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

Part1 ⚓

```
# Business Problem
# Analyze the data and generate insights that could help Netflix ijn deciding which type of shows/movies to produce and how they can grow the business
```

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

	show_id	type	title	director	cast	country	date_added	release_year	r
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020.0	P
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021.0	T
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021.0	T
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021.0	T
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar,	India	September 24, 2021	2021.0	T

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5398 entries, 0 to 5397
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   show_id       5398 non-null   object 
 1   type          5398 non-null   object 
 2   title         5397 non-null   object 
 3   director      3515 non-null   object 
 4   cast          4903 non-null   object 
 5   country       4735 non-null   object 
 6   date_added    5397 non-null   object 
 7   release_year  5397 non-null   float64
 8   rating        5397 non-null   object 
 9   duration      5397 non-null   object 
 10  listed_in     5397 non-null   object 
 11  description   5397 non-null   object 
dtypes: float64(1), object(11)
memory usage: 506.2+ KB
```

```
df.shape
```

```
(5398, 12)
```

Null_value ✈

```
df.isnull().sum()
```

```
show_id            0
type               0
```

```
title              1
director        1883
cast             495
country          663
date_added         1
release_year       1
rating             1
duration           1
listed_in          1
description        1
dtype: int64
```

Note-->

1. director has 2634 null value which is 30% of total(aprox)
2. cast has 825 null value which is 9.36% of total
3. counter has 831 null value which is 9.43% of total

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5398 entries, 0 to 5397
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   show_id       5398 non-null   object 
 1   type          5398 non-null   object 
 2   title         5397 non-null   object 
 3   director      3515 non-null   object 
 4   cast          4903 non-null   object 
 5   country       4735 non-null   object 
 6   date_added    5397 non-null   object 
 7   release_year  5397 non-null   float64
 8   rating        5397 non-null   object 
 9   duration      5397 non-null   object 
 10  listed_in     5397 non-null   object 
 11  description   5397 non-null   object 
dtypes: float64(1), object(11)
memory usage: 506.2+ KB
```

Unique_value ✈

```
for i in (df.columns):
  print(i,"-->",df[i].nunique())
```

```
show_id --> 5398
type --> 3
title --> 5397
director --> 2721
cast --> 4749
country --> 475
date_added --> 1177
release_year --> 62
rating --> 11
duration --> 208
listed_in --> 430
description --> 5381
```

Note:--> show_id and title are unique

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

```
array(['Kirsten Johnson', nan, 'Julien Leclercq', ..., 'Lekh Tandon',
       'Doesjka van Hoogdalem', 'Sushil Majumdar'], dtype=object)
```

```
# value of rating
df['rating'].value_counts()
```

```
TV-MA    2246
TV-14    1373
TV-PG     465
R         367
TV-Y      230
```

```
PG-13     226
TV-Y7     209
TV-G      144
PG        123
G          13
NC-17       1
Name: rating, dtype: int64
```

Note--> There is 3 mismached value and this value are of different columns

```
c=df['director'].apply(lambda x:str(x).split(', ')).tolist()
c
```

```
['Jeff Wadlow'],
['Charles Martin'],
['Stella Corradi'],
['Roland Emmerich'],
['Kevin Macdonald'],
['Ann Deborah Fishman'],
['Chris Gorak'],
['Peter Jackson'],
['Roger Kumble'],
['Jonathan Lynn'],
['Courtney Hunt'],
['Pierre Greco', 'Nancy Florence Savard'],
['Andrew Davis'],
['Kevin Smith'],
['nan'],
['Tosin Igho'],
['Chaitanya Tamhane'],
['Oriol Paulo'],
['Mike Rianda', 'Jeff Rowe'],
['Johannes Roberts'],
['nan'],
['Robert Pulcini', 'Shari Springer Berman'],
['nan'],
['Pedro Antonio'],
['nan'],
['nan'],
['John Wells'],
['Jonathan Liebesman'],
['Maria Pulera'],
['nan'],
['Santhosh Viswanath'],
['Seema Pahwa'],
['nan'],
['Ozan Açıktan'],
['Meltem Bozoflu'],
['Hakan Algül'],
['Selçuk Aydemir', 'Birkan Pusa'],
['Selçuk Aydemir'],
['Ömer Faruk Sorak'],
['Şenol Sönmez'],
['Alexis Morante'],
['Burak Aksak'],
['Kıvanç Baruönü'],
['Kıvanç Baruönü'],
['Rindala Kodeih'],
['Kongkiat Khomsiri'],
['Bedran Güzel'],
['Hakan Algül'],
['Marwan Nabil'],
['MIKIKO', 'Daito Manabe'],
['nan'],
['Kayode Kasum'],
['Yılmaz Erdoğan', 'Ömer Faruk Sorak'],
['Takashi Shimizu'],
['nan'],
['nan'],
['Joe Penna'],
...]
```

```
df_1=pd.DataFrame(c,index=df['title'])
df_1
```

title	0	1	2	3	4	5	6	7	8	9	10
Dick Johnson Is Dead	Kirsten Johnson	None	None	None	None	None	None	None	None	None	None
Blood & Water	nan	None	None	None	None	None	None	None	None	None	None
Kota Factory	nan	None	None	None	None	None	None	None	None	None	None

unnesting

```
#unnnesting director columns
c=df['director'].apply(lambda x:str(x).split(', ')).tolist()
df_1=pd.DataFrame(c,index=df['title'])
df_1=df_1.stack()   #melting the data
df_1=pd.DataFrame(df_1.reset_index())
df_1
```

	title	level_1	0
0	Dick Johnson Is Dead	0	Kirsten Johnson
1	Blood & Water	0	nan
2	Ganglands	0	Julien Leclercq
3	Jailbirds New Orleans	0	nan
4	Kota Factory	0	nan
...	...	...	...
5811	Breakout	0	nan
5812	Hans Teeuwen: Real Rancour	0	Doesjka van Hoogdalem
5813	Intersection	0	nan
5814	Lal Patthar	0	Sushil Majumdar
5815	NaN	0	nan

5816 rows × 3 columns

```
# more cleaning df_1
df_1.drop(['level_1'],axis=1,inplace=True)
df_1.rename(columns={0:'director'},inplace=True)
df_1
```

	title	director
0	Dick Johnson Is Dead	Kirsten Johnson
1	Blood & Water	nan
2	Ganglands	Julien Leclercq
3	Jailbirds New Orleans	nan
4	Kota Factory	nan
...	...	...
5811	Breakout	nan
5812	Hans Teeuwen: Real Rancour	Doesjka van Hoogdalem
5813	Intersection	nan
5814	Lal Patthar	Sushil Majumdar
5815	NaN	nan

5816 rows × 2 columns

```
#unnnesting cast columns
c=df['cast'].apply(lambda x:str(x).split(', ')).tolist()
df_2=pd.DataFrame(c,index=df['title'])
df_2=df_2.stack()   #melting the data
df_2=pd.DataFrame(df_2.reset_index())
df_2
```

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

```
# more cleaning df_2
df_2.drop(['level_1'],axis=1,inplace=True)
df_2.rename(columns={0:'Actor'},inplace=True)
df_2
```

	title	Actor
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
...	...	...
40642	Lal Patthar	Dulari
40643	Lal Patthar	Chandramohan
40644	Lal Patthar	Sujata Bakshi
40645	Lal Patthar	Meena T.
40646	NaN	nan

40647 rows × 2 columns

```
#unnnesting listed_in columns
c=df['listed_in'].apply(lambda x:str(x).split(', ')).tolist()
df_3=pd.DataFrame(c,index=df['title'])
df_3=df_3.stack()   #melting the data
df_3=pd.DataFrame(df_3.reset_index())
df_3
```

	title	level_1	0
0	Dick Johnson Is Dead	0	Documentaries
1	Blood & Water	0	International TV Shows
2	Blood & Water	1	TV Dramas
3	Blood & Water	2	TV Mysteries
4	Ganglands	0	Crime TV Shows
...	...	...	...
12069	Intersection	2	TV Dramas
12070	Lal Patthar	0	Classic Movies
12071	Lal Patthar	1	Dramas
12072	Lal Patthar	2	International Movies
12073	NaN	0	nan

12074 rows × 3 columns

```
# more cleaning df_3
df_3.drop(['level_1'],axis=1,inplace=True)
df_3.rename(columns={0:'Genre'},inplace=True)
df_3
```

	title	Genre
0	Dick Johnson Is Dead	Documentaries
1	Blood & Water	International TV Shows
2	Blood & Water	TV Dramas
3	Blood & Water	TV Mysteries

```
#unnnesting country columns
c=df['country'].apply(lambda x:str(x).split(', ')).tolist()
df_4=pd.DataFrame(c,index=df['title'])
df_4=df_4.stack()   #melting the data
df_4=pd.DataFrame(df_4.reset_index())
df_4
```

	title	level_1	0
0	Dick Johnson Is Dead	0	United States
1	Blood & Water	0	South Africa
2	Ganglands	0	nan
3	Jailbirds New Orleans	0	nan
4	Kota Factory	0	India
...	...	...	...
6450	Breakout	0	nan
6451	Hans Teeuwen: Real Rancour	0	Netherlands
6452	Intersection	0	Turkey
6453	Lal Patthar	0	India
6454	NaN	0	nan

6455 rows × 3 columns

```
# more cleaning df_4
df_4.drop(['level_1'],axis=1,inplace=True)
df_4.rename(columns={0:'country'},inplace=True)
df_4
```

	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
...	...	...
6450	Breakout	nan
6451	Hans Teeuwen: Real Rancour	Netherlands
6452	Intersection	Turkey
6453	Lal Patthar	India
6454	NaN	nan

6455 rows × 2 columns

merge

```
# merging all the unnesting dataframe
```

```
d=pd.merge(df_1,df_2,on='title',how='inner')
d_1=pd.merge(d,df_3,on='title',how='inner')
d_2=pd.merge(d_1,df_4,on='title',how='inner')
d_2
```

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

```
d_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 123851 entries, 0 to 123850
Data columns (total 5 columns):
 #   Column    Non-Null Count   Dtype 
---  ------    --------------   ----- 
 0   title     123850 non-null  object
 1   director  123851 non-null  object
 2   Actor     123851 non-null  object
 3   Genre     123851 non-null  object
 4   country   123851 non-null  object
dtypes: object(5)
memory usage: 5.7+ MB
```

```
d_2.isnull().sum()
```

```
title       0
director    0
Actor       0
Genre       0
country     0
dtype: int64
```

```
#finally merging with original datframe
required_features=df[['title','show_id','type','date_added','release_year','rating','duration']]
merge_df=pd.merge(d_2,required_features,on='title',how='left')
merge_df
```

	title	director	Actor	Genre	country	show_id	type	date_added	release_y
0	Dick Johnson Is Dead	Kirsten Johnson	nan	Documentaries	United States	s1	Movie	September 25, 2021	20
1	Blood & Water	nan	Ama Qamata	International TV Shows	South Africa	s2	TV Show	September 24, 2021	20
2	Blood & Water	nan	Ama Qamata	TV Dramas	South Africa	s2	TV Show	September 24, 2021	20
3	Blood & Water	nan	Ama Qamata	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	20
4	Blood & Water	nan	Khosi Ngema	International TV Shows	South Africa	s2	TV Show	September 24, 2021	20
...	...	...	...	...	...	...	...	...	
123846	Lal Patthar	Sushil Majumdar	Sujata Bakshi	International Movies	India	s5397	Movie	July 1, 2017	19
	Lal	Sushil	Meena						

```
#checking null in new datframe

merge_df.isnull().sum()
```

```
title           1
director        0
Actor           0
Genre           0
country         0
show_id         0
type            0
date_added      1
release_year    1
rating          1
duration        1
dtype: int64
```

```
merge_df['duration'].value_counts()
```

```
1 Season     26765
2 Seasons     7567
3 Seasons     4042
102 min       2284
94 min        2259
```

```
              ...
8 min            2
9 min            2
19 min           2
11 min           1
41 min           1
Name: duration, Length: 208, dtype: int64
```

```
# fill the missing value of duration

merge_df['duration'].fillna(merge_df['rating'],inplace = True)
```

```
# and remuving unwanted data from rating
merge_df.loc[merge_df['rating'].str.contains('min',na=False)]="no_rating"
```

```
#checking the rasults
merge_df['rating'].value_counts()
```

```
TV-MA    53206
TV-14    29281
R        10720
TV-PG     9406
PG-13     6889
PG        4764
TV-Y7     4233
TV-Y      2700
TV-G      2051
G          580
NC-17       20
Name: rating, dtype: int64
```

```
# date_added null values (imputing)
merge_df[merge_df['date_added'].isnull()]
```

	title	director	Actor	Genre	country	show_id	type	date_added	release_year	rating
							TV			

```
merge_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 123851 entries, 0 to 123850
Data columns (total 11 columns):
 #   Column        Non-Null Count   Dtype 
---  ------        --------------   ----- 
 0   title         123850 non-null  object
 1   director      123851 non-null  object
 2   Actor         123851 non-null  object
 3   Genre         123851 non-null  object
 4   country       123851 non-null  object
 5   show_id       123851 non-null  object
 6   type          123851 non-null  object
 7   date_added    123850 non-null  object
 8   release_year  123850 non-null  object
 9   rating        123850 non-null  object
 10  duration      123850 non-null  object
dtypes: object(11)
memory usage: 11.3+ MB
```

```
# creating a new data frame with mode of date_added
a=merge_df.groupby(['release_year'])[['date_added']].agg(pd.Series.mode).reset_index()
a
```

	release_year	date_added
0	1925.0	December 30, 2018
1	1945.0	February 9, 2021
2	1954.0	June 18, 2020
3	1956.0	June 18, 2020
4	1958.0	June 18, 2020
...	...	...
57	2017.0	September 1, 2017
58	2018.0	November 30, 2018
59	2019.0	October 4, 2019
60	2020.0	May 8, 2020
61	2021.0	August 6, 2021

62 rows × 2 columns

```
# mergeing the new dataframe with old one
merge_df_1=pd.merge(merge_df,a,on='release_year',how='inner')
merge_df_1.head()
```

	title	director	Actor	Genre	country	show_id	type	date_added_x	release_yea
0	Dick Johnson Is Dead	Kirsten Johnson	nan	Documentaries	United States	s1	Movie	September 25, 2021	2020.
1	Europe's Most Dangerous Man: Otto	Pedro de Echave	nan	Documentaries	nan	s17	Movie	September 22, 2021	2020.

```
# filling date_added_x with date_added_y
merge_df_1['date_added_x'].fillna(merge_df_1['date_added_y'],inplace=True)
```

```
merge_df_1.drop(['date_added_y'],axis=1,inplace=True)
merge_df_1.rename({"date_added_x":"date_added"},axis=1,inplace=True)
```

```
merge_df_1.isnull().sum()
```

```
title           0
director        0
Actor           0
Genre           0
country         0
show_id         0
type            0
date_added      0
release_year    0
rating          0
duration        0
dtype: int64
```

working with duration colunms

```
merge_df_1['duration'].value_counts()
```

```
1 Season     26765
2 Seasons     7567
3 Seasons     4042
102 min       2284
94 min        2259
             ...
19 min           2
9 min            2
8 min            2
11 min           1
41 min           1
Name: duration, Length: 208, dtype: int64
```

```
#1. create a copy of dataframe
#2. take all rows with min written as a seprate group
#3. remove min from each row
```

```
new_df=merge_df_1.copy()
```

```
new_df_1=new_df.loc[new_df['duration'].str.contains('min')]
```

```
new_df_1
```

	title	director	Actor	Genre	country	show_id	type	date_added	release_
0	Dick Johnson Is Dead	Kirsten Johnson	nan	Documentaries	United States	s1	Movie	September 25, 2021	2
	Europe's								

```
new_df_1['duration']=new_df['duration'].str.replace(' min','')
```

```
<ipython-input-47-2ac6cca7f338>:1: SettingWithCopyWarning:
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
  new_df_1['duration']=new_df['duration'].str.replace(' min','')
```

Garcia

```
new_df_1.head()
```

	title	director	Actor	Genre	country	show_id	type	date_added	release_year
0	Dick Johnson Is Dead	Kirsten Johnson	nan	Documentaries	United States	s1	Movie	September 25, 2021	2020.0
1	Europe's Most Dangerous Man: Otto	Pedro de Echave	nan	Documentaries	nan	s17	Movie	September 22, 2021	2020.0

```
#converting a columns from object to int
new_df_1['duration']=new_df_1['duration'].astype('int')
```

```
<ipython-input-49-68c09db1e096>:2: SettingWithCopyWarning:
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
  new_df_1['duration']=new_df_1['duration'].astype('int')
```

```
new_df_1['duration'].describe()
```

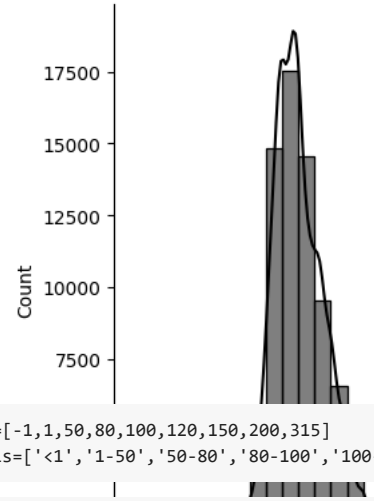
```
count    80202.000000
mean       107.780005
std         25.320469
min          3.000000
25%         94.000000
50%        105.000000
75%        121.000000
max        312.000000
Name: duration, dtype: float64
```

```
new_df_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 80202 entries, 0 to 123849
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   title         80202 non-null  object
 1   director      80202 non-null  object
 2   Actor         80202 non-null  object
 3   Genre         80202 non-null  object
 4   country       80202 non-null  object
 5   show_id       80202 non-null  object
 6   type          80202 non-null  object
 7   date_added    80202 non-null  object
 8   release_year  80202 non-null  object
 9   rating        80202 non-null  object
 10  duration      80202 non-null  int64 
dtypes: int64(1), object(10)
memory usage: 7.3+ MB
```

```
# obseving the distribution
sns.displot(new_df_1['duration'],
            kde=True,bins=(30),
            color='black'
            )
```

```
<seaborn.axisgrid.FacetGrid at 0x7a847c342d40>
```




```
bins=[-1,1,50,80,100,120,150,200,315]
labels=['<1','1-50','50-80','80-100','100-120','120-150','150-200','200-315']
```

```
new_df_1['duration_copy']=pd.cut(new_df_1['duration'],bins=bins,labels=labels)
```

```
<ipython-input-54-fecb6fdb27d8>:1: SettingWithCopyWarning:
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
  new_df_1['duration_copy']=pd.cut(new_df_1['duration'],bins=bins,labels=labels)
```

```
new_df_1.head()
```

	title	director	Actor	Genre	country	show_id	type	date_added	release_year
0	Dick Johnson Is Dead	Kirsten Johnson	nan	Documentaries	United States	s1	Movie	September 25, 2021	2020.0
1	Europe's Most Dangerous Man: Otto	Pedro de Echave	nan	Documentaries	nan	s17	Movie	September 22, 2021	2020.0

```
new_df_1['duration_copy'].value_counts()
```

```
100-120    28442
80-100     25883
120-150    16325
50-80       3942
150-200     3468
1-50        1744
200-315      398
<1             0
Name: duration_copy, dtype: int64
```

```
merge_df['date_added'].values
```

```
array(['September 25, 2021', 'September 24, 2021', 'September 24, 2021',
       ..., 'July 1, 2017', 'July 1, 2017', nan], dtype=object)
```

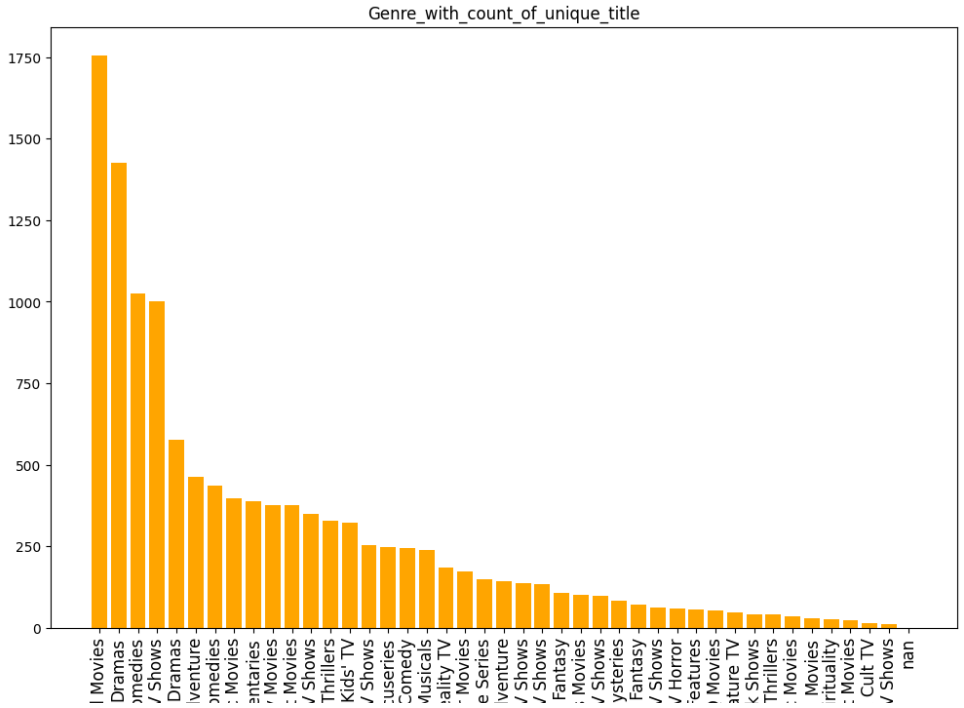
some busciness questions

Q1)list out top 3 Genres

```
#number of distinct title on the basis of genre
f=merge_df.groupby('Genre')['title'].nunique().reset_index()
f=f.sort_values(by=['title'],ascending=False)
f
```

	Genre	title
16	International Movies	1755
12	Dramas	1426
7	Comedies	1025
17	International TV Shows	1000
34	TV Dramas	576
0	Action & Adventure	464
33	TV Comedies	436
24	Romantic Movies	398
10	Documentaries	389
4	Children & Family Movies	377
15	Independent Movies	376
8	Crime TV Shows	349
41	Thrillers	328
18	Kids' TV	321
25	Romantic TV Shows	253
11	Docuseries	248
30	Stand-Up Comedy	244
22	Music & Musicals	237
23	Reality TV	185
14	Horror Movies	173
2	Anime Series	149
32	TV Action & Adventure	143
3	British TV Shows	136
28	Spanish-Language TV Shows	133
26	Sci-Fi & Fantasy	106
29	Sports Movies	102
19	Korean TV Shows	97
36	TV Mysteries	82
37	TV Sci-Fi & Fantasy	70
40	Teen TV Shows	62
35	TV Horror	58
1	Anime Features	56
20	LGBTQ Movies	54
27	Science & Nature TV	46
31	Stand-Up Comedy & Talk Shows	42

```
plt.figure(figsize=(12,8))
plt.bar(f['Genre'],f['title'],color='orange')
plt.xticks(rotation=90,fontsize=12)
plt.title('Genre_with_count_of_unique_title' ,fontsize=12)
plt.show()
```



Note--> Top 3 genre are

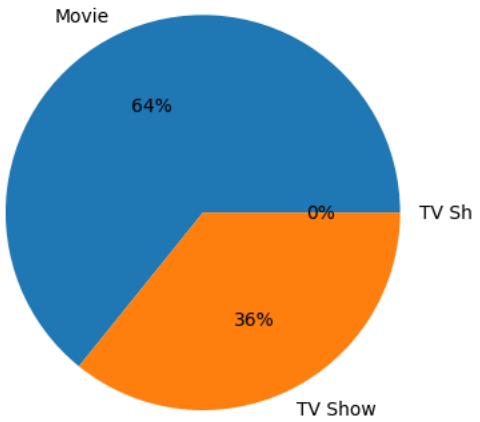
1. International Movies
2. Drames
3. Comedies

Q2)what is the percentage of type in dataframe

```
f=merge_df.groupby('type')['title'].nunique().reset_index()
f=f.sort_values(by=['title'],ascending=False)
f
```

	type	title
0	Movie	3464
2	TV Show	1933
1	TV Sh	0

```
plt.pie(f['title'],labels=f['type'],autopct='%.0f%%')
plt.show()
```



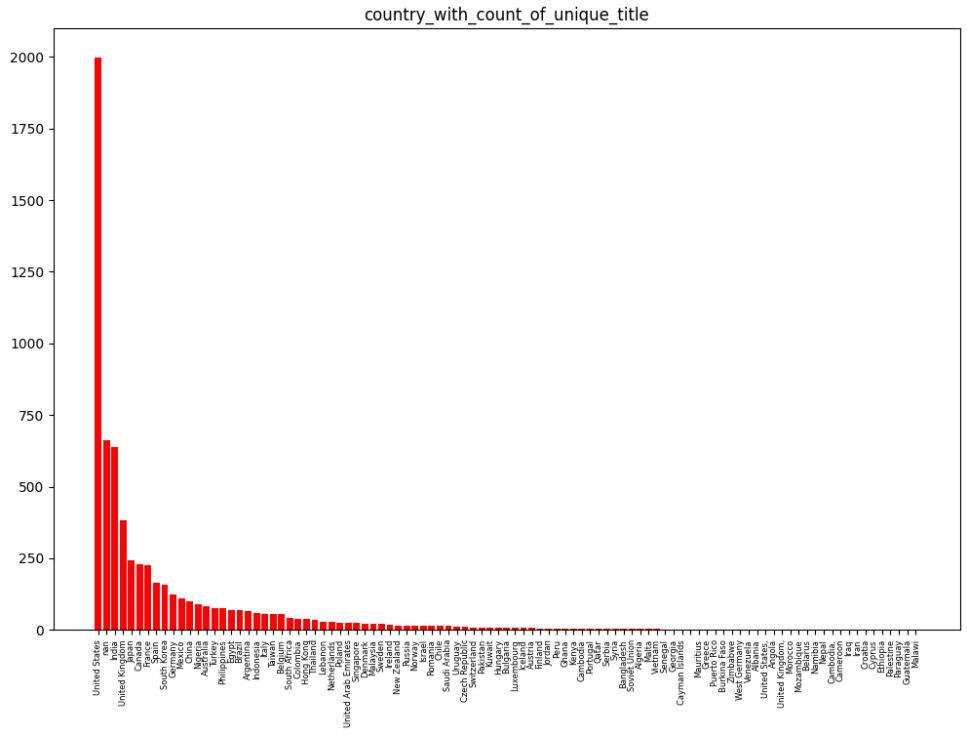
Note-->Movies and Tv_show are having 70:30 distribution

Q3)what content is available in different countries

```
p=merge_df.groupby('country')[['title']].nunique().reset_index().sort_values(by=['title'],ascending=False)
p
```

	country	title
91	United States	1999
98	nan	662
37	India	638
89	United Kingdom	383
44	Japan	244
...	...	...
26	Ethiopia	1
64	Palestine	1
65	Paraguay	1
33	Guatemala	1
50	Malawi	1

```
plt.figure(figsize=(12,8))
plt.bar(p['country'],p['title'],color='red')
plt.xticks(rotation=90,fontsize=6)
plt.title('country_with_count_of_unique_title' ,fontsize=12)
plt.show()
```



Q4) top 5 leading countries in terms of overall content creation on Netflix?

```
p[:5]
```

	country	title
91	United States	1999
98	nan	662
37	India	638
89	United Kingdom	383
44	Japan	244

Q4) How has the number of movies released per year changed over the last 20-30 years?

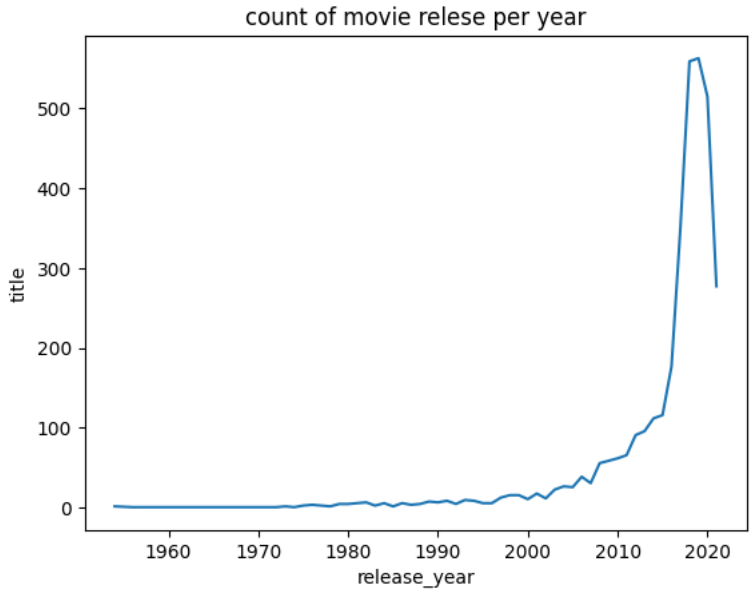
```
merge_df['year_added']=merge_df['date_added'].str.split(',', expand=True)[1]
merge_df
```

	title	director	Actor	Genre	country	show_id	type	date_added	release_y
0	Dick Johnson Is Dead	Kirsten Johnson	nan	Documentaries	United States	s1	Movie	September 25, 2021	20
1	Blood & Water	nan	Ama Qamata	International TV Shows	South Africa	s2	TV Show	September 24, 2021	20
2	Blood & Water	nan	Ama Qamata	TV Dramas	South Africa	s2	TV Show	September 24, 2021	20
3	Blood & Water	nan	Ama Qamata	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	20
4	Blood & Water	nan	Khosi Ngema	International TV Shows	South Africa	s2	TV Show	September 24, 2021	20
...	...	...	...	...	...	...	...	...	
123846	Lal Patthar	Sushil Majumdar	Sujata Bakshi	International Movies	India	s5397	Movie	July 1, 2017	19
	Lal	Sushil	Meena						

```
a=merge_df[merge_df['type']=='Movie']
b=a.groupby('release_year')['title'].nunique().reset_index()
b.head()
```

	release_year	title
0	1954.0	2
1	1956.0	1
2	1958.0	1
3	1959.0	1
4	1960.0	1

```
sns.lineplot(data=b, x='release_year', y='title')
plt.title('count of movie relese per year')
plt.show()
```



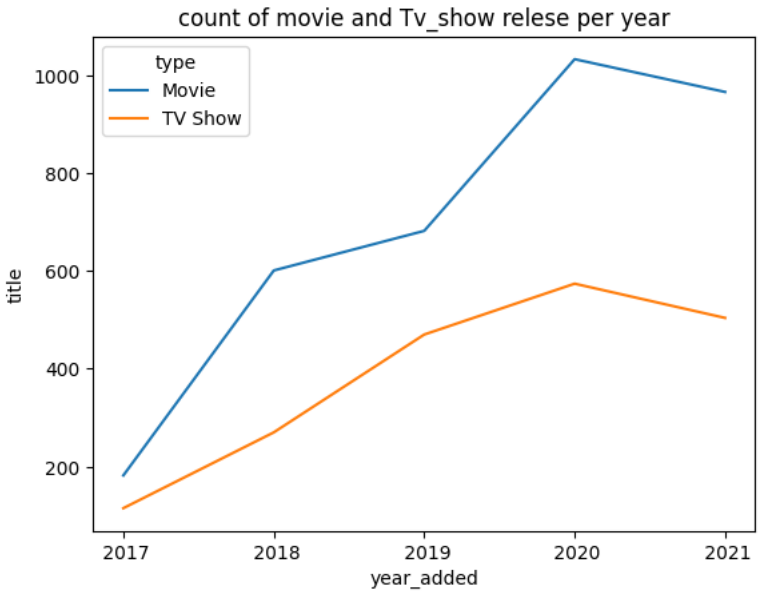
Note--> From 2000 to 2017 there was exponencial growth but fom 2018 there was a decline

Q5) Does Netflix has more focus on TV Shows than movies in recent years

```
b=merge_df.groupby(['year_added','type'])['title'].nunique().reset_index()
b
```

	year_added	type	title
0	2017	Movie	182
1	2017	TV Show	115
2	2018	Movie	601
3	2018	TV Show	270
4	2019	Movie	682
5	2019	TV Show	470
6	2020	Movie	1033

```
sns.lineplot(data=b, x='year_added', y='title',hue='type')
plt.title('count of movie and Tv_show relese per year')
plt.show()
```



Note-->we can observe that there is a growth and decline in both Movies and Tv-show

INSIGHT

-->director has 2634 null value which is 30% of total(aprox) cast has 825 null value which is 9.36% of total counter has 831 null value which is 9.43% of total

->Top 3 genre are

1. International Movies
2. Drames
3. Comedies

-->Movies and Tv_show are having 70:30 distribution

-->Top 5 countey with most content are (United state,United Kingdom ,Japan,India)

-->From 2000 to 2017 there was exponencial growth but fom 2018 there was a decline

Double-click (or enter) to edit