import required libraries

In [3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Read the dataset

In [197…
```
movies = pd.read_csv(r'C:\Users\91939\Desktop\AI&DS\12thAug\archive\movie.csv')
movies
```

Out[197…

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy
...	...	...	...
27273	131254	Kein Bund für's Leben (2007)	Comedy
27274	131256	Feuer, Eis & Dosenbier (2002)	Comedy
27275	131258	The Pirates (2014)	Adventure
27276	131260	Rentun Ruusu (2001)	(no genres listed)
27277	131262	Innocence (2014)	Adventure|Fantasy|Horror

27278 rows × 3 columns

In [5]:
```
print(type(movies))
```

```
<class 'pandas.core.frame.DataFrame'>
```

In [6]:
```
movies.shape
```

Out[6]:
```
(27278, 3)
```

In [7]:
```
movies.columns
```

Out[7]:
```
Index(['movieId', 'title', 'genres'], dtype='object')
```

In [8]:
```
tags = pd.read_csv(r'C:\Users\91939\Desktop\AI&DS\12thAug\archive\tag.csv')
tags.head()
```

Out[8]:

	userId	movieId	tag	timestamp
0	18	4141	Mark Waters	2009-04-24 18:19:40
1	65	208	dark hero	2013-05-10 01:41:18
2	65	353	dark hero	2013-05-10 01:41:19
3	65	521	noir thriller	2013-05-10 01:39:43
4	65	592	dark hero	2013-05-10 01:41:18

In [9]:
```
tags.shape
```

Out[9]:
```
(465564, 4)
```

In [10]:
```
tags.columns
```

Out[10]:
```
Index(['userId', 'movieId', 'tag', 'timestamp'], dtype='object')
```

In [11]:
```
ratings = pd.read_csv(r'C:\Users\91939\Desktop\AI&DS\12thAug\archive\rating.csv'
ratings.head()
```

Out[11]:

	userId	movieId	rating	timestamp
0	1	2	3.5	2005-04-02 23:53:47
1	1	29	3.5	2005-04-02 23:31:16
2	1	32	3.5	2005-04-02 23:33:39
3	1	47	3.5	2005-04-02 23:32:07
4	1	50	3.5	2005-04-02 23:29:40

In [12]:
```
ratings.shape
```

Out[12]:
```
(20000263, 4)
```

In [13]:
```
ratings.columns
```

Out[13]:
```
Index(['userId', 'movieId', 'rating', 'timestamp'], dtype='object')
```

In [17]:
```
del tags['timestamp']
```

In [18]:
```
tags.columns
```

Out[18]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [24]:
```
#del ratings['timestamp']
```

In [20]:
```
ratings.columns
```

Out[20]:
```
Index(['userId', 'movieId', 'rating'], dtype='object')
```

In [25]:
```
ratings.head(1)
```

Out[25]:

	userId	movieId	rating
0	1	2	3.5

In [26]:
```
tags.head(1)
```

Out[26]:

	userId	movieId	tag
0	18	4141	Mark Waters

In [27]:
```
tags.head()
```

Out[27]:

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero
2	65	353	dark hero
3	65	521	noir thriller
4	65	592	dark hero

iloc - returns the individual data by indexing

In [28]:
```
tags.iloc[0]
```

Out[28]:
```
userId              18
movieId           4141
tag        Mark Waters
Name: 0, dtype: object
```

In [29]:
```
tags.iloc[0,2]
```

Out[29]:
```
'Mark Waters'
```

In [33]:
```
tags.iloc[0][2]
```

Out[33]:
```
'Mark Waters'
```

In [34]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [35]:
```
tags.iloc[0:5]
```

Out[35]:

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero
2	65	353	dark hero
3	65	521	noir thriller
4	65	592	dark hero

Data Structures

Series

In [36]:
```
row_0 = tags.iloc[0]
type(row_0)
```

Out[36]:
```
pandas.core.series.Series
```

In [37]:
```
print(row_0)
```

```
userId              18
movieId           4141
tag        Mark Waters
Name: 0, dtype: object
```

In [38]:
```
row_0.index
```

Out[38]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [39]:
```
row_0['userId']
```

Out[39]:
```
18
```

In [40]:
```
'rating' in row_0
```

Out[40]:
```
False
```

In [41]:
```
'movieId' in row_0
```

Out[41]:
```
True
```

In [42]:
```
row_0.name
```

Out[42]:
```
0
```

In [43]:
```
row_0 = row_0.rename('firstRow')
row_0.name
```

Out[43]:
```
'firstRow'
```

Dataframes

In [44]:
```
tags.head()
```

Out[44]:

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero
2	65	353	dark hero
3	65	521	noir thriller
4	65	592	dark hero

In [45]:
```
tags.index
```

Out[45]:
```
RangeIndex(start=0, stop=465564, step=1)
```

In [46]:
```
tags.columns
```

Out[46]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [47]:
```
tags.iloc[ [0,11,500] ]
```

Out[47]:

	userId	movieId	tag
0	18	4141	Mark Waters
11	65	1783	noir thriller
500	342	55908	entirely dialogue

In [48]:
```
tags.describe()
```

Out[48]:

	userId	movieId
count	465564.000000	465564.000000
mean	68712.354263	32627.762920
std	41877.674053	36080.241157
min	18.000000	1.000000
25%	28780.000000	2571.000000
50%	70201.000000	7373.000000
75%	107322.000000	62235.000000
max	138472.000000	131258.000000

Descriptive Statistics

In [49]:
```
ratings
```

Out[49]:

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5
2	1	32	3.5
3	1	47	3.5
4	1	50	3.5
...	...	...	...
20000258	138493	68954	4.5
20000259	138493	69526	4.5
20000260	138493	69644	3.0
20000261	138493	70286	5.0
20000262	138493	71619	2.5

20000263 rows × 3 columns

In [50]:
```
ratings['rating']
```

Out[50]:
```
0           3.5
1           3.5
2           3.5
3           3.5
4           3.5
           ...
20000258    4.5
20000259    4.5
20000260    3.0
20000261    5.0
20000262    2.5
Name: rating, Length: 20000263, dtype: float64
```

In [51]:
```
ratings['rating'].describe()
```

Out[51]:
```
count    2.000026e+07
mean     3.525529e+00
std      1.051989e+00
min      5.000000e-01
25%      3.000000e+00
50%      3.500000e+00
75%      4.000000e+00
max      5.000000e+00
Name: rating, dtype: float64
```

In [52]:
```
ratings.columns
```

Out[52]:
```
Index(['userId', 'movieId', 'rating'], dtype='object')
```

In [53]:
```
ratings['userId'].describe()
```

Out[53]:
```
count    2.000026e+07
mean     6.904587e+04
std      4.003863e+04
min      1.000000e+00
25%      3.439500e+04
50%      6.914100e+04
75%      1.036370e+05
max      1.384930e+05
Name: userId, dtype: float64
```

In [54]:
```
ratings.describe()
```

Out[54]:

	userId	movieId	rating
count	2.000026e+07	2.000026e+07	2.000026e+07
mean	6.904587e+04	9.041567e+03	3.525529e+00
std	4.003863e+04	1.978948e+04	1.051989e+00
min	1.000000e+00	1.000000e+00	5.000000e-01
25%	3.439500e+04	9.020000e+02	3.000000e+00
50%	6.914100e+04	2.167000e+03	3.500000e+00
75%	1.036370e+05	4.770000e+03	4.000000e+00
max	1.384930e+05	1.312620e+05	5.000000e+00

In [55]:
```
ratings['rating'].mean()
```

Out[55]:
```
3.5255285642993797
```

In [56]:
```
ratings['rating'].min()
```

Out[56]:
```
0.5
```

In [57]:
```
ratings['rating'].max()
```

Out[57]:
```
5.0
```

In [58]:
```
ratings['rating'].std()
```

Out[58]:
```
1.051988919275684
```

In [59]:
```
ratings['rating'].mode()
```

Out[59]:
```
0    4.0
Name: rating, dtype: float64
```

In [60]:
```
ratings.mode()
```

Out[60]:

	userId	movieId	rating
0	118205	296	4.0

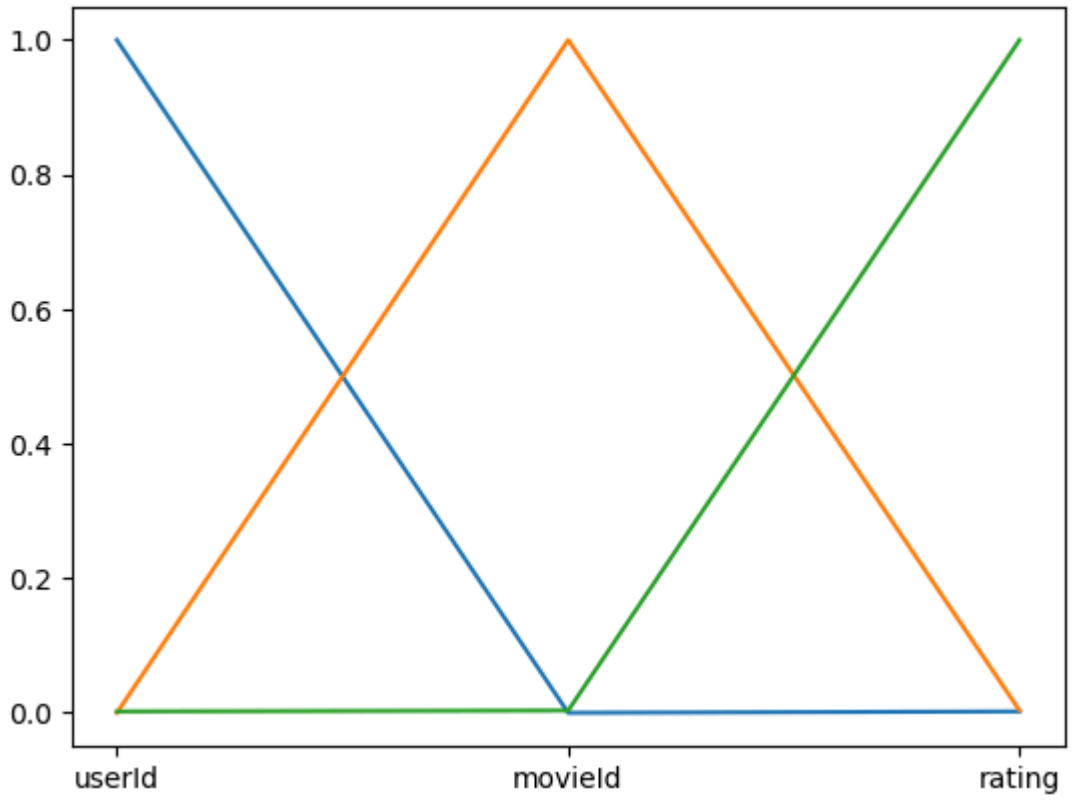
In [61]:
```
ratings.corr()
```

Out[61]:

	userId	movieId	rating
userId	1.000000	-0.000850	0.001175
movieId	-0.000850	1.000000	0.002606
rating	0.001175	0.002606	1.000000

In [62]:
```
plt.plot(ratings.corr())
```

Out[62]:
```
[<matplotlib.lines.Line2D at 0x13e8035a360>,
 <matplotlib.lines.Line2D at 0x13e8034e360>,
 <matplotlib.lines.Line2D at 0x13e8035ca70>]
```



In [63]:
```
filter1 = ratings['rating'] > 10
print(filter1)
```

```
0           False
1           False
2           False
3           False
4           False
            ...
20000258    False
20000259    False
20000260    False
20000261    False
20000262    False
Name: rating, Length: 20000263, dtype: bool
```

In [64]:
```
filter1.any()
```

Out[64]:
```
False
```

In [65]:
```
filter2 = ratings['rating'] > 0
filter2.all()
```

Out[65]:
```
True
```

Data Cleaning: Handling Missing Data

In [66]:
```
movies.shape
```

Out[66]:
```
(27278, 3)
```

In [67]:
```
movies.isnull()
```

Out[67]:

	movieId	title	genres
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
...	...	...	...
27273	False	False	False
27274	False	False	False
27275	False	False	False
27276	False	False	False
27277	False	False	False

27278 rows × 3 columns

In [68]:
```
movies.isnull().any()
```

Out[68]:
```
movieId    False
title      False
genres     False
dtype: bool
```

In [69]:
```
ratings.shape
```

Out[69]:
```
(20000263, 3)
```

In [71]:
```
ratings.isnull().any().any()
```

Out[71]:
```
False
```

In [72]:
```
ratings.isnull().any()
```

Out[72]:
```
userId     False
movieId    False
rating     False
dtype: bool
```

In [73]:
```
ratings.isnull()
```

Out[73]:

	userId	movieId	rating
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
...	...	...	...
20000258	False	False	False
20000259	False	False	False
20000260	False	False	False
20000261	False	False	False
20000262	False	False	False

20000263 rows × 3 columns

In [75]:
```
tags.shape
```

Out[75]:
```
(465564, 3)
```

In [76]:
```
tags.isnull().any().any()
```

Out[76]:
```
True
```

In [77]:
```
tags=tags.dropna()
```

In [78]:
```
tags
```

Out[78]:

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero
2	65	353	dark hero
3	65	521	noir thriller
4	65	592	dark hero
...	...	...	...
465559	138446	55999	dragged
465560	138446	55999	Jason Bateman
465561	138446	55999	quirky
465562	138446	55999	sad
465563	138472	923	rise to power

465548 rows × 3 columns

In [79]:
```
tags.isnull().any().any()
```

Out[79]:
```
False
```

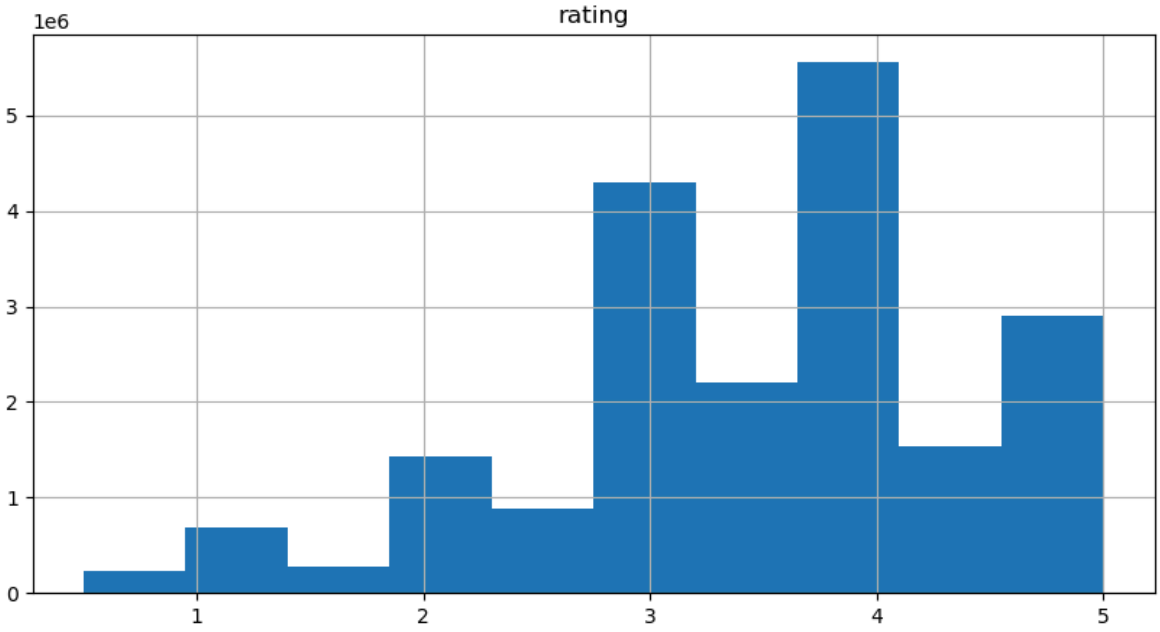
In [80]:
```
tags.shape
```

Out[80]:
```
(465548, 3)
```

Data Visualization

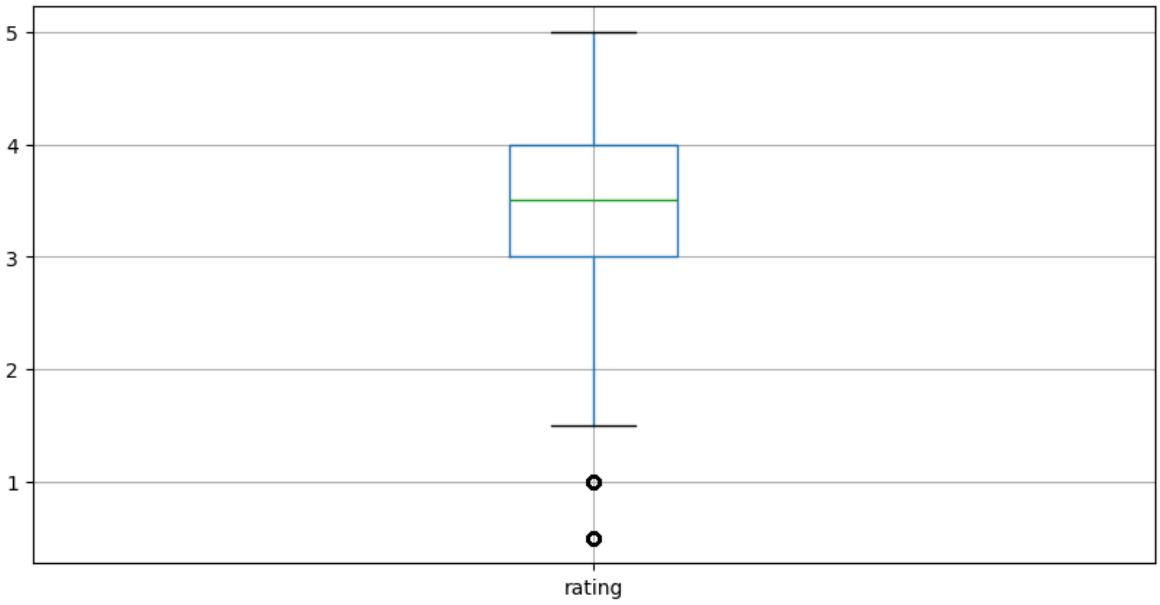
In [81]:
```
%matplotlib inline

ratings.hist(column='rating', figsize=(10,5))
```

Out[81]:
```
array([[<Axes: title={'center': 'rating'}>]], dtype=object)
```



In [82]:
```
ratings.boxplot(column='rating', figsize=(10,5))
```

Out[82]: <Axes: >



In [83]:
```
ratings['rating'].min()
```

Out[83]: 0.5

In [84]:
```
ratings['rating'].max()
```

Out[84]: 5.0

In [85]:
```
ratings['rating'].median()
```

Out[85]: 3.5

Slicing Out Columns

In [86]:
```
tags['tag'].head()
```

Out[86]:
```
0      Mark Waters
1        dark hero
2        dark hero
3    noir thriller
4        dark hero
Name: tag, dtype: object
```

In [89]:
```
movies[['title','genres']].head()
```

Out[89]:

	title	genres
0	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	Jumanji (1995)	Adventure|Children|Fantasy
2	Grumpier Old Men (1995)	Comedy|Romance
3	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	Father of the Bride Part II (1995)	Comedy

In [90]:
```
ratings.head(10)
```

Out[90]:

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5
2	1	32	3.5
3	1	47	3.5
4	1	50	3.5
5	1	112	3.5
6	1	151	4.0
7	1	223	4.0
8	1	253	4.0
9	1	260	4.0

In [91]:
```
ratings[-10:]
```

Out[91]:

	userId	movieId	rating
20000253	138493	60816	4.5
20000254	138493	61160	4.0
20000255	138493	65682	4.5
20000256	138493	66762	4.5
20000257	138493	68319	4.5
20000258	138493	68954	4.5
20000259	138493	69526	4.5
20000260	138493	69644	3.0
20000261	138493	70286	5.0
20000262	138493	71619	2.5

In [92]:
```
ratings
```

Out[92]:

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5
2	1	32	3.5
3	1	47	3.5
4	1	50	3.5
...	...	...	...
20000258	138493	68954	4.5
20000259	138493	69526	4.5
20000260	138493	69644	3.0
20000261	138493	70286	5.0
20000262	138493	71619	2.5

20000263 rows × 3 columns

In [93]:
```
tags
```

Out[93]:

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero
2	65	353	dark hero
3	65	521	noir thriller
4	65	592	dark hero
...	...	...	...
465559	138446	55999	dragged
465560	138446	55999	Jason Bateman
465561	138446	55999	quirky
465562	138446	55999	sad
465563	138472	923	rise to power

465548 rows × 3 columns

In [94]:
```
tags['tag'][-10:]
```

Out[94]:
```
465554                 visually appealing
465555                    family friendly
465556    Scary Movies To See on Halloween
465557                           Peter Pan
465558                 visually appealing
465559                           dragged
465560                      Jason Bateman
465561                             quirky
465562                                sad
465563                      rise to power
Name: tag, dtype: object
```

In [102…
```
tag_counts = tags['tag'].value_counts()
tag_counts[:10]
```

Out[102…
```
tag
sci-fi            3384
based on a book   3281
atmospheric       2917
comedy            2779
action            2657
surreal           2427
BD-R              2334
twist ending      2323
funny             2072
dystopia          1991
Name: count, dtype: int64
```

In [103…
```
tag_counts = tags['tag'].unique()
tag_counts[:10]
```

Out[103…
```
array(['Mark Waters', 'dark hero', 'noir thriller', 'bollywood',
       'screwball comedy', 'mars', 'neo-noir', 'jesus', 'dragon',
       'conspiracy theory'], dtype=object)
```

In [104…
```
tag_counts = tags.tag.unique()
tag_counts[:10]
```

Out[104…
```
array(['Mark Waters', 'dark hero', 'noir thriller', 'bollywood',
       'screwball comedy', 'mars', 'neo-noir', 'jesus', 'dragon',
       'conspiracy theory'], dtype=object)
```

In [110…
```
'dark hero' in tags
```

Out[110…
```
False
```

In [133…
```
tags[tags['tag'] == 'dark hero']
```

Out[133…

	userId	movieId	tag
1	65	208	dark hero
2	65	353	dark hero
4	65	592	dark hero
21	65	6874	dark hero
1353	558	5146	dark hero
…	…	…	…
460229	136015	6874	dark hero
462399	137717	5146	dark hero
462400	137717	7373	dark hero
462466	137717	60684	dark hero
462467	137717	91529	dark hero

194 rows × 3 columns

In [123…
```
print(len('Mark Waters'))
```

```
11
```

In [117…
```
tags[tags['tag'] == 'sci-fi']
```

Out[117…

	userId	movieId	tag
162	129	4878	sci-fi
259	190	2011	sci-fi
303	318	260	sci-fi
386	342	2571	sci-fi
438	342	8914	sci-fi
…	…	…	…
464719	138301	27904	sci-fi
464793	138301	52328	sci-fi
464865	138301	70286	sci-fi
464961	138301	109487	sci-fi
464970	138301	112623	sci-fi

3384 rows × 3 columns

In [107…
```
tags
```

Out[107…

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero
2	65	353	dark hero
3	65	521	noir thriller
4	65	592	dark hero
…	…	…	…
465559	138446	55999	dragged
465560	138446	55999	Jason Bateman
465561	138446	55999	quirky
465562	138446	55999	sad
465563	138472	923	rise to power

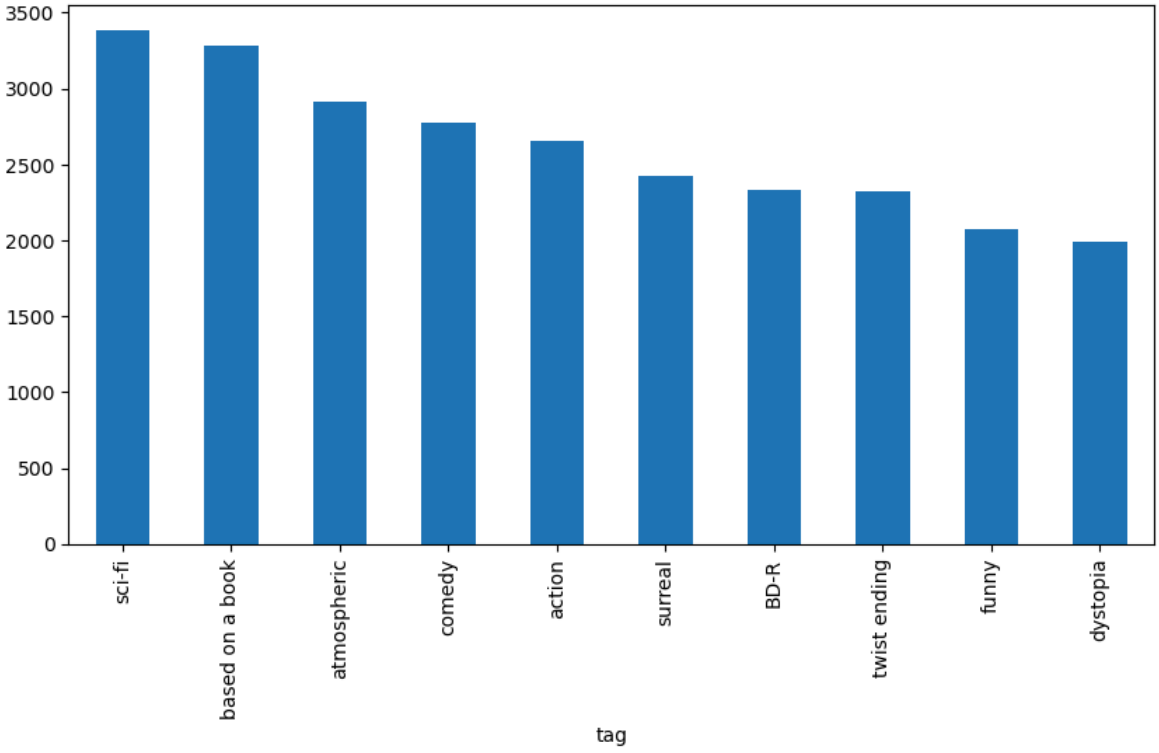
465548 rows × 3 columns

In [136…
```
tag_counts = tags['tag'].value_counts()
tag_counts
```

Out[136…
```
tag
sci-fi                            3384
based on a book                   3281
atmospheric                       2917
comedy                            2779
action                            2657
                                  ...
Paul Adelstein                       1
the wig                              1
killer fish                          1
genetically modified monsters        1
topless scene                        1
Name: count, Length: 38643, dtype: int64
```

In [144…
```
tag_counts[:10].plot(kind='bar', figsize=(10,5))
```

Out[144…
```
<Axes: xlabel='tag'>
```



In [172…
```
tags.head(10)
```

Out[172…

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero
2	65	353	dark hero
3	65	521	noir thriller
4	65	592	dark hero
5	65	668	bollywood
6	65	898	screwball comedy
7	65	1248	noir thriller
8	65	1391	mars
9	65	1617	neo-noir

Filters for Selecting Rows

In [145…

```
is_highly_rated = ratings['rating'] >= 5.0
ratings[is_highly_rated][30:50]
```

Out[145...

	userId	movieId	rating
239	3	50	5.0
242	3	175	5.0
244	3	223	5.0
245	3	260	5.0
246	3	316	5.0
247	3	318	5.0
248	3	329	5.0
252	3	457	5.0
253	3	480	5.0
254	3	490	5.0
256	3	541	5.0
258	3	593	5.0
263	3	858	5.0
264	3	904	5.0
267	3	924	5.0
268	3	953	5.0
271	3	1060	5.0
272	3	1073	5.0
275	3	1084	5.0
276	3	1089	5.0

In [146...

```
is_action= movies['genres'].str.contains('Action')
movies[is_action][5:15]
```

Out[146...

	movieId	title	genres
22	23	Assassins (1995)	Action|Crime|Thriller
41	42	Dead Presidents (1995)	Action|Crime|Drama
43	44	Mortal Kombat (1995)	Action|Adventure|Fantasy
50	51	Guardian Angel (1994)	Action|Drama|Thriller
65	66	Lawnmower Man 2: Beyond Cyberspace (1996)	Action|Sci-Fi|Thriller
69	70	From Dusk Till Dawn (1996)	Action|Comedy|Horror|Thriller
70	71	Fair Game (1995)	Action
75	76	Screamers (1995)	Action|Sci-Fi|Thriller
77	78	Crossing Guard, The (1995)	Action|Crime|Drama|Thriller
85	86	White Squall (1996)	Action|Adventure|Drama

In [147...

```
movies[is_action].head(15)
```

Out[147...

	movieId	title	genres
5	6	Heat (1995)	Action|Crime|Thriller
8	9	Sudden Death (1995)	Action
9	10	GoldenEye (1995)	Action|Adventure|Thriller
14	15	Cutthroat Island (1995)	Action|Adventure|Romance
19	20	Money Train (1995)	Action|Comedy|Crime|Drama|Thriller
22	23	Assassins (1995)	Action|Crime|Thriller
41	42	Dead Presidents (1995)	Action|Crime|Drama
43	44	Mortal Kombat (1995)	Action|Adventure|Fantasy
50	51	Guardian Angel (1994)	Action|Drama|Thriller
65	66	Lawnmower Man 2: Beyond Cyberspace (1996)	Action|Sci-Fi|Thriller
69	70	From Dusk Till Dawn (1996)	Action|Comedy|Horror|Thriller
70	71	Fair Game (1995)	Action
75	76	Screamers (1995)	Action|Sci-Fi|Thriller
77	78	Crossing Guard, The (1995)	Action|Crime|Drama|Thriller
85	86	White Squall (1996)	Action|Adventure|Drama

Group By and Aggregate

In [148...

```
ratings_count = ratings[['movieId','rating']].groupby('rating').count()
ratings_count
```

Out[148...

rating	movieId
0.5	239125
1.0	680732
1.5	279252
2.0	1430997
2.5	883398
3.0	4291193
3.5	2200156
4.0	5561926
4.5	1534824
5.0	2898660

In [149...
```
average_rating = ratings[['movieId','rating']].groupby('movieId').mean()
average_rating.head()
```

Out[149...

movieId	rating
1	3.921240
2	3.211977
3	3.151040
4	2.861393
5	3.064592

In [150...
```
average_rating = ratings[['movieId','rating']].groupby('movieId').median()
average_rating.head()
```

Out[150...

movieId	rating
1	4.0
2	3.0
3	3.0
4	3.0
5	3.0

In [151...
```
movie_count = ratings[['movieId','rating']].groupby('movieId').count()
movie_count.head()
```

Out[151…

movieId	rating
1	49695
2	22243
3	12735
4	2756
5	12161

In [152…

```
movie_count = ratings[['movieId','rating']].groupby('movieId').count()
movie_count.tail()
```

Out[152…

movieId	rating
131254	1
131256	1
131258	1
131260	1
131262	1

merge dataframes

In [153…

```
tags.head()
```

Out[153…

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero
2	65	353	dark hero
3	65	521	noir thriller
4	65	592	dark hero

In [154…

```
movies.head()
```

Out[154…

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

In [161…

```
t = movies.merge(tags, on='movieId', how='inner')
t.head()
```

Out[161…

	movieId	title	genres	userId	tag
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1644	Watched
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	computer animation
2	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	Disney animated feature
3	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	Pixar animation
4	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	TÃ©a Leoni does not star in this movie

In [165…

```
t[t['tag'] == 'Mark Waters']
```

Out[165...

	movieId	title	genres	userId	tag
84036	1648	House of Yes, The (1997)	Comedy|Drama	70201	Mark Waters
162637	4141	Head Over Heels (2001)	Comedy|Romance	18	Mark Waters
162639	4141	Head Over Heels (2001)	Comedy|Romance	70201	Mark Waters
212960	6593	Freaky Friday (2003)	Children|Comedy|Fantasy	70201	Mark Waters
234375	7451	Mean Girls (2004)	Comedy	70201	Mark Waters
234395	7451	Mean Girls (2004)	Comedy	77463	Mark Waters
281094	36525	Just Like Heaven (2005)	Comedy|Fantasy|Romance	70201	Mark Waters
336445	58105	Spiderwick Chronicles, The (2008)	Adventure|Children|Drama|Fantasy|IMAX	70201	Mark Waters
336458	58105	Spiderwick Chronicles, The (2008)	Adventure|Children|Drama|Fantasy|IMAX	107711	Mark Waters
413659	87483	Mr. Popper's Penguins (2011)	Comedy	70201	Mark Waters

Combine aggreagation, merging, and filters to get useful analytics

In [166...

```
avg_ratings= ratings.groupby('movieId', as_index=False).mean()
del avg_ratings['userId']
avg_ratings.head()
```

Out[166...

	movieId	rating
0	1	3.921240
1	2	3.211977
2	3	3.151040
3	4	2.861393
4	5	3.064592

In [167…
```
box_office = movies.merge(avg_ratings, on='movieId', how='inner')
box_office.tail()
```

Out[167…

	movieId	title	genres	rating
26739	131254	Kein Bund für's Leben (2007)	Comedy	4.0
26740	131256	Feuer, Eis & Dosenbier (2002)	Comedy	4.0
26741	131258	The Pirates (2014)	Adventure	2.5
26742	131260	Rentun Ruusu (2001)	(no genres listed)	3.0
26743	131262	Innocence (2014)	Adventure|Fantasy|Horror	4.0

In [168…
```
is_highly_rated = box_office['rating'] >= 4.0
box_office[is_highly_rated][-5:]
```

Out[168…

	movieId	title	genres	rating
26737	131250	No More School (2000)	Comedy	4.0
26738	131252	Forklift Driver Klaus: The First Day on the Jo…	Comedy|Horror	4.0
26739	131254	Kein Bund für's Leben (2007)	Comedy	4.0
26740	131256	Feuer, Eis & Dosenbier (2002)	Comedy	4.0
26743	131262	Innocence (2014)	Adventure|Fantasy|Horror	4.0

In [169…
```
is_highly_rated = box_office['rating'] >= 4.0
box_office[is_highly_rated].tail()
```

Out[169…

	movieId	title	genres	rating
26737	131250	No More School (2000)	Comedy	4.0
26738	131252	Forklift Driver Klaus: The First Day on the Jo…	Comedy|Horror	4.0
26739	131254	Kein Bund für's Leben (2007)	Comedy	4.0
26740	131256	Feuer, Eis & Dosenbier (2002)	Comedy	4.0
26743	131262	Innocence (2014)	Adventure|Fantasy|Horror	4.0

In [170…
```
is_Adventure = box_office['genres'].str.contains('Adventure')
box_office[is_Adventure][:5]
```

Out[170…

	movieId	title	genres	rating
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	3.921240
1	2	Jumanji (1995)	Adventure|Children|Fantasy	3.211977
7	8	Tom and Huck (1995)	Adventure|Children	3.142049
9	10	GoldenEye (1995)	Action|Adventure|Thriller	3.430029
12	13	Balto (1995)	Adventure|Animation|Children	3.272416

In [171…

```
box_office[is_Adventure & is_highly_rated][-5:]
```

Out[171…

	movieId	title	genres	rating
26611	130586	Itinerary of a Spoiled Child (1988)	Adventure|Drama	4.5
26655	130996	The Beautiful Story (1992)	Adventure|Drama|Fantasy	5.0
26667	131050	Stargate SG-1 Children of the Gods - Final Cut...	Adventure|Sci-Fi|Thriller	5.0
26736	131248	Brother Bear 2 (2006)	Adventure|Animation|Children|Comedy|Fantasy	4.0
26743	131262	Innocence (2014)	Adventure|Fantasy|Horror	4.0

Vectorized String Operations

In [172…

```
movies.head()
```

Out[172…

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

Split 'genres' into multiple columns

In [175…

```
movies['genres']
```

Out[175…
```
0        Adventure|Animation|Children|Comedy|Fantasy
1                          Adventure|Children|Fantasy
2                                       Comedy|Romance
3                                 Comedy|Drama|Romance
4                                               Comedy
                              ...
27273                                           Comedy
27274                                           Comedy
27275                                        Adventure
27276                                (no genres listed)
27277                           Adventure|Fantasy|Horror
Name: genres, Length: 27278, dtype: object
```

In [181…
```
movie_genres = movies['genres'].str.split('|', expand=True)
```

In [179…
```
movie_genres[:10]
```

Out[179…

	0	1	2	3	4	5	6	7	8	9
0	Adventure	Animation	Children	Comedy	Fantasy	None	None	None	None	None
1	Adventure	Children	Fantasy	None	None	None	None	None	None	None
2	Comedy	Romance	None	None	None	None	None	None	None	None
3	Comedy	Drama	Romance	None	None	None	None	None	None	None
4	Comedy	None	None	None	None	None	None	None	None	None
5	Action	Crime	Thriller	None	None	None	None	None	None	None
6	Comedy	Romance	None	None	None	None	None	None	None	None
7	Adventure	Children	None	None	None	None	None	None	None	None
8	Action	None	None	None	None	None	None	None	None	None
9	Action	Adventure	Thriller	None	None	None	None	None	None	None

Add a new column for comedy genre flag

In [185…
```
movie_genres['isComedy'] = movies['genres'].str.contains('Comedy')
```

In [187…
```
movie_genres[:10]
```

Out[187...

	0	1	2	3	4	5	6	7	8	9
0	Adventure	Animation	Children	Comedy	Fantasy	None	None	None	None	None
1	Adventure	Children	Fantasy	None	None	None	None	None	None	None
2	Comedy	Romance	None	None	None	None	None	None	None	None
3	Comedy	Drama	Romance	None	None	None	None	None	None	None
4	Comedy	None	None	None	None	None	None	None	None	None
5	Action	Crime	Thriller	None	None	None	None	None	None	None
6	Comedy	Romance	None	None	None	None	None	None	None	None
7	Adventure	Children	None	None	None	None	None	None	None	None
8	Action	None	None	None	None	None	None	None	None	None
9	Action	Adventure	Thriller	None	None	None	None	None	None	None

Extract year from title e.g. (2007)

In [198...
```
movies.columns
```

Out[198...
```
Index(['movieId', 'title', 'genres'], dtype='object')
```

In [199...
```
movies.head()
```

Out[199...

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

In [200...
```
movies['year'] = movies['title'].str.extract('.*\((.*)\).*', expand=True)
```

In [193...
```
movies.tail()
```

Out[193...

	movieId	title	genres	year
27273	131254	Kein Bund für's Leben (2007)	Comedy	2007
27274	131256	Feuer, Eis & Dosenbier (2002)	Comedy	2002
27275	131258	The Pirates (2014)	Adventure	2014
27276	131260	Rentun Ruusu (2001)	(no genres listed)	2001
27277	131262	Innocence (2014)	Adventure|Fantasy|Horror	2014

Parsing Timestamps

In [201…
```
tags = pd.read_csv(r'C:\Users\91939\Desktop\AI&DS\12thAug\archive\tag.csv')
```

In [202…
```
tags.dtypes
```

Out[202…
```
userId        int64
movieId       int64
tag          object
timestamp    object
dtype: object
```

In [203…
```
tags.head(5)
```

Out[203…

	userId	movieId	tag	timestamp
0	18	4141	Mark Waters	2009-04-24 18:19:40
1	65	208	dark hero	2013-05-10 01:41:18
2	65	353	dark hero	2013-05-10 01:41:19
3	65	521	noir thriller	2013-05-10 01:39:43
4	65	592	dark hero	2013-05-10 01:41:18

In [206…
```
tags['parsed_time'] = pd.to_datetime(tags['timestamp'])
```

In [207…
```
tags['parsed_time'].dtype
```

Out[207…
```
dtype('<M8[ns]')
```

In [208…
```
tags.head(2)
```

Out[208…

	userId	movieId	tag	timestamp	parsed_time
0	18	4141	Mark Waters	2009-04-24 18:19:40	2009-04-24 18:19:40
1	65	208	dark hero	2013-05-10 01:41:18	2013-05-10 01:41:18

In [209…
```
greater_than_t = tags['parsed_time'] > '2015-02-01'

selected_rows = tags[greater_than_t]

tags.shape, selected_rows.shape
```

Out[209…
```
((465564, 5), (12130, 5))
```

In [213…
```
tags.sort_values(by='parsed_time', ascending=True)[:10]
```

Out[213...

	userId	movieId	tag	timestamp	parsed_time
333932	100371	2788	monty python	2005-12-24 13:00:10	2005-12-24 13:00:10
333927	100371	1732	coen brothers	2005-12-24 13:00:36	2005-12-24 13:00:36
333924	100371	1206	stanley kubrick	2005-12-24 13:00:48	2005-12-24 13:00:48
333923	100371	1193	jack nicholson	2005-12-24 13:02:51	2005-12-24 13:02:51
333939	100371	5004	peter sellers	2005-12-24 13:03:19	2005-12-24 13:03:19
333922	100371	47	morgan freeman	2005-12-24 13:03:32	2005-12-24 13:03:32
333921	100371	47	brad pitt	2005-12-24 13:03:32	2005-12-24 13:03:32
333936	100371	4011	brad pitt	2005-12-24 13:03:51	2005-12-24 13:03:51
333937	100371	4011	guy ritchie	2005-12-24 13:03:51	2005-12-24 13:03:51
333920	100371	32	bruce willis	2005-12-24 13:04:02	2005-12-24 13:04:02

In [214...

```
average_rating = ratings[['movieId','rating']].groupby('movieId', as_index=False
average_rating.tail()
```

Out[214...

	movieId	rating
26739	131254	4.0
26740	131256	4.0
26741	131258	2.5
26742	131260	3.0
26743	131262	4.0