Travail à rendre

In [11]:
```
import pandas as pd
import matplotlib.pyplot as plt
movies = pd.read_csv('./movielens/movies.csv', sep=',')
movies.head()
```

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

In [3]:
```
# Tags
tags = pd.read_csv('./movielens/tags.csv', sep=',')
del tags['timestamp']
tags.head()
```

Out[3]:

	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 [4]:
```
# Ratings

ratings = pd.read_csv('./movielens/ratings.csv', sep=',', parse_dates=['timestamp'])
del ratings['timestamp']
ratings.head()
```

Out[4]:

	userId	movieId	rating
0	1	2	3.5
1	1	29	3.5
2	1	32	3.5

	userId	movieId	rating
3	1	47	3.5
4	1	50	3.5

1. Essayez d'autres techniques de netoyage dans Pandas vu dans le cours

In [5]:
```
print(movies.shape)
print(tags.shape)
print(ratings.shape)
print("#############")
movies = movies.dropna()
tags = tags.dropna()
ratings = ratings.dropna()
print("#############")
print(movies.shape)
print(tags.shape)
print(ratings.shape)
```

```
(27278, 3)
(465564, 3)
(20000263, 3)
#############
#############
(27278, 3)
(465548, 3)
(20000263, 3)
```

2. Essayez de rempalcer les valeurs numériques manquantes par la moyennes des valeurs, les valeurs les plus fréquentes et comparez vos résultats

In [7]:
```
# Tags
tags = pd.read_csv('./movielens/tags.csv', sep=',')
del tags['timestamp']
tags.head()
```

Out[7]:

	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 [8]:
```
import numpy as np

tags.replace([np.nan], np.mean, inplace=True)
```

In [9]:
```
tags.describe()
```

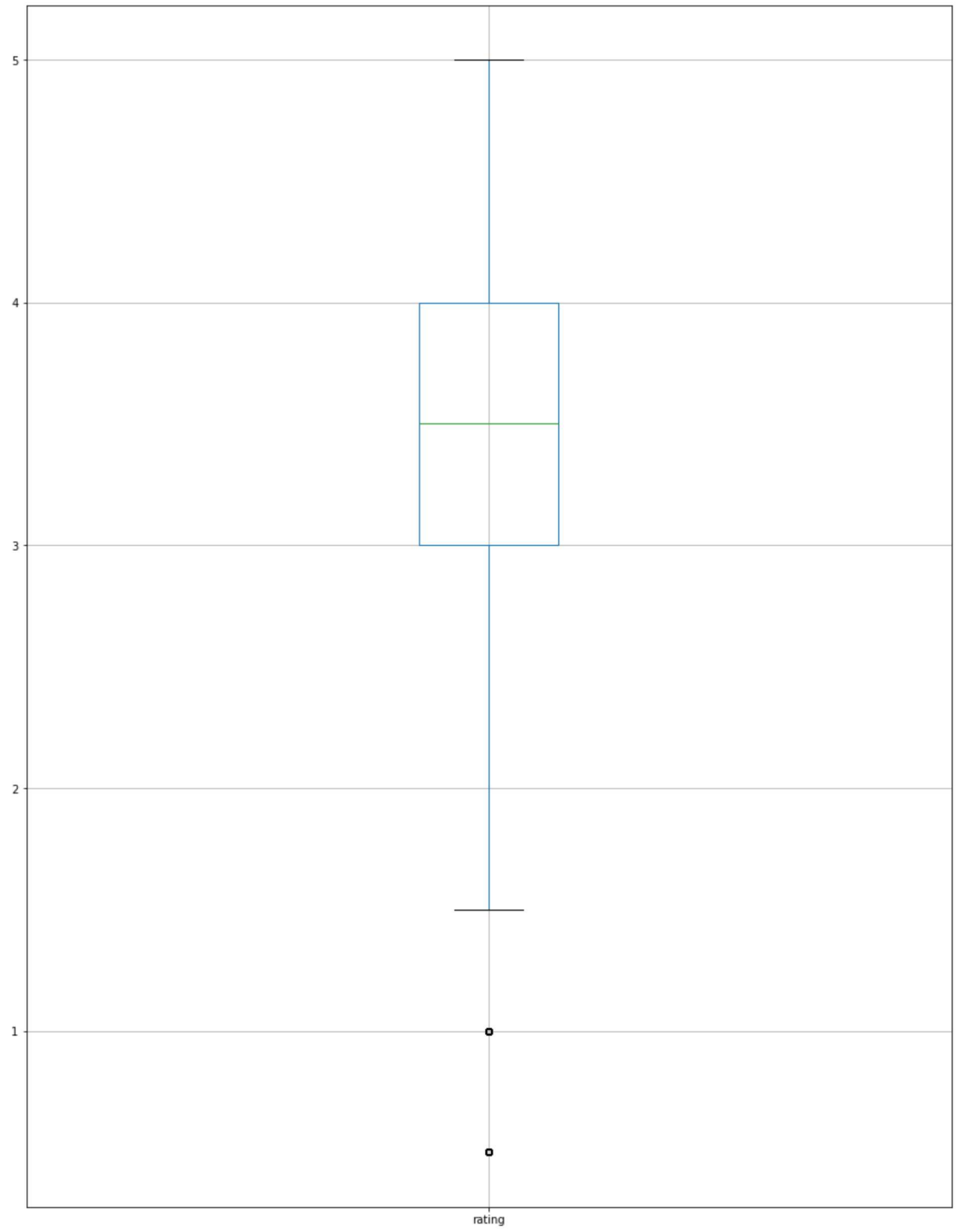
Out[9]:

	userId	movieId
count	465564.000000	465564.000000

	userId	movieId
mean	68712.354263	32627.762920
std	41877.674053	36080.241157
min	18.000000	1.000000
25%	28780.000000	2571.000000
50%	70201.000000	7373.000000
75%	107322.000000	62235.000000
max	138472.000000	131258.000000

3. Est ce qu'il y a des points qui sont des valeurs extrêmes? outliers

In [12]:
```
ratings.boxplot(column='rating', figsize=(15,20))
plt.show()
```



Dans le graphique ci-dessus, on peut clairement voir que les valeurs inferieurs à 1.5 agissent comme des valeurs aberrantes.

4. Afficher les films évalués par un utilisateur donné

par son ID.

In [13]:
```
t = movies.merge(tags, on='movieId', how='inner')
t.head()
```

Out[13]:

	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

5. Visualisez le nombre de films produits par année

In [14]:
```
movies['year'] = movies['title'].str.extract('.*\((.*)\).*', expand=True)
```

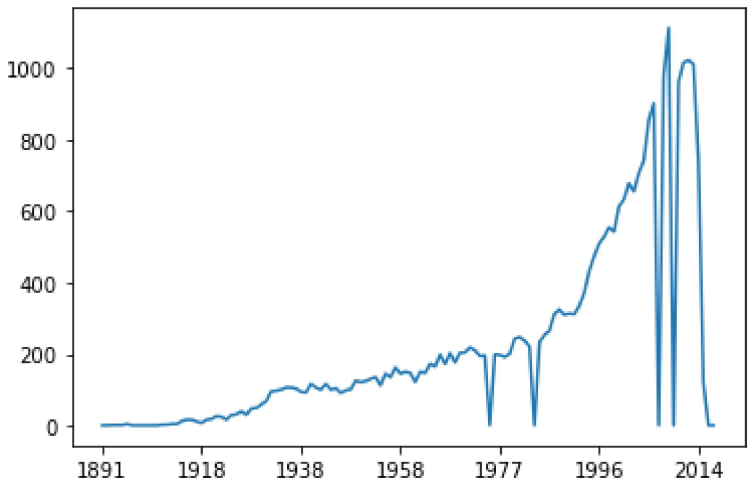
In [15]:
```
movies_count = movies['year'].value_counts()
```

In [16]:
```
movies_count = movies_count.sort_index()
movies_count
```

Out[16]:
```
1891                          1
1893                          1
1894                          2
1895                          2
1896                          2
                           ...
2013                       1011
2014                        740
2015                        120
Bicicleta, cullera, poma      1
Das Millionenspiel            1
Name: year, Length: 124, dtype: int64
```

In [17]:
```
movies_count.plot()
```

Out[17]:
```
<AxesSubplot:>
```



6. Visualiser l'occurence de chaque genre (drama, Animation, romance,...) de film dans le data set.

In [18]:
```
movie_genres = movies['genres'].str.split('|', expand=True)
print(type(movie_genres))
```

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

In [19]:
```
movie_genres
```

Out[19]:

	0	1	2	3	4	5	6	7	8	9
0	Adventure	Animation	Children	Comedy	Fantasy	None	None	None	None	None
1	Adventure	Children	Fantasy	None	None	None	None	None	None	None
2	Comedy	Romance	None	None	None	None	None	None	None	None
3	Comedy	Drama	Romance	None	None	None	None	None	None	None
4	Comedy	None	None	None	None	None	None	None	None	None
...	...	...	...	...	...	...	...	...	...	...
27273	Comedy	None	None	None	None	None	None	None	None	None
27274	Comedy	None	None	None	None	None	None	None	None	None
27275	Adventure	None	None	None	None	None	None	None	None	None
27276	(no genres listed)	None	None	None	None	None	None	None	None	None
27277	Adventure	Fantasy	Horror	None	None	None	None	None	None	None

27278 rows × 10 columns

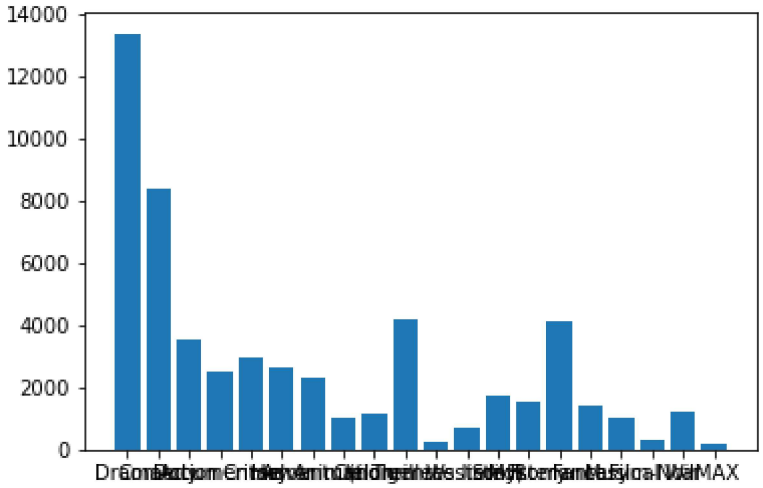
In [20]:
```
dict = {}
k = 0

for i in movie_genres:
    value_count = movie_genres[i].value_counts()
    for j in value_count.index:
        if k == 0:
            dict[j] = value_count[j]
        elif k == 1:
            dict[j] = dict[j] + value_count[j]
    k = 1
print(dict)
```

```
{'Drama': 13344, 'Comedy': 8374, 'Action': 3520, 'Documentary': 2471, 'Crime': 2939,
'Horror': 2611, 'Adventure': 2329, 'Animation': 1027, 'Children': 1139, 'Thriller':
4178, '(no genres listed)': 246, 'Western': 676, 'Sci-Fi': 1743, 'Mystery': 1514, 'R
omance': 4127, 'Fantasy': 1412, 'Musical': 1036, 'Film-Noir': 330, 'War': 1194, 'IMA
X': 196}
```

In [21]:
```
names = list(dict.keys())
values = list(dict.values())

plt.bar(range(len(dict)), values, tick_label=names)
plt.show()
```

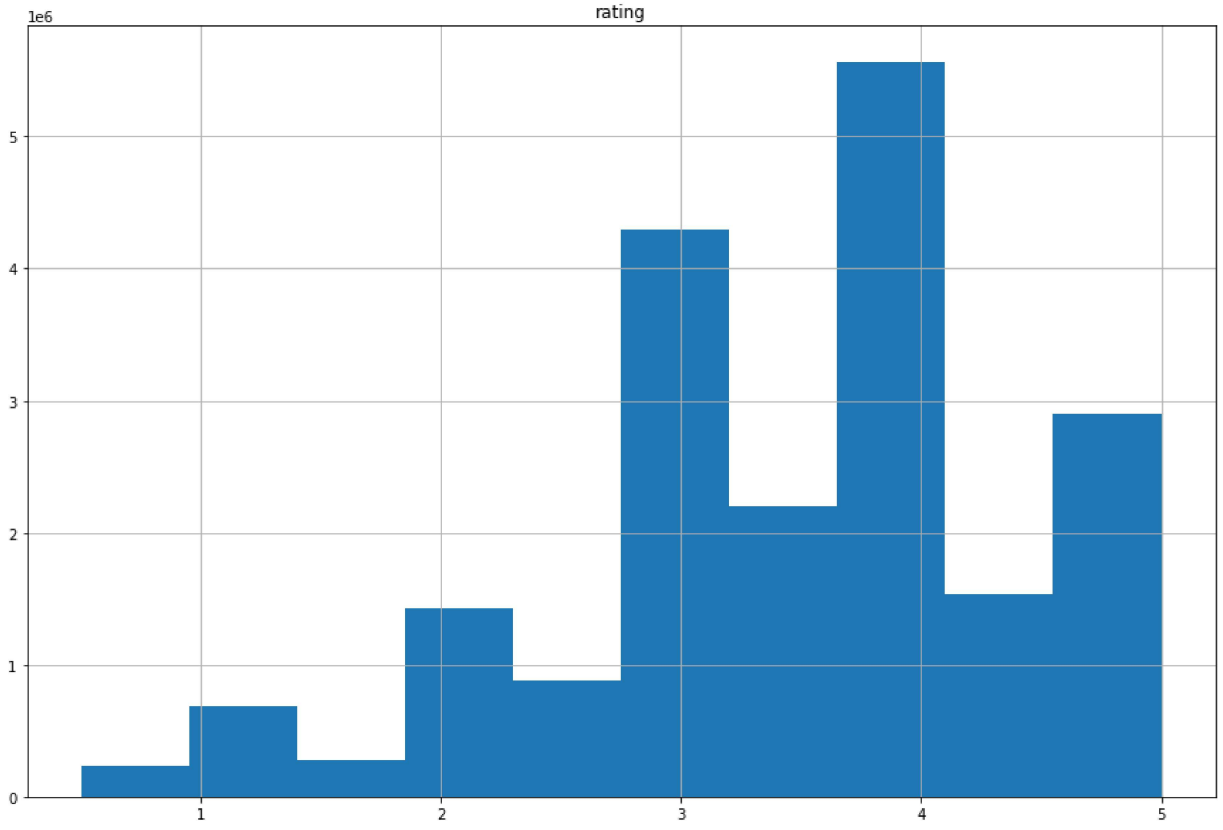


7. Essayez de pousser votre analyse exploratoire en faisant d'autres visualisation

Ratings par films :

In [22]:
```
ratings.hist(column='rating', figsize=(15,10))
```

Out[22]:
```
array([[<AxesSubplot:title={'center':'rating'}>]], dtype=object)
```

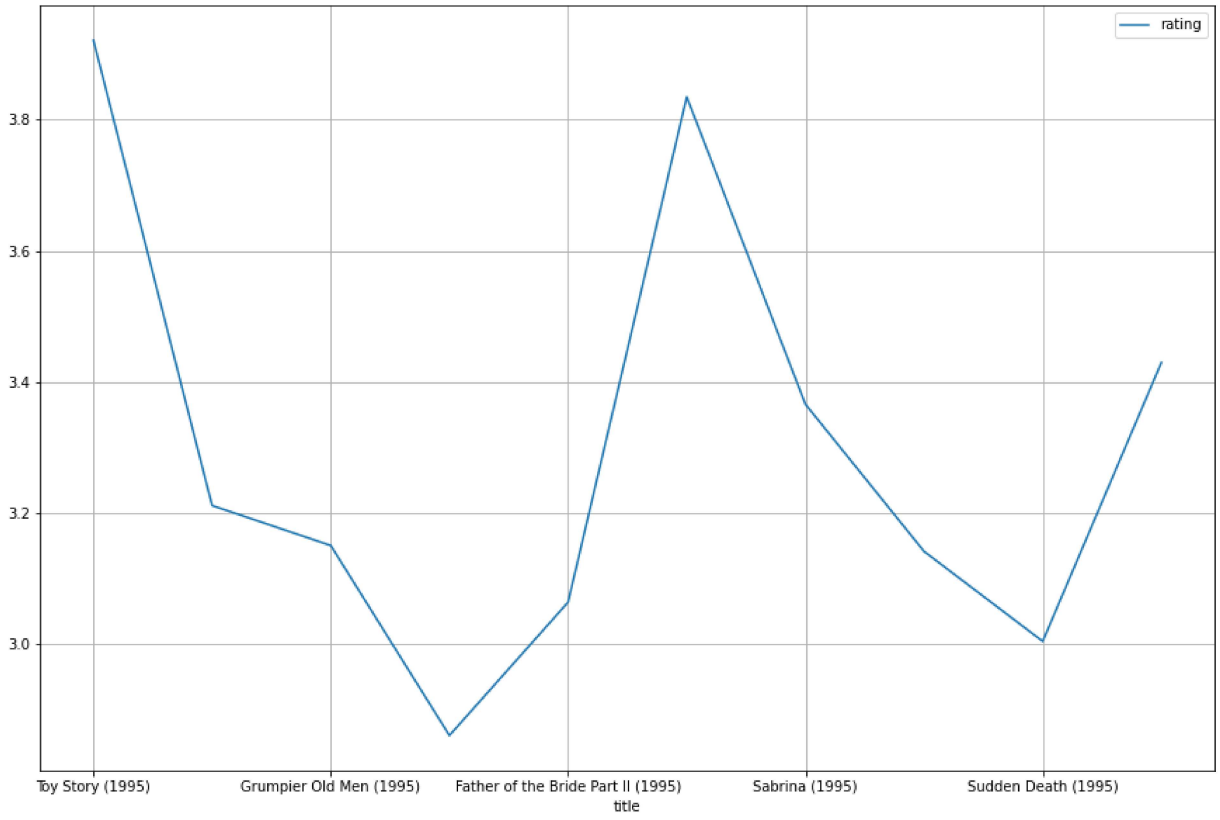


Ratings :

In [23]:
```
average_rating = ratings[['movieId','rating']].groupby('movieId', as_index=False).me
joined = movies.merge(average_rating, on='movieId', how='inner')
```

In [24]:
```
joined[:10].plot(x='title', y='rating', figsize=(15,10), grid=True)
```

Out[24]: <AxesSubplot:xlabel='title'>



Omar Naitelhaj