In [1]:

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

In [2]:

```
df = pd.read_csv("C:/Users/91981/Desktop/Mansi/scaler/Python Visualisation/netflix.csv")
```

In [3]:

```
df.head()
```

Out[3]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	c
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	S
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	

	show_id	type	title	director	cast	country	date_added	release_year	rating	c
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	S

In [4]:

```
df.shape
```

Out[4]:

```
(8807, 12)
```

In [5]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

In [6]:

```
df.columns
```

Out[6]:

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

In [7]:

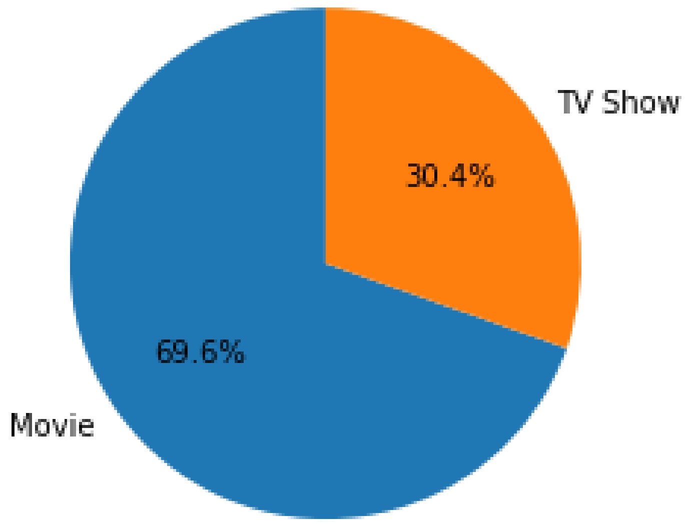
```
type_counts = df['type'].value_counts()
type_counts
```

Out[7]:

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

```
In [8]:
```

```
plt.pie(type_counts, labels= type_counts.index, autopct='%1.1f%%', startangle=90)
plt.show()
```



In [9]:

```
df.describe(include = 'all')
```

Out[9]:

	show_id	type	title	director	cast	country	date_added	release_year	rat
count	8807	8807	8807	6173	7982	7976	8797	8807.000000	8
unique	8807	2	8807	4528	7692	748	1767	NaN	
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	NaN	
freq	1	6131	1	19	19	2818	109	NaN	3
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2014.180198	N
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.819312	N
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1925.000000	N
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2013.000000	N
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2017.000000	N
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2019.000000	N
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2021.000000	N

In [10]:

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

Out[10]:

```
Rajiv Chilaka                     19
Raúl Campos, Jan Suter            18
Marcus Raboy                      16
Suhas Kadav                       16
Jay Karas                         14
                                  ..
Raymie Muzquiz, Stu Livingston     1
Joe Menendez                       1
Eric Bross                         1
Will Eisenberg                     1
Mozez Singh                        1
Name: director, Length: 4528, dtype: int64
```

In [11]:

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

```
Out[11]:
```

```
David Attenborough
19
Vatsal Dubey, Julie Tejwani, Rupa Bhimani, Jigna Bhardwaj, Rajesh Kava, Mous
am, Swapnil
14
Samuel West
10
Jeff Dunham
7
David Spade, London Hughes, Fortune Feimster
6
..
Michael Peña, Diego Luna, Tenoch Huerta, Joaquin Cosio, José María Yazpik, M
att Letscher, Alyssa Diaz
1
Nick Lachey, Vanessa Lachey
1
Takeru Sato, Kasumi Arimura, Haru, Kentaro Sakaguchi, Takayuki Yamada, Kendo
Kobayashi, Ken Yasuda, Arata Furuta, Suzuki Matsuo, Koichi Yamadera, Arata I
ura, Chikako Kaku, Kotaro Yoshida     1
Toyin Abraham, Sambasa Nzeribe, Chioma Chukwuka Akpotha, Chioma Omeruah, Chi
wetalu Agu, Dele Odule, Femi Adebayo, Bayray McNwizu, Biodun Stephen
1
Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanana, Manish Chaudhary, Meghna Ma
lik, Malkeet Rauni, Anita Shabdish, Chittaranjan Tripathy
```

In [12]:

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

Out[12]:

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

```
1
Name: cast, Length: 7692, dtype: int64
```

In [13]:

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

Out[13]:

```
2018    1147
2017    1032
2019    1030
2020     953
2016     902
        ...
1959       1
1925       1
1961       1
1947       1
1966       1
Name: release_year, Length: 74, dtype: int64
```

In [14]:

```
df['date_added'] = pd.to_datetime(df['date_added'])
df['DA_year'] = df['date_added'].dt.year
df['DA_month'] = df['date_added'].dt.month_name().str[:3]
```

In [15]:

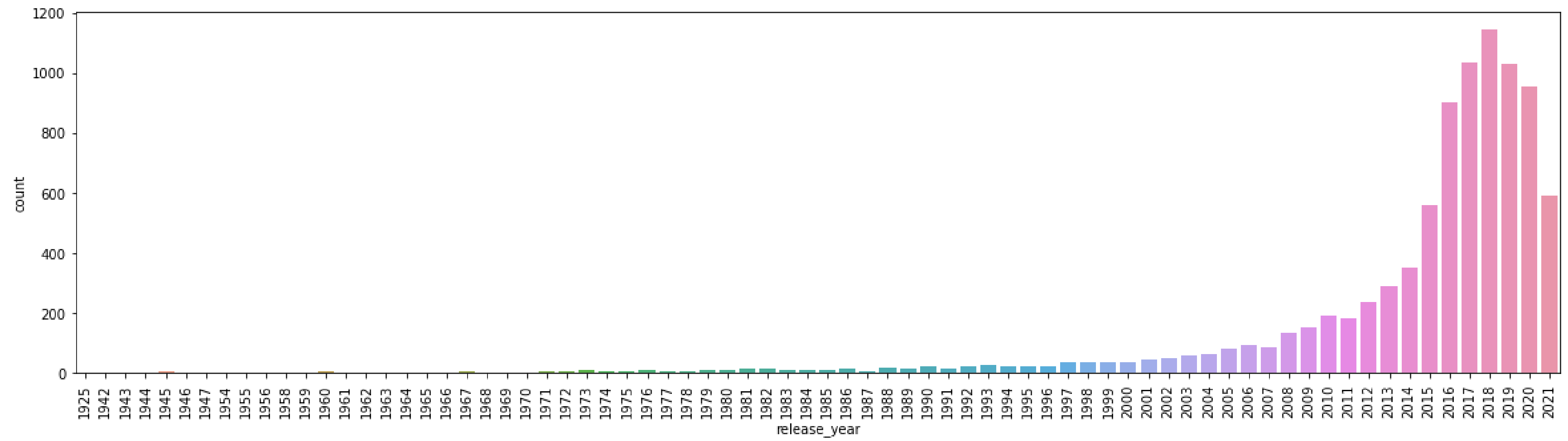
```
## Univariate and Bivariate Ananlysis for unnested data alongwith missing value imoutation
```

In [16]:

```
# Univariate Analysis
```

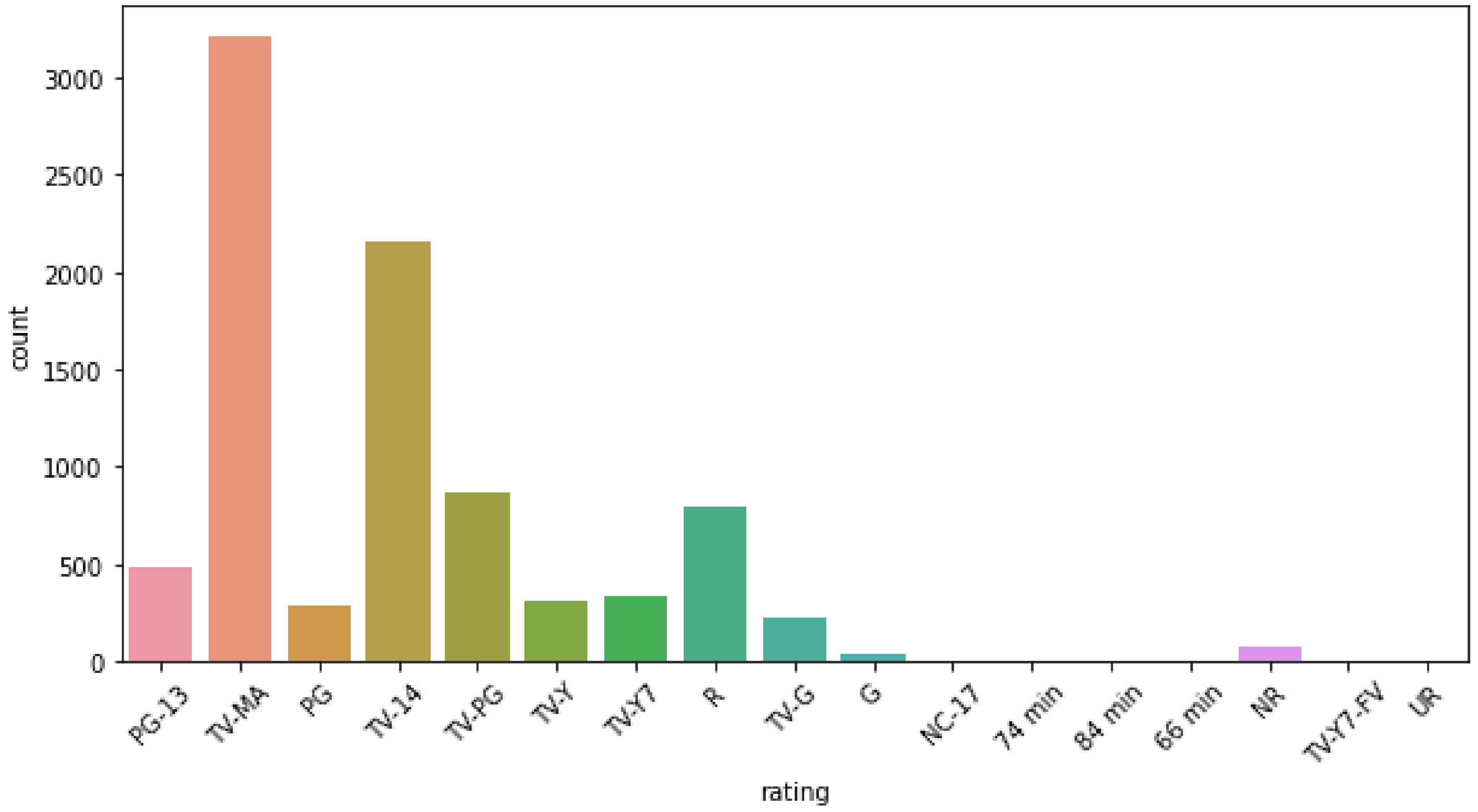
In [17]:

```python
fig = plt.figure(figsize = (20,5))
sns.countplot(x= 'release_year', data = df)
plt.xticks(rotation = 90)
plt.show()
```



In [18]:

```
fig = plt.figure(figsize = (10,5))
sns.countplot(x= 'rating', data = df)
plt.xticks(rotation = 45)
plt.show()
```



In [19]:

```
df.loc[df['rating']=='74 min']
```

Out[19]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duratio
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	2017-04-04	2017	74 min	NaI

```
In [20]:
df.loc[df['rating']=='84 min']
Out[20]:
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	dura
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	2016-09-16	2010	84 min	

In [21]:

```
df.loc[df['rating']=='66 min']
```

Out[21]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	dura
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	2016-08-15	2015	66 min	I

In [22]:

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

Out[22]:

```
Dramas, International Movies                           362
Documentaries                                          359
Stand-Up Comedy                                        334
Comedies, Dramas, International Movies                 274
Dramas, Independent Movies, International Movies       252
                                                      ...
Kids' TV, TV Action & Adventure, TV Dramas               1
TV Comedies, TV Dramas, TV Horror                        1
Children & Family Movies, Comedies, LGBTQ Movies         1
Kids' TV, Spanish-Language TV Shows, Teen TV Shows       1
Cult Movies, Dramas, Thrillers                           1
Name: listed_in, Length: 514, dtype: int64
```

In [23]:

```
# Missing value Imputation
```

In [24]:

```
df['director'] = df['director'].fillna('missing')
```

In [25]:

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

Out[25]:

```
0
```

In [26]:

```
df['cast'] = df['cast'].fillna('missing')
```

In [27]:

```
country_mode = df['country'].mode()[0]
country_mode
```

Out[27]:

```
'United States'
```

In [28]:

```
df['country'] = df['country'].fillna(country_mode)
```

In [29]:

```
df['date_added'].fillna(df['date_added'].mode()[0], inplace = True)
```

In [30]:

```
# Correcting the wrongly written values in the rating columns to duration column for the ra
# imputing the corresponding rating cells with nan values...This will treat the missing val
# missing value in the rating column instead...post this step we will have to do missing va
# column
```

In [31]:

```
df.at[5541,'duration'] = df.at[5541,'rating']
df.at[5794,'duration'] = df.at[5794,'rating']
df.at[5813,'duration'] = df.at[5813,'rating']
```

In [32]:

```
df.at[5813,'duration']
```

Out[32]:

```
'66 min'
```

In [33]:

```
df.at[5541,'rating'] = np.nan
df.at[5794,'rating'] = np.nan
df.at[5813,'rating'] = np.nan
```

In [34]:

```
df.at[5541,'rating']
```

Out[34]:

```
nan
```

In [35]:

```
df['rating'].fillna(df['rating'].mode()[0], inplace = True)
```

In [36]:

```
# Convert date_added column to datetime and add year and month column
```

In [37]:

```
df['date_added'] = pd.to_datetime(df['date_added'])
df['DA_year'] = df['date_added'].dt.year
df['DA_month'] = df['date_added'].dt.month_name().str[:3]
```

In [38]:

```
# Treating duration column to remove min and seasons string, so that we can use the informa
```

In [39]:

```python
df['duration']= df['duration'].str.split(' ').apply(lambda x: x[0])
df.head(20)
```

Out[39]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duratic
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	missing	United States	2021-09-25	2020	PG-13	
1	s2	TV Show	Blood & Water	missing	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	
		TV		Julien	Sami Bouajila, Tracy	United			TV-	

In [41]:

```
df['duration'] = df['duration'].astype(int)
```

In [42]:

```
df.info()
```

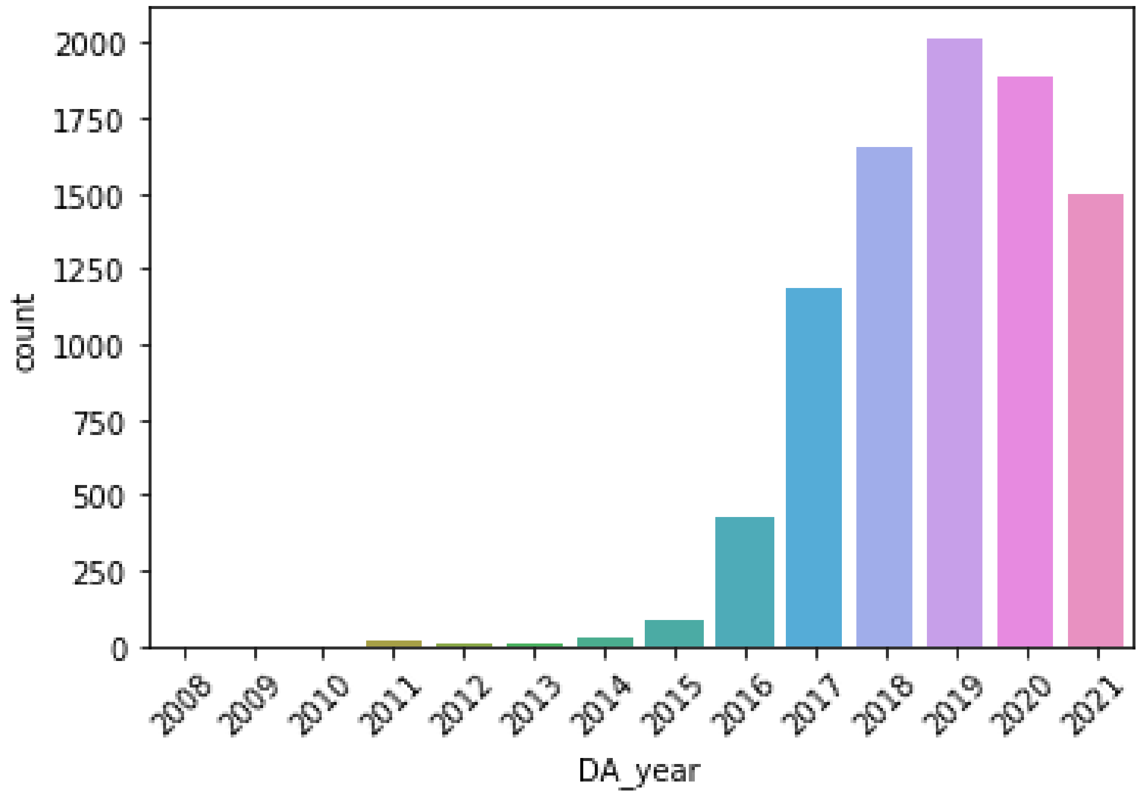
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype         
---  ------        --------------  -----         
 0   show_id       8807 non-null   object        
 1   type          8807 non-null   object        
 2   title         8807 non-null   object        
 3   director      8807 non-null   object        
 4   cast          8807 non-null   object        
 5   country       8807 non-null   object        
 6   date_added    8807 non-null   datetime64[ns]
 7   release_year  8807 non-null   int64         
 8   rating        8807 non-null   object        
 9   duration      8807 non-null   int32         
 10  listed_in     8807 non-null   object        
 11  description   8807 non-null   object        
 12  DA_year       8807 non-null   int64         
 13  DA_month      8807 non-null   object        
dtypes: datetime64[ns](1), int32(1), int64(2), object(10)
memory usage: 929.0+ KB
```

In [161]:

```
# Univariate and bivariate analysis
```

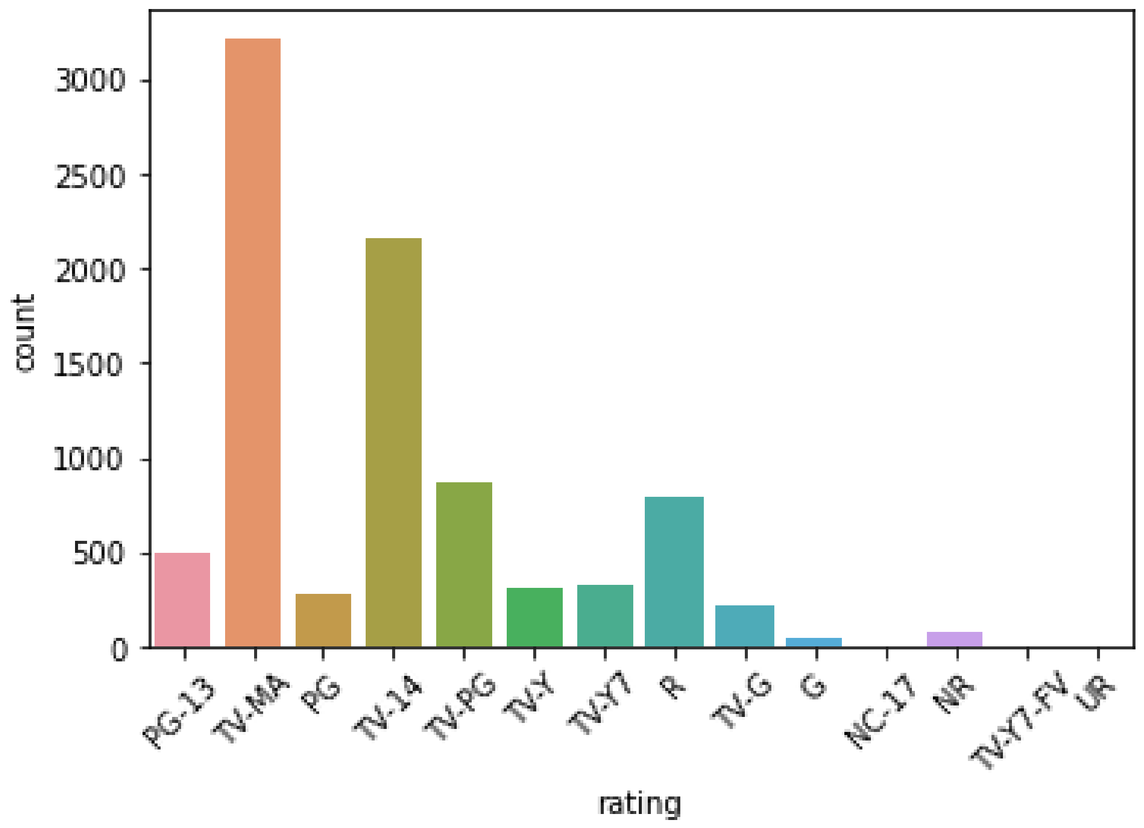
In [162]:

```
sns.countplot(x = 'DA_year', data = df)
plt.xticks(rotation = 45)
plt.show()
```



In [163]:

```
sns.countplot(x = 'rating', data = df)
plt.xticks(rotation = 45)
plt.show()
```



In [164]:

```
# creating different dataframe for movies and TV shows
```

In [165]:

```
df_movies = df.loc[df['type'] == 'Movie']
df_TV = df.loc[df['type']=='TV Show']
```

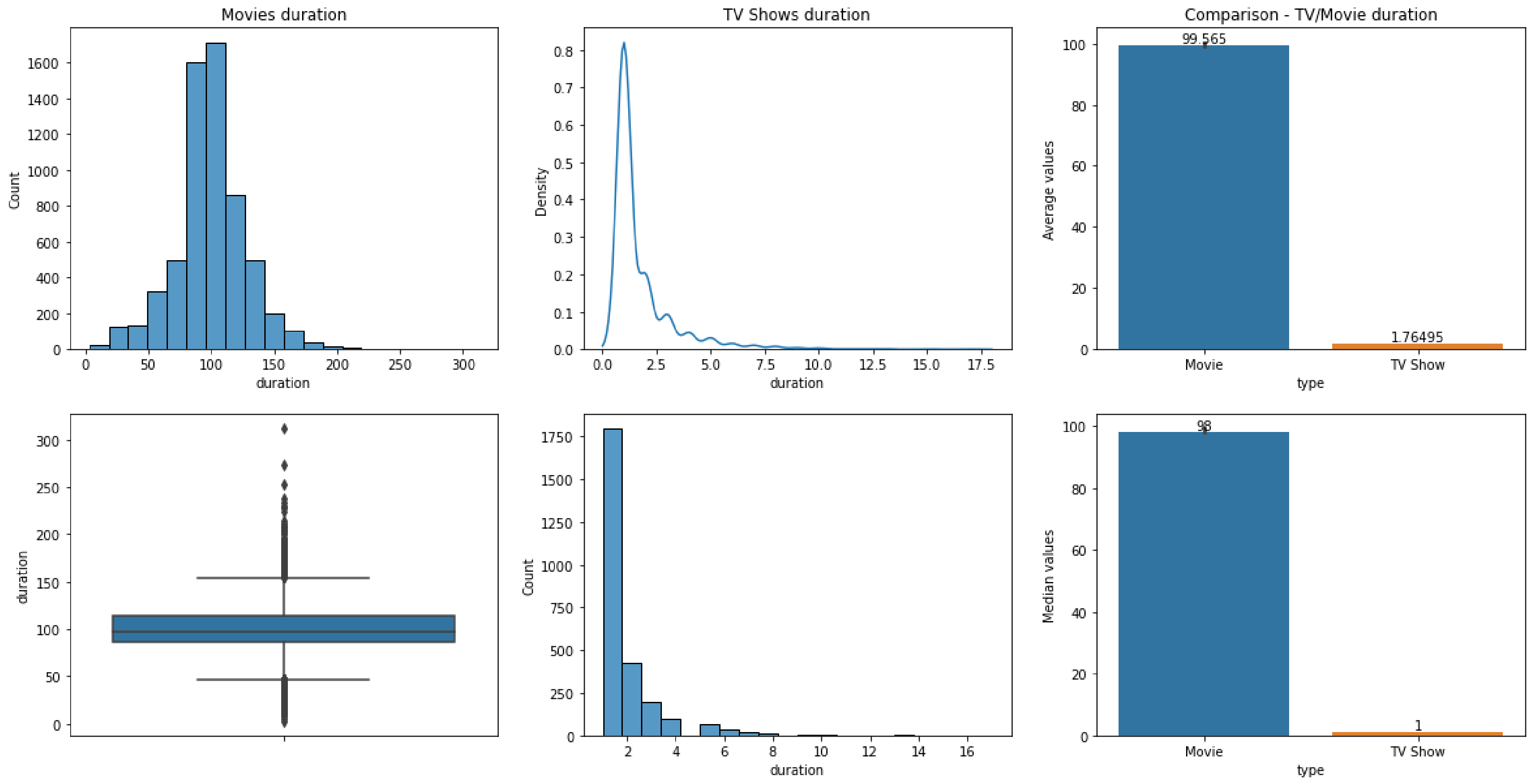
In [166]:

```
fig = plt.figure(figsize = (20,10))
plt.subplot(2,3,1)
sns.histplot(df_movies['duration'], bins = 20)
plt.title('Movies duration')

plt.subplot(2,3,4)
sns.boxplot(y= 'duration', data = df_movies)

plt.subplot(2,3,2)
sns.kdeplot(df_TV['duration'])
plt.title('TV Shows duration')

plt.subplot(2,3,5)
sns.histplot(df_TV['duration'], bins = 20)

plt.subplot(2,3,3)
ax = sns.barplot(x = 'type', y = 'duration', data = df)
ax.bar_label(ax.containers[0])
plt.ylabel('Average values')
plt.title('Comparison - TV/Movie duration')

plt.subplot(2,3,6)
ax = sns.barplot(x = 'type', y = 'duration', data = df, estimator = np.median)
ax.bar_label(ax.containers[0])
```

```
plt.ylabel('Median values')
```

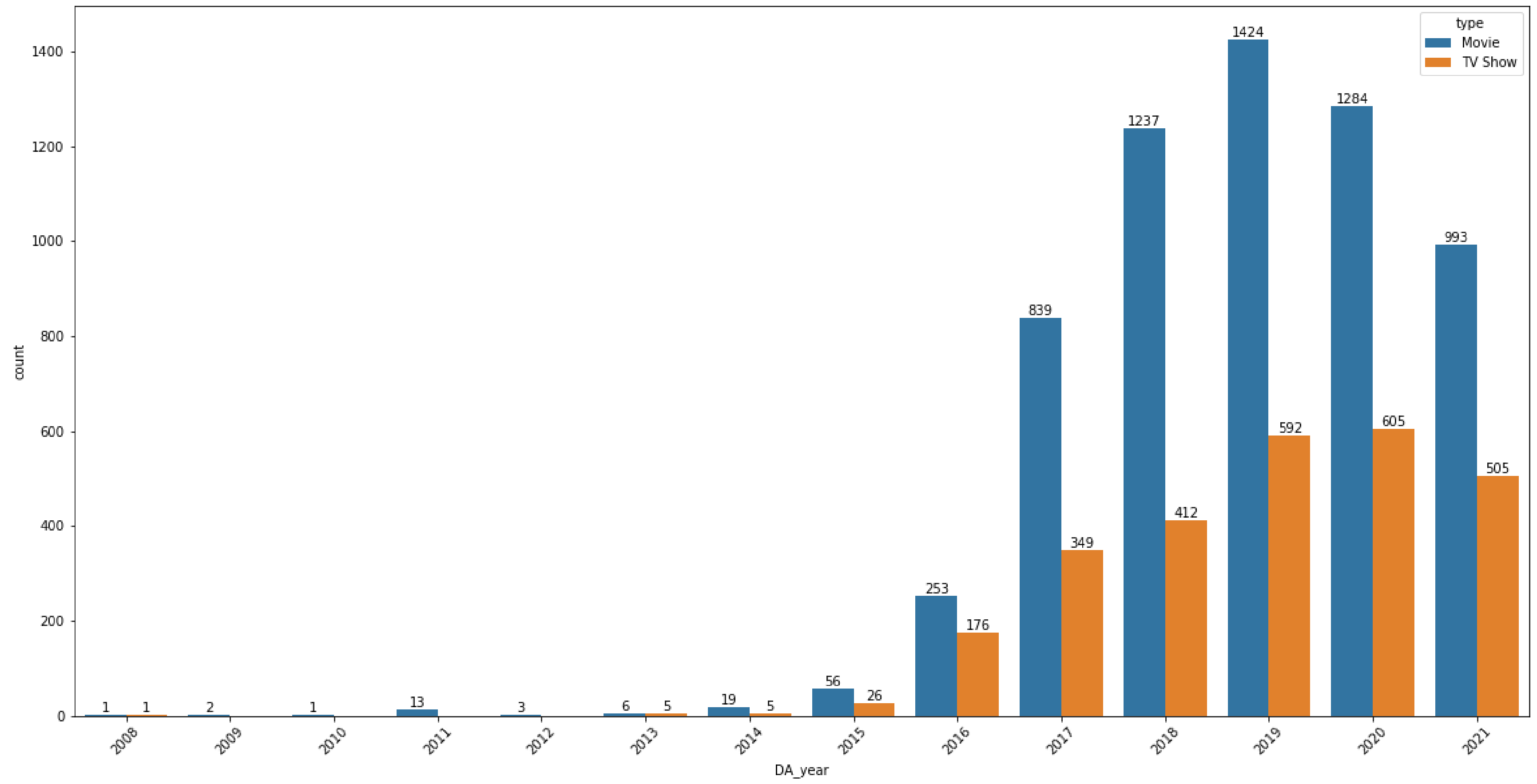
Out[166]:

```
Text(0, 0.5, 'Median values')
```



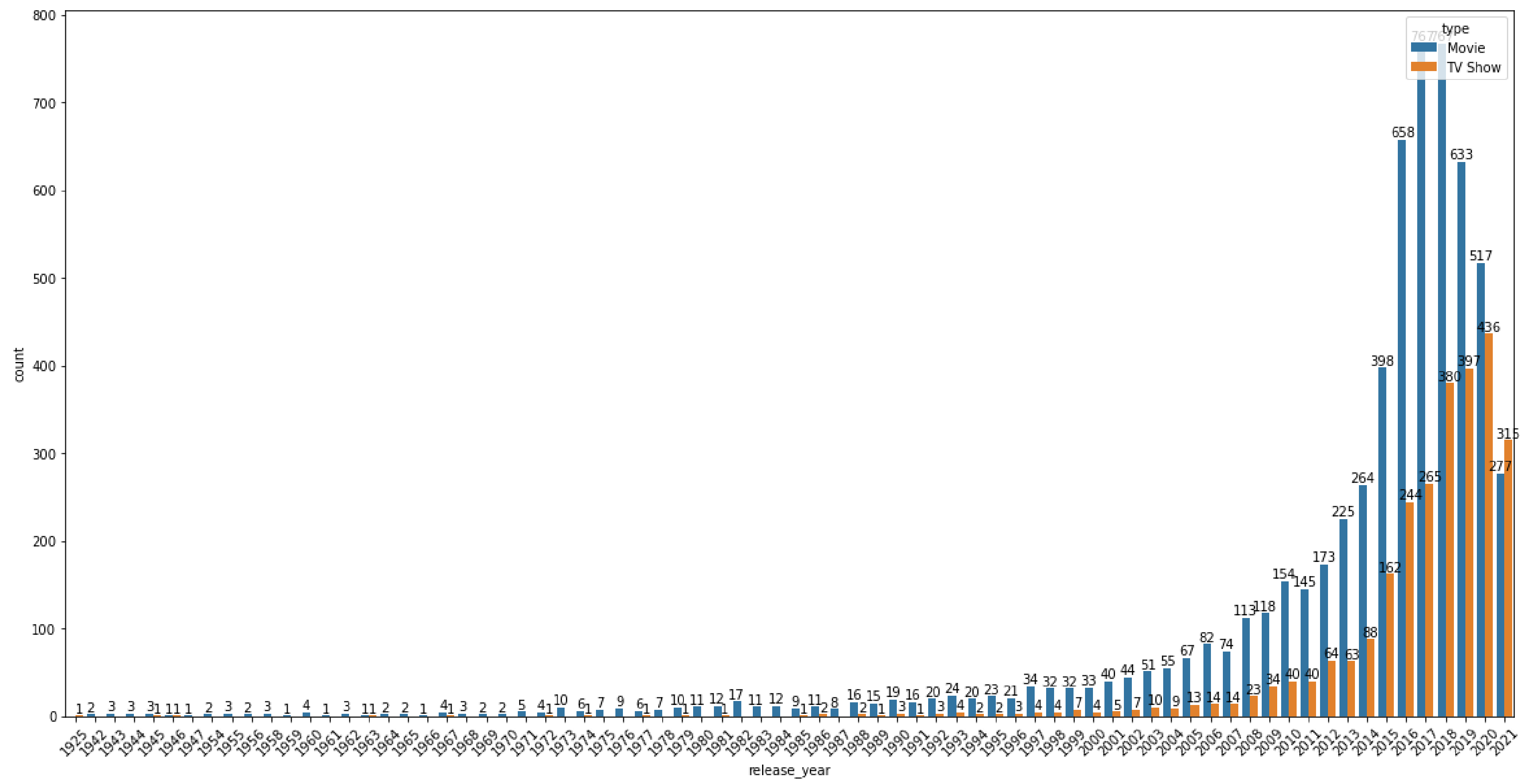
In [56]:

```
fig = plt.figure(figsize = (20,10))
ax = sns.countplot(x= 'DA_year', data = df, hue = 'type')
for i in ax.containers:
    ax.bar_label(i,)
plt.xticks(rotation = 45)
plt.show()
```



In [57]:

```
fig = plt.figure(figsize = (20,10))
ax = sns.countplot(x= 'release_year', data = df, hue = 'type')
for i in ax.containers:
    ax.bar_label(i,)
plt.xticks(rotation = 45)
plt.show()
```



In [81]:

```
df['date_added'].min()
```

Out[81]:

```
Timestamp('2008-01-01 00:00:00')
```

In [82]:

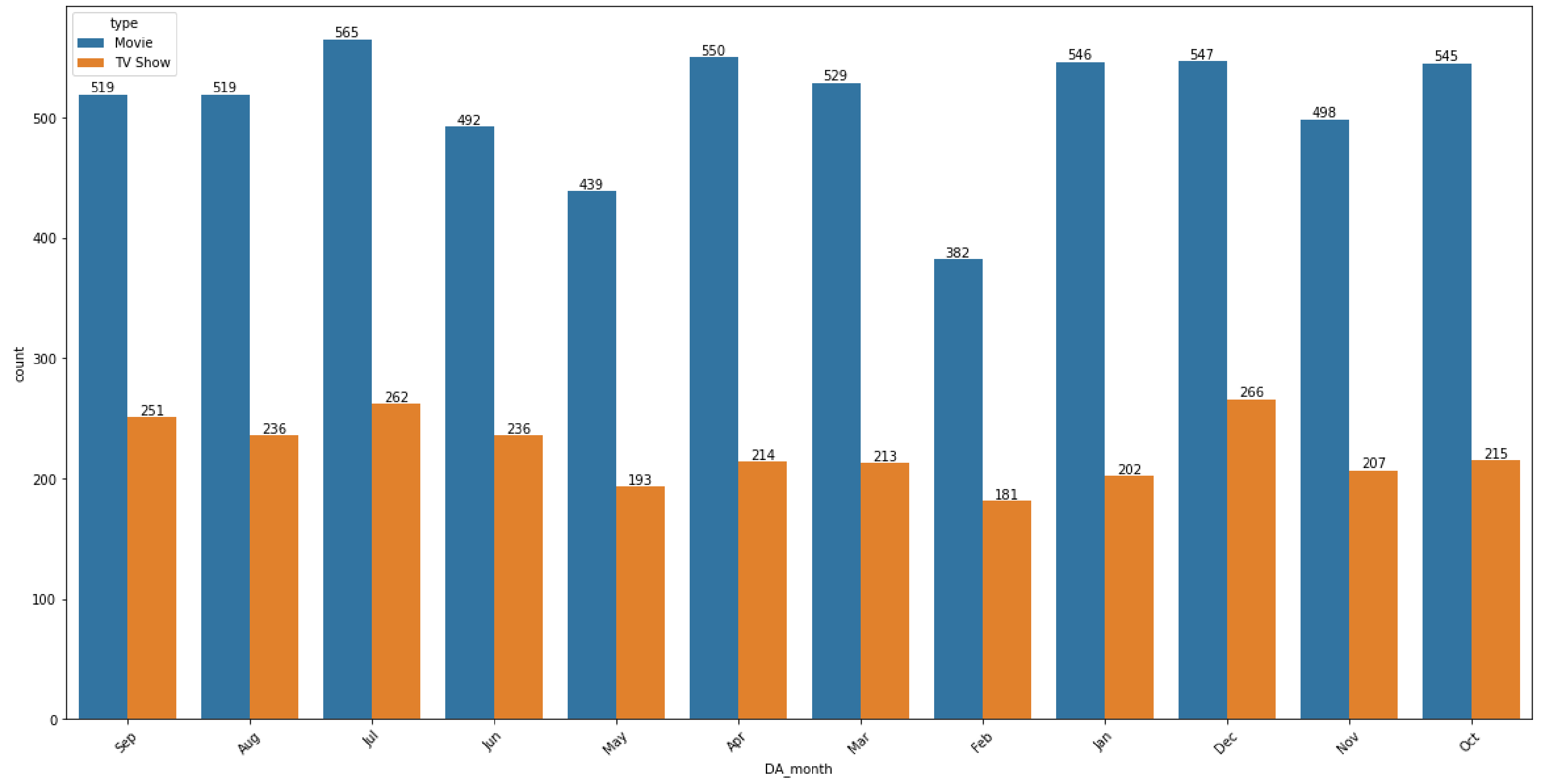
```
df['date_added'].max()
```

Out[82]:

```
Timestamp('2021-09-25 00:00:00')
```

In [60]:

```
fig = plt.figure(figsize = (20,10))
ax = sns.countplot(x= 'DA_month', data = df, hue = 'type')
for i in ax.containers:
    ax.bar_label(i,)
plt.xticks(rotation = 45)
plt.show()
```



In [ ]:

```
# To see if there is a lag in the movies or TV shows production and the time that they are
```

In [43]:

```
df_cut = df.loc[df['release_year']>= 2015]
df_cut.head()
```

Out[43]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	c
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	missing	United States	2021-09-25	2020	PG-13	
1	s2	TV Show	Blood & Water	missing	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	2021	TV-MA	
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	2021	TV-MA	
3	s4	TV Show	Jailbirds New Orleans	missing	missing	United States	2021-09-24	2021	TV-MA	

	show_id	type	title	director	cast	country	date_added	release_year	rating	c
4	s5	TV Show	Kota Factory	missing	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-24	2021	TV-MA	

In [44]:

```
df_cut['lag'] = df_cut['DA_year']-df_cut['release_year']
df_cut['lag'].value_counts()
```

```
C:\Users\91981\AppData\Local\Temp\ipykernel_7456\2803299873.py:1: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  df_cut['lag'] = df_cut['DA_year']-df_cut['release_year']
```

Out[44]:

```
0     3216
1     1562
2      684
3      394
4      207
5       98
6       41
-1      12
-2       1
-3       1
Name: lag, dtype: int64
```

In [46]:

```python
# Unnesting the data - cast column
```

In [47]:

```python
cast_split = df['cast'].apply(lambda x: str(x).split(',')).to_list()
```

In [48]:

```
df_new = pd.DataFrame(cast_split, index= df['title'])
df_new.head()
```

Out[48]:

	0	1	2	3	4	5	6	7
title								
Dick Johnson Is Dead	missing	None	None	None	None	None	None	Non
Blood & Water	Ama Qamata	Khosi Ngema	Gail Mabalane	Thabang Molaba	Dillon Windvogel	Natasha Thahane	Arno Greeff	Xolil Tshabalal
Ganglands	Sami Bouajila	Tracy Gotoas	Samuel Jouy	Nabiha Akkari	Sofia Lesaffre	Salim Kechiouche	Noureddine Farihi	Geert Va Rampelber
Jailbirds New Orleans	missing	None	None	None	None	None	None	Non
Kota Factory	Mayur More	Jitendra Kumar	Ranjan Raj	Alam Khan	Ahsaas Channa	Revathi Pillai	Urvi Singh	Arun Kuma

5 rows × 50 columns

In [49]:

```
df_new = df_new.stack()
```

In [50]:

```
df_new.head(30)
```

```
Out[50]:
```

```
title
Dick Johnson Is Dead    0                  missing
Blood & Water           0               Ama Qamata
                        1              Khosi Ngema
                        2            Gail Mabalane
                        3           Thabang Molaba
                        4         Dillon Windvogel
                        5         Natasha Thahane
                        6              Arno Greeff
                        7        Xolile Tshabalala
                        8          Getmore Sithole
                        9          Cindy Mahlangu
                        10           Ryle De Morny
                        11         Greteli Fincham
                        12    Sello Maake Ka-Ncube
                        13             Odwa Gwanya
                        14          Mekaila Mathys
                        15           Sandi Schultz
                        16          Duane Williams
                        17         Shamilla Miller
                        18        Patrick Mofokeng
Ganglands               0            Sami Bouajila
                        1             Tracy Gotoas
                        2             Samuel Jouy
                        3            Nabiha Akkari
                        4           Sofia Lesaffre
                        5         Salim Kechiouche
                        6        Noureddine Farihi
```

In [51]:

```python
df_new = pd.DataFrame(df_new)
df_new.reset_index(inplace = True)
df_new.head()
```

Out[51]:

	title	level_1	0
0	Dick Johnson Is Dead	0	missing
1	Blood & Water	0	Ama Qamata
2	Blood & Water	1	Khosi Ngema
3	Blood & Water	2	Gail Mabalane
4	Blood & Water	3	Thabang Molaba

```
                          7      Geert Van Rampelberg
                          8           Bakary Diombera
Jailbirds New Orleans     0                   missing
dtype: object
```

In [52]:

```
df_new.drop('level_1', axis = 1, inplace = True)
df_new.columns = ['title','cast']
```

In [53]:

```
df_new.shape
```

Out[53]:

```
(64951, 2)
```

In [54]:

```
df_new['cast'].value_counts()[:10]
```

Out[54]:

```
missing                825
 Anupam Kher            39
 Rupa Bhimani           31
 Takahiro Sakurai       30
 Julie Tejwani          28
 Om Puri                27
Shah Rukh Khan          26
 Rajesh Kava            26
 Andrea Libman          25
 Paresh Rawal           25
Name: cast, dtype: int64
```

In [55]:

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

Out[55]:

```
39297
```

In [56]:

```
# Unnesting the data - Director column
```

In [57]:

```
dir_split = df['director'].apply(lambda x: str(x).split(',')).to_list()
df_dir = pd.DataFrame(dir_split,index = df['title'])
df_dir = df_dir.stack()
df_dir = pd.DataFrame(df_dir)
```

In [58]:

```
df_dir.head(20)
```

Out[58]:

		0
title		
Dick Johnson Is Dead	0	Kirsten Johnson
Blood & Water	0	missing
Ganglands	0	Julien Leclercq
Jailbirds New Orleans	0	missing
Kota Factory	0	missing
Midnight Mass	0	Mike Flanagan
My Little Pony: A New Generation	0	Robert Cullen
	1	José Luis Ucha
Sankofa	0	Haile Gerima
The Great British Baking Show	0	Andy Devonshire
The Starling	0	Theodore Melfi
Vendetta: Truth, Lies and The Mafia	0	missing
Bangkok Breaking	0	Kongkiat Komesiri
Je Suis Karl	0	Christian Schwochow
Confessions of an Invisible Girl	0	Bruno Garotti

		0
title		
Crime Stories: India Detectives	**0**	missing
Dear White People	**0**	missing
Europe's Most Dangerous Man: Otto Skorzeny in Spain	**0**	Pedro de Echave García
	1	Pablo Azorín Williams
Falsa identidad	**0**	missing

In [59]:

```
df_dir = pd.DataFrame(df_dir)
df_dir.reset_index(inplace=True)
df_dir.head()
```

Out[59]:

	title	level_1	0
0	Dick Johnson Is Dead	0	Kirsten Johnson
1	Blood & Water	0	missing
2	Ganglands	0	Julien Leclercq
3	Jailbirds New Orleans	0	missing
4	Kota Factory	0	missing

In [60]:

```
df_dir.drop('level_1', axis= 1, inplace = True)
```

In [61]:

```
df_dir.shape
```

Out[61]:

```
(9612, 2)
```

In [62]:

```
df_dir.columns = ['title','director']
```

In [63]:

```
df_dir['director'].value_counts()[:10]
```

Out[63]:

```
missing                2634
Rajiv Chilaka            22
 Jan Suter               18
Raúl Campos              18
Marcus Raboy             16
Suhas Kadav              16
Jay Karas                15
Cathy Garcia-Molina      13
Martin Scorsese          12
Jay Chapman              12
Name: director, dtype: int64
```

In [64]:

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

Out[64]:

```
5121
```

In [65]:

```
# unnesting - Listed_in variable
```

In [66]:

```
gen_split = df['listed_in'].apply(lambda x: str(x).split(',')).to_list()
df_gen = pd.DataFrame(gen_split,index = df['title'])
df_gen = df_gen.stack()
df_gen = pd.DataFrame(df_gen)
df_gen.reset_index(inplace = True)
df_gen.drop('level_1', axis = 1, inplace = True)
df_gen.columns = ['title', 'listed_in']
```

```
In [67]:
```

```
df_gen.head()
```

```
Out[67]:
```

	title	listed_in
0	Dick Johnson Is Dead	Documentaries
1	Blood & Water	International TV Shows
2	Blood & Water	TV Dramas
3	Blood & Water	TV Mysteries
4	Ganglands	Crime TV Shows

In [68]:

```
df_gen['listed_in'].value_counts()[:10]
```

Out[68]:

```
 International Movies     2624
Dramas                    1600
Comedies                  1210
Action & Adventure         859
Documentaries              829
 Dramas                    827
International TV Shows     774
 Independent Movies        736
 TV Dramas                 696
 Romantic Movies           613
Name: listed_in, dtype: int64
```

In [69]:

```
df_gen.shape
```

Out[69]:

```
(19323, 2)
```

In [70]:

```
df_gen['listed_in'].nunique()
```

Out[70]:

```
73
```

In [71]:

```
# Unnesting country column
```

In [72]:

```
c_split = df['country'].apply(lambda x: str(x).split(',')).to_list()
df_c = pd.DataFrame(c_split,index = df['title'])
df_c = df_c.stack()
df_c = pd.DataFrame(df_c)
df_c.reset_index(inplace = True)
df_c.drop('level_1', axis = 1, inplace = True)
df_c.columns = ['title', 'country']
```

In [73]:

```
df_c.head()
```

Out[73]:

	title	country
0	Dick Johnson Is Dead	United States
1	Blood & Water	South Africa
2	Ganglands	United States
3	Jailbirds New Orleans	United States
4	Kota Factory	India

In [74]:

```
df_c['country'].value_counts()[:10]
```

Out[74]:

```
United States     4042
India             1008
United Kingdom     628
 United States     479
Canada             271
Japan              259
France             212
South Korea        211
Spain              181
 France            181
Name: country, dtype: int64
```

In [75]:

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

Out[75]:

```
197
```

In [76]:

```
df_c.shape
```

Out[76]:

```
(10850, 2)
```

In [77]:

```
# Merging the datasets to the final dataset - for this we will have to drop the nested colu
# and director from the original dataset
```

In [78]:

```
df_st1 =df
df_st1.drop(['cast','country','director','listed_in'], axis=1, inplace = True)
```

In [79]:

```
df_st1.head()
```

```
Out[79]:
```

	show_id	type	title	date_added	release_year	rating	duration	description	DA_year
0	s1	Movie	Dick Johnson Is Dead	2021-09-25	2020	PG-13	90	As her father nears the end of his life, filmm...	2021
1	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021
2	s3	TV Show	Ganglands	2021-09-24	2021	TV-MA	1	To protect his family from a powerful drug lor...	2021
3	s4	TV Show	Jailbirds New Orleans	2021-09-24	2021	TV-MA	1	Feuds, flirtations and toilet talk go down amo...	2021

	show_id	type	title	date_added	release_year	rating	duration	description	DA_year
4	s5	TV Show	Kota Factory	2021-09-24	2021	TV-MA	2	In a city of coaching centers known to train l...	2021

In [80]:

```
# Merge df_new to df_st1 (Adding cast info)
```

In [81]:

```
df_st2 = df_st1.merge(df_new, on = 'title', how = 'left')
df_st2.shape
```

Out[81]:

```
(64951, 11)
```

In [82]:

```
# merge df_dir to df_st2 ( Adding director information)
```

In [83]:

```
df_st3 = df_st2.merge(df_dir, on = 'title', how = 'left')
df_st3.shape
```

Out[83]:

```
(70812, 12)
```

In [84]:

```
# merge df_c to df_st3 ( Adding country information)
```

In [85]:

```
df_st4 = df_st3.merge(df_c, on = 'title', how = 'left')
```

In [86]:

```
df_st4.shape
```

Out[86]:

```
(89415, 13)
```

In [87]:

```
# merge df_gen to df_st4 (Adding genre information)
```

In [88]:

```
final = df_st4.merge(df_gen, on = 'title', how = 'left')
```

In [89]:

```
final.shape
```

Out[89]:

```
(202065, 14)
```

In [90]:

```
final.drop_duplicates(inplace = True)
```

In [91]:

```
final.shape
```

Out[91]:

```
(202058, 14)
```

In [92]:

```
final.head()
```

Out[92]:

	show_id	type	title	date_added	release_year	rating	duration	description	DA_year
0	s1	Movie	Dick Johnson Is Dead	2021-09-25	2020	PG-13	90	As her father nears the end of his life, filmm...	2021
1	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021
2	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021
3	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021

	show_id	type	title	date_added	release_year	rating	duration	description	DA_year
4	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021

In [93]:

```
final.to_csv('C:/Users/91981/Desktop/Mansi/scaler/Python Visualisation/Netflix_final.csv',
```

In [94]:

```
# To understand key metrics for deriving business insights, we ddivide the final dataset in
```

In [95]:

```
movie_f = final.loc[final['type']=='Movie']
TV_f = final.loc[final['type']=='TV Show']
```

In [96]:

```
movie_f.head()
```

Out[96]:

	show_id	type	title	date_added	release_year	rating	duration	description	DA_ye
0	s1	Movie	Dick Johnson Is Dead	2021-09-25	2020	PG-13	90	As her father nears the end of his life, filmm...	20
159	s7	Movie	My Little Pony: A New Generation	2021-09-24	2021	PG	91	Equestria's divided. But a bright-eyed hero be...	20
160	s7	Movie	My Little Pony: A New Generation	2021-09-24	2021	PG	91	Equestria's divided. But a bright-eyed hero be...	20
161	s7	Movie	My Little Pony: A New Generation	2021-09-24	2021	PG	91	Equestria's divided. But a bright-eyed hero be...	20

	show_id	type	title	date_added	release_year	rating	duration	description	DA_ye
162	s7	Movie	My Little Pony: A New Generation	2021-09-24	2021	PG	91	Equestria's divided. But a bright-eyed hero be...	20

In [97]:

```
# Popular actor director pairs
```

In [98]:

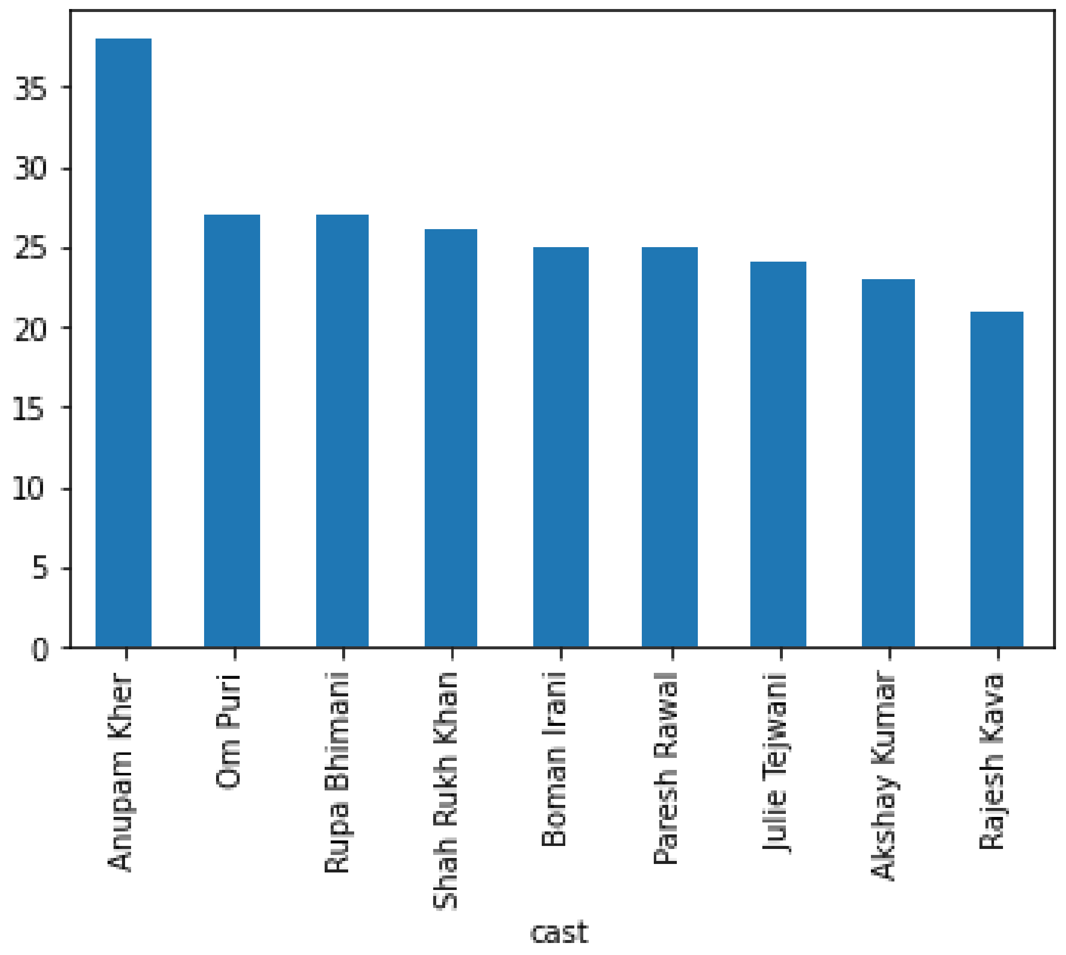
```
movie_f.groupby(['cast','director'])['title'].nunique().sort_values(ascending = False)[:10
```

Out[98]:

```
cast                  director
missing               missing          54
 Julie Tejwani        Rajiv Chilaka    19
 Rajesh Kava          Rajiv Chilaka    19
 Rupa Bhimani         Rajiv Chilaka    18
 Jigna Bhardwaj       Rajiv Chilaka    18
Vatsal Dubey          Rajiv Chilaka    16
 Swapnil              Rajiv Chilaka    13
 Mousam               Rajiv Chilaka    13
David Spade           missing          11
 Fortune Feimster     missing          11
Name: title, dtype: int64
```

In [161]:

```
m_cast = movie_f.groupby(['cast'])['title'].nunique().sort_values(ascending = False)
m_cast[1:10].plot(kind = 'bar')
plt.show()
```



```
In [144]:
```

```
movie_f.groupby(['director'])['title'].nunique().sort_values(ascending = False)[:10]
```

```
Out[144]:

director
missing                188
Rajiv Chilaka           22
 Jan Suter              18
Raúl Campos             18
Suhas Kadav             16
Jay Karas               15
Marcus Raboy            15
Cathy Garcia-Molina     13
Martin Scorsese         12
Youssef Chahine         12
Name: title, dtype: int64
```

```
In [145]:
```

```
TV_f.groupby(['cast'])['title'].nunique().sort_values(ascending = False)[:10]
```

```
Out[145]:
```

```
cast
missing                   350
 Takahiro Sakurai          24
 Yuki Kaji                 17
 Junichi Suwabe            17
 Ai Kayano                 17
 Daisuke Ono               14
David Attenborough         14
 Takehito Koyasu           13
 Yoshimasa Hosoya          13
 Yuichi Nakamura           13
Name: title, dtype: int64
```

In [102]:

```
TV_f.groupby(['director'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[102]:

```
director
missing                      2446
Ken Burns                       3
Alastair Fothergill             3
 Gautham Vasudev Menon          2
Iginio Straffi                  2
Joe Berlinger                   2
Jung-ah Im                      2
Rob Seidenglanz                 2
Shin Won-ho                     2
Stan Lathan                     2
Name: title, dtype: int64
```

In [103]:

```
TV_f.groupby(['cast','director'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[103]:

```
cast                 director
missing              missing     298
 Takahiro Sakurai    missing      23
 Yuki Kaji           missing      16
 Junichi Suwabe      missing      16
 Ai Kayano           missing      15
 Yoshimasa Hosoya    missing      13
 Yuichi Nakamura     missing      13
 Daisuke Ono         missing      13
 Takehito Koyasu     missing      12
David Attenborough   missing      11
Name: title, dtype: int64
```

In [104]:

```
# Popular genre by type(movies/TV shows)
```

In [141]:

```
movie_f.groupby(['listed_in'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[141]:

```
listed_in
 International Movies       2624
Dramas                      1600
Comedies                    1210
Action & Adventure           859
Documentaries                829
 Dramas                      827
 Independent Movies          736
 Romantic Movies             613
Children & Family Movies     605
 Thrillers                   512
Name: title, dtype: int64
```

```
In [142]:
```

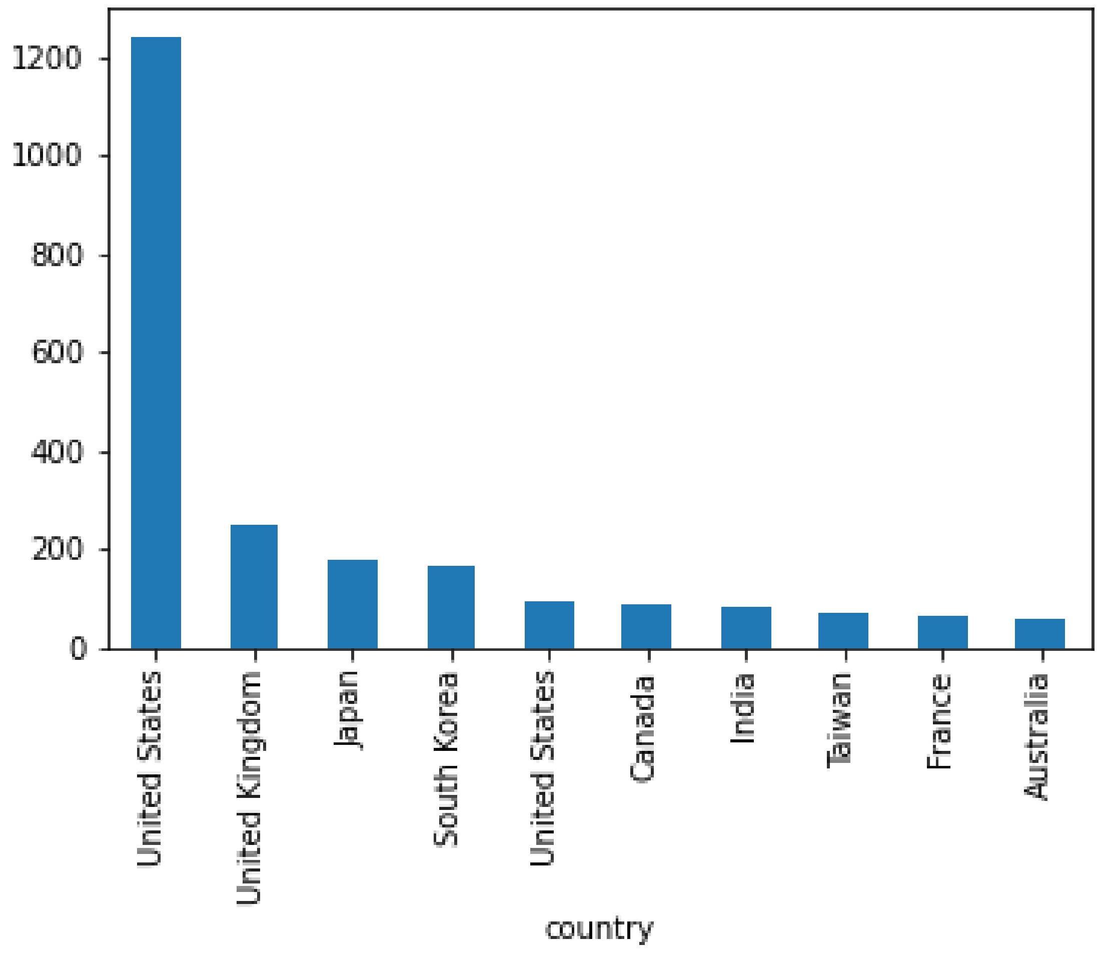
```python
TV_f.groupby(['listed_in'])['title'].nunique().sort_values(ascending = False)[:10]
```

```
Out[142]:
```

```
listed_in
International TV Shows     774
 TV Dramas                 696
 International TV Shows    577
 TV Comedies               461
Crime TV Shows             399
Kids' TV                   388
 Romantic TV Shows         338
British TV Shows           253
Docuseries                 221
Anime Series               176
Name: title, dtype: int64
```

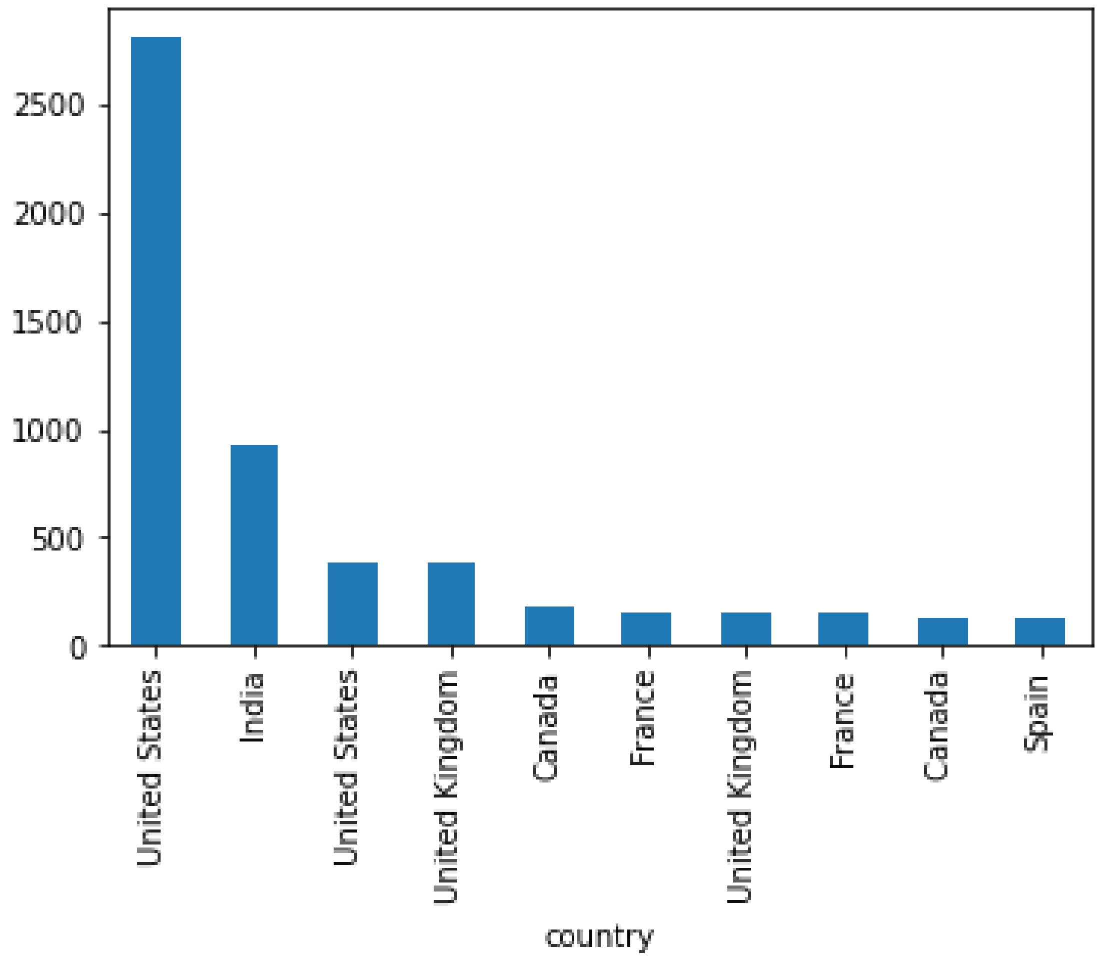
In [163]:

```
TV_f.groupby(['country'])['title'].nunique().sort_values(ascending = False)[:10].plot(kind
plt.show()
```



In [162]:

```
movie_f.groupby(['country'])['title'].nunique().sort_values(ascending = False)[:10].plot(k
plt.show()
```



```
In [109]:
```

```
movie_f.groupby(['DA_month'])['title'].nunique().sort_values(ascending = False)
```

```
Out[109]:
```

```
DA_month
Jul    565
Apr    550
Dec    547
Jan    546
Oct    545
Mar    529
Aug    519
Sep    519
Nov    498
Jun    492
May    439
Feb    382
Name: title, dtype: int64
```

In [110]:

```
TV_f.groupby(['DA_month'])['title'].nunique().sort_values(ascending = False)
```

Out[110]:

```
DA_month
Dec    266
Jul    262
Sep    251
Aug    236
Jun    236
Oct    215
Apr    214
Mar    213
Nov    207
Jan    202
May    193
Feb    181
Name: title, dtype: int64
```

In [111]:

```python
movie_f['DA_week'] = movie_f['date_added'].dt.isocalendar().week
TV_f['DA_week'] = TV_f['date_added'].dt.isocalendar().week
```

```
C:\Users\91981\AppData\Local\Temp\ipykernel_7456\548383386.py:1: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  movie_f['DA_week'] = movie_f['date_added'].dt.isocalendar().week
C:\Users\91981\AppData\Local\Temp\ipykernel_7456\548383386.py:2: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  TV_f['DA_week'] = TV_f['date_added'].dt.isocalendar().week
```

In [112]:

```
movie_f.head()
```

Out[112]:

	show_id	type	title	date_added	release_year	rating	duration	description	DA_ye
0	s1	Movie	Dick Johnson Is Dead	2021-09-25	2020	PG-13	90	As her father nears the end of his life, filmm...	20
159	s7	Movie	My Little Pony: A New Generation	2021-09-24	2021	PG	91	Equestria's divided. But a bright-eyed hero be...	20
160	s7	Movie	My Little Pony: A New Generation	2021-09-24	2021	PG	91	Equestria's divided. But a bright-eyed hero be...	20
161	s7	Movie	My Little Pony: A New Generation	2021-09-24	2021	PG	91	Equestria's divided. But a bright-eyed hero be...	20

	show_id	type	title	date_added	release_year	rating	duration	description	DA_ye
162	s7	Movie	My Little Pony: A New Generation	2021-09-24	2021	PG	91	Equestria's divided. But a bright-eyed hero be...	20

In [113]:

```
genre_counts = movie_f.groupby(['DA_week', 'listed_in']).size().reset_index(name='Count')

most_popular_genre = genre_counts.groupby('DA_week')['listed_in', 'Count'].apply(

lambda x: x.loc[x['Count'].idxmax()]

)
most_popular_genre
```

```
C:\Users\91981\AppData\Local\Temp\ipykernel_7456\4030375247.py:3: FutureWarn
ing: Indexing with multiple keys (implicitly converted to a tuple of keys) w
ill be deprecated, use a list instead.
  most_popular_genre = genre_counts.groupby('DA_week')['listed_in', 'Coun
t'].apply(
Out[113]:
```

	listed_in	Count
DA_week		
1	International Movies	1177
2	International Movies	314
3	International Movies	351
4	International Movies	266
5	International Movies	713
6	Comedies	254
7	International Movies	504
8	International Movies	359
9	International Movies	1071
10	International Movies	553
11	International Movies	476
12	International Movies	306
13	Dramas	675
14	International Movies	412
15	Comedies	669

	listed_in	Count
DA_week		
16	International Movies	549
17	International Movies	403
18	International Movies	805
19	International Movies	330
20	International Movies	383
21	International Movies	412
22	Dramas	562
23	International Movies	553
24	International Movies	452
25	International Movies	613
26	International Movies	877
27	International Movies	792
28	Dramas	466
29	International Movies	429
30	International Movies	422
31	International Movies	876

	listed_in	Count
DA_week		
32	International Movies	223
33	International Movies	437
34	International Movies	586
35	International Movies	837
36	International Movies	423
37	International Movies	529
38	International Movies	542
39	Children & Family Movies	717
40	International Movies	922
41	International Movies	455
42	International Movies	450
43	International Movies	503
44	International Movies	812
45	International Movies	288
46	International Movies	259
47	International Movies	206

```
In [114]:
TV_f.head()
```

DA_week	listed_in	Count
48	International Movies	940
49	International Movies	335
50	International Movies	494
51	Dramas	406
52	International Movies	291
53	Action & Adventure	218

Out[114]:

	show_id	type	title	date_added	release_year	rating	duration	description	DA_year	DA
1	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021	
2	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021	
3	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021	
4	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021	

	show_id	type	title	date_added	release_year	rating	duration	description	DA_year	DA
5	s2	TV Show	Blood & Water	2021-09-24	2021	TV-MA	2	After crossing paths at a party, a Cape Town t...	2021	

In [115]:

```
TV_counts = TV_f.groupby(['DA_week', 'listed_in']).size().reset_index(name='Count')

TV_popular_genre = TV_counts.groupby('DA_week')['listed_in', 'Count'].apply(

lambda x: x.loc[x['Count'].idxmax()]

)
TV_popular_genre
```

```
C:\Users\91981\AppData\Local\Temp\ipykernel_7456\3697046180.py:3: FutureWarn
ing: Indexing with multiple keys (implicitly converted to a tuple of keys) w
ill be deprecated, use a list instead.
  TV_popular_genre = TV_counts.groupby('DA_week')['listed_in', 'Count'].appl
y(
```

Out[115]:

	listed_in	Count
DA_week		
1	Anime Series	135
2	TV Dramas	149
3	International TV Shows	98
4	TV Dramas	147
5	Kids' TV	202
6	TV Dramas	117
7	TV Dramas	246
8	International TV Shows	155
9	International TV Shows	123
10	International TV Shows	124
11	International TV Shows	139
12	TV Dramas	164
13	International TV Shows	280
14	International TV Shows	169
15	TV Dramas	161

DA_week	listed_in	Count
16	International TV Shows	85
17	International TV Shows	142
18	TV Dramas	211
19	TV Dramas	336
20	International TV Shows	137
21	International TV Shows	138
22	TV Dramas	230
23	TV Dramas	149
24	TV Dramas	298
25	International TV Shows	206
26	TV Dramas	340
27	International TV Shows	319
28	International TV Shows	86
29	International TV Shows	88
30	TV Action & Adventure	91
31	International TV Shows	235

DA_week	listed_in	Count
32	International TV Shows	130
33	TV Dramas	267
34	TV Dramas	138
35	TV Dramas	307
36	TV Dramas	126
37	TV Dramas	144
38	TV Dramas	179
39	International TV Shows	136
40	British TV Shows	167
41	International TV Shows	136
42	TV Dramas	122
43	TV Mysteries	71
44	International TV Shows	182
45	International TV Shows	117
46	TV Dramas	167
47	International TV Shows	116

In [116]:

```
# Similar analysis for the top countries where Netflix has a strong presence like United St
```

	listed_in	Count
DA_week		
48	International TV Shows	220
49	International TV Shows	168
50	International TV Shows	239
51	TV Dramas	200
52	TV Dramas	157
53	Kids' TV	583

In [117]:

```
df_US = final.loc[final['country'].str.contains('United States|US')]
df_india = final.loc[final['country'].str.contains('India')]
df_UK = final.loc[final['country'].str.contains('United Kingdom|UK')]
```

In [118]:

```
df_US.shape
```

Out[118]:

```
(71246, 14)
```

In [119]:

```
df_india.shape
```

Out[119]:

```
(22814, 14)
```

In [120]:

```
df_UK.shape
```

Out[120]:

```
(12965, 14)
```

In [121]:

```
genre_counts = df_US.groupby(['DA_month', 'listed_in']).size().reset_index(name='Count')

most_popular_genre = genre_counts.groupby('DA_month')['listed_in', 'Count'].apply(

lambda x: x.loc[x['Count'].idxmax()]

)
most_popular_genre
```

```
C:\Users\91981\AppData\Local\Temp\ipykernel_7456\1146318166.py:3: FutureWarn
ing: Indexing with multiple keys (implicitly converted to a tuple of keys) w
ill be deprecated, use a list instead.
  most_popular_genre = genre_counts.groupby('DA_month')['listed_in', 'Coun
t'].apply(
```

Out[121]:

	listed_in	Count
DA_month		
Apr	Comedies	865
Aug	Dramas	561
Dec	Children & Family Movies	505
Feb	Dramas	442
Jan	Comedies	714
Jul	Children & Family Movies	687
Jun	Action & Adventure	579
Mar	Dramas	428
May	Dramas	408
Nov	Action & Adventure	522
Oct	Children & Family Movies	768
Sep	Action & Adventure	516

In [122]:

```
# Popular actor, director, genre in each country - US
```

In [146]:

```
df_US.groupby(['cast'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[146]:

```
cast
missing             561
 Rupa Bhimani        25
 Andrea Libman       22
 Fred Tatasciore     21
 Julie Tejwani       21
Adam Sandler         20
 Rajesh Kava         19
 Vincent Tong        18
 Jigna Bhardwaj      18
 Fortune Feimster    16
Name: title, dtype: int64
```

In [147]:

```
df_US.groupby(['director'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[147]:

```
director
missing              1349
Rajiv Chilaka          17
Marcus Raboy           16
Suhas Kadav            15
Jay Karas              15
Jay Chapman            12
Martin Scorsese        12
Steven Spielberg       11
Don Michael Paul       10
Shannon Hartman         9
Name: title, dtype: int64
```

In [148]:

```
df_US.groupby(['cast','director'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[148]:

```
cast                director
missing             missing          241
 Fortune Feimster   missing           15
 Rupa Bhimani       Rajiv Chilaka     15
 Julie Tejwani      Rajiv Chilaka     15
 Rajesh Kava        Rajiv Chilaka     15
 Jigna Bhardwaj     Rajiv Chilaka     15
Vatsal Dubey        Rajiv Chilaka     14
 Swapnil            Rajiv Chilaka     12
 Mousam             Rajiv Chilaka     12
 Vincent Tong       missing           11
Name: title, dtype: int64
```

```
In [159]:
```

```
df_US.groupby(['cast','type'])['title'].nunique().sort_values(ascending = False)[1:11]
```

```
Out[159]:
```

```
cast                 type
missing              TV Show    239
 Rupa Bhimani        Movie       23
Adam Sandler         Movie       20
 Julie Tejwani       Movie       19
 Rajesh Kava         Movie       17
 Jigna Bhardwaj      Movie       16
 Andrea Libman       Movie       15
 Fred Tatasciore     Movie       15
 Alfred Molina       Movie       15
 Molly Shannon       Movie       14
Name: title, dtype: int64
```

```
In [127]:
```

```
# Ananlysis for India
```

In [149]:

```
df_india.groupby(['cast','type'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[149]:

```
cast                type 
 Anupam Kher        Movie    36
 Om Puri            Movie    26
Shah Rukh Khan      Movie    25
 Boman Irani        Movie    25
 Paresh Rawal       Movie    25
Akshay Kumar        Movie    23
missing             Movie    20
 Naseeruddin Shah   Movie    20
 Kareena Kapoor     Movie    20
Amitabh Bachchan    Movie    20
Name: title, dtype: int64
```

In [150]:

```
df_india.groupby(['cast'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[150]:

```
cast
missing               39
 Anupam Kher          36
 Om Puri              26
 Paresh Rawal         25
Shah Rukh Khan        25
 Boman Irani          25
Akshay Kumar          23
 Naseeruddin Shah     20
 Kareena Kapoor       20
Amitabh Bachchan      20
Name: title, dtype: int64
```

In [151]:

```
df_india.groupby(['director'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[151]:

```
director
missing                 85
David Dhawan             9
Anurag Kashyap           7
Ram Gopal Varma          7
Sooraj R. Barjatya       6
Ashutosh Gowariker       6
Anees Bazmee             6
Imtiaz Ali               6
Rajkumar Santoshi        6
Priyadarshan             6
Name: title, dtype: int64
```

```
In [152]:
```

```
df_india.groupby(['cast','director'])['title'].nunique().sort_values(ascending = False)[:1
```

```
Out[152]:
```

```
cast               director
missing            missing                 18
 Anupam Kher       David Dhawan             6
 Alok Nath         Sooraj R. Barjatya       5
 Julie Tejwani     Rajiv Chilaka            4
 Rajesh Kava       Rajiv Chilaka            4
Salman Khan        Sooraj R. Barjatya       4
 Mohnish Bahl      Sooraj R. Barjatya       4
 Rajpal Yadav      Priyadarshan             4
 Asrani            Hrishikesh Mukherjee     3
 Rupa Bhimani      Rajiv Chilaka            3
Name: title, dtype: int64
```

In [132]:

```
df_india.groupby(['listed_in'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[132]:

```
listed_in
 International Movies    826
Dramas                   415
Comedies                 271
 Dramas                  247
 Independent Movies      166
Action & Adventure       137
 Romantic Movies         120
 Music & Musicals         96
 Thrillers                91
 Comedies                 52
Name: title, dtype: int64
```

In [133]:

```
# Analysis for UK
```

In [154]:

```
df_UK.groupby(['cast','type'])['title'].nunique().sort_values(ascending = False)[1:11]
```

Out[154]:

```
cast                    type
missing                 TV Show    39
David Attenborough      TV Show    13
 Michael Palin          Movie       9
 John Cleese            Movie       9
 Brendan Gleeson        Movie       8
 Helena Bonham Carter   Movie       8
 Terry Gilliam          Movie       7
 Terry Jones            Movie       7
 Eddie Marsan           Movie       7
 Judi Dench             Movie       7
Name: title, dtype: int64
```

```
In [155]:
```

```python
df_UK.groupby(['director'])['title'].nunique().sort_values(ascending = False)[:10]
```

```
Out[155]:
```

```
director
missing                    267
Alastair Fothergill          4
Edward Cotterill             4
Martin Campbell              3
Orlando von Einsiedel        3
Tom Hooper                   3
Terry Gilliam                3
Vince Marcello               3
Blair Simmons                3
Jerry Rothwell               3
Name: title, dtype: int64
```

In [156]:

```
df_UK.groupby(['cast','director'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[156]:

```
cast                director
missing             missing               43
David Attenborough  missing               13
 Terry Jones        missing                5
 Eric Idle          missing                5
 Michael Palin      missing                5
David Attenborough  Alastair Fothergill    4
 Brendan Coyle      missing                4
 Terry Gilliam      missing                4
 Molly Ringwald     Vince Marcello         3
missing             Jerry Rothwell         3
Name: title, dtype: int64
```

In [157]:

```
df_UK.groupby(['cast'])['title'].nunique().sort_values(ascending = False)[:10]
```

Out[157]:

```
cast
missing                  97
David Attenborough       17
 Michael Palin           14
 Eric Idle               12
 Terry Jones             12
 John Cleese             12
 Terry Gilliam           11
 Helena Bonham Carter     9
 Jim Broadbent            8
 Brendan Gleeson          8
Name: title, dtype: int64
```

In [ ]:

```
```