In [ ]:
```
## imdb
```

In [1]:
```
import pandas as pd
```

In [57]:
```
ratings=pd.read_csv(r"C:\Users\91630\Downloads\rating.csv")
```

In [58]:
```
ratings
```

Out[58]:

	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 [85]:
```
tags=pd.read_csv(r"C:\Users\91630\Downloads\tag.csv",sep=',')
```

In [87]:
```
tags
```

Out[87]:

	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 [91]:
```
movies=pd.read_csv(r"C:\Users\91630\Downloads\movie.csv",sep=',')
```

In [93]:
```
movies
```

Out[93]:

	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 [83]:
```
print(type(movies))
```

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

In [71]:
```
print(type(ratings))
```

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

In [73]:
```
print(type(tags))
```

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

In [75]:
```
movies.head(20)
```

Out[75]:

	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 [77]:
```
tags.head()
```

Out[77]:

	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 [95]:
```
rating.head()
```

Out[95]:

	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 [97]:
```
del ratings['timestamp']
del tags['timestamp']
```

data structures

In [100…
```
row_0=tags.iloc[0]
type(row_0)
```

Out[100…
```
pandas.core.series.Series
```

In [102…
```
print(row_0)
```

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

In [104…
```
row_0.index
```

Out[104…
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [106…
```
row_0['userId']
```

Out[106…
```
18
```

In [108…
```
'rating' in row_0
```

Out[108…
```
False
```

In [110…
```
row_0.name
```

Out[110…
```
0
```

In [112…
```
row_0=row_0.rename('firstRow')
row_0.name
```

Out[112…
```
'firstRow'
```

dataframes

In [115…
```
tags.head()
```

Out[115…

	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 [117…
```
tags.index
```

Out[117…
```
RangeIndex(start=0, stop=465564, step=1)
```

In [119…
```
tags.columns
```

Out[119…
```
Index(['userId', 'movieId', 'tag'], dtype='object')
```

In [121…
```
tags.iloc[[0,11,500]]
```

Out[121…

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

Descriptive statistics

In [124…
```
ratings['rating'].describe()
```

Out[124…
```
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 [126...]
```
ratings.describe()
```

Out[126...]

	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 [128...]
```
ratings['rating'].mean()
```

Out[128...]
```
3.5255285642993797
```

In [130...]
```
ratings.mean()
```

Out[130...]
```
userId     69045.872583
movieId     9041.567330
rating         3.525529
dtype: float64
```

In [132...]
```
ratings['rating'].max()
```

Out[132...]
```
5.0
```

In [134...]
```
ratings['rating'].std()
```

Out[134...]
```
1.0519889192756 84
```

In [136...]
```
ratings['rating'].mode()
```

Out[136...]
```
0    4.0
Name: rating, dtype: float64
```

In [138...]
```
ratings.corr()
```

Out[138...]

	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 [142...]
```
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[142… False

In [144…
```
filter2=ratings['rating']>0
filter2.all()
```

Out[144… True

data cleaning:handling missing data

In [151…
```
movies.shape
```

Out[151… (27278, 3)

In [153…
```
movies.isnull().any().any()
```

Out[153… False

. thats nice!no null values!

In [156…
```
ratings.shape
```

Out[156… (20000263, 3)

In [158…
```
ratings.isnull().any().any()
```

Out[158… False

In [160…
```
tags.shape
```

Out[160… (465564, 3)

In [162…
```
tags.isnull().any().any()
```

Out[162… True

. we have some tags which are NULL.

In [165…
```
tags=tags.dropna()
```

In [167…
```
tags.isnull().any().any()
```

Out[167… False

In [169…
```
tags.shape
```

Out[169…
```
(465548, 3)
```

. thats nice! No NULL values! Notice the njmber of lines have reduced.

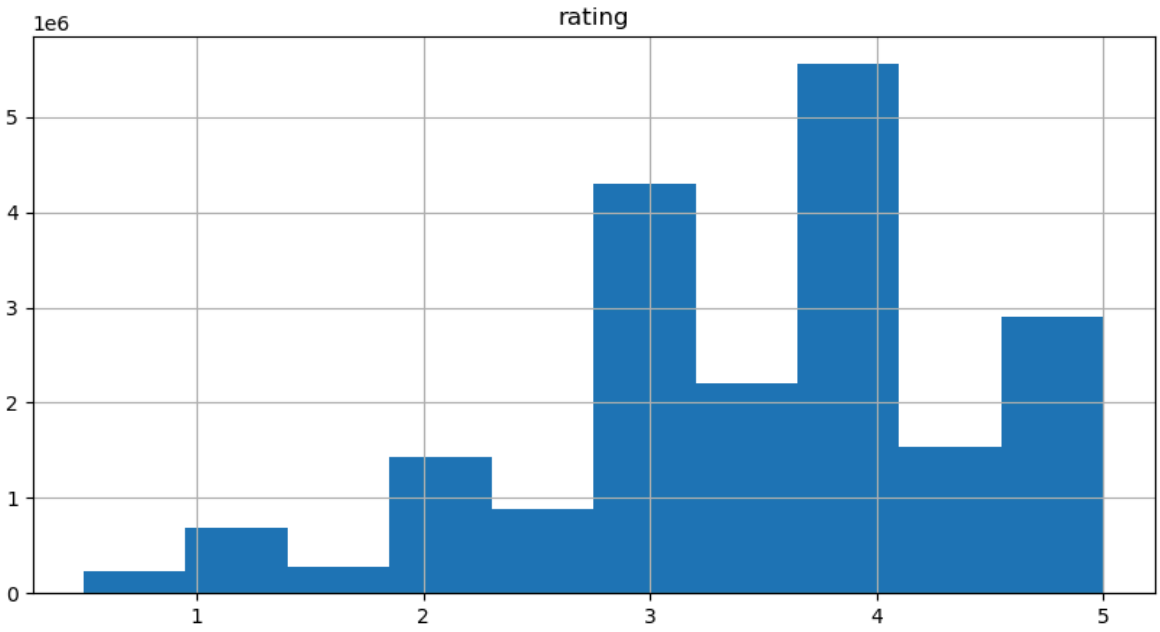
data viisualization

In [187…
```
import matplotlib.pyplot as plt
```

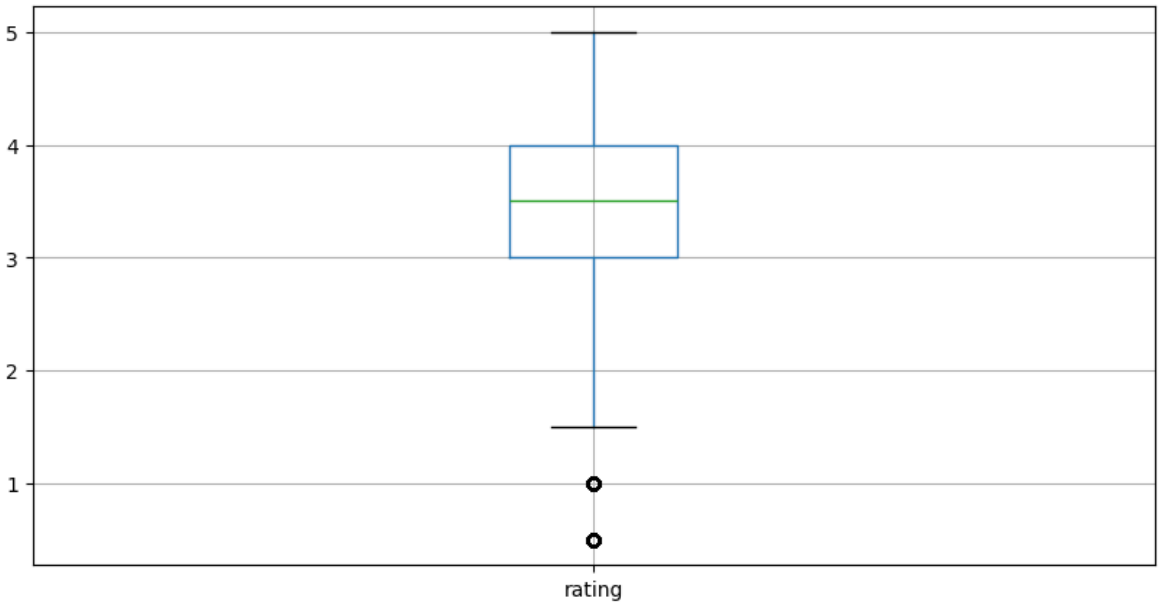
In [195…
```
%matplotlib inline
ratings.hist(column='rating',figsize=(10,5))
```

Out[195…
```
array([[<Axes: title={'center': 'rating'}>]], dtype=object)
```

In [197…
```
plt.show()
```



In [209…
```
ratings.boxplot(column='rating',figsize=(10,5))
plt.show()
```



slicing out columns

In [214…
```
tags['tag'].head()
```

Out[214…
```
0      Mark Waters
1        dark hero
2        dark hero
3    noir thriller
4        dark hero
Name: tag, dtype: object
```

In [216…
```
movies[['title','genres']].head()
```

Out[216…

	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 [218…
```
rating[-10:]
```

Out[218…

	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 [220…

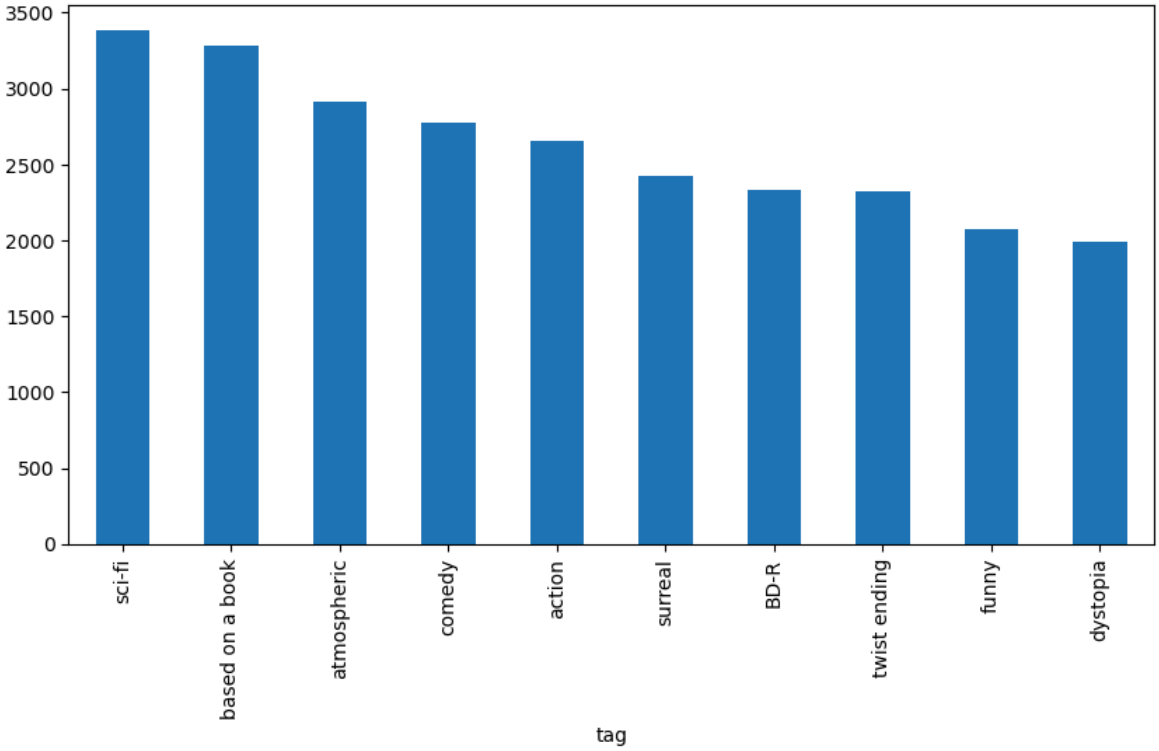
```
tag_counts=tags['tag'].value_counts()
tag_counts[-10:]
```

Out[220…

```
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 [228…

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



In [230…

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

Out[230…

	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 [232…

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

Out[232...

	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 [234...

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

Out[234...

	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 snd Aggregate

In [237...

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

Out[237...

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 [243...

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

Out[243...

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

In [245...

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

Out[245...

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

In [249...

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

Out[249...

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

merge dataframes

In [252...
```
tags.head()
```

Out[252...

	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 [254...
```
movies.head()
```

Out[254...

	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 [256...
```
t=movies.merge(tags,on='movieId',how='inner')
t.head()
```

Out[256...

	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

combine aggreagation,merging,and filters to get useful analytics

In [267...

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

Out[267...

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

In [269...

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

Out[269...

	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 [271...

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

Out[271...

	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 [273...

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

Out[273...

	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 [275...

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

Out[275...

	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 [278...

```
movies.head()
```

Out[278...

	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 [281...
```
movie_genres=movies['genres'].str.split('l',expand=True)
```

In [283...
```
movie_genres[:10]
```

Out[283...

	0	1	2	3	4
0	Adventure|Animation|Chi	dren|Comedy|Fantasy	None	None	None
1	Adventure|Chi	dren|Fantasy	None	None	None
2	Comedy|Romance	None	None	None	None
3	Comedy|Drama|Romance	None	None	None	None
4	Comedy	None	None	None	None
5	Action|Crime|Thri		er	None	None
6	Comedy|Romance	None	None	None	None
7	Adventure|Chi	dren	None	None	None
8	Action	None	None	None	None
9	Action|Adventure|Thri		er	None	None

add a new column for comedy genre flag

In [288...
```
movie_genres['isComedy']=movies['genres'].str.contains('Comedy')
```

In [290...
```
movie_genres[:10]
```

Out[290…

	0	1	2	3	4	isComedy
0	Adventure|Animation|Chi	dren|Comedy|Fantasy	None	None	None	True
1	Adventure|Chi	dren|Fantasy	None	None	None	False
2	Comedy|Romance	None	None	None	None	True
3	Comedy|Drama|Romance	None	None	None	None	True
4	Comedy	None	None	None	None	True
5	Action|Crime|Thri	er	None	None	None	False
6	Comedy|Romance	None	None	None	None	True
7	Adventure|Chi	dren	None	None	None	False
8	Action	None	None	None	None	False
9	Action|Adventure|Thri	er	None	None	None	False

Extract year from title e.g.(2007)

In [299…
```
movies['year']=movies['title'].str.extract('.*\((.*)\).*',expand=True)
```

```
<>:1: SyntaxWarning: invalid escape sequence '\('
<>:1: SyntaxWarning: invalid escape sequence '\('
C:\Users\91630\AppData\Local\Temp\ipykernel_19688\3287904146.py:1: SyntaxWarning:
invalid escape sequence '\('
  movies['year']=movies['title'].str.extract('.*\((.*)\).*',expand=True)
```

In [302…
```
movies.tail()
```

Out[302…

	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 [309…
```
tags.dtypes
```

Out[309…
```
userId      int64
movieId     int64
tag        object
dtype: object
```

In [311…
```
tags.head(5)
```

Out[311…

	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 [313…

```
tags['parsed_time']=pd.to_datetime(tags['timestamp'],unit='s')
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3805, in Index.get
_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

KeyError: 'timestamp'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[313], line 1
----> 1 tags['parsed_time']=pd.to_datetime(tags['timestamp'],unit='s')

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:4102, in DataFrame.__geti
tem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3812, in Index.get
_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'timestamp'
```

In [317…

```
tags['parsed_time'].dtype
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3805, in Index.get
_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

KeyError: 'parsed_time'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[317], line 1
----> 1 tags['parsed_time'].dtype

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:4102, in DataFrame.__geti
tem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3812, in Index.get
_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'parsed_time'
```

In [319…

```
tags.head(2)
```

Out[319…

	userId	movieId	tag
0	18	4141	Mark Waters
1	65	208	dark hero

In [333…

```
greater_than_t=tags['parsed_time']>'2015-02-01'
selected_rows=tags[greater_than_t]
tags.shape,selected_rows.shape
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3805, in Index.get
_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

KeyError: 'parse_time'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[333], line 1
----> 1 greater_than_t=tags['parse_time']>'2015-02-01'
      2 selected_rows=tags[greater_than_t]
      3 tags.shape,selected_rows.shape

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:4102, in DataFrame.__geti
tem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3812, in Index.get
_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'parse_time'
```

In [335…

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

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_19688\3410031928.py in ?()
----> 1 tags.sort_values(by='parsed_time', ascending=True)[:10]

~\anaconda3\Lib\site-packages\pandas\core\frame.py in ?(self, by, axis, ascendin
g, inplace, kind, na_position, ignore_index, key)
   7185             )
   7186         elif len(by):
   7187             # len(by) == 1
   7188
-> 7189             k = self._get_label_or_level_values(by[0], axis=axis)
   7190
   7191             # need to rewrap column in Series to apply key function
   7192             if key is not None:

~\anaconda3\Lib\site-packages\pandas\core\generic.py in ?(self, key, axis)
   1907             values = self.xs(key, axis=other_axes[0])._values
   1908         elif self._is_level_reference(key, axis=axis):
   1909             values = self.axes[axis].get_level_values(key)._values
   1910         else:
-> 1911             raise KeyError(key)
   1912
   1913         # Check for duplicates
   1914         if values.ndim > 1:

KeyError: 'parsed_time'
```

In [337…

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

Out[337…

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

In [339…

```
joined = movies.merge(average_rating, on='movieId', how='inner')
joined.head()
joined.corr()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[339], line 3
      1 joined = movies.merge(average_rating, on='movieId', how='inner')
      2 joined.head()
----> 3 joined.corr()

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:11049, in DataFrame.corr
(self, method, min_periods, numeric_only)
  11047 cols = data.columns
  11048 idx = cols.copy()
> 11049 mat = data.to_numpy(dtype=float, na_value=np.nan, copy=False)
  11051 if method == "pearson":
  11052     correl = libalgos.nancorr(mat, minp=min_periods)

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:1993, in DataFrame.to_num
py(self, dtype, copy, na_value)
   1991 if dtype is not None:
   1992     dtype = np.dtype(dtype)
-> 1993 result = self._mgr.as_array(dtype=dtype, copy=copy, na_value=na_value)
   1994 if result.dtype is not dtype:
   1995     result = np.asarray(result, dtype=dtype)

File ~\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:1694, in Blo
ckManager.as_array(self, dtype, copy, na_value)
   1692         arr.flags.writeable = False
   1693 else:
-> 1694     arr = self._interleave(dtype=dtype, na_value=na_value)
   1695     # The underlying data was copied within _interleave, so no need
   1696     # to further copy if copy=True or setting na_value
   1698 if na_value is lib.no_default:

File ~\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:1753, in Blo
ckManager._interleave(self, dtype, na_value)
   1751     else:
   1752         arr = blk.get_values(dtype)
-> 1753     result[rl.indexer] = arr
   1754     itemmask[rl.indexer] = 1
   1756 if not itemmask.all():

ValueError: could not convert string to float: 'Toy Story (1995)'
```

In [ ]:

In [ ]: