In [46]:
```
import pandas as pd
```

In [48]:
```
movies=pd.read_csv(r'C:\Users\arsha_4tjdyqj\Downloads\archive\movie.csv')
```

In [50]:
```
movies
```

Out[50]:

	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 [52]:
```
movies.head(1)
```

Out[52]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy

In [54]:
```
ratings=pd.read_csv(r'C:\Users\arsha_4tjdyqj\Downloads\archive\rating.csv')
```

In [55]:
```
tags=pd.read_csv(r'C:\Users\arsha_4tjdyqj\Downloads\archive\tag.csv')
```

In [56]:
```
print(movies.shape)
print(ratings.shape)
print(tags.shape)
```

```
(27278, 3)
(20000263, 4)
(465564, 4)
```

In [57]:
```
del ratings['timestamp']
del tags['timestamp']
```

In [63]:
```
tags.head()
```

Out[63]:

	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

In [66]:
```
row_0=tags.iloc[0]
type(row_0)
```

Out[66]:
```
pandas.core.series.Series
```

In [68]:
```
print(row_0)
```

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

In [70]:
```
row_0.index
```

Out[70]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [78]:
```
tags.columns
```

Out[78]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [80]:
```
row_0['userId']
```

Out[80]:
```
18
```

In [82]:
```
'rating' in row_0
```

Out[82]:
```
False
```

In [84]:
```
row_0.name
```

Out[84]:
```
0
```

In [86]:
```
row_0=row_0.rename('firstRow')
row_0.name
```

Out[86]:
```
'firstRow'
```

In [88]:
```
tags
```

Out[88]:

	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

465564 rows × 3 columns

DATA FRAMES

In [90]:
```
tags.head()
```

Out[90]:

	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 [92]:
```
tags.index
```

Out[92]:
```
RangeIndex(start=0, stop=465564, step=1)
```

In [94]:
```
tags.columns
```

Out[94]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [96]:
```
tags.iloc[[0,11,500]]
```

Out[96]:

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

Descriptive Statistics

In [98]:
```
ratings
```

Out[98]:

	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 [100…
```
ratings['rating'].describe()
```

Out[100…
```
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 [104…
```
ratings.describe()
```

Out[104…

	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 [106…
```
ratings['rating'].mean()
```

Out[106…
```
3.5255285642993797
```

In [108…
```
ratings.mean()
```

Out[108…
```
userId     69045.872583
movieId     9041.567330
rating         3.525529
dtype: float64
```

In [110…
```
ratings['rating'].min()
```

Out[110…
```
0.5
```

In [112…
```
ratings['rating'].max()
```

Out[112…
```
5.0
```

In [114…
```
ratings['rating'].std()
```

Out[114…
```
1.051988919275684
```

In [116…
```
ratings['rating'].mode()
```

Out[116…
```
0    4.0
Name: rating, dtype: float64
```

In [118…
```
ratings.corr()
```

Out[118…

	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 [120…
```
filter1=ratings['rating']>10
print(filter1)
filter1.any()
```

```
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
```

Out[120… False

In [125…
```
filter2=ratings['rating']>0
filter2.all()
```

Out[125… True

In [127…
```
ratings
```

Out[127…

	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

Data Cleaning: Handling Missing Data

In [129…
```
movies.shape
```

Out[129…
```
(27278, 3)
```

In [133…
```
movies.isnull().any()
```

Out[133…
```
movieId    False
title      False
genres     False
dtype: bool
```

In [135…
```
movies.isnull().any().any()
```

Out[135…
```
False
```

In [137…
```
ratings.shape
```

Out[137…
```
(20000263, 3)
```

In [139…
```
ratings.isnull().any().any()
```

Out[139…
```
False
```

In [141…
```
tags.shape
```

Out[141…
```
(465564, 3)
```

In [143…
```
tags.isnull().any().any()
```

Out[143…
```
True
```

we have some tags which ara null

In [147…
```
tags=tags.dropna() #dropna funtion is used to rows that contains the null values
```

In [149…
```
tags.isnull().any().any()
```

Out[149…
```
False
```

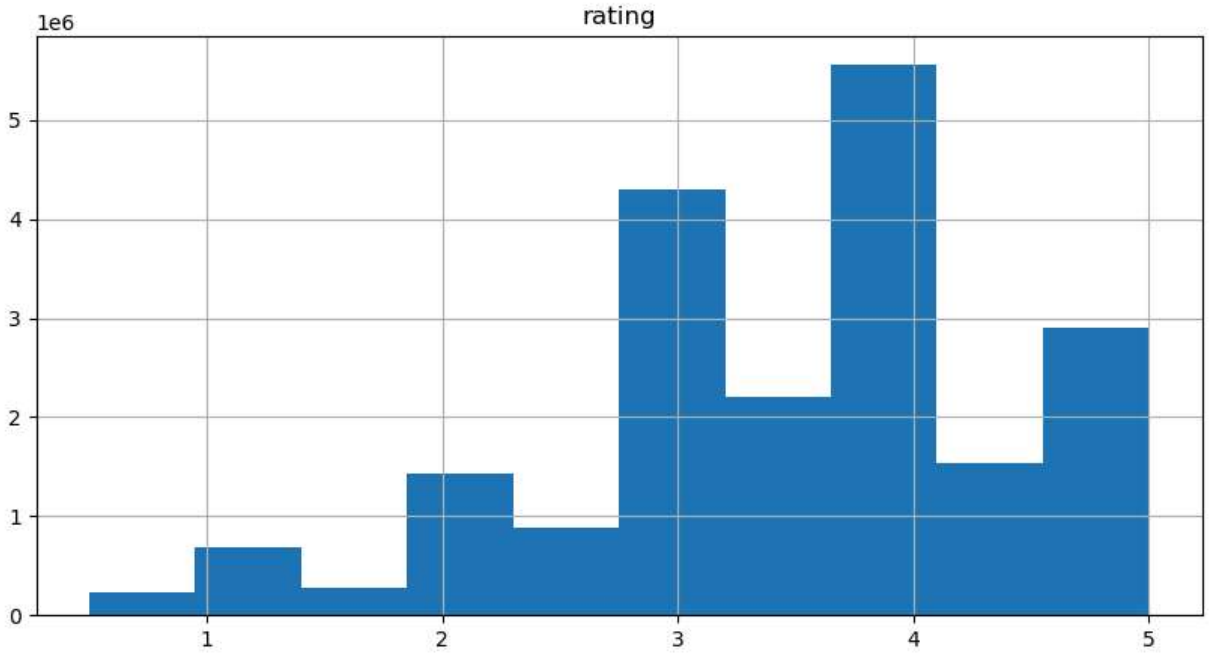
In [151…
```
tags.shape #the shape is different from the above where as the null values are remo
```

Out[151…
```
(465548, 3)
```

Data Visualization

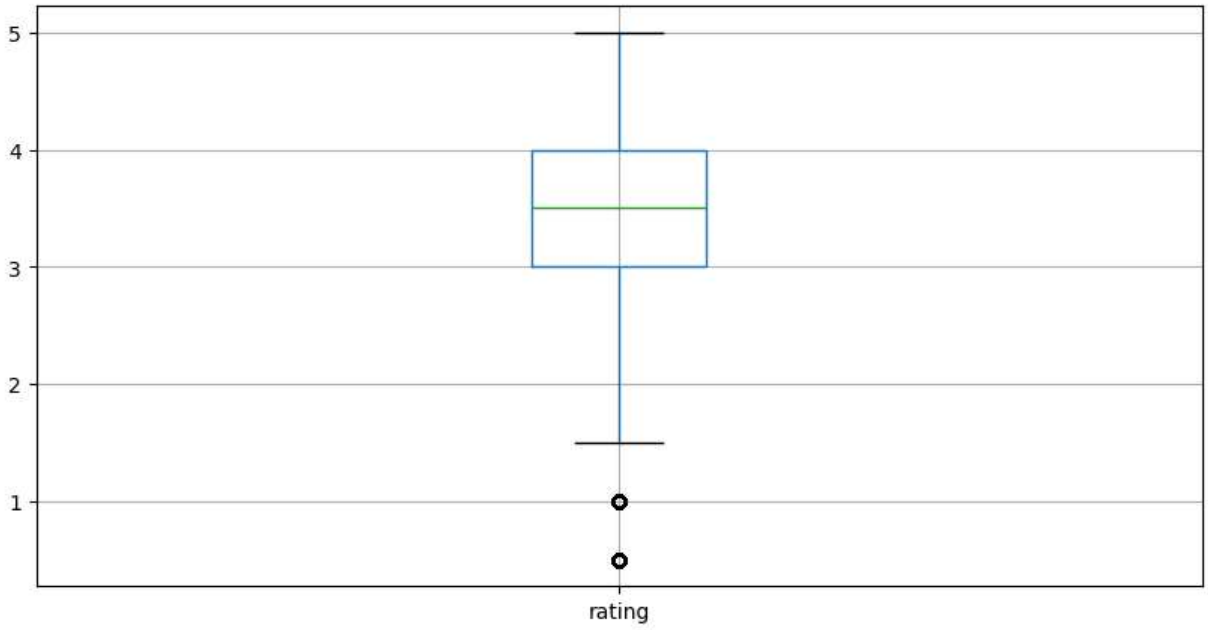
In [155…
```
import matplotlib.pyplot as plt
%matplotlib inline
ratings.hist(column='rating',figsize=(10,5))
```

Out[155…
```
array([[<Axes: title={'center': 'rating'}>]], dtype=object)
```



In [164…
```
ratings.boxplot(column='rating',figsize=(10,5))
```

Out[164…
```
<Axes: >
```



Slicing Out Columns

In [166…
```
tags['tag'].head()
```

Out[166…
```
0      Mark Waters
1        dark hero
2        dark hero
3    noir thriller
4        dark hero
Name: tag, dtype: object
```

In [168…
```
tags
```

Out[168…

	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 [170…
```
movies
```

Out[170…

	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 [178…
```
movies[['title','genres']].head(10)
```

Out[178…

	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
5	Heat (1995)	Action|Crime|Thriller
6	Sabrina (1995)	Comedy|Romance
7	Tom and Huck (1995)	Adventure|Children
8	Sudden Death (1995)	Action
9	GoldenEye (1995)	Action|Adventure|Thriller

In [182…

```
ratings[-10:]
```

Out[182…

	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 [188…

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

Out[188…

```
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 [190…

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

Out[190…

```
<Axes: xlabel='tag'>
```