1: Problem Statement and Analysing basic metrics.

Data Import

In [81]:

```
# downloading data to working directory
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv
```

```
--2023-02-18 07:34:34--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv (h
ttps://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/original/netflix.csv)
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 99.84.170.22, 99.84.170.112, 99.84.170.176,
...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|99.84.170.22|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3399671 (3.2M) [text/plain]
Saving to: ‘netflix.csv.1’

netflix.csv.1       100%[===================>]   3.24M  --.-KB/s    in 0.07s

2023-02-18 07:34:34 (46.4 MB/s) - ‘netflix.csv.1’ saved [3399671/3399671]
```

In [82]:

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

In [83]:

```
# assigning data to object
df=pd.read_csv("/content/netflix.csv")
```

In [84]:

```
#Exploring first three rows of data set
df.head(3)
```

Out[84]:

	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 Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...

In [85]:

```
#Exploring last three rows of data set
df.tail(3)
```

Out[85]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	November 1, 2019	2009	R	88 min	Comedies, Horror Movies	Looking to survive in a world taken over by zo...
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020	2006	PG	88 min	Children & Family Movies, Comedies	Dragged from civilian life, a former superhero...
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019	2015	TV-14	111 min	Dramas, International Movies, Music & Musicals	A scrappy but poor boy worms his way into a ty...

2: Shape of data, data types of all the attributes,conversion of categorical attributes to 'category'missing value detection and statistical summary.

In [86]:

```
#shape of data set
df.shape
```

Out[86]:

```
(8807, 12)
```

In [87]:

```
# information about the data
# column names, datatypes, non-null values, 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
```

In [88]:

```
#count of null values in each column
df.isna().sum()
```

Out[88]:

```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             4
duration           3
listed_in          0
description        0
dtype: int64
```

In [89]:

```
# Total null values
df.isna().sum().sum()
```

Out[89]:

```
4307
```

In [90]:

```
# Statistical summary
df.describe(include = 'all',datetime_is_numeric=True)
```

Out[90]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
count	8807	8807	8807	6173	7982	7976	8797	8807.000000	8803	8804	8807	8807
unique	8807	2	8807	4528	7692	748	1767	NaN	17	220	514	8775
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	NaN	TV-MA	1 Season	Dramas, International Movies	Paranormal activity at a lush, abandoned prope...
freq	1	6131	1	19	19	2818	109	NaN	3207	1793	362	4
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2014.180198	NaN	NaN	NaN	NaN
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.819312	NaN	NaN	NaN	NaN
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1925.000000	NaN	NaN	NaN	NaN
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2013.000000	NaN	NaN	NaN	NaN
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2017.000000	NaN	NaN	NaN	NaN
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2019.000000	NaN	NaN	NaN	NaN
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2021.000000	NaN	NaN	NaN	NaN

In [91]:

```
# Statistical Summary
df.describe(include=object).T
```

Out[91]:

	count	unique	top	freq
show_id	8807	8807	s1	1
type	8807	2	Movie	6131
title	8807	8807	Dick Johnson Is Dead	1
director	6173	4528	Rajiv Chilaka	19
cast	7982	7692	David Attenborough	19
country	7976	748	United States	2818
date_added	8797	1767	January 1, 2020	109
rating	8803	17	TV-MA	3207
duration	8804	220	1 Season	1793
listed_in	8807	514	Dramas, International Movies	362
description	8807	8775	Paranormal activity at a lush, abandoned prope...	4

Data Type correction and dropping the 'description' column

In [92]:

```
# correction: by converting date_added in to datetime format
df['date_added'] = pd.to_datetime(df['date_added'])
```

In [93]:

```
# droping description columns from the dataframe
df.drop('description', axis = 1,inplace = True)
```

In [94]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 11 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   datetime64[ns]
 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
dtypes: datetime64[ns](1), int64(1), object(9)
memory usage: 757.0+ KB
```

missing value exploration

In [95]:

```
# missing value detection
df.loc[df['duration'].isna()]
```

Out[95]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	2017-04-04	2017	74 min	NaN	Movies
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	2016-09-16	2010	84 min	NaN	Movies
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	2016-08-15	2015	66 min	NaN	Movies

In [96]:

```
df['duration'][df['duration'].isna()] = df['rating'][df['duration'].isna()]
df["duration"]=df["duration"].apply(lambda x:str(x).split()[0])
df["duration"]=df["duration"].astype(int)
df
```

```
<ipython-input-96-0695f7581a9c>: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-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
  df['duration'][df['duration'].isna()] = df['rating'][df['duration'].isna()]
```

Out[96]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	90	Documentaries
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	2	International TV Shows, TV Dramas, TV Mysteries
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	2021-09-24	2021	TV-MA	1	Crime TV Shows, International TV Shows, TV Act...
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	2021-09-24	2021	TV-MA	1	Docuseries, Reality TV
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-24	2021	TV-MA	2	International TV Shows, Romantic TV Shows, TV ...
...	...	...	...	...	...	...	...	...	...	...	...
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	2019-11-20	2007	R	158	Cult Movies, Dramas, Thrillers
8803	s8804	TV Show	Zombie Dumb	NaN	NaN	NaN	2019-07-01	2018	TV-Y7	2	Kids' TV, Korean TV Shows, TV Comedies
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	2019-11-01	2009	R	88	Comedies, Horror Movies
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	2020-01-11	2006	PG	88	Children & Family Movies, Comedies
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	2019-03-02	2015	TV-14	111	Dramas, International Movies, Music & Musicals

8807 rows × 11 columns

In [97]:

```
# filling the missing value in date_added
df['date_added'] = df['date_added'].fillna(df['date_added'].max())
```

In [98]:

```
# filling the missing value in rating
df['rating'] = df['rating'].fillna(df['rating'].mode()[0])
```

In [99]:

```
# Cross checking of missing values
df.isna().sum()
```

Out[99]:

```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added         0
release_year       0
rating             0
duration           0
listed_in          0
dtype: int64
```

Comment:

1.country,cast and director columns have high number of missing values so the missing value operation will be done in the end of analysis

2.The most significant number of missing data is in the director column, which is unimportant for our current analysis. However, we will not erase this missing data so that we do not lose other important information regarding this data.

3:Non-Graphical Analysis: Value counts and unique attributes

In [100]:

```
df[df["type"]=="Movie"]["duration"].describe().round(2)
```

Out[100]:

```
count    6131.00
mean       99.56
std        28.29
min         3.00
25%        87.00
50%        98.00
75%       114.00
max       312.00
Name: duration, dtype: float64
```

In [101]:

```
df[df["type"]=="TV Show"]["duration"].describe().round(2)
```

Out[101]:

```
count    2676.00
mean        1.76
std         1.58
min         1.00
25%         1.00
50%         1.00
75%         2.00
max        17.00
Name: duration, dtype: float64
```

In [102]:

```
df.nunique()
```

Out[102]:

```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1714
release_year      74
rating            17
duration         210
listed_in        514
dtype: int64
```

In [103]:

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

Out[103]:

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

In [104]:

```
T1=df[["title","director","type"]]
T1["director"]=T1["director"].str.split(",")
T_1=T1.explode("director")
T_1
```

```
<ipython-input-104-3ee975dec179>: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-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
  T1["director"]=T1["director"].str.split(",")
```

Out[104]:

	title	director	type
0	Dick Johnson Is Dead	Kirsten Johnson	Movie
1	Blood & Water	NaN	TV Show
2	Ganglands	Julien Leclercq	TV Show
3	Jailbirds New Orleans	NaN	TV Show
4	Kota Factory	NaN	TV Show
...	...	...	...
8802	Zodiac	David Fincher	Movie
8803	Zombie Dumb	NaN	TV Show
8804	Zombieland	Ruben Fleischer	Movie
8805	Zoom	Peter Hewitt	Movie
8806	Zubaan	Mozez Singh	Movie

9612 rows × 3 columns

In [105]:

```
T_1["director"].value_counts()
```

Out[105]:

```
Rajiv Chilaka       22
Raúl Campos         18
 Jan Suter          18
Marcus Raboy        16
Suhas Kadav         16
                    ..
Will Eisenberg       1
Marina Seresesky     1
Kenny Leon           1
James Dearden        1
Mozez Singh          1
Name: director, Length: 5120, dtype: int64
```

In [106]:

```
T2=df[["title","cast","type"]]
T2["cast"]=T2["cast"].str.split(",")
T_2=T2.explode("cast")
T_2
```

```
<ipython-input-106-84c1c9056fa6>: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-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
  T2["cast"]=T2["cast"].str.split(",")
```

Out[106]:

	title	cast	type
0	Dick Johnson Is Dead	NaN	Movie
1	Blood & Water	Ama Qamata	TV Show
1	Blood & Water	Khosi Ngema	TV Show
1	Blood & Water	Gail Mabalane	TV Show
1	Blood & Water	Thabang Molaba	TV Show
...	...	...	...
8806	Zubaan	Manish Chaudhary	Movie
8806	Zubaan	Meghna Malik	Movie
8806	Zubaan	Malkeet Rauni	Movie
8806	Zubaan	Anita Shabdish	Movie
8806	Zubaan	Chittaranjan Tripathy	Movie

64951 rows × 3 columns

In [107]:

```
T_2["cast"].value_counts()
```

Out[107]:

```
 Anupam Kher              39
 Rupa Bhimani             31
 Takahiro Sakurai         30
 Julie Tejwani            28
 Om Puri                  27
                          ..
 Vedika                    1
 Tedros Teclebrhan         1
 Maryam Zaree              1
 Melanie Straub            1
 Chittaranjan Tripathy     1
Name: cast, Length: 39296, dtype: int64
```

In [108]:

```
T3=df[["title","country"]]
T3["country"]=T3["country"].str.split(",")
T_3=T3.explode("country")
T_3
```

```
<ipython-input-108-bca3d36199fa>: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-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
  T3["country"]=T3["country"].str.split(",")
```

Out[108]:

	title	country
0	Dick Johnson Is Dead	United States
1	Blood & Water	South Africa
2	Ganglands	NaN
3	Jailbirds New Orleans	NaN
4	Kota Factory	India
...	...	...
8802	Zodiac	United States
8803	Zombie Dumb	NaN
8804	Zombieland	United States
8805	Zoom	United States
8806	Zubaan	India

10850 rows × 2 columns

In [109]:

```
T_3["country"].value_counts()
```

Out[109]:

```
United States     3211
India             1008
United Kingdom     628
 United States     479
Canada             271
                  ...
 Ecuador             1
Iran                 1
Cyprus               1
 Mongolia            1
 Montenegro          1
Name: country, Length: 197, dtype: int64
```

In [110]:

```
T4=df[["title","listed_in","type"]]
T4["listed_in"]=T4["listed_in"].str.split(",")
T_4=T4.explode("listed_in")
T_4
```

```
<ipython-input-110-9eebfd11a41b>: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-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
  T4["listed_in"]=T4["listed_in"].str.split(",")
```

Out[110]:

	title	listed_in	type
0	Dick Johnson Is Dead	Documentaries	Movie
1	Blood & Water	International TV Shows	TV Show
1	Blood & Water	TV Dramas	TV Show
1	Blood & Water	TV Mysteries	TV Show
2	Ganglands	Crime TV Shows	TV Show
...	...	...	...
8805	Zoom	Children & Family Movies	Movie
8805	Zoom	Comedies	Movie
8806	Zubaan	Dramas	Movie
8806	Zubaan	International Movies	Movie
8806	Zubaan	Music & Musicals	Movie

19323 rows × 3 columns

In [111]:

```
T_4["listed_in"].value_counts()
```

Out[111]:

```
 International Movies       2624
Dramas                      1600
Comedies                    1210
Action & Adventure           859
Documentaries                829
                            ...
Romantic Movies                3
Spanish-Language TV Shows      2
LGBTQ Movies                   1
TV Sci-Fi & Fantasy            1
Sports Movies                  1
Name: listed_in, Length: 73, dtype: int64
```

In [112]:

```
df[df["type"]=="TV Show"]['release_year'].value_counts()
```

Out[112]:

```
2020    436
2019    397
2018    380
2021    315
2017    265
2016    244
2015    162
2014     88
2012     64
2013     63
2010     40
2011     40
2009     34
2008     23
2006     14
2007     14
2005     13
2003     10
2004      9
1999      7
2002      7
2001      5
1993      4
2000      4
1997      4
1998      4
1990      3
1996      3
1992      3
1995      2
1994      2
1988      2
1986      2
1989      1
1967      1
1985      1
1946      1
1981      1
1972      1
1979      1
1977      1
1991      1
1974      1
1925      1
1945      1
1963      1
Name: release_year, dtype: int64
```

In [113]:

```
df[df["type"]=="Movie"]['release_year'].value_counts()
```

Out[113]:

```
2017    767
2018    767
2016    658
2019    633
2020    517
       ... 
1966      1
1961      1
1946      1
1963      1
1947      1
Name: release_year, Length: 73, dtype: int64
```

In [114]:

```
df['date_added'] = pd.to_datetime(df["date_added"])
df['year'] = df['date_added'].dt.year
df['month'] = df['date_added'].dt.month
```

In [115]:

```
df[df["type"]=="TV Show"]['year'].value_counts()
```

Out[115]:

```
2020    595
2019    592
2021    515
2018    412
2017    349
2016    176
2015     26
2014      5
2013      5
2008      1
Name: year, dtype: int64
```

In [116]:

```
df[df["type"]=="Movie"]['year'].value_counts()
```

Out[116]:

```
2019    1424
2020    1284
2018    1237
2021     993
2017     839
2016     253
2015      56
2014      19
2011      13
2013       6
2012       3
2009       2
2008       1
2010       1
Name: year, dtype: int64
```

In [117]:

```
df[df["type"]=="TV Show"]['month'].value_counts()
```

Out[117]:

```
12    266
7     262
9     261
8     236
6     236
10    215
4     214
3     213
11    207
5     193
1     192
2     181
Name: month, dtype: int64
```

In [118]:

```
df[df["type"]=="Movie"]['month'].value_counts()
```

Out[118]:

```
7     565
4     550
12    547
1     546
10    545
3     529
9     519
8     519
11    498
6     492
5     439
2     382
Name: month, dtype: int64
```

In [119]:

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

Out[119]:

```
TV-MA       3211
TV-14       2160
TV-PG        863
R            799
PG-13        490
TV-Y7        334
TV-Y         307
PG           287
TV-G         220
NR            80
G             41
TV-Y7-FV       6
NC-17          3
UR             3
74 min         1
84 min         1
66 min         1
Name: rating, dtype: int64
```

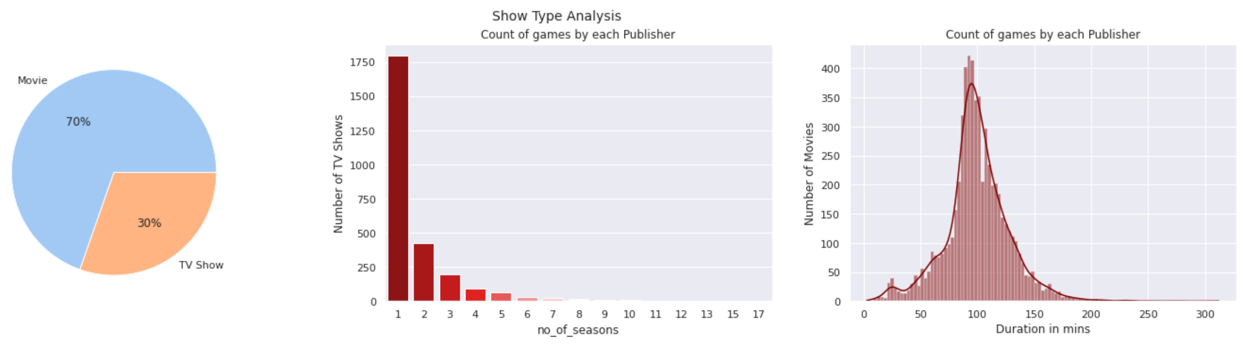
4:Visual Analysis - Univariate, Bivariate after pre-processing of the data

6: Insights based on Non-Graphical and Visual Analysis through comments

In [120]:

```
fig = plt.figure(figsize=(25,5))

plt.subplot(1, 3, 1)
data = df["type"].value_counts()
labels = ['Movie',"TV Show"]
colors = sns.color_palette('pastel')[0:7]
plt.pie(data, labels = labels, colors = colors, autopct='%.0f%%')

plt.subplot(1, 3, 2)
sns.countplot(df[df['type']=='TV Show']['duration'],x = 'duration',palette='seismic_r' )
plt.xlabel("no_of_seasons", fontsize=12)
plt.ylabel("Number of TV Shows", fontsize=12)
plt.title('Count of games by each Publisher', fontsize=12)

plt.subplot(1, 3, 3)
sns.histplot(df[df['type']=='Movie']['duration'],color='maroon',kde = True)
plt.xlabel("Duration in mins", fontsize=12)
plt.ylabel("Number of Movies", fontsize=12)
plt.title('Count of games by each Publisher', fontsize=12)

fig.suptitle('Show Type Analysis',fontsize=14)
plt.show()
```

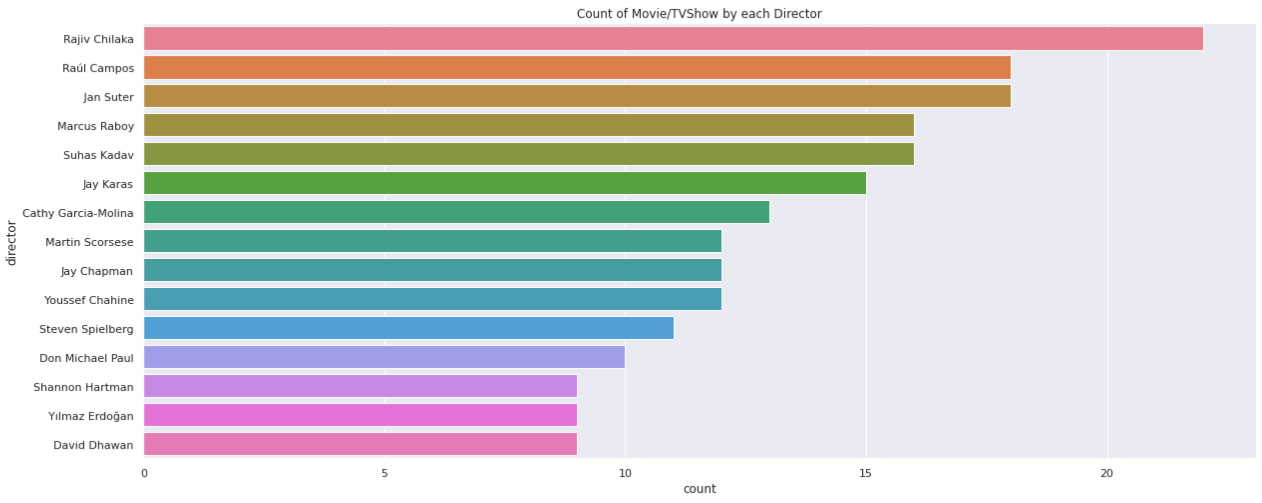
```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keywo
rd arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an
explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



###Comment: On Netflix 70% of content belongs to Movies and remaining 30% belongs to TV Show. The highest number of TV show having only one season and most of the movies duration varies from 50 minutes to 150 minutes

In [121]:

```
plt.figure(figsize=(20,8))
plt.title('Count of Movie/TVShow by each Director', fontsize=12)
sns.set(style="darkgrid")
ax = sns.countplot(y="director", data=T_1, palette="husl", order=T_1['director'].value_counts().index[0:15])

```

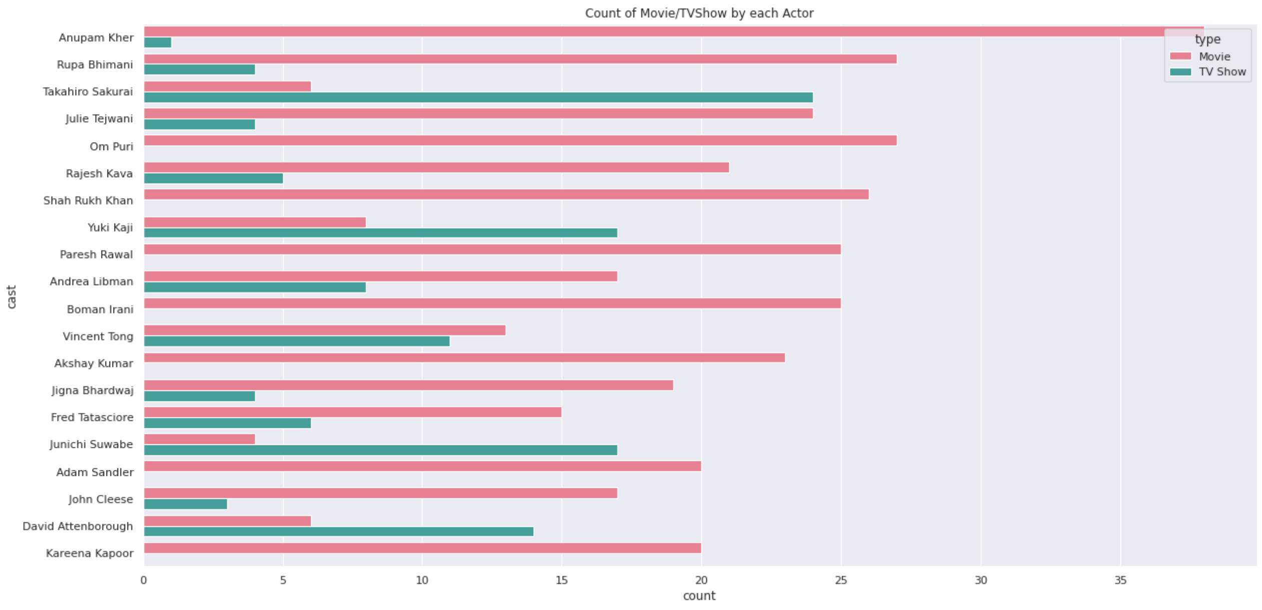


comment

Rajive Chilaka is the director who is having highest movies count on Netflix.

