NETFLIX - DATA EXPLORATION AND VISUALISATION Netflix is one of the most popular media and video streaming platforms (OTT). They have over 10000 movies or tv shows available on their platform, as of mid-2021, they have over 222M Subscribers globally. This tabular dataset consists of listings of all the movies and tv shows available on Netflix, along with details such as - cast, directors, ratings, release year, duration, etc. At the same time its a bussiness with crores of turnover. Problem Statement: Hence it is important to find and forecast the type of content people are actually willing to watch. So we should do the analyse the data given and get insights that help Netflix to grow better.

Importing the required Libraries

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [ ]:
```
!pip install pandas_profiling
```

```
Requirement already satisfied: pandas_profiling in /usr/local/lib/python3.10/dist-pa
ckages (3.6.6)
Requirement already satisfied: ydata-profiling in /usr/local/lib/python3.10/dist-pac
kages (from pandas_profiling) (4.8.3)
Requirement already satisfied: scipy<1.14,>=1.4.1 in /usr/local/lib/python3.10/dist-
packages (from ydata-profiling->pandas_profiling) (1.11.4)
Requirement already satisfied: pandas!=1.4.0,<3,>1.1 in /usr/local/lib/python3.10/di
st-packages (from ydata-profiling->pandas_profiling) (2.0.3)
Requirement already satisfied: matplotlib<3.9,>=3.2 in /usr/local/lib/python3.10/dis
t-packages (from ydata-profiling->pandas_profiling) (3.7.1)
Requirement already satisfied: pydantic>=2 in /usr/local/lib/python3.10/dist-package
s (from ydata-profiling->pandas_profiling) (2.7.4)
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in /usr/local/lib/python3.10/dist-
packages (from ydata-profiling->pandas_profiling) (6.0.1)
Requirement already satisfied: jinja2<3.2,>=2.11.1 in /usr/local/lib/python3.10/dist
-packages (from ydata-profiling->pandas_profiling) (3.1.4)
Requirement already satisfied: visions[type_image_path]<0.7.7,>=0.7.5 in /usr/local/
lib/python3.10/dist-packages (from ydata-profiling->pandas_profiling) (0.7.6)
Requirement already satisfied: numpy<2,>=1.16.0 in /usr/local/lib/python3.10/dist-pa
ckages (from ydata-profiling->pandas_profiling) (1.25.2)
Requirement already satisfied: htmlmin==0.1.12 in /usr/local/lib/python3.10/dist-pac
kages (from ydata-profiling->pandas_profiling) (0.1.12)
Requirement already satisfied: phik<0.13,>=0.11.1 in /usr/local/lib/python3.10/dist-
packages (from ydata-profiling->pandas_profiling) (0.12.4)
Requirement already satisfied: requests<3,>=2.24.0 in /usr/local/lib/python3.10/dist
-packages (from ydata-profiling->pandas_profiling) (2.31.0)
Requirement already satisfied: tqdm<5,>=4.48.2 in /usr/local/lib/python3.10/dist-pac
kages (from ydata-profiling->pandas_profiling) (4.66.4)
Requirement already satisfied: seaborn<0.14,>=0.10.1 in /usr/local/lib/python3.10/di
st-packages (from ydata-profiling->pandas_profiling) (0.13.1)
Requirement already satisfied: multimethod<2,>=1.4 in /usr/local/lib/python3.10/dist
-packages (from ydata-profiling->pandas_profiling) (1.11.2)
Requirement already satisfied: statsmodels<1,>=0.13.2 in /usr/local/lib/python3.10/d
ist-packages (from ydata-profiling->pandas_profiling) (0.14.2)
Requirement already satisfied: typeguard<5,>=3 in /usr/local/lib/python3.10/dist-pac
kages (from ydata-profiling->pandas_profiling) (4.3.0)
Requirement already satisfied: imagehash==4.3.1 in /usr/local/lib/python3.10/dist-pa
ckages (from ydata-profiling->pandas_profiling) (4.3.1)
Requirement already satisfied: wordcloud>=1.9.1 in /usr/local/lib/python3.10/dist-pa
ckages (from ydata-profiling->pandas_profiling) (1.9.3)
Requirement already satisfied: dacite>=1.8 in /usr/local/lib/python3.10/dist-package
s (from ydata-profiling->pandas_profiling) (1.8.1)
Requirement already satisfied: numba<1,>=0.56.0 in /usr/local/lib/python3.10/dist-pa
ckages (from ydata-profiling->pandas_profiling) (0.58.1)
Requirement already satisfied: PyWavelets in /usr/local/lib/python3.10/dist-packages
(from imagehash==4.3.1->ydata-profiling->pandas_profiling) (1.6.0)
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (fr
om imagehash==4.3.1->ydata-profiling->pandas_profiling) (9.4.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-pac
kages (from jinja2<3.2,>=2.11.1->ydata-profiling->pandas_profiling) (2.1.5)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib<3.9,>=3.2->ydata-profiling->pandas_profiling) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packag
es (from matplotlib<3.9,>=3.2->ydata-profiling->pandas_profiling) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib<3.9,>=3.2->ydata-profiling->pandas_profiling) (4.53.0)
```

```
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib<3.9,>=3.2->ydata-profiling->pandas_profiling) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pac
kages (from matplotlib<3.9,>=3.2->ydata-profiling->pandas_profiling) (24.1)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib<3.9,>=3.2->ydata-profiling->pandas_profiling) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dis
t-packages (from matplotlib<3.9,>=3.2->ydata-profiling->pandas_profiling) (2.8.2)
Requirement already satisfied: llvmlite<0.42,>=0.41.0dev0 in /usr/local/lib/python3.
10/dist-packages (from numba<1,>=0.56.0->ydata-profiling->pandas_profiling) (0.41.1)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packag
es (from pandas!=1.4.0,<3,>1.1->ydata-profiling->pandas_profiling) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-pack
ages (from pandas!=1.4.0,<3,>1.1->ydata-profiling->pandas_profiling) (2024.1)
Requirement already satisfied: joblib>=0.14.1 in /usr/local/lib/python3.10/dist-pack
ages (from phik<0.13,>=0.11.1->ydata-profiling->pandas_profiling) (1.4.2)
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/d
ist-packages (from pydantic>=2->ydata-profiling->pandas_profiling) (0.7.0)
Requirement already satisfied: pydantic-core==2.18.4 in /usr/local/lib/python3.10/di
st-packages (from pydantic>=2->ydata-profiling->pandas_profiling) (2.18.4)
Requirement already satisfied: typing-extensions>=4.6.1 in /usr/local/lib/python3.1
0/dist-packages (from pydantic>=2->ydata-profiling->pandas_profiling) (4.12.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.1
0/dist-packages (from requests<3,>=2.24.0->ydata-profiling->pandas_profiling) (3.3.
2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packag
es (from requests<3,>=2.24.0->ydata-profiling->pandas_profiling) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-
packages (from requests<3,>=2.24.0->ydata-profiling->pandas_profiling) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-
packages (from requests<3,>=2.24.0->ydata-profiling->pandas_profiling) (2024.6.2)
Requirement already satisfied: patsy>=0.5.6 in /usr/local/lib/python3.10/dist-packag
es (from statsmodels<1,>=0.13.2->ydata-profiling->pandas_profiling) (0.5.6)
Requirement already satisfied: attrs>=19.3.0 in /usr/local/lib/python3.10/dist-packa
ges (from visions[type_image_path]<0.7.7,>=0.7.5->ydata-profiling->pandas_profiling)
(23.2.0)
Requirement already satisfied: networkx>=2.4 in /usr/local/lib/python3.10/dist-packa
ges (from visions[type_image_path]<0.7.7,>=0.7.5->ydata-profiling->pandas_profiling)
(3.3)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from
patsy>=0.5.6->statsmodels<1,>=0.13.2->ydata-profiling->pandas_profiling) (1.16.0)
```

In [ ]:
```
from ydata_profiling import ProfileReport
```

In [ ]:
```
!gdown https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/origi
```

```
Downloading...
From: https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/000/940/origina
l/netflix.csv
To: /content/netflix.csv
100% 3.40M/3.40M [00:00<00:00, 23.3MB/s]
```

In [ ]:
```
netflix_df=pd.read_csv('netflix.csv')
print('Data Set read successfully')
```

```
Data Set read successfully
```

Analysing basic metrics of the Netflix Dataset

In [ ]:
```
#deep copy of the dataset

netflix = netflix_df.copy()
```

In [ ]:
```
netflix
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021
...	...	...	...	...	...	...	...	...
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
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody	United States	November 1, 2019	2009

	show_id	type	title	director	cast	country	date_added	release_year
					Harrelson, Emma Stone, ...			
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020	2006
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019	2015

8807 rows × 12 columns

Observations

Shape of the data

In [ ]:
```
netflix.shape
```

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

Insights: data constains 8807 rows and 12 columns

Data Set Information: This dataset contains 8700 rows and 12 columns(features).

Data description/Attribute Information: The dataset consists of a list of all the TV shows/movies available on Netflix:

Show_id: Unique ID for every Movie / Tv Show

Type: Identifier - A Movie or TV Show

Title: Title of the Movie / Tv Show

Director: Director of the Movie

Cast: Actors involved in the movie/show

Country: Country where the movie/show was produced

Date_added: Date it was added on Netflix

Release_year: Actual Release year of the movie/show

Rating: TV Rating of the movie/show

Duration: Total Duration - in minutes or number of seasons

Listed_in: Genre

Description: The summary description

Data types of all the attributes

In [ ]:
```
netflix.dtypes
```

Out[ ]:
```
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
```

Insights: dtypes shows the all datatypes such as object and int64

In [ ]:
```
netflix.size
```

Out[ ]:
```
105684
```

In [ ]:
```
netflix.head()
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year	rati
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	T M
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	T M
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	T M
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	T M

Insights: shows first five rows of the dataset

In [ ]:
```
netflix.tail()
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year
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
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	November 1, 2019	2009
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020	2006
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019	2015

Insights: Observed that last five rows of the dataset

In [ ]:
```
netflix.duplicated().sum()
```

Out[ ]:
```
0
```

Insights:Shows no duplicates in the dataset

In [ ]:
```
netflix.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
```

Insights:The above information shows that there are some Null values in the Dataset. It also shows that the total number of rows, names and number of columns and their respective datatypes.

In [ ]:
```
ProfileReport(netflix)
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Overview

Dataset statistics

Number of variables	12
Number of observations	8807
Missing cells	4307
Missing cells (%)	4.1%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	825.8 KiB
Average record size in memory	96.0 B

Variable types

Text	8
Categorical	2
DateTime	1
Numeric	1

Alerts

`director` has 2634 (29.9%) missing values	**Missing**
`cast` has 825 (9.4%) missing values	**Missing**

Out[ ]:

Statistical Data

In [ ]:
```
netflix.describe(include='object')
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	rating	du
count	8807	8807	8807	6173	7982	7976	8797	8803	
unique	8807	2	8807	4528	7692	748	1767	17	
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	TV-MA	1 S
freq	1	6131	1	19	19	2818	109	3207	

Insights: describe with object gives only count,unique, top and frequency

In [ ]:
```
netflix.describe(include='all')
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_yea
count	8807	8807	8807	6173	7982	7976	8797	8807.00000
unique	8807	2	8807	4528	7692	748	1767	Na
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	Na
freq	1	6131	1	19	19	2818	109	Na
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2014.18019
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.81931
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1925.00000
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2013.00000
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2017.00000
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2019.00000
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2021.00000

Insights: describe which includes all shows values in each coulmn names of count, mean,std,min,25%,50%,75% and max

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

Out[ ]:

	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

Insights: describe shows only numerical values of count, mean,std,min,25%,50%,75% and max

**Missing value detection

missing value detection/Null Values

In [ ]:
```
netflix.isnull().sum()
```

Out[ ]:
```
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
```

Insights:There are missing values in director,case,country,date_addded,release_year,rating and duration attributes.

In [ ]:
```
netflix['date_added'].head(1)
```

Out[ ]:
```
0    September 25, 2021
Name: date_added, dtype: object
```

In [ ]:
```
netflix['date_added'].tail(1)
```

Out[ ]:
```
8806    March 2, 2019
Name: date_added, dtype: object
```

Insights:The first day and last day on which the movie/show is added is 02/03/2019 and 25/09/2021

In [ ]:
```
netflix.values
```

Out[ ]:
```
array([['s1', 'Movie', 'Dick Johnson Is Dead', ..., '90 min',
        'Documentaries',
        'As her father nears the end of his life, filmmaker Kirsten Johnson stages
his death in inventive and comical ways to help them both face the inevitable.'],
       ['s2', 'TV Show', 'Blood & Water', ..., '2 Seasons',
        'International TV Shows, TV Dramas, TV Mysteries',
        'After crossing paths at a party, a Cape Town teen sets out to prove wheth
er a private-school swimming star is her sister who was abducted at birth.'],
       ['s3', 'TV Show', 'Ganglands', ..., '1 Season',
        'Crime TV Shows, International TV Shows, TV Action & Adventure',
        'To protect his family from a powerful drug lord, skilled thief Mehdi and
his expert team of robbers are pulled into a violent and deadly turf war.'],
       ...,
       ['s8805', 'Movie', 'Zombieland', ..., '88 min',
        'Comedies, Horror Movies',
        'Looking to survive in a world taken over by zombies, a dorky college stud
ent teams with an urban roughneck and a pair of grifter sisters.'],
       ['s8806', 'Movie', 'Zoom', ..., '88 min',
        'Children & Family Movies, Comedies',
        'Dragged from civilian life, a former superhero must train a new crop of y
outhful saviors when the military preps for an attack by a familiar villain.'],
       ['s8807', 'Movie', 'Zubaan', ..., '111 min',
        'Dramas, International Movies, Music & Musicals',
        "A scrappy but poor boy worms his way into a tycoon's dysfunctional famil
y, while facing his fear of music and the truth about his past."]],
      dtype=object)
```

In [ ]:
```
netflix.columns
```

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

Non-Graphical Analysis: Value counts and unique attributes

In [ ]:
```
netflix["show_id"].value_counts()
```

Out[ ]:
```
show_id
s1       1
s5875    1
s5869    1
s5870    1
s5871    1
        ..
s2931    1
s2930    1
s2929    1
s2928    1
s8807    1
Name: count, Length: 8807, dtype: int64
```

Insights:

It Shows the unique value in show id column.

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

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

Insights: There are two unique category Movie & TV Show and Movies are dominant content on netflix platform

In [ ]:
```
netflix["title"].value_counts()
```

Out[ ]:
```
title
Dick Johnson Is Dead                     1
Ip Man 2                                 1
Hannibal Buress: Comedy Camisado         1
Turbo FAST                               1
Masha's Tales                            1
                                       ..
Love for Sale 2                          1
ROAD TO ROMA                             1
Good Time                                1
Captain Underpants Epic Choice-o-Rama    1
Zubaan                                   1
Name: count, Length: 8807, dtype: int64
```

In [ ]:
```
netflix["cast"].value_counts()
```

Out[ ]:
```
cast
David Attenborough
19
Vatsal Dubey, Julie Tejwani, Rupa Bhimani, Jigna Bhardwaj, Rajesh Kava, Mousam, Sw
apnil
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, Matt Le
tscher, Alyssa Diaz
1
Nick Lachey, Vanessa Lachey
1
Takeru Sato, Kasumi Arimura, Haru, Kentaro Sakaguchi, Takayuki Yamada, Kendo Kobay
ashi, Ken Yasuda, Arata Furuta, Suzuki Matsuo, Koichi Yamadera, Arata Iura, Chikak
o Kaku, Kotaro Yoshida     1
Toyin Abraham, Sambasa Nzeribe, Chioma Chukwuka Akpotha, Chioma Omeruah, Chiwetalu
Agu, Dele Odule, Femi Adebayo, Bayray McNwizu, Biodun Stephen
1
Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanana, Manish Chaudhary, Meghna Malik, M
alkeet Rauni, Anita Shabdish, Chittaranjan Tripathy
1
Name: count, Length: 7692, dtype: int64
```

In [ ]:
```
netflix["director"].value_counts()
```

Out[ ]:
```
director
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: count, Length: 4528, dtype: int64
```

In [ ]:
```
netflix["date_added"].value_counts()
```

Out[ ]:
```
date_added
January 1, 2020      109
November 1, 2019      89
March 1, 2018         75
December 31, 2019     74
October 1, 2018       71
                     ...
December 4, 2016       1
November 21, 2016      1
November 19, 2016      1
November 17, 2016      1
January 11, 2020       1
Name: count, Length: 1767, dtype: int64
```

In [ ]:
```
netflix["release_year"].value_counts()
```

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

Insights: Content released in range of 74 unique years between 1925 - 2021 and Most of the content released in year 2018

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

Out[ ]:
```
duration
1 Season     1793
2 Seasons     425
3 Seasons     199
90 min        152
94 min        146
              ...
16 min          1
186 min         1
193 min         1
189 min         1
191 min         1
Name: count, Length: 220, dtype: int64
```

In [ ]:
```
netflix["country"].value_counts()
```

Out[ ]:
```
country
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: count, Length: 748, dtype: int64
```

In [ ]:
```
netflix["listed_in"].value_counts()
```

Out[ ]:
```
listed_in
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: count, Length: 514, dtype: int64
```

In [ ]:
```
netflix["rating"].value_counts()
```

Out[ ]:
```
rating
TV-MA       3207
TV-14       2160
TV-PG        863
R            799
PG-13        490
TV-Y7        334
TV-Y         307
PG           287
TV-G         220
NR            80
G             41
TV-Y7-FV       6
NC-17          3
UR             3
74 min         1
84 min         1
66 min         1
Name: count, dtype: int64
```

In [ ]:
```
netflix.nunique()
```

Out[ ]:
```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1767
release_year      74
rating            17
duration         220
listed_in        514
description     8775
dtype: int64
```

Insights: It shows the values of each columns

Pre-processing of the data

In [ ]:
```
netflix["rating"].value_counts()
```

Out[ ]:
```
rating
TV-MA       3207
TV-14       2160
TV-PG        863
R            799
PG-13        490
TV-Y7        334
TV-Y         307
PG           287
TV-G         220
NR            80
G             41
TV-Y7-FV       6
NC-17          3
UR             3
74 min         1
84 min         1
66 min         1
Name: count, dtype: int64
```

In [ ]:
```
netflix[netflix['duration'].isna()==True]
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year	rating
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2017	74 min
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	September 16, 2016	2010	84 min
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	August 15, 2016	2015	66 min

Insights:

Three rows of rating column contains the data of the duration column. So replacing those values

Transfer value from rating to duration

In [ ]:
```
netflix['duration']=netflix['duration'].fillna(netflix['rating'])
```

In [ ]:
```
netflix.iloc[[5541,5794,5813]]
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year	rating
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2017	74 min
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	September 16, 2016	2010	84 min
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	August 15, 2016	2015	66 min

Insights: deleted all values which are in Rating column shifted to Duration column

Replace value of rating to np.nan

In [ ]:
```
netflix['rating'][[5541,5794,5813]] = np.nan
```

In [ ]:
```
netflix['rating'][[5541,5794,5813]] #Checking whether the Nan values are replaced o
```

Out[ ]:
```
5541    NaN
5794    NaN
5813    NaN
Name: rating, dtype: object
```

Insights: Those values are replaced with Nan

In [ ]:
```
netflix.isnull().sum()
```

Out[ ]:
```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             7
duration           0
listed_in          0
description        0
dtype: int64
```

Insights: It shows the mising values of director,cast,country,date_added and rating columns

Converted mix date format to timestamp format

In [ ]:
```
#converted object datatype data_added column to datetime
netflix['date_added'] = pd.to_datetime(netflix['date_added'], format='mixed')
netflix.head(1)
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	du
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	9

In [ ]:
```
#converted object datatype release_year column to datetime
netflix['date_added'] = pd.to_datetime(netflix['date_added'], format='%Y')
netflix.head(1)
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	du
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	2021-09-25	2020	PG-13	9

In [ ]:
```
# Conversion null value of date_added column from release_year
netflix['date_added'].fillna(netflix['release_year'], inplace=True)
```

Handling of Null Values

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

Out[ ]:
```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added         0
release_year       0
rating             7
duration           0
listed_in          0
description        0
dtype: int64
```

In [ ]:
```
netflix.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    8807 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8800 non-null   object
 9   duration      8807 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
```

For categorical variables with null values, update those rows as unknown_column_name. Example : Replace missing value with Unknown Actor for missing value in Actors column.

Renamed the Listed_in to genre column

In [ ]:
```
netflix.rename({'listed_in':'genre'},axis=1,inplace=True)
```

Converted Null values to Unknown

In [ ]:
```
netflix['cast'].fillna('unknown_actor', inplace=True)
netflix['country'].fillna('unknown_country', inplace=True)
netflix['director'].fillna('unknown_director', inplace=True)
netflix['genre'].fillna('unknown_genre', inplace=True)
```

Replace with 0 for continuous variables having null values.

In [ ]:
```
netflix['type'].fillna(0,inplace=True)
netflix['rating'].fillna(0, inplace=True)
```

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

Out[ ]:
```
show_id         0
type            0
title           0
director        0
cast            0
country         0
date_added      0
release_year    0
rating          0
duration        0
genre           0
description     0
dtype: int64
```

Insights: After adding unknown and zero all columns shows with zeros

Visual Analysis - Univariate, Bivariate after pre-processing of the data

Note: Pre-processing involves unnesting of the data in columns like Actor, Director, Country Un-nesting the columns a. Un-nest the columns those have cells with multiple comma separated values by creating multiple rows *checking the nested columns for each and every columns*

In [ ]:
```
netflix[netflix['show_id'].apply(lambda x: "," in str(x))]
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration

In [ ]:
```
netflix[netflix['type'].apply(lambda x: "," in str(x))]
```

Out[ ]:

show_id	type	title	director	cast	country	date_added	release_year	rating	duration

In [ ]:
```
netflix[netflix['title'].apply(lambda x: "," in str(x))].head(2)
```

Out[ ]:

	show_id	type	title	director	cast	country	date_a
10	s11	TV Show	Vendetta: Truth, Lies and The Mafia	unknown_director	unknown_actor	unknown_country	2021-(00:
140	s141	Movie	El patrón, radiografía de un crimen	Sebastián Schindel	Joaquín Furriel, Luis Ziembrowski, Guillermo P...	Argentina, Venezuela	2021-(00:

In [ ]:
```
netflix[netflix['director'].apply(lambda x:"," in str(x))].head(2)
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	rel
6	s7	Movie	My Little Pony: A New Generation	Robert Cullen, José Luis Ucha	Vanessa Hudgens, Kimiko Glenn, James Marsden, ...	unknown_country	2021-09-24 00:00:00	
16	s17	Movie	Europe's Most Dangerous Man: Otto Skorzeny in ...	Pedro de Echave García, Pablo Azorín Williams	unknown_actor	unknown_country	2021-09-22 00:00:00	

In [ ]:
```
netflix[netflix['cast'].apply(lambda x:"," in str(x))].head(2)
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	r
1	s2	TV Show	Blood & Water	unknown_director	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24 00:00:00	
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	unknown_country	2021-09-24 00:00:00	

In [ ]:
```
netflix[netflix['country'].apply(lambda x:"," in str(x))].head(2)
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year
7	s8	Movie	Sankofa	Haile Gerima	Kofi Ghanaba, Oyafunmike Ogunlano, Alexandra D...	United States, Ghana, Burkina Faso, United Kin...	2021-09-24 00:00:00	1993
12	s13	Movie	Je Suis Karl	Christian Schwochow	Luna Wedler, Jannis Niewöhner, Milan Peschel, ...	Germany, Czech Republic	2021-09-23 00:00:00	2021

In [ ]:
```
netflix[netflix['release_year'].apply(lambda x:"," in str(x))]
```

Out[ ]:

show_id	type	title	director	cast	country	date_added	release_year	rating	duration

In [ ]:
```
netflix[netflix['date_added'].apply(lambda x:"," in str(x))].head(2)
```

Out[ ]:

show_id	type	title	director	cast	country	date_added	release_year	rating	duration

In [ ]:
```
netflix[netflix['rating'].apply(lambda x:"," in str(x))]
```

Out[ ]:

show_id	type	title	director	cast	country	date_added	release_year	rating	duration

In [ ]:
```
netflix[netflix['duration'].apply(lambda x:"," in str(x))]
```

Out[ ]:

show_id	type	title	director	cast	country	date_added	release_year	rating	duration

In [ ]:
```
netflix[netflix['description'].apply(lambda x:"," in str(x))].head()
```

Out[ ]:

	show_id	type	title	director	cast	country	date_add
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	unknown_actor	United States	2021-09-00:00:
1	s2	TV Show	Blood & Water	unknown_director	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-00:00:
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	unknown_country	2021-09-00:00:
3	s4	TV Show	Jailbirds New Orleans	unknown_director	unknown_actor	unknown_country	2021-09-00:00:
4	s5	TV Show	Kota Factory	unknown_director	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-00:00:

Insights: Found that multiple comma seperated values observed multilpe columns, so need to go for data cleaning process. Recommendations:To check the comma's through column wise nested before unnesting the document.found the Unnesting for these columns:title,director,cast,country and listed_in date_added and description can be acceptable.

In [ ]:
```
netflix.head()
```

Out[ ]:

	show_id	type	title	director	cast	country	date_add
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	unknown_actor	United States	2021-09-00:00:
1	s2	TV Show	Blood & Water	unknown_director	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-00:00:
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	unknown_country	2021-09-00:00:
3	s4	TV Show	Jailbirds New Orleans	unknown_director	unknown_actor	unknown_country	2021-09-00:00:
4	s5	TV Show	Kota Factory	unknown_director	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-00:00:

Unnested for the columns

unnesting the columns ---director, cast, genre,country

In [ ]:
```
# creating director frame:
df_director =  netflix[['title', 'director']]
df_director["unnested_df"] = df_director['director'].apply(lambda x: str(x).split(
df_director = df_director.explode("unnested_df")
df_director.head()
```

Out[ ]:

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

In [ ]:
```
# creating cast frame:
df_cast =  netflix[['title', 'cast']]
df_cast["unnested_cast"] = df_cast['cast'].apply(lambda x: str(x).split(", "))
df_cast = df_cast.explode("unnested_cast")
df_cast.head()
```

Out[ ]:

	title	cast	unnested_cast
0	Dick Johnson Is Dead	unknown_actor	unknown_actor
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Ama Qamata
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Khosi Ngema
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Gail Mabalane
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Thabang Molaba

In [ ]:
```
# creating listed_in frame:
df_genre =  netflix[['title','genre']]
df_genre["unnested_genre"] = df_genre['genre'].apply(lambda x: str(x).split(", "))
df_genre = df_genre.explode("unnested_genre")
df_genre.head()
```

Out[ ]:

	title	genre	unnested_genre
0	Dick Johnson Is Dead	Documentaries	Documentaries
1	Blood & Water	International TV Shows, TV Dramas, TV Mysteries	International TV Shows
1	Blood & Water	International TV Shows, TV Dramas, TV Mysteries	TV Dramas
1	Blood & Water	International TV Shows, TV Dramas, TV Mysteries	TV Mysteries
2	Ganglands	Crime TV Shows, International TV Shows, TV Act...	Crime TV Shows

In [ ]:
```
# creating country frame:
df_country =  netflix[["title", "country"]]
df_country["unnested_country"] = df_country["country"].apply(lambda x: str(x).spli
df_country = df_country.explode("unnested_country")
df_country.head()
```

Out[ ]:

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

Merging all columns--

In [ ]:
```
#Merging director column with main column
new_df=pd.merge(left=netflix,
                right=df_director,
                on="title",
                how="inner")
```

In [ ]:
```
new_df.head(2)
```

Out[ ]:

	show_id	type	title	director_x	cast	country	date_added	release
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	unknown_actor	United States	2021-09-25 00:00:00	
1	s2	TV Show	Blood & Water	unknown_director	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24 00:00:00	

In [ ]:
```
#Merging cast column with main column
new_df=pd.merge(right=new_df,
                left=df_cast,
                on='title',
                how='inner')
```

In [ ]:
```
new_df.head(2)
```

Out[ ]:

	title	cast_x	unnested_cast	show_id	type	director_x	cast_y
0	Dick Johnson Is Dead	unknown_actor	unknown_actor	s1	Movie	Kirsten Johnson	unknown_acto
1	Blood & Water	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	Ama Qamata	s2	TV Show	unknown_director	Ama Qamata Khosi Ngema Gail Mabalane Thaban..

In [ ]:
```
#Merging listed_in column with main column
new_df=pd.merge(right=new_df,
                left=df_genre,
                on='title',
                how='inner')
```

In [ ]:
```
new_df.shape
```

Out[ ]:
```
(161216, 18)
```

In [ ]:
```
#Merging country column with main column
new_df=pd.merge(right=new_df,
                left=df_country,
```

```
                on='title',
                how='inner')
```

In [ ]:
```
new_df.head()
```

Out[ ]:

	title	country_x	unnested_country	genre_x	unnested_genre	cast_x	u
0	Dick Johnson Is Dead	United States	United States	Documentaries	Documentaries	unknown_actor	u
1	Blood & Water	South Africa	South Africa	International TV Shows, TV Dramas, TV Mysteries	International TV Shows	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	
2	Blood & Water	South Africa	South Africa	International TV Shows, TV Dramas, TV Mysteries	International TV Shows	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	
3	Blood & Water	South Africa	South Africa	International TV Shows, TV Dramas, TV Mysteries	International TV Shows	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	
4	Blood & Water	South Africa	South Africa	International TV Shows, TV Dramas, TV Mysteries	International TV Shows	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	

In [ ]:
```
new_df.columns
```

Out[ ]:
```
Index(['title', 'country_x', 'unnested_country', 'genre_x', 'unnested_genre',
       'cast_x', 'unnested_cast', 'show_id', 'type', 'director_x', 'cast_y',
       'country_y', 'date_added', 'release_year', 'rating', 'duration',
       'genre_y', 'description', 'director_y', 'unnested_df'],
      dtype='object')
```

In [ ]:
```
nnew_df=new_df.copy()
```

In [ ]:
```
nnew_df.drop(columns=['genre_x','country_x','cast_x','director_x','genre_y','countr
```

In [ ]:
```
nnew_df.shape
```

Out[ ]: (201991, 12)

In [ ]:
```
nnew_df.duplicated().sum()
```

Out[ ]: 55

In [ ]:
```
nnew_df.drop_duplicates(keep='first',inplace=True)
```

In [ ]:
```
nnew_df.shape
```

Out[ ]: (201936, 12)

In [ ]:
```
n_df=nnew_df.copy()
```

In [ ]:
```
n_df.head(2)
```

Out[ ]:

	title	unnested_country	unnested_genre	unnested_cast	show_id	type	date_added
0	Dick Johnson Is Dead	United States	Documentaries	unknown_actor	s1	Movie	2021-09-25 00:00:00
1	Blood & Water	South Africa	International TV Shows	Ama Qamata	s2	TV Show	2021-09-24 00:00:00

In [ ]:
```
n_df.rename({'unnested_genre':'genre','unnested_country':'country','unnested_direct
```

In [ ]:
```
n_df.head(2)
```

Out[ ]:

	title	country	genre	cast	show_id	type	date_added	release_y
0	Dick Johnson Is Dead	United States	Documentaries	unknown_actor	s1	Movie	2021-09-25 00:00:00	2(
1	Blood & Water	South Africa	International TV Shows	Ama Qamata	s2	TV Show	2021-09-24 00:00:00	2(

In [ ]:
```
n_df.shape
```

Out[ ]:
```
(201936, 12)
```

In [ ]:
```
n_df.isnull().sum()
```

Out[ ]:
```
title           0
country         0
genre           0
cast            0
show_id         0
type            0
date_added      0
release_year    0
rating          0
duration        0
description     0
unnested_df     0
dtype: int64
```

Univariate analysis

Find the counts of each categorical variable using non- graphical analysis.

Hint : We want you to find the values counts of each category for the given column

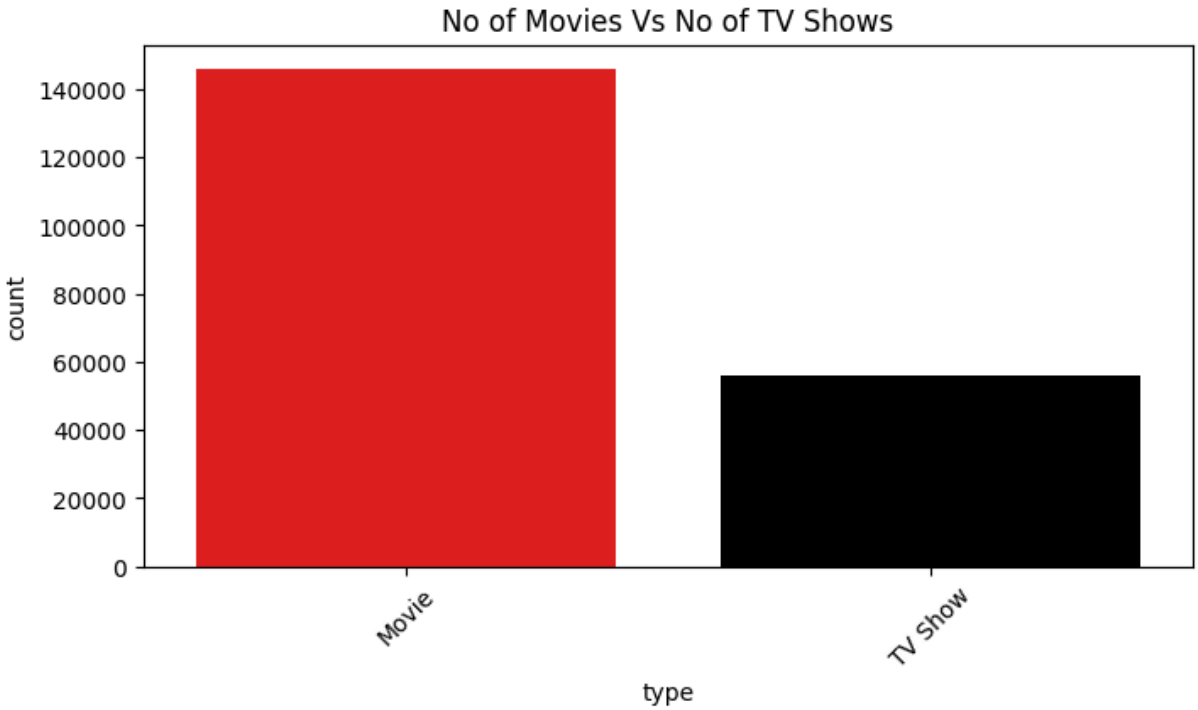
In [ ]:
```
n_df['type'].value_counts().head()
```

Out[ ]:
```
type
Movie      145788
TV Show     56148
Name: count, dtype: int64
```

Find the counts of each categorical variable both using graphical analysis.

Hint : We want you to find the values counts of each category for the given column

In [ ]:
```
color_palette={'Movie':'red','TV Show':'black'}
plt.figure(figsize=(8,4))
Type_Count=sns.countplot(data=n_df,x='type',palette=color_palette)
plt.xticks(rotation = 45)
plt.title('No of Movies Vs No of TV Shows')
plt.show()
```



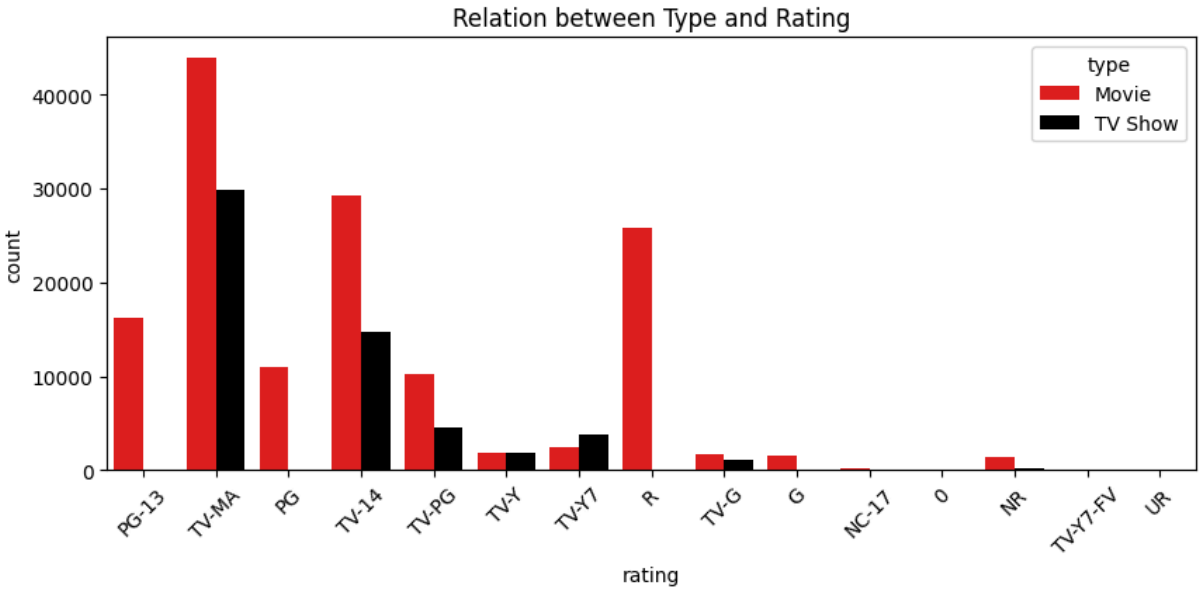
Insights: More Number of movies were added in Netflix than TV shows

What does 'good' look like?

In [ ]:
```
n_df['rating'].value_counts().head()
```

Out[ ]:
```
rating
TV-MA    73819
TV-14    43925
R        25859
PG-13    16246
TV-PG    14926
Name: count, dtype: int64
```

In [ ]:
```
color_palette={'Movie':'red','TV Show':'black'}
plt.figure(figsize=(10,4))
Type_rating=sns.countplot(data=n_df,x='rating',hue='type',palette=color_palette)
plt.xticks(rotation = 45)
plt.title('Relation between Type and Rating')
plt.show()
```



Observations:

By comparing the heights of the bars within each rating category, we can observe the relative frequency of each rating for movies and TV shows.

The color palette effectively distinguishes between movies (red) and TV shows (black), making it easy to differentiate between the two types of content.

The countplot reveals that the majority of ratings fall into the categories of TV-MA and TV-14

TV-14 - unsuitable for children under 14 years of age.

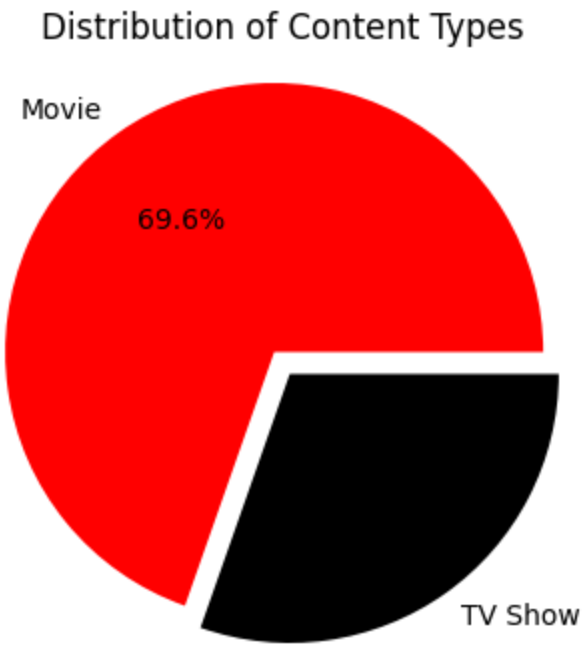
TV-MA - Mature Audience Only. Intended for adults and may be unsuitable for children under 17 years of age

In [ ]:
```
type_df=n_df.groupby(['type']).agg({'title':'nunique'}).reset_index()
type_df
```

Out[ ]:

	type	title
0	Movie	6131
1	TV Show	2676

In [ ]:
```
color_palette={'Movie':'red','TV Show':'black'}
plt.figure(figsize=(6, 4))
plt.pie(type_df['title'], labels=type_df['type'], autopct='%1.1f%%', startangle=0,e
plt.title('Distribution of Content Types')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle
plt.show()
```



Insights:

In our dataset, movies constitute roughly 70% of the content, while TV shows make up the remaining 30%.

Comparison of tv shows vs. movies.

a. Find the number of movies produced in each country and pick the top 10 countries. Hint : We want you to apply group by each country and find the count of unique titles of movies

In [ ]:
```
Movies=n_df[n_df['type']=='Movie']
```

In [ ]:
```
Movies.head()
```

Out[ ]:

	title	country	genre	cast	show_id	type	date_add
0	Dick Johnson Is Dead	United States	Documentaries	unknown_actor	s1	Movie	2021-09-00:00
159	My Little Pony: A New Generation	unknown_country	Children & Family Movies	Vanessa Hudgens	s7	Movie	2021-09-00:00
160	My Little Pony: A New Generation	unknown_country	Children & Family Movies	Vanessa Hudgens	s7	Movie	2021-09-00:00
161	My Little Pony: A New Generation	unknown_country	Children & Family Movies	Kimiko Glenn	s7	Movie	2021-09-00:00
162	My Little Pony: A New Generation	unknown_country	Children & Family Movies	Kimiko Glenn	s7	Movie	2021-09-00:00

In [ ]:
```
Movies.shape
```

Out[ ]:
```
(145788, 12)
```

In [ ]:
```
Movies.duplicated().sum()
```

Out[ ]:
```
0
```

Find the number of Tv-Shows produced in each country and pick the top 10 countries.

Hint : We want you to apply group by each country and find the count of unique titles of Tv-shows

---

In [ ]:
```
netflix_df.head()
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year	rati
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	T M
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	T M
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	T M
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	T M

Find the number of Tv-Shows produced in each country and pick the top 10 countries.

Hint : We want you to apply group by each country and find the count of unique titles of Tv-shows

In [546...

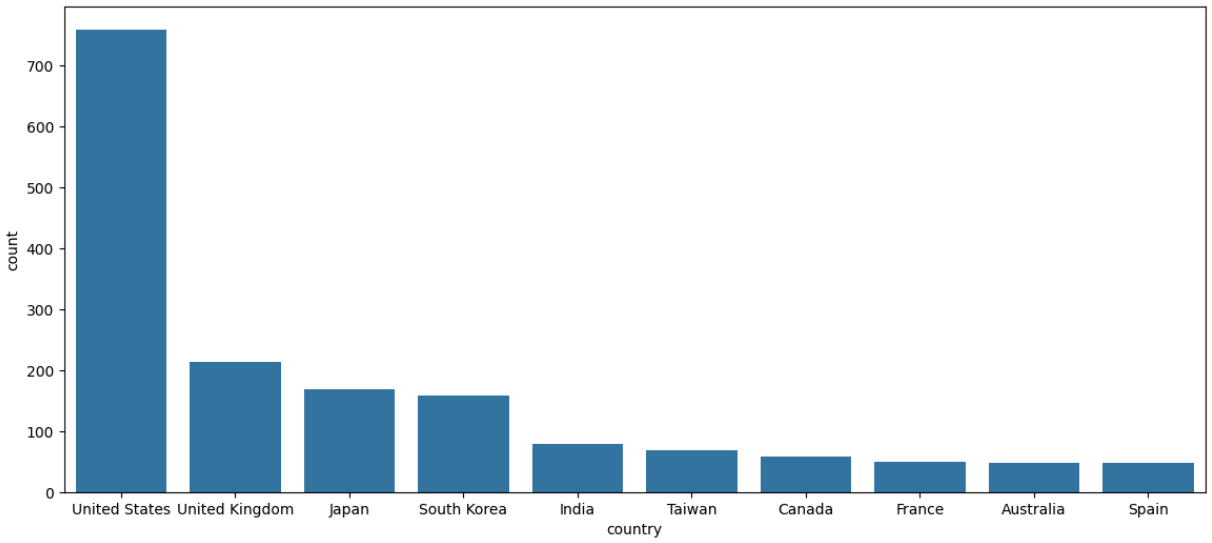
```
top_10_directors = n_df[n_df['type'] == 'TV Show']
top_10_directors = top_10_directors.groupby('cast')['title'].nunique().reset_index(
top10_TvShow = top10_TvShow_count.sort_values(by = 'count',  ascending = False).hea
top10_TvShow
```

Out[546...

	country	count
160	United States	760
140	United Kingdom	213
83	Japan	169
120	South Korea	158
66	India	79
132	Taiwan	68
17	Canada	59
47	France	49
4	Australia	48
125	Spain	48

In [ ]:
```
plt.figure(figsize=(14,6))
sns.barplot(data = top10_TvShow, x ='country', y = 'count')
```

Out[ ]: <Axes: xlabel='country', ylabel='count'>



Comparison of tv shows vs. movies.

In [ ]:
```
top10_movies_Tv = pd.merge(top10_movies, top10_TvShow, on = 'country' , how = 'left
top10_movies_Tv
```

Out[ ]:

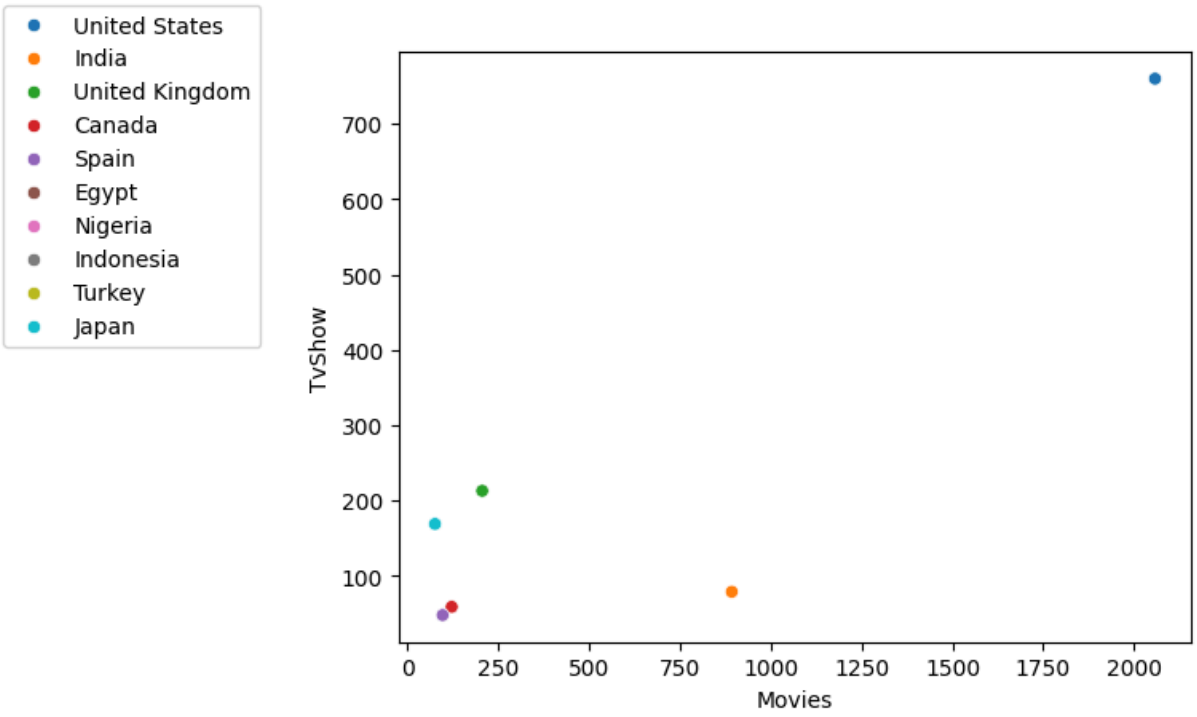
	country	count_x	count_y
0	United States	2058	760.0
1	India	893	79.0
2	United Kingdom	206	213.0
3	Canada	122	59.0
4	Spain	97	48.0
5	Egypt	92	NaN
6	Nigeria	86	NaN
7	Indonesia	77	NaN
8	Turkey	76	NaN
9	Japan	76	169.0

In [ ]:
```
top10_movies_Tv.rename(columns = {'count_x' : 'Movies' , 'count_y' : 'TvShow' },  i
top10_movies_Tv
```

Out[ ]:

	country	Movies	TvShow
0	United States	2058	760.0
1	India	893	79.0
2	United Kingdom	206	213.0
3	Canada	122	59.0
4	Spain	97	48.0
5	Egypt	92	NaN
6	Nigeria	86	NaN
7	Indonesia	77	NaN
8	Turkey	76	NaN
9	Japan	76	169.0

In [ ]:
```
sns.scatterplot( data = top10_movies_Tv, x = 'Movies', y = 'TvShow', hue= 'country'
plt.legend(loc=(-0.5,.5))
plt.show()
```



Insights:

Comparing the number of TV shows vs movies for each country:

United States: Produces a very high number of both movies and TV shows. Leads in both categories by a large margin.

India: Produces a large number of movies. Produces fewer TV shows compared to movies.

United Kingdom: Balanced production of both movies and TV shows. Neither category significantly outweighs the other.

Japan: Produces more TV shows than movies. Strong focus on television content.

China: Produces more movies compared to TV shows. Movies are a stronger focus than TV shows.

France, Germany, Spain, Canada: Moderate and similar levels of production for both movies and TV shows. Balanced but lower output compared to larger countries.

Unknown Country: Produces a moderate number of TV shows but fewer movies. Higher focus on TV shows relative to movies.

In summary, the United States dominates in both categories, India focuses more on movies, the UK has a balanced approach, Japan emphasizes TV shows, and China leans towards movies. France, Germany, Spain, and Canada have moderate production in both categories.

What is the best time to launch a TV show?

In [ ]:
```
Movies['date_added'] = pd.to_datetime(Movies['date_added'], errors='coerce', infer_
```

In [ ]:
```
Movies['day_added']=Movies['date_added'].dt.day
Movies['month_added']=Movies['date_added'].dt.month
Movies['year_added']=Movies['date_added'].dt.year
```

In [ ]:
```
Movies['day_name'] = Movies['date_added'].dt.strftime('%A')
```

In [ ]:
```
Movies.columns
```

Out[ ]:
```
Index(['title', 'country', 'genre', 'cast', 'show_id', 'type', 'date_added',
       'release_year', 'rating', 'duration', 'description', 'unnested_df',
       'day_added', 'month_added', 'year_added', 'day_name'],
      dtype='object')
```

Find which is the best month to release the Tv-show or the movie. Do the analysis separately for Tv-shows and Movies

Hint : We expect you to create a new column and group by each month and count the total number of movies/ tv shows.

In [ ]:
```
mm=Movies.groupby(Movies['date_added'].dt.strftime('%B'))[['show_id']].nunique().re
mm
```

Out[ ]:

	date_added	show_id
5	July	565
0	April	550
2	December	547
4	January	546
10	October	545
7	March	529
1	August	519
11	September	519
9	November	498
6	June	492
8	May	439
3	February	382

Insights

July appears to be the optimal month for releasing movies on Netflix, based on observed trends or data analysis, while December are favored for adding TV shows.

**Find which is the best week to release the Tv-show or the movie. Do the analysis separately for Tv-shows and Movies

Hint : We expect you to create a new column and group by each week and count the total number of movies/ tv shows.

In [ ]:
```
Movies.groupby(['day_name'])[['show_id']].nunique().reset_index().sort_values(by='s
```

Out[ ]:

	day_name	show_id
0	Friday	1566
4	Thursday	1053
6	Wednesday	906
5	Tuesday	852
1	Monday	628
3	Sunday	569
2	Saturday	557

Insights

Friday emerges as the preferred day for Netflix to introduce new movies into its catalogue, potentially reflecting strategic scheduling to coincide with peak viewing periods over the weekend.

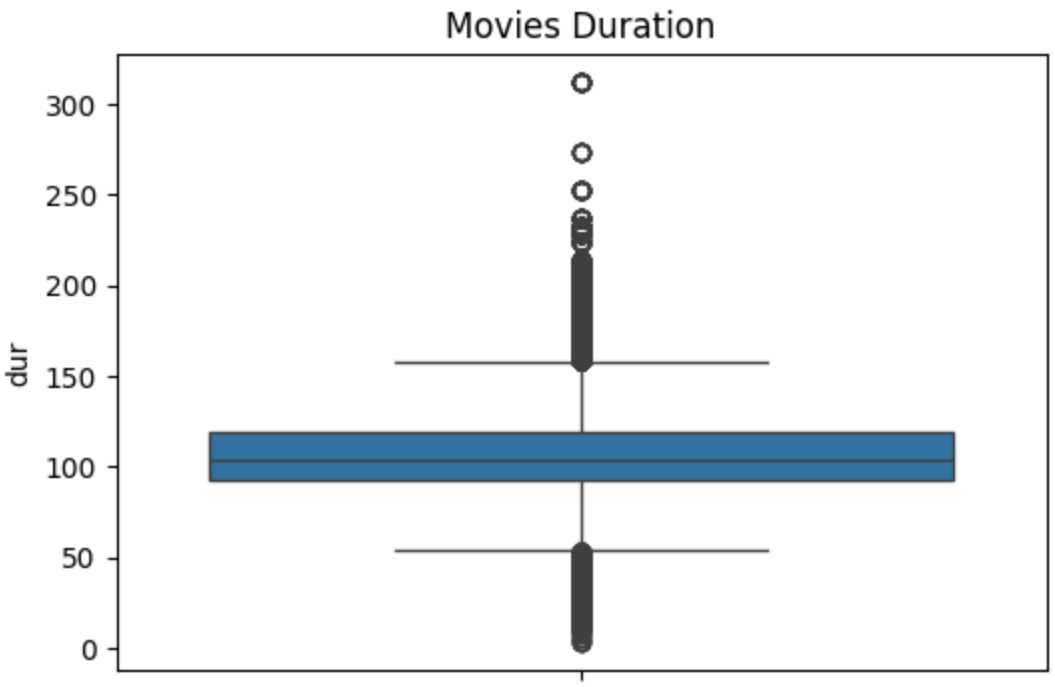
What is the best time to launch a TV show?

In [504…
```
Movies=n_df[n_df['type']=='Movie']
```

In [505…
```
Movies[['dur','n']]=Movies['duration'].str.split(" ",expand=True)
```

In [506…
```
Movies['dur']=Movies['dur'].astype(int)
```

In [507…
```
plt.figure(figsize=(6,4))
sns.boxplot(data=Movies['dur'])
plt.title('Movies Duration')
plt.show()
```



Insights:

The median duration of movies on Netflix is 100 minutes. This indicates that half of the movies available on the platform have a duration of 100 minutes or less, while the other half have a duration of 100 minutes or more.

In [508…
```
TV_Show=n_df[n_df['type']=='TV Show']
```

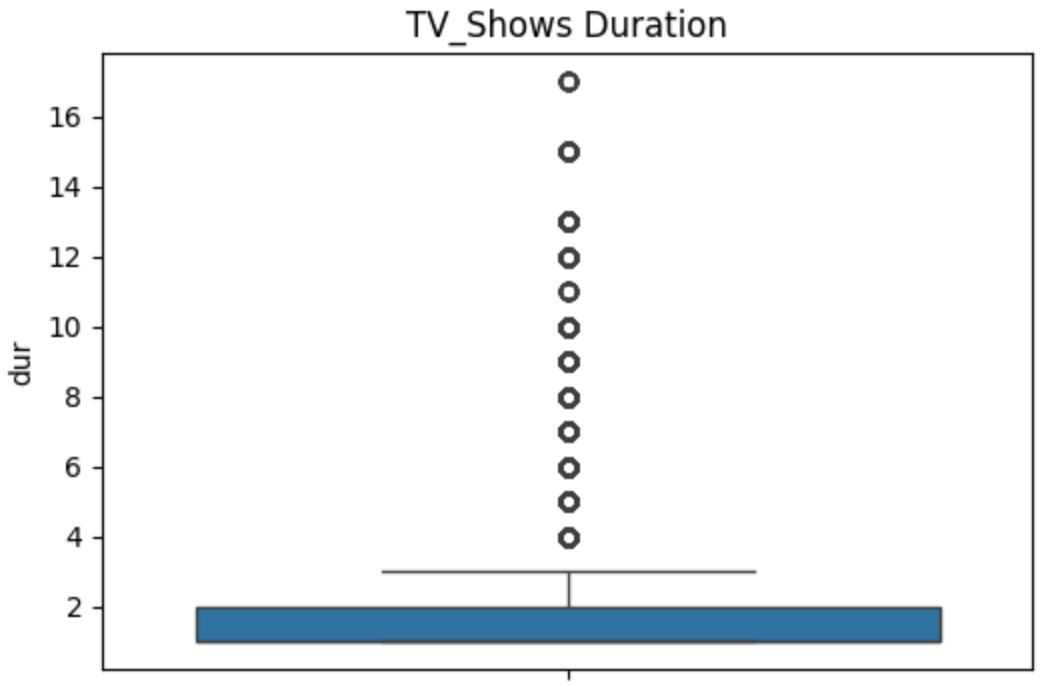
In [509…
```
TV_Show[['dur','n']]=TV_Show['duration'].str.split(" ",expand=True)
```

In [510…
```
TV_Show['dur']=TV_Show['dur'].astype(int)
```

In [511…
```
TV_Show['dur'].value_counts()
```

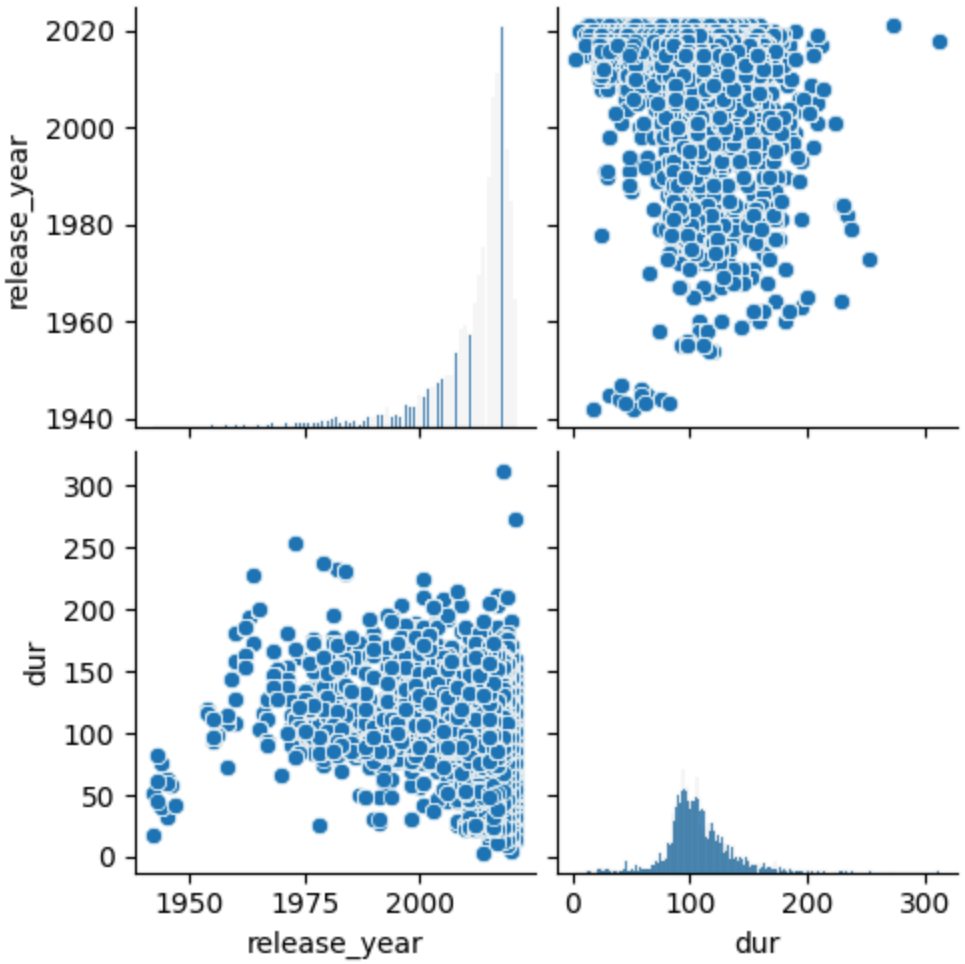
Out[511…
```
dur
1     35035
2      9559
3      5084
4      2134
5      1698
7       843
6       633
8       286
9       257
10      220
13      132
12      111
15       96
17       30
11       30
Name: count, dtype: int64
```

In [512…
```
plt.figure(figsize=(6,4))
sns.boxplot(data=TV_Show['dur'])
plt.title('TV_Shows Duration')
plt.show()
```

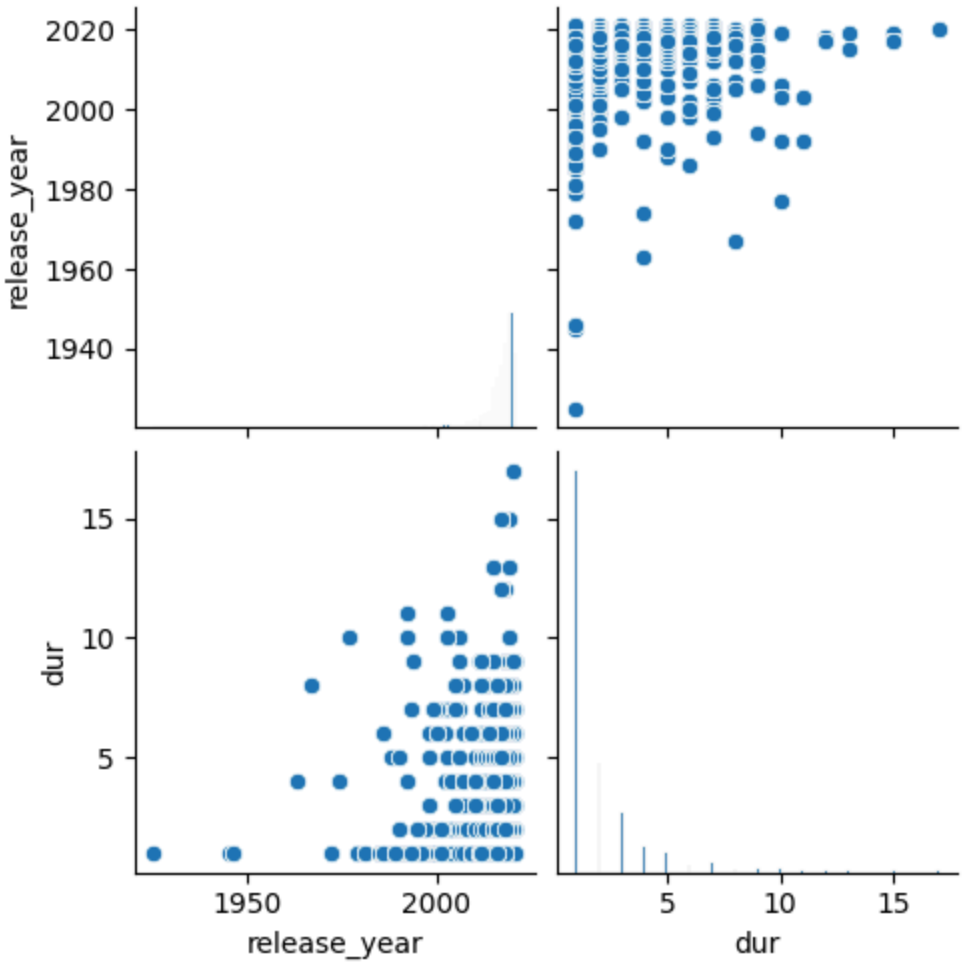


Insight:The maximum number of TV shows on Netflix contains one season. This suggests that a significant portion of TV shows available on the platform have a single season

In [513…
```
sns.pairplot(data=Movies)
plt.show()
```



In [514…
```
sns.pairplot(data=TV_Show)
plt.show()
```



Insights:

About half of the movies available on Netflix have durations ranging from 90 to 110 minutes.

Identify the top 10 actor who have appeared in most movies or TV shows.

Hint : # We want you to group by each actor and find the count of unique titles of Tv-shows/movies

Considering the actor in Movies

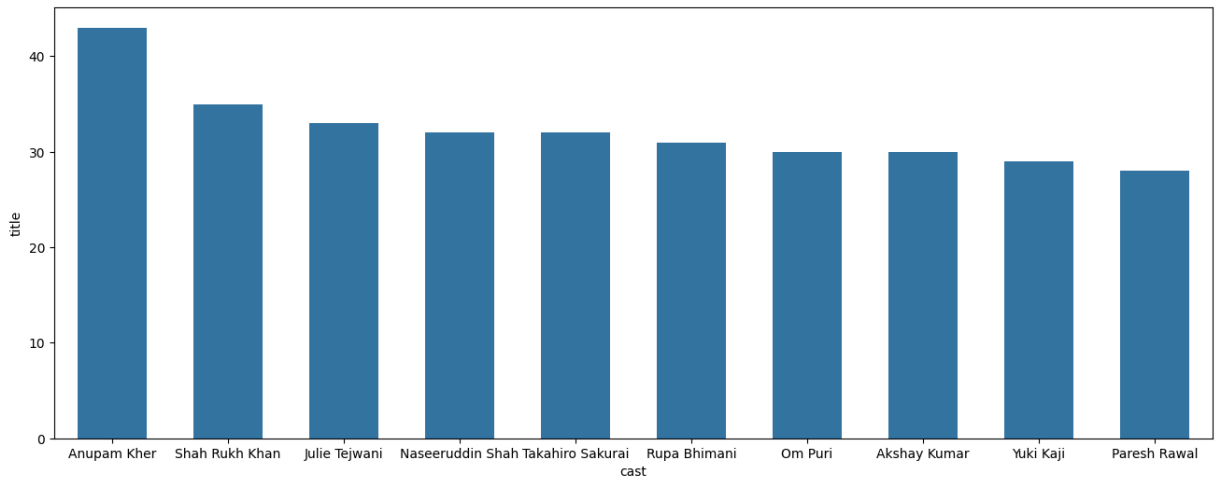
In [538…

```
top_actor = n_df.groupby('cast')['show_id'].nunique().sort_values(ascending  = Fals
top_actor.drop(0, inplace = True)
top_actor
```

Out[538…

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

In [ ]:
```
plt.figure(figsize = (16,6))
sns.barplot(x='cast', y='title', data=top_actor, width=0.6)
plt.show("Top")
```



Considering the directors in Movies

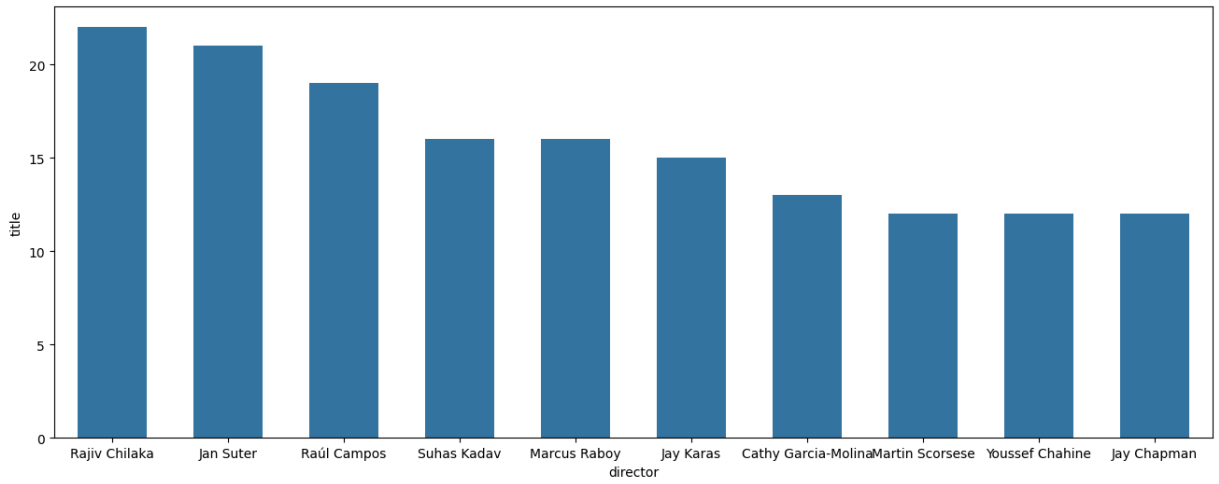
In [531…
```
# We want you to group by each director and find the count of unique titles of Tv-s
top_directors = n_df.groupby('director')['title'].nunique().sort_values(ascending
top_directors.drop(0, inplace = True)
top_directors
```

Out[531…

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

In [540…

```
plt.figure(figsize = (16,6))
sns.barplot(x='director', y='title', data=top_directors, width=0.6)
plt.show("Top")
```



Considering the directors for TV shows

In [541…

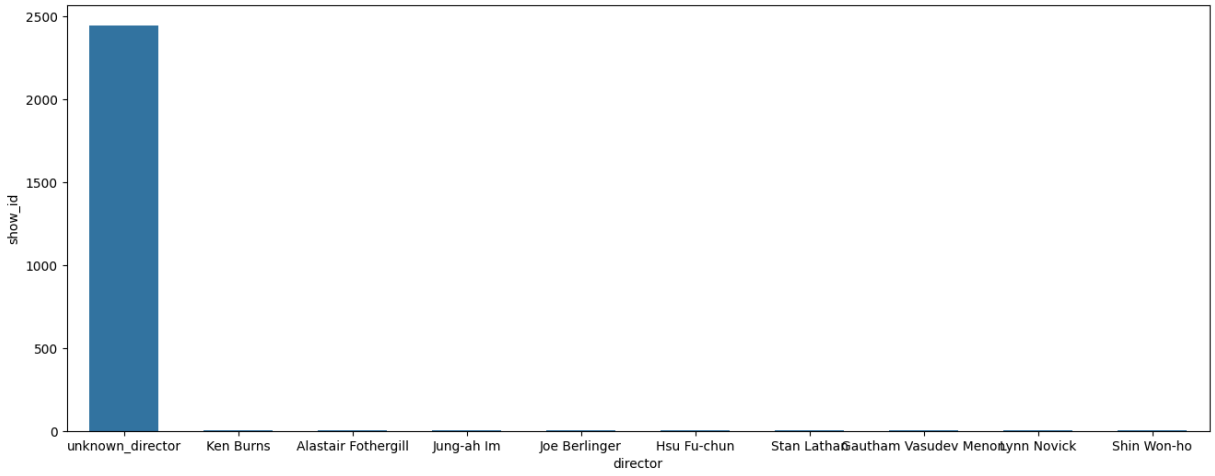
```
dt=TV_Show.groupby(['director'])[['show_id']].nunique().reset_index().sort_values(b
dt
```

Out[541...

	director	show_id
299	unknown_director	2446
146	Ken Burns	3
8	Alastair Fothergill	3
140	Jung-ah Im	2
128	Joe Berlinger	2
100	Hsu Fu-chun	2
259	Stan Lathan	2
84	Gautham Vasudev Menon	2
168	Lynn Novick	2
251	Shin Won-ho	2

In [544...

```
plt.figure(figsize = (16,6))
sns.barplot(x='director', y='show_id', data=dt, width=0.6)
plt.show("Top")
```



Considering the actors in TV shows

In [542...

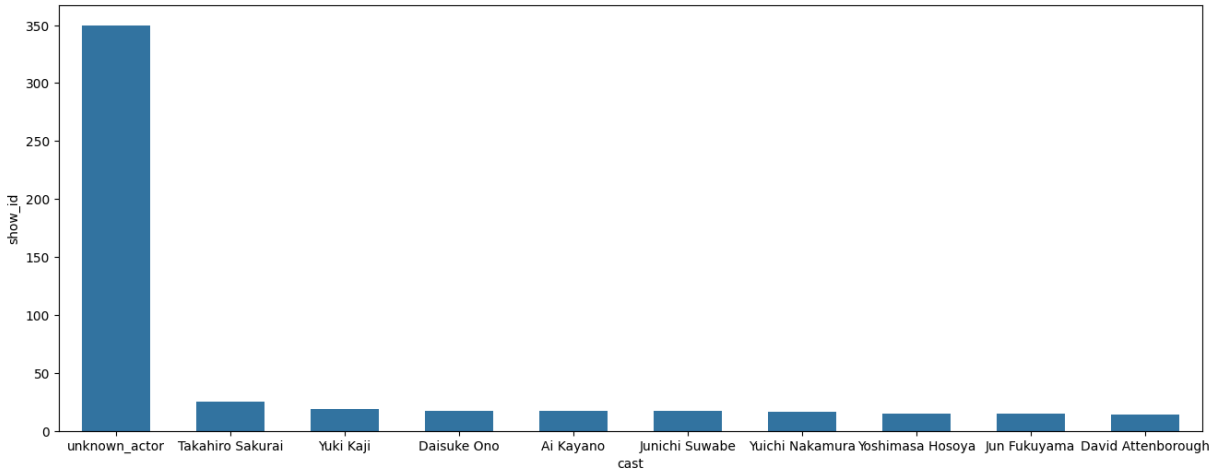
```
at=TV_Show.groupby(['cast'])[['show_id']].nunique().reset_index().sort_values(by='s
at
```

Out[542...

	cast	show_id
14805	unknown_actor	350
13230	Takahiro Sakurai	25
14580	Yuki Kaji	19
2874	Daisuke Ono	17
252	Ai Kayano	17
6804	Junichi Suwabe	17
14564	Yuichi Nakamura	16
14496	Yoshimasa Hosoya	15
6761	Jun Fukuyama	15
3127	David Attenborough	14

In [545...

```
plt.figure(figsize = (16,6))
sns.barplot(x='cast', y='show_id', data=at, width=0.6)
plt.show("Top")
```



Analysis of actors/directors of different types of shows/movies.

Insights: Now it seems we have a different scenario.

By looking at the numbers, it's possible to see that Anupam Kher made 36 appearence on total and 25 of them were on movies.

He's not like to beeing found on TV shows. Meanwhile, Takahiro Sakurai appeared 27 times overall but 21 of them were on TV shows.

Which genre movies are more popular or produced more

We want you to apply the word cloud on the genre columns to know which kind of genre is produced

In [ ]:
```
!pip install wordcloud
```

```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.10/dist-packages
(1.9.3)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.10/dist-packag
es (from wordcloud) (1.25.2)
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (fr
om wordcloud) (9.4.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages
(from wordcloud) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib->wordcloud) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packag
es (from matplotlib->wordcloud) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib->wordcloud) (4.53.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib->wordcloud) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pac
kages (from matplotlib->wordcloud) (24.1)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib->wordcloud) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dis
t-packages (from matplotlib->wordcloud) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages
(from python-dateutil>=2.7->matplotlib->wordcloud) (1.16.0)
```

In [ ]:
```
df_genre=n_df.groupby(['genre','type']).agg({"title":"nunique"}).reset_index()
```

In [ ]:
```
df_genre
```

Out[ ]:

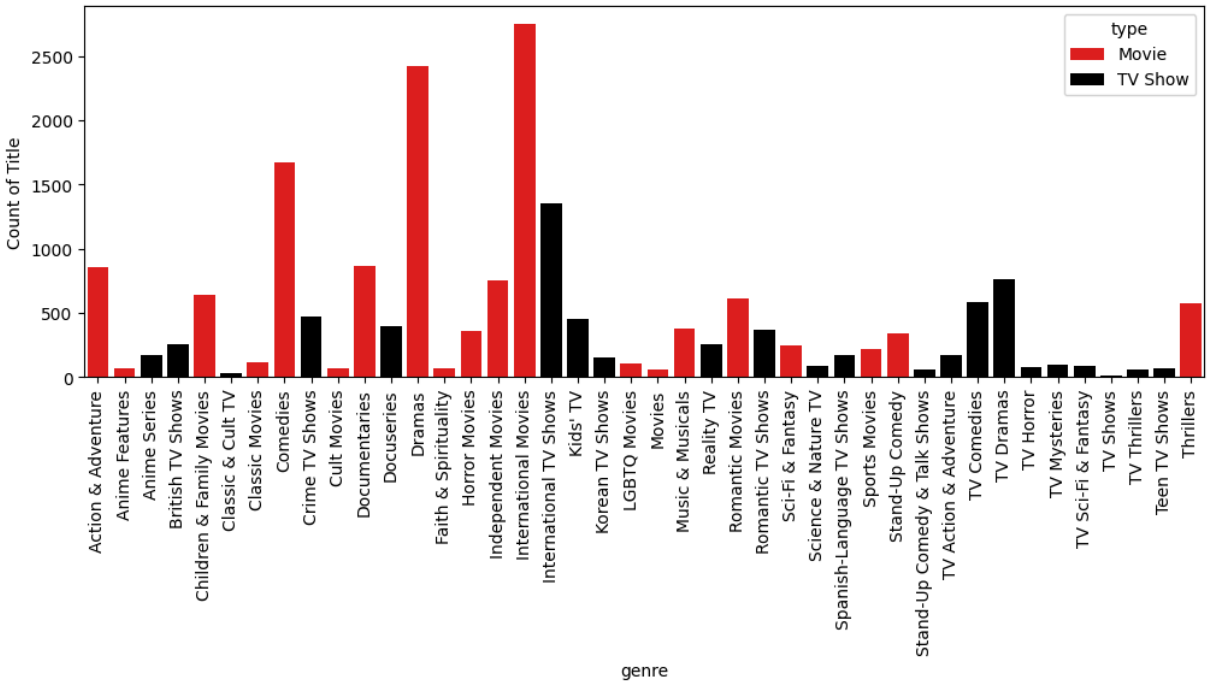
	genre	type	title
0	Action & Adventure	Movie	859
1	Anime Features	Movie	71
2	Anime Series	TV Show	176
3	British TV Shows	TV Show	253
4	Children & Family Movies	Movie	641
5	Classic & Cult TV	TV Show	28
6	Classic Movies	Movie	116
7	Comedies	Movie	1674
8	Crime TV Shows	TV Show	470
9	Cult Movies	Movie	71
10	Documentaries	Movie	869
11	Docuseries	TV Show	395
12	Dramas	Movie	2427
13	Faith & Spirituality	Movie	65
14	Horror Movies	Movie	357
15	Independent Movies	Movie	756
16	International Movies	Movie	2752
17	International TV Shows	TV Show	1351
18	Kids' TV	TV Show	451
19	Korean TV Shows	TV Show	151
20	LGBTQ Movies	Movie	102
21	Movies	Movie	57
22	Music & Musicals	Movie	375
23	Reality TV	TV Show	255
24	Romantic Movies	Movie	616
25	Romantic TV Shows	TV Show	370
26	Sci-Fi & Fantasy	Movie	243
27	Science & Nature TV	TV Show	92
28	Spanish-Language TV Shows	TV Show	174
29	Sports Movies	Movie	219

	genre	type	title
30	Stand-Up Comedy	Movie	343
31	Stand-Up Comedy & Talk Shows	TV Show	56
32	TV Action & Adventure	TV Show	168
33	TV Comedies	TV Show	581
34	TV Dramas	TV Show	763
35	TV Horror	TV Show	75
36	TV Mysteries	TV Show	98
37	TV Sci-Fi & Fantasy	TV Show	84
38	TV Shows	TV Show	16
39	TV Thrillers	TV Show	57
40	Teen TV Shows	TV Show	69
41	Thrillers	Movie	577

In [ ]:
```
from wordcloud import WordCloud
movie_genre = n_df[n_df['type'] == 'Movie']

text = str(list(movie_genre['genre'])).replace(',','').replace("'",'').replace('"',

color = sns.color_palette("dark:red", as_cmap=True)

wordcld = WordCloud(max_words = 150, width = 2000,  height = 800,background_color =

plt.figure(figsize=(15, 7))
plt.imshow(wordcld,interpolation = 'bilinear')
plt.axis('off')
plt.show()
```

In [ ]:
```
TvShow_genre = n_df[n_df['type'] == 'TV Show']

text = str(list(TvShow_genre['genre'])).replace(',','').replace("'",'').replace('"'

color = sns.color_palette("dark:red", as_cmap=True)

wordcld = WordCloud(max_words = 150, width = 2000,  height = 800,background_color =

plt.figure(figsize=(15, 7))
plt.imshow(wordcld,interpolation = 'bilinear')
plt.axis('off')
plt.show()
```

In [ ]:
```
color_palette={'Movie':'red','TV Show':'black'}
plt.figure(figsize=(12,4))
genre_count=sns.barplot(data=df_genre,x='genre',y='title',hue='type',palette=color_
plt.xticks(rotation=90)
plt.ylabel("Count of Title")
plt.show()
```



Insights:

International productions, dramas, and comedies play significant roles in shaping both the movie and TV show landscapes. These genres contribute significantly to the diversity and richness of content available across various platforms, capturing the attention of audiences worldwide.

**Find After how many days the movie will be added to Netflix after the release of the movie (you can consider the recent past data)

Hint : We want you to get the difference between the columns having date added information and release year information and get the mode of difference. This will give an insight into what will be the better time to add in Netflix

In [ ]:
```
n_df['release_year'].unique()
```

Out[ ]:
```
array([2020, 2021, 1993, 2018, 1996, 1998, 1997, 2010, 2013, 2017, 1975,
       1978, 1983, 1987, 2012, 2001, 2014, 2002, 2003, 2004, 2011, 2008,
       2009, 2007, 2005, 2006, 1994, 2015, 2019, 2016, 1982, 1989, 1990,
       1991, 1999, 1986, 1992, 1984, 1980, 1961, 2000, 1995, 1985, 1976,
       1959, 1988, 1981, 1972, 1964, 1945, 1954, 1979, 1958, 1956, 1963,
       1970, 1973, 1925, 1974, 1960, 1966, 1971, 1962, 1969, 1977, 1967,
       1968, 1965, 1946, 1942, 1955, 1944, 1947, 1943])
```

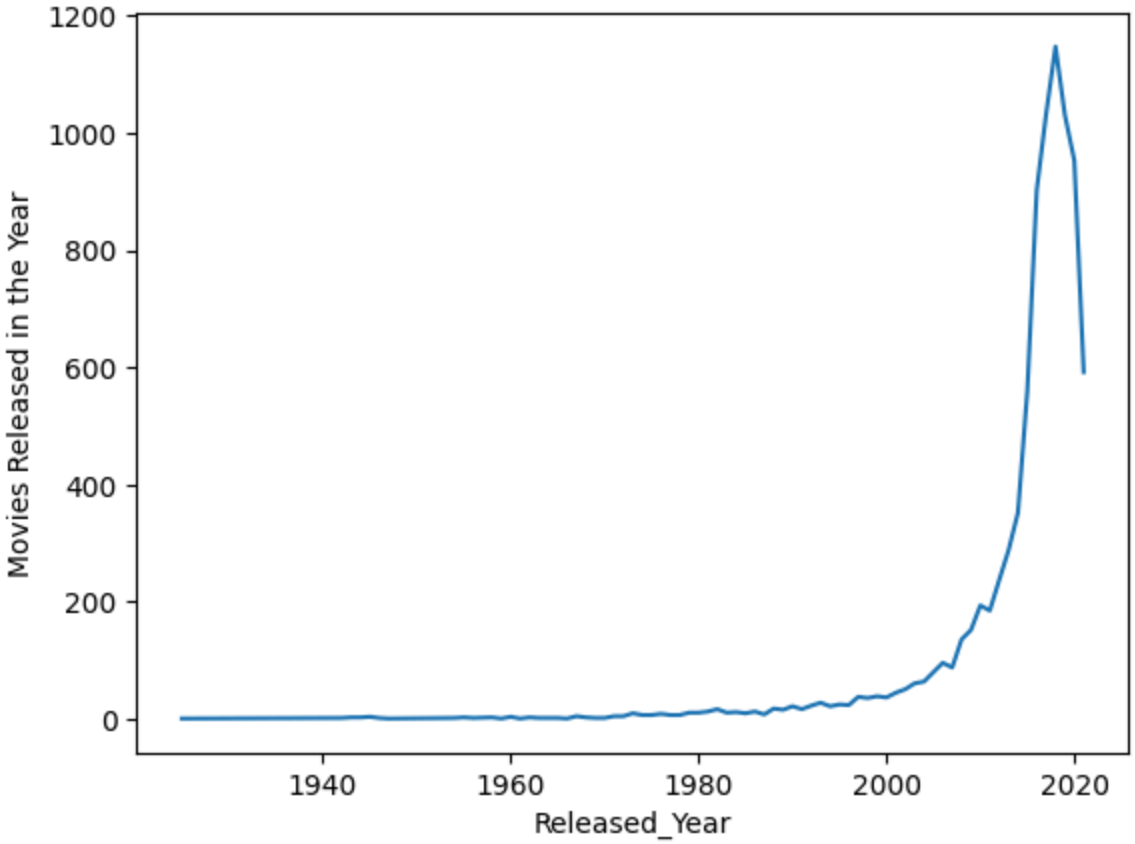
In [ ]:
```
df_year=n_df.groupby(['release_year']).agg({"title":"nunique"}).reset_index()
df_year
```

Out[ ]:

	release_year	title
0	1925	1
1	1942	2
2	1943	3
3	1944	3
4	1945	4
...	...	...
69	2017	1032
70	2018	1147
71	2019	1030
72	2020	953
73	2021	592

74 rows × 2 columns

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



Insights:

The availability of content on Netflix has experienced steady growth since 2008, with a notable surge in new releases after 2015. However, this upward trend was interrupted and experienced a decline during and after the COVID-19 pandemic.

In [ ]:
```
n_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 201936 entries, 0 to 201990
Data columns (total 12 columns):
 #   Column        Non-Null Count   Dtype 
---  ------        --------------   ----- 
 0   title         201936 non-null  object
 1   country       201936 non-null  object
 2   genre         201936 non-null  object
 3   cast          201936 non-null  object
 4   show_id       201936 non-null  object
 5   type          201936 non-null  object
 6   date_added    201936 non-null  object
 7   release_year  201936 non-null  int64 
 8   rating        201936 non-null  object
 9   duration      201936 non-null  object
 10  description   201936 non-null  object
 11  director      201936 non-null  object
dtypes: int64(1), object(11)
memory usage: 20.0+ MB
```

Heat Map Analysis

In [468…
```
Movies['date_added'] = pd.to_datetime(Movies['date_added'], errors='coerce', infer_
```

In [444…
```
Movies['day_added']=Movies['date_added'].dt.day
Movies['month_added']=Movies['date_added'].dt.month
Movies['year_added']=Movies['date_added'].dt.year
```

In [480…
```
Movies['day_name'] = Movies['date_added'].dt.strftime('%A')
```

In [445…
```
Movies.columns
```

Out[445…
```
Index(['title', 'country', 'genre', 'cast', 'show_id', 'type', 'date_added',
       'release_year', 'rating', 'duration', 'description', 'director',
       'day_added', 'month_added', 'year_added'],
      dtype='object')
```

In [481…
```
TV_Show=n_df[n_df['type']=='TV Show']
```

In [482…
```
TV_Show.columns
```

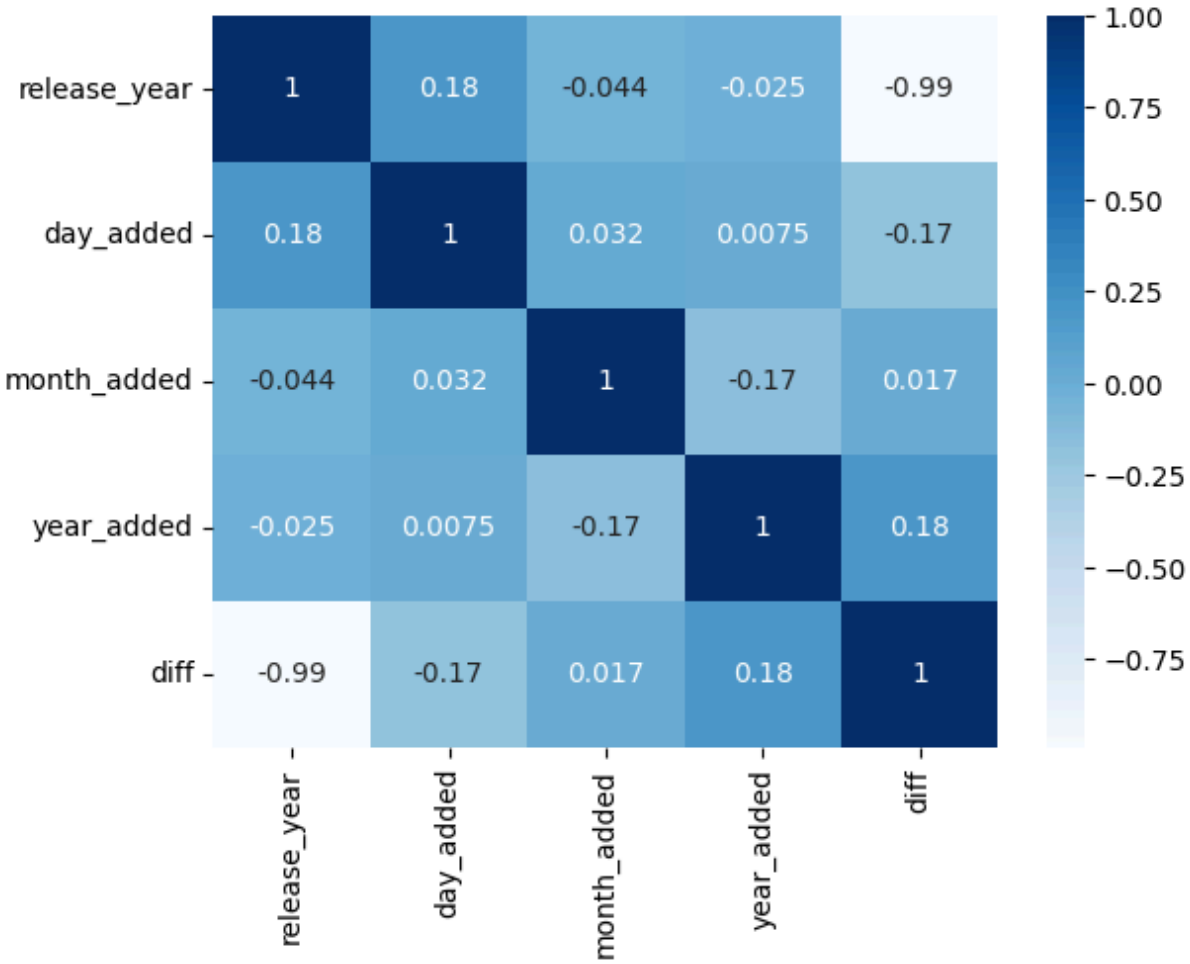
Out[482…
```
Index(['title', 'country', 'genre', 'cast', 'show_id', 'type', 'date_added',
       'release_year', 'rating', 'duration', 'description', 'director'],
      dtype='object')
```

In [485…
```
movies_df = Movies.select_dtypes(include=['number'])

movies_df.corr()
```

Out[485…

	release_year	day_added	month_added	year_added	diff
release_year	1.000000	0.178679	-0.044143	-0.024627	-0.987994
day_added	0.178679	1.000000	0.031945	0.007549	-0.174687
month_added	-0.044143	0.031945	1.000000	-0.170080	0.017161
year_added	-0.024627	0.007549	-0.170080	1.000000	0.178779
diff	-0.987994	-0.174687	0.017161	0.178779	1.000000

In [486…
```
sns.heatmap(movies_df.corr(), cmap= "Blues", annot=True)
plt.show()
```



Insights: This will give an insight into what will be the better time to add in Netflix month_added and year_added