Netflix Overview:

Netflix is a leading global media and video streaming platform renowned for its vast library of movies and TV shows. With over 222 million subscribers worldwide as of mid 2021, Netflix has revolutionized the entertainment industry by providing on-demand access to a diverse range of content. From blockbuster movies to original series, Netflix caters to a broad audience demographic, offering personalized recommendations and an immersive viewing experience. Dataset Description: The dataset provided contains comprehensive information about the TV shows and movies available on Netflix. Here's a breakdown of the key attributes included in the dataset:

1. Show_id: A unique identifier for each movie or TV show listed on Netflix.
2. Type: Indicates whether the entry is a movie or TV show.
3. Title: The title of the movie or TV show.
4. Director: The director(s) responsible for the production.
5. Cast: Actors and actresses involved in the movie or TV show.
6. Country: The country where the movie or TV show was produced.
7. Date_added: The date when the content was added to Netflix.
8. Release_year: The actual release year of the movie or TV show.
9. Rating: The TV rating assigned to the content.
10. Duration: The total duration of the content, either in minutes or number of seasons.
11. Listed_in: Genre classification of the content.
12. Description: A summary description of the movie or TV show.

Problem Statement:

Netflix, a prominent media and video streaming platform with over 222 million global subscribers, seeks to leverage data analytics to enhance its content strategy and drive business growth. The company aims to analyze its extensive dataset, containing information about movies and TV shows available on the platform, to derive actionable insights. Specifically, Netflix wants to identify trends, preferences, and patterns within the data to inform decisions on the types of content to produce and how to expand its market presence across different countries.

Scope of Exploratory Data Analysis (EDA):

The scope of this EDA is to delve into the dataset provided by Netflix, which includes detailed information about the content available on the platform. The analysis will encompass various facets, including but not limited to:

1. Distribution of content across different countries: Understanding the geographical spread of content and identifying regional preferences.

2. Trends in movie releases over the past few decades: Examining how the number and types of movies released on Netflix have evolved over time.
   A. Comparison between TV shows and movies: Analyzing the relative popularity and consumption patterns of TV shows versus movies.
3. Optimal timing for launching TV shows: Identifying patterns in viewer engagement and determining the best times to release new TV shows.
4. Analysis of actors and directors: Exploring the influence of actors and directors on the popularity of content.
5. Focus on TV shows versus movies: Assessing whether Netflix has shifted its focus towards producing more TV shows in recent years. The EDA aims to uncover insights and trends within the data that can guide decision making processes at Netflix. By conducting a comprehensive analysis, Netflix can refine its content strategy, tailor offerings to specific markets, and capitalize on emerging opportunities for growth.

Assumptions:

1. The dataset provided contains accurate and up-to-date information about Netflix's content offerings.
2. External factors such as cultural trends and market dynamics are not explicitly considered in this analysis.
3. The analysis is based solely on the information available in the dataset, and additional data sources are not incorporated at this stage. Through this EDA, Netflix aims to gain a deeper understanding of its audience preferences, optimize content production, and drive continued success in the highly competitive streaming industry

In [37]:

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

In [38]:

```python
df = pd.read_csv("/content/netflix_titles.csv")
df.shape
# There are total 8807 rows and 12 different columns
# in the dataset.
```

Out[38]:

```
(8807, 12)
```

The dataset consists of 8,807 entries with 12 attributes

In [39]:

```python
df.sample(5)
```

Out[39]:

	show_id	type	title	director	cast	country	date_added	release_year
537	s538	TV Show	Terrace House: Opening New Doors	NaN	You, Reina Triendl, Yoshimi Tokui, Azusa Babaz...	Japan	July 6, 2021	2018
4805	s4806	Movie	The Maus	Yayo Herrero	Alma Terzic, August Wittgenstein, Aleksandar S...	Spain	June 30, 2018	2017
3152	s3153	Movie	My Wife and I	Bunmi Ajakaiye	Ramsey Nouah, Omoni Oboli, Dorcas Shola Fapson...	South Africa	December 13, 2019	2017
893	s894	Movie	Wave of Cinema: Filosofi Kopi	Adriano Rudiman	Maliq & D'Essentials, Fourtwnty, Gede Robi, Na...	NaN	May 13, 2021	2020
1849	s1850	Movie	Kartini: Princess of Java	Hanung Bramantyo	Dian Sastrowardoyo, Ayushita, Acha Septriasa, ...	Indonesia, Netherlands	October 15, 2020	2017

In [40]:

```
df.title.nunique()
# The number of unique titles are same as the number of rows in the
# dataframe which suggest that the titles are unique and can be treated
# as a key in future
```

Out[40]:

```
8807
```

In [41]:

```
df.show_id.nunique()
# The number of unique show_id are same as the number of rows in
# the dataframe which suggest that the show_id are unique and can be treated
# as a key in future
```

Out[41]:

```
8807
```

In [42]:

```
df.isna().sum()
```

Out[42]:

```
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
```

In [43]:

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

Out[43]:

	title	cast
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

In [44]:

```
df_cast.shape
```

Out[44]:

```
(64951, 2)
```

In [45]:

```
df_cast[df_cast["cast"]=="nan"].count()
```

Out[45]:

```
title    825
cast     825
dtype: int64
```

In [46]:

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

Out[46]:

	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

In [47]:

```
df_director.shape
```

Out[47]:

```
(9612, 2)
```

In [48]:

```
df_dir_cast = df_director.merge(df_cast,on="title",how="inner")
df_dir_cast
```

Out[48]:

	title	director	cast
0	Dick Johnson Is Dead	Kirsten Johnson	nan
1	Blood & Water	nan	Ama Qamata
2	Blood & Water	nan	Khosi Ngema
3	Blood & Water	nan	Gail Mabalane
4	Blood & Water	nan	Thabang Molaba
...	...	...	...
70807	Zubaan	Mozez Singh	Manish Chaudhary
70808	Zubaan	Mozez Singh	Meghna Malik
70809	Zubaan	Mozez Singh	Malkeet Rauni
70810	Zubaan	Mozez Singh	Anita Shabdish
70811	Zubaan	Mozez Singh	Chittaranjan Tripathy

70812 rows × 3 columns

In [49]:

```
df.columns
```

Out[49]:

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

In [50]:

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

Out[50]:

	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
...	...	...
10845	Zodiac	United States
10846	Zombie Dumb	nan
10847	Zombieland	United States
10848	Zoom	United States
10849	Zubaan	India

10850 rows × 2 columns

In [51]:

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

Out[51]:

	title	listed_in
0	Dick Johnson Is Dead	Documentaries
1	Blood & Water	International TV Shows
2	Blood & Water	TV Dramas
3	Blood & Water	TV Mysteries
4	Ganglands	Crime TV Shows
...	...	...
19318	Zoom	Children & Family Movies
19319	Zoom	Comedies
19320	Zubaan	Dramas
19321	Zubaan	International Movies
19322	Zubaan	Music & Musicals

19323 rows × 2 columns

In [52]:

```
df_country_list = df_country.merge(df_listed_in
                                   ,on="title",how="inner")
df_country_list
```

Out[52]:

	title	country	listed_in
0	Dick Johnson Is Dead	United States	Documentaries
1	Blood & Water	South Africa	International TV Shows
2	Blood & Water	South Africa	TV Dramas
3	Blood & Water	South Africa	TV Mysteries
4	Ganglands	nan	Crime TV Shows
...	...	...	...
23759	Zoom	United States	Children & Family Movies
23760	Zoom	United States	Comedies
23761	Zubaan	India	Dramas
23762	Zubaan	India	International Movies
23763	Zubaan	India	Music & Musicals

23764 rows × 3 columns

In [53]:

```
df_country_list_dir_cast = df_country_list.merge(df_dir_cast
                                                  ,on="title",how="inner")
df_country_list_dir_cast
```

Out[53]:

	title	country	listed_in	director	cast
0	Dick Johnson Is Dead	United States	Documentaries	Kirsten Johnson	nan
1	Blood & Water	South Africa	International TV Shows	nan	Ama Qamata
2	Blood & Water	South Africa	International TV Shows	nan	Khosi Ngema
3	Blood & Water	South Africa	International TV Shows	nan	Gail Mabalane
4	Blood & Water	South Africa	International TV Shows	nan	Thabang Molaba
...	...	...	...	...	...
202060	Zubaan	India	Music & Musicals	Mozez Singh	Manish Chaudhary
202061	Zubaan	India	Music & Musicals	Mozez Singh	Meghna Malik
202062	Zubaan	India	Music & Musicals	Mozez Singh	Malkeet Rauni
202063	Zubaan	India	Music & Musicals	Mozez Singh	Anita Shabdish
202064	Zubaan	India	Music & Musicals	Mozez Singh	Chittaranjan Tripathy

202065 rows × 5 columns

In [54]:

```
df.columns
```

Out[54]:

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

Final dataframe created merging country,listed_in,director and cast

In [55]:

```
df_final = df_country_list_dir_cast.merge(df[['show_id', 'type',
                                               'title','date_added',
       'release_year', 'rating', 'duration', 'description']]
                                          ,on="title",how="inner")
df_final.head(3)
```

Out[55]:

	title	country	listed_in	director	cast	show_id	type	date_added	release_year
0	Dick Johnson Is Dead	United States	Documentaries	Kirsten Johnson	nan	s1	Movie	September 25, 2021	2020
1	Blood & Water	South Africa	International TV Shows	nan	Ama Qamata	s2	TV Show	September 24, 2021	2021
2	Blood & Water	South Africa	International TV Shows	nan	Khosi Ngema	s2	TV Show	September 24, 2021	2021

In [56]:

```
df_final.isna().sum()
```

Out[56]:

```
title             0
country           0
listed_in         0
director          0
cast              0
show_id           0
type              0
date_added      158
release_year      0
rating           67
duration          3
description       0
dtype: int64
```

In [57]:

```
df_final[df_final["duration"].isnull()]

#Shows the 3 wrong values in "rating" and the 3 missing values in "duration"
```

Out[57]:

	title	country	listed_in	director	cast	show_id	type	date_added	release_year	ra
126582	Louis C.K. 2017	United States	Movies	Louis C.K.	Louis C.K.	s5542	Movie	April 4, 2017	2017	
131648	Louis C.K.: Hilarious	United States	Movies	Louis C.K.	Louis C.K.	s5795	Movie	September 16, 2016	2010	
131782	Louis C.K.: Live at the Comedy Store	United States	Movies	Louis C.K.	Louis C.K.	s5814	Movie	August 15, 2016	2015	

In [58]:

```
df_final["duration"].fillna(df_final["rating"]
                            ,inplace=True)

# filling the 3 missing values in "duration" by the
# corresponding values in "rating"
```

In [59]:

```
df_final.isna().sum()

# We see that 3 missing values in "duration" are filled now
```

Out[59]:

```
title             0
country           0
listed_in         0
director          0
cast              0
show_id           0
type              0
date_added      158
release_year      0
rating           67
duration          0
description       0
dtype: int64
```

In [60]:

```
df_final.loc[126582,"rating"]="unknown rating"
df_final.loc[131648,"rating"]="unknown rating"
df_final.loc[131782,"rating"]="unknown rating"

# This fills the 3 rows in "rating" having wrong values from
# corresponding "duration" column , with "unknown"
```

In [61]:

```
df_final.isna().sum()
```

Out[61]:
```
title             0
country           0
listed_in         0
director          0
cast              0
show_id           0
type              0
date_added      158
release_year      0
rating           67
duration          0
description       0
dtype: int64
```

In [62]:
```
df_final.head(3)
```

Out[62]:

	title	country	listed_in	director	cast	show_id	type	date_added	release_year
0	Dick Johnson Is Dead	United States	Documentaries	Kirsten Johnson	nan	s1	Movie	September 25, 2021	2020
1	Blood & Water	South Africa	International TV Shows	nan	Ama Qamata	s2	TV Show	September 24, 2021	2021
2	Blood & Water	South Africa	International TV Shows	nan	Khosi Ngema	s2	TV Show	September 24, 2021	2021

In [68]:
```
df_final["mode_year"]= df_final.groupby('release_year')["date_added"].transform(
    lambda x: x.mode().iloc[0])
df_final.sample(5)

# Creates a new column "mode_year" which has the most
# frequent date added for each release year.
```

Out[68]:

	title	country	listed_in	director	cast	show_id	type	date_added	rel
19658	Attack on Titan	Japan	Anime Series	nan	Saki Fujita	s779	TV Show	June 2, 2021	
9089	Eyes of a Thief	France	Independent Movies	Najwa Najjar	Souad Massi	s366	Movie	July 30, 2021	
118630	DreamWorks Home: For the Holidays	United States	Children & Family Movies	nan	Matt Jones	s5141	Movie	December 1, 2017	
25519	Four Sisters Before the Wedding	Philippines	Children & Family Movies	Mae Czarina Cruz	Irma Adlawan	s1031	Movie	April 16, 2021	
182062	Shakti: The Power	India	Thrillers	Krishna Vamshi	Divya Dutta	s7995	Movie	March 1, 2018	

In [69]:

```
df_final["date_added"].fillna(df_final["mode_year"],inplace=True)

# Fills the missing values in "date_added" with the mode of "date_added"
# for the corresponding release year.
```

In [70]:

```
df_final.isna().sum()

#  We see that the 158 missing values in "date_added" are now filled.
```

Out[70]:

```
title            0
country          0
listed_in        0
director         0
cast             0
show_id          0
type             0
date_added       0
release_year     0
rating          67
duration         0
description      0
mode_year        0
dtype: int64
```

In [ ]:

```
# df_final.groupby('release_year').apply(lambda x: x["date_added"].fillna(x["releas
```

In [71]:

```
df_final.isna().sum()
```

```
Out[71]: title            0
         country          0
         listed_in        0
         director         0
         cast             0
         show_id          0
         type             0
         date_added       0
         release_year     0
         rating          67
         duration         0
         description      0
         mode_year        0
         dtype: int64
```

Lets fix the 67 missing values in "rating" Now

```
In [72]: df_final[df_final["rating"].isna()]
```

Out[72]:

	title	country	listed_in	director	cast	show_id	type	date_added	relea
135172	13TH: A Conversation with Oprah Winfrey & Ava ...	nan	Movies	nan	Oprah Winfrey	s5990	Movie	January 26, 2017	
135173	13TH: A Conversation with Oprah Winfrey & Ava ...	nan	Movies	nan	Ava DuVernay	s5990	Movie	January 26, 2017	
154424	Gargantia on the Verdurous Planet	Japan	Anime Series	nan	Kaito Ishikawa	s6828	TV Show	December 1, 2016	
154425	Gargantia on the Verdurous Planet	Japan	Anime Series	nan	Hisako Kanemoto	s6828	TV Show	December 1, 2016	
154426	Gargantia on the Verdurous Planet	Japan	Anime Series	nan	Ai Kayano	s6828	TV Show	December 1, 2016	
...	...	...	...	...	...	...	...	...	
172016	My Honor Was Loyalty	Italy	Dramas	Alessandro Pepe	Francesco Migliore	s7538	Movie	March 1, 2017	
172017	My Honor Was Loyalty	Italy	Dramas	Alessandro Pepe	Albrecht Weimer	s7538	Movie	March 1, 2017	
172018	My Honor Was Loyalty	Italy	Dramas	Alessandro Pepe	Giulia Dichiaro	s7538	Movie	March 1, 2017	
172019	My Honor Was Loyalty	Italy	Dramas	Alessandro Pepe	Alessandra Oriti Niosi	s7538	Movie	March 1, 2017	
172020	My Honor Was Loyalty	Italy	Dramas	Alessandro Pepe	Andreas Segeritz	s7538	Movie	March 1, 2017	

67 rows × 13 columns

In [73]:

```
df_final["rating"].fillna("unknown rating",inplace=True)

#This replaces the missing values in "rating" with "unknown rating"
```

In [74]:

```
df_final.isna().sum()
# We see that there are no missing values now but
# there are string null "nan" in "director","cast","country"
```

Out[74]:

```
title           0
country         0
listed_in       0
director        0
cast            0
show_id         0
type            0
date_added      0
release_year    0
rating          0
duration        0
description     0
mode_year       0
dtype: int64
```

In [76]:

```
df_final.eq("nan").sum()

# We see that there are no missing values now but there
# are string null "nan" in "director","cast","country"
```

Out[76]:

```
title               0
country         11897
listed_in           0
director        50643
cast             2149
show_id             0
type                0
date_added          0
release_year        0
rating              0
duration            0
description         0
mode_year           0
dtype: int64
```

In [77]:

```
df_final["cast"].replace(['nan'],['unknown actor'],inplace=True)
df_final["director"].replace(['nan'],['unknown director'],inplace=True)
df_final.sample(10)

#Replaces "nan" in "cast" and "director" columns with "unknown
#actor" and "unknown director" respectively
```

Out[77]:

	title	country	listed_in	director	cast	show_id	type	date_ad
9436	All American	United States	TV Dramas	unknown director	Taye Diggs	s383	TV Show	July 2
43372	Heritages	Switzerland	Documentaries	Philippe Aractingi	Diane Aractingi	s1821	Movie	October 2
105222	Edgar Rice Burroughs' Tarzan and Jane	nan	Crime TV Shows	unknown director	Paul Dobson	s4512	TV Show	October 2
9574	The Operative	Israel	Dramas	Yuval Adler	Cas Anvar	s390	Movie	July 2
185668	Surat Dari Praha	Indonesia	Romantic Movies	Angga Dwimas Sasongko	Tio Pakusadewo	s8132	Movie	Octob 2
175497	Pasión de Gavilanes	Colombia	Romantic TV Shows	unknown director	Gloria Gómez	s7711	TV Show	May 4, 2
772	Ankahi Kahaniya	nan	Independent Movies	Saket Chaudhary	Delzad Hiwale	s31	Movie	Septem 17, 2
144807	Cappuccino	India	Dramas	Noushad	Sharanya	s6419	Movie	July 1, 2
148234	Dad	United States	Dramas	Gary David Goldberg	Ethan Hawke	s6549	Movie	Ap 2
5299	Shootout at Lokhandwala	India	Action & Adventure	Apoorva Lakhia	Amitabh Bachchan	s216	Movie	August 2

In [78]:

```
# At this moment "mode_year" is no longer needed,
# so lets get rid of this column
df_final.drop(["mode_year"],axis=1,inplace=True)
```

In [79]:

```
df_final.head(2)
```

Out[79]:

	title	country	listed_in	director	cast	show_id	type	date_added	release_year
0	Dick Johnson Is Dead	United States	Documentaries	Kirsten Johnson	unknown actor	s1	Movie	September 25, 2021	2020
1	Blood & Water	South Africa	International TV Shows	unknown director	Ama Qamata	s2	TV Show	September 24, 2021	2021

Lets work on filling the "nan" values in "country" column with director's country and if director's country is missing, we fill it with corresponding cast's country

In [80]:

```
df_final["director_country"]=df_final.groupby(
    "director")["country"].transform(lambda x : x.mode().iloc[0])
df_final["cast_country"]=df_final.groupby(
    "cast")["country"].transform(
        lambda x: x.mode().iloc[0]
        if not x.mode().isnull().any() else x.value_counts().index[1])
df_final.head()

# This creates column "director_country" containing
# the mode of the country for each director
# This creates column "cast_country" containing the mode of the country
# for each cast and if the mode is null, it contains the next most frequent value
```

Out[80]:

	title	country	listed_in	director	cast	show_id	type	date_added	release_yea
0	Dick Johnson Is Dead	United States	Documentaries	Kirsten Johnson	unknown actor	s1	Movie	September 25, 2021	202
1	Blood & Water	South Africa	International TV Shows	unknown director	Ama Qamata	s2	TV Show	September 24, 2021	202
2	Blood & Water	South Africa	International TV Shows	unknown director	Khosi Ngema	s2	TV Show	September 24, 2021	202
3	Blood & Water	South Africa	International TV Shows	unknown director	Gail Mabalane	s2	TV Show	September 24, 2021	202
4	Blood & Water	South Africa	International TV Shows	unknown director	Thabang Molaba	s2	TV Show	September 24, 2021	202

Lets create a slice of the dataframe having "nan" values in "country"

In [81]:

```
df_slice = df_final.loc[df_final["country"]=="nan"].copy()
df_slice
```

Out[81]:

	title	country	listed_in	director	cast	show_id	type	date_added	release_y
58	Ganglands	nan	Crime TV Shows	Julien Leclercq	Sami Bouajila	s3	TV Show	September 24, 2021	20
59	Ganglands	nan	Crime TV Shows	Julien Leclercq	Tracy Gotoas	s3	TV Show	September 24, 2021	20
60	Ganglands	nan	Crime TV Shows	Julien Leclercq	Samuel Jouy	s3	TV Show	September 24, 2021	20
61	Ganglands	nan	Crime TV Shows	Julien Leclercq	Nabiha Akkari	s3	TV Show	September 24, 2021	20
62	Ganglands	nan	Crime TV Shows	Julien Leclercq	Sofia Lesaffre	s3	TV Show	September 24, 2021	20
...	...	...	...	...	...	...	...	...	
201498	YOM	nan	Kids' TV	unknown director	Mayur Vyas	s8786	TV Show	June 7, 2018	20
201499	YOM	nan	Kids' TV	unknown director	Ketan Kava	s8786	TV Show	June 7, 2018	20
202006	Zombie Dumb	nan	Kids' TV	unknown director	unknown actor	s8804	TV Show	July 1, 2019	20
202007	Zombie Dumb	nan	Korean TV Shows	unknown director	unknown actor	s8804	TV Show	July 1, 2019	20
202008	Zombie Dumb	nan	TV Comedies	unknown director	unknown actor	s8804	TV Show	July 1, 2019	20

11897 rows × 14 columns

In [82]:
```
df_slice["country"]=df_slice.apply(
    lambda x : x["director_country"] if x["director_country"]!="nan"
    else (x["cast_country"] if x["cast_country"]!="nan"
          else "unknown country" ), axis=1)
# updates the "country" column with "director_country" if it is not "nan"
# else updates with "cast_country" if it is not "nan" else
# updates with "unknown country"
```

In [83]:
```
df_slice[df_slice["country"]=="nan"]
```

Out[83]:

title	country	listed_in	director	cast	show_id	type	date_added	release_year	rating	duratio

In [84]:
```
df_final.update(df_slice)
df_final.sample(5)
# updates the original dataframe with the slice of the dataframe and reflect the ch
```

Out[84]:

	title	country	listed_in	director	cast	show_id	type	date_added	r
51441	Sin City	unknown country	International Movies	Pascal Amanfo	Yvonne Nelson	s2165	Movie	August 5, 2020	
184811	Stranger than Fiction	United Kingdom	Comedies	Marc Forster	Kristin Chenoweth	s8102	Movie	October 1, 2020	
53008	The Millions	Nigeria	Comedies	Toka McBaror	Blossom Chukwujekwu	s2232	Movie	July 17, 2020	
117367	Disjointed	United States	TV Comedies	unknown director	Elizabeth Alderfer	s5083	TV Show	January 12, 2018	
57046	From A to B	Lebanon	Comedies	Ali F. Mostafa	Ahd	s2403	Movie	June 11, 2020	

In [85]:
```
df_final[df_final["country"]=="nan"]
```

Out[85]:

title	country	listed_in	director	cast	show_id	type	date_added	release_year	rating	duratio

In [86]:
```
df_final.drop(["director_country","cast_country"],axis=1,inplace=True)
```

In [87]:
```
df_final.eq("nan").sum()
# So we see that there are no "nan" values in any of the columns now
```

Out[87]:

```
title           0
country         0
listed_in       0
director        0
cast            0
show_id         0
type            0
date_added      0
release_year    0
rating          0
duration        0
description     0
dtype: int64
```

Data Cleaning Over*

In [88]:

```
df_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 202065 entries, 0 to 202064
Data columns (total 12 columns):
 #   Column        Non-Null Count   Dtype 
---  ------        --------------   ----- 
 0   title         202065 non-null  object
 1   country       202065 non-null  object
 2   listed_in     202065 non-null  object
 3   director      202065 non-null  object
 4   cast          202065 non-null  object
 5   show_id       202065 non-null  object
 6   type          202065 non-null  object
 7   date_added    202065 non-null  object
 8   release_year  202065 non-null  int64 
 9   rating        202065 non-null  object
 10  duration      202065 non-null  object
 11  description   202065 non-null  object
dtypes: int64(1), object(11)
memory usage: 18.5+ MB
```

In [89]:

```
df_final.sample(5)
```

Out[89]:

	title	country	listed_in	director	cast	show_id	type	date_added	release
51100	GAME ON: A Comedy Crossover Event	United States	Kids' TV	unknown director	Tia Mowry-Hardrict	s2145	TV Show	August 10, 2020	
106753	My Friend Pinto	India	International Movies	Raaghav Dar	Karim Hajee	s4589	Movie	October 1, 2018	
74025	The Vendor	unknown country	Comedies	Odunlade Adekola	Tunde Bernard	s3094	Movie	December 27, 2019	
35366	Jenni Rivera: Mariposa de Barrio	United States	Spanish-Language TV Shows	unknown director	Tony Garza	s1458	TV Show	January 1, 2021	
10534	2 Weeks in Lagos	Nigeria	International Movies	Kathryn Fasegha	Okey Uzoeshi	s439	Movie	July 16, 2021	

In [90]:

```
df_final.nunique()
```

Out[90]:

```
title            8807
country           198
listed_in          73
director         5121
cast            39297
show_id          8807
type                2
date_added       1767
release_year       74
rating             15
duration          220
description      8775
dtype: int64
```

Observation : The dataset contains entries for 8807 movies and TV shows and covers around 198 different countries.

In [91]:

```
df_final.describe(include="object")
```

Out[91]:

	title	country	listed_in	director	cast	show_id	type	date_added	rating
count	202065	202065	202065	202065	202065	202065	202065	202065	202065
unique	8807	198	73	5121	39297	8807	2	1767	15
top	Kahlil Gibran's The Prophet	United States	International Movies	unknown director	unknown actor	s7165	Movie	January 1, 2020	TV-MA
freq	700	55428	27141	50643	2149	700	145917	3730	73915

Observations:

- The United states appears to be the most popular country for content production.
- The platform focuses on mature audiences since the "TV-MA" is most frequent in rating.
- The platform focuses more on movies compared to the TV shows since Movies is most frequency in type category.

In [92]:

```
df_final.describe()
```

Out[92]:

	release_year
count	202065.000000
mean	2013.448950
std	9.013616
min	1925.000000
25%	2012.000000
50%	2016.000000
75%	2019.000000
max	2021.000000

Observation: The dataset contains movies and TV-shows released between 1925 to 2021.Nearly half the content present on the platform has been added after 2016.

In [93]:

```
df_final.groupby("type")["title"].nunique()
```

Out[93]:

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

Observations:

- Total number of movies: 6131
- Total number of TV Shows: 2676

In [94]:

```
df_final.groupby("country")["title"].nunique().sort_values(ascending=False)[:10]
```

```
Out[94]: country
         United States      3749
         India              1048
         United Kingdom      650
          United States      490
         unknown country     283
         Canada              279
         Japan               273
         France              216
         South Korea         213
          France             192
         Name: title, dtype: int64
```

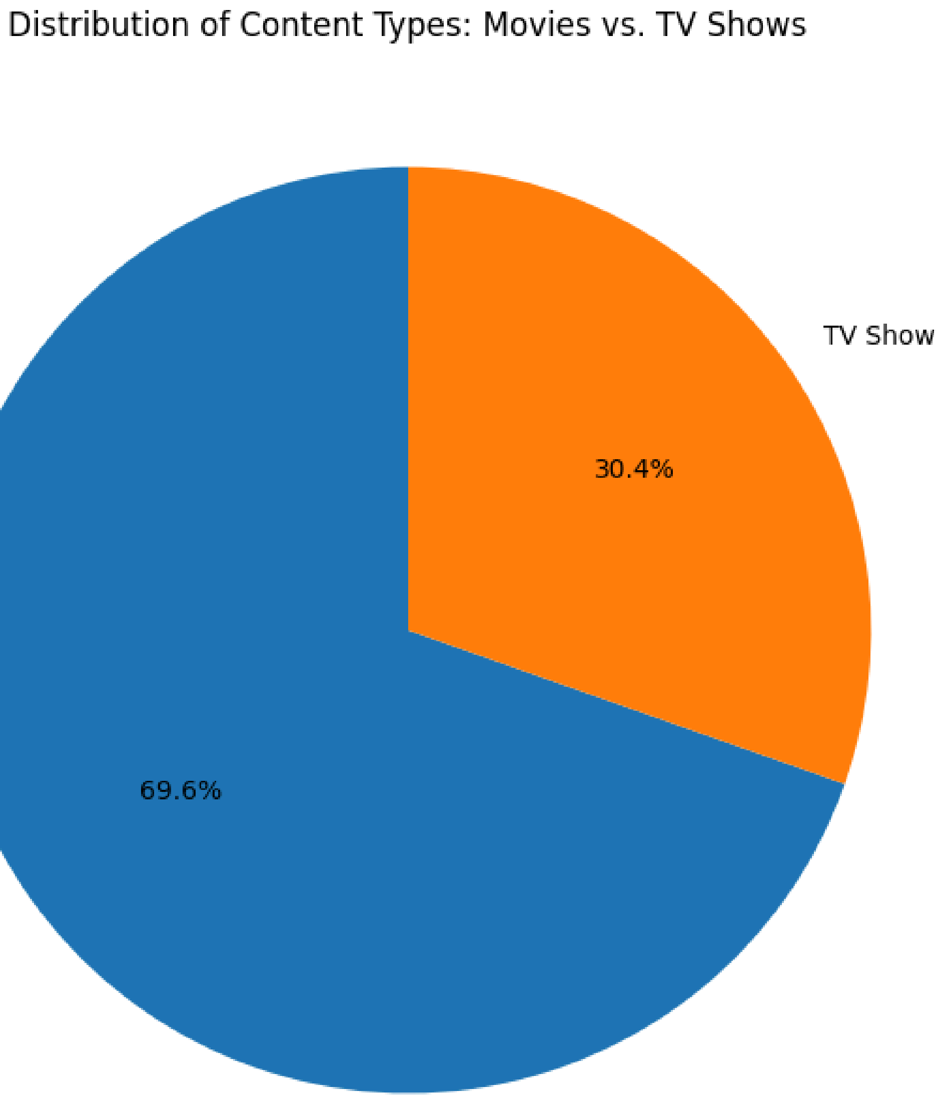
Observation: Top 10 countries producing content where the unites states tops the list followed by india.

```
In [95]: df_final.groupby("rating")["title"].nunique().sort_values(ascending=False)[:10]
```

```
Out[95]: 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
         Name: title, dtype: int64
```

Observation: Top 10 categories of content on the platform where the content for matured audieces tops the list.

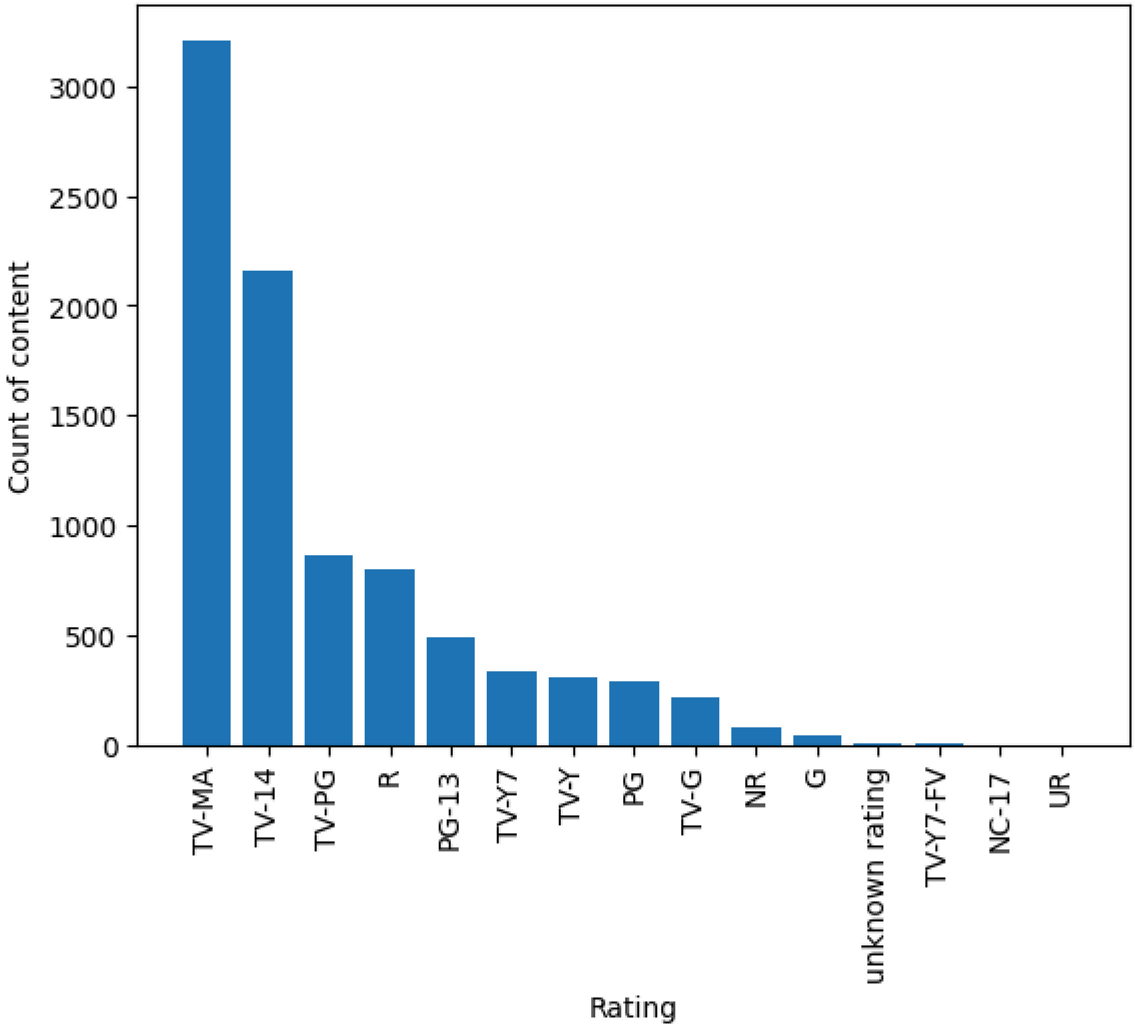
```
In [96]: type_counts = df_final.groupby("type")["title"].nunique()
         labels = type_counts.index
         sizes = type_counts.values

         plt.figure(figsize=(8, 8))
         plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90)
         plt.title('Distribution of Content Types: Movies vs. TV Shows')
         plt.show()
```



Observation : The count of Movies is significantly higher than that of TV Shows, indicating that Netflix has a more extensive catalog of movies.

In [97]:
```
count_of_content = df_final.groupby("rating")["title"].nunique().values
genre = df_final.groupby("rating")["title"].nunique().index
data = list(zip(genre, count_of_content))
data_sorted = sorted(data, key=lambda x: x[1], reverse=True)
genre_sorted, count_of_content_sorted = zip(*data_sorted)
genre_sorted = np.array(genre_sorted)
count_of_content_sorted = np.array(count_of_content_sorted)
```

In [98]:
```
x_bar=genre_sorted
y_bar = count_of_content_sorted
plt.bar(x_bar, y_bar)
plt.xticks(rotation=90)
plt.xlabel("Rating")
plt.ylabel("Count of content")
plt.show()
```



Observation : The majority of the content is rated "TV-MA" followed by "TV-14", indicating a focus on mature audiences and teenagers.

Top 10 Most Frequent Directors on Netflix

In [100…

```
df_final.groupby("director")["title"].nunique().sort_values(
    ascending=False)[1:10]
```

Out[100]:

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

Observation: A large number of director names are missing in the dataset. Of the known names, rajiv chilaka has directed moost number of movies and TV shows.

In [101…

```
num_of_mov = df_final.groupby("director")["title"].nunique().sort_values(
    ascending=False)[1:10].values
name_of_dir = df_final.groupby("director")["title"].nunique().sort_values(
    ascending=False)[1:10].index
name_of_dir
```

Out[101]:

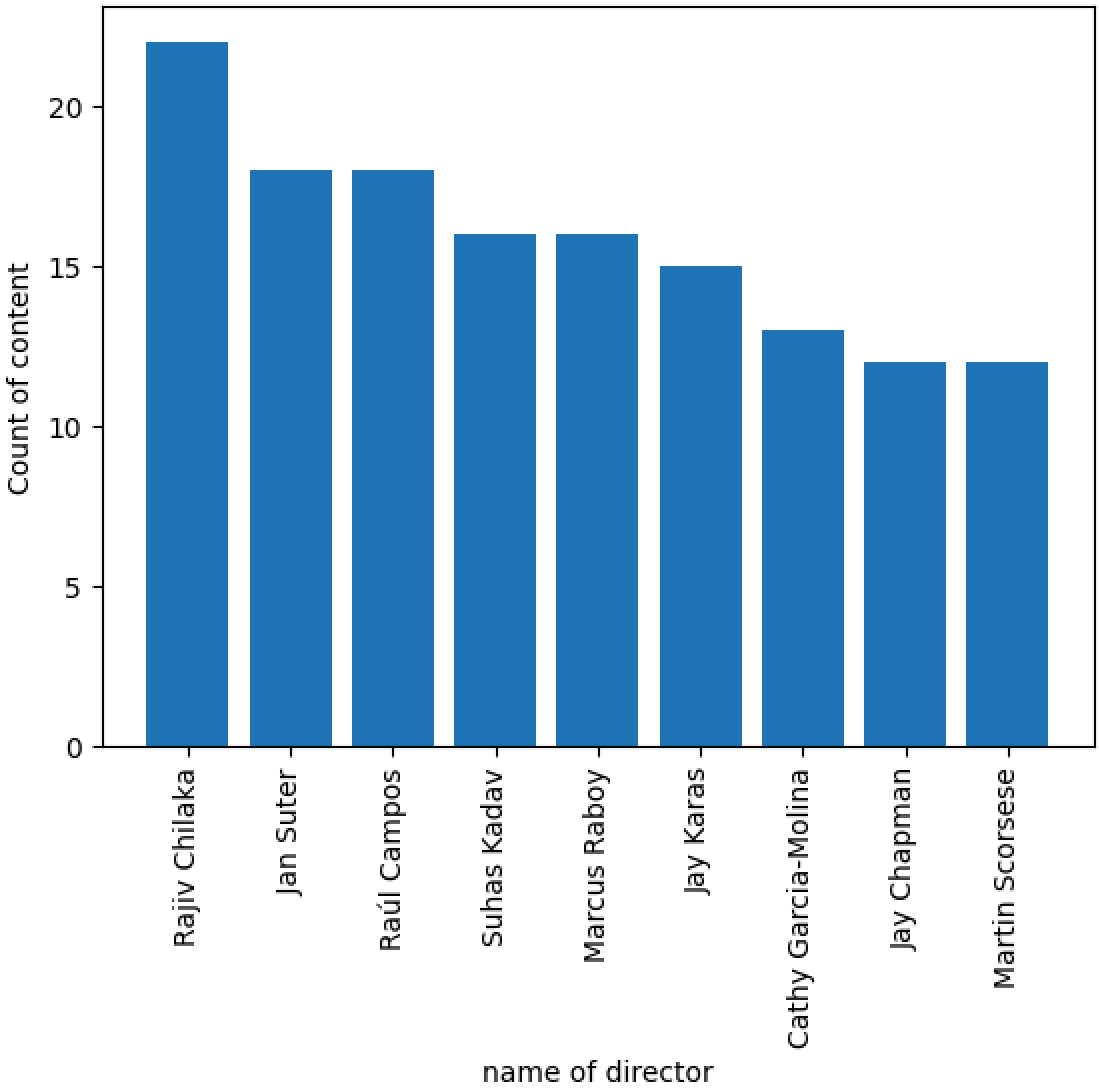
```
Index(['Rajiv Chilaka', ' Jan Suter', 'Raúl Campos', 'Suhas Kadav',
       'Marcus Raboy', 'Jay Karas', 'Cathy Garcia-Molina', 'Jay Chapman',
       'Martin Scorsese'],
      dtype='object', name='director')
```

In [102…

```
num_of_mov = df_final.groupby("director")["title"].nunique().sort_values(
    ascending=False)[1:10].values
name_of_dir = df_final.groupby("director")["title"].nunique().sort_values(
    ascending=False)[1:10].index
data_dir = list(zip(name_of_dir, num_of_mov))
data_dir_sorted = sorted(data_dir, key=lambda x: x[1], reverse=True)
name_of_dir_sorted, num_of_mov_sorted = zip(*data_dir_sorted)
name_of_dir_sorted = np.array(name_of_dir_sorted)
num_of_mov_sorted = np.array(num_of_mov_sorted)
```

In [103…

```
x_bar=name_of_dir_sorted
y_bar = num_of_mov_sorted
plt.bar(x_bar, y_bar)
plt.xticks(rotation=90)
plt.xlabel("name of director")
plt.ylabel("Count of content")
plt.show()
```



Observation: A large number of director names are missing in the dataset. Of the known names, rajiv chilaka has directed moost number of movies and TV shows.

In [104…

```
df_temp1 = df_final[["title","release_year"]]
df_temp_unique= df_temp1.drop_duplicates(keep='first')
```

In [105…

```
df_temp_unique
```

Out[105]:

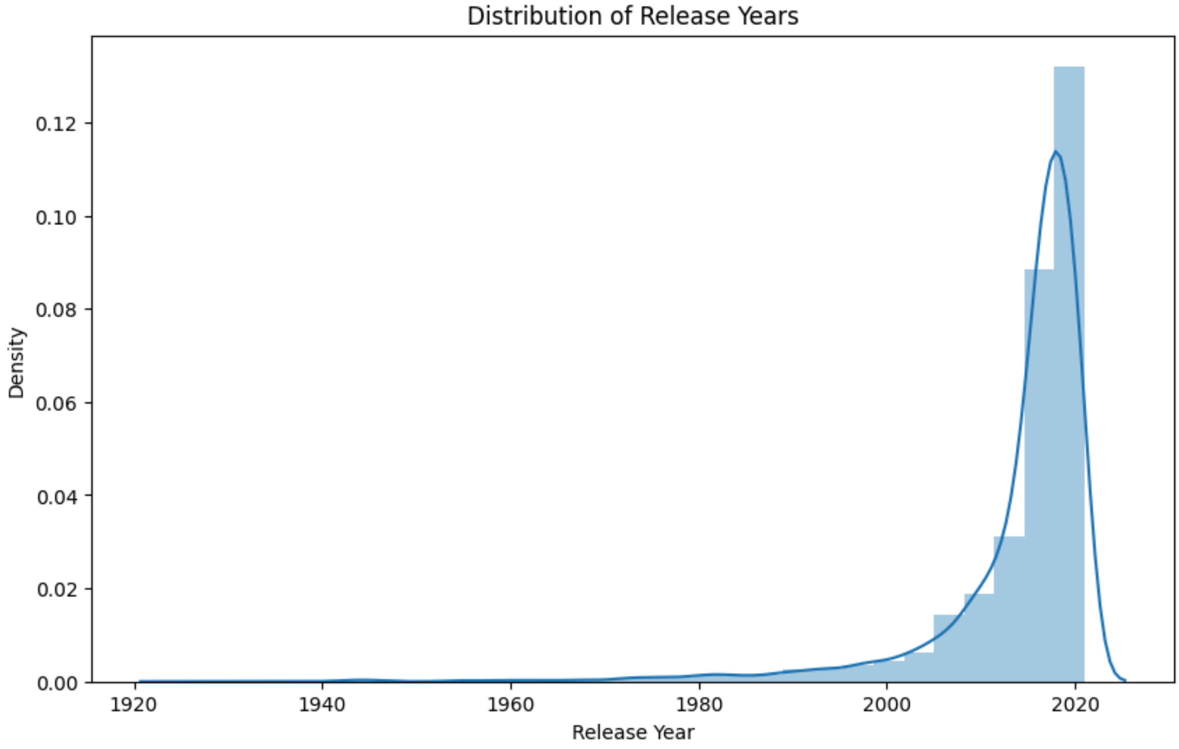
	title	release_year
0	Dick Johnson Is Dead	2020
1	Blood & Water	2021
58	Ganglands	2021
85	Jailbirds New Orleans	2021
87	Kota Factory	2021
...	...	...
201976	Zodiac	2007
202006	Zombie Dumb	2018
202009	Zombieland	2009
202023	Zoom	2006
202041	Zubaan	2015

8807 rows × 2 columns

In [106…

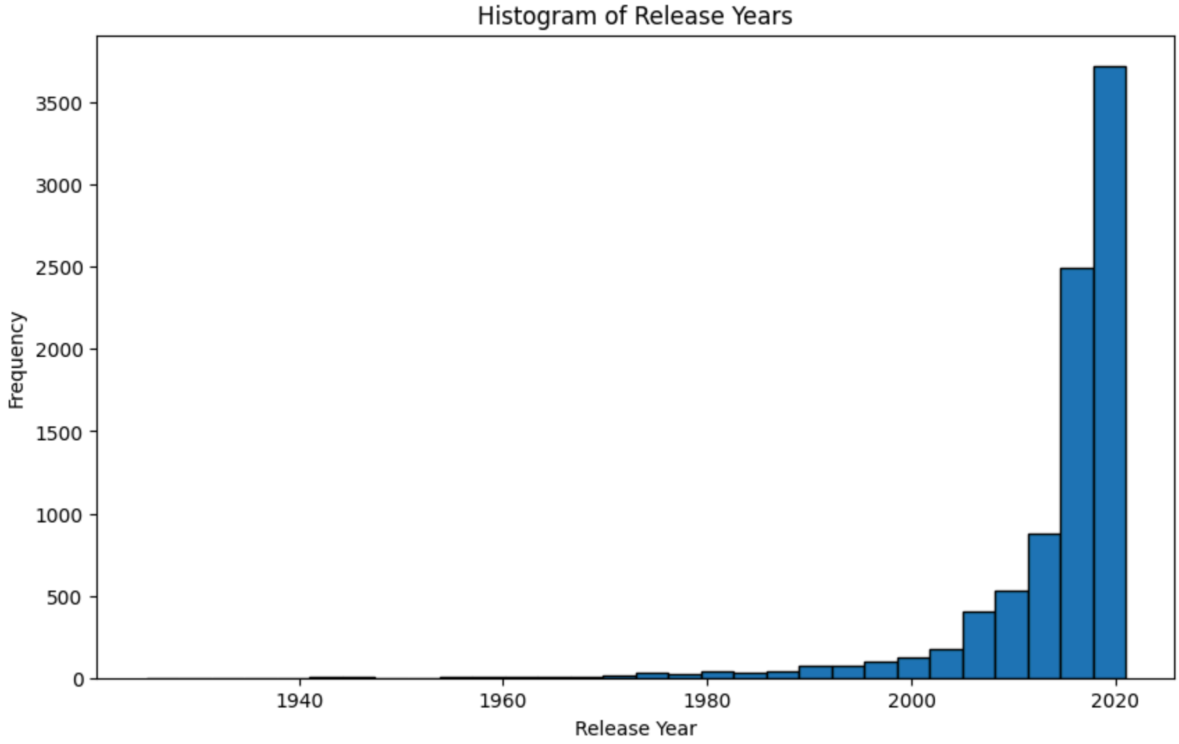
```
plt.figure(figsize=(10, 6))
sns.distplot(df_temp_unique['release_year'], kde=True, bins=30)
plt.title('Distribution of Release Years')
plt.xlabel('Release Year')
plt.ylabel('Density')
plt.show()
```

```
<ipython-input-106-5f73c342a0e4>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df_temp_unique['release_year'], kde=True, bins=30)
```



In [107…

```
plt.figure(figsize=(10, 6))
plt.hist(df_temp_unique['release_year'], bins=30, edgecolor='black')
plt.title('Histogram of Release Years')
plt.xlabel('Release Year')
plt.ylabel('Frequency')
plt.show()
```



Observation :The distribution of release years is right-skewed, indicating that most of the content on Netflix is relatively new, with a significant amount released in the last decade.

In [108…

```
df_temp2 = df_final[["title","type","rating"]]
df_temp2_unique= df_temp2.drop_duplicates(keep='first')
```

In [109…

```
df_temp2_unique
```

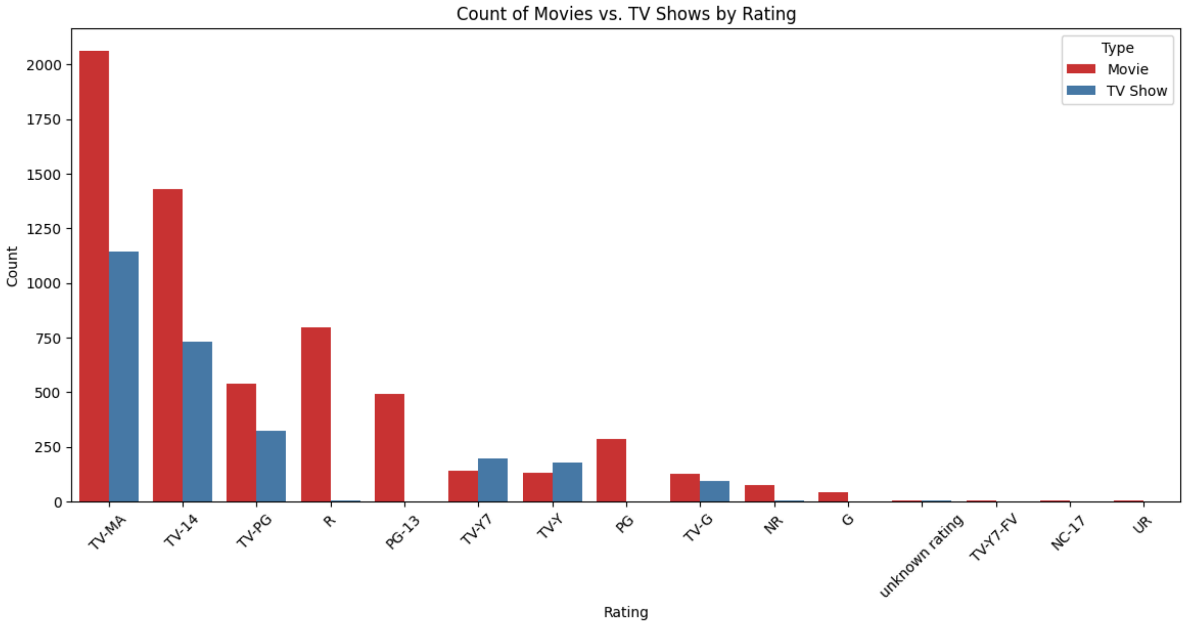
Out[109]:

	title	type	rating
0	Dick Johnson Is Dead	Movie	PG-13
1	Blood & Water	TV Show	TV-MA
58	Ganglands	TV Show	TV-MA
85	Jailbirds New Orleans	TV Show	TV-MA
87	Kota Factory	TV Show	TV-MA
...	...	...	...
201976	Zodiac	Movie	R
202006	Zombie Dumb	TV Show	TV-Y7
202009	Zombieland	Movie	R
202023	Zoom	Movie	PG
202041	Zubaan	Movie	TV-14

8807 rows × 3 columns

In [110…

```
plt.figure(figsize=(14, 6))
sns.countplot(x='rating', hue='type', data=df_temp2_unique,
              order=df_temp2_unique['rating'].value_counts().index,
              palette='Set1')
plt.title('Count of Movies vs. TV Shows by Rating')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.legend(title='Type')
plt.show()
```



Observations:

- Both Movies and TV Shows predominantly fall under the "TV-MA" and "TV-14" ratings.

- The distribution of ratings between Movies and TV Shows is somewhat similar, though Movies have a higher count in most rating categories.

In [111…

```
df_temp3 = df_final[["title","date_added","listed_in"]]
df_temp3
# df_temp3_unique= df_temp2.drop_duplicates(keep='first')
```

Out[111]:

	title	date_added	listed_in
0	Dick Johnson Is Dead	September 25, 2021	Documentaries
1	Blood & Water	September 24, 2021	International TV Shows
2	Blood & Water	September 24, 2021	International TV Shows
3	Blood & Water	September 24, 2021	International TV Shows
4	Blood & Water	September 24, 2021	International TV Shows
...	...	...	...
202060	Zubaan	March 2, 2019	Music & Musicals
202061	Zubaan	March 2, 2019	Music & Musicals
202062	Zubaan	March 2, 2019	Music & Musicals
202063	Zubaan	March 2, 2019	Music & Musicals
202064	Zubaan	March 2, 2019	Music & Musicals

202065 rows × 3 columns

In [112…

```
df_temp3["date_added"] = pd.to_datetime(df_temp3["date_added"],format="mixed")
df_temp3["year_added"]=df_temp3["date_added"].dt.year
df_temp3.drop_duplicates(keep="first",inplace=True)
```

```
<ipython-input-112-94815534e2c7>: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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_temp3["date_added"] = pd.to_datetime(df_temp3["date_added"],format="mixed")
<ipython-input-112-94815534e2c7>: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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_temp3["year_added"]=df_temp3["date_added"].dt.year
<ipython-input-112-94815534e2c7>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_temp3.drop_duplicates(keep="first",inplace=True)
```

In [113…

```
df_temp3
```

Out[113]:

	title	date_added	listed_in	year_added
0	Dick Johnson Is Dead	2021-09-25	Documentaries	2021
1	Blood & Water	2021-09-24	International TV Shows	2021
20	Blood & Water	2021-09-24	TV Dramas	2021
39	Blood & Water	2021-09-24	TV Mysteries	2021
58	Ganglands	2021-09-24	Crime TV Shows	2021
...	...	...	...	...
202023	Zoom	2020-01-11	Children & Family Movies	2020
202032	Zoom	2020-01-11	Comedies	2020
202041	Zubaan	2019-03-02	Dramas	2019
202049	Zubaan	2019-03-02	International Movies	2019
202057	Zubaan	2019-03-02	Music & Musicals	2019

19323 rows × 4 columns

In [114…

```
df_temp3= df_temp3.groupby("year_added")["listed_in"].value_counts().sort_values(
    ascending=False).reset_index()
```

In [115…

```
df_heat = df_temp3[["year_added","count"]]
df_heat
```

Out[115]:

	year_added	count
0	2018	642
1	2019	583
2	2020	538
3	2021	392
4	2017	375
...	...	...
487	2015	1
488	2015	1
489	2015	1
490	2016	1
491	2021	1

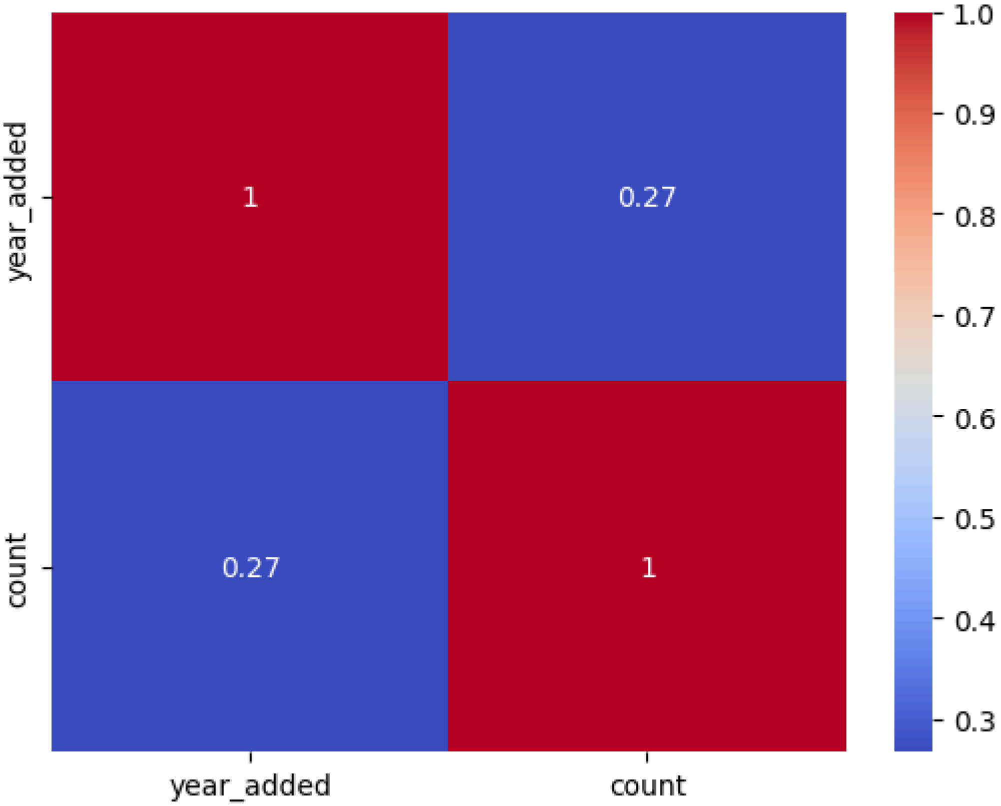
492 rows × 2 columns

In [116…

```
sns.heatmap(df_heat.corr(), annot=True, cmap='coolwarm')
```

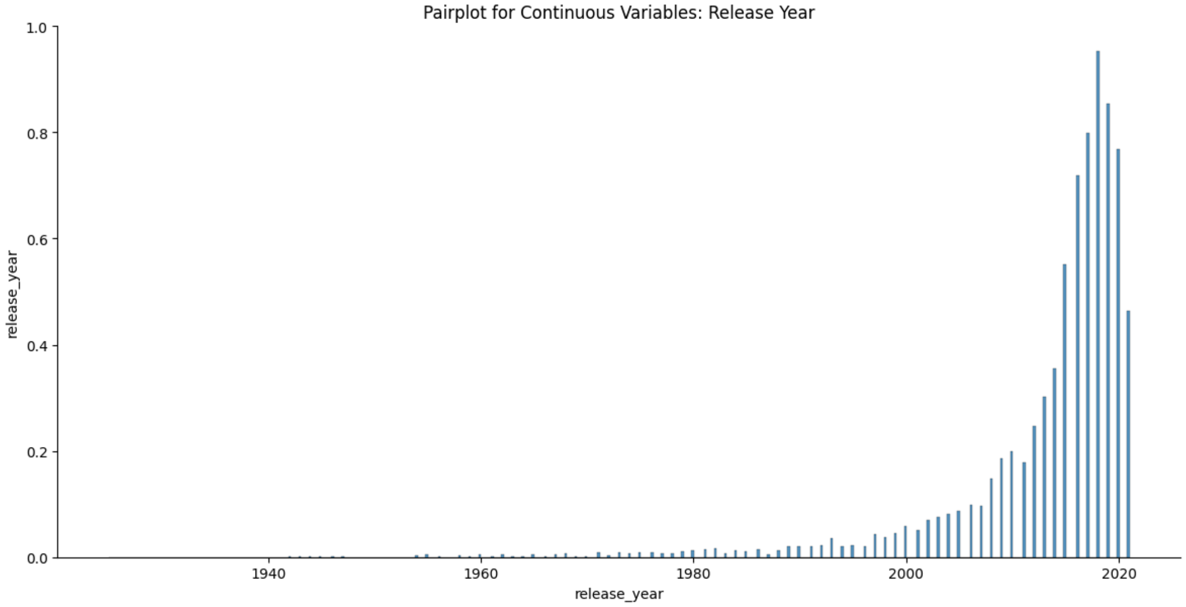
Out[116]:

```
<Axes: >
```



In [117…

```
sns.pairplot(df_final[['release_year']], kind='scatter', height=6, aspect=2)
plt.title('Pairplot for Continuous Variables: Release Year')
plt.show()
```



How does the Type of content vary with release year

In [118…

```
df_temp4 = df_final[["title","type","release_year"]]
df_temp4_unique= df_temp4.drop_duplicates(keep='first',inplace=True)
```

```
<ipython-input-118-f3e4eac1c3a7>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_temp4_unique= df_temp4.drop_duplicates(keep='first',inplace=True)
```

In [119…

```
df_temp4
```

Out[119]:

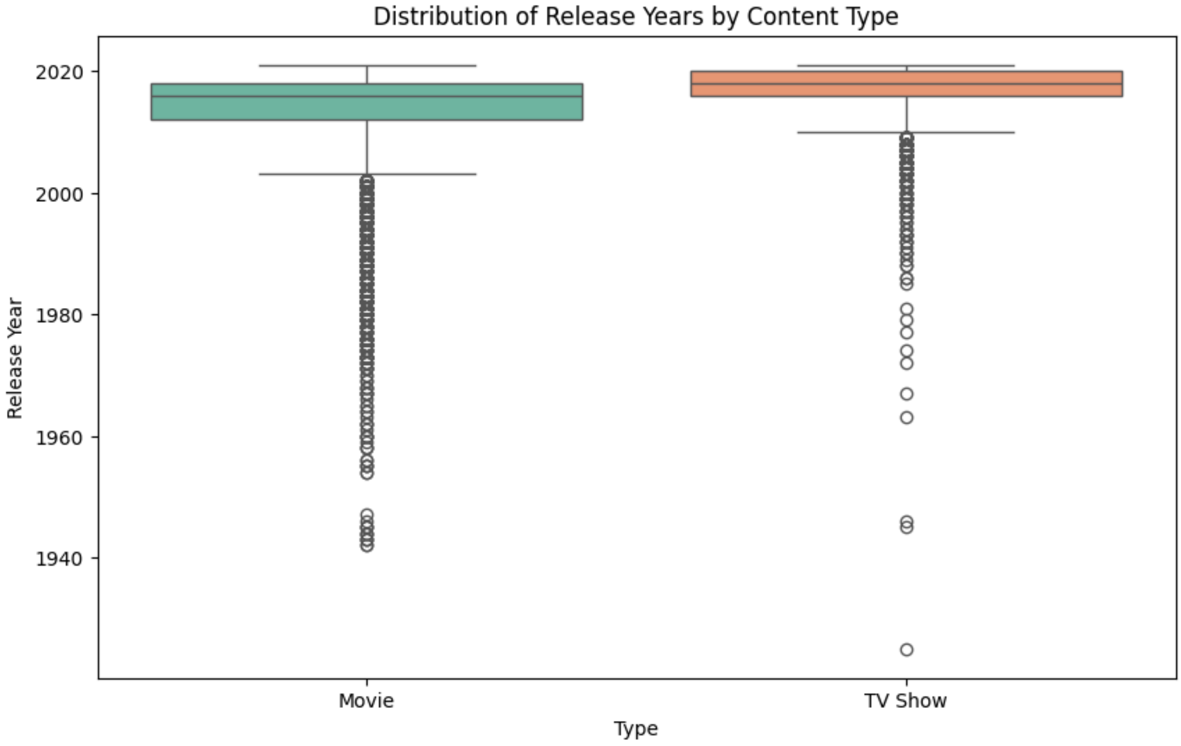
	title	type	release_year
0	Dick Johnson Is Dead	Movie	2020
1	Blood & Water	TV Show	2021
58	Ganglands	TV Show	2021
85	Jailbirds New Orleans	TV Show	2021
87	Kota Factory	TV Show	2021
...	...	...	...
201976	Zodiac	Movie	2007
202006	Zombie Dumb	TV Show	2018
202009	Zombieland	Movie	2009
202023	Zoom	Movie	2006
202041	Zubaan	Movie	2015

8807 rows × 3 columns

In [120…

```
plt.figure(figsize=(10, 6))
sns.boxplot(x='type', y='release_year', data=df_temp4, palette='Set2')
plt.title('Distribution of Release Years by Content Type')
plt.xlabel('Type')
plt.ylabel('Release Year')
plt.show()
```

```
<ipython-input-120-641b5535444f>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.
14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x='type', y='release_year', data=df_temp4, palette='Set2')
```



Observation: The median release year for TV Shows is more recent compared to Movies.

How does the content added varies monthwise

In [121…
```
df_final.head()
```

Out[121]:

	title	country	listed_in	director	cast	show_id	type	date_added	release_yea
0	Dick Johnson Is Dead	United States	Documentaries	Kirsten Johnson	unknown actor	s1	Movie	September 25, 2021	202
1	Blood & Water	South Africa	International TV Shows	unknown director	Ama Qamata	s2	TV Show	September 24, 2021	202
2	Blood & Water	South Africa	International TV Shows	unknown director	Khosi Ngema	s2	TV Show	September 24, 2021	202
3	Blood & Water	South Africa	International TV Shows	unknown director	Gail Mabalane	s2	TV Show	September 24, 2021	202
4	Blood & Water	South Africa	International TV Shows	unknown director	Thabang Molaba	s2	TV Show	September 24, 2021	202

In [122…
```
df_temp5=df_final[["title","date_added"]]
df_temp5["date_added"] = pd.to_datetime(df_temp5["date_added"],format="mixed")
df_temp5["month_added"]=df_temp5["date_added"].dt.month

df_temp5.drop_duplicates(keep="first",inplace=True)
```

```
<ipython-input-122-8a556989e8a9>: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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_temp5["date_added"] = pd.to_datetime(df_temp5["date_added"],format="mixed")
<ipython-input-122-8a556989e8a9>:3: 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/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_temp5["month_added"]=df_temp5["date_added"].dt.month
<ipython-input-122-8a556989e8a9>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_temp5.drop_duplicates(keep="first",inplace=True)
```

In [123…

```
df_temp5
```

Out[123]:

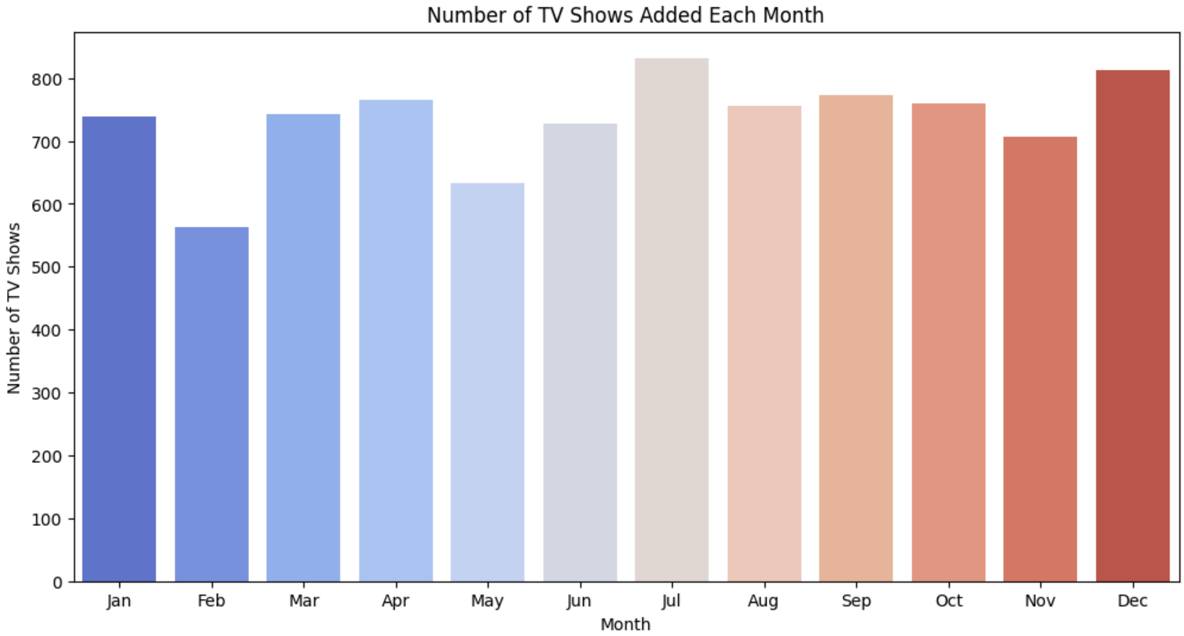
	title	date_added	month_added
0	Dick Johnson Is Dead	2021-09-25	9
1	Blood & Water	2021-09-24	9
58	Ganglands	2021-09-24	9
85	Jailbirds New Orleans	2021-09-24	9
87	Kota Factory	2021-09-24	9
...	...	...	...
201976	Zodiac	2019-11-20	11
202006	Zombie Dumb	2019-07-01	7
202009	Zombieland	2019-11-01	11
202023	Zoom	2020-01-11	1
202041	Zubaan	2019-03-02	3

8807 rows × 3 columns

In [124…

```
monthwise_add = df_temp5["month_added"].value_counts().sort_index()
# monthwise_add
plt.figure(figsize=(12, 6))
sns.barplot(x=monthwise_add.index, y=monthwise_add.values, palette='coolwarm')
plt.title('Number of TV Shows Added Each Month')
plt.xlabel('Month')
plt.ylabel('Number of TV Shows')
plt.xticks(ticks=range(0, 12), labels=['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'J
plt.show()
```

```
<ipython-input-124-fb0ad1e120f2>:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.
14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=monthwise_add.index, y=monthwise_add.values, palette='coolwarm')
```



Data-Backed Business Insights

1. Content is diverse

The Netflix dataset is diversified having the movies and TV shows produced in 748 unique countries and covers a wide array of genres. The top three countries contributing to the content are the United States (3749 titles), India (1048 titles), and the United Kingdom (650 titles). Business Interpretation: This broad geographical and genre-based diversity suggests that Netflix is well-positioned to cater to a global audience with varied tastes. This is a strong asset for market penetration and customer retention.

1. Focus on Recent Content Insight: A large portion of Netflix's content has been released in recent years. For instance, the years after 2016 makes up nearly 50% of the total content on the platform.

Business Interpretation: This focus on newer content is in alighment with current viewer preferences for fresh and relevant material. It also indicates that Netflix is actively keeping its content up-to-date, which is essential for maintaining subscriber interest and attracting new customers.

1. Ratings and Target Demographic Quantifiable Insight: The ratings 'TV-MA' and 'TV-14' dominate the content on Netflix.

Business Interpretation: The predominance of these ratings suggests that Netflix's primary target audience is mature and teen audiences. Content strategies targeting these demographics are likely to be more successful.

Data-Backed Recommendations

1. Expand Older TV Show Portfolio :-

Quantifiable Insight: The median release year for TV Shows is more recent compared to Movies.

Recommendation: Given this focus on newer TV Shows, Netflix should consider adding more classic TV Shows to its platform to attract a wider age group

1. Regional Customization:-

Quantifiable Insight: Content from the United States, India, and the United Kingdom makes up nearly 50% of the total content. Recommendation: With content available from 198 different countries, Netflix has the opportunity to further expand its offerings based on regional popularity. This could lead to an increase in local subscriptions and customer satisfaction.

1. Explore other Genres and Ratings:-

Quantifiable Insight: Ratings 'TV-MA' and 'TV-14' account for nearly 60% of all content. Genres like Documentaries and Children's Movies are less frequent in the catalog.

Recommendation: Netflix could diversify its ratings by exploring underrepresented genres and ratings to attract a more diverse audience.

1. Seasonal Releases:-

Quantifiable Insight: There is a sudden spike in the amount of content added during December and January, suggesting these are peak months for new releases. Recommendation: Given this seasonal trend, Netflix could focus on releasing highly anticipated new seasons or exclusive content during these months to capitalize on increased viewership.