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

In [2]:
```
data = pd.read_csv("IMDB-Movie-Data.csv")
```

1. Display top 5 rows of dataset

In [60]:
```
data.head(5)
```

Out[60]:

	Rank	Title	Genre	Description	Director	Actors	Yea
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	201
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	201
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	201
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	201
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	201

2. Display last 5 rows of dataset

In [61]:
```
data.tail(5)
```

Out[61]:

	Rank	Title	Genre	Description	Director	Actors	Year	Ru (Mir
995	996	Secret in Their Eyes	Crime,Drama,Mystery	A tight-knit team of rising investigators, alo...	Billy Ray	Chiwetel Ejiofor, Nicole Kidman, Julia Roberts...	2015	
996	997	Hostel: Part II	Horror	Three American college students studying abroa...	Eli Roth	Lauren German, Heather Matarazzo, Bijou Philli...	2007	
997	998	Step Up 2: The Streets	Drama,Music,Romance	Romantic sparks occur between two dance studen...	Jon M. Chu	Robert Hoffman, Briana Evigan, Cassie Ventura,...	2008	
998	999	Search Party	Adventure,Comedy	A pair of friends embark on a mission to reuni...	Scot Armstrong	Adam Pally, T.J. Miller, Thomas Middleditch,Sh...	2014	
999	1000	Nine Lives	Comedy,Family,Fantasy	A stuffy businessman finds himself trapped ins...	Barry Sonnenfeld	Kevin Spacey, Jennifer Garner, Robbie Amell,Ch...	2016	

3. Find Shape of Our Dataset (Number of Rows And Number of Columns)

In [5]:
```
data.shape
```

Out[5]:
```
(1000, 12)
```

In [6]:
```
print("No of Rows :",data.shape[0])
print("No of Columns :",data.shape[1])
```

```
No of Rows : 1000
No of Columns : 12
```

4. Getting Information About Our Dataset Like Total Number Rows, Total Number of Columns, Datatypes of Each Column And Memory Requirement

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Rank                1000 non-null   int64  
 1   Title               1000 non-null   object 
 2   Genre               1000 non-null   object 
 3   Description         1000 non-null   object 
 4   Director            1000 non-null   object 
 5   Actors              1000 non-null   object 
 6   Year                1000 non-null   int64  
 7   Runtime (Minutes)   1000 non-null   int64  
 8   Rating              1000 non-null   float64
 9   Votes               1000 non-null   int64  
 10  Revenue (Millions)  872 non-null    float64
 11  Metascore           936 non-null    float64
dtypes: float64(3), int64(4), object(5)
memory usage: 93.9+ KB
```

5. Check Missing Values In The Dataset

```
In [8]: data.isnull().values.any()
```

```
Out[8]: True
```

```
In [9]: data.isnull().value_counts('Metascore')
```

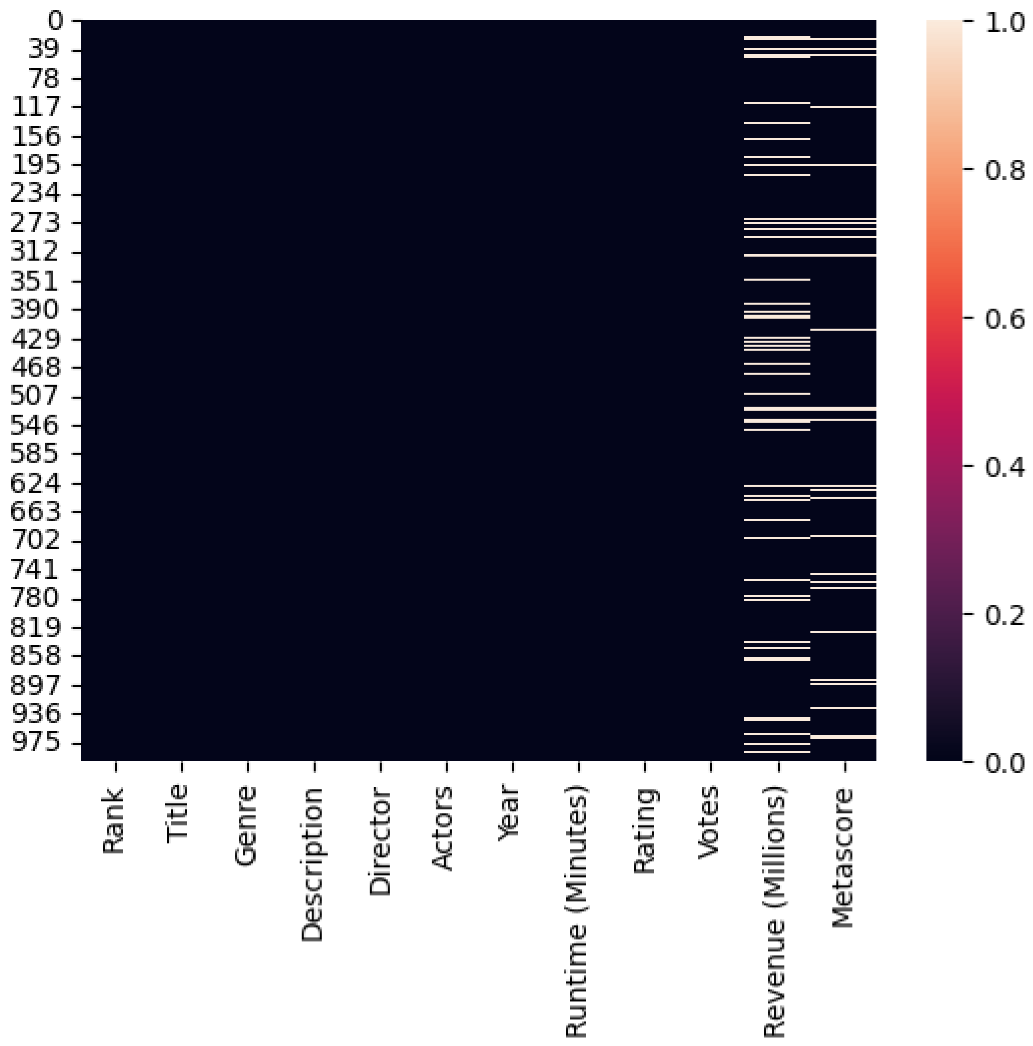
```
Out[9]: Metascore
False    936
True      64
dtype: int64
```

```
In [10]: data.isnull().sum()
```

```
Out[10]: Rank                    0
Title                   0
Genre                   0
Description             0
Director                0
Actors                  0
Year                    0
Runtime (Minutes)       0
Rating                  0
Votes                   0
Revenue (Millions)    128
Metascore              64
dtype: int64
```

```
In [11]: sns.heatmap(data.isnull())
```

```
Out[11]: <AxesSubplot: >
```



In [12]:
```
data.isnull().sum() * 100 / len(data)
```

Out[12]:
```
Rank                   0.0
Title                  0.0
Genre                  0.0
Description            0.0
Director               0.0
Actors                 0.0
Year                   0.0
Runtime (Minutes)      0.0
Rating                 0.0
Votes                  0.0
Revenue (Millions)    12.8
Metascore              6.4
dtype: float64
```

6. Drop All The Missing Values

In [13]:
```
df = data.dropna(axis = 0)
```

In [14]:
```
df.isnull().sum()
```

Out[14]:
```
Rank                  0
Title                 0
Genre                 0
Description           0
Director              0
Actors                0
Year                  0
Runtime (Minutes)     0
Rating                0
Votes                 0
Revenue (Millions)    0
Metascore             0
dtype: int64
```

In [15]:
```
df.isnull().values.any()
```

Out[15]:
```
False
```

7. Check For Duplicate Data

In [16]:
```
data.duplicated().sum()
```

Out[16]:
```
0
```

In [17]:
```
data.duplicated().values.any()
```

Out[17]:
```
False
```

In [18]:
```
df1 = data.drop_duplicates()
```

8. Get Overall Statistics About The DataFrame

In [19]:
```
data.describe()
```

Out[19]:

	Rank	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metasco
count	1000.000000	1000.000000	1000.000000	1000.000000	1.000000e+03	872.000000	936.0000
mean	500.500000	2012.783000	113.172000	6.723200	1.698083e+05	82.956376	58.9850
std	288.819436	3.205962	18.810908	0.945429	1.887626e+05	103.253540	17.1947
min	1.000000	2006.000000	66.000000	1.900000	6.100000e+01	0.000000	11.0000
25%	250.750000	2010.000000	100.000000	6.200000	3.630900e+04	13.270000	47.0000
50%	500.500000	2014.000000	111.000000	6.800000	1.107990e+05	47.985000	59.5000
75%	750.250000	2016.000000	123.000000	7.400000	2.399098e+05	113.715000	72.0000
max	1000.000000	2016.000000	191.000000	9.000000	1.791916e+06	936.630000	100.0000

In [20]:
```
data.describe(include = 'all')
```

Out[20]:

	Rank	Title	Genre	Description	Director	Actors	Year
count	1000.000000	1000	1000	1000	1000	1000	1000.000000
unique	NaN	999	207	1000	644	996	NaN
top	NaN	The Host	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	Ridley Scott	Jennifer Lawrence, Josh Hutcherson, Liam Hemsw...	NaN
freq	NaN	2	50	1	8	2	NaN
mean	500.500000	NaN	NaN	NaN	NaN	NaN	2012.783000
std	288.819436	NaN	NaN	NaN	NaN	NaN	3.205962
min	1.000000	NaN	NaN	NaN	NaN	NaN	2006.000000
25%	250.750000	NaN	NaN	NaN	NaN	NaN	2010.000000
50%	500.500000	NaN	NaN	NaN	NaN	NaN	2014.000000
75%	750.250000	NaN	NaN	NaN	NaN	NaN	2016.000000
max	1000.000000	NaN	NaN	NaN	NaN	NaN	2016.000000

9. Display Title of The Movie Having Runtime Greater Than or equal to 180 Minutes

In [21]:
```
data[data['Runtime (Minutes)'] >= 180]['Title']
```

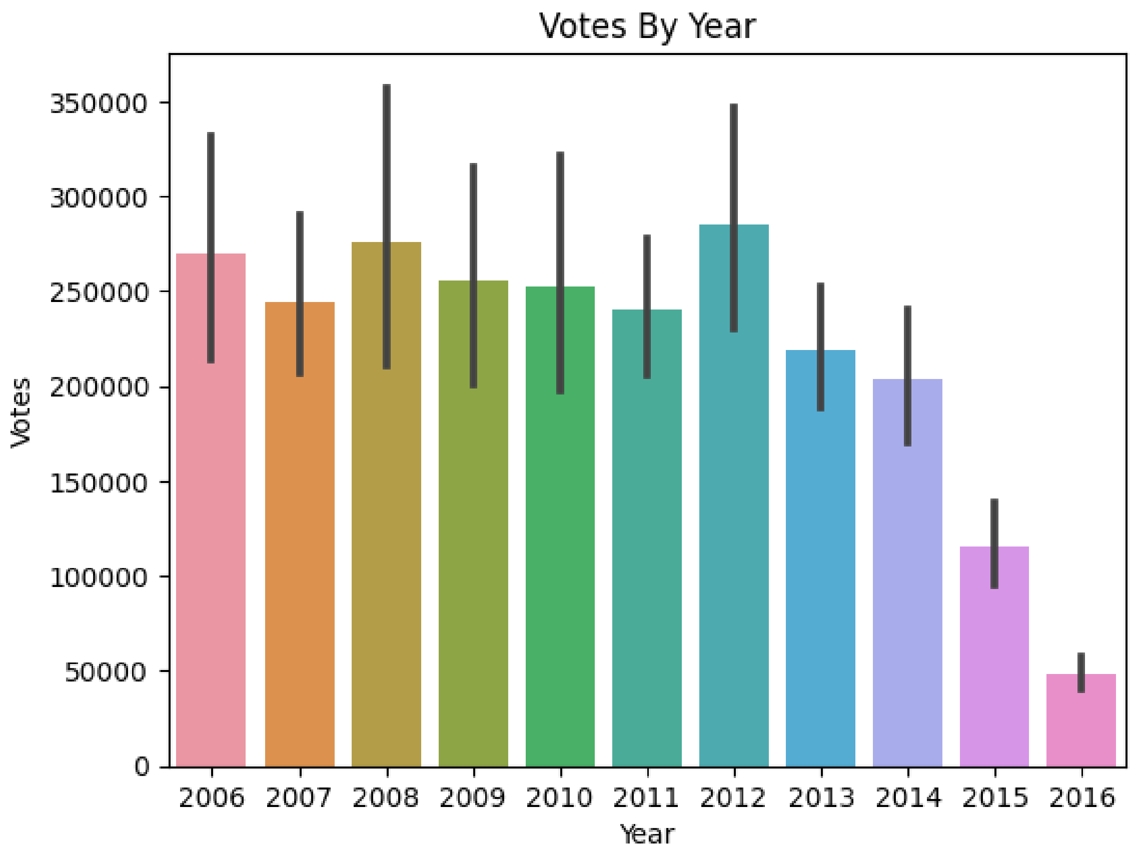
Out[21]:
```
82      The Wolf of Wall Street
88            The Hateful Eight
311              La vie d'Adèle
828                 Grindhouse
965              Inland Empire
Name: Title, dtype: object
```

10. In Which Year There Was The Highest Average Voting?

In [22]:
```
data.groupby('Year')['Votes'].mean().sort_values(ascending = False)
```

Out[22]:
```
Year
2012    285226.093750
2008    275505.384615
2006    269289.954545
2009    255780.647059
2010    252782.316667
2007    244331.037736
2011    240790.301587
2013    219049.648352
2014    203930.224490
2015    115726.220472
2016     48591.754209
Name: Votes, dtype: float64
```

In [23]:
```python
sns.barplot(data=data, x='Year',y='Votes')
plt.title('Votes By Year')
plt.show()
```

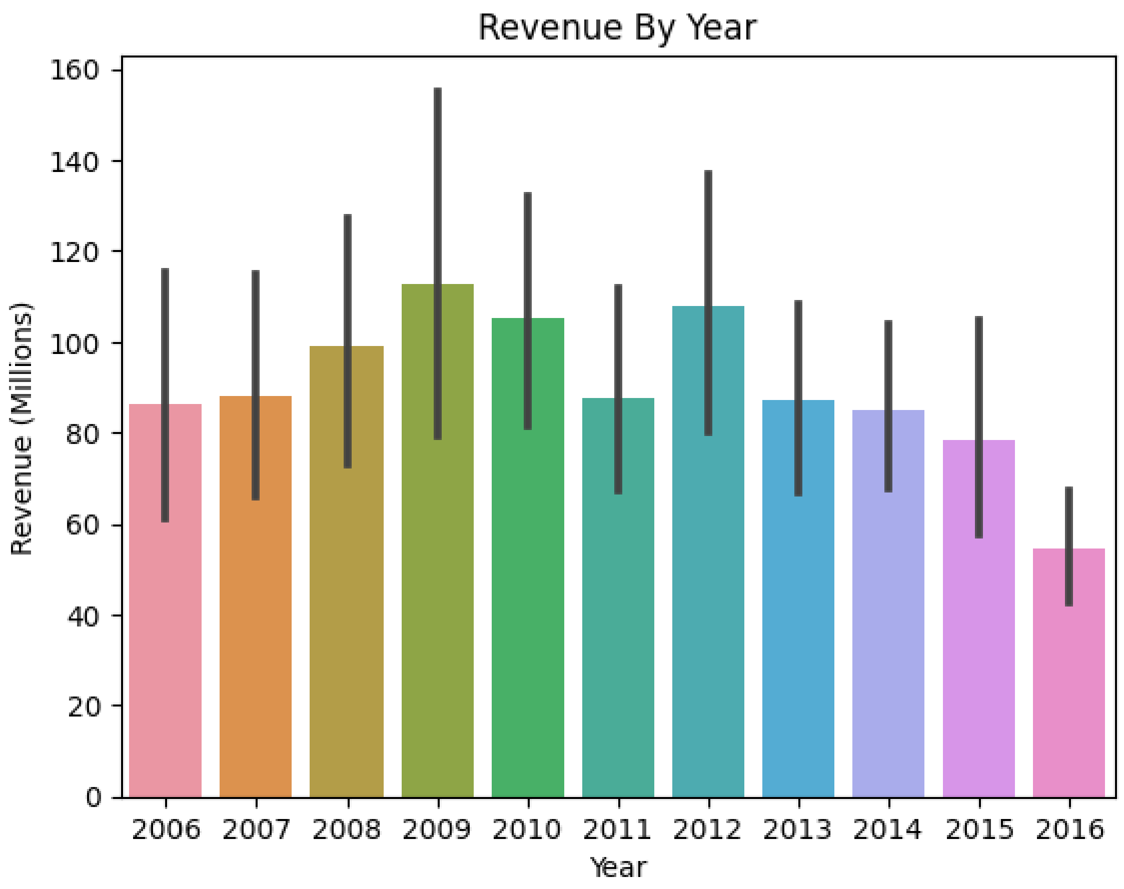


11. In Which Year There Was The Highest Average Revenue?

In [24]:
```python
data.groupby('Year')['Revenue (Millions)'].mean().sort_values(ascending = False)
```

Out[24]:
```
Year
2009    112.601277
2012    107.973281
2010    105.081579
2008     99.082745
2007     87.882245
2011     87.612258
2013     87.121818
2006     86.296667
2014     85.078723
2015     78.355044
2016     54.690976
Name: Revenue (Millions), dtype: float64
```

In [25]:
```python
sns.barplot(data=data, x='Year',y='Revenue (Millions)')
plt.title('Revenue By Year')
plt.show()
```



12. Find The Average Rating For Each Director

In [26]:
```python
data.groupby('Director')['Rating'].mean().sort_values(ascending = False)
```

```
Out[26]: Director
         Nitesh Tiwari         8.80
         Christopher Nolan     8.68
         Olivier Nakache       8.60
         Makoto Shinkai        8.60
         Aamir Khan            8.50
                               ...
         Micheal Bafaro        3.50
         Jonathan Holbrook     3.20
         Shawn Burkett         2.70
         James Wong            2.70
         Jason Friedberg       1.90
         Name: Rating, Length: 644, dtype: float64
```

13. Display Top 10 Lengthy Movies Title and Runtime

```
In [27]: df.columns
```

```
Out[27]: Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
                'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
                'Metascore'],
               dtype='object')
```

```
In [28]: dataTop10 = df.sort_values('Runtime (Minutes)', ascending = False)
         dataTop10.head(10)[['Title','Runtime (Minutes)']].set_index('Title')
```

Out[28]:

	Runtime (Minutes)
Title	
The Hateful Eight	187
The Wolf of Wall Street	180
La vie d'Adèle	180
Cloud Atlas	172
3 Idiots	170
Pirates of the Caribbean: At World's End	169
Interstellar	169
The Hobbit: An Unexpected Journey	169
The Curious Case of Benjamin Button	166
Transformers: Age of Extinction	165

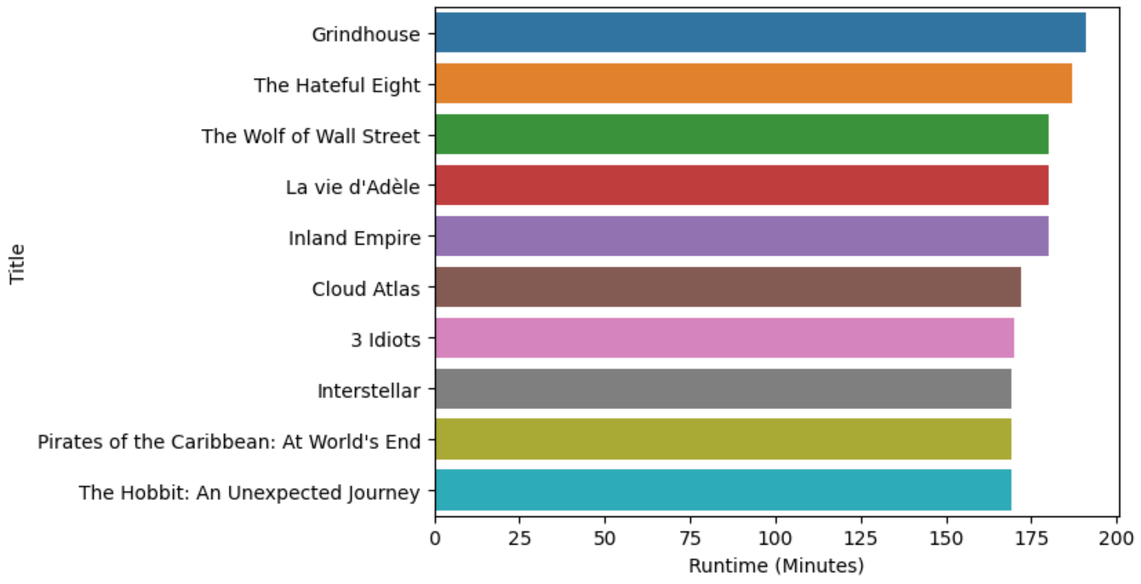
```
In [29]: top10_len = data.nlargest(10,'Runtime (Minutes)')[['Title','Runtime (Minutes)']]
         .set_index('Title')
         top10_len
```

Out[29]:

Title	Runtime (Minutes)
Grindhouse	191
The Hateful Eight	187
The Wolf of Wall Street	180
La vie d'Adèle	180
Inland Empire	180
Cloud Atlas	172
3 Idiots	170
Interstellar	169
Pirates of the Caribbean: At World's End	169
The Hobbit: An Unexpected Journey	169

In [30]:
```
sns.barplot(data=top10_len, x='Runtime (Minutes)', y=top10_len.index)
```

Out[30]:
```
<AxesSubplot: xlabel='Runtime (Minutes)', ylabel='Title'>
```



14. Display Number of Movies Per Year

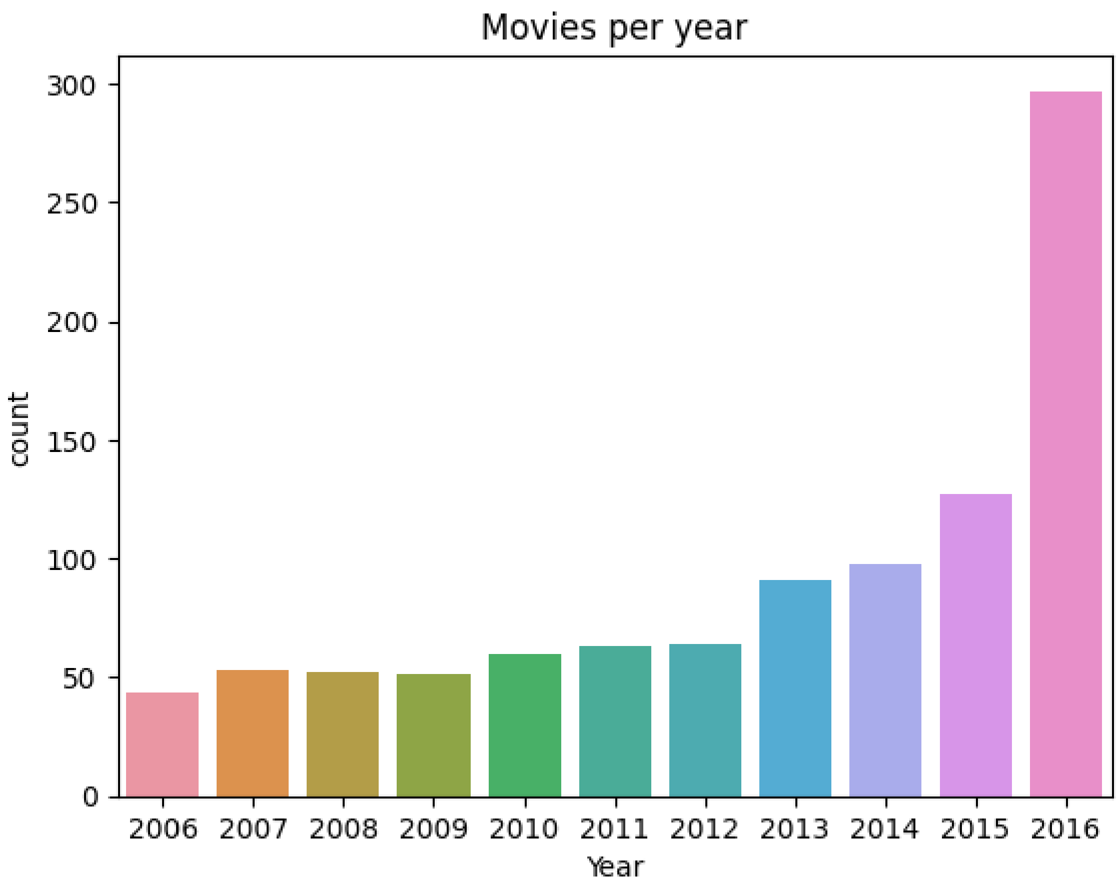
In [31]:
```
df_MoviesPerYear = data.groupby('Year')['Rank'].count()
df_MoviesPerYear
```

Out[31]:
```
Year
2006     44
2007     53
2008     52
2009     51
2010     60
2011     63
2012     64
2013     91
2014     98
2015    127
2016    297
Name: Rank, dtype: int64
```

In [32]:
```
data['Year'].value_counts()
```

Out[32]:
```
2016    297
2015    127
2014     98
2013     91
2012     64
2011     63
2010     60
2007     53
2008     52
2009     51
2006     44
Name: Year, dtype: int64
```

In [33]:
```
sns.countplot(data=data, x='Year')
plt.title('Movies per year')
plt.show()
```



15. Find Most Popular Movie Title (Highest Revenue)

In [34]:
```
data.head(2)
```

Out[34]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014	121
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012	124

In [35]:
```
data.nlargest(5,'Revenue (Millions)')[['Title','Revenue (Millions)']]
```

Out[35]:

	Title	Revenue (Millions)
50	Star Wars: Episode VII - The Force Awakens	936.63
87	Avatar	760.51
85	Jurassic World	652.18
76	The Avengers	623.28
54	The Dark Knight	533.32

In [36]:
```
data[data['Revenue (Millions)'].max() == data['Revenue (Millions)']][['Title','R
```

Out[36]:

	Title	Revenue (Millions)
50	Star Wars: Episode VII - The Force Awakens	936.63

In [37]:
```
data.nsmallest(5,'Revenue (Millions)')[['Title','Revenue (Millions)']].set_index
```

Out[37]:

Title	Revenue (Millions)
A Kind of Murder	0.00
Dead Awake	0.01
Wakefield	0.01
Lovesong	0.01
Love, Rosie	0.01

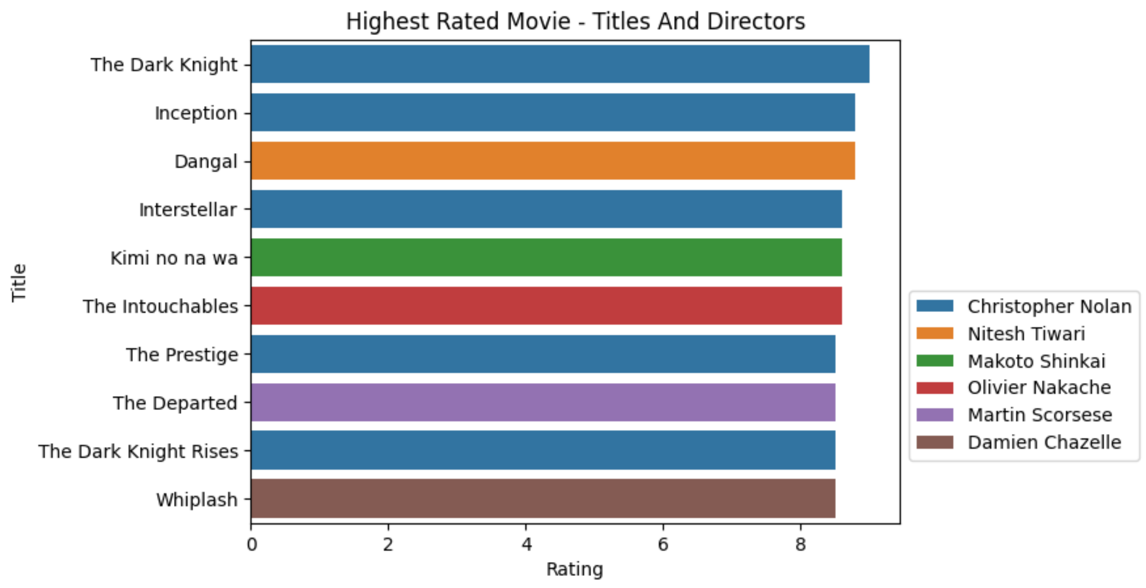
16. Display Top 10 Highest Rated Movie Titles And its Directors

In [38]:
```
top10_highrated_mvi = data.nlargest(10,'Rating')[['Title','Rating','Director']].
top10_highrated_mvi
```

Out[38]:

