In [423]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [424]:
```
file_path = r'C:\Users\INACHAVH\PycharmProjects\CodingElements\practice\Data\Netf
df = pd.read_csv(file_path)
```

In [425]:
```
df.shape
```

Out[425]:
```
(8807, 12)
```

In [426]:
```
# Calculating missing data
for i in df.columns:
    null_rate = df[i].isna().sum()/len(df) * 100
    if null_rate > 0 :
        print("{} null rate: {}%".format(i,round(null_rate,2)))
```

```
director null rate: 29.91%
cast null rate: 9.37%
country null rate: 9.44%
date_added null rate: 0.11%
rating null rate: 0.05%
duration null rate: 0.03%
```

In [427]:
```
df.count()
```

Out[427]:
```
show_id         8807
type            8807
title           8807
director        6173
cast            7982
country         7976
date_added      8797
release_year    8807
rating          8803
duration        8804
listed_in       8807
description     8807
dtype: int64
```

Business Problem : Analyze the data and generate insights that could help Netflix in deciding which type of shows/movies to produce and how they can grow the business in different countries

In [428]:
```
##understand Column type and Not-null
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 [429]:
```
# df['duration'] = str(df['duration'])
```

In [430]:
```
df.describe(include = object)
```

Out[430]:

	show_id	type	title	director	cast	country	date_added	rating	duration	
count	8807	8807	8807	6173	7982	7976	8797	8803	8804	
unique	8807	2	8807	4528	7692	748	1767	17	220	
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	TV-MA	1 Season	Int
freq	1	6131	1	19	19	2818	109	3207	1793	

In [431]:
```
'''
Problem Solving Approach :-
    1. Provide what kind of recommendation can be provided using Actor.
        -  No. of movies that actor has worked in.
        -  Most Frequent actor/ Director on the platform.
        -  For a parcticular genre, find most popular actor/director.
        -  Popular actor in a country.
        -  Most frequent actor with respect to Rating.
        -  Favorite actor for a director.

    2. Provide what kind of recommendation can be provided using Movie.

    3. Provide what kind of recommendation can be provided using Director.

    4. Provide what kind of recommendation can be provided using Country.

'''
```

Out[431]:
```
'\nProblem Solving Approach :-\n    1. Provide what kind of recommendation can
be provided using Actor.\n        -  No. of movies that actor has worked in.\n
-  Most Frequent actor/ Director on the platform.\n        -  For a parcticular
genre, find most popular actor/director.\n        -  Popular actor in a countr
y.\n        -  Most frequent actor with respect to Rating.\n        -  Favorite
actor for a director.\n        \n    2. Provide what kind of recommendation can
be provided using Movie.\n        \n    3. Provide what kind of recommendation
can be provided using Director.\n    \n    4. Provide what kind of recommendati
on can be provided using Country.\n    \n'
```

In [432]:
```
'''
 Pre-processing the dataset.
    1. Un-nesting of rows. -- Done
    2. Null values
    3. Duration in string -> integers  --Done
    4. Date_added string -> Date time  -- Done
    5.
'''
```

Out[432]:
```
'\n Pre-processing the dataset.\n    1. Un-nesting of rows. -- Done\n    2. Nul
l values\n    3. Duration in string -> integers  --Done\n    4. Date_added stri
ng -> Date time  -- Done\n    5. \n'
```

In [433]:
```
# ##Count number of seasons
# df['season_count'] = df.apply(lambda x : x['duration'].split(" ")[0] if "Seasor
# df['duration'] = df.apply(lambda x : x['duration'].split(" ")[0] if "Seasons" r
```

In [434]:
```
df['duration']
```

Out[434]:
```
0          90 min
1       2 Seasons
2        1 Season
3        1 Season
4       2 Seasons
           ...
8802      158 min
8803    2 Seasons
8804       88 min
8805       88 min
8806      111 min
Name: duration, Length: 8807, dtype: object
```

In [ ]:
```

```

In [435]:
```
## split cast into list and create separate row for each actor
df['cast'] = df['cast'].apply(lambda x : str(x).split(', ')).to_list()
df = df.explode('cast').reset_index(drop=True)
```

In [436]:
```
## split director into list and create separate row for each director
df['director'] = df['director'].apply(lambda x : str(x).split(', ')).to_list()
df = df.explode('director').reset_index(drop=True)
```

In [437]:
```
## split lissted_in into list and create separate row for each listed_in
df['listed_in'] = df['listed_in'].apply(lambda x : str(x).split(', ')).to_list()
df = df.explode('listed_in').reset_index(drop=True)
```

In [438]:
```
#### split country into list and create separate row for each country
df['country'] = df['country'].apply(lambda x : str(x).split(', ')).to_list()
df = df.explode('country').reset_index(drop=True)
```

In [ ]:
```

```

In [439]:
```
df.head()
```

Out[439]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	nan	United States	September 25, 2021	2020	PG-13	90 min
1	s2	TV Show	Blood & Water	nan	Ama Qamata	South Africa	September 24, 2021	2021	TV-MA	2 Seasons
2	s2	TV Show	Blood & Water	nan	Ama Qamata	South Africa	September 24, 2021	2021	TV-MA	2 Seasons
3	s2	TV Show	Blood & Water	nan	Ama Qamata	South Africa	September 24, 2021	2021	TV-MA	2 Seasons
4	s2	TV Show	Blood & Water	nan	Khosi Ngema	South Africa	September 24, 2021	2021	TV-MA	2 Seasons

In [440]:
```
## convert date(string) to datetime
df['date_added'] = pd.to_datetime(df['date_added'])
```

In [441]:
```
df['cast'].astype(str)
```

Out[441]:
```
0                          nan
1                  Ama Qamata
2                  Ama Qamata
3                  Ama Qamata
4                 Khosi Ngema
                     ...
201986          Anita Shabdish
201987          Anita Shabdish
201988    Chittaranjan Tripathy
201989    Chittaranjan Tripathy
201990    Chittaranjan Tripathy
Name: cast, Length: 201991, dtype: object
```

In [442]:
```
## check for NaN
df['cast'].value_counts(dropna=False)
```

Out[442]:
```
nan               2146
Liam Neeson        161
Alfred Molina      160
John Krasinski     139
Salma Hayek        130
                  ...
Dario Yazbek         1
Corinne Foxx         1
Jacob Craner         1
Laila Berzins        1
Richard Ryan         1
Name: cast, Length: 36440, dtype: int64
```

In [443]:
```
df['cast'].isna()
```

Out[443]:
```
0         False
1         False
2         False
3         False
4         False
          ...
201986    False
201987    False
201988    False
201989    False
201990    False
Name: cast, Length: 201991, dtype: bool
```

In [444]:
```
## check for NaN
df['director'].value_counts(dropna=False)
```

Out[444]:
```
nan                    50643
Martin Scorsese          419
Youssef Chahine          409
Cathy Garcia-Molina      356
Steven Spielberg         355
                       ...
Richard Maurice            1
Richard E. Norman          1
Spencer Williams           1
Oscar Micheaux             1
Kirsten Johnson            1
Name: director, Length: 4994, dtype: int64
```

In [445]:
```
## check for NaN
df['duration'].value_counts(dropna=False)
```

Out[445]:
```
1 Season     35035
2 Seasons     9559
3 Seasons     5084
94 min        4343
106 min       4040
              ...
5 min            3
NaN              3
8 min            2
11 min           2
9 min            2
Name: duration, Length: 221, dtype: int64
```

In [446]:
```
## Add year_added column
df["year_added"] = df['date_added'].dt.year
## add month_added column
df["month_added"] = df['date_added'].dt.month
```

In [447]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 201991 entries, 0 to 201990
Data columns (total 14 columns):
 #   Column        Non-Null Count   Dtype         
---  ------        --------------   -----         
 0   show_id       201991 non-null  object        
 1   type          201991 non-null  object        
 2   title         201991 non-null  object        
 3   director      201991 non-null  object        
 4   cast          201991 non-null  object        
 5   country       201991 non-null  object        
 6   date_added    201833 non-null  datetime64[ns]
 7   release_year  201991 non-null  int64         
 8   rating        201924 non-null  object        
 9   duration      201988 non-null  object        
 10  listed_in     201991 non-null  object        
 11  description   201991 non-null  object        
 12  year_added    201833 non-null  float64       
 13  month_added   201833 non-null  float64       
dtypes: datetime64[ns](1), float64(2), int64(1), object(10)
memory usage: 21.6+ MB
```

In [448]:
```
df.head()
```

Out[448]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	nan	United States	2021-09-25	2020	PG-13	90 min
1	s2	TV Show	Blood & Water	nan	Ama Qamata	South Africa	2021-09-24	2021	TV-MA	2 Seasons
2	s2	TV Show	Blood & Water	nan	Ama Qamata	South Africa	2021-09-24	2021	TV-MA	2 Seasons
3	s2	TV Show	Blood & Water	nan	Ama Qamata	South Africa	2021-09-24	2021	TV-MA	2 Seasons
4	s2	TV Show	Blood & Water	nan	Khosi Ngema	South Africa	2021-09-24	2021	TV-MA	2 Seasons

Now we select one entity and try to find recommendation.

Using Actor

1. Provide what kind of recommendation can be provided using Actor.
    - No. of movies that actor has worked in.
    - Most Frequent actor/ Director on the platform.
    - For a parcticular genre, find most popular actor/director.
    - Popular actor in a country.
    - Most frequent actor with respect to Rating.
    - Favorite actor for a director.

In [449]:
```
## No. of movies that actor has worked in
df_movie = df[df['type'] == 'Movie']
len(df_movie)  ## No. of records of type 'Movie' -- 145843
df_movie.groupby('cast')['cast'].count().sort_values(ascending=False)

# ##Observation :-
#     1. Liam Nesson worked in 161 movies.
#     2. John Krasinski woked in 157 movies.
#     3. Tiffany Kathryn,Donny Boaz,Tiffany Shepis,Tiffany Snow  acted in 1 movie
```

Out[449]:
```
cast
nan                1328
Liam Neeson         161
Alfred Molina       157
John Krasinski      138
Salma Hayek         130
                   ...
Tiffany Kathryn       1
Donny Boaz            1
Tiffany Shepis        1
Tiffany Snow          1
 Jr.                  1
Name: cast, Length: 25952, dtype: int64
```

In [450]:
```
df_movie.head()
```

Out[450]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	du
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	nan	United States	2021-09-25	2020	PG-13	9
159	s7	Movie	My Little Pony: A New Generation	Robert Cullen	Vanessa Hudgens	nan	2021-09-24	2021	PG	9
160	s7	Movie	My Little Pony: A New Generation	José Luis Ucha	Vanessa Hudgens	nan	2021-09-24	2021	PG	9
161	s7	Movie	My Little Pony: A New Generation	Robert Cullen	Kimiko Glenn	nan	2021-09-24	2021	PG	9
162	s7	Movie	My Little Pony: A New Generation	José Luis Ucha	Kimiko Glenn	nan	2021-09-24	2021	PG	9

In [451]:
```
## Most popular director in 'movies'
df_movie.groupby('director')['director'].count().sort_values(ascending=False)

##observations :
# Most popular director in 'movie' is 'Martin Scorsese'
# 2nd most Famous director in 'movie' is 'Youssef Chahine'
```

Out[451]:
```
director
nan                     1285
Martin Scorsese          419
Youssef Chahine          409
Cathy Garcia-Molina      356
Steven Spielberg         355
                        ...
Michelle Esrick            1
J. Michael Long            1
C.J. Wallis                1
Caio Cobra                 1
Jon Rudberg                1
Name: director, Length: 4778, dtype: int64
```

In [452]:
```
## get max count of actor
def get_max_actor_count(a):
    return a['cast'].value_counts().idxmax()
    #return a['cast'].value_counts().nunique()
## for a particular 'genre' find the most popular actor in a "Movie"
df_movie.groupby('listed_in').apply(get_max_actor_count)
```

Out[452]:
```
listed_in
Action & Adventure              Luci Christian
Anime Features                     John Swasey
Children & Family Movies         John Krasinski
Classic Movies                 Burgess Meredith
Comedies                            Tara Strong
Cult Movies                         Keith David
Documentaries                               nan
Dramas                              Liam Neeson
Faith & Spirituality           Abdelilah Wahbi
Horror Movies                     Lorenza Izzo
Independent Movies                James Franco
International Movies                        nan
LGBTQ Movies                                nan
Movies                        David Attenborough
Music & Musicals                            nan
Romantic Movies                  Michelle Yeoh
Sci-Fi & Fantasy                 Luci Christian
Sports Movies                               nan
Stand-Up Comedy                     Kevin Hart
Thrillers                          Nicolas Cage
dtype: object
```

In [485]:
```
#pd.pivot_table(data=df_movie, index = 'cast',values = 'show_id', aggfunc = (lamb
# df['release_year'].value_counts(ascending=False,dropna=False)
```

In [454]:
```
## get max count of drector
def get_max_director_count(a):
    return a['director'].value_counts().idxmax()

## for a particular 'genre' find the most popular actor in a "Movie"
df_movie.groupby('listed_in').apply(get_max_director_count)
```

Out[454]:
```
listed_in
Action & Adventure                  Martin Campbell
Anime Features                    Toshiya Shinohara
Children & Family Movies                        nan
Classic Movies                    Youssef Chahine
Comedies                                        nan
Cult Movies                          Edgar Wright
Documentaries                                   nan
Dramas                            Martin Scorsese
Faith & Spirituality                  David Batty
Horror Movies                           James Wan
Independent Movies                 Lars von Trier
International Movies                            nan
LGBTQ Movies                            Jun Lana
Movies                                          nan
Music & Musicals                                nan
Romantic Movies                 Cathy Garcia-Molina
Sci-Fi & Fantasy                      Peter Jackson
Sports Movies                 Juan José Campanella
Stand-Up Comedy                                 nan
Thrillers                  Fernando González Molina
dtype: object
```

In [455]:
```
## Most Popular 'Movie' Actor in a country
def popular_country(c):
    return c['cast'].value_counts().idxmax()

df_movie.groupby(['country']).apply(popular_country)

##Observation
# Below mentioned are the coutry wise favorite actor in a Movie
```

Out[455]:
```
country
                Khaled Abol El Naga
Afghanistan           Sohrab Nazari
Albania              Marco Giallini
Algeria         Khaled Abol El Naga
Angola              Paulo Americano
                               ...
Venezuela                      nan
Vietnam                 Mai Cat Vi
West Germany                   nan
Zimbabwe                       nan
nan                            nan
Length: 123, dtype: object
```

In [456]:
```
## Most Popular 'Movie' Director in a country

def popular_country(c):
    return c['director'].value_counts().idxmax()

df_movie.groupby(['country']).apply(popular_country)

##Observation
# Below mentioned are the coutry wise favorite director in a Movie
```

Out[456]:
```
country
                          Najwa Najjar
Afghanistan          Pieter-Jan De Pue
Albania               Antonio Morabito
Algeria                Youssef Chahine
Angola                    Chris Roland
                                 ...
Venezuela           Sebastián Schindel
Vietnam                      Victor Vu
West Germany          Jacek Koprowicz
Zimbabwe              Tomas Brickhill
nan                               nan
Length: 123, dtype: object
```

In [457]:
```
##Most frequent actor with respect to rating in a "Movie".
def count_actor(c):
    return c['cast'].value_counts().idxmax()

df_movie.groupby(['rating']).apply(count_actor)
```

Out[457]:
```
rating
66 min           Louis C.K.
74 min           Louis C.K.
84 min           Louis C.K.
G             Pierce Brosnan
NC-17        Catherine Salée
NR                       nan
PG             Alfred Molina
PG-13            John Swasey
R                Ben Whishaw
TV-14                    nan
TV-G                     nan
TV-MA                    nan
TV-PG                    nan
TV-Y           Andrea Libman
TV-Y7          Julie Tejwani
TV-Y7-FV                 nan
UR                 Ira Max
dtype: object
```

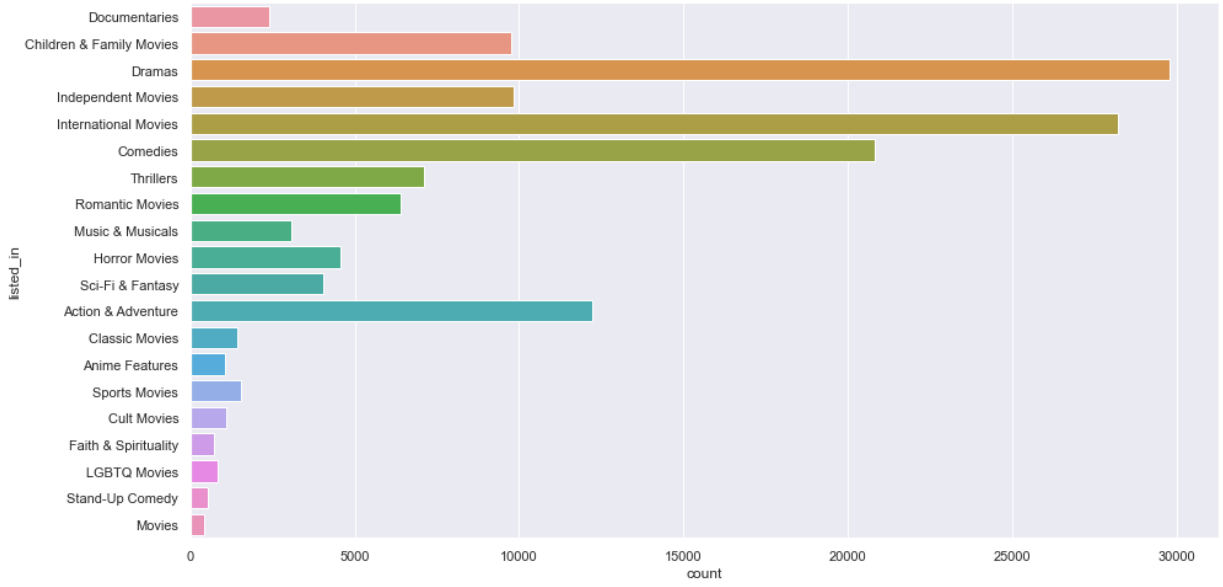
In [458]:
```
## Most frequent director with respect to rating in a "Movie".
def count_actor(c):
    return c['director'].value_counts().idxmax()

df_movie.groupby(['rating']).apply(count_actor)
```

Out[458]:
```
rating
66 min               Louis C.K.
74 min               Louis C.K.
84 min               Louis C.K.
G                   Robert Vince
NC-17       Abdellatif Kechiche
NR                Lars von Trier
PG                  Raja Gosnell
PG-13             Peter Jackson
R                    Tom Hooper
TV-14               Umesh Mehra
TV-G                        nan
TV-MA                       nan
TV-PG                       nan
TV-Y                        nan
TV-Y7                       nan
TV-Y7-FV             Mario Cambi
UR                Sylvie Verheyde
dtype: object
```

In [459]:
```
## get max count of drector
def get_max_director_count(a):
    return a['director'].value_counts().idxmax()

## for a particular 'genre' find the most popular actor in a "Movie"
df_movie.groupby('listed_in').apply(get_max_director_count)
```

Out[459]:
```
listed_in
Action & Adventure                 Martin Campbell
Anime Features                   Toshiya Shinohara
Children & Family Movies                       nan
Classic Movies                   Youssef Chahine
Comedies                                       nan
Cult Movies                        Edgar Wright
Documentaries                                  nan
Dramas                           Martin Scorsese
Faith & Spirituality                 David Batty
Horror Movies                          James Wan
Independent Movies                Lars von Trier
International Movies                           nan
LGBTQ Movies                          Jun Lana
Movies                                         nan
Music & Musicals                               nan
Romantic Movies                Cathy Garcia-Molina
Sci-Fi & Fantasy                     Peter Jackson
Sports Movies                Juan José Campanella
Stand-Up Comedy                                nan
Thrillers                  Fernando González Molina
dtype: object
```

In [460]:
```
## Different movies based on 'genre'
sns.set(rc = {'figure.figsize':(15,8)})
sns.countplot(y = df_movie['listed_in'],data = df_movie)
plt.show()
#sns.barplot(genre_count.index,genre_count.values)
```



In [461]:
```
## No. of movies that actor has worked in
df_tvshow = df[df['type'] == 'TV Show']

## Most popular Actor in 'tv-shows'
df_tvshow.groupby('cast')['cast'].count().sort_values(ascending=False)

##observations :
# Most popular Actor in 'Tv shows' is 'David Attenborough'
# 2nd most Famous Actor in 'Tv shows' is 'Takahiro Sakurai'
```

Out[461]:
```
cast
nan                   818
David Attenborough     82
Takahiro Sakurai       56
Yuki Kaji              45
Ai Kayano              41
                      ...
Richard E. Grant        1
David Pittu             1
Richard Harrison        1
David Nichtern          1
 Jr.                    1
Name: cast, Length: 14864, dtype: int64
```

In [462]:
```
## Most popular director in 'tv-shows'
df_tvshow.groupby('director')['director'].count().sort_values(ascending=False)

##observations :
# Most popular director in 'Tv shows' is 'Noam Murro'.
# 2nd most Famous director in 'Tv shows' is 'Thomas Astruc'.
```

Out[462]:
```
director
nan                  49358
Noam Murro             189
Thomas Astruc          160
Alan Poul              104
Houda Benyamina        104
                     ...
Garrett Bradley          1
Fernando Moro            1
Oliver Stone             1
Julia Reichert           1
Richard E. Norman        1
Name: director, Length: 300, dtype: int64
```

In [463]:
```
## for a particular 'genre' find the most popular actor in a "tv-show".
## get max count of actor
def get_max_actor_count(a):
    return a['cast'].value_counts().idxmax()

## for a particular 'genre' find the most popular actor in a "Movie"
df_tvshow.groupby('listed_in').apply(get_max_actor_count)
```

Out[463]:
```
listed_in
Anime Series                    Takahiro Sakurai
British TV Shows                             nan
Classic & Cult TV              John Dunsworth
Crime TV Shows                               nan
Docuseries                                   nan
International TV Shows                       nan
Kids' TV                                     nan
Korean TV Shows                        Bae Doona
Reality TV                                   nan
Romantic TV Shows                            nan
Science & Nature TV                          nan
Spanish-Language TV Shows                    nan
Stand-Up Comedy & Talk Shows                 nan
TV Action & Adventure                Lena Headey
TV Comedies                                  nan
TV Dramas                           Joanna Kulig
TV Horror                        Jon Jon Briones
TV Mysteries                                 nan
TV Sci-Fi & Fantasy                  Lena Headey
TV Shows                           Prayaga Martin
TV Thrillers                       Hsia Teng-hung
Teen TV Shows                    Yuichi Nakamura
dtype: object
```

In [464]:
```
## get max count of director
def get_max_director_count(a):
    return a['director'].value_counts().idxmax()

## for a particular 'genre' find the most popular actor in a "tv-show"
df_tvshow.groupby('listed_in').apply(get_max_director_count)
```

Out[464]:
```
listed_in
Anime Series                                nan
British TV Shows                            nan
Classic & Cult TV                           nan
Crime TV Shows                              nan
Docuseries                                  nan
International TV Shows                      nan
Kids' TV                                    nan
Korean TV Shows                             nan
Reality TV                                  nan
Romantic TV Shows                           nan
Science & Nature TV                         nan
Spanish-Language TV Shows                   nan
Stand-Up Comedy & Talk Shows                nan
TV Action & Adventure                       nan
TV Comedies                                 nan
TV Dramas                                   nan
TV Horror                                   nan
TV Mysteries                                nan
TV Sci-Fi & Fantasy                         nan
TV Shows                    Gautham Vasudev Menon
TV Thrillers                                nan
Teen TV Shows                               nan
dtype: object
```

In [465]:
```
## Most Popular 'tv-show' Actor in a country
def popular_country(c):
    return c['cast'].value_counts().idxmax()

df_tvshow.groupby(['country']).apply(popular_country)

##Observation
# Below mentioned are the coutry wise favorite actor in a tv-show
```

Out[465]:
```
country
                          Jung Hae-in
Argentina                 Chino Darín
Australia                         nan
Austria                Robert Finster
Azerbaijan         Aras Bulut İynemli
                              ...
United Kingdom                    nan
United States                     nan
Uruguay                           nan
West Germany           Graham Chapman
nan                               nan
Length: 67, dtype: object
```

In [466]:
```
##Most frequent actor with respect to rating in a "Movie".
def count_actor(c):
    return c['cast'].value_counts().idxmax()

df_tvshow.groupby(['rating']).apply(count_actor)
```

Out[466]:
```
rating
NR                  Tim Pigott-Smith
R            Mads Sjøgård Pettersen
TV-14                             nan
TV-G                              nan
TV-MA                             nan
TV-PG                             nan
TV-Y                              nan
TV-Y7                             nan
TV-Y7-FV                  Al Mukadam
dtype: object
```

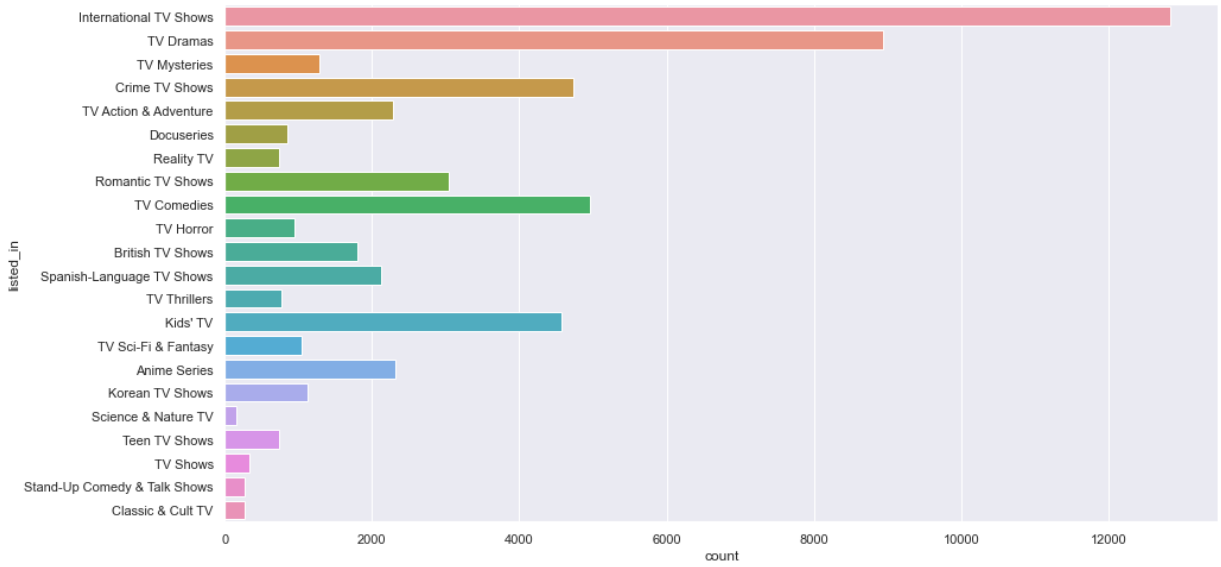
In [467]:
```
##Most frequent actor with respect to rating in a "Movie".
def count_actor(c):
    return c['director'].value_counts().idxmax()

df_tvshow.groupby(['rating']).apply(count_actor)
```

Out[467]:
```
rating
NR          nan
R           nan
TV-14       nan
TV-G        nan
TV-MA       nan
TV-PG       nan
TV-Y        nan
TV-Y7       nan
TV-Y7-FV    nan
dtype: object
```

In [468]:
```
## Most Popular 'tv-show' Director in a country

def popular_country(c):
    return c['director'].value_counts().idxmax()

df_tvshow.groupby(['country']).apply(popular_country)

##Observation
# Below mentioned are the country wise favorite director
```

Out[468]:
```
country
                    nan
Argentina           nan
Australia           nan
Austria             nan
Azerbaijan          nan
                   ...
United Kingdom      nan
United States       nan
Uruguay             nan
West Germany        nan
nan                 nan
Length: 67, dtype: object
```

In [469]:
```
## Different tv show based on 'genre'
sns.set(rc = {'figure.figsize':(15,8)})
sns.countplot(y = df_tvshow['listed_in'],data = df_tvshow)
plt.show()
#sns.barplot(genre_count.index,genre_count.values)
```



In [470]:
```
##favorite actor for a director in "movie"
def director_actor(d):
    return d['cast'].value_counts().idxmax()

df_movie.groupby('director').apply(director_actor)
```

Out[470]:
```
director
A. L. Vijay                           G.V. Prakash Kumar
A. Raajdheep                               Vikram Prabhu
A. Salaam                                  Shashi Kapoor
A.R. Murugadoss                                    Vijay
Aadish Keluskar                          Khushboo Upadhyay
                                         ...
Éric Warin                                  Bronwen Mantel
Ísold Uggadóttir          Kristín Thóra Haraldsdóttir
Óskar Thór Axelsson          Jóhannes Haukur Jóhannesson
Ömer Faruk Sorak                                Cem Yılmaz
Şenol Sönmez                                     Ali Sunal
Length: 4778, dtype: object
```

In [471]:
```
##favorite actor for a director in "tv-show"
def director_actor(d):
    return d['cast'].value_counts().idxmax()

df_tvshow.groupby('director').apply(director_actor)
```

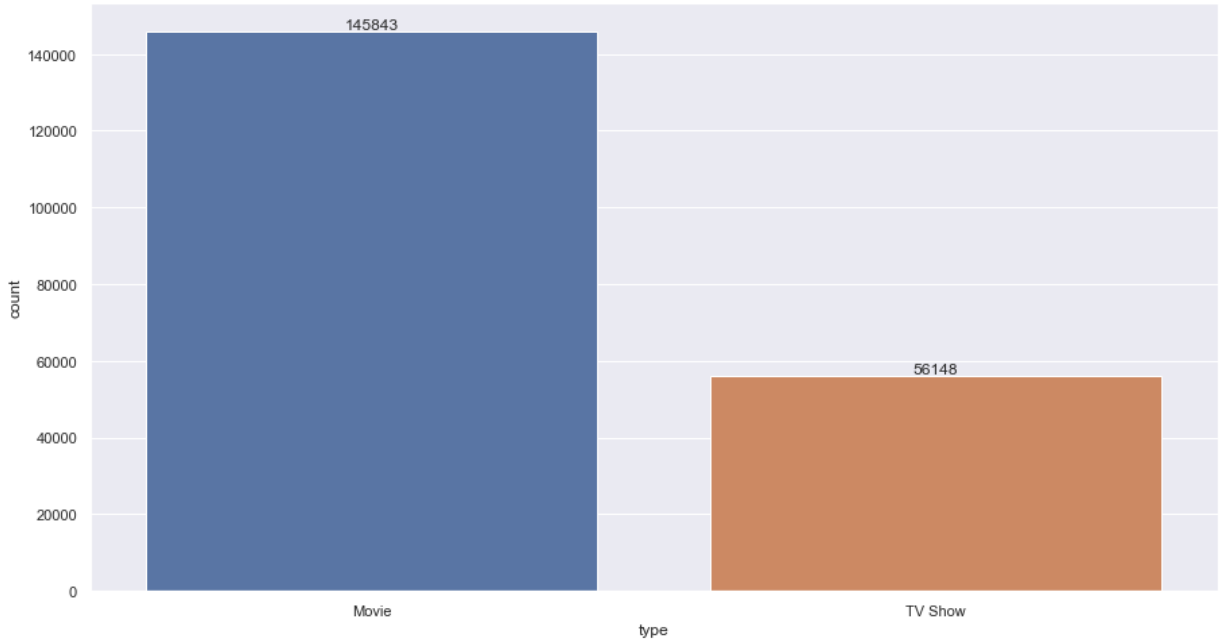
Out[471]:
```
director
Abhishek Chaubey              Manoj Bajpayee
Aco Tenriyagelli              Adinia Wirasti
Adrien Lagier                           Fary
Adrián García Bogliano                   nan
Ah Loong                                 nan
                                     ...
YC Tom Lee                     Wen Chen-ling
Yasuhiro Irie                      Romi Park
Yim Pilsung                 Lee Ji-eun (IU)
Ziad Doueiri                   Eric Cantona
nan                                      nan
Length: 300, dtype: object
```

Visual Analysis

Categorical Variable(s)

In [472]:
```
## Plot no. of movies and tv-shows
ax = sns.countplot(x = 'type',data = df)
ax.bar_label(ax.containers[0])
```

Out[472]:
```
[Text(0, 0, '145843'), Text(0, 0, '56148')]
```

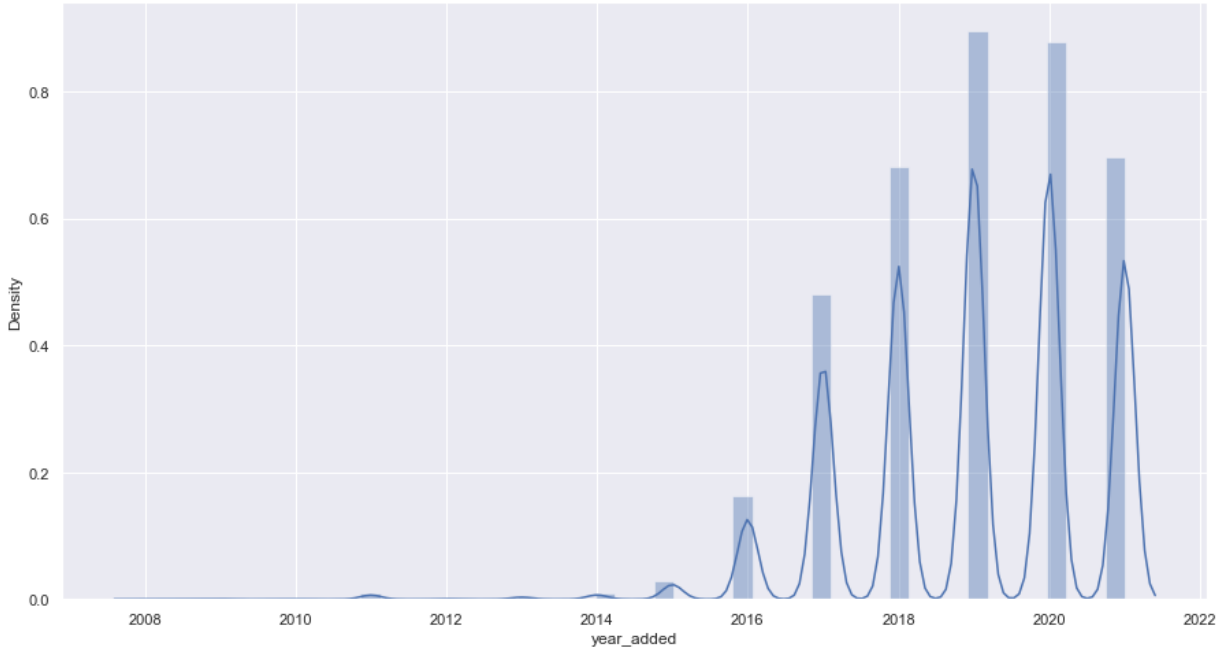


Continuous Variable(s)

In [473]:
```
sns.distplot(df['year_added'])
plt.plot()
```

```
c:\python39\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `di
stplot` is a deprecated function and will be removed in a future version. Pleas
e adapt your code to use either `displot` (a figure-level function with similar
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

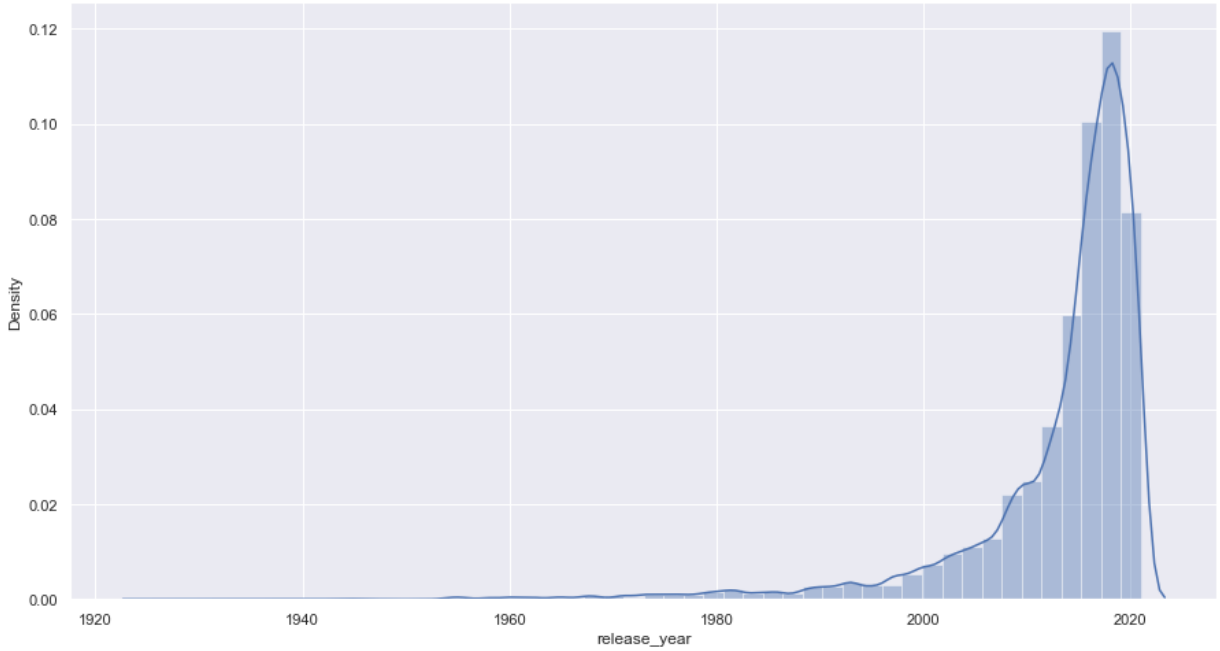
Out[473]: []



In [474]:
```
sns.distplot(df['release_year'])
plt.plot()
```

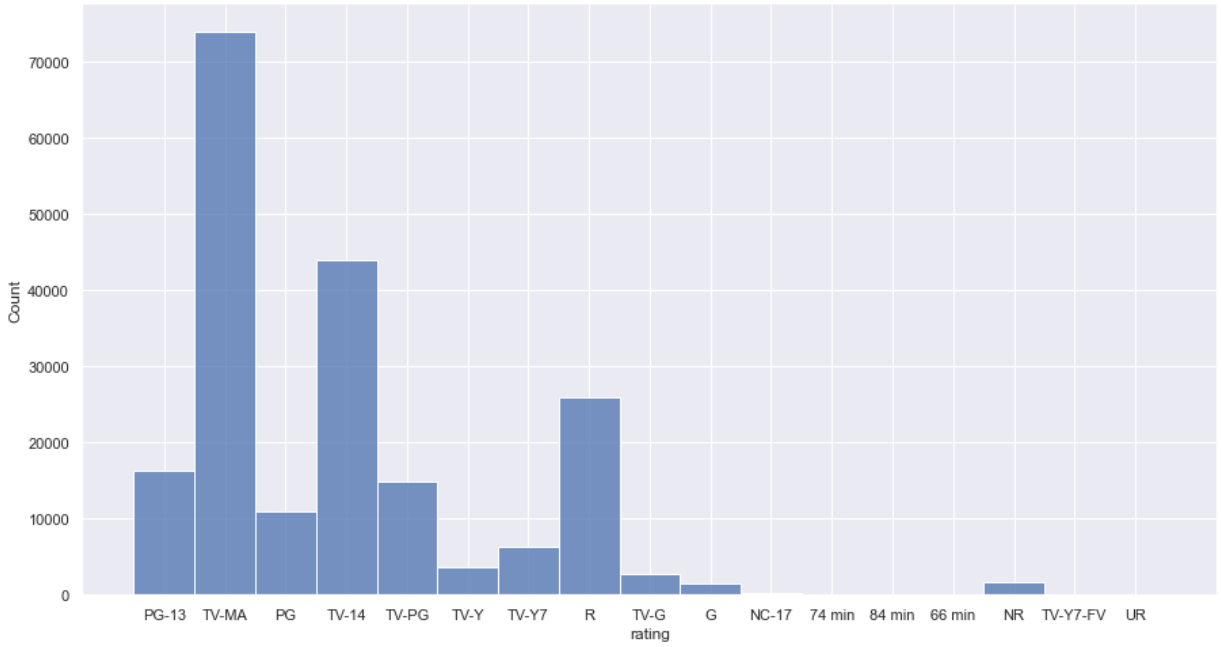
```
c:\python39\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `di
stplot` is a deprecated function and will be removed in a future version. Pleas
e adapt your code to use either `displot` (a figure-level function with similar
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[474]: []

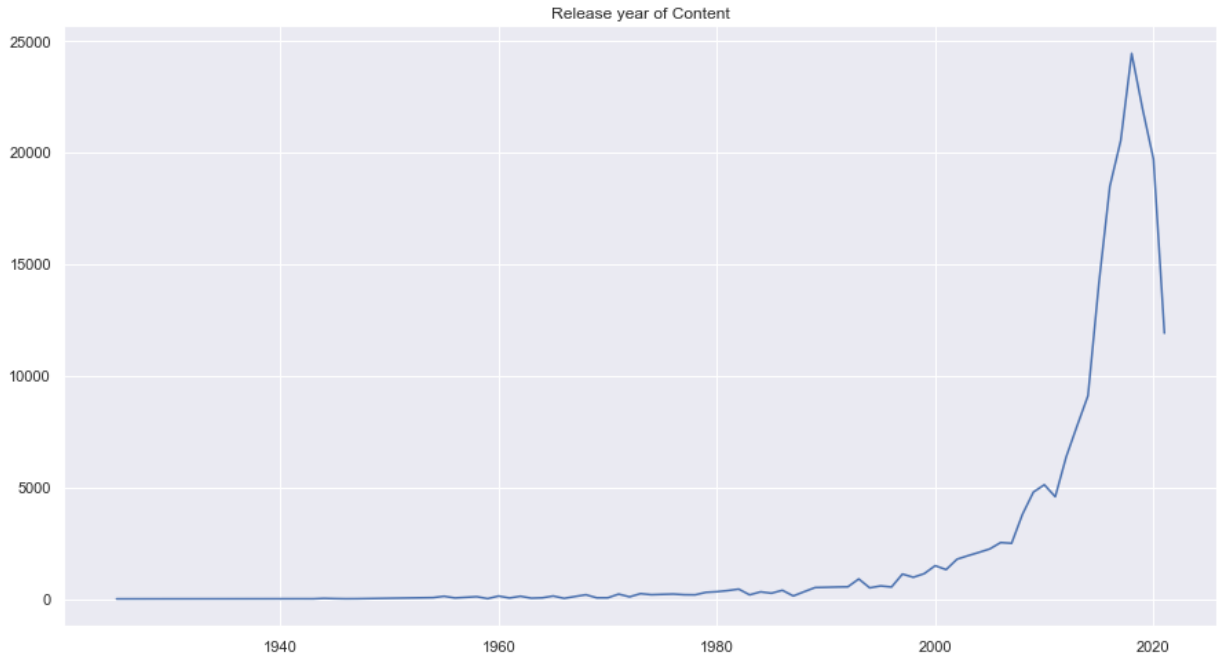


In [475]:
```
sns.histplot(x = 'rating',data =df)
plt.plot()
```

Out[475]:
```
[]
```

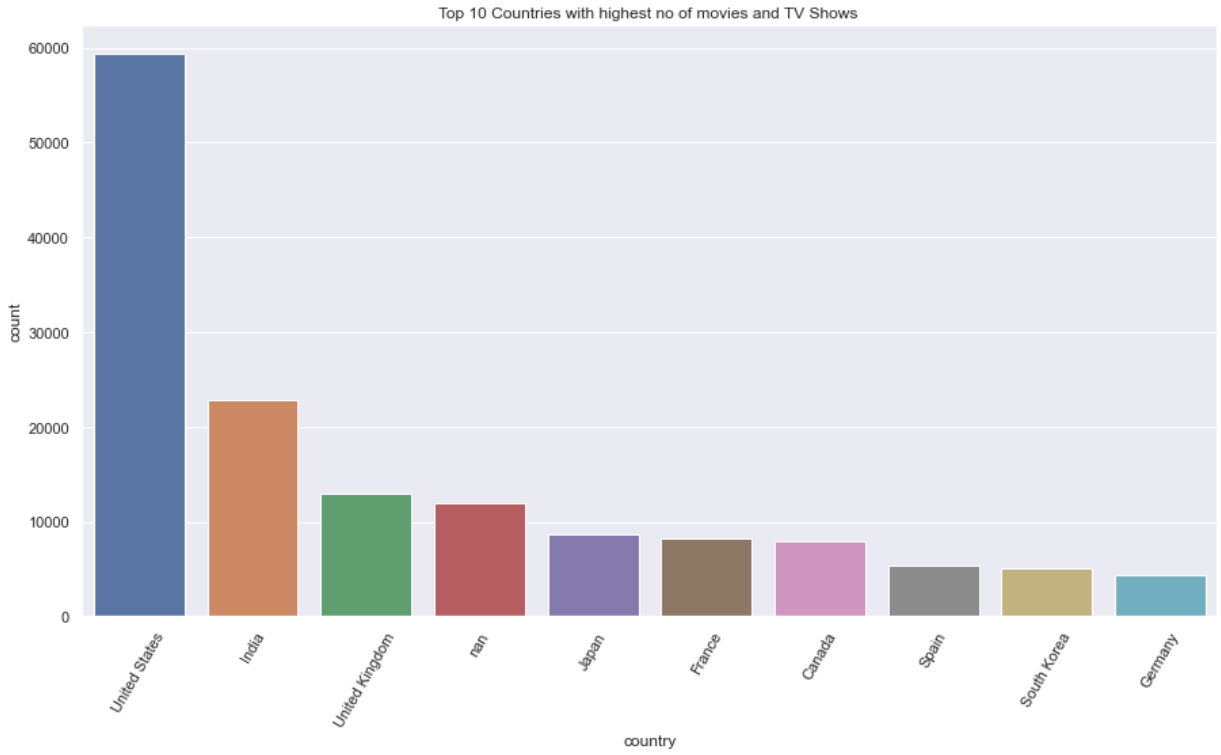


In [476]:
```
plt.title("Release year of Content")
plt.plot(df.groupby(by = ["release_year"]).release_year.count());
```



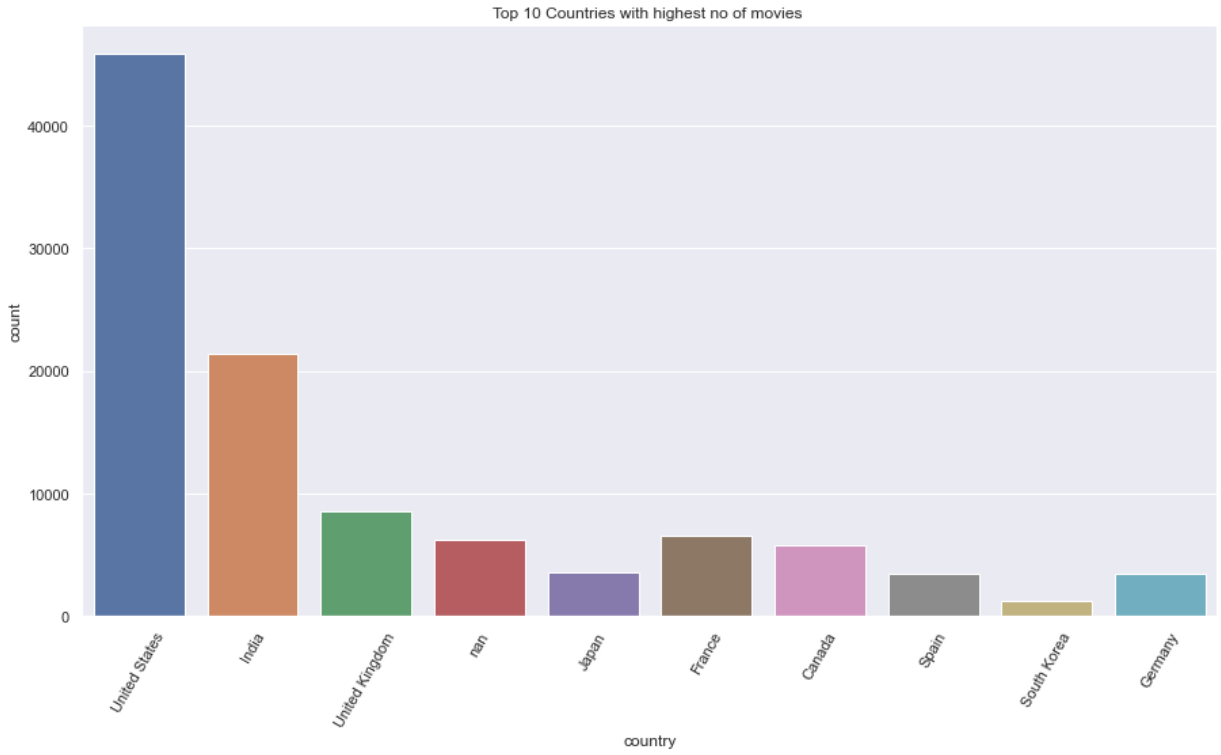
In [477]:
```
##Which country has the highest no of movies and TV shows?
plt.title("Top 10 Countries with highest no of movies and TV Shows ")
plt.xticks(rotation=60)
sns.countplot(x = df.country, order = df['country'].value_counts().index[0:10])
```

Out[477]:
```
<AxesSubplot:title={'center':'Top 10 Countries with highest no of movies and TV
Shows '}, xlabel='country', ylabel='count'>
```



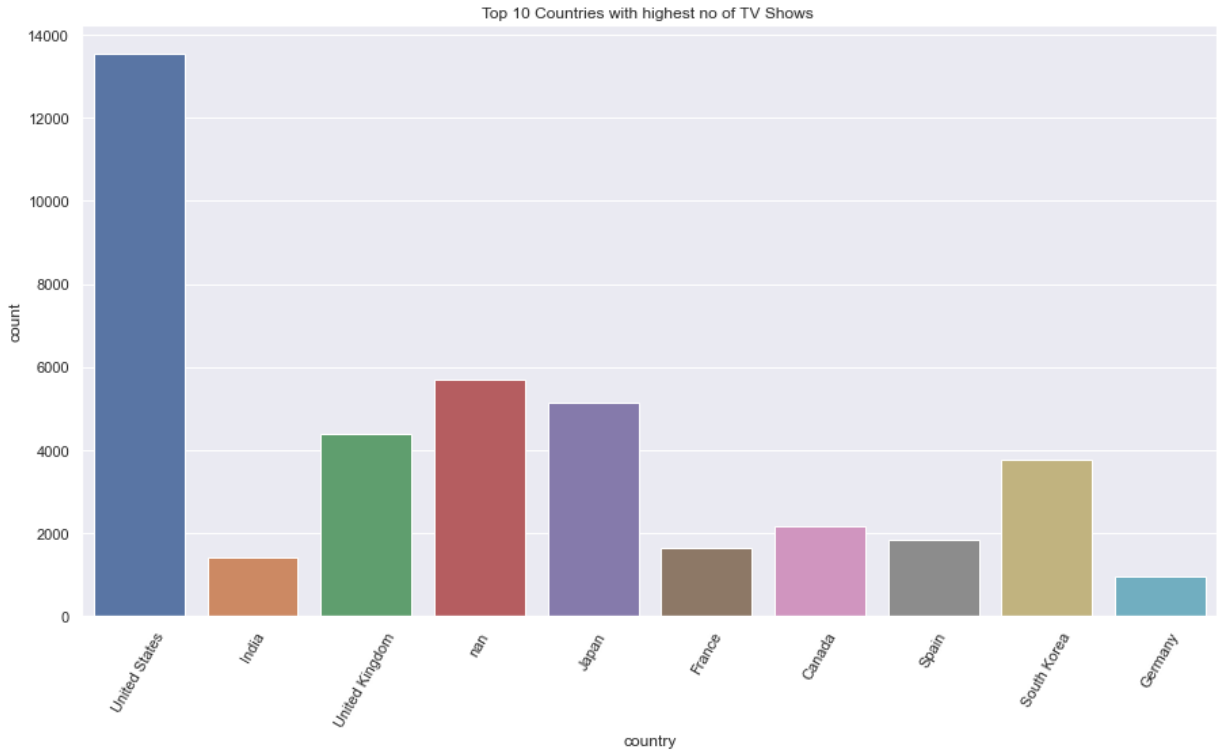
In [478]:
```
##Which country has the highest no of movies and TV shows?
plt.title("Top 10 Countries with highest no of movies ")
plt.xticks(rotation=60)
sns.countplot(x = df_movie.country, order = df['country'].value_counts().index[0:
```

Out[478]:
```
<AxesSubplot:title={'center':'Top 10 Countries with highest no of movies '}, xl
abel='country', ylabel='count'>
```



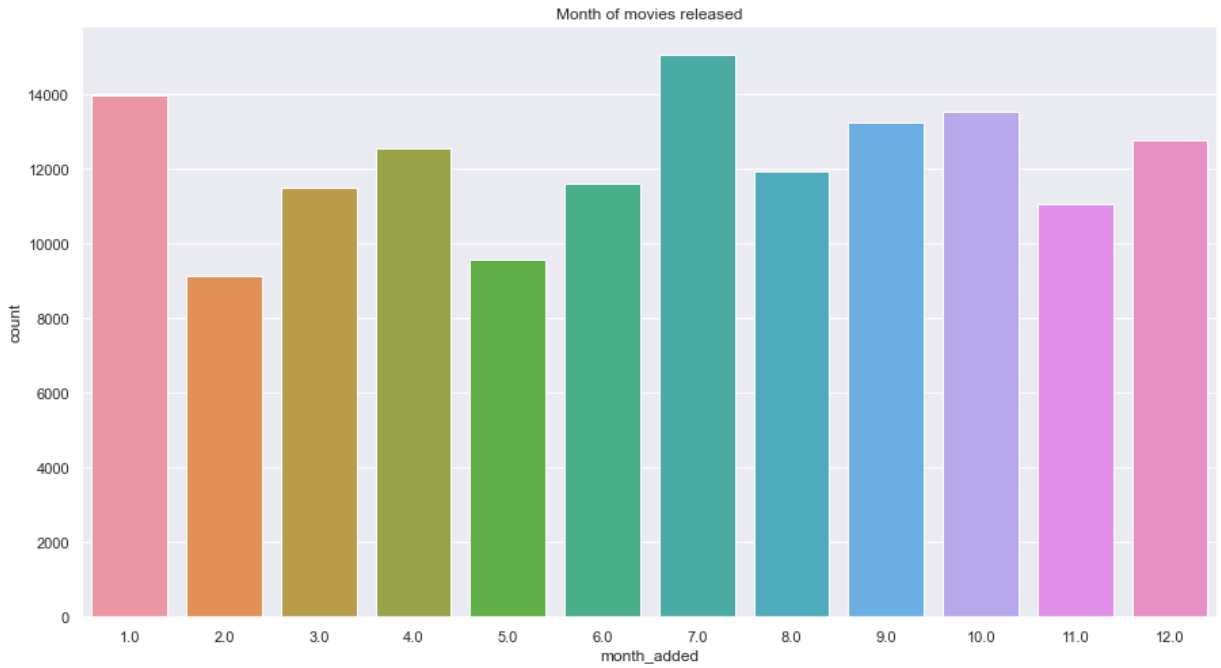
In [479]:
```
##Which country has the highest no of movies and TV shows?
plt.title("Top 10 Countries with highest no of TV Shows ")
plt.xticks(rotation=60)
sns.countplot(x = df_tvshow.country, order = df['country'].value_counts().index[0
```

Out[479]:
```
<AxesSubplot:title={'center':'Top 10 Countries with highest no of TV Shows '},
xlabel='country', ylabel='count'>
```



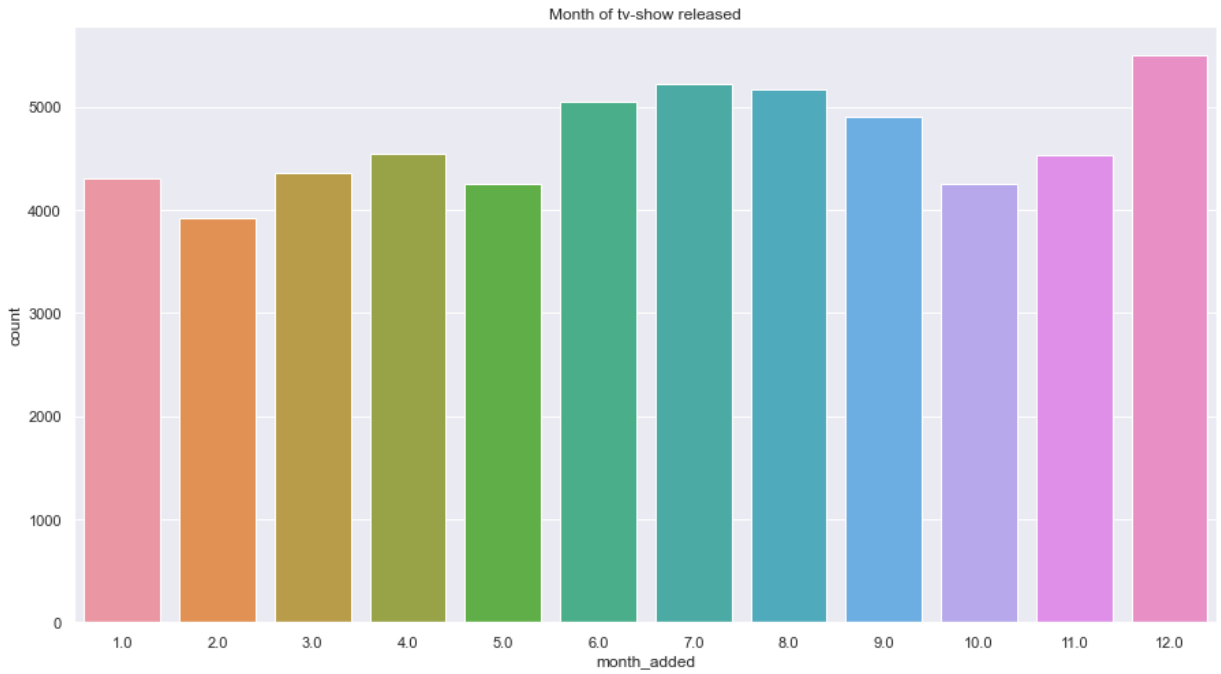
In [480]:
```
## Which month are the most no of movies released?
plt.title("Month of movies released")
sns.countplot(x= df_movie['month_added'])
```

Out[480]:
```
<AxesSubplot:title={'center':'Month of movies released'}, xlabel='month_added',
ylabel='count'>
```

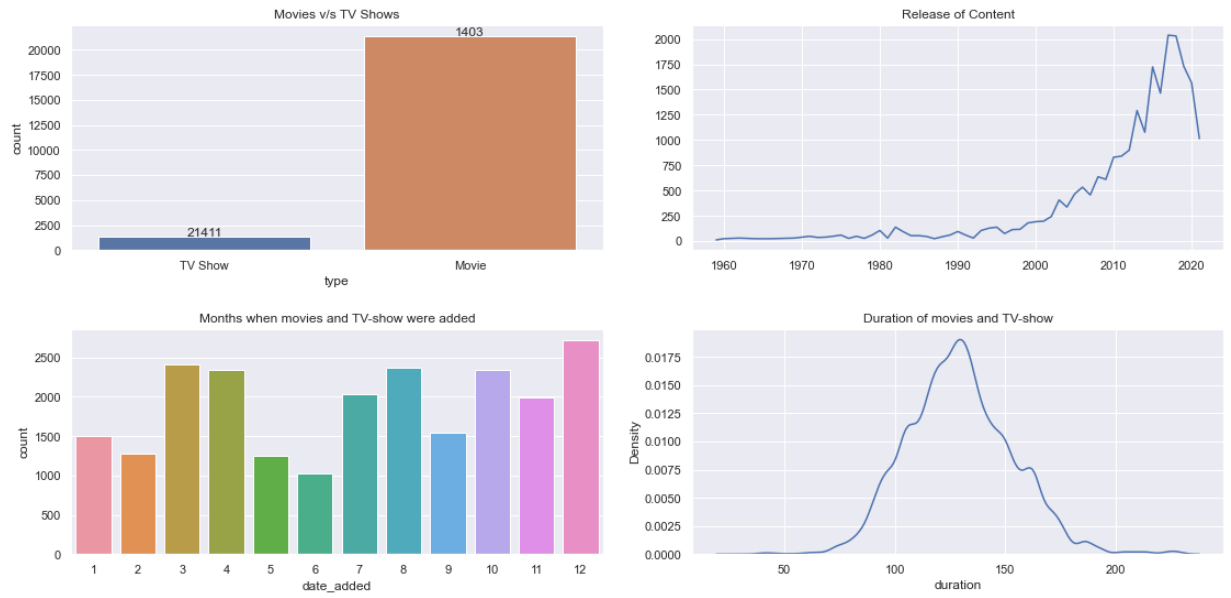


In [481]:
```
## Which month are the most no of movies released?
plt.title("Month of tv-show released")
sns.countplot(x= df_tvshow['month_added'])
```

Out[481]:
```
<AxesSubplot:title={'center':'Month of tv-show released'}, xlabel='month_adde
d', ylabel='count'>
```



In [484]:
```
#making a df for indian content
india_content= df[df["country"].str.contains('India')]
fig, axes = plt.subplots(2, 2, figsize=(16, 8))

#first graph
axes[0][0].set_title("Movies v/s TV Shows")
graph = sns.countplot(x=india_content.type, ax=axes[0,0]);
i=0
for p in graph.patches:
    height = p.get_height()
    graph.text(p.get_x()+p.get_width()/2., height + 0.1,
        india_content['type'].value_counts()[i],ha="center")
    i += 1

#graph 2
axes[0][1].set_title("Release of Content")
axes[0][1].plot(india_content.groupby(by = ["release_year"]).release_year.count()

#graph 3
axes[1][0].set_title("Months when movies and TV-show were added")
sns.countplot(x= india_content.date_added.dt.month, ax=axes[1,0])

#graph 4
axes[1][1].set_title("Duration of movies and TV-show")
indian_movies = india_content[india_content.type == "Movie"]
indian_movies_duration = indian_movies["duration"].str.replace("min","")
sns.kdeplot(data=indian_movies_duration.astype(str).astype(int))

plt.tight_layout(pad=2)
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: