The project is to analyse the rating behaviour of Romance movies over 145 years.

Starting from 1874 until 2019

Importing Libraries

In [86]:
```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

In [87]:
```
movies= pd.read_csv("D:\\python\\additional\\ml-25m\\ml-25m\\movies.csv")
```

In [88]:
```
movies.head(10)
```

Out[88]:

	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

In [89]:
```
movies.shape
```

Out[89]:
```
(62423, 3)
```

In [90]:
```
movies.describe()
```

Out[90]:

	movieId
count	62423.000000
mean	122220.387646
std	63264.744844
min	1.000000
25%	82146.500000
50%	138022.000000
75%	173222.000000
max	209171.000000

Split the years from the tilte column to a separate column called 'year'

In [91]:
```
movies['year']= movies['title'].str.extract('.*\((.*)\).*', expand=True)
```

In [92]:
```
movies.head()
```

Out[92]:

	movieId	title	genres	year
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1995
1	2	Jumanji (1995)	Adventure|Children|Fantasy	1995
2	3	Grumpier Old Men (1995)	Comedy|Romance	1995
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	1995
4	5	Father of the Bride Part II (1995)	Comedy	1995

Read the 'ratings' dataframe

In [93]:
```
rating= pd.read_csv("D:\\python\\additional\\ml-25m\\ml-25m\\ratings.csv")
```

In [94]:
```
rating.head(5)
```

Out[94]:

	userId	movieId	rating	timestamp
0	1	296	5.0	1147880044
1	1	306	3.5	1147868817
2	1	307	5.0	1147868828
3	1	665	5.0	1147878820
4	1	899	3.5	1147868510

In [95]:
```
del rating['timestamp']
```

In [96]:
```
rating.shape
```

Out[96]:
```
(25000095, 3)
```

In [97]:
```
rating.describe()
```

Out[97]:

	userId	movieId	rating
count	2.500010e+07	2.500010e+07	2.500010e+07
mean	8.118928e+04	2.138798e+04	3.533854e+00
std	4.679172e+04	3.919886e+04	1.060744e+00
min	1.000000e+00	1.000000e+00	5.000000e-01
25%	4.051000e+04	1.196000e+03	3.000000e+00
50%	8.091400e+04	2.947000e+03	3.500000e+00
75%	1.215570e+05	8.623000e+03	4.000000e+00
max	1.625410e+05	2.091710e+05	5.000000e+00

Apply groupby function on 'rating'

In [98]:
```
avg_rat = rating.groupby('movieId', as_index = False).mean()
```

In [99]:
```
avg_rat.head(5)
```

Out[99]:

	movieId	userId	rating
0	1	81294.564728	3.893708
1	2	81358.542554	3.251527
2	3	81343.694934	3.142028
3	4	81266.193024	2.853547
4	5	81002.872460	3.058434

Create the 'cinema' dataframe which is the merging between 'movies' and 'ratings' dataframs.

In [100]:
```
cinema = movies.merge(avg_rat, on = 'movieId', how = 'inner')
print ('Shape',cinema.shape)
```

```
Shape (59047, 6)
```

In [111]:
```
cinema.head(5)
```

Out[111]:

	movieId	title	genres	year	userId	rating
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1995	81294.564728	3.893708
1	2	Jumanji (1995)	Adventure|Children|Fantasy	1995	81358.542554	3.251527
2	3	Grumpier Old Men (1995)	Comedy|Romance	1995	81343.694934	3.142028
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	1995	81266.193024	2.853547
4	5	Father of the Bride Part II (1995)	Comedy	1995	81002.872460	3.058434

Creating the 'Romance' subdataframe as 'rom'

In [102]:
```
rom = cinema[(cinema.genres == 'Documentary')]
```

In [103]:
```
rom.head()
```

Out[103]:

	movieId	title	genres	year	userId	rating
76	77	Nico Icon (1995)	Documentary	1995	81352.909953	3.402844
97	99	Heidi Fleiss: Hollywood Madam (1995)	Documentary	1995	84324.905689	3.101048
106	108	Catwalk (1996)	Documentary	1996	88036.365217	3.073913
114	116	Anne Frank Remembered (1995)	Documentary	1995	83564.330286	3.911429
126	128	Jupiter's Wife (1994)	Documentary	1994	83893.154930	3.485915

In [104]:
```
print(rom.shape)
print(rom.dtypes)
```

```
(4603, 6)
movieId     int64
title      object
genres     object
year       object
userId    float64
rating    float64
dtype: object
```

Changing the type of 'year' coloum from object to numeric

In [105]:
```
rom['year']= pd.to_numeric(rom['year'], errors='coerce')
```

```
C:\Users\hp\anaconda3\lib\site-packages\ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

In [106]:
```
rom.dtypes
```

Out[106]:
```
movieId     int64
title      object
genres     object
year      float64
userId    float64
rating    float64
dtype: object
```

to know the minimum and the maximum year

In [107]:
```
print('minimum',rom['year'].min())
print('maximum',rom['year'].max())
```

```
minimum 1874.0
maximum 2019.0
```

Data visualization

scatter plot

In [108]:
```
plt.scatter(x=rom['year'], y=rom['rating'] , color= 'red', alpha= 0.6)
plt.title('Rating of Romantic movies from 1874 to 2019', fontsize= 20)
plt.xlabel('Year', fontsize=15)
plt.ylabel('Rating', fontsize= 15)
plt.grid(True)
plt.show()
```