Netflix Exploratory Data Analysis

🎯 Goal of Project

The goal of this project is to conduct a comprehensive Exploratory Data Analysis (EDA) on the Netflix dataset. By examining viewing patterns, popular genres, and content distribution, we aim to uncover valuable insights that can inform Netflix's content strategy and improve user experience. This analysis will help identify trends, understand user preferences, and highlight areas for potential growth and development within the platform's vast content library.

📁 Source of Data : Kaggle

The dataset used for this project is sourced from Kaggle, a popular platform for data science and machine learning competitions. The Netflix dataset on Kaggle includes information about various TV shows and movies available on Netflix, such as titles, genres, release dates, and ratings. This dataset provides a comprehensive overview of the content available on Netflix and serves as the foundation for our Exploratory Data Analysis (EDA)

🔄 Work Flow

- step 1 : importing all the related libraries
- step 2 : load the data set
- step 3 : Basic Understanding of Data
- step 4 : check Dtypes of columns
- step 5 : Basic information of Data
- step 6 : Data processing
- step 7 : Handling Missing Value
- step 8 : Feature Engneering Column Extraction
- step 9 : Data Quality Management: Addressing Duplicate Entries and Inconsistencies
- step 10 : Analysis and Insights
- step 11 : Conclusion

📚 Importing the required libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime as dt
pd.set_option('display.max_rows',None)
pd.set_option('display.max_columns',None)
```

📥 Loading the Dataset

In [2]:
```
df=pd.read_csv("C:\\Users\\Rahul\\Desktop\\Download_desktop\\archive\\netf
```

🔍 Basic Understanding of Data

Preview Of Data 👀

Previewing the data allows us to understand the structure of the dataset, including the number of columns, their names, and the type of data stored in each column. This understanding helps you plan your analysis effectively and choose appropriate techniques and tools.

In [3]:
```
df.sample(10)
```

Out[3]:

	show_id	type	title	director	cast	country	date_added	release
7636	s7637	Movie	Occupation	Luke Sparke	Daniel Ewing, Temuera Morrison, Stephanie Jaco...	Australia	January 3, 2019	
2779	s2780	Movie	Crip Camp: A Disability Revolution	NaN	NaN	United States	March 25, 2020	
5032	s5033	TV Show	The Frankenstein Chronicles	NaN	Sean Bean, Tom Ward, Richie Campbell, Vanessa ...	United Kingdom	February 20, 2018	
3488	s3489	TV Show	The Inmate	NaN	Ignacio Serricchio, Ana Claudia Talancón, Flav...	Mexico	September 26, 2019	
3801	s3802	TV Show	WHAT / IF	NaN	Renée Zellweger, Jane Levy, Blake Jenner, Keit...	United States	May 24, 2019	
945	s946	Movie	Sitting in Limbo	Stella Corradi	Patrick Robinson, Nadine Marshall, Pippa Benne...	United Kingdom	May 1, 2021	
4749	s4750	Movie	Coco y Raulito: Carrusel de ternura	Raúl Campos, Jan Suter	Coco Celis, Raúl Meneses	Mexico	July 27, 2018	
1574	s1575	TV Show	The Surgeon's Cut	NaN	NaN	United States	December 9, 2020	
4305	s4306	TV Show	Inside the Real Narcos	NaN	Jason Fox	NaN	December 14, 2018	
3296	s3297	Movie	Let It Snow	Luke Snellin	Isabela Moner, Shameik Moore, Kiernan Shipka, ...	United States	November 8, 2019	

How Big is the Data 📏

In [4]:
```
df.shape
```

Out[4]:
```
(8807, 12)
```

🔍 Inference :

- There are total **12 Attributes/column** available in the dataset.
- There are total **8807 Record/rows** available in the dataset.

Fecthing the columns name 📝

In [5]:
```
for i in df.columns:
    print(i)
```

```
show_id
type
title
director
cast
country
date_added
release_year
rating
duration
listed_in
description
```

Column name :

show_id , type , title , director , cast , country , date_added , release_year , rating , duration , listed_in and description

Columns Description :

1. **show_id** 🔢: Unique identifier for each show or movie.
2. **type** 📺: Indicates whether the entry is a TV show or a movie.
3. **title** 🎬: The title of the show or movie.
4. **director** 👨‍💼: The director(s) of the show or movie.
5. **cast** 👥: The cast or actors featured in the show or movie.
6. **country** 🌍: The country or countries where the show or movie was produced or filmed.
7. **date_added** 🗓️: The date when the show or movie was added to the Netflix catalog.
8. **release_year** 🗓️: The year when the show or movie was originally released.
9. **rating** ⭐: The content rating assigned to the show or movie.
10. **duration** 🕛: The duration or length of the show or movie.

11. **listed_in** 📋: The categories or genres the show or movie is classified under.
12. **description** 📝: A brief description or summary of the plot or premise of the show or movie.

📝 Checking Dtypes of columns

In [6]:
```
df1=df.dtypes.to_frame()
df1
```

Out[6]:

	0
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

In [7]:
```
df1.rename(columns={0:"Dtypes"}).T
```

Out[7]:

	show_id	type	title	director	cast	country	date_added	release_year
Dtypes	object	object	object	object	object	object	object	int64

🔍 Inference :

- From the above output, we can see that all columns have the appropriate data types except for 'date_added,' which is currently an object. We need to convert it to datetime for better analysis.

📝 Basic information of Data

In [8]:
```
df.info()
```

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

🔍 Inference :

- There are a few columns with null or missing values that we also need to handle.

Here's an outline of preprocessing steps which we are going to follow

- Remove duplicates: Ensure there are no duplicate entries in the dataset.
- Handle missing values: Address missing data points.
- Imputation: Fill missing values with mean, median, mode, or other appropriate values.
- Dropping: Remove rows or columns with excessive missing values.
- Correct errors: Fix any inaccuracies or inconsistencies in the data.
- Convert data types: Ensure all columns have appropriate data types (e.g., dates as datetime objects).
- Create new features: Based on existing data to enhance the model

Checking the missing value of each columns

In [9]:
```
df.isnull().sum().to_frame().rename(columns={0:"Number of missing v
```

Out[9]:

	show_id	type	title	director	cast	country	date_added	release_year
Number of missing value	0	0	0	2634	825	831	10	0

🔍 Inference :

- The director column has the highest number of missing values: **2638**.
- The cast column has **825** missing values.
- The Country columns has **831** missing values.
- The date_added column has **10** missing values.
- The rating column has **4** missing values.
- The duration column has **3** missing values.

Checking Missing Value Precentage of each Columns

In [10]:
```
print("-------------------------------------------------------------
print("-------------------------------------------------------------
print("|||Columns Name|||---------->>>|||Missing Value Percentage|
print("-------------------------------------------------------------
print("-------------------------------------------------------------
for i in df.columns:
    if df[i].isnull().sum()>0:
        print(i,"------------------------>>>",df[i].isnull().sum()
```

```
-------------------------------------------------------------
-------------------------------------------------------------
|||Columns Name|||---------->>>|||Missing Value Percentage|||
-------------------------------------------------------------
-------------------------------------------------------------
director ------------------------>>> 29.908027705234474
cast ------------------------>>> 9.367548540933349
country ------------------------>>> 9.43567616668559
date_added ------------------------>>> 0.11354604292040422
rating ------------------------>>> 0.04541841716816169
duration ------------------------>>> 0.034063812876121265
```

🔍 Inference :

- All columns with null values have a percentage below 30%, so they will not significantly affect the data insights. However, we will handle

each column individually, using domain knowledge to appropriately fill in the missing values.

Taking director columns

In [11]:
```
df["director"].nunique()
```

Out[11]:
```
4528
```

There are 4528 unique values, so we will replace NaN values with 'Not Available'

In [12]:
```
df["director"]=df["director"].fillna("Not Available")
```

Verify the replacement

In [13]:
```
df["director"].isnull().sum()
```

Out[13]:
```
0
```

Taking Cast Columns

In [14]:
```
df["cast"].unique()
```

Out[14]:
```
array([nan,
       'Ama Qamata, Khosi Ngema, Gail Mabalane, Thabang Molaba, Dil
lon Windvogel, Natasha Thahane, Arno Greeff, Xolile Tshabalala, Get
more Sithole, Cindy Mahlangu, Ryle De Morny, Greteli Fincham, Sello
Maake Ka-Ncube, Odwa Gwanya, Mekaila Mathys, Sandi Schultz, Duane W
illiams, Shamilla Miller, Patrick Mofokeng',
       'Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabiha Akkari, So
fia Lesaffre, Salim Kechiouche, Noureddine Farihi, Geert Van Rampel
berg, Bakary Diombera',
       ...,
       'Jesse Eisenberg, Woody Harrelson, Emma Stone, Abigail Bresl
in, Amber Heard, Bill Murray, Derek Graf',
       'Tim Allen, Courteney Cox, Chevy Chase, Kate Mara, Ryan Newm
an, Michael Cassidy, Spencer Breslin, Rip Torn, Kevin Zegers',
       'Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanana, Manish Cha
udhary, Meghna Malik, Malkeet Rauni, Anita Shabdish, Chittaranjan T
ripathy'],
      dtype=object)
```

In [15]:
```
df["cast"].nunique()
```

Out[15]:
```
7692
```

In [16]:
```
df["cast"].mode()
```

Out[16]:
```
0    David Attenborough
Name: cast, dtype: object
```

In [17]:
```
df["cast"].value_counts().to_frame().head(70)
```

Out[17]:

cast	count
David Attenborough	19
Vatsal Dubey, Julie Tejwani, Rupa Bhimani, Jigna Bhardwaj, Rajesh Kava, Mousam, Swapnil	14
Samuel West	10
Jeff Dunham	7
David Spade, London Hughes, Fortune Feimster	6
Kevin Hart	6
Craig Sechler	6
Michela Luci, Jamie Watson, Eric Peterson, Anna Claire Bartlam, Nicolas Aqui, Cory Doran, Julie Lemieux, Derek McGrath	6
Bill Burr	5
Iliza Shlesinger	5
Jim Gaffigan	5
Jay O. Sanders	4
Aishwarya Rajesh, Vidhu, Surya Ganapathy, Madhuri, Pavel Navageethan, Avantika Vandanapu	4
Mike Birbiglia	4
Bill Hicks	4
Sam Kinison	4
Vir Das	4
Dave Chappelle	4
Damandeep Singh Baggan, Smita Malhotra, Baba Sehgal	4
Tom Segura	4
Sonal Kaushal, Rupa Bhimani, Julie Tejwani, Sabina Malik, Jigna Bharadhwaj, Rajesh Kawa	4
Prabhas, Rana Daggubati, Anushka Shetty, Tamannaah Bhatia, Sathyaraj, Nassar, Ramya Krishnan, Sudeep	4
Jim Jefferies	4
Jerry Seinfeld	4
Michela Luci, Jamie Watson, Anna Claire Bartlam, Dante Zee, Eric Peterson	4
Prabhas, Rana Daggubati, Anushka Shetty, Tamannaah Bhatia, Sathyaraj, Ramya Krishnan, Nassar, Subbaraju	3
R. Kelly	3
Derren Brown	3
Jo Koy	3
Peter Coyote	3
Nadia Ramlee, Chio Su-Ping, Jeremy Linn, Marlon Dance-Hooi	3

cast	count
Rachael Stirling	3
Brian Regan	3
Julie Tejwani, SABINA MALIK, Jigna Bharadhwaj, Rupa Bhimani, Lalit Agarwal, Rajesh Shukla, Disha, Rajesh Kawa, Raju	3
Taapsee Pannu, Vinodhini, Parvathi T, Ramya Subramanian, Sanchana Natarajan, Anish Kuruvilla, David Solomon Raja	3
Chris D'Elia	3
Samantha Ruth Prabhu, Lakshmi, Rajendraprasad, Naga Shourya, Rao Ramesh, Teja Sajja, Pragathi, Jagapathi Babu, Aishwarya, Urvashi	3
Katt Williams	3
Colin Quinn	3
Paco Ignacio Taibo II	3
Aziz Ansari	3
Chelsea Handler	3
Sebastian Maniscalco	3
Russell Peters	3
John Mulaney	3
Stephen Fry, Alex Marty	3
Bo Burnham	3
Marc Maron	3
Bob Ross	3
Bert Kreischer	3
Louis C.K.	3
Gabriel Iglesias	3
Morgan Freeman	3
Eric Meyers	3
Tim Allen	2
Jack Whitehall	2
Piers Morgan	2
Saurav Chakraborty	2
Todd Glass	2
Gavin Lewis, Theodore Barnes, Shelby Simmons, Cynthia Kaye McWilliams	2
Tom Papa	2
Chris Rock	2
Bettany Hughes	2
Chloe Marsden, Aaron Marsden, María Esnoz	2

cast	count
Luis Brandoni, China Zorrilla, Antonio Gasalla, Julio De Grazia, Betiana Blum, Monica Villa, Juan Manuel Tenuta, Andrea Tenuta, Cecilia Rossetto, Enrique Pinti	2
Andrew M. Gray, Ciara Hanna, John Mark Loudermilk, Christina Masterson, Azim Rizk, Shailesh Prajapati, Ian Harcourt	2
Nayantara, Vaibhav Reddy, Pasupathy, Harshvardhan Rane, Thagubothu Ramesh, Vinay Varma, Dheer Charan Srivastav, D. Narsingh Rao	2
You, Reina Triendl, Yoshimi Tokui, Azusa Babazono, Ryota Yamasato, Shono Hayama	2
Sommore	2
Marie Kondo	2

There are 7692 unique values, so we will replace NaN values with 'Not Available'

In [18]:
```
df["cast"]=df["cast"].fillna("Not Available")
```

Verify the replacement

In [19]:
```
df["cast"].isnull().sum()
```

Out[19]:
```
0
```

Taking Country Columns

In [ ]:
```

```

In [20]:
```
df["country"].nunique()
```

Out[20]:
```
748
```

In [21]:
```
df["country"].unique()
```

Out[21]:
```
array(['United States', 'South Africa', nan, 'India',
       'United States, Ghana, Burkina Faso, United Kingdom, German
y, Ethiopia',
       'United Kingdom', 'Germany, Czech Republic', 'Mexico', 'Turk
ey',
       'Australia', 'United States, India, France', 'Finland',
       'China, Canada, United States',
       'South Africa, United States, Japan', 'Nigeria', 'Japan',
       'Spain, United States', 'France', 'Belgium',
       'United Kingdom, United States', 'United States, United King
dom',
       'France, United States', 'South Korea', 'Spain',
       'United States, Singapore', 'United Kingdom, Australia, Fran
ce',
       'United Kingdom, Australia, France, United States',
       'United States, Canada', 'Germany, United States',
       'South Africa, United States', 'United States, Mexico',
       'United States, Italy, France, Japan',
       'United States, Italy, Romania, United Kingdom',
       'Australia, United States', 'Argentina, Venezuela',
       'United States, United Kingdom, Canada', 'China, Hong Kong',
       'Russia', 'Canada', 'Hong Kong', 'United States, China, Hong
Kong',
       'Italy, United States', 'United States, Germany',
       'United Kingdom, Canada, United States', ', South Korea',
       'Ireland', 'India, Nepal',
       'New Zealand, Australia, France, United States', 'Italy',
       'Italy, Brazil, Greece', 'Argentina', 'Jordan', 'Colombia',
       'United States, Japan', 'Belgium, United Kingdom',
       'Switzerland, United Kingdom, Australia', 'Israel, United St
ates',
       'Canada, United States', 'Brazil', 'Argentina, Spain', 'Taiw
an',
       'United States, Nigeria', 'Bulgaria, United States',
       'Spain, United Kingdom, United States', 'United States, Chin
a',
       'United States, France',
       'Spain, France, United Kingdom, United States',
       ', France, Algeria', 'Poland', 'Germany',
       'France, Israel, Germany, United States, United Kingdom',
       'New Zealand', 'Saudi Arabia', 'Thailand', 'Indonesia',
       'Egypt, Denmark, Germany', 'United States, Switzerland',
       'Hong Kong, Canada, United States', 'Kuwait, United States',
       'France, Canada, United States, Spain',
       'France, Netherlands, Singapore', 'France, Belgium',
       'Ireland, United States, United Kingdom', 'Egypt', 'Malaysi
a',
       'Israel', 'Australia, New Zealand', 'United Kingdom, German
y',
       'Belgium, Netherlands', 'South Korea, Czech Republic',
       'Australia, Germany', 'Vietnam', 'United Kingdom, Belgium',
       'United Kingdom, Australia, United States',
       'France, Japan, United States',
       'United Kingdom, Germany, Spain, United States',
       'United Kingdom, United States, France, Italy',
       'United States, Germany, Canada',
       'United States, France, Italy, United Kingdom',
       'United States, United Kingdom, Germany, Hungary',
       'United States, New Zealand', 'Sweden', 'China', 'Lebanon',
       'Romania', 'Finland, Germany', 'Lebanon, Syria', 'Philippine
s',
       'Iceland', 'Denmark', 'United States, India',
       'Philippines, Singapore, Indonesia',
       'China, United States, Canada', 'Lebanon, United Arab Emirat
```

```
       'China, Hong Kong, United States', 'United Kingdom, New Zeal
and',
       'Czech Republic, United Kingdom, France',
       'Australia, United Kingdom, Canada', 'Jamaica, United State
s',
       'Australia, United Kingdom, United States, New Zealand, Ital
y, France',
       'France, United States, Canada',
       'United Kingdom, France, Canada, Belgium, United States',
       'Denmark, United Kingdom, Sweden', 'United States, Hong Kon
g',
       'United States, Kazakhstan',
       'Argentina, France, United States, Germany, Qatar',
       'United States, Germany, United Kingdom',
       'United States, Germany, United Kingdom, Italy',
       'United States, New Zealand, United Kingdom',
       'Finland, United States', 'Spain, France, Uruguay',
       'France, Canada, United States', 'United States, Canada, Chi
na',
       'Ireland, Canada, Luxembourg, United States, United Kingdom,
Philippines, India',
       'United States, Czech Republic, United Kingdom', 'Israel, Ge
rmany',
       'Mexico, France',
       'Israel, Germany, Poland, Luxembourg, Belgium, France, Unite
d States',
       'Austria, United States', 'United Kingdom, Lithuania',
       'United States, Greece, United Kingdom',
       'United Kingdom, China, United States, India',
       'United States, Sweden, Norway',
       'United Kingdom, United States, Morocco',
       'United States, United Kingdom, Morocco',
       'Spain, Canada, United States',
       'United States, India, United Arab Emirates',
       'United Kingdom, Canada, France, United States',
       'India, Germany, France',
       'Belgium, Ireland, Netherlands, Germany, Afghanistan',
       'France, Canada, Italy, United States, China',
       'Ireland, United Kingdom, Greece, France, Netherlands',
       'Denmark, Indonesia, Finland, Norway, United Kingdom, Israe
l, France, United States, Germany, Netherlands',
       'New Zealand, United States',
       'United States, Australia, South Africa, United Kingdom',
       'United States, Germany, Mexico',
       'Somalia, Kenya, Sudan, South Africa, United States',
       'United States, Canada, Japan, Panama',
       'United Kingdom, Spain, Belgium', 'Serbia, South Korea, Slov
enia',
       'Denmark, United Kingdom, South Africa, Sweden, Belgium',
       'Germany, Canada, United States',
       'Ireland, Canada, United States, United Kingdom',
       'New Zealand, United Kingdom, Australia',
       'United Kingdom, Australia, Canada, United States',
       'Germany, United States, Italy', 'United States, Venezuela',
       'United Kingdom, Canada, Japan',
       'United Kingdom, United States, Czech Republic',
       'United Kingdom, China, United States',
       'United Kingdom, Brazil, Germany',
       'United Kingdom, Namibia, South Africa, Zimbabwe, United Sta
tes',
       'Canada, United States, India, United Kingdom',
       'Switzerland, United Kingdom, United States',
       'United Kingdom, India, Sweden',
       'United States, Brazil, India, Uganda, China',
```

```
        'Peru, United States, United Kingdom',
        'Germany, United States, United Kingdom, Canada',
        'Canada, India, Thailand, United States, United Arab Emirate
s',
        'United States, East Germany, West Germany',
        'France, Netherlands, South Africa, Finland',
        'Egypt, Austria, United States', 'Russia, Spain',
        'Croatia, Slovenia, Serbia, Montenegro', 'Japan, Canada',
        'United States, France, South Korea, Indonesia',
        'United Arab Emirates, Jordan'], dtype=object)
```

In [22]:
```
df["country"].value_counts().to_frame().head(100)
```

Out[22]:

country	count
United States	2818
India	972
United Kingdom	419
Japan	245
South Korea	199
Canada	181
Spain	145
France	124
Mexico	110
Egypt	106
Turkey	105
Nigeria	95
Australia	87
Taiwan	81
Indonesia	79
Brazil	77
Philippines	75
United Kingdom, United States	75
United States, Canada	73
Germany	67
China	66
Thailand	61
Argentina	56
Hong Kong	53
United States, United Kingdom	47
Italy	45
Canada, United States	45
Colombia	35
South Africa	30
France, Belgium	27
Poland	24
Singapore	23
Malaysia	22
Netherlands	19
Pakistan	18

country	count
United States, Germany	17
United States, Japan	16
United States, France	16
Hong Kong, China	16
Lebanon	15
Russia	15
Chile	14
United Arab Emirates	14
China, Hong Kong	14
United States, Mexico	14
Denmark	13
Australia, United States	13
Mexico, United States	13
Sweden	13
Germany, United States	13
Israel	13
New Zealand	12
Japan, United States	12
United States, Australia	11
Norway	11
France, United States	10
India, United States	10
Ireland	10
United States, India	9
Belgium	9
Argentina, Spain	8
United Kingdom, Canada, United States	8
United States, China	8
Saudi Arabia	8
Vietnam	7
United States, France, Japan	7
United Kingdom, France	6
Romania	6
Kuwait	6
South Korea, United States	6

country	count
United States, Czech Republic	5
Italy, France	5
United States, New Zealand	5
Spain, France	5
Austria	5
United States, Italy	5
India, France	5
Israel, United States	5
Iceland	5
South Africa, United States	4
United Kingdom, France, United States	4
United States, Russia	4
Italy, United States	4
United Kingdom, United States, Spain, Germany, Greece, Canada	4
United States, United Kingdom, France	4
Peru	4
United States, Spain	4
Poland, United States	4
United States, Bulgaria	4
Canada, United States, United Kingdom	4
Hungary	4
Mexico, Spain	4
Hong Kong, United States	4
United States, United Kingdom, Australia	4
United Kingdom, Canada	4
United Kingdom, Germany	4
Spain, Germany	3
United States, Sweden	3
Sweden, United States	3
United Kingdom, Japan, United States	3

In [23]:
```
df["country"].mode()
```

Out[23]:
```
0    United States
Name: country, dtype: object
```

Since the values in the 'country' column are categorical, we will fill the missing values with the mode. The United States, with the highest

frequency of 2818, is also present in most of the hybrid country entries.

In [24]:
```
df["country"]=df["country"].fillna("United States")
```

Verify the replacement

In [25]:
```
df["country"].isnull().sum()
```

Out[25]:
```
0
```

Taking Rating column

In [26]:
```
df["rating"].unique()
```

Out[26]:
```
array(['PG-13', 'TV-MA', 'PG', 'TV-14', 'TV-PG', 'TV-Y', 'TV-Y7',
'R',
       'TV-G', 'G', 'NC-17', '74 min', '84 min', '66 min', 'NR', na
n,
       'TV-Y7-FV', 'UR'], dtype=object)
```

Here's the meaning of each rating:

- 'G': Suitable for all ages.
- 'TV-Y': Intended for children of all ages.
- 'TV-Y7': May not be suitable for children under 7.
- 'TV-G': General audience.
- 'TV-PG': Parental guidance suggested.
- 'PG': Parental guidance suggested - Some material may not be suitable for children.
- 'TV-14': May be unsuitable for children under 14.
- 'PG-13': Parents strongly cautioned - Some material may be inappropriate for children under 13.
- 'TV-MA': Mature audience - Content is specifically designed to be viewed by adults and may be unsuitable for children under 17.
- 'R': Restricted - Restricted to viewers over a certain age (typically 17 or 18).
- 'NC-17': No one 17 and under admitted.
- 'NR': Not rated.
- 'TV-Y7-FV': Suitable for children ages 7 and up, with fantasy violence.
- 'UR': Unrated.

In [27]:
```
df["rating"].mode()
```

Out[27]:
```
0    TV-MA
Name: rating, dtype: object
```

In [28]:
```
df["rating"].value_counts()
```

Out[28]:
```
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
```

Since 'UR' represents 'Unrated', we will fill the NaN values in the 'rating' column with 'UR'

In [29]:
```
df["rating"]=df["rating"].fillna("UR")
```

Verify the replacement

In [30]:
```
df["rating"].isnull().sum()
```

Out[30]:
```
0
```

Taking duration column

In [31]:
```
df["duration"].nunique()
```

Out[31]:
```
220
```

In [32]:
```
df["duration"].unique()
```

Out[32]:
```
array(['90 min', '2 Seasons', '1 Season', '91 min', '125 min',
       '9 Seasons', '104 min', '127 min', '4 Seasons', '67 min', '9
4 min',
       '5 Seasons', '161 min', '61 min', '166 min', '147 min', '103
min',
       '97 min', '106 min', '111 min', '3 Seasons', '110 min', '105
min',
       '96 min', '124 min', '116 min', '98 min', '23 min', '115 mi
n',
       '122 min', '99 min', '88 min', '100 min', '6 Seasons', '102
min',
       '93 min', '95 min', '85 min', '83 min', '113 min', '13 min',
       '182 min', '48 min', '145 min', '87 min', '92 min', '80 mi
n',
       '117 min', '128 min', '119 min', '143 min', '114 min', '118
min',
       '108 min', '63 min', '121 min', '142 min', '154 min', '120 m
in',
       '82 min', '109 min', '101 min', '86 min', '229 min', '76 mi
n',
       '89 min', '156 min', '112 min', '107 min', '129 min', '135 m
in',
       '136 min', '165 min', '150 min', '133 min', '70 min', '84 mi
n',
       '140 min', '78 min', '7 Seasons', '64 min', '59 min', '139 m
in',
       '69 min', '148 min', '189 min', '141 min', '130 min', '138 m
in',
       '81 min', '132 min', '10 Seasons', '123 min', '65 min', '68
min',
       '66 min', '62 min', '74 min', '131 min', '39 min', '46 min',
       '38 min', '8 Seasons', '17 Seasons', '126 min', '155 min',
       '159 min', '137 min', '12 min', '273 min', '36 min', '34 mi
n',
       '77 min', '60 min', '49 min', '58 min', '72 min', '204 min',
       '212 min', '25 min', '73 min', '29 min', '47 min', '32 min',
       '35 min', '71 min', '149 min', '33 min', '15 min', '54 min',
       '224 min', '162 min', '37 min', '75 min', '79 min', '55 mi
n',
       '158 min', '164 min', '173 min', '181 min', '185 min', '21 m
in',
       '24 min', '51 min', '151 min', '42 min', '22 min', '134 mi
n',
       '177 min', '13 Seasons', '52 min', '14 min', '53 min', '8 mi
n',
       '57 min', '28 min', '50 min', '9 min', '26 min', '45 min',
       '171 min', '27 min', '44 min', '146 min', '20 min', '157 mi
n',
       '17 min', '203 min', '41 min', '30 min', '194 min', '15 Seas
ons',
       '233 min', '237 min', '230 min', '195 min', '253 min', '152
min',
       '190 min', '160 min', '208 min', '180 min', '144 min', '5 mi
n',
       '174 min', '170 min', '192 min', '209 min', '187 min', '172
min',
       '16 min', '186 min', '11 min', '193 min', '176 min', '56 mi
n',
       '169 min', '40 min', '10 min', '3 min', '168 min', '312 mi
n',
       '153 min', '214 min', '31 min', '163 min', '19 min', '12 Sea
sons',
       nan, '179 min', '11 Seasons', '43 min', '200 min', '196 mi
n',
```

```
       '167 min', '178 min', '228 min', '18 min', '205 min', '201 m
in',
       '191 min'], dtype=object)
```

In [33]:
```
df["duration"].value_counts().head(20)
```

Out[33]:
```
duration
1 Season     1793
2 Seasons     425
3 Seasons     199
90 min        152
94 min        146
97 min        146
93 min        146
91 min        144
95 min        137
96 min        130
92 min        129
102 min       122
98 min        120
99 min        118
101 min       116
88 min        116
103 min       114
106 min       111
100 min       108
89 min        106
Name: count, dtype: int64
```

In [34]:
```
df[df["duration"].isnull()]
```

Out[34]:

	show_id	type	title	director	cast	country	date_added	release_y
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2(
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	September 16, 2016	2(
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	August 15, 2016	2(

Since there are 3 null values in the 'duration' column, we will replace them with the corresponding values from the 'rating' column. Additionally, we will change the 'rating' column values to 'UR'.

In [35]:
```
df["duration"]=df["duration"].fillna(df["rating"])
```

In [36]:
```
df.iloc[5541,8]="UR"
```

In [37]:
```
df.iloc[5794,8]="UR"
```

In [38]:
```
df.iloc[5813,8]="UR"
```

Verify the replacement

In [39]:
```
df.iloc[[5813,5541,5794],]
```

Out[39]:

	show_id	type	title	director	cast	country	date_added	release_y
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	August 15, 2016	2(
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	April 4, 2017	2(
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	September 16, 2016	2(

Taking date_added Column

In [40]:
```
df["date_added"].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 8807 entries, 0 to 8806
Series name: date_added
Non-Null Count  Dtype
--------------  -----
8797 non-null   object
dtypes: object(1)
memory usage: 68.9+ KB
```

Since the data type of the 'date_added' column is currently object, we will convert it to datetime.

In [41]:
```
df["date_added"]=pd.to_datetime(df["date_added"], errors="coerce")
```

Verify the changes

In [42]:
```
df["date_added"].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 8807 entries, 0 to 8806
Series name: date_added
Non-Null Count  Dtype
--------------  -----
8709 non-null   datetime64[ns]
dtypes: datetime64[ns](1)
memory usage: 68.9 KB
```

Since we will extract month_added, day_name_added, day_added, and year_added columns from the date_added column

In [43]:
```
df["month_added"]=df.date_added.dt.month_name()
df["day_name_added"]=df.date_added.dt.day_name()
df["day_added"]=df.date_added.dt.day
df["year_added"]=df.date_added.dt.year
```

Verifying the results

In [44]:
```
df.sample(2)
```

Out[44]:

	show_id	type	title	director	cast	country	date_added	rel
5693	s5694	Movie	For the Love of Spock	Adam Nimoy	Leonard Nimoy, William Shatner, George Takei, ...	Canada, United States	2016-12-02	
5359	s5360	Movie	Raising the Bar	Clay Glen	Kelli Berglund, Lili Karamalikis, Tess Fowler,...	Australia	2017-08-01	

Checking the duplicate entries

In [45]:
```
df.duplicated().sum()
```

Out[45]:
```
0
```

🔍 Inference :

- Their is no duplicates values in the data

Detecting the inconsistency of the data

In [46]:
```
for i in df.columns:
    print(i)
    print( i ,"has",df[i].nunique(),"unique values")
    print("----------------------------------------Unique Value-
    print("-------------------------------------------------------
    print(df[i].unique())
    #for j in df[i]:
        #print(j)
    print("=======================================================
    print("=======================================================
```

```
show_id
show_id has 8807 unique values
----------------------------------------Unique Value----------
--------------------------------------
------------------------------------------------------------
--------------------------------------
['s1' 's2' 's3' ... 's8805' 's8806' 's8807']
============================================================
======================================
============================================================
======================================
type
type has 2 unique values
----------------------------------------Unique Value----------
--------------------------------------
------------------------------------------------------------
--------------------------------------
['Movie' 'TV Show']
============================================================
======================================
============================================================
======================================
title
title has 8807 unique values
----------------------------------------Unique Value----------
--------------------------------------
------------------------------------------------------------
--------------------------------------
['Dick Johnson Is Dead' 'Blood & Water' 'Ganglands' ... 'Zombiel
and'
 'Zoom' 'Zubaan']
============================================================
======================================
============================================================
======================================
director
director has 4529 unique values
----------------------------------------Unique Value----------
--------------------------------------
------------------------------------------------------------
--------------------------------------
['Kirsten Johnson' 'Not Available' 'Julien Leclercq' ... 'Majid
Al Ansari'
 'Peter Hewitt' 'Mozez Singh']
============================================================
======================================
============================================================
======================================
cast
cast has 7693 unique values
----------------------------------------Unique Value----------
--------------------------------------
------------------------------------------------------------
--------------------------------------
['Not Available'
 'Ama Qamata, Khosi Ngema, Gail Mabalane, Thabang Molaba, Dillon
Windvogel, Natasha Thahane, Arno Greeff, Xolile Tshabalala, Getm
ore Sithole, Cindy Mahlangu, Ryle De Morny, Greteli Fincham, Sel
lo Maake Ka-Ncube, Odwa Gwanya, Mekaila Mathys, Sandi Schultz, D
uane Williams, Shamilla Miller, Patrick Mofokeng'
 'Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabiha Akkari, Sofia
Lesaffre, Salim Kechiouche, Noureddine Farihi, Geert Van Rampelb
erg, Bakary Diombera'
 ...
```

```
 'Jesse Eisenberg, Woody Harrelson, Emma Stone, Abigail Breslin,
Amber Heard, Bill Murray, Derek Graf'
 'Tim Allen, Courteney Cox, Chevy Chase, Kate Mara, Ryan Newman,
Michael Cassidy, Spencer Breslin, Rip Torn, Kevin Zegers'
 'Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanana, Manish Chaudh
ary, Meghna Malik, Malkeet Rauni, Anita Shabdish, Chittaranjan T
ripathy']
================================================================
====================================
================================================================
====================================
country
country has 748 unique values
----------------------------------------Unique Value-----------
------------------------------------
--------------------------------------------------------------
------------------------------------
['United States' 'South Africa' 'India'
 'United States, Ghana, Burkina Faso, United Kingdom, Germany, E
thiopia'
 'United Kingdom' 'Germany, Czech Republic' 'Mexico' 'Turkey' 'A
ustralia'
 'United States, India, France' 'Finland' 'China, Canada, United
States'
 'South Africa, United States, Japan' 'Nigeria' 'Japan'
 'Spain, United States' 'France' 'Belgium' 'United Kingdom, Unit
ed States'
 'United States, United Kingdom' 'France, United States' 'South
Korea'
 'Spain' 'United States, Singapore' 'United Kingdom, Australia,
France'
 'United Kingdom, Australia, France, United States'
 'United States, Canada' 'Germany, United States'
 'South Africa, United States' 'United States, Mexico'
 'United States, Italy, France, Japan'
 'United States, Italy, Romania, United Kingdom'
 'Australia, United States' 'Argentina, Venezuela'
 'United States, United Kingdom, Canada' 'China, Hong Kong' 'Rus
sia'
 'Canada' 'Hong Kong' 'United States, China, Hong Kong'
 'Italy, United States' 'United States, Germany'
 'United Kingdom, Canada, United States' ', South Korea' 'Irelan
d'
 'India, Nepal' 'New Zealand, Australia, France, United States'
'Italy'
 'Italy, Brazil, Greece' 'Argentina' 'Jordan' 'Colombia'
 'United States, Japan' 'Belgium, United Kingdom'
 'Switzerland, United Kingdom, Australia' 'Israel, United State
s'
 'Canada, United States' 'Brazil' 'Argentina, Spain' 'Taiwan'
 'United States, Nigeria' 'Bulgaria, United States'
 'Spain, United Kingdom, United States' 'United States, China'
 'United States, France' 'Spain, France, United Kingdom, United
States'
 ', France, Algeria' 'Poland' 'Germany'
 'France, Israel, Germany, United States, United Kingdom' 'New Z
ealand'
 'Saudi Arabia' 'Thailand' 'Indonesia' 'Egypt, Denmark, Germany'
 'United States, Switzerland' 'Hong Kong, Canada, United States'
 'Kuwait, United States' 'France, Canada, United States, Spain'
 'France, Netherlands, Singapore' 'France, Belgium'
 'Ireland, United States, United Kingdom' 'Egypt' 'Malaysia' 'Is
rael'
 'Australia, New Zealand' 'United Kingdom, Germany' 'Belgium, Ne
```

```
therlands'
 'South Korea, Czech Republic' 'Australia, Germany' 'Vietnam'
 'United Kingdom, Belgium' 'United Kingdom, Australia, United St
ates'
 'France, Japan, United States'
 'United Kingdom, Germany, Spain, United States'
 'United Kingdom, United States, France, Italy'
 'United States, Germany, Canada'
 'United States, France, Italy, United Kingdom'
 'United States, United Kingdom, Germany, Hungary'
 'United States, New Zealand' 'Sweden' 'China' 'Lebanon' 'Romani
a'
 'Finland, Germany' 'Lebanon, Syria' 'Philippines' 'Iceland' 'De
nmark'
 'United States, India' 'Philippines, Singapore, Indonesia'
 'China, United States, Canada' 'Lebanon, United Arab Emirates'
 'Canada, United States, Denmark' 'United Arab Emirates'
 'Mexico, France, Colombia' 'Netherlands' 'Germany, United State
s, France'
 'United States, Bulgaria'
 'United Kingdom, France, Germany, United States' 'Norway, Denma
rk'
 'Syria, France, Lebanon, Qatar' 'United States, Czech Republic'
 'Mauritius' 'Canada, South Africa' 'Austria' 'Mexico, Brazil'
 'Germany, France' 'Mexico, United States'
 'United Kingdom, France, Spain, United States' 'United States,
Australia'
 'United States, United Kingdom, France' 'United States, Russia'
 'United States, United Kingdom, New Zealand' 'Australia, United
Kingdom'
 'Canada, Nigeria, United States'
 'France, United States, United Kingdom, Canada' 'France, United
Kingdom'
 'India, United Kingdom' 'Canada, United States, Mexico'
 'United Kingdom, Germany, United States'
 'Czech Republic, United Kingdom, United States' 'China, United
Kingdom'
 'Italy, United Kingdom' 'China, Taiwan'
 'United States, Brazil, Japan, Spain, India'
 'United States, China, United Kingdom' 'Cameroon'
 'Lebanon, Palestine, Denmark, Qatar' 'Japan, United States'
 'Uruguay, Germany' 'Egypt, Saudi Arabia'
 'United Kingdom, France, Poland, Germany, United States'
 'Ireland, Switzerland, United Kingdom, France, United States'
 'United Kingdom, South Africa, France'
 'Ireland, United Kingdom, France, Germany' 'Russia, United Stat
es'
 'United Kingdom, United States, France' 'United Kingdom,'
 'United States, India, United Kingdom' 'Kenya' 'Spain, Argentin
a'
 'India, United Kingdom, France, Qatar' 'Belgium, France'
 'Argentina, Chile' 'United States, Thailand' 'Chile, Brazil'
 'United States, Colombia' 'Canada, United States, United Kingdo
m'
 'Uruguay' 'Luxembourg' 'United States, Cambodia, Romania' 'Bang
ladesh'
 'Spain, Belgium, United States'
 'United Kingdom, United States, Australia'
 'Canada, United States, France' 'Portugal, United States'
 'Portugal, Spain' 'India, United States' 'United Kingdom, Irela
nd'
 'United Kingdom, Spain, United States' 'Hungary, United States'
 'United States, South Korea' 'Canada, United States, Cayman Isl
ands'
```

```
 'International TV Shows, Spanish-Language TV Shows, TV Horror'
 'Crime TV Shows, TV Action & Adventure, TV Thrillers'
 'Music & Musicals, Stand-Up Comedy' 'British TV Shows, TV Comed
ies'
 'TV Comedies, TV Sci-Fi & Fantasy, Teen TV Shows'
 'TV Comedies, TV Sci-Fi & Fantasy'
 'Romantic TV Shows, Spanish-Language TV Shows, TV Comedies'
 'Crime TV Shows, International TV Shows, TV Sci-Fi & Fantasy'
 'British TV Shows, International TV Shows, Romantic TV Shows'
 "Crime TV Shows, Kids' TV"
 'Horror Movies, International Movies, Sci-Fi & Fantasy'
 'TV Comedies, TV Mysteries'
 'Cult Movies, Horror Movies, Independent Movies'
 'British TV Shows, Docuseries, TV Comedies' 'Comedies, Document
aries'
 'Reality TV, Science & Nature TV, TV Action & Adventure'
 'TV Comedies, TV Dramas, TV Mysteries'
 'Crime TV Shows, TV Comedies, Teen TV Shows'
 "Docuseries, Kids' TV, Science & Nature TV"
 'Reality TV, Spanish-Language TV Shows'
 'Action & Adventure, Anime Features, Sci-Fi & Fantasy'
 "Crime TV Shows, Kids' TV, TV Comedies"
 'Dramas, Faith & Spirituality, Independent Movies'
 'Documentaries, Faith & Spirituality'
 'British TV Shows, International TV Shows, Stand-Up Comedy & Ta
lk Shows'
 'Comedies, Dramas, Faith & Spirituality' 'Classic & Cult TV, TV
Comedies'
 'Dramas, Romantic Movies, Sports Movies'
 'Stand-Up Comedy & Talk Shows, TV Mysteries, TV Sci-Fi & Fantas
y'
 'TV Sci-Fi & Fantasy, TV Thrillers'
 'Comedies, Independent Movies, Music & Musicals'
 'Comedies, Cult Movies, Independent Movies'
 'Documentaries, Dramas, International Movies'
 'British TV Shows, TV Horror, TV Thrillers'
 'British TV Shows, Docuseries, Science & Nature TV'
 'Children & Family Movies, Comedies, Cult Movies' 'Sports Movie
s'
 'Sci-Fi & Fantasy' 'Comedies, LGBTQ Movies'
 'Comedies, Independent Movies, Thrillers'
 'Classic Movies, Cult Movies, Dramas'
 'British TV Shows, TV Comedies, TV Dramas'
 'Action & Adventure, Children & Family Movies, Independent Movi
es'
 'Action & Adventure, Documentaries, International Movies'
 'Children & Family Movies, Independent Movies'
 'Comedies, Cult Movies, Dramas'
 'International TV Shows, TV Horror, TV Thrillers'
 'Classic Movies, Thrillers' 'Crime TV Shows, TV Dramas, TV Horr
or'
 'British TV Shows, Docuseries, Reality TV'
 'Documentaries, LGBTQ Movies, Music & Musicals'
 'Classic Movies, Dramas, Romantic Movies'
 'Crime TV Shows, Romantic TV Shows, Spanish-Language TV Shows'
 'Classic Movies, Cult Movies, Horror Movies'
 'Anime Series, Crime TV Shows, TV Thrillers'
 'Children & Family Movies, Classic Movies'
 'Classic Movies, Comedies, International Movies'
 'Comedies, Sci-Fi & Fantasy' 'Action & Adventure, Cult Movies,
Dramas'
 'Documentaries, Faith & Spirituality, Music & Musicals'
 'British TV Shows, Classic & Cult TV, TV Comedies'
 'International Movies, Sports Movies' 'International TV Shows'
```

```
 "Classic & Cult TV, Kids' TV, Spanish-Language TV Shows"
 'Romantic TV Shows, Spanish-Language TV Shows, TV Dramas'
 'Children & Family Movies, Comedies, Faith & Spirituality'
 'British TV Shows, Crime TV Shows, TV Dramas'
 'Classic Movies, Dramas, Music & Musicals'
 'Cult Movies, Horror Movies, Thrillers'
 'Action & Adventure, Classic Movies, Sci-Fi & Fantasy'
 'TV Action & Adventure, TV Comedies'
 'Classic Movies, Comedies, Music & Musicals' 'Independent Movie
s'
 'Documentaries, Horror Movies'
 'Classic & Cult TV, TV Horror, TV Mysteries'
 'Comedies, Faith & Spirituality, International Movies'
 'Dramas, Horror Movies, Sci-Fi & Fantasy'
 'British TV Shows, TV Dramas, TV Sci-Fi & Fantasy'
 'Comedies, Cult Movies, Horror Movies'
 'Comedies, Cult Movies, Sports Movies' 'Classic Movies, Documen
taries'
 'Action & Adventure, Faith & Spirituality, Sci-Fi & Fantasy'
 'Action & Adventure, Children & Family Movies'
 'International TV Shows, Reality TV, TV Action & Adventure'
 'Docuseries, Science & Nature TV, TV Dramas' 'Anime Features'
 'Action & Adventure, Horror Movies, Independent Movies'
 'Action & Adventure, Classic Movies, International Movies'
 'Cult Movies, Independent Movies, Thrillers'
 'Crime TV Shows, TV Comedies'
 'Classic Movies, Cult Movies, Documentaries'
 "Classic & Cult TV, Kids' TV, TV Comedies"
 'Classic Movies, Dramas, LGBTQ Movies'
 'Classic Movies, Dramas, Sports Movies' 'Action & Adventure, Cu
lt Movies'
 'Action & Adventure, Comedies, Music & Musicals'
 'Classic Movies, Horror Movies, Thrillers'
 'Classic Movies, Comedies, Independent Movies'
 'Children & Family Movies, Classic Movies, Dramas'
 'Dramas, Faith & Spirituality, Sports Movies'
 'Classic Movies, Comedies, Romantic Movies'
 'Dramas, Horror Movies, Music & Musicals'
 'Classic Movies, Independent Movies, Thrillers'
 'Children & Family Movies, Faith & Spirituality'
 'Classic Movies, Comedies, Sports Movies'
 'Comedies, Dramas, Sports Movies'
 'Action & Adventure, Romantic Movies, Sci-Fi & Fantasy'
 'Classic & Cult TV, TV Sci-Fi & Fantasy'
 'Comedies, Cult Movies, LGBTQ Movies'
 'Comedies, Horror Movies, Sci-Fi & Fantasy'
 'Action & Adventure, Comedies, Horror Movies'
 'Classic & Cult TV, Crime TV Shows, TV Dramas'
 'Action & Adventure, Documentaries, Sports Movies'
 'International Movies, LGBTQ Movies, Romantic Movies'
 'Cult Movies, Dramas, Thrillers']
==================================================================
======================================
==================================================================
======================================
description
description has 8775 unique values
----------------------------------------Unique Value----------
--------------------------------------
------------------------------------------------------------------
--------------------------------------
['As her father nears the end of his life, filmmaker Kirsten Joh
nson stages his death in inventive and comical ways to help them
both face the inevitable.'
```

```
 'After crossing paths at a party, a Cape Town teen sets out to
prove whether a private-school swimming star is her sister who w
as abducted at birth.'
 'To protect his family from a powerful drug lord, skilled thief
Mehdi and his expert team of robbers are pulled into a violent a
nd deadly turf war.'
 ...
 'Looking to survive in a world taken over by zombies, a dorky c
ollege student teams with an urban roughneck and a pair of grift
er sisters.'
 'Dragged from civilian life, a former superhero must train a ne
w crop of youthful saviors when the military preps for an attack
by a familiar villain.'
 "A scrappy but poor boy worms his way into a tycoon's dysfuncti
onal family, while facing his fear of music and the truth about
his past."]
================================================================
====================================
================================================================
====================================
month_added
month_added has 12 unique values
-----------------------------------------Unique Value-----------
-----------------------------------
----------------------------------------------------------------
-----------------------------------
['September' 'August' 'July' 'June' 'May' 'April' 'March' 'Febru
ary'
 'January' 'December' 'November' 'October' nan]
================================================================
====================================
================================================================
====================================
day_name_added
day_name_added has 7 unique values
-----------------------------------------Unique Value-----------
-----------------------------------
----------------------------------------------------------------
-----------------------------------
['Saturday' 'Friday' 'Thursday' 'Wednesday' 'Tuesday' 'Monday'
'Sunday'
 nan]
================================================================
====================================
================================================================
====================================
day_added
day_added has 31 unique values
-----------------------------------------Unique Value-----------
-----------------------------------
----------------------------------------------------------------
-----------------------------------
[25. 24. 23. 22. 21. 20. 19. 17. 16. 15. 14. 11. 10.  9.  8.  7.
6.  5.
  4.  3.  2.  1. 31. 29. 28. 27. 26. 18. 13. 12. 30. nan]
================================================================
====================================
================================================================
====================================
year_added
year_added has 14 unique values
-----------------------------------------Unique Value-----------
-----------------------------------
----------------------------------------------------------------
```

```
------------------------------------
[2021. 2020. 2019. 2018. 2017. 2016. 2015. 2014. 2013. 2012. 201
1. 2009.
 2008.   nan 2010.]
====================================================================
====================================
====================================================================
====================================
```

🔍 Inference :

- There are no inconsistencies in the data. However, the show_id column contains an extra 's' at the beginning which we can remove. For example, we will replace 's1' with '1'.

Taking show_id columns

In [47]:
```
df["show_id"]= df["show_id"].apply(lambda x : x[1:])
```

Verifying the results

In [48]:
```
df.head()
```

Out[48]:

	show_id	type	title	director	cast	country	date_added	re
0	1	Movie	Dick Johnson Is Dead	Kirsten Johnson	Not Available	United States	2021-09-25	
1	2	TV Show	Blood & Water	Not Available	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	2021-09-24	
2	3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	United States	2021-09-24	
3	4	TV Show	Jailbirds New Orleans	Not Available	Not Available	United States	2021-09-24	
4	5	TV Show	Kota Factory	Not Available	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	2021-09-24	

🔍 Inference :

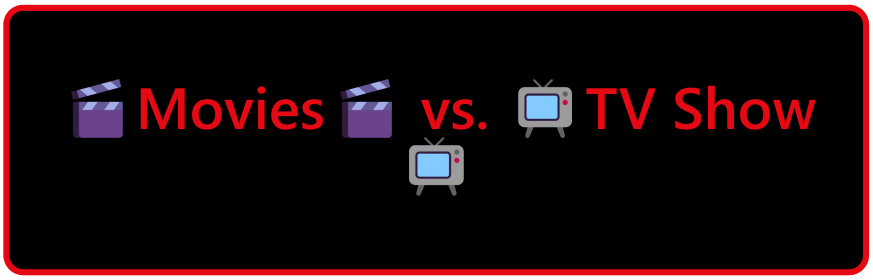
- We have remove all the inconsistency. Now,There are no inconsistencies in the data.

🔍 Analysis and Insights 📊

Key aspects of analyzing Netflix data and deriving insights

- **Content Trends**: Identify popular genres, types of content (movies, TV shows), and their evolution over time.
- **Geographical Insights**: Determine content origin and regional viewing preferences to tailor offerings.

- **Viewership Behavior**: Analyze viewer habits, peak viewing times, and preferences across demographics.
- **Content Details**: Study content duration, number of seasons per show, and upcoming releases.
- **Performance Metrics**: Evaluate viewer engagement, ratings, and impact on viewership for content optimization.



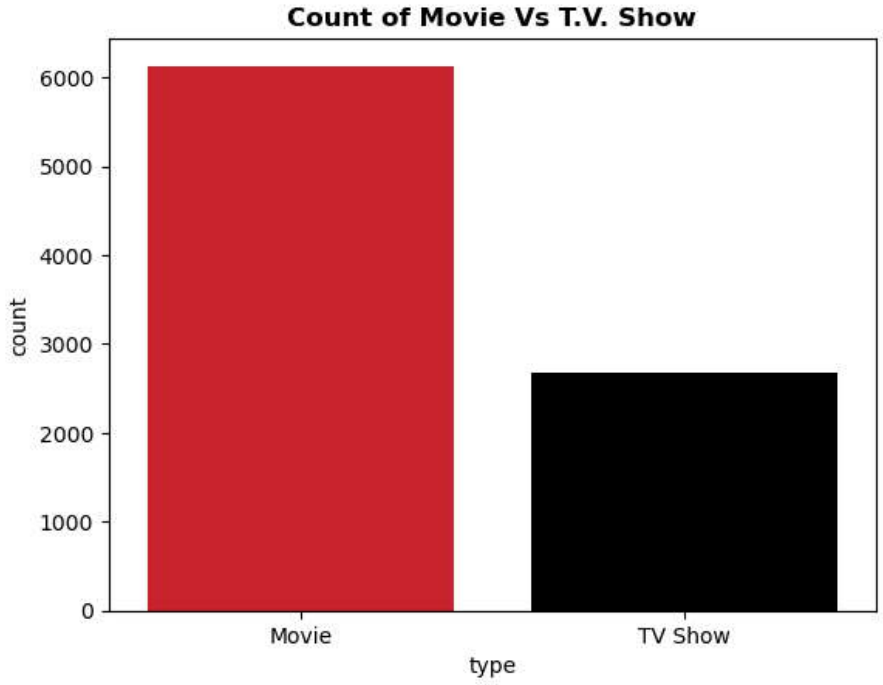
In [49]:
```
df["type"].unique()
```

Out[49]:
```
array(['Movie', 'TV Show'], dtype=object)
```

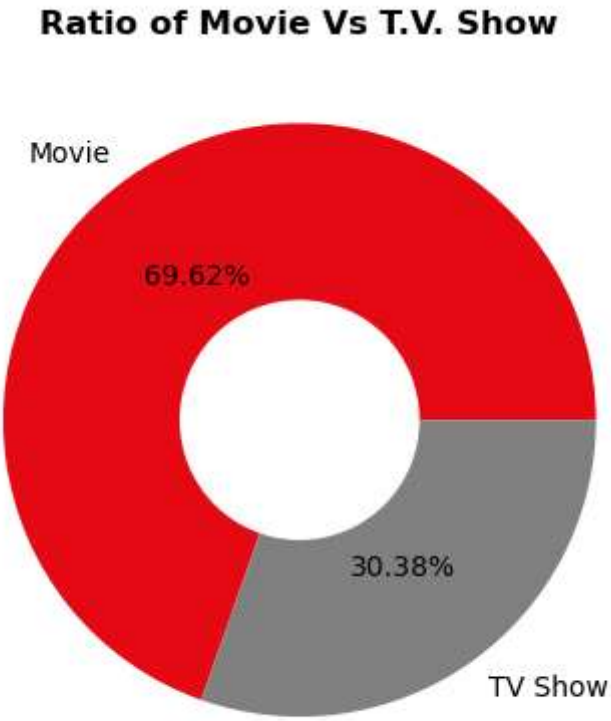
In [50]:
```
df["type"].value_counts()
```

Out[50]:
```
type
Movie      6131
TV Show    2676
Name: count, dtype: int64
```

In [51]:
```
sns.countplot(x="type",data=df,palette=['#E50914','#000000'])
plt.title("Count of Movie Vs T.V. Show",fontweight='bold');
```



In [52]:
```
plt.pie(df["type"].value_counts(),labels=df["type"].value_cour
plt.title("Ratio of Movie Vs T.V. Show",fontweight='bold');
circle = plt.Circle( (0,0), 0.4, color='white')
p=plt.gcf()
p.gca().add_artist(circle);
```

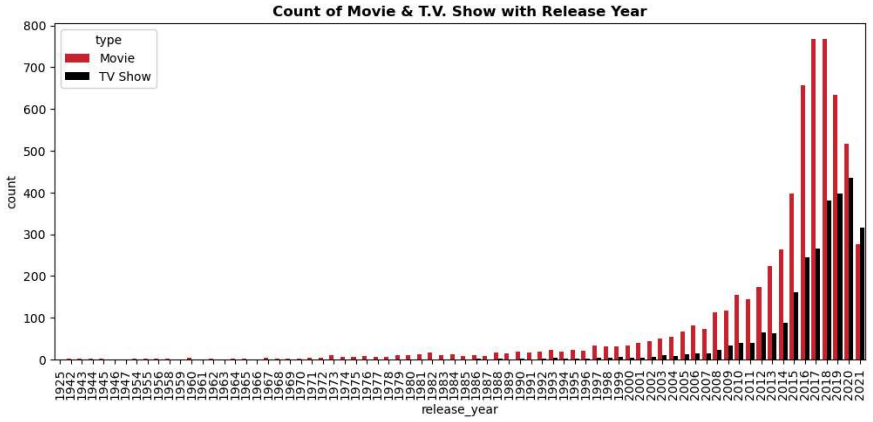


🔍 Inference :

- The Netflix catalog features a significantly higher number of movies 🎬, totaling 6,131, compared to 2,676 TV shows 📺.
- Movies 🎬 make up approximately 70% of the content, while TV shows 📺 account for around 30%.

📽️ Release Year 🕰️

In [53]:
```
plt.figure(figsize=(12,5))
sns.countplot(x="release_year",data=df,palette=['#E50914','#
plt.title("Count of Movie & T.V. Show with Release Year",fon
plt.xticks(rotation=90);
```

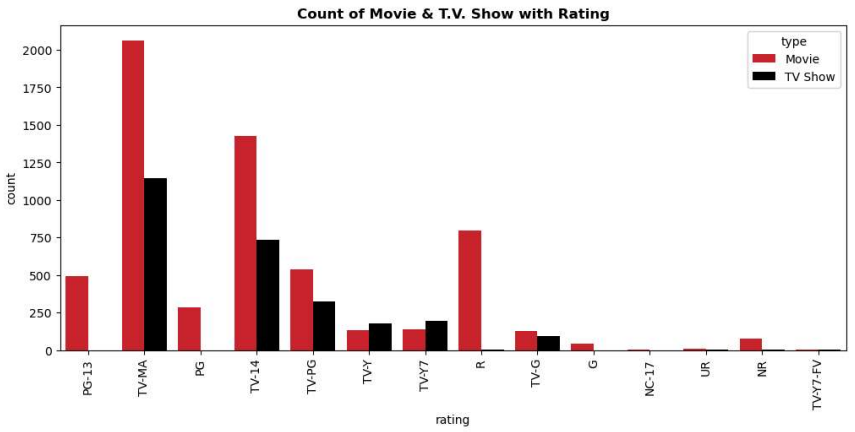


🔍 Inference :

- Most of the content in the dataset has been released within the last seven years.

🎬 Content Ratings🌟

In [54]:
```
plt.figure(figsize=(12,5))
sns.countplot(x="rating",data=df,palette=['#E50914','#00000
plt.title("Count of Movie & T.V. Show with Rating",fontweig
plt.xticks(rotation=90);
```



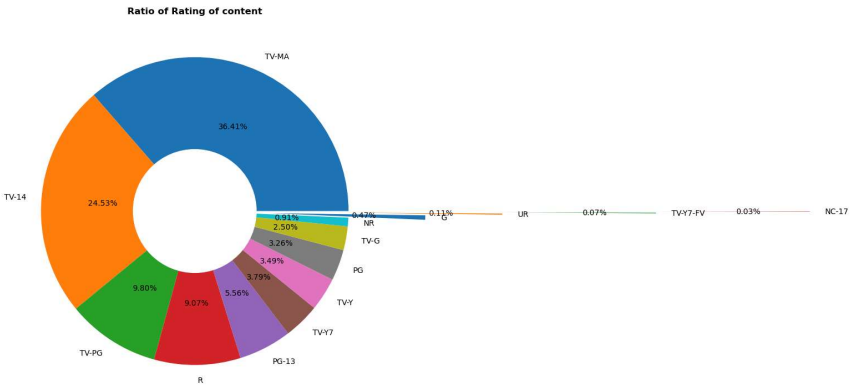
In [55]:
```
df["rating"].value_counts().to_frame()
```

Out[55]:

rating	count
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
UR	10
TV-Y7-FV	6
NC-17	3

In [56]:
```
plt.figure(figsize=(9,9))
plt.pie(df["rating"].value_counts(),labels=df["rating"].val
```

```
plt.title("Ratio of Rating of content",fontweight="bold");
circle = plt.Circle( (0,0), 0.4, color='white')
p=plt.gcf()
p.gca().add_artist(circle);
```

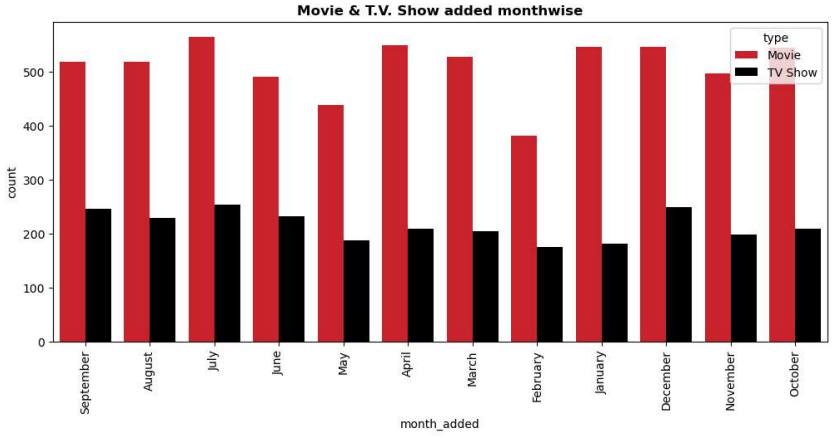


🔍 Inference :

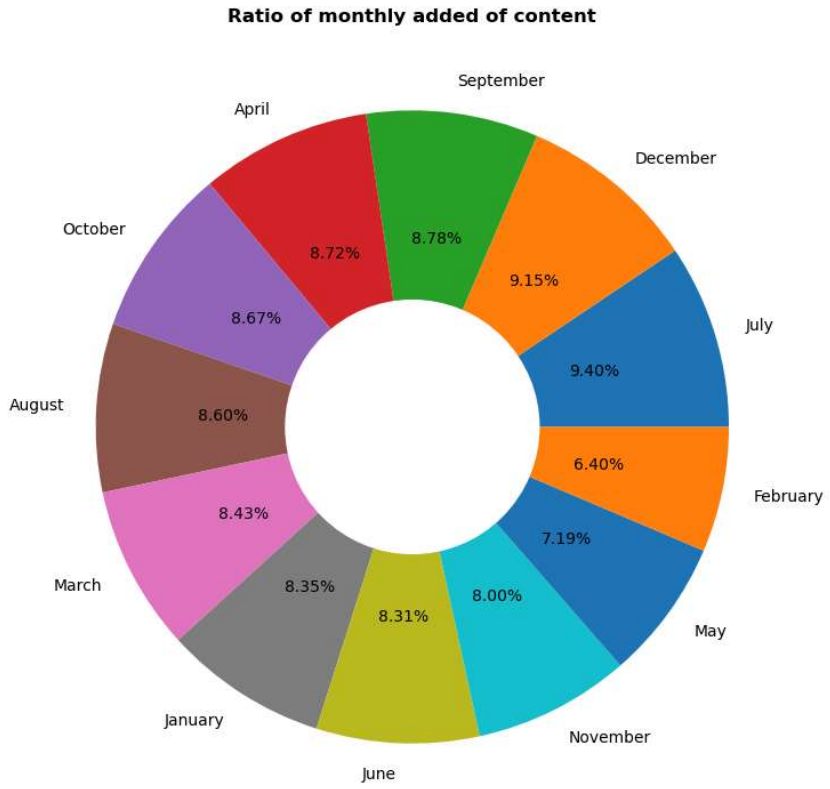
- The dataset shows that the majority of content is rated TV-MA, TV-14, TV-PG, and R.

Monthly Content Additions

In [57]:
```
plt.figure(figsize=(12,5))
sns.countplot(x="month_added",data=df,palette=['#E50914',
plt.title("Movie & T.V. Show added monthwise",fontweight=
plt.xticks(rotation=90);
```



In [58]:
```
plt.figure(figsize=(9,9))
plt.pie(df["month_added"].value_counts(),labels=df["month_
plt.title("Ratio of monthly added of content",fontweight='
circle = plt.Circle( (0,0), 0.4, color='white')
p=plt.gcf()
p.gca().add_artist(circle);
```



In [59]:
```
df["month_added"].value_counts().to_frame()
```

Out[59]:

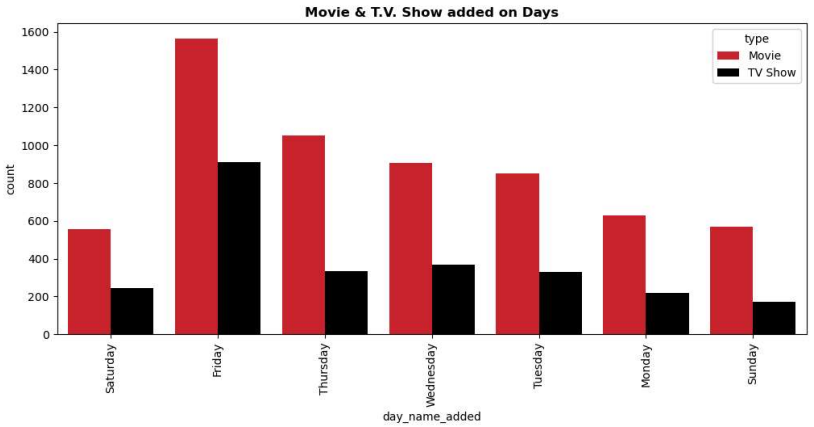
month_added	count
July	819
December	797
September	765
April	759
October	755
August	749
March	734
January	727
June	724
November	697
May	626
February	557

🔍 Inference :

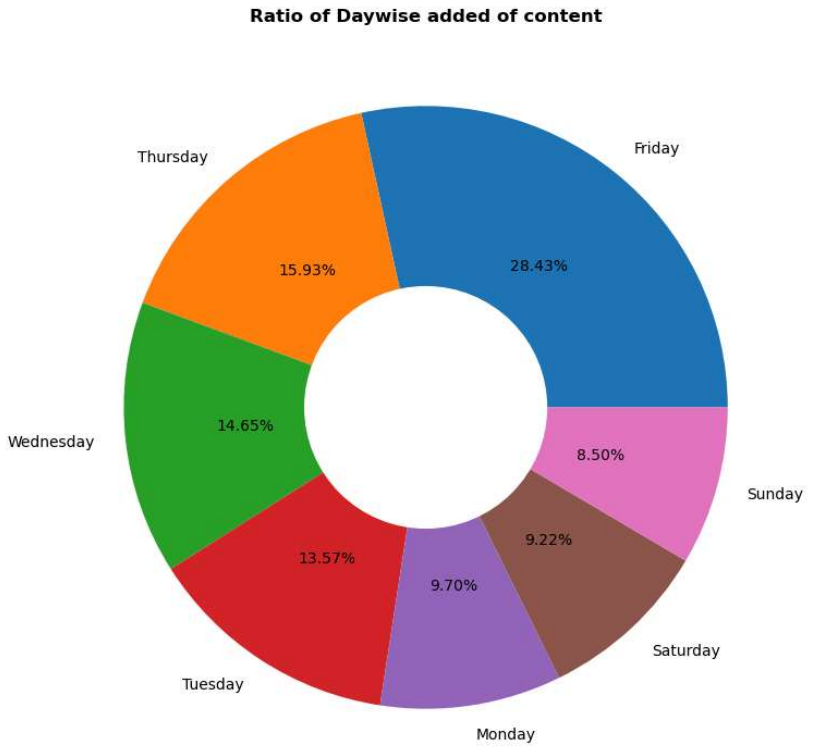
- Most of the content was added in July, with 819 titles, while February saw the least additions with 557 titles. However, the difference is not particularly significant.

Daily Content Additions

In [60]:
```
plt.figure(figsize=(12,5))
sns.countplot(x="day_name_added",data=df,palette=['#E509
plt.title("Movie & T.V. Show added on Days",fontweight='
plt.xticks(rotation=90);
```



In [61]:
```
plt.figure(figsize=(9,9))
plt.pie(df["day_name_added"].value_counts(),labels=df["d
plt.title("Ratio of Daywise added of content",fontweight
circle = plt.Circle( (0,0), 0.4, color='white')
p=plt.gcf()
p.gca().add_artist(circle);
```



In [62]:
```
df["day_name_added"].value_counts().to_frame()
```

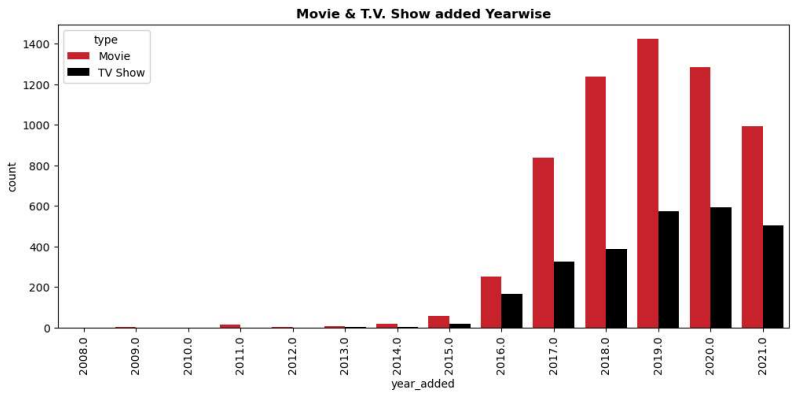
Out[62]:

day_name_added	count
Friday	2476
Thursday	1387
Wednesday	1276
Tuesday	1182
Monday	845
Saturday	803
Sunday	740

🔍 Inference :

- Most of the content is added on Fridays, with 2,476 titles, whereas Sundays have the least additions, with 740 titles.

In [63]:
```
plt.figure(figsize=(12,5))
sns.countplot(x="year_added",data=df,palette=['#E50914'
plt.title("Movie & T.V. Show added Yearwise",fontweight
plt.xticks(rotation=90);
```



In [64]:
```
yr=df["year_added"].value_counts().to_frame()
```

In [65]:
```
yr
```

Out[65]:

year_added	count
2019.0	1999
2020.0	1878
2018.0	1625
2021.0	1498
2017.0	1164
2016.0	418
2015.0	73
2014.0	23
2011.0	13
2013.0	10
2012.0	3
2009.0	2
2008.0	2
2010.0	1

There are many small values before 2016, making it difficult to represent them in a pie chart. To address this, we combined all the values from 2008 to 2015 into one category.

In [66]:
```
yr_d={}
befor2016=0
for i in yr["count"].index:
    if i>2015:
        yr_d[int(i)]=yr.loc[i,"count"]

    else:
        befor2016+=yr.loc[i,"count"]
yr_d["2018-2015"]=befor2016
print(yr_d)
```

```
{2019: 1999, 2020: 1878, 2018: 1625, 2021: 1498, 2017:
1164, 2016: 418, '2018-2015': 127}
```

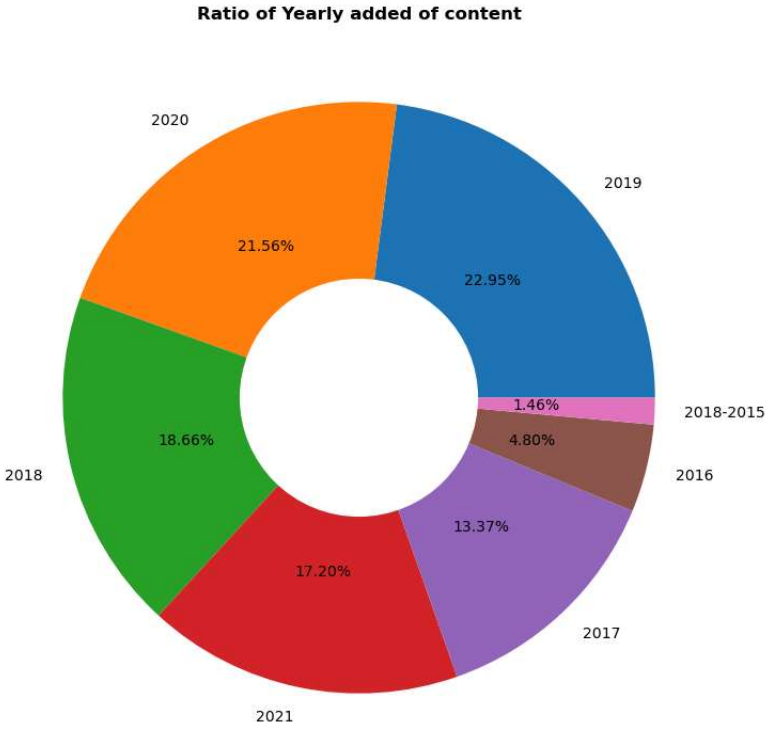
In [67]:
```
yr_d.values()
```

Out[67]:
```
dict_values([1999, 1878, 1625, 1498, 1164, 418, 127])
```

In [68]:
```
yr_d.keys()
```

Out[68]:
```
dict_keys([2019, 2020, 2018, 2021, 2017, 2016, '2018-2
015'])
```

In [69]:
```
plt.figure(figsize=(9,9))
plt.pie(yr_d.values(),labels=yr_d.keys(),autopct="%.2f%
plt.title("Ratio of Yearly added of content",fontweight
circle = plt.Circle( (0,0), 0.4, color='white')
p=plt.gcf()
p.gca().add_artist(circle);
```



🔍 Inference :

- The maximum content was added in the year 2019, which is around 2000 items, while the least content was added in 2010, with only one item.
- We can also see that most of the content, around 90%, was added in the last few years.

🌎 Regional Content Analysis

There are numerous countries and hybrid entities, making it challenging to represent in a count plot and pie chart. Therefore, we trimmed the data to focus on the top 15 countries for analysis.

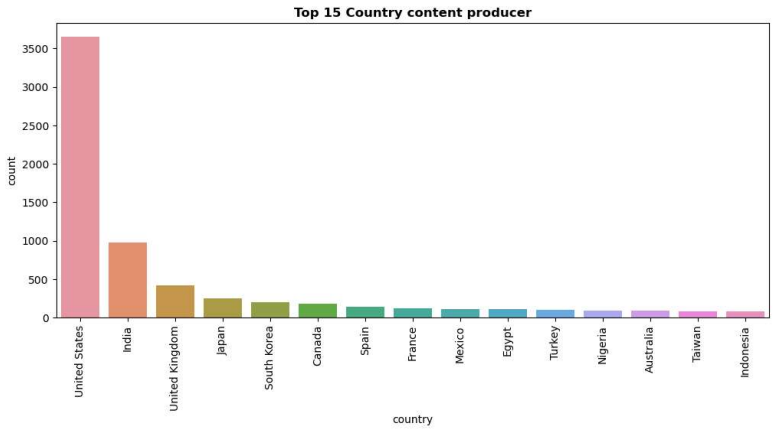
In [70]:
```
top_15_country=df["country"].value_counts().to_frame(
```

In [71]:
```
top_15_country
```

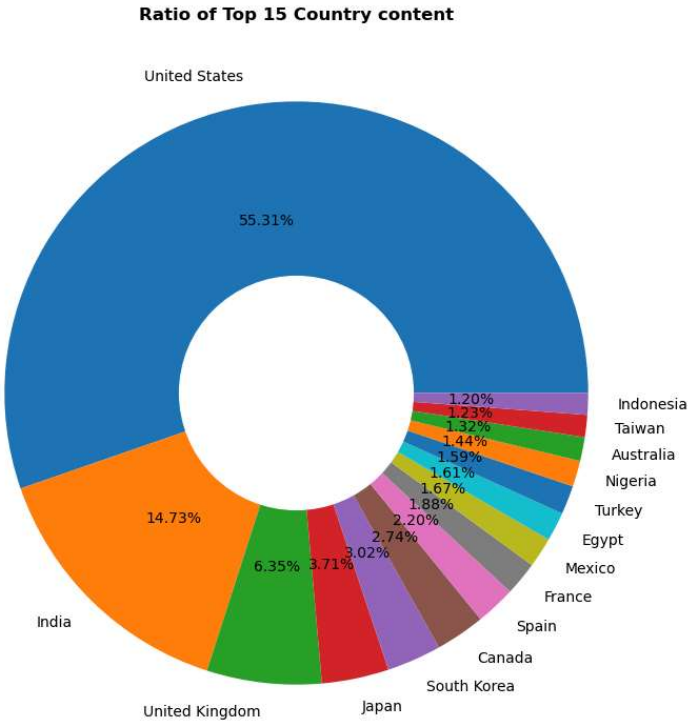
Out[71]:

country	count
United States	3649
India	972
United Kingdom	419
Japan	245
South Korea	199
Canada	181
Spain	145
France	124
Mexico	110
Egypt	106
Turkey	105
Nigeria	95
Australia	87
Taiwan	81
Indonesia	79

In [72]:
```
plt.figure(figsize=(12,5))
sns.barplot(x=top_15_country.index,y="count",data=top
plt.title("Top 15 Country content producer",fontweigh
plt.xticks(rotation=90);
```



In [73]:
```
plt.figure(figsize=(9,9))
plt.pie(top_15_country["count"],labels=top_15_country
plt.title("Ratio of Top 15 Country content",fontweigh
circle = plt.Circle( (0,0), 0.4, color='white')
p=plt.gcf()
p.gca().add_artist(circle);
```



🔍 Inference :

- The majority of the content consistently comes from the United States, comprising around 55%.
- When watching any content, whether movies or TV shows, one out of every two originates from the United States.

🕰️ Analyzing Content Length ⏱️

There are numerous durations and seasons, which makes it challenging to represent in a count plot and pie chart. Therefore, we narrowed down the data to focus on the top 20 durations for analysis.

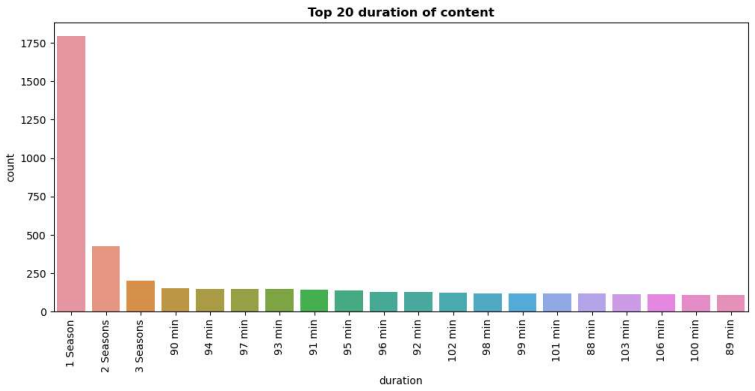
In [74]:
```
top_20_duration=df["duration"].value_counts().head(2
```

In [75]:
```
top_20_duration
```

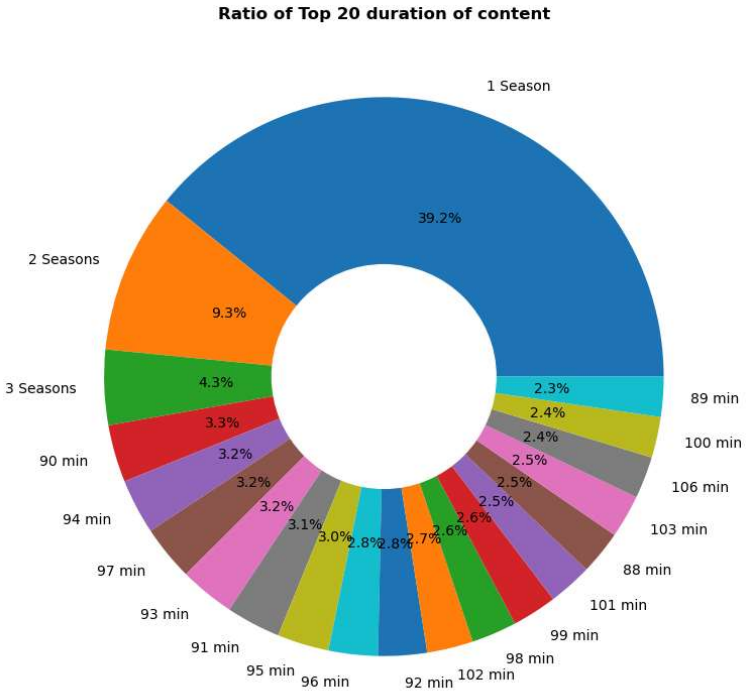
Out[75]:

duration	count
1 Season	1793
2 Seasons	425
3 Seasons	199
90 min	152
94 min	146
97 min	146
93 min	146
91 min	144
95 min	137
96 min	130
92 min	129
102 min	122
98 min	120
99 min	118
101 min	116
88 min	116
103 min	114
106 min	111
100 min	108
89 min	106

In [76]:
```
plt.figure(figsize=(12,5))
sns.barplot(x=top_20_duration.index,y="count",data=t
plt.title("Top 20 duration of content",fontweight="b
plt.xticks(rotation=90);
```



In [77]:
```
plt.figure(figsize=(9,9))
plt.pie(top_20_duration["count"],labels=top_20_durat
plt.title("Ratio of Top 20 duration of content",font
circle = plt.Circle( (0,0), 0.4, color='white')
p=plt.gcf()
p.gca().add_artist(circle);
```



🔍 Inference :

- Most of the content currently has only Season 1, with Seasons 2 and 3 still upcoming.
- In terms of duration, the majority of the content ranges from 90 to 100 minutes.

🎬 Conclusion

Netflix's content analysis reveals that most shows are single-season, with many awaiting subsequent seasons. Movies constitute 70% of the catalog, with durations typically ranging from 90 to 100 minutes, catering to movie viewing habits.

Around 55% of Netflix content originates from the United States, emphasizing its reliance on American productions. Content additions peaked in 2019 with 2000 items, while 2010 saw the fewest additions. Approximately 90% of current content has been added in recent years, reflecting Netflix's strategy of continuous library refreshment.

Content uploads are busiest on Fridays (2,476 titles) and lowest on Sundays (740 titles). July saw the most additions (819 titles), contrasting with

February's least (557 titles), though differences are minor.

Content ratings span TV-MA, TV-14, TV-PG, and R, catering to diverse audience preferences. Netflix's strategy of diverse, recent, and predominantly American content resonates well globally, ensuring a broad subscriber base.