Welcome to another lesson on Data Analysis / Data Science using the amazing Libraries in Python { Numpy, Pandas & Matplotlib }.

This is part of sharing my learning and growth process on this career path. And also, to hope that this will encourage you to keep doing things right, even if it means doing it poorly till you gain mastery of it.

In this lesson, I'll be working with IMDB Movies Dataset which is readily available online for public access.

This Dataset contains 3 Sheets, and we're going to work on the 3 of them, by cleaning, exploring, analyzing, deriving insights and making visual plots as the case may require,

So, Let's Get Started !!!

The first step will be to import the required Libraries which is required for the task.

These include Numpy, Pandas and Matplotlib. Remember to include "%matplotlib inline" . This will enable your plots to display in Jupyter Notebook. Skipping this step will require you to type plt.show() after every plotting code.

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

Now, import the first sheet of the excel file and read the first lines of the Dataset.

Note the use of .T on the head() function. This is just an optional method to ensure the Dataset is Transposed and the full view is obtained

In [2]:
```
file = pd.read_excel("movies.xls")
file.head().T
```

Out[2]:

	0	1	2	3	4
Title	Intolerance: Love's Struggle Throughout the Ages	Over the Hill to the Poorhouse	The Big Parade	Metropolis	Pandora's Box
Year	1916	1920	1925	1927	1929
Genres	Drama|History|War	Crime|Drama	Drama|Romance|War	Drama|Sci-Fi	Crime|Drama|Romance
Language	NaN	NaN	NaN	German	German
Country	USA	USA	USA	Germany	Germany
Content Rating	Not Rated	NaN	Not Rated	Not Rated	Not Rated
Duration	123	110	151	145	110
Aspect Ratio	1.33	1.33	1.33	1.33	1.33
Budget	385907	100000	245000	6e+06	NaN
Gross Earnings	NaN	3e+06	NaN	26435	9950
Director	D.W. Griffith	Harry F. Millarde	King Vidor	Fritz Lang	Georg Wilhelm Pabst
Actor 1	Lillian Gish	Stephen Carr	John Gilbert	Brigitte Helm	Louise Brooks
Actor 2	Mae Marsh	Johnnie Walker	Renée Adorée	Gustav Fröhlich	Francis Lederer
Actor 3	Walter Long	Mary Carr	Claire Adams	Rudolf Klein-Rogge	Fritz Kortner
Facebook Likes - Director	204	0	54	756	21
Facebook Likes - Actor 1	436	2	81	136	426
Facebook Likes - Actor 2	22	2	12	23	20
Facebook Likes - Actor 3	9	0	6	18	3
Facebook Likes - cast Total	481	4	108	203	455
Facebook likes - Movie	691	0	226	12000	926
Facenumber in posters	1	1	0	1	1
User Votes	10718	5	4849	111841	7431

	0	1	2	3	4
Reviews by Users	88	1	45	413	84
Reviews by Crtiics	69	1	48	260	71
IMDB Score	8	4.8	8.3	8.3	8

Now, import the second sheet of the Excel File and read the first 5 lines of the Dataset

In [3]:
```
file1 = pd.read_excel("movies.xls", sheet_name = 1)
file1.head().T
```

Out[3]:

	0	1	2	3	
Title	102 Dalmatians	28 Days	3 Strikes	Aberdeen	All the Pretty Hors
Year	2000	2000	2000	2000	20
Genres	Adventure|Comedy|Family	Comedy|Drama	Comedy	Drama	Drama|Romance|West
Language	English	English	English	English	Engl
Country	USA	USA	USA	UK	U
Content Rating	G	PG-13	R	NaN	PG-
Duration	100	103	82	106	2
Aspect Ratio	1.85	1.37	1.85	1.85	2.
Budget	8.5e+07	4.3e+07	6e+06	6.5e+06	5.7e+
Gross Earnings	6.69416e+07	3.70355e+07	9.82134e+06	64148	1.55271e+
Director	Kevin Lima	Betty Thomas	DJ Pooh	Hans Petter Moland	Billy Bob Thornt
Actor 1	Ioan Gruffudd	Steve Buscemi	Mo'Nique	Charlotte Rampling	Matt Dam
Actor 2	Eric Idle	Viggo Mortensen	Mike Epps	Sara-Marie Maltha	Henry Thom
Actor 3	Jim Carter	Elizabeth Perkins	Faizon Love	Stellan Skarsgård	Sam Shepa
Facebook Likes - Director	36	84	69	19	
Facebook Likes - Actor 1	2000	12000	939	844	130
Facebook Likes - Actor 2	795	10000	706	2	8
Facebook Likes - Actor 3	439	664	585	0	8
Facebook Likes - cast Total	4182	23864	3354	846	150
Facebook likes - Movie	372	0	118	260	6
Facenumber in posters	1	1	1	0	
User Votes	26413	34597	1415	2601	113
Reviews by Users	77	194	10	35	1

	0	1	2	3
Reviews by Crtiics	84	116	22	28
IMDB Score	4.8	6	4	7.3

Now, import the third sheet of the excel file and read the first 5 lines of the Dataset

In [4]:
```
file2 = pd.read_excel("movies.xls", sheet_name = 2)
file2.head().T
```

Out[4]:

	0	1	2	3
Title	127 Hours	3 Backyards	3	8: The Mormon Proposition
Year	2010	2010	2010	2010
Genres	Adventure|Biography|Drama|Thriller	Drama	Comedy|Drama|Romance	Documentary
Language	English	English	German	English
Country	USA	USA	Germany	USA
Content Rating	R	R	Unrated	R
Duration	94	88	119	80
Aspect Ratio	1.85	NaN	2.35	1.78
Budget	1.8e+07	300000	NaN	2.5e+06
Gross Earnings	1.83295e+07	NaN	59774	99851
Director	Danny Boyle	Eric Mendelsohn	Tom Tykwer	Reed Cowan
Actor 1	James Franco	Embeth Davidtz	Devid Striesow	Dustin Lance Black
Actor 2	Treat Williams	Edie Falco	Sebastian Schipper	Emily Pearson
Actor 3	Kate Burton	Kathryn Erbe	Sophie Rois	Gavin Newsom
Facebook Likes - Director	0	5	670	0
Facebook Likes - Actor 1	11000	795	24	191
Facebook Likes - Actor 2	642	659	20	12
Facebook Likes - Actor 3	223	301	9	5
Facebook Likes - cast Total	11984	1884	69	210
Facebook likes - Movie	63000	92	2000	0
Facenumber in posters	0	0	0	0
User Votes	279179	554	4212	1138
Reviews by Users	440	23	18	30

	0	1	2	3
Reviews by Crtiics	450	20	76	28
IMDB Score	7.6	5.2	6.8	7.1

Now that the 3 sheets in the Excel File have been imported and read, you can proceed to further actions like joining the 3 files together into a single Dataset.

But before then, confirm the state of these datasets so that you will know if further actions on the datasets are successful

In [5]:
```
print(file.shape)
print(file1.shape)
print(file2.shape)
```

```
(1338, 25)
(2100, 25)
(1604, 25)
```

You can as well confirm if the column names for each of the files or Dataset are equal. Doing this now will avoid complications when you want to join the 3 Datasets together.

Moreover, there's no harm in double-checking the status of the Dataset

In [6]:
```
for A,B,C in zip(file.columns, file1.columns, file2.columns):
    if A == B == C:
        print(True)
        print(A,'=',B,'=',C,"\n")
    else:
        print(False)
```

```
True
Title = Title = Title

True
Year = Year = Year

True
Genres = Genres = Genres

True
Language = Language = Language

True
Country = Country = Country

True
Content Rating = Content Rating = Content Rating

True
Duration = Duration = Duration

True
Aspect Ratio = Aspect Ratio = Aspect Ratio

True
Budget = Budget = Budget

True
Gross Earnings = Gross Earnings = Gross Earnings

True
Director = Director = Director

True
Actor 1 = Actor 1 = Actor 1

True
Actor 2 = Actor 2 = Actor 2

True
Actor 3 = Actor 3 = Actor 3

True
Facebook Likes - Director = Facebook Likes - Director = Facebook Likes
- Director

True
Facebook Likes - Actor 1 = Facebook Likes - Actor 1 = Facebook Likes -
Actor 1

True
Facebook Likes - Actor 2 = Facebook Likes - Actor 2 = Facebook Likes -
Actor 2

True
Facebook Likes - Actor 3 = Facebook Likes - Actor 3 = Facebook Likes -
Actor 3
```

```
True
Facebook Likes - cast Total = Facebook Likes - cast Total = Facebook L
ikes - cast Total

True
Facebook likes - Movie = Facebook likes - Movie = Facebook likes - Mov
ie

True
Facenumber in posters = Facenumber in posters = Facenumber in posters

True
User Votes = User Votes = User Votes

True
Reviews by Users = Reviews by Users = Reviews by Users

True
Reviews by Crtiics = Reviews by Crtiics = Reviews by Crtiics

True
IMDB Score = IMDB Score = IMDB Score
```

Now, it's time to join the 3 files together. Here, the pd.concat() will be utilized.

In [7]:
```
dataset = pd.concat([file, file1, file2])
print(dataset.shape)
dataset.head()
```

```
(5042, 25)
```

Out[7]:

	Title	Year	Genres	Language	Country	Content Rating	Duration	Aspect Ratio	B
0	Intolerance: Love's Struggle Throughout the Ages	1916.0	Drama|History|War	NaN	USA	Not Rated	123.0	1.33	385
1	Over the Hill to the Poorhouse	1920.0	Crime|Drama	NaN	USA	NaN	110.0	1.33	100
2	The Big Parade	1925.0	Drama|Romance|War	NaN	USA	Not Rated	151.0	1.33	245
3	Metropolis	1927.0	Drama|Sci-Fi	German	Germany	Not Rated	145.0	1.33	6000
4	Pandora's Box	1929.0	Crime|Drama|Romance	German	Germany	Not Rated	110.0	1.33	

5 rows × 25 columns

How are you sure that what you've done is correct ?

You can write a small code to confirm the authenticity of the last step

Remember when dealing with data, you cannot be overbearing as double-checking is concerned. Any mistake in the handling of data will affect all your actions and results across-the-board. And in most cases than not, it's a flawed outcome.

In [8]:
```
print("The index size of file is :",len(file.index),"\n")
print("The index size of file1 is :",len(file1.index),"\n")
print("The index size o file2 is :",len(file2.index),"\n")
print("The index size of the combined dataset is :",len(file1) + len(f
ile) + len(file2))
print("The index size of dataset is :",len(dataset.index))
```

```
The index size of file is : 1338

The index size of file1 is : 2100

The index size o file2 is : 1604

The index size of the combined dataset is : 5042
The index size of dataset is : 5042
```

And just a simple process like the one displayed below can save you tons of stress that might arise later because you didn't double-check

In [9]:
```
combined_dataset = len(file.index) + len(file1.index) + len(file2.inde
x)
if len(dataset.index) == combined_dataset :
    print("Your concatenation process is correct. You can Proceed")
else :
    print("Your Concatenation process has some erroei. Kindly rectify
 it before you proceed")
```

```
Your concatenation process is correct. You can Proceed
```

And now that this process is confirmed accurate, you can proceed to get familiar with the dadaset that you will now be working with.

Check its Statistical details, its attributes and all other necessary things that will give you a fair idea of the dataset

In [10]:
```
dataset.shape
```

Out[10]:
```
(5042, 25)
```

In [11]:
```
dataset.size
```

Out[11]:
```
126050
```

In [12]:
```
dataset.dtypes
```

Out[12]:
```
Title                          object
Year                          float64
Genres                         object
Language                       object
Country                        object
Content Rating                 object
Duration                      float64
Aspect Ratio                  float64
Budget                        float64
Gross Earnings                float64
Director                       object
Actor 1                        object
Actor 2                        object
Actor 3                        object
Facebook Likes - Director     float64
Facebook Likes - Actor 1      float64
Facebook Likes - Actor 2      float64
Facebook Likes - Actor 3      float64
Facebook Likes - cast Total     int64
Facebook likes - Movie          int64
Facenumber in posters         float64
User Votes                      int64
Reviews by Users              float64
Reviews by Crtiics            float64
IMDB Score                    float64
dtype: object
```

In [13]:
```
pd.value_counts(dataset.dtypes)
```

Out[13]:
```
float64    13
object      9
int64       3
dtype: int64
```

In [14]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5042 entries, 0 to 1603
Data columns (total 25 columns):
Title                          5042 non-null object
Year                           4935 non-null float64
Genres                         5042 non-null object
Language                       5031 non-null object
Country                        5038 non-null object
Content Rating                 4740 non-null object
Duration                       5028 non-null float64
Aspect Ratio                   4714 non-null float64
Budget                         4551 non-null float64
Gross Earnings                 4159 non-null float64
Director                       4938 non-null object
Actor 1                        5035 non-null object
Actor 2                        5029 non-null object
Actor 3                        5020 non-null object
Facebook Likes - Director      4938 non-null float64
Facebook Likes - Actor 1       5035 non-null float64
Facebook Likes - Actor 2       5029 non-null float64
Facebook Likes - Actor 3       5020 non-null float64
Facebook Likes - cast Total    5042 non-null int64
Facebook likes - Movie         5042 non-null int64
Facenumber in posters          5029 non-null float64
User Votes                     5042 non-null int64
Reviews by Users               5022 non-null float64
Reviews by Crtiics             4993 non-null float64
IMDB Score                     5042 non-null float64
dtypes: float64(13), int64(3), object(9)
memory usage: 846.9+ KB
```

From the above details using the info() attribute of the dataset, it's observed that the index column of the dataset is not well labeled.

It's necessary to check it out and correct it accordingly

Even though this could've been corrected when the 3 datasets were concatenated, (because pd.concat takes the "ignore_index" parameter), it's not too much of hassle to correct that below with just 1 or 2 lines of code(s)

That will be corrected below and check again

In [15]:
```
new_index = range(0,5042)
dataset.index = new_index
dataset.index
```

Out[15]:
```
RangeIndex(start=0, stop=5042, step=1)
```

In [16]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5042 entries, 0 to 5041
Data columns (total 25 columns):
Title                          5042 non-null object
Year                           4935 non-null float64
Genres                         5042 non-null object
Language                       5031 non-null object
Country                        5038 non-null object
Content Rating                 4740 non-null object
Duration                       5028 non-null float64
Aspect Ratio                   4714 non-null float64
Budget                         4551 non-null float64
Gross Earnings                 4159 non-null float64
Director                       4938 non-null object
Actor 1                        5035 non-null object
Actor 2                        5029 non-null object
Actor 3                        5020 non-null object
Facebook Likes - Director      4938 non-null float64
Facebook Likes - Actor 1       5035 non-null float64
Facebook Likes - Actor 2       5029 non-null float64
Facebook Likes - Actor 3       5020 non-null float64
Facebook Likes - cast Total    5042 non-null int64
Facebook likes - Movie         5042 non-null int64
Facenumber in posters          5029 non-null float64
User Votes                     5042 non-null int64
Reviews by Users               5022 non-null float64
Reviews by Crtiics             4993 non-null float64
IMDB Score                     5042 non-null float64
dtypes: float64(13), int64(3), object(9)
memory usage: 807.6+ KB
```

In [17]:
```
dataset.describe()
```

Out[17]:

	Year	Duration	Aspect Ratio	Budget	Gross Earnings	Facebook Likes - Director	Faceb
count	4935.000000	5028.000000	4714.000000	4.551000e+03	4.159000e+03	4938.000000	503
mean	2002.470517	107.201074	2.220403	3.975262e+07	4.846841e+07	686.621709	656
std	12.474599	25.197441	1.385113	2.061149e+08	6.845299e+07	2813.602405	1502
min	1916.000000	7.000000	1.180000	2.180000e+02	1.620000e+02	0.000000	
25%	1999.000000	93.000000	1.850000	6.000000e+06	5.340988e+06	7.000000	61
50%	2005.000000	103.000000	2.350000	2.000000e+07	2.551750e+07	49.000000	98
75%	2011.000000	118.000000	2.350000	4.500000e+07	6.230944e+07	194.750000	1100
max	2016.000000	511.000000	16.000000	1.221550e+10	7.605058e+08	23000.000000	64000

Okay, time to check up for duplicate values. And whatever the outcome is will be treated accordingly

In [18]:
```
dataset.duplicated().sum()
```

Out[18]: 45

From the above results, there's and indication that there's a total of 45 duplicate values in the dataset.

Instead of just taking that blindly, why not let's dig deeper to actually

In [19]:
```
dup_values = dataset[dataset.duplicated()]
dup_values
```

Out[19]:

	Title	Year	Genres	Language	Country	Content Rating	Durati
131	Night of the Living Dead	1968.0	Drama|Horror|Mystery	English	USA	Unrated	9
167	The French Connection	1971.0	Action|Crime|Drama|Thriller	English	USA	R	10
237	Halloween	1978.0	Horror|Thriller	English	USA	R	10
296	History of the World: Part I	1981.0	Comedy	English	USA	R	9
321	Cat People	1982.0	Fantasy|Horror|Thriller	English	USA	R	9
381	Footloose	1984.0	Drama|Music|Romance	English	USA	PG	10
498	Dangerous Liaisons	1988.0	Drama|Romance	English	USA	R	11
579	Total Recall	1990.0	Action|Sci-Fi	English	USA	R	11
851	Hamlet	1996.0	Drama	English	UK	PG-13	15
1011	The Full Monty	1997.0	Comedy|Drama|Music	English	UK	R	9
1139	The Love Letter	1998.0	Fantasy|Romance	English	USA	Unrated	9
1449	Snatch	2000.0	Comedy|Crime	English	UK	R	10
1463	The Claim	2000.0	Drama|Romance|Western	English	UK	R	11
1561	From Hell	2001.0	Horror|Mystery|Thriller	English	USA	R	12
1618	O	2001.0	Drama|Romance|Thriller	English	USA	R	9
1660	The Fast and the Furious	2001.0	Action|Crime|Thriller	English	USA	PG-13	10
1721	Big Fat Liar	2002.0	Adventure|Comedy|Family	English	USA	PG	8
1744	Crossroads	2002.0	Comedy|Drama	English	USA	PG-13	9
1778	Hero	2002.0	Action|Adventure|History	Mandarin	China	PG-13	8
1849	Stealing Harvard	2002.0	Comedy|Crime	English	USA	PG-13	8
2118	Crash	2004.0	Crime|Drama|Thriller	English	Germany	R	11
2232	The Alamo	2004.0	Drama|History|War|Western	English	USA	PG-13	13
2282	Wicker Park	2004.0	Drama|Mystery|Romance|Thriller	English	USA	PG-13	11
2707	The Illusionist	2006.0	Drama|Mystery|Romance|Thriller	English	USA	PG-13	11
2757	A Dog's Breakfast	2007.0	Comedy	English	Canada	NaN	8
2791	Death at a Funeral	2007.0	Comedy	English	USA	R	8

	Title	Year	Genres	Language	Country	Content Rating	Durati
3192	A Woman, a Gun and a Noodle Shop	2009.0	Comedy|Drama	Mandarin	China	R	9
3272	Halloween II	2009.0	Horror	English	USA	R	11
3555	My Soul to Take	2010.0	Horror|Mystery|Thriller	English	USA	R	10
4049	The Avengers	2012.0	Action|Adventure|Sci-Fi	English	USA	PG-13	17
4082	The Possession	2012.0	Horror|Thriller	English	USA	PG-13	9
4088	The Twilight Saga: Breaking Dawn - Part 2	2012.0	Adventure|Drama|Fantasy|Romance	English	USA	PG-13	11
4335	Trance	2013.0	Crime|Drama|Mystery|Thriller	English	UK	R	10
4443	Hercules	2014.0	Action|Adventure	English	USA	PG-13	10
4467	Left Behind	2014.0	Action|Drama|Fantasy|Mystery|Thriller	English	USA	PG-13	11
4538	The Calling	2014.0	Thriller	English	USA	R	10
4588	Unbroken	2014.0	Biography|Drama|Sport|War	English	USA	PG-13	13
4669	Fantastic Four	2015.0	Action|Adventure|Sci-Fi	English	USA	PG-13	10
4674	Forsaken	2015.0	Drama|Western	English	Canada	R	9
4721	Pan	2015.0	Adventure|Family|Fantasy	English	USA	PG	11
4819	Victor Frankenstein	2015.0	Drama|Horror|Sci-Fi|Thriller	English	USA	PG-13	11
4838	Bad Moms	2016.0	Comedy	English	USA	R	10
4863	Godzilla Resurgence	2016.0	Action|Adventure|Drama|Horror|Sci-Fi	Japanese	Japan	NaN	12
4917	The Legend of Tarzan	2016.0	Action|Adventure|Drama|Romance	English	USA	PG-13	11
4998	Saving Grace	NaN	Drama|Fantasy	English	USA	TV-MA	6

45 rows × 25 columns

Now that you have the duplicate values, from my experience, i will say don't just take it as it is presented. You can dig a little deeper. This way, you'll be double sure of what to delete and what to retain.

So, why not write a little code and display the duplicate values. That way, you can visualize and make informed decisions

In [20]:
```
create = []
for num in dup_values.index:
    create.append(num-1)
    create.append(num)
    create.append(num+1)
create
```

Out[20]: [130,
 131,
 132,
 166,
 167,
 168,
 236,
 237,
 238,
 295,
 296,
 297,
 320,
 321,
 322,
 380,
 381,
 382,
 497,
 498,
 499,
 578,
 579,
 580,
 850,
 851,
 852,
 1010,
 1011,
 1012,
 1138,
 1139,
 1140,
 1448,
 1449,
 1450,
 1462,
 1463,
 1464,
 1560,
 1561,
 1562,
 1617,
 1618,
 1619,
 1659,
 1660,
 1661,
 1720,
 1721,
 1722,
 1743,
 1744,
 1745,
 1777,
 1778,
 1779,

1848,
1849,
1850,
2117,
2118,
2119,
2231,
2232,
2233,
2281,
2282,
2283,
2706,
2707,
2708,
2756,
2757,
2758,
2790,
2791,
2792,
3191,
3192,
3193,
3271,
3272,
3273,
3554,
3555,
3556,
4048,
4049,
4050,
4081,
4082,
4083,
4087,
4088,
4089,
4334,
4335,
4336,
4442,
4443,
4444,
4466,
4467,
4468,
4537,
4538,
4539,
4587,
4588,
4589,
4668,
4669,
4670,

4673,
4674,
4675,
4720,
4721,
4722,
4818,
4819,
4820,
4837,
4838,
4839,
4862,
4863,
4864,
4916,
4917,
4918,
4997,
4998,
4999]

In [21]:
```
dup_show = dataset.iloc[create]
display(dup_show[ :45 ])
display(dup_show[45:91])
display(dup_show[91: ])
```

	Title	Year	Genres	Language	Country	Co Ra
130	Night of the Living Dead	1968.0	Drama|Horror|Mystery	English	USA	Un
131	Night of the Living Dead	1968.0	Drama|Horror|Mystery	English	USA	Un
132	Oliver!	1968.0	Drama|Family|Musical	English	UK	
166	The French Connection	1971.0	Action|Crime|Drama|Thriller	English	USA	
167	The French Connection	1971.0	Action|Crime|Drama|Thriller	English	USA	
168	The Night Visitor	1971.0	Crime|Horror|Thriller	English	USA	
236	Halloween	1978.0	Horror|Thriller	English	USA	
237	Halloween	1978.0	Horror|Thriller	English	USA	
238	Halloween	1978.0	Horror|Thriller	English	USA	
295	History of the World: Part I	1981.0	Comedy	English	USA	
296	History of the World: Part I	1981.0	Comedy	English	USA	
297	Inchon	1981.0	Drama|History|War	English	South Korea	
320	Cat People	1982.0	Fantasy|Horror|Thriller	English	USA	
321	Cat People	1982.0	Fantasy|Horror|Thriller	English	USA	
322	Class of 1984	1982.0	Action|Crime|Drama|Thriller	English	Canada	
380	Footloose	1984.0	Drama|Music|Romance	English	USA	
381	Footloose	1984.0	Drama|Music|Romance	English	USA	
382	Friday the 13th: The Final Chapter	1984.0	Horror|Thriller	English	USA	
497	Dangerous Liaisons	1988.0	Drama|Romance	English	USA	
498	Dangerous Liaisons	1988.0	Drama|Romance	English	USA	
499	Die Hard	1988.0	Action|Thriller	English	USA	
578	Total Recall	1990.0	Action|Sci-Fi	English	USA	
579	Total Recall	1990.0	Action|Sci-Fi	English	USA	
580	Tremors	1990.0	Comedy|Horror|Sci-Fi	English	USA	P

	Title	Year	Genres	Language	Country	Co Ra
850	Hamlet	1996.0	Drama	English	UK	P
851	Hamlet	1996.0	Drama	English	UK	P
852	Happy Gilmore	1996.0	Comedy|Sport	English	USA	P
1010	The Full Monty	1997.0	Comedy|Drama|Music	English	UK	
1011	The Full Monty	1997.0	Comedy|Drama|Music	English	UK	
1012	The Game	1997.0	Drama|Mystery|Thriller	English	USA	
1138	The Love Letter	1998.0	Fantasy|Romance	English	USA	Un
1139	The Love Letter	1998.0	Fantasy|Romance	English	USA	Un
1140	The Magic Sword: Quest for Camelot	1998.0	Adventure|Animation|Comedy|Drama|Family|Fantas...	English	USA	
1448	Snatch	2000.0	Comedy|Crime	English	UK	
1449	Snatch	2000.0	Comedy|Crime	English	UK	
1450	Snow Day	2000.0	Adventure|Comedy|Family	English	USA	
1462	The Claim	2000.0	Drama|Romance|Western	English	UK	
1463	The Claim	2000.0	Drama|Romance|Western	English	UK	
1464	The Contender	2000.0	Drama|Thriller	English	USA	
1560	From Hell	2001.0	Horror|Mystery|Thriller	English	USA	
1561	From Hell	2001.0	Horror|Mystery|Thriller	English	USA	
1562	Get Over It	2001.0	Comedy|Romance	English	USA	P
1617	O	2001.0	Drama|Romance|Thriller	English	USA	
1618	O	2001.0	Drama|Romance|Thriller	English	USA	
1619	Ocean's Eleven	2001.0	Crime|Thriller	English	USA	P

45 rows × 25 columns

	Title	Year	Genres	Language	Country	Content Rating	Durati
1659	The Fast and the Furious	2001.0	Action|Crime|Thriller	English	USA	PG-13	10
1660	The Fast and the Furious	2001.0	Action|Crime|Thriller	English	USA	PG-13	10
1661	The Fast and the Furious	2001.0	Action|Crime|Thriller	English	USA	PG-13	10
1720	Big Fat Liar	2002.0	Adventure|Comedy|Family	English	USA	PG	8
1721	Big Fat Liar	2002.0	Adventure|Comedy|Family	English	USA	PG	8
1722	Big Trouble	2002.0	Comedy|Crime|Thriller	English	USA	PG-13	7
1743	Crossroads	2002.0	Comedy|Drama	English	USA	PG-13	9
1744	Crossroads	2002.0	Comedy|Drama	English	USA	PG-13	9
1745	Cypher	2002.0	Mystery|Romance|Sci-Fi|Thriller	English	USA	R	9
1777	Hero	2002.0	Action|Adventure|History	Mandarin	China	PG-13	8
1778	Hero	2002.0	Action|Adventure|History	Mandarin	China	PG-13	8
1779	Hey Arnold! The Movie	2002.0	Adventure|Animation|Comedy|Family	English	USA	PG	7
1848	Stealing Harvard	2002.0	Comedy|Crime	English	USA	PG-13	8
1849	Stealing Harvard	2002.0	Comedy|Crime	English	USA	PG-13	8
1850	Stolen Summer	2002.0	Drama	English	USA	PG	9
2117	Crash	2004.0	Crime|Drama|Thriller	English	Germany	R	11
2118	Crash	2004.0	Crime|Drama|Thriller	English	Germany	R	11
2119	D.E.B.S.	2004.0	Action|Comedy|Romance	English	USA	PG-13	9
2231	The Alamo	2004.0	Drama|History|War|Western	English	USA	PG-13	13
2232	The Alamo	2004.0	Drama|History|War|Western	English	USA	PG-13	13
2233	The Aviator	2004.0	Biography|Drama	English	USA	PG-13	17
2281	Wicker Park	2004.0	Drama|Mystery|Romance|Thriller	English	USA	PG-13	11
2282	Wicker Park	2004.0	Drama|Mystery|Romance|Thriller	English	USA	PG-13	11
2283	Wimbledon	2004.0	Comedy|Romance|Sport	English	UK	PG-13	9
2706	The Illusionist	2006.0	Drama|Mystery|Romance|Thriller	English	USA	PG-13	11
2707	The Illusionist	2006.0	Drama|Mystery|Romance|Thriller	English	USA	PG-13	11

	Title	Year	Genres	Language	Country	Content Rating	Durati
2708	The Lake House	2006.0	Drama|Fantasy|Romance	English	USA	PG	9
2756	A Dog's Breakfast	2007.0	Comedy	English	Canada	NaN	8
2757	A Dog's Breakfast	2007.0	Comedy	English	Canada	NaN	8
2758	A Mighty Heart	2007.0	Biography|Drama|History|Thriller|War	English	USA	R	10
2790	Death at a Funeral	2007.0	Comedy	English	USA	R	8
2791	Death at a Funeral	2007.0	Comedy	English	USA	R	8
2792	Death Sentence	2007.0	Action|Crime|Thriller	English	USA	R	11
3191	A Woman, a Gun and a Noodle Shop	2009.0	Comedy|Drama	Mandarin	China	R	9
3192	A Woman, a Gun and a Noodle Shop	2009.0	Comedy|Drama	Mandarin	China	R	9
3193	Adam	2009.0	Drama|Romance	English	USA	PG-13	9
3271	Halloween II	2009.0	Horror	English	USA	R	11
3272	Halloween II	2009.0	Horror	English	USA	R	11
3273	Hannah Montana: The Movie	2009.0	Comedy|Drama|Family|Music|Romance	English	USA	G	10
3554	My Soul to Take	2010.0	Horror|Mystery|Thriller	English	USA	R	10
3555	My Soul to Take	2010.0	Horror|Mystery|Thriller	English	USA	R	10
3556	Nanny McPhee Returns	2010.0	Comedy|Family|Fantasy	English	UK	PG	10
4048	The Avengers	2012.0	Action|Adventure|Sci-Fi	English	USA	PG-13	17
4049	The Avengers	2012.0	Action|Adventure|Sci-Fi	English	USA	PG-13	17
4050	The Bourne Legacy	2012.0	Action|Adventure|Thriller	English	USA	PG-13	13
4081	The Possession	2012.0	Horror|Thriller	English	USA	PG-13	9

46 rows × 25 columns

	Title	Year	Genres	Language	Country	Content Rating	Duratic
4082	The Possession	2012.0	Horror|Thriller	English	USA	PG-13	92
4083	The Raven	2012.0	Crime|Mystery|Thriller	English	USA	R	110
4087	The Twilight Saga: Breaking Dawn - Part 2	2012.0	Adventure|Drama|Fantasy|Romance	English	USA	PG-13	115
4088	The Twilight Saga: Breaking Dawn - Part 2	2012.0	Adventure|Drama|Fantasy|Romance	English	USA	PG-13	115
4089	The Vow	2012.0	Drama|Romance	English	USA	PG-13	104
4334	Trance	2013.0	Crime|Drama|Mystery|Thriller	English	UK	R	101
4335	Trance	2013.0	Crime|Drama|Mystery|Thriller	English	UK	R	101
4336	Treachery	2013.0	Drama|Thriller	English	USA	NaN	67
4442	Hercules	2014.0	Action|Adventure	English	USA	PG-13	101
4443	Hercules	2014.0	Action|Adventure	English	USA	PG-13	101
4444	Hidden Away	2014.0	Drama|Romance	Spanish	Spain	Unrated	96
4466	Left Behind	2014.0	Action|Drama|Fantasy|Mystery|Thriller	English	USA	PG-13	110
4467	Left Behind	2014.0	Action|Drama|Fantasy|Mystery|Thriller	English	USA	PG-13	110
4468	Let's Be Cops	2014.0	Comedy	English	USA	R	104
4537	The Calling	2014.0	Thriller	English	USA	R	108
4538	The Calling	2014.0	Thriller	English	USA	R	108
4539	The Equalizer	2014.0	Action|Crime|Thriller	English	USA	R	132
4587	Unbroken	2014.0	Biography|Drama|Sport|War	English	USA	PG-13	137
4588	Unbroken	2014.0	Biography|Drama|Sport|War	English	USA	PG-13	137
4589	Unfriended	2014.0	Horror|Mystery|Thriller	English	USA	R	83
4668	Fantastic Four	2015.0	Action|Adventure|Sci-Fi	English	USA	PG-13	100
4669	Fantastic Four	2015.0	Action|Adventure|Sci-Fi	English	USA	PG-13	100
4670	Fear Clinic	2015.0	Horror	English	USA	R	95
4673	Forsaken	2015.0	Drama|Western	English	Canada	R	90
4674	Forsaken	2015.0	Drama|Western	English	Canada	R	90
4675	Freeheld	2015.0	Biography|Drama|Romance	English	USA	PG-13	103
4720	Pan	2015.0	Adventure|Family|Fantasy	English	USA	PG	111

	Title	Year	Genres	Language	Country	Content Rating	Duratic
4721	Pan	2015.0	Adventure|Family|Fantasy	English	USA	PG	111
4722	Pan	2015.0	Adventure|Family|Fantasy	English	USA	PG	111
4818	Victor Frankenstein	2015.0	Drama|Horror|Sci-Fi|Thriller	English	USA	PG-13	110
4819	Victor Frankenstein	2015.0	Drama|Horror|Sci-Fi|Thriller	English	USA	PG-13	110
4820	Victor Frankenstein	2015.0	Drama|Horror|Sci-Fi|Thriller	English	USA	PG-13	110
4837	Bad Moms	2016.0	Comedy	English	USA	R	100
4838	Bad Moms	2016.0	Comedy	English	USA	R	100
4839	Batman v Superman: Dawn of Justice	2016.0	Action|Adventure|Sci-Fi	English	USA	PG-13	183
4862	Godzilla Resurgence	2016.0	Action|Adventure|Drama|Horror|Sci-Fi	Japanese	Japan	NaN	120
4863	Godzilla Resurgence	2016.0	Action|Adventure|Drama|Horror|Sci-Fi	Japanese	Japan	NaN	120
4864	Hail, Caesar!	2016.0	Comedy|Mystery	English	UK	PG-13	106
4916	The Legend of Tarzan	2016.0	Action|Adventure|Drama|Romance	English	USA	PG-13	110
4917	The Legend of Tarzan	2016.0	Action|Adventure|Drama|Romance	English	USA	PG-13	110
4918	The Little Ponderosa Zoo	2016.0	Family	English	USA	NaN	84
4997	Saving Grace	NaN	Drama|Fantasy	English	USA	TV-MA	60
4998	Saving Grace	NaN	Drama|Fantasy	English	USA	TV-MA	60
4999	Scream: The TV Series	NaN	Crime|Drama|Horror|Mystery|Thriller	English	USA	TV-14	45

44 rows × 25 columns

In [22]:
```
display(dup_show.duplicated().sum())
display(dataset.duplicated().sum())
```

45

45

Now, it's confirmed. The duplicate values in the original dataset is the same as the one in the extracted dataset.

This is not a waste of time for the following reasons.

1. It'll build your skills working and digging deeper into data

2. You'll be sure of what you're doing

3. You can visualize the data and see for yourself how the duplicate values look like. In my experience, duplicate values are either directly before or after the original version of it. In this case, both situation are present. That's experience which you'll never get if you fail to dive deep enough.

So, for what it's worth, you should do this once in a while as time permits

Now, you can drop the duplicate values contained in the original dataset and check again.

After that, you can continue with other processes

In [23]:
```
dataset.drop_duplicates(inplace = True)
dataset.duplicated().sum()
```

Out[23]: 0

Here we are. Now it's time to check for missing values.

Some rigorous work will be done here, so fasten your seatbelt tight.

I'm just kidding about that but pay attention to this part. It's a very important aspect of data preparation (Science) process

In [24]:
```
not_missing = dataset.notna().sum()
not_missing
```

Out[24]:
```
Title                          4997
Year                           4891
Genres                         4997
Language                       4986
Country                        4993
Content Rating                 4697
Duration                       4983
Aspect Ratio                   4671
Budget                         4511
Gross Earnings                 4124
Director                       4894
Actor 1                        4990
Actor 2                        4984
Actor 3                        4975
Facebook Likes - Director      4894
Facebook Likes - Actor 1       4990
Facebook Likes - Actor 2       4984
Facebook Likes - Actor 3       4975
Facebook Likes - cast Total    4997
Facebook likes - Movie         4997
Facenumber in posters          4984
User Votes                     4997
Reviews by Users               4977
Reviews by Crtiics             4949
IMDB Score                     4997
dtype: int64
```

As some random values of duplicated dataset has been deleted, the index of the dataset would have been altered. Check this again and correct it accordingly

In [25]:
```
display(len(dataset.index))
#The total length of the index in the dataset is 4997. Now, the index
 should be reconfigured accordingly
new_index1 = range(0,4997)
dataset.index = new_index1
dataset.index
```

```
4997
```

Out[25]:
```
RangeIndex(start=0, stop=4997, step=1)
```

In [26]:
```
missing_values = dataset.isnull().sum()
missing_values
```

Out[26]:
```
Title                          0
Year                         106
Genres                         0
Language                      11
Country                        4
Content Rating               300
Duration                      14
Aspect Ratio                 326
Budget                       486
Gross Earnings               873
Director                     103
Actor 1                        7
Actor 2                       13
Actor 3                       22
Facebook Likes - Director    103
Facebook Likes - Actor 1       7
Facebook Likes - Actor 2      13
Facebook Likes - Actor 3      22
Facebook Likes - cast Total    0
Facebook likes - Movie         0
Facenumber in posters         13
User Votes                     0
Reviews by Users              20
Reviews by Crtiics            48
IMDB Score                     0
dtype: int64
```

In [27]:
```
for column, percent in zip(dataset.columns, missing_values) :
    if percent > 0 :
        print(column,":", percent)
```

```
Year : 106
Language : 11
Country : 4
Content Rating : 300
Duration : 14
Aspect Ratio : 326
Budget : 486
Gross Earnings : 873
Director : 103
Actor 1 : 7
Actor 2 : 13
Actor 3 : 22
Facebook Likes - Director : 103
Facebook Likes - Actor 1 : 7
Facebook Likes - Actor 2 : 13
Facebook Likes - Actor 3 : 22
Facenumber in posters : 13
Reviews by Users : 20
Reviews by Crtiics : 48
```

In [28]:
```
outcome = []
for M,N in zip(missing_values,not_missing) :
    outcome.append((M/(M+N)*100))
for P,Q,R in zip(dataset.columns, outcome, missing_values) :
    if Q > 0 :
        print(P,"=>",R,"=>",Q,"%")
```

```
Year => 106 => 2.1212727636581947 %
Language => 11 => 0.22013207924754855 %
Country => 4 => 0.08004802881729037 %
Content Rating => 300 => 6.0036021612967785 %
Duration => 14 => 0.2801681008605163 %
Aspect Ratio => 326 => 6.523914348609165 %
Budget => 486 => 9.72583550130078 %
Gross Earnings => 873 => 17.470482289373624 %
Director => 103 => 2.061236742045227 %
Actor 1 => 7 => 0.14008405043025815 %
Actor 2 => 13 => 0.2601560936561937 %
Actor 3 => 22 => 0.4402641584950971 %
Facebook Likes - Director => 103 => 2.061236742045227 %
Facebook Likes - Actor 1 => 7 => 0.14008405043025815 %
Facebook Likes - Actor 2 => 13 => 0.2601560936561937 %
Facebook Likes - Actor 3 => 22 => 0.4402641584950971 %
Facenumber in posters => 13 => 0.2601560936561937 %
Reviews by Users => 20 => 0.40024014408645187 %
Reviews by Crtiics => 48 => 0.9605763458074845 %
```

This seems a little more clear. It's very much easier to see the columns that has null values, the number of null values it contains and the percentage of the total values of the column which the null values represent.

When it comes to treating null values, there are many approach to it and how to treat it varies according to the situation.

Some of the condition is the quantity of the missing values we're talking about.

Is the value not recorded or it does not exist.

Whatever the condition is determines what you do and how you do it.

In this case, each column will be scrutinized separately, anf depending on what is revealed, the appropriate approach will be applied accordingly.

In [29]:
```
dataset[dataset.Year.isnull()].index
```

Out[29]:
```
Int64Index([4891, 4892, 4893, 4894, 4895, 4896, 4897, 4898, 4899, 4900,
            ...
            4987, 4988, 4989, 4990, 4991, 4992, 4993, 4994, 4995, 4996],
           dtype='int64', length=106)
```

The code above indicates the positions of the null values. They're towards the end of the dataset. Now we'll call it out, to have a visual clue

In [30]:
```
display(dataset[4540:4600])
display(dataset[4780:4801])
display(dataset[4840 : 4900])
```

	Title	Year	Genres	Language	Country
4540	The Water Diviner	2014.0	Drama|War	English	Australia
4541	They Came Together	2014.0	Comedy	English	USA
4542	Think Like a Man Too	2014.0	Comedy|Romance	English	USA
4543	This Is Where I Leave You	2014.0	Comedy|Drama	English	USA
4544	Tiger Orange	2014.0	Drama	English	USA
4545	Top Five	2014.0	Comedy|Romance	English	USA
4546	Top Spin	2014.0	Documentary	English	USA
4547	Transcendence	2014.0	Drama|Mystery|Romance|Sci-Fi|Thriller	English	UK
4548	Transformers: Age of Extinction	2014.0	Action|Adventure|Sci-Fi	English	USA
4549	Trash	2014.0	Adventure|Crime|Drama|Mystery|Thriller	Portuguese	UK
4550	Tusk	2014.0	Comedy|Drama|Horror	English	USA
4551	Unbroken	2014.0	Biography|Drama|Sport|War	English	USA
4552	Unfriended	2014.0	Horror|Mystery|Thriller	English	USA
4553	United Passions	2014.0	Drama|History|Sport	English	France
4554	Unsullied	2014.0	Action|Horror|Thriller	English	USA
4555	Viy	2014.0	Adventure|Fantasy|Mystery|Thriller	Russian	Russia
4556	When the Game Stands Tall	2014.0	Drama|Family|Sport	English	USA
4557	While We're Young	2014.0	Comedy|Drama	English	USA
4558	Whiplash	2014.0	Drama|Music	English	USA
4559	Wicked Blood	2014.0	Action|Crime|Drama|Thriller	English	USA
4560	Wild	2014.0	Adventure|Biography|Drama	English	USA
4561	Winter's Tale	2014.0	Drama|Fantasy|Mystery|Romance	English	USA
4562	Wish I Was Here	2014.0	Comedy|Drama	English	USA
4563	Wolves	2014.0	Action|Horror	English	France
4564	X-Men: Days of Future Past	2014.0	Action|Adventure|Fantasy|Sci-Fi|Thriller	English	USA
4565	Z Storm	2014.0	Action|Crime	Cantonese	Hong Kong

	Title	Year	Genres	Language	Country
4566	#Horror	2015.0	Drama|Horror|Mystery|Thriller	English	USA
4567	10 Days in a Madhouse	2015.0	Drama	English	USA
4568	90 Minutes in Heaven	2015.0	Drama	English	USA
4569	A Tale of Three Cities	2015.0	Drama	Chinese	China
4570	A Warrior's Tail	2015.0	Adventure|Animation|Fantasy	Russian	Russia
4571	Abandoned	2015.0	Drama	English	New Zealand
4572	Accidental Love	2015.0	Comedy|Romance	English	USA
4573	Adulterers	2015.0	Crime|Drama|Thriller	English	USA
4574	Aloha	2015.0	Comedy|Drama|Romance	English	USA
4575	Aloha	2015.0	Comedy|Drama|Romance	English	USA
4576	Alvin and the Chipmunks: The Road Chip	2015.0	Adventure|Animation|Comedy|Family|Fantasy|Music	English	USA
4577	America Is Still the Place	2015.0	History	English	USA
4578	American Hero	2015.0	Action|Comedy|Drama|Sci-Fi	English	USA
4579	Animal Kingdom: Let's go Ape	2015.0	Adventure|Animation|Comedy|Family	French	France
4580	Anomalisa	2015.0	Animation|Comedy|Drama|Romance	English	USA
4581	Antarctic Edge: 70° South	2015.0	Adventure|Documentary	English	USA
4582	Ant-Man	2015.0	Action|Adventure|Comedy|Sci-Fi	English	USA
4583	Area 51	2015.0	Horror|Sci-Fi|Thriller	English	USA
4584	Avengers: Age of Ultron	2015.0	Action|Adventure|Sci-Fi	English	USA
4585	AWOL-72	2015.0	Thriller	English	USA
4586	Baahubali: The Beginning	2015.0	Action|Adventure|Drama|Fantasy|War	Telugu	India
4587	Bizarre	2015.0	Drama|Musical|Romance	English	France
4588	Black Mass	2015.0	Biography|Crime|Drama	English	USA
4589	Blackhat	2015.0	Action|Crime|Drama|Mystery|Thriller	English	USA
4590	Bleeding Hearts	2015.0	Horror	English	USA
4591	Bridge of Spies	2015.0	Drama|History|Thriller	English	USA
4592	Broken Horses	2015.0	Action|Crime|Drama|Mystery|Thriller	English	USA

	Title	Year	Genres	Language	Country
4593	Brooklyn	2015.0	Drama|Romance	English	UK
4594	Brotherly Love	2015.0	Drama	English	USA
4595	Burnt	2015.0	Comedy|Drama	English	USA
4596	By the Sea	2015.0	Drama|Romance	English	USA
4597	Captive	2015.0	Crime|Drama|Thriller	English	USA
4598	Censored Voices	2015.0	Documentary|History	Hebrew	Israel
4599	Chain of Command	2015.0	Action|Thriller	English	USA

60 rows × 25 columns

	Title	Year	Genres	Language	Country	Content Rating	Dura
4780	Walter	2015.0	Comedy|Drama	English	USA	NaN	9
4781	We Are Your Friends	2015.0	Drama|Music|Romance	English	UK	R	9
4782	Western Religion	2015.0	Adventure|Drama|Fantasy|Thriller|Western	English	USA	NaN	1(
4783	Wild Card	2015.0	Action|Crime|Drama|Thriller	English	USA	R	9
4784	Wind Walkers	2015.0	Action|Horror|Thriller	English	USA	R	9
4785	Windsor Drive	2015.0	Mystery|Thriller	English	USA	NaN	9
4786	Woman in Gold	2015.0	Biography|Drama|History	English	UK	PG-13	1(
4787	Zipper	2015.0	Drama|Thriller	English	USA	R	1(
4788	10 Cloverfield Lane	2016.0	Drama|Horror|Mystery|Sci-Fi|Thriller	English	USA	PG-13	1(
4789	13 Hours	2016.0	Action|Drama|Thriller|War	English	USA	R	1⁴
4790	A Beginner's Guide to Snuff	2016.0	Comedy|Horror|Thriller	English	USA	NaN	8
4791	Airlift	2016.0	Action|Drama|History|Thriller|War	Hindi	India	NaN	1:
4792	Alice Through the Looking Glass	2016.0	Adventure|Family|Fantasy	English	USA	PG	1'
4793	Allegiant	2016.0	Action|Adventure|Mystery|Sci-Fi|Thriller	English	USA	PG-13	1:
4794	Alleluia! The Devil's Carnival	2016.0	Horror|Musical	English	USA	NaN	9
4795	Antibirth	2016.0	Horror	English	USA	NaN	9
4796	Bad Moms	2016.0	Comedy	English	USA	R	1(
4797	Batman v Superman: Dawn of Justice	2016.0	Action|Adventure|Sci-Fi	English	USA	PG-13	1[
4798	Ben-Hur	2016.0	Adventure|Drama|History	English	USA	PG-13	1⁴
4799	Ben-Hur	2016.0	Adventure|Drama|History	English	USA	PG-13	1⁴
4800	Ben-Hur	2016.0	Adventure|Drama|History	English	USA	PG-13	1⁴

21 rows × 25 columns

	Title	Year	Genres	Language	Country	Conte Ratir
4840	Money Monster	2016.0	Crime|Drama|Thriller	English	USA	
4841	Mr. Church	2016.0	Drama	English	USA	PG-1
4842	My Big Fat Greek Wedding 2	2016.0	Comedy|Family|Romance	English	USA	PG-1
4843	Neighbors 2: Sorority Rising	2016.0	Comedy	English	USA	
4844	Nerve	2016.0	Adventure|Crime|Mystery|Sci-Fi|Thriller	English	USA	PG-1
4845	Now You See Me 2	2016.0	Action|Adventure|Comedy|Crime|Mystery|Thriller	English	USA	PG-1
4846	Operation Chromite	2016.0	Action|Drama|History|War	English	South Korea	Na
4847	Our Kind of Traitor	2016.0	Thriller	English	UK	
4848	Pete's Dragon	2016.0	Adventure|Family|Fantasy	English	USA	P
4849	Pride and Prejudice and Zombies	2016.0	Action|Horror|Romance	English	USA	PG-1
4850	Race	2016.0	Biography|Drama|Sport	English	Canada	PG-1
4851	Restoration	2016.0	Horror	English	USA	Na
4852	Ride Along 2	2016.0	Action|Comedy	English	USA	PG-1
4853	Risen	2016.0	Action|Adventure|Drama|Mystery	English	USA	PG-1
4854	Rodeo Girl	2016.0	Family	English	USA	P
4855	Sausage Party	2016.0	Adventure|Animation|Comedy|Fantasy	English	USA	
4856	Star Trek Beyond	2016.0	Action|Adventure|Sci-Fi|Thriller	English	USA	PG-1
4857	Suicide Squad	2016.0	Action|Adventure|Comedy|Sci-Fi	English	USA	PG-1
4858	Teenage Mutant Ninja Turtles: Out of the Shadows	2016.0	Action|Adventure|Comedy|Sci-Fi	English	USA	PG-1
4859	The 5th Wave	2016.0	Action|Adventure|Sci-Fi|Thriller	English	USA	PG-1
4860	The Angry Birds Movie	2016.0	Action|Animation|Comedy|Family	English	USA	P
4861	The BFG	2016.0	Adventure|Family|Fantasy	English	UK	P
4862	The Birth of a Nation	2016.0	Biography|Drama	English	USA	

	Title	Year	Genres	Language	Country	Conte Ratir
4863	The Boss	2016.0	Comedy	English	USA	
4864	The Boy	2016.0	Horror|Mystery|Thriller	English	USA	PG-1
4865	The Conjuring 2	2016.0	Horror|Mystery|Thriller	English	USA	
4866	The Dog Lover	2016.0	Drama	English	USA	F
4867	The Finest Hours	2016.0	Action|Drama|History|Thriller	English	USA	PG-1
4868	The Forest	2016.0	Horror|Mystery|Thriller	English	USA	PG-1
4869	The Huntsman: Winter's War	2016.0	Action|Adventure|Drama|Fantasy	English	USA	PG-1
4870	The Infiltrator	2016.0	Biography|Crime|Drama|Thriller	English	UK	
4871	The Jungle Book	2016.0	Adventure|Drama|Family|Fantasy	English	UK	F
4872	The Jungle Book	2016.0	Adventure|Drama|Family|Fantasy	English	UK	F
4873	The Legend of Tarzan	2016.0	Action|Adventure|Drama|Romance	English	USA	PG-1
4874	The Little Ponderosa Zoo	2016.0	Family	English	USA	Na
4875	The Masked Saint	2016.0	Action|Biography|Crime|Drama|Family|Fantasy	English	Canada	PG-1
4876	The Neon Demon	2016.0	Horror|Thriller	English	France	
4877	The Perfect Match	2016.0	Comedy|Romance	English	USA	
4878	The Purge: Election Year	2016.0	Action|Horror|Sci-Fi|Thriller	English	France	
4879	The Secret Life of Pets	2016.0	Animation|Comedy|Family	English	Japan	F
4880	The Shallows	2016.0	Drama|Horror|Thriller	English	USA	PG-1
4881	The Veil	2016.0	Horror	English	USA	
4882	The Wailing	2016.0	Fantasy|Horror|Mystery|Thriller	Korean	South Korea	N Rate
4883	The Young Messiah	2016.0	Drama	English	USA	PG-1
4884	Triple 9	2016.0	Action|Crime|Drama|Thriller	English	USA	
4885	Two Lovers and a Bear	2016.0	Drama|Romance	English	Canada	Na
4886	Warcraft	2016.0	Action|Adventure|Fantasy	English	USA	PG-1

	Title	Year	Genres	Language	Country	Conte Ratir
4887	Xi you ji zhi: Sun Wukong san da Baigu Jing	2016.0	Action|Adventure|Fantasy	English	China	Na
4888	X-Men: Apocalypse	2016.0	Action|Adventure|Sci-Fi	English	USA	PG-1
4889	Yoga Hosers	2016.0	Comedy|Fantasy|Horror|Thriller	English	USA	PG-1
4890	Zoolander 2	2016.0	Comedy	English	USA	PG-1
4891	10,000 B.C.	NaN	Comedy	NaN	NaN	Na
4892	12 Monkeys	NaN	Adventure|Drama|Mystery|Sci-Fi|Thriller	English	USA	TV-1
4893	3rd Rock from the Sun	NaN	Comedy|Family|Sci-Fi	English	USA	TV-P
4894	A Touch of Frost	NaN	Crime|Drama|Mystery	English	UK	Na
4895	Anger Management	NaN	Comedy|Romance	English	USA	Na
4896	Animal Kingdom	NaN	Crime|Drama	English	USA	Na
4897	Anne of Green Gables	NaN	Drama|Family	English	Canada	TV-
4898	Arthur	NaN	Animation|Comedy|Family	English	Canada	TV
4899	Bewitched	NaN	Comedy|Family|Fantasy	English	USA	TV-

60 rows × 25 columns

Look again through the trend of the results obtained above. There's a reason i extracted that much of info. Look throught the date column, that it's arranged in ascending order. This is a matter of unrecorded data, not that it doesn't exist. Moreover, the missing values in the Year column are from the 3rd sheet of the Excel file

Assuming you contacted the source of your data and it confirmed the year of those movies as 2016. so fill in 2016 for the missing values in that column

In [31]:
```
pd.value_counts(dataset.Year).sort_index()
```

Out[31]:
```
1916.0      1
1920.0      1
1925.0      1
1927.0      1
1929.0      2
           ...
2012.0    218
2013.0    236
2014.0    248
2015.0    222
2016.0    103
Name: Year, Length: 91, dtype: int64
```

In [32]:
```
dataset.Year.fillna(method = "ffill", inplace = True)
display(dataset.Year.isnull().sum())
```

```
0
```

In [33]:
```
null_lang = dataset[dataset.Language.isnull()]
null_lang
```

Out[33]:

	Title	Year	Genres	Language	Country	C
0	Intolerance: Love's Struggle Throughout the Ages	1916.0	Drama|History|War	NaN	USA	
1	Over the Hill to the Poorhouse	1920.0	Crime|Drama	NaN	USA	
2	The Big Parade	1925.0	Drama|Romance|War	NaN	USA	
206	Silent Movie	1976.0	Comedy|Romance	NaN	USA	
2602	Love's Abiding Joy	2006.0	Drama|Family|Western	NaN	USA	
2851	September Dawn	2007.0	Drama|History|Romance|Western	NaN	USA	
4322	A Fine Step	2014.0	Drama	NaN	USA	
4333	Alpha and Omega 4: The Legend of the Saw Tooth...	2014.0	Action|Adventure|Animation|Comedy|Drama|Family...	NaN	USA	
4831	Kickboxer: Vengeance	2016.0	Action	NaN	USA	
4891	10,000 B.C.	2016.0	Comedy	NaN	NaN	
4989	Unforgettable	2016.0	Drama|Mystery	NaN	USA	

11 rows × 25 columns

A higher percentage of the movies made in USA are in English Language. And upon confirming from the Data source, that's true except for only one.

So, delete, that one and fill the language of the rest as english

In [34]:
```
dataset.Language.fillna("English", inplace = True )
dataset.drop( [4891], inplace = True )
display(dataset.Language.isnull().sum())
```

0

In [35]:
```
null_country = dataset[dataset.Country.isnull()]
null_country
```

Out[35]:

	Title	Year	Genres	Language	Country	Content Rating	Duration	Asp R
4368	Dawn Patrol	2014.0	Drama|Thriller	English	NaN	NaN	88.0	2
4923	Gone, Baby, Gone	2016.0	Comedy|Drama|Reality-TV|Romance	English	NaN	TV-14	43.0	1
4945	Preacher	2016.0	Adventure|Drama|Fantasy|Mystery	English	NaN	TV-MA	60.0	16

3 rows × 25 columns

The missing values for country corresponds to English under the language column. And upon confirmation from the data source, the missing values under Country column was confirmed to be "USA". So, fill it in appropriately.

In [36]:
```
dataset.Country.fillna("USA", inplace = True)
display(dataset.Country.isnull().sum())
```

0

In [37]:
```
null_duration = dataset[dataset.Duration.isnull()]
null_duration
```

Out[37]:

	Title	Year	Genres	Language	Country	Content Rating	Duration	As F
1387	Hum To Mohabbat Karega	2000.0	Action|Comedy|Romance|Thriller	Hindi	India	NaN	NaN	
2303	Dil Jo Bhi Kahey...	2005.0	Romance	English	India	NaN	NaN	
2689	The Naked Ape	2006.0	Comedy|Drama	English	USA	NaN	NaN	
3183	Black Water Transit	2009.0	Crime|Drama	English	USA	NaN	NaN	
3479	Harry Potter and the Deathly Hallows: Part I	2010.0	Fantasy	English	UK	NaN	NaN	
3530	N-Secure	2010.0	Crime|Drama|Thriller	English	USA	R	NaN	
3704	Harry Potter and the Deathly Hallows: Part II	2011.0	Action|Fantasy	English	UK	NaN	NaN	
3996	Should've Been Romeo	2012.0	Comedy|Drama	English	USA	NaN	NaN	
4113	Barfi	2013.0	Comedy|Romance	Kannada	India	NaN	NaN	
4371	Destiny	2014.0	Action|Adventure|Fantasy|Sci-Fi	English	USA	NaN	NaN	
4663	Karachi se Lahore	2015.0	Comedy|Family	Urdu	Pakistan	NaN	NaN	
4696	Romantic Schemer	2015.0	Romance	English	USA	PG-13	NaN	
4992	War & Peace	2016.0	Drama|History|Romance|War	English	UK	TV-14	NaN	1
4994	Wolf Creek	2016.0	Drama|Horror|Thriller	English	Australia	NaN	NaN	

14 rows × 25 columns

After contacting your data supplier, assuming you're provided with the corresponding duration of the missing values. So, you'll fill it in accordingly. In this case, the missing values under Duration column will be filled with the column median value.

In [38]:
```
duration_median = dataset.Duration.median()
dataset.Duration.fillna(duration_median, inplace = True )
display(dataset.Duration.isnull().sum())
```

0

In [39]:
```
null_actor1 = dataset[dataset["Actor 1"].isnull()]
null_actor1
```

Out[39]:

	Title	Year	Genres	Language	Country	Content Rating	Duration	Aspect Ratio	B
1513	Ayurveda: Art of Being	2001.0	Documentary	English	India	NaN	102.0	1.85	30(
1817	Sex with Strangers	2002.0	Documentary|Drama	English	USA	NaN	105.0	1.33	
2411	The Blood of My Brother	2005.0	Documentary|War	English	USA	NaN	90.0	1.66	12(
3759	Pink Ribbons, Inc.	2011.0	Documentary	English	Canada	Not Rated	97.0	NaN	120(
3819	The Harvest/La Cosecha	2011.0	Documentary	English	USA	NaN	80.0	NaN	56(
4251	The Brain That Sings	2013.0	Documentary|Family	Arabic	United Arab Emirates	NaN	62.0	NaN	12[
4611	Counting	2015.0	Documentary	English	USA	NaN	111.0	1.78	5(

7 rows × 25 columns

Upon contacting the Data supplier, there's no provision for the missing values of Actor 1, Actor 2, Actor 3.

The next step will be to drop the missing values in those columns

In [40]:
```
dataset.drop(null_actor1.index, inplace = True )
```

In [41]:
```
null_actor2 = dataset[dataset["Actor 2"].isnull()]
null_actor2
```

Out[41]:

	Title	Year	Genres	Language	Country	Content Rating	Dura
4102	All Is Lost	2013.0	Action|Adventure|Drama	English	USA	PG-13	1(
4119	Bending Steel	2013.0	Documentary	English	USA	NaN	!
4598	Censored Voices	2015.0	Documentary|History	Hebrew	Israel	NaN	8
4966	The Bachelor	2016.0	Game-Show|Reality-TV|Romance	English	USA	NaN	(
4996	Yu-Gi-Oh! Duel Monsters	2016.0	Action|Adventure|Animation|Family|Fantasy	Japanese	Japan	NaN	2

5 rows × 25 columns

In [42]:
```
dataset.drop(null_actor2.index, inplace = True )
```

In [43]:
```
null_actor3 = dataset[dataset["Actor 3"].isnull()]
null_actor3
```

Out[43]:

	Title	Year	Genres	Language	Country	Content Rating	Duration
22	Fantasia	1940.0	Animation|Family|Fantasy|Music	English	USA	G	120.0
162	Pink Narcissus	1971.0	Drama|Fantasy	English	USA	Not Rated	65.0
1678	Winged Migration	2001.0	Documentary	English	France	G	81.0
1753	Gerry	2002.0	Adventure|Drama|Mystery	English	USA	R	103.0
2397	Sisters in Law	2005.0	Documentary	English	Cameroon	Not Rated	104.0
2499	An Inconvenient Truth	2006.0	Documentary	English	USA	PG	96.0
2961	Dolphins and Whales 3D: Tribes of the Ocean	2008.0	Adventure|Documentary|Short	English	UK	NaN	42.0
4696	Romantic Schemer	2015.0	Romance	English	USA	PG-13	103.0
4985	The Streets of San Francisco	2016.0	Action|Crime|Drama|Mystery	English	USA	NaN	120.0

9 rows × 25 columns

In [44]:
```
dataset.drop(null_actor3.index, inplace = True )
```

In [45]:
```
null_director = dataset[dataset["Facebook Likes - Director"].isnull()]
null_index = null_director.index
```

Upon contacting the data supplier, you couldn't get concrete figures about the missing values for this column. So you decided to treat it with discretion. And as such, all the null values in this column will be deleted

In [46]:
```
dataset.drop(null_index, inplace = True)
```

In [47]:
```
null_user_review = dataset[dataset["Reviews by Users"].isnull()]
null_user_review
```

Out[47]:

	Title	Year	Genres	Language	Country	Content Rating	Dura
2805	Jesus People	2007.0	Comedy|Short	English	USA	NaN	
2902	The Touch	2007.0	Romance|Short	English	USA	NaN	
2951	Childless	2008.0	Drama	English	USA	R	
3183	Black Water Transit	2009.0	Crime|Drama	English	USA	NaN	1
3448	Death Calls	2010.0	Action|Adventure|Mystery|Romance|Thriller	English	USA	R	
4311	Water & Power	2013.0	Crime|Drama	English	USA	Not Rated	
4337	Amidst the Devil's Wings	2014.0	Action|Crime|Drama	English	USA	NaN	
4463	Perfect Cowboy	2014.0	Drama	English	USA	NaN	1
4577	America Is Still the Place	2015.0	History	English	USA	NaN	
4673	Me You and Five Bucks	2015.0	Comedy|Drama|Romance	English	USA	NaN	
4700	Running Forever	2015.0	Family	English	USA	NaN	
4770	To Be Frank, Sinatra at 100	2015.0	Documentary	English	UK	NaN	
4790	A Beginner's Guide to Snuff	2016.0	Comedy|Horror|Thriller	English	USA	NaN	
4912	Del 1 - Män som hatar kvinnor	2016.0	Action|Crime|Mystery|Thriller	Swedish	Sweden	NaN	
4986	Towering Inferno	2016.0	Comedy	English	Canada	NaN	

15 rows × 25 columns

The data supplier could not provide the missing values for this column, hence, the null values will be dropped

In [48]:
```
dataset.drop(null_user_review.index, inplace = True)
```

In [49]:
```
null_critic_review = dataset[dataset["Reviews by Crtiics"].isnull()]
null_critic_review
```

Out[49]:

	Title	Year	Genres	Language	Country	Content Rating	Duration	As F
336	The Ballad of Gregorio Cortez	1982.0	Western	English	USA	NaN	105.0	
1128	The Love Letter	1998.0	Fantasy|Romance	English	USA	Unrated	99.0	
2127	Guiana 1838	2004.0	Drama	English	USA	Unrated	120.0	
2171	On the Downlow	2004.0	Drama	English	USA	NaN	84.0	
2342	Insomnia Manica	2005.0	Thriller	English	USA	NaN	127.0	
2440	The Mongol King	2005.0	Crime|Drama	English	USA	PG-13	84.0	
2689	The Naked Ape	2006.0	Comedy|Drama	English	USA	NaN	103.0	
2743	Arnolds Park	2007.0	Mystery|Thriller	English	USA	PG-13	103.0	
3233	Flying By	2009.0	Drama|Family|Music	English	USA	PG-13	90.0	
3326	Steppin: The Movie	2009.0	Comedy|Music	English	USA	PG-13	138.0	
3339	The Deported	2009.0	Comedy	English	USA	PG-13	90.0	
3508	Lies in Plain Sight	2010.0	Drama	English	USA	TV-PG	89.0	
3831	The Ridges	2011.0	Drama|Horror|Thriller	English	USA	NaN	143.0	
3851	We Have Your Husband	2011.0	Crime|Drama|Thriller	English	USA	TV-PG	87.0	
4094	A True Story	2013.0	Comedy	English	USA	NaN	96.0	
4169	Her Cry: La Llorona Investigation	2013.0	Horror	English	USA	Not Rated	89.0	
4321	8 Days	2014.0	Drama|Thriller	English	USA	PG-13	90.0	
4361	Butterfly Girl	2014.0	Documentary	English	USA	NaN	78.0	
4435	Light from the Darkroom	2014.0	Action|Drama|Thriller	English	USA	PG-13	90.0	
4471	Rise of the Entrepreneur: The Search for a Bet...	2014.0	Documentary	English	USA	G	52.0	
4571	Abandoned	2015.0	Drama	English	New Zealand	NaN	86.0	
4625	Dutch Kills	2015.0	Crime|Drama|Thriller	English	USA	NaN	90.0	
4643	Growing Up Smith	2015.0	Comedy|Drama|Family	English	USA	PG-13	102.0	

	Title	Year	Genres	Language	Country	Content Rating	Duration	As F
4722	Teeth and Blood	2015.0	Horror	English	USA	NaN	96.0	1
4822	Hands of Stone	2016.0	Action|Biography|Drama|Sport	English	Panama	R	105.0	

25 rows × 25 columns

The data supplier couldn't make provisions for the missing values in this column as well. Hence, it'll be dropped

In [50]:
```
dataset.drop(null_critic_review.index, inplace = True)
```

Both "Content Rating" and "Aspect Ratio" columns which have missing values will be dropped. The Data supplier couldn't make provisions for the missing values in both cases

In [51]:
```
null_content_rating = dataset[dataset["Content Rating"].isnull()]
dataset.drop(null_content_rating.index, inplace = True)
```

In [52]:
```
null_aspect_ratio = dataset[dataset["Aspect Ratio"].isnull()]
dataset.drop(null_aspect_ratio.index, inplace = True)
```

In [53]:
```
null_facenumbers = dataset[dataset["Facenumber in posters"].isnull()]
facenumbers_index = null_facenumbers.index
```

When the data supplier was contacted for info on the missing values in this column, his response is a value equal to mean of the column. And the mean value for the column will be filled in for null values accordingly

In [54]:
```
dataset["Facenumber in posters"].fillna(dataset["Facenumber in poster
s"].mean(), inplace = True)
```

There are only 2 columns left containing null values. First we'll drop the Budget columns, so the null values in the "Gross Earnings column will be reduced. Next, we'll fill the remaining null values with the median values of the Gross Earning column

In [55]:
```
print("Number of missing values in Gross Earnings\' column before dele
ting Budget Column :", dataset["Gross Earnings"].isnull().sum())
null_budget = dataset[dataset.Budget.isnull()]
dataset.drop(null_budget.index, inplace = True)
print("Number of missing values in Gross Earnings\' column after delet
ing Budget Column :", dataset["Gross Earnings"].isnull().sum())
```

```
Number of missing values in Gross Earnings' column before deleting Bud
get Column : 461
Number of missing values in Gross Earnings' column after deleting Budg
et Column : 404
```

In [56]:
```
dataset["Gross Earnings"].fillna(dataset["Gross Earnings"].median(), i
nplace = True)
```

Note that most of the values dropped in this tutorial are for teaching or example sake. There's a certain values (mostly, less than 10% of the total data) that can be dropped. Any other scenario should be treated with great care. This will help you avoid the trap of losing valuable chunks of data.

In [57]:
```
dataset.isna().sum()
```

Out[57]:
```
Title                          0
Year                           0
Genres                         0
Language                       0
Country                        0
Content Rating                 0
Duration                       0
Aspect Ratio                   0
Budget                         0
Gross Earnings                 0
Director                       0
Actor 1                        0
Actor 2                        0
Actor 3                        0
Facebook Likes - Director      0
Facebook Likes - Actor 1       0
Facebook Likes - Actor 2       0
Facebook Likes - Actor 3       0
Facebook Likes - cast Total    0
Facebook likes - Movie         0
Facenumber in posters          0
User Votes                     0
Reviews by Users               0
Reviews by Crtiics             0
IMDB Score                     0
dtype: int64
```

In [58]:
```
pd.value_counts(dataset["Facenumber in posters"])
```

Out[58]:
```
0.000000     1768
1.000000     1046
2.000000      592
3.000000      314
4.000000      174
5.000000       84
6.000000       58
7.000000       34
8.000000       33
9.000000       11
10.000000       8
1.351131        8
11.000000       5
15.000000       4
12.000000       4
13.000000       2
14.000000       1
19.000000       1
31.000000       1
43.000000       1
Name: Facenumber in posters, dtype: int64
```

All the missing values in the dataset have been treated. The data is clean and is ready for further exploration and analysis.

But this has been a fair lesson to wrap your head around and lay your hands on.

This document will be saved into an excel file and the exploration, analysis and insights and visualization will be saved for the second part. Make sure to check it out

In [59]:
```
dataset.index
```

Out[59]:
```
Int64Index([   0,    2,    3,    5,    6,    7,    8,    9,   10,   1
1,
            ...
            4876, 4878, 4879, 4880, 4881, 4883, 4884, 4886, 4888, 489
0],
           dtype='int64', length=4149)
```

The index column has been altered again as a result of data cleaning process. Let's rewrite it in an ordered manner

In [60]:
```
final_index = range(1,4150)
dataset.index = final_index
dataset.index
```

Out[60]:
```
RangeIndex(start=1, stop=4150, step=1)
```

In [61]:
```
dataset.shape
```

Out[61]:
```
(4149, 25)
```

In [62]:
```
dataset.size
```

Out[62]:
```
103725
```

In [63]:
```
dataset.dtypes
```

Out[63]:
```
Title                          object
Year                          float64
Genres                         object
Language                       object
Country                        object
Content Rating                 object
Duration                      float64
Aspect Ratio                  float64
Budget                        float64
Gross Earnings                float64
Director                       object
Actor 1                        object
Actor 2                        object
Actor 3                        object
Facebook Likes - Director     float64
Facebook Likes - Actor 1      float64
Facebook Likes - Actor 2      float64
Facebook Likes - Actor 3      float64
Facebook Likes - cast Total     int64
Facebook likes - Movie          int64
Facenumber in posters         float64
User Votes                      int64
Reviews by Users              float64
Reviews by Crtiics            float64
IMDB Score                    float64
dtype: object
```

In [64]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4149 entries, 1 to 4149
Data columns (total 25 columns):
Title                          4149 non-null object
Year                           4149 non-null float64
Genres                         4149 non-null object
Language                       4149 non-null object
Country                        4149 non-null object
Content Rating                 4149 non-null object
Duration                       4149 non-null float64
Aspect Ratio                   4149 non-null float64
Budget                         4149 non-null float64
Gross Earnings                 4149 non-null float64
Director                       4149 non-null object
Actor 1                        4149 non-null object
Actor 2                        4149 non-null object
Actor 3                        4149 non-null object
Facebook Likes - Director      4149 non-null float64
Facebook Likes - Actor 1       4149 non-null float64
Facebook Likes - Actor 2       4149 non-null float64
Facebook Likes - Actor 3       4149 non-null float64
Facebook Likes - cast Total    4149 non-null int64
Facebook likes - Movie         4149 non-null int64
Facenumber in posters          4149 non-null float64
User Votes                     4149 non-null int64
Reviews by Users               4149 non-null float64
Reviews by Crtiics             4149 non-null float64
IMDB Score                     4149 non-null float64
dtypes: float64(13), int64(3), object(9)
memory usage: 664.6+ KB
```

In [65]:
```
dataset.describe()
```

Out[65]:

	Year	Duration	Aspect Ratio	Budget	Gross Earnings	Facebook Likes - Director	Faceb
count	4149.000000	4149.000000	4149.000000	4.149000e+03	4.149000e+03	4149.000000	414
mean	2001.688841	109.782598	2.113620	4.242499e+07	5.016976e+07	785.356471	725
std	12.482498	22.707194	0.558098	2.153378e+08	6.665142e+07	3015.762769	1511
min	1916.000000	20.000000	1.180000	2.180000e+02	1.620000e+02	0.000000	
25%	1998.000000	95.000000	1.850000	8.000000e+06	1.021401e+07	10.000000	69
50%	2004.000000	105.000000	2.350000	2.000000e+07	2.997598e+07	58.000000	100
75%	2010.000000	120.000000	2.350000	4.800000e+07	6.065204e+07	222.000000	1200
max	2016.000000	330.000000	16.000000	1.221550e+10	7.605058e+08	23000.000000	64000

In [66]:
```
dataset.to_excel("movies_cleaned_for_part2.xlsx", index = False)
```

So, this marks the end of another lesson in the process of Data Science using Python Libraries like Numpy, Pandas and Matplotlib.

The simplest way to learn is by doing, so roll up your sleeves and get to work. Be encouraged no matter the progress you're making, keep doing it even if it means doing it poorly till you get it right.

It's only a matter of some more practice before you actually get it right. With a little more of commitment, it'll all come naturally and you'll gain mastery of the process and system

Till i bring another lesson your way,

Happy Learning ! ! !

In [ ]: