```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving IMDB-Movie-Data.csv to IMDB-Movie-Data.csv

```
import pandas as pd
import numpy as np

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

1] Find the number of movies in the dataset.

```
num_movies = len(df)
print(num_movies)
```

```
1000
```

2] Find the average rating of all movies.

```
average_rating = df['Rating'].mean()
print(average_rating)
```

```
6.723199999999999
```

3] Find the highest-rated movie.

```
highest_rated_movie = df.loc[df['Rating'].idxmax()]
print(highest_rated_movie[['Title', 'Rating']])
```

```
Title     The Dark Knight
Rating                9.0
Name: 54, dtype: object
```

4]Find the number of unique genres.

```
unique_genres = df['Genre'].nunique()
print(unique_genres)
```

```
207
```

5] Find the movie with the maximum number of votes.

```
most_voted_movie = df.loc[df['Votes'].idxmax()]
print(most_voted_movie[['Title', 'Votes']])
```

```
Title    The Dark Knight
Votes            1791916
Name: 54, dtype: object
```

6] List the top 5 movies with the highest revenue.

```
top5_revenue = df.sort_values(by='Revenue (Millions)', ascending=False).head(5)
print(top5_revenue[['Title', 'Revenue (Millions)']])
```

```
                                          Title  Revenue (Millions)
50  Star Wars: Episode VII - The Force Awakens              936.63
87                                      Avatar              760.51
85                              Jurassic World              652.18
76                                The Avengers              623.28
54                             The Dark Knight              533.32
```

7] Find how many movies were released each year.

```
movies_per_year = df['Year'].value_counts().sort_index()
print(movies_per_year)
```

```
Year
2006     44
2007     53
2008     52
2009     51
2010     60
2011     63
2012     64
2013     91
2014     98
2015    127
2016    297
Name: count, dtype: int64
```

8] Find the correlation between Rating and Revenue.

```
correlation = df['Rating'].corr(df['Revenue (Millions)'])
print(f"Correlation between Rating and Revenue: {correlation:.2f}")
```

```
Correlation between Rating and Revenue: 0.22
```

9] Find the average Runtime of movies.

```
average_runtime = df['Runtime (Minutes)'].mean()
print(f"Average Runtime: {average_runtime:.2f} minutes")
```

```
Average Runtime: 113.17 minutes
```

10] How many unique Directors are there?

```
unique_directors = df['Director'].nunique()
print(f"Unique Directors: {unique_directors}")
```

```
Unique Directors: 644
```

11] List the top 10 Directors with most movies.

```
top_directors = df['Director'].value_counts().head(10)
print(top_directors)
```

```
Director
Ridley Scott          8
David Yates           6
M. Night Shyamalan    6
Paul W.S. Anderson    6
Michael Bay           6
Zack Snyder           5
Denis Villeneuve      5
Woody Allen           5
Peter Berg            5
Danny Boyle           5
Name: count, dtype: int64
```

12] Find the average Revenue for each Genre.

```
# First split genres
df_genre = df.assign(Genre=df['Genre'].str.split(','))
df_genre = df_genre.explode('Genre')
df_genre['Genre'] = df_genre['Genre'].str.strip()

avg_revenue_genre = df_genre.groupby('Genre')['Revenue (Millions)'].mean().sort_values(as
print(avg_revenue_genre)
```

```
Genre
Animation    191.223404
Adventure    154.177024
Sci-Fi       135.552545
Fantasy      131.850108
Family       126.175714
Action       124.494476
Western      111.824000
```

```
Musical       81.642000
Comedy        75.750784
Thriller      69.577255
Sport         65.042500
Crime         61.804769
Biography     55.801600
Mystery       54.627640
War           53.433000
History       52.923846
Drama         49.844205
Music         47.070000
Romance       42.503023
Horror        36.705269
Name: Revenue (Millions), dtype: float64
```

13] Find the average Runtime of movies.

```
average_runtime = df['Runtime (Minutes)'].mean()
print(f"Average Runtime: {average_runtime:.2f} minutes")
```

```
Average Runtime: 113.17 minutes
```

14] Find the top 5 longest movies by Runtime.

```
top5_runtime = df.sort_values(by='Runtime (Minutes)', ascending=False).head(5)
print(top5_runtime[['Title', 'Runtime (Minutes)']])
```

```
                     Title  Runtime (Minutes)
828              Grindhouse                191
88         The Hateful Eight                187
965            Inland Empire                180
311           La vie d'Adèle                180
82   The Wolf of Wall Street                180
```

15] Find how many movies have a Rating greater than 8.

```
highly_rated_movies = df[df['Rating'] > 8].shape[0]
print(f"Movies with Rating > 8: {highly_rated_movies}")
```

```
Movies with Rating > 8: 59
```

16] List movies with missing Metascore values.

```
missing_metascore = df[df['Metascore'].isnull()]
print(missing_metascore[['Title']])
```

```
                        Title
25           Paris pieds nus
26   Bahubali: The Beginning
27                Dead Awake
39                 5- 25- 77
42   Don't Fuck in the Woods
..                       ...
967                 The Walk
969          The Lone Ranger
971                Disturbia
989                    Selma
992     Take Me Home Tonight

[64 rows x 1 columns]
```

17]Find top 5 Actors with most appearances.

```
# Split actors
df_actor = df.assign(Actors=df['Actors'].str.split(','))
df_actor = df_actor.explode('Actors')
df_actor['Actors'] = df_actor['Actors'].str.strip()

top5_actors = df_actor['Actors'].value_counts().head(5)
print(top5_actors)
```

```
Actors
Mark Wahlberg     15
Hugh Jackman      14
Christian Bale    13
Brad Pitt         13
Channing Tatum    12
Name: count, dtype: int64
```

18] How many movies have Revenue above 100 million?

```
high_revenue_movies = df[df['Revenue (Millions)'] > 100].shape[0]
print(f"Movies with Revenue > $100 Million: {high_revenue_movies}")
```

```
Movies with Revenue > $100 Million: 250
```

19] Find the average Metascore of movies released after 2010.

```
avg_metascore_post2010 = df[df['Year'] > 2010]['Metascore'].mean()
print(f"Average Metascore (Post 2010): {avg_metascore_post2010:.2f}")
```

```
Average Metascore (Post 2010): 58.54
```

20] Find the Genre with highest average Rating.

```
df_genre = df.assign(Genre=df['Genre'].str.split(','))
df_genre = df_genre.explode('Genre')
df_genre['Genre'] = df_genre['Genre'].str.strip()

highest_rated_genre = df_genre.groupby('Genre')['Rating'].mean().sort_values(ascending=Fal
print(highest_rated_genre)
```

```
Genre
War          7.353846
Animation    7.324490
Biography    7.290123
History      7.127586
Music        7.075000
Sport        7.011111
Drama        6.953801
Musical      6.940000
Mystery      6.886792
Crime        6.786667
Adventure    6.772201
Western      6.771429
Sci-Fi       6.716667
Romance      6.685816
Family       6.684314
Comedy       6.647670
Action       6.614521
Thriller     6.593333
Fantasy      6.548515
Horror       6.089916
Name: Rating, dtype: float64
```