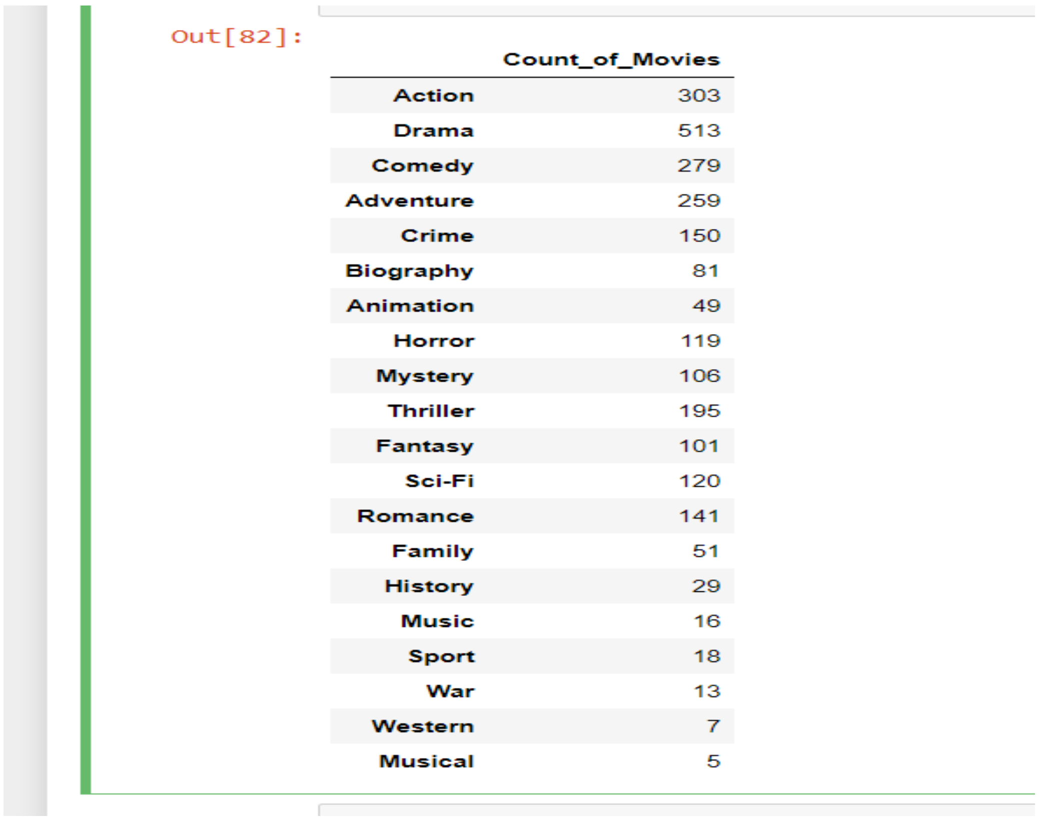
Assignment: Data Cleanup

- Using the IMDB-Movie-Data.csv dataset:
    - Tabulate the number of movies in each Genre
        - Note the multi-value format!

```
import pandas as pd

filename = "D:/Spring_2021/AIT 580_Prof.Harry Foxwell/8. week 8 _data cleaning and project 2/IMDB-Movie-Data.csv"

df3 = pd.read_csv(filename)

df_splittedGenre= df3['Genre'].str.split(';').apply(pd.Series)

df_splittedGenre.head()

df_splittedGenre.columns = ['Genre1','Genre2','Genre3']

df3.head()

df_splittedGenre

df3 = pd.concat([df3, df_splittedGenre],axis=1)

df3.drop("Genre", axis = 1, inplace=True)

df8=df3['Genre2'].value_counts().to_frame()

df5=df3['Genre1'].value_counts().to_frame()

df6=df3['Genre3'].value_counts().to_frame()

df8['Genre1'].replace(np.nan,0,inplace=True)

df8['Genre2'].replace(np.nan,0,inplace=True)

df8['Genre3'].replace(np.nan,0,inplace=True)

df8.sum()

df9=df8.T

df9.sum()

df9.dtypes

df10=df9.sum().to_frame()

df11=df10.astype('int')

df11.columns=['Count_of_Movies']

df11
```



Out[82]:

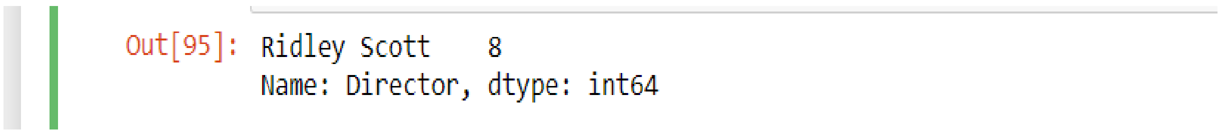
	Count_of_Movies
Action	303
Drama	513
Comedy	279
Adventure	259
Crime	150
Biography	81
Animation	49
Horror	119
Mystery	106
Thriller	195
Fantasy	101
Sci-Fi	120
Romance	141
Family	51
History	29
Music	16
Sport	18
War	13
Western	7
Musical	5

These are the count of movies in each genre with highest in Drama and least in Musical.

- Which Director directed the most movies?

(df3['Director'] == ' ').value_counts().head(5)

(df3['Director']).value_counts().head(1)



```
Out[95]: Ridley Scott    8
         Name: Director, dtype: int64
```

Ridley Scott has directed most movies with a count of 8

- Which Actor acted in the most movies?

dfa=df3['Actor1'].str.strip().value_counts().to_frame()

dfb=df3['Actor2'].str.strip().value_counts().to_frame()

dfc=df3['Actor3'].str.strip().value_counts().to_frame()

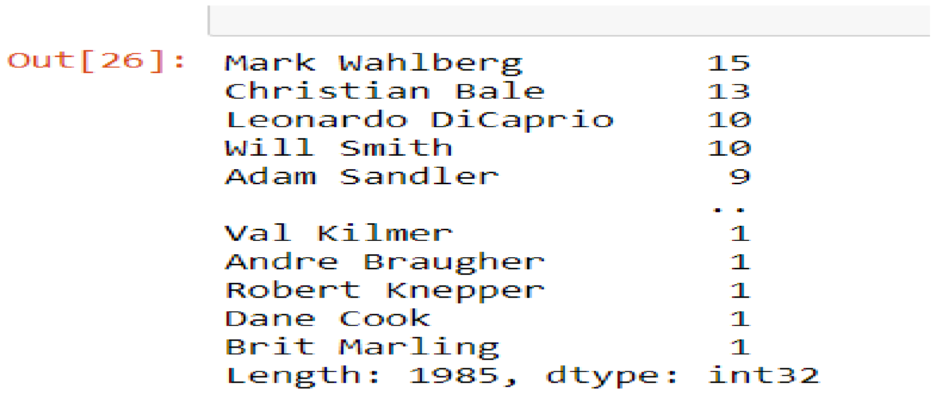
dfd=df3['Actor4'].str.strip().value_counts().to_frame()

dfe=pd.concat([dfa,dfb,dfc,dfd],axis=1)

t=dfe.T.sum()

df17=t.astype('int')

df17[0:]



```
Out[26]:  Mark Wahlberg         15
          Christian Bale        13
          Leonardo DiCaprio     10
          Will Smith            10
          Adam Sandler           9
                                ..
          Val Kilmer             1
          Andre Braugher         1
          Robert Knepper         1
          Dane Cook              1
          Brit Marling           1
          Length: 1985, dtype: int32
```

This shows the actors who acted in most movies in descending order. Mark Wahlberg acted in most movies with a count of 15.

- List the movies in which *Anthony Hopkins* appears

a1=df3['Actor1'].str.strip()

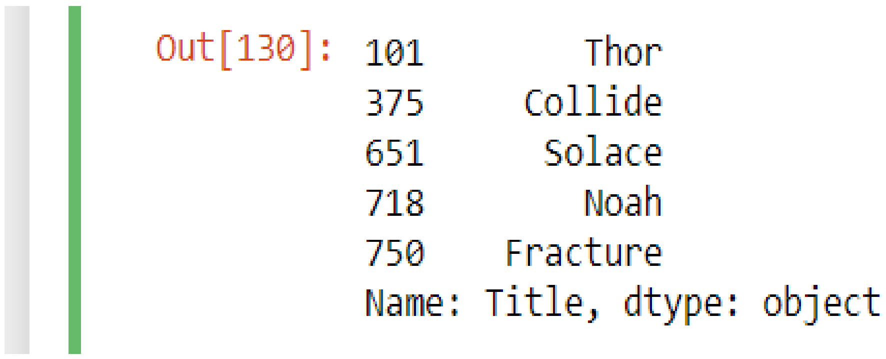
a2=df3['Actor2'].str.strip()

a3=df3['Actor3'].str.strip()

a4=df3['Actor4'].str.strip()

movie_anthony=df3.query('Actor1.str.strip() == "Anthony Hopkins" or Actor2.str.strip() == "Anthony Hopkins" or Actor3.str.strip() == "Anthony Hopkins" or Actor4.str.strip() == " Anthony Hopkins "')

movie_anthony['Title']



```
Out[130]: 101         Thor
          375      Collide
          651       Solace
          718         Noah
          750     Fracture
          Name: Title, dtype: object
```

The movies in which Anthony Hopkins acted are a total of 5 namely Thor, Collide, Solace, Noah, Fracture

- List the movies that involve an *alien*

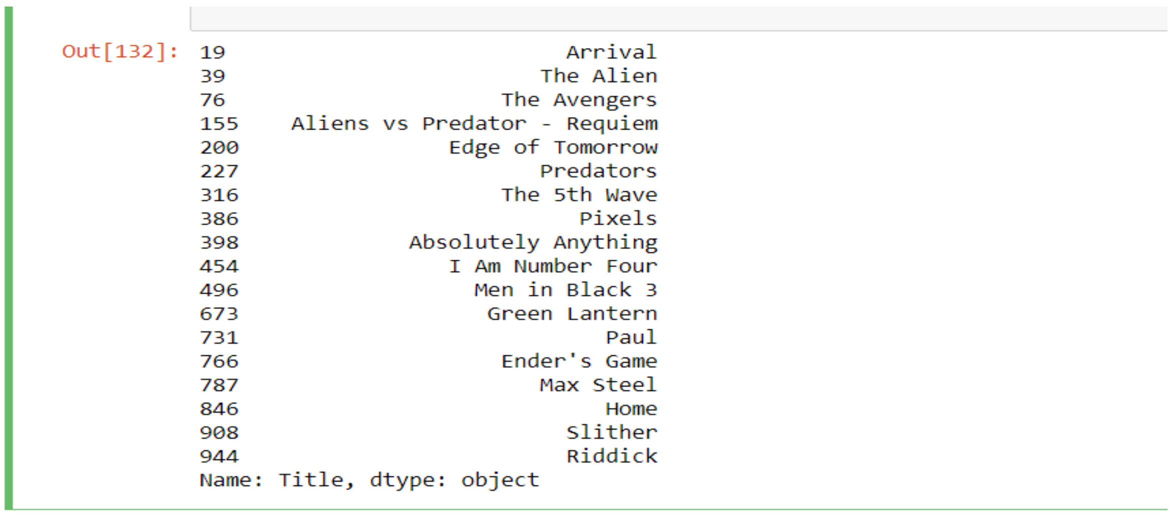
df3["Title"] = df3["Title"].apply(lambda x: x.replace("5/25/77", "The Alien"))

# Used a new title ‘The Alien’ as this movie’s description contained alien and this cell is incorrectly populated with some date in the Title column. I haven’t deleted the row because may be this movie had the highest IMDB rating which is what we predict in this dataset and we shouldn’t lose this data. We should consult the people or organization who has given us the data and correct it and proceed further. As of now, I have added a new title ‘The Alien’ in the place of wrongly populated date in the Title column.

movie_anthony=df3["Description"].str.contains('alien',case=False)

movie_anthony.value_counts()

df3[movie_anthony[0:] == True].Title

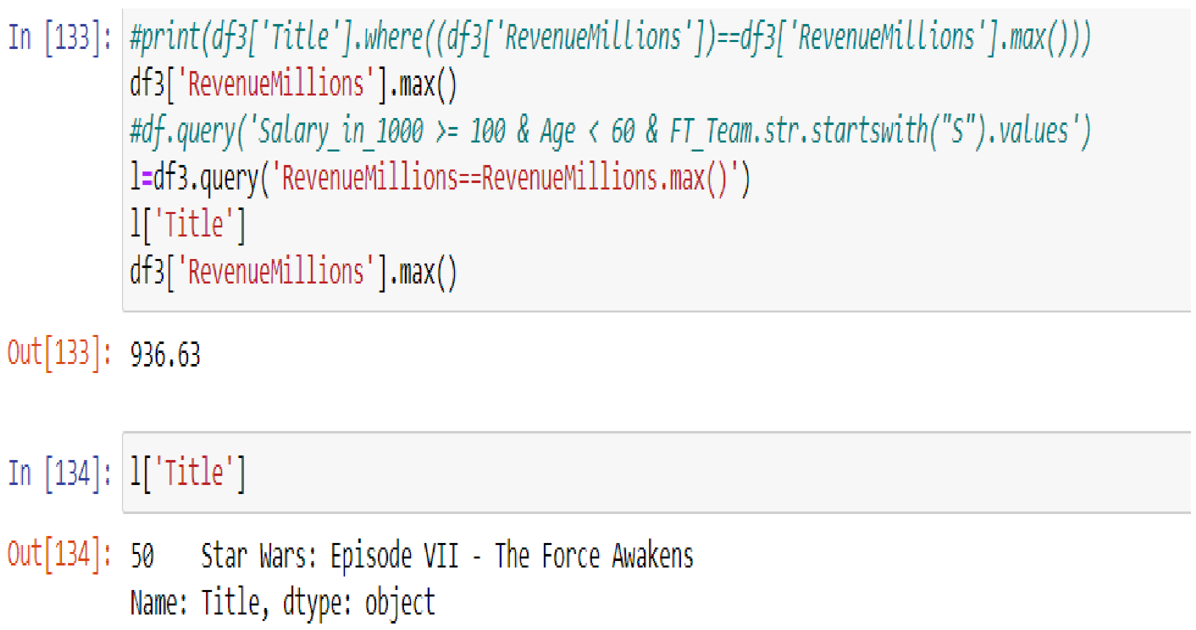


```
Out[132]: 19                             Arrival
          39                           The Alien
          76                        The Avengers
          155      Aliens vs Predator - Requiem
          200                  Edge of Tomorrow
          227                          Predators
          316                       The 5th Wave
          386                             Pixels
          398                 Absolutely Anything
          454                   I Am Number Four
          496                    Men in Black 3
          673                     Green Lantern
          731                               Paul
          766                       Ender's Game
          787                          Max Steel
          846                               Home
          908                            Slither
          944                            Riddick
          Name: Title, dtype: object
```

The above mentioned movies involved an Alien

- Which movie had the highest RevenueMillions?

  df3['RevenueMillions'].max()
  l=df3.query('RevenueMillions==RevenueMillions.max()')
  l['Title']



```
In [133]: #print(df3['Title'].where((df3['RevenueMillions'])==df3['RevenueMillions'].max()))
          df3['RevenueMillions'].max()
          #df.query('Salary_in_1000 >= 100 & Age < 60 & FT_Team.str.startswith("S").values')
          l=df3.query('RevenueMillions==RevenueMillions.max()')
          l['Title']
          df3['RevenueMillions'].max()

Out[133]: 936.63

In [134]: l['Title']

Out[134]: 50    Star Wars: Episode VII - The Force Awakens
          Name: Title, dtype: object
```

Star Wars: Episode VII – The Force Awakens, has the highest revenue with 936.63 millions.

- How could you estimate the RevenueMillions for the movie *The Last Airbender*?
  - Calculate a range of possible values

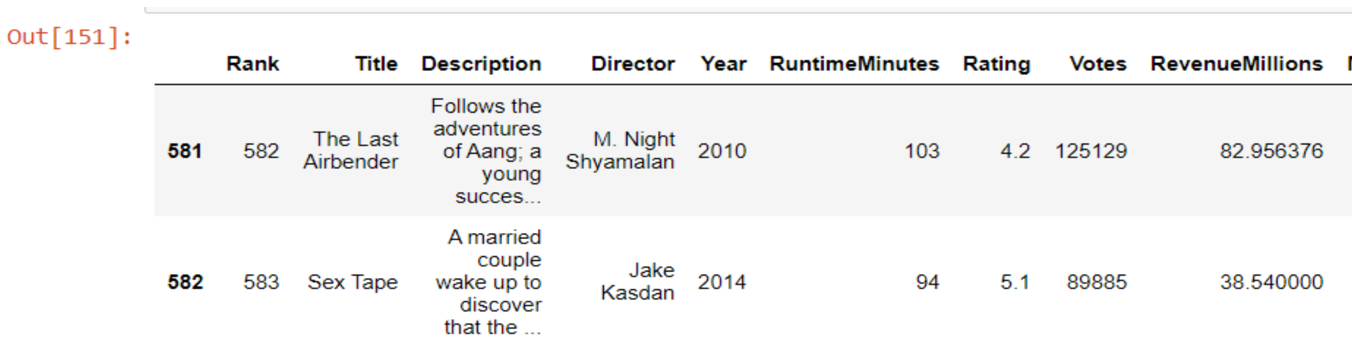
The range of possible values can be

a. The mean Revenue in millions - 82.96 M
b. The max Revenue in millions - 936.63 M
c. As The Last Air bender movie is released in 2010, the average revenue in that particular year can also be a possible value – 105.08 M
d. The runtime is 103 minutes for this movie, so average revenue with 103 minutes runtime can also be a possible value – 83.6 M
e. As The Last Air bender movie is released in 2010, the max revenue in that particular year can also be a possible value – 414.98 M
f. Similarly we can take the average or mode or max or min values of Revenue according to Ratings and Metascore also

   a. The mean Revenue in millions - 82.96 M

import numpy as np

```
df3['RevenueMillions'].max()

avg_revenue= df3['RevenueMillions'].mean(axis=0)

round(avg_revenue,2)

df3['RevenueMillions'].replace(np.nan,avg_revenue,inplace=True)

df3[581:583]
```

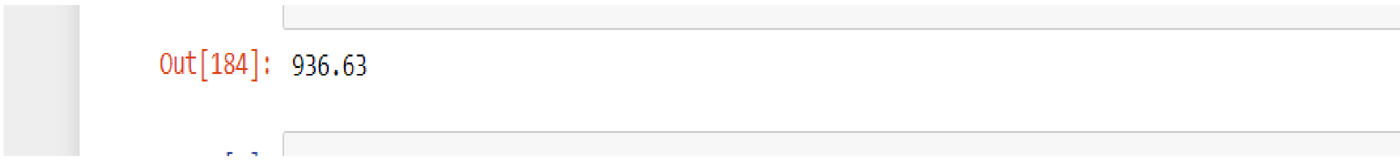


Out[151]:

	Rank	Title	Description	Director	Year	RuntimeMinutes	Rating	Votes	RevenueMillions
581	582	The Last Airbender	Follows the adventures of Aang; a young succes...	M. Night Shyamalan	2010	103	4.2	125129	82.956376
582	583	Sex Tape	A married couple wake up to discover that the ...	Jake Kasdan	2014	94	5.1	89885	38.540000

b. The max Revenue in millions - 936.63 M

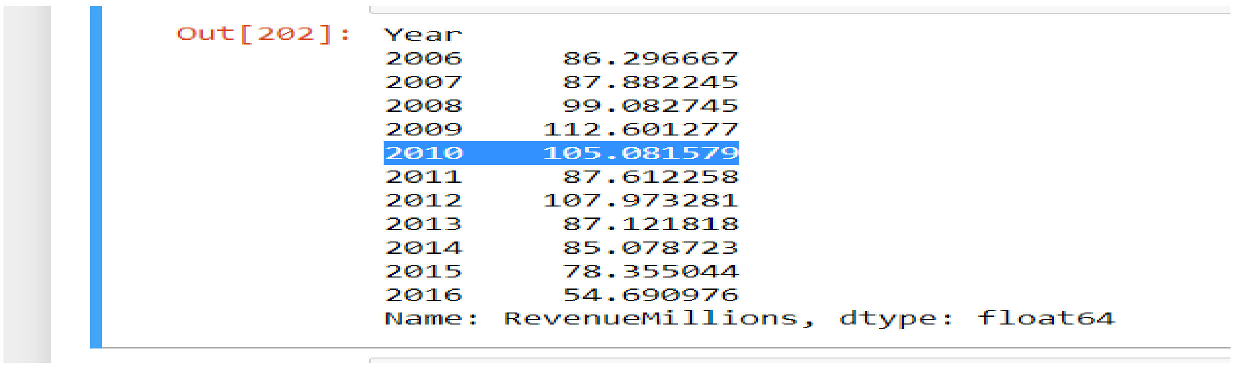
```
s=df3['RevenueMillions'].max()

s
```



```
Out[184]: 936.63
```

c. As The Last Air bender movie is released in 2010, the average revenue in that particular year can also be a possible value – 105.08 M

```
a=df3['Year'].to_frame()

b=df3['RevenueMillions'].to_frame()

df8=pd.concat([a,b],axis=1)

df8

df8.groupby('Year')['RevenueMillions'].mean()
```


```
Out[202]:  Year
           2006      86.296667
           2007      87.882245
           2008      99.082745
           2009     112.601277
           2010     105.081579
           2011      87.612258
           2012     107.973281
           2013      87.121818
           2014      85.078723
           2015      78.355044
           2016      54.690976
           Name: RevenueMillions, dtype: float64
```

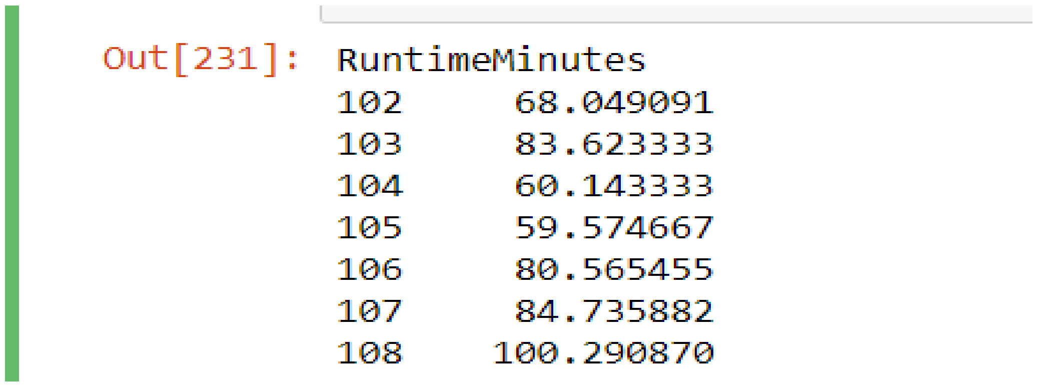
d. The runtime is 103 minutes for this movie, so average revenue with 103 minutes runtime can also be a possible value – 83.62M

u=df3['RuntimeMinutes'].to_frame()

df10=pd.concat([u,b],axis=1)

df10

df10.groupby('RuntimeMinutes')['RevenueMillions'].mean()[24:40]


```
Out[231]:  RuntimeMinutes
           102       68.049091
           103       83.623333
           104       60.143333
           105       59.574667
           106       80.565455
           107       84.735882
           108      100.290870
```

e. As The Last Air bender movie is released in 2010, the max revenue in that particular year can also be a possible value - 414.98 M

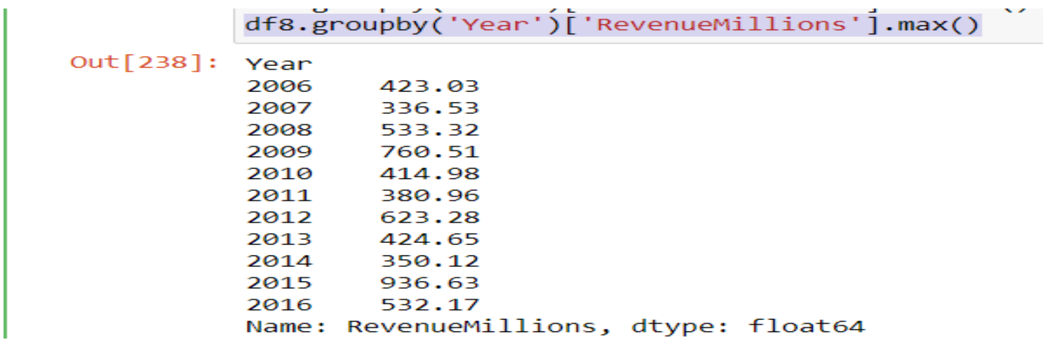
a=df3['Year'].to_frame()

b=df3['RevenueMillions'].to_frame()

df8=pd.concat([a,b],axis=1)

df8

df8.groupby('Year')['RevenueMillions'].max()



```
df8.groupby('Year')['RevenueMillions'].max()
Out[238]: Year
          2006    423.03
          2007    336.53
          2008    533.32
          2009    760.51
          2010    414.98
          2011    380.96
          2012    623.28
          2013    424.65
          2014    350.12
          2015    936.63
          2016    532.17
          Name: RevenueMillions, dtype: float64
```

- Using the movie_sample_dataset.csv dataset:
    - Identify and correct errors in the dataset

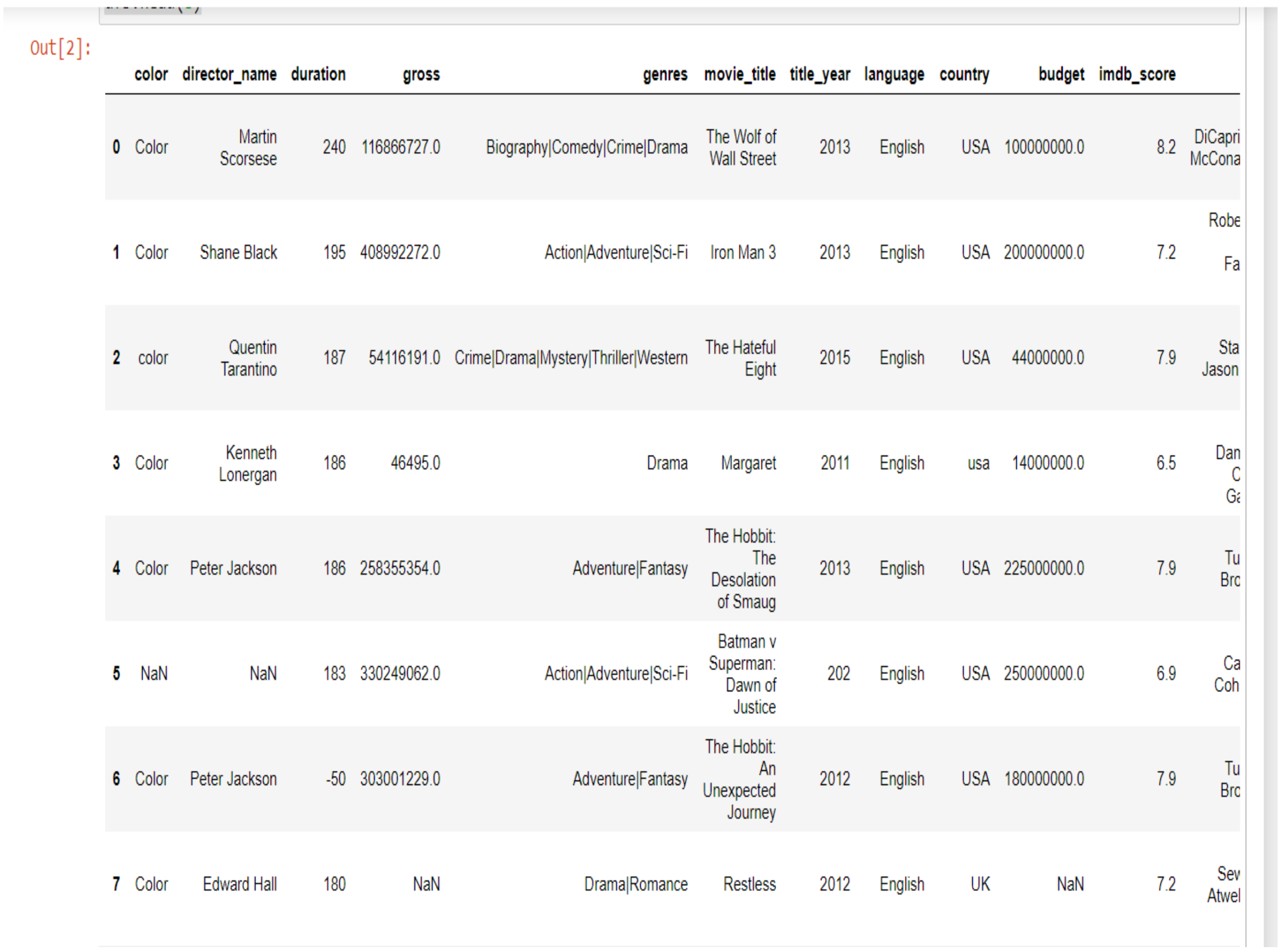
a. Loaded the Dataset

import pandas as pd

filename = "D:/Spring_2021/AIT 580_Prof.Harry Foxwell/8. week 8 _data cleaning and project 2/movie_sample_dataset.csv"

df3 = pd.read_csv(filename)

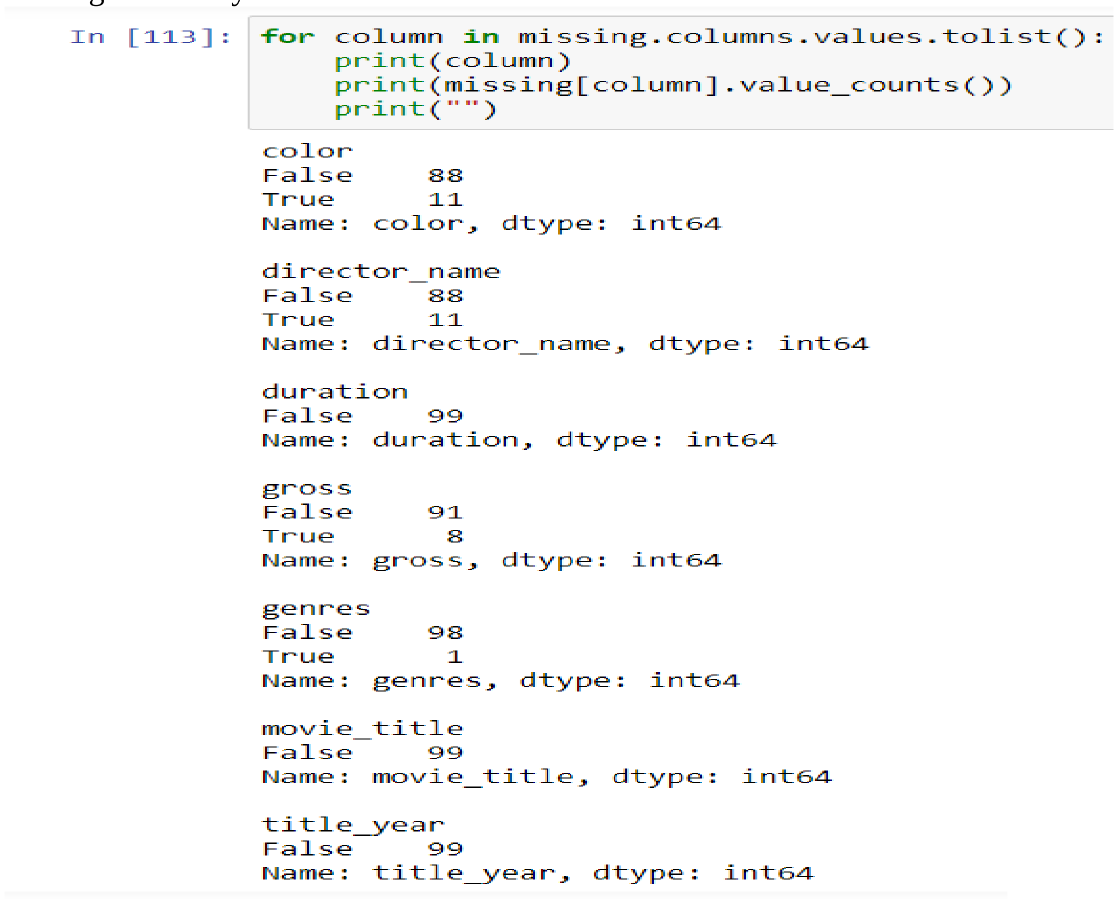
df3.head(8)



Out[2]:

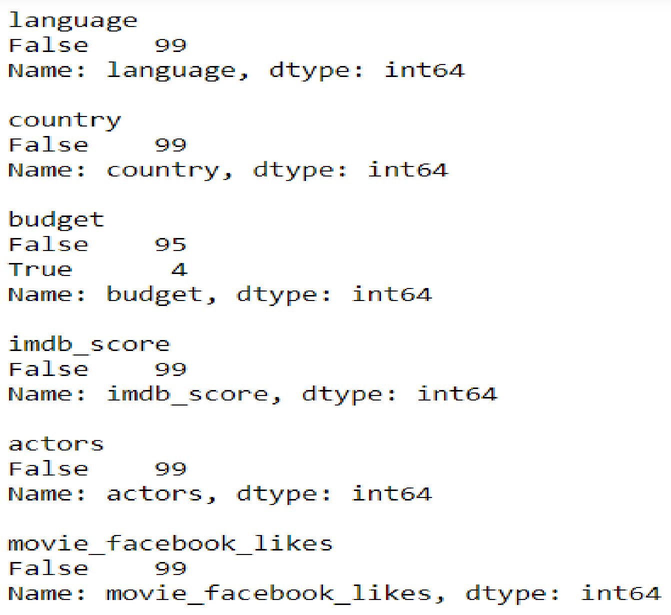
	color	director_name	duration	gross	genres	movie_title	title_year	language	country	budget	imdb_score	
0	Color	Martin Scorsese	240	116866727.0	Biography|Comedy|Crime|Drama	The Wolf of Wall Street	2013	English	USA	100000000.0	8.2	DiCapri McCona
1	Color	Shane Black	195	408992272.0	Action|Adventure|Sci-Fi	Iron Man 3	2013	English	USA	200000000.0	7.2	Robe Fa
2	color	Quentin Tarantino	187	54116191.0	Crime|Drama|Mystery|Thriller|Western	The Hateful Eight	2015	English	USA	44000000.0	7.9	Sta Jason
3	Color	Kenneth Lonergan	186	46495.0	Drama	Margaret	2011	English	usa	14000000.0	6.5	Dan C Gi
4	Color	Peter Jackson	186	258355354.0	Adventure|Fantasy	The Hobbit: The Desolation of Smaug	2013	English	USA	225000000.0	7.9	Tu Brc
5	NaN	NaN	183	330249062.0	Action|Adventure|Sci-Fi	Batman v Superman: Dawn of Justice	202	English	USA	250000000.0	6.9	Ca Coh
6	Color	Peter Jackson	-50	303001229.0	Adventure|Fantasy	The Hobbit: An Unexpected Journey	2012	English	USA	180000000.0	7.9	Tu Brc
7	Color	Edward Hall	180	NaN	Drama|Romance	Restless	2012	English	UK	NaN	7.2	Sev Atwel

b. Finding how many null values are there in each column



```
In [113]: for column in missing.columns.values.tolist():
              print(column)
              print(missing[column].value_counts())
              print("")
```

```
color
False    88
True     11
Name: color, dtype: int64

director_name
False    88
True     11
Name: director_name, dtype: int64

duration
False    99
Name: duration, dtype: int64

gross
False    91
True      8
Name: gross, dtype: int64

genres
False    98
True      1
Name: genres, dtype: int64

movie_title
False    99
Name: movie_title, dtype: int64

title_year
False    99
Name: title_year, dtype: int64
```



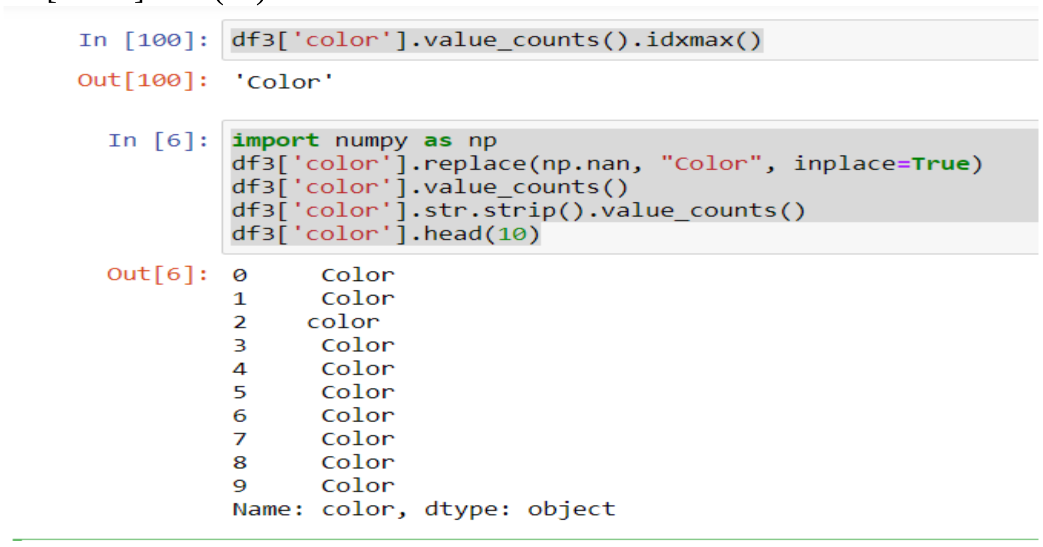
```
language
False    99
Name: language, dtype: int64

country
False    99
Name: country, dtype: int64

budget
False    95
True      4
Name: budget, dtype: int64

imdb_score
False    99
Name: imdb_score, dtype: int64

actors
False    99
Name: actors, dtype: int64

movie_facebook_likes
False    99
Name: movie_facebook_likes, dtype: int64
```

1. Color column

The most repeated color type of a Movie is 'Color'. So filled the empty cells in the first column with 'Color'

```
df3['color'].value_counts().idxmax()
import numpy as np
```

```
df3['color'].replace(np.nan, "Color", inplace=True)
df3['color'].value_counts()
df3['color'].str.strip().value_counts()
df3['color'].head(10)
```

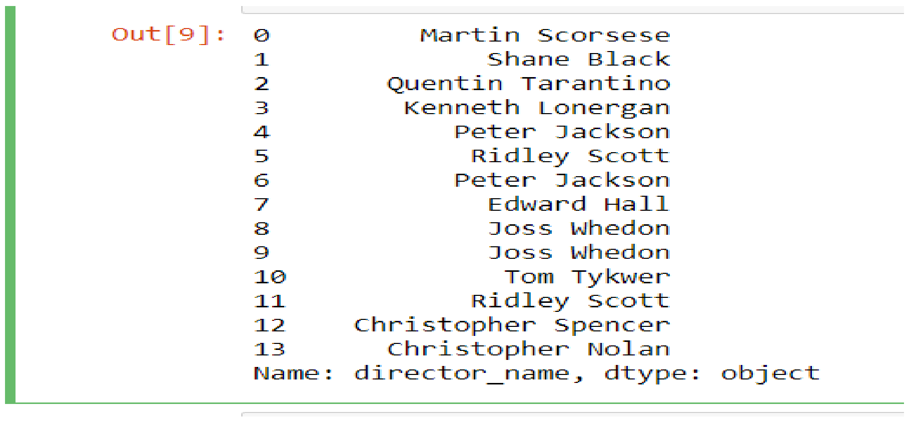


```
In [100]: df3['color'].value_counts().idxmax()

Out[100]: 'Color'

In [6]: import numpy as np
        df3['color'].replace(np.nan, "Color", inplace=True)
        df3['color'].value_counts()
        df3['color'].str.strip().value_counts()
        df3['color'].head(10)

Out[6]: 0     Color
        1     Color
        2    Color
        3     Color
        4     Color
        5     Color
        6     Color
        7     Color
        8     Color
        9     Color
        Name: color, dtype: object
```

2. Director_name column

Replaced Null value and empty cells in the second column with ‘Ridley Scott’ as he has directed the most movies. This is a general approach. The more apt approach would be to contact the person or the organization for the correct missing data.

```
df3['director_name'].value_counts().idxmax()

df3['director_name'].replace(np.nan, "Ridley Scott", inplace=True)

df3['director_name'].replace('Null', "Ridley Scott", inplace=True)

df3['director_name'].value_counts()

df3['director_name'].str.strip().value_counts()

df3['director_name'].head(14)
```



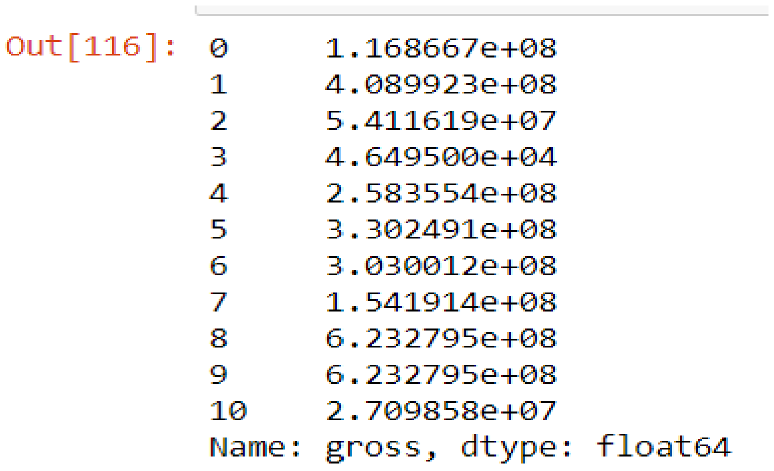
```
Out[9]: 0         Martin Scorsese
        1             Shane Black
        2       Quentin Tarantino
        3        Kenneth Lonergan
        4           Peter Jackson
        5            Ridley Scott
        6           Peter Jackson
        7             Edward Hall
        8             Joss Whedon
        9             Joss Whedon
        10             Tom Tykwer
        11           Ridley Scott
        12    Christopher Spencer
        13      Christopher Nolan
        Name: director_name, dtype: object
```

3. Gross

```
avg_gross= df3['gross'].mean(axis=0)
```

avg_gross

df3['gross'].replace(np.nan,avg_gross,inplace=True)

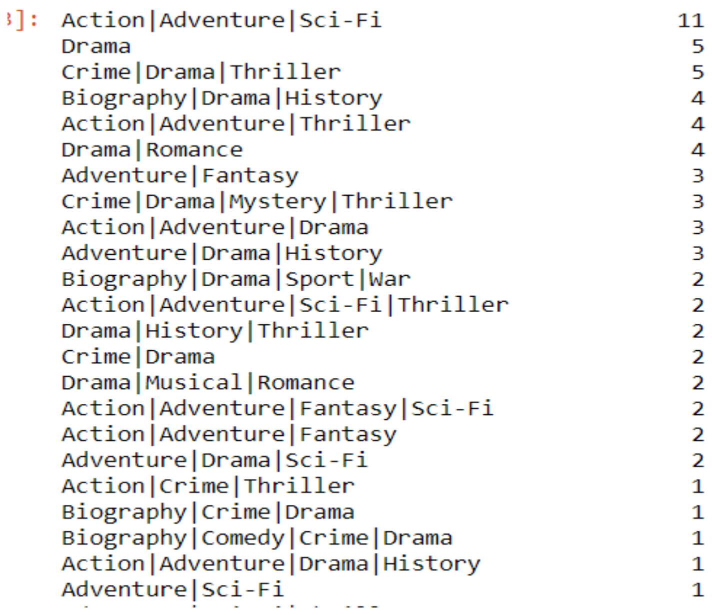


```
Out[116]: 0     1.168667e+08
          1     4.089923e+08
          2     5.411619e+07
          3     4.649500e+04
          4     2.583554e+08
          5     3.302491e+08
          6     3.030012e+08
          7     1.541914e+08
          8     6.232795e+08
          9     6.232795e+08
          10    2.709858e+07
          Name: gross, dtype: float64
```

4. Genres Column

The most repeated genre of movies is Action| Adventure | Sci-Fi which can be seen through value_counts(). So I have filled the missing values with this genre.

df3['genres'].value_counts()
df3['genres'].replace(np.nan, "Action|Adventure|Sci-Fi", inplace=True)



```
3]: Action|Adventure|Sci-Fi                 11
    Drama                                     5
    Crime|Drama|Thriller                      5
    Biography|Drama|History                   4
    Action|Adventure|Thriller                 4
    Drama|Romance                             4
    Adventure|Fantasy                         3
    Crime|Drama|Mystery|Thriller              3
    Action|Adventure|Drama                    3
    Adventure|Drama|History                   3
    Biography|Drama|Sport|War                 2
    Action|Adventure|Sci-Fi|Thriller          2
    Drama|History|Thriller                    2
    Crime|Drama                               2
    Drama|Musical|Romance                     2
    Action|Adventure|Fantasy|Sci-Fi           2
    Action|Adventure|Fantasy                  2
    Adventure|Drama|Sci-Fi                    2
    Action|Crime|Thriller                     1
    Biography|Crime|Drama                     1
    Biography|Comedy|Crime|Drama              1
    Action|Adventure|Drama|History            1
    Adventure|Sci-Fi                          1
```

5. Budget Column

Replace the Missing values in this Column with mean of all the budgets.

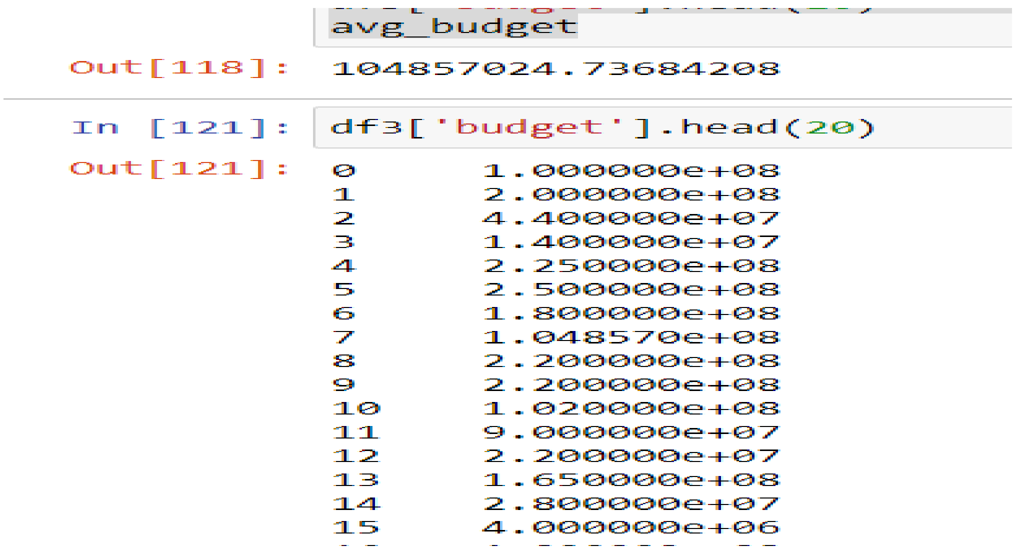
avg_budget= df3['budget'].mean(axis=0)

avg_budget

df3['budget'].replace(np.nan,avg_budget,inplace=True)

df3['budget'].head(10)

avg_budget



```
            avg_budget
Out[118]:   104857024.73684208

In  [121]:  df3['budget'].head(20)
Out[121]:   0        1.000000e+08
            1        2.000000e+08
            2        4.400000e+07
            3        1.400000e+07
            4        2.250000e+08
            5        2.500000e+08
            6        1.800000e+08
            7        1.048570e+08
            8        2.200000e+08
            9        2.200000e+08
            10       1.020000e+08
            11       9.000000e+07
            12       2.200000e+07
            13       1.650000e+08
            14       2.800000e+07
            15       4.000000e+06
```

6. Now we can see that there are no missing/null values in the all columns

missing=df3.isnull()

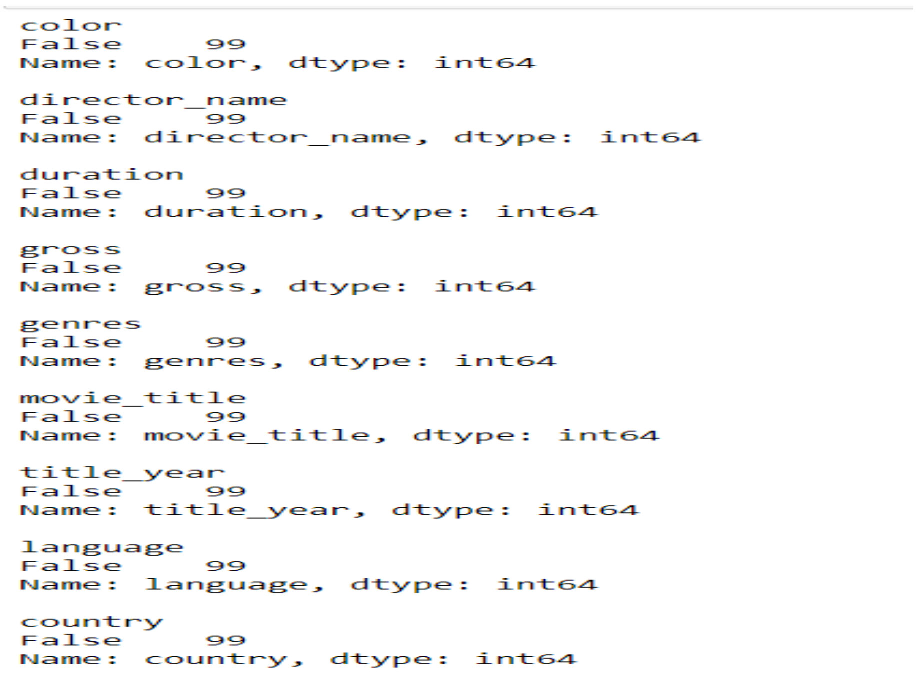
missing.tail()

for column in missing.columns.values.tolist():

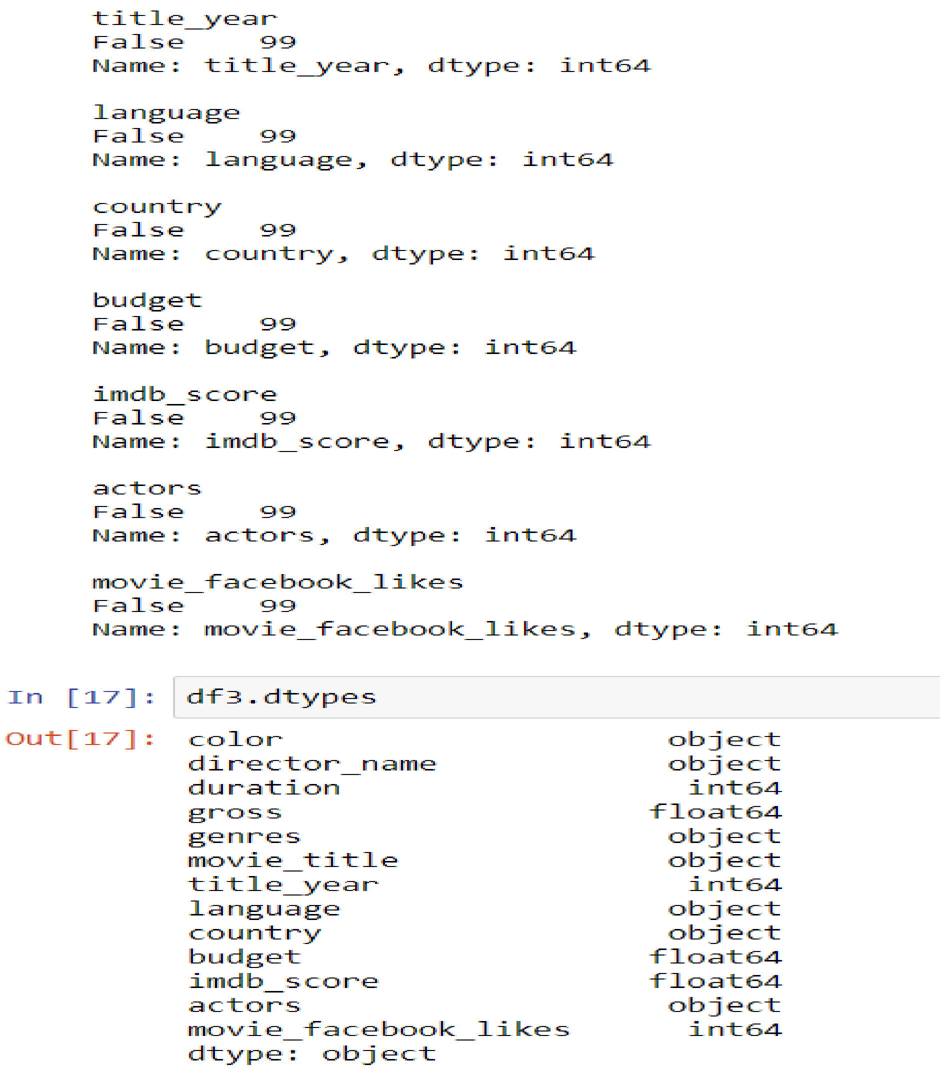
    print(column)

    print(missing[column].value_counts())

    print("")



```
color
False    99
Name: color, dtype: int64

director_name
False    99
Name: director_name, dtype: int64

duration
False    99
Name: duration, dtype: int64

gross
False    99
Name: gross, dtype: int64

genres
False    99
Name: genres, dtype: int64

movie_title
False    99
Name: movie_title, dtype: int64

title_year
False    99
Name: title_year, dtype: int64

language
False    99
Name: language, dtype: int64

country
False    99
Name: country, dtype: int64
```

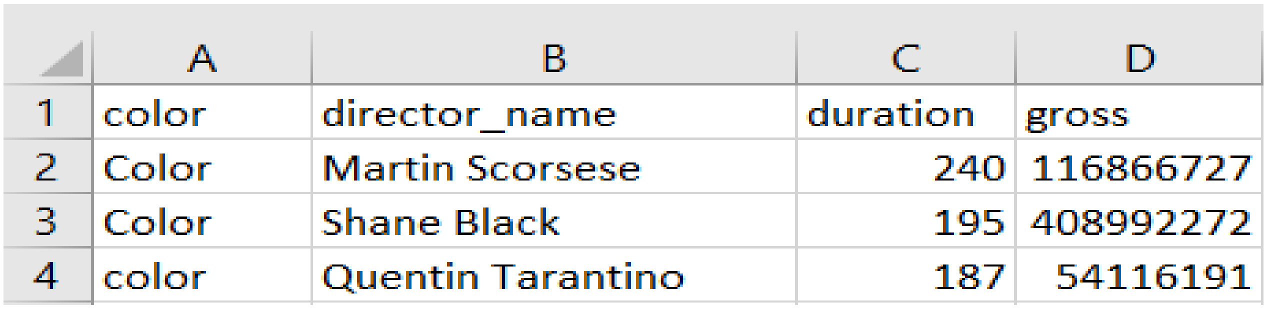


```
title_year
False    99
Name: title_year, dtype: int64

language
False    99
Name: language, dtype: int64

country
False    99
Name: country, dtype: int64

budget
False    99
Name: budget, dtype: int64

imdb_score
False    99
Name: imdb_score, dtype: int64

actors
False    99
Name: actors, dtype: int64

movie_facebook_likes
False    99
Name: movie_facebook_likes, dtype: int64
```

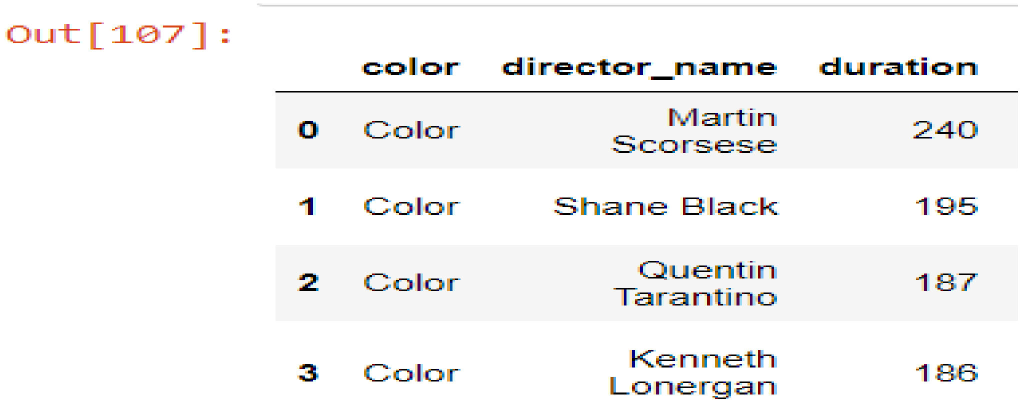
```
In [17]: df3.dtypes

Out[17]: color                      object
         director_name              object
         duration                    int64
         gross                     float64
         genres                     object
         movie_title                object
         title_year                  int64
         language                   object
         country                    object
         budget                    float64
         imdb_score                float64
         actors                     object
         movie_facebook_likes        int64
         dtype: object
```

7. In the color column one value is color instead of Color (the one with duration 187 minutes)(case-sensitive)

```
df3["color"] = df3["color"].apply(lambda x: x.replace("color", "Color"))
```



	A	B	C	D
1	color	director_name	duration	gross
2	Color	Martin Scorsese	240	116866727
3	Color	Shane Black	195	408992272
4	color	Quentin Tarantino	187	54116191

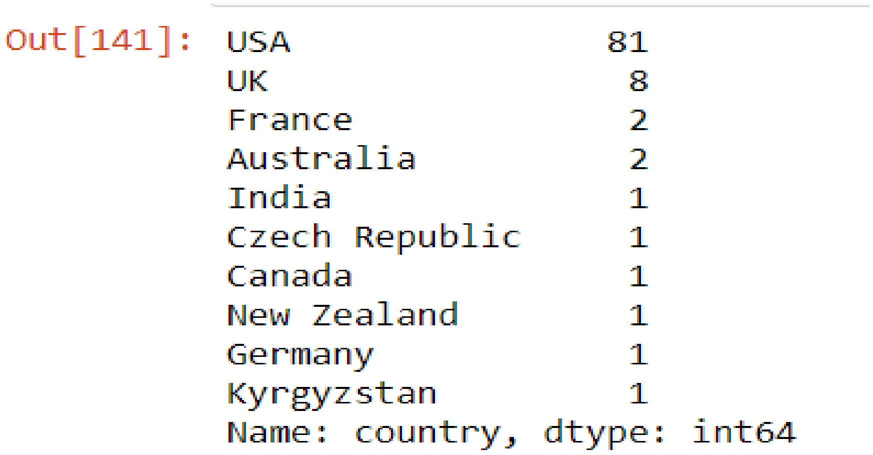


Out[107]:

	color	director_name	duration
0	Color	Martin Scorsese	240
1	Color	Shane Black	195
2	Color	Quentin Tarantino	187
3	Color	Kenneth Lonergan	186

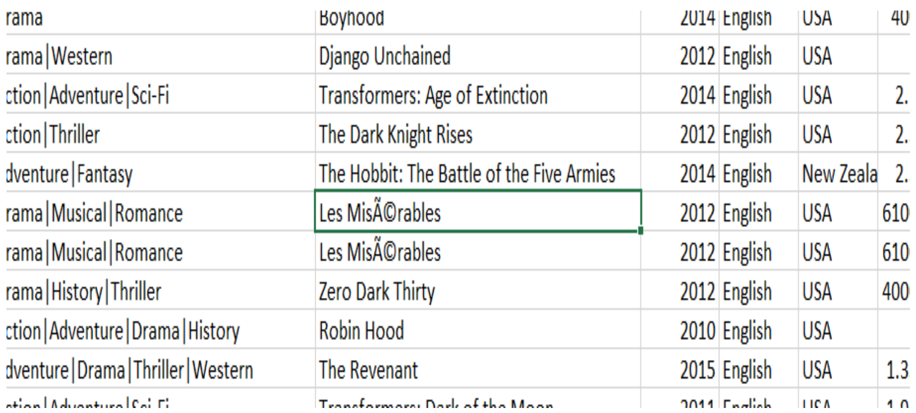
8. USA in Country column is in three variations i.e, usa, USA, United States. Changed all these variations into single thing i.e., USA

```
df3["country"] = df3["country"].apply(lambda x: x.replace("usa", "USA"))

df3["country"] = df3["country"].apply(lambda x: x.replace("United States", "USA"))

(df3["country"]).value_counts()
```

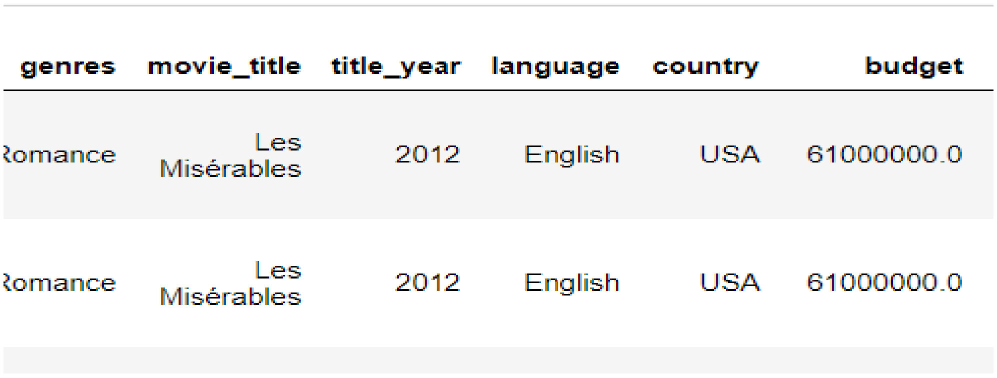


```
Out[141]: USA                 81
          UK                   8
          France               2
          Australia            2
          India                1
          Czech Republic       1
          Canada               1
          New Zealand          1
          Germany              1
          Kyrgyzstan           1
          Name: country, dtype: int64
```

9. Some latin characters in the words are automatically converted into understandable english language when loaded into data frame

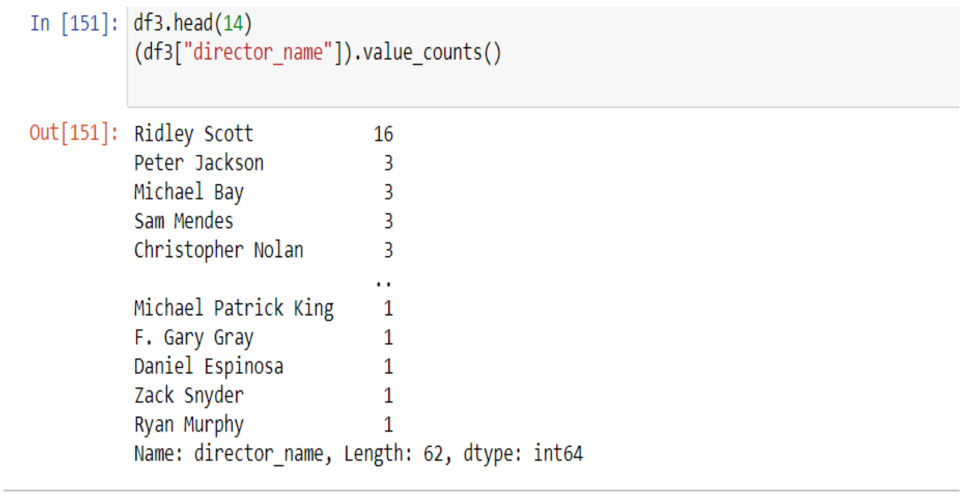


rama	Boyhood	2014	English	USA	40
rama|Western	Django Unchained	2012	English	USA	
ction|Adventure|Sci-Fi	Transformers: Age of Extinction	2014	English	USA	2.
ction|Thriller	The Dark Knight Rises	2012	English	USA	2.
dventure|Fantasy	The Hobbit: The Battle of the Five Armies	2014	English	New Zeala	2.
rama|Musical|Romance	Les MisÃ©rables	2012	English	USA	610
rama|Musical|Romance	Les MisÃ©rables	2012	English	USA	610
rama|History|Thriller	Zero Dark Thirty	2012	English	USA	400
ction|Adventure|Drama|History	Robin Hood	2010	English	USA	
dventure|Drama|Thriller|Western	The Revenant	2015	English	USA	1.3
ction|Adventure|Sci-Fi	Transformers: Dark of the Moon	2011	English	USA	1.9



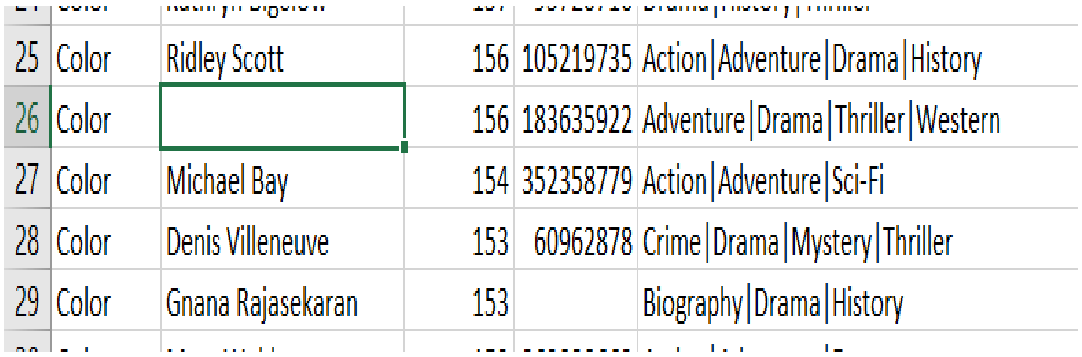
genres	movie_title	title_year	language	country	budget
Romance	Les Misérables	2012	English	USA	61000000.0
Romance	Les Misérables	2012	English	USA	61000000.0

10. There are a few director names missing and the correct method to find out this data is contacting the organization with this data. As of now, these missing values of directors are filled with the director who directed most movies and he is Ridley Scott

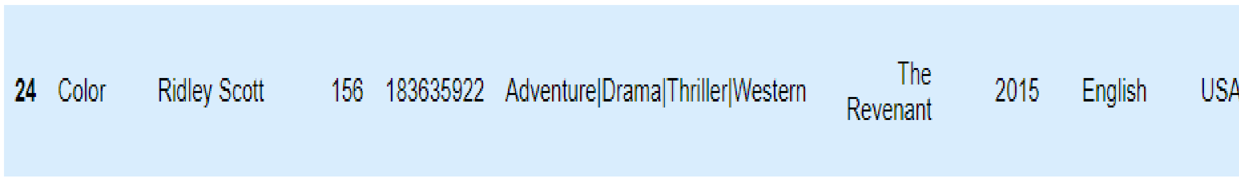


```
In [151]: df3.head(14)
          (df3["director_name"]).value_counts()

Out[151]: Ridley Scott           16
          Peter Jackson           3
          Michael Bay             3
          Sam Mendes              3
          Christopher Nolan       3
                                 ..
          Michael Patrick King    1
          F. Gary Gray            1
          Daniel Espinosa         1
          Zack Snyder             1
          Ryan Murphy             1
          Name: director_name, Length: 62, dtype: int64
```

```
df3['director_name'].replace(np.nan, "Ridley Scott", inplace=True)
```



25	Color	Ridley Scott	156	105219735	Action|Adventure|Drama|History
26	Color		156	183635922	Adventure|Drama|Thriller|Western
27	Color	Michael Bay	154	352358779	Action|Adventure|Sci-Fi
28	Color	Denis Villeneuve	153	60962878	Crime|Drama|Mystery|Thriller
29	Color	Gnana Rajasekaran	153		Biography|Drama|History



24	Color	Ridley Scott	156	183635922	Adventure|Drama|Thriller|Western	The Revenant	2015	English	USA

11. Movie duration can never be negative, so I have replaced the negative values with the positive mean of the duration. We can also choose to just simply change the negative value to positive value with abs() function in python

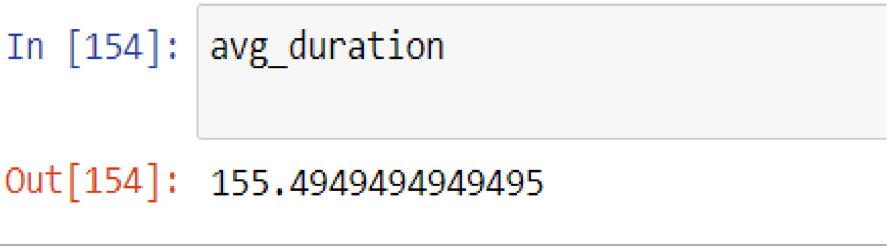
avg_duration=df3['duration'].mean(axis=0)

num = df3['duration']._get_numeric_data()

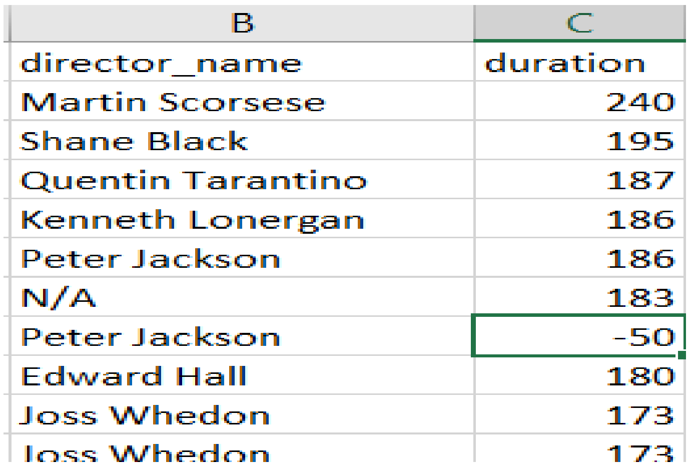
import math

truncatedvalue=math.trunc(avg_duration)

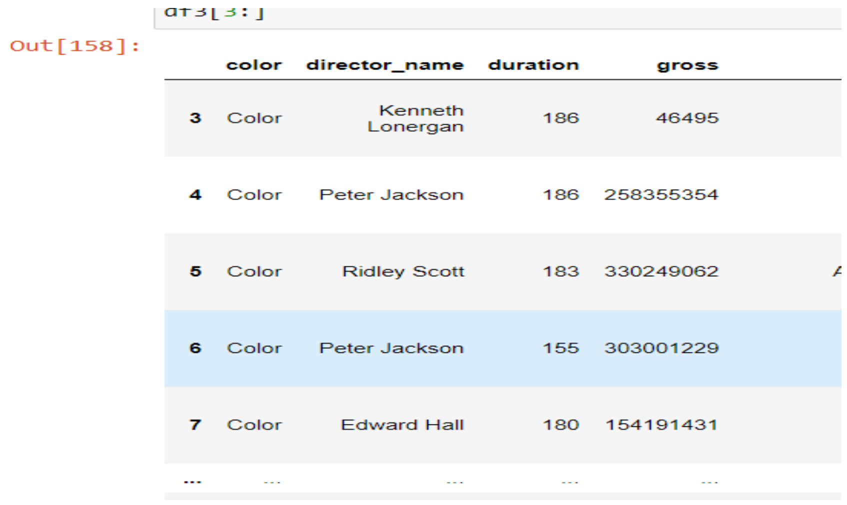
num[num < 0] = truncatedvalue



```
In [154]: avg_duration

Out[154]: 155.4949494949495
```



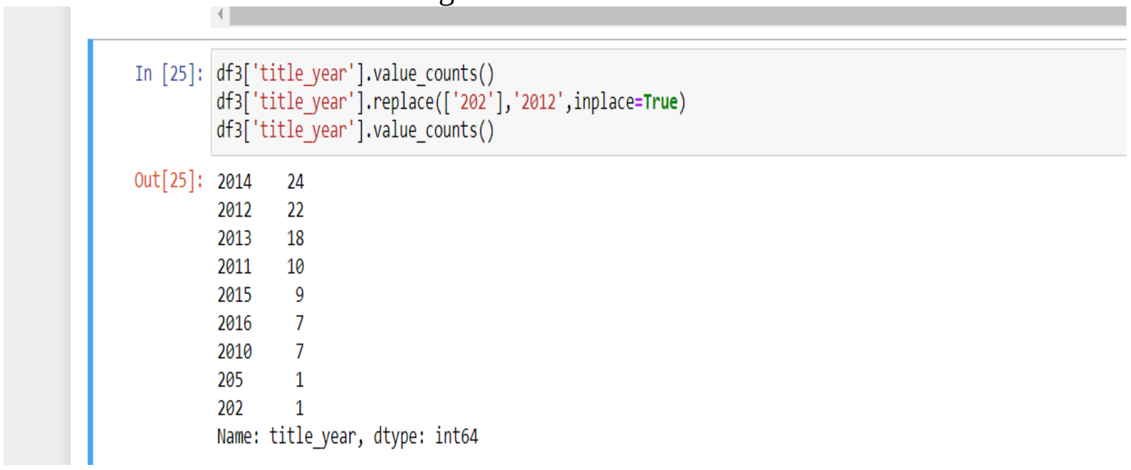
B	C
director_name	duration
Martin Scorsese	240
Shane Black	195
Quentin Tarantino	187
Kenneth Lonergan	186
Peter Jackson	186
N/A	183
Peter Jackson	-50
Edward Hall	180
Joss Whedon	173
Joss Whedon	173



```
Out[158]:
```

	color	director_name	duration	gross
3	Color	Kenneth Lonergan	186	46495
4	Color	Peter Jackson	186	258355354
5	Color	Ridley Scott	183	330249062
6	Color	Peter Jackson	155	303001229
7	Color	Edward Hall	180	154191431
...	...	...	...	...

12. Some years are incorrectly written which can be seen below. i.e., 202 and 205. These values needs to be changed.



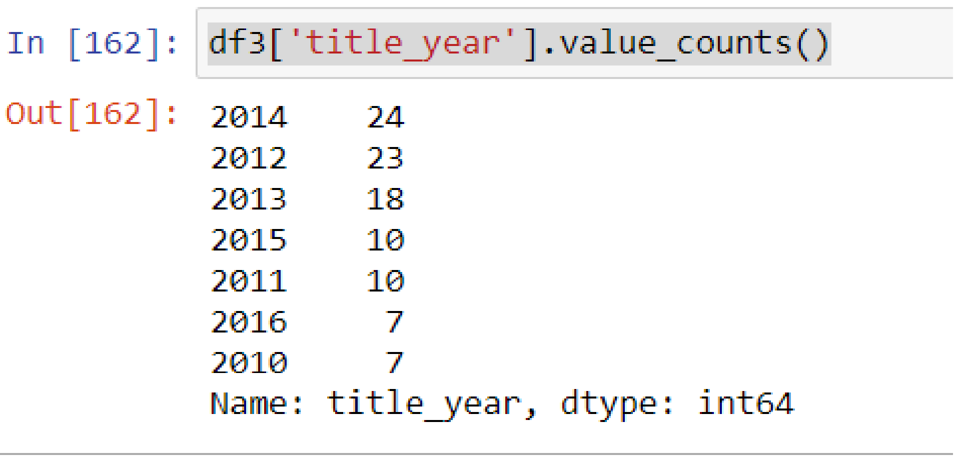
```
In [25]: df3['title_year'].value_counts()
         df3['title_year'].replace(['202'],'2012',inplace=True)
         df3['title_year'].value_counts()

Out[25]: 2014    24
         2012    22
         2013    18
         2011    10
         2015     9
         2016     7
         2010     7
         205      1
         202      1
         Name: title_year, dtype: int64
```

num = df3['title_year']._get_numeric_data()

num[num == 205] = 2015

num[num == 202] = 2012

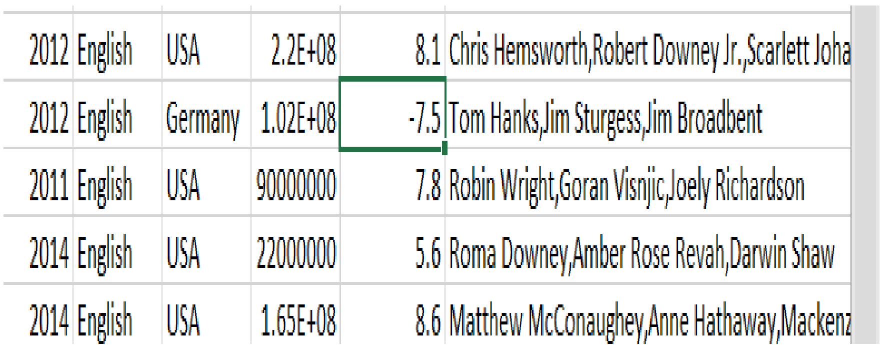
df3['title_year'].value_counts()



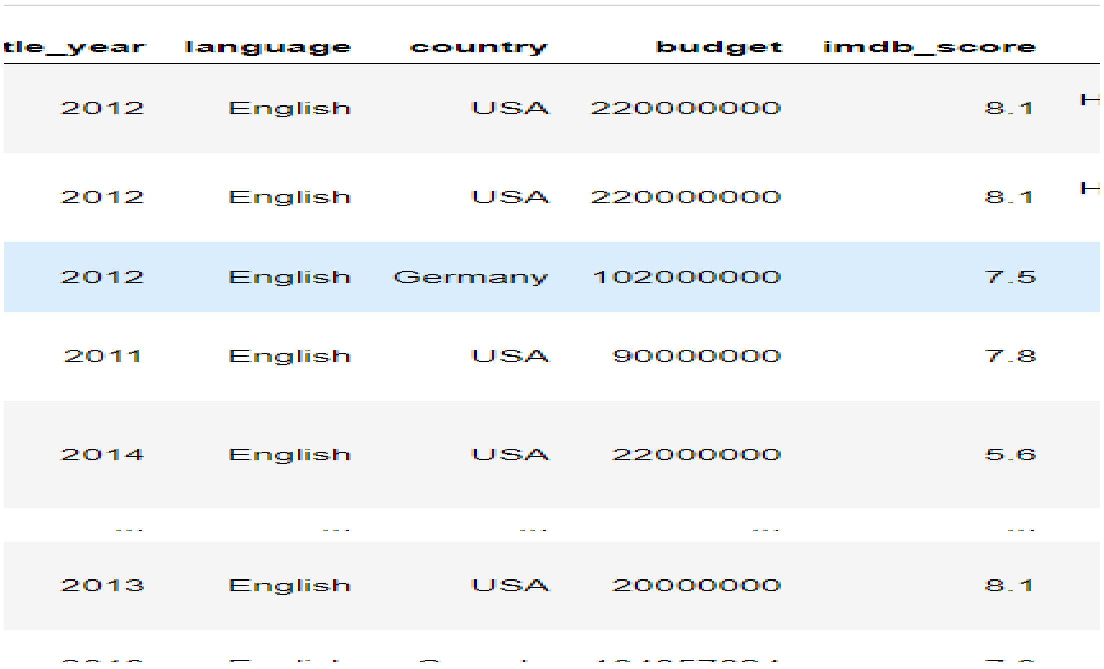
```
In [162]: df3['title_year'].value_counts()

Out[162]: 2014    24
          2012    23
          2013    18
          2015    10
          2011    10
          2016     7
          2010     7
          Name: title_year, dtype: int64
```

13. In general IMDB_scores will in in range 0 to 10 and they generally will not be negative. So, the negative values in the data set needs to be removed.

```
num = df3['imdb_score']._get_numeric_data()

num[num < 0] = abs(num)

(df3['imdb_score']>0).value_counts()

df3['imdb_score'].value_counts()

(df3['imdb_score']>0).value_counts()

df3[8:]
```



2012	English	USA	2.2E+08	8.1	Chris Hemsworth,Robert Downey Jr.,Scarlett Joha
2012	English	Germany	1.02E+08	-7.5	Tom Hanks,Jim Sturgess,Jim Broadbent
2011	English	USA	90000000	7.8	Robin Wright,Goran Visnjic,Joely Richardson
2014	English	USA	22000000	5.6	Roma Downey,Amber Rose Revah,Darwin Shaw
2014	English	USA	1.65E+08	8.6	Matthew McConaughey,Anne Hathaway,Mackenz



tle_year	language	country	budget	imdb_score
2012	English	USA	220000000	8.1
2012	English	USA	220000000	8.1
2012	English	Germany	102000000	7.5
2011	English	USA	90000000	7.8
2014	English	USA	22000000	5.6
...	...	...	...	...
2013	English	USA	20000000	8.1

14. Splitted the Actors and Genres into different columns for easy view

df_splittedGenre= df3['genres'].str.split('|').apply(pd.Series)

df_splittedGenre.head()

df_splittedGenre.columns = ['Genre1','Genre2','Genre3','Genre4','Genre5']

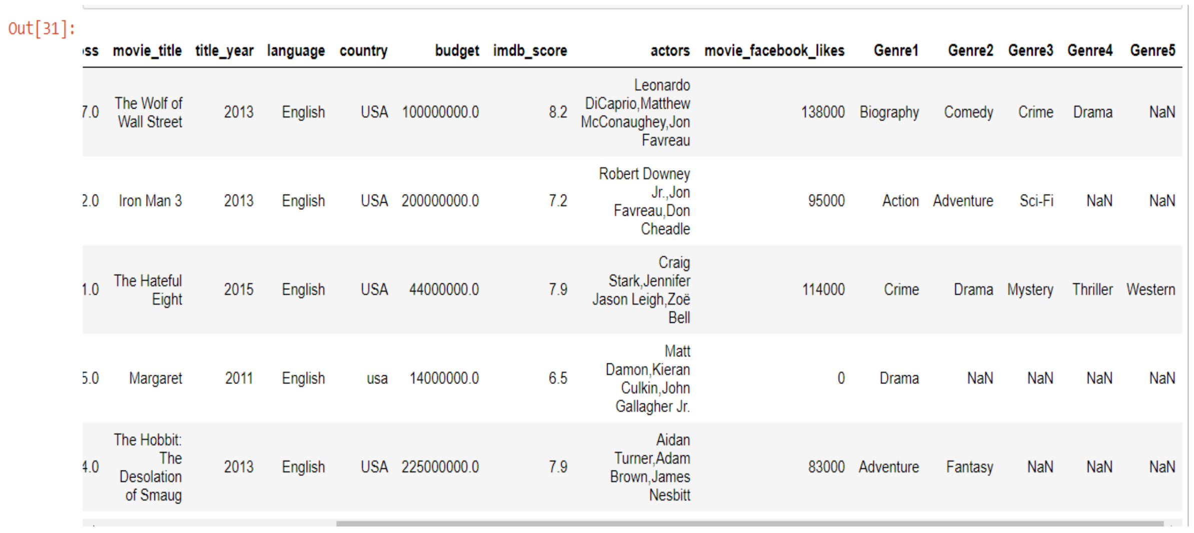
df3.head()

df_splittedGenre

df3 = pd.concat([df3, df_splittedGenre],axis=1)

df3.drop("genres", axis = 1, inplace=True)

df3.head()


Out[31]:

ss	movie_title	title_year	language	country	budget	imdb_score	actors	movie_facebook_likes	Genre1	Genre2	Genre3	Genre4	Genre5
7.0	The Wolf of Wall Street	2013	English	USA	100000000.0	8.2	Leonardo DiCaprio,Matthew McConaughey,Jon Favreau	138000	Biography	Comedy	Crime	Drama	NaN
2.0	Iron Man 3	2013	English	USA	200000000.0	7.2	Robert Downey Jr.,Jon Favreau,Don Cheadle	95000	Action	Adventure	Sci-Fi	NaN	NaN
1.0	The Hateful Eight	2015	English	USA	44000000.0	7.9	Craig Stark,Jennifer Jason Leigh,Zoë Bell	114000	Crime	Drama	Mystery	Thriller	Western
5.0	Margaret	2011	English	usa	14000000.0	6.5	Matt Damon,Kieran Culkin,John Gallagher Jr.	0	Drama	NaN	NaN	NaN	NaN
4.0	The Hobbit: The Desolation of Smaug	2013	English	USA	225000000.0	7.9	Aidan Turner,Adam Brown,James Nesbitt	83000	Adventure	Fantasy	NaN	NaN	NaN

For further analysis on Genres, we can replace the NaN values with mode of the Genre.

df3_splitactors=df3['actors'].str.split(',').apply(pd.Series)

df3_splitactors.head()

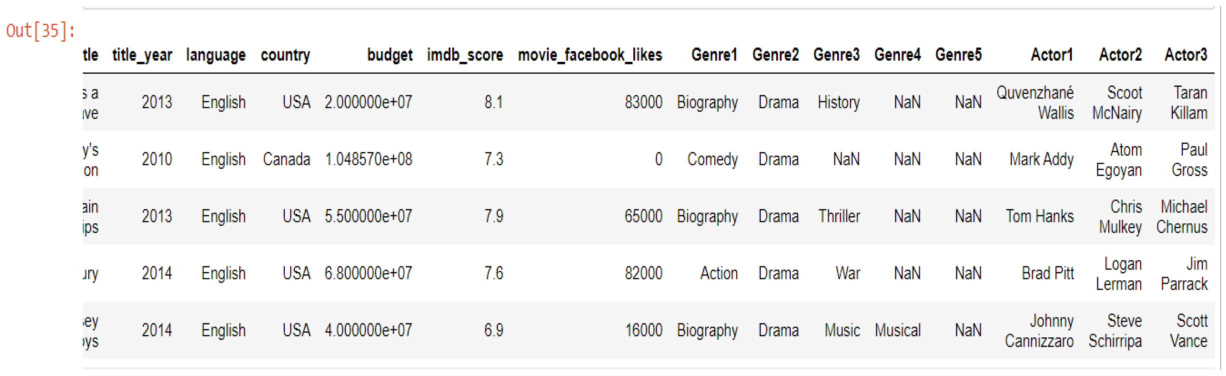
df3_splitactors.columns=['Actor1','Actor2','Actor3']

df3_splitactors.head()

df3=pd.concat([df3,df3_splitactors],axis=1)

df3.head()

df3.drop('actors',axis=1,inplace=True)



Out[35]:

tle	title_year	language	country	budget	imdb_score	movie_facebook_likes	Genre1	Genre2	Genre3	Genre4	Genre5	Actor1	Actor2	Actor3
s a ive	2013	English	USA	2.000000e+07	8.1	83000	Biography	Drama	History	NaN	NaN	Quvenzhané Wallis	Scoot McNairy	Taran Killam
y's on	2010	English	Canada	1.048570e+08	7.3	0	Comedy	Drama	NaN	NaN	NaN	Mark Addy	Atom Egoyan	Paul Gross
ain ips	2013	English	USA	5.500000e+07	7.9	65000	Biography	Drama	Thriller	NaN	NaN	Tom Hanks	Chris Mulkey	Michael Chernus
ıry	2014	English	USA	6.800000e+07	7.6	82000	Action	Drama	War	NaN	NaN	Brad Pitt	Logan Lerman	Jim Parrack
ey ıys	2014	English	USA	4.000000e+07	6.9	16000	Biography	Drama	Music	Musical	NaN	Johnny Cannizzaro	Steve Schirripa	Scott Vance

- For all of the above, explain your methods (use Python, R, SQL, Excel, direct edit, etc).

I have used Python programming Language through Jupyter Notebooks. For creating the data frames, packages namely pandas and numpy were handy.

All the values before replacing, needs to be verified from the specific organizations. Only then we can get the values and go for further data Visualization or modelling. So, in order to clean the data given, I have replaced missing values with mean or max based on particular conditions and they may not be the final values.