Title	Rating	Director
The Dark Knight	9.0	Christopher Nolan
Inception	8.8	Christopher Nolan
Dangal	8.8	Nitesh Tiwari
Interstellar	8.6	Christopher Nolan
Kimi no na wa	8.6	Makoto Shinkai
The Intouchables	8.6	Olivier Nakache
The Prestige	8.5	Christopher Nolan
The Departed	8.5	Martin Scorsese
The Dark Knight Rises	8.5	Christopher Nolan
Whiplash	8.5	Damien Chazelle

In [39]:
```
sns.barplot(data = top10_highrated_mvi, x='Rating', y=top10_highrated_mvi.index,
plt.title('Highest Rated Movie - Titles And Directors')
plt.legend(bbox_to_anchor=[1, 0.5], loc=2)
plt.show()
```



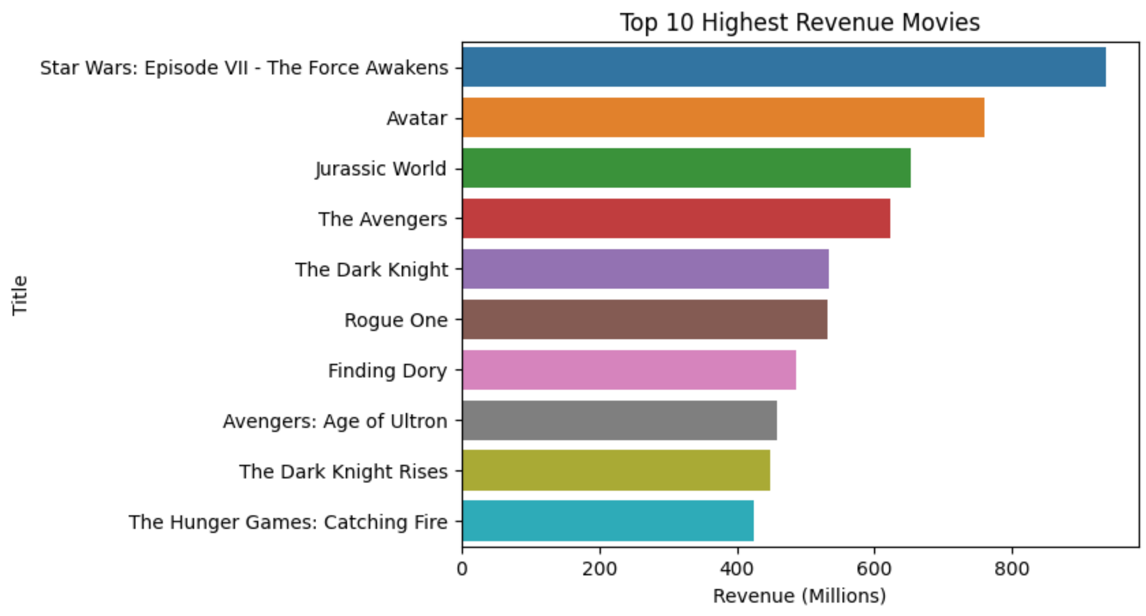
17. Display Top 10 Highest Revenue Movie Titles

In [40]:
```
top10HighRevenue_mvi = data.nlargest(10,'Revenue (Millions)')[['Title','Revenue
top10HighRevenue_mvi
```

Out[40]:

	Revenue (Millions)
Title	
Star Wars: Episode VII - The Force Awakens	936.63
Avatar	760.51
Jurassic World	652.18
The Avengers	623.28
The Dark Knight	533.32
Rogue One	532.17
Finding Dory	486.29
Avengers: Age of Ultron	458.99
The Dark Knight Rises	448.13
The Hunger Games: Catching Fire	424.65

In [41]:
```
sns.barplot(data = top10HighRevenue_mvi, x='Revenue (Millions)' , y = top10HighR
plt.title('Top 10 Highest Revenue Movies')
plt.show()
```



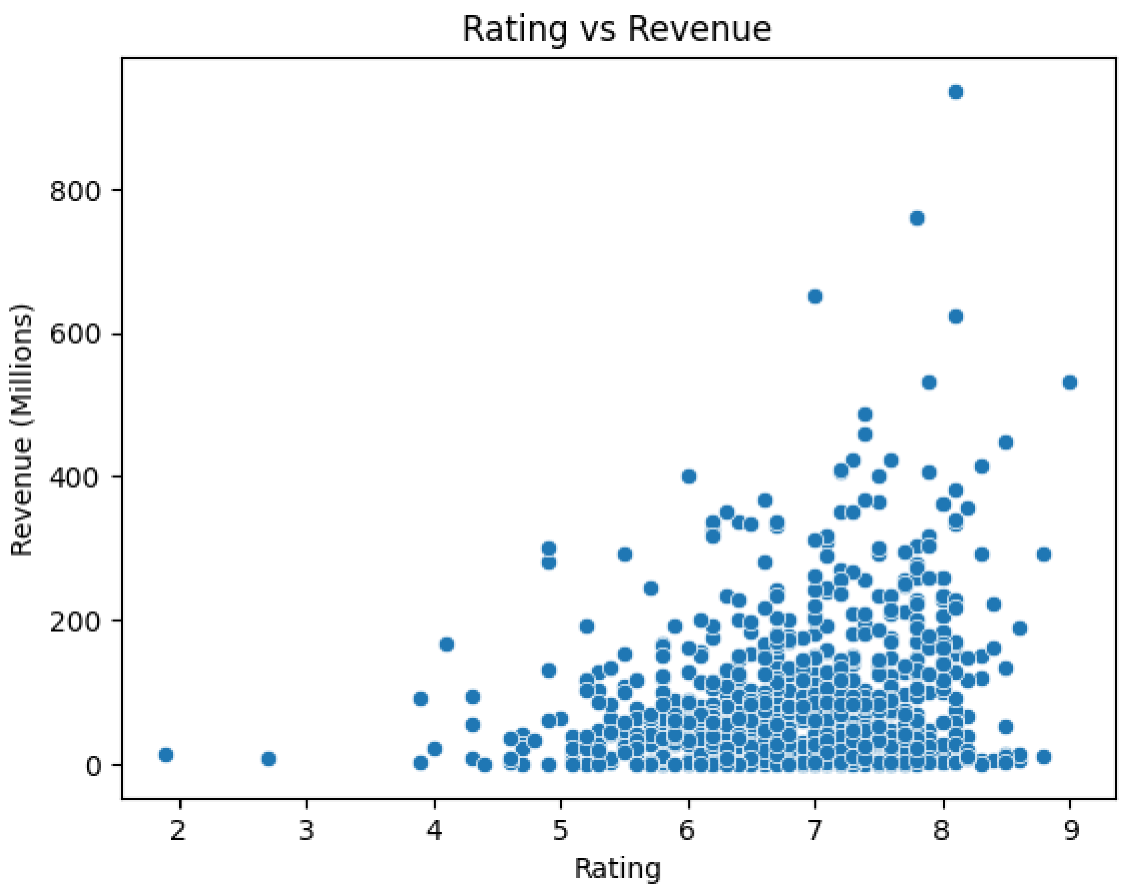
18. Find Average Rating of Movies Year Wise

In [42]:
```python
data.groupby('Year')['Rating'].mean().sort_values(ascending = False)
```

Out[42]:
```
Year
2007    7.133962
2006    7.125000
2009    6.960784
2012    6.925000
2011    6.838095
2014    6.837755
2010    6.826667
2013    6.812088
2008    6.784615
2015    6.602362
2016    6.436700
Name: Rating, dtype: float64
```

19. Does Rating Affect The Revenue?

In [43]:
```python
sns.scatterplot(data = data, x='Rating', y= 'Revenue (Millions)' )
plt.title('Rating vs Revenue')
plt.show()
```



Yes, Rating directly affects the Revenue.

20. Classify Movies Based on Ratings [Excellent, Good, and Average]

In [44]:
```
def rating(rating):
    if rating >= 7.0:
        return 'Excellent'
    elif rating >= 6.0:
        return 'Good'
    else:
        return 'Average'
```

In [45]:
```
data['Rating_Cat'] = data['Rating'].apply(rating)
data.head(10)
```

Out[45]:

	Rank	Title	Genre	Description	Director	Actors	Y
0	1	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2
1	2	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2
2	3	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2
3	4	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2
4	5	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	2
5	6	The Great Wall	Action,Adventure,Fantasy	European mercenaries searching for black powde...	Yimou Zhang	Matt Damon, Tian Jing, Willem Dafoe, Andy Lau	2
6	7	La La Land	Comedy,Drama,Music	A jazz pianist falls for an aspiring actress i...	Damien Chazelle	Ryan Gosling, Emma Stone, Rosemarie DeWitt, J....	2
7	8	Mindhorn	Comedy	A has-been actor best known for playing the ti...	Sean Foley	Essie Davis, Andrea Riseborough, Julian Barrat...	2
8	9	The Lost City of Z	Action,Adventure,Biography	A true-life drama, centering on British explor...	James Gray	Charlie Hunnam, Robert Pattinson, Sienna Mille...	2
9	10	Passengers	Adventure,Drama,Romance	A spacecraft traveling to a distant colony pla...	Morten Tyldum	Jennifer Lawrence, Chris Pratt, Michael Sheen,...	2

21. Count Number of Action Movies

In [46]:
```
len(data[data['Genre'].str.contains('Action', case = False)])
```

Out[46]:
```
303
```

22. Find Unique Values From Genre

In [47]:
```
data['Genre']
```

Out[47]:
```
0       Action,Adventure,Sci-Fi
1      Adventure,Mystery,Sci-Fi
2               Horror,Thriller
3       Animation,Comedy,Family
4      Action,Adventure,Fantasy
                 ...
995        Crime,Drama,Mystery
996                     Horror
997        Drama,Music,Romance
998           Adventure,Comedy
999      Comedy,Family,Fantasy
Name: Genre, Length: 1000, dtype: object
```

In [56]:
```
list1 = []
for value in data['Genre']:
    list1.append(value.split(','))
```

In [57]:
```
list_d =[]
for item in list1:
    for item1 in item:
        list_d.append(item1)
```

In [58]:
```
uni_list = []
for item in list_d:
    if item not in uni_list:
        uni_list.append(item)
```

In [59]:
```
uni_list
```

```
Out[59]: ['Action',
          'Adventure',
          'Sci-Fi',
          'Mystery',
          'Horror',
          'Thriller',
          'Animation',
          'Comedy',
          'Family',
          'Fantasy',
          'Drama',
          'Music',
          'Biography',
          'Romance',
          'History',
          'Crime',
          'Western',
          'War',
          'Musical',
          'Sport']
```

23. How Many Films of Each Genre Were Made?

```
In [54]: from collections import Counter
```

```
In [55]: Counter(list_d)
```

```
Out[55]: Counter({'Action': 303,
                  'Adventure': 259,
                  'Sci-Fi': 120,
                  'Mystery': 106,
                  'Horror': 119,
                  'Thriller': 195,
                  'Animation': 49,
                  'Comedy': 279,
                  'Family': 51,
                  'Fantasy': 101,
                  'Drama': 513,
                  'Music': 16,
                  'Biography': 81,
                  'Romance': 141,
                  'History': 29,
                  'Crime': 150,
                  'Western': 7,
                  'War': 13,
                  'Musical': 5,
                  'Sport': 18})
```

```
In [ ]:
```