Netflix Case Study

By Gautam Naik (gautamnaik1994@gmail.com)

Google Collab Link: https://colab.research.google.com/github/gautamnaik1994/NetflixDataAnalysisCaseStudy/blob/main/CaseStudy.ipynb
Github Repo Link: https://github.com/gautamnaik1994/NetflixDataAnalysisCaseStudy
Github Pages Link: https://gautamnaik1994.github.io/NetflixDataAnalysisCaseStudy/

Business Problem

- Help Netflix in deciding which type of shows/movies to produce
- How to grow the business in different countries

Metric

- Since there is not data about views count, user star rating we are going to use the count of content added to Netflix as the metric.
- We will also use the count of cast, director, rating as measure of popularity.

Table of contents

In [ ]:
```
import pandas as pd
import numpy as np
import duckdb
import seaborn as sns
import matplotlib.gridspec as gridspec
import matplotlib.pyplot as plt
import datetime
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
sns.set_style('darkgrid')
pd.reset_option('display.max_rows')
```

Data Cleaning and Splitting

In [ ]:
```
df=pd.read_csv('./netflix.csv')
df.sample(10)
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
7165	s7166	Movie	Kajraare	Pooja Bhatt	Himesh Reshammiya, Sara Loren, Amrita Singh, G...	India	October 22, 2017	2010	TV-14	113 min	Dramas, International Movies, Music & Musicals	A popular singer on the run poses as a bartend...
6270	s6271	Movie	Becoming Jane	Julian Jarrold	Anne Hathaway, James McAvoy, Julie Walters, Ja...	United Kingdom, Ireland	January 12, 2019	2007	PG-13	121 min	Dramas, Romantic Movies	A passionate romance with roguish barrister To...
6026	s6027	TV Show	9 Months That Made You	NaN	Demetri Goritsas	United States	March 1, 2017	2016	TV-PG	1 Season	British TV Shows, Docuseries, Science & Nature TV	Witness the wonders of human gestation through...
8735	s8736	Movie	Who's That Knocking at My Door?	Martin Scorsese	Zina Bethune, Harvey Keitel, Anne Collette, Le...	United States	July 1, 2019	1967	R	90 min	Classic Movies, Dramas, Independent Movies	A woman's revelation that she was once raped s...
7541	s7542	Movie	My Little Pony Equestria Girls: Friendship Games	Ishi Rudell, Jayson Thiessen	Tara Strong, Rebecca Shoichet, Ashleigh Ball, ...	United States, Canada	December 1, 2015	2015	TV-Y	72 min	Children & Family Movies, Comedies	Rainbow Dash, Applejack and friends compete ag...
8156	s8157	Movie	Teen Patti	Leena Yadav	Amitabh Bachchan, Madhavan, Ben Kingsley, Shra...	India	March 1, 2018	2010	TV-PG	137 min	Dramas, International Movies, Thrillers	Luck brings together math expert Perci Trachte...
8733	s8734	Movie	White Island	Benjamin Turner	Lyndon Ogbourne, Billy Zane, Billy Boyd, Joel ...	United Kingdom	September 25, 2017	2016	TV-MA	91 min	Comedies, Independent Movies, Thrillers	Returning to Ibiza after several years walking...
7714	s7715	Movie	Patriot Games	Phillip Noyce	Harrison Ford, Anne Archer, Patrick Bergin, Se...	United States	January 1, 2020	1992	R	117 min	Action & Adventure	CIA desk jockey Jack Ryan plunges into the hea...
3976	s3977	TV Show	The Eagle of El-Se'eed	NaN	Mohamed Ramadan, Sayed Rajab, Dorra Zarrouk, D...	NaN	March 25, 2019	2018	TV-14	1 Season	Crime TV Shows, International TV Shows, TV Act...	A police officer and a drug lord become embroi...
4001	s4002	TV Show	Green Door	NaN	Jam Hsiao, Bea Hayden Kuo, Enno Cheng, Hsieh Y...	Taiwan	March 17, 2019	2019	TV-MA	1 Season	International TV Shows, Romantic TV Shows, TV ...	A troubled psychologist returns from the U.S. ...

In [ ]:
```
df = df.drop(["description","title"], axis=1)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   director      6173 non-null   object
 3   cast          7982 non-null   object
 4   country       7976 non-null   object
 5   date_added    8797 non-null   object
 6   release_year  8807 non-null   int64 
 7   rating        8803 non-null   object
 8   duration      8804 non-null   object
 9   listed_in     8807 non-null   object
dtypes: int64(1), object(9)
memory usage: 688.2+ KB
```

In [ ]:
```
df.isna().sum()
```

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

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

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

Out[ ]:
```
rating
TV-MA       3207
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: count, dtype: int64
```

In [ ]:
```
df["type"]=df["type"].astype("category")
mask=df["rating"].isin(["74 min","84 min","66 min"])
df.loc[mask, "duration"] = df.loc[mask, "rating"]
df.loc[mask, "rating"] = df["rating"].mode().iloc[0]
df["rating"]=df["rating"].fillna(df["rating"].mode().iloc[0])
df["date_added"] = pd.to_datetime(df["date_added"] , format='%B %d, %Y', errors="coerce")
mask=df["date_added"].isna()
df.loc[mask, "date_added"] = df.loc[mask, "release_year"].apply(lambda x: max(pd.to_datetime(x + 1, format="%Y"), pd.Timestamp(datetime.date(2006, 1, 1))).date() )
```

In [ ]:
```
# df.set_index('show_id')['cast'].str.split(', ', expand=True).stack().reset_index(name='cast').drop('level_1', axis=1)
```

Separating nested data

In [ ]:
```
cast=df['cast'].apply(lambda x: str(x).split(', ')).tolist()
cast_df=pd.DataFrame(cast,index=df['show_id'])
cast_df=cast_df.stack().reset_index(name='cast').drop('level_1', axis=1).set_index('show_id')
cast_df.replace("nan", float('nan'), inplace=True)
# mask = cast_df[cast_df['cast'] == ''].index
# cast_df.drop(mask, inplace=True)
# cast_df

director=df['director'].apply(lambda x: str(x).split(', ')).tolist()
director_df=pd.DataFrame(director,index=df['show_id'])
director_df=director_df.stack().reset_index(name='director').drop('level_1', axis=1).set_index('show_id')
director_df.replace("nan", float('nan'), inplace=True)
# director_df

country=df['country'].apply(lambda x: str(x).split(', ')).tolist()
country_df=pd.DataFrame(country,index=df['show_id'])
country_df=country_df.stack().reset_index(name='country').drop('level_1', axis=1).set_index('show_id')
country_df.replace("nan", float('nan'), inplace=True)
mask = country_df[country_df['country'] == ''].index
country_df.drop(mask, inplace=True)
# country_df.replace(" ", float('nan'), inplace=True)
# country_df

listed=df['listed_in'].apply(lambda x: str(x).split(', ')).tolist()
listed_df=pd.DataFrame(listed,index=df['show_id'])
listed_df=listed_df.stack().reset_index(name='listed_in').drop('level_1', axis=1).set_index('show_id')
listed_df.replace("nan", float('nan'), inplace=True)
# listed_df
df.drop(["cast","country","director","listed_in"], axis=1, inplace=True)
```

Adding date related columns

In [ ]:
```
df["duration"]=df["duration"].apply(lambda x: x.split(" ")[0])
df["date_added_year_month"] = df["date_added"].dt.strftime('%Y-%m')
df["date_added_year"]=df["date_added"].dt.year
df["date_added_month"]=df["date_added"].dt.month
df["date_added_month_name"]=df["date_added"].dt.month_name()
df["dat_added_period"]=pd.cut(df["date_added_year"], bins=[0,2005, 2010, 2015,2022 ], labels=["2005","2006-2010","2011-2015","2016-2022"])

movies_df=df.loc[df["type"]=="Movie"]
tv_shows_df=df.loc[df["type"]=="TV Show"]
movies_df.head()
tv_shows_df.head()
```

Out[ ]:

	show_id	type	date_added	release_year	rating	duration	date_added_year_month	date_added_year	date_added_month	date_added_month_name	dat_added_period
0	s1	Movie	2021-09-25	2020	PG-13	90	2021-09	2021	9	September	2016-2022
6	s7	Movie	2021-09-24	2021	PG	91	2021-09	2021	9	September	2016-2022
7	s8	Movie	2021-09-24	1993	TV-MA	125	2021-09	2021	9	September	2016-2022
9	s10	Movie	2021-09-24	2021	PG-13	104	2021-09	2021	9	September	2016-2022
12	s13	Movie	2021-09-23	2021	TV-MA	127	2021-09	2021	9	September	2016-2022

Out[ ]:

	show_id	type	date_added	release_year	rating	duration	date_added_year_month	date_added_year	date_added_month	date_added_month_name	dat_added_period
1	s2	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022
2	s3	TV Show	2021-09-24	2021	TV-MA	1	2021-09	2021	9	September	2016-2022
3	s4	TV Show	2021-09-24	2021	TV-MA	1	2021-09	2021	9	September	2016-2022
4	s5	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022
5	s6	TV Show	2021-09-24	2021	TV-MA	1	2021-09	2021	9	September	2016-2022

In [ ]:
```
country_df.value_counts()
cast_df.value_counts()
director_df.value_counts()
```

Out[ ]:
```
country
United States     3689
India             1046
United Kingdom     804
Canada             445
France             392
                  ...
Kazakhstan           1
Jamaica              1
Slovakia             1
Ethiopia             1
Afghanistan          1
Name: count, Length: 126, dtype: int64
====================
```

Out[ ]:
```
cast
Anupam Kher            43
Shah Rukh Khan         35
Julie Tejwani          33
Naseeruddin Shah       32
Takahiro Sakurai       32
                       ..
Chinmay Kambli          1
Kumiko Aso              1
Kumarakom Vasudevan     1
Kumar Varun             1
Ṣọpẹ́ Dìrísù            1
Name: count, Length: 36439, dtype: int64
====================
```

Out[ ]:
```
director
Rajiv Chilaka              22
Jan Suter                  21
Raúl Campos                19
Suhas Kadav                16
Marcus Raboy               16
                           ..
Brandon Camp                1
Juan Antin                  1
Juan Antonio de la Riva     1
Juan Camilo Pinzon          1
María Jose Cuevas           1
Name: count, Length: 4993, dtype: int64
```

In [ ]:
```
country_df["country"].mode().iloc[0]
```

Out[ ]:
```
'United States'
```

In [ ]:
```
director_df.fillna("Unknown", inplace=True)
cast_df.fillna("Unknown", inplace=True)
country_df=country_df.fillna('Unknown')
```

In [ ]:
```
country_df.reset_index(inplace=True)
cast_df.reset_index(inplace=True)
listed_df.reset_index(inplace=True)
director_df.reset_index(inplace=True)
```

In [ ]:
```
country_df.isna().sum()
director_df.isna().sum()
listed_df.isna().sum()
tv_shows_df.isna().sum()
movies_df.isna().sum()
df.isna().sum()
cast_df.isna().sum()
```

Out[ ]:
```
show_id    0
country    0
dtype: int64
```

Out[ ]:
```
show_id     0
director    0
dtype: int64
```

Out[ ]:
```
show_id      0
listed_in    0
dtype: int64
```

Out[ ]:
```
show_id                 0
type                    0
date_added              0
release_year            0
rating                  0
duration                0
date_added_year_month   0
date_added_year         0
date_added_month        0
date_added_month_name   0
dat_added_period        0
dtype: int64
```

Out[ ]:
```
show_id                 0
type                    0
date_added              0
release_year            0
rating                  0
duration                0
date_added_year_month   0
date_added_year         0
date_added_month        0
date_added_month_name   0
dat_added_period        0
dtype: int64
```

Out[ ]:
```
show_id                 0
type                    0
date_added              0
release_year            0
rating                  0
duration                0
date_added_year_month   0
date_added_year         0
date_added_month        0
date_added_month_name   0
dat_added_period        0
dtype: int64
```

Out[ ]:
```
show_id    0
cast       0
dtype: int64
```

Exporting data to separate files

In [ ]:
```
country_df.to_csv("country.csv", index=False)
director_df.to_csv("director.csv", index=False)
cast_df.to_csv("cast.csv", index=False)
listed_df.to_csv("listed.csv", index=False)
df.to_csv("data.csv", index=False)
movies_df.to_csv("movies.csv", index=False)
tv_shows_df.to_csv("tv_shows.csv", index=False)
```

EDA and Insights

In [ ]:
```
country_df=pd.read_csv("./country.csv")
cast_df=pd.read_csv("./cast.csv")
listed_df=pd.read_csv("./listed.csv")
movies_df=pd.read_csv("./movies.csv", parse_dates=["date_added"])
tv_shows_df=pd.read_csv("./tv_shows.csv",parse_dates=["date_added"])
director_df=pd.read_csv("./director.csv")
df=pd.read_csv("./data.csv",parse_dates=["date_added"])
# df["type"]=df["type"].astype("category")
# movies_df["type"]=movies_df["type"].astype("category")
# tv_shows_df["type"]=tv_shows_df["type"].astype("category")
```

In [ ]:
```
df.describe( include="all" )
```

Out[ ]:

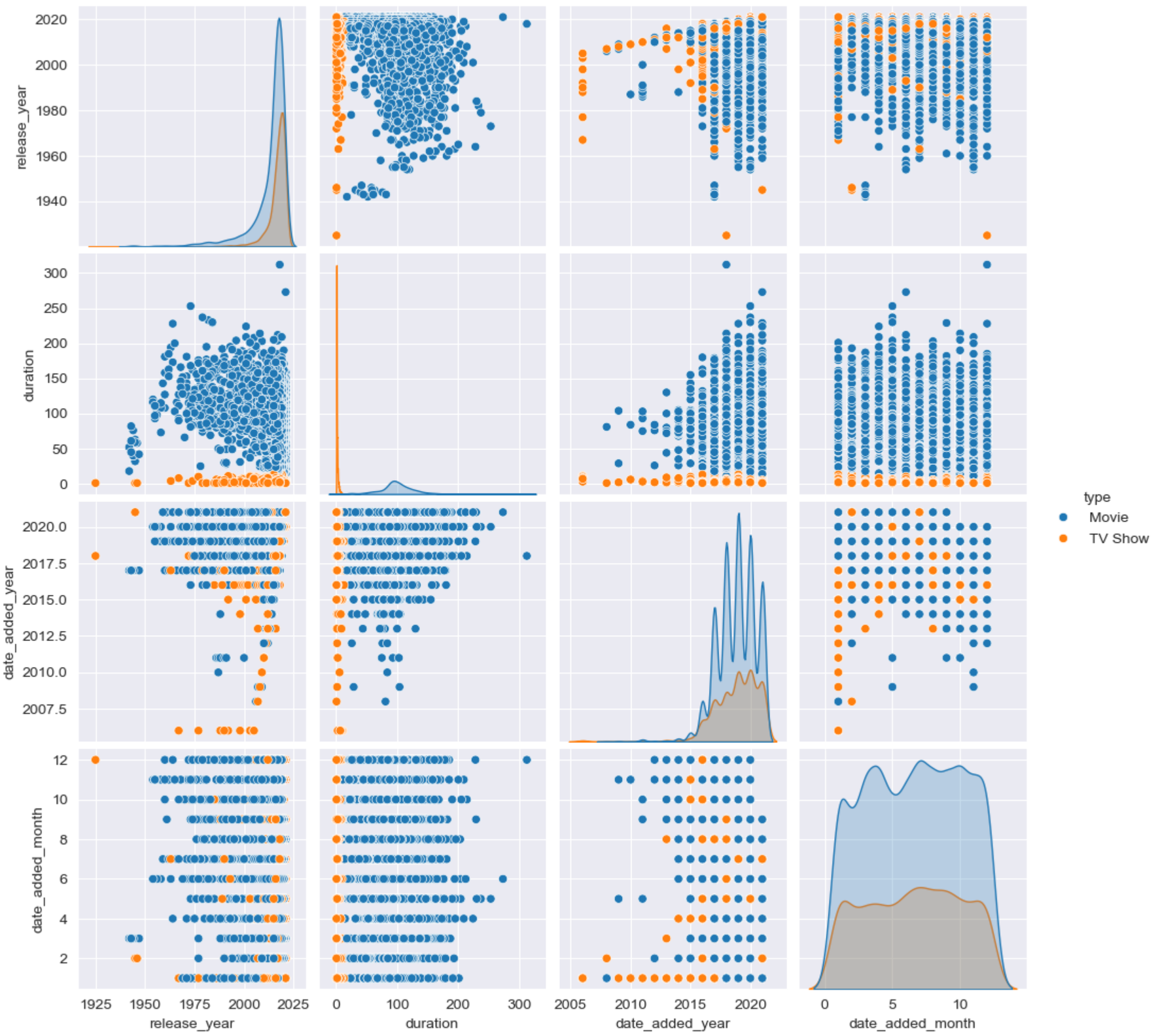
	show_id	type	date_added	release_year	rating	duration	date_added_year_month	date_added_year	date_added_month	date_added_month_name	dat_added_period
count	8807	8807	8807	8807.000000	8807	8807.000000	8807	8807.000000	8807.000000	8807	8807
unique	8807	2	NaN	NaN	14	NaN	114	NaN	NaN	12	3
top	s1	Movie	NaN	NaN	TV-MA	NaN	2021-07	NaN	NaN	January	2016-2022
freq	1	6131	NaN	NaN	3214	NaN	257	NaN	NaN	825	8640
mean	NaN	NaN	2019-05-04 22:43:58.174179584	2014.180198	NaN	69.848530	NaN	2018.843874	6.590439	NaN	NaN
min	NaN	NaN	2006-01-01 00:00:00	1925.000000	NaN	1.000000	NaN	2006.000000	1.000000	NaN	NaN
25%	NaN	NaN	2018-04-01 00:00:00	2013.000000	NaN	2.000000	NaN	2018.000000	4.000000	NaN	NaN
50%	NaN	NaN	2019-07-01 00:00:00	2017.000000	NaN	88.000000	NaN	2019.000000	7.000000	NaN	NaN
75%	NaN	NaN	2020-08-18 00:00:00	2019.000000	NaN	106.000000	NaN	2020.000000	10.000000	NaN	NaN
max	NaN	NaN	2021-09-25 00:00:00	2021.000000	NaN	312.000000	NaN	2021.000000	12.000000	NaN	NaN
std	NaN	NaN	NaN	8.819312	NaN	50.806431	NaN	1.660223	3.463441	NaN	NaN

Insights

- There are 8807 unique shows and movies availble in the dataset.

In [ ]:
```
sns.pairplot(df, hue="type")
```

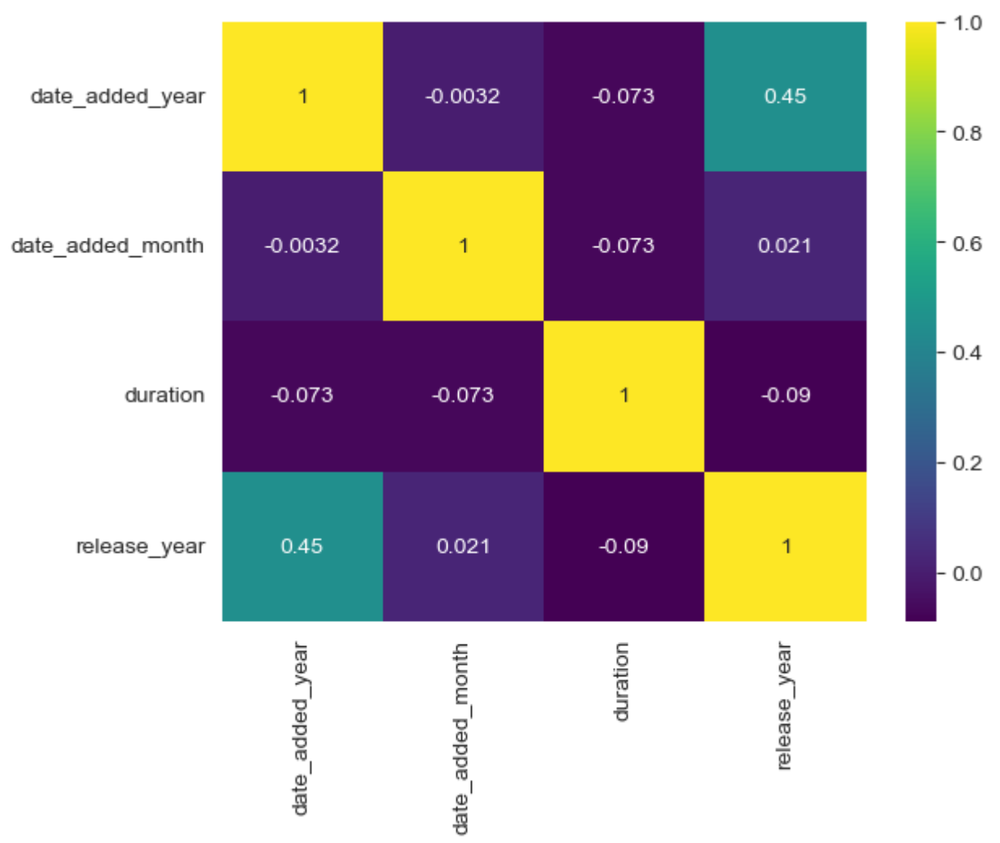
Out[ ]:
```
<seaborn.axisgrid.PairGrid at 0x1b0277210>
```



Insights

- No particular paatern is observed in above plot

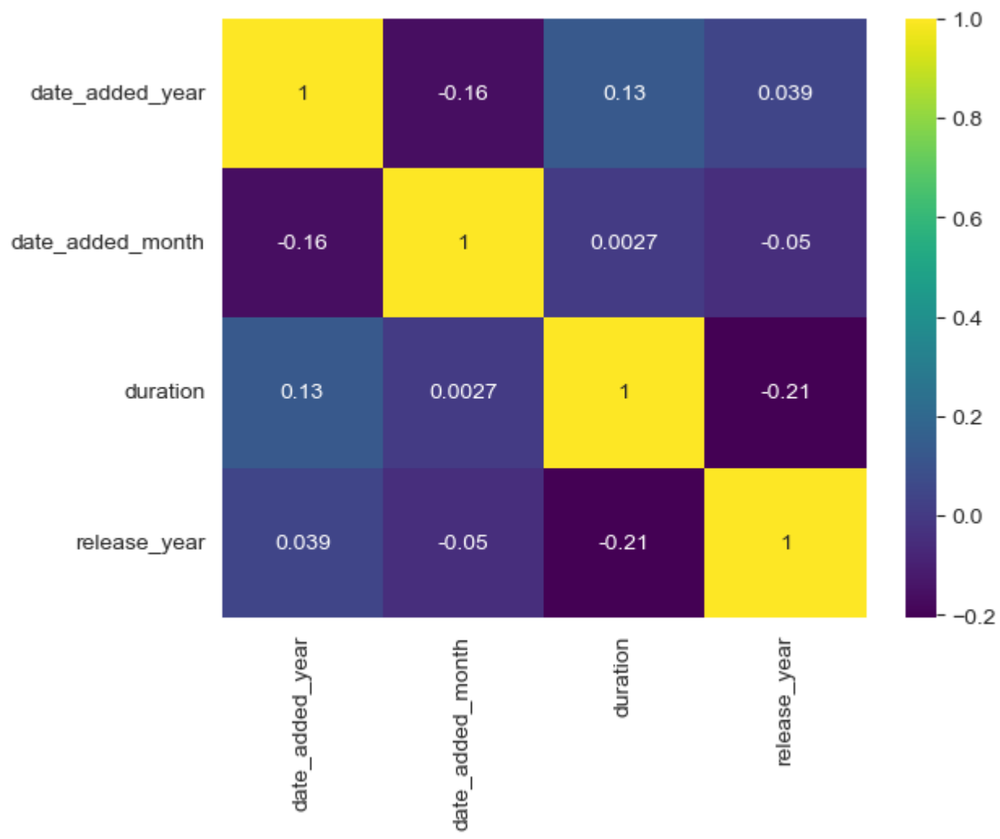
In [ ]:
```
sns.heatmap(tv_shows_df[["date_added_year","date_added_month","duration","release_year"]].corr(), annot=True, cmap="viridis");
```



Insights

- There is no strong correlation between the columns of the tv show data.

In [ ]:
```
sns.heatmap(movies_df[["date_added_year","date_added_month","duration","release_year"]].corr(), annot=True, cmap="viridis");
```



Insights

- There is no strong correlation between the columns of the movie data.

Country Analysis

In [ ]:
```
country_df.describe()
```

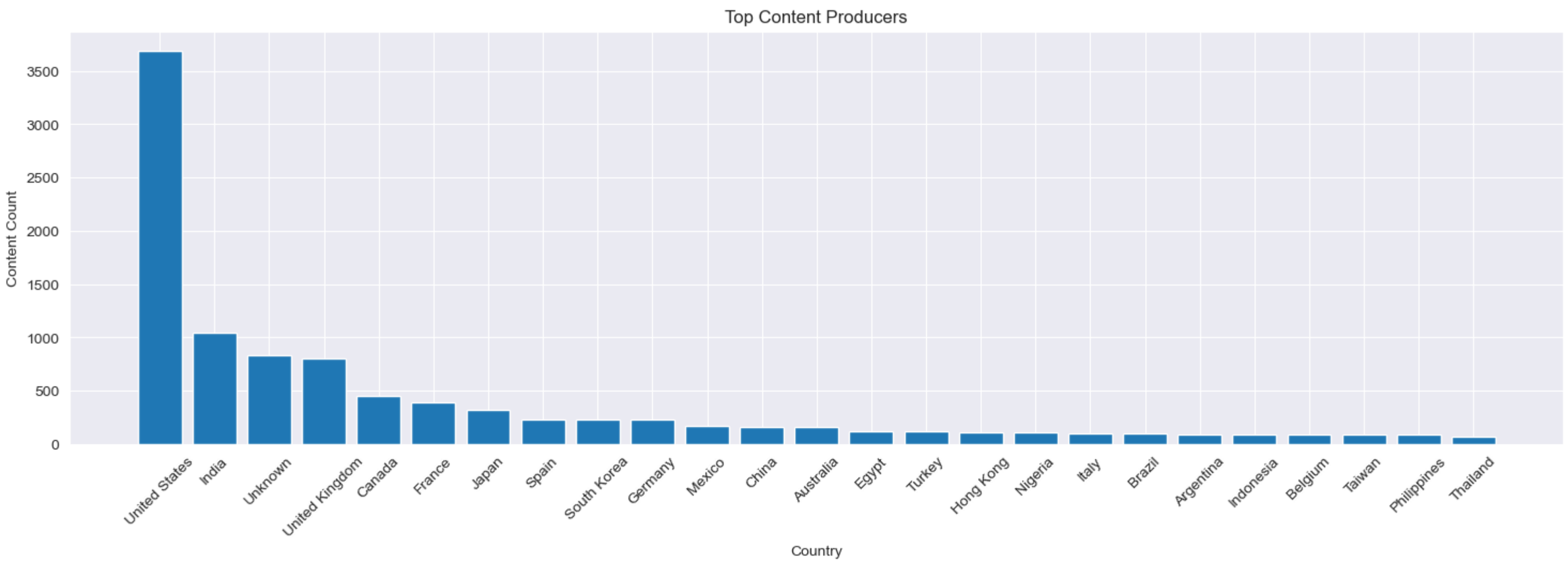
Out[ ]:

	show_id	country
count	10840	10840
unique	8805	127
top	s6234	United States
freq	12	3689

In [ ]:
```
cdf=country_df["country"].value_counts()[:25]
cdf
```

Out[ ]:
```
country
United States     3689
India             1046
Unknown            831
United Kingdom     804
Canada             445
France             392
Japan              318
Spain              232
South Korea        230
Germany            226
Mexico             169
China              162
Australia          160
Egypt              117
Turkey             113
Hong Kong          105
Nigeria            103
Italy              100
Brazil              97
Argentina           91
Indonesia           90
Belgium             90
Taiwan              89
Philippines         83
Thailand            70
Name: count, dtype: int64
```

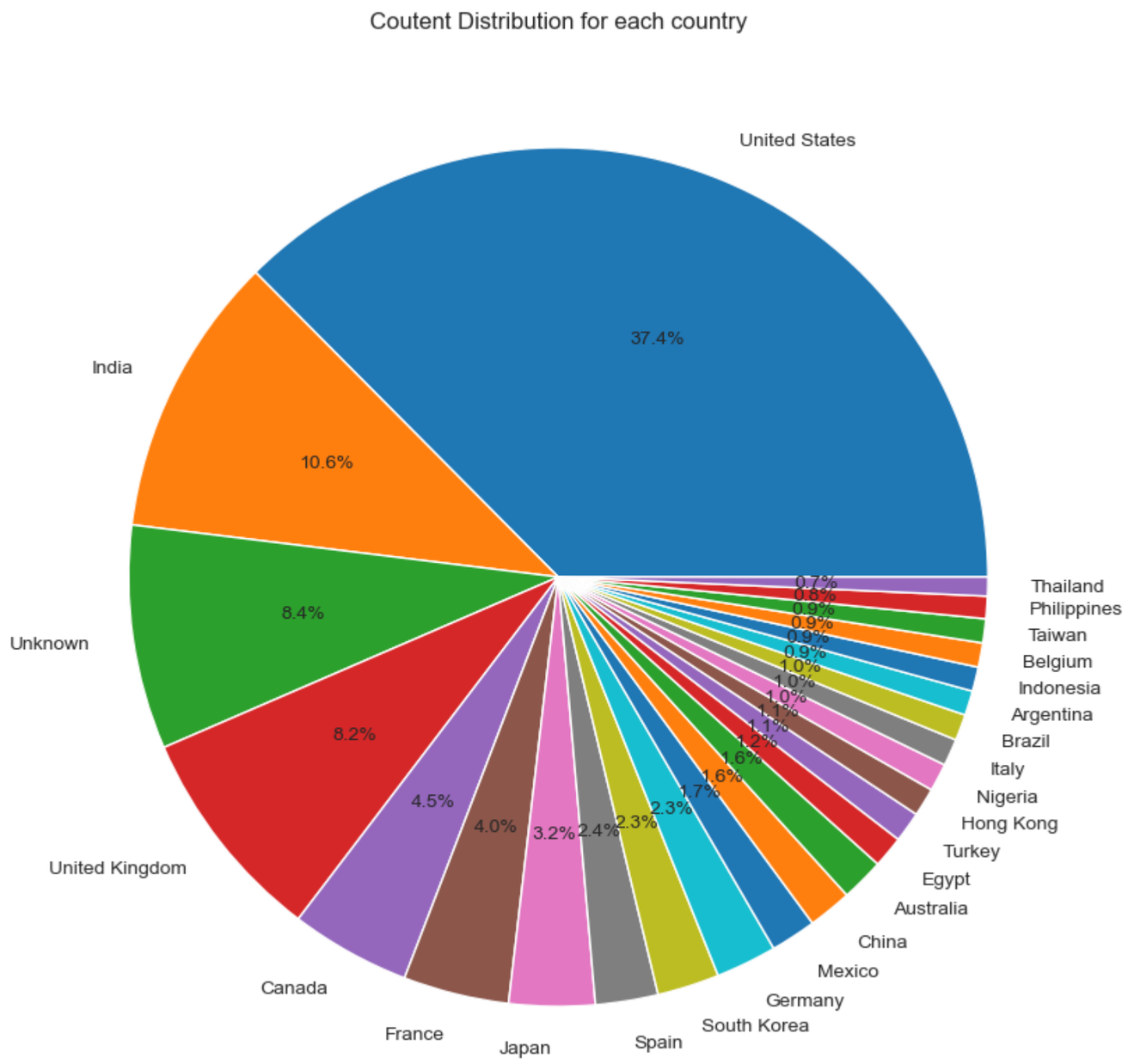
In [ ]:
```
plt.figure(figsize=(18,5))
plt.bar(cdf.index,cdf)
plt.xticks(rotation=45)
plt.ylabel("Content Count")
plt.xlabel("Country")
plt.title("Top Content Producers");
```



Insights

- From above graph we can see that USA is a top content producer.
- Countries like India, UK, Canada, France, Japan, Spain, South Korea and Germany have lot of scope for improvement.
- Countries after Germany have a very high scope for improvement.

In [ ]:
```
plt.figure(figsize=(10,10))
plt.pie(cdf, labels=cdf.index, autopct= '%1.1f%%')
plt.title("Coutent Distribution for each country");
```



Insights

- 37.4% content is produced by USA and 10.6% is produced by India, indicating the popularity of the content available on Netflix
- This is because Hollywood and Bollywood are biggest film industry in the world.

In [ ]:
```
improvement_countries= country_df["country"].value_counts()[1:25].drop(index="Unknown").index.to_list()
improvement_countries
# non_top_3_countrries
```

Out[ ]:
```
['India',
 'United Kingdom',
 'Canada',
 'France',
 'Japan',
 'Spain',
 'South Korea',
 'Germany',
 'Mexico',
 'China',
 'Australia',
 'Egypt',
 'Turkey',
 'Hong Kong',
 'Nigeria',
 'Italy',
 'Brazil',
 'Argentina',
 'Indonesia',
 'Belgium',
 'Taiwan',
 'Philippines',
 'Thailand']
```

Insights

- We will focus on above countries the most as there is a higher chance of growth if we invest in producing content for them

Cast Analysis

In [ ]:
```
cast_df.describe()
```

Out[ ]:

	show_id	cast
count	64951	64951
unique	8807	36440
top	s1855	Unknown
freq	50	825

In [ ]:
```
merge_df=df.merge(cast_df,on='show_id',how='inner')
merge_df.head()
```

Out[ ]:

	show_id	type	date_added	release_year	rating	duration	date_added_year_month	date_added_year	date_added_month	date_added_month_name	dat_added_period	cast
0	s1	Movie	2021-09-25	2020	PG-13	90	2021-09	2021	9	September	2016-2022	Unknown
1	s2	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022	Ama Qamata
2	s2	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022	Khosi Ngema
3	s2	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022	Gail Mabalane
4	s2	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022	Thabang Molaba

In [ ]:
```
cdf = cast_df["cast"].value_counts()[:11].reset_index()
cdf
```

Out[ ]:

	cast	count
0	Unknown	825
1	Anupam Kher	43
2	Shah Rukh Khan	35
3	Julie Tejwani	33
4	Naseeruddin Shah	32
5	Takahiro Sakurai	32
6	Rupa Bhimani	31
7	Akshay Kumar	30
8	Om Puri	30
9	Yuki Kaji	29
10	Amitabh Bachchan	28

Insights

- Above table shows the list of top cast present in movies and tv shows
- There is lot of missing values in this data, which have been replaced by "Unknown"
- Anupam Kher appears to be in maximum number of content present on Netflix

In [ ]:
```
cdf = cdf.iloc[1:11]
mdf=merge_df.loc[merge_df["type"]=="Movie"]["cast"].value_counts()[1:11].reset_index()
tdf=merge_df.loc[merge_df["type"]=="TV Show"]["cast"].value_counts()[1:11].reset_index()
print("Overall Top Cast")
cdf
print("Movie Top Cast")
mdf
print("TV Show Top Cast")
tdf
```

```
Overall Top Cast
```

Out[ ]:

	cast	count
1	Anupam Kher	43
2	Shah Rukh Khan	35
3	Julie Tejwani	33
4	Naseeruddin Shah	32
5	Takahiro Sakurai	32
6	Rupa Bhimani	31
7	Akshay Kumar	30
8	Om Puri	30
9	Yuki Kaji	29
10	Amitabh Bachchan	28

```
Movie Top Cast
```

Out[ ]:

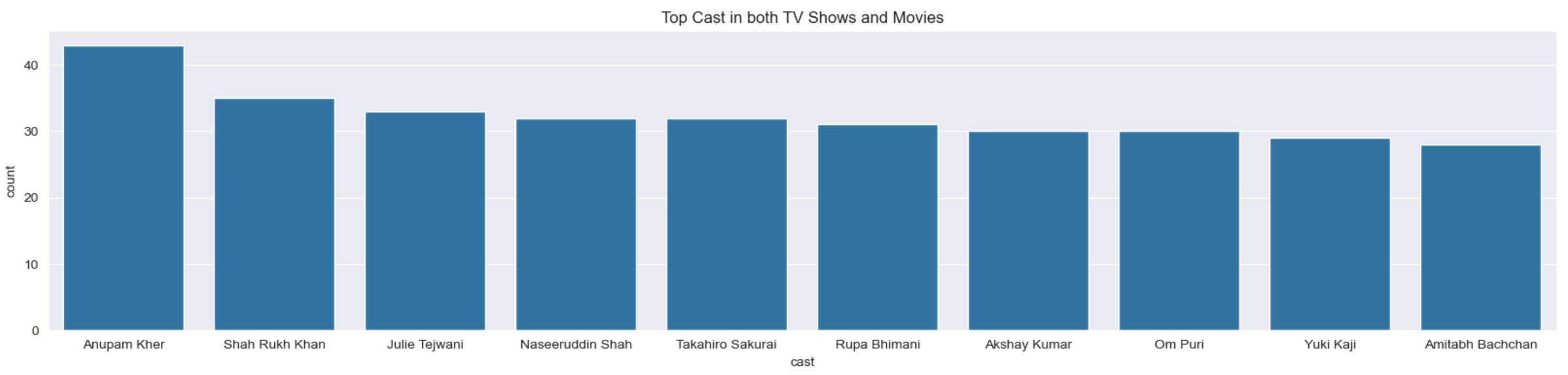
	cast	count
0	Anupam Kher	42
1	Shah Rukh Khan	35
2	Naseeruddin Shah	32
3	Om Puri	30
4	Akshay Kumar	30
5	Paresh Rawal	28
6	Julie Tejwani	28
7	Amitabh Bachchan	28
8	Rupa Bhimani	27
9	Boman Irani	27

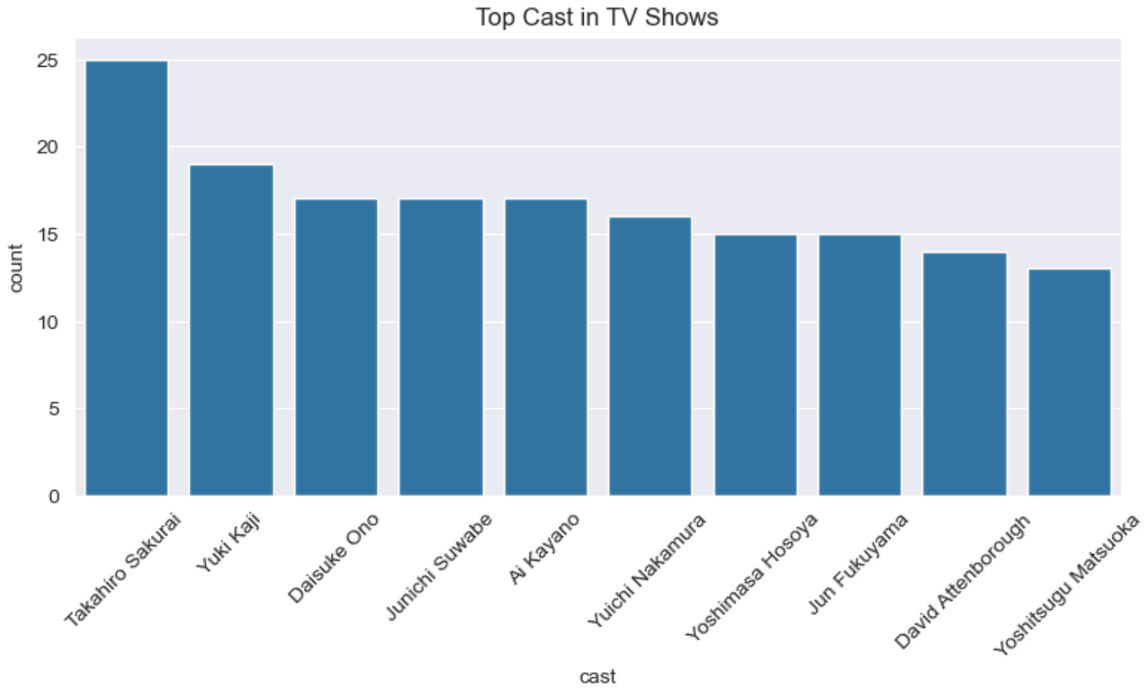
```
TV Show Top Cast
```

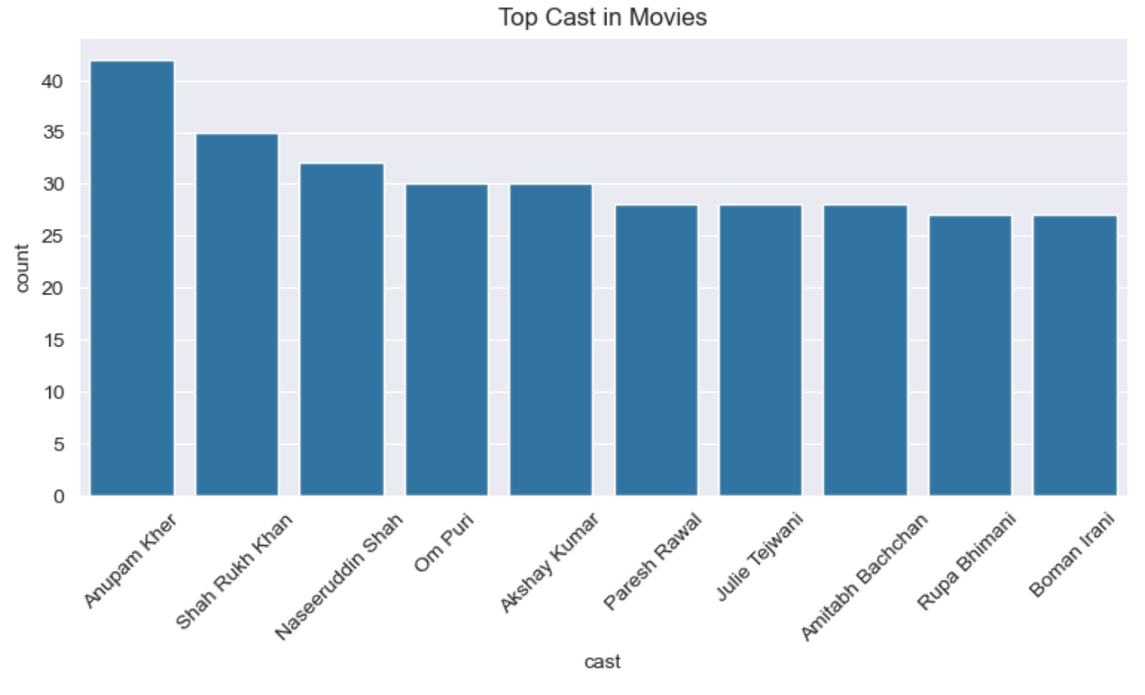
Out[ ]:

	cast	count
0	Takahiro Sakurai	25
1	Yuki Kaji	19
2	Daisuke Ono	17
3	Junichi Suwabe	17
4	Ai Kayano	17
5	Yuichi Nakamura	16
6	Yoshimasa Hosoya	15
7	Jun Fukuyama	15
8	David Attenborough	14
9	Yoshitsugu Matsuoka	13

In [ ]:
```
fig = plt.figure(figsize=(20, 10))
gs = gridspec.GridSpec(2, 2, width_ratios=[1, 1], hspace=0.5)


ax1 = plt.subplot(gs[0, :])
sns.barplot(data=cdf, x="cast", y="count", ax=ax1)
# ax1.tick_params(axis='x', labelrotation=45)
ax1.set_title("Top Cast in both TV Shows and Movies")

ax2 = plt.subplot(gs[1, 0])
sns.barplot(data=tdf, x="cast", y="count", ax=ax2)
ax2.tick_params(axis='x', labelrotation=45)
ax2.set_title("Top Cast in TV Shows")


ax3 = plt.subplot(gs[1, 1])
sns.barplot(data=mdf, x="cast", y="count", ax=ax3)
ax3.set_title("Top Cast in Movies")
ax3.tick_params(axis='x', labelrotation=45);
```







Insights

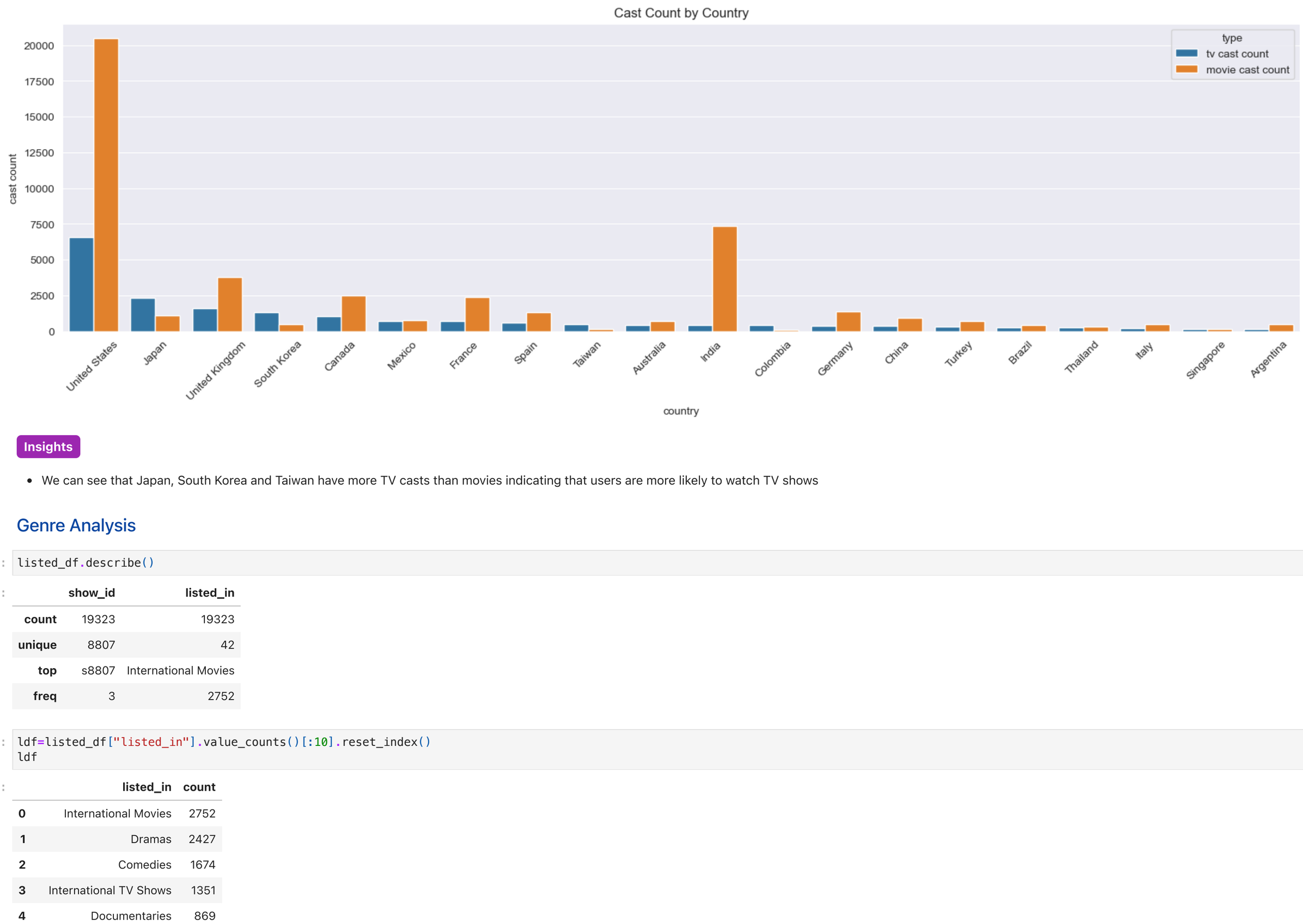
- Above visuals shows the list of top cast present in movies and tv shows
- There appears to be lot of Indian actors in the movies section
- Takahiro Sakurai is the most popular actor in the TV show section
- Anupam Kher is the most popular actor in the movies section and overall section

In [ ]:
```
cast_count_df=duckdb.sql("""
           with cte as (
            select distinct show_id, type,country, cast_df.cast from df
            join cast_df using(show_id)
            join country_df using(show_id)
            where country != 'Unknown' and cast_df.cast != 'Unknown'
           )
           select country, sum(case when type ='TV Show' then 1 else 0 end) "tv cast count",
           sum(case when type ='Movie' then 1 else 0 end) "movie cast count" from cte
           group by country order by "tv cast count" desc
""").df()
cast_count_df.head(20)
```

Out[ ]:

	country	tv cast count	movie cast count
0	United States	6578.0	20484.0
1	Japan	2329.0	1124.0
2	United Kingdom	1572.0	3751.0
3	South Korea	1306.0	481.0
4	Canada	1033.0	2486.0
5	Mexico	710.0	734.0
6	France	685.0	2408.0
7	Spain	600.0	1303.0
8	Taiwan	508.0	131.0
9	Australia	450.0	707.0
10	India	412.0	7338.0
11	Colombia	401.0	108.0
12	Germany	357.0	1403.0
13	China	346.0	913.0
14	Turkey	291.0	697.0
15	Brazil	280.0	403.0
16	Thailand	272.0	305.0
17	Italy	204.0	485.0
18	Singapore	157.0	127.0
19	Argentina	154.0	457.0

In [ ]:
```
temp_df=cast_count_df.head(20).melt(id_vars=['country'],var_name="type", value_name='cast count')
plt.figure(figsize=(20,5))
lp=sns.barplot(data=temp_df, x="country", y="cast count", hue="type")
plt.xticks(rotation=45)
plt.title("Cast Count by Country");
```



Insights

- We can see that Japan, South Korea and Taiwan have more TV casts than movies indicating that users are more likely to watch TV shows

Genre Analysis

In [ ]:
```
listed_df.describe()
```

Out[ ]:

	show_id	listed_in
count	19323	19323
unique	8807	42
top	s8807	International Movies
freq	3	2752

In [ ]:
```
ldf=listed_df["listed_in"].value_counts()[:10].reset_index()
ldf
```

Out[ ]:

	listed_in	count
0	International Movies	2752
1	Dramas	2427
2	Comedies	1674
3	International TV Shows	1351
4	Documentaries	869
5	Action & Adventure	859
6	TV Dramas	763
7	Independent Movies	756
8	Children & Family Movies	641
9	Romantic Movies	616

In [ ]:
```
merge_df=df.merge(listed_df,on='show_id',how='inner')
merge_df.head()
```

Out[ ]:

	show_id	type	date_added	release_year	rating	duration	date_added_year_month	date_added_year	date_added_month	date_added_month_name	dat_added_period	listed_in
0	s1	Movie	2021-09-25	2020	PG-13	90	2021-09	2021	9	September	2016-2022	Documentaries
1	s2	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022	International TV Shows
2	s2	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022	TV Dramas
3	s2	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022	TV Mysteries
4	s3	TV Show	2021-09-24	2021	TV-MA	1	2021-09	2021	9	September	2016-2022	Crime TV Shows

In [ ]:
```
mdf=merge_df.loc[merge_df["type"]=="Movie"]["listed_in"].value_counts()[:10].reset_index()
tdf=merge_df.loc[merge_df["type"]=="TV Show"]["listed_in"].value_counts()[:10].reset_index()
```

```
ldf
mdf
tdf
```

Out[ ]:

	listed_in	count
0	International Movies	2752
1	Dramas	2427
2	Comedies	1674
3	International TV Shows	1351
4	Documentaries	869
5	Action & Adventure	859
6	TV Dramas	763
7	Independent Movies	756
8	Children & Family Movies	641
9	Romantic Movies	616

Out[ ]:

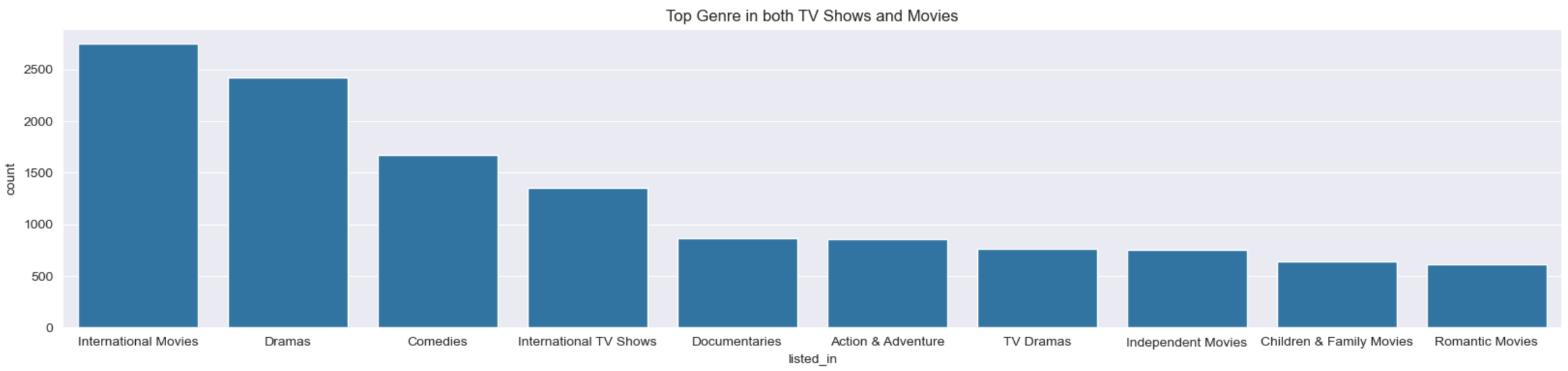
	listed_in	count
0	International Movies	2752
1	Dramas	2427
2	Comedies	1674
3	Documentaries	869
4	Action & Adventure	859
5	Independent Movies	756
6	Children & Family Movies	641
7	Romantic Movies	616
8	Thrillers	577
9	Music & Musicals	375

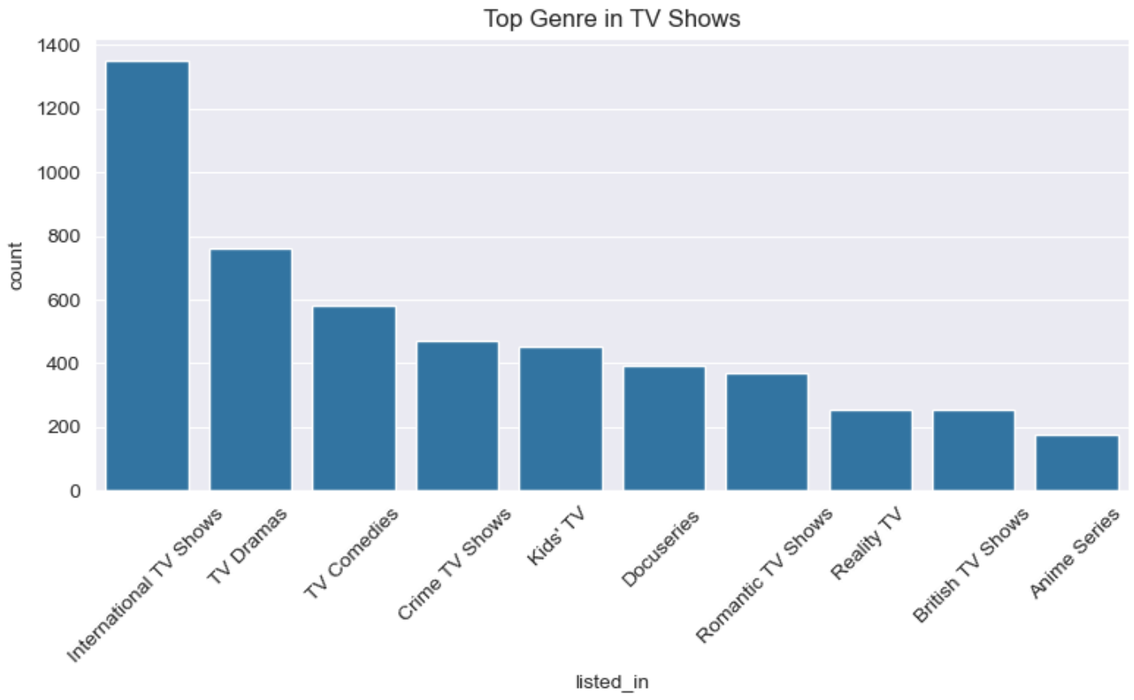
Out[ ]:

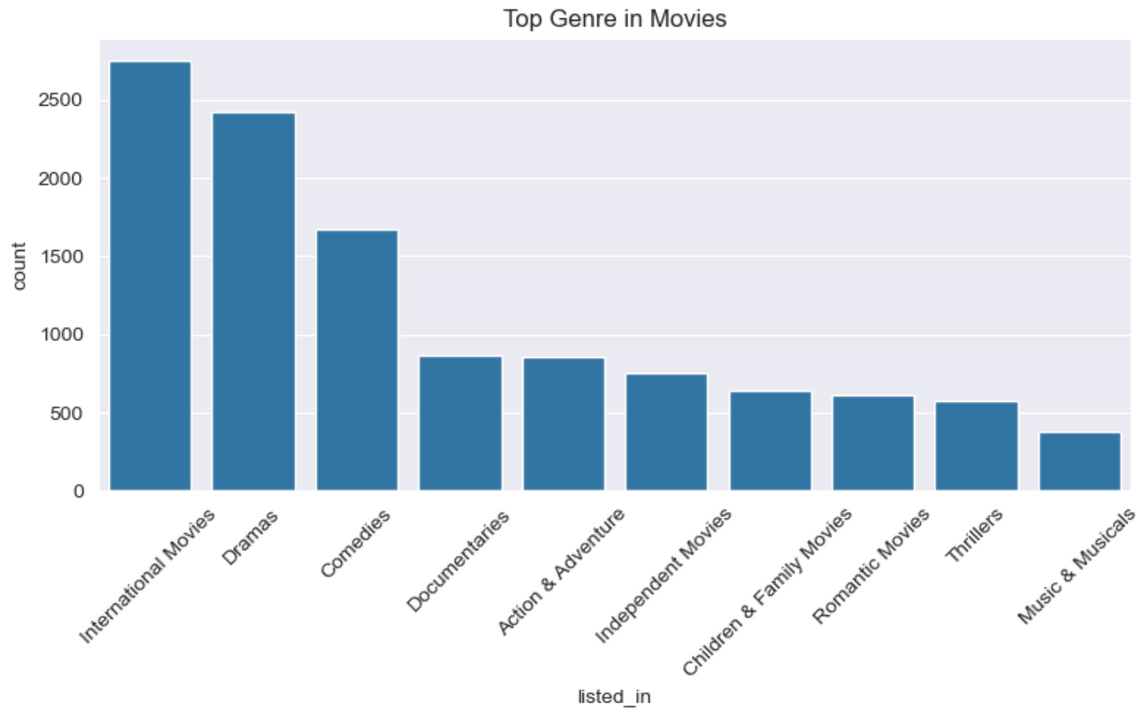
	listed_in	count
0	International TV Shows	1351
1	TV Dramas	763
2	TV Comedies	581
3	Crime TV Shows	470
4	Kids' TV	451
5	Docuseries	395
6	Romantic TV Shows	370
7	Reality TV	255
8	British TV Shows	253
9	Anime Series	176

In [ ]:

```
fig = plt.figure(figsize=(20, 10))
gs = gridspec.GridSpec(2, 2, width_ratios=[1, 1], hspace=0.5)


ax1 = plt.subplot(gs[0, :])
sns.barplot(data=ldf, x="listed_in", y="count", ax=ax1)
# ax1.tick_params(axis='x', labelrotation=45)
ax1.set_title("Top Genre in both TV Shows and Movies")

ax2 = plt.subplot(gs[1, 0])
sns.barplot(data=tdf, x="listed_in", y="count", ax=ax2)
ax2.tick_params(axis='x', labelrotation=45)
ax2.set_title("Top Genre in TV Shows")


ax3 = plt.subplot(gs[1, 1])
sns.barplot(data=mdf, x="listed_in", y="count", ax=ax3)
ax3.set_title("Top Genre in Movies")
ax3.tick_params(axis='x', labelrotation=45);
```





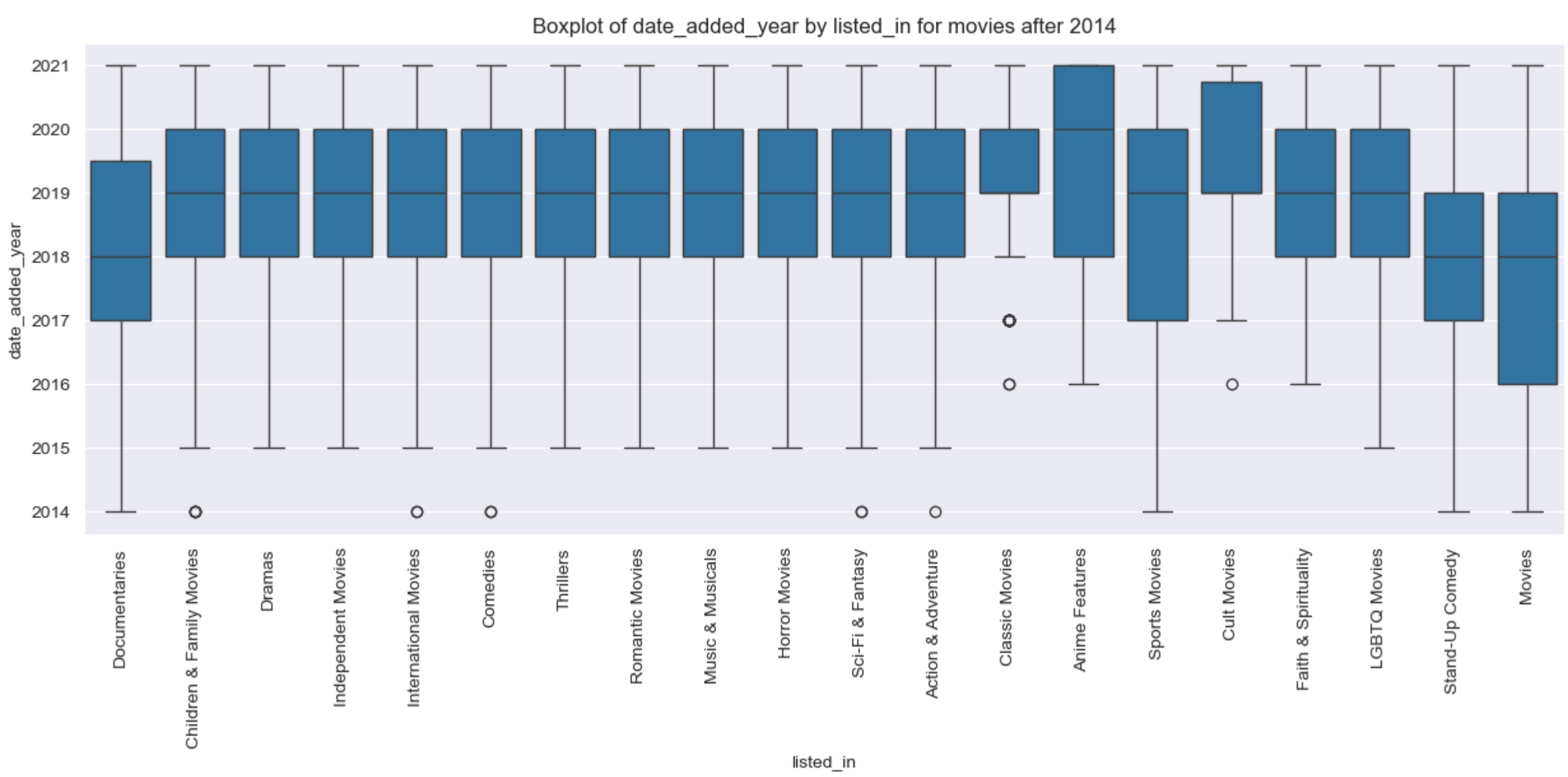


Insights

- Above graphs shows the list of top genres present in movies and tv show category
- It appears that users like watching international movies/tv shows, dramas and comedies the most
- This indicates there is high demand for International content in both TV and movies sectiion

Movie Genre Analysis

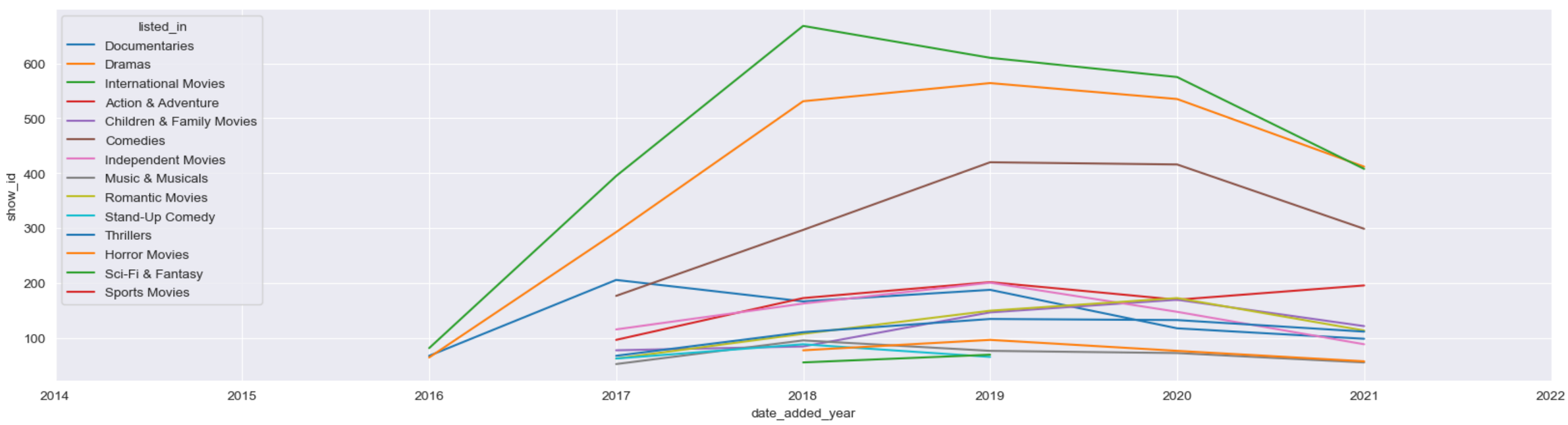
In [ ]:
```
merge_df=movies_df.merge(listed_df, on='show_id', how='inner')
merge_df_trunc = merge_df.loc[merge_df['date_added_year'] >= 2014]
plt.figure(figsize=(15, 5))
sns.boxplot(data=merge_df_trunc, x="listed_in", y="date_added_year")
plt.title("Boxplot of date_added_year by listed_in for movies after 2014")
plt.xticks(rotation=90);
```



Insights

- Majority of the content has added during 2018 - 2020

In [ ]:
```
temp_df=merge_df.groupby(["date_added_year","listed_in"])["show_id"].count().reset_index()
temp_df=temp_df.loc[temp_df["show_id"]>50]
plt.figure(figsize=(20,5))
sns.lineplot(data=temp_df, y="show_id", x="date_added_year", hue="listed_in", palette="tab10" )
plt.xlim((2014,2022));
```

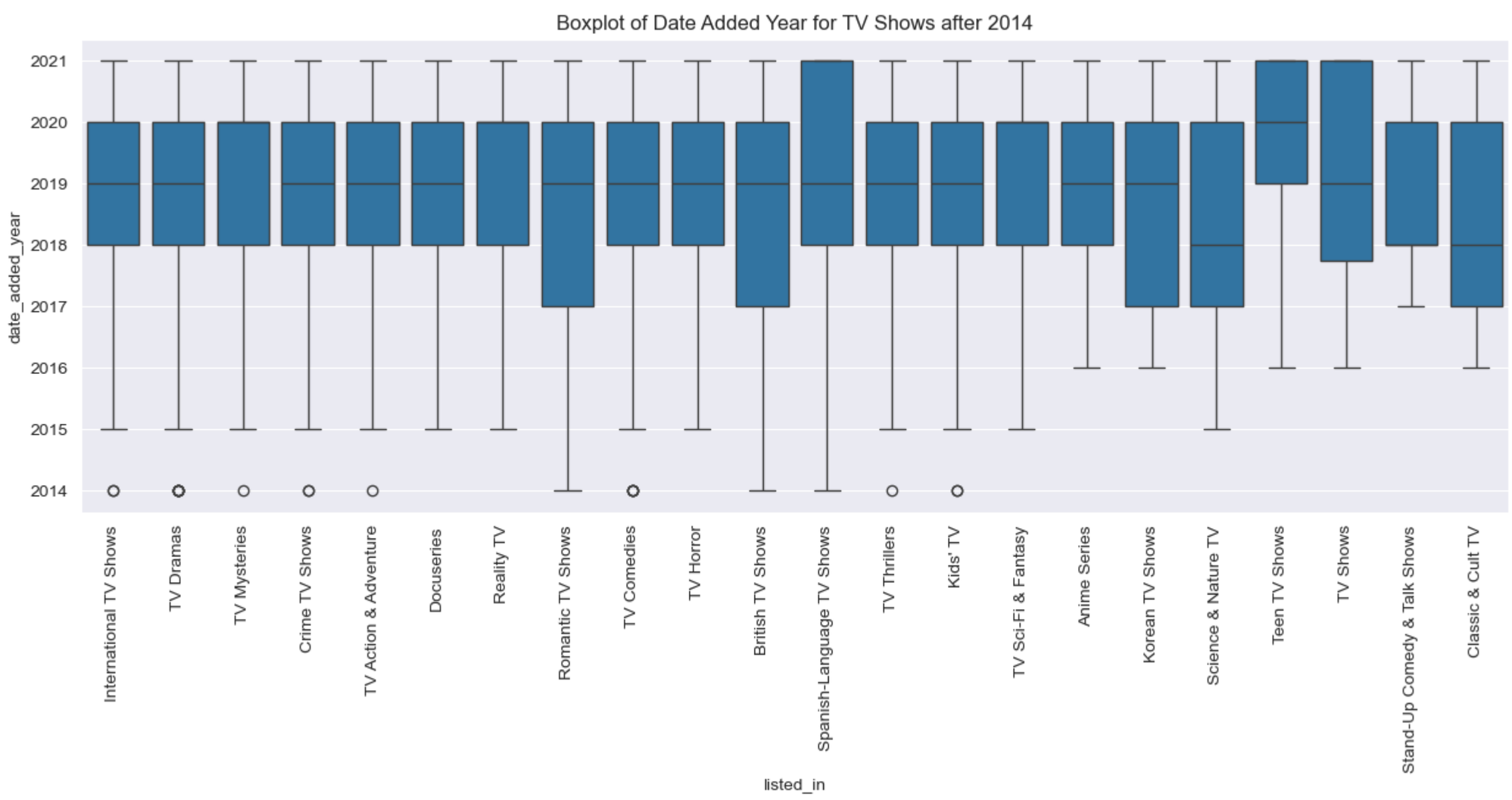


Insights

- International movies, Dramas and Comedies have been added the most over last few years indicating popularity of those genre in movie section

TV Show Genre Analysis

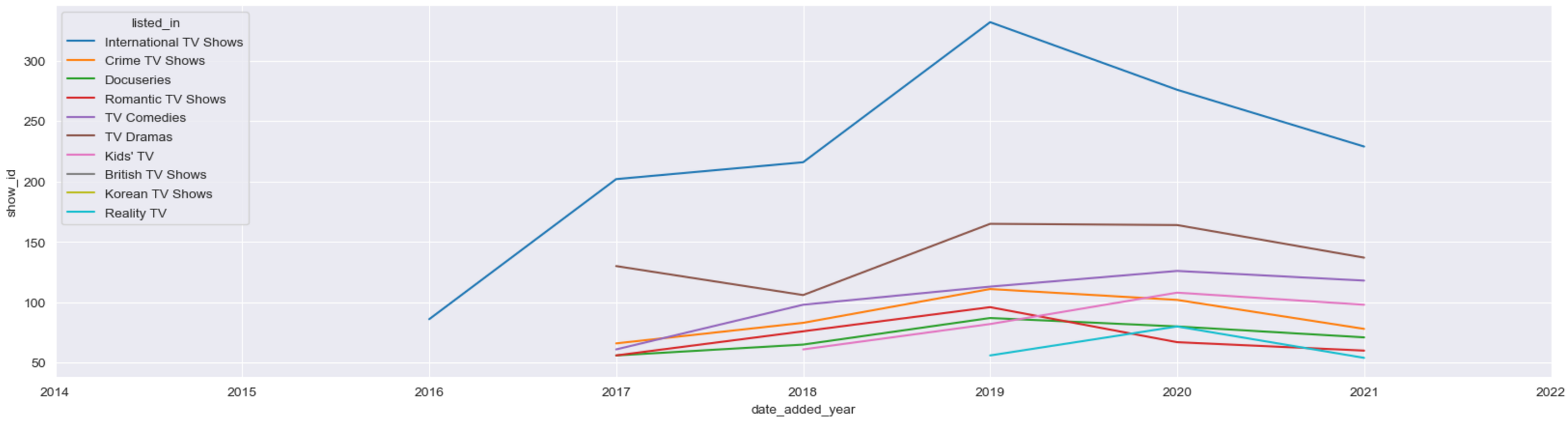
In [ ]:
```
merge_df=tv_shows_df.merge(listed_df, on='show_id', how='inner')
merge_df_trunc = merge_df.loc[merge_df['date_added_year'] >= 2014]
plt.figure(figsize=(15, 5))
sns.boxplot(data=merge_df_trunc, x="listed_in", y="date_added_year")
plt.title('Boxplot of Date Added Year for TV Shows after 2014')
plt.xticks(rotation=90);
```



Insights

- Majority of the content has added during 2018 - 2020 indicating Netflix popularity.

In [ ]:
```
temp_df=merge_df.groupby(["date_added_year","listed_in"])["show_id"].count().reset_index()
temp_df=temp_df.loc[temp_df["show_id"]>50]
plt.figure(figsize=(20,5))
sns.lineplot(data=temp_df, y="show_id", x="date_added_year", hue="listed_in", palette="tab10" )
plt.xlim((2014,2022));
```



Insights

- International TV shows have been added the most over last few years indicating popularity of this genre in TV show section

Director Analysis

In [ ]:
```
director_df.describe()
```

Out[ ]:

	show_id	director
count	9612	9612
unique	8807	4994
top	s5888	Unknown
freq	13	2634

In [ ]:
```
merge_df=df.merge(director_df,on='show_id',how='inner')
merge_df.head()
```

Out[ ]:

	show_id	type	date_added	release_year	rating	duration	date_added_year_month	date_added_year	date_added_month	date_added_month_name	dat_added_period	director
0	s1	Movie	2021-09-25	2020	PG-13	90	2021-09	2021	9	September	2016-2022	Kirsten Johnson
1	s2	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022	Unknown
2	s3	TV Show	2021-09-24	2021	TV-MA	1	2021-09	2021	9	September	2016-2022	Julien Leclercq
3	s4	TV Show	2021-09-24	2021	TV-MA	1	2021-09	2021	9	September	2016-2022	Unknown
4	s5	TV Show	2021-09-24	2021	TV-MA	2	2021-09	2021	9	September	2016-2022	Unknown

In [ ]:
```
ddf = director_df["director"].value_counts()[:11].reset_index()
ddf
```

Out[ ]:

	director	count
0	Unknown	2634
1	Rajiv Chilaka	22
2	Jan Suter	21
3	Raúl Campos	19
4	Suhas Kadav	16
5	Marcus Raboy	16
6	Jay Karas	15
7	Cathy Garcia-Molina	13
8	Martin Scorsese	12
9	Jay Chapman	12
10	Youssef Chahine	12

Insights

- Above table shows the list of top directors present in movies and tv shows
- There is lot of missing values in this data, which have been replaced by "Unknown"

In [ ]:
```
ddf = ddf.iloc[1:11]
mdf=merge_df.loc[merge_df["type"]=="Movie"]["director"].value_counts()[1:11].reset_index()
tdf=merge_df.loc[merge_df["type"]=="TV Show"]["director"].value_counts()[1:11].reset_index()
ddf
mdf
tdf
```

Out[ ]:

	director	count
1	Rajiv Chilaka	22
2	Jan Suter	21
3	Raúl Campos	19
4	Suhas Kadav	16
5	Marcus Raboy	16
6	Jay Karas	15
7	Cathy Garcia-Molina	13
8	Martin Scorsese	12
9	Jay Chapman	12
10	Youssef Chahine	12

Out[ ]:

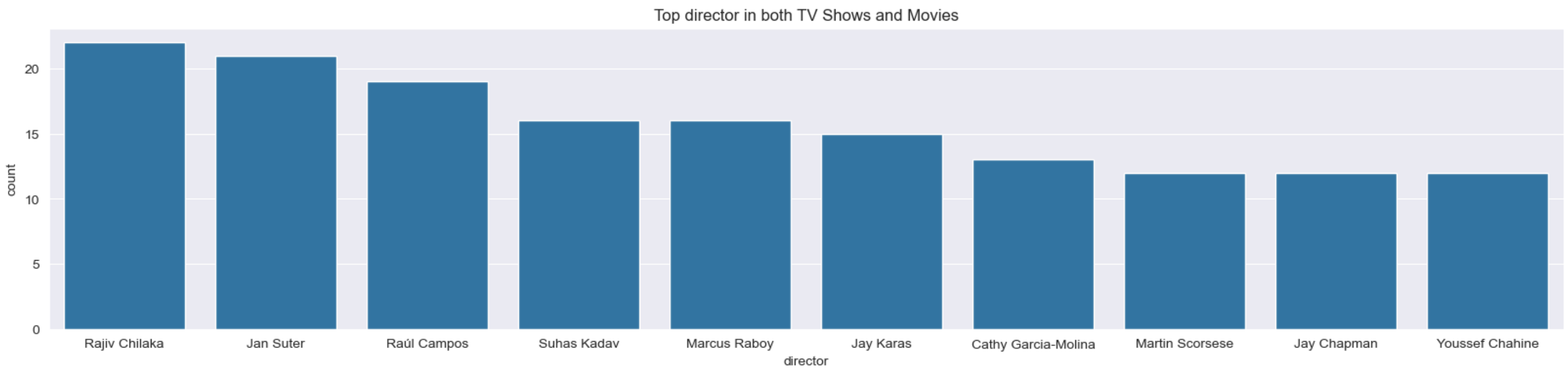
	director	count
0	Rajiv Chilaka	22
1	Jan Suter	21
2	Raúl Campos	19
3	Suhas Kadav	16
4	Jay Karas	15
5	Marcus Raboy	15
6	Cathy Garcia-Molina	13
7	Youssef Chahine	12
8	Martin Scorsese	12
9	Jay Chapman	12

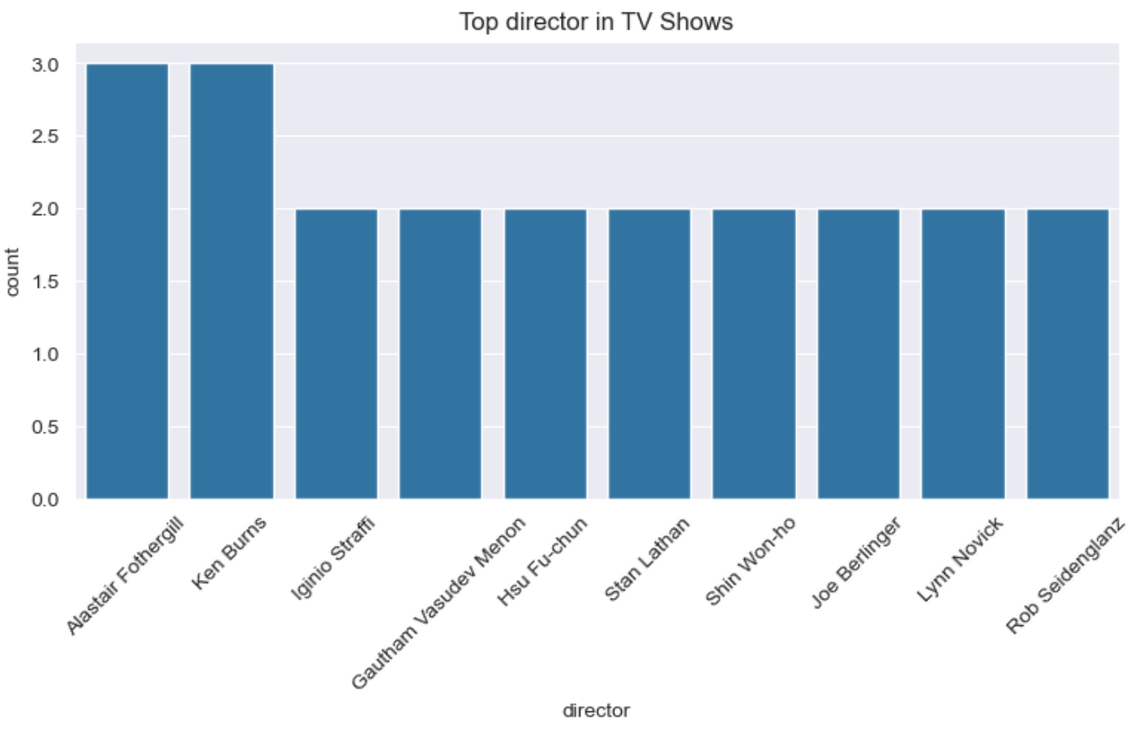
Out[ ]:

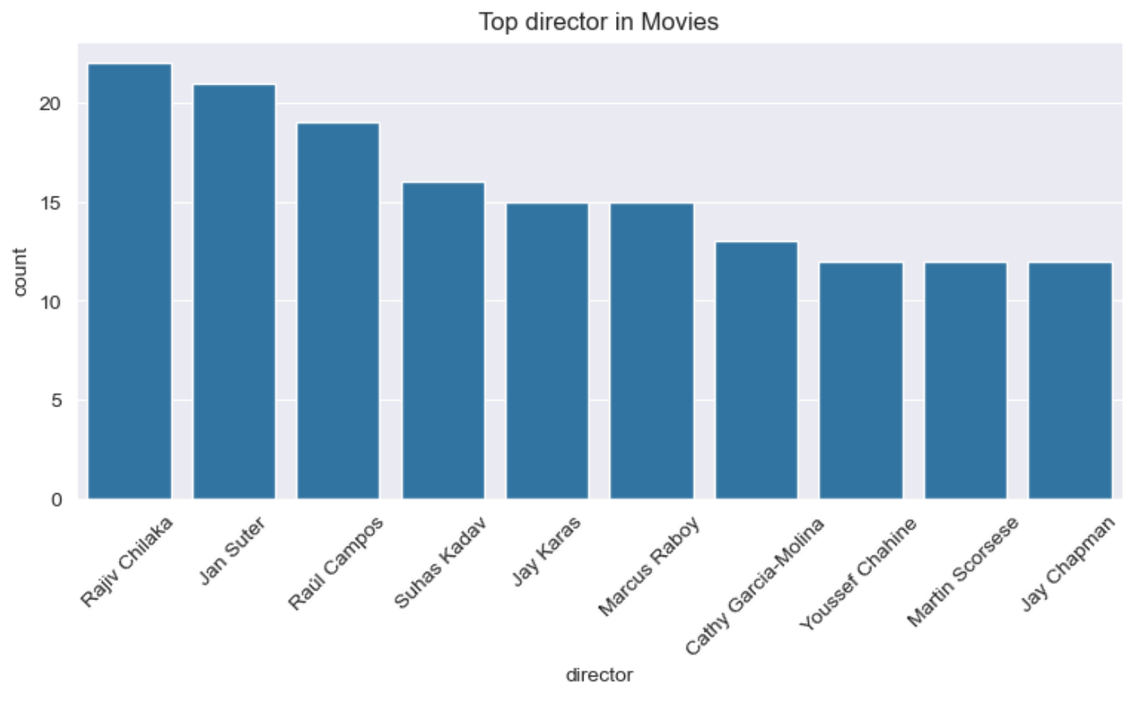
	director	count
0	Alastair Fothergill	3
1	Ken Burns	3
2	Iginio Straffi	2
3	Gautham Vasudev Menon	2
4	Hsu Fu-chun	2
5	Stan Lathan	2
6	Shin Won-ho	2
7	Joe Berlinger	2
8	Lynn Novick	2
9	Rob Seidenglanz	2

In [ ]:
```
fig = plt.figure(figsize=(20, 10))
gs = gridspec.GridSpec(2, 2, width_ratios=[1, 1], hspace=0.5)


ax1 = plt.subplot(gs[0, :])
sns.barplot(data=ddf, x="director", y="count", ax=ax1)
ax1.set_title("Top director in both TV Shows and Movies")

ax2 = plt.subplot(gs[1, 0])
```

```
sns.barplot(data=tdf, x="director", y="count", ax=ax2)
ax2.tick_params(axis='x', labelrotation=45)
ax2.set_title("Top director in TV Shows")


ax3 = plt.subplot(gs[1, 1])
sns.barplot(data=mdf, x="director", y="count", ax=ax3)
ax3.set_title("Top director in Movies")
ax3.tick_params(axis='x', labelrotation=45);
```







Insights

- Above graphs shows the list of top directors present in movies and tv shows

Release Timeline Analysis

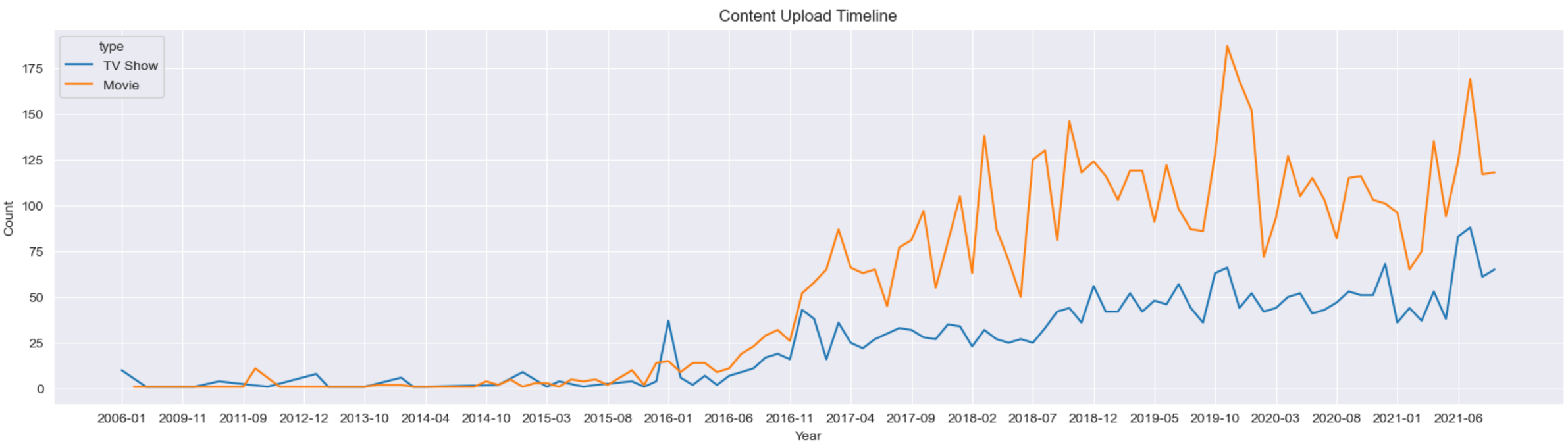
Analysis of all time data

In [ ]:
```
stream_timeline=df.groupby(["date_added_year_month","type"])["show_id"].count().reset_index().set_index("date_added_year_month")
stream_timeline.head()
```

Out[ ]:

date_added_year_month	type	show_id
2006-01	TV Show	10
2008-01	Movie	1
2008-02	TV Show	1
2009-01	TV Show	1
2009-05	Movie	1

In [ ]:
```
plt.figure(figsize=(20,5))
# fig, ax = plt.subplots(figsize=(20, 5))
l=sns.lineplot(data=stream_timeline, x="date_added_year_month", y="show_id", hue="type")
l.set(xlabel="Year", ylabel="Count");
l.set_title("Content Upload Timeline");
l.set_xticks(l.get_xticks()[::5]);
# plt.gca().xaxis.set_major_locator(plt.MultipleLocator(10))
# plt.gca().xaxis.set_minor_locator(plt.MultipleLocator(1))
```



Insights

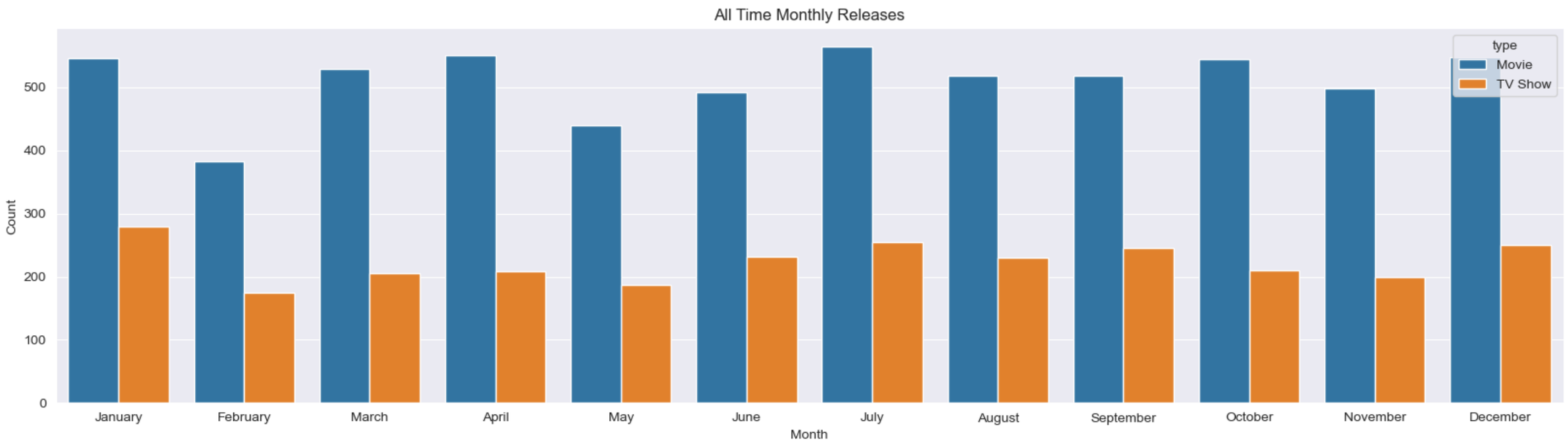
- From above graph, we can see that majority of content got added after 2015.
- Over the years, the amount of tv shows added were comparatively less than movies.
- This shows that movies are more popular than tv shows.

In [ ]:
```
monthly_release=df.groupby(["date_added_month","date_added_month_name","type"])["show_id"].count().reset_index().set_index("date_added_month")
monthly_release.head()
```

Out[ ]:

date_added_month	date_added_month_name	type	show_id
1	January	Movie	546
1	January	TV Show	279
2	February	Movie	382
2	February	TV Show	175
3	March	Movie	529

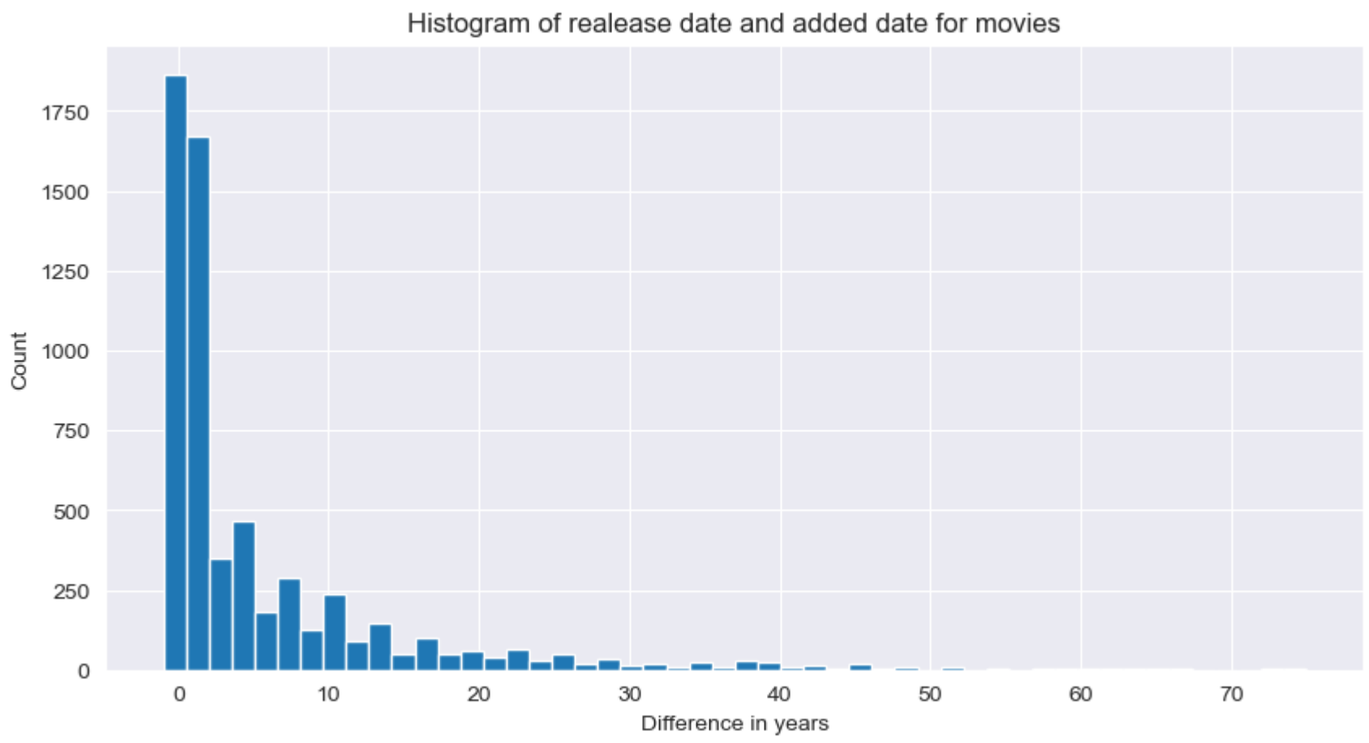
In [ ]:
```
plt.figure(figsize=(20,5))

lp=sns.barplot(data=monthly_release, x="date_added_month_name", y="show_id", hue="type")
lp.set(xlabel="Month", ylabel="Count");
lp.set_title("All Time Monthly Releases");
```



Insights

- From above plot we can see that January, April, July and December months has highest number of content added.
- This seems to be ideal time to release content

In [ ]:
```
plt.figure(figsize=(10,5))
(movies_df["date_added_year"] - movies_df["release_year"]).hist(bins=50)
plt.title("Histogram of realease date and added date for movies")
plt.xlabel("Difference in years")
plt.ylabel("Count");
```



Insights

- Majority of the movies are added under 1 year after the release date.
- This shows that people are eager to watch the movie after its theatrical release date.
- Netflix should produce more original content

Analysis of latest data

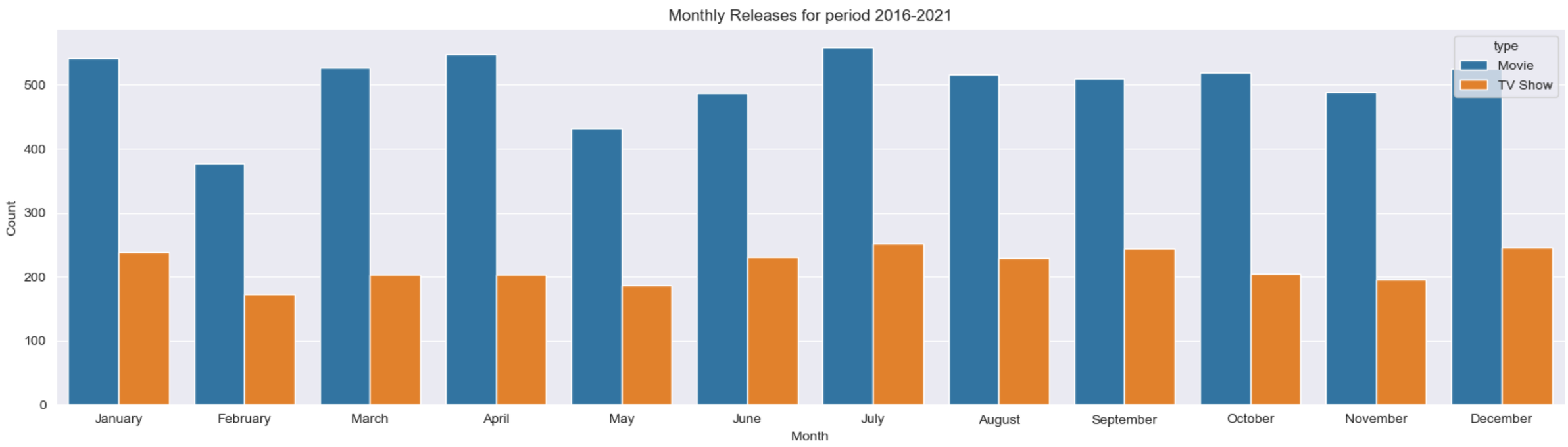
In [ ]:
```
latest_data=df.loc[df["dat_added_period"].isin(["2016-2022"])]
```

In [ ]:
```
monthly_release_latest=latest_data.groupby(["date_added_month","date_added_month_name","type"])["show_id"].count().reset_index().set_index("date_added_month")
monthly_release.head()
```

Out[ ]:

date_added_month	date_added_month_name	type	show_id
1	January	Movie	546
1	January	TV Show	279
2	February	Movie	382
2	February	TV Show	175
3	March	Movie	529

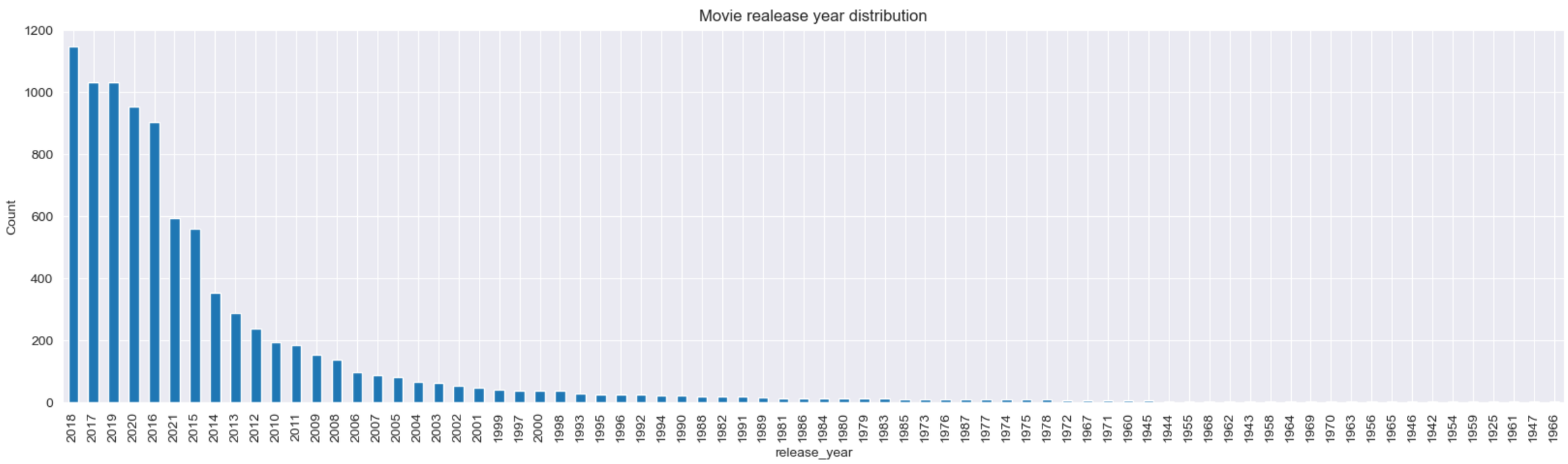
In [ ]:
```
plt.figure(figsize=(20,5))

lp=sns.barplot(data=monthly_release_latest, x="date_added_month_name", y="show_id", hue="type")
lp.set(xlabel="Month", ylabel="Count");
lp.set_title("Monthly Releases for period 2016-2021");
```



Insights

- From above plot we can see that January, April, July and December months has highest number of content added even for recent data.
- This seems to be ideal time to release content

In [ ]:
```
plt.figure(figsize=(20,5))
df["release_year"].value_counts().plot(kind="bar");
plt.ylabel("Count");
plt.title("Movie realease year distribution");
```



Insights

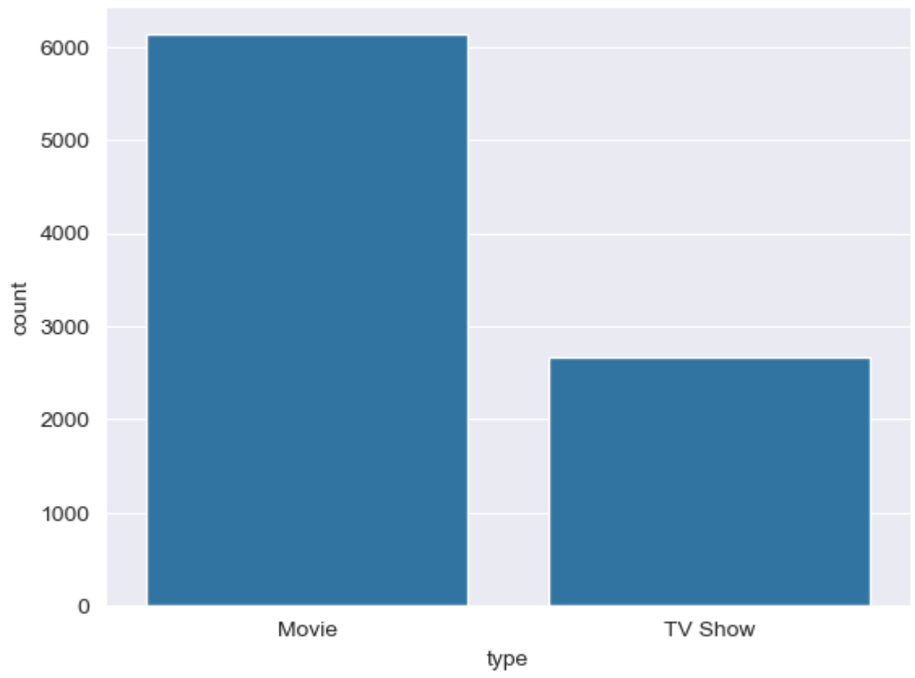
- From above graph, we can see that majority of content that is added was realeased after 2014
- This shows that more and more content producers are using Netflix to distribute their content.

Movie and TV Show Distribution Analysis

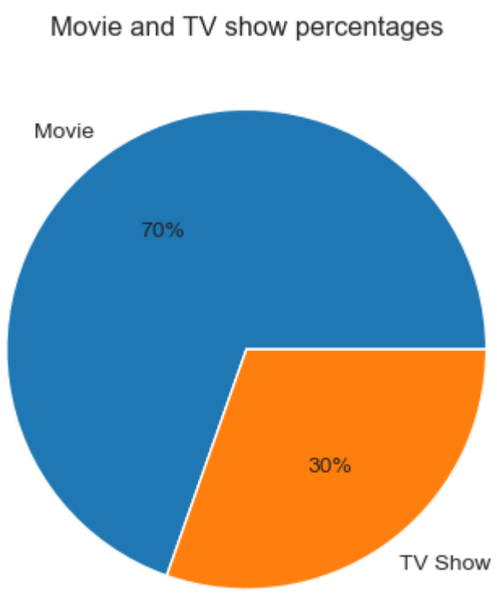
In [ ]:
```
pdf=df["type"].value_counts()
pdf
```

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

In [ ]:
```
sns.countplot(data=df, x="type");
```



In [ ]:
```
plt.pie(pdf, labels=pdf.index, autopct='%.0f%%')
plt.title("Movie and TV show percentages");
```



Insights

- From above plot we can say that movies are more popular than TV shows
- This shows that users are more interested in stories that get over in under 2hrs.

TV Show popularity analysis

In [ ]:
```
tmdf=df.merge(country_df, on="show_id").groupby(["country","type"])["show_id"].count().reset_index().rename(columns={"show_id":"count"}).sort_values(by="count", ascending=False)
tmdf
```

Out[ ]:

	country	type	count
175	United States	Movie	2751
66	India	Movie	962
176	United States	TV Show	938
172	United Kingdom	Movie	532
178	Unknown	Movie	440
...	...	...	...
98	Malta	TV Show	1
1	Albania	Movie	1
91	Lithuania	Movie	1
90	Liechtenstein	Movie	1
94	Malawi	Movie	1

188 rows × 3 columns

In [ ]:
```
tmdf=tmdf.pivot(index="country", columns="type", values="count").fillna(0)
tmdf.loc[tmdf["TV Show"]>tmdf["Movie"]].sort_values(by="TV Show", ascending=False)
```

Out[ ]:

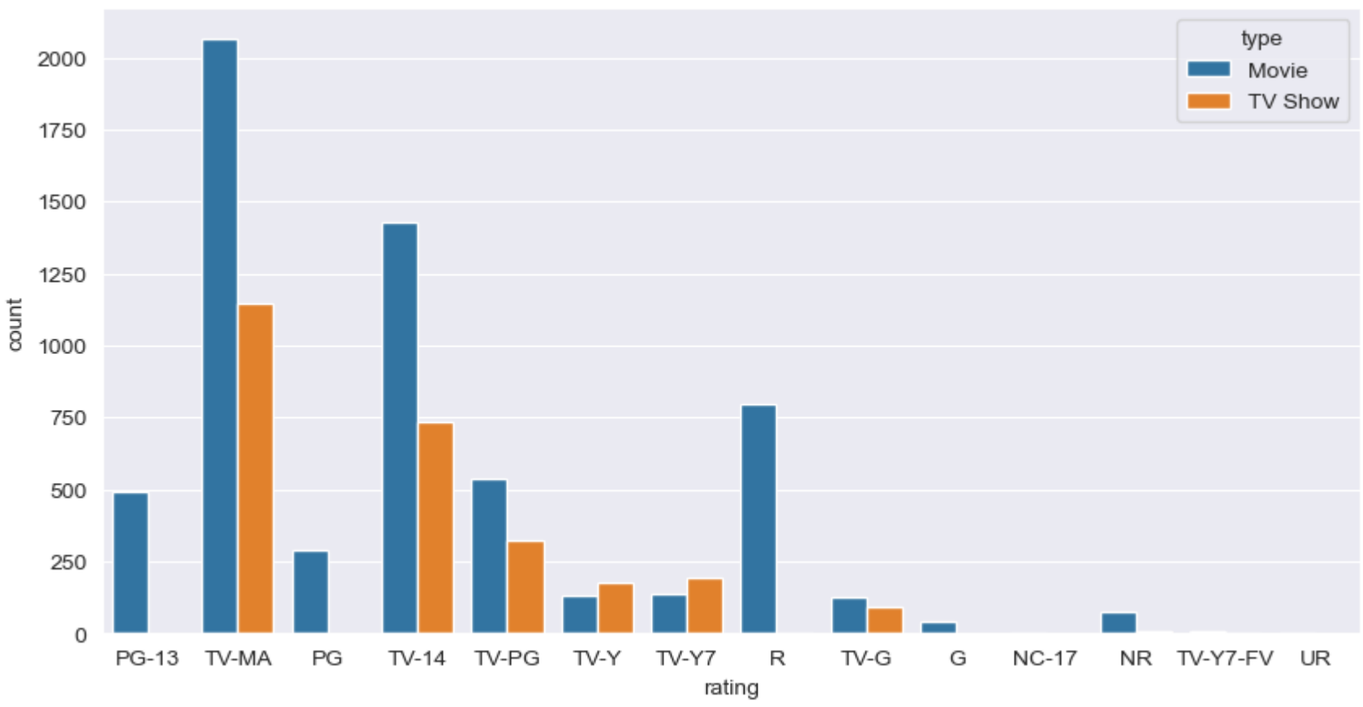
type	Movie	TV Show
country		
Japan	119.0	199.0
South Korea	61.0	169.0
Taiwan	19.0	70.0
Colombia	20.0	32.0
Singapore	18.0	23.0
Russia	11.0	16.0
Ukraine	1.0	2.0
Azerbaijan	0.0	1.0
Belarus	0.0	1.0
Cuba	0.0	1.0
Cyprus	0.0	1.0
Puerto Rico	0.0	1.0

Insights

- Above table consist of those countries which have higher count of tv shows as compared to movies
- We can see that Japan, South Korea, Taiwan have the biggest difference in number of tv shows compared to movies.
- Netflix should add more tv shows in above countries

Rating Analyisis

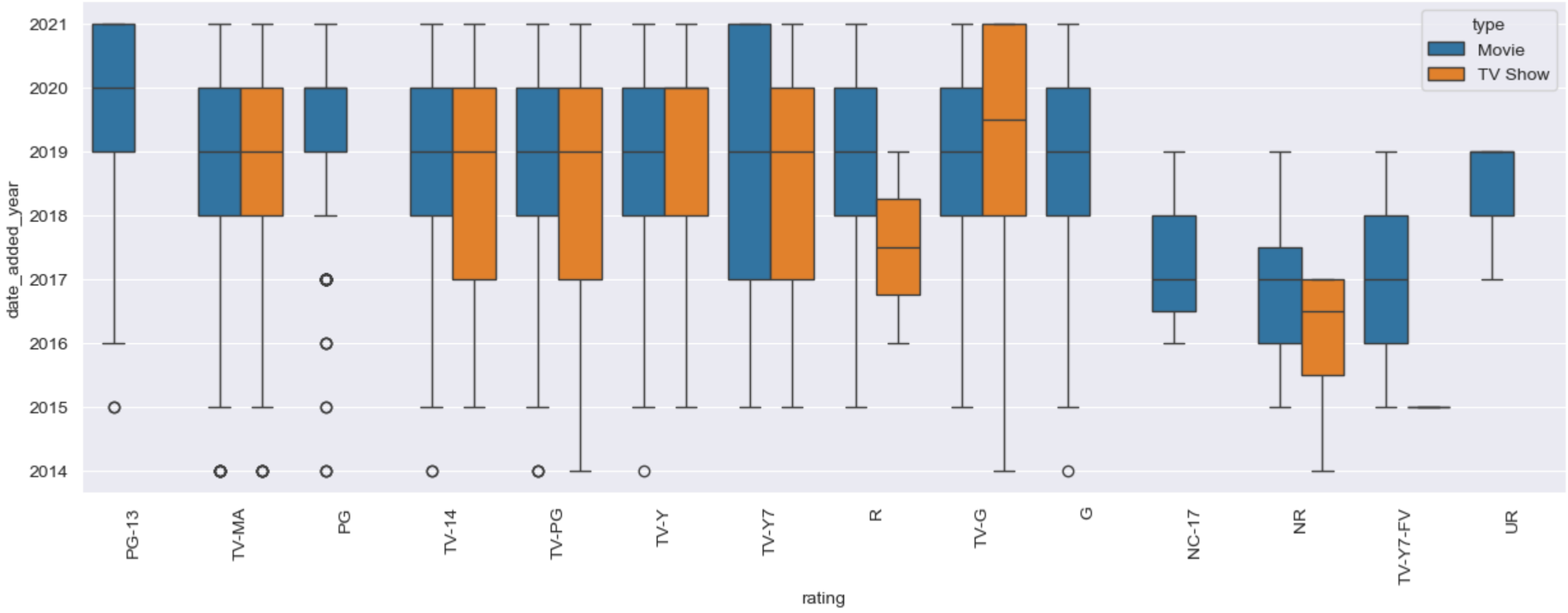
In [ ]:
```
plt.figure(figsize=(10,5))
sns.countplot(data=df, x="rating", hue="type");
```



Insights

- From above plot we can say that "TV-MA" and "TV-14 rated content is the most popular
- This plot also shows that majority of the users are of age 14 and above

In [ ]:
```
plt.figure(figsize=(15, 5))
rating_df = df.loc[df['date_added_year'] >= 2014]
sns.boxplot(data=rating_df, x="rating", y="date_added_year", hue="type")
plt.xticks(rotation=90);
```



Insights

- Majority of the content was added during 2017 - 2020
- It looks like NC-17, NR, TV-Y7-FV rated content was stopped after 2019, indicating less popularity among users

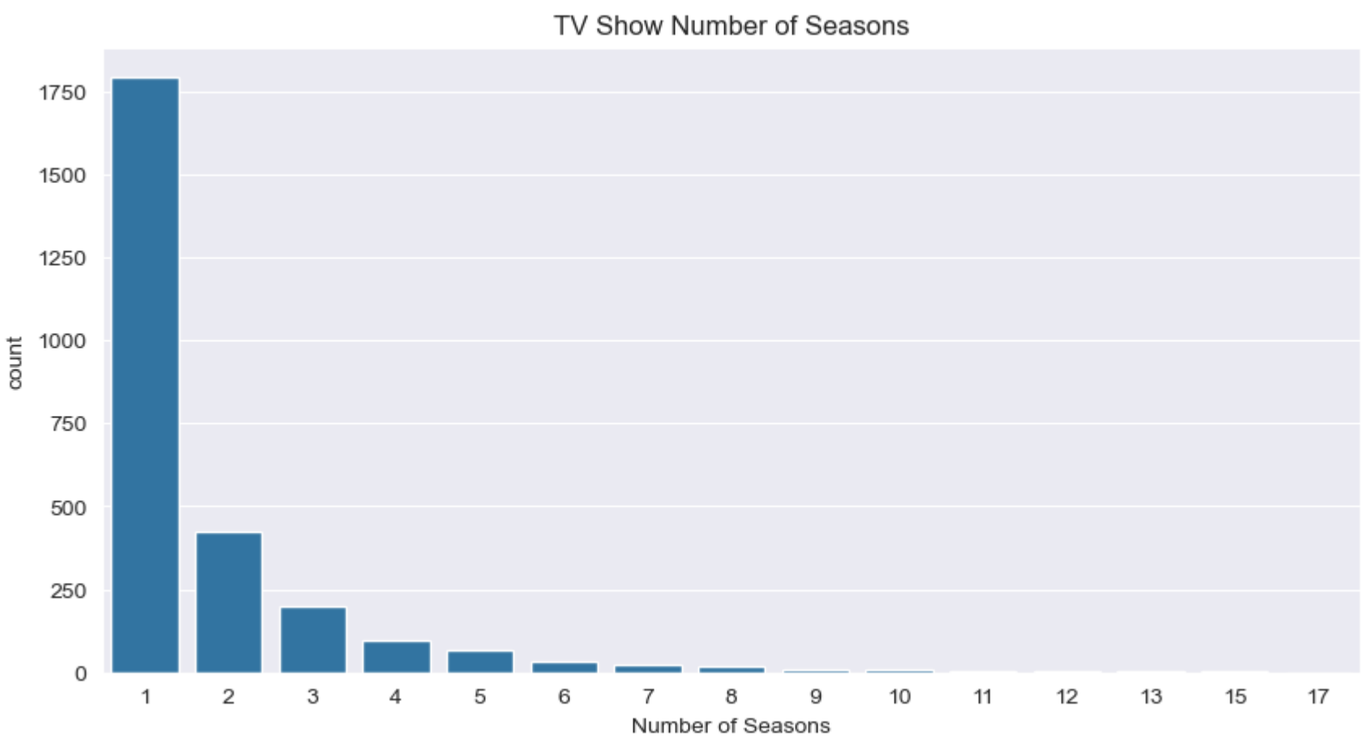
Duration Anaylsis

TV Show Analysis

In [ ]:
```
tv_shows_df["duration"].value_counts()
```

Out[ ]:
```
duration
1     1793
2      425
3      199
4       95
5       65
6       33
7       23
8       17
9        9
10       7
13       3
15       2
12       2
11       2
17       1
Name: count, dtype: int64
```

In [ ]:
```
plt.figure(figsize=(10,5))
sns.countplot(data=tv_shows_df, x="duration")
plt.title("TV Show Number of Seasons")
plt.xlabel("Number of Seasons");
```



Insights

- According to histogram data, we can see that majority of the tv shows have a duration of 1 season.
- This shows that majority of the users like tv shows that end their story in 1 season.

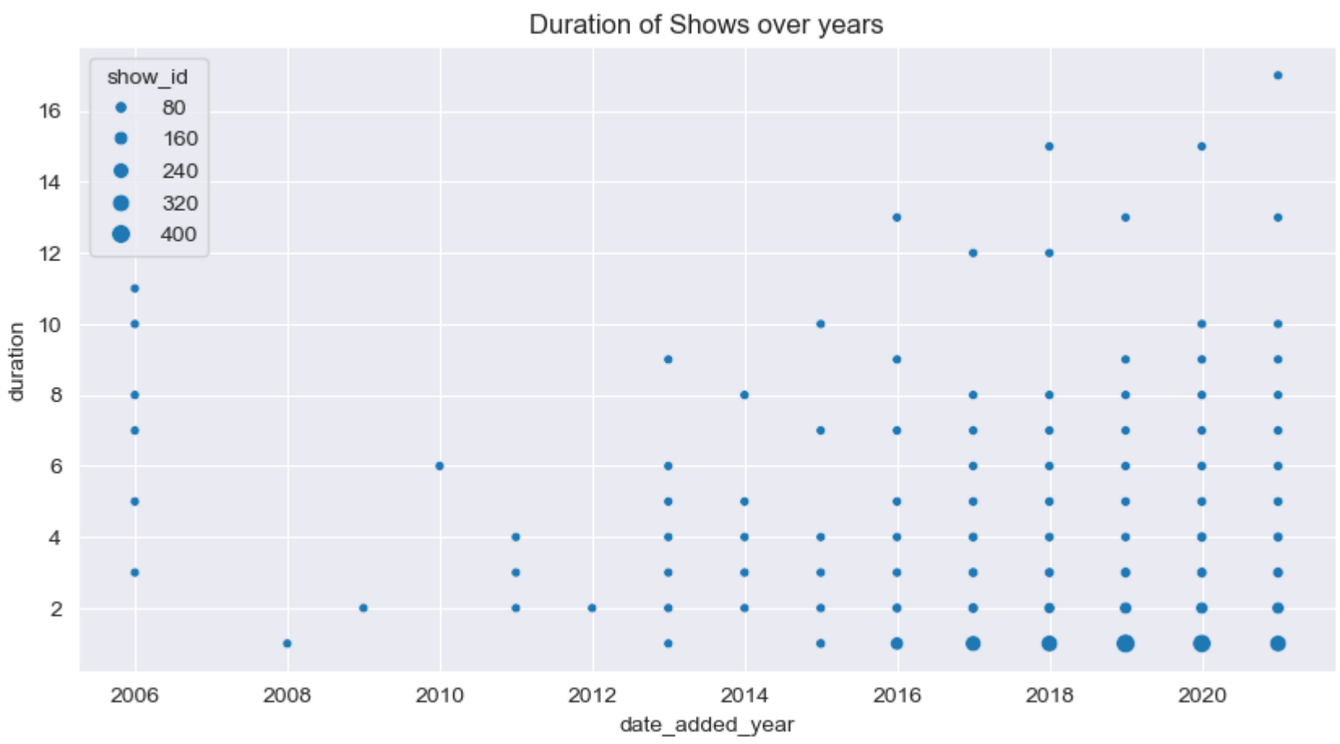
In [ ]:
```
temp_df=tv_shows_df.groupby(["date_added_year","duration"])["show_id"].count().reset_index()
temp_df
```

Out[ ]:

	date_added_year	duration	show_id
0	2006	3	1
1	2006	5	2
2	2006	7	1
3	2006	8	2
4	2006	10	2
...	...	...	...
86	2021	8	1
87	2021	9	2
88	2021	10	1
89	2021	13	1
90	2021	17	1

91 rows × 3 columns

In [ ]:
```
plt.figure(figsize=(10,5))
sns.scatterplot(data=temp_df, x="date_added_year", y="duration", size="show_id")
plt.title("Duration of Shows over years");
```



Insights

- We can say that as small amount of longer duration shows were added to the platform over the years.
- These are the classic shows that have released during the 2000s or before.This indicates that there are some users who would want to watch older shows.
- Netflix should add more of these to the platform.

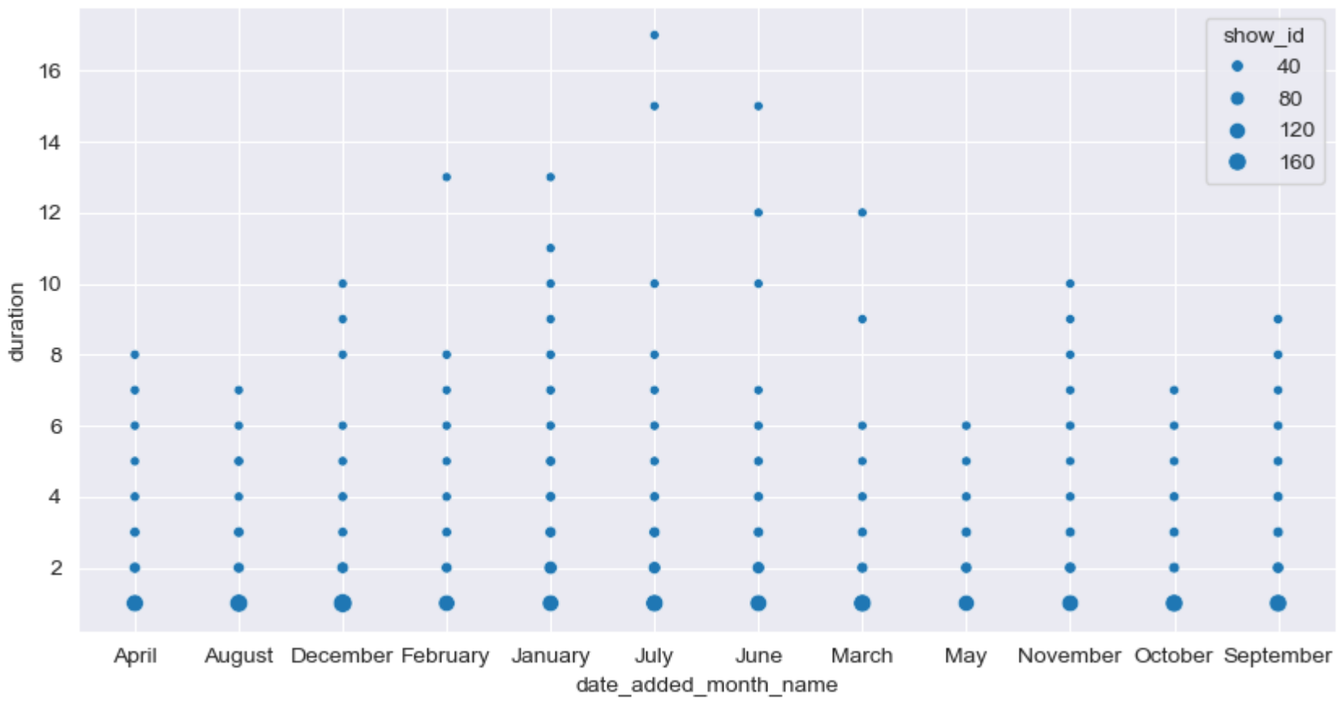
In [ ]:
```
temp_df=tv_shows_df.groupby(["date_added_month_name","duration"])["show_id"].count().reset_index()
temp_df
```

Out[ ]:

	date_added_month_name	duration	show_id
0	April	1	148
1	April	2	29
2	April	3	14
3	April	4	6
4	April	5	3
...	...	...	...
101	September	5	7
102	September	6	4
103	September	7	2
104	September	8	4
105	September	9	5

106 rows × 3 columns

In [ ]:
```
plt.figure(figsize=(10,5))
sns.scatterplot(data=temp_df, x="date_added_month_name", y="duration", size="show_id");
```



Insights

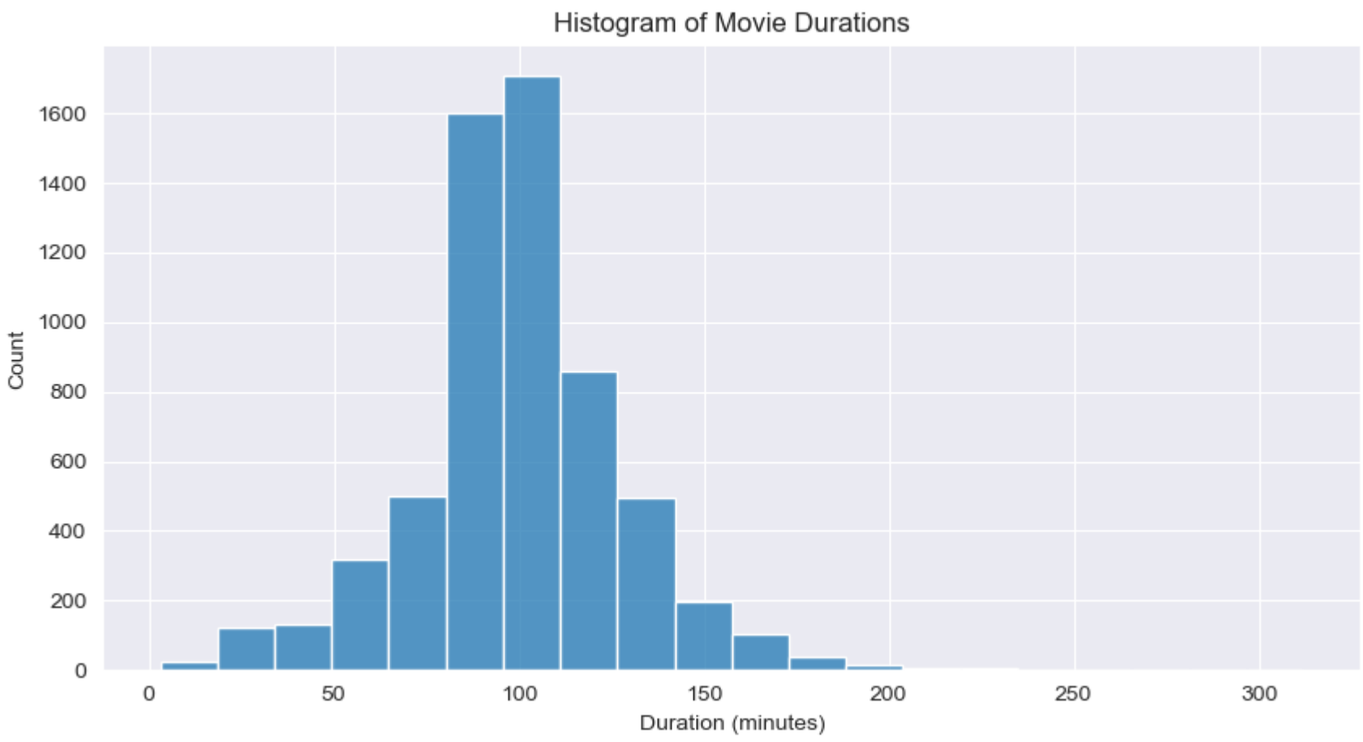
- There seems to be no pattern of number of season wrt added month of the year

Movie Analysis

In [ ]:
```
movies_df["duration"].value_counts()
```

Out[ ]:
```
duration
90     152
94     146
97     146
93     146
91     144
      ...
208      1
5        1
16       1
186      1
191      1
Name: count, Length: 205, dtype: int64
```

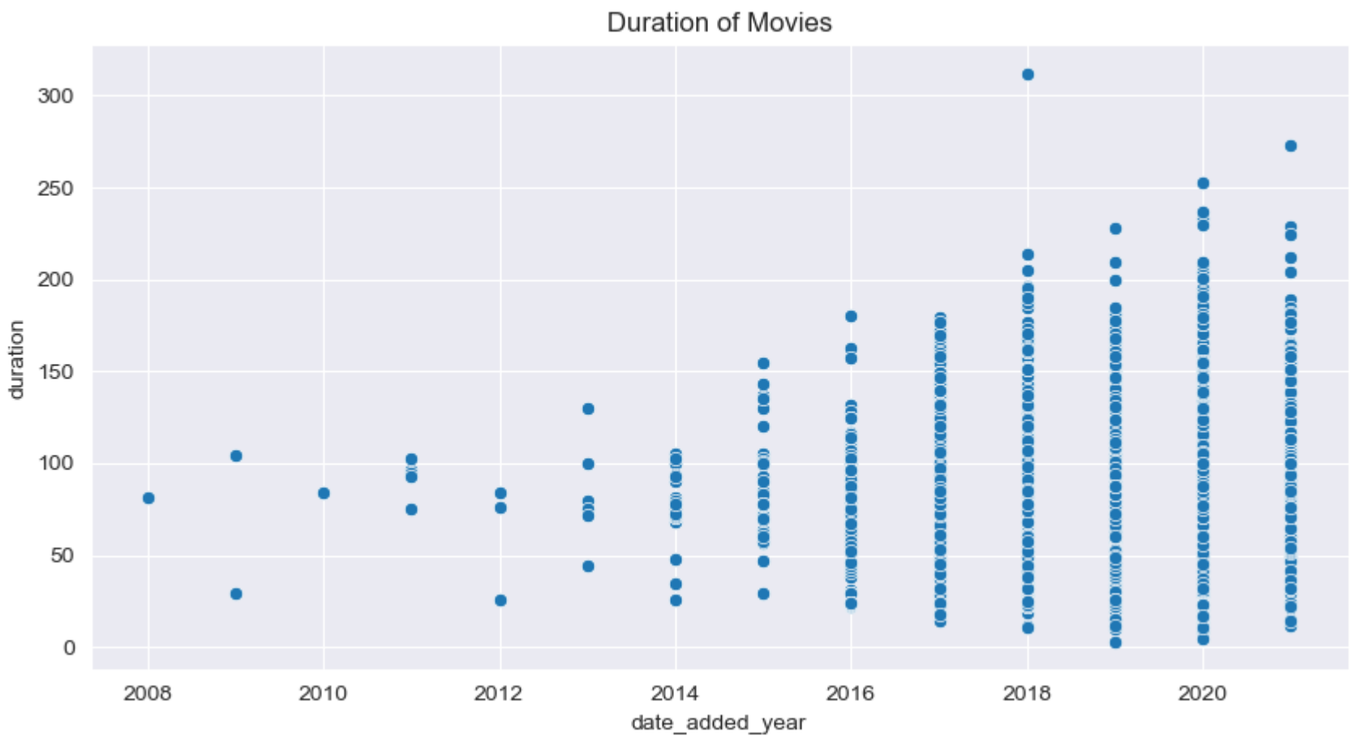
In [ ]:
```
plt.figure(figsize=(10,5))
sns.histplot(data=movies_df, x="duration", bins=20)
plt.title("Histogram of Movie Durations")
plt.xlabel("Duration (minutes)");
```



Insights

- According to histogram data, we can see that majority of the movies have a duration of 90min - 120min.
- This shows that people are highly interested in movies that are around 2 hrs.

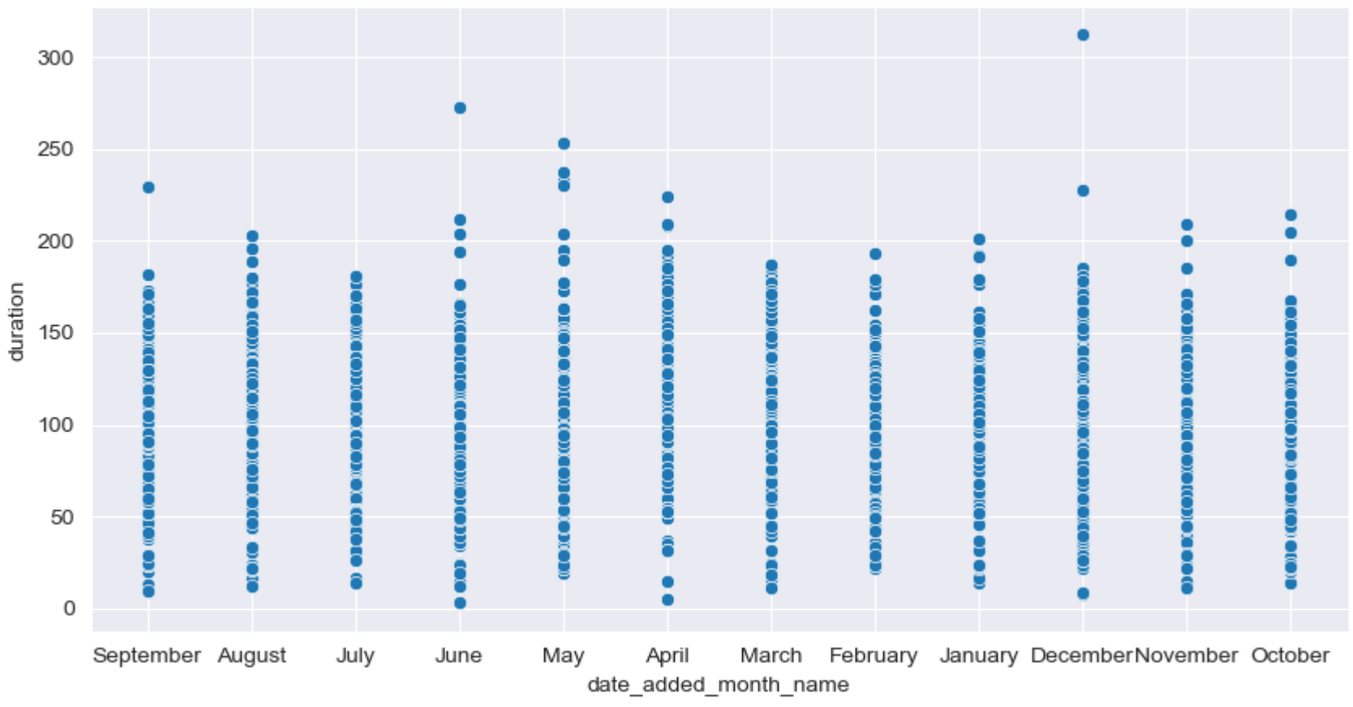
In [ ]:
```
plt.figure(figsize=(10,5))
sns.scatterplot(data=movies_df, x="date_added_year", y="duration");
plt.title("Duration of Movies");
```



Insights

- There appears to be both increase and decrease in movie duration over the years.

In [ ]:
```
plt.figure(figsize=(10,5))
sns.scatterplot(data=movies_df, x="date_added_month_name", y="duration");
```



Insights

- There seems to be no pattern of duration of movies wrt added month of the year

Recommendations

General Recommendations

Netflix is very popular in US with over 37% of content belonging to it. Because of this, it is important to focus on other countries having lesser content.
By analyzing Netflix data, we have come up with following recommendations:

- Content should be released on platform during the month of July, December and January.
- Content should not be released during the month of February.
- More movies should be produced/added to Netflix as movies are more popular than TV shows.
- Netflix should added more content in International movies, Dramas and Comedies genres.
- Movie timings should be around 120 minutes.
- TV shows should contain 1 season as these are more popular than shows having multiple season.
- In Japan, South Korea and Taiwan, TV shows are more popular than movies. More TV shows should be produced in these countries.
- Movies should be added to Netflix within 1 year of release date.
- Netflix should add more of classic tv shows to their platform for catering to older audiences.

Content Recommendations

In this analysis, we are going to look at improving business in countries that are in top 25 list excluding US. We will do this by ranking Genre, Cast, Rating and Director for each country.
Netflix can use these ranking to determine the best content for each country.

In [ ]:
```
filter_country_df = country_df.loc[country_df["country"].isin(improvement_countries)]
```

Top genres for each country

In [ ]:
```
merge_df = df.merge(filter_country_df, on="show_id")
merge_df = merge_df.merge(listed_df, on="show_id")
```

In [ ]:
```
gp=merge_df.groupby(["country","type","listed_in"])["show_id"].count().reset_index().rename(columns={"show_id":"count"})
gp["rank"]=gp.groupby(["country","type"])["count"].rank(method="first",ascending=False)
gp=gp.loc[gp["rank"] < 3].sort_values(by='country')
suggested_movie_genres =  gp.loc[(gp["type"]=='Movie')].sort_values(by=["country","rank"]).drop(["type","count"], axis=1)
suggested_tv_genres =  gp.loc[(gp["type"]=='TV Show')].sort_values(by=["country","rank"]).drop(["type","count"], axis=1)
```

In [ ]:
```
pd.set_option('display.max_rows', None)
suggested_movie_genres.merge(suggested_tv_genres, on=['country','rank']).rename(columns={'listed_in_x':'Movie Genre', 'listed_in_y':'TV Genres'})
```

Out[ ]:

	country	Movie Genre	rank	TV Genres
0	Argentina	International Movies	1.0	Spanish-Language TV Shows
1	Argentina	Dramas	2.0	International TV Shows
2	Australia	Dramas	1.0	International TV Shows
3	Australia	International Movies	2.0	Kids' TV
4	Belgium	International Movies	1.0	International TV Shows
5	Belgium	Dramas	2.0	Crime TV Shows
6	Brazil	International Movies	1.0	International TV Shows
7	Brazil	Dramas	2.0	TV Dramas
8	Canada	Comedies	1.0	Kids' TV
9	Canada	Dramas	2.0	TV Dramas
10	China	International Movies	1.0	International TV Shows
11	China	Action & Adventure	2.0	Romantic TV Shows
12	Egypt	International Movies	1.0	International TV Shows
13	Egypt	Comedies	2.0	TV Dramas
14	France	International Movies	1.0	International TV Shows
15	France	Dramas	2.0	Kids' TV
16	Germany	International Movies	1.0	International TV Shows
17	Germany	Dramas	2.0	TV Dramas
18	Hong Kong	International Movies	1.0	International TV Shows
19	Hong Kong	Action & Adventure	2.0	TV Dramas
20	India	International Movies	1.0	International TV Shows
21	India	Dramas	2.0	TV Dramas
22	Indonesia	International Movies	1.0	International TV Shows
23	Indonesia	Dramas	2.0	TV Dramas
24	Italy	International Movies	1.0	International TV Shows
25	Italy	Dramas	2.0	TV Dramas
26	Japan	International Movies	1.0	International TV Shows
27	Japan	Anime Features	2.0	Anime Series
28	Mexico	International Movies	1.0	Spanish-Language TV Shows
29	Mexico	Dramas	2.0	International TV Shows
30	Nigeria	International Movies	1.0	International TV Shows
31	Nigeria	Dramas	2.0	TV Dramas
32	Philippines	International Movies	1.0	International TV Shows
33	Philippines	Dramas	2.0	Anime Series
34	South Korea	International Movies	1.0	International TV Shows
35	South Korea	Dramas	2.0	Korean TV Shows
36	Spain	International Movies	1.0	International TV Shows
37	Spain	Dramas	2.0	Spanish-Language TV Shows
38	Taiwan	International Movies	1.0	International TV Shows
39	Taiwan	Dramas	2.0	Romantic TV Shows
40	Thailand	International Movies	1.0	International TV Shows
41	Thailand	Horror Movies	2.0	TV Dramas
42	Turkey	International Movies	1.0	International TV Shows
43	Turkey	Comedies	2.0	TV Dramas
44	United Kingdom	Dramas	1.0	British TV Shows
45	United Kingdom	International Movies	2.0	International TV Shows

Insights

- Above table consist of top movie genre and tv genres in each country.
- Netflix can use this data to determine the most popular genre in each country.
  for eg: Australians like **Drama** movies and **International TV Shows**

Top rated content for each country

In [ ]:
```
gp=merge_df.groupby(["country","type","rating"])["show_id"].count().reset_index().rename(columns={"show_id":"count"})
gp["rank"]=gp.groupby(["country","type"])["count"].rank(method="first",ascending=False)
gp=gp.loc[gp["rank"] < 3].sort_values(by='country')
suggested_tv_ratings =  gp.loc[(gp["type"]=='TV Show')].sort_values(by=["country","rank"]).drop(["count","type"], axis=1)
suggested_movie_ratings =  gp.loc[(gp["type"]=='Movie')].sort_values(by=["country","rank"]).drop(["count","type"], axis=1)
```

In [ ]:
```
suggested_movie_ratings.merge(suggested_tv_ratings, on=['country','rank']).rename(columns={'rating_x':'Movie Rating', 'rating_y':'TV Rating'})
```

Out[ ]:

	country	Movie Rating	rank	TV Rating
0	Argentina	TV-MA	1.0	TV-MA
1	Argentina	TV-14	2.0	TV-14
2	Australia	R	1.0	TV-MA
3	Australia	TV-MA	2.0	TV-PG
4	Belgium	TV-MA	1.0	TV-MA
5	Belgium	R	2.0	TV-14
6	Brazil	TV-MA	1.0	TV-MA
7	Brazil	TV-14	2.0	TV-PG
8	Canada	R	1.0	TV-MA
9	Canada	TV-MA	2.0	TV-14
10	China	TV-14	1.0	TV-14
11	China	TV-MA	2.0	TV-MA
12	Egypt	TV-14	1.0	TV-14
13	Egypt	TV-MA	2.0	TV-MA
14	France	TV-MA	1.0	TV-MA
15	France	R	2.0	TV-Y
16	Germany	TV-MA	1.0	TV-MA
17	Germany	R	2.0	TV-14
18	Hong Kong	TV-14	1.0	TV-MA
19	Hong Kong	TV-MA	2.0	TV-14
20	India	TV-14	1.0	TV-MA
21	India	TV-MA	2.0	TV-14
22	Indonesia	TV-14	1.0	TV-14
23	Indonesia	TV-PG	2.0	TV-MA
24	Italy	TV-MA	1.0	TV-MA
25	Italy	TV-14	2.0	TV-Y7
26	Japan	TV-MA	1.0	TV-14
27	Japan	TV-PG	2.0	TV-MA
28	Mexico	TV-MA	1.0	TV-MA
29	Mexico	R	2.0	TV-14
30	Nigeria	TV-14	1.0	TV-MA
31	Nigeria	TV-MA	2.0	TV-14
32	Philippines	TV-14	1.0	TV-MA
33	South Korea	TV-MA	1.0	TV-14
34	South Korea	TV-14	2.0	TV-MA
35	Spain	TV-MA	1.0	TV-MA
36	Spain	TV-14	2.0	TV-PG
37	Taiwan	TV-MA	1.0	TV-14
38	Taiwan	TV-14	2.0	TV-MA
39	Thailand	TV-MA	1.0	TV-MA
40	Thailand	TV-14	2.0	TV-14
41	Turkey	TV-MA	1.0	TV-MA
42	Turkey	TV-14	2.0	TV-14
43	United Kingdom	R	1.0	TV-MA
44	United Kingdom	TV-MA	2.0	TV-PG

Insights

- Above table shows top movie and tv ratings in each country.
- Netflix can use this data to predict what type of content users want to watch.
- For example, Japan likes TV-M movies and TV-14 rated TV shows. This shows that most of the tv users here under the age of 14.

Top cast in each country

In [ ]:
```
pd.reset_option('display.max_rows')
```

In [ ]:
```
merge_df = df.merge(filter_country_df, on="show_id")
merge_df = merge_df.merge(cast_df, on="show_id")
merge_df=merge_df.loc[merge_df["cast"] != "Unknown"]
gp=merge_df.groupby(["country","type","cast"])["show_id"].count().reset_index().rename(columns={"show_id":"count"})
gp["rank"]=gp.groupby(["country","type"])["count"].rank(method="first",ascending=False)
gp=gp.loc[gp["rank"] < 3].sort_values(by='country')
suggested_movie_cast =  gp.loc[(gp["type"]=='Movie')].sort_values(by=["country","rank"]).drop(["type","count"], axis=1)
suggested_tv_cast =  gp.loc[(gp["type"]=='TV Show')].sort_values(by=["country","rank"]).drop(["count","type"], axis=1)
```

In [ ]:
```
pd.set_option('display.max_rows', None)
suggested_movie_cast.merge(suggested_tv_cast, on=['country','rank']).rename(columns={'cast_x':'Movie Cast', 'cast_y':'TV Cast'})
```

Out[ ]:

	country	Movie Cast	rank	TV Cast
0	Argentina	Andrea Frigerio	1.0	Chino Darín
1	Argentina	Joaquín Furriel	2.0	Fabio Aste
2	Australia	Sam Neill	1.0	Alex Dimitriades
3	Australia	Emily Morris	2.0	Danielle Cormack
4	Belgium	Matthias Schoenaerts	1.0	Charlotte Timmers
5	Belgium	Bérénice Bejo	2.0	Jeroen Perceval
6	Brazil	Dalton Vigh	1.0	Jonathan Haagensen
7	Brazil	Eduardo Galvão	2.0	Wallie Ruy
8	Canada	John Paul Tremblay	1.0	Ashleigh Ball
9	Canada	Robb Wells	2.0	Vincent Tong
10	China	Donnie Yen	1.0	Dilraba Dilmurat
11	China	Jackie Chan	2.0	James Wen
12	Egypt	Ahmed Helmy	1.0	Ahmed Dawood
13	Egypt	Hassan Hosny	2.0	Sawsan Badr
14	France	Wille Lindberg	1.0	Jason Narvy
15	France	Benoît Magimel	2.0	Paul Schrier
16	Germany	Charly Hübner	1.0	Anna Maria Mühe
17	Germany	Daniel Brühl	2.0	David Attenborough
18	Hong Kong	Donnie Yen	1.0	Justin Cheung
19	Hong Kong	Lam Suet	2.0	Adam Pak
20	India	Anupam Kher	1.0	Nishka Raheja
21	India	Shah Rukh Khan	2.0	Rajesh Kava
22	Indonesia	Reza Rahadian	1.0	Aci Resti
23	Indonesia	Maudy Koesnaedi	2.0	Adjis Doaibu
24	Italy	Riccardo Scamarcio	1.0	Suzy Myers
25	Italy	Elio Germano	2.0	Alyson Leigh Rosenfeld
26	Japan	Yuki Kaji	1.0	Takahiro Sakurai
27	Japan	Chie Nakamura	2.0	Yuki Kaji
28	Mexico	Cassandra Ciangherotti	1.0	Damián Alcázar
29	Mexico	Fernando Becerril	2.0	Erik Hayser
30	Nigeria	Blossom Chukwujekwu	1.0	Bimbo Manuel
31	Nigeria	Tina Mba	2.0	Jude Chukwuka
32	Philippines	Kathryn Bernardo	1.0	Allen Dizon
33	Philippines	Joross Gamboa	2.0	Alora Sasam
34	South Korea	Kyeong-yeong Lee	1.0	Cho Seong-ha
35	South Korea	Dal-su Oh	2.0	Kim Won-hae
36	Spain	Mario Casas	1.0	Carlos Cuevas
37	Spain	Carmen Machi	2.0	Eloy Azorín
38	Taiwan	Chang Chen	1.0	Amanda Chou
39	Taiwan	Chen Yi-wen	2.0	Jack Lee
40	Thailand	Sahajak Boonthanakit	1.0	Chutavuth Pattarakampol
41	Thailand	Arisara Thongborisut	2.0	Kanyawee Songmuang
42	Turkey	Demet Akbağ	1.0	Alican Yücesoy
43	Turkey	Cezmi Baskın	2.0	Haluk Bilginer
44	United Kingdom	John Cleese	1.0	David Attenborough
45	United Kingdom	Judi Dench	2.0	Eric Idle

Insights

- Above table shows which cast is the most popular in each country.
- Netflix can use this data to predict what type of content users want to watch.
- Using these casts, Netflix can produce their original content and distribute in their respective countries.

Top directors in each country

In [ ]:
```
merge_df = df.merge(filter_country_df, on="show_id")
merge_df = merge_df.merge(director_df, on="show_id")
merge_df=merge_df.loc[merge_df["director"] != "Unknown"]
gp=merge_df.groupby(["country","type","director"])["show_id"].count().reset_index().rename(columns={"show_id":"count"})
gp=gp.loc[gp["count"] > 0]
gp["rank"]=gp.groupby(["country","type"])["count"].rank(method="first",ascending=False)
gp=gp.loc[gp["rank"] < 3].sort_values(by='country')
suggested_movie_director =  gp.loc[(gp["type"]=='Movie')].sort_values(by=["country","rank"]).drop(["type","count"], axis=1)
suggested_tv_director =  gp.loc[(gp["type"]=='TV Show')].sort_values(by=["country","rank"]).drop(["count","type"], axis=1)
```

In [ ]:
```
pd.set_option('display.max_rows', None)
suggested_movie_director.merge(suggested_tv_director, on=['country','rank'], how="left")\
.rename(columns={'director_x':'Movie Director', 'director_y':'TV Director'})\
.replace(np.nan, '', regex=True)
```

Out[ ]:

	country	Movie Director	rank	TV Director
0	Argentina	Jan Suter	1.0	Alejandro Hartmann
1	Argentina	Raúl Campos	2.0	Hernán Guerschuny
2	Australia	Clay Glen	1.0	Mat King
3	Australia	Jane Campion	2.0	
4	Belgium	Jalil Lespert	1.0	Alain Brunard
5	Belgium	Lars von Trier	2.0	Wouter Bouvijn
6	Brazil	Diego Pignataro	1.0	Andrucha Waddington
7	Brazil	Lucas Margutti	2.0	Carla Barros
8	Canada	Justin G. Dyck	1.0	Alastair Fothergill
9	Canada	Mike Clattenburg	2.0	Gary Howsam
10	China	Wilson Yip	1.0	Han Qing
11	China	Johnnie To	2.0	He Xiaofeng
12	Egypt	Youssef Chahine	1.0	
13	Egypt	Sameh Abdulaziz	2.0	
14	France	Thierry Donard	1.0	Adrien Lagier
15	France	Youssef Chahine	2.0	Alan Poul
16	Germany	Detlev Buck	1.0	Alan Poul
17	Germany	Fernando González Molina	2.0	Alastair Fothergill
18	Hong Kong	Johnnie To	1.0	
19	Hong Kong	Wong Jing	2.0	
20	India	David Dhawan	1.0	Gautham Vasudev Menon
21	India	Anurag Kashyap	2.0	Anurag Kashyap
22	Indonesia	Hanung Bramantyo	1.0	Jay Oliva
23	Indonesia	Riri Riza	2.0	
24	Italy	Dino Risi	1.0	Iginio Straffi
25	Italy	Francesco Imperato	2.0	Cecilia Peck
26	Japan	Toshiya Shinohara	1.0	Caroline Sá
27	Japan	Masahiko Murata	2.0	Chico Pereira
28	Mexico	Jan Suter	1.0	Adrián García Bogliano
29	Mexico	Raúl Campos	2.0	Alejandro Lozano
30	Nigeria	Kunle Afolayan	1.0	BB Sasore
31	Nigeria	Omoni Oboli	2.0	Kemi Adetiba
32	Philippines	Cathy Garcia-Molina	1.0	Jay Oliva
33	Philippines	Mae Czarina Cruz	2.0	Richard Arellano
34	South Korea	Bong Joon Ho	1.0	Jung-ah Im
35	South Korea	Mark A.Z. Dippé	2.0	Shin Won-ho
36	Spain	Fernando González Molina	1.0	Alastair Fothergill
37	Spain	Hernán Zin	2.0	Carlos Sedes
38	Taiwan	Ang Lee	1.0	Chang Chin-jung
39	Taiwan	Brody Chu	2.0	Chen Hung-yi
40	Thailand	Banjong Pisanthanakun	1.0	Cheewatan Pusitsuksa
41	Thailand	Poj Arnon	2.0	Kongkiat Khomsiri
42	Turkey	Yılmaz Erdoğan	1.0	Ahmet Katıksız
43	Turkey	Hakan Algül	2.0	Neslihan Yesilyurt
44	United Kingdom	Edward Cotterill	1.0	Alastair Fothergill
45	United Kingdom	Blair Simmons	2.0	Alan Poul

Insights

- Above table shows which director is the most popular in each country.
- Netflix can use this data to predict what type of content users want to watch.
- Using these directors, Netflix can produce their original content and distribute in their respective countries.