In [1]:

```
# Import the numpy and pandas packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Task 1: Reading and Inspection

- ### Subtask 1.1: Import and read

Import and read the movie database. Store it in a variable called `movies`.

In [2]:

```
#write your code here
movies = pd.read_csv('IMDB_Movies.csv')
OrgData = movies
movies
```

Out[2]:

	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name
0	Color	James Cameron	723.0	178.0	0.0	855.0	Joel David Moore
1	Color	Gore Verbinski	302.0	169.0	563.0	1000.0	Orlando Bloom
2	Color	Sam Mendes	602.0	148.0	0.0	161.0	Rory Kinnear
3	Color	Christopher Nolan	813.0	164.0	22000.0	23000.0	Christian Bale
4	NaN	Doug Walker	NaN	NaN	131.0	NaN	Rob Walker
...	...	...	...	...	...	...	...
5038	Color	Scott Smith	1.0	87.0	2.0	318.0	Daphne Zuniga
5039	Color	NaN	43.0	43.0	NaN	319.0	Valorie Curry
5040	Color	Benjamin Roberds	13.0	76.0	0.0	0.0	Maxwell Moody
5041	Color	Daniel Hsia	14.0	100.0	0.0	489.0	Daniel Henney
5042	Color	Jon Gunn	43.0	90.0	16.0	16.0	Brian Herzlinger

5043 rows × 28 columns

- ### Subtask 1.2: Inspect the dataframe

Inspect the dataframe's columns, shapes, variable types etc.

In [65]:

```
#write your code here
movies.shape
```

Out[65]:

```
(5043, 28)
```

In [66]:

```
movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5043 entries, 0 to 5042
Data columns (total 28 columns):
color                        5024 non-null object
director_name                4939 non-null object
num_critic_for_reviews       4993 non-null float64
duration                     5028 non-null float64
director_facebook_likes      4939 non-null float64
actor_3_facebook_likes       5020 non-null float64
actor_2_name                 5030 non-null object
actor_1_facebook_likes       5036 non-null float64
gross                        4159 non-null float64
genres                       5043 non-null object
actor_1_name                 5036 non-null object
movie_title                  5043 non-null object
num_voted_users              5043 non-null int64
cast_total_facebook_likes    5043 non-null int64
actor_3_name                 5020 non-null object
facenumber_in_poster         5030 non-null float64
plot_keywords                4890 non-null object
movie_imdb_link              5043 non-null object
num_user_for_reviews         5023 non-null object
language                     5031 non-null object
country                      5038 non-null object
content_rating               4740 non-null object
budget                       4551 non-null float64
title_year                   4935 non-null float64
actor_2_facebook_likes       5030 non-null float64
imdb_score                   5043 non-null float64
aspect_ratio                 4714 non-null float64
movie_facebook_likes         5043 non-null int64
dtypes: float64(12), int64(3), object(13)
memory usage: 1.1+ MB
```

Task 2: Cleaning the Data

- **### Subtask 2.1: Inspect Null values**

Find out the number of Null values in all the columns and rows. Also, find the percentage of Null values in each column. Round off the percentages upto two decimal places.

In [67]:

```
# Write your code for column-wise null count here
movies.isnull().sum(axis=0).sort_values(ascending=False)
```

Out[67]:

```
gross                        884
budget                       492
aspect_ratio                 329
content_rating               303
plot_keywords                153
title_year                   108
director_name                104
director_facebook_likes      104
num_critic_for_reviews        50
actor_3_name                  23
actor_3_facebook_likes        23
num_user_for_reviews          20
color                         19
duration                      15
facenumber_in_poster          13
actor_2_name                  13
actor_2_facebook_likes        13
language                      12
actor_1_name                   7
actor_1_facebook_likes         7
country                        5
```

```
movie_facebook_likes         0
genres                       0
movie_title                  0
num_voted_users              0
movie_imdb_link              0
imdb_score                   0
cast_total_facebook_likes    0
dtype: int64
```

In [68]:

```
# Write your code for row-wise null count here
movies.isnull().sum(axis=1).sort_values(ascending=False)
```

Out[68]:

```
279     15
4       13
4945    11
2241    11
2342    10
        ..
2708     0
2707     0
2706     0
2705     0
0        0
Length: 5043, dtype: int64
```

In [69]:

```
# Write your code for column-wise null percentages here
movies.isnull().sum(axis=0).sort_values(ascending=False)/len(movies) * 100
```

Out[69]:

```
gross                        17.529248
budget                        9.756098
aspect_ratio                  6.523895
content_rating                6.008328
plot_keywords                 3.033908
title_year                    2.141582
director_name                 2.062265
director_facebook_likes       2.062265
num_critic_for_reviews        0.991473
actor_3_name                  0.456078
actor_3_facebook_likes        0.456078
num_user_for_reviews          0.396589
color                         0.376760
duration                      0.297442
facenumber_in_poster          0.257783
actor_2_name                  0.257783
actor_2_facebook_likes        0.257783
language                      0.237954
actor_1_name                  0.138806
actor_1_facebook_likes        0.138806
country                       0.099147
movie_facebook_likes          0.000000
genres                        0.000000
movie_title                   0.000000
num_voted_users               0.000000
movie_imdb_link               0.000000
imdb_score                    0.000000
cast_total_facebook_likes     0.000000
dtype: float64
```

- **### Subtask 2.2: Drop unecessary columns**

For this assignment, you will mostly be analyzing the movies with respect to the ratings, gross collection, popularity of movies, etc. So many of the columns in this dataframe are not required. So it is advised to drop the following columns.

- color
- director_facebook_likes
- actor_1_facebook_likes
- actor_2_facebook_likes
- actor_3_facebook_likes
- actor_2_name
- cast_total_facebook_likes
- actor_3_name
- duration
- facenumber_in_poster
- content_rating
- country
- movie_imdb_link
- aspect_ratio
- plot_keywords

In [70]:

```
# Write your code for dropping the columns here. It is advised to keep inspecting the dat
aframe after each set of opera
movies = movies.drop([

    'color',
    'director_facebook_likes',
    'actor_1_facebook_likes',
    'actor_2_facebook_likes',
    'actor_3_facebook_likes',
    'actor_2_name',
    'cast_total_facebook_likes',
    'actor_3_name',
    'duration',
    'facenumber_in_poster',
    'content_rating',
    'country',
    'movie_imdb_link',
    'aspect_ratio',
    'plot_keywords'],axis=1)
```

In [71]:

```
movies
```

Out[71]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	num_votec
0	James Cameron	723.0	760505847.0	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	Avatar	
1	Gore Verbinski	302.0	309404152.0	Action|Adventure|Fantasy	Johnny Depp	Pirates of the Caribbean: At World's End	
2	Sam Mendes	602.0	200074175.0	Action|Adventure|Thriller	Christoph Waltz	Spectre	
3	Christopher Nolan	813.0	448130642.0	Action|Thriller	Tom Hardy	The Dark Knight Rises	1
4	Doug Walker	NaN	NaN	Documentary	Doug Walker	Star Wars: Episode VII - The Force Awakens ...	
...	...	...	...	...	...	...	

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	num_votec
5038	Scott Smith	1.0	NaN	Comedy|Drama	Eric Mabius	Signed Sealed Delivered	
5039	NaN	43.0	NaN	Crime|Drama|Mystery|Thriller	Natalie Zea	The Following	
5040	Benjamin Roberds	13.0	NaN	Drama|Horror|Thriller	Eva Boehnke	A Plague So Pleasant	
5041	Daniel Hsia	14.0	10443.0	Comedy|Drama|Romance	Alan Ruck	Shanghai Calling	
5042	Jon Gunn	43.0	85222.0	Documentary	John August	My Date with Drew	

5043 rows × 13 columns

- **### Subtask 2.3: Drop unecessary rows using columns with high Null percentages**

Now, on inspection you might notice that some columns have large percentage (greater than 5%) of Null values. Drop all the rows which have Null values for such columns.

In [72]:

```
# Write your code for dropping the rows here
round(movies.isnull().sum().sort_values(ascending=False)/len(movies)*100,2)
```

Out[72]:

```
gross                     17.53
budget                     9.76
title_year                 2.14
director_name              2.06
num_critic_for_reviews     0.99
num_user_for_reviews       0.40
language                   0.24
actor_1_name               0.14
movie_facebook_likes       0.00
imdb_score                 0.00
num_voted_users            0.00
movie_title                0.00
genres                     0.00
dtype: float64
```

In [73]:

```
movies = movies[movies['gross'].notnull()]
movies = movies[movies['budget'].notnull()]
```

In [74]:

```
round(movies.isnull().sum().sort_values(ascending=False)/len(movies)*100,2)
```

Out[74]:

```
language                  0.08
actor_1_name              0.08
num_critic_for_reviews    0.03
movie_facebook_likes      0.00
imdb_score                0.00
title_year                0.00
budget                    0.00
num_user_for_reviews      0.00
num_voted_users           0.00
movie_title               0.00
genres                    0.00
gross                     0.00
director_name             0.00
dtype: float64
```

- **### Subtask 2.4: Drop unecessary rows**

Some of the rows might have greater than five NaN values. Such rows aren't of much use for the analysis and hence, should be removed.

In [75]:

```
# Write your code for dropping the rows here
(movies.isnull().sum(axis=1).sort_values(ascending=False) > 5).sum()
```

Out[75]:

```
0
```

In [76]:

```
movies = movies[movies.isnull().sum(axis=1).sort_values(ascending=False) <= 5]
movies
```

```
c:\users\karan\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launche
r.py:1: UserWarning: Boolean Series key will be reindexed to match DataFrame index.
  """Entry point for launching an IPython kernel.
```

Out[76]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	nu
0	James Cameron	723.0	760505847.0	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	Avatar	
1	Gore Verbinski	302.0	309404152.0	Action|Adventure|Fantasy	Johnny Depp	Pirates of the Caribbean: At World's End	
2	Sam Mendes	602.0	200074175.0	Action|Adventure|Thriller	Christoph Waltz	Spectre	
3	Christopher Nolan	813.0	448130642.0	Action|Thriller	Tom Hardy	The Dark Knight Rises	
5	Andrew Stanton	462.0	73058679.0	Action|Adventure|Sci-Fi	Daryl Sabara	John Carter	
...	...	...	...	...	...	...	
5033	Shane Carruth	143.0	424760.0	Drama|Sci-Fi|Thriller	Shane Carruth	Primer	
5034	Neill Dela Llana	35.0	70071.0	Thriller	Ian Gamazon	Cavite	
5035	Robert Rodriguez	56.0	2040920.0	Action|Crime|Drama|Romance|Thriller	Carlos Gallardo	El Mariachi	
5037	Edward Burns	14.0	4584.0	Comedy|Drama	Kerry Bishé	Newlyweds	
5042	Jon Gunn	43.0	85222.0	Documentary	John August	My Date with Drew	

3891 rows × 13 columns

- **### Subtask 2.5: Fill NaN values**

You might notice that the `language` column has some NaN values. Here, on inspection, you will see that it is safe to replace all the missing values with `'English'`.

In [77]:

```
# Write your code here
round(movies.isnull().sum().sort_values(ascending=False)/len(movies)*100,2)
```

Out[77]:

```
language                  0.08
actor_1_name              0.08
num_critic_for_reviews    0.03
movie_facebook_likes      0.00
imdb_score                0.00
title_year                0.00
budget                    0.00
num_user_for_reviews      0.00
num_voted_users           0.00
movie_title               0.00
genres                    0.00
gross                     0.00
director_name             0.00
dtype: float64
```

In [78]:

```
movies.groupby('language').language.count().sort_values(ascending=False)
```

Out[78]:

```
language
English       3707
French          37
Spanish         26
Mandarin        15
German          13
Japanese        12
Hindi           10
Cantonese        8
Italian          7
Korean           5
Portuguese       5
Norwegian        4
Hebrew           3
Persian          3
Dutch            3
Danish           3
Thai             3
Dari             2
Indonesian       2
Aboriginal       2
Icelandic        1
Hungarian        1
Arabic           1
Aramaic          1
Bosnian          1
Telugu           1
Czech            1
Swedish          1
Russian          1
Romanian         1
Dzongkha         1
None             1
Filipino         1
Mongolian        1
Maya             1
Kazakh           1
Vietnamese       1
Zulu             1
Name: language, dtype: int64
```

In [79]:

```
movies.language.describe()
```

Out[79]:

```
count        3888
unique         38
top       English
```

```
freq          3707
Name: language, dtype: object
```

In [80]:

```
movies.language = movies.language.fillna('English')
```

In [81]:

```
round(movies.isnull().sum().sort_values(ascending=False)/len(movies)*100,2)
```

Out[81]:

```
actor_1_name              0.08
num_critic_for_reviews    0.03
movie_facebook_likes      0.00
imdb_score                0.00
title_year                0.00
budget                    0.00
language                  0.00
num_user_for_reviews      0.00
num_voted_users           0.00
movie_title               0.00
genres                    0.00
gross                     0.00
director_name             0.00
dtype: float64
```

- **### Subtask 2.6: Check the number of retained rows**

You might notice that two of the columns viz. `num_critic_for_reviews` and `actor_1_name` have small percentages of NaN values left. You can let these columns as it is for now. Check the number and percentage of the rows retained after completing all the tasks above.

In [84]:

```
# Write your code for checking number of retained rows
len(movies)/len(OrgData) * 100
```

Out[84]:

```
77.15645449137418
```

Checkpoint 1: You might have noticed that we still have around `77%` of the rows!

Task 3: Data Analysis

- **### Subtask 3.1: Change the unit of columns Convert the unit of the `budget` and `gross` columns from `$` to `million $`.**

In [87]:

```
# Write your code for unit conversion here
movies['budget'] = movies['budget']/1000000
movies['gross'] = movies['gross']/1000000
```

In [88]:

```
movies
```

Out[88]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	nun
0	James Cameron	723.0	760.505847	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	Avatar	

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	num
1	Gore Verbinski	302.0	309.404152	Action|Adventure|Fantasy	Johnny Depp	Pirates of the Caribbean: At World's End	
2	Sam Mendes	602.0	200.074175	Action|Adventure|Thriller	Christoph Waltz	Spectre	
3	Christopher Nolan	813.0	448.130642	Action|Thriller	Tom Hardy	The Dark Knight Rises	
5	Andrew Stanton	462.0	73.058679	Action|Adventure|Sci-Fi	Daryl Sabara	John Carter	
...	...	...	...	...	...	...	
5033	Shane Carruth	143.0	0.424760	Drama|Sci-Fi|Thriller	Shane Carruth	Primer	
5034	Neill Dela Llana	35.0	0.070071	Thriller	Ian Gamazon	Cavite	
5035	Robert Rodriguez	56.0	2.040920	Action|Crime|Drama|Romance|Thriller	Carlos Gallardo	El Mariachi	
5037	Edward Burns	14.0	0.004584	Comedy|Drama	Kerry Bishé	Newlyweds	
5042	Jon Gunn	43.0	0.085222	Documentary	John August	My Date with Drew	

3891 rows × 13 columns

- ### Subtask 3.2: Find the movies with highest profit

  1. Create a new column called `profit` which contains the difference of the two columns: `gross` and `budget`.
  2. Sort the dataframe using the `profit` column as reference.
  3. Extract the top ten profiting movies in descending order and store them in a new dataframe - `top10`

In [89]:

```
# Write your code for creating the profit column here
movies['profit'] = movies['gross'] - movies['budget']
movies
```

Out[89]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	num
0	James Cameron	723.0	760.505847	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	Avatar	
1	Gore Verbinski	302.0	309.404152	Action|Adventure|Fantasy	Johnny Depp	Pirates of the Caribbean: At World's End	
2	Sam Mendes	602.0	200.074175	Action|Adventure|Thriller	Christoph Waltz	Spectre	
3	Christopher Nolan	813.0	448.130642	Action|Thriller	Tom Hardy	The Dark Knight Rises	
5	Andrew Stanton	462.0	73.058679	Action|Adventure|Sci-Fi	Daryl Sabara	John Carter	
...	...	...	...	...	...	...	
5033	Shane Carruth	143.0	0.424760	Drama|Sci-Fi|Thriller	Shane Carruth	Primer	
5034	Neill Dela	35.0	0.070071	Thriller	Ian Gamazon	Cavite	

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	nun
5034	Ilana	35.0	0.070071	Thriller	Ian Gamazon	Cavite	
5035	Robert Rodriguez	56.0	2.040920	Action|Crime|Drama|Romance|Thriller	Carlos Gallardo	El Mariachi	
5037	Edward Burns	14.0	0.004584	Comedy|Drama	Kerry Bishé	Newlyweds	
5042	Jon Gunn	43.0	0.085222	Documentary	John August	My Date with Drew	

3891 rows × 14 columns

```
In [90]:
```

```python
# Write your code for sorting the dataframe here
movies.sort_values(by='profit',ascending=False)
```

```
Out[90]:
```

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	nu
0	James Cameron	723.0	760.505847	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	Avatar	
29	Colin Trevorrow	644.0	652.177271	Action|Adventure|Sci-Fi|Thriller	Bryce Dallas Howard	Jurassic World	
26	James Cameron	315.0	658.672302	Drama|Romance	Leonardo DiCaprio	Titanic	
3024	George Lucas	282.0	460.935665	Action|Adventure|Fantasy|Sci-Fi	Harrison Ford	Star Wars: Episode IV - A New Hope	
3080	Steven Spielberg	215.0	434.949459	Family|Sci-Fi	Henry Thomas	E.T. the Extra-Terrestrial	
...	...	...	...	...	...	...	
2334	Katsuhiro Ôtomo	105.0	0.410388	Action|Adventure|Animation|Family|Sci-Fi|Thriller	William Hootkins	Steamboy	
2323	Hayao Miyazaki	174.0	2.298191	Adventure|Animation|Fantasy	Minnie Driver	Princess Mononoke	
3005	Lajos Koltai	73.0	0.195888	Drama|Romance|War	Marcell Nagy	Fateless	
3859	Chan-wook Park	202.0	0.211667	Crime|Drama	Min-sik Choi	Lady Vengeance	
2988	Joon-ho Bong	363.0	2.201412	Comedy|Drama|Horror|Sci-Fi	Doona Bae	The Host	

3891 rows × 14 columns

```
In [91]:
```

```python
# Write your code for top10 movies
top10 = movies.sort_values(by='profit',ascending=False).head(10)
top10
```

```
Out[91]:
```

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title
0	James Cameron	723.0	760.505847	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	Avatar
29	Colin Trevorrow	644.0	652.177271	Action|Adventure|Sci-Fi|Thriller	Bryce Dallas Howard	Jurassic World
26	James Cameron	315.0	658.672302	Drama|Romance	Leonardo DiCaprio	Titanic
						Star Wars:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title
3024	George Lucas	282.0	460.935665	Action|Adventure|Fantasy|Sci-Fi	Harrison Ford	Star Wars: Episode IV - A New Hope
3080	Steven Spielberg	215.0	434.949459	Family|Sci-Fi	Henry Thomas	E.T. the Extra-Terrestrial
794	Joss Whedon	703.0	623.279547	Action|Adventure|Sci-Fi	Chris Hemsworth	The Avengers
17	Joss Whedon	703.0	623.279547	Action|Adventure|Sci-Fi	Chris Hemsworth	The Avengers
509	Roger Allers	186.0	422.783777	Adventure|Animation|Drama|Family|Musical	Matthew Broderick	The Lion King
240	George Lucas	320.0	474.544677	Action|Adventure|Fantasy|Sci-Fi	Natalie Portman	Star Wars: Episode I - The Phantom Menace
66	Christopher Nolan	645.0	533.316061	Action|Crime|Drama|Thriller	Christian Bale	The Dark Knight

- ### Subtask 3.3: Drop duplicate values

After you found out the top 10 profiting movies, you might have notice a duplicate value. So, it seems like the dataframe has duplicate values as well. Drop the duplicate values from the dataframe and repeat `Subtask 3.2`.

In [92]:

```
# Write your code for dropping duplicate values here
movies.drop_duplicates(keep='first',inplace=True)
```

In [93]:

```
# Write code for repeating subtask 2 here
movies
```

Out[93]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	num
0	James Cameron	723.0	760.505847	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	Avatar	
1	Gore Verbinski	302.0	309.404152	Action|Adventure|Fantasy	Johnny Depp	Pirates of the Caribbean: At World's End	
2	Sam Mendes	602.0	200.074175	Action|Adventure|Thriller	Christoph Waltz	Spectre	
3	Christopher Nolan	813.0	448.130642	Action|Thriller	Tom Hardy	The Dark Knight Rises	
5	Andrew Stanton	462.0	73.058679	Action|Adventure|Sci-Fi	Daryl Sabara	John Carter	
...	...	...	...	...	...	...	
5033	Shane Carruth	143.0	0.424760	Drama|Sci-Fi|Thriller	Shane Carruth	Primer	
5034	Neill Dela Llana	35.0	0.070071	Thriller	Ian Gamazon	Cavite	
5035	Robert Rodriguez	56.0	2.040920	Action|Crime|Drama|Romance|Thriller	Carlos Gallardo	El Mariachi	

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	num
5037	Edward Burns	14.0	0.004584	Comedy|Drama	Kerry Bishé	Newlyweds	
5042	Jon Gunn	43.0	0.085222	Documentary	John August	My Date with Drew	

3856 rows × 14 columns

In [94]:

```
top10 = movies.sort_values(by='profit',ascending=False).head(10)
top10
```

Out[94]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title
0	James Cameron	723.0	760.505847	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	Avatar
29	Colin Trevorrow	644.0	652.177271	Action|Adventure|Sci-Fi|Thriller	Bryce Dallas Howard	Jurassic World
26	James Cameron	315.0	658.672302	Drama|Romance	Leonardo DiCaprio	Titanic
3024	George Lucas	282.0	460.935665	Action|Adventure|Fantasy|Sci-Fi	Harrison Ford	Star Wars: Episode IV - A New Hope
3080	Steven Spielberg	215.0	434.949459	Family|Sci-Fi	Henry Thomas	E.T. the Extra-Terrestrial
17	Joss Whedon	703.0	623.279547	Action|Adventure|Sci-Fi	Chris Hemsworth	The Avengers
509	Roger Allers	186.0	422.783777	Adventure|Animation|Drama|Family|Musical	Matthew Broderick	The Lion King
240	George Lucas	320.0	474.544677	Action|Adventure|Fantasy|Sci-Fi	Natalie Portman	Star Wars: Episode I - The Phantom Menace
66	Christopher Nolan	645.0	533.316061	Action|Crime|Drama|Thriller	Christian Bale	The Dark Knight
439	Gary Ross	673.0	407.999255	Adventure|Drama|Sci-Fi|Thriller	Jennifer Lawrence	The Hunger Games

In [ ]:

```

```

Checkpoint 2: You might spot two movies directed by `James Cameron` in the list.

- ### Subtask 3.4: Find IMDb Top 250

  1. Create a new dataframe `IMDb_Top_250` and store the top 250 movies with the highest IMDb Rating (corresponding to the column: `imdb_score`). Also make sure that for all of these movies, the `num_voted_users` is greater than 25,000. Also add a `Rank` column containing the values 1 to 250 indicating the ranks of the corresponding films.
  2. Extract all the movies in the `IMDb_Top_250` dataframe which are not in the English language and store them in a new dataframe named `Top_Foreign_Lang_Film`.

In [101]:

```
# Write your code for extracting the top 250 movies as per the IMDb score here. Make sure
that you store it in a new dataframe
# and name that dataframe as 'IMDb_Top_250'
IMDb_Top_250 = movies[movies['num_voted_users'] > 25000].sort_values(by='imdb_score',asc
ending=False).head(250)
IMDb_Top_250
```

Out[101]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	num_v
1937	Frank Darabont	199.0	28.341469	Crime|Drama	Morgan Freeman	The Shawshank Redemption	
3466	Francis Ford Coppola	208.0	134.821952	Crime|Drama	Al Pacino	The Godfather	
2837	Francis Ford Coppola	149.0	57.300000	Crime|Drama	Robert De Niro	The Godfather: Part II	
66	Christopher Nolan	645.0	533.316061	Action|Crime|Drama|Thriller	Christian Bale	The Dark Knight	
4498	Sergio Leone	181.0	6.100000	Western	Clint Eastwood	The Good, the Bad and the Ugly	
...	...	...	...	...	...	...	
4931	John Carney	232.0	9.437933	Drama|Music|Romance	Glen Hansard	Once	
2605	Ang Lee	287.0	128.067808	Action|Drama|Romance	Chen Chang	Crouching Tiger, Hidden Dragon	
3029	David O. Russell	410.0	93.571803	Biography|Drama|Sport	Christian Bale	The Fighter	
2177	Tim Burton	111.0	56.362352	Fantasy|Romance	Johnny Depp	Edward Scissorhands	
2487	George Cukor	82.0	72.000000	Drama|Family|Musical|Romance	Jeremy Brett	My Fair Lady	

250 rows × 14 columns

In [102]:

```
IMDb_Top_250['Rank'] = IMDb_Top_250['imdb_score'].rank(method='first',ascending=False)
IMDb_Top_250
```

Out[102]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	num_v
1937	Frank Darabont	199.0	28.341469	Crime|Drama	Morgan Freeman	The Shawshank Redemption	
3466	Francis Ford Coppola	208.0	134.821952	Crime|Drama	Al Pacino	The Godfather	
2837	Francis Ford Coppola	149.0	57.300000	Crime|Drama	Robert De Niro	The Godfather: Part II	
66	Christopher Nolan	645.0	533.316061	Action|Crime|Drama|Thriller	Christian Bale	The Dark Knight	
4498	Sergio Leone	181.0	6.100000	Western	Clint Eastwood	The Good, the Bad and the Ugly	
...	...	...	...	...	...	...	
4931	John Carney	232.0	9.437933	Drama|Music|Romance	Glen Hansard	Once	
						Crouching	

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	num_vo
2605	Ang Lee	287.0	128.067808	Action|Drama|Romance	Chen Chang	Tiger Hidden Dragon	
3029	David O. Russell	410.0	93.571803	Biography|Drama|Sport	Christian Bale	The Fighter	
2177	Tim Burton	111.0	56.362352	Fantasy|Romance	Johnny Depp	Edward Scissorhands	
2487	George Cukor	82.0	72.000000	Drama|Family|Musical|Romance	Jeremy Brett	My Fair Lady	

250 rows × 15 columns

In [106]:

```
IMDb_Top_250[IMDb_Top_250['language']!='English']
```

Out[106]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name
4498	Sergio Leone	181.0	6.100000	Western	Clint Eastwood
4747	Akira Kurosawa	153.0	0.269061	Action|Adventure|Drama	Takashi Shimura
4029	Fernando Meirelles	214.0	7.563397	Crime|Drama	Alice Braga
2373	Hayao Miyazaki	246.0	10.049886	Adventure|Animation|Family|Fantasy	Bunta Sugawara
4259	Florian Henckel von Donnersmarck	215.0	11.284657	Drama|Thriller	Sebastian Koch
4921	Majid Majidi	46.0	0.925402	Drama|Family	Bahare Seddiqi
2323	Hayao Miyazaki	174.0	2.298191	Adventure|Animation|Fantasy	Minnie Driver
2970	Wolfgang Petersen	96.0	11.433134	Adventure|Drama|Thriller|War	Jürgen Prochnow
4105	Chan-wook Park	305.0	2.181290	Drama|Mystery|Thriller	Min-sik Choi
4659	Asghar Farhadi	354.0	7.098492	Drama|Mystery	Shahab Hosseini
1329	S.S. Rajamouli	44.0	6.498000	Action|Adventure|Drama|Fantasy|War	Tamannaah Bhatia
1298	Jean-Pierre Jeunet	242.0	33.201661	Comedy|Romance	Mathieu Kassovitz
2734	Fritz Lang	260.0	0.026435	Drama|Sci-Fi	Brigitte Helm
4033	Thomas Vinterberg	349.0	0.610968	Drama	Thomas Bo Larsen
2829	Oliver Hirschbiegel	192.0	5.501940	Biography|Drama|History|War	Thomas Kretschmann
2551	Guillermo del Toro	406.0	37.623143	Drama|Fantasy|War	Ivana Baquero
4000	Juan José Campanella	262.0	20.167424	Drama|Mystery|Thriller	Ricardo Darín
3550	Denis Villeneuve	226.0	6.857096	Drama|Mystery|War	Lubna Azabal
2047	Hayao Miyazaki	212.0	4.710455	Adventure|Animation|Family|Fantasy	Christian Bale

	director_name	num_critic_for_reviews	gross	genres	actor_1_name
2830	Alejandro Amenábar	157.0	2.086345	Biography|Drama|Romance	Belén Rueda
2914	Je-kyu Kang	86.0	1.110186	Action|Drama|War	Min-sik Choi
4461	Thomas Vinterberg	98.0	1.647780	Drama	Ulrich Thomsen
3553	José Padilha	142.0	0.008060	Action|Crime|Drama|Thriller	Wagner Moura
3423	Katsuhiro Ôtomo	150.0	0.439162	Action|Animation|Sci-Fi	Mitsuo Iwata
4267	Alejandro G. Iñárritu	157.0	5.383834	Drama|Thriller	Adriana Barraza
3456	Vincent Paronnaud	242.0	4.443403	Animation|Biography|Drama|War	Catherine Deneuve
3344	Karan Johar	210.0	4.018695	Adventure|Drama|Thriller	Shah Rukh Khan
4144	Walter Salles	71.0	5.595428	Drama	Fernanda Montenegro
4284	Ari Folman	231.0	2.283276	Animation|Biography|Documentary|Drama|History|War	Ari Folman
4897	Sergio Leone	122.0	3.500000	Action|Drama|Western	Clint Eastwood
1171	Yimou Zhang	283.0	0.084961	Action|Adventure|History	Jet Li
2863	Clint Eastwood	251.0	13.753931	Drama|History|War	Yuki Matsuzaki
3264	Michael Haneke	447.0	0.225377	Drama|Romance	Isabelle Huppert
3510	Yash Chopra	29.0	2.921738	Drama|Musical|Romance	Shah Rukh Khan
3677	Christophe Barratier	112.0	3.629758	Drama|Music	Jean-Baptiste Maunier
4415	Fabián Bielinsky	94.0	1.221261	Crime|Drama|Thriller	Ricardo Darín
4640	Cristian Mungiu	233.0	1.185783	Drama	Anamaria Marinca
2605	Ang Lee	287.0	128.067808	Action|Drama|Romance	Chen Chang

Checkpoint 3: Can you spot `Veer-Zaara` in the dataframe?

- **### Subtask 3.5: Find the best directors**

  1. **Group the dataframe using the `director_name` column.**
  2. **Find out the top 10 directors for whom the mean of `imdb_score` is the highest and store them in a new dataframe `top10director`.**

```
In [109]:
```

```
# Write your code for extracting the top 10 directors here
top10director = movies.groupby('director_name').imdb_score.mean().sort_values(ascending=
False).head(10)
top10director
```

Out[109]:

```
director_name
Charles Chaplin       8.600000
Tony Kaye             8.600000
Ron Fricke            8.500000
Damien Chazelle       8.500000
Majid Majidi          8.500000
Alfred Hitchcock      8.500000
Sergio Leone          8.433333
Christopher Nolan     8.425000
Asghar Farhadi        8.400000
Richard Marquand      8.400000
Name: imdb_score, dtype: float64
```

Checkpoint 4: No surprises that `Damien Chazelle` (director of Whiplash and La La Land) is in this list.

- **### Subtask 3.6: Find popular genres**

You might have noticed the `genres` column in the dataframe with all the genres of the movies seperated by a pipe (`|`). Out of all the movie genres, the first two are most significant for any film.

1. **Extract the first two genres from the `genres` column and store them in two new columns: `genre_1` and `genre_2`. Some of the movies might have only one genre. In such cases, extract the single genre into both the columns, i.e. for such movies the `genre_2` will be the same as `genre_1`.**
2. **Group the dataframe using `genre_1` as the primary column and `genre_2` as the secondary column.**
3. **Find out the 5 most popular combo of genres by finding the mean of the gross values using the `gross` column and store them in a new dataframe named `PopGenre`.**

In [116]:

```
TempGenre = movies.genres.str.split('|',expand=True).iloc[:,0:2]
TempGenre.columns=['genre_1','genre_2']
TempGenre.genre_2.fillna(TempGenre.genre_1,inplace=True)
TempGenre
```

Out[116]:

	genre_1	genre_2
0	Action	Adventure
1	Action	Adventure
2	Action	Adventure
3	Action	Thriller
5	Action	Adventure
...	...	...
5033	Drama	Sci-Fi
5034	Thriller	Thriller
5035	Action	Crime
5037	Comedy	Drama
5042	Documentary	Documentary

3856 rows × 2 columns

In [117]:

```
movies = pd.concat([movies,TempGenre],axis=1)
movies
```

Out[117]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie_title	nun
0	James Cameron	723.0	760.505847	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder	Avatar	
1	Gore Verbinski	302.0	309.404152	Action|Adventure|Fantasy	Johnny Depp	Pirates of the Caribbean: At World's End	
2	Sam Mendes	602.0	200.074175	Action|Adventure|Thriller	Christoph Waltz	Spectre	
3	Christopher Nolan	813.0	448.130642	Action|Thriller	Tom Hardy	The Dark Knight Rises	
5	Andrew Stanton	462.0	73.058679	Action|Adventure|Sci-Fi	Daryl Sabara	John Carter	
...	...	...	...	...	...	...	
5033	Shane Carruth	143.0	0.424760	Drama|Sci-Fi|Thriller	Shane Carruth	Primer	
5034	Neill Dela Llana	35.0	0.070071	Thriller	Ian Gamazon	Cavite	
5035	Robert Rodriguez	56.0	2.040920	Action|Crime|Drama|Romance|Thriller	Carlos Gallardo	El Mariachi	
5037	Edward Burns	14.0	0.004584	Comedy|Drama	Kerry Bishé	Newlyweds	
5042	Jon Gunn	43.0	0.085222	Documentary	John August	My Date with Drew	

3856 rows × 16 columns

In [121]:

```
movies.groupby(['genre_1','genre_2']).gross.mean().sort_values(ascending=False).head(5)
```

Out[121]:

```
genre_1    genre_2
Family     Sci-Fi         434.949459
Adventure  Sci-Fi         228.627758
           Family         118.919540
           Animation      116.998550
Action     Adventure      109.595465
Name: gross, dtype: float64
```

Checkpoint 5: Well, as it turns out. `Family + Sci-Fi` is the most popular combo of genres out there!

- **### Subtask 3.7: Find the critic-favorite and audience-favorite actors**

  1. **Create three new dataframes namely, `Meryl_Streep`, `Leo_Caprio`, and `Brad_Pitt` which contain the movies in which the actors: 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' are the lead actors. Use only the `actor_1_name` column for extraction. Also, make sure that you use the names 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' for the said extraction.**
  2. **Append the rows of all these dataframes and store them in a new dataframe named `Combined`.**
  3. **Group the combined dataframe using the `actor_1_name` column.**
  4. **Find the mean of the `num_critic_for_reviews` and `num_user_for_review` and identify the actors which have the highest mean.**

In [126]:

```
# Write your code for creating three new dataframes here
Meryl_Streep = movies[movies['actor_1_name']=='Meryl Streep']
Leo_Caprio = movies[movies['actor_1_name']=='Leonardo DiCaprio']
```

```
Brad_Pitt = movies[movies['actor_1_name']=='Brad Pitt']
```

In [131]:

```
Combined = Meryl_Streep.append([Leo_Caprio,Brad_Pitt])
Combined
```

Out[131]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name
410	Nancy Meyers	187.0	112.703470	Comedy|Drama|Romance	Meryl Streep
1106	Curtis Hanson	42.0	46.815748	Action|Adventure|Crime|Thriller	Meryl Streep
1204	Nora Ephron	252.0	94.125426	Biography|Drama|Romance	Meryl Streep
1408	David Frankel	208.0	124.732962	Comedy|Drama|Romance	Meryl Streep
1483	Robert Redford	227.0	14.998070	Drama|Thriller|War	Meryl Streep
1575	Sydney Pollack	66.0	87.100000	Biography|Drama|Romance	Meryl Streep
1618	David Frankel	234.0	63.536011	Comedy|Drama|Romance	Meryl Streep
1674	Carl Franklin	64.0	23.209440	Drama	Meryl Streep
1925	Stephen Daldry	174.0	41.597830	Drama|Romance	Meryl Streep
2781	Phyllida Lloyd	331.0	29.959436	Biography|Drama|History	Meryl Streep
3135	Robert Altman	211.0	20.338609	Comedy|Drama|Music	Meryl Streep
26	James Cameron	315.0	658.672302	Drama|Romance	Leonardo DiCaprio
50	Baz Luhrmann	490.0	144.812796	Drama|Romance	Leonardo DiCaprio
97	Christopher Nolan	642.0	292.568851	Action|Adventure|Sci-Fi|Thriller	Leonardo DiCaprio
179	Alejandro G. Iñárritu	556.0	183.635922	Adventure|Drama|Thriller|Western	Leonardo DiCaprio
257	Martin Scorsese	267.0	102.608827	Biography|Drama	Leonardo DiCaprio
296	Quentin Tarantino	765.0	162.804648	Drama|Western	Leonardo DiCaprio
307	Edward Zwick	166.0	57.366262	Adventure|Drama|Thriller	Leonardo DiCaprio
308	Martin Scorsese	606.0	116.866727	Biography|Comedy|Crime|Drama	Leonardo DiCaprio
326	Martin Scorsese	233.0	77.679638	Crime|Drama	Leonardo DiCaprio
361	Martin Scorsese	352.0	132.373442	Crime|Drama|Thriller	Leonardo DiCaprio
452	Martin Scorsese	490.0	127.968405	Mystery|Thriller	Leonardo DiCaprio
641	Ridley Scott	238.0	39.380442	Action|Drama|Thriller	Leonardo DiCaprio
911	Steven Spielberg	194.0	164.435221	Biography|Crime|Drama	Leonardo DiCaprio
990	Danny Boyle	118.0	39.778599	Adventure|Drama|Thriller	Leonardo DiCaprio

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	R
1114	Sam Mendes	323.0	22.877808	Drama|Romance	Leonardo DiCaprio	
1422	Randall Wallace	83.0	56.876365	Action|Adventure	Leonardo DiCaprio	th
1453	Clint Eastwood	392.0	37.304950	Biography|Crime|Drama	Leonardo DiCaprio	
1560	Sam Raimi	63.0	18.636537	Action|Thriller|Western	Leonardo DiCaprio	a
2067	Jerry Zaks	45.0	12.782508	Drama	Leonardo DiCaprio	
2757	Baz Luhrmann	106.0	46.338728	Drama|Romance	Leonardo DiCaprio	
3476	Baz Luhrmann	490.0	144.812796	Drama|Romance	Leonardo DiCaprio	
101	David Fincher	362.0	127.490802	Drama|Fantasy|Romance	Brad Pitt	
147	Wolfgang Petersen	220.0	133.228348	Adventure	Brad Pitt	
254	Steven Soderbergh	198.0	125.531634	Crime|Thriller	Brad Pitt	
255	Doug Liman	233.0	186.336103	Action|Comedy|Crime|Romance|Thriller	Brad Pitt	
382	Tony Scott	142.0	0.026871	Action|Crime|Thriller	Brad Pitt	
400	Steven Soderbergh	186.0	183.405771	Crime|Thriller	Brad Pitt	
470	David Ayer	406.0	85.707116	Action|Drama|War	Brad Pitt	
611	Jean-Jacques Annaud	76.0	37.901509	Adventure|Biography|Drama|History|War	Brad Pitt	:
683	David Fincher	315.0	37.023395	Drama	Brad Pitt	
792	Patrick Gilmore	98.0	26.288320	Adventure|Animation|Comedy|Drama|Family|Fantas...	Brad Pitt	Lo
940	Neil Jordan	120.0	105.264608	Drama|Fantasy|Horror	Brad Pitt	In t -
1490	Terrence Malick	584.0	13.303319	Drama|Fantasy	Brad Pitt	
1722	Andrew Dominik	273.0	3.904982	Biography|Crime|Drama|History|Western	Brad Pitt	As J
2204	Alejandro G. Iñárritu	285.0	34.300771	Drama	Brad Pitt	
2333	Angelina Jolie Pitt	131.0	0.531009	Drama|Romance	Brad Pitt	
2682	Andrew Dominik	414.0	14.938570	Crime|Thriller	Brad Pitt	|
2898	Tony Scott	122.0	12.281500	Action|Crime|Drama|Romance|Thriller	Brad Pitt	

In [ ]:

```
num_critic_for_reviews and num_user_for_review
```

In [133]:

```
Combined.groupby('actor_1_name').num_critic_for_reviews.mean()
```

Out[133]:

```
actor_1_name
Brad Pitt            245.000000
Leonardo DiCaprio    330.190476
Meryl Streep         181.454545
Name: num_critic_for_reviews, dtype: float64
```

In [135]:

```
Combined
```

Out[135]:

	director_name	num_critic_for_reviews	gross	genres	actor_1_name
410	Nancy Meyers	187.0	112.703470	Comedy|Drama|Romance	Meryl Streep
1106	Curtis Hanson	42.0	46.815748	Action|Adventure|Crime|Thriller	Meryl Streep
1204	Nora Ephron	252.0	94.125426	Biography|Drama|Romance	Meryl Streep
1408	David Frankel	208.0	124.732962	Comedy|Drama|Romance	Meryl Streep
1483	Robert Redford	227.0	14.998070	Drama|Thriller|War	Meryl Streep
1575	Sydney Pollack	66.0	87.100000	Biography|Drama|Romance	Meryl Streep
1618	David Frankel	234.0	63.536011	Comedy|Drama|Romance	Meryl Streep
1674	Carl Franklin	64.0	23.209440	Drama	Meryl Streep
1925	Stephen Daldry	174.0	41.597830	Drama|Romance	Meryl Streep
2781	Phyllida Lloyd	331.0	29.959436	Biography|Drama|History	Meryl Streep
3135	Robert Altman	211.0	20.338609	Comedy|Drama|Music	Meryl Streep
26	James Cameron	315.0	658.672302	Drama|Romance	Leonardo DiCaprio
50	Baz Luhrmann	490.0	144.812796	Drama|Romance	Leonardo DiCaprio
97	Christopher Nolan	642.0	292.568851	Action|Adventure|Sci-Fi|Thriller	Leonardo DiCaprio
179	Alejandro G. Iñárritu	556.0	183.635922	Adventure|Drama|Thriller|Western	Leonardo DiCaprio
257	Martin Scorsese	267.0	102.608827	Biography|Drama	Leonardo DiCaprio
296	Quentin Tarantino	765.0	162.804648	Drama|Western	Leonardo DiCaprio
307	Edward Zwick	166.0	57.366262	Adventure|Drama|Thriller	Leonardo DiCaprio
308	Martin Scorsese	606.0	116.866727	Biography|Comedy|Crime|Drama	Leonardo DiCaprio
326	Martin Scorsese	233.0	77.679638	Crime|Drama	Leonardo DiCaprio
361	Martin Scorsese	352.0	132.373442	Crime|Drama|Thriller	Leonardo DiCaprio
	Martin				Leonardo

	director_name	num_critic_for_reviews	gross	genres	actor_1_name	
452	Martin Scorsese	490.0	127.968405	Mystery|Thriller	Leonardo DiCaprio	Sl
641	Ridley Scott	238.0	39.380442	Action|Drama|Thriller	Leonardo DiCaprio	
911	Steven Spielberg	194.0	164.435221	Biography|Crime|Drama	Leonardo DiCaprio	
990	Danny Boyle	118.0	39.778599	Adventure|Drama|Thriller	Leonardo DiCaprio	
1114	Sam Mendes	323.0	22.877808	Drama|Romance	Leonardo DiCaprio	R
1422	Randall Wallace	83.0	56.876365	Action|Adventure	Leonardo DiCaprio	th
1453	Clint Eastwood	392.0	37.304950	Biography|Crime|Drama	Leonardo DiCaprio	
1560	Sam Raimi	63.0	18.636537	Action|Thriller|Western	Leonardo DiCaprio	a
2067	Jerry Zaks	45.0	12.782508	Drama	Leonardo DiCaprio	
2757	Baz Luhrmann	106.0	46.338728	Drama|Romance	Leonardo DiCaprio	
3476	Baz Luhrmann	490.0	144.812796	Drama|Romance	Leonardo DiCaprio	
101	David Fincher	362.0	127.490802	Drama|Fantasy|Romance	Brad Pitt	
147	Wolfgang Petersen	220.0	133.228348	Adventure	Brad Pitt	
254	Steven Soderbergh	198.0	125.531634	Crime|Thriller	Brad Pitt	
255	Doug Liman	233.0	186.336103	Action|Comedy|Crime|Romance|Thriller	Brad Pitt	
382	Tony Scott	142.0	0.026871	Action|Crime|Thriller	Brad Pitt	
400	Steven Soderbergh	186.0	183.405771	Crime|Thriller	Brad Pitt	
470	David Ayer	406.0	85.707116	Action|Drama|War	Brad Pitt	
611	Jean-Jacques Annaud	76.0	37.901509	Adventure|Biography|Drama|History|War	Brad Pitt	
683	David Fincher	315.0	37.023395	Drama	Brad Pitt	
792	Patrick Gilmore	98.0	26.288320	Adventure|Animation|Comedy|Drama|Family|Fantas...	Brad Pitt	L
940	Neil Jordan	120.0	105.264608	Drama|Fantasy|Horror	Brad Pitt	In
1490	Terrence Malick	584.0	13.303319	Drama|Fantasy	Brad Pitt	
1722	Andrew Dominik	273.0	3.904982	Biography|Crime|Drama|History|Western	Brad Pitt	As J
2204	Alejandro G. Iñárritu	285.0	34.300771	Drama	Brad Pitt	
2333	Angelina Jolie Pitt	131.0	0.531009	Drama|Romance	Brad Pitt	
2682	Andrew Dominik	414.0	14.938570	Crime|Thriller	Brad Pitt	

	director_name	num_critic_for_reviews	gross	genres	actor_1_name
2898	Tony Scott	122.0	12.281500	Action|Crime|Drama|Romance|Thriller	Brad Pitt

In [138]:

```
Combined.num_user_for_reviews = Combined.num_user_for_reviews.astype('int')
Combined.num_user_for_reviews
```

Out[138]:

```
410      214
1106      69
1204     277
1408     631
1483     298
1575     200
1618     178
1674     112
1925     660
2781     350
3135     280
26      2528
50       753
97      2803
179     1188
257      799
296     1193
307      657
308     1138
326     1166
361     2054
452      964
641      263
911      667
990      548
1114     414
1422     244
1453     279
1560     216
2067      71
2757     506
3476     753
101      822
147     1694
254      627
255      798
382      361
400      845
470      701
611      119
683     2968
792       91
940      406
1490     975
1722     415
2204     908
2333      61
2682     369
2898     460
Name: num_user_for_reviews, dtype: int32
```

In [139]:

```
Combined.groupby('actor_1_name').num_user_for_reviews.mean()
```

Out[139]:

```
actor_1_name
Brad Pitt            742.352941
Leonardo DiCaprio    914.476190
Meryl Streep         297.181818
```

```
Name: num_user_for_reviews, dtype: float64
```

In [140]:

```
Combined.groupby('actor_1_name')[['num_critic_for_reviews','num_user_for_reviews']].mean(
)
```

Out[140]:

	num_critic_for_reviews	num_user_for_reviews
actor_1_name		
Brad Pitt	245.000000	742.352941
Leonardo DiCaprio	330.190476	914.476190
Meryl Streep	181.454545	297.181818

In [ ]:

```

```

Checkpoint 6: `Leonardo` **has aced both the lists!**