- Importing essential libraries

In [104…
```
import pandas as pd
import numpy as np
import seaborn as sns
%matplotlib inline
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

- Importing data and checking the data for first 10 rows

In [104…
```
data = pd.read_csv('C:/Users/hp/Jupyter/Datasets/netflix.csv')
data.head(3)
```

Out[1041]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	li
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Docume
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	Interr TV Sh Dra My
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Cr Interr TV Sh

Problem Statement and Analysing basic metrics

1. The aim of this case study is to analyze Netflix platform shows.
2. The given data is for the shows added on Netflix between year 2008 to 2021.
3. The Netflix dataset has information about TV shows and movies available on platform.
4. It includes comma separated value and missing values in some columns which requires featuring engineering before analysis.
5. There are 6131 movies and 2676 TV shows in given data set.
6. No duplicates are available in data.
7. Aim of this data study to improve business by recommending which shows to produce.

In [104…
```
# The data given contains 8807 rows and 12 columns
data.shape
```

Out[1042]:
```
(8807, 12)
```

In [104…
```
data.loc[data['release_year']==1925]
```

Out[1043]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in
4250	s4251	TV Show	Pioneers: First Women Filmmakers*	NaN	NaN	NaN	December 30, 2018	1925	TV-14	1 Season	TV Shows

In [104…
```
data[data.duplicated()]
```

Out[1044]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description

In [104…
```
print('Oldest movie release year: ',data['release_year'].min())
print('Latest movie release year: ',data['release_year'].max())
```

```
Oldest movie release year:  1925
Latest movie release year:  2021
```

Observations on the shape of data, data types of all the attributes

In [104…
```
data.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 [104…]
```
# We are getting only release year which is the only numerical column.
data.describe()
```

Out[1047]:

	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

In [104…]
```
data.describe(include=object)
```

Out[1048]:

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

In [104…]
```
# There are 105684 elements in dataset
data.size
```

Out[1049]:
```
105684
```

In [105…]
```
# List of all columns in given data
data.columns
```

Out[1050]:
```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

In [105…]
```
# Except the release_year column all other columns are having object data type
data.dtypes
```

Out[1051]:
```
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
```

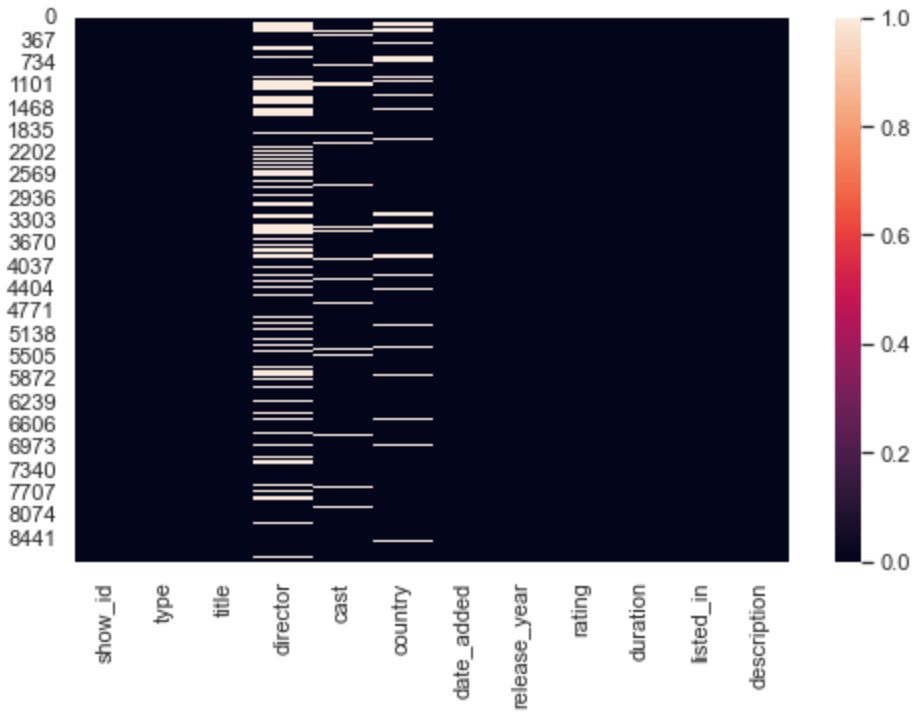
Missing Value & Outlier check

In [105…
```
# Count of null values in each column
data.isna().sum().sort_values(ascending=False) # director column has maximum missing val
```

Out[1052]:
```
director        2634
country          831
cast             825
date_added        10
rating             4
duration           3
show_id            0
type               0
title              0
release_year       0
listed_in          0
description        0
dtype: int64
```

In [105…
```
plt.figure(figsize=(8,5))
sns.heatmap(data.isna()) # Null values represent by white lines
```

Out[1053]:
```
<AxesSubplot:>
```



. Missing values for categorical variables are replaced with **Unavailable**

In [105…
```
data.fillna({'rating':'Unavailable','cast':'Unavailable','country':'Unavailable','direct
```

In [105…
```
data.isna().sum().sort_values(ascending=False)
```

Out[1055]:
```
date_added      10
duration         3
show_id          0
```

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

. Checking missing values in **Rating** column

In [105…
```
# There are three values which are misplaced we will replace them with most occuring val

data.loc[[5541,5794,5813],'rating'] = [np.NaN,np.NaN,np.NaN]
data['rating'].fillna(data['rating'].mode()[0],inplace=True)
```

In [105…
```
data['rating'].unique()
```

Out[1057]:
```
array(['PG-13', 'TV-MA', 'PG', 'TV-14', 'TV-PG', 'TV-Y', 'TV-Y7', 'R',
       'TV-G', 'G', 'NC-17', 'NR', 'Unavailable', 'TV-Y7-FV', 'UR'],
      dtype=object)
```

Different **rating categories** are replaced in 4 bands

- Kids
- Teenagers
- Parental Guidance
- Adults

In [105…
```
Age_Rating = {'PG-13':'Teenagers',
'TV-14':'Teenagers',
'TV-MA':'Adults',
'UR':'Adults',
'NR':'Adults',
'NC-17':'Adults',
'R':'Adults',
'TV-Y':'Kids',
'TV-G':'Kids',
'G':'Kids',
'TV-Y7':'Parental guidance',
'TV-Y7-FV':'Parental guidance',
'PG':'Parental guidance',
'TV-PG':'Parental guidance',
'Unavailable':'Unavailable'}
```

In [105…
```
data['Age_Ratings'] = data['rating'].replace(Age_Rating)
```

In [106…
```
data.drop(columns='rating',inplace=True) # Dropping the original column as new categoric
data['Age_Ratings'].value_counts()
```

Out[1060]:
```
Adults              4095
Teenagers           2650
Parental guidance   1490
Kids                 568
Unavailable            4
Name: Age_Ratings, dtype: int64
```

Non-Graphical Analysis: Value counts and unique attributes

In [106…
```
#Given data has 6132 movies and 2676 tv show entries

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

Out[1061]:
```
Movie      6131
TV Show    2676
Name: type, dtype: int64
```

In [106…
```
# We can see unique values in type column

data['type'].unique()
```

Out[1062]:
```
array(['Movie', 'TV Show'], dtype=object)
```

In [106…
```
data.nunique().sort_values()
```

Out[1063]:
```
type              2
Age_Ratings       5
release_year     74
duration        220
listed_in       514
country         749
date_added     1767
director       4529
cast           7693
description    8775
show_id        8807
title          8807
dtype: int64
```

- We will change data type of **date_added** column to datetime64[ns]

In [106…
```
# Data type will change to datetime64[ns]
data['date_added'] = pd.to_datetime(data['date_added'])
data['date_added'].dt.year.value_counts()
```

Out[1064]:
```
2019.0    2016
2020.0    1879
2018.0    1649
2021.0    1498
2017.0    1188
2016.0     429
2015.0      82
2014.0      24
2011.0      13
2013.0      11
2012.0       3
2009.0       2
2008.0       2
2010.0       1
Name: date_added, dtype: int64
```

- Adding two new column as **year** and **month** for analysis

In [106…
```
data['year'] = data['date_added'].dt.year
data['month'] = data['date_added'].dt.month_name()
data.head(2)
```

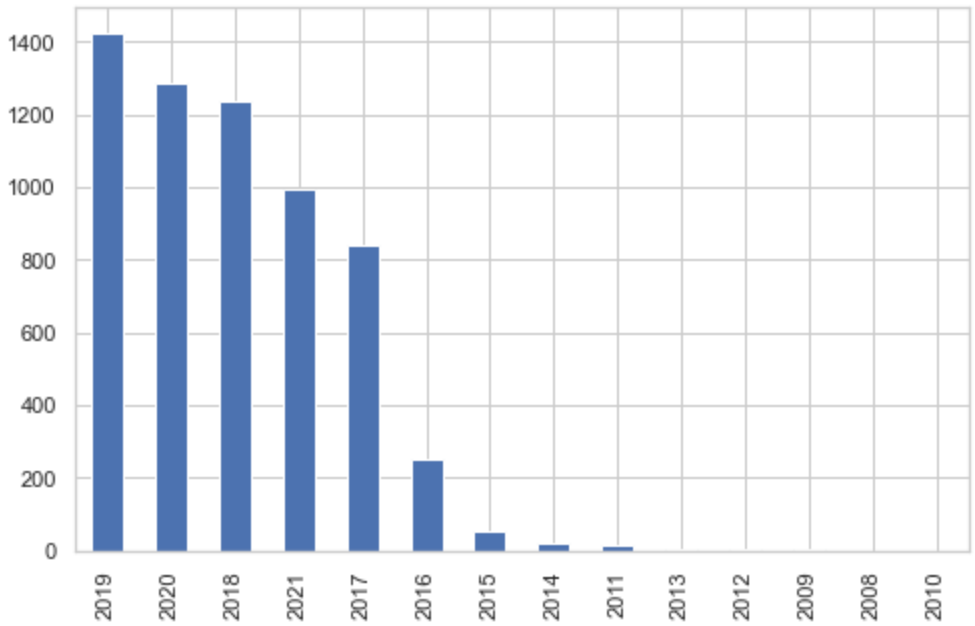
Out[1065]:

	show_id	type	title	director	cast	country	date_added	release_year	duration	listed_in
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	Unavailable	United States	2021-09-25	2020	90 min	Documentaries

1	s2	TV Show	Blood & Water	Unavailable	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	2 Seasons	International TV Shows, TV Dramas, TV Mysteries

Analysis on movies data

In [106…
```
movies_data = data.loc[data['type']=='Movie']
plt.figure(figsize=(8,5))
movies_data['date_added'].dt.year.value_counts().plot(kind='bar')
plt.show()
```



In [106…
```
# Checking long duration movie
movies_data[['Minutes','Unit']]= movies_data['duration'].str.split(' ',expand=True)
```

In [106…
```
# Converting new column from object to numeric
movies_data['Minutes']= pd.to_numeric(movies_data['Minutes'])
```

In [106…
```
# The maximun run time of movie is 312 which is 'Black Mirror: Bandersnatch'
movies_data['Minutes'].max()
```

Out[1069]:
```
312.0
```

In [107…
```
# Average time of movies is 99.58 minutes
round(movies_data['Minutes'].mean(),2)
```

Out[1070]:
```
99.58
```

Conclusion:
- Large number of movies content were added in year 2019.
- From above details we can see Black Mirror: Bandersnatch is longest duration movie with 312 minutes
- Average time for the movies is 100 minutes approximately.

Analysis on TV shows data

In [107…
```
tv_show_data = data.loc[data['type']=='TV Show']
```

In [107…
```
tv_show_data.loc[tv_show_data['duration']=='17 Seasons']['title'].value_counts()
```

Out[1072]:
```
Grey's Anatomy    1
Name: title, dtype: int64
```

In [107…
```
tv_show_data1 = tv_show_data[['title','duration']].groupby('duration')['title'].value_co
tv_show_data1.drop(columns='title',inplace=True)
tv_show_data1.reset_index(inplace=True)
tv_show_data1.groupby('duration').nunique().sort_values(by='title',ascending=False).rese
```

Out[1073]:

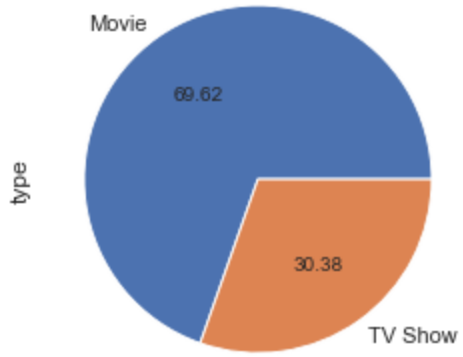
	duration	title
0	1 Season	1793
1	2 Seasons	425
2	3 Seasons	199
3	4 Seasons	95
4	5 Seasons	65
5	6 Seasons	33
6	7 Seasons	23
7	8 Seasons	17
8	9 Seasons	9
9	10 Seasons	7
10	13 Seasons	3
11	11 Seasons	2
12	12 Seasons	2
13	15 Seasons	2
14	17 Seasons	1

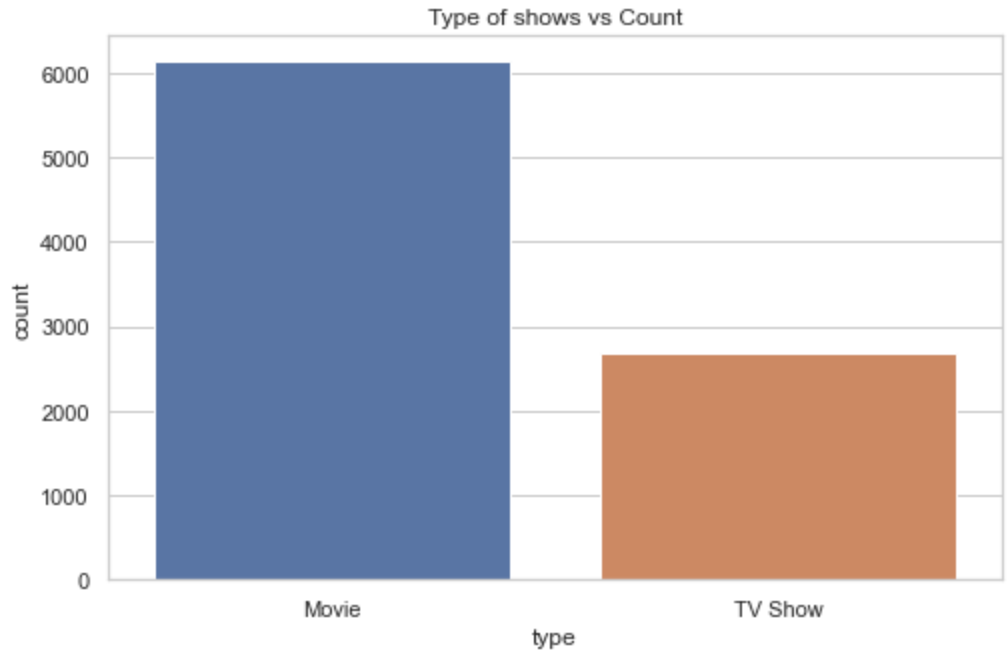
Conclusion:
- Large number of movies content were added in year 2019.
- Netflix has one show with 17 seasons i.e Grey's Anatomy.
- There are 1793 shows which ended with only 1 season or there next season are not avaiable in given data.
- Average time for the movies is 100 minutes approximately

Visual Analysis - Univariate, Bivariate

In [107…
```
data['type'].value_counts(normalize=True)*100
data['type'].value_counts().plot(kind='pie',autopct='%.2f')

plt.subplot()
plt.figure(figsize=(8,5))
sns.set_theme(style="whitegrid")
sns.countplot(x='type',data=data)
```

```
plt.title('Type of shows vs Count')
plt.show()
```



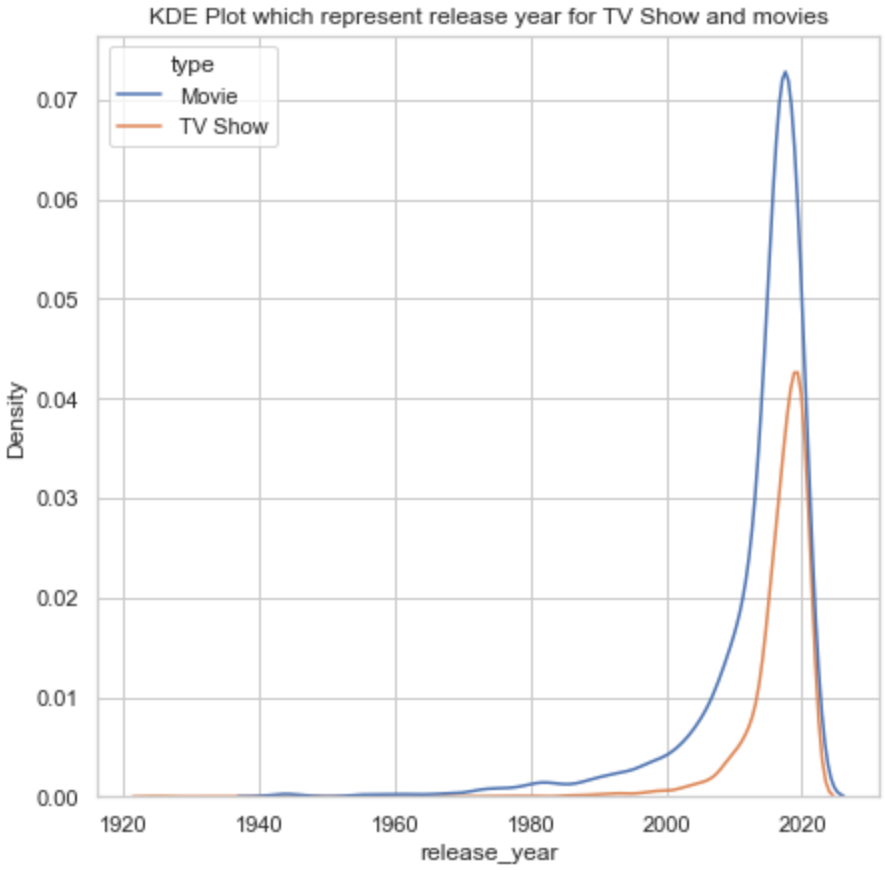


Conclusion: From above graph we can predict 70% of movies are part of content available on Netflix

In [ ]:

In [107…

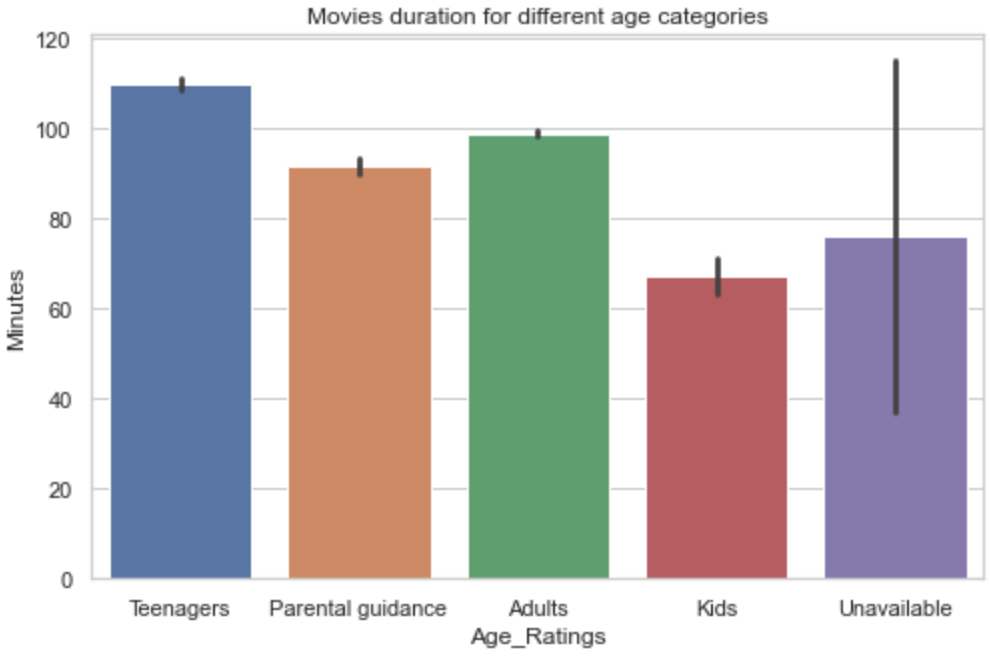
```
plt.figure(figsize=(7,7))
sns.kdeplot(data['release_year'],hue=data['type'])
plt.title('KDE Plot which represent release year for TV Show and movies')
plt.show()
```



Conclusion: From above graph we can predict more movies and tv shows were added between year 2017-2019

In [ ]:

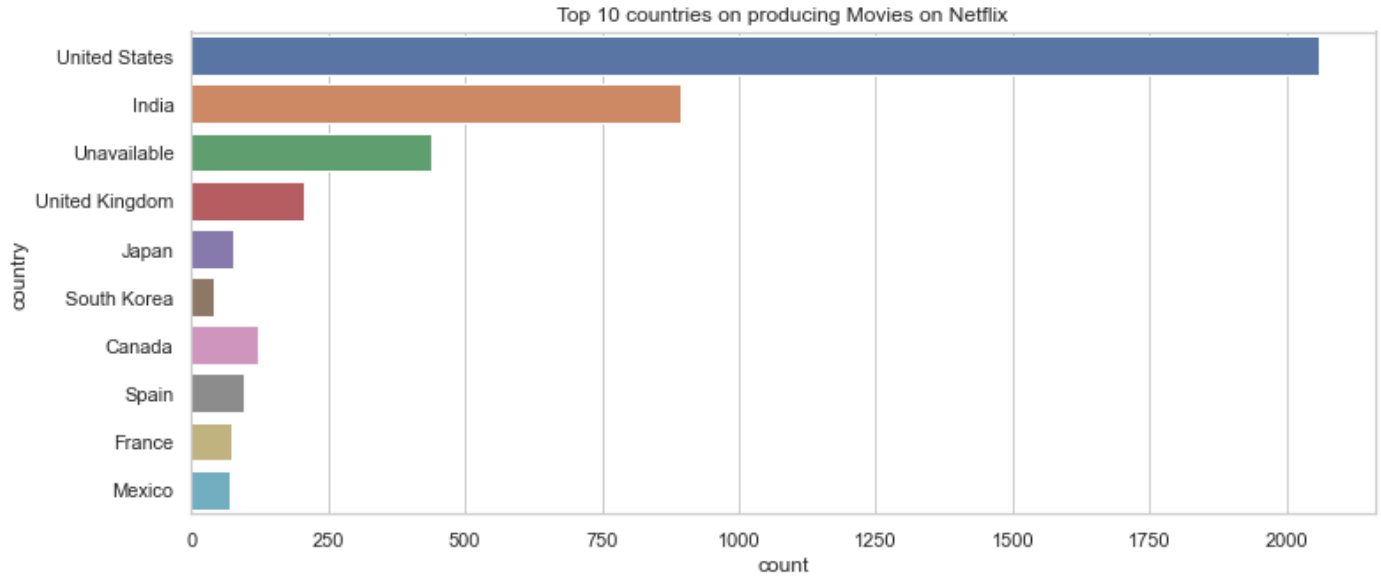
In [107…
```
plt.figure(figsize=(8,5))
plt.title('Movies duration for different age categories')
sns.barplot(x=movies_data['Age_Ratings'],y=movies_data['Minutes'])
plt.show()
```

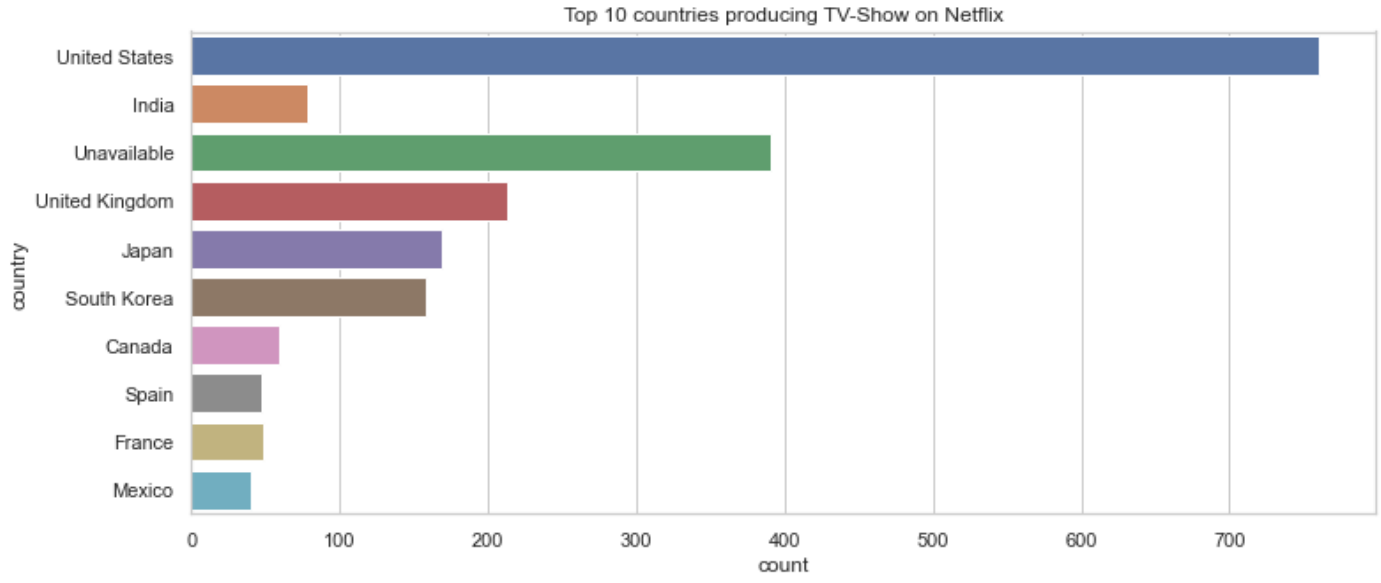


Conclusion:
1. We can see teenagers movies are large in duration.
2. Kids movies are smaller as compared to other age groups

In [ ]:

In [107…
```
plt.figure(figsize=(12,5))
sns.set_theme(style="whitegrid")
sns.countplot(y='country',order =data['country'].value_counts().index[0:10] ,data=movies
plt.title('Top 10 countries on producing Movies on Netflix')

plt.figure(figsize=(12,5))
sns.countplot(y='country',order =data['country'].value_counts().index[0:10] ,data=tv_sho
plt.title('Top 10 countries producing TV-Show on Netflix')
plt.show()
```

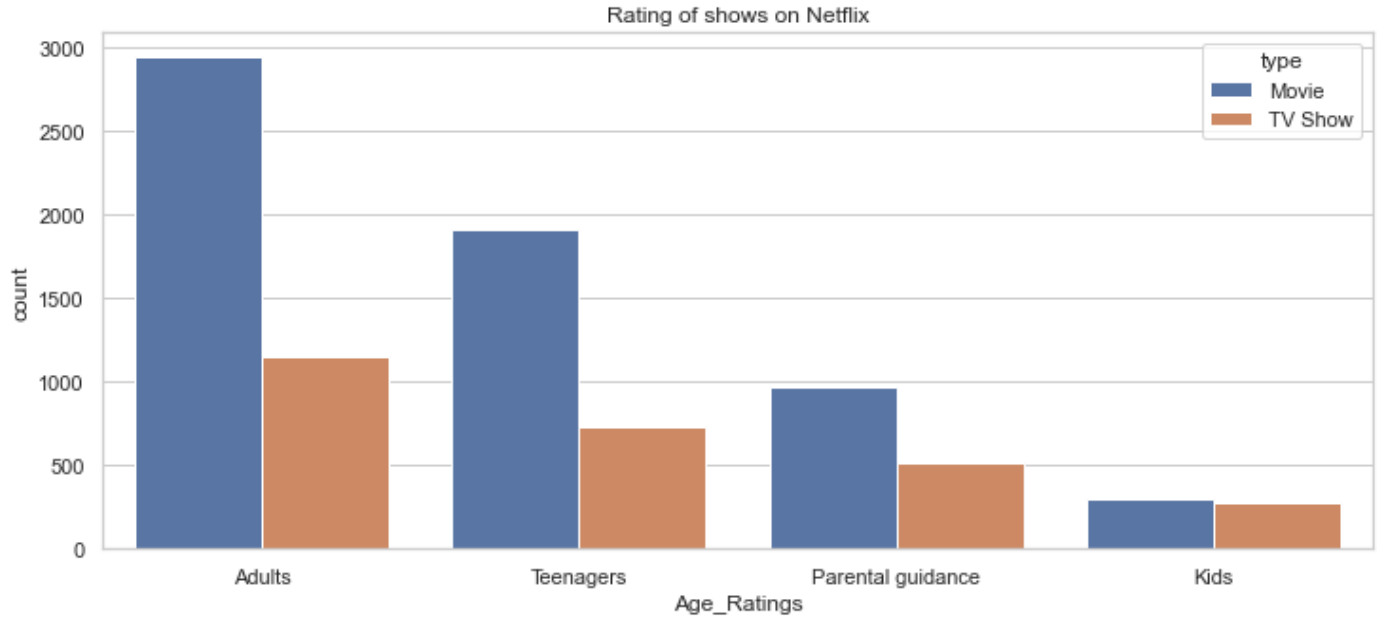




Conclusion:
1. United States is leader in producing both Movies and TV shows.
2. United Kingdom prodcues more TV shows followed by United States.
3. While India produces more movies followed by United States.
4. Japan and South Korea are also big contributor to TV shows

In [ ]:

In [107…
```
plt.figure(figsize=(12,5))
sns.countplot(x='Age_Ratings',order = data['Age_Ratings'].value_counts().index[0:4],data
plt.title('Rating of shows on Netflix')
plt.show()
```

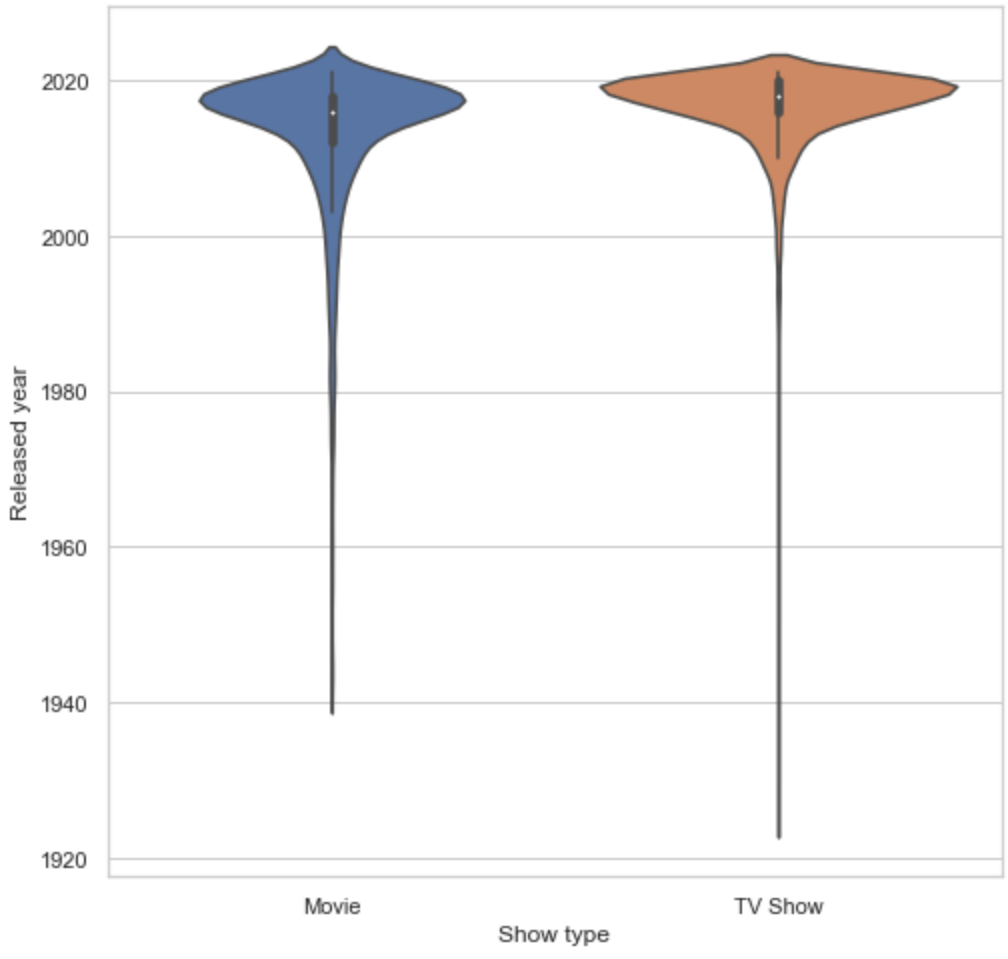


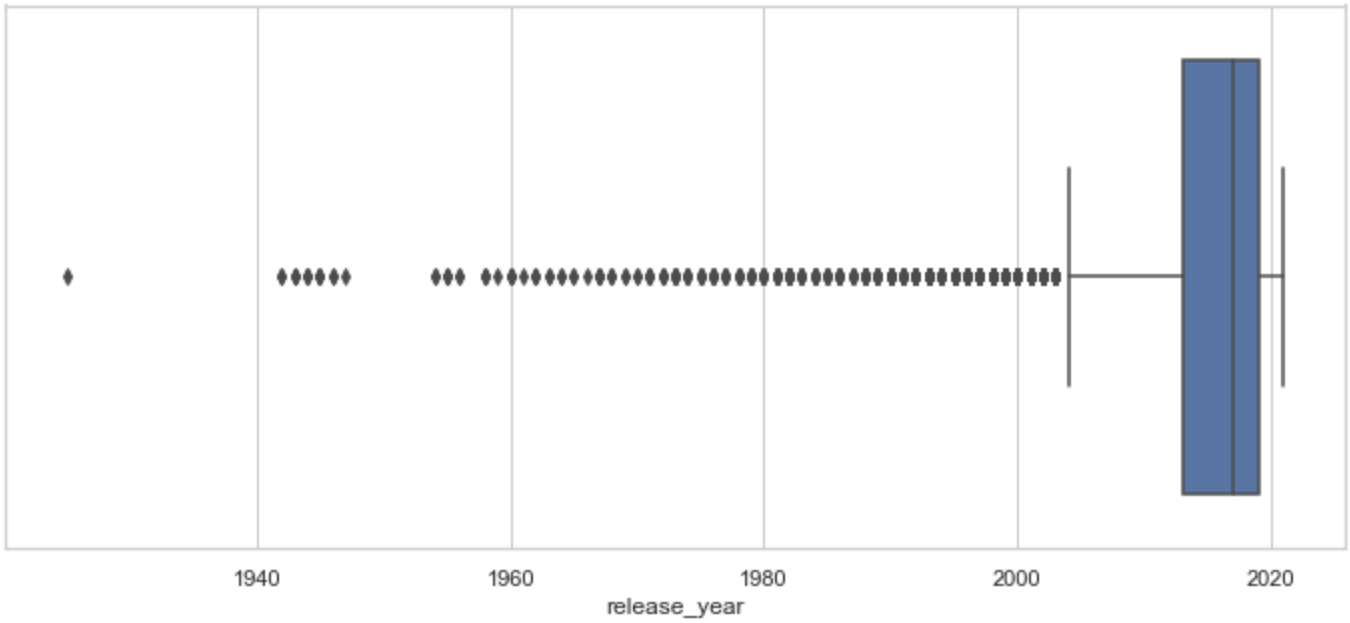
Conclusion: Netflix has more adult content that other categories

In [ ]:

In [108…

```
plt.figure(figsize=(8,8))
sns.violinplot(x='type',y='release_year',data=data)
plt.xlabel('Show type')
plt.ylabel('Released year')
plt.show()

plt.figure(figsize=(12,5))
sns.boxplot(data['release_year'])
plt.show()
```

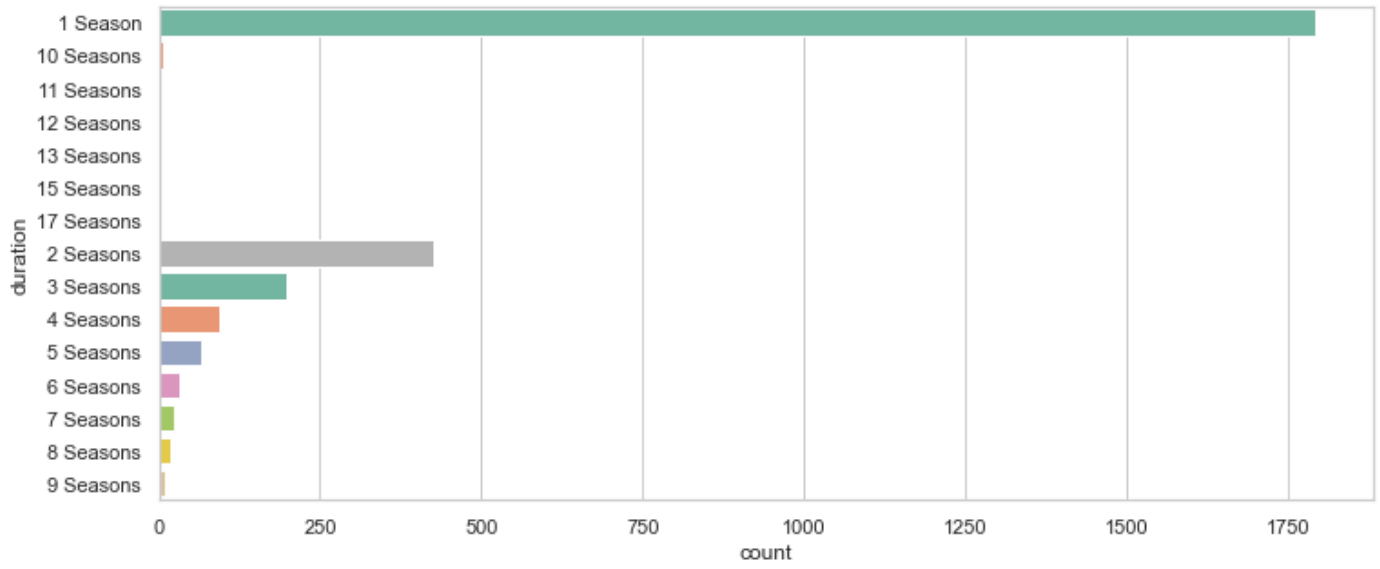




Conclusion:
1. Most content on Netflix is release in year 2019 and 2020.
2. Before 2000 content is very less and can be seen as outlier in box plot
3. In year 2019 highest number of shows and movies were added on platform.
4. In year 2010 lowest number of shows and movies added.

In [ ]:

In [108…
```
plt.figure(figsize=(12,5))
sns.countplot(y='duration',data=tv_show_data1,palette = "Set2")
plt.show()
```

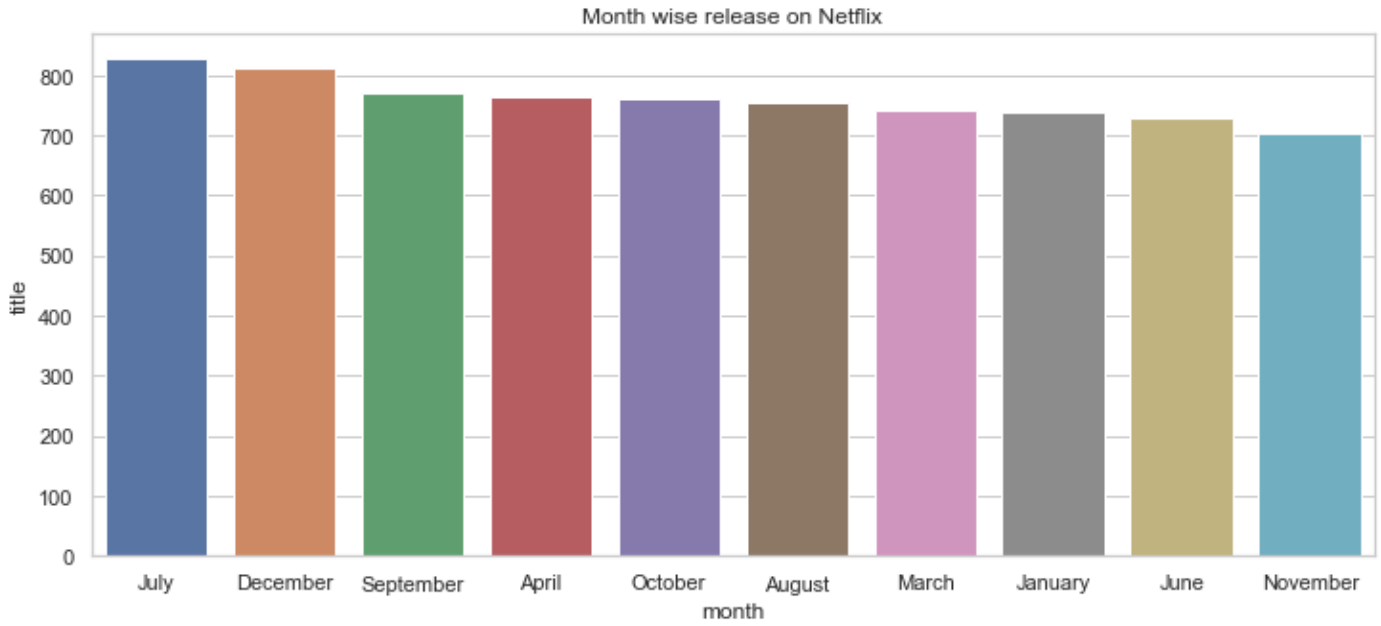


Conclusion:
1. There are nearly 1800 tv shows with season 1.
2. Most shows end up in 1 or 2 season.
3. Only one show is available with 17 seasons.

In [ ]:

In [108…
```
best_month = data.groupby('month')['title'].nunique()
best_month = pd.DataFrame(best_month).reset_index()
best_month = best_month.sort_values(by='title',ascending=False).head(10)
best_month
plt.figure(figsize=(12,5))
sns.barplot(x='month',y='title',data = best_month)
```

```
plt.title('Month wise release on Netflix')
plt.show()
```

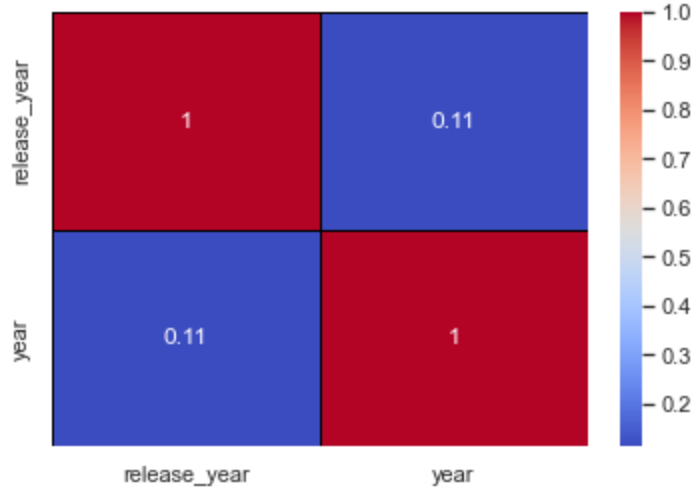


Conclusion: This graph predicts most content is released in month of July and December.

Correlation: Heatmaps, Pairplots

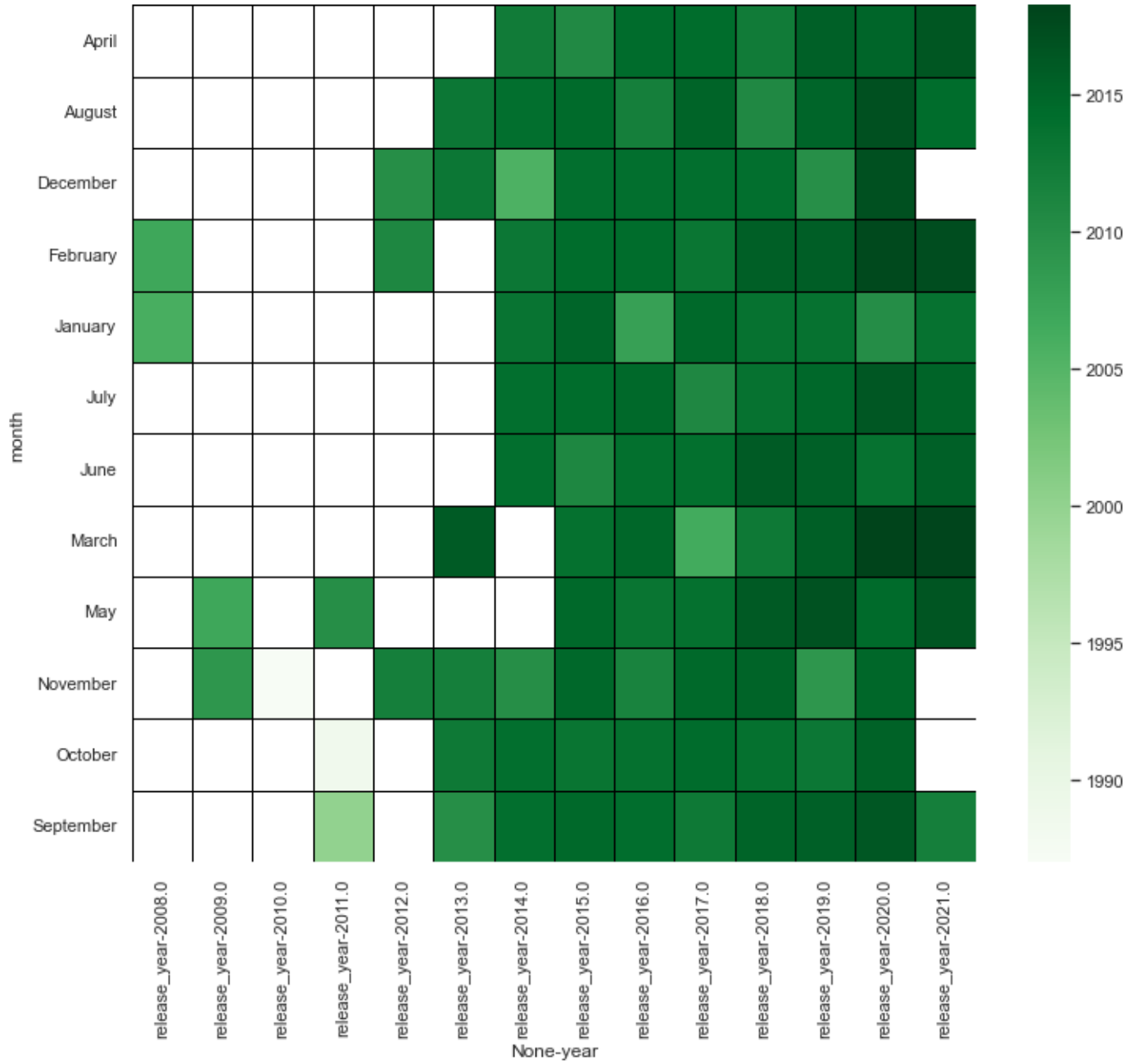
In [108…
```
tc = data.corr()
sns.heatmap(tc,annot=True,cmap='coolwarm',linecolor='black',linewidths=1)
```

Out[1083]: <AxesSubplot:>



In [108…
```
plt.figure(figsize=(12,10))
ft = data.pivot_table(index='month',columns='year')
sns.heatmap(ft,cmap='Greens',linecolor='black',linewidths=1)
```

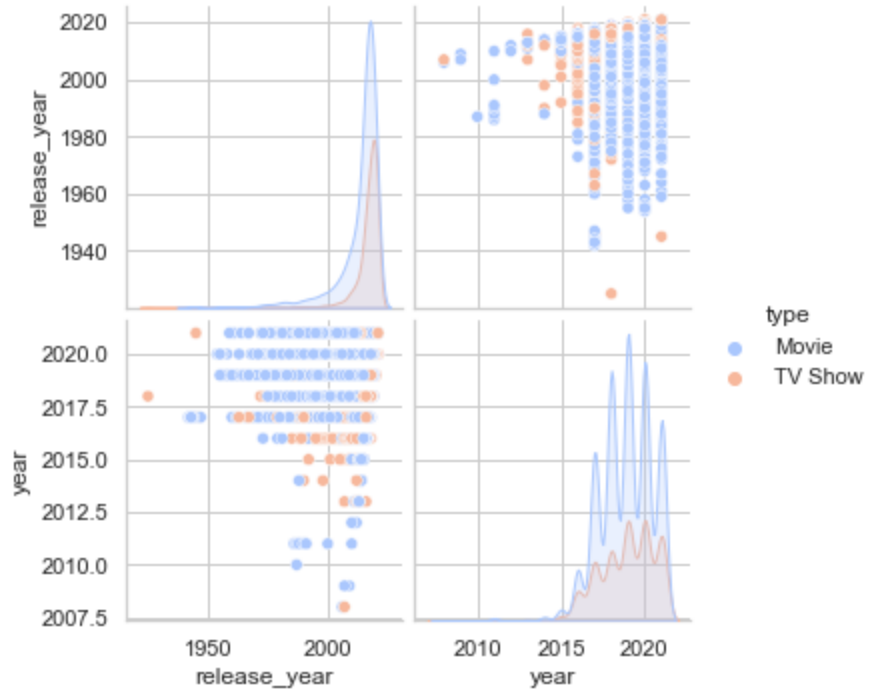
Out[1084]: <AxesSubplot:xlabel='None-year', ylabel='month'>



In [108…

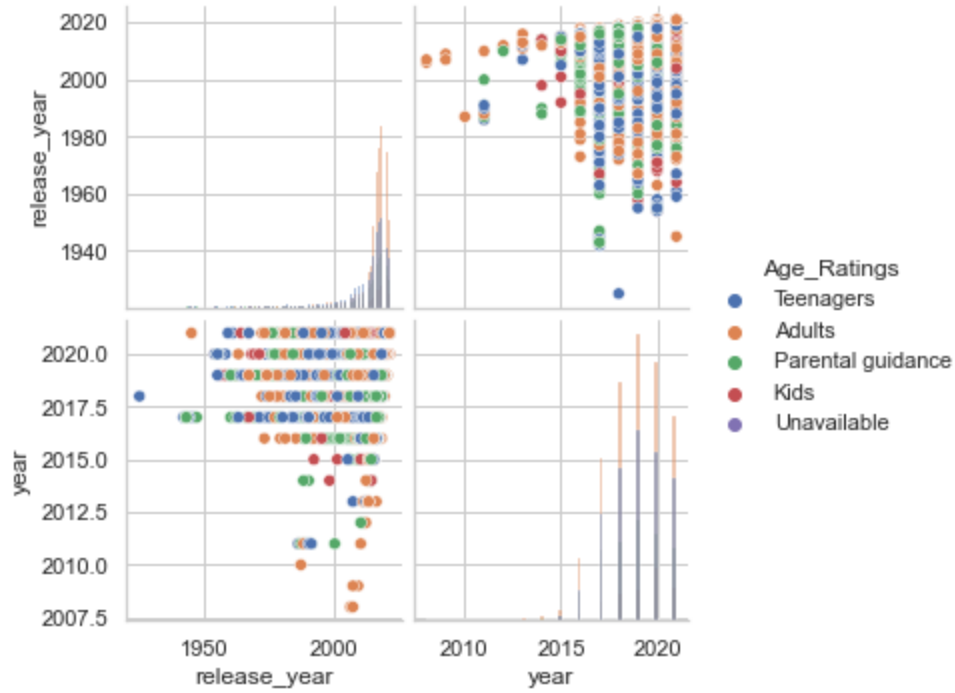
```
plt.figure(figsize=(10,10))
sns.pairplot(data,hue='type',palette='coolwarm')
plt.show()
```

```
<Figure size 720x720 with 0 Axes>
```



In [108…
```
plt.figure(figsize=(10,10))
sns.pairplot(data,hue='Age_Ratings',diag_kind="hist",height=2.5)
plt.show()
```

```
<Figure size 720x720 with 0 Axes>
```



Insights based on Non-Graphical and Visual Analysis

Following steps are done for unnesting data for the columns Director, Cast, Country and Genre(Listed_in)

In [108…
```
# Unpivoting the Cast column
```

In [108…
```
# Step 1 - Spliting the value and applying lambda function to all values
```

```
constraints_cast = data['cast'].apply(lambda x: str(x).split(', ')).to_list()

# Step 2 - Converting series into dataframe with index as title column
df_cast = pd.DataFrame(constraints_cast,index=data['title'])

# Step 3: Using stack
df_cast = df_cast.stack()

#Step 4: Converting series into dataframe
df_cast = pd.DataFrame(df_cast)

#Step 5: Reset index from title to implicit index
df_cast.reset_index(inplace=True)

# Step 6: Renaming the columns with relevant name
df_cast = df_cast.rename(columns  = {0:'Cast'})

# Keeping the releavant column in final data
df_cast.drop(columns=['level_1'],inplace=True)
df_cast.head(5)
```

Out[1088]:

	title	Cast
0	Dick Johnson Is Dead	Unavailable
1	Blood & Water	Ama Qamata
2	Blood & Water	Khosi Ngema
3	Blood & Water	Gail Mabalane
4	Blood & Water	Thabang Molaba

In [108…

```
# Same steps will be followed for director, listed_in and country column
```

In [109…

```
# Unpivoting the director column
constraints_director = data['director'].apply(lambda x: str(x).split(', ')).to_list()
df_director = pd.DataFrame(constraints_director,index=data['title'])
df_director = df_director.stack()
df_director = pd.DataFrame(df_director)
df_director.reset_index(inplace=True)
df_director = df_director.rename(columns  = {0:'Director'})
df_director.drop(columns=['level_1'],inplace=True)
df_director.head(5)

# Unpivoting the listed_in column
constraints_genres = data['listed_in'].apply(lambda x:str(x).split(', ')).to_list()
df_genre = pd.DataFrame(constraints_genres,index=data['title'])
df_genre = df_genre.stack()
df_genre = pd.DataFrame(df_genre)
df_genre.reset_index(inplace=True)
df_genre.drop(columns=['level_1'],inplace=True)
df_genre = df_genre.rename(columns={0:'listed_in'})

# Unpivoting the country column
constraints_country = data['country'].apply(lambda x:str(x).split(', ')).to_list()
df_country = pd.DataFrame(constraints_country,index=data['title'])
df_country = df_country.stack()
df_country = pd.DataFrame(df_country)
df_country.reset_index(inplace=True)
df_country.drop(columns=['level_1'],inplace=True)
df_country = df_country.rename(columns={0:'Country'})
```

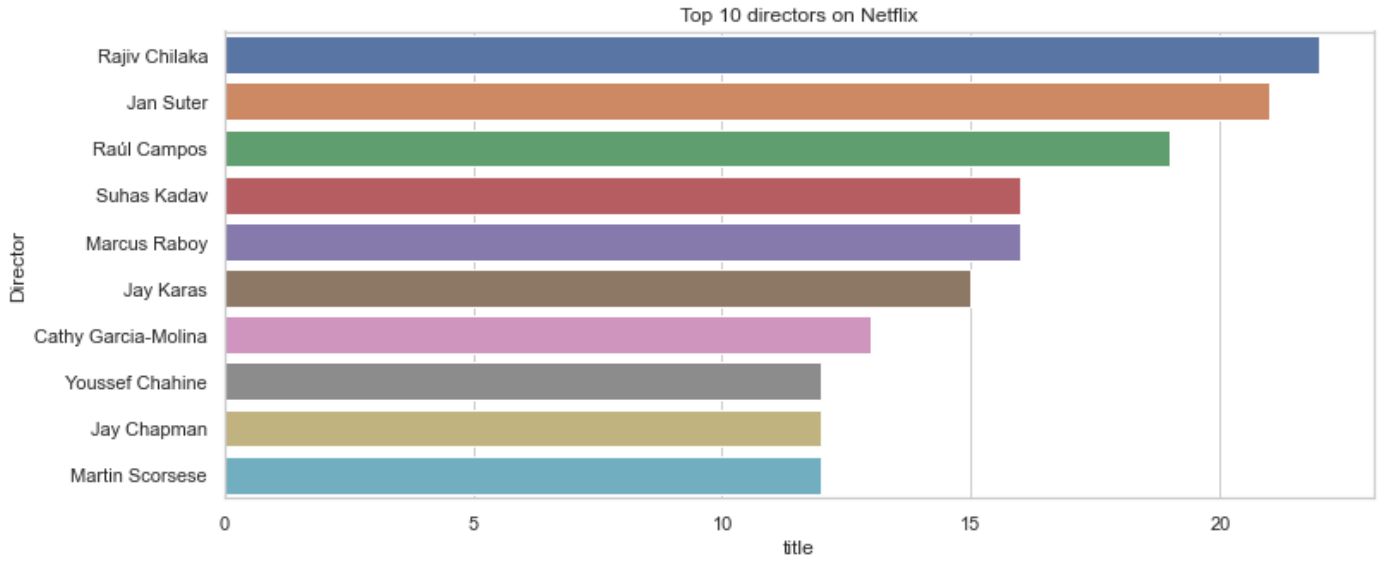
In [109…

```
df1 = pd.merge(df_cast,df_director,on='title') # Director and case dataset are merged as
df2 = pd.merge(df_country,df_genre,on='title') # Country and Genre(listed_in) dataset ar
```

In [109…
```
df = pd.merge(df1,df2,on='title')
```

In [109…
```
final_data= pd.merge(data,df,on='title')
final_data.drop(columns=['cast','director','country','listed_in_x','description'],inplac
```

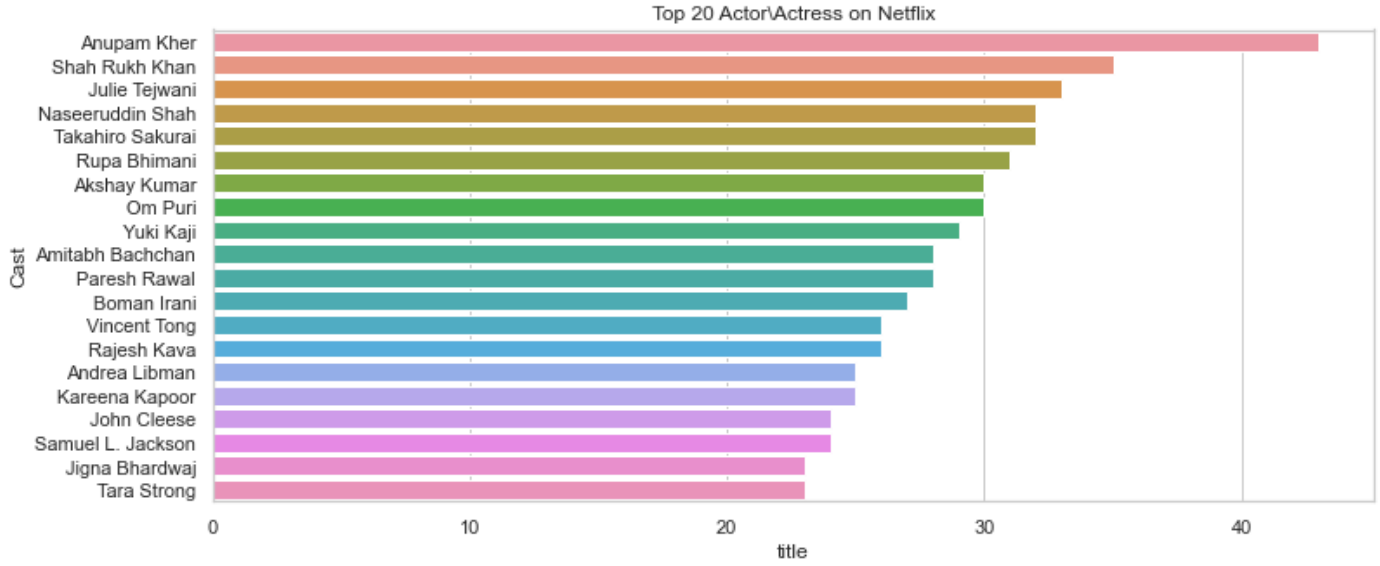
In [109…
```
popular_director = final_data.groupby(['Director'])['title'].nunique()
popular_director = pd.DataFrame(popular_director).reset_index()
popular_director = popular_director[popular_director['Director'] != 'Unavailable']
popular_director = popular_director.sort_values(by='title',ascending=False).head(10)

plt.figure(figsize=(12,5))
sns.barplot(y='Director',x='title',data = popular_director)
plt.title('Top 10 directors on Netflix')
plt.show()
```



Conclusion: :This graph shows top 10 director on Netflix

In [ ]:
```
```

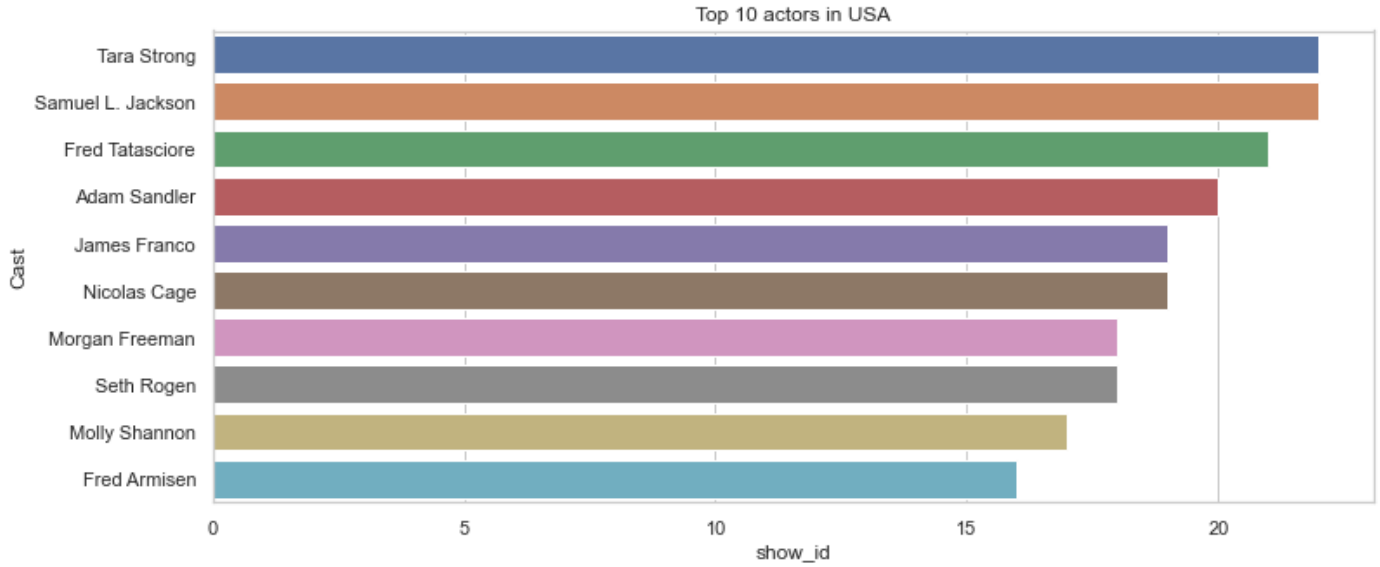
In [109…
```
# Popular actor\actress

final_data.head(5)
popular_cast = final_data.groupby('Cast')['title'].nunique()
popular_cast = pd.DataFrame(popular_cast).reset_index()
popular_cast = popular_cast[popular_cast['Cast'] != 'Unavailable']
popular_cast = popular_cast.sort_values(by='title',ascending=False).head(20)

plt.figure(figsize=(12,5))
sns.barplot(y='Cast',x='title',data = popular_cast)
plt.title('Top 20 Actor\Actress on Netflix')
plt.show()
```



Conclusion:
1. Anupan Kher is actor with most number of shows\movies on platform followed by Shah Rukh Khan
2. He has 43 shows while shah rukh khan has 35 shows.

In [ ]:

In [109…
```
Popular_in_USA= final_data[(final_data['Country']=='United States') & (final_data['Cast'
Popular_in_USA = Popular_in_USA[['show_id','title','listed_in_y','Cast']]
Popular_in_USA = Popular_in_USA.groupby(['Cast'])['show_id'].nunique()
Popular_in_USA = pd.DataFrame(Popular_in_USA).reset_index()
Popular_in_USA = Popular_in_USA.sort_values(by='show_id',ascending=False).head(10)

plt.figure(figsize=(12,5))
sns.barplot(x='show_id',y='Cast',data = Popular_in_USA)
plt.title('Top 10 actors in USA')
plt.show()
```

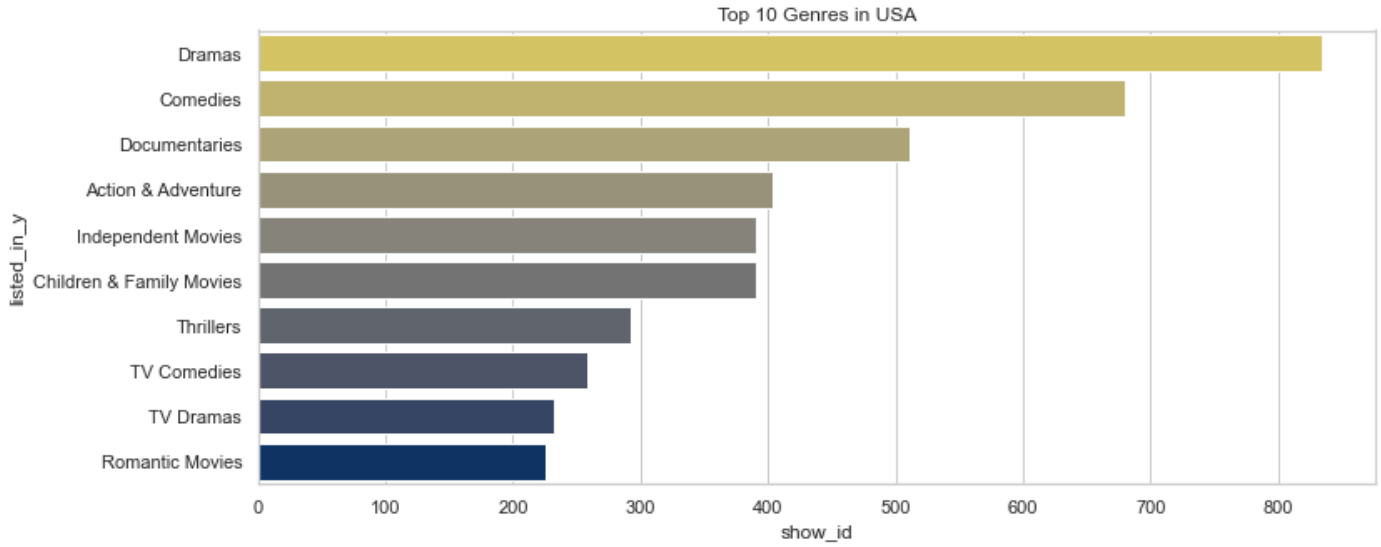


Conclusion: Tara Strong and Samuel L. Jackson are leading actors in United States.

In [ ]:

In [109…
```
USA = final_data[final_data['Country']=='United States']
USA = USA[['show_id','title','listed_in_y','Cast']]
USA = USA.groupby(['listed_in_y'])['show_id'].nunique()
USA = pd.DataFrame(USA).reset_index()
USA = USA.sort_values(by='show_id',ascending=False).head(10)
```

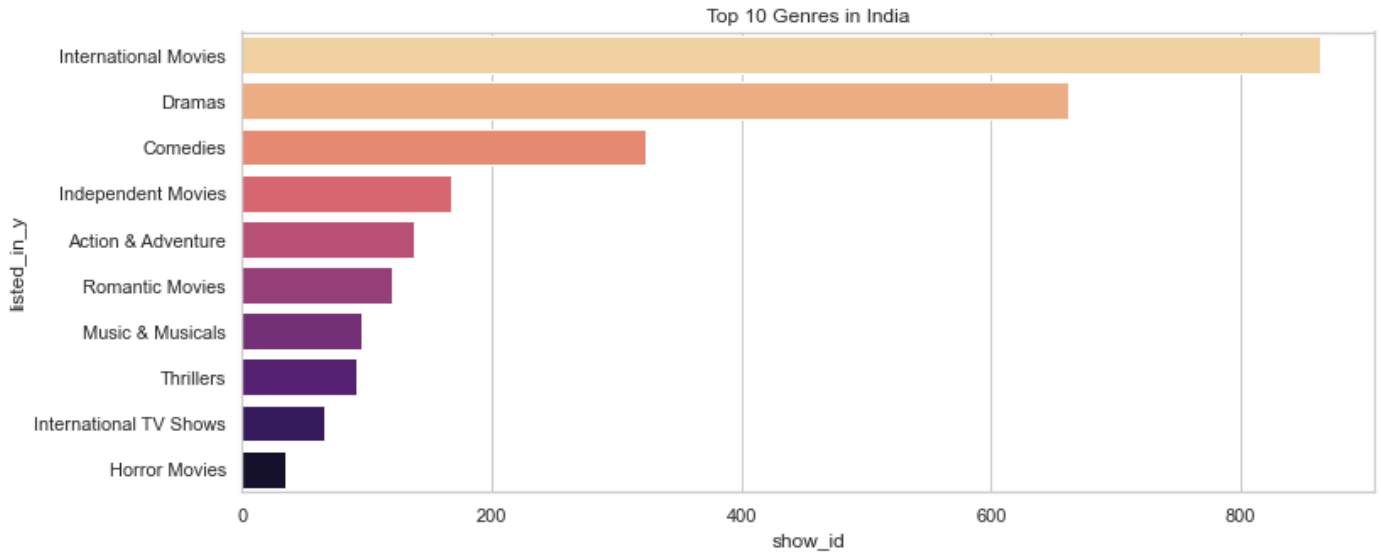
```
plt.figure(figsize=(12,5))
sns.barplot(x='show_id',y='listed_in_y',palette="cividis_r",data = USA)
plt.title('Top 10 Genres in USA')
plt.show()
```



Conclusion: Dramas and Comedies are most popular generes in USA.

In [ ]:

In [109…
```
India = final_data[final_data['Country']=='India']
India = India[['show_id','title','listed_in_y','Cast']]
India = India.groupby(['listed_in_y'])['show_id'].nunique()
India = pd.DataFrame(India).reset_index()
India = India.sort_values(by='show_id',ascending=False).head(10)


plt.figure(figsize=(12,5))
sns.barplot(x='show_id',y='listed_in_y',palette='magma_r',data = India)
plt.title('Top 10 Genres in India')
plt.show()
```

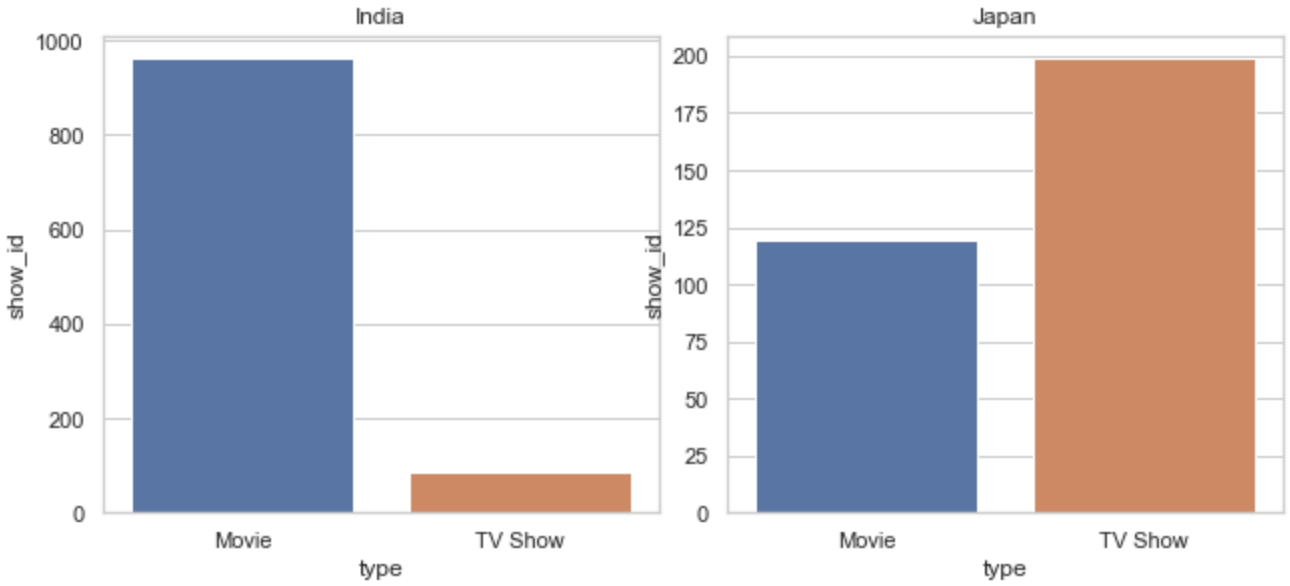


Conclusion: International movies are popular in India.

In [ ]:

In [109…
```
# Understanding content produced in different countries

fig, axes = plt.subplots(1,2,figsize=(9,4))
```

```
fig.tight_layout()
fig.subplots_adjust(hspace=0.125, wspace=0.125)

data_country1 = final_data[final_data['Country'] == 'India']
data_country1 = data_country1.groupby(['type'])['show_id'].nunique()
data_country1 = pd.DataFrame(data_country1).reset_index()
sns.barplot(x='type',y='show_id',data=data_country1,ax=axes[0]).set(title='India')


data_country2 = final_data[final_data['Country'] == 'Japan']
data_country2 = data_country2.groupby(['type'])['show_id'].nunique()
data_country2 = pd.DataFrame(data_country2).reset_index()
sns.barplot(x='type',y='show_id',data=data_country2,ax=axes[1]).set(title='Japan')

plt.show()
```



Conclusion: Movies produced in India are more than TV shows which is opposite to Japan

In [ ]:

In [110…
```
final_data.loc[final_data['listed_in_y']=='Documentaries']['Country'].value_counts().hea
```

Out[1100]:
```
United States     1205
United Kingdom     255
Unavailable        211
France             116
Canada              95
Spain               50
Mexico              35
India               32
Germany             27
Italy               23
Name: Country, dtype: int64
```

Conclusion: Top 10 countries for producing Documentaries

In [ ]:

In [110…
```
less_content_countries = final_data.groupby(['Country'])['title'].nunique()
less_content_countries = pd.DataFrame(less_content_countries).reset_index()
less_content_countries.sort_values(by='title',ascending=False).tail(10)
```

Out[1101]:

	Country	title
54	Jamaica	1

84	Paraguay	1
83	Panama	1
99	Slovakia	1
82	Palestine	1
101	Somalia	1
70	Mongolia	1
34	Ecuador	1
33	East Germany	1
36	Ethiopia	1

Conclusion: Bottom 10 countries with less content

In [110…
```
Popular_actor_director = final_data.groupby(['Director','Cast'])['title'].nunique()
Popular_actor_director = pd.DataFrame(Popular_actor_director).reset_index()
Popular_actor_director = Popular_actor_director[(Popular_actor_director['Director'] !='U
Popular_actor_director.sort_values(by='title',ascending=False).head(10)
```

Out[1102]:

	Director	Cast	title
35755	Rajiv Chilaka	Julie Tejwani	19
35761	Rajiv Chilaka	Rajesh Kava	19
35754	Rajiv Chilaka	Jigna Bhardwaj	18
35762	Rajiv Chilaka	Rupa Bhimani	18
35770	Rajiv Chilaka	Vatsal Dubey	16
35767	Rajiv Chilaka	Swapnil	13
35758	Rajiv Chilaka	Mousam	13
43526	Suhas Kadav	Saurav Chakraborty	8
62338	Yılmaz Erdoğan	Yılmaz Erdoğan	7
39089	S.S. Rajamouli	Sathyaraj	7

Conclusion: Director Rajiv Chilaka has most content with Rajesh Kava and Julie Tejwani

In [ ]:
```
```

In [110…
```
kids_show = final_data.groupby(['Country','Age_Ratings'])['title'].nunique()
kids_show = pd.DataFrame(kids_show).reset_index()
kids_show = kids_show.loc[kids_show['Age_Ratings']=='Kids']
kids_show.sort_values(by='title',ascending=False).head(5)
```

Out[1103]:

	Country	Age_Ratings	title
291	United States	Kids	255
277	Unavailable	Kids	110
45	Canada	Kids	64
286	United Kingdom	Kids	63
83	France	Kids	29

Conclusion: Top 5 kids show producers country list

Business Insights

1. In year 2019 highest number of shows were added on platform.
2. From the count plot it looks like Netflix is still investing more on the movies in compare to the web series.
3. 70% of content is movies and 30% of content is TV shows.
4. Dramas and comedies are two popular genre in USA.
5. International movies and dramas are most popular genres on Netflix.
6. Documentries are produced by United states and United Kingdom.
7. The number of content titles on Netflix continued to increase from 2012 to 2019
8. Tara Strong is most popular actor in United States.

Movies

1. International movies are most watched genre in India
2. Rajiv Chilaka is most popular director on Netflix. He has 22 movies on platform.
3. Average time of movies is 99.58 minutes.
4. Netflix has more content for adults as compare to other categories which can be seen from countplot on age ratings column.

Tv Shows

1. From given data 'Pioneers: First Women Filmmakers' is the oldest TV show which was released in 1925.
2. Most of Netflix shows(1793) end with season 1. Netflix has only 1 show with 17 seasons(Grey's Anatomy) and 2 shows with 17 seasons.
3. Maximum number of shows are added in month of July and December.

Recommendations

Movies

1. Netflix should target to add more movies in Unites states and India as compare to TV Series.
2. Netflix should release more content other than with category for adults.
3. Company should target countries like East Germany, Slovakia,Ethopia where less content is published.
4. Duration of movies should be kept between 90 mins to 110 mins.
5. Comedy shows should be added in more numbers in United States.

TV Shows

1. Kids shows content should be made avilable in more countries to increase popularity of this shows.
2. Company should target to add more TV shows in Japan and South Korea.
3. After 2021 trend shows people are most interested in TV shows than movies.