In [3]:
```
import pandas as pd
```

In [5]:
```
ratings = pd.read_csv(r'C:\Users\velug\Downloads\rating\rating.csv')
```

In [6]:
```
ratings
```

Out[6]:

	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
...	...	...	...	...
20000258	138493	68954	4.5	2009-11-13 15:42:00
20000259	138493	69526	4.5	2009-12-03 18:31:48
20000260	138493	69644	3.0	2009-12-07 18:10:57
20000261	138493	70286	5.0	2009-11-13 15:42:24
20000262	138493	71619	2.5	2009-10-17 20:25:36

20000263 rows × 4 columns

In [7]:
```
movies = pd.read_csv(r'C:\Users\velug\Downloads\rating\movie.csv')
```

In [11]:
```
movies
```

Out[11]:

	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 [13]:
```
tags = pd.read_csv(r'C:\Users\velug\Downloads\rating\tag.csv')
```

In [15]:
```
tags
```

Out[15]:

	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
...	...	...	...	...
465559	138446	55999	dragged	2013-01-23 23:29:32
465560	138446	55999	Jason Bateman	2013-01-23 23:29:38
465561	138446	55999	quirky	2013-01-23 23:29:38
465562	138446	55999	sad	2013-01-23 23:29:32
465563	138472	923	rise to power	2007-11-02 21:12:47

465564 rows × 4 columns

In [17]:
```
del ratings['timestamp'] # here i delete the coloumn timestamp using function de
del tags['timestamp']    # here i delete the coloumn timestamp
```

In [19]:
```
ratings
```

Out[19]:

	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 [21]:
```
row_0 = tags.iloc[0] # Purely integer-location based indexing for selection by p

type(row_0)
```

Out[21]:
```
pandas.core.series.Series
```

In [23]:
```
print(row_0)
```

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

In [25]:
```
row_0.index
```

Out[25]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [27]:
```
row_0['userId']
```

Out[27]:
```
18
```

In [29]:
```
'rating' in row_0
```

Out[29]:
```
False
```

In [31]:
```
row_0.name
```

Out[31]:
```
0
```

In [33]:
```
row_0 = row_0.rename('firstRow')
row_0.name
```

Out[33]: 'firstRow'

In [35]:
```
ratings
```

Out[35]:

	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 [37]:
```
tags.head()
```

Out[37]:

	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 [39]:
```
tags.index
```

Out[39]: RangeIndex(start=0, stop=465564, step=1)

In [41]:
```
tags.columns
```

Out[41]: Index(['userId', 'movieId', 'tag'], dtype='object')

In [43]:
```
tags.iloc[ [0,11,500] ]
```

Out[43]:

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

In [45]:
```
ratings['rating'].describe() # the describe function  Generate descriptive stati
                             # Analyzes both numeric and object series, as well
                             #The output will vary depending on what is provided
```

Out[45]:
```
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 [47]:
```
ratings.describe()
```

Out[47]:

	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 [48]:
```
ratings['rating'].mean()
```

Out[48]:
```
3.5255285642993797
```

In [49]:
```
ratings.mean() # Return the mean of the values over the requested axis.
```

Out[49]:
```
userId     69045.872583
movieId     9041.567330
rating         3.525529
dtype: float64
```

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

Out[51]:
```
1.051988919275684
```

In [55]:
```
ratings.std() # Return sample standard deviation over requested axis.
```

Out[55]:
```
userId     40038.626653
movieId    19789.477445
rating         1.051989
dtype: float64
```

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

Out[57]:
```
0    4.0
Name: rating, dtype: float64
```

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

Out[59]:

	userId	movieId	rating
0	118205	296	4.0

In [60]:
```
ratings.corr()#Compute pairwise correlation of columns, excluding NA/null values
```

Out[60]:

	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 [61]:
```
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[61]:
```
False
```

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

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

In [67]:
```
movies.shape # Return a tuple representing the dimensionality of the DataFrame.
```

Out[67]:
```
(27278, 3)
```

In [69]:
```
movies.isnull() #DataFrame.isnull is an alias for DataFrame.isna.Detect missing
                #it gives us if there was any missing value
```

Out[69]:

	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 [71]:
```
movies.isna() # .isnull & isna perform same task.
```

Out[71]:

	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 [73]:
```
movies.isnull().any()
```

Out[73]:
```
movieId    False
title      False
genres     False
dtype: bool
```

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

Out[75]: (20000263, 3)

In [77]:
```
ratings.isnull()
```

Out[77]:

	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 [79]:
```
ratings.isnull().any()
```

Out[79]:
```
userId     False
movieId    False
rating     False
dtype: bool
```

In [81]:
```
ratings.isnull().any().any() # no null values
```

Out[81]: False

In [83]:
```
tags.shape
```

Out[83]: (465564, 3)

In [85]:
```
tags.isnull()
```

Out[85]:

	userId	movieId	tag
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
...	...	...	...
465559	False	False	False
465560	False	False	False
465561	False	False	False
465562	False	False	False
465563	False	False	False

465564 rows × 3 columns

In [87]:
```
tags.isnull().any() # here true is appear
```

Out[87]:
```
userId     False
movieId    False
tag         True
dtype: bool
```

In [89]:
```
tags.isnull().any().any() # (true)which means that we have some missing values
```

Out[89]:
```
True
```

In [91]:
```
tags=tags.dropna() # Remove missing values.
```

In [93]:
```
tags.isnull().any().any() # dropna() fuctions removes the missing values clean t
```

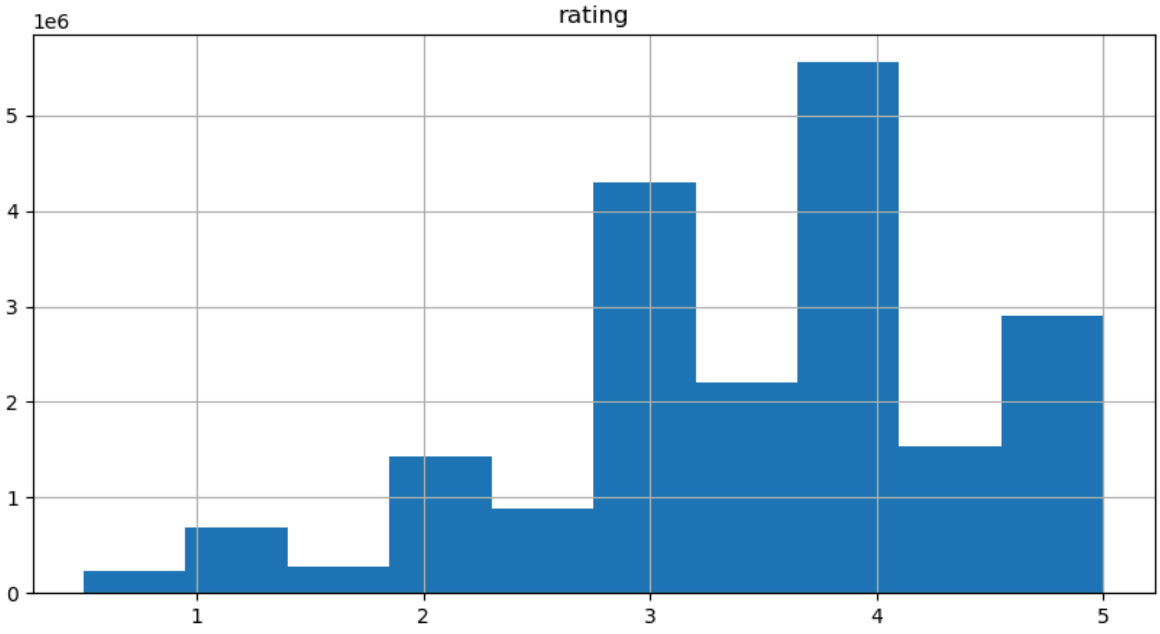
Out[93]:
```
False
```

In [95]:
```
tags.shape # before cleaning data tags.shape(465564, 3) data with missing values
           # after cleaning the data(465548, 3) data after removing missing valu
```

Out[95]:
```
(465548, 3)
```

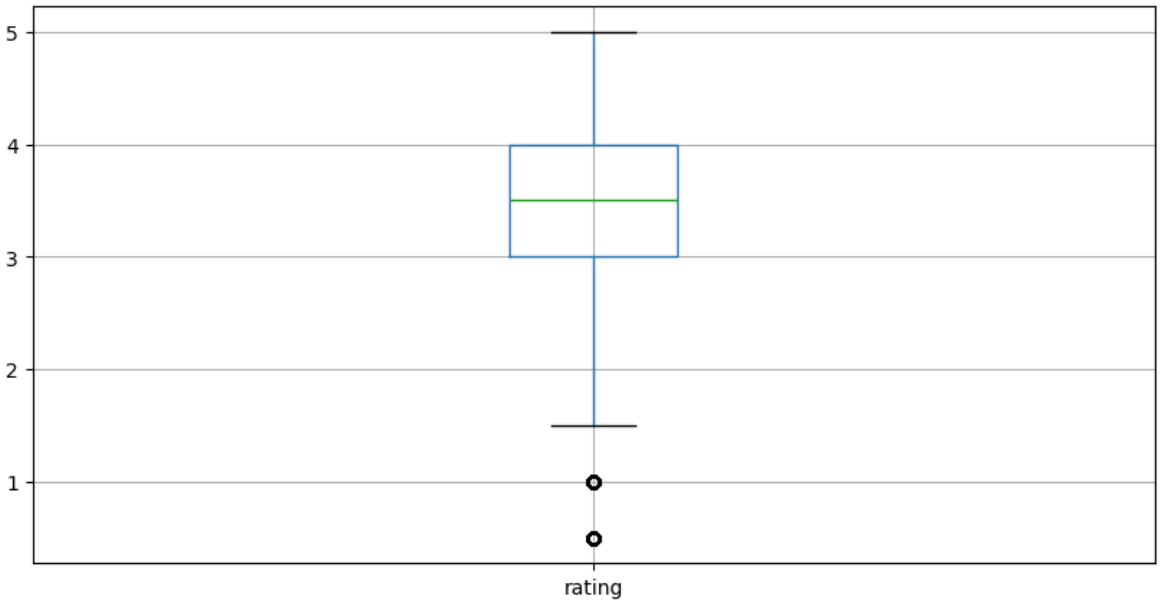
In [97]:
```
%matplotlib inline

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

Out[97]:
```
array([[<Axes: title={'center': 'rating'}>]], dtype=object)
```



In [98]:
```
ratings.boxplot(column='rating', figsize=(10,5)) # boxplot  A box plot is a meth
```

Out[98]: <Axes: >



In [100…
```
tags['tag'].head() #  Return the first `n(5)` rows.
```

Out[100…
```
0      Mark Waters
1        dark hero
2        dark hero
3    noir thriller
4        dark hero
Name: tag, dtype: object
```

In [101…
```
tags['tag'].tail() # Return the last `n(5)` rows.
```

Out[101…
```
465559          dragged
465560    Jason Bateman
465561           quirky
465562              sad
465563    rise to power
Name: tag, dtype: object
```

In [102…
```
movies.head()
```

Out[102…

	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 [103…
```
movies[['title','genres']]
```

Out[103…

	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
…	…	…
27273	Kein Bund für's Leben (2007)	Comedy
27274	Feuer, Eis & Dosenbier (2002)	Comedy
27275	The Pirates (2014)	Adventure
27276	Rentun Ruusu (2001)	(no genres listed)
27277	Innocence (2014)	Adventure|Fantasy|Horror

27278 rows × 2 columns

In [104…
```
movies[['title','genres']].head()
```

Out[104…

	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 [105…
```
ratings[-10:]
```

Out[105…

	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 [106…

```
ratings[10:] # if we use 10 : after the positive it begins
```

Out[106…

	userId	movieId	rating
10	1	293	4.0
11	1	296	4.0
12	1	318	4.0
13	1	337	3.5
14	1	367	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

20000253 rows × 3 columns

In [107…

```
ratings[:10] # if we use : 10 after the negaitive it ends upto the index value 1
```

Out[107…

	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 [108…

```
tag_counts = tags['tag'].value_counts()
tag_counts[-10:]
```

Out[108…

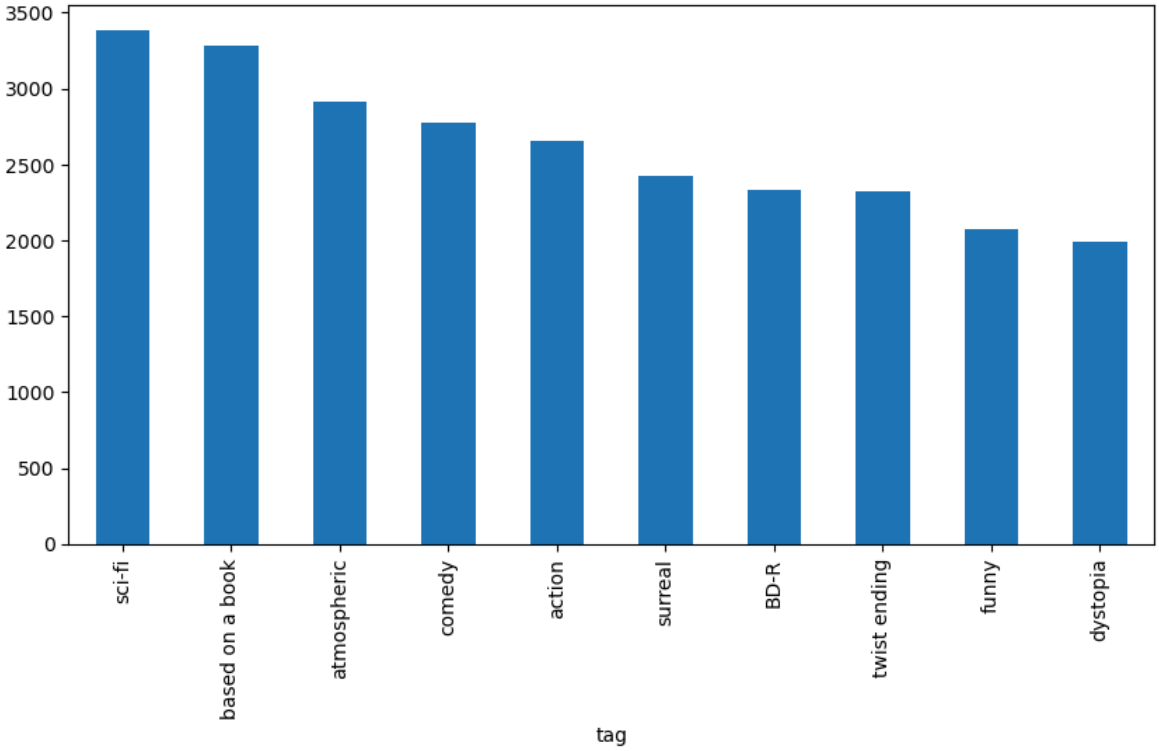
```
tag
missing child                  1
Ron Moore                      1
Citizen Kane                   1
mullet                         1
biker gang                     1
Paul Adelstein                 1
the wig                        1
killer fish                    1
genetically modified monsters  1
topless scene                  1
Name: count, dtype: int64
```

In [109…

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

Out[109…

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


3500
3000
2500
2000
1500
1000
500
0
sci-fi
based on a book
atmospheric
comedy
action
surreal
BD-R
twist ending
funny
dystopia
tag

In [ ]: