KAGGLE PROJECT (MOVIES ANALYSISES RATING)

In [1]:
```
import pandas as pd       # here importing libraries
```

In [2]:
```
 # READ THE DATASET
 # in this notebeek , we will be using 3 CSV files
 # RATINGS.CSV : userId,movieId,rating, timestamp
 # TAGS.CSV : userId,movieId, tag, timestamp
 # MOVIES.CSV : movieId, title, genres
```

In [4]:
```
movies = pd.read_csv(r'C:\Users\Hanshu\Desktop\EXCEL 1 kaggale\movie.csv')
movies
```

Out[4]:

	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 [7]:
```
movies.head(20)  # (0-19 records displyed)
```

Out[7]:

	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
5	6	Heat (1995)	Action|Crime|Thriller
6	7	Sabrina (1995)	Comedy|Romance
7	8	Tom and Huck (1995)	Adventure|Children
8	9	Sudden Death (1995)	Action
9	10	GoldenEye (1995)	Action|Adventure|Thriller
10	11	American President, The (1995)	Comedy|Drama|Romance
11	12	Dracula: Dead and Loving It (1995)	Comedy|Horror
12	13	Balto (1995)	Adventure|Animation|Children
13	14	Nixon (1995)	Drama
14	15	Cutthroat Island (1995)	Action|Adventure|Romance
15	16	Casino (1995)	Crime|Drama
16	17	Sense and Sensibility (1995)	Drama|Romance
17	18	Four Rooms (1995)	Comedy
18	19	Ace Ventura: When Nature Calls (1995)	Comedy
19	20	Money Train (1995)	Action|Comedy|Crime|Drama|Thriller

In [8]:
```
tags = pd.read_csv(r'C:\Users\Hanshu\Desktop\EXCEL 1 kaggale\tag.csv')
tags
```

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
...	...	...	...	...
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 [9]:
```
tags.head()    # by default 5 records (0-4)
```

Out[9]:

	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 [11]:
```
ratings = pd.read_csv(r'C:\Users\Hanshu\Desktop\EXCEL 1 kaggale\rating.csv')
ratings
```

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
...	...	...	...	...
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 [12]:

```
del ratings['timestamp']
del tags['timestamp']
```

In [16]:

```
ratings       # here timestamp remove bcz above step we are using DEL
```

Out[16]:

	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 [15]:

```
tags.head()
```

Out[15]:

	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 [21]:
```
row_0 = tags.iloc[0]      # in TAGS , first record information
row_0                      # iloc[0] -- first record infor (oth index)
                          # get output in the form of pandas SERIES
```

Out[21]:
```
userId               18
movieId            4141
tag         Mark Waters
Name: 0, dtype: object
```

In [18]:
```
type(row_0)
```

Out[18]:
```
pandas.core.series.Series
```

In [19]:
```
print(row_0)
```

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

In [23]:
```
row_0.index   # we get columns names of pands SERIES
```

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

In [24]:
```
row_0['userId']
```

Out[24]:
```
18
```

In [26]:
```
'rating' in row_0    # here used membership operators
```

Out[26]:
```
False
```

In [27]:
```
row_0.name
```

Out[27]:
```
0
```

In [29]:
```
row_0 = row_0.rename('firstROW')    # here rename the na,e as 'firstROW'
row_0
```

Out[29]:
```
userId                18
movieId             4141
tag          Mark Waters
Name: firstROW, dtype: object
```

DataFrames

In [30]:
```
tags.head()
```

Out[30]:

	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 [32]:
```
tags.index
```

Out[32]:
```
RangeIndex(start=0, stop=465564, step=1)
```

In [33]:
```
tags.columns
```

Out[33]:
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [34]:
```
tags.iloc[ [0,11,500] ]
```

Out[34]:

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

Descriptive Statistics

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

Out[35]:
```
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 [36]:
```
ratings.describe()
```

Out[36]:

	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 [39]:
```
ratings['rating'].mean()
```
Out[39]:
```
3.5255285642993797
```

In [40]:
```
ratings.mean()
```
Out[40]:
```
userId      69045.872583
movieId      9041.567330
rating          3.525529
dtype: float64
```

In [41]:
```
ratings['rating'].min()
```
Out[41]:
```
0.5
```

In [42]:
```
ratings['rating'].max()
```
Out[42]:
```
5.0
```

In [43]:
```
ratings['rating'].std()
```
Out[43]:
```
1.051988919275684
```

In [44]:
```
ratings['rating'].mode()
```
Out[44]:
```
0    4.0
Name: rating, dtype: float64
```

In [45]:
```
ratings.corr()
```
Out[45]:

	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 [47]:
```
filter1 = ratings['rating'] > 10
filter1
```

```
Out[47]: 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 [49]: 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[49]: False
```

```
In [50]: filter2 = ratings['rating'] > 0
         filter2.all()
```

```
Out[50]: True
```

Data Cleaning : Handling Missing Data

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

```
Out[51]: (27278, 3)
```

```
In [53]: movies.isnull().any().any()  # NO NULL Values
```

```
Out[53]: False
```

```
In [54]: ratings.shape
```

```
Out[54]: (20000263, 3)
```

```
In [55]: tags.shape
```

```
Out[55]: (465564, 3)
```

```
In [57]: tags.isnull().any().any()    # here NULL Values will have
```

```
Out[57]:  True
```

```
In [58]:  tags = tags.dropna()
```

```
In [59]:  tags.isnull().any().any()   # now no NULL values BCZ in above step we are droppi
```

```
Out[59]:  False
```

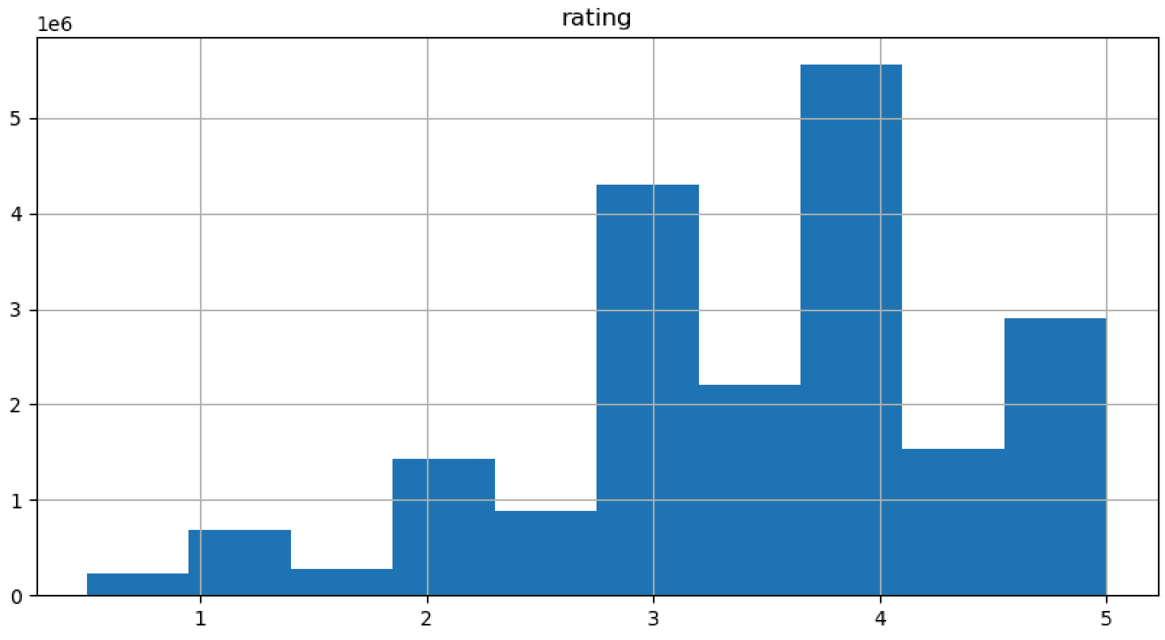
```
In [60]:  tags.shape  # NO NULL VALUES ,check before step(tags.shape) & now this step numb
```

```
Out[60]:  (465548, 3)
```

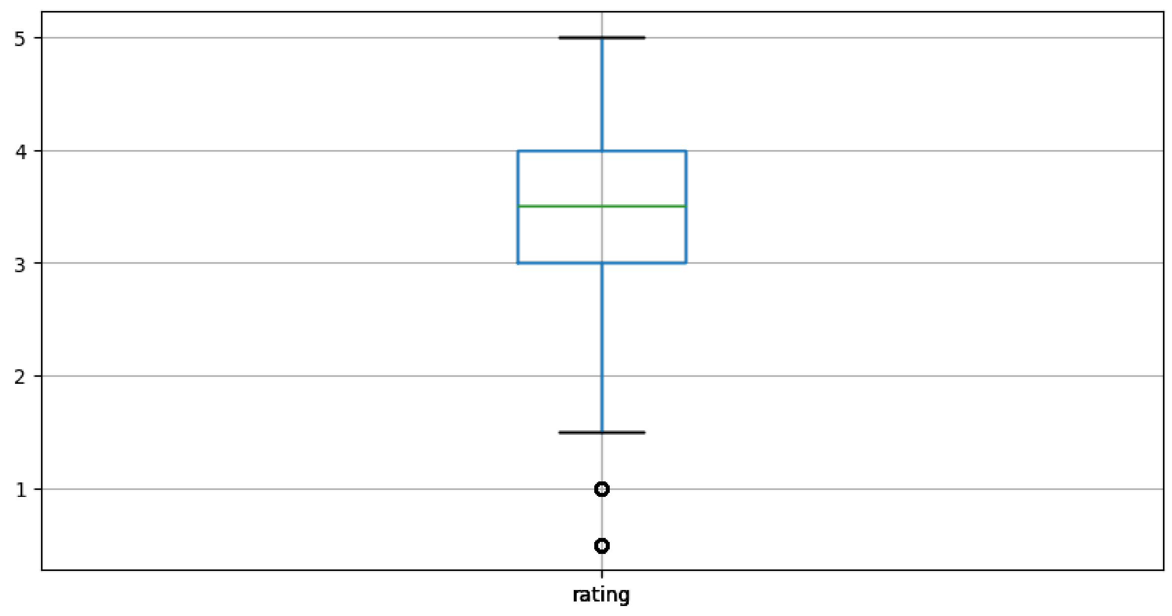
Data Visualization

```
In [69]:  import matplotlib.pyplot as plt
          %matplotlib inline

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



```
In [72]:  ratings.boxplot(column = 'rating' , figsize=(10,5))
          plt.show()
```



Slicing Out Columns

In [77]:
```
tags['tag'].head()
```

Out[77]:
```
0       Mark Waters
1         dark hero
2         dark hero
3     noir thriller
4         dark hero
Name: tag, dtype: object
```

In [81]:
```
movies[['title','genres']] . head()
```

Out[81]:

	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 [82]:
```
ratings[-10:]
```

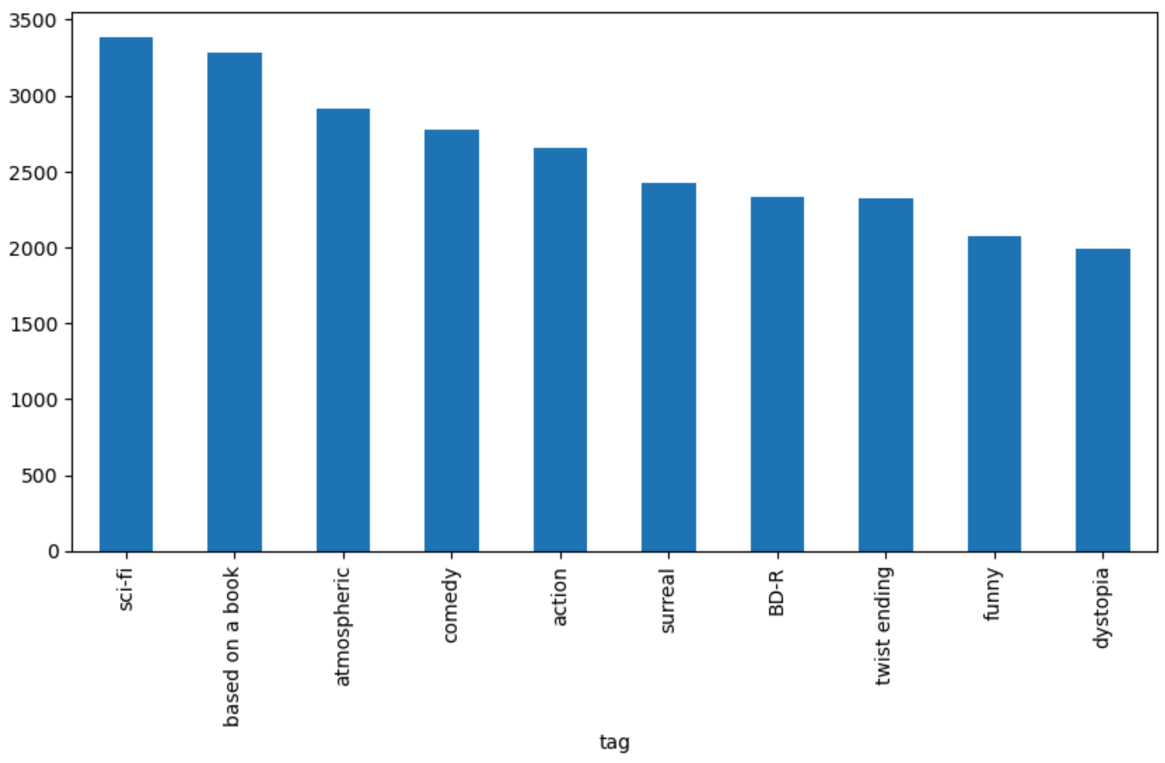
Out[82]:

	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 [83]:
```
tag_counts = tags['tag'].value_counts()
tag_counts[-10:]
```

Out[83]:
```
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 [89]:
```
tag_counts[:10].plot(kind = 'bar' , figsize= (10,5))
plt.show()
```



Filter for Selecting Rows

```
In [90]: is_high_rated = ratings['rating'] >= 5.0
         ratings[is_high_rated][30:50]
```

Out[90]:

	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 [92]:
```
is_action = movies['genres'].str.contains('Action')
movies[is_action][5:15]
```

Out[92]:

	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 [93]:
```
movies[is_action].head(15)
```

Out[93]:

	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 [94]:
```
ratings_count = ratings[['movieId','rating']].groupby('rating').count()
ratings_count
```

Out[94]:

	movieId
rating	
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 [97]:
```
average_rating = ratings[['movieId', 'rating']].groupby('movieId').count()
average_rating.head()
```

Out[97]:

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

In [98]:
```
average_rating = ratings[['movieId','rating']].groupby('movieId').mean()
average_rating.head()
```

Out[98]:

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

In [99]:
```
movie_count = ratings[['movieId','rating']].groupby('movieId').count()
movie_count.head()
```

Out[99]:

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

In [100…

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

Out[100…

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

Merge Dataframes

In [101…

```
tags.head()
```

Out[101…

	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 [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…

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

Out[103…

	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 [106…

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

Out[106…

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

In [ ]: