.show()
```

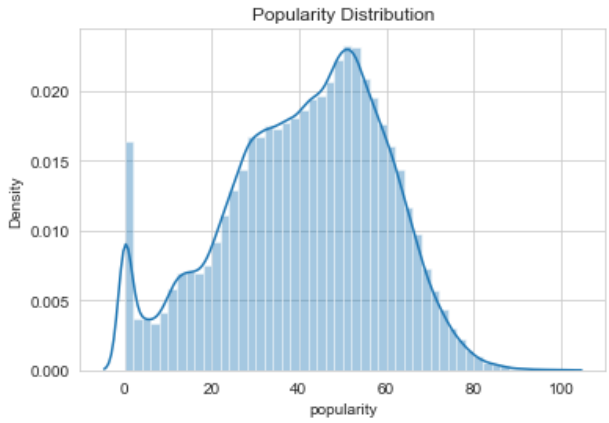



In [20]:
```
# Scatter plot for numerical columns
dataframe.plot(kind='scatter', x='loudness', y='energy')
plt.show()
```



In [21]:
```
sns.distplot(dataframe['popularity']).set_title('Popularity Distribution')
```

Out[21]:
```
Text(0.5, 1.0, 'Popularity Distribution')
```



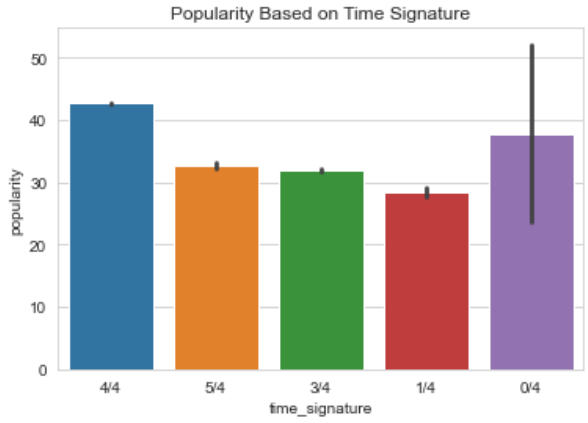

In [22]:
```
dataframe.corr()
```

Out[22]:

| | popularity | acousticness | danceability | duration_sec | energy | instrumentalness | liveness | loudness | speechiness |
|---|---|---|---|---|---|---|---|---|---|
| **popularity** | 1.000000 | -0.381295 | 0.256564 | 0.002348 | 0.248922 | -0.210983 | -0.167995 | 0.363011 | -0.151076 |
| **acousticness** | -0.381295 | 1.000000 | -0.364546 | 0.011203 | -0.725576 | 0.316154 | 0.069004 | -0.690202 | 0.150935 |
| **danceability** | 0.256564 | -0.364546 | 1.000000 | -0.125781 | 0.325807 | -0.364941 | -0.041684 | 0.438668 | 0.134560 |
| **duration_sec** | 0.002348 | 0.011203 | -0.125781 | 1.000000 | -0.030550 | 0.076021 | 0.023783 | -0.047618 | -0.016171 |
| **energy** | 0.248922 | -0.725576 | 0.325807 | -0.030550 | 1.000000 | -0.378957 | 0.192801 | 0.816088 | 0.145120 |
| **instrumentalness** | -0.210983 | 0.316154 | -0.364941 | 0.076021 | -0.378957 | 1.000000 | -0.134198 | -0.506320 | -0.177147 |
| **liveness** | -0.167995 | 0.069004 | -0.041684 | 0.023783 | 0.192801 | -0.134198 | 1.000000 | 0.045686 | 0.510147 |
| **loudness** | 0.363011 | -0.690202 | 0.438668 | -0.047618 | 0.816088 | -0.506320 | 0.045686 | 1.000000 | -0.002273 |
| **speechiness** | -0.151076 | 0.150935 | 0.134560 | -0.016171 | 0.145120 | -0.177147 | 0.510147 | -0.002273 | 1.000000 |
| **tempo** | 0.081039 | -0.238247 | 0.021939 | -0.028456 | 0.228774 | -0.104133 | -0.051355 | 0.228364 | -0.081541 |
| **valence** | 0.060076 | -0.325798 | 0.547154 | -0.141811 | 0.436771 | -0.307522 | 0.011804 | 0.399901 | 0.023842 |

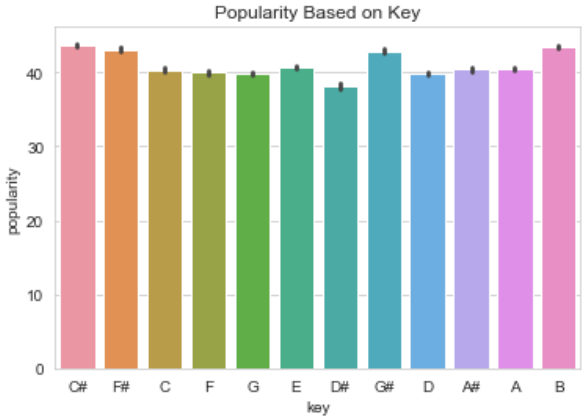
In [23]:
```
sns.barplot(x = 'time_signature', y = 'popularity', data = dataframe)
plt.title('Popularity Based on Time Signature')
```

Out[23]:
```
Text(0.5, 1.0, 'Popularity Based on Time Signature')
```



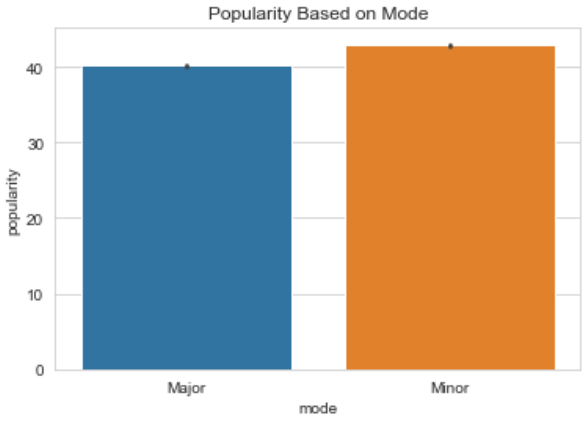
In [24]:
```
sns.barplot(x = 'key', y = 'popularity', data = dataframe)
plt.title('Popularity Based on Key')
```

Out[24]: Text(0.5, 1.0, 'Popularity Based on Key')



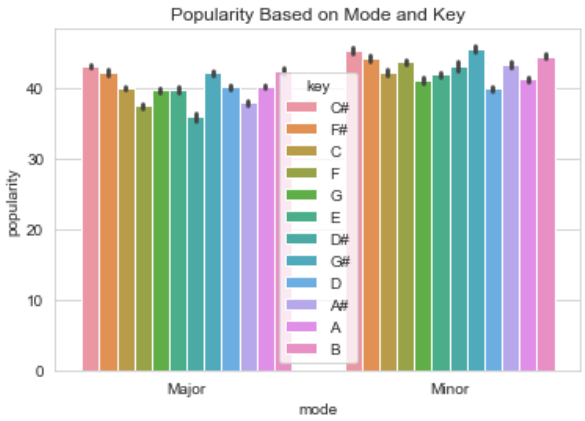

In [25]:
```
sns.barplot(x = 'mode', y = 'popularity', data = dataframe)
plt.title('Popularity Based on Mode')
```

Out[25]: Text(0.5, 1.0, 'Popularity Based on Mode')



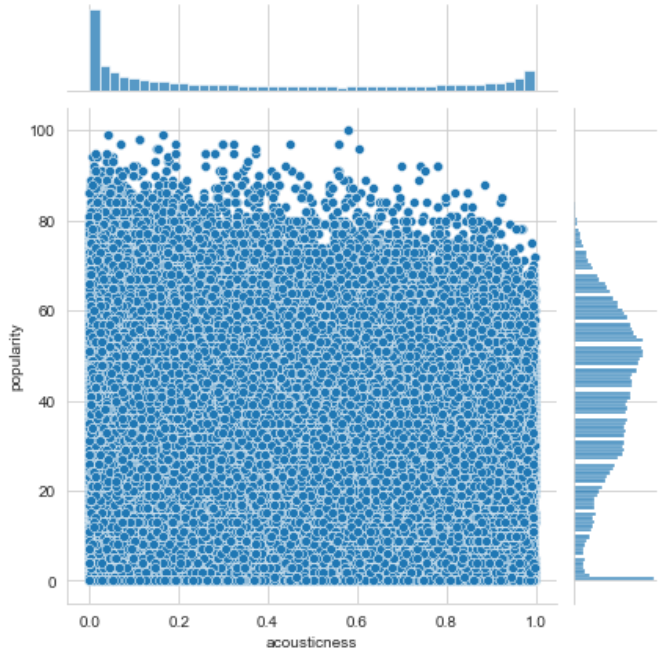

In [26]:
```
sns.barplot(x = 'mode', y = 'popularity', hue = 'key', data = dataframe)
plt.title('Popularity Based on Mode and Key')
```

Out[26]: Text(0.5, 1.0, 'Popularity Based on Mode and Key')



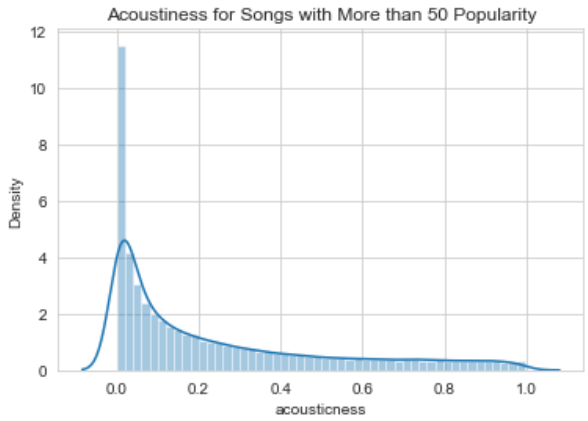
In [27]:
```
sns.jointplot(x = 'acousticness', y = 'popularity', data = dataframe)
```

Out[27]:
```
<seaborn.axisgrid.JointGrid at 0x205dd484610>
```



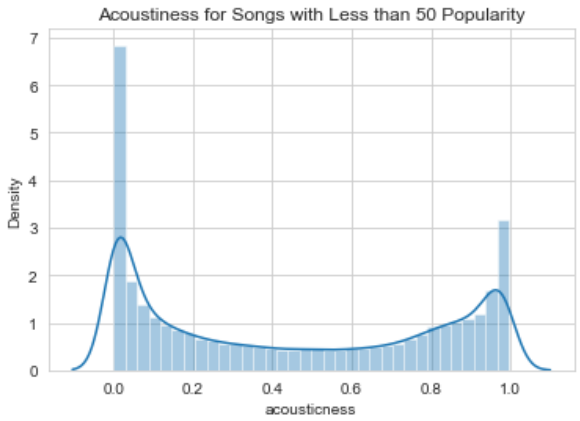

In [28]:
```
popular_above_50 = dataframe[dataframe.popularity > 50]
sns.distplot(popular_above_50['acousticness'])
plt.title('Acoustiness for Songs with More than 50 Popularity')
```

Out[28]:
```
Text(0.5, 1.0, 'Acoustiness for Songs with More than 50 Popularity')
```



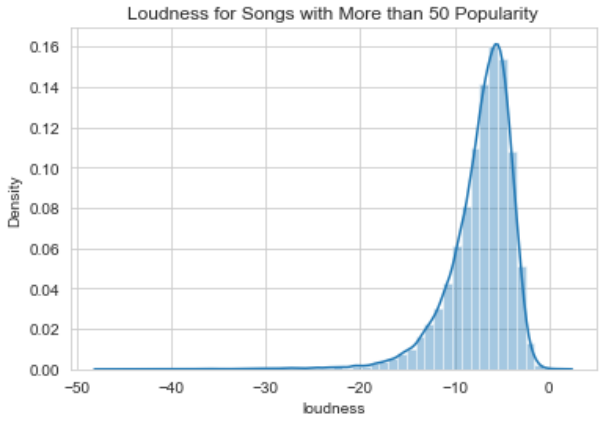
In [29]:
```
popular_below_50 = dataframe[dataframe.popularity < 50]
sns.distplot(popular_below_50['acousticness'])
plt.title('Acoustiness for Songs with Less than 50 Popularity')
```

Out[29]: Text(0.5, 1.0, 'Acoustiness for Songs with Less than 50 Popularity')



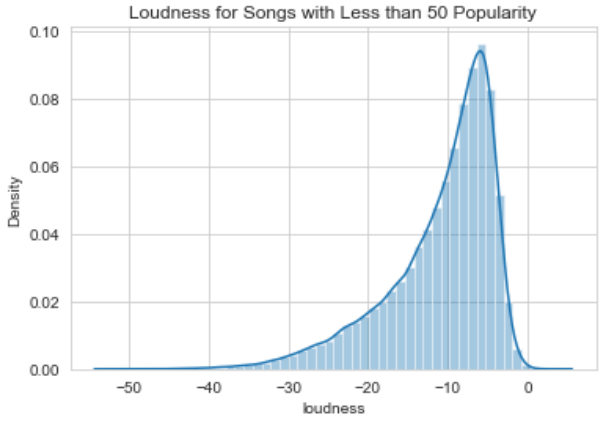

In [30]:
```
sns.distplot(popular_above_50['loudness'])
plt.title('Loudness for Songs with More than 50 Popularity')
```

Out[30]: Text(0.5, 1.0, 'Loudness for Songs with More than 50 Popularity')




In [31]:
```
popular_below_50 = dataframe[dataframe.popularity < 50]
sns.distplot(popular_below_50['loudness'])
plt.title('Loudness for Songs with Less than 50 Popularity')
```

Out[31]: Text(0.5, 1.0, 'Loudness for Songs with Less than 50 Popularity')



In [32]:
```
sns.pairplot(dataframe)
```

Out[32]: <seaborn.axisgrid.PairGrid at 0x205e509c220>

In [33]:
```
list_of_keys = dataframe['key'].unique()
for i in range(len(list_of_keys)):
    dataframe.loc[dataframe['key'] == list_of_keys[i], 'key'] = i
dataframe.sample(5)
```

Out[33]:

| | genre | artist_name | track_name | track_id | popularity | acousticness | danceability | duration_sec | ener |
|---|---|---|---|---|---|---|---|---|---|
| **152289** | R&B | The Weeknd | Starboy | 7MXVkk9YMctZqd1Srtv4MB | 85 | 0.141000 | 0.678 | 230.453 | 0.58 |
| **204356** | Soundtrack | Jóhann Jóhannsson | Fragment II | 4SP8QTXvCVNHjFTCEuuL1G | 33 | 0.156000 | 0.167 | 132.187 | 0.08 |
| **110663** | Pop | Rex Orange County | A Song About Being Sad | 6JI3ZOX6MUOlyoUc721bsX | 69 | 0.891000 | 0.568 | 136.000 | 0.26 |
| **11701** | Alternative | Mastodon | Steambreather | 4Ut80ggQbyiJN2pGCs7VfB | 49 | 0.000041 | 0.358 | 303.107 | 0.98 |
| **75260** | Children's Music | Children Songs Company | Take This - Instrumental | 6t44FUJonKndkVG2s0qj1E | 0 | 0.006760 | 0.838 | 321.080 | 0.76 |

In [34]:
```
dataframe.loc[dataframe["mode"] == 'Major', "mode"] = 1
dataframe.loc[dataframe["mode"] == 'Minor', "mode"] = 0
dataframe.sample(5)
```

Out[34]:

| | genre | artist_name | track_name | track_id | popularity | acousticness | danceability | duration_sec | energy |
|---|---|---|---|---|---|---|---|---|---|
| **24265** | Electronic | Nightmares On Wax | Look Up | 1tqU9QZxCG8DPfXPH1I25f | 49 | 0.02170 | 0.927 | 354.000 | 0.647 |
| **30805** | Anime | MAXIMUM THE HORMONE | Usukimi Billy ~koigimi sairoku hen~ | 3I1wiaafi69Nekjwq6wmsD | 27 | 0.00043 | 0.338 | 200.653 | 0.982 |
| **178238** | Jazz | Christian Scott aTunde Adjuah | Ancestral Recall (feat. Saul Williams) | 1CZvusy1pGEyqKpCyDR6LU | 38 | 0.00601 | 0.540 | 367.450 | 0.806 |
| **134660** | Reggae | Sublime | Garbage Grove | 204Mp03UDldypQSOIUmRdQ | 29 | 0.01000 | 0.793 | 132.373 | 0.542 |
| **22260** | Electronic | Portishead | Glory Box | 3Ty7OTBNSigGEpeW2PqcsC | 67 | 0.22800 | 0.481 | 305.560 | 0.409 |

In [35]:
```
list_of_time_signatures = dataframe['time_signature'].unique()
for i in range(len(list_of_time_signatures)):
    dataframe.loc[dataframe['time_signature'] == list_of_time_signatures[i], 'time_signature'] = i
dataframe.sample(5)
```

Out[35]:

| | genre | artist_name | track_name | track_id | popularity | acousticness | danceability | duration_sec | energy | ins |
|---|---|---|---|---|---|---|---|---|---|---|
| **21090** | Dance | Niall Horan | Slow Hands - Basic Tape Remix | 3XJhDig11wzszomesWNOnV | 49 | 0.00806 | 0.723 | 182.608 | 0.720 | |
| **57050** | R&B | Sean Kingston | Dumb Love | 10QJkBWQJXPr3TYaPuH6gR | 55 | 0.35600 | 0.758 | 186.773 | 0.646 | |
| **61208** | R&B | LL Cool J | Around The Way Girl | 6jL1SnyXcXiKOmw4M2RnmT | 52 | 0.00268 | 0.614 | 248.493 | 0.445 | |
| **44473** | Folk | Raffi | Mr. Sun | 6n3qzK1msumNEERJI17dWi | 47 | 0.39800 | 0.515 | 74.787 | 0.378 | |
| **29890** | Anime | DECO*27 | EGOMAMA | 77IxrKgYoLuHorNeZqHQog | 26 | 0.00244 | 0.528 | 267.173 | 0.901 | |

In [36]:
```
dataframe.loc[dataframe['popularity'] < 57, 'popularity'] = 0
dataframe.loc[dataframe['popularity'] >= 57, 'popularity'] = 1
dataframe.loc[dataframe['popularity'] == 1]
```

Out[36]:

| | genre | artist_name | track_name | track_id | popularity | acousticness | danceability | duration_sec | energy | ins |
|---|---|---|---|---|---|---|---|---|---|---|
| **135** | R&B | Mary J. Blige | Be Without You - Kendu Mix | 2YegxR5As7BeQuVp2U6pek | 1 | 0.08300 | 0.724 | 246.333 | 0.689 | |
| **136** | R&B | Rihanna | Desperado | 6KFaHC9G178beAp7P0Vi5S | 1 | 0.32300 | 0.685 | 186.467 | 0.610 | |
| **137** | R&B | Yung Bleu | Ice On My Baby (feat. Kevin Gates) - Remix | 6muW8cSjJ3rusKJ0vH5olw | 1 | 0.06750 | 0.762 | 199.520 | 0.520 | |
| **138** | R&B | Surfaces | Heaven Falls / Fall on Me | 7yHqOZfsXYlicyoMt62yC6 | 1 | 0.36000 | 0.563 | 240.597 | 0.366 | |
| **139** | R&B | Olivia O'Brien | Love Myself | 4XzgjxGKqULifVf7mnDIQK | 1 | 0.59600 | 0.653 | 213.947 | 0.621 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **230312** | Soul | James Brown | Get Up (I Feel Like Being A) Sex Machine - Pts... | 6hpmTwgNCz81H2bFEREx29 | 1 | 0.27300 | 0.833 | 318.800 | 0.661 | |
| **230782** | Soul | Alex Hepburn | If You Stay | 4sJoIeb8zWYCLHSLM0az3b | 1 | 0.04530 | 0.719 | 194.554 | 0.702 | |
| **230817** | Soul | Paloma Faith | Make Your Own Kind of Music | 5jsFFhABp2FkasGr4QcQd6 | 1 | 0.00862 | 0.567 | 163.840 | 0.753 | |
| **230946** | Soul | James Brown | Papa's Got A Brand New Bag - Pt. 1 | 5aZzmPUv5a2nna9sxBrmpL | 1 | 0.51900 | 0.775 | 128.973 | 0.725 | |
| **231493** | Soul | Simply Red | The Air That I Breathe | 4Sfq2ZuUK9tS66eXqCCKRF | 1 | 0.21000 | 0.660 | 262.827 | 0.560 | |

49104 rows × 18 columns

In [37]:
```
pip install xgboost
```

```
Requirement already satisfied: xgboost in c:\users\sanka\anaconda3\lib\site-packages (2.0.3)
Requirement already satisfied: numpy in c:\users\sanka\anaconda3\lib\site-packages (from xgboost) (1.19.2)
Requirement already satisfied: scipy in c:\users\sanka\anaconda3\lib\site-packages (from xgboost) (1.5.2)
Note: you may need to restart the kernel to use updated packages.
```

In [38]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC, LinearSVC
from xgboost import XGBClassifier

from sklearn.metrics import make_scorer, accuracy_score, roc_auc_score
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
```

In [39]:
```
features = ["acousticness", "danceability", "duration_sec", "energy", "instrumentalness", "key", "liveness"
            "mode", "speechiness", "tempo", "time_signature", "valence"]
```

In [40]:
```
training = dataframe.sample(frac = 0.8,random_state = 420)
X_train = training[features]
y_train = training['popularity']
X_test = dataframe.drop(training.index)[features]
```

In [41]:
```
X_train, X_valid, y_train, y_valid = train_test_split(X_train, y_train, test_size = 0.2, random_state = 420
```

In [42]:
```
LR_Model = LogisticRegression()
LR_Model.fit(X_train, y_train)
LR_Predict = LR_Model.predict(X_valid)
LR_Accuracy = accuracy_score(y_valid, LR_Predict)
print("Accuracy: " + str(LR_Accuracy))
```

```
Accuracy: 0.7900418949403802
```

In [43]:
```
data = dataframe.copy()
data.head()
```

Out[43]:

| | genre | artist_name | track_name | track_id | popularity | acousticness | danceability | duration_sec | energy | instrume |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Movie | Henri Salvador | C'est beau de faire un Show | 0BRjO6ga9RKCKjfDqeFgWV | 0 | 0.611 | 0.389 | 99.373 | 0.910 | |
| **1** | Movie | Martin & les fées | Perdu d'avance (par Gad Elmaleh) | 0BjC1NfoEOOusryehmNudP | 0 | 0.246 | 0.590 | 137.373 | 0.737 | |
| **2** | Movie | Joseph Williams | Don't Let Me Be Lonely Tonight | 0CoSDzoNIKCRs124s9uTVy | 0 | 0.952 | 0.663 | 170.267 | 0.131 | |
| **3** | Movie | Henri Salvador | Dis-moi Monsieur Gordon Cooper | 0Gc6TVm52BwZD07Ki6tIvf | 0 | 0.703 | 0.240 | 152.427 | 0.326 | |
| **4** | Movie | Fabien Nataf | Ouverture | 0IuslXpMROHdEPvSl1fTQK | 0 | 0.950 | 0.331 | 82.625 | 0.225 | |

In [44]:
```
data = data.drop(labels=["track_id", "artist_name", "genre", "track_name"], axis=1)
```

In [45]:
```
data.head()
```

Out[45]:

| | popularity | acousticness | danceability | duration_sec | energy | instrumentalness | key | liveness | loudness | mode | speechiness | tem |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 0.611 | 0.389 | 99.373 | 0.910 | 0.000 | 0 | 0.3460 | -1.828 | 1 | 0.0525 | 166.9 |
| **1** | 0 | 0.246 | 0.590 | 137.373 | 0.737 | 0.000 | 1 | 0.1510 | -5.559 | 0 | 0.0868 | 174.0 |
| **2** | 0 | 0.952 | 0.663 | 170.267 | 0.131 | 0.000 | 2 | 0.1030 | -13.879 | 0 | 0.0362 | 99.4 |
| **3** | 0 | 0.703 | 0.240 | 152.427 | 0.326 | 0.000 | 0 | 0.0985 | -12.178 | 1 | 0.0395 | 171.7 |
| **4** | 0 | 0.950 | 0.331 | 82.625 | 0.225 | 0.123 | 3 | 0.2020 | -21.150 | 1 | 0.0456 | 140.5 |

In [46]:
```
#Test-Train Split
from sklearn.model_selection import train_test_split

X = data.drop("danceability", axis=1)
y = dataframe["danceability"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=21)
print("Shape of X_train",X_train.shape)
print("Shape of X_test",X_test.shape)
print("Shape of y_train", y_train.shape)
print("Shape of y_test", y_test.shape)
```

```
Shape of X_train (162907, 13)
Shape of X_test (69818, 13)
Shape of y_train (162907,)
Shape of y_test (69818,)
```

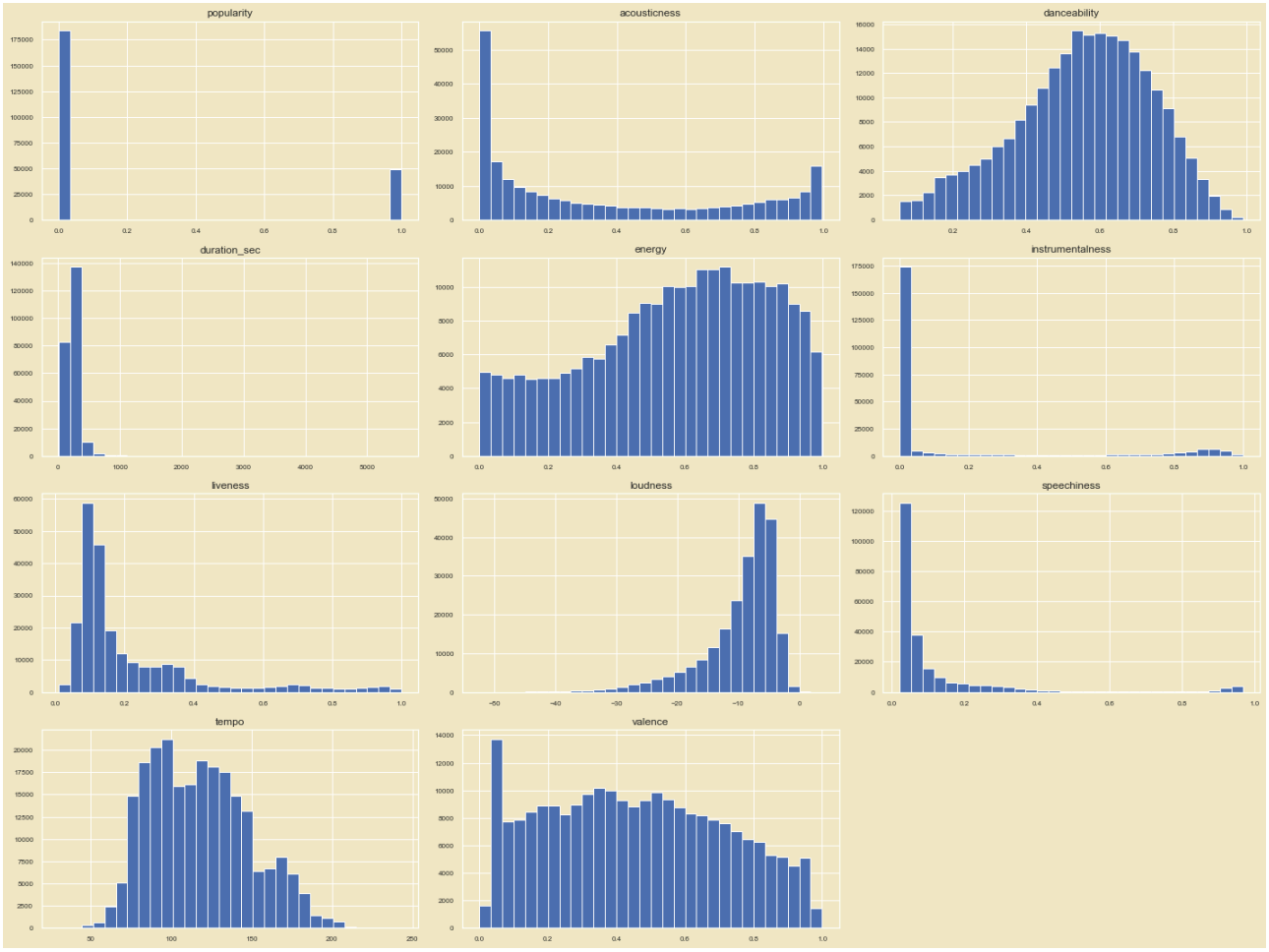
In [47]:
```
num_cols = data[data.columns[(data.dtypes == 'float64') | (data.dtypes == 'int64')]]
num_cols.shape
```

Out[47]:
```
(232725, 11)
```

In [48]:
```
num_cols.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 232725 entries, 0 to 232724
Data columns (total 11 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   popularity        232725 non-null  int64  
 1   acousticness      232725 non-null  float64
 2   danceability      232725 non-null  float64
 3   duration_sec      232725 non-null  float64
 4   energy            232725 non-null  float64
 5   instrumentalness  232725 non-null  float64
 6   liveness          232725 non-null  float64
 7   loudness          232725 non-null  float64
 8   speechiness       232725 non-null  float64
 9   tempo             232725 non-null  float64
 10  valence           232725 non-null  float64
dtypes: float64(10), int64(1)
memory usage: 19.5 MB
```

In [49]:
```
#Checking distribution of numerical columns
sns.set_style('darkgrid')
sns.set(rc={"axes.facecolor":"#F2EAC5","figure.facecolor":"#F2EAC5"})
num_cols.hist(figsize=(20,15), bins=30, xlabelsize=8, ylabelsize=8)
plt.tight_layout()
plt.show()
```




In [50]:
```
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

# Assuming that `data` is your original DataFrame
X = data.drop("danceability", axis=1)
scaler = StandardScaler()
scaled_features = scaler.fit_transform(X)

kmeans = KMeans(n_clusters=7, random_state=48)
data['cluster'] = kmeans.fit_predict(scaled_features)
```

In [51]:
```
kmeans = KMeans(n_clusters=7, random_state=48)
data['cluster'] = kmeans.fit_predict(scaled_features)
```

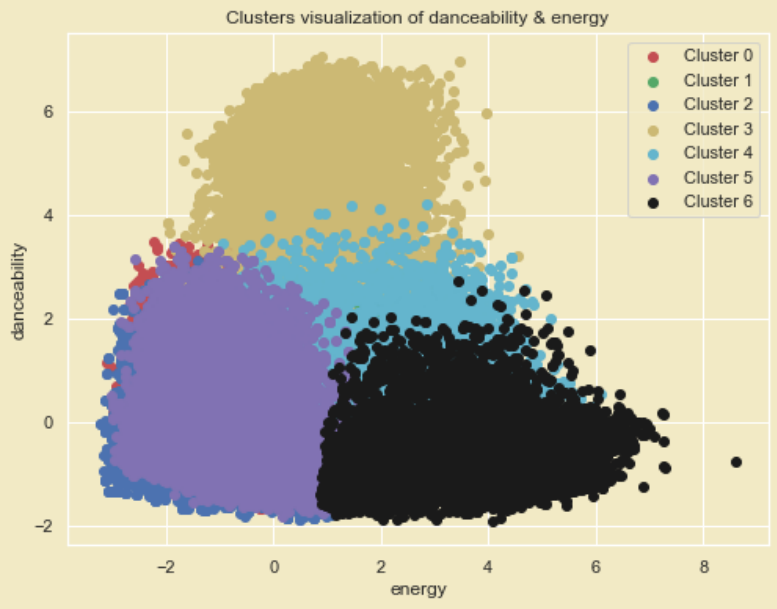
In [52]:
```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
reduced_features = pca.fit_transform(scaled_features)
```

In [53]:
```
# Create a scatter plot
plt.figure(figsize=(8, 6))
colors = ['r', 'g', 'b', 'y', 'c', 'm', 'k']  # Colors for the clusters

# Plot each cluster
for i in range(7):

    idx = data['cluster'] == i
    plt.scatter(reduced_features[idx, 0], reduced_features[idx, 1], c=colors[i], label=f'Cluster {i}')

# Adjust visuals
plt.title('Clusters visualization of danceability & energy')
plt.xlabel('energy')
plt.ylabel('danceability')
plt.legend()
plt.grid(True)
plt.show()
```




In [ ]: