In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df= pd.read_csv("mymoviedb.csv",lineterminator = '\n')
```

In [3]:
```
df.head()
```

Out[3]:

	Release_Date	Title	Overview	Popularity	Vote_Count	Vote_Average	Original_Language	Genre	Poster_Url
0	2021-12-15	Spider-Man: No Way Home	Peter Parker is unmasked and no longer able to...	5083.954	8940	8.3	en	Action, Adventure, Science Fiction	https://image.tmdb.org/t/p/original/1g0dhYtq4...
1	2022-03-01	The Batman	In his second year of fighting crime, Batman u...	3627.658	1151	8.1	en	Crime, Mystery, Thriller	https://image.tmdb.org/t/p/original/74xTEgt7R3...
2	2022-02-25	No Exit	Stranded at a rest stop in the mountains durin...	2618.087	122	6.3	en	Thriller	https://image.tmdb.org/t/p/original/vDHsLnOWKl...
3	2021-11-24	Encanto	The tale of an extraordinary family, the Madri...	2402.201	5076	7.7	en	Animation, Comedy, Family, Fantasy	https://image.tmdb.org/t/p/original/4j0PNHkMr5...
4	2021-12-22	The King's Man	As a collection of history's worst tyrants and...	1895.511	1793	7.0	en	Action, Adventure, Thriller, War	https://image.tmdb.org/t/p/original/aq4Pwv5Xeu...

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9827 entries, 0 to 9826
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Release_Date  9827 non-null   object
 1   Title         9827 non-null   object
 2   Overview      9827 non-null   object
 3   Popularity    9827 non-null   float64
 4   Vote_Count    9827 non-null   int64
 5   Vote_Average  9827 non-null   float64
 6   Original_Language  9827 non-null   object
 7   Genre         9827 non-null   object
 8   Poster_Url    9827 non-null   object
dtypes: float64(3), int64(1), object(5)
memory usage: 691.1 KB
```

In [5]:
```
df.duplicated()
```

Out[5]:
```
0       False
1       False
2       False
3       False
4       False
        ...
9822    False
9823    False
9824    False
9825    False
9826    False
Length: 9827, dtype: bool
```

In [6]:
```
df.describe()
```

Out[6]:

	Popularity	Vote_Count	Vote_Average
count	9827.000000	9827.000000	9827.000000
mean	40.326088	1392.805536	6.439534
std	108.873998	2011.208627	1.129759
min	13.354000	0.000000	0.000000
25%	16.128500	146.000000	5.900000
50%	21.199000	444.000000	6.500000
75%	35.191500	1576.000000	7.100000
max	5083.954000	31077.000000	10.000000

In [7]:
```
## Exploratory summary

## We have dataframe consisting of 9827 rows and 9 columns.
## our dataset looks a bit big with no nans but duplicated values.
##Release_Date column needs to be casted into date and to extract only.
## Overview and Poster_Url columns are useless to us.
## Genre is noticeable column in popularity column
## Vote_Average better be categorized for proper analysis.
## Genre column has commas separated values and white spaces that need to be handled and casted into category.
```

In [8]:
```
df["Release_Date"] = pd.to_datetime(df["Release_Date"])
print(df["Release_Date"].dtype)
```

```
datetime64[ns]
```

In [9]:
```
df["Release_Date"] = df["Release_Date"].dt.year
df["Release_Date"].dtypes
```

Out[9]:
```
dtypes(int32)
```

In [10]:
```
df.head()
```

Out[10]:

	Release_Date	Title	Overview	Popularity	Vote_Count	Vote_Average	Original_Language	Genre	Poster_Url
0	2021	Spider-Man: No Way Home	Peter Parker is unmasked and no longer able to...	5083.954	8940	8.3	en	Action, Adventure, Science Fiction	https://image.tmdb.org/t/p/original/1g0dhYtq4...
1	2022	The Batman	In his second year of fighting crime, Batman u...	3627.658	1151	8.1	en	Crime, Mystery, Thriller	https://image.tmdb.org/t/p/original/74xTEgt7R3...
2	2022	No Exit	Stranded at a rest stop in the mountains durin...	2618.087	122	6.3	en	Thriller	https://image.tmdb.org/t/p/original/vDHsLnOWKl...
3	2021	Encanto	The tale of an extraordinary family, the Madri...	2402.201	5076	7.7	en	Animation, Comedy, Family, Fantasy	https://image.tmdb.org/t/p/original/4j0PNHkMr5...
4	2021	The King's Man	As a collection of history's worst tyrants and...	1895.511	1793	7.0	en	Action, Adventure, Thriller, War	https://image.tmdb.org/t/p/original/aq4Pwv5Xeu...

dropping the columns

In [11]:
```
cols=['Overview','Original_Language','Poster_Url']
df.drop(cols, axis= 1 ,inplace=True)
df.columns
```

Out[11]:
```
Index(['Release_Date', 'Title', 'Popularity', 'Vote_Count', 'Vote_Average',
       'Genre'],
      dtype='object')
```

In [12]:
```
df.head()
```

Out[12]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	8.3	Action, Adventure, Science Fiction
1	2022	The Batman	3627.658	1151	8.1	Crime, Mystery, Thriller
2	2022	No Exit	2618.087	122	6.3	Thriller
3	2021	Encanto	2402.201	5076	7.7	Animation, Comedy, Family, Fantasy
4	2021	The King's Man	1895.511	1793	7.0	Action, Adventure, Thriller, War

categorizing vote_average column

we could cut the vote_average values and make 4 categories:popular, average ,below average, not popular to describe it more using categorize_col() function provided above

In [13]:
```
def categorize_col(df, col, labels):
    edges = df[col].describe()['min'],
            df[col].describe()['25%'],
            df[col].describe()['50%'],
            df[col].describe()['75%'],
            df[col].describe()['max']

    df[col] = pd.cut(df[col], edges, labels = labels, duplicates='drop')
    return df
```

In [14]:
```
labels = ['not_popular', 'below_average', 'average','popular']
df = categorize_col(df, col='Vote_Average', labels=labels)
df['Vote_Average'].unique()
```

Out[14]:
```
['popular', 'below_average', 'average', 'not_popular', NaN]
Categories (4, object): ['not_popular' < 'below_average' < 'average' < 'popular']
```

In [15]:
```
df.head()
```

Out[15]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action, Adventure, Science Fiction
1	2022	The Batman	3627.658	1151	popular	Crime, Mystery, Thriller
2	2022	No Exit	2618.087	122	below_average	Thriller
3	2021	Encanto	2402.201	5076	popular	Animation, Comedy, Family, Fantasy
4	2021	The King's Man	1895.511	1793	average	Action, Adventure, Thriller, War

In [16]:
```
df['Vote_Average'].value_counts()
```

```
Vote_Average
not_popular    2451
popular        2450
average        2412
below_average  2412
Name: count, dtype: int64
```

In [17]:
```
df.dropna(inplace=True)
```

In [18]:
```
df.isna().sum()
```

Out[18]:
```
Release_Date    0
Title           0
Popularity      0
Vote_Count      0
Vote_Average    0
Genre           0
dtype: int64
```

In [19]:
```
df.head()
```

Out[19]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action, Adventure, Science Fiction
1	2022	The Batman	3627.658	1151	popular	Crime, Mystery, Thriller
2	2022	No Exit	2618.087	122	below_average	Thriller
3	2021	Encanto	2402.201	5076	popular	Animation, Comedy, Family, Fantasy
4	2021	The King's Man	1895.511	1793	average	Action, Adventure, Thriller, War

we have split genres into list and then explode our dataframe to have only one genre per row for movie

In [20]:
```
df["Genre"]=df["Genre"].str.split(", ")
df = df.explode("Genre").reset_index(drop = True)
df.head()
```

Out[20]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action
1	2021	Spider-Man: No Way Home	5083.954	8940	popular	Adventure
2	2021	Spider-Man: No Way Home	5083.954	8940	popular	Science Fiction
3	2022	The Batman	3627.658	1151	popular	Crime
4	2022	The Batman	3627.658	1151	popular	Mystery

In [21]:
```
# casting column into category

df["Genre"] = df["Genre"].astype("category")
df["Genre"].dtypes
```

Out[21]:
```
CategoricalDtype(categories=['Action', 'Adventure', 'Animation', 'Comedy', 'Crime',
                  'Documentary', 'Drama', 'Family', 'Fantasy', 'History',
                  'Horror', 'Music', 'Mystery', 'Romance', 'Science Fiction',
                  'TV Movie', 'Thriller', 'War', 'Western'],
                 , ordered=False, categories_dtype=object)
```

In [22]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25552 entries, 0 to 25551
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Release_Date  25552 non-null  int32
 1   Title         25552 non-null  object
 2   Popularity    25552 non-null  float64
 3   Vote_Count    25552 non-null  int64
 4   Vote_Average  25552 non-null  category
 5   Genre         25552 non-null  category
dtypes: category(2), float64(1), int32(1), int64(1), object(1)
memory usage: 764.8 KB
```

In [23]:
```
df.nunique()
```

Out[23]:
```
Release_Date    100
Title          9415
Popularity     8584
Vote_Count     3267
Vote_Average      4
Genre            19
dtype: int64
```

In [24]:
```
df.head()
```

Out[24]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action
1	2021	Spider-Man: No Way Home	5083.954	8940	popular	Adventure
2	2021	Spider-Man: No Way Home	5083.954	8940	popular	Science Fiction
3	2022	The Batman	3627.658	1151	popular	Crime
4	2022	The Batman	3627.658	1151	popular	Mystery

Data visualization

In [25]:
```
sns.set_style("whitegrid")
```

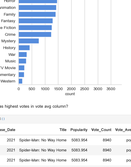
which is the most frequent genre of movies released on netflix?

In [26]:
```
df['Genre'].describe()
```

Out[26]:
```
count    25552
unique      19
top      Drama
freq      3715
Name: Genre, dtype: object
```

In [27]:
```
sns.catplot(y="Genre",data = df, kind="count",order=df['Genre'].value_counts().index,color="#42B7D1")
plt.title('Genre column distribution')
plt.show()
```

```
C:\Users\Ignatenko\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\categorical.py:641: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  grouped_ests = ests.groupby(grouped_vals)
C:\Users\Ignatenko\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\categorical.py:641: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  grouped_ests = ests.groupby(grouped_vals)
```



which has highest votes in vote avg column?

In [28]:
```
df.head()
```

Out[28]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action
1	2021	Spider-Man: No Way Home	5083.954	8940	popular	Adventure
2	2021	Spider-Man: No Way Home	5083.954	8940	popular	Science Fiction
3	2022	The Batman	3627.658	1151	popular	Crime
4	2022	The Batman	3627.658	1151	popular	Mystery

In [29]:
```
sns.catplot(y="Vote_Average",data = df, kind="count",order=df["Vote_Average"].value_counts().index,color="#42B7D1")
plt.title('Vote distribution')
plt.show()
```

```
C:\Users\Ignatenko\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\categorical.py:641: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  grouped_ests = ests.groupby(grouped_vals)
C:\Users\Ignatenko\AppData\Local\Programs\Python\Python39\lib\site-packages\seaborn\categorical.py:641: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  grouped_ests = ests.groupby(grouped_vals)
```

Which movie got the highest popularity? what's its genre?

In [30]:
```
df.head()
```

Out[30]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action
1	2021	Spider-Man: No Way Home	5083.954	8940	popular	Adventure
2	2021	Spider-Man: No Way Home	5083.954	8940	popular	Science Fiction
3	2022	The Batman	3627.658	1151	popular	Crime
4	2022	The Batman	3627.658	1151	popular	Mystery

In [31]:
```
df[df['Popularity'] == df['Popularity'].max()]
```

Out[31]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action
1	2021	Spider-Man: No Way Home	5083.954	8940	popular	Adventure
2	2021	Spider-Man: No Way Home	5083.954	8940	popular	Science Fiction

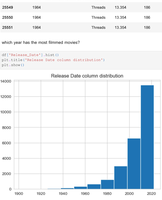
In [32]:
```
df[df['Popularity'] == df['Popularity'].min()]
```

Out[32]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
25546	2021	The United States vs. Billie Holiday	13.354	152	average	Music
25547	2021	The United States vs. Billie Holiday	13.354	152	average	Drama
25548	2021	The United States vs. Billie Holiday	13.354	152	average	History
25549	1984	Threads	13.354	186	popular	War
25550	1984	Threads	13.354	186	popular	Drama
25551	1984	Threads	13.354	186	popular	Science Fiction

which year has the most filmed movies?

In [33]:
```
df["Release_Date"].hist()
plt.title('Release Date column distribution')
plt.show()
```



Conclusion:-

Q1 what is the most frequent genre in the dataset? ->Genre is the most frequent genre in our data set and has appeared more than 14% of the times among 19 other genres.

Q2 What genre has highest votes? ->We have 25.9% of our dataset with popular vote (9520 rows).drama again gets the highest popularity among fans by being having more than 19.5% movies

Q3 what movie got the highest popularity ? whats its genre? ->Spider-man: no way home ,has the highest rate in our dataset and it has genres of Action, Adventure, and science fiction.

Q4 what movie got the lowest popularity ? whats its genre? ->The united states , thread has the lowest rate in our dataset and it has genres of music, drama , war ,sci-fi.

Q5 which year has the most filmed movies? ->Year 2020 has the highest filming rate in our dataset.