In [122]:

```
plt.figure(figsize=(20,10))
plt.title('Count of Movie/TVShow by each Actor', fontsize=12)
sns.set(style="darkgrid")
ax = sns.countplot(y="cast", data=T_2, palette="husl", order=T_2['cast'].value_counts().index[0:20],hue="type")
```



In [122]:

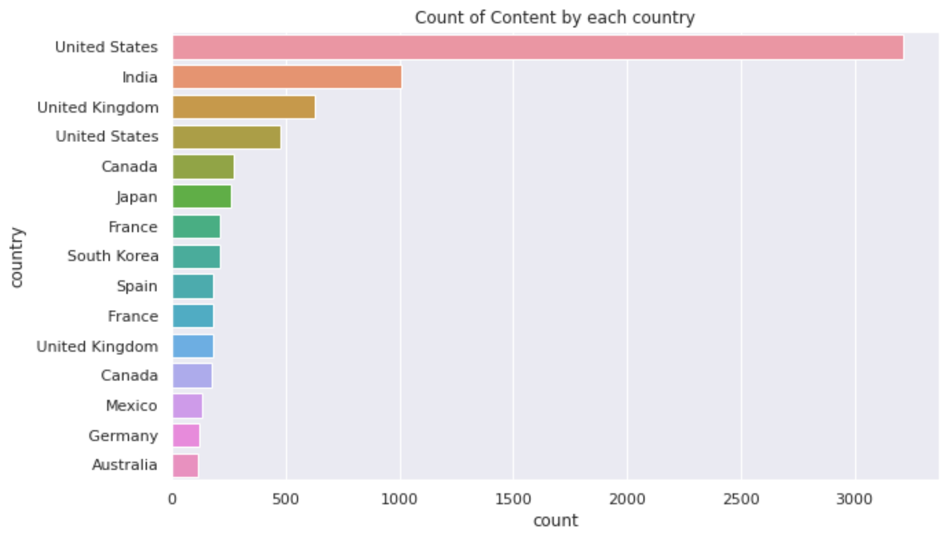
```

```

comment

In movie category, Anupum kher is the actor who is having highest movies count in movie category and in TV show category, Takahiro Sakurai is the actor who is having highest number of TV shows on Netflix.

In [123]:

```
plt.figure(figsize=(10,6))
plt.title('Count of Content by each country', fontsize=12)
sns.set(style="darkgrid")
ax = sns.countplot(y="country", data=T_3, order=T_3['country'].value_counts().index[0:15])
```

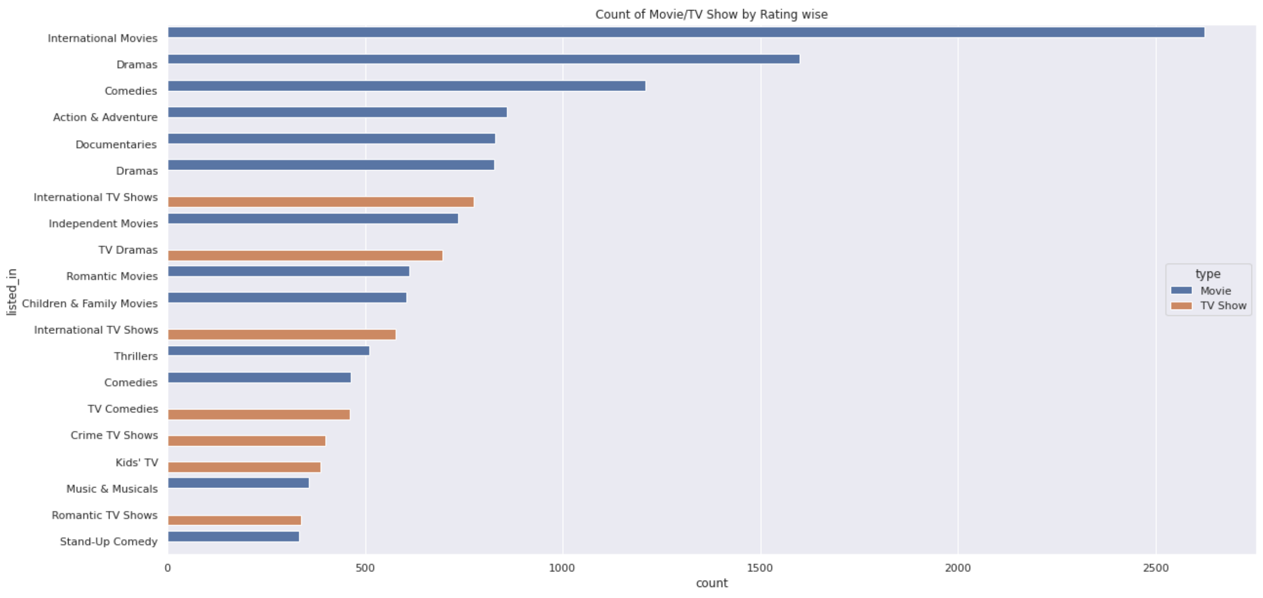


Comment

The Countplot shows that the United States is the leader in producing content for the platform, followed by India, the United Kingdom, and Canada.

In [124]:

```
plt.figure(figsize=(20,10))
sns.set(style="darkgrid")
plt.title('Count of Movie/TV Show by Rating wise', fontsize=12)
ax = sns.countplot(y="listed_in", data=T_4, order=T_4['listed_in'].value_counts().index[0:20],hue="type")
```



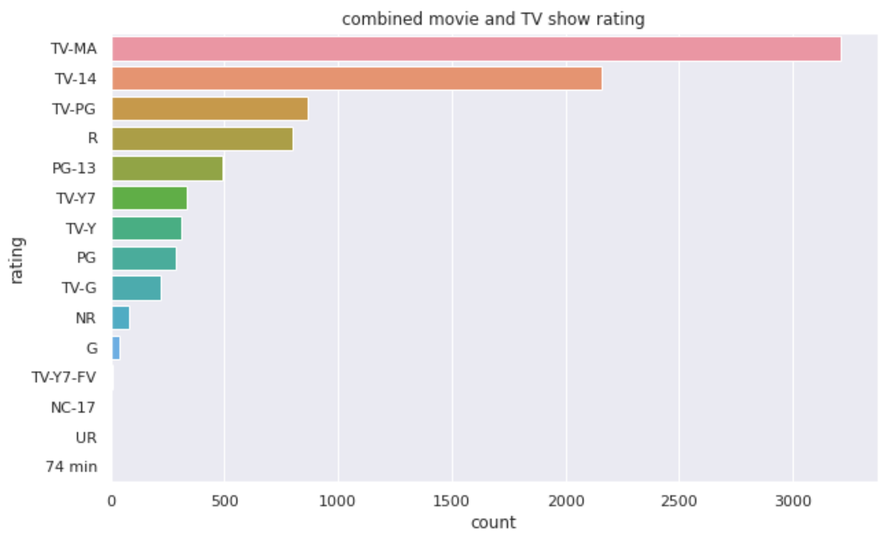
Comment:

The main genres that are always part of TV shows and also movies are International (films and TV shows), dramas, and comedies. Which sounds like genres you can see with a family members.

Comment:

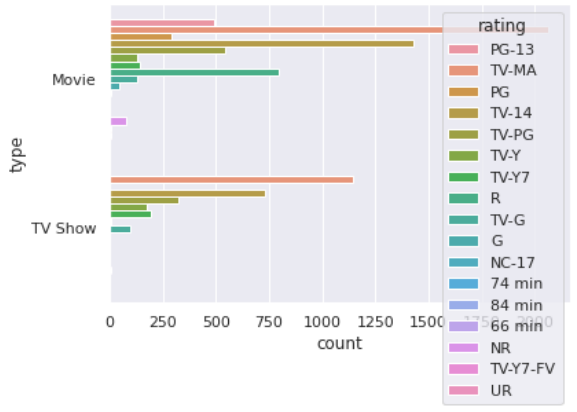
In [125]:

```
plt.figure(figsize=(10,6))
sns.set(style="darkgrid")
plt.title("combined movie and TV show rating")
ax = sns.countplot(y="rating", data=df, order=df['rating'].value_counts().index[0:15])
```



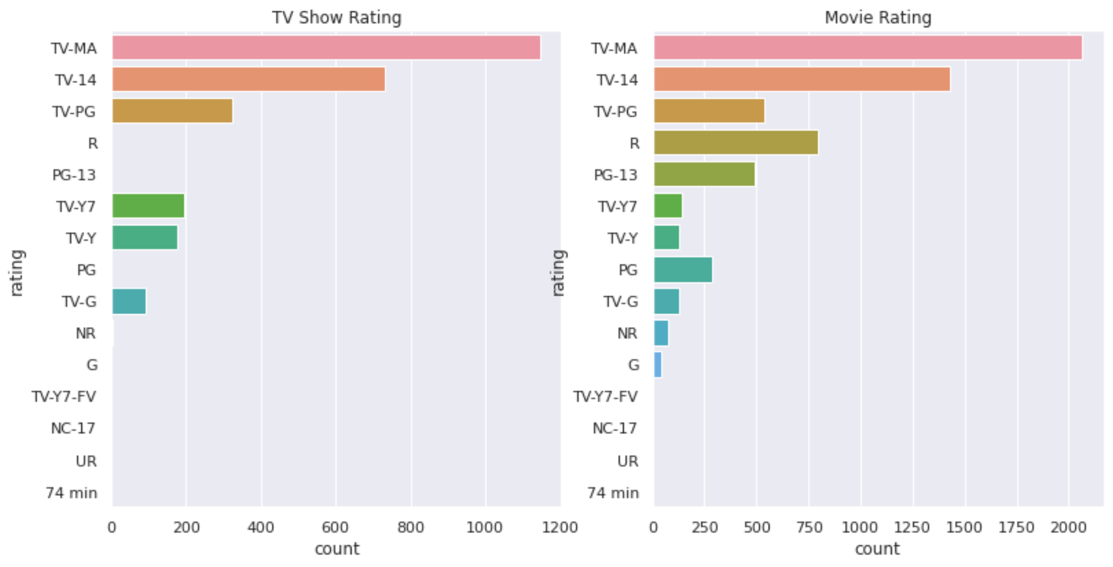
In [126]:

```
sns.countplot(data=df, y="type", hue="rating")
plt.show()
```



In [127]:

```
plt.figure(figsize=(20,14))
plt.subplot(2, 3, 1)
sns.set(style="darkgrid")
plt.xlabel("Rating count")
plt.ylabel("Rating")
plt.title("TV Show Rating");
ax = sns.countplot(y="rating", data=df[df["type"]=="TV Show"], order=df['rating'].value_counts().index[0:15])
plt.subplot(2, 3, 2)
plt.xlabel("Rating count")
plt.ylabel("Rating")
plt.title("Movie Rating");
sns.set(style="darkgrid")
ax = sns.countplot(y="rating", data=df[df["type"]=="Movie"], order=df['rating'].value_counts().index[0:15])
```

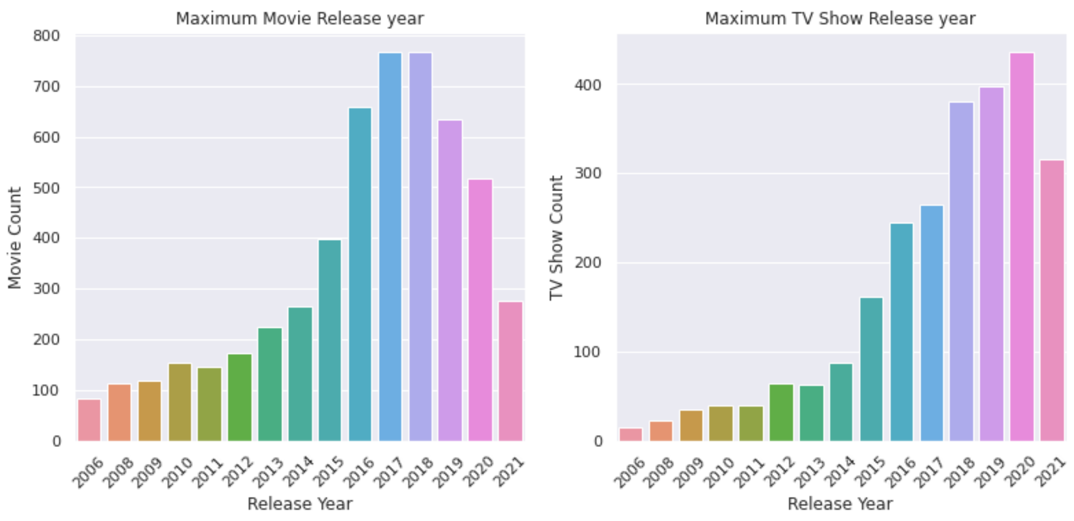


Comment:

Most of the content available is for the adult audience (TV-MA). Another large portion is the TV-14 classification, that is, programs that may contain material considered inappropriate for children under 14 years of age because they may contain moderate violence and offensive language. We can say with this that the massive audience of Netflix is made up of the adult audience.

In [128]:

```
plt.figure(figsize=(20,12))
plt.subplot(2, 3, 1)
sns.barplot(data=df, x=df[df["type"]=="Movie"]['release_year'].value_counts().index[:15],
            y=df[df["type"]=="Movie"]['release_year'].value_counts().head(15))
plt.xticks(rotation=45)
plt.xlabel("Release Year")
plt.ylabel("Movie Count")
plt.title("Maximum Movie Release year");
plt.subplot(2, 3, 2)
sns.barplot(data=df, x=df[df["type"]=="TV Show"]['release_year'].value_counts().index[:15],
            y=df[df["type"]=="TV Show"]['release_year'].value_counts().head(15))
plt.xticks(rotation=45)
plt.xlabel("Release Year")
plt.ylabel("TV Show Count")
plt.title("Maximum TV Show Release year");
```



comment:

As per our visual analysis trend of growth of number of movies and TV shows was similar till 2016.Afterword growth of movies was at a higher pace compared to growth of TV show.As of 2019, there is a drop in movie viewing. In 2020, while movie viewing has dropped,but TV show viewing has maintained its status until it starts falling in 2021,but at a slower pace.

In [129]:

```
T3=df[["title","type","country"]]
T3["country"]=T3["country"].str.split(",")
T_3=T3.explode("country")
T_3
```

```
<ipython-input-129-3b9b1c7e0bea>: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-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
  T3["country"]=T3["country"].str.split(",")
```

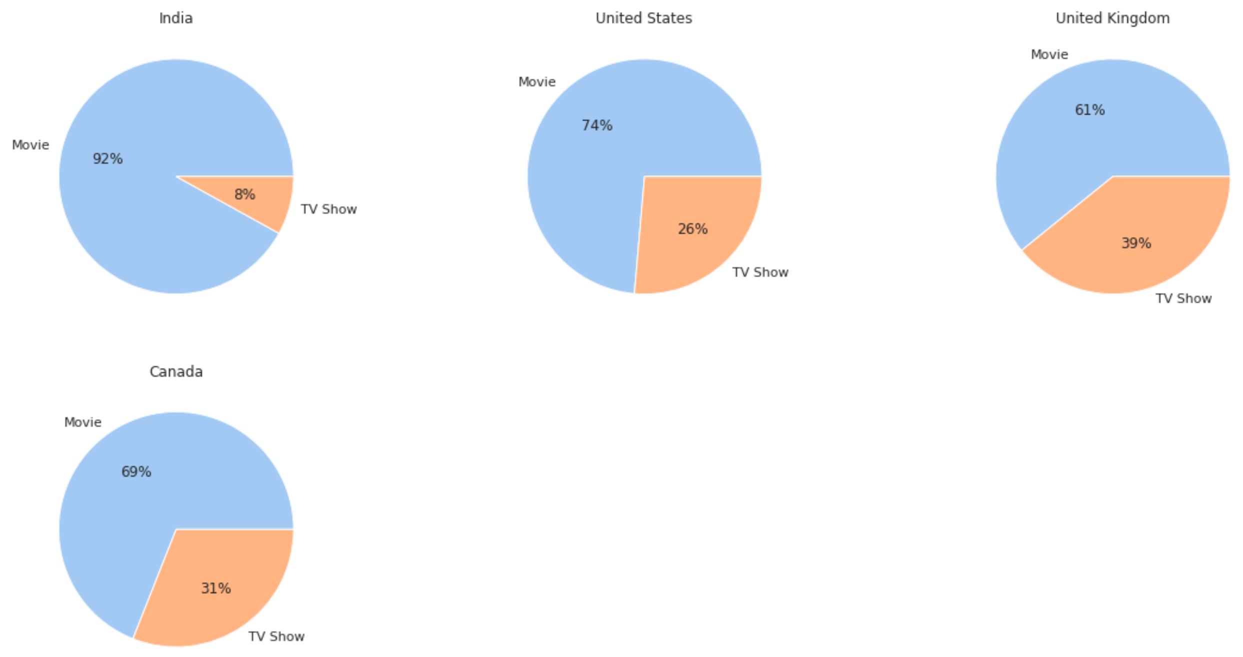
Out[129]:

	title	type	country
0	Dick Johnson Is Dead	Movie	United States
1	Blood & Water	TV Show	South Africa
2	Ganglands	TV Show	NaN
3	Jailbirds New Orleans	TV Show	NaN
4	Kota Factory	TV Show	India
...	...	...	...
8802	Zodiac	Movie	United States
8803	Zombie Dumb	TV Show	NaN
8804	Zombieland	Movie	United States
8805	Zoom	Movie	United States
8806	Zubaan	Movie	India

10850 rows × 3 columns

In [130]:

```
plt.figure(figsize=(20,10))
plt.subplot(2, 3, 1)
plt.title('India', fontsize=12)
df_1=T_3.loc[T_3["country"]=="India"]
data = df_1["type"].value_counts()
labels = ['Movie',"TV Show"]
colors = sns.color_palette('pastel')[0:7]
plt.pie(data,  labels = labels, colors = colors, autopct='%.0f%%')
plt.subplot(2, 3, 2)
plt.title('United States', fontsize=12)
df_2=T_3.loc[T_3["country"]=="United States"]
data = df_2["type"].value_counts()
labels = ['Movie',"TV Show"]
colors = sns.color_palette('pastel')[0:7]
plt.pie(data,  labels = labels, colors = colors, autopct='%.0f%%')
plt.subplot(2, 3, 3)
plt.title('United Kingdom', fontsize=12)
df_3=T_3.loc[T_3["country"]=="United Kingdom"]
data = df_3["type"].value_counts()
labels = ['Movie',"TV Show"]
colors = sns.color_palette('pastel')[0:7]
plt.pie(data,  labels = labels, colors = colors, autopct='%.0f%%')
plt.subplot(2, 3, 4)
plt.title('Canada', fontsize=12)
df_4=T_3.loc[T_3["country"]=="Canada"]
data = df_4["type"].value_counts()
labels = ['Movie',"TV Show"]
colors = sns.color_palette('pastel')[0:7]
plt.pie(data,  labels = labels, colors = colors, autopct='%.0f%%')
plt.show()
```



Comment:

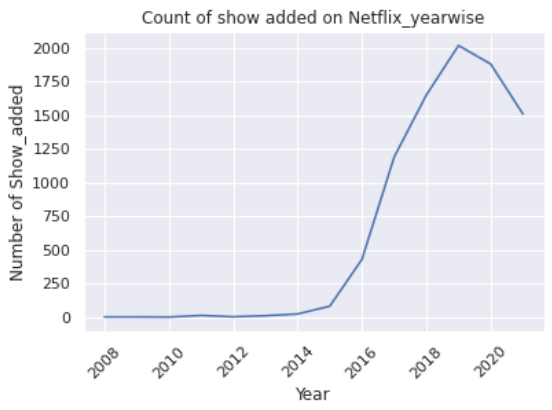
India's massive production is concentrated in films, the highest proportion among the countries analyzed. This can be explained by its large film industry (Bollywood).

In [130]:

```
 
```

In [131]:

```
df_1=df.groupby(df["year"])[["show_id"]].count().sort_values(by=["show_id"],ascending=False).reset_index()
plt.xticks(rotation=45)
plt.xlabel("Year")
plt.ylabel("Number of Show_added")
plt.title("Count of show added on Netflix_yearwise")
ax=sns.lineplot(data=df_1, x=df_1["year"], y=df_1["show_id"])
```

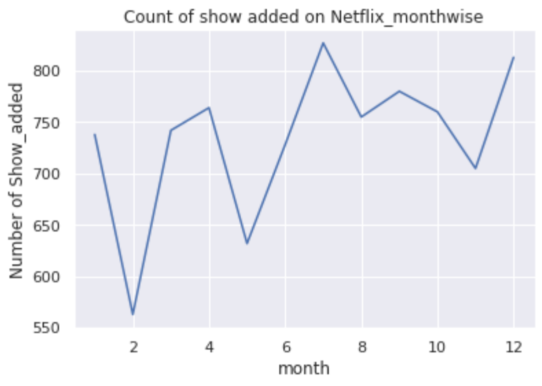


comment

All the months with the highest added number of TV show and Movie are in the period from 2018 to 2021 with reference to date_added column in data set .

In [132]:

```
df_1=df.groupby(df["month"])[["show_id"]].count().sort_values(by=["show_id"],ascending=False).reset_index()
plt.xlabel("month")
plt.ylabel("Number of Show_added")
plt.title("Count of show added on Netflix_monthwise")
ax=sns.lineplot(data=df_1, x=df_1["month"], y=df_1["show_id"])
```



comment

As a general observation December and July are the months with the highest number of TV Show and Movie added and February is the month with the worst when least number of TV Show and Movie added on Netflix. The peak of number of TV Show and Movie attended in months of January(2020), July(2021) and November(2019) may be due to the highest presence of audience in the evaluated period.

In [133]:

```
T5=df[["type","title","country","duration"]]
T5["country"]=T5["country"].str.split(",")
T_5=T5.explode("country")
T_5
```

```
<ipython-input-133-fc8d1ffdd28b>: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-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
  T5["country"]=T5["country"].str.split(",")
```

Out[133]:

	type	title	country	duration
0	Movie	Dick Johnson Is Dead	United States	90
1	TV Show	Blood & Water	South Africa	2
2	TV Show	Ganglands	NaN	1
3	TV Show	Jailbirds New Orleans	NaN	1
4	TV Show	Kota Factory	India	2
...	...	...	...	...
8802	Movie	Zodiac	United States	158
8803	TV Show	Zombie Dumb	NaN	2
8804	Movie	Zombieland	United States	88
8805	Movie	Zoom	United States	88
8806	Movie	Zubaan	India	111

10850 rows × 4 columns

In [134]:

```
top5_genre = T_4["listed_in"].value_counts().index[:5]
top5_title = T_5['title'].value_counts().index[:5]
top5_country = T_5['country'].value_counts().index[:5]
top5_type = T_5['type'].value_counts().index[:2]
top5_data = T_5.loc[(T_5['country'].isin(top5_country)) & (T_5['type'].isin(top5_type)) & (T_5['type'].isin(top5_type))]
top5_data
```

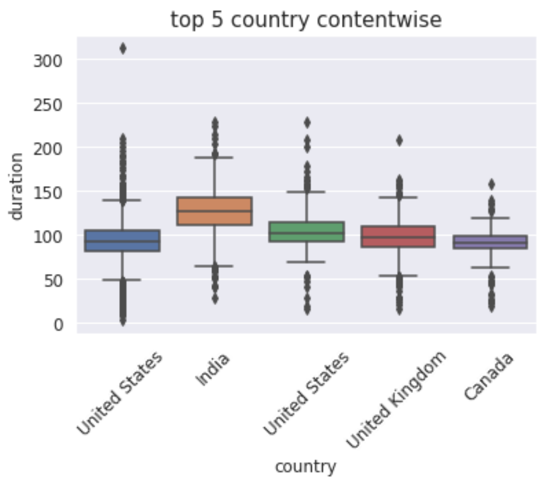
Out[134]:

	type	title	country	duration
0	Movie	Dick Johnson Is Dead	United States	90
4	TV Show	Kota Factory	India	2
7	Movie	Sankofa	United States	125
8	TV Show	The Great British Baking Show	United Kingdom	9
9	Movie	The Starling	United States	104
...	...	...	...	...
8799	Movie	Zenda	India	120
8802	Movie	Zodiac	United States	158
8804	Movie	Zombieland	United States	88
8805	Movie	Zoom	United States	88
8806	Movie	Zubaan	India	111

5597 rows × 4 columns

In [135]:

```
sns.boxplot(x='country', y='duration', data=top5_data[top5_data["type"]=="Movie"])
plt.xticks(rotation=45,fontsize=12)
plt.yticks(fontsize=12)
plt.title('top 5 country contentwise', fontsize=15)
plt.show()
```

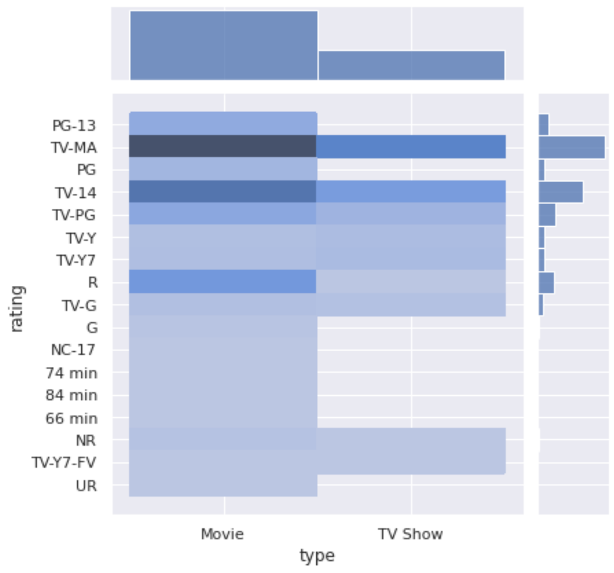


Comment

it signifies that indian movies duration is larger than the other foreign country movies duration

In [136]:

```
sns.jointplot(data=df, x="type", y="rating",kind="hist")
plt.show()
```

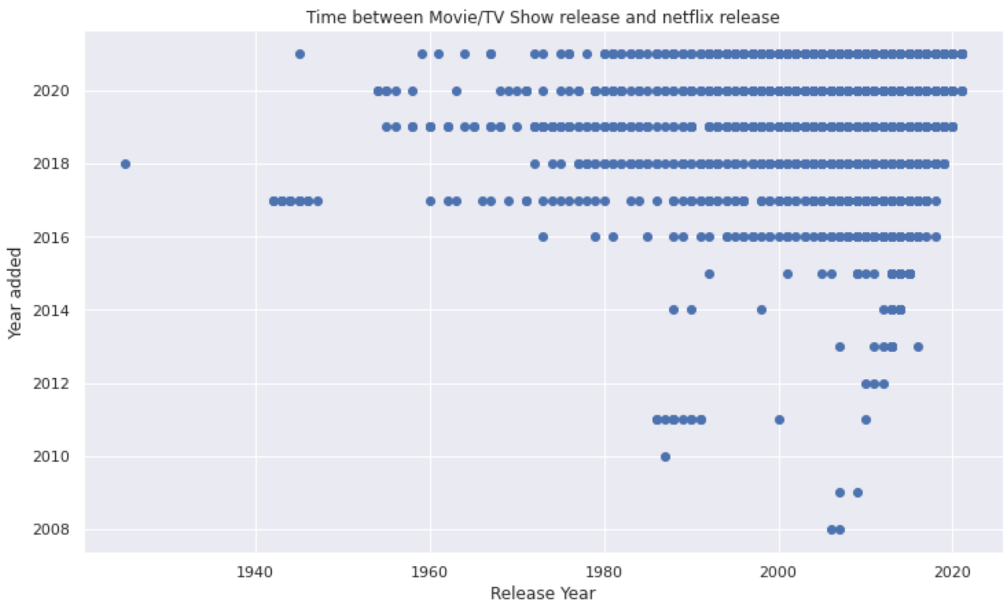


In [137]:

```
x=df["release_year"]
y=df["year"]
plt.figure(figsize=(12,7))
plt.scatter(x,y)
plt.title("Time between Movie/TV Show release and netflix release")
plt.xlabel("Release Year")
plt.ylabel("Year added")
```

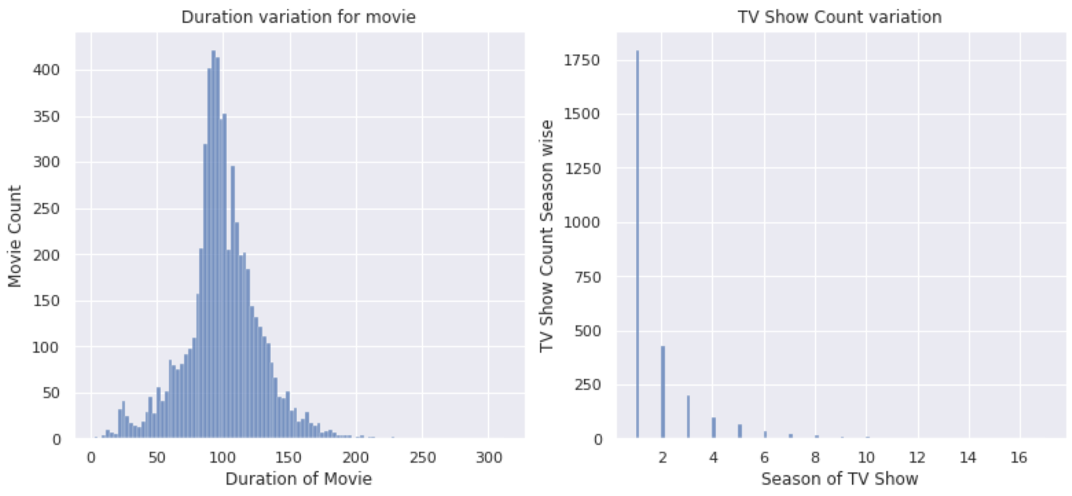
Out[137]:

```
Text(0, 0.5, 'Year added')
```



In [138]:

```
plt.figure(figsize=(20,12))
plt.subplot(2, 3, 1)
df_mov=df[df["type"]=="Movie"]
plt.xlabel("Duration of Movie")
plt.ylabel("Movie Count")
plt.title("Duration variation for movie")
ax=sns.histplot(data=df_mov,x="duration")
plt.subplot(2, 3, 2)
df_TV=df[df["type"]=="TV Show"]
plt.xlabel("Season of TV Show")
plt.ylabel("TV Show Count Season wise")
plt.title("TV Show Count variation")
ax=sns.histplot(data=df_TV,x="duration",)
```



Comment:

The average length of movies varies from 80 to 120 min.

Most TV shows watched have only one season.

5. Missing Value & Outlier check (Treatment optional)

In [139]:

```
# Filling data of cast with director columns
T_5=pd.merge(T_1,T_2, on="title",how="inner")
T_5.isna().sum()
```

Out[139]:

```
title           0
director    19013
type_x          0
cast          960
type_y          0
dtype: int64
```

In [140]:

```
df1=T_5.groupby(['director'])['cast'].agg(pd.Series.mode).to_frame().reset_index().rename(columns={'cast':'actor_mod'})
df2=T_5.merge(df1,on='director',how='left')
df2=df2.fillna({'cast':df2.actor_mod}).drop('actor_mod',axis=1)
df2
```

Out[140]:

	title	director	type_x	cast	type_y
0	Dick Johnson Is Dead	Kirsten Johnson	Movie	[]	Movie
1	Blood & Water	NaN	TV Show	Ama Qamata	TV Show
2	Blood & Water	NaN	TV Show	Khosi Ngema	TV Show
3	Blood & Water	NaN	TV Show	Gail Mabalane	TV Show
4	Blood & Water	NaN	TV Show	Thabang Molaba	TV Show
...	...	...	...	...	...
70807	Zubaan	Mozez Singh	Movie	Manish Chaudhary	Movie
70808	Zubaan	Mozez Singh	Movie	Meghna Malik	Movie
70809	Zubaan	Mozez Singh	Movie	Malkeet Rauni	Movie
70810	Zubaan	Mozez Singh	Movie	Anita Shabdish	Movie
70811	Zubaan	Mozez Singh	Movie	Chittaranjan Tripathy	Movie

70812 rows × 5 columns

In [141]:

```
df2.isna().sum()
```

Out[141]:

```
title           0
director    19013
type_x          0
cast          352
type_y          0
dtype: int64
```

Comment

missing values in column cast is reduces

In [142]:

```
#filling the country value with rating
T3=df[["type","country","rating"]]
T3["country"]=T3["country"].str.split(",")
T_3=T3.explode("country")
T_3.isna().sum()
```

```
<ipython-input-142-3b1f97578394>:3: 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 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
  T3["country"]=T3["country"].str.split(",")
```

Out[142]:

```
type          0
country     831
rating        0
dtype: int64
```

In [143]:

```
df1=T_3.groupby(['rating',"type"])['country'].agg(pd.Series.mode).to_frame().reset_index().rename(columns={'country':'country_mod
df2=T_3.merge(df1,on=['rating',"type"],how='left')
df2=df2.fillna({'country':df2.country_mod}).drop('country_mod',axis=1)
df2
```

Out[143]:

	type	country	rating
0	Movie	United States	PG-13
1	TV Show	South Africa	TV-MA
2	TV Show	United States	TV-MA
3	TV Show	United States	TV-MA
4	TV Show	India	TV-MA
...	...	...	...
10845	Movie	United States	R
10846	TV Show	United States	TV-Y7
10847	Movie	United States	R
10848	Movie	United States	PG
10849	Movie	India	TV-14

10850 rows × 3 columns

In [144]:

```
df2.isna().sum()
```

Out[144]:

```
type       0
country    0
rating     0
dtype: int64
```

7: Business Insights:

1. In the Northern hemisphere, the schools and universities observe summer holidays between June and July while in Southern hemisphere, it is observed from December to January. On Netflix, the months with the highest added content peaks are July and December. It signifies that the best time to release new content on Netflix is during these month.
2. Our data set shows that although the OTT platform like Netflix have been around before 2008, a high frequency of content additions has been observed since 2016.
3. New content (Movie category) showed strong growth from 2016 to 2018. However, there was significant reduction in releases in 2021, possibly caused by the Covid-19 pandemic. This reduction also occurred in content (TV Show category), but the rate of reduction was much less. We can say that audience's presence changed to TV show.
4. On Netflix, the significant contributer in content-wise countries are like USA and India. Where as india's contribution in TV shows is only 8 % and USA having contribution approximate 26 %. So we can say that as audience perefence are changes since 2020 onwards,netflix can more focused towards TV shows in india.
5. As shown in the analysed dataset, most movies are between one and two hours long and many TV shows have only one season. This is due to a large production of shows with only one season, cancellations of new seasons and other factors.
6. It has been observed that the 'Mature Audience only' content is the rating with highest shows count in both TV Shows and Movies.
7. USA & Canada have similarity in Genre('Dramas', 'Comedies') popularity in both TV Shows and Movies. Similarly India & France have similarity in Genre('International Movies', 'Dramas') popularity in Movies.
8. Rajiv Chilaka director producing more movies in 'Children & Family Movies' Genre with same actors.

8.Recommendations:

1. Since TV Shows are in trending, So Netflix should add more content in popular Genre category country-wise.
2. As Netflix has added less shows in month February, that could be best time to release more new shows to have higher probability in more viewers.
3. TV Shows with less seasons and movies with 80-120 minutes duration on 'Dramas' & 'Comedies' Genre is preferable.
4. Since, USA and Canada have similar popularity in Genre, but have less movies in Canada. We can produce more movies to increase business in Canada. Similarly, India and France has common interest. We can provide movies to attract both Audience.
5. As the Rajiv Chilaka popular director, netflix can produce more movies with Rajiv Chilaka in category of 'Children & Family Movies'.