IMDB DATASET

In [5]:

```
# How many rows are there in the IMDB dataset?

import pandas as pd
import numpy as np
df = pd.read_csv("IMDB-Movie-Data.xlsx.csv")
print(df)

# What is the 75th percentile of rating in the IMDB dataset?

df.describe()
```

```
       ID                    Title      Genre             Director  Year  \
0       1  Guardians of the Galaxy     Action           James Gunn  2014   
1       2               Prometheus  Adventure         Ridley Scott  2012   
2       3                    Split     Horror   M. Night Shyamalan  2016   
3       4                     Sing  Animation  Christophe Lourdelet  2016   
4       5            Suicide Squad     Action           David Ayer  2016   
..    ...                      ...        ...                  ...   ...   
995   996     Secret in Their Eyes      Crime            Billy Ray  2015   
996   997          Hostel: Part II     Horror             Eli Roth  2007   
997   998   Step Up 2: The Streets      Drama         Jon M. Chu  2008   
998   999             Search Party  Adventure       Scot Armstrong  2014   
999  1000               Nine Lives     Comedy     Barry Sonnenfeld  2016   

     Runtime_minutes  Rating   Votes  Revenue_millions  
0                121     8.1  757074            333.13  
1                124     7.0  485820            126.46  
2                117     7.3  157606            138.12  
3                108     7.2   60545            270.32  
4                123     6.2  393727            325.02  
..               ...     ...     ...               ...  
995              111     6.2   27585               NaN  
996               94     5.5   73152             17.54  
997               98     6.2   70699             58.01  
998               93     5.6    4881               NaN  
999               87     5.3   12435             19.64  

[1000 rows x 9 columns]
```

Out[5]:

	ID	Year	Runtime_minutes	Rating	Votes	Revenue_millions
count	1000.000000	1000.000000	1000.000000	1000.000000	1.000000e+03	872.000000
mean	500.500000	2012.783000	113.172000	6.723200	1.698083e+05	82.956376
std	288.819436	3.205962	18.810908	0.945429	1.887626e+05	103.253540
min	1.000000	2006.000000	66.000000	1.900000	6.100000e+01	0.000000
25%	250.750000	2010.000000	100.000000	6.200000	3.630900e+04	13.270000
50%	500.500000	2014.000000	111.000000	6.800000	1.107990e+05	47.985000
75%	750.250000	2016.000000	123.000000	7.400000	2.399098e+05	113.715000
max	1000.000000	2016.000000	191.000000	9.000000	1.791916e+06	936.630000

In [6]:

```
# How many NA values are there in the field 'Revenue'?

df.isna().value_counts()
```

Out[6]:

```
ID     Title  Genre  Director  Year   Runtime_minutes  Rating  Votes  Revenue_millions
False  False  False  False     False  False            False   False  False
872

                                                                      True
128
dtype: int64
```

In [7]:

```
df = pd.read_csv("IMDB-Movie-Data.xlsx.csv")
df
```

Out[7]:

	ID	Title	Genre	Director	Year	Runtime_minutes	Rating	Votes	Revenue_millions
0	1	Guardians of the Galaxy	Action	James Gunn	2014	121	8.1	757074	333.13
1	2	Prometheus	Adventure	Ridley Scott	2012	124	7.0	485820	126.46
2	3	Split	Horror	M. Night Shyamalan	2016	117	7.3	157606	138.12
3	4	Sing	Animation	Christophe Lourdelet	2016	108	7.2	60545	270.32
4	5	Suicide Squad	Action	David Ayer	2016	123	6.2	393727	325.02
...	...	...	...	...	...	...	...	...	...
995	996	Secret in Their Eyes	Crime	Billy Ray	2015	111	6.2	27585	NaN
996	997	Hostel: Part II	Horror	Eli Roth	2007	94	5.5	73152	17.54
997	998	Step Up 2: The Streets	Drama	Jon M. Chu	2008	98	6.2	70699	58.01
998	999	Search Party	Adventure	Scot Armstrong	2014	93	5.6	4881	NaN
999	1000	Nine Lives	Comedy	Barry Sonnenfeld	2016	87	5.3	12435	19.64

1000 rows × 9 columns

In [8]:

```
# How many movies have revenue higher than 75 million?

revenue = 75.000000
filter1 = df[df["Revenue_millions"] > revenue ]
filter2 = filter1.count()
filter2
```

Out[8]:

```
ID                  318
Title               318
Genre               318
Director            318
Year                318
Runtime_minutes     318
Rating              318
Votes               318
Revenue_millions    318
dtype: int64
```

In [8]:

```
# How many movies have revenue greater than 50 million but rating less than 7?

revenue = 50.000000
filter3 = df[(df["Revenue_millions"] > revenue) & (df["Rating"] < 7)]
filter4 = filter3.count()
filter4
```

Out[8]:

```
ID                  211
```

```
Title              211
Genre              211
Director           211
Year               211
Runtime_minutes    211
Rating             211
Votes              211
Revenue_millions   211
dtype: int64
```

In [9]:

```
# What is the total revenue generated by movies in the year 2015?

x = df[(df["Year"]==2015)]
x1 = x.iloc[:,8]
x2 = x1.sum()
x2
```

Out[9]:

```
8854.119999999999
```

In [10]:

```
# What is the average rating for the genre adventure in the year 2015?

df
x = df[(df["Year"]==2015) & (df["Genre"]=="Adventure")]
x2 = x["Rating"].mean()
x2
```

Out[10]:

```
6.8
```

In [11]:

```
# What is the average duration of movies in rows 75 to 150? Please note that the rows in
python start from 0.

x = df.iloc[75:150]
x
avg = x["Runtime_minutes"].mean()
avg
```

Out[11]:

```
127.61333333333333
```

In [12]:

```
# Which year generated the highest revenue?

df.groupby(by="Year").sum().sort_values(by="Revenue_millions",ascending=False)
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_5340\3506442192.py:3: FutureWarning: The default
value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric
_only will default to False. Either specify numeric_only or select only columns which sho
uld be valid for the function.
  df.groupby(by="Year").sum().sort_values(by="Revenue_millions",ascending=False)
```

Out[12]:

	ID	Runtime_minutes	Rating	Votes	Revenue_millions
Year					
2016	111804	31890	1911.7	14431751	11211.65
2015	62407	14541	838.5	14697230	8854.12
2014	50272	11220	670.1	19985162	7997.40

Year	ID	Runtime_minutes	Rating	Votes	Revenue_millions
2013	50617	10562	619.9	19933518	7666.72
2012	34104	7623	443.2	18254470	6910.29
2010	35239	6668	409.6	15166939	5989.65
2011	37143	7220	430.8	15169789	5431.96
2009	29928	5922	355.0	13044813	5292.26
2008	32533	5763	352.8	14326280	5053.22
2007	30559	6446	378.1	12949545	4306.23
2006	25894	5317	313.5	11848758	3624.46

In [13]:

```
# What is the maximum revenue out of (10,20,30,40,50) rows?

df
x = df.iloc[10:51:10]
x1 = x["Revenue_millions"].max()
x1
```

Out[13]:

```
936.63
```

In [14]:

```
# How many movies with the genres 'Adventure', 'Action', 'Horror', and
# 'Crime' exist in the IMDB dataset?

df[(df["Genre"].isin(["Adventure",'Action',"Horror","Crime"]))].count()
```

Out[14]:

```
ID                  485
Title               485
Genre               485
Director            485
Year                485
Runtime_minutes     485
Rating              485
Votes               485
Revenue_millions    436
dtype: int64
```

REMOVE AND REPLACE NULL VALUES CLEANED IMDB DATASET

In [15]:

```
df = df.dropna(how='any')
df
```

Out[15]:

	ID	Title	Genre	Director	Year	Runtime_minutes	Rating	Votes	Revenue_millions
0	1	Guardians of the Galaxy	Action	James Gunn	2014	121	8.1	757074	333.13
1	2	Prometheus	Adventure	Ridley Scott	2012	124	7.0	485820	126.46
2	3	Split	Horror	M. Night Shyamalan	2016	117	7.3	157606	138.12
3	4	Sing	Animation	Christophe Lourdelet	2016	108	7.2	60545	270.32
4	5	Suicide Squad	Action	David Ayer	2016	123	6.2	393727	325.02
...	...	...	...	...	...	...	...	...	...
993	994	Resident Evil: Afterlife	Action	Paul W.S. Anderson	2010	97	5.9	140900	60.13

	ID	Title	Genre	Director	Year	Runtime_minutes	Rating	Votes	Revenue_millions
993	994	Resident Evil: Afterlife	Action	Paul W.S. Anderson	2010	97	5.9	140900	60.13
994	995	Project X	Comedy	Nima Nourizadeh	2012	88	6.7	164088	54.72
996	997	Hostel: Part II	Horror	Eli Roth	2007	94	5.5	73152	17.54
997	998	Step Up 2: The Streets	Drama	Jon M. Chu	2008	98	6.2	70699	58.01
999	1000	Nine Lives	Comedy	Barry Sonnenfeld	2016	87	5.3	12435	19.64

872 rows × 9 columns

In [16]:

```
# Create a genre-level report with metrics average rating, the average number
# of votes, and average revenue. What is the average rating of the 'Horror' genre?
# (Round to 2 decimal places)

a = df[(df["Genre"]=="Horror")]
a1 = a["Rating"].mean()
a2 = round(a1,2)
a2
```

Out[16]:

```
6.1
```

In [17]:

```
# What is the  % revenue of the movie 'Split' in its respective genre and year?

x= df[(df["Title"]=="Split")]
print(x)
x1 = df[(df["Genre"]=="Horror") & (df["Year"]==2016)]
print(x1)
x2 = x1["Revenue_millions"].sum()
print(x2)
perc = x["Revenue_millions"] * 100 / x2
print(perc)
print('\n')
print("% revenue of the movie 'Split' in its respective genre and year :", perc)
```

```
   ID  Title   Genre             Director  Year  Runtime_minutes  Rating  \
2   3  Split  Horror  M. Night Shyamalan  2016              117     7.3   

    Votes  Revenue_millions  
2  157606            138.12  
      ID                 Title   Genre             Director  Year  \
2      3                 Split  Horror    M. Night Shyamalan  2016   
27    28            Dead Awake  Horror        Phillip Guzman  2016   
97    98              The Void  Horror      Jeremy Gillespie  2016   
116  117        The Neon Demon  Horror  Nicolas Winding Refn  2016   
178  179      The Conjuring 2  Horror             James Wan  2016   
193  194                Morgan  Horror            Luke Scott  2016   
258  259            Lights Out  Horror     David F. Sandberg  2016   
461  462               The Boy  Horror    William Brent Bell  2016   
531  532        Friend Request  Horror      Simon Verhoeven  2016   
723  724           Blair Witch  Horror          Adam Wingard  2016   
748  749  Ouija: Origin of Evil  Horror         Mike Flanagan  2016   
776  777                    31  Horror            Rob Zombie  2016   

     Runtime_minutes  Rating   Votes  Revenue_millions  
2                117     7.3  157606            138.12  
27                99     4.7     523              0.01  
97                90     5.8    9247              0.15  
116              118     6.2   50359              1.33  
178              134     7.4  137203            102.46  
193               92     5.8   22107              3.91  
258               81     6.4   69823             67.24  
461               97     6.0   51235             35.79  
531               92     5.4   12758             64.03  
723               89     5.1   26088             20.75  
748               99     6.1   30035             34.90  
776              102     5.1   10871              0.78  
```

```
469.46999999999999
2    29.42041
Name: Revenue_millions, dtype: float64


% revenue of the movie 'Split' in its respective genre and year : 2    29.42041
Name: Revenue_millions, dtype: float64
```

In [18]:

```
# Add a column 'Votes_norm' in the IMDB dataset using apply() function where Votes_norm i
s

# [Votes - min(Votes)]*10/[max(votes) - min(votes)

# The above formula is the normalization formula and converts votes into a scale of 0-10.


# What is the average 'Votes_norm' ? (Round to two decimal places)


df['Votes_norm'] = df['Votes'].apply(lambda x: ((x - df['Votes'].min()) * 10) / (df['Vot
es'].max() - df['Votes'].min()))
avg_votes = df["Votes_norm"].mean()
round_avg = round(avg_votes,2)
round_avg
```

Out[18]:

```
1.06
```

In [43]:

```
# Add a column 'Total_rating'  in the IMDB dataset using apply()
# function where Total_rating is 'Rating'+ 'Votes_norm'.What is the highest 'Total_rating
' ?

df["Total_rating"] = df["Rating"] + df["Votes_norm"]
highest_Rating = df["Total_rating"].max()
highest_Rating
```

Out[43]:

```
19.0
```

In [57]:

```
# How many directors have created movies in the highest number of genres?


# Step 1: Group the dataframe by 'director' and 'genre' columns and count the number of g
enres each director has created movies in
director_genre_counts = df.groupby(['Director', 'Genre']).size().reset_index(name='count
')

# Step 2: Find the maximum count of genres among the directors
max_genre_count = director_genre_counts['count'].max()

# Step 3: Filter the grouped dataframe to select only the directors who have created movi
es in the highest number of genres

directors_with_max_genres = director_genre_counts[director_genre_counts['count'] == max_
genre_count]['Director']

num_directors_highest_genres = directors_with_max_genres.nunique()

num_directors_highest_genres
```

Out[57]:

```
1
```

In [ ]: