In [6]:
```
%%html
<style>
@import url('https://fonts.googleapis.com/css?family=Ewert|Roboto&effect=3d|');
body {background-color: gainsboro;}
a {color: #37c9e1; font-family: 'Roboto';}
h1 {color: #37c9e1; font-family: 'Orbitron'; text-shadow: 4px 4px 4px #aaa;}
h2, h3 {color: slategray; font-family: 'Orbitron'; text-shadow: 4px 4px 4px #aaa;}
h4 {color: #818286; font-family: 'Roboto';}
span {font-family:'Roboto'; color:black; text-shadow: 5px 5px 5px #aaa;}
div.output_area pre{font-family:'Roboto'; font-size:110%; color:lightblue;}
</style>
```

Pandas with Data Science

https://www.kaggle.com/code/harunshimanto/pandas-with-data-science-ai

In [8]:
```
import pandas as pd
```

In [9]:
```
movies = pd.read_csv(r'Datasets/link.csv') # reading the datasets
```

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

Out[10]:
```
Index(['movieId', 'imdbId', 'tmdbId'], dtype='object')
```

In [11]:
```
movies.head()
```

Out[11]:

	movieId	imdbId	tmdbId
0	1	114709	862.0
1	2	113497	8844.0
2	3	113228	15602.0
3	4	114885	31357.0
4	5	113041	11862.0

In [13]:
```
tags = pd.read_csv(r'Datasets/tag.csv', sep = ',')
print(type(tags))
tags.head(10)
```

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

Out[13]:

	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
5	65	668	bollywood	2013-05-10 01:37:56
6	65	898	screwball comedy	2013-05-10 01:42:40
7	65	1248	noir thriller	2013-05-10 01:39:43
8	65	1391	mars	2013-05-10 01:40:55
9	65	1617	neo-noir	2013-05-10 01:43:37

In [14]:
```
ratings = pd.read_csv(r'Datasets/rating.csv', sep = ',', parse_dates=['timestamp'])
ratings.head(10)
```

Out[14]:

	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
5	1	112	3.5	2004-09-10 03:09:00
6	1	151	4.0	2004-09-10 03:08:54
7	1	223	4.0	2005-04-02 23:46:13
8	1	253	4.0	2005-04-02 23:35:40
9	1	260	4.0	2005-04-02 23:33:46

Series

In [15]:
```
row_0 = tags.iloc[0]
print(type(row_0))
print(row_0)
```

```
<class 'pandas.core.series.Series'>
userId                        18
movieId                     4141
tag                 Mark Waters
timestamp    2009-04-24 18:19:40
Name: 0, dtype: object
```

In [16]:
```
row_0.index
```

Out[16]:
```
Index(['userId', 'movieId', 'tag', 'timestamp'], dtype='object')
```

In [17]:
```
'rating' in row_0
```

Out[17]:
```
False
```

In [18]:
```
row_0.name
```

Out[18]:
```
0
```

In [19]:
```
row_0 = row_0.rename('first row')
row_0.name
```

Out[19]:
```
'first row'
```

Data Frames

In [21]:
```
tags.head()
```

Out[21]:

	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 [22]:
```
tags.index
```

Out[22]:
```
RangeIndex(start=0, stop=465564, step=1)
```

In [23]:
```
tags.columns
```

Out[23]:
```
Index(['userId', 'movieId', 'tag', 'timestamp'], dtype='object')
```

In [26]:
```
tags.iloc[[2,207,50000, 200000]]
```

Out[26]:

	userId	movieId	tag	timestamp
2	65	353	dark hero	2013-05-10 01:41:19
207	129	60756	ending	2011-02-08 05:55:56
50000	11081	1253	eerie	2007-03-02 22:28:47
200000	58612	27611	reboot	2013-03-31 06:06:14

Descriptive statistics

In [27]:
```
ratings.describe()
```

Out[27]:

	userId	movieId	rating	timestamp
count	2.000026e+07	2.000026e+07	2.000026e+07	20000263
mean	6.904587e+04	9.041567e+03	3.525529e+00	2004-11-20 02:32:01.677113984
min	1.000000e+00	1.000000e+00	5.000000e-01	1995-01-09 11:46:44
25%	3.439500e+04	9.020000e+02	3.000000e+00	2000-08-20 18:55:45
50%	6.914100e+04	2.167000e+03	3.500000e+00	2004-12-20 15:18:06
75%	1.036370e+05	4.770000e+03	4.000000e+00	2008-11-02 16:11:57.500000
max	1.384930e+05	1.312620e+05	5.000000e+00	2015-03-31 06:40:02
std	4.003863e+04	1.978948e+04	1.051989e+00	NaN

In [28]:
```
ratings.columns
```

Out[28]:
```
Index(['userId', 'movieId', 'rating', 'timestamp'], dtype='object')
```

In [30]:
```
ratings['rating'].describe()
```

Out[30]:
```
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 [31]:
```
ratings.mean()
```

Out[31]:
```
userId                          69045.872583
movieId                          9041.56733
rating                             3.525529
timestamp    2004-11-20 02:32:01.677113984
dtype: object
```

In [33]:
```
ratings['rating'].min()
```

Out[33]:
```
np.float64(0.5)
```

In [34]:
```
ratings['rating'].max()
```

Out[34]:
```
np.float64(5.0)
```

In [35]:
```
ratings['rating'].std()
```

Out[35]:
```
np.float64(1.0519889192942418)
```

In [36]:
```
ratings['rating'].mode()
```

Out[36]:
```
0    4.0
Name: rating, dtype: float64
```

In [37]:
```
ratings.corr()
```

Out[37]:

	userId	movieId	rating	timestamp
userId	1.000000	-0.000850	0.001175	-0.003101
movieId	-0.000850	1.000000	0.002606	0.459096
rating	0.001175	0.002606	1.000000	-0.000512
timestamp	-0.003101	0.459096	-0.000512	1.000000

In [38]:
```
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[38]: np.False_

In [40]:
```
filter2 = ratings['rating'] > 0
print(filter2)
filter2.any()
```

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

Out[40]: np.True_

Data cleaning

In [41]:
```
movies.shape
```

Out[41]: (27278, 3)

In [43]:
```
movies.isnull().any().any()  # Checks for any true value in columns returned by isnull()
```

Out[43]:
```
np.True_
```

In [42]:
```
movies.isnull().any()  # To check which column has the true values returned by isnull()
```

Out[42]:
```
movieId    False
imdbId     False
tmdbId      True
dtype: bool
```

In [44]:
```
ratings.shape
```

Out[44]:
```
(20000263, 4)
```

In [49]:
```
ratings.isnull().any().any()
```

Out[49]:
```
np.False_
```

In [47]:
```
tags.isnull().any().any()
```

Out[47]:
```
np.True_
```

In [48]:
```
tags.isnull().any()
```

Out[48]:
```
userId       False
movieId      False
tag           True
timestamp    False
dtype: bool
```

In [50]:
```
# Drop the null values in tags and movies
tags = tags.dropna()
movies = movies.dropna()
```

In [53]:
```
tags.isnull().any().any()
```

Out[53]:
```
np.False_
```

In [51]:
```
tags.shape
```

Out[51]:
```
(465548, 4)
```

In [54]:
```
movies.isnull().any().any()
```

Out[54]:
```
np.False_
```

In [52]:
```
movies.shape
```

Out[52]:
```
(27026, 3)
```

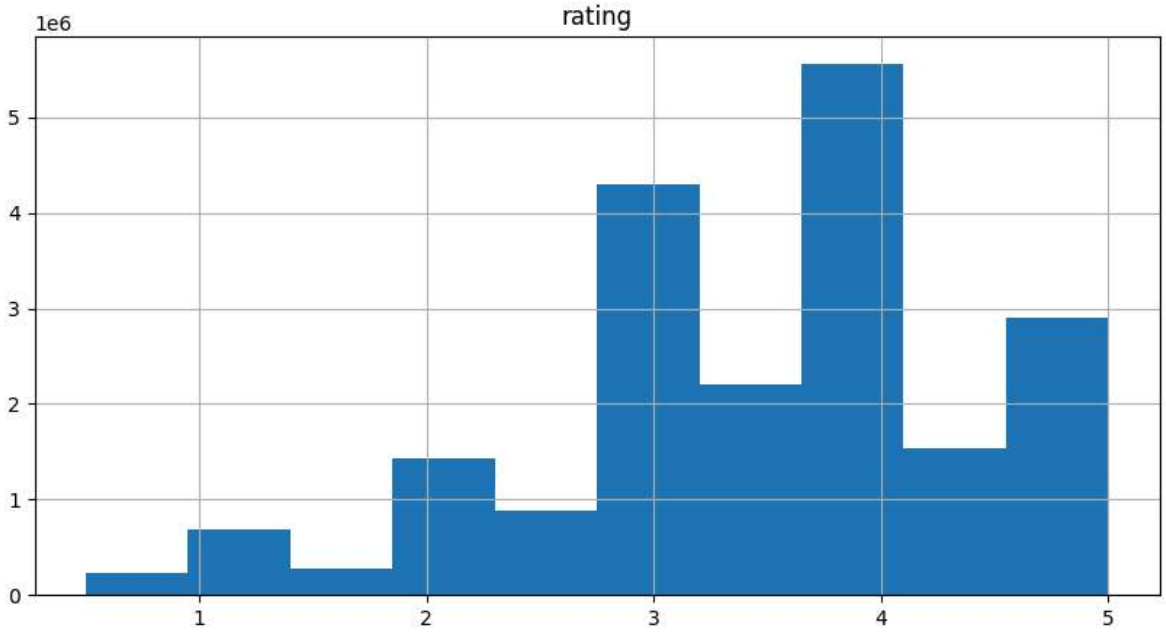
The number of rows have reduced in both tags and movies

Data visualization using matplotlib

In [57]:
```
%matplotlib inline
```

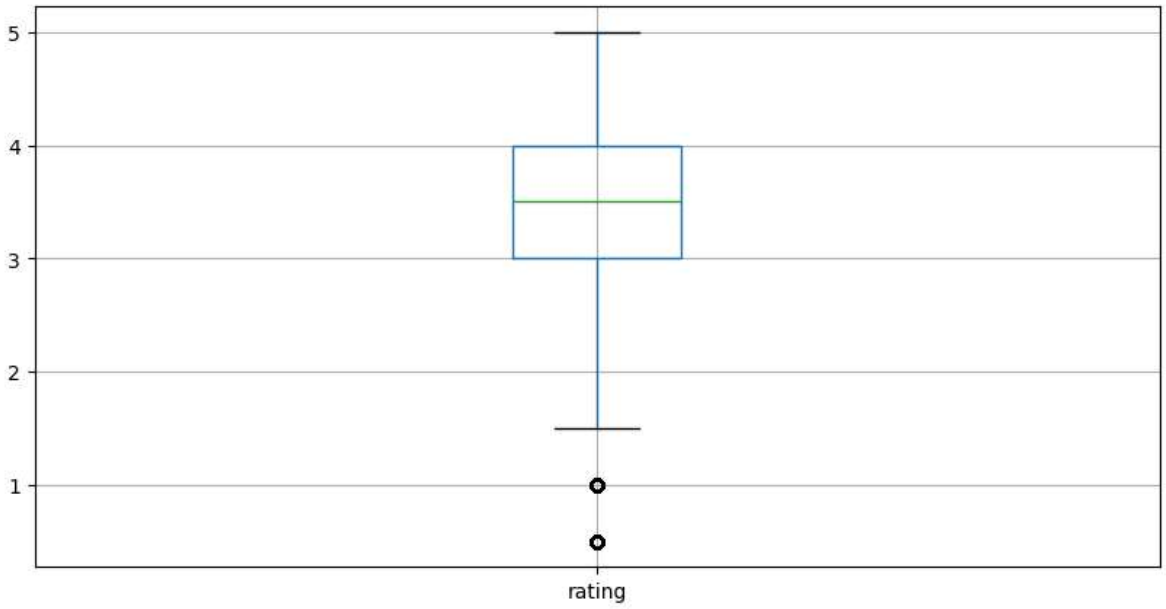
```
ratings.hist(column = 'rating' , figsize=(10,5))
```

Out[57]:
```
array([[<Axes: title={'center': 'rating'}>]], dtype=object)
```



In [58]:
```
ratings.boxplot(column = 'rating' , figsize=(10,5))
```

Out[58]:
```
<Axes: >
```



Slicing out columns

In [59]:
```
tags['tag'].head()
```

Out[59]:
```
0      Mark Waters
1        dark hero
2        dark hero
3    noir thriller
4        dark hero
Name: tag, dtype: object
```

In [61]:
```
movies.columns
```

Out[61]:
```
Index(['movieId', 'imdbId', 'tmdbId'], dtype='object')
```

In [62]:
```
movies[['movieId','imdbId']].head()
```

Out[62]:

	movieId	imdbId
0	1	114709
1	2	113497
2	3	113228
3	4	114885
4	5	113041

In [63]:
```
ratings[-10:]
```

Out[63]:

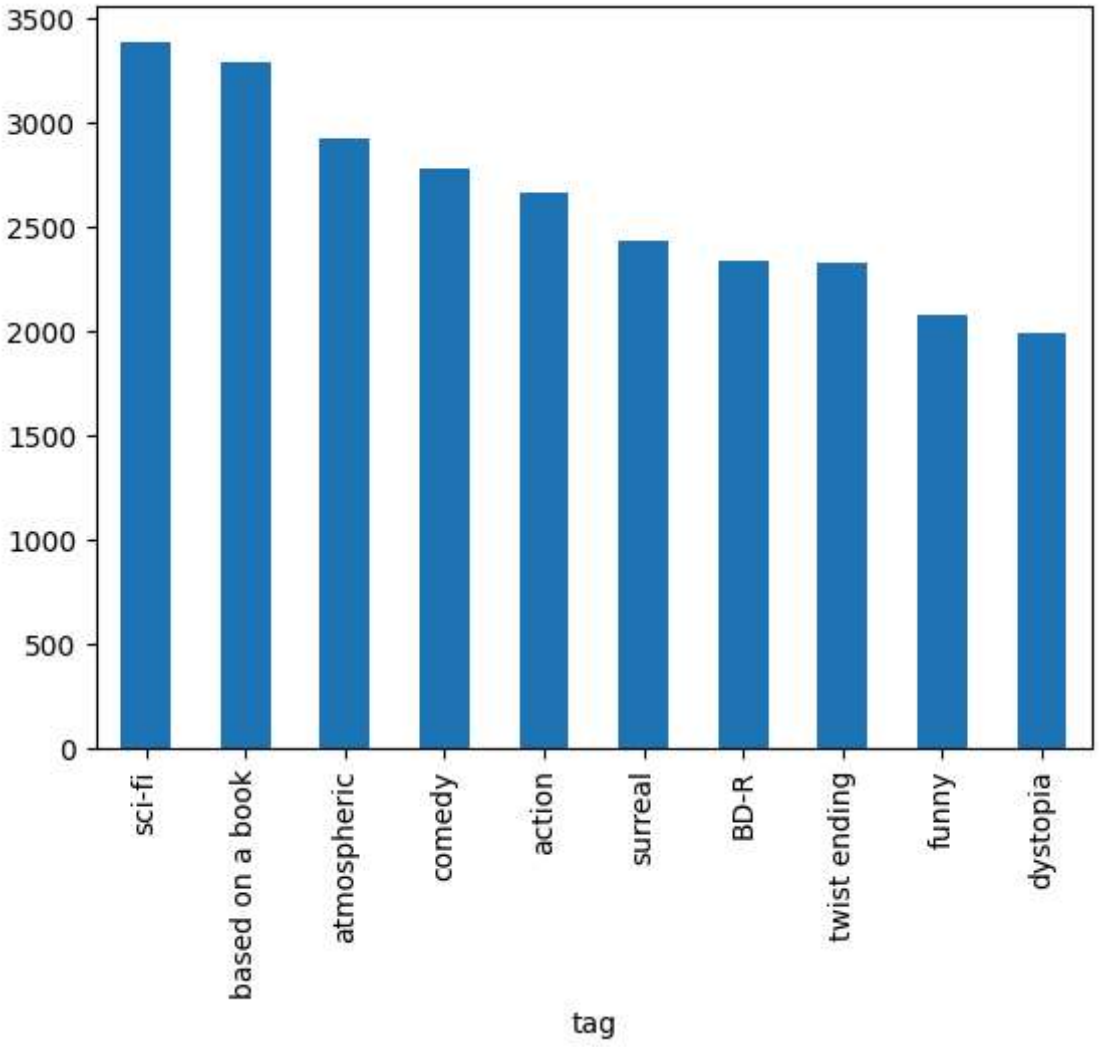
	userId	movieId	rating	timestamp
20000253	138493	60816	4.5	2009-12-03 18:32:43
20000254	138493	61160	4.0	2009-11-16 16:55:37
20000255	138493	65682	4.5	2009-10-17 21:52:53
20000256	138493	66762	4.5	2009-10-17 18:50:08
20000257	138493	68319	4.5	2009-12-07 18:15:20
20000258	138493	68954	4.5	2009-11-13 15:42:00
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

In [65]:
```
tag_counts = tags['tag'].value_counts()
tag_counts
```

Out[65]:
```
tag
sci-fi                3384
based on a book       3281
atmospheric           2917
comedy                2779
action                2657
                      ...
Diamond Dallas Page      1
I'm Devon Butler!        1
No arguement             1
Really Bad               1
Botox                    1
Name: count, Length: 38643, dtype: int64
```

In [68]:
```
tag_counts[:10].plot(kind = 'bar')
```

Out[68]: <Axes: xlabel='tag'>



In [ ]: