Project Title: Netflix Data Analysis: Unveiling Movie Trend and User Preferences

Dive into a full data science project where we analyze Netflix data to uncover movie trends, understand user preferences, and develop strategies for personalized recommendations. This video guides you through data cleaning, preprocessing, and visualization, culminating in insights that can enhance the Netflix user experience.

In [6]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [7]:
```python
data=pd.read_csv("mymoviedb.csv", lineterminator = "\n")
```

In [8]:
```python
data
```

Out[8]:

	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/1g0dhYtq4i...
1	2022-03-01	The Batman	In his second year of fighting crime, Batman u...	3827.658	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...
...	...	...	...	...	...	...	...	...	...
9822	1973-10-15	Badlands	A dramatization of the Starkweather-Fugate kil...	13.357	896	7.6	en	Drama, Crime	https://image.tmdb.org/t/p/original/z81rBzHNgi...
9823	2020-10-01	Violent Delights	A female vampire falls in love with a man she ...	13.356	8	3.5	es	Horror	https://image.tmdb.org/t/p/original/4b6HY7rud6...
9824	2016-05-06	The Offering	When young and successful reporter Jamie finds...	13.355	94	5.0	en	Mystery, Thriller, Horror	https://image.tmdb.org/t/p/original/h4uMM1wOhz...
9825	2021-03-31	The United States vs. Billie Holiday	Billie Holiday spent much of her career being ...	13.354	152	6.7	en	Music, Drama, History	https://image.tmdb.org/t/p/original/vEzkxuE2sJ...
9826	1984-09-23	Threads	Documentary style account of a nuclear holocau...	13.354	186	7.8	en	War, Drama, Science Fiction	https://image.tmdb.org/t/p/original/lBhU4U9Eeh...

9827 rows × 9 columns

In [9]:
```python
data.head(5)
```

Out[9]:

	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/1g0dhYtq4i...
1	2022-03-01	The Batman	In his second year of fighting crime, Batman u...	3827.658	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 [10]:
```python
data.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(2), int64(1), object(6)
memory usage: 691.1+ KB
```

In [11]:
```python
data["Genre"].head()
```

Out[11]:
```
0    Action, Adventure, Science Fiction
1              Crime, Mystery, Thriller
2                              Thriller
3    Animation, Comedy, Family, Fantasy
4      Action, Adventure, Thriller, War
Name: Genre, dtype: object
```

In [12]:
```python
# to find the duplicate data in the column
data.duplicated().sum()
```

Out[12]: np.int64(0)

In [13]:
```python
data.describe()
```

Out[13]:

	Popularity	Vote_Count	Vote_Average
count	9827.000000	9827.000000	9827.000000
mean	40.326088	1392.805536	6.439534
std	108.873998	2611.206907	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	1376.000000	7.100000
max	5083.954000	31077.000000	10.000000

Expolatory Summary

we have a dataframe consist of 9827 rows and 9 columns.

our dataset looks bit tidy with no NaNs nor duplicate values.

Release_Date column needs to be casted in date time and to extract only the year value.

Overview, Original_Language and Poster-Url woudn't so useful during analysis, so we'll drop them.

there is noicable outliers in popularity column.

Vote_average better be catagorized for proper analysis.

Genre column has coma separated values and white spaces that needs to be handled and casted into catagory.Exploration summary.

In [14]:
```python
data.head(2)
```

Out[14]:

	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/1g0dhYtq4i...
1	2022-03-01	The Batman	In his second year of fighting crime, Batman u...	3827.658	1151	8.1	en	Crime, Mystery, Thriller	https://image.tmdb.org/t/p/original/74xTEgt7R3...

In [15]:
```python
# changing it into date time format
data["Release_Date"] = pd.to_datetime(data["Release_Date"])

print(data["Release_Date"].dtype)
```

```
datetime64[ns]
```

In [16]:
```python
# we want only year
data["Release_Date"] = data["Release_Date"].dt.year

data["Release_Date"].dtype
```

Out[16]: dtype('int32')

In [17]:
```python
data.head(2)
```

Out[17]:

	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/1g0dhYtq4i...
1	2022	The Batman	In his second year of fighting crime, Batman u...	3827.658	1151	8.1	en	Crime, Mystery, Thriller	https://image.tmdb.org/t/p/original/74xTEgt7R3...

In [18]:
```python
# dropping the columns
```

In [19]:
```python
cols =["Overview","Original_Language","Poster_Url"]
```

In [20]:
```python
data.drop(cols, axis=1,inplace= True)
data.columns
```

Out[20]:
```
Index(['Release_Date', 'Title', 'Popularity', 'Vote_Count', 'Vote_Average',
       'Genre'],
      dtype='object')
```

In [21]:
```python
data.head(2)
```

Out[21]:

	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	3827.658	1151	8.1	Crime, Mystery, Thriller

Catagorizing Vote_Average column

We would cut the Vote_Average values and make 4 catagories: popular average below_avg not_popular to decribe it more using catigorize_col() function provided alone

In [22]:
```python
def categorize_col(data, col, labels):
    edges = [data[col].describe()['min'],
             data[col].describe()['25%'],
             data[col].describe()['50%'],
             data[col].describe()['75%'],
             data[col].describe()['max']]
    data[col] = pd.cut(data[col], edges, labels = labels, duplicates= "drop")
    return data
```

In [23]:
```python
labels = ["not_popular", "below_avg", "average", "popular"]

categorize_col(data, 'Vote_Average', labels)

data["Vote_Average"].unique()
```

Out[23]:
```
['popular', 'below_avg', 'average', 'not_popular', NaN]
Categories (4, object): ['not_popular' < 'below_avg' < 'average' < 'popular']
```

In [24]:
```python
data.head()
```

Out[24]:

	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	3827.658	1151	popular	Crime, Mystery, Thriller
2	2022	No Exit	2618.087	122	below_avg	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 [25]:
```python
#value counts group wise category in Vote_Average
data["Vote_Average"].value_counts()
```

Out[25]:
```
Vote_Average
not_popular    2467
popular        2450
average        2412
below_avg      2398
Name: count, dtype: int64
```

In [26]:
```python
data.dropna(inplace = True)

data.isna().sum()
```

Out[26]:
```
Release_Date    0
Title           0
Popularity      0
Vote_Count      0
Vote_Average    0
Genre           0
dtype: int64
```

In [27]:
```python
data.head(2)
```

Out[27]:

	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	3827.658	1151	popular	Crime, Mystery, Thriller

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

In [28]:
```python
data['Genre']=data['Genre'].str.split(', ')

data = data.explode('Genre').reset_index(drop=True)
data.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	3827.658	1151	popular	Crime
4	2022	The Batman	3827.658	1151	popular	Mystery

In [29]:
```python
# casting column into category

data['Genre']= data['Genre'].astype('category')

data['Genre'].dtypes
```

Out[29]:
```
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 [30]:
```python
data.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: 749.6+ KB
```

In [31]:
```python
data.nunique()
```

Out[31]:
```
Release_Date     100
Title           9415
Popularity      8088
Vote_Count      3265
Vote_Average       4
Genre             19
dtype: int64
```

Date Visualisation

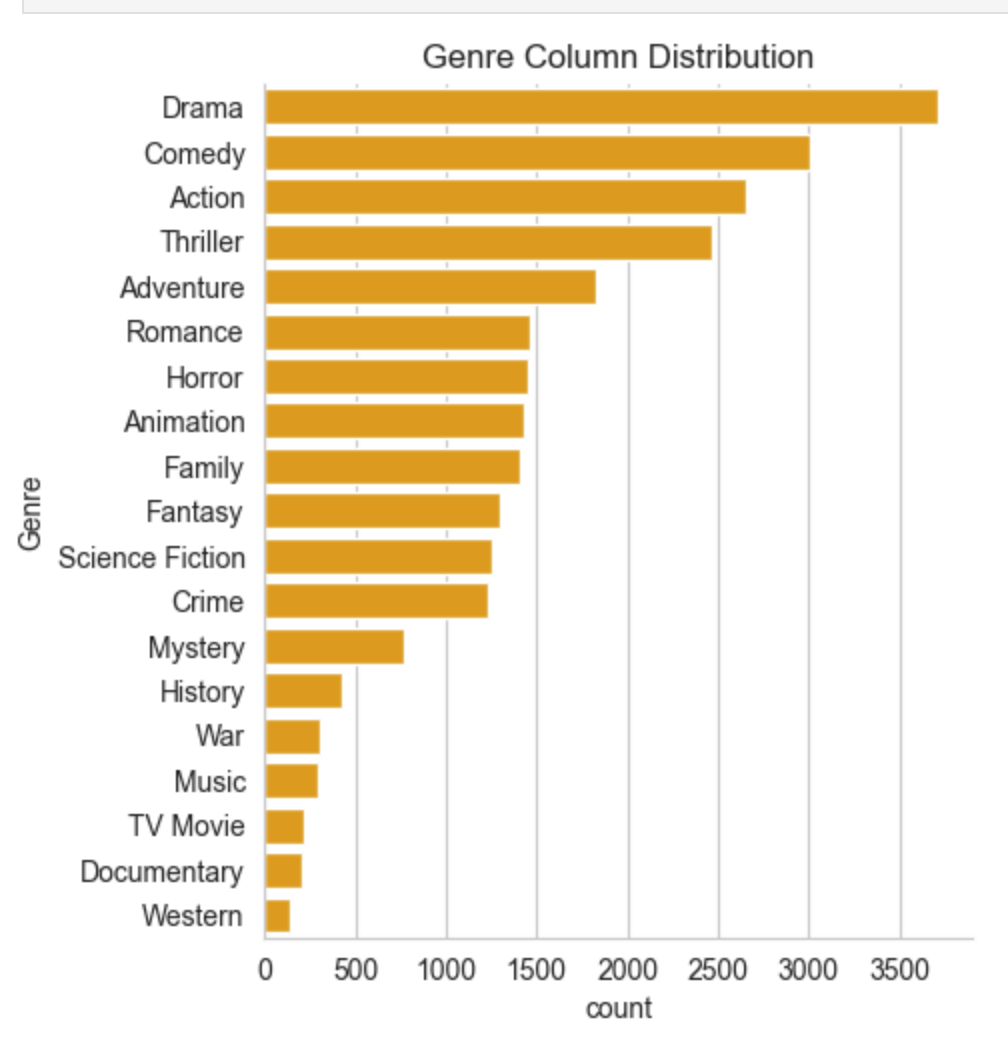
In [32]:
```python
sns.set_style('whitegrid')
```

1) What is the most frequent genre of movies relesased on Netflix?

In [33]:
```python
data['Genre'].describe()
```

Out[33]:
```
count     25552
unique       19
top       Drama
freq       3715
Name: Genre, dtype: object
```

In [34]:
```python
sns.catplot(y='Genre', data=data, kind ="count",
            order = data['Genre'].value_counts().index,
            color="orange")
plt.title('Genre Column Distribution')
plt.show()
```



Answer: Drama

2) Which has highest votes in vote average colummn ?

In [36]:
```python
data.head(2)
```

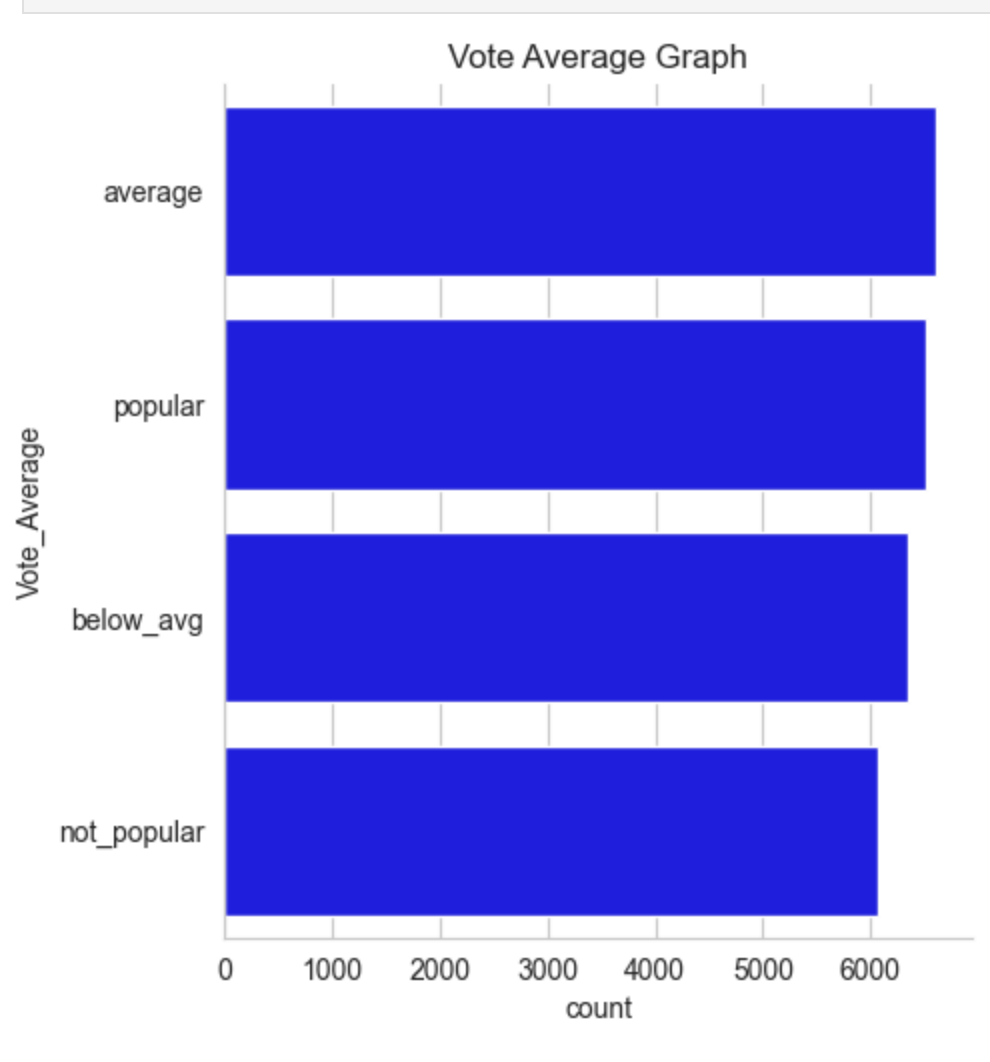
Out[36]:

	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

In [38]:
```python
data['Vote_Average'].describe()
```

Out[38]:
```
count       25552
unique          4
top       average
freq         6613
Name: Vote_Average, dtype: object
```

In [40]:
```python
sns.catplot(y = "Vote_Average", data=data, kind = "count",
            order=data['Vote_Average'].value_counts().index,
            color = "blue")
plt.title('Vote Average Graph')
plt.show()
```



Answer: Average

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

In [42]:
```python
data.head(2)
```

Out[42]:

	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

In [45]:
```python
data[data['Popularity'] == data['Popularity'].max()]
```

Out[45]:

	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

Answer: Spider-Man: No Way Home

Genre: Action, Adventure, Science Fiction

4) Which movie got the lowest polpularity? what's its genre?

In [48]:
```python
data.head(5)
```

Out[48]:

	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	3827.658	1151	popular	Crime
4	2022	The Batman	3827.658	1151	popular	Mystery

In [50]:
```python
data[data['Popularity']==data['Popularity'].min()]
```

Out[50]:

	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

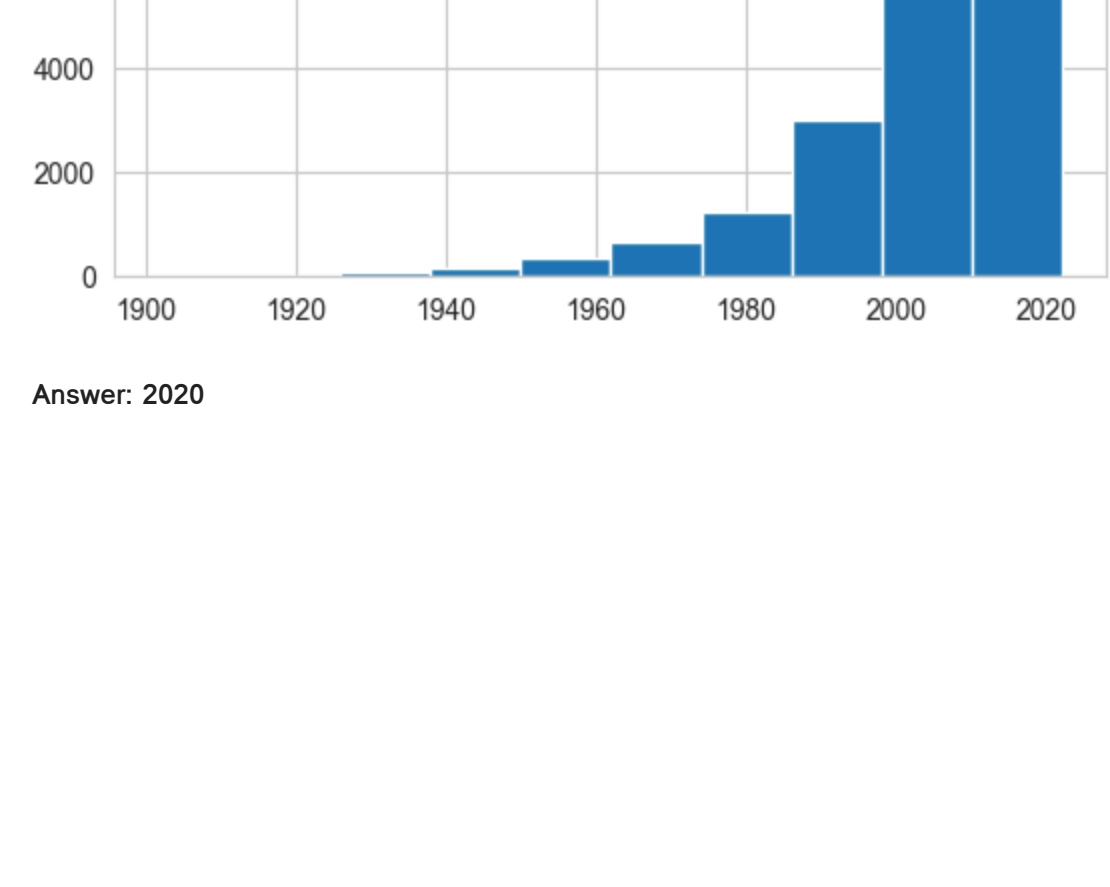
Answer: The United States vs. Billie Holiday; Genre: Music, Drama, History

And

Threads; Genre: War, Drama, Science Fiction

5) Which year has the most filmmed movies?

In [69]:
```python
data['Release_Date'].hist()
plt.title("Release Date Column Distribution")
plt.show()
```



Answer: 2020