```python
import pandas as pd
df=pd.read_csv("imdb_movies.csv")
df
```

```
                                         names      date_x  score  \
0                                    Creed III  03/02/2023   73.0   
1                     Avatar: The Way of Water  12/15/2022   78.0   
2                  The Super Mario Bros. Movie  04/05/2023   76.0   
3                                      Mummies  01/05/2023   70.0   
4                                    Supercell  03/17/2023   61.0   
...                                        ...         ...    ...   
10173                       20th Century Women  12/28/2016   73.0   
10174  Delta Force 2: The Colombian Connection  08/24/1990   54.0   
10175                         The Russia House  12/21/1990   61.0   
10176          Darkman II: The Return of Durant  07/11/1995   55.0   
10177        The Swan Princess: A Royal Wedding  07/20/2020   70.0   

                                                   genre  \
0                                          Drama, Action   
1                     Science Fiction, Adventure, Action   
2             Animation, Adventure, Family, Fantasy, Comedy   
3             Animation, Comedy, Family, Adventure, Fantasy   
4                                                 Action   
...                                                  ...   
10173                                              Drama   
10174                                             Action   
10175                            Drama, Thriller, Romance   
10176  Action, Adventure, Science Fiction, Thriller, ...   
10177                          Animation, Family, Fantasy   

                                                overview  \
0      After dominating the boxing world, Adonis Cree...   
1      Set more than a decade after the events of the...   
2      While working underground to fix a water main,...   
3      Through a series of unfortunate events, three ...   
4      Good-hearted teenager William always lived in ...   
...                                                  ...   
10173  In 1979 Santa Barbara, California, Dorothea Fi...   
10174  When DEA agents are taken captive by a ruthles...   
10175  Barley Scott Blair, a Lisbon-based editor of R...   
10176  Darkman and Durant return and they hate each o...   
10177  Princess Odette and Prince Derek are going to ...   

                                                    crew  \
0      Michael B. Jordan, Adonis Creed, Tessa Thompso...   
1      Sam Worthington, Jake Sully, Zoe Saldaña, Neyt...   
2      Chris Pratt, Mario (voice), Anya Taylor-Joy, P...   
3      Óscar Barberán, Thut (voice), Ana Esther Albor...   
4      Skeet Ulrich, Roy Cameron, Anne Heche, Dr Quin...   
...                                                  ...   
```

```
10173  Annette Bening, Dorothea Fields, Lucas Jade Zu...
10174  Chuck Norris, Col. Scott McCoy, Billy Drago, R...
10175  Sean Connery, Bartholomew 'Barley' Scott Blair...
10176  Larry Drake, Robert G. Durant, Arnold Vosloo, ...
10177  Nina Herzog, Princess Odette (voice), Yuri Low...

                                    orig_title     status  \
0                                    Creed III   Released
1                    Avatar: The Way of Water   Released
2                  The Super Mario Bros. Movie   Released
3                                       Momias   Released
4                                    Supercell   Released
...                                        ...        ...
10173                     20th Century Women   Released
10174  Delta Force 2: The Colombian Connection   Released
10175                      The Russia House   Released
10176         Darkman II: The Return of Durant   Released
10177       The Swan Princess: A Royal Wedding   Released

                  orig_lang     budget_x       revenue country
0                   English   75000000.0  2.716167e+08      AU
1                   English  460000000.0  2.316795e+09      AU
2                   English  100000000.0  7.244590e+08      AU
3       Spanish, Castilian   12300000.0  3.420000e+07      AU
4                   English   77000000.0  3.409420e+08      US
...                     ...          ...           ...     ...
10173               English    7000000.0  9.353729e+06      US
10174               English    9145817.8  6.698361e+06      US
10175               English   21800000.0  2.299799e+07      US
10176               English  116000000.0  4.756613e+08      US
10177               English   92400000.0  5.394018e+08      GB

[10178 rows x 12 columns]

df.columns

Index(['names', 'date_x', 'score', 'genre', 'overview', 'crew',
'orig_title',
       'status', 'orig_lang', 'budget_x', 'revenue', 'country'],
      dtype='object')
```

Data Overview and Basic Exploration

```
#Use .info() to understand the data types and missing values. What
potential issues can you spot?
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10178 entries, 0 to 10177
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   names       10178 non-null  object 
 1   date_x      10178 non-null  object 
 2   score       10178 non-null  float64
 3   genre       10093 non-null  object 
 4   overview    10178 non-null  object 
 5   crew        10122 non-null  object 
 6   orig_title  10178 non-null  object 
 7   status      10178 non-null  object 
 8   orig_lang   10178 non-null  object 
 9   budget_x    10178 non-null  float64
 10  revenue     10178 non-null  float64
 11  country     10178 non-null  object 
dtypes: float64(3), object(9)
memory usage: 954.3+ KB
```

```
# Describe the main characteristics of each column using .describe(). 
What can you infer from the mean, median, and distribution of 
numerical columns?
df.describe()
```

	score	budget_x	revenue
count	10178.000000	1.017800e+04	1.017800e+04
mean	63.497052	6.488238e+07	2.531401e+08
std	13.537012	5.707565e+07	2.777880e+08
min	0.000000	1.000000e+00	0.000000e+00
25%	59.000000	1.500000e+07	2.858898e+07
50%	65.000000	5.000000e+07	1.529349e+08
75%	71.000000	1.050000e+08	4.178021e+08
max	100.000000	4.600000e+08	2.923706e+09

Data Cleaning

```
#Which columns contain missing values? How would you handle them?
df.isna().sum()
```

```
names          0
date_x         0
score          0
genre         85
overview       0
crew          56
orig_title     0
status         0
```

```
orig_lang      0
budget_x       0
revenue        0
country        0
dtype: int64
```

```
#Are there any columns where data types need conversion (e.g., date, ratings)? Explain your decision.
#before ->date_x in a format of object
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10178 entries, 0 to 10177
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   names       10178 non-null  object 
 1   date_x      10178 non-null  object 
 2   score       10178 non-null  float64
 3   genre       10093 non-null  object 
 4   overview    10178 non-null  object 
 5   crew        10122 non-null  object 
 6   orig_title  10178 non-null  object 
 7   status      10178 non-null  object 
 8   orig_lang   10178 non-null  object 
 9   budget_x    10178 non-null  float64
 10  revenue     10178 non-null  float64
 11  country     10178 non-null  object 
dtypes: float64(3), object(9)
memory usage: 954.3+ KB
```

```
df["date_x"]=pd.to_datetime(df["date_x"])
```

```
#after -> now you can see the the date_x in a format of datetime
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10178 entries, 0 to 10177
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   names       10178 non-null  object        
 1   date_x      10178 non-null  datetime64[ns]
 2   score       10178 non-null  float64       
 3   genre       10093 non-null  object        
 4   overview    10178 non-null  object        
 5   crew        10122 non-null  object        
 6   orig_title  10178 non-null  object        
 7   status      10178 non-null  object        
 8   orig_lang   10178 non-null  object        
```

```
 9   budget_x    10178 non-null  float64
 10  revenue     10178 non-null  float64
 11  country     10178 non-null  object
dtypes: datetime64[ns](1), float64(3), object(8)
memory usage: 954.3+ KB
```

Univariate Analysis: Explore each column individually.

```
df.describe()
```

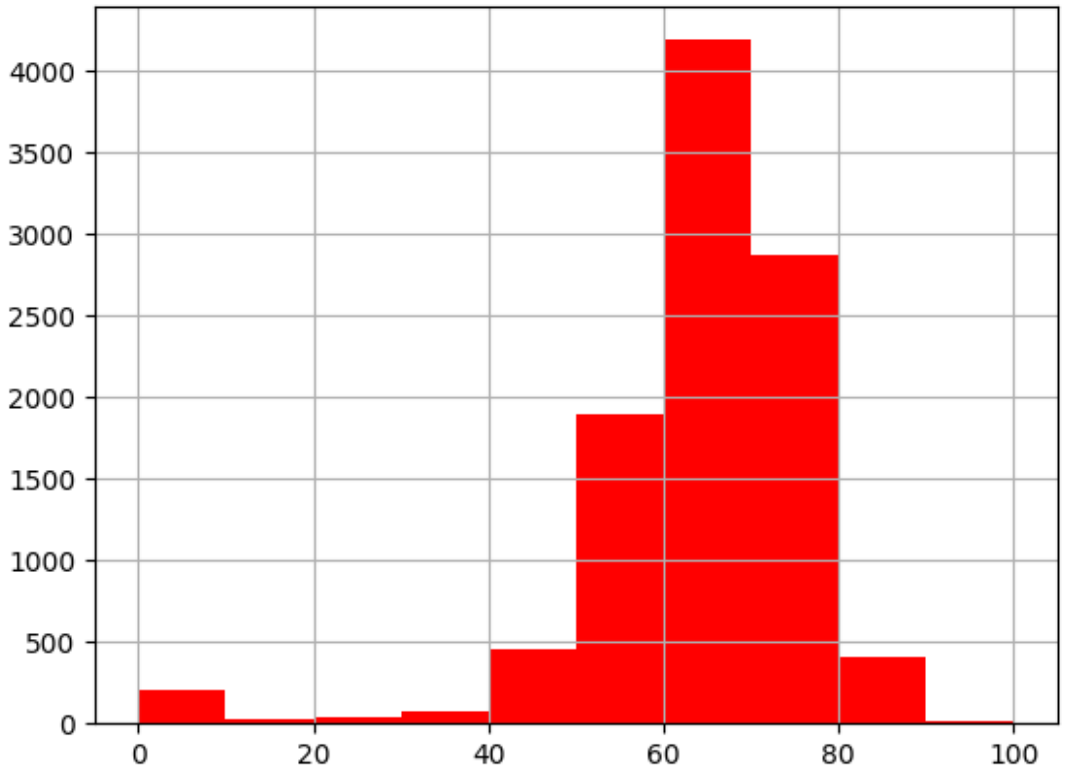
```
                                date_x          score      budget_x
revenue
count                            10178  10178.000000  1.017800e+04
1.017800e+04
mean   2008-06-15 06:16:37.445470720     63.497052  6.488238e+07
2.531401e+08
min              1903-05-15 00:00:00      0.000000  1.000000e+00
0.000000e+00
25%              2001-12-25 06:00:00     59.000000  1.500000e+07
2.858898e+07
50%              2013-05-09 00:00:00     65.000000  5.000000e+07
1.529349e+08
75%              2019-10-17 00:00:00     71.000000  1.050000e+08
4.178021e+08
max              2023-12-31 00:00:00    100.000000  4.600000e+08
2.923706e+09
std                              NaN     13.537012  5.707565e+07
2.777880e+08
```

```
df.columns
```

```
Index(['names', 'date_x', 'score', 'genre', 'overview', 'crew',
'orig_title',
       'status', 'orig_lang', 'budget_x', 'revenue', 'country'],
      dtype='object')
```

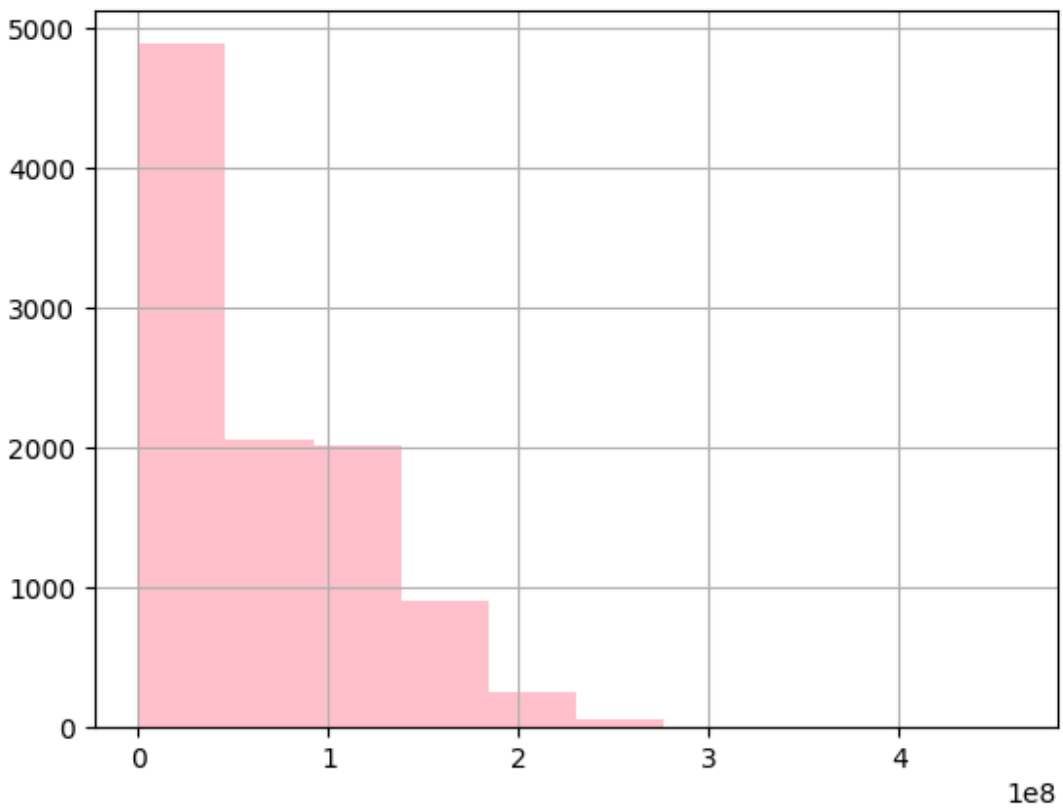
```
#check skew/outliers (for numberical columns)
df["score"].hist(color="red")   #1
```

```
<Axes: >
```



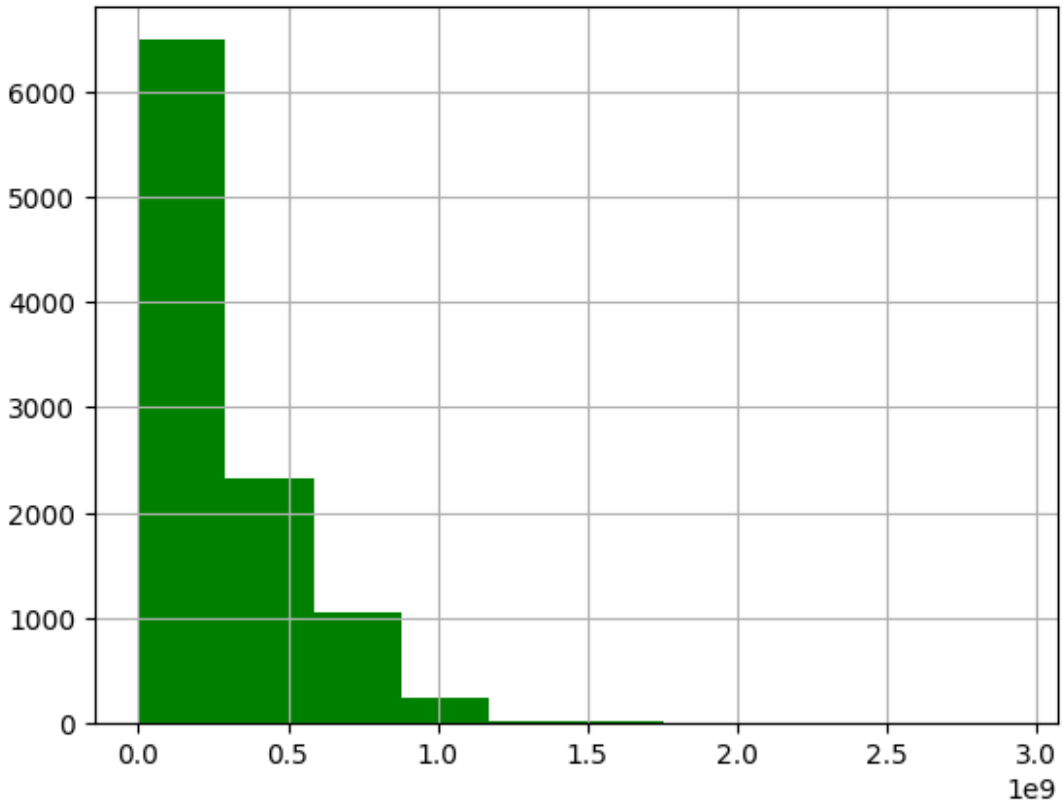

```
df["budget_x"].hist(color="pink") #2
```

```
<Axes: >
```




```
df["revenue"].hist(color="green")  #3
```

```
<Axes: >
```




```
# for categorical columns
df["genre"].value_counts()
```

```
genre
Drama                                                556
Comedy                                               373
Drama, Romance                                       268
Horror                                               260
Horror, Thriller                                     202
                                                    ...
Action, Animation, Crime, Drama                        1
Adventure, Animation, Family, Action                   1
Drama, Animation, Family, Comedy, Fantasy              1
Science Fiction, War                                   1
Action, Adventure, Science Fiction, Thriller, Horror   1
Name: count, Length: 2303, dtype: int64
```

```
df["status"].value_counts()
```

```
status
Released           10131
Post Production       31
In Production         16
Name: count, dtype: int64
```

```
df["orig_lang"].value_counts()
```

orig_lang	
English	7417
Japanese	714
Spanish, Castilian	397
Korean	388
French	285
Chinese	153
Cantonese	145
Italian	142
German	93
Russian	66
Tagalog	43
Portuguese	35
Thai	34
Norwegian	29
Hindi	26
Polish	26
Danish	23
Dutch, Flemish	22
Swedish	22
Turkish	22
Indonesian	11
Malayalam	7
Greek	6
Tamil	6
Telugu	6
Finnish	6
Persian	5
Vietnamese	4
Arabic	4
Ukrainian	4
Icelandic	3
No Language	3
Romanian	2
Kannada	2
Czech	2
Central Khmer	2
Malay	2
Latvian	2
Bengali	2
Hungarian	2
Catalan, Valencian	2
Macedonian	1
Oriya	1
Bokmål, Norwegian, Norwegian Bokmål	1
Marathi	1
Basque	1
Galician	1

```
Irish                                    1
Serbian                                  1
Gujarati                                 1
Serbo-Croatian                           1
Latin                                    1
Dzongkha                                 1
Slovak                                   1
Name: count, dtype: int64
```

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

```
country
AU    4885
US    2750
JP     538
KR     361
FR     222
GB     174
ES     153
HK     125
IT     123
MX     105
CN      93
DE      88
CA      67
RU      52
IN      43
PH      43
AR      41
BR      38
TH      30
DK      24
PL      22
TR      20
NO      16
NL      16
CO      14
TW      13
ID      12
IE      11
CL       9
SE       9
BE       7
PE       7
FI       6
GR       6
CH       5
SU       5
UA       4
SG       4
```

```
VN       3
HU       3
ZA       3
IR       2
PR       2
CZ       2
GT       2
IS       2
SK       2
UY       2
AT       2
MY       2
LV       1
KH       1
PT       1
XC       1
IL       1
MU       1
PY       1
DO       1
BO       1
BY       1
Name: count, dtype: int64
```

```
#date columns
#basic info
print(df["date_x"].min(),"\n",df["date_x"].max())
```

```
1903-05-15 00:00:00
 2023-12-31 00:00:00
```

```
#movies per year
df["date_x"].dt.year.value_counts().sort_index()
```

```
date_x
1903      1
1907      1
1915      2
1920      1
1923      3
       ...
2019    470
2020    449
2021    627
2022    954
2023    403
Name: count, Length: 99, dtype: int64
```

```
#string length
df["overview"].str.len()
```

```
0        458
1        272
2        252
3        230
4        403
        ...
10173    406
10174    147
10175    288
10176    218
10177    201
Name: overview, Length: 10178, dtype: int64
```

```
df["crew"].isna().sum()
```

```
np.int64(56)
```

```
df.columns
```

```
Index(['names', 'date_x', 'score', 'genre', 'overview', 'crew',
'orig_title',
       'status', 'orig_lang', 'budget_x', 'revenue', 'country'],
      dtype='object')
```

```
# What is the distribution of movie runtimes? Plot a histogram and
describe its shape.
print("There is no runtimes column in the dataset")
```

```
There is no runtimes column in the dataset
```

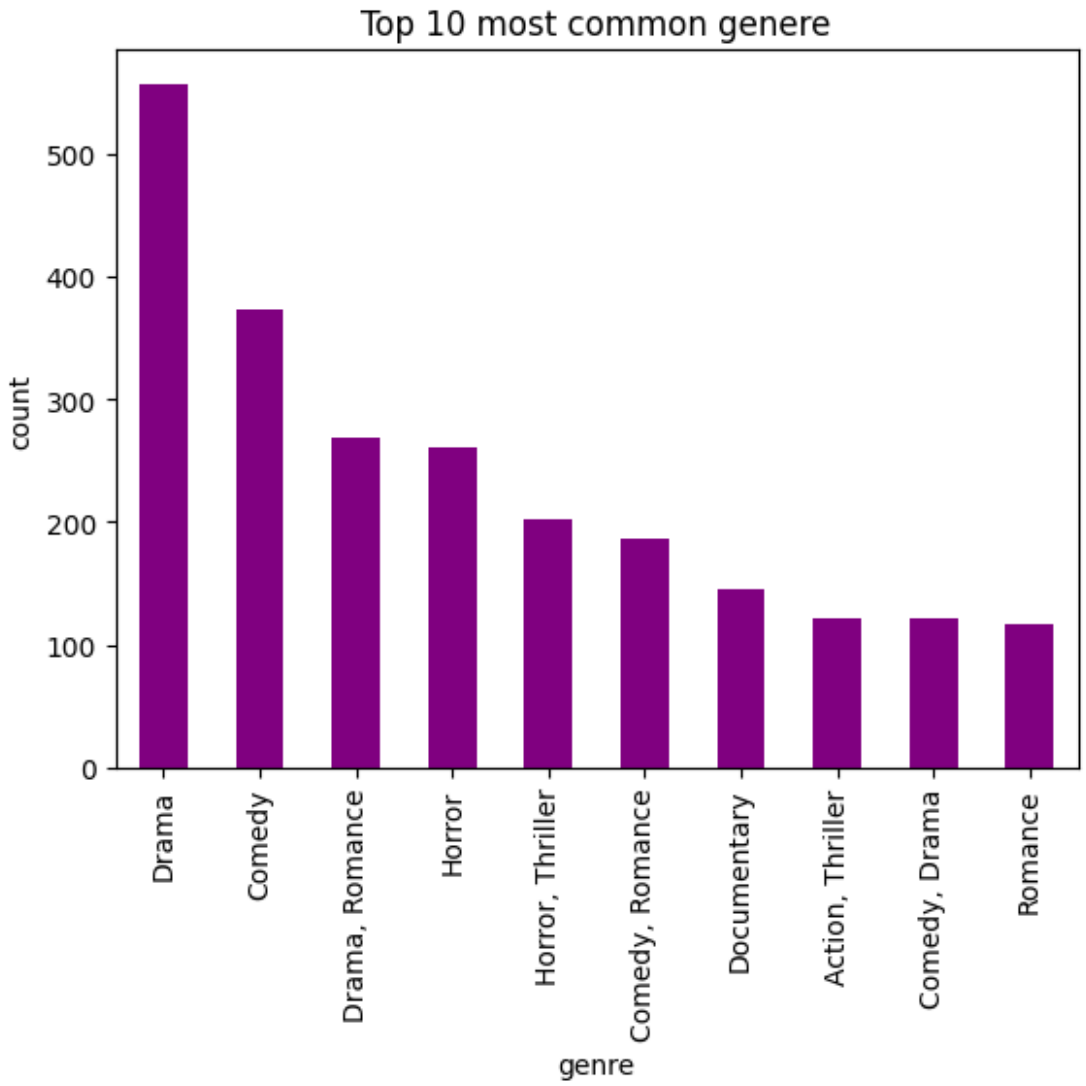
```
# What are the most common genres in the dataset? Use a bar chart to
show their distribution.
import matplotlib.pyplot as plt
df["genre"]
```

```
0                                            Drama, Action
1                        Science Fiction, Adventure, Action
2              Animation, Adventure, Family, Fantasy, Comedy
3              Animation, Comedy, Family, Adventure, Fantasy
4                                                    Action
                                ...
10173                                                 Drama
10174                                                Action
10175                              Drama, Thriller, Romance
10176    Action, Adventure, Science Fiction, Thriller, ...
10177                            Animation, Family, Fantasy
Name: genre, Length: 10178, dtype: object
```

```
genre_count=df["genre"].value_counts()
genre_count.head(10).plot(kind="bar",color="purple")
```

```
plt.title("Top 10 most common genere")
plt.ylabel("count")
```

```
Text(0, 0.5, 'count')
```



Bivariate Analysis: Explore relationships between two variables.

```
# Is there a relationship between a movie's runtime and its rating?
Plot a scatter plot and describe any observed trend.
if 'runtime' in df.columns:
    plt.figure(figsize=(8, 6))
    sns.scatterplot(x='runtime', y='score', data=data)
```

```
    plt.title('Runtime vs. Score')
    plt.xlabel('Runtime (minutes)')
    plt.ylabel('Score')
    plt.show()
else:
    print("Runtime column not found in the dataset.")
```

```
Runtime column not found in the dataset.
```

```
# How do ratings vary by genre? Use a boxplot to visualize the
differences in ratings across genres.
df.columns
# we don't have rating columns so we use columns
```

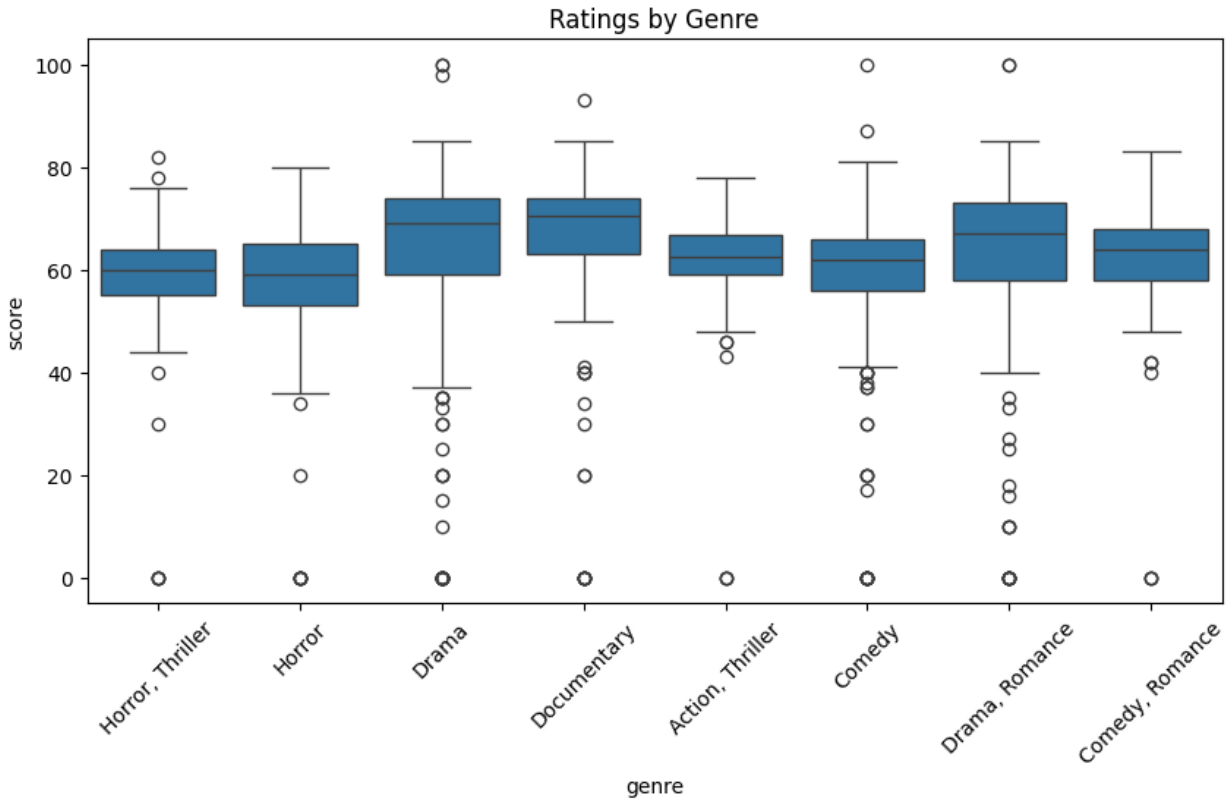
```
Index(['names', 'date_x', 'score', 'genre', 'overview', 'crew',
'orig_title',
       'status', 'orig_lang', 'budget_x', 'revenue', 'country'],
      dtype='object')
```

```
genre_df = df[['genre', 'score']].dropna()
genre_df['genre'] = genre_df['genre'].str.split(', ')
genre_df = genre_df.explode('genre') #eplode is to convert spilt into
rows
```

```
top_genres = genre_df['genre'].value_counts().head(8).index
genre_df = genre_df[genre_df['genre'].isin(top_genres)]
genre_df.head(10)
```

```
                genre  score
10  Horror, Thriller   58.0
11            Horror   55.0
17  Horror, Thriller   65.0
22             Drama   74.0
25  Horror, Thriller   65.0
28       Documentary   58.0
34            Horror   60.0
36  Horror, Thriller   63.0
37  Action, Thriller   54.0
44             Drama   81.0
```

```
import seaborn as sns
plt.figure(figsize=(10, 5))
sns.boxplot(x='genre', y='score', data=genre_df)
plt.title('Ratings by Genre')
plt.xticks(rotation=45)
plt.show()
```




```
# Correlation between votes and ratings (if available)
if 'votes' in df.columns:
    plt.figure(figsize=(8, 6))
    sns.scatterplot(x='votes', y='score', data=df)
    plt.title('Votes vs. Score')
    plt.xlabel('Number of Votes')
    plt.ylabel('Score')
    plt.show()
    print("Correlation Coefficient:", df['votes'].corr(df['score']))
else:
    print("Votes column not found in the dataset.")
```

```
Votes column not found in the dataset.
```

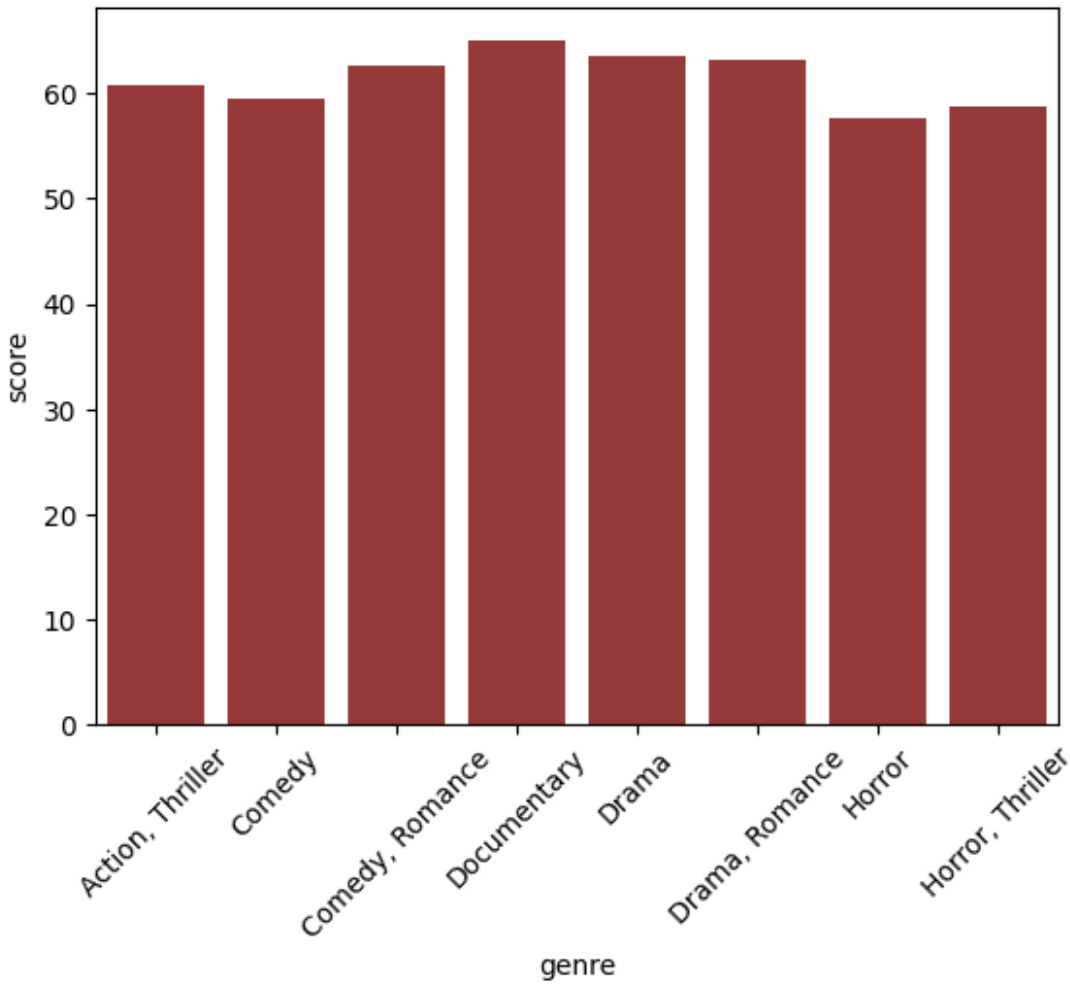
Genre-Specific Analysis

```
#Which genre has the highest average rating?
#Calculate the average rating for each genre and plot the results.
import pandas as pd
gd=genre_df.groupby("genre").mean()
gd
```

```
                        score
genre
Action, Thriller    60.803279
Comedy              59.530831
Comedy, Romance     62.689840
Documentary         64.952055
Drama               63.627698
Drama, Romance      63.164179
Horror              57.657692
Horror, Thriller    58.732673
```

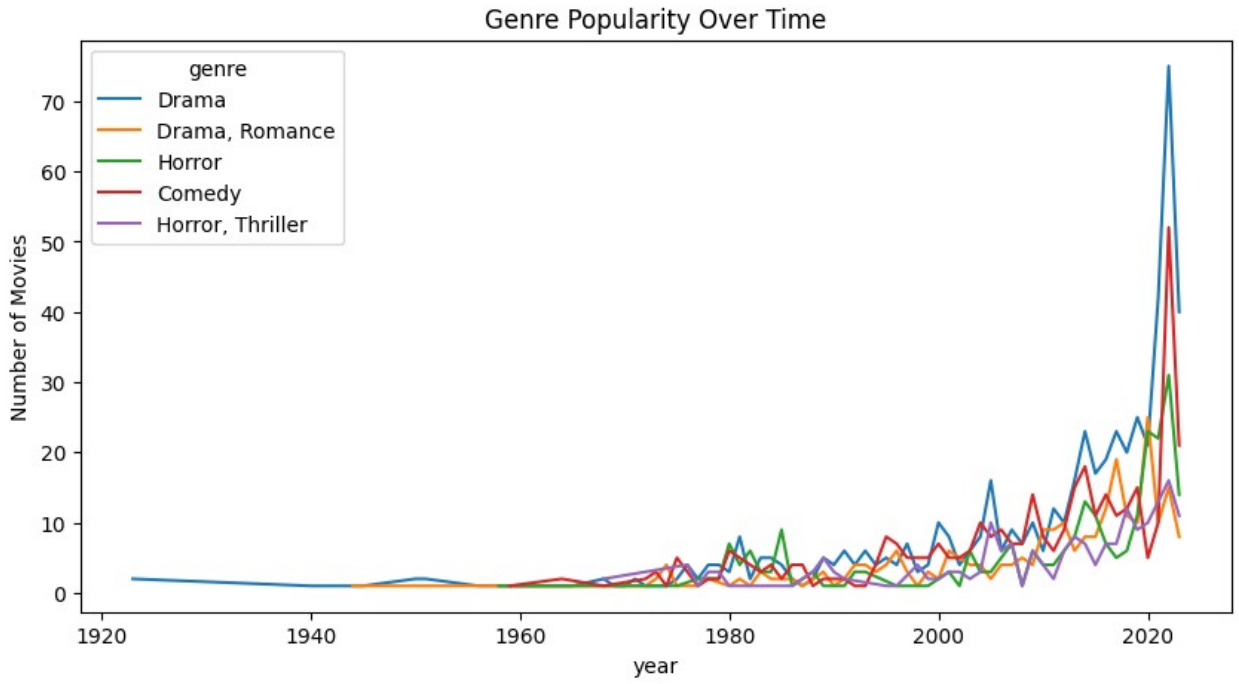
```
a=sns.barplot(data=gd,x="genre",y="score",color="brown")
plt.xticks(rotation=45)
a
```

```
<Axes: xlabel='genre', ylabel='score'>
```



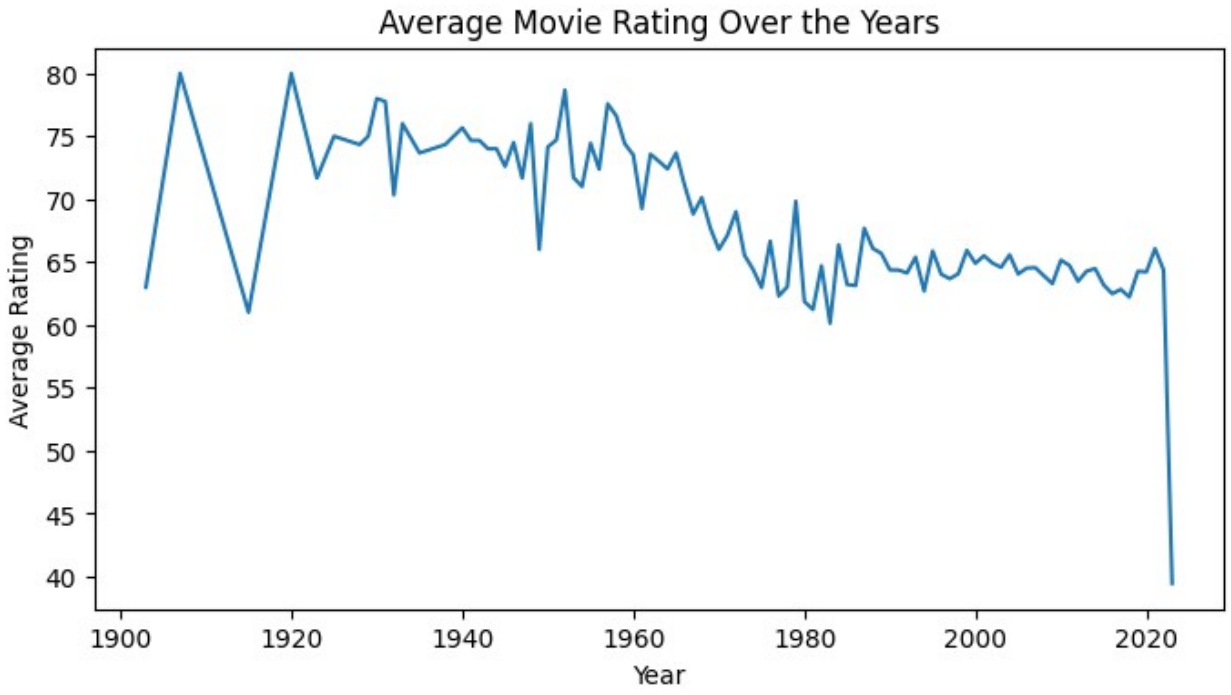

```
#How does the popularity of genres vary over time? Plot the number of
movies released per genre each year.
```

```
genre_year_df = df[["genre","date_x"]]
genre_year_df = genre_year_df.dropna()
genre_year_df['year'] = genre_year_df['date_x'].dt.year
genre_year_df['genre'] = genre_year_df['genre'].str.split(', ')
genre_year_df = genre_year_df.explode('genre')
genre_year_count = (
    genre_year_df
    .groupby(['year', 'genre'])
    .size()
    .reset_index(name='movie_count')
)
top_genres = genre_year_df['genre'].value_counts().head(5).index
genre_year_count =
genre_year_count[genre_year_count['genre'].isin(top_genres)]
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10,5))
sns.lineplot(
    data=genre_year_count,
    x='year',
    y='movie_count',
    hue='genre'
)
plt.title('Genre Popularity Over Time')
plt.ylabel('Number of Movies')
plt.show()
```



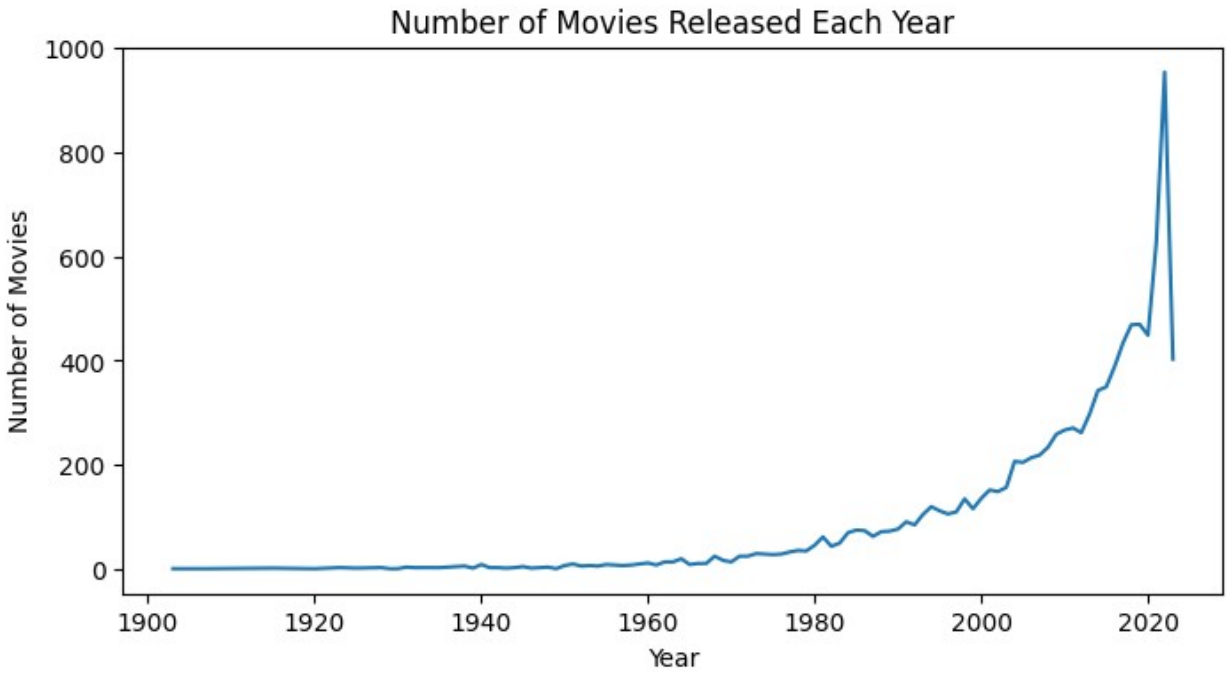
Year and Trend Analysis

```
rating_year_df = df[['date_x', 'score']]
rating_year_df = rating_year_df.dropna()
rating_year_df['year'] = rating_year_df['date_x'].dt.year
avg_rating_year = rating_year_df.groupby('year')['score'].mean()
import matplotlib.pyplot as plt

plt.figure(figsize=(8,4))
avg_rating_year.plot(kind='line')
plt.title('Average Movie Rating Over the Years')
plt.xlabel('Year')
plt.ylabel('Average Rating')
plt.show()
```




```
#Which years had the highest and lowest number of movie releases? Plot
the number of movies released each year.
year_df = df[['date_x']]
year_df = year_df.dropna()
year_df['year'] = year_df['date_x'].dt.year
movies_per_year = year_df['year'].value_counts().sort_index()
import matplotlib.pyplot as plt

plt.figure(figsize=(8,4))
movies_per_year.plot(kind='line')
plt.title('Number of Movies Released Each Year')
```

```
plt.xlabel('Year')
plt.ylabel('Number of Movies')
plt.show()
movies_per_year.idxmax(), movies_per_year.max()
```

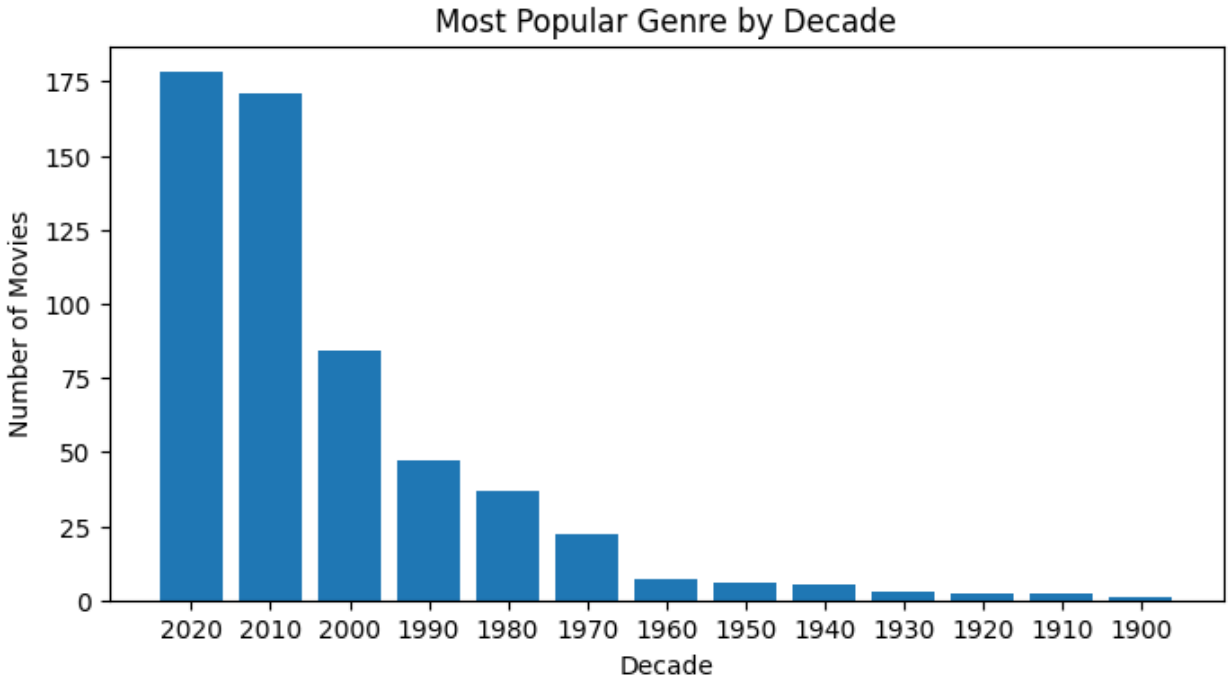



```
(np.int32(2022), np.int64(954))
```

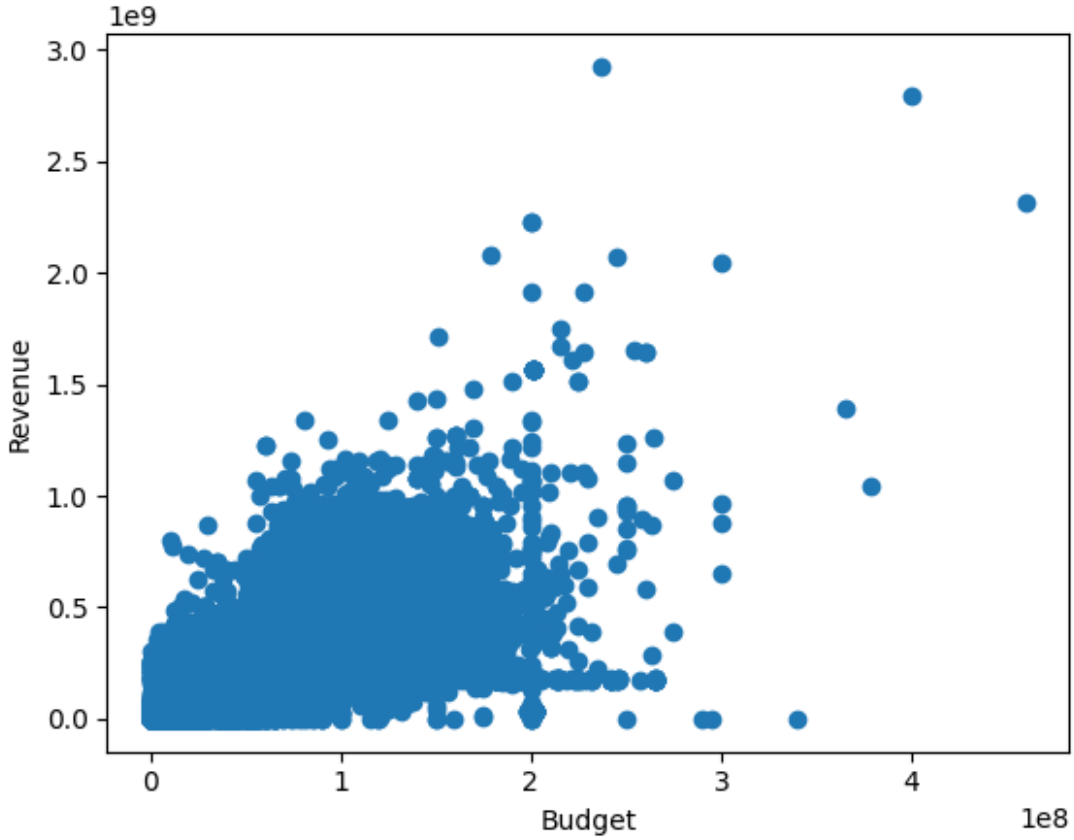
movies_per_year.idxmax(), movies_per_year.max()

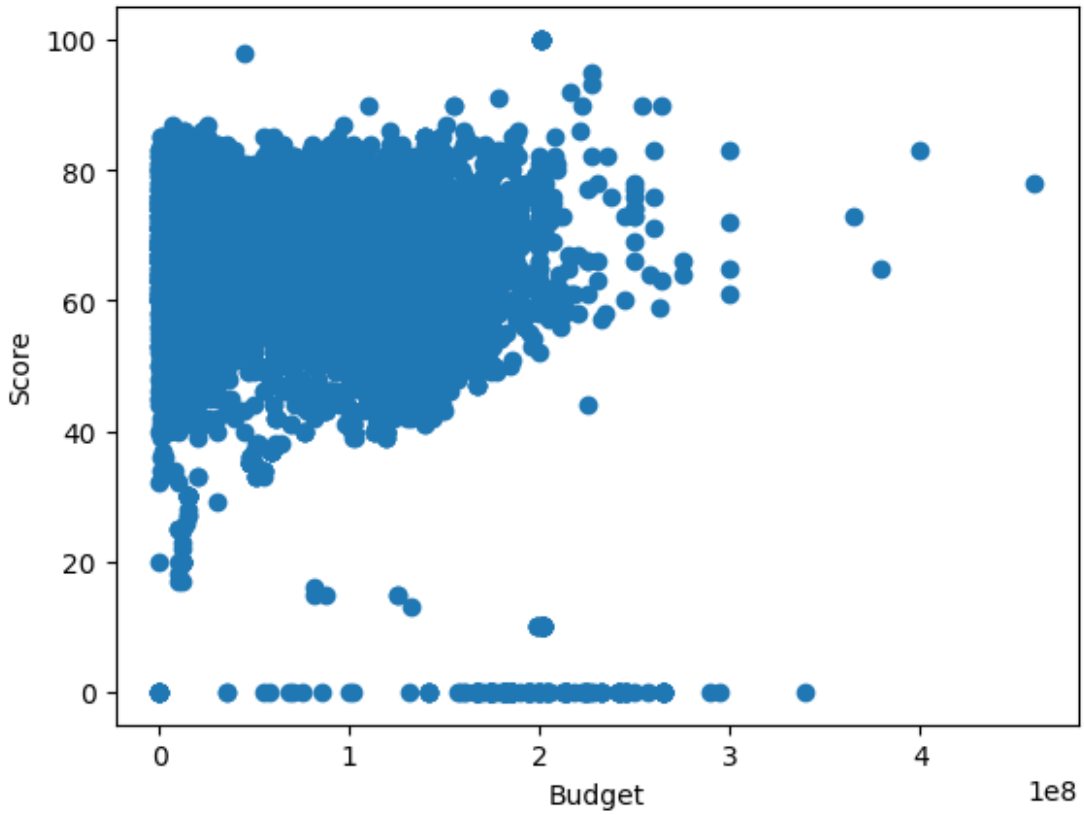
```
# Which genres are most popular in each decade? Create a bar plot
showing the most frequent genres by decade.
decade_df = df[['date_x', 'genre']]
decade_df = decade_df.dropna()
decade_df['year'] = decade_df['date_x'].dt.year
decade_df['decade'] = (decade_df['year'] // 10) * 10
decade_df['genre'] = decade_df['genre'].str.split(', ')
decade_df = decade_df.explode('genre')
decade_genre_count = (
    decade_df
    .groupby(['decade', 'genre'])
    .size()
    .reset_index(name='movie_count')
)
top_genre_decade = decade_genre_count.sort_values(
```

```
    ['decade', 'movie_count'], ascending=False
).groupby('decade').head(1)
import matplotlib.pyplot as plt

plt.figure(figsize=(8,4))
plt.bar(top_genre_decade['decade'].astype(str),
        top_genre_decade['movie_count'])
plt.xlabel('Decade')
plt.ylabel('Number of Movies')
plt.title('Most Popular Genre by Decade')
plt.show()
```




```
# Plot a heatmap or pairplot to examine relationships between budget, revenue, scores.
num_df = df[['budget_x', 'revenue', 'score']].dropna()
plt.scatter(num_df['budget_x'], num_df['revenue'])
plt.xlabel('Budget')
plt.ylabel('Revenue')
plt.show()
plt.scatter(num_df['budget_x'], num_df['score'])
plt.xlabel('Budget')
plt.ylabel('Score')
plt.show()
num_df.corr()
```


1e9
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Revenue
0
1
2
3
4
Budget
1e8




```
          budget_x   revenue     score
budget_x   1.00000  0.673830 -0.235470
revenue    0.67383  1.000000  0.096533
score     -0.23547  0.096533  1.000000
```

```
#Are there specific genres or release years with higher-rated movies?
Group by genre and year, then analyze the average rating.
rating_df = df[['genre', 'date_x', 'score']].dropna()
rating_df['year'] = rating_df['date_x'].dt.year
rating_df['genre'] = rating_df['genre'].str.split(', ')
rating_df = rating_df.explode('genre')
avg_rating = (
    rating_df
    .groupby(['year', 'genre'])['score']
    .mean()
    .reset_index()
)
```

Insights and Summary

```
print("""
 9. Insights and Summary
 Q1. Three Major Insights from the Analysis
```

```
 Insight 1: Genre Popularity Changes Over Time
Drama and Comedy appear most frequently
Different decades show different dominant genres
Audience preferences change with time

 Insight 2: Budget and Revenue are Positively Related
Higher-budget movies generally earn higher revenue
Scatter plots show a clear upward trend
Big budget does not always guarantee high ratings

 Insight 3: Ratings Vary by Genre and Year
Some genres consistently have higher average ratings
Average ratings fluctuate across years
Movie quality and audience response are not constant over time

 Q2. Additional Questions & Data for Deeper Analysis
 More Questions to Explore
Do higher-rated movies always make more revenue?
Which directors or actors are associated with high ratings?
Are movies getting longer or shorter over time?
Do certain countries produce higher-rated movies?

 Additional Data That Would Help
Runtime of movies
Number of votes (popularity measure)
Director and cast details
Marketing or promotion budget
Streaming vs theatrical release data""")


 9. Insights and Summary
 Q1. Three Major Insights from the Analysis
 Insight 1: Genre Popularity Changes Over Time
Drama and Comedy appear most frequently
Different decades show different dominant genres
Audience preferences change with time

 Insight 2: Budget and Revenue are Positively Related
Higher-budget movies generally earn higher revenue
Scatter plots show a clear upward trend
Big budget does not always guarantee high ratings

 Insight 3: Ratings Vary by Genre and Year
Some genres consistently have higher average ratings
Average ratings fluctuate across years
Movie quality and audience response are not constant over time

 Q2. Additional Questions & Data for Deeper Analysis
 More Questions to Explore
Do higher-rated movies always make more revenue?
```

Which directors or actors are associated with high ratings?
Are movies getting longer or shorter over time?
Do certain countries produce higher-rated movies?

 Additional Data That Would Help
Runtime of movies
Number of votes (popularity measure)
Director and cast details
Marketing or promotion budget
Streaming vs theatrical release data