```
#importing libraries
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns


! gdown 1zrj16zATT0vIiQa4NVqxlGQUOs0DDCSt
```

```
Downloading...
From: https://drive.google.com/uc?id=1zrj16zATT0vIiQa4NVqxlGQUOs0DDCSt
To: /content/netflix.csv
100% 3.40M/3.40M [00:00<00:00, 213MB/s]
```

1. Defining Problem Statement and Analysing basic metrics

NETFLIX IS A MULTI-NATIONAL STREAMING COMPANY WHICH PRODUCES

- MOVIES AND TV WEB SERIES EVERY YEAR AND ALL AROUND THE GLOBE
- ANALYSING THE NETFLIX DATASET AND COMPARING INDIA WITH DIFFERENT COUNTRIES
- TAKE CONCLUSION THROUGH VISUAL AND DESCRIPTIVE ANALYSIS

```
df = pd.read_csv('/content/netflix.csv')
df.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating
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
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas,	NaN	September 24, 2021	2021	TV-MA

```
#length of data

len(df)
```

```
8807
```

2. Observations on the shape of data, data types of all the attributes, conversion of categorical attributes to 'category' (If required), missing value detection, statistical summary

```
df.columns
```

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

```
df.shape

#the dataset has 12 columns with 8807 rows
```

```
(8807, 12)
```

```
df.dtypes

# Here notice that all except release_year column all the remaining attributes are Objects. Which are nothing but Strings.
```

```
show_id         object
type            object
title           object
director        object
cast            object
country         object
date_added      object
release_year     int64
rating          object
duration        object
listed_in       object
description     object
dtype: object
```

```
# Showing for the above datatypes that they are indeed string values

col_type = []
for i in df.columns:
  col_type.append([i,type(i)])
for i in col_type:
   print(i)
```

```
['show_id', <class 'str'>]
['type', <class 'str'>]
['title', <class 'str'>]
['director', <class 'str'>]
['cast', <class 'str'>]
['country', <class 'str'>]
['date_added', <class 'str'>]
['release_year', <class 'str'>]
['rating', <class 'str'>]
['duration', <class 'str'>]
['listed_in', <class 'str'>]
['description', <class 'str'>]
```

```
#INFO

df.info()
```

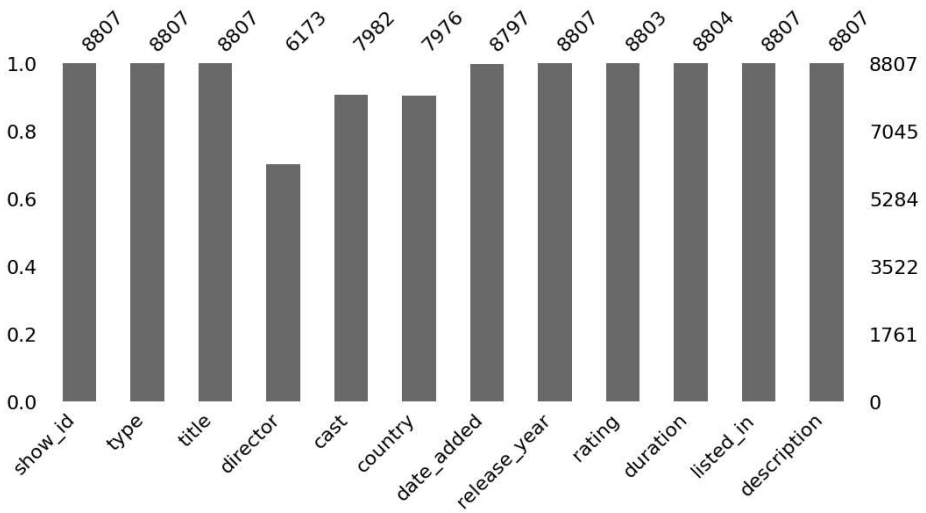
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 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
```

```
#Checking Missing Values

df1 = df.isnull().sum()
df1
```

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

```
description        0
dtype: int64
```

```
#Missing Values in bar plot

import missingno as msno
msno.bar(df, figsize=(12,5))
plt.show()
```




```
#Statistical Summary--by default gives for int datatype only

df.describe()
```

	release_year
count	8807.000000
mean	2014.180198
std	8.819312
min	1925.000000
25%	2013.000000
50%	2017.000000
75%	2019.000000
max	2021.000000

```
#Statistical Summary including object datatypes in the dataset

df.describe(include=[np.object])
```

```
<ipython-input-69-554b7518cb2b>:1: DeprecationWarning: `np.object` is a deprecated alias fo
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/
  df.describe(include=[np.object])
```

```
#Statistical Summary including the complete dataset

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

	show_id	type	title	director	cast	country	date_added	release_year	r
count	8807	8807	8807	6173	7982	7976	8797	8807.000000	
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	
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2014.180198	
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.819312	
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1925.000000	
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2013.000000	
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2017.000000	
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2019.000000	
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2021.000000	

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

```
#GIves bottom 5 data from the datase

df.tail()
```

	show_id	type	title	director	cast	country	date_added	release_year	ra
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	November 20, 2019	2007	
8803	s8804	TV Show	Zombie Dumb	NaN	NaN	NaN	July 1, 2019	2018	T
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson,	United States	November 1, 2019	2009	

Value Counts

```
#Value count for type

df['type'].value_counts().reset_index()
```

	index	type
0	Movie	6131
1	TV Show	2676

```
#Value count for title

df['title'].value_counts().reset_index()
```

	index	title
0	Dick Johnson Is Dead	1
1	Ip Man 2	1
2	Hannibal Buress: Comedy Camisado	1
3	Turbo FAST	1
4	Masha's Tales	1
...	...	...
8802	Love for Sale 2	1
8803	ROAD TO ROMA	1
8804	Good Time	1
8805	Captain Underpants Epic Choice-o-Rama	1
8806	Zubaan	1

8807 rows × 2 columns

```
#Value count for director

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

	index	director
0	Rajiv Chilaka	19
1	Raúl Campos, Jan Suter	18
2	Marcus Raboy	16
3	Suhas Kadav	16
4	Jay Karas	14
...	...	...
4523	Raymie Muzquiz, Stu Livingston	1
4524	Joe Menendez	1
4525	Eric Bross	1
4526	Will Eisenberg	1
4527	Mozez Singh	1

4528 rows × 2 columns

```
#Value count for cast

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

	index	cast
0	David Attenborough	19
1	Vatsal Dubey, Julie Tejwani, Rupa Bhimani, Jig...	14
2	Samuel West	10
3	Jeff Dunham	7
4	David Spade, London Hughes, Fortune Feimster	6
...	...	...
7687	Michael Peña, Diego Luna, Tenoch Huerta, Joaqu...	1
7688	Nick Lachey, Vanessa Lachey	1
7689	Takeru Sato, Kasumi Arimura, Haru, Kentaro Sak...	1
7690	Toyin Abraham, Sambasa Nzeribe, Chioma Chukwuk...	1
7691	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	1

7692 rows × 2 columns

```
#Value count for country

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

	index	country
0	United States	2818
1	India	972
2	United Kingdom	419
3	Japan	245
4	South Korea	199
...	...	...
743	Romania, Bulgaria, Hungary	1
744	Uruguay, Guatemala	1
745	France, Senegal, Belgium	1
746	Mexico, United States, Spain, Colombia	1
747	United Arab Emirates, Jordan	1

748 rows × 2 columns

```
#Value count for date_added

df['date_added'].value_counts().reset_index()
```

	index	date_added
0	January 1, 2020	109
1	November 1, 2019	89
2	March 1, 2018	75
3	December 31, 2019	74
4	October 1, 2018	71
...	...	...
1762	December 4, 2016	1
1763	November 21, 2016	1
1764	November 19, 2016	1
1765	November 17, 2016	1
1766	January 11, 2020	1

1767 rows × 2 columns

```
#Value count for release_year

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

	index	release_year
0	2018	1147

```
#Value count for rating

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

	index	rating
0	TV-MA	3207
1	TV-14	2160
2	TV-PG	863
3	R	799
4	PG-13	490
5	TV-Y7	334
6	TV-Y	307
7	PG	287
8	TV-G	220
9	NR	80
10	G	41
11	TV-Y7-FV	6
12	NC-17	3
13	UR	3
14	74 min	1
15	84 min	1
16	66 min	1

```
#Value count for duration

df['duration'].value_counts().reset_index()
```

	index	duration
0	1 Season	1793
1	2 Seasons	425
2	3 Seasons	199
3	90 min	152
4	94 min	146
...	...	...
215	16 min	1
216	186 min	1
217	193 min	1
218	189 min	1
219	191 min	1

220 rows × 2 columns

```
#Value count for listed_in

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

	index	listed_in
0	Dramas, International Movies	362
1	Documentaries	359
2	Stand-Up Comedy	334
3	Comedies, Dramas, International Movies	274
4	Dramas, Independent Movies, International Movies	252
...	...	...
509	Kids' TV, TV Action & Adventure, TV Dramas	1
510	TV Comedies, TV Dramas, TV Horror	1

```
#Value count for description

df['description'].value_counts().reset_index()
```

	index	description
0	Paranormal activity at a lush, abandoned prope...	4
1	Challenged to compose 100 songs before he can ...	3
2	A surly septuagenarian gets another chance at ...	3
3	Multiple women report their husbands as missin...	3
4	Secrets bubble to the surface after a sensual ...	2
...	...	...
8770	Sent away to evade an arranged marriage, a 14-...	1
8771	When his partner in crime goes missing, a smal...	1
8772	During 1962's Cuban missile crisis, a troubled...	1
8773	A teen's discovery of a vintage Polaroid camer...	1
8774	A scrappy but poor boy worms his way into a ty...	1

8775 rows × 2 columns

```
# the date_added column datatype is object, and we need to change it into datetime datatype

df['date_added'] = pd.to_datetime(df['date_added'])
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   datetime64[ns]
 7   release_year  8807 non-null   int64
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: datetime64[ns](1), int64(1), object(10)
memory usage: 825.8+ KB
```

Unique Values

```
# As the show_id column is unique, lets start with type column

print(df.type.unique())
print(df.type.nunique())
```

```
['Movie' 'TV Show']
2
```

```
# using simple for loop for all the columns

for i in df.columns:
  print('The Unique values and nunique values for', i,'are',df[i].unique(),'number of uniue values :',df[i].nunique())
  print('****************************************************************************************************')
```

```
The Unique values and nunique values for show_id are ['s1' 's2' 's3' ... 's8805' 's8806' 's8807'] number of uniue values : 8807
****************************************************************************************************
The Unique values and nunique values for type are ['Movie' 'TV Show'] number of uniue values : 2
****************************************************************************************************
The Unique values and nunique values for title are ['Dick Johnson Is Dead' 'Blood & Water' 'Ganglands' ... 'Zombieland'
 'Zoom' 'Zubaan'] number of uniue values : 8807
****************************************************************************************************
The Unique values and nunique values for director are ['Kirsten Johnson' nan 'Julien Leclercq' ... 'Majid Al Ansari'
 'Peter Hewitt' 'Mozez Singh'] number of uniue values : 4528
****************************************************************************************************
The Unique values and nunique values for cast are [nan
 'Ama Qamata, Khosi Ngema, Gail Mabalane, Thabang Molaba, Dillon Windvogel, Natasha Thahane, Arno Greeff, Xolile Tshabalala, Getmore
 'Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabiha Akkari, Sofia Lesaffre, Salim Kechiouche, Noureddine Farihi, Geert Van Rampelberg,
 ...
 'Jesse Eisenberg, Woody Harrelson, Emma Stone, Abigail Breslin, Amber Heard, Bill Murray, Derek Graf'
 'Tim Allen, Courteney Cox, Chevy Chase, Kate Mara, Ryan Newman, Michael Cassidy, Spencer Breslin, Rip Torn, Kevin Zegers'
 'Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanana, Manish Chaudhary, Meghna Malik, Malkeet Rauni, Anita Shabdish, Chittaranjan Tripat
****************************************************************************************************
The Unique values and nunique values for country are ['United States' 'South Africa' nan 'India'
 'United States, Ghana, Burkina Faso, United Kingdom, Germany, Ethiopia'
 'United Kingdom' 'Germany, Czech Republic' 'Mexico' 'Turkey' 'Australia'
 'United States, India, France' 'Finland' 'China, Canada, United States'
 'South Africa, United States, Japan' 'Nigeria' 'Japan'
 'Spain, United States' 'France' 'Belgium' 'United Kingdom, United States'
 'United States, United Kingdom' 'France, United States' 'South Korea'
 'Spain' 'United States, Singapore' 'United Kingdom, Australia, France'
 'United Kingdom, Australia, France, United States'
 'United States, Canada' 'Germany, United States'
 'South Africa, United States' 'United States, Mexico'
 'United States, Italy, France, Japan'
 'United States, Italy, Romania, United Kingdom'
 'Australia, United States' 'Argentina, Venezuela'
 'United States, United Kingdom, Canada' 'China, Hong Kong' 'Russia'
 'Canada' 'Hong Kong' 'United States, China, Hong Kong'
 'Italy, United States' 'United States, Germany'
 'United Kingdom, Canada, United States' ', South Korea' 'Ireland'
 'India, Nepal' 'New Zealand, Australia, France, United States' 'Italy'
 'Italy, Brazil, Greece' 'Argentina' 'Jordan' 'Colombia'
 'United States, Japan' 'Belgium, United Kingdom'
 'Switzerland, United Kingdom, Australia' 'Israel, United States'
 'Canada, United States' 'Brazil' 'Argentina, Spain' 'Taiwan'
 'United States, Nigeria' 'Bulgaria, United States'
 'Spain, United Kingdom, United States' 'United States, China'
 'United States, France' 'Spain, France, United Kingdom, United States'
 ', France, Algeria' 'Poland' 'Germany'
 'France, Israel, Germany, United States, United Kingdom' 'New Zealand'
 'Saudi Arabia' 'Thailand' 'Indonesia' 'Egypt, Denmark, Germany'
 'United States, Switzerland' 'Hong Kong, Canada, United States'
 'Kuwait, United States' 'France, Canada, United States, Spain'
 'France, Netherlands, Singapore' 'France, Belgium'
 'Ireland, United States, United Kingdom' 'Egypt' 'Malaysia' 'Israel'
 'Australia, New Zealand' 'United Kingdom, Germany' 'Belgium, Netherlands'
 'South Korea, Czech Republic' 'Australia, Germany' 'Vietnam'
 'United Kingdom, Belgium' 'United Kingdom, Australia, United States'
 'France, Japan, United States'
 'United Kingdom, Germany, Spain, United States'
 'United Kingdom, United States, France, Italy'
```

Pre Processing Data

```
#unnesting the directors column, i.e- creating separate lines for each director in a movie

constraint1=df['director'].apply(lambda x: str(x).split(', ')).tolist()
df_new1=pd.DataFrame(constraint1,index=df['title'])
df_new1=df_new1.stack()
df_new1=pd.DataFrame(df_new1.reset_index())
df_new1.rename(columns={0:'Directors'},inplace=True)
df_new1.drop(['level_1'],axis=1,inplace=True)
df_new1.head()
```

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

```
#unnesting the cast column, i.e- creating separate lines for each cast member in a movie

constraint2=df['cast'].apply(lambda x: str(x).split(', ')).tolist()
df_new2=pd.DataFrame(constraint2,index=df['title'])
df_new2=df_new2.stack()
df_new2=pd.DataFrame(df_new2.reset_index())
df_new2.rename(columns={0:'Actors'},inplace=True)
df_new2.drop(['level_1'],axis=1,inplace=True)
df_new2.head()
```

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

```
#unnesting the listed_in column, i.e- creating separate lines for each genre in a movie

constraint3=df['listed_in'].apply(lambda x: str(x).split(', ')).tolist()
df_new3=pd.DataFrame(constraint3,index=df['title'])
df_new3=df_new3.stack()
df_new3=pd.DataFrame(df_new3.reset_index())
df_new3.rename(columns={0:'Genre'},inplace=True)
df_new3.drop(['level_1'],axis=1,inplace=True)
df_new3.head()
```

	title	Genre
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

```
#unnesting the country column, i.e- creating separate lines for each country in a movie

constraint4=df['country'].apply(lambda x: str(x).split(', ')).tolist()
df_new4=pd.DataFrame(constraint4,index=df['title'])
df_new4=df_new4.stack()
df_new4=pd.DataFrame(df_new4.reset_index())
df_new4.rename(columns={0:'country'},inplace=True)
df_new4.drop(['level_1'],axis=1,inplace=True)
df_new4.head()
```

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

```
#merging the unnested director data with unnested actors data

df_new5=df_new2.merge(df_new1,on=['title'],how='inner')

#merging the above merged data with unnested genre data

df_new6=df_new5.merge(df_new3,on=['title'],how='inner')

#merging the above merged data with unnested country data

df_new=df_new6.merge(df_new4,on=['title'],how='inner')


#replacing nan values of director and actor by Unknown Actor and Director

df_new['Actors'].replace(['nan'],['Unknown Actor'],inplace=True)
df_new['Directors'].replace(['nan'],['Unknown Director'],inplace=True)
df_new['country'].replace(['nan'],[np.nan],inplace=True)
df_new.head()
```

	title	Actors	Directors	Genre	country
0	Dick Johnson Is Dead	Unknown Actor	Kirsten Johnson	Documentaries	United States
1	Blood & Water	Ama Qamata	Unknown Director	International TV Shows	South Africa
2	Blood & Water	Ama Qamata	Unknown Director	TV Dramas	South Africa
3	Blood & Water	Ama Qamata	Unknown Director	TV Mysteries	South Africa
4	Blood & Water	Khosi Ngema	Unknown Director	International TV Shows	South Africa

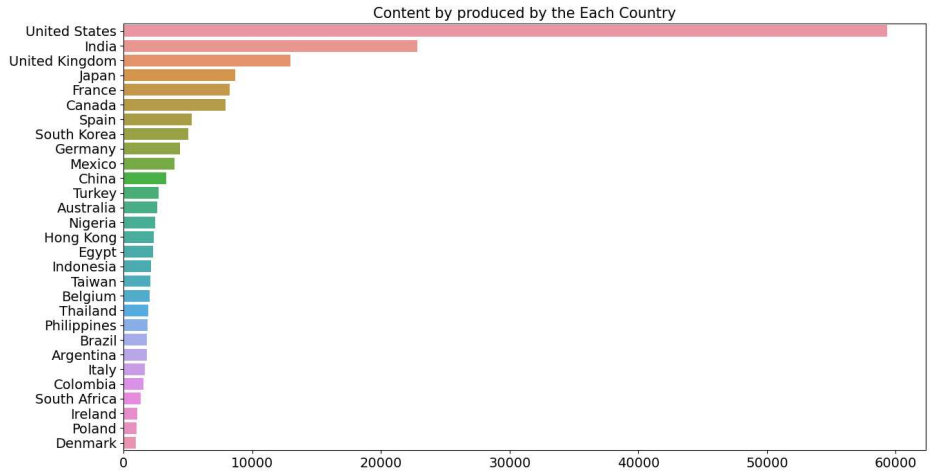
```
#merging our unnested data with the original data
df_final=df_new.merge(df[['show_id', 'type', 'title', 'date_added',
       'release_year', 'rating', 'duration']],on=['title'],how='left')
df_final.head()
```

	title	Actors	Directors	Genre	country	show_id	type	date_added	release_
0	Dick Johnson Is Dead	Unknown Actor	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	
1	Blood & Water	Ama Qamata	Unknown Director	International TV Shows	South Africa	s2	TV Show	September 24, 2021	
2	Blood & Water	Ama Qamata	Unknown Director	TV Dramas	South Africa	s2	TV Show	September 24, 2021	

```
#Now we are done with missing values, but the dates are still not quite right…

df_final['date_added'] = pd.to_datetime(df['date_added'])
df_final['month_added'] = df_final['date_added'].dt.month
df_final['month_name_added'] = df_final['date_added'].dt.month_name()
df_final['year_added'] = df_final['date_added'].dt.year
df_final.head()
```

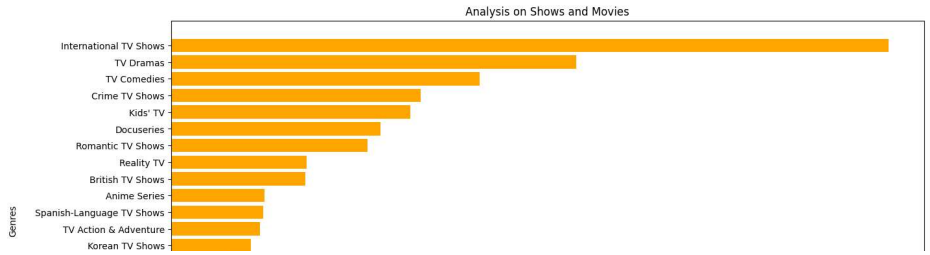
	title	Actors	Directors	Genre	country	show_id	type	date_added	release_
0	Dick Johnson Is Dead	Unknown Actor	Kirsten Johnson	Documentaries	United States	s1	Movie	2021-09-25	
1	Blood & Water	Ama Qamata	Unknown Director	International TV Shows	South Africa	s2	TV Show	2021-09-24	
2	Blood & Water	Ama Qamata	Unknown Director	TV Dramas	South Africa	s2	TV Show	2021-09-24	

```
df_countries = df_final['country'].value_counts()[df_final['country'].value_counts(normalize=True)> 0.005]


# barplotting the number of content per each country
plt.figure(figsize=(15,8))
plt.title('Content by produced by the Each Country', fontsize=15)
plt.tick_params(labelsize=14)
sns.barplot(y=df_countries.index, x=df_countries.values)
plt.show()
```

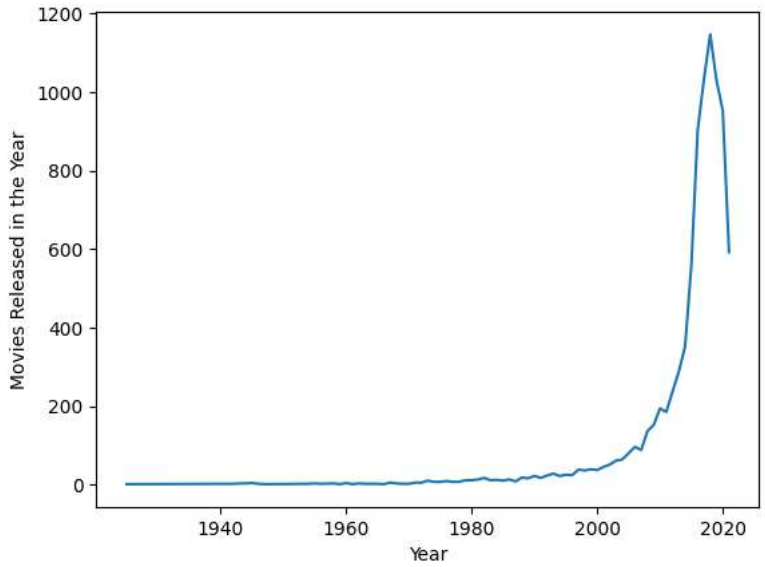



```
df_shows=df_final[df_final['type']=='TV Show']
df_movies=df_final[df_final['type']=='Movie']
```

```
df_genre=df_shows.groupby(['Genre']).agg({"title":"nunique"}).reset_index().sort_values(by=['title'],ascending=False)
plt.figure(figsize=(15,8))
plt.barh(df_genre[::-1]['Genre'], df_genre[::-1]['title'],color=['orange'])
plt.xlabel('Frequency of Genres')
plt.ylabel('Genres')
plt.title('Analysis on Shows and Movies',fontsize=12)
plt.show()
```

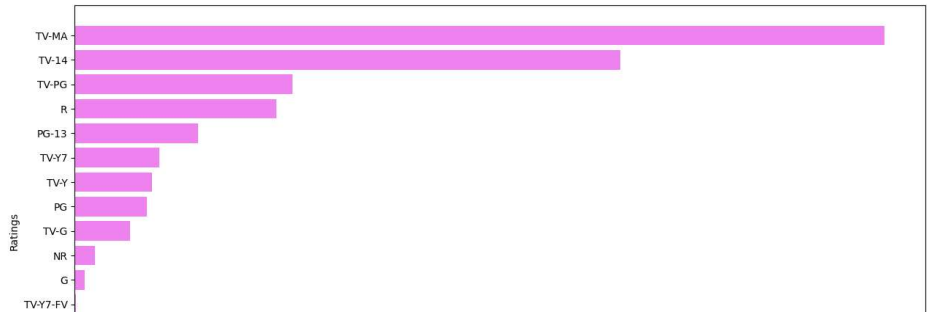



```
df_year=df_final.groupby(['release_year']).agg({"title":"nunique"}).reset_index()
sns.lineplot(data=df_year, x='release_year', y='title')
plt.ylabel("Movies Released in the Year")
plt.xlabel("Year")
plt.show()
```



The Amount of Content across Netflix has increased from 2008 continuously till 2019. Then started decreasing from here(probably due to Covid)

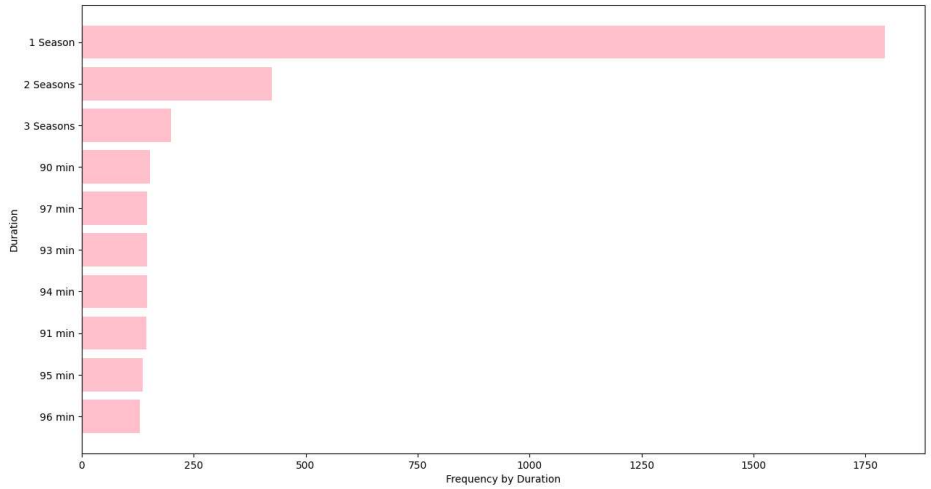
```
df_rating=df_final.groupby(['rating']).agg({"title":"nunique"}).reset_index().sort_values(by=['title'],ascending=False)
plt.figure(figsize=(15,8))
plt.barh(df_rating[::-1]['rating'], df_rating[::-1]['title'],color=['violet'])
plt.xlabel('Frequency by Ratings')
plt.ylabel('Ratings')
plt.show()
```



The highly rated content on Netflix is based on for Mature Audiences, R Rated, content not for audience under 14 and those which require Parental Guidance

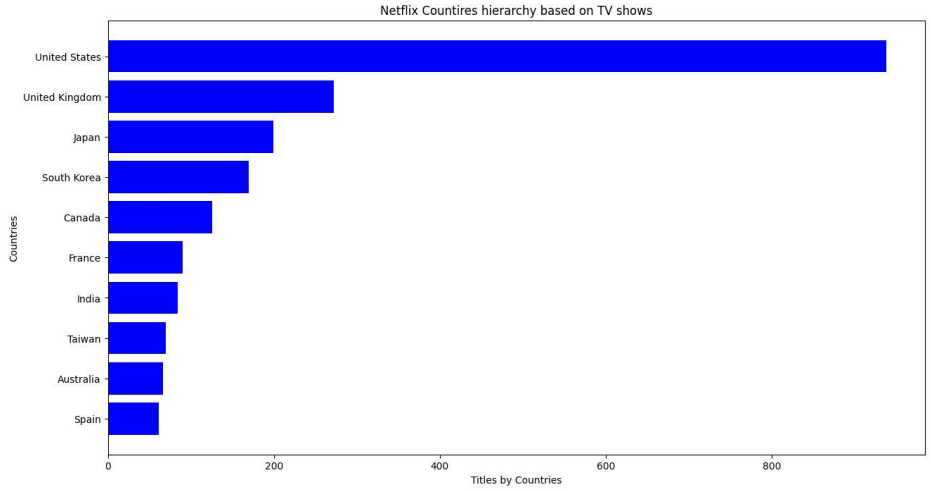
66 min

```
df_duration=df_final.groupby(['duration']).agg({"title":"nunique"}).reset_index().sort_values(by=['title'],ascending=False)[:10]
plt.figure(figsize=(15,8))
plt.barh(df_duration[::-1]['duration'], df_duration[::-1]['title'],color=['pink'])
plt.xlabel('Frequency by Duration')
plt.ylabel('Duration')
plt.show()
```

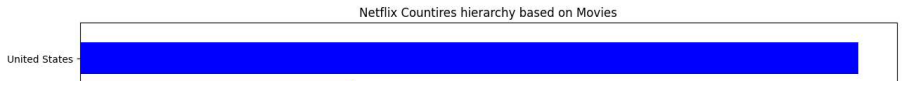


The Most Watched content in our dataset is 80-100 mins.These must be movies and Shows having only 1 Season.

```
df_country=df_shows.groupby(['country']).agg({"title":"nunique"}).reset_index().sort_values(by=['title'],ascending=False)[:10]
plt.figure(figsize=(15,8))
plt.barh(df_country[::-1]['country'], df_country[::-1]['title'],color=['blue'])
plt.xlabel('Titles by Countries')
plt.ylabel('Countries')
plt.title('Netflix Countires hierarchy based on TV shows',fontsize=12)
plt.show()
```




```
df_country=df_movies.groupby(['country']).agg({"title":"nunique"}).reset_index().sort_values(by=['title'],ascending=False)[:10]
plt.figure(figsize=(15,8))
plt.barh(df_country[::-1]['country'], df_country[::-1]['title'],color=['blue'])
plt.xlabel('Titles by Countries')
plt.ylabel('Countries')
plt.title('Netflix Countires hierarchy based on Movies',fontsize=12)
plt.show()
```

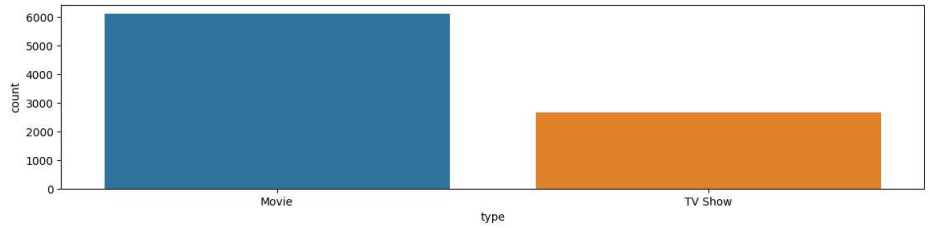


United States is leading across both TV Shows and Movies, And India is much more prevalent in Movies as compared TV Shows. Even UK is giving good numbers in both TV Shows and Movies

However the number of Movies created in India outweigh the sum of TV Shows and Movies across UK since India was rated as second in net sum of whole content across Netflix.

```
plt.figure(figsize=(14, 3))
sns.countplot(x='type',data = df)
#ratio of the movie / tv shows
```

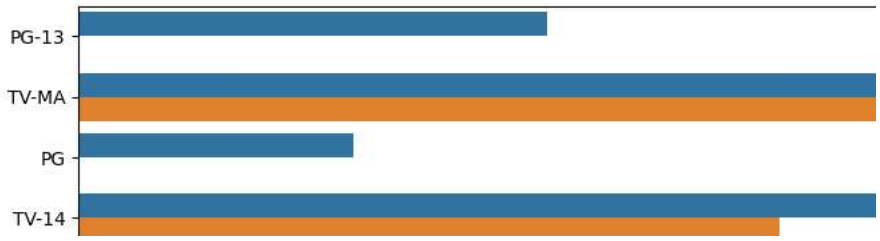
```
<Axes: xlabel='type', ylabel='count'>
```



there is a clear majority of movies released by netflix

```
plt.figure(figsize = (20,10))
sns.countplot(y='rating',data = df,hue='type')
```

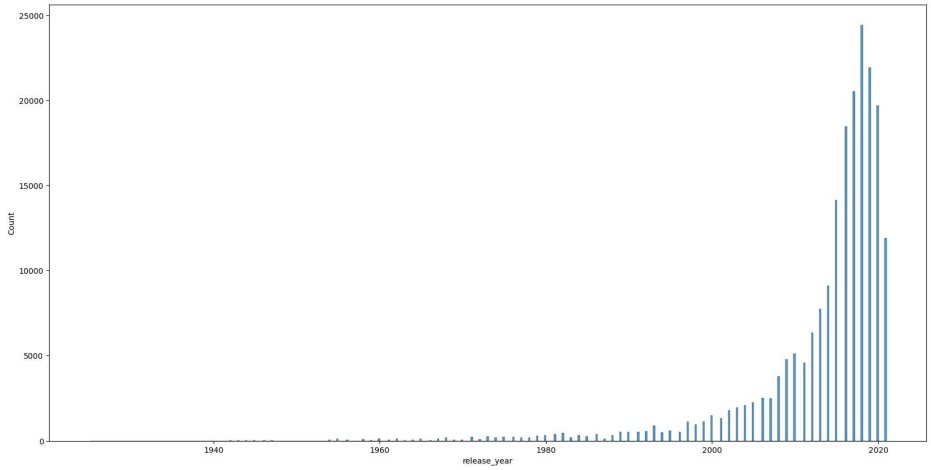
```
<Axes: xlabel='count', ylabel='rating'>
```



- The highly rated content on Netflix TV Shows and Movies is based on for Mature Audiences

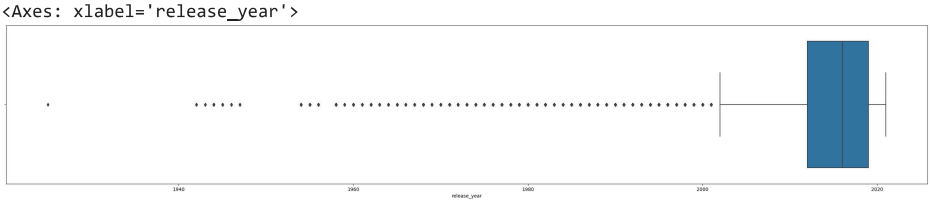
```
plt.figure(figsize = (20,10))
sns.histplot(data=df_final['release_year'])

#histplot foe year
```

```
<Axes: xlabel='release_year', ylabel='Count'>
```



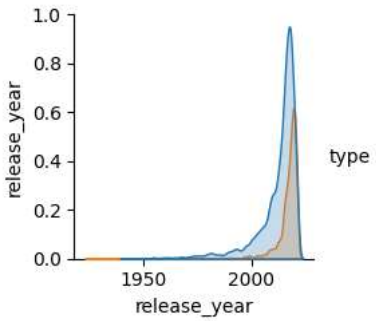

```
# boxplot based on release_year

plt.figure(figsize = (35,6))
sns.boxplot(data = df_final,x='release_year')
```

```
<Axes: xlabel='release_year'>
```



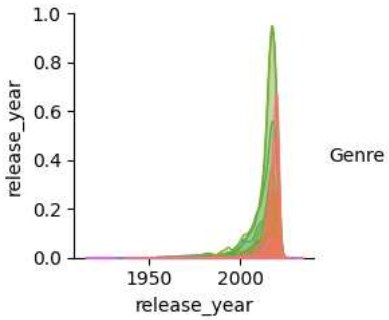
```
#pair  plot of Type and release year

plt.figure(figsize = (12,10))
sns.pairplot(data=df_final,hue='type')
```

```
<seaborn.axisgrid.PairGrid at 0x7f24aaf5fd30>
<Figure size 1200x1000 with 0 Axes>
```




```
#pair  plot of Type and release year

plt.figure(figsize = (12,10))
sns.pairplot(data=df_final,hue='Genre')
```

```
<seaborn.axisgrid.PairGrid at 0x7f24aacfa860>
<Figure size 1200x1000 with 0 Axes>
```



Business insights :

1. The analsysis shows us that the there is high amt of movies produced per year than tv s
2. corona virus has the impacted the content quantity
3. the usa and india are the top 2 countries content wise
4. the content targeted in india is teens while the content being targeted at usa is adult
5. lack of child content produced in india
6. india and south korean have similar taste and usa and uk audience have similar taste
7. lack of diverse conetent for indian audience

Recommendations:

1. Produce more tv shows in high markets regions like india, United Kingdom with diverse quantity
2. More movies targetting teanage audience
3. More children quanity should be created

✓ 0s completed at 12:33 AM