Explanatory Data Analysis & Data Presentation (Movies Dataset)

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
pd.options.display.max_columns = 30
pd.options.display.float_format = '{:.2f}'.format
```

In [2]:
```
df = pd.read_csv("X:\Data Science\Projects\Work with Pandas, SQL Databases
```

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

Out[3]:

	id	title	tagline	release_date	genres	belongs_to_collection	orig
0	862	Toy Story	NaN	1995-10-30	Animation|Comedy|Family	Toy Story Collection	
1	8844	Jumanji	Roll the dice and unleash the excitement!	1995-12-15	Adventure|Fantasy|Family	NaN	
2	15602	Grumpier Old Men	Still Yelling. Still Fighting. Still Ready for...	1995-12-22	Romance|Comedy	Grumpy Old Men Collection	
3	31357	Waiting to Exhale	Friends are the people who let you be yourself...	1995-12-22	Comedy|Drama|Romance	NaN	
4	11862	Father of the Bride Part II	Just When His World Is Back To Normal... He's ...	1995-02-10	Comedy	Father of the Bride Collection	

In [4]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44691 entries, 0 to 44690
Data columns (total 22 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   id                     44691 non-null  int64
 1   title                  44691 non-null  object
 2   tagline                20284 non-null  object
 3   release_date           44657 non-null  object
 4   genres                 42586 non-null  object
 5   belongs_to_collection  4463 non-null   object
 6   original_language      44681 non-null  object
 7   budget_musd            8854 non-null   float64
 8   revenue_musd           7385 non-null   float64
 9   production_companies   33356 non-null  object
 10  production_countries   38835 non-null  object
 11  vote_count             44691 non-null  float64
 12  vote_average           42077 non-null  float64
 13  popularity             44691 non-null  float64
 14  runtime                43179 non-null  float64
 15  overview               43740 non-null  object
 16  spoken_languages       41094 non-null  object
 17  poster_path            44467 non-null  object
 18  cast                   42502 non-null  object
 19  cast_size              44691 non-null  int64
 20  crew_size              44691 non-null  int64
 21  director               43960 non-null  object
dtypes: float64(6), int64(3), object(13)
memory usage: 7.5+ MB
```

In [5]:
```
1 df["release_date"] = pd.to_datetime(df['release_date'])
```

In [6]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44691 entries, 0 to 44690
Data columns (total 22 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   id                     44691 non-null  int64
 1   title                  44691 non-null  object
 2   tagline                20284 non-null  object
 3   release_date           44657 non-null  datetime64[ns]
 4   genres                 42586 non-null  object
 5   belongs_to_collection  4463 non-null   object
 6   original_language      44681 non-null  object
 7   budget_musd            8854 non-null   float64
 8   revenue_musd           7385 non-null   float64
 9   production_companies   33356 non-null  object
 10  production_countries   38835 non-null  object
 11  vote_count             44691 non-null  float64
 12  vote_average           42077 non-null  float64
 13  popularity             44691 non-null  float64
 14  runtime                43179 non-null  float64
 15  overview               43740 non-null  object
 16  spoken_languages       41094 non-null  object
 17  poster_path            44467 non-null  object
 18  cast                   42502 non-null  object
 19  cast_size              44691 non-null  int64
 20  crew_size              44691 non-null  int64
 21  director               43960 non-null  object
dtypes: datetime64[ns](1), float64(6), int64(3), object(12)
memory usage: 7.5+ MB
```

In [7]:
```
1 df.genres[1]
```

Out[7]:
```
'Adventure|Fantasy|Family'
```

In [8]:
```
1 df.cast[1]
```

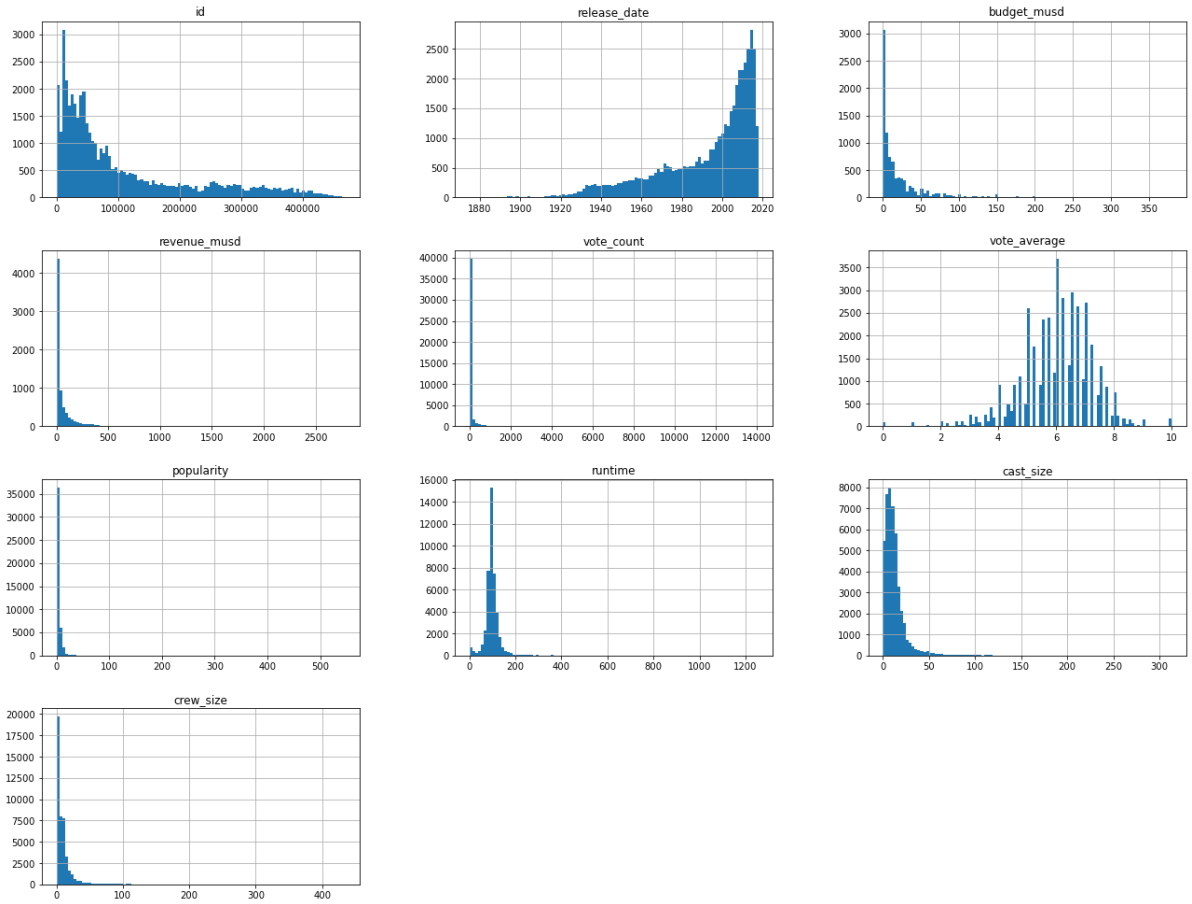
Out[8]:
```
'Robin Williams|Jonathan Hyde|Kirsten Dunst|Bradley Pierce|Bonnie Hunt|Bebe N
euwirth|David Alan Grier|Patricia Clarkson|Adam Hann-Byrd|Laura Bell Bundy|Ja
mes Handy|Gillian Barber|Brandon Obray|Cyrus Thiedeke|Gary Joseph Thorup|Leon
ard Zola|Lloyd Berry|Malcolm Stewart|Annabel Kershaw|Darryl Henriques|Robyn D
riscoll|Peter Bryant|Sarah Gilson|Florica Vlad|June Lion|Brenda Lockmuller'
```

In [9]:
```
1 df.describe()
```

Out[9]:

	id	budget_musd	revenue_musd	vote_count	vote_average	popularity	runtime
count	44691.00	8854.00	7385.00	44691.00	42077.00	44691.00	43179.00
mean	107186.24	21.67	68.97	111.65	6.00	2.96	97.57
std	111806.36	34.36	146.61	495.32	1.28	6.04	34.65
min	2.00	0.00	0.00	0.00	0.00	0.00	1.00
25%	26033.50	2.00	2.41	3.00	5.30	0.40	86.00
50%	59110.00	8.20	16.87	10.00	6.10	1.15	95.00
75%	154251.00	25.00	67.64	35.00	6.80	3.77	107.00
max	469172.00	380.00	2787.97	14075.00	10.00	547.49	1256.00

In [10]:
```
1 df.hist(figsize = (22,17), bins = 100)
2 plt.show()
```



In [11]:
```
1 df.budget_musd.value_counts(dropna = False).head(20)
```

Out[11]:
```
NaN      35837
5.00       286
10.00      258
20.00      243
2.00       241
15.00      226
3.00       220
25.00      206
1.00       195
30.00      189
4.00       180
6.00       173
12.00      171
40.00      166
8.00       155
35.00      141
0.50       141
7.00       137
50.00      124
```

In [12]:
```
1 df.revenue_musd.value_counts(dropna = False).head(20)
```

Out[12]:
```
NaN      37306
12.00       20
10.00       19
11.00       19
2.00        18
6.00        17
5.00        14
0.50        13
8.00        13
0.00        12
14.00       12
7.00        11
3.00        10
1.00        10
20.00       10
0.00         9
1.50         9
4.00         9
30.00        8
4.10         8
Name: revenue_musd, dtype: int64
```

In [13]:
```
1 df.vote_average.value_counts(dropna = False)
```

Out[13]:
```
NaN     2614
6.00    2421
5.00    1969
7.00    1859
6.50    1705
        ...
9.40       3
9.60       1
9.80       1
0.70       1
1.10       1
Name: vote_average, Length: 93, dtype: int64
```

In [14]:
```
1 df.vote_count.value_counts()
```

Out[14]:
```
1.00       3186
2.00       3056
3.00       2729
0.00       2614
4.00       2442
           ...
2755.00       1
1187.00       1
4200.00       1
3322.00       1
2712.00       1
Name: vote_count, Length: 1820, dtype: int64
```

In [15]:
```
1 df.describe(include = "object")
```

Out[15]:

	title	tagline	genres	belongs_to_collection	original_language	production_companie
count	44691	20284	42586	4463	44681	3335
unique	41605	20171	4044	1691	89	2253
top	Cinderella	Based on a true story.	Drama	The Bowery Boys	en	Metro-Goldwyn-Maye (MGM
freq	11	7	4935	29	31755	74

In [16]:
```
1 df[df.title == "Cinderella"]
```

Out[16]:

	id	title	tagline	release_date	genres	belongs_to
984	11224	Cinderella	The greatest love story ever told.	1950-03-04	Family|Fantasy|Animation|Romance	Cinderell
12988	42884	Cinderella	NaN	1997-11-02	TV Movie|Family|Fantasy|Music|Romance	
23254	92349	Cinderella	NaN	1914-12-28	Fantasy|Drama	
23265	105875	Cinderella	The version children love!	2002-08-06	Animation|Family|Fantasy	
28073	261985	Cinderella	NaN	2011-10-30	Family|TV Movie	
28340	150689	Cinderella	Midnight is just the beginning.	2015-03-12	Romance|Fantasy|Family|Drama	
33805	42651	Cinderella	NaN	1947-11-28	Comedy|Family|Fantasy	
35114	44459	Cinderella	NaN	1957-03-31	Drama|Romance	
35116	289673	Cinderella	NaN	2000-01-01	NaN	
40439	114108	Cinderella	NaN	1899-10-01	Fantasy|Horror|Science Fiction|Family	
44633	98604	Cinderella	NaN	2012-02-14	Comedy|Romance	

The best and the worst movies

In [17]:
```
from IPython.display import HTML
```

In [18]:
```
df_best = df[["poster_path","title","budget_musd", "revenue_musd","vote_co
df_best
```

Out[18]:

	poster_path	title	budget_musd	revenue_musd	vote_co
0	<img src='http://image.tmdb.org/t/p/w185//uXDf...	Toy Story	30.00	373.55	5415
1	<img src='http://image.tmdb.org/t/p/w185//vgpX...	Jumanji	65.00	262.80	2413
2	<img src='http://image.tmdb.org/t/p/w185//1FSX...	Grumpier Old Men	NaN	NaN	92
3	<img src='http://image.tmdb.org/t/p/w185//4wjG...	Waiting to Exhale	16.00	81.45	34
4	<img src='http://image.tmdb.org/t/p/w185//lf9R...	Father of the Bride Part II	NaN	76.58	173
...	...	...	...	...	
44686	<img src='http://image.tmdb.org/t/p/w185//pfC8...	Subdue	NaN	NaN	1
44687	<img src='http://image.tmdb.org/t/p/w185//xZkm...	Century of Birthing	NaN	NaN	3
44688	<img src='http://image.tmdb.org/t/p/w185//eGga...	Betrayal	NaN	NaN	6
44689	<img src='http://image.tmdb.org/t/p/w185//aorB...	Satan Triumphant	NaN	NaN	0
44690	<img src='http://image.tmdb.org/t/p/w185//oxFE...	Queerama	NaN	NaN	0

44691 rows × 7 columns

In [19]:
```
df_best["profit_musd"] = df.revenue_musd.sub(df.budget_musd)
df_best["return"] = df.revenue_musd.div(df.budget_musd)
```

In [20]:
```
df_best
```

Out[20]:

	poster_path	title	budget_musd	revenue_musd	vote_co
0	<img src='http://image.tmdb.org/t/p/w185//uXDf...	Toy Story	30.00	373.55	5415
1	<img src='http://image.tmdb.org/t/p/w185//vgpX...	Jumanji	65.00	262.80	2413
2	<img src='http://image.tmdb.org/t/p/w185//1FSX...	Grumpier Old Men	NaN	NaN	92
3	<img src='http://image.tmdb.org/t/p/w185//4wjG...	Waiting to Exhale	16.00	81.45	34
4	<img src='http://image.tmdb.org/t/p/w185//lf9R...	Father of the Bride Part II	NaN	76.58	173
...	...	...	...	...	
44686	<img src='http://image.tmdb.org/t/p/w185//pfC8...	Subdue	NaN	NaN	1
44687	<img src='http://image.tmdb.org/t/p/w185//xZkm...	Century of Birthing	NaN	NaN	3
44688	<img src='http://image.tmdb.org/t/p/w185//eGga...	Betrayal	NaN	NaN	6
44689	<img src='http://image.tmdb.org/t/p/w185//aorB...	Satan Triumphant	NaN	NaN	0
44690	<img src='http://image.tmdb.org/t/p/w185//oxFE...	Queerama	NaN	NaN	0

44691 rows × 9 columns

In [21]:
```
df_best.columns  = ["", "Title", "Budget", "Revenue", "Votes", "Average Ra
```

In [22]:
```
df_best.set_index("Title", inplace = True)
```

In [23]:
```
df_best
```

Out[23]:

Title		Budget	Revenue	Votes	Average Rating	Popular
Toy Story	<img src='http://image.tmdb.org/t/p/w185//uXDf...	30.00	373.55	5415.00	7.70	21.
Jumanji	<img src='http://image.tmdb.org/t/p/w185//vgpX...	65.00	262.80	2413.00	6.90	17.
Grumpier Old Men	<img src='http://image.tmdb.org/t/p/w185//1FSX...	NaN	NaN	92.00	6.50	11.
Waiting to Exhale	<img src='http://image.tmdb.org/t/p/w185//4wjG...	16.00	81.45	34.00	6.10	3.
Father of the Bride Part II	<img src='http://image.tmdb.org/t/p/w185//lf9R...	NaN	76.58	173.00	5.70	8.
...	...	...	...	...	...	
Subdue	<img src='http://image.tmdb.org/t/p/w185//pfC8...	NaN	NaN	1.00	4.00	0.
Century of Birthing	<img src='http://image.tmdb.org/t/p/w185//xZkm...	NaN	NaN	3.00	9.00	0.
Betrayal	<img src='http://image.tmdb.org/t/p/w185//eGga...	NaN	NaN	6.00	3.80	0.
Satan Triumphant	<img src='http://image.tmdb.org/t/p/w185//aorB...	NaN	NaN	0.00	NaN	0.
Queerama	<img src='http://image.tmdb.org/t/p/w185//oxFE...	NaN	NaN	0.00	NaN	0.

44691 rows × 8 columns

In [24]:
```
df_best.iloc[0,0]
```

Out[24]:
```
"<img src='http://image.tmdb.org/t/p/w185//uXDfjJbdP4ijW5hWSBrPrlKpxab.jpg' s
tyle='height:100px;'>"
```

In [25]:
```
subset = df_best.iloc[0:5,0:2]
subset
```

Out[25]:

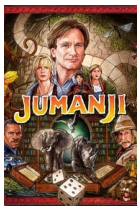
Title		Budget
Toy Story	<img src='http://image.tmdb.org/t/p/w185//uXDf...	30.00
Jumanji	<img src='http://image.tmdb.org/t/p/w185//vgpX...	65.00
Grumpier Old Men	<img src='http://image.tmdb.org/t/p/w185//1FSX...	NaN
Waiting to Exhale	<img src='http://image.tmdb.org/t/p/w185//4wjG...	16.00
Father of the Bride Part II	<img src='http://image.tmdb.org/t/p/w185//lf9R...	NaN

In [26]:
```
1 subset.to_html(escape = False)
```

Out[26]:
```
'<table border="1" class="dataframe">\n  <thead>\n    <tr style="text-align:
right;">\n      <th></th>\n      <th></th>\n      <th>Budget</th>\n    </tr>
\n    <tr>\n      <th>Title</th>\n      <th></th>\n      <th></th>\n    </tr>
\n  </thead>\n  <tbody>\n    <tr>\n      <th>Toy Story</th>\n      <td><img s
rc=\'http://image.tmdb.org/t/p/w185//uXDfjJbdP4ijW5hWSBrPrlKpxab.jpg\' style=
\'height:100px;\'></td>\n      <td>30.00</td>\n    </tr>\n    <tr>\n      <th
>Jumanji</th>\n      <td><img src=\'http://image.tmdb.org/t/p/w185//vgpXmVaVy
UL7GGiDeiK1mKEKzcX.jpg\' style=\'height:100px;\'></td>\n      <td>65.00</td>
\n    </tr>\n    <tr>\n      <th>Grumpier Old Men</th>\n      <td><img src=
\'http://image.tmdb.org/t/p/w185//1FSXpj5e8l4KH6nVFO5SPUeraOt.jpg\' style=\'h
eight:100px;\'></td>\n      <td>NaN</td>\n    </tr>\n    <tr>\n      <th>Wait
ing to Exhale</th>\n      <td><img src=\'http://image.tmdb.org/t/p/w185//4wjG
MwPsdlvi025ZqR4rXnFDvBz.jpg\' style=\'height:100px;\'></td>\n      <td>16.00
</td>\n    </tr>\n    <tr>\n      <th>Father of the Bride Part II</th>\n
<td><img src=\'http://image.tmdb.org/t/p/w185//lf9RTErt8BSLQy98aSFblElvsCQ.jp
g\' style=\'height:100px;\'></td>\n      <td>NaN</td>\n    </tr>\n  </tbody>
\n</table>'
```

In [27]:
```
1 HTML(subset.to_html(escape = False))
```

Out[27]:

Title		Budget
Toy Story		30.00
Jumanji		65.00
Grumpier Old Men		NaN
Waiting to Exhale		16.00
Father of the Bride Part II		NaN

In [28]:
```
1 df_best.sort_values(by = 'Average Rating', ascending = False)
```

Out[28]:

Title		Budget	Revenue	Votes	Average Rating	Popula
Portrait of a Young Man in Three Movements	NaN	NaN	NaN	1.00	10.00	0
Brave Revolutionary	<img src='http://image.tmdb.org/t/p/w185//zAb2...	NaN	NaN	1.00	10.00	0
Other Voices Other Rooms	<img src='http://image.tmdb.org/t/p/w185//4ifP...	NaN	NaN	1.00	10.00	0
The Lion of Thebes	<img src='http://image.tmdb.org/t/p/w185//tdOc...	NaN	NaN	1.00	10.00	1
Katt Williams: Priceless: Afterlife	<img src='http://image.tmdb.org/t/p/w185//wKrH...	NaN	NaN	2.00	10.00	0
...	...	...	...	...	...	
Altar of Fire	<img src='http://image.tmdb.org/t/p/w185//iJ78...	NaN	NaN	0.00	NaN	0
The Wonders of Aladdin	<img src='http://image.tmdb.org/t/p/w185//AvfX...	NaN	NaN	0.00	NaN	0
Deep Hearts	<img src='http://image.tmdb.org/t/p/w185//8jI4...	NaN	NaN	0.00	NaN	0
Satan Triumphant	<img src='http://image.tmdb.org/t/p/w185//aorB...	NaN	NaN	0.00	NaN	0
Queerama	<img src='http://image.tmdb.org/t/p/w185//oxFE...	NaN	NaN	0.00	NaN	0

44691 rows × 8 columns

In [29]:
```
df_best.sort_values(by = "ROI", ascending = False)
```

Out[29]:

Title		Budget	Revenue	Votes	Average Rating	Popular
Less Than Zero	<img src='http://image.tmdb.org/t/p/w185//1GY0...	0.00	12.40	77.00	6.10	4
Modern Times	<img src='http://image.tmdb.org/t/p/w185//7uoi...	0.00	8.50	881.00	8.10	8
Welcome to Dongmakgol	<img src='http://image.tmdb.org/t/p/w185//5iGV...	0.00	33.58	49.00	7.70	4
Aquí Entre Nos	<img src='http://image.tmdb.org/t/p/w185//oflx...	0.00	2.76	3.00	6.00	0
The Karate Kid, Part II	<img src='http://image.tmdb.org/t/p/w185//mSne...	0.00	115.10	457.00	5.90	9
...	...	...	...	...	...	
Subdue	<img src='http://image.tmdb.org/t/p/w185//pfC8...	NaN	NaN	1.00	4.00	0
Century of Birthing	<img src='http://image.tmdb.org/t/p/w185//xZkm...	NaN	NaN	3.00	9.00	0
Betrayal	<img src='http://image.tmdb.org/t/p/w185//eGga...	NaN	NaN	6.00	3.80	0
Satan Triumphant	<img src='http://image.tmdb.org/t/p/w185//aorB...	NaN	NaN	0.00	NaN	0
Queerama	<img src='http://image.tmdb.org/t/p/w185//oxFE...	NaN	NaN	0.00	NaN	0

44691 rows × 8 columns

In [30]:
```
df_best.loc[df_best.Budget >= 5]
```

Out[30]:

Title		Budget	Revenue	Votes	Average Rating	Popularity
Toy Story	<img src='http://image.tmdb.org/t/p/w185//uXDf...	30.00	373.55	5415.00	7.70	21.95
Jumanji	<img src='http://image.tmdb.org/t/p/w185//vgpX...	65.00	262.80	2413.00	6.90	17.02
Waiting to Exhale	<img src='http://image.tmdb.org/t/p/w185//4wjG...	16.00	81.45	34.00	6.10	3.86
Heat	<img src='http://image.tmdb.org/t/p/w185//lbf2...	60.00	187.44	1886.00	7.70	17.92
Sabrina	<img src='http://image.tmdb.org/t/p/w185//z1oN...	58.00	NaN	141.00	6.20	6.68
...	...	...	...	...	...	..
Sivaji: The Boss	<img src='http://image.tmdb.org/t/p/w185//nqS3...	12.00	19.00	25.00	6.90	1.32
Good Guys Go to Heaven, Bad Guys Go to Pattaya	<img src='http://image.tmdb.org/t/p/w185//cfVB...	5.40	NaN	153.00	5.30	5.61
The Visitors: Bastille Day	<img src='http://image.tmdb.org/t/p/w185//kBlm...	25.87	NaN	167.00	4.00	7.29
House of the Long Shadows	<img src='http://image.tmdb.org/t/p/w185//21ZG...	7.50	NaN	18.00	6.10	1.19
Antidur	<img src='http://image.tmdb.org/t/p/w185//f5XO...	5.00	1.41	1.00	1.00	0.04

5449 rows × 8 columns

In [31]:
```
df_best.Budget.fillna(0, inplace = True)
df_best.Votes.fillna(0, inplace = True)
```

In [32]:
```
1 df_best.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 44691 entries, Toy Story to Queerama
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0                   44467 non-null  object 
 1   Budget          44691 non-null  float64
 2   Revenue         7385 non-null   float64
 3   Votes           44691 non-null  float64
 4   Average Rating  42077 non-null  float64
 5   Popularity      44691 non-null  float64
 6   Profit          5371 non-null   float64
 7   ROI             5371 non-null   float64
dtypes: float64(7), object(1)
memory usage: 3.1+ MB
```

In [33]:
```
1 def best_worst(n, by, ascending = False, min_bud = 0, min_votes = 0):
2 
3     df2 = df_best.loc[(df_best.Budget >= min_bud) & (df_best.Votes >= min_
4                       ["", by]].sort_values(by, ascending = ascending).head
5     return HTML(df2.to_html(escape = False))
```

Movies Top 5 - Highest Revenue

In [34]:
```
1 best_worst(n = 5, by = 'Revenue')
```

Out[34]:

Title		Revenue
Avatar		2787.97
Star Wars: The Force Awakens		2068.22
Titanic		1845.03
The Avengers		1519.56
Jurassic World		1513.53

Movies Top 5 Highest Budget

In [35]:
```
1 best_worst(n = 5, by = 'Budget')
```

Out[35]:

Title		Budget
Pirates of the Caribbean: On Stranger Tides		380.00
Pirates of the Caribbean: At World's End		300.00
Avengers: Age of Ultron		280.00
Superman Returns		270.00
Transformers: The Last Knight		260.00

MoviesTop 5 - Highest Profit

In [36]:
```
1 best_worst(5, "Profit")
```

Out[36]:

Title		Profit
Avatar		2550.97
Star Wars: The Force Awakens		1823.22
Titanic		1645.03
Jurassic World		1363.53
Furious 7		1316.25

Movies Top 5 - Lowest Profit

In [37]:
```
1 best_worst(5, "Profit", ascending = True)
```

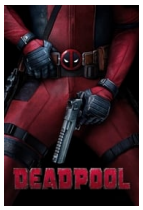
Out[37]:

Title		Profit
The Lone Ranger		-165.71
The Alamo		-119.18
Mars Needs Moms		-111.01
Valerian and the City of a Thousand Planets		-107.45
The 13th Warrior		-98.30

Movies Top 5 - Highest ROI

In [38]:
```
1 best_worst(5, "ROI", min_bud = 50)
```

Out[38]:

Title		ROI
Minions		15.63
Jurassic Park		14.60
The Twilight Saga: New Moon		14.20
Deadpool		13.50
Despicable Me 2		12.77

Movies Top - Lowest ROI

In [39]:
```
best_worst(5, "ROI", ascending = True, min_bud = 100)
```

Out[39]:

Title		ROI
The Adventures of Pluto Nash		0.07
The Alamo		0.18
Live by Night		0.21
Mars Needs Moms		0.26
The Lone Ranger		0.35

Movies Top 5 - Most Votes

In [40]:
```
1 best_worst(6, "Votes")
```

Out[40]:

Title		Votes
Inception		14075.00
The Dark Knight		12269.00
Avatar		12114.00
The Avengers		12000.00
Deadpool		11444.00
Interstellar		11187.00

Movies Top 5 Highest Rating

In [41]:
```
1 best_worst(5,"Average Rating", min_votes = 2000)
```

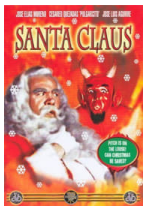
Out[41]:

Title		Average Rating
The Shawshank Redemption		8.50
The Godfather		8.50
Life Is Beautiful		8.30
Spirited Away		8.30
One Flew Over the Cuckoo's Nest		8.30

Movies Top 5 - Lowest Rating

In [42]:
```
best_worst(5, "Average Rating", ascending = True, min_votes = 10)
```

Out[42]:

Title		Average Rating
Call Me by Your Name		0.00
Extinction: Nature Has Evolved		0.00
How to Talk to Girls at Parties		0.00
Santa Claus		1.60
The Beast of Yucca Flats		1.60

Movies Top 5 - Most Popular

In [43]:
```
1 best_worst(5, "Popularity")
```

Out[43]:

Title		Popularity
Minions		547.49
Wonder Woman		294.34
Beauty and the Beast		287.25
Baby Driver		228.03
Big Hero 6		213.85

Find Your Next Movie

Search 1 : Science Fiction Action Movie with Bruce Willis(High Rating)

In [44]:
```
1 df.genres[0]
```

Out[44]:
```
'Animation|Comedy|Family'
```

In [45]:
```
mask_genres = df.genres.str.contains("Action") & df.genres.str.contains("S
mask_genres
```

Out[45]:
```
0        False
1        False
2        False
3        False
4        False
         ...
44686    False
44687    False
44688    False
44689    False
44690    False
Name: genres, Length: 44691, dtype: bool
```

In [46]:
```
df.cast[0]
```

Out[46]:
```
'Tom Hanks|Tim Allen|Don Rickles|Jim Varney|Wallace Shawn|John Ratzenberger|A
nnie Potts|John Morris|Erik von Detten|Laurie Metcalf|R. Lee Ermey|Sarah Free
man|Penn Jillette'
```

In [47]:
```
mask_actor = df.cast.str.contains("Bruce Willis")
mask_actor
```

Out[47]:
```
0        False
1        False
2        False
3        False
4        False
         ...
44686    False
44687    False
44688    False
44689    False
44690      NaN
Name: cast, Length: 44691, dtype: object
```

In [48]:
```
df.loc[mask_actor & mask_genres, ["title","vote_average"]].sort_values(by
```

Out[48]:

	title	vote_average
1448	The Fifth Element	7.30
19218	Looper	6.60
1786	Armageddon	6.50
14135	Surrogates	5.90
20333	G.I. Joe: Retaliation	5.40
27619	Vice	4.10

In [49]:
```
bruce = df.loc[mask_actor & mask_genres, ["title","poster_path","vote_aver
```

In [50]:
```
HTML(bruce.to_html(escape = False))
```

Out[50]:

	title	poster_path	vote_average
1448	The Fifth Element		7.30
19218	Looper		6.60
1786	Armageddon		6.50
14135	Surrogates		5.90
20333	G.I. Joe: Retaliation		5.40
27619	Vice		4.10

Search 2: Movies with Uma Thurma and directed by Quentin Tarantio(low runtime)

In [51]:
```
df.director
```

Out[51]:
```
0           John Lasseter
1            Joe Johnston
2          Howard Deutch
3         Forest Whitaker
4           Charles Shyer
               ...
44686    Hamid Nematollah
44687            Lav Diaz
44688      Mark L. Lester
44689    Yakov Protazanov
44690       Daisy Asquith
Name: director, Length: 44691, dtype: object
```

In [52]:
```
mask_director = df.director == "Quentin Tarantino"
mask_director
```

Out[52]:
```
0        False
1        False
2        False
3        False
4        False
         ...
44686    False
44687    False
44688    False
44689    False
44690    False
Name: director, Length: 44691, dtype: bool
```

In [53]:
```
mask_actor = df.cast.str.contains("Uma Thurman")
```

In [54]:
```
quentin = df.loc[mask_director & mask_actor,["title", "poster_path",'runti
```

In [55]:
```
1 HTML(quentin.to_html(escape = False))
```

Out[55]:

title	poster_path	runtime
Kill Bill: Vol. 1		111.00
Kill Bill: Vol. 2		136.00
Pulp Fiction		154.00

Search 3: Most Successful Pixar Movies between 2010 and 2015 (high Revenue)

In [56]:
```
1 df.production_companies[1]
```

Out[56]:
```
'TriStar Pictures|Teitler Film|Interscope Communications'
```

In [57]:
```
1 mask_studio = df.production_companies.str.contains("Pixar").fillna(False)
2 mask_studio
```

Out[57]:
```
0         True
1        False
2        False
3        False
4        False
         ...
44686    False
44687    False
44688    False
44689    False
44690    False
Name: production_companies, Length: 44691, dtype: bool
```

In [58]:
```
1 df.release_date
```

Out[58]:
```
0       1995-10-30
1       1995-12-15
2       1995-12-22
3       1995-12-22
4       1995-02-10
           ...
44686         NaT
44687   2011-11-17
44688   2003-08-01
44689   1917-10-21
44690   2017-06-09
Name: release_date, Length: 44691, dtype: datetime64[ns]
```

In [59]:
```
1 mask_time = df.release_date.between("2010-01-01","2015-12-31")
```

In [60]:
```
1 pixar = df.loc[mask_studio & mask_time, ["title", "poster_path", "revenue_
```

In [61]:
```
1 HTML(pixar.to_html(escape= False))
```

Out[61]:

	title	poster_path	revenue_musd	release_date
15236	Toy Story 3		1066.97	2010-06-16
29957	Inside Out		857.61	2015-06-09
20888	Monsters University		743.56	2013-06-20

Search 4: Action or Thriller Movies with original language English and minimum Rating of & 7.5 (most recent)

In [62]:
```
df.head(1)
```

Out[62]:

	id	title	tagline	release_date	genres	belongs_to_collection	original_langu
0	862	Toy Story	NaN	1995-10-30	Animation|Comedy|Family	Toy Story Collection	

In [63]:
```
mask_genres = df.genres.str.contains("Action") | df.genres.str.contains("T
mask_genres.head(2)
```

Out[63]:
```
0    False
1    False
Name: genres, dtype: bool
```

In [64]:
```
mask_original_language = df.original_language == "en"
mask_original_language.head(2)
```

Out[64]:
```
0    True
1    True
Name: original_language, dtype: bool
```

In [65]:
```
mask_vote_average = df.vote_average >= 7.5
mask_vote_average.head(2)
```

Out[65]:
```
0     True
1    False
Name: vote_average, dtype: bool
```

In [66]:
```
mask_vote_count = df.vote_count >= 10
mask_vote_count.head(2)
```

Out[66]:
```
0    True
1    True
Name: vote_count, dtype: bool
```

In [67]:
```
next_movie = df.loc[mask_genres & mask_original_language & mask_vote_avera
```

In [68]:
```
1 HTML(next_movie.to_html(escape = False))
```

Out[68]:

title	poster_path	genres	vote_average	vote_count	release_da
London Spy		Romance|Crime|Drama|Mystery|Thriller	8.80	12.00	2015-11-
I Am So Proud of You		Animation|Action|Thriller|Science Fiction	8.30	12.00	2008-01-
Pulp Fiction		Thriller|Crime	8.30	8670.00	1994-09-
Black Mirror: White Christmas		Drama|Horror|Mystery|Science Fiction|Thriller|TV Movie	8.30	211.00	2014-12-
The Dark Knight		Drama|Action|Crime|Thriller	8.30	12269.00	2008-07-

What are the most comman Words in Movie Titles and Taglines ?

In [69]:
```
1 from wordcloud import WordCloud
```

In [70]:
```
df.head()
```

Out[70]:

	id	title	tagline	release_date	genres	belongs_to_collection	orig
0	862	Toy Story	NaN	1995-10-30	Animation|Comedy|Family	Toy Story Collection	
1	8844	Jumanji	Roll the dice and unleash the excitement!	1995-12-15	Adventure|Fantasy|Family	NaN	
2	15602	Grumpier Old Men	Still Yelling. Still Fighting. Still Ready for...	1995-12-22	Romance|Comedy	Grumpy Old Men Collection	
3	31357	Waiting to Exhale	Friends are the people who let you be yourself...	1995-12-22	Comedy|Drama|Romance	NaN	
4	11862	Father of the Bride Part II	Just When His World Is Back To Normal... He's ...	1995-02-10	Comedy	Father of the Bride Collection	

In [71]:
```
df.tagline[1]
```

Out[71]:
```
'Roll the dice and unleash the excitement!'
```

In [72]:
```
df.overview[1]
```

Out[72]:
```
"When siblings Judy and Peter discover an enchanted board game that opens the
door to a magical world, they unwittingly invite Alan -- an adult who's been
trapped inside the game for 26 years -- into their living room. Alan's only h
ope for freedom is to finish the game, which proves risky as all three find t
hemselves running from giant rhinoceroses, evil monkeys and other terrifying
creatures."
```

In [73]:
```
title = df.title.dropna()
overview = df.overview.dropna()
tagline = df.tagline.dropna()
```

In [74]:
```
1 title
```

Out[74]:
```
0                          Toy Story
1                            Jumanji
2                   Grumpier Old Men
3                  Waiting to Exhale
4        Father of the Bride Part II
                    ...
44686                          Subdue
44687           Century of Birthing
44688                       Betrayal
44689              Satan Triumphant
44690                       Queerama
Name: title, Length: 44691, dtype: object
```

In [75]:
```
1 ' '.join(title)
```

Out[75]:
```
'Toy Story Jumanji Grumpier Old Men Waiting to Exhale Father of the Bride P
art II Heat Sabrina Tom and Huck Sudden Death GoldenEye The American Presid
ent Dracula: Dead and Loving It Balto Nixon Cutthroat Island Casino Sense a
nd Sensibility Four Rooms Ace Ventura: When Nature Calls Money Train Get Sh
orty Copycat Assassins Powder Leaving Las Vegas Othello Now and Then Persua
sion The City of Lost Children Shanghai Triad Dangerous Minds Twelve Monkey
s Wings of Courage Babe Carrington Dead Man Walking Across the Sea of Time
It Takes Two Clueless Cry, the Beloved Country Richard III Dead Presidents
Restoration Mortal Kombat To Die For How To Make An American Quilt Se7en Po
cahontas When Night Is Falling The Usual Suspects Guardian Angel Mighty Aph
rodite Lamerica The Big Green Georgia Kids of the Round Table Home for the
Holidays The Postman The Confessional The Indian in the Cupboard Eye for an
Eye Mr. Holland\'s Opus Don\'t Be a Menace to South Central While Drinking
Your Juice in the Hood Two If by Sea Bio-Dome Lawnmower Man 2: Beyond Cyber
space Two Bits French Twist Friday From Dusk Till Dawn Fair Game Kicking an
d Screaming Les Miserables Bed of Roses Big Bully Screamers Nico Icon The C
rossing Guard The Juror The White Balloon Things to Do in Denver When You
\'re Dead Antonia\'s Line Once Upon a Time... When We Were Colored Last Sum
mer in the Hamptons Angels and Insects White Squall Dunston Checks In Black
```

In [76]:
```
1 title_corpus = ' '.join(title)
2 overview_corpus = ' '.join(overview)
3 tagline_corpus = ' '.join(tagline)
```

In [77]:
```
1 title_corpus
```

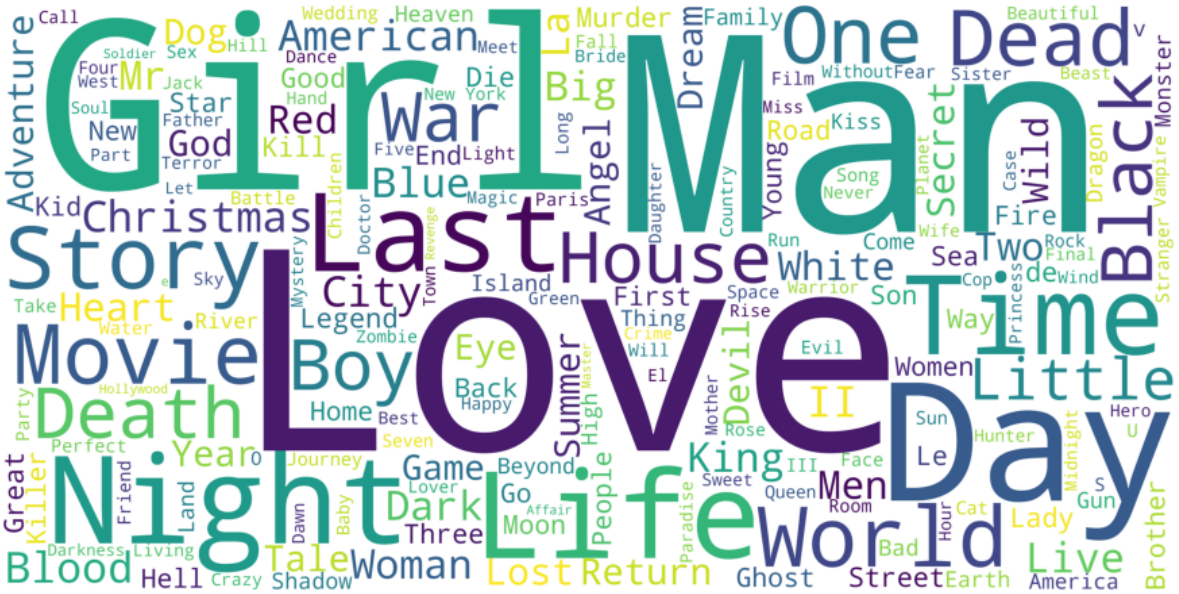
Out[77]:
```
'Toy Story Jumanji Grumpier Old Men Waiting to Exhale Father of the Bride P
art II Heat Sabrina Tom and Huck Sudden Death GoldenEye The American Presid
ent Dracula: Dead and Loving It Balto Nixon Cutthroat Island Casino Sense a
nd Sensibility Four Rooms Ace Ventura: When Nature Calls Money Train Get Sh
orty Copycat Assassins Powder Leaving Las Vegas Othello Now and Then Persua
sion The City of Lost Children Shanghai Triad Dangerous Minds Twelve Monkey
s Wings of Courage Babe Carrington Dead Man Walking Across the Sea of Time
It Takes Two Clueless Cry, the Beloved Country Richard III Dead Presidents
Restoration Mortal Kombat To Die For How To Make An American Quilt Se7en Po
cahontas When Night Is Falling The Usual Suspects Guardian Angel Mighty Aph
rodite Lamerica The Big Green Georgia Kids of the Round Table Home for the
Holidays The Postman The Confessional The Indian in the Cupboard Eye for an
Eye Mr. Holland\'s Opus Don\'t Be a Menace to South Central While Drinking
Your Juice in the Hood Two If by Sea Bio-Dome Lawnmower Man 2: Beyond Cyber
space Two Bits French Twist Friday From Dusk Till Dawn Fair Game Kicking an
d Screaming Les Miserables Bed of Roses Big Bully Screamers Nico Icon The C
rossing Guard The Juror The White Balloon Things to Do in Denver When You
\'re Dead Antonia\'s Line Once Upon a Time... When We Were Colored Last Sum
mer in the Hamptons Angels and Insects White Squall Dunston Checks In Black
```

In [78]:
```
1 title_wordcloud = WordCloud(background_color = 'White', height = 2000, wid
```

In [79]:
```
1 title_wordcloud
```

Out[79]:
```
<wordcloud.wordcloud.WordCloud at 0x280be989430>
```

In [80]:
```
1 plt.figure(figsize =(16,8))
2 plt.imshow(title_wordcloud)
3 plt.axis('off')
4 plt.show()
```



In [81]:
```
tagline_wordcloud = WordCloud(background_color = 'white', height = 2000, w
plt.figure(figsize = (16,8))
plt.imshow(tagline_wordcloud)
plt.axis('off')
plt.show()
```

In [82]:
```
overview_wordcloud = WordCloud(background_color = 'white', height = 2000,
plt.figure(figsize = (16,9))
plt.imshow(overview_wordcloud)
plt.axis('off')
plt.show()
```

Are Franchises more successful?

In [83]:
```
1 df.belongs_to_collection
```

Out[83]:
```
0                  Toy Story Collection
1                                   NaN
2             Grumpy Old Men Collection
3                                   NaN
4        Father of the Bride Collection
                      ...
44686                               NaN
44687                               NaN
44688                               NaN
44689                               NaN
44690                               NaN
Name: belongs_to_collection, Length: 44691, dtype: object
```

In [84]:
```
1 df["Franchise"] = df.belongs_to_collection.notna()
```

In [85]:
```
1 df.Franchise
```

Out[85]:
```
0         True
1        False
2         True
3        False
4         True
         ...
44686    False
44687    False
44688    False
44689    False
44690    False
Name: Franchise, Length: 44691, dtype: bool
```

In [86]:
```
1 df.Franchise.value_counts()
```

Out[86]:
```
False    40228
True      4463
Name: Franchise, dtype: int64
```

Franchise vs Stand-alone: Average Revenue

In [87]:
```
1 df.groupby("Franchise").revenue_musd.mean()
```

Out[87]:
```
Franchise
False    44.74
True    165.71
Name: revenue_musd, dtype: float64
```

Franchise vs Stand-alone: Return on Investment / Profitability

In [88]:
```
df["ROI"] = df["revenue_musd"] / df["budget_musd"]
```

In [89]:
```
df.ROI
```

Out[89]:
```
0       12.45
1        4.04
2         NaN
3        5.09
4         NaN
         ...
44686     NaN
44687     NaN
44688     NaN
44689     NaN
44690     NaN
Name: ROI, Length: 44691, dtype: float64
```

In [90]:
```
df.groupby("Franchise").ROI.median()
```

Out[90]:
```
Franchise
False   1.62
True    3.71
Name: ROI, dtype: float64
```

Franchise va Stand- Alone: Average Budget

In [91]:
```
df.groupby("Franchise").budget_musd.mean()
```

Out[91]:
```
Franchise
False   18.05
True    38.32
Name: budget_musd, dtype: float64
```

Franchise vs Stand-Alone: Average Rating

In [92]:
```
df.groupby("Franchise").vote_average.mean()
```

Out[92]:
```
Franchise
False   6.01
True    5.96
Name: vote_average, dtype: float64
```

In [93]:
```
df.groupby("Franchise").agg({'budget_musd': "mean", "revenue_musd": "mean"
                             "popularity": "mean","ROI":"median","vote_cou
```

Out[93]:

Franchise	budget_musd	revenue_musd	vote_average	popularity	ROI	vote_count
False	18.05	44.74	6.01	2.59	1.62	78.29
True	38.32	165.71	5.96	6.25	3.71	412.39

In [94]:
```
1 df
```

Out[94]:

	id	title	tagline	release_date	genres	belongs_to_collectio
0	862	Toy Story	NaN	1995-10-30	Animation|Comedy|Family	Toy Story Collectic
1	8844	Jumanji	Roll the dice and unleash the excitement!	1995-12-15	Adventure|Fantasy|Family	Na
2	15602	Grumpier Old Men	Still Yelling. Still Fighting. Still Ready for...	1995-12-22	Romance|Comedy	Grumpy Old Me Collectic
3	31357	Waiting to Exhale	Friends are the people who let you be yourself...	1995-12-22	Comedy|Drama|Romance	Na
4	11862	Father of the Bride Part II	Just When His World Is Back To Normal... He's ...	1995-02-10	Comedy	Father of the Bric Collectic
...	...	...	...	...	...	
44686	439050	Subdue	Rising and falling between a man and woman	NaT	Drama|Family	Na
44687	111109	Century of Birthing	NaN	2011-11-17	Drama	Na
44688	67758	Betrayal	A deadly game of wits.	2003-08-01	Action|Drama|Thriller	Na
44689	227506	Satan Triumphant	NaN	1917-10-21	NaN	Na
44690	461257	Queerama	NaN	2017-06-09	NaN	Na

44691 rows × 24 columns

Most Successful Franchises

In [95]:
```
1 df.belongs_to_collection
```

Out[95]:
```
0                  Toy Story Collection
1                                   NaN
2             Grumpy Old Men Collection
3                                   NaN
4        Father of the Bride Collection
                      ...
44686                               NaN
44687                               NaN
44688                               NaN
44689                               NaN
44690                               NaN
Name: belongs_to_collection, Length: 44691, dtype: object
```

In [96]:
```
1 df.belongs_to_collection.value_counts()
```

Out[96]:
```
The Bowery Boys                  29
Totò Collection                  27
Zatôichi: The Blind Swordsman    26
James Bond Collection            26
The Carry On Collection          25
                                 ..
Salt and Pepper Collection        1
Deadpool Collection               1
Ant-Man Collection                1
Elvira Collection                 1
Red Lotus Collection              1
Name: belongs_to_collection, Length: 1691, dtype: int64
```

In [97]:
```
1 franchises = df.groupby("belongs_to_collection").agg({"title":"count", "bu
2                                                       "revenue_musd": ["su
3                                                      "popularity": "mean",
```

In [98]:
```
1 franchises
```

Out[98]:

	title	budget_musd		revenue_musd		vote_average	popularity	ROI	vo
	count	sum	mean	sum	mean	mean	mean	median	
belongs_to_collection									
... Has Fallen Collection	2	130.00	65.00	366.78	183.39	6.00	13.01	2.86	
00 Schneider Filmreihe	1	0.00	NaN	0.00	NaN	6.50	1.93	NaN	
08/15 Collection	1	0.00	NaN	0.00	NaN	5.90	0.63	NaN	
100 Girls Collection	2	0.00	NaN	0.00	NaN	5.15	3.08	NaN	
101 Dalmatians (Animated) Collection	2	4.00	4.00	215.88	215.88	6.25	13.06	53.97	
...	...	...	...	...	...	...	...	...	
Сказки Чуковского	1	0.00	NaN	0.00	NaN	3.00	0.73	NaN	
Чебурашка и крокодил Гена	1	0.00	NaN	0.00	NaN	6.70	0.88	NaN	
Что Творят мужчины! (Коллекция)	2	2.00	2.00	0.00	NaN	3.15	1.30	NaN	
男はつらいよ シリーズ	3	0.00	NaN	0.00	NaN	7.00	0.04	NaN	
식객 시리즈	2	0.00	NaN	0.00	NaN	4.95	0.16	NaN	

1691 rows × 9 columns

In [99]:
```
1 franchises.nlargest(10, ("title", "count"))
```

Out[99]:

	title	budget_musd		revenue_musd		vote_average	popularity	ROI
	count	sum	mean	sum	mean	mean	mean	median
belongs_to_collection								
The Bowery Boys	29	0.00	NaN	0.00	NaN	6.67	0.20	NaN
Totò Collection	27	0.00	NaN	0.00	NaN	6.84	1.05	NaN
James Bond Collection	26	1539.65	59.22	7106.97	273.35	6.34	13.45	6.13
Zatôichi: The Blind Swordsman	26	0.00	NaN	0.00	NaN	6.40	1.10	NaN
The Carry On Collection	25	0.00	NaN	0.00	NaN	6.17	3.22	NaN
Charlie Chan (Sidney Toler) Collection	21	0.00	NaN	0.00	NaN	6.61	0.63	NaN
Pokémon Collection	20	250.72	50.14	601.87	66.87	6.05	4.37	4.46
Godzilla (Showa) Collection	16	2.81	0.56	0.00	NaN	5.97	3.42	NaN
Charlie Chan (Warner Oland) Collection	15	0.00	NaN	0.00	NaN	6.66	0.66	NaN
Dragon Ball Z (Movie) Collection	15	5.00	5.00	112.12	56.06	6.61	6.18	12.35

In [100]:
```
1 franchises.nlargest(10,("revenue_musd","sum"))
```

Out[100]:

	title	budget_musd		revenue_musd		vote_average	popularity	ROI
	count	sum	mean	sum	mean	mean	mean	median
belongs_to_collection								
Harry Potter Collection	8	1280.00	160.00	7707.37	963.42	7.54	26.25	6.17
Star Wars Collection	8	854.35	106.79	7434.49	929.31	7.38	23.41	8.24
James Bond Collection	26	1539.65	59.22	7106.97	273.35	6.34	13.45	6.13
The Fast and the Furious Collection	8	1009.00	126.12	5125.10	640.64	6.66	10.80	4.94
Pirates of the Caribbean Collection	5	1250.00	250.00	4521.58	904.32	6.88	53.97	3.45
Transformers Collection	5	965.00	193.00	4366.10	873.22	6.14	14.43	5.20
Despicable Me Collection	6	299.00	74.75	3691.07	922.77	6.78	106.72	12.76
The Twilight Collection	5	385.00	77.00	3342.11	668.42	5.84	29.50	10.27
Ice Age Collection	5	429.00	85.80	3216.71	643.34	6.38	16.08	8.26
Jurassic Park Collection	4	379.00	94.75	3031.48	757.87	6.50	10.77	7.03

In [101]:
```
1 franchises.nlargest(10,("budget_musd","sum"))
```

Out[101]:

	title	budget_musd		revenue_musd		vote_average	popularity	ROI
	count	sum	mean	sum	mean	mean	mean	median
belongs_to_collection								
James Bond Collection	26	1539.65	59.22	7106.97	273.35	6.34	13.45	6.13
Harry Potter Collection	8	1280.00	160.00	7707.37	963.42	7.54	26.25	6.17
Pirates of the Caribbean Collection	5	1250.00	250.00	4521.58	904.32	6.88	53.97	3.45
The Fast and the Furious Collection	8	1009.00	126.12	5125.10	640.64	6.66	10.80	4.94
X-Men Collection	6	983.00	163.83	2808.83	468.14	6.82	9.71	3.02
Transformers Collection	5	965.00	193.00	4366.10	873.22	6.14	14.43	5.20
Star Wars Collection	8	854.35	106.79	7434.49	929.31	7.38	23.41	8.24
The Hobbit Collection	3	750.00	250.00	2935.52	978.51	7.23	25.21	3.83
The Terminator Collection	5	661.40	132.28	1845.33	369.07	6.54	21.89	2.84
Mission: Impossible Collection	5	650.00	130.00	2778.98	555.80	6.60	16.51	4.55

In [102]:
```
1 franchises[franchises[("vote_count", "mean")] >= 1000].nlargest(10,("vote_
```

Out[102]:

	title	budget_musd		revenue_musd		vote_average	popularity	ROI
	count	sum	mean	sum	mean	mean	mean	median
belongs_to_collection								
The Lord of the Rings Collection	3	266.00	88.67	2916.54	972.18	8.03	30.27	11.73
The Godfather Collection	3	73.00	24.33	429.38	143.13	7.97	31.64	3.66
Blade Runner Collection	1	28.00	28.00	33.14	33.14	7.90	96.27	1.18
The Man With No Name Collection	3	2.00	0.67	35.50	11.83	7.83	14.17	25.00
The Dark Knight Collection	3	585.00	195.00	2463.72	821.24	7.80	57.42	4.34
Guardians of the Galaxy Collection	2	370.00	185.00	1636.74	818.37	7.75	119.31	4.43
Kill Bill Collection	2	60.00	30.00	333.11	166.55	7.70	23.40	5.55
Kingsman Collection	1	81.00	81.00	414.35	414.35	7.60	28.22	5.12
How to Train Your Dragon Collection	2	310.00	155.00	1104.00	552.00	7.55	13.34	3.60
Harry Potter Collection	8	1280.00	160.00	7707.37	963.42	7.54	26.25	6.17

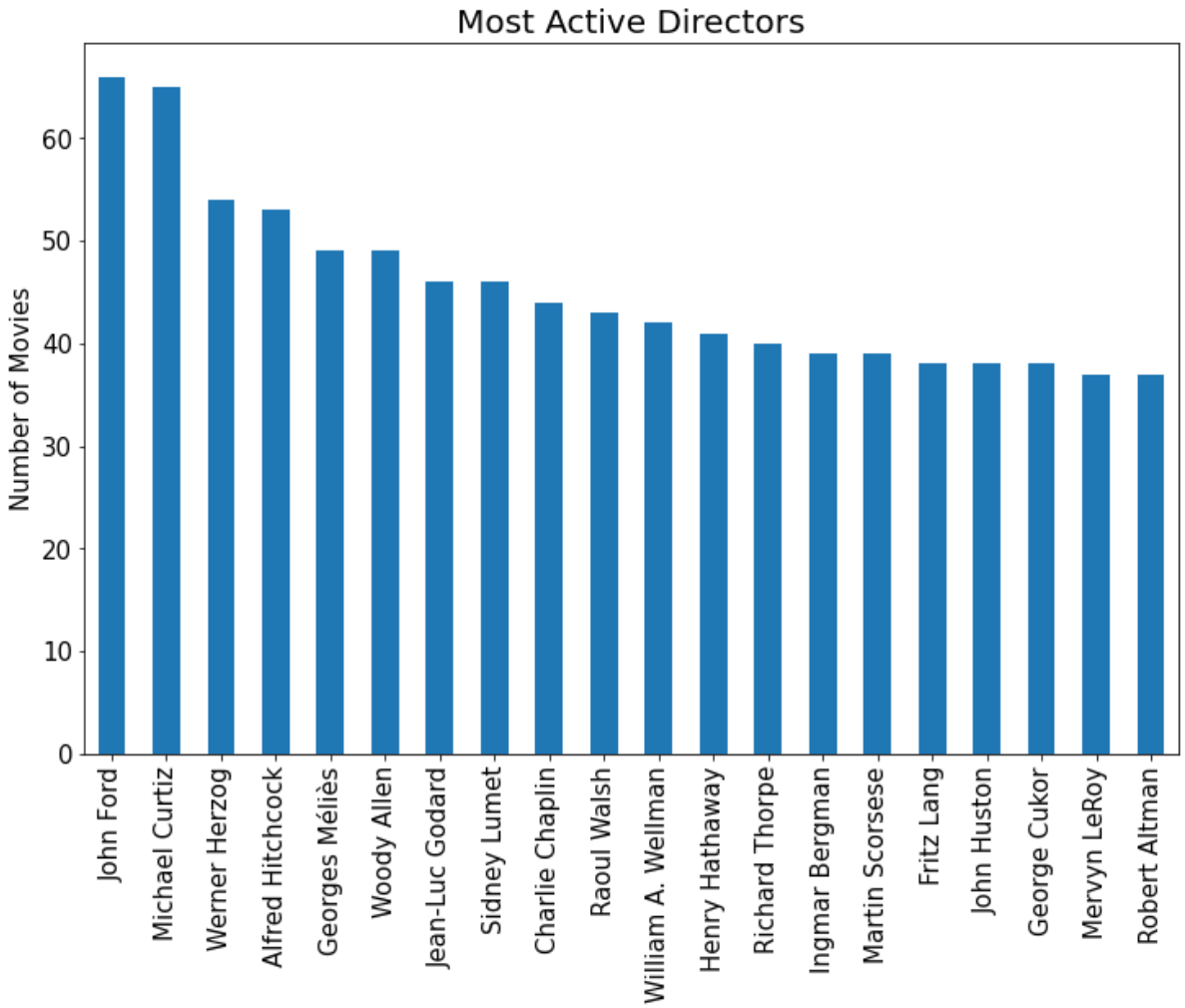
In [103]:
```
1 df.director
```

Out[103]:
```
0           John Lasseter
1            Joe Johnston
2          Howard Deutch
3        Forest Whitaker
4          Charles Shyer
              ...
44686    Hamid Nematollah
44687           Lav Diaz
44688      Mark L. Lester
44689    Yakov Protazanov
44690       Daisy Asquith
Name: director, Length: 44691, dtype: object
```

In [104]:
```
1 df.director.value_counts().head(20)
```

Out[104]:
```
John Ford             66
Michael Curtiz        65
Werner Herzog         54
Alfred Hitchcock      53
Georges Méliès        49
Woody Allen           49
Jean-Luc Godard       46
Sidney Lumet          46
Charlie Chaplin       44
Raoul Walsh           43
William A. Wellman    42
Henry Hathaway        41
Richard Thorpe        40
Ingmar Bergman        39
Martin Scorsese       39
Fritz Lang            38
John Huston           38
George Cukor          38
Mervyn LeRoy          37
Robert Altman         37
Name: director, dtype: int64
```

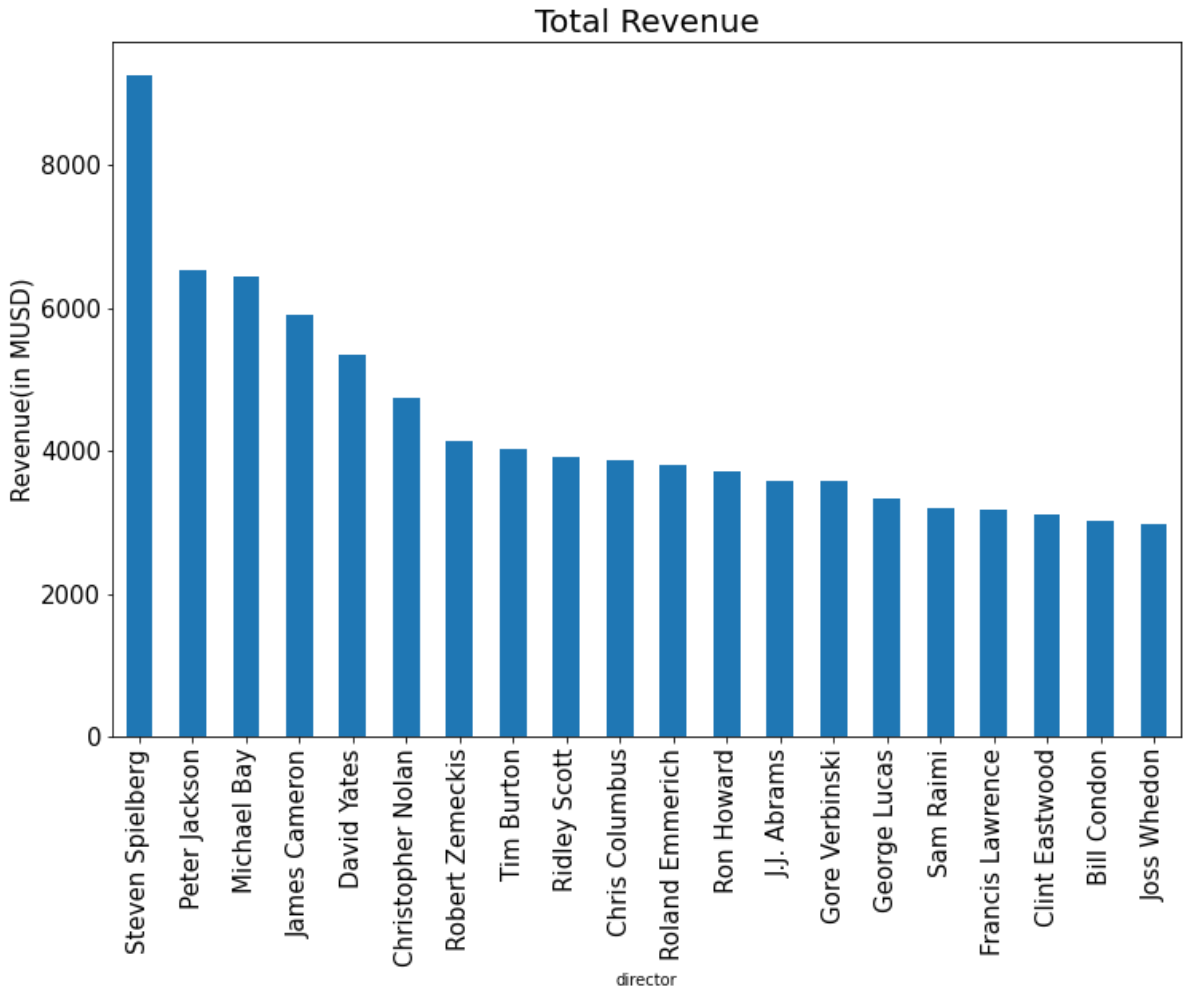
In [105]:
```
plt.figure(figsize = (12,8))
df.director.value_counts().head(20).plot(kind = 'bar', fontsize = 15)
plt.title("Most Active Directors", fontsize = 20)
plt.ylabel("Number of Movies", fontsize = 15)
plt.show()
```



In [106]:
```
1 df.groupby("director").revenue_musd.sum().nlargest(20)
```

Out[106]:
```
director
Steven Spielberg    9256.62
Peter Jackson       6528.24
Michael Bay         6437.47
James Cameron       5900.61
David Yates         5334.56
Christopher Nolan   4747.41
Robert Zemeckis     4138.23
Tim Burton          4032.92
Ridley Scott        3917.53
Chris Columbus      3866.84
Roland Emmerich     3798.40
Ron Howard          3714.15
J.J. Abrams         3579.22
Gore Verbinski      3575.34
George Lucas        3341.55
Sam Raimi           3193.79
Francis Lawrence    3183.34
Clint Eastwood      3100.68
Bill Condon         3017.30
Joss Whedon         2963.83
Name: revenue_musd, dtype: float64
```

In [107]:
```
plt.figure(figsize = (12,8))
df.groupby("director").revenue_musd.sum().nlargest(20).plot(kind = 'bar',
plt.title("Total Revenue", fontsize = 20)
plt.ylabel("Revenue(in MUSD)", fontsize = 15)
plt.show()
```



In [108]:
```
directors = df.groupby("director").agg({"title":"count", "vote_average":'m
```

In [109]:
```
1 directors
```

Out[109]:

director	title	vote_average	vote_count
Dale Trevillion\t	2	4.00	4.00
Davide Manuli	1	6.90	10.00
E.W. Swackhamer	1	5.90	5.00
Vitaliy Vorobyov	1	5.50	3.00
Yeon Sang-Ho	4	6.60	1039.00
...	...	...	...
Ярополк Лапшин	1	10.00	1.00
پیمان معادی	1	6.00	2.00
塩谷 直義	1	7.20	40.00
杰森·莫玛	1	5.80	28.00
진모영	1	6.00	6.00

17349 rows × 3 columns

In [110]:
```
directors[(directors.vote_count >= 10000) & (directors.title >=10)].nlarge
```

Out[110]:

director	title	vote_average	vote_count
Hayao Miyazaki	14	7.70	14700.00
Christopher Nolan	11	7.62	67344.00
Quentin Tarantino	10	7.49	45910.00
Wes Anderson	10	7.37	11743.00
David Fincher	10	7.35	37588.00
Martin Scorsese	39	7.22	35541.00
Peter Jackson	13	7.14	47571.00
Joel Coen	17	7.02	18139.00
James Cameron	11	6.93	33736.00
Stanley Kubrick	16	6.91	18214.00
James Mangold	10	6.90	16607.00
Steven Spielberg	33	6.89	62266.00
Danny Boyle	14	6.87	16504.00
Guy Ritchie	10	6.80	19626.00
Robert Zemeckis	19	6.79	37666.00
Terry Gilliam	14	6.76	10049.00
Tim Burton	21	6.73	36922.00
John Lasseter	10	6.72	18337.00
Ang Lee	14	6.71	11164.00
Antoine Fuqua	12	6.71	15519.00

Find out Most Successful Horror Movies director

In [111]:
```
df.genres = df.genres.astype(str)
```

In [112]:
```
1 df.loc[df.genres.str.contains("Horror")].groupby("director").revenue_musd.
```

Out[112]:
```
director
Paul W.S. Anderson    982.29
James Wan             861.31
Wes Craven            834.93
Francis Lawrence      816.23
Ridley Scott          689.00
Marc Forster          531.87
Steven Spielberg      500.10
William Friedkin      466.40
Darren Lynn Bousman   456.34
M. Night Shyamalan    375.37
Henry Joost           349.07
David R. Ellis        348.74
Adrian Lyne           346.11
James DeMonaco        316.70
Stephen Sommers       311.46
Gore Verbinski        275.91
Guillermo del Toro    261.63
John R. Leonetti      255.27
Fede Alvarez          254.64
Jordan Peele          252.43
Name: revenue_musd, dtype: float64
```

Most Successful Actors (Part 1)

In [113]:
```
1 df.cast
```

Out[113]:
```
0        Tom Hanks|Tim Allen|Don Rickles|Jim Varney|Wal...
1        Robin Williams|Jonathan Hyde|Kirsten Dunst|Bra...
2        Walter Matthau|Jack Lemmon|Ann-Margret|Sophia ...
3        Whitney Houston|Angela Bassett|Loretta Devine|...
4        Steve Martin|Diane Keaton|Martin Short|Kimberl...
                               ...
44686              Leila Hatami|Kourosh Tahami|Elham Korda
44687    Angel Aquino|Perry Dizon|Hazel Orencio|Joel To...
44688    Erika Eleniak|Adam Baldwin|Julie du Page|James...
44689    Iwan Mosschuchin|Nathalie Lissenko|Pavel Pavlo...
44690                                                  NaN
Name: cast, Length: 44691, dtype: object
```

In [114]:
```
1 df.head()
```

Out[114]:

	id	title	tagline	release_date	genres	belongs_to_collection	orig
0	862	Toy Story	NaN	1995-10-30	Animation|Comedy|Family	Toy Story Collection	
1	8844	Jumanji	Roll the dice and unleash the excitement!	1995-12-15	Adventure|Fantasy|Family	NaN	
2	15602	Grumpier Old Men	Still Yelling. Still Fighting. Still Ready for...	1995-12-22	Romance|Comedy	Grumpy Old Men Collection	
3	31357	Waiting to Exhale	Friends are the people who let you be yourself...	1995-12-22	Comedy|Drama|Romance	NaN	
4	11862	Father of the Bride Part II	Just When His World Is Back To Normal... He's ...	1995-02-10	Comedy	Father of the Bride Collection	

In [116]:
```
1 df.set_index("id", inplace = True)
```

In [117]:
```
1 df.head()
```

Out[117]:

id	title	tagline	release_date	genres	belongs_to_collection	origina
862	Toy Story	NaN	1995-10-30	Animation|Comedy|Family	Toy Story Collection	
8844	Jumanji	Roll the dice and unleash the excitement!	1995-12-15	Adventure|Fantasy|Family	NaN	
15602	Grumpier Old Men	Still Yelling. Still Fighting. Still Ready for...	1995-12-22	Romance|Comedy	Grumpy Old Men Collection	
31357	Waiting to Exhale	Friends are the people who let you be yourself...	1995-12-22	Comedy|Drama|Romance	NaN	
11862	Father of the Bride Part II	Just When His World Is Back To Normal... He's ...	1995-02-10	Comedy	Father of the Bride Collection	

In [121]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 44691 entries, 862 to 461257
Data columns (total 23 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   title                  44691 non-null  object
 1   tagline                20284 non-null  object
 2   release_date           44657 non-null  datetime64[ns]
 3   genres                 44691 non-null  object
 4   belongs_to_collection  4463 non-null   object
 5   original_language      44681 non-null  object
 6   budget_musd            8854 non-null   float64
 7   revenue_musd           7385 non-null   float64
 8   production_companies   33356 non-null  object
 9   production_countries   38835 non-null  object
 10  vote_count             44691 non-null  float64
 11  vote_average           42077 non-null  float64
 12  popularity             44691 non-null  float64
 13  runtime                43179 non-null  float64
 14  overview               43740 non-null  object
 15  spoken_languages       41094 non-null  object
 16  poster_path            44467 non-null  object
 17  cast                   42502 non-null  object
 18  cast_size              44691 non-null  int64
 19  crew_size              44691 non-null  int64
 20  director               43960 non-null  object
 21  Franchise              44691 non-null  bool
 22  ROI                    5371 non-null   float64
dtypes: bool(1), datetime64[ns](1), float64(7), int64(2), object(12)
memory usage: 7.9+ MB
```

In [122]:
```
1 df.cast.str.split("|")
```

Out[122]:
```
id
862       [Tom Hanks, Tim Allen, Don Rickles, Jim Varney...
8844      [Robin Williams, Jonathan Hyde, Kirsten Dunst,...
15602     [Walter Matthau, Jack Lemmon, Ann-Margret, Sop...
31357     [Whitney Houston, Angela Bassett, Loretta Devi...
11862     [Steve Martin, Diane Keaton, Martin Short, Kim...
                                ...
439050          [Leila Hatami, Kourosh Tahami, Elham Korda]
111109    [Angel Aquino, Perry Dizon, Hazel Orencio, Joe...
67758     [Erika Eleniak, Adam Baldwin, Julie du Page, J...
227506    [Iwan Mosschuchin, Nathalie Lissenko, Pavel Pa...
461257                                                  NaN
Name: cast, Length: 44691, dtype: object
```

In [124]:
```
1 act = df.cast.str.split("|", expand = True)
```

In [125]:
```
1 act
```

Out[125]:

id	0	1	2	3	4	5	6	7	
862	Tom Hanks	Tim Allen	Don Rickles	Jim Varney	Wallace Shawn	John Ratzenberger	Annie Potts	John Morris	
8844	Robin Williams	Jonathan Hyde	Kirsten Dunst	Bradley Pierce	Bonnie Hunt	Bebe Neuwirth	David Alan Grier	Patricia Clarkson	H
15602	Walter Matthau	Jack Lemmon	Ann-Margret	Sophia Loren	Daryl Hannah	Burgess Meredith	Kevin Pollak	None	
31357	Whitney Houston	Angela Bassett	Loretta Devine	Lela Rochon	Gregory Hines	Dennis Haysbert	Michael Beach	Mykelti Williamson	
11862	Steve Martin	Diane Keaton	Martin Short	Kimberly Williams-Paisley	George Newbern	Kieran Culkin	BD Wong	Peter Michael Goetz	M
...	...	...	...	...	...	...	...	...	
439050	Leila Hatami	Kourosh Tahami	Elham Korda	None	None	None	None	None	
111109	Angel Aquino	Perry Dizon	Hazel Orencio	Joel Torre	Bart Guingona	Soliman Cruz	Roeder	Angeli Bayani	
67758	Erika Eleniak	Adam Baldwin	Julie du Page	James Remar	Damian Chapa	Louis Mandylor	Tom Wright	Jeremy Lelliott	Q
227506	Iwan Mosschuchin	Nathalie Lissenko	Pavel Pavlov	Aleksandr Chabrov	Vera Orlova	None	None	None	
461257	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

44691 rows × 313 columns

In [136]:
```
act.stack().reset_index(level = 1, drop = True).to_frame()
```

Out[136]:

	0
id	
862	Tom Hanks
862	Tim Allen
862	Don Rickles
862	Jim Varney
862	Wallace Shawn
...	...
227506	Iwan Mosschuchin
227506	Nathalie Lissenko
227506	Pavel Pavlov
227506	Aleksandr Chabrov
227506	Vera Orlova

557703 rows × 1 columns

In [138]:
```
act = act.stack().reset_index(level = 1, drop = True).to_frame()
```

In [139]:
```
act
```

Out[139]:

	0
id	
862	Tom Hanks
862	Tim Allen
862	Don Rickles
862	Jim Varney
862	Wallace Shawn
...	...
227506	Iwan Mosschuchin
227506	Nathalie Lissenko
227506	Pavel Pavlov
227506	Aleksandr Chabrov
227506	Vera Orlova

557703 rows × 1 columns

In [140]:
```
act.columns = ['Actor']
```

In [141]:
```
act = act.merge(df[["title", "revenue_musd", "vote_average", "popularity"]
```

In [142]:
```
act
```

Out[142]:

id	Actor	title	revenue_musd	vote_average	popularity
2	Turo Pajala	Ariel	NaN	7.10	3.86
2	Susanna Haavisto	Ariel	NaN	7.10	3.86
2	Matti Pellonpää	Ariel	NaN	7.10	3.86
2	Eetu Hilkamo	Ariel	NaN	7.10	3.86
3	Matti Pellonpää	Shadows in Paradise	NaN	7.10	2.29
...	...	...	...	...	...
469172	Vasco Sequeira	Manuel on the Island of Wonders	NaN	NaN	0.00
469172	Armanda Bacelar	Manuel on the Island of Wonders	NaN	NaN	0.00
469172	Rafael de Sousa	Manuel on the Island of Wonders	NaN	NaN	0.00
469172	José Antônio Gomes	Manuel on the Island of Wonders	NaN	NaN	0.00
469172	Isabel Branco	Manuel on the Island of Wonders	NaN	NaN	0.00

557703 rows × 5 columns

In [157]:
```
act.Actor.unique()
```

Out[157]:
```
array(['Turo Pajala', 'Susanna Haavisto', 'Matti Pellonpää', ...,
       'Armanda Bacelar', 'Rafael de Sousa', 'José Antônio Gomes'],
      dtype=object)
```

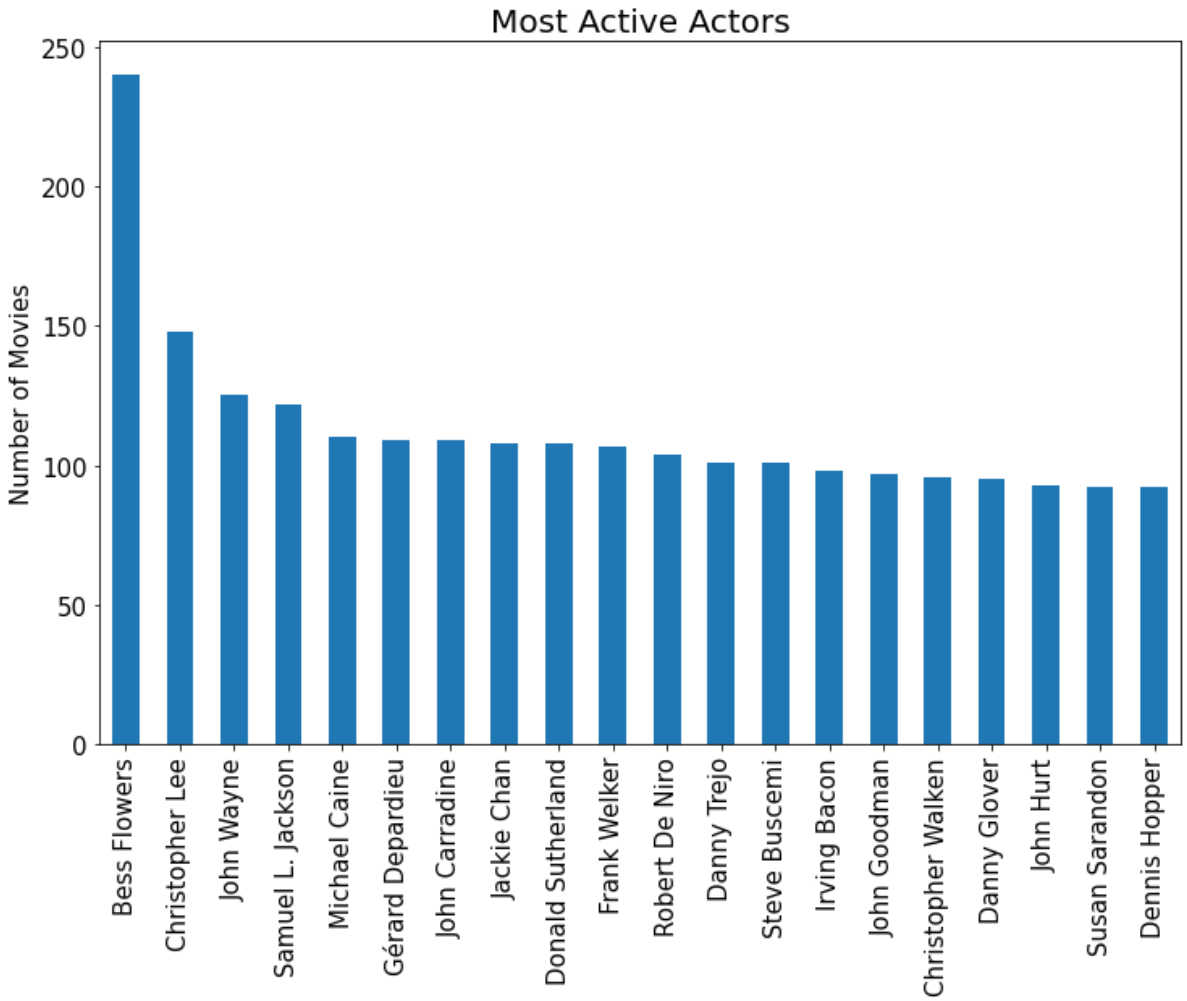
In [159]:
```
act.Actor.nunique()
```

Out[159]:
```
201501
```

In [161]:
```
1 act.Actor.value_counts().head(20)
```

Out[161]:
```
Bess Flowers          240
Christopher Lee       148
John Wayne            125
Samuel L. Jackson     122
Michael Caine         110
Gérard Depardieu      109
John Carradine        109
Jackie Chan           108
Donald Sutherland     108
Frank Welker          107
Robert De Niro        104
Danny Trejo           101
Steve Buscemi         101
Irving Bacon           98
John Goodman           97
Christopher Walken     96
Danny Glover           95
John Hurt              93
Susan Sarandon         92
Dennis Hopper          92
Name: Actor, dtype: int64
```

In [163]:
```
plt.figure(figsize = (12,8))
act.Actor.value_counts().head(20).plot(kind ="bar", fontsize = 15)
plt.title("Most Active Actors", fontsize = 20)
plt.ylabel("Number of Movies", fontsize = 15)
plt.show()
```



In [176]:
```
agg = act.groupby("Actor").agg(Total_Revenue = ("revenue_musd", 'sum'),
                               Mean_Revenue = ("revenue_musd", "mean"),
                              Mean_Rating = ("vote_average", "mean"),
                              Mean_Pop = ("popularity", "mean"),
                              Total_Movies = ("Actor", "count"))
```

In [178]:
```
1 agg.nlargest(10, "Total_Movies")
```

Out[178]:

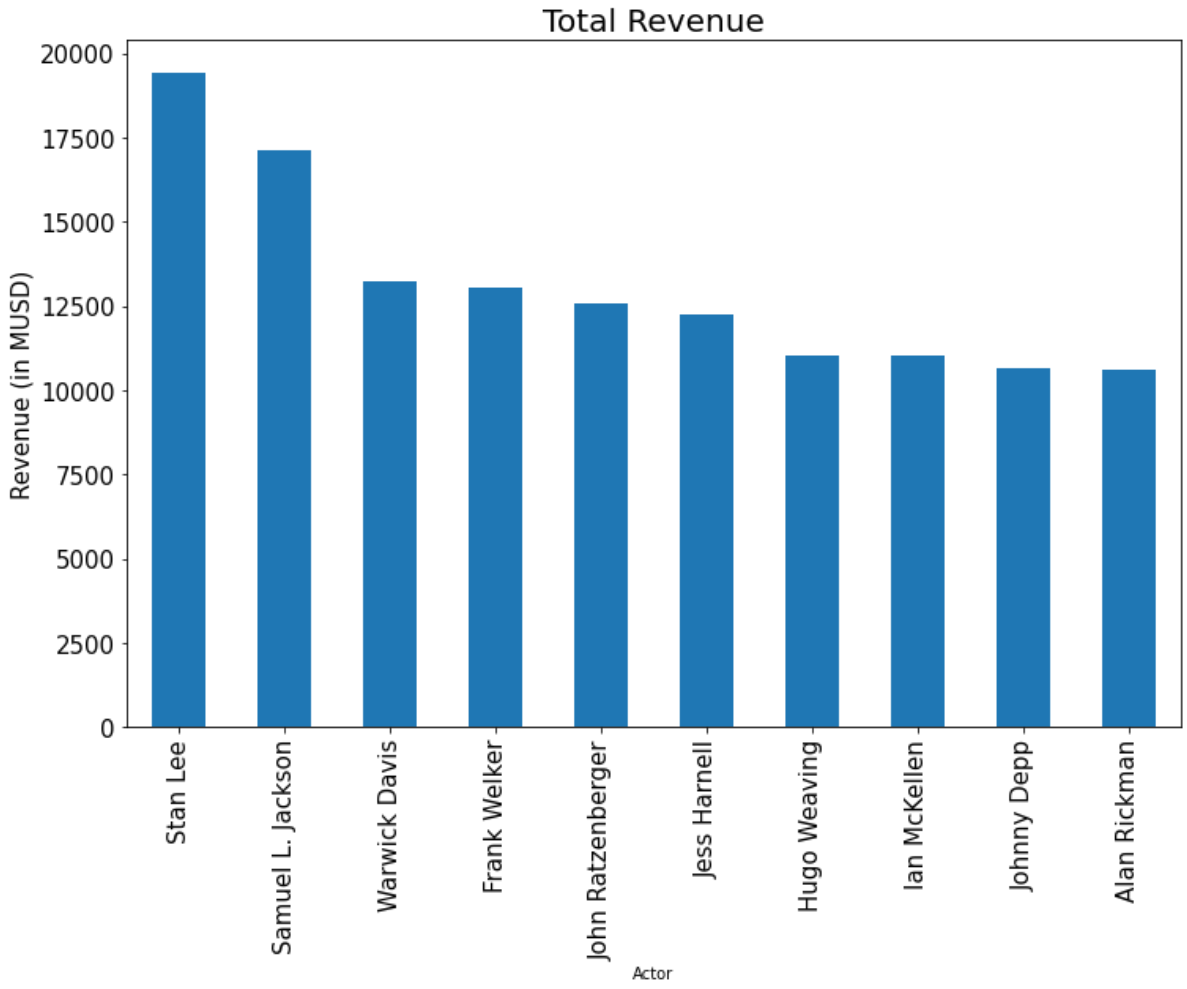
Actor	Total_Revenue	Mean_Revenue	Mean_Rating	Mean_Pop	Total_Movies
Bess Flowers	368.91	14.76	6.18	2.03	240
Christopher Lee	9417.05	324.73	5.91	4.75	148
John Wayne	236.09	11.24	5.71	3.09	125
Samuel L. Jackson	17109.62	213.87	6.27	11.70	122
Michael Caine	8053.40	191.75	6.27	8.27	110
Gérard Depardieu	1247.61	95.97	6.05	3.70	109
John Carradine	255.84	19.68	5.55	2.43	109
Donald Sutherland	5390.77	138.22	6.23	7.00	108
Jackie Chan	4699.19	146.85	6.28	5.86	108
Frank Welker	13044.15	326.10	6.31	9.57	107

In [179]:
```
1 agg.nlargest(10, "Total_Revenue")
```

Out[179]:

Actor	Total_Revenue	Mean_Revenue	Mean_Rating	Mean_Pop	Total_Movies
Stan Lee	19414.96	647.17	6.51	29.94	48
Samuel L. Jackson	17109.62	213.87	6.27	11.70	122
Warwick Davis	13256.03	662.80	6.29	13.09	34
Frank Welker	13044.15	326.10	6.31	9.57	107
John Ratzenberger	12596.13	449.86	6.48	10.96	46
Jess Harnell	12234.61	611.73	6.44	10.92	35
Hugo Weaving	11027.58	459.48	6.47	10.97	40
Ian McKellen	11015.59	478.94	6.35	15.45	44
Johnny Depp	10653.76	217.42	6.44	12.38	69
Alan Rickman	10612.63	353.75	6.72	10.40	45

In [185]:
```
plt.figure(figsize = (12,8))
agg.Total_Revenue.nlargest(10).plot(kind = 'bar', fontsize = 15)
plt.title("Total Revenue", fontsize = 20)
plt.ylabel("Revenue (in MUSD)", fontsize = 15)
plt.show()
```



In [186]:
```
agg.Mean_Revenue.nlargest(10)
```

Out[186]:
```
Actor
April Marie Thomas   2787.97
Ashley Jeffery       2787.97
Austin Wilson        2787.97
Brandon Jelkes       2787.97
Bravita A. Threatt   2787.97
Carvon Futrell       2787.97
Chris Mala           2787.97
Christa Oliver       2787.97
Christopher Nolen    2787.97
Colin Bleasdale      2787.97
Name: Mean_Revenue, dtype: float64
```

In [187]:
```
1 agg.Mean_Revenue.nlargest(10)
```

Out[187]:
```
Actor
April Marie Thomas   2787.97
Ashley Jeffery       2787.97
Austin Wilson        2787.97
Brandon Jelkes       2787.97
Bravita A. Threatt   2787.97
Carvon Futrell       2787.97
Chris Mala           2787.97
Christa Oliver       2787.97
Christopher Nolen    2787.97
Colin Bleasdale      2787.97
Name: Mean_Revenue, dtype: float64
```

In [188]:
```
1 act[act.Actor == "Ashley Jeffery"]
```

Out[188]:

id	Actor	title	revenue_musd	vote_average	popularity
19995	Ashley Jeffery	Avatar	2787.97	7.20	185.07

the problem here is that Actors acted in only one's movie that is Avatar which has the highest mean income for those actors.

In [189]:
```
1 agg[agg.Total_Movies >= 10].nlargest(10, "Mean_Revenue")
```

Out[189]:

Actor	Total_Revenue	Mean_Revenue	Mean_Rating	Mean_Pop	Total_Movies
Gloria Stuart	1845.03	1845.03	6.37	3.48	18
Keith Richards	2967.71	989.24	6.46	5.03	23
Zoë Wanamaker	976.48	976.48	6.33	6.82	10
James Cameron	1862.08	931.04	7.06	4.69	12
Matthew Lewis	7915.31	879.48	7.37	23.10	11
Luke de Woolfson	1720.67	860.34	5.72	8.77	11
Yuri Lowenthal	1708.16	854.08	6.19	19.88	17
Dominic Monaghan	3289.61	822.40	6.05	10.62	11
Philip Ng	821.71	821.71	5.92	6.35	10
Peter Mayhew	4820.72	803.45	6.70	12.30	11

In [190]:
```
1 agg[agg.Total_Movies >= 10].nlargest(10, "Mean_Rating")
```

Out[190]:

Actor	Total_Revenue	Mean_Revenue	Mean_Rating	Mean_Pop	Total_Movies
Masao Hayama	0.00	NaN	8.84	0.32	10
David Attenborough	0.00	NaN	8.27	2.15	11
Emil Jannings	0.00	NaN	7.78	1.70	10
Halit Akçatepe	0.21	0.21	7.78	0.74	10
Yo Oizumi	511.21	102.24	7.72	7.51	13
Şener Şen	11.07	3.69	7.69	0.91	16
Ayşen Gruda	0.91	0.46	7.68	0.74	10
Akira Tani	0.33	0.16	7.65	5.04	12
Joseph Oliveira	3543.44	354.34	7.64	34.45	10
Björk	51.33	25.67	7.62	2.36	10

In [191]:
```
1 agg[agg.Total_Movies >= 10].nlargest(10, "Mean_Pop")
```

Out[191]:

Actor	Total_Revenue	Mean_Revenue	Mean_Rating	Mean_Pop	Total_Movies
Katy Mixon	1519.57	151.96	5.84	51.97	12
Terry Notary	6947.21	694.72	6.47	51.58	11
Goran D. Kleut	2568.94	366.99	6.00	42.32	10
Mark Smith	2195.52	243.95	6.55	40.08	11
Jon Hamm	3449.35	191.63	6.33	39.42	25
Gal Gadot	5449.53	495.41	6.33	37.39	11
Ava Acres	6272.36	482.49	5.99	36.26	21
Emma Watson	9639.20	535.51	6.77	35.97	19
Joseph Oliveira	3543.44	354.34	7.64	34.45	10
Keith Jardine	1062.49	212.50	5.96	32.00	11

In [ ]:
```
1
```