```
# NETFLIX BUSINESS CASE STUDY
```

To analyse,interprete and visualize the given Netflix data and to solve the related problems to get insights we need functions and methods, so we must import Python libraries into our work notebook.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

To get the data into our work space we use the below code(to read csv files) and saving the whole set of data into a single variable(dataframe) which makes analysis easier

```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv
```

```
--2023-08-07 14:05:21--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 99.84.178.226, 99.84.178.93, 99.84.178.132, ...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|99.84.178.226|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3399671 (3.2M) [text/plain]
Saving to: ‘netflix.csv’

netflix.csv         100%[===================>]   3.24M  --.-KB/s    in 0.06s

2023-08-07 14:05:21 (57.5 MB/s) - ‘netflix.csv’ saved [3399671/3399671]
```

```
df = pd.read_csv('netflix.csv')
```

```
df.head(2)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
					Ama Qamata,							

Head functions gives the first 5 rows of the data which helps to avoid loading the entire data everytime for analysis

UNDERSTANDING THE GIVEN DATA

Its important to understand the data before analysing it as it will give an idea about how to handle the data and extract the desired information from it.

```
# To get first few rows
```

```
df.head(1)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is	Kirsten	NaN	United	September	2020	PG-13	90 min	Documentaries	As her father nears the end of his life,

```
# To get last few rows
```

```
df.tail(1)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias,	India	March 2, 2019	2015	TV-14	111 min	Dramas, International Movies, Music	A scrappy but poor boy worms his way

```
# To get n number of random rows from the dataset

df.sample(n=2)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
8699	s8700	Movie	Warehoused	Jack Zagha Kababie	José Carlos Ruiz, Hoze Meléndez	Mexico	November 1, 2017	2015	TV-14	92 min	Comedies, Dramas, International Movies	A soon-to-be-retiring Mr. Lino teaches 20-some...

```
# TO GET NO. OF ROWS & COLUMNS:

df.shape
```

```
(8807, 12)
```

```
# TO GET TOTAL ELEMENTS IN THE DATASET (i.e., the dot product of no. of rows & columns)

df.size
```

```
105684
```

```
# To get index

df.index
```

```
RangeIndex(start=0, stop=8807, step=1)
```

```
# TO GET THE NAMES OF THE COLUMNS

df.columns
```

```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

```
# TO GET THE NAMES OF THE COLUMNS(alternate method)

df.keys()
```

```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

```
# To get memory usage of each column

df.memory_usage()
```

```
Index            128
show_id        70456
type           70456
title          70456
director       70456
cast           70456
country        70456
date_added     70456
release_year   70456
rating         70456
duration       70456
listed_in      70456
description    70456
dtype: int64
```

```
# TO GET THE TOTAL INFORMATION ABOUT THE DATASET.
# info function let us know the columns with their data types and no. of non-null values & the memory usage


df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
```

```
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

From the above analysis we get to know that all columns are object data type except

"release_year" column which is of integer type

TO ANALYSE THE BASIC METRICS

```
# To get the data type of each column

df.dtypes
```

```
show_id         object
type            object
title           object
director        object
cast            object
country         object
date_added      object
release_year     int64
rating          object
duration        object
listed_in       object
description     object
dtype: object
```

STATISTICAL SUMMERY

```
df.describe()
```

	release_year
count	8807.000000
mean	2014.180198
std	8.819312
min	1925.000000
25%	2013.000000
50%	2017.000000
75%	2019.000000
max	2021.000000

Describe function returns the glimpse of the data with the statistical values from all over the data just to predict the normal ranges and average ranges to the particular elements. Note: it will display only the numerical values and return from the numerical values.

```
# To get statistical values for the object data type

df.describe(include = object)
```

	show_id	type	title	director	cast	country	date_added	rating	duration	listed_in	description
count	8807	8807	8807	6173	7982	7976	8797	8803	8804	8807	8807

```
# Accessing the rows with their iloc(integer location) values

df.iloc[:3]
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV	After crossing paths at a party, a Cape

```
# Accessing selected range of rows using external location values

df.loc[3:5]
```

	show_id	type	title	director	cast	country	date_added	release
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	
					Mayur More,			

```
# Accessing the specified columns for all rows using external location

df.loc[:,['title','director','release_year']]
```

	title	director	release_year
0	Dick Johnson Is Dead	Kirsten Johnson	2020
1	Blood & Water	NaN	2021
2	Ganglands	Julien Leclercq	2021
3	Jailbirds New Orleans	NaN	2021
4	Kota Factory	NaN	2021
...	...	...	...
8802	Zodiac	David Fincher	2007
8803	Zombie Dumb	NaN	2018
8804	Zombieland	Ruben Fleischer	2009
8805	Zoom	Peter Hewitt	2006
8806	Zubaan	Mozez Singh	2015

SEPERATING MOVIES & TV SHOWS

The given dataset has details of both movies and tv shows which has to be seperated to do deeper analysis. Since, analysing parameters belonging to similar category will only gives a meaningfull insight.

```
movie = df[df['type'] == 'Movie']
movie.sample(2)
```

	show_id	type	title	director	cast	country	date_added	re
946	s947	Movie	Stargate	Roland Emmerich	Kurt Russell, James Spader, Jaye	United States, France	May 1, 2021	

```
tvshow = df[df['type'] == 'TV Show']
tvshow.sample(2)
```

	show_id	type	title	director	cast	country	date_add
459	s460	TV Show	Never Have I Ever	NaN	Maitreyi Ramakrishnan, Poorna Jagannathan,	United States	July 1 20

Now, we have two dataframes where one has only the movie details and the other has the TV show details. Thus analysis these dataframes would give more reliable results

TO CLEAN THE DATASET

CHECK FOR DUPLICATE VALUES

```
movie.duplicated().sum()
```

```
0
```

```
tvshow.duplicated().sum()
```

```
0
```

```
# To know the presence of null values in the dataset
# It returns boolean value: True - indiates presence of a null value
```

```
movie.isnull().head(2)
```

	show_id	type	title	director	cast	country	date_added	release_yea
0	False	False	False	False	True	False	False	Fals

```
tvshow.isnull().head(2)
```

	show_id	type	title	director	cast	country	date_added	release_yea
1	False	False	False	True	False	False	False	Fals

To clean the dataset i.e., to fill the NA values with a proxy value (here - 'Mode' of that column)

FOR MOVIE DATASET

```
# To get the total no. of null values in each columns
```

```
movie.isna().sum()
```

```
show_id          0
type             0
title            0
director       188
cast           475
country        440
date_added       0
release_year     0
rating           2
duration         3
listed_in        0
description      0
dtype: int64
```

```
movie['director'].replace([np.nan],movie['director'].mode())
```

```
0                     Kirsten Johnson
6       Robert Cullen, José Luis Ucha
7                       Haile Gerima
9                     Theodore Melfi
12                Christian Schwochow
                    ...
8801                  Majid Al Ansari
8802                    David Fincher
8804                  Ruben Fleischer
8805                     Peter Hewitt
8806                      Mozez Singh
Name: director, Length: 6131, dtype: object
```

```
movie['director'].replace([np.nan],'UNKNOWN',inplace = True)
```

```
<ipython-input-33-e22d17a9a91d>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  movie['director'].replace([np.nan],'UNKNOWN',inplace = True)
```

```
movie['cast'] = movie['cast'].replace([np.nan],'UNKNOWN')
```

```
<ipython-input-34-0dfec4179870>: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
  movie['cast'] = movie['cast'].replace([np.nan],'UNKNOWN')
```

```
movie['duration'].replace([np.nan],movie['duration'].mode(),inplace = True)
```

```
<ipython-input-35-783f1b6bd517>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  movie['duration'].replace([np.nan],movie['duration'].mode(),inplace = True)
```

```
movie['country'].replace([np.nan],movie['country'].mode(),inplace = True)
```

```
<ipython-input-37-cceba0ab9944>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  movie['country'].replace([np.nan],movie['country'].mode(),inplace = True)
```

```
movie['rating'].replace([np.nan],movie['rating'].mode(),inplace = True)
```

```
<ipython-input-36-b1ff06a5a6c8>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  movie['rating'].replace([np.nan],movie['rating'].mode(),inplace = True)
```

```
# To check whether all NA values are replaced by proxy

movie.isna().sum()
```

```
show_id         0
type            0
title           0
director        0
cast            0
country         0
date_added      0
release_year    0
rating          0
duration        0
listed_in       0
description     0
dtype: int64
```

FOR TV SHOW DATASET

```
tvshow.isna().sum()
```

```
show_id            0
type               0
title              0
director        2446
cast             350
country          391
date_added        10
release_year       0
rating             2
duration           0
listed_in          0
description        0
dtype: int64
```

```
tvshow['director'].replace([np.nan],tvshow['director'].mode())
```

```
1       Alastair Fothergill
2           Julien Leclercq
3       Alastair Fothergill
4       Alastair Fothergill
5             Mike Flanagan
               ...
8795    Alastair Fothergill
8796    Alastair Fothergill
8797    Alastair Fothergill
8800    Alastair Fothergill
8803    Alastair Fothergill
Name: director, Length: 2676, dtype: object
```

```
tvshow['director'].replace([np.nan],'UNKNOWN',inplace = True)
```

```
<ipython-input-41-241f582692df>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  tvshow['director'].replace([np.nan],'UNKNOWN',inplace = True)
```

```
tvshow['cast'].replace([np.nan],'UNKNOWN',inplace = True)
```

```
<ipython-input-42-827d2e90419e>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  tvshow['cast'].replace([np.nan],'UNKNOWN',inplace = True)
```

```
tvshow['country'].replace([np.nan],'UNKNOWN',inplace = True)
```

```
<ipython-input-43-00228d44a18f>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  tvshow['country'].replace([np.nan],'UNKNOWN',inplace = True)
```

```
tvshow['rating'].replace([np.nan],tvshow['rating'].mode(),inplace = True)
```

```
<ipython-input-44-5f05f045b6c9>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  tvshow['rating'].replace([np.nan],tvshow['rating'].mode(),inplace = True)
```

```
tvshow['date_added'].replace([np.nan],tvshow['date_added'].mode(),inplace = True)
```

```
<ipython-input-45-aa5489de10f2>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  tvshow['date_added'].replace([np.nan],tvshow['date_added'].mode(),inplace = True)
```

```
tvshow.isna().sum()
```

```
show_id         0
type            0
title           0
director        0
cast            0
country         0
date_added      0
release_year    0
rating          0
duration        0
listed_in       0
description     0
dtype: int64
```

PRE-PROCESSING THE DATA (UNNESTING DATAS IN COLUMN)

unnesting movies:

```
# Unnesting multiple names in cast column which is seperated by comma

movie_c = movie['cast'].str.split(',' , expand = True).stack()
```

```
movie_c = movie_c.reset_index(level = 1,drop=True).to_frame('cast')

movie_c['show_id'] = movie['show_id']

movie_c
```

	cast	show_id
0	UNKNOWN	s1
6	Vanessa Hudgens	s7
6	Kimiko Glenn	s7
6	James Marsden	s7
6	Sofia Carson	s7
...	...	...
8806	Manish Chaudhary	s8807
8806	Meghna Malik	s8807
8806	Malkeet Rauni	s8807
8806	Anita Shabdish	s8807
8806	Chittaranjan Tripathy	s8807

```
movie_cast = movie
movie_cast = movie_c.merge(movie_cast, on ='show_id', how='left')

movie_cast.head(2)
```

	cast_x	show_id	type	title	director	cast_y	country	da
0	UNKNOWN	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	UNKNOWN	United States	S

```
movie_cast.drop(columns = ['cast_y'], inplace = True)

movie_cast.rename(columns ={'cast_x':'cast'},inplace = True)
movie_cast.head(2)
```

	cast	show_id	type	title	director	country	date_added	re
0	UNKNOWN	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	United States	September 25, 2021	

```
movie_dir= movie['director'].str.split(',' , expand = True).stack()
movie_dir = movie_dir.reset_index(level = 1,drop=True).to_frame('director')
movie_dir['show_id'] = movie['show_id']


movie_director = movie_dir.merge(movie, on ='show_id', how='left')
movie_director.drop(columns = ['director_y'], inplace = True)
movie_director.rename(columns ={'director_x':'director'},inplace = True)
movie_director.sample(2)
```

	director	show_id	type	title	cast	country	date_added
6329	Norman Jewison	s8250	Movie	The Cincinnati Kid	Steve McQueen, Edward G. Robinson	United States	November 1, 2019

```
movie_director['director'].nunique()
```

```
4887
```

```
movie_cty= movie['country'].str.split(',' , expand = True).stack()
movie_cty = movie_cty.reset_index(level = 1,drop=True).to_frame('country')
movie_cty['show_id'] = movie['show_id']


movie_country = movie_cty.merge(movie, on ='show_id', how='left')
movie_country.drop(columns = ['country_y'], inplace = True)
movie_country.rename(columns ={'country_x':'country'},inplace = True)
movie_country.sample(2)
```

	country	show_id	type	title	director	cast	date_added	release_year	rating	duration	listed_in	description
1160	France	s1388	Movie	The Next Three Days	Paul Haggis	Russell Crowe, Elizabeth Banks, Brian Dennehy,...	January 22, 2021	2010	PG-13	133 min	Dramas, Thrillers	When his wife becomes a murder suspect and is ...

Pulkit

```
movie_list= movie['listed_in'].str.split(',' , expand = True).stack()
movie_list = movie_list.reset_index(level = 1,drop=True).to_frame('listed_in')
movie_list['show_id'] = movie['show_id']


movie_listed_in = movie_list.merge(movie, on ='show_id', how='left')
movie_listed_in.drop(columns = ['listed_in_y'], inplace = True)
movie_listed_in.rename(columns ={'listed_in_x':'listed_in'},inplace = True)
movie_listed_in.sample(2)
```

	listed_in	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	description
7814	Comedies	s5533	Movie	Our Lovers	Miguel Ángel Lamata	Eduardo Noriega, Michelle Jenner, Fele Martíne...	Spain	April 14, 2017	2016	TV-MA	92 min	A love-challenged man and a woman he meets at ...

▼ unnesting tv show:

```
tvshow_c= tvshow['cast'].str.split(',' , expand = True).stack()
tvshow_c = tvshow_c.reset_index(level = 1,drop=True).to_frame('cast')
tvshow_c['show_id'] = tvshow['show_id']


tvshow_cast = tvshow_c.merge(tvshow, on ='show_id', how='left')
tvshow_cast.drop(columns = ['cast_y'], inplace = True)
tvshow_cast.rename(columns ={'cast_x':'cast'},inplace = True)
tvshow_cast.sample(2)
```

	cast	show_id	type	title	director	country	date_added	release_year	rating	duration	listed_in	description
12857	Héctor Segura	s4431	TV Show	Creators	UNKNOWN	Argentina	November 1, 2018	2015	TV-Y	2 Seasons	Kids' TV, Spanish-Language TV Shows	Two brilliant scientists discover the hidden v...
	Avesha		TV				December				International TV Shows,	A series of incidents in

```
tvshow_dir= tvshow['director'].str.split(',' , expand = True).stack()
tvshow_dir = tvshow_dir.reset_index(level = 1,drop=True).to_frame('director')
tvshow_dir['show_id'] = tvshow['show_id']


tvshow_director = tvshow_dir.merge(tvshow, on ='show_id', how='left')
tvshow_director.drop(columns = ['director_y'], inplace = True)
tvshow_director.rename(columns ={'director_x':'director'},inplace = True)
tvshow_director.sample(2)
```

	director	show_id	type	title	cast	country	date_added	release_year	rating	duration	listed_in	description
2168	UNKNOWN	s5903	TV Show	H2O: Mermaid Adventures	Sonja Ball, Holly Gauthier-Frankel, Thor Bisho...	Germany, Australia, France, China	July 15, 2015	2015	TV-Y7	2 Seasons	Kids' TV	Three high school friends who turn into mermai...
					Song							

```
tvshow_cty= tvshow['country'].str.split(',' , expand = True).stack()
tvshow_cty = tvshow_cty.reset_index(level = 1,drop=True).to_frame('country')
tvshow_cty['show_id'] = tvshow['show_id']


tvshow_country = tvshow_cty.merge(tvshow, on ='show_id', how='left')
tvshow_country.drop(columns = ['country_y'], inplace = True)
tvshow_country.rename(columns ={'country_x':'country'},inplace = True)
tvshow_country.sample(2)
```

	country	show_id	type	title	director	cast	date_added	release_year	rating	duration	listed_in	description
682	Sweden	s1741	TV Show	Love & Anarchy	UNKNOWN	Ida Engvoll, Björn Mosten, Carla Sehn, Reine B...	November 4, 2020	2020	TV-MA	1 Season	International TV Shows, Romantic TV Shows, TV ...	A married consultant and a young IT tech kick ...
						Christian						

```
tvshow_list= tvshow['listed_in'].str.split(',' , expand = True).stack()
tvshow_list = tvshow_list.reset_index(level = 1,drop=True).to_frame('listed_in')
tvshow_list['show_id'] = tvshow['show_id']


tvshow_listed_in = tvshow_list.merge(tvshow, on ='show_id', how='left')
tvshow_listed_in.drop(columns = ['listed_in_y'], inplace = True)
tvshow_listed_in.rename(columns ={'listed_in_x':'listed_in'},inplace = True)
tvshow_listed_in.sample(2)
```

	listed_in	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	description
25	TV Comedies	s16	TV Show	Dear White People	UNKNOWN	Logan Browning, Brandon P. Bell, DeRon Horton,...	United States	September 22, 2021	2021	TV-MA	4 Seasons	Students of color navigate the daily slights a...

This reality

NON-GRAPHICAL ANALYSIS:

```
# To get unique number of elements in each column.

df.nunique()
```

```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1767
release_year      74
rating            17
duration         220
listed_in        514
description     8775
dtype: int64
```

```
# To get unique elements in a specific column.

df['country'].unique
```

```
<bound method Series.unique of 0       United States
1        South Africa
2                 NaN
3                 NaN
4               India
             ...
8802    United States
8803              NaN
8804    United States
8805    United States
8806            India
Name: country, Length: 8807, dtype: object>
```

```
# To get the frequency of occurence of each unique elements in a specific column.

df['country'].value_counts()
```

```
United States                              2818
India                                       972
United Kingdom                              419
Japan                                       245
South Korea                                 199
                                           ...
Romania, Bulgaria, Hungary                    1
Uruguay, Guatemala                            1
France, Senegal, Belgium                      1
Mexico, United States, Spain, Colombia        1
United Arab Emirates, Jordan                  1
Name: country, Length: 748, dtype: int64
```

```
df['type'].value_counts()
```

```
Movie      6131
TV Show    2676
Name: type, dtype: int64
```

```
df[['type','director']].value_counts()
```

```
type     director
Movie    Rajiv Chilaka             19
         Raúl Campos, Jan Suter    18
         Suhas Kadav               16
         Marcus Raboy              15
         Jay Karas                 14
                                   ..
         Jasmine D'Souza            1
         Jason Bourque              1
         Jason Cohen                1
         Jason James                1
TV Show  Ziad Doueiri               1
Length: 4576, dtype: int64
```

```
df[['type','rating']].value_counts()
```

```
type     rating
Movie    TV-MA       2062
         TV-14       1427
TV Show  TV-MA       1145
Movie    R            797
TV Show  TV-14        733
Movie    TV-PG        540
         PG-13        490
TV Show  TV-PG        323
Movie    PG           287
TV Show  TV-Y7        195
         TV-Y         176
Movie    TV-Y7        139
         TV-Y         131
         TV-G         126
TV Show  TV-G          94
Movie    NR            75
         G             41
         TV-Y7-FV       5
TV Show  NR             5
Movie    NC-17          3
         UR             3
TV Show  R              2
Movie    66 min         1
         74 min         1
         84 min         1
TV Show  TV-Y7-FV       1
dtype: int64
```

```
movie_cast['cast'].nunique()
```

```
27880
```

```
tvshow_country['country'].nunique()
```

```
103
```

DATA TYPE CONVERSIONS

```
# converting date_added column to datetime data type to make date operation easier.

df['date_added'] = pd.to_datetime(df['date_added'])

df.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
					Sami Bouajila,						Crime TV	To protect his

```
# To drop specific column from the original dataset

df_drop = df.drop(columns=['show_id'])

df_drop.head()
```

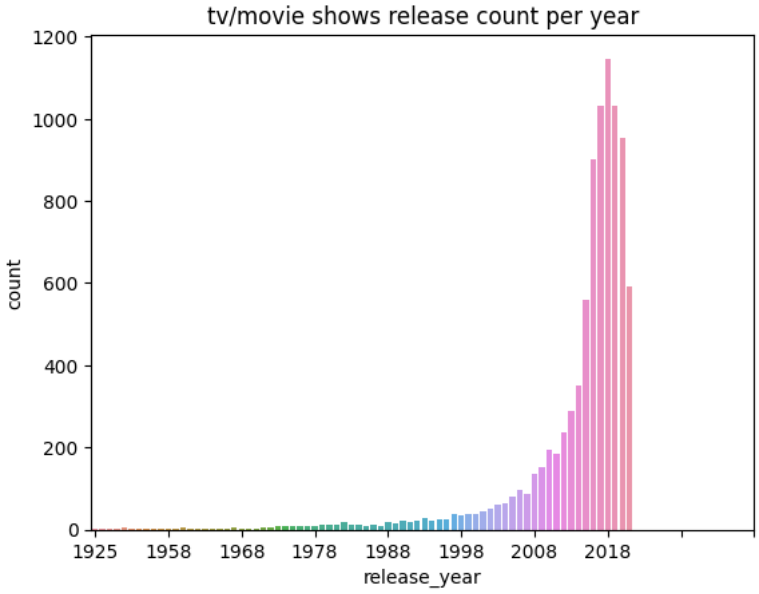
	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
2	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	2021-09-24	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...

```
# to check whether show-id is dropped from the original datasat

df_drop.keys()
```

```
Index(['type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

VISUAL ANALYSIS

```
# Count plot for shows released per year

sns.countplot(data = df, x='release_year')
plt.title('tv/movie shows release count per year')
plt.xticks(range(0,100,10))
plt.show()
```

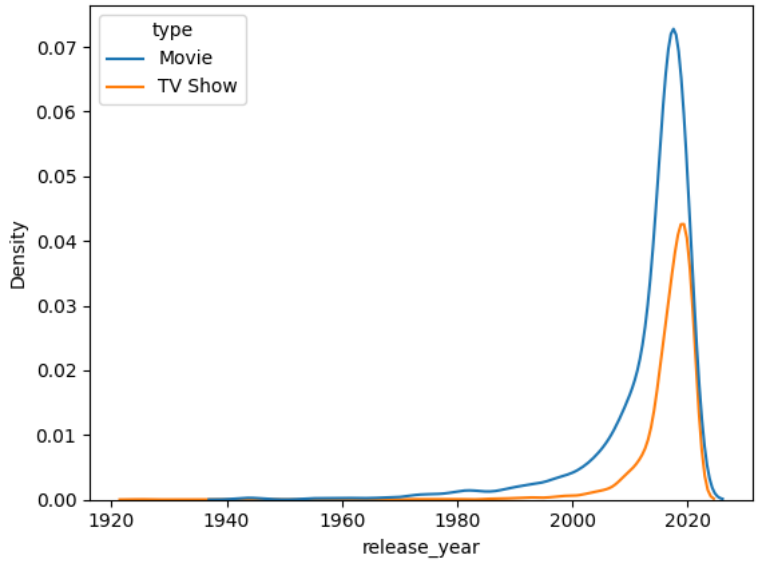


INFERENCE:

Number of movies released per year has a gradual growth between 1998 and 2008 but for the past 10 years there is an exponential growth with a slight surge after 2018

```
sns.kdeplot( data = df, x='release_year',hue='type')
```

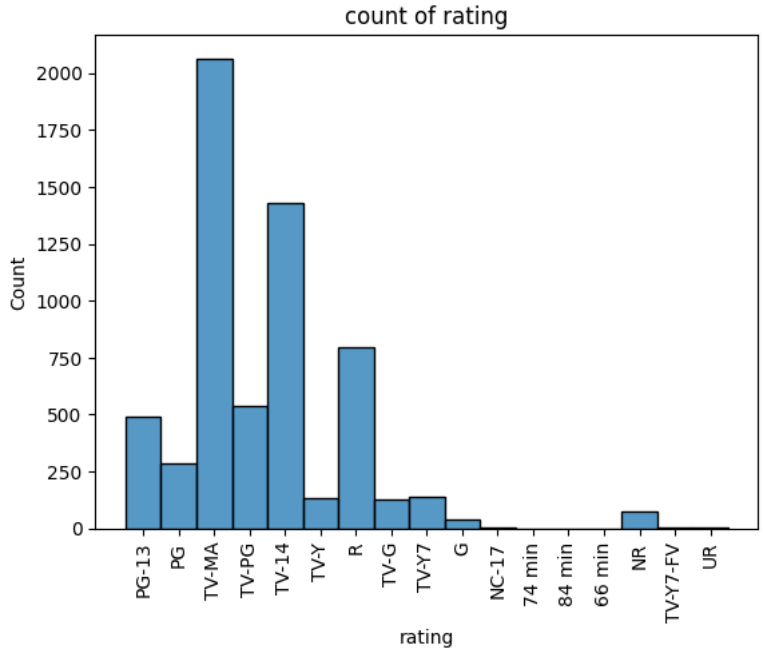
```
<Axes: xlabel='release_year', ylabel='Density'>
```




```
(df['rating']=='TV-MA').value_counts()
```

```
False    5600
True     3207
Name: rating, dtype: int64
```

```
# Histogram of ratings of all the movies

sns.histplot(movie['rating'],bins=20)
plt.xticks(rotation=90)
plt.title('count of rating')
plt.show()
```

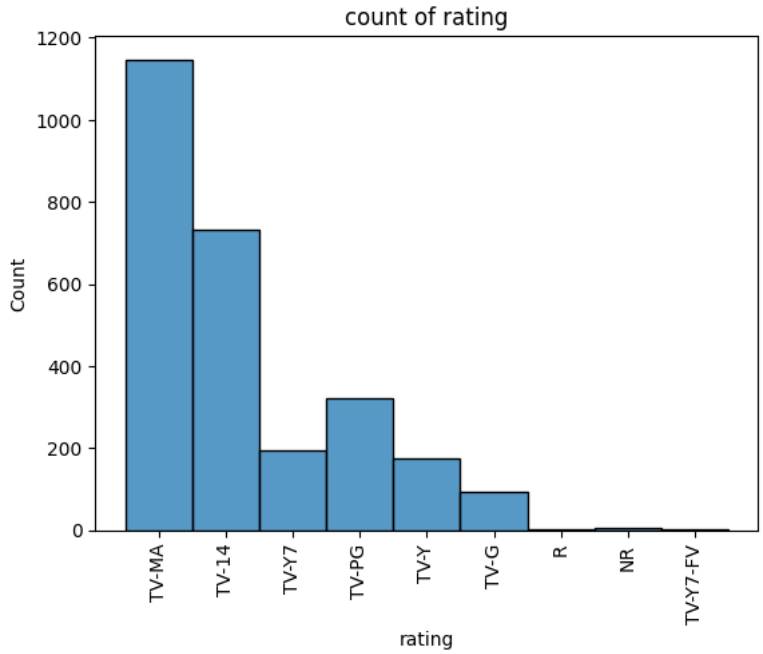


INFERENCE:

The movie with the rating "TV-MA" has a very good count(most popular) whereas ratings like "NC-17,74 min,84 min,66 min, TV-Y7-FV, UR" have a very poor count(least popular)

```
# Histogram of ratings of all the tv showa

sns.histplot(tvshow['rating'],bins=20)
plt.xticks(rotation=90)
```

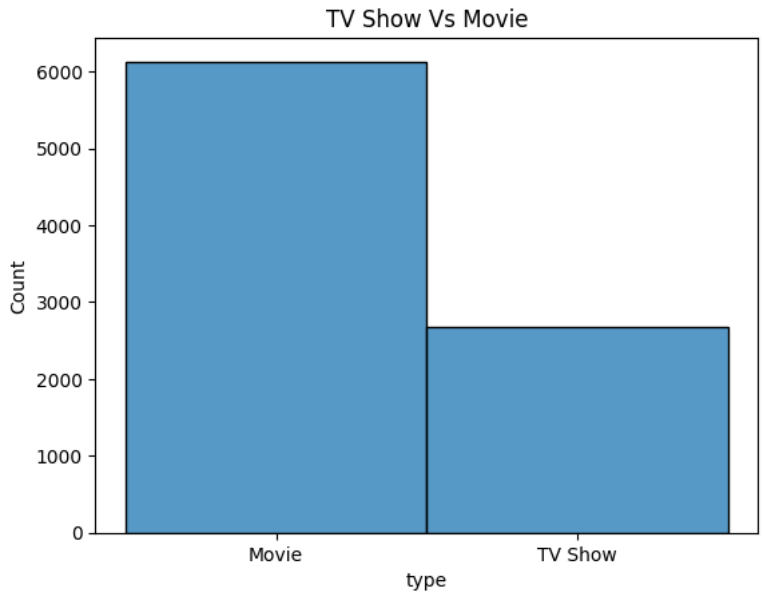
```
plt.title('count of rating')
plt.show()
```



INFERENCE:

The Tv show with the rating "TV-MA" has a very good count(most popular) whereas ratings like "R,TV-Y7-FV,NR" have a very poor count(least popular)

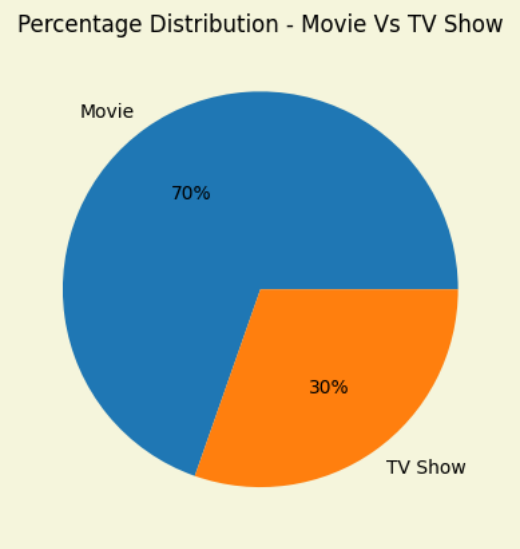
```
sns.histplot(data = df, x='type')
plt.title('TV Show Vs Movie')
plt.show()
```



INFERENCE:

It is more evident that Netflix has more focus on Movie rather than on TV Show thats why there are more movies released then that of TV Show. In other words, Movies are more popular than TV show so Netflix has to focus on launching new varieties of TV show to gain popularity

```
type_counts = df['type'].value_counts()

plt.figure(facecolor="beige", frameon=True)
plt.title("Percentage Distribution - Movie Vs TV Show")
plt.pie(type_counts.values, labels=type_counts.index, autopct='%.0f%%')
plt.show()
```



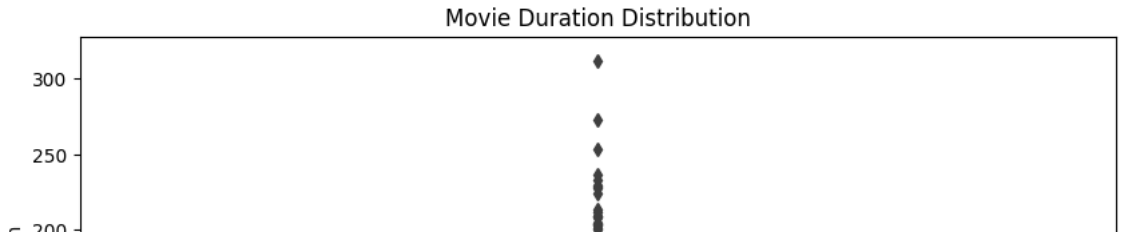
INFERENCE:

It is very obvious that TV Show contributes only 30% whereas Movie contributes 70%

```
df['movie_duration'] = np.where(df['type']=='Movie', df['duration'].str[:-4], None)
df['movie_duration'] = df['movie_duration'].astype('float')
df.sample(3)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
1600	s1601	Movie	Chico Bon Bon and the Very Berry Holiday	Darragh O'Connell	Robbie Daymond, Dayci Brookshire, Anthony Tede...	NaN	2020-12-03	2020	TV-Y	25 min	Children & Family Movies	The Fix-It Force makes a plan to hit every hom...
2894	s2895	TV Show	Puerta 7	NaN	Dolores Fonzi, Esteban Lamothe, Carlos	Argentina	2020-02-21	2020	TV-MA	1 Season	Crime TV Shows, International TV Shows,	A determined woman works to rid an Argentine

```
# Boxplot shows the median and the outliers

plt.figure(figsize=(10,5))
sns.boxplot(data=df, y='movie_duration')
plt.title('Movie Duration Distribution')
plt.show()
```



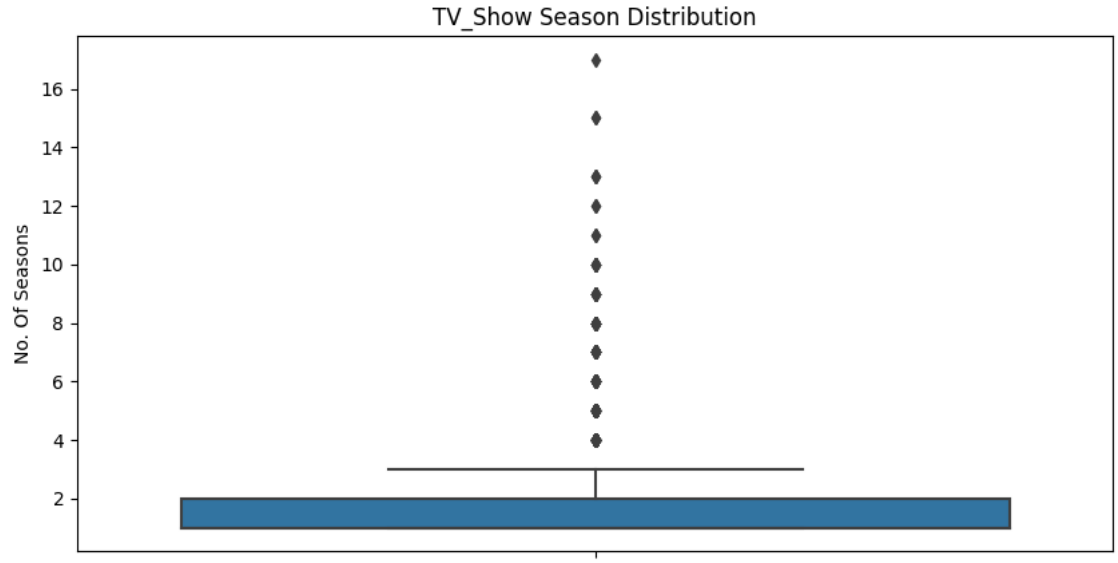
▼ INFERENCE:

The average duration of a movie is around 90 min where movies with duration as short as 10 min(approx.) and as long as 310 min(approx.) are also released.

```
df['tvshow_duration'] = np.where(df['type']=='TV Show', df['duration'].str[:-7], None)
df['tvshow_duration'] = df['tvshow_duration'].astype('float')
df.sample(3)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	movi
5464	s5465	TV Show	The Day Will Come	NaN	Sofie Gråbøl, Lars Mikkelsen, Harald Kaiser He...	Denmark	2017-06-01	2016	TV-14	1 Season	International TV Shows, TV Dramas	In 1967, two brothers from Copenhagen, Denmark...	
2947	s2948	TV Show	Arrow	James Bamford	Stephen Amell, Katie Cassidy, David Ramsey, Wi...	United States	2020-02-05	2019	TV-14	8 Seasons	Crime TV Shows, TV Action & Adventure	Based on DC Comics' Green Arrow, an affluent p...	

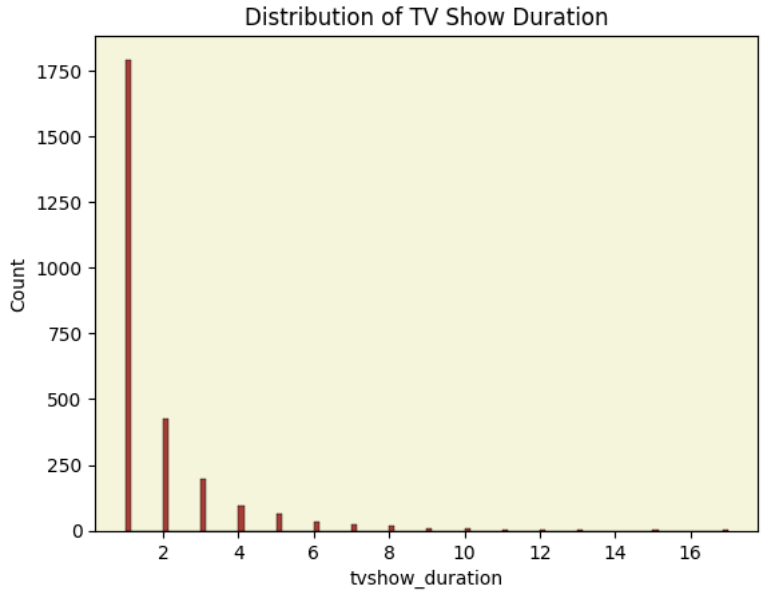
```
# Boxplot shows the median and the outliers

plt.figure(figsize=(10,5))
sns.boxplot(data=df, y='tvshow_duration')
plt.ylabel('No. Of Seasons')
plt.title('TV_Show Season Distribution')
plt.show()
```



▼ INFERENCE:

Most of the shows has 1 or 2 seasons with few exceptions where a show has even 17 seasons too.

```
sns.histplot(data=df, x='tvshow_duration',color='darkred')
plt.title('Distribution of TV Show Duration')
ax = plt.gca()
ax.set_facecolor('Beige')
plt.show()
```




```
uniq_director = df.groupby('director')['show_id'].count().sort_values(ascending = False)
uniq_director
```

```
director
Rajiv Chilaka                             19
Raúl Campos, Jan Suter                    18
Suhas Kadav                               16
Marcus Raboy                              16
Jay Karas                                 14
                                          ..
Jos Humphrey                               1
Jose Gomez                                 1
Jose Javier Reyes                          1
Joseduardo Giordano, Sergio Goyri Jr.      1
Khaled Youssef                             1
Name: show_id, Length: 4528, dtype: int64
```

```
india_df = df[df['country']=='India']

india_df.head(2)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	movie_
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-24	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train I...	
					Prashanth, Aishwarya						Comedies,	When the	

```
india_df['director'].nunique()
```

```
645
```

```
mov_india_df = india_df.where(india_df['type'] == 'Movie')

mov_india_df.sample(2)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	movi
3299	s3300	Movie	U Turn	Pawan Kumar	Roger Narayan, Shraddha Srinath, Dileep Raj, K...	India	2019-11-07	2016.0	TV-14	121 min	International Movies, Thrillers	A reporter must hunt for the truth behind a st...	

Rajkummar

```
director_df = mov_india_df.groupby('director').agg({'title':'count'}).sort_values('title',ascending = False).reset_index()
director_df.rename(columns = {'title':'count'},inplace=True)

director_df.head(3)
```

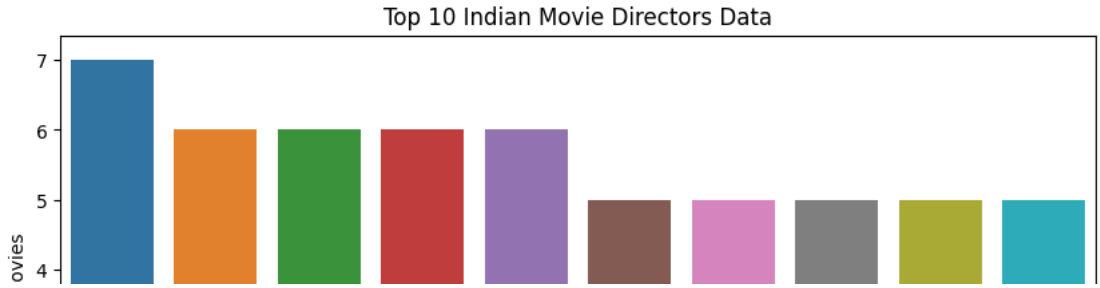
	director	count
0	David Dhawan	9
1	Ram Gopal Varma	7
2	Anees Bazmee	6

```
top10_director = director_df[1:11]

top10_director
```

	director	count
1	Ram Gopal Varma	7
2	Anees Bazmee	6
3	Rajkumar Santoshi	6
4	Sooraj R. Barjatya	6
5	Imtiaz Ali	6
6	Anurag Kashyap	5
7	Ashutosh Gowariker	5
8	Indra Kumar	5
9	Rohit Shetty	5
10	Umesh Mehra	5

```
plt.figure(figsize=(10,5))
sns.barplot(data = top10_director, x='director', y='count')
plt.xticks(rotation=90)
plt.ylabel('No.Of Movies')
plt.title('Top 10 Indian Movie Directors Data')
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



INFERERNCE:

Above plot gives details about the most active and successful movie directors in India

```
temp = df[df['type'] == 'Movie']
temp.head(2)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	mov
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...	
6	s7	Movie	My Little Pony: A	Robert Cullen,	Vanessa Hudgens, Kimiko Glenn	NaN	2021-09-24	2021	PG	91 min	Children &	Equestria's divided. But	

```
mov_world_df = temp[temp['country'] != 'India']

mov_world_df.sample(2)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	m
1497	s1498	Movie	Hello, Love, Goodbye	Cathy Garcia-Molina	Kathryn Bernardo, Alden Richards, Maymay Entra...	Philippines	2020-12-24	2019	TV-14	117 min	Dramas, International Movies, Romantic Movies	In Hong Kong, the lives of two overseas Filipi...	
			Bad	David	Jack Cutmore-Scott, Lili	Singapore,						A player who uses the	

```
world_director_df = mov_world_df.groupby('director').agg({'title':'count'}).sort_values('title',ascending = False).reset_index()
world_director_df.rename(columns = {'title':'count'},inplace=True)

world_director_df.head(3)
```

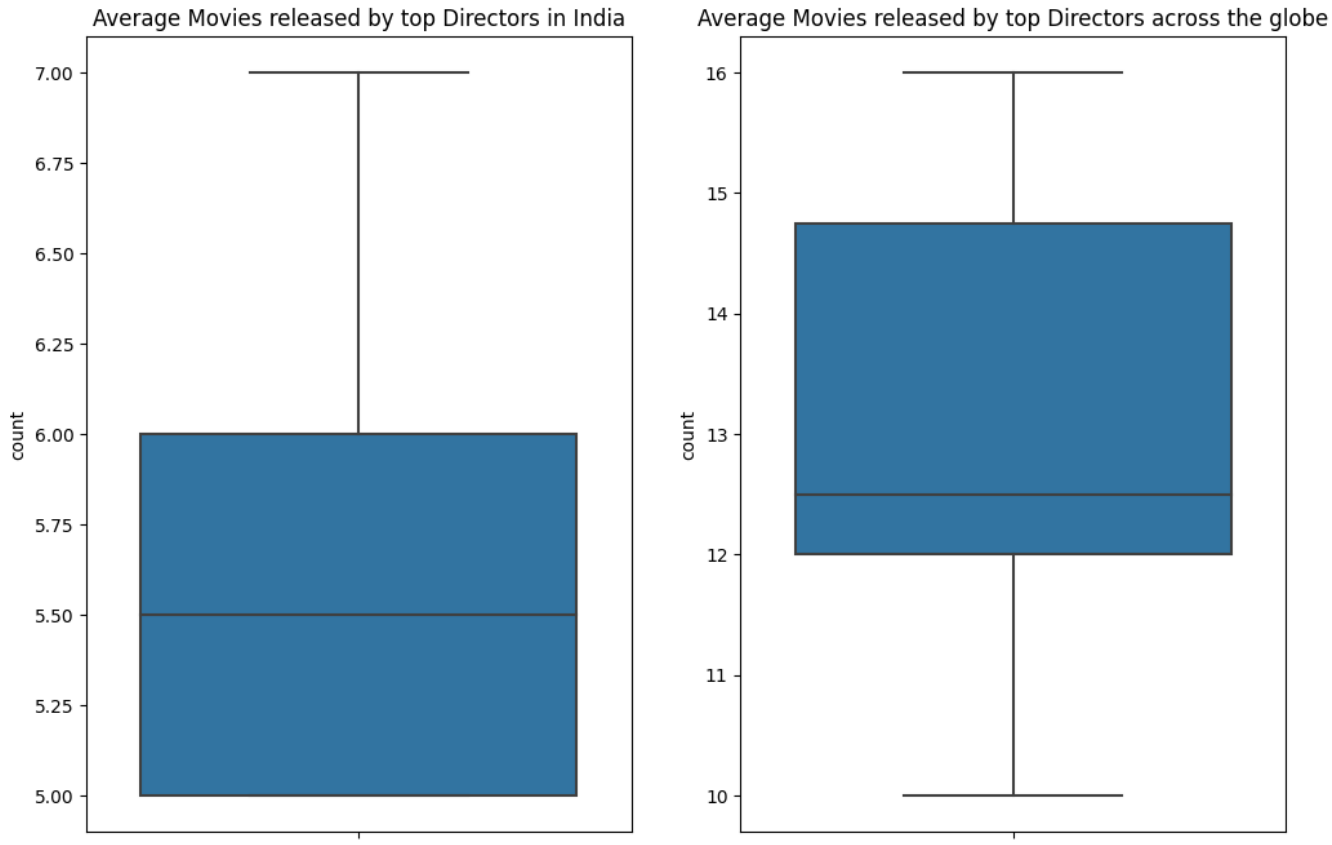
	director	count
0	Raúl Campos, Jan Suter	18
1	Rajiv Chilaka	16
2	Suhas Kadav	15

```
top10_world_director = world_director_df[1:11]

top10_world_director
```

	director	count
1	Rajiv Chilaka	16
2	Suhas Kadav	15
3	Marcus Raboy	15
4	Jay Karas	14
5	Cathy Garcia-Molina	13
6	Youssef Chahine	12
7	Martin Scorsese	12
8	Jay Chapman	12
9	Steven Spielberg	11

```
plt.figure(figsize=(12,8))

plt.subplot(1,2,1)
sns.boxplot(data = top10_director, y='count')
plt.title('Average Movies released by top Directors in India')

plt.subplot(1,2,2)
sns.boxplot(data = top10_world_director, y='count')
plt.title('Average Movies released by top Directors across the globe')
```

```
Text(0.5, 1.0, 'Average Movies released by top Directors across the globe')
```

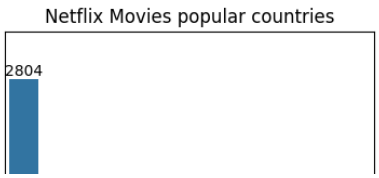


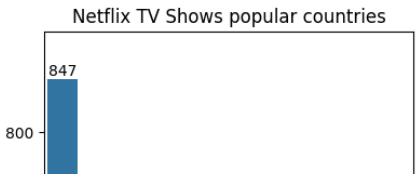
INFERENCE:

By comparing the BOXPLOTS of average movies by top directors from India and rest of the world, we can see that average release of Indian directors is only 6 movies whereas that of directors from rest of the world is 15 movies . so, Netflix to get new movies released often , they have to focus on international movies.

```
# countries where netflix is popular

# select top 10 countries

mov_top10_country = movie_country['country'].value_counts().head(10)
tv_top10_country = tvshow_country['country'].value_counts().head(10)
```

```
# plotting barchart

plt.figure(figsize=(15,12))


plt.subplot(1,3,1)
barplot1 = sns.barplot(x=mov_top10_country.index, y=mov_top10_country.values)
plt.title('Netflix Movies popular countries')
plt.xticks(rotation = 90)
plt.xlabel('Country')
plt.ylabel('Movies Count')



plt.subplot(1,3,3)
barplot2 = sns.barplot(x=tv_top10_country.index, y=tv_top10_country.values)
plt.title('Netflix TV Shows popular countries')
plt.xticks(rotation = 90)
plt.xlabel('Country')
plt.ylabel('TV Show Count')


# to add values at top of each bars

for index, value in enumerate(mov_top10_country.values):
    barplot1.text(index, value, str(value), ha='center', va='bottom')


for index, value in enumerate(tv_top10_country.values):
    barplot2.text(index, value, str(value), ha='center', va='bottom')




plt.show()
```

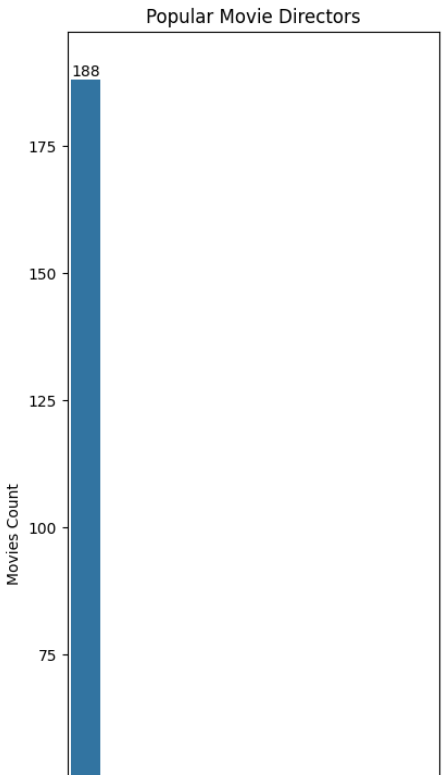


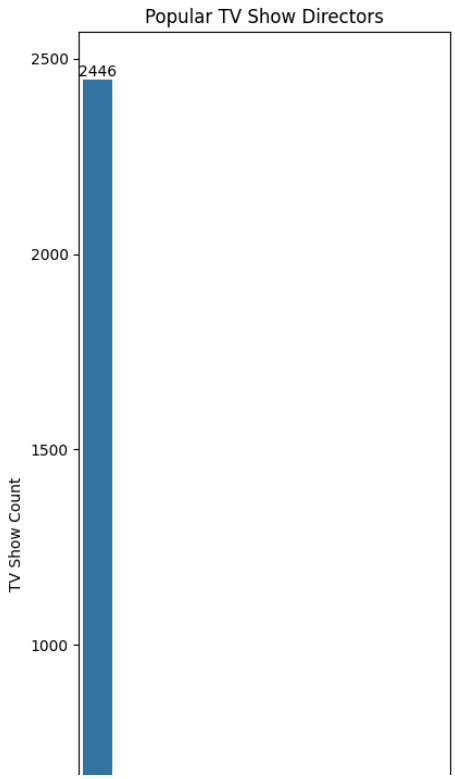


INFERENCE:

In either cases Netflix is most popular in United States. Netflix Movies are second most popular in India. Netflix TV Shows are second most popular in United Kingdom.

```
# popular directors

# select top 10 directors

mov_top10_director = movie_director['director'].value_counts().head(10)
tv_top10_director = tvshow_director['director'].value_counts().head(10)


# plotting barchart

plt.figure(figsize=(15,12))


plt.subplot(1,3,1)
barplot1 = sns.barplot(x=mov_top10_director.index, y=mov_top10_director.values)
plt.title('Popular Movie Directors')
plt.xticks(rotation = 90)
plt.xlabel('Directors')
plt.ylabel('Movies Count')



plt.subplot(1,3,3)
barplot2 = sns.barplot(x=tv_top10_director.index, y=tv_top10_director.values)
plt.title('Popular TV Show Directors')
plt.xticks(rotation = 90)
plt.xlabel('Directors')
plt.ylabel('TV Show Count')


# to add values at top of each bars

for index, value in enumerate(mov_top10_director.values):
    barplot1.text(index, value, str(value), ha='center', va='bottom')


for index, value in enumerate(tv_top10_director.values):
    barplot2.text(index, value, str(value), ha='center', va='bottom')




plt.show()
```

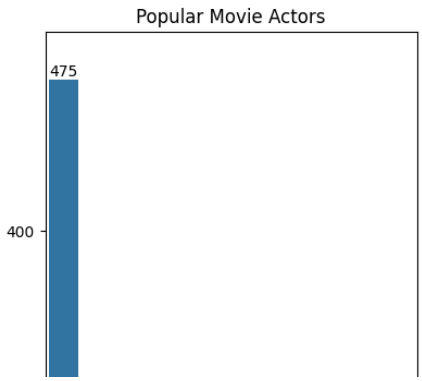


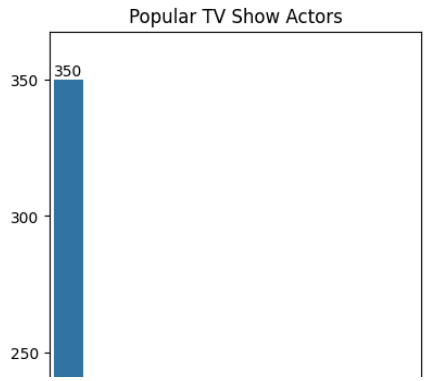


INFERENCE:

Movies by Rajiv chilaka occupies most of the Netflix's content. Records of TV Show director is not sufficient

```
# popular Actors

# select top 10 actors

mov_top10_cast = movie_cast['cast'].value_counts().head(10)
tv_top10_cast = tvshow_cast['cast'].value_counts().head(10)


# plotting barchart

plt.figure(figsize=(15,12))


plt.subplot(1,3,1)
barplot1 = sns.barplot(x=mov_top10_cast.index, y=mov_top10_cast.values)
plt.title('Popular Movie Actors')
plt.xticks(rotation = 90)
plt.xlabel('cast')
plt.ylabel('Movies Count')



plt.subplot(1,3,3)
barplot2 = sns.barplot(x=tv_top10_cast.index, y=tv_top10_cast.values)
plt.title('Popular TV Show Actors')
plt.xticks(rotation = 90)
plt.xlabel('cast')
plt.ylabel('TV Show Count')


# to add values at top of each bars

for index, value in enumerate(mov_top10_cast.values):
    barplot1.text(index, value, str(value), ha='center', va='bottom')


for index, value in enumerate(tv_top10_cast.values):
    barplot2.text(index, value, str(value), ha='center', va='bottom')




plt.show()
```

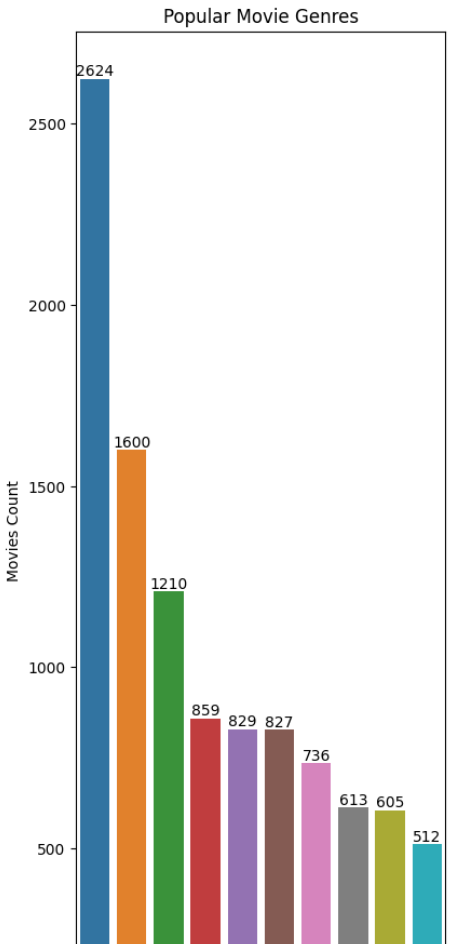


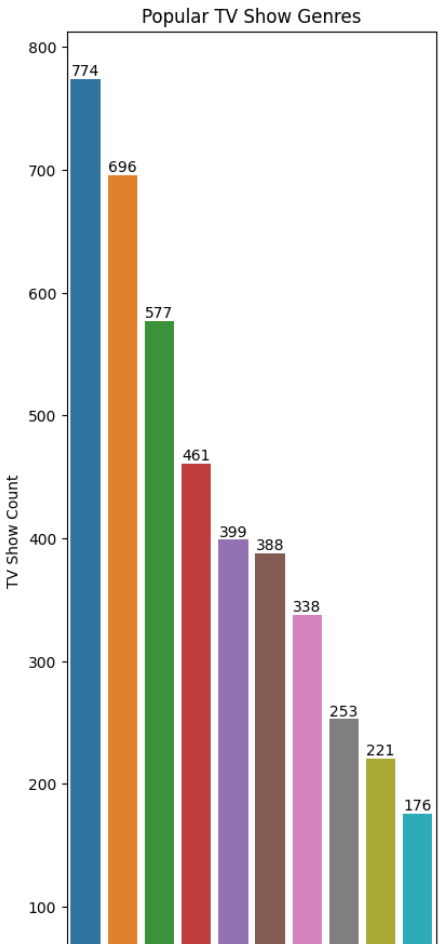


INFERENCE:

Most of the top 10 Actors are from India and Anupam Kher is casted in most Movies. Takahiro Sakurai is casted in most TV Shows

```
# popular genre

# select top 10 genres

mov_top10_listed_in = movie_listed_in['listed_in'].value_counts().head(10)
tv_top10_listed_in = tvshow_listed_in['listed_in'].value_counts().head(10)


# plotting barchart

plt.figure(figsize=(15,12))


plt.subplot(1,3,1)
barplot1 = sns.barplot(x=mov_top10_listed_in.index, y=mov_top10_listed_in.values)
plt.title('Popular Movie Genres')
plt.xticks(rotation = 90)
plt.xlabel('Genre')
plt.ylabel('Movies Count')



plt.subplot(1,3,3)
barplot2 = sns.barplot(x=tv_top10_listed_in.index, y=tv_top10_listed_in.values)
plt.title('Popular TV Show Genres')
plt.xticks(rotation = 90)
plt.xlabel('Genre')
plt.ylabel('TV Show Count')


# to add values at top of each bars

for index, value in enumerate(mov_top10_listed_in.values):
    barplot1.text(index, value, str(value), ha='center', va='bottom')


for index, value in enumerate(tv_top10_listed_in.values):
    barplot2.text(index, value, str(value), ha='center', va='bottom')




plt.show()
```



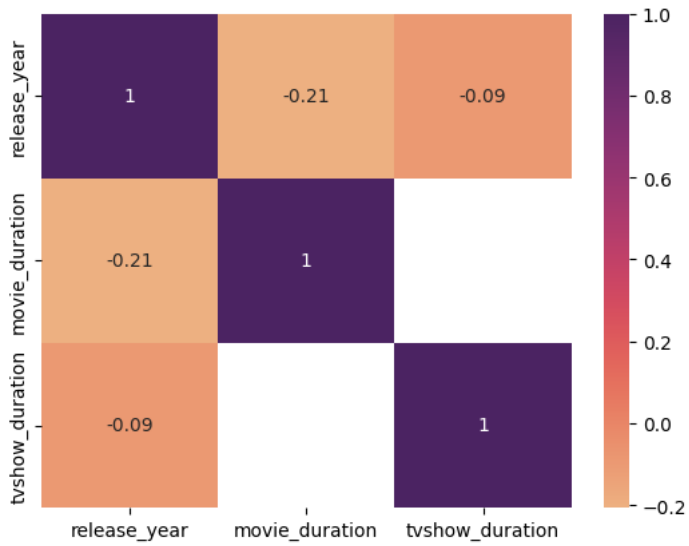


INFERENCE:

International Movies & TV Shows are the most popular genre followed by Dramas in Netflix

```
sns.heatmap(df.corr(), cmap='flare', annot = True)
plt.show()
```

```
<ipython-input-104-5c41d89ded57>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future v
  sns.heatmap(df.corr(), cmap='flare', annot = True)
```

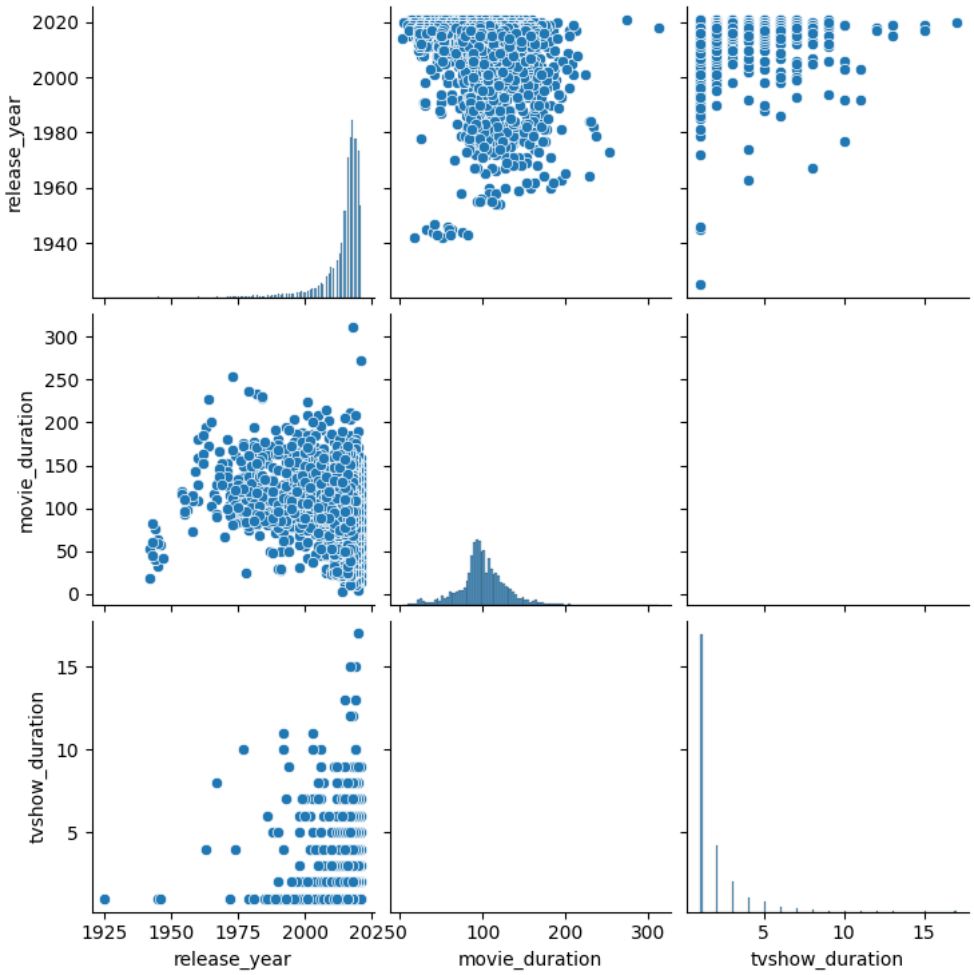


INFERENCE:

There is a weak correlation between tvshow_duration and release_year, which means there is not much increase in seasons released for a show with the years. There is a better correlatiob between movie_duration and release_year, which means the duration of movies extended a bit with the years.

```
# pairplot gives complete relation between all the range of statistical attributes in data
sns.pairplot(data = df)
```

```
<seaborn.axisgrid.PairGrid at 0x7bb070b6db40>
```



INFERENCE:

Earlier there were only 1 or 2 seasons released per tv show but now a days there are as high as 19 seasons too, and movie durations too increased from around 50 min to 200 min with a maximum duration of even 310 min is also avilable

MISSING VALUES:

```
df[df['duration'].isnull()]
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	movi
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	2017-04-04	2017	74 min	NaN	Movies	Louis C.K. muses on religion, eternal love, gi...	
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	2016-09-16	2010	84 min	NaN	Movies	Emmy-winning comedy writer Louis C.K. brings h...	

```
df['duration'].isna().sum()
```

```
3
```

```
df[df['duration'].isnull()].fillna('None')
```

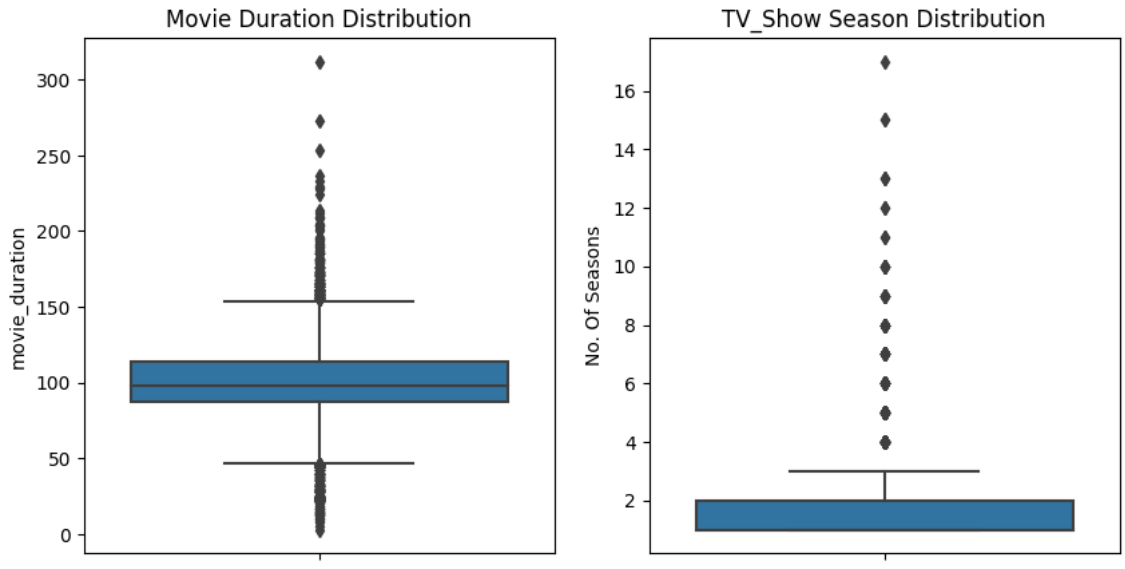
	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	movi
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	2017-04-04	2017	74 min	None	Movies	Louis C.K. muses on religion, eternal love, gi...	
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	2016-09-16	2010	84 min	None	Movies	Emmy-winning comedy writer Louis C.K. brings h...	

INFERENCE:

From the above table there are null values in duration but in respective rating column, it has values in minutes. There, is probably an error in the dataframe.

OUTLIER CHECK:

```
plt.figure(figsize=(10,5))

plt.subplot(1,2,1)
sns.boxplot(data=df, y='movie_duration')
plt.title('Movie Duration Distribution')


plt.subplot(1,2,2)
sns.boxplot(data=df, y='tvshow_duration')
plt.ylabel('No. Of Seasons')
plt.title('TV_Show Season Distribution')

plt.show()
```



INFERENCE:

Outliers gives idea about the exception cases. Here, in case of movies the outliers extend till from 5 min till 310 min. In case of TV shhowa it raises from 4 to 17 seasons at the max

```
df[df['movie_duration'] == 312 ]
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
4253	s4254	Movie	Black Mirror: Bandersnatch	NaN	Fionn Whitehead, Will Poulter, Craig Parkinson	United States	2018-12-28	2018	TV-MA	312 min	Dramas, International Movies, Sci-Fi & Fantasy	In 1984, a young programmer begins to question

```
df[df['movie_duration'] == 3 ]
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	movie_dura
3777	s3778	Movie	Silent	Limbert Fabian, Brandon	NaN	United States	2019-06-04	2014	TV-Y	3 min	Children & Family Movies, Sci-Fi &	"Silent" is an animated short film	

```
df[df['tvshow_duration'] == 17 ]
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	movie_
548	s549	TV Show	Grey's Anatomy	NaN	Ellen Pompeo, Sandra Oh, Katherine	United States	2021-07-03	2020	TV-14	17 Seasons	Romantic TV Shows, TV	Intern (and eventual resident) Meredith	

INFERENCE:

Either its the movie or TV show, the major outliers(max or min) are from UNITED STATES

```
df.keys()
```

```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description',
       'movie_duration', 'tvshow_duration'],
      dtype='object')
```

Relation between two attributes

```
df.groupby('release_year').agg({'title':'count'}).reset_index()
```

	release_year	title
0	1925	1
1	1942	2
2	1943	3
3	1944	3
4	1945	4
...	...	...
69	2017	1032
70	2018	1147
71	2019	1030
72	2020	953
73	2021	592

74 rows × 2 columns

INFERENCE:

The no of movies released per year increased with the years

```
data_movie = movie.groupby(['country','release_year']).agg({'title':'count'}).sort_values('title', ascending = False).reset_index()
data_movie.rename(columns={'title':'count'}, inplace = True)
data_movie
```

	country	release_year	count
0	United States	2017	321
1	United States	2018	300
2	United States	2019	285
3	United States	2020	233
4	United States	2016	230
...	...	...	...
1477	Israel, United States	2015	1
1478	Israel, Sweden, Germany, Netherlands	2015	1
1479	Israel, Germany, Poland, Luxembourg, Belgium, ...	2013	1
1480	Israel, Germany, France	2016	1
1481	Zimbabwe	2017	1

1482 rows × 3 columns

INFERENCE:

For the past 5 Years maximum movies contents in Netflix are from United States

```
data_tvshow = tvshow.groupby(['country','release_year']).agg({'title':'count'}).sort_values('title', ascending = False).reset_index()
data_tvshow.rename(columns={'title':'count'}, inplace = True)
data_tvshow
```

	country	release_year	count
0	United States	2020	159
1	United States	2019	134
2	United States	2018	110
3	UNKNOWN	2021	101
4	United States	2021	89
...	...	...	...
584	Japan	2002	1
585	Japan	2001	1
586	Japan	2000	1
587	Japan	1999	1
588	Uruguay, Germany	2021	1

589 rows × 3 columns

INFERENCE:

For the past 4 Years maximum TV Show contents in Netflix are from United States

```
df.groupby(['country','director']).agg({'title':'count'}).reset_index().sort_values('title',ascending = False)
```

	country	director	title
3709	United States	Marcus Raboy	15
3390	United States	Jay Karas	14
1949	Philippines	Cathy Garcia-Molina	13
3389	United States	Jay Chapman	12
1796	Mexico	Raúl Campos, Jan Suter	9

```
df.describe()
```

	release_year	movie_duration	tvshow_duration
count	8807.000000	6128.000000	2676.000000
mean	2014.180198	99.577187	1.764948
std	8.819312	28.290593	1.582752
min	1925.000000	3.000000	1.000000
25%	2013.000000	87.000000	1.000000
50%	2017.000000	98.000000	1.000000
75%	2019.000000	114.000000	2.000000
max	2021.000000	312.000000	17.000000

```
df[df['movie_duration'] == 312 ]
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
4253	s4254	Movie	Black Mirror: Bandersnatch	NaN	Fionn Whitehead, Will Poulter, Craig Parkinson	United States	2018-12-28	2018	TV-MA	312 min	Dramas, International Movies, Sci-Fi & Fantasy	In 1984, a young programmer begins to question

```
movie.groupby('release_year').agg({'title':'count'}).reset_index().sort_values('title',ascending = False)
```

	release_year	title
69	2018	767
68	2017	767
67	2016	658
70	2019	633
71	2020	517
...	...	...
12	1961	1
14	1963	1
17	1966	1
5	1947	1
4	1946	1

73 rows × 2 columns

```
movie['date_added'] = pd.to_datetime(movie['date_added'])
movie['month'] = movie['date_added'].dt.month_name()
movie.groupby('month').agg({'title':'count'}).reset_index().sort_values('title',ascending = False)
```

```
<ipython-input-125-2ffe52d797e9>: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
  movie['date_added'] = pd.to_datetime(movie['date_added'])
<ipython-input-125-2ffe52d797e9>:2: 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
  movie['month'] = movie['date_added'].dt.month_name()
```

	month	title
5	July	565
0	April	550
2	December	547
4	January	546
10	October	545
7	March	529
1	August	519

INFERENCE:

Most of the Movies are released in the month of July followed by April.

```
movie['day'] = movie['date_added'].dt.day_name()
movie.groupby('day').agg({'title':'count'}).reset_index().sort_values('title',ascending = False)
```

```
<ipython-input-126-da5abee6920b>: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
  movie['day'] = movie['date_added'].dt.day_name()
```

	day	title
0	Friday	1566
4	Thursday	1053
6	Wednesday	906
5	Tuesday	852
1	Monday	628
3	Sunday	569
2	Saturday	557

INFERENCE:

The most preferable day to release a movie is Friday and Thursday comes next.

```
movie[movie['rating']=='TV-MA'].sample(2)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	mo
7402	s7403	Movie	Mara	Clive Tonge	Olga Kurylenko, Javier Botet, Mitch Eakins	United Kingdom	2019-01-05	2017	TV-MA	99 min	Horror Movies, Thrillers	When criminal psychologist Kate Fuller investi	Janu

```
tvshow.groupby('release_year').agg({'title':'count'}).reset_index().sort_values('title',ascending = False)
```

	release_year	title
44	2020	436
43	2019	397
42	2018	380
45	2021	315
41	2017	265
40	2016	244
39	2015	162
38	2014	88
36	2012	64
37	2013	63
35	2011	40
34	2010	40
33	2009	34
32	2008	23
31	2007	14
30	2006	14
29	2005	13
27	2003	10
28	2004	9
26	2002	7
23	1999	7
25	2001	5
21	1997	4
17	1993	4
22	1998	4
24	2000	4
14	1990	3
16	1992	3
20	1996	3
18	1994	2
19	1995	2
12	1988	2
11	1986	2
7	1977	1
3	1963	1
2	1946	1
4	1967	1
5	1972	1
6	1974	1
13	1989	1
8	1979	1
9	1981	1
10	1985	1
15	1991	1

```
tvshow['date_added'] = pd.to_datetime(tvshow['date_added'])
tvshow['month'] = tvshow['date_added'].dt.month_name()
tvshow.groupby('month').agg({'title':'count'}).reset_index().sort_values('title',ascending = False)
```

```
<ipython-input-129-1c288893c8e3>: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
  tvshow['date_added'] = pd.to_datetime(tvshow['date_added'])
<ipython-input-129-1c288893c8e3>:2: 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
  tvshow['month'] = tvshow['date_added'].dt.month_name()
```

	month	title
5	July	272
2	December	266
11	September	251
1	August	236
6	June	236
10	October	215
0	April	214
7	March	213
9	November	207
8	May	193

INFERENCE:

December is the best month to release a TV Show

```
tvshow['day'] = tvshow['date_added'].dt.day_name()
tvshow.groupby('day').agg({'title':'count'}).reset_index().sort_values('title',ascending = False)
```

```
<ipython-input-130-96331c41cdf4>: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
  tvshow['day'] = tvshow['date_added'].dt.day_name()
```

	day	title
0	Friday	932
6	Wednesday	382
5	Tuesday	355
4	Thursday	343
2	Saturday	259
1	Monday	223
3	Sunday	182

INFERENCE:

Friday being the best day for the release of TV Show whereas Wednesday is the second best

```
tvshow[tvshow['rating']=='TV-MA'].sample(2)
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
3981	s3982	TV Show	Delhi Crime	UNKNOWN	Shefali Shah, Rajesh Tailang, Rasika Dugal, Ad...	India	2019-03-22	2019	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Dramas	As Delhi reels in the aftermath of a gang rape...
					Christian Malheiros						Crime TV Shows	Three teens

▾ BUSINESS INSIGHTS: