Importing the library

In [2]:
```
import pandas as pd
import numpy as np
import requests
from tqdm import tqdm
from bs4 import BeautifulSoup
from tqdm import tqdm
import time

from selenium import webdriver
from webdriver_manager.chrome import ChromeDriverManager
```

Chrome Driver - Web Scraping

In [2]:
```
driver = webdriver.Chrome(ChromeDriverManager().install())
link = 'https://www.imdb.com/chart/toptv/'
driver.get(link)
html = driver.page_source
soup = BeautifulSoup(html,'html.parser')
```

```
/tmp/ipykernel_24038/243864188.py:1: DeprecationWarning: executable_path has been deprecated, plea
se pass in a Service object
  driver = webdriver.Chrome(ChromeDriverManager().install())
```

webscraping 250 shows | basic details

In [3]:
```
data = []

for shows in soup.find('tbody',class_ ='lister-list').find_all('tr'):
    tv_id    = shows.find('td',class_ = 'titleColumn').find('a').get('href').split('/')[-2]
    tv_name  = shows.find('td',class_ = 'titleColumn').find('a').text.strip()
    year     = int(shows.find('span',class_ = 'secondaryInfo').text.strip()[1:-1])
    ratings  = float(shows.find('strong').text.strip())
    data.append([tv_id,tv_name,year,ratings])

df = pd.DataFrame(data,columns = ['tv_id','tv_name','release_year','ratings'])
df.to_csv('top_250_shows.csv',index = False)
```

Checking the code for any one web page series

In [4]:
```
for i in df['tv_id'][1:]:
    link = 'https://www.imdb.com/title/'+i
    driver.get(link)
    sp = BeautifulSoup(driver.page_source,'html.parser')
    print(sp.find_all('a')[1].get('href'))

    break
```

```
https://www.imdb.com/calendar/?ref_=nv_mv_cal (https://www.imdb.com/calendar/?ref_=nv_mv_cal)
```

Webscraping entire 250 imdb top series

In [5]:
```
data2 = []

for i in tqdm(df['tv_id']):

    link = 'https://www.imdb.com/title/' + i
    driver.get(link)
    time.sleep(1)

    sp = BeautifulSoup(driver.page_source,'html.parser')

    ## scraping the number of episodes
    try:
        episodes = int(sp.find('div',class_ = 'sc-8862e651-2 JWTyb').find('div').find_all('span')[1]
    except:
        episode = np.nan

    ## scraping the series type
    try:
        series_type = sp.find('div',class_ ='sc-b5e8e7ce-2 AIESV').find_all('li')[0].text.strip()
    except:
        series_type = np.nan

    ## scraping the certificate
    try:
        certificate = (sp.find('div',class_ ='sc-b5e8e7ce-2 AIESV').find_all('li')[2].find('span').t
    except:
        certificate = 'No Certification'

    ## scraping the duration
    try:
        duration    = sp.find('div',class_ ='sc-b5e8e7ce-2 AIESV').find_all('li')[3].text
    except:
        duration    = 'Not Available'

    ## scraping the cast id and cast name
    try:
        cast_id        = ','.join([cast.find('div',class_ = 'sc-bfec09a1-7 dpBDvu' ).find('a').get(
    except:
        cast_id = np.nan
    try:
        cast_name      = ','.join([cast.find('div',class_ = 'sc-bfec09a1-7 dpBDvu' ).find('a').text
```

```
    except:
        cast_name = np.nan

    ## scraping the tagline
    try:
        tagline = sp.find('ul',class_ = 'ipc-metadata-list ipc-metadata-list--dividers-all sc-fba22a
    except:
        tagline = 'No Taglines'

    ## scraping the genre
    try:
        genre = ','.join([i.text for i in sp.find('div',class_ = 'ipc-chip-list__scroller').find_all
    except:
        genre = np.nan


    data2.append([series_type,episodes,certificate,duration,cast_id,cast_name,tagline,genre])

ls = ['series_type','episodes','certificate','duration','cast_id','cast_name','tagline','genre']
for i in range(len(ls)):
    df[ls[i]] = np.array(data2)[:,i]
```

```
100%|██████████████████████████████████████| 250/250 [34:31<00:00,  8.29s/it]
```

Saving the top 250 shows dataset

In [6]:
```
df.to_csv('top_250_shows.csv',index = False)
```

In [7]:
```
df.head()
```

Out[7]:

	tv_id	tv_name	release_year	ratings	series_type	episodes	certificate	duration	
0	tt5491994	Planet Earth II	2016	9.4	TV Mini Series	6	U	4h 58m	nm0041003,nm0118096,nm1769336,nm4830788,n
1	tt0903747	Breaking Bad	2008	9.4	TV Series	62	15	49m	nm0186505,nm0666739,nm0348152,nm1336827,n
2	tt0795176	Planet Earth	2006	9.4	TV Mini Series	11	PG	8h 58m	nm0000244,nm0041003,nm0238419,nm8603319,n
3	tt0185906	Band of Brothers	2001	9.4	TV Mini Series	10	15	9h 54m	nm0342241,nm0507073,nm0515296,nm0853169,n
4	tt7366338	Chernobyl	2019	9.3	TV Mini Series	5	15	5h 30m	nm2976580,nm0364813,nm0001745,nm1835523,n

Preprocessing the top 250 shows

In [3]:
```
df = pd.read_csv('top_250_shows.csv')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250 entries, 0 to 249
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   tv_id         250 non-null    object
 1   tv_name       250 non-null    object
 2   release_year  250 non-null    int64
 3   ratings       250 non-null    float64
 4   series_type   250 non-null    object
 5   episodes      250 non-null    int64
 6   certificate   250 non-null    object
 7   duration      250 non-null    object
 8   cast_id       250 non-null    object
 9   cast_name     250 non-null    object
 10  tagline       250 non-null    object
 11  genre         250 non-null    object
dtypes: float64(1), int64(2), object(9)
memory usage: 23.6+ KB
```

In [4]:
```
df.describe()
```

Out[4]:

	release_year	ratings	episodes
count	250.000000	250.000000	250.000000
mean	2006.928000	8.653200	68.736000
std	12.512597	0.220099	92.832417
min	1955.000000	8.400000	2.000000
25%	2001.000000	8.500000	14.000000
50%	2010.000000	8.600000	34.500000
75%	2016.000000	8.800000	78.000000
max	2023.000000	9.400000	744.000000

In [5]:
```
df.isnull().sum()
```

Out[5]:
```
tv_id           0
tv_name         0
release_year    0
ratings         0
series_type     0
episodes        0
certificate     0
duration        0
cast_id         0
cast_name       0
tagline         0
genre           0
dtype: int64
```

Pre-processing the duration columns

In [44]:
```
duration = []
for i in df['duration']:
    if 'm' in (i.split(' ')[0]):
        duration.append(i.split(' ')[0][:-1])
    elif 'h' in i.split(' ')[0] and len(i.split(' ')) == 2 :
        duration.append(int(i.split(' ')[0][:-1]) * 60 + int(i.split(' ')[1][:-1]))
    elif 'h' in i.split(' ')[0] and len(i.split(' ')) == 1:
        duration.append(int(i.split(' ')[0][:-1])*60)
    else:
        duration.append(i)

df['duration'] = duration
```

Final Save

In [46]:
```
df.to_csv('top_250_shows.csv',index = False )
```

In [48]:
```
df.head()
```

Out[48]:

	tv_id	tv_name	release_year	ratings	series_type	episodes	certificate	duration	
0	tt5491994	Planet Earth II	2016	9.4	TV Mini Series	6	U	298	nm0041003,nm0118096,nm1769336,nm4830788,n
1	tt0903747	Breaking Bad	2008	9.4	TV Series	62	15	49	nm0186505,nm0666739,nm0348152,nm1336827,n
2	tt0795176	Planet Earth	2006	9.4	TV Mini Series	11	PG	538	nm0000244,nm0041003,nm0238419,nm8603319,n
3	tt0185906	Band of Brothers	2001	9.4	TV Mini Series	10	15	594	nm0342241,nm0507073,nm0515296,nm0853169,n
4	tt7366338	Chernobyl	2019	9.3	TV Mini Series	5	15	330	nm2976580,nm0364813,nm0001745,nm1835523,n