In [1]:
```
#importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading and exploring the dataset

In [2]:
```
# fetching the datasest
df=pd.read_csv('netflix.csv')
df.sample(7)
```

Out[2]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	descri
5026	s5027	TV Show	Best Lover	NaN	Da-hae Lee, Zhou Mi, Wang Qiuzi, Kwang-hyun Pa...	South Korea, China	February 23, 2018	2016	TV-14	1 Season	International TV Shows, Romantic TV Shows, TV ...	Two y star dif backgr
8096	s8097	Movie	Stop at Nothing: The Lance Armstrong Story	Alex Holmes	NaN	Australia, United Kingdom, United States, New ...	February 15, 2015	2014	NR	100 min	Documentaries, International Movies, Sports Mo...	An a dupe worl his t
7773	s7774	TV Show	Power Rangers Samurai	NaN	Alex Heartman, Erika Fong, Hector David Jr., N...	United States	January 1, 2016	2011	TV-Y7	1 Season	Kids' TV	A gene of F Ra must m
954	s955	Movie	The Yeti Adventures	Pierre Greco, Nancy Florence Savard	Rachelle Lefevre, Noel Fisher, Colm Feore, Jul...	Canada	May 1, 2021	2018	TV-PG	85 min	Children & Family Movies, Comedies	An ex detecti off f
4850	s4851	Movie	Ibiza	Alex Richanbach	Gillian Jacobs, Vanessa Bayer, Phoebe Robinson...	United States	May 25, 2018	2018	TV-MA	94 min	Comedies, Romantic Movies	Har s Spain import
2805	s2806	Movie	XV: Beyond the Tryline	Pierre Deschamps	NaN	United Kingdom	March 18, 2020	2016	TV-14	91 min	Documentaries, Sports Movies	Set ag the F World this
1473	s1474	TV Show	Chilling Adventures of Sabrina	NaN	Kiernan Shipka, Ross Lynch, Miranda Otto, Lucy...	United States	December 31, 2020	2020	TV-14	4 Seasons	TV Horror, TV Mysteries, TV Sci-Fi & Fantasy	Magi mis colli half-hu

In [3]:
```
df.shape
```

Out[3]:
```
(8807, 12)
```

There are 8807 rows and 12 columns in the datset

In [4]:
```
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
```

- as we can see there is only one int column and rest are object type

In [ ]:
```
```

In [5]:
```
# no. of unique values in each columns
df.nunique()
```

Out[5]:
```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1767
release_year      74
rating            17
duration         220
listed_in        514
description     8775
dtype: int64
```

In [ ]:
```
```

In [6]:
```
#checking for count of null values in eac column
df.isnull().sum()
```

Out[6]:
```
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
```

- So director column has a lot of Null values followed by country, cast, date_added, rating, duration

In [7]:
```
df.describe(include='all')
```

Out[7]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
count	8807	8807	8807	6173	7982	7976	8797	8807.000000	8803	8804	8807	8807
unique	8807	2	8807	4528	7692	748	1767	NaN	17	220	514	8775
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	NaN	TV-MA	1 Season	Dramas, International Movies	Paranormal activity at a lush, abandoned prope...
freq	1	6131	1	19	19	2818	109	NaN	3207	1793	362	4
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2014.180198	NaN	NaN	NaN	NaN
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.819312	NaN	NaN	NaN	NaN
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1925.000000	NaN	NaN	NaN	NaN
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2013.000000	NaN	NaN	NaN	NaN
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2017.000000	NaN	NaN	NaN	NaN
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2019.000000	NaN	NaN	NaN	NaN
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2021.000000	NaN	NaN	NaN	NaN

In [8]:
```
# fetching the count of all unique values in 'type' column
df['type'].value_counts()
```

Out[8]:
```
Movie      6131
TV Show    2676
Name: type, dtype: int64
```

In [9]:
```
# fetching the count of all unique values in 'rating' column
df['rating'].value_counts()
```

Out[9]:
```
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: rating, dtype: int64
```

In [ ]:

In [10]:
```
# there are multiple directors in some of the cells, so splitting those
a1 = df['director'].apply(lambda x: str(x).split(', ')).tolist()
a1
```

Out[10]:
```
[['Kirsten Johnson'],
 ['nan'],
 ['Julien Leclercq'],
 ['nan'],
 ['nan'],
 ['Mike Flanagan'],
 ['Robert Cullen', 'José Luis Ucha'],
 ['Haile Gerima'],
 ['Andy Devonshire'],
 ['Theodore Melfi'],
 ['nan'],
 ['Kongkiat Komesiri'],
 ['Christian Schwochow'],
 ['Bruno Garotti'],
 ['nan'],
 ['nan'],
 ['Pedro de Echave García', 'Pablo Azorín Williams'],
 ['nan'],
 ['Adam Salky'],
```

In [ ]:
```
```

Unnesting the Dataset

In [11]:
```
# unnesting the directors column, i.e. creating a separate row for every director-title combination
# stack() function stacks the columns into a single column

a1 = df['director'].apply(lambda x: str(x).split(', ')).tolist()
df1 = pd.DataFrame(a1, index=df['title'])
df1 = df1.stack()

df1 = pd.DataFrame(df1.reset_index())
df1.head(10)
```

Out[11]:

	title	level_1	0
0	Dick Johnson Is Dead	0	Kirsten Johnson
1	Blood & Water	0	nan
2	Ganglands	0	Julien Leclercq
3	Jailbirds New Orleans	0	nan
4	Kota Factory	0	nan
5	Midnight Mass	0	Mike Flanagan
6	My Little Pony: A New Generation	0	Robert Cullen
7	My Little Pony: A New Generation	1	José Luis Ucha
8	Sankofa	0	Haile Gerima
9	The Great British Baking Show	0	Andy Devonshire

In [12]:
```
df1.rename(columns={0:'director'}, inplace=True)
df1.drop(columns = ['level_1'], inplace=True)
df1.head(10)
```

Out[12]:

	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
5	Midnight Mass	Mike Flanagan
6	My Little Pony: A New Generation	Robert Cullen
7	My Little Pony: A New Generation	José Luis Ucha
8	Sankofa	Haile Gerima
9	The Great British Baking Show	Andy Devonshire

In [13]:
```
# unnesting the cast column, i.e. creating a separate row for every cast-title combination
# stack() function stacks the columns into a single column
a2=df['cast'].apply(lambda x: str(x).split(', ')).tolist()
df2=pd.DataFrame(a2,index=df['title'])
df2=df2.stack()

df2=pd.DataFrame(df2.reset_index())
df2.rename(columns = {0:'actor'},inplace=True)
df2.drop(columns = ['level_1'],inplace=True)
df2.head(12)
```

Out[13]:

	title	actor
0	Dick Johnson Is Dead	nan
1	Blood & Water	Ama Qamata
2	Blood & Water	Khosi Ngema
3	Blood & Water	Gail Mabalane
4	Blood & Water	Thabang Molaba
5	Blood & Water	Dillon Windvogel
6	Blood & Water	Natasha Thahane
7	Blood & Water	Arno Greeff
8	Blood & Water	Xolile Tshabalala
9	Blood & Water	Getmore Sithole
10	Blood & Water	Cindy Mahlangu
11	Blood & Water	Ryle De Morny

In [14]:
```
# unnesting the listed_in column, i.e. creating a separate row for every genre-title combination
# stack() function stacks the columns into a single column
a3=df['listed_in'].apply(lambda x: str(x).split(', ')).tolist()
df3=pd.DataFrame(a3, index=df['title'])
df3=df3.stack()

df3=pd.DataFrame(df3.reset_index())
df3.rename(columns={0:'genre'},inplace=True)
df3.drop(columns = ['level_1'], inplace=True)
df3.head(12)
```

Out[14]:

	title	genre
0	Dick Johnson Is Dead	Documentaries
1	Blood & Water	International TV Shows
2	Blood & Water	TV Dramas
3	Blood & Water	TV Mysteries
4	Ganglands	Crime TV Shows
5	Ganglands	International TV Shows
6	Ganglands	TV Action & Adventure
7	Jailbirds New Orleans	Docuseries
8	Jailbirds New Orleans	Reality TV
9	Kota Factory	International TV Shows
10	Kota Factory	Romantic TV Shows
11	Kota Factory	TV Comedies

In [15]:
```
# unnesting the country column, i.e. creating a separate row for every country-title combination
# stack() function stacks the columns into a single column
a4=df['country'].apply(lambda x: str(x).split(', ')).tolist()
df4=pd.DataFrame(a4,index=df['title'])
df4=df4.stack()

df4=pd.DataFrame(df4.reset_index())
df4.rename(columns={0:'country'},inplace=True)
df4.drop(columns = ['level_1'], inplace=True)
df4.head(12)
```

Out[15]:

	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
5	Midnight Mass	nan
6	My Little Pony: A New Generation	nan
7	Sankofa	United States
8	Sankofa	Ghana
9	Sankofa	Burkina Faso
10	Sankofa	United Kingdom
11	Sankofa	Germany

In [16]:
```
# merging the unnested director data with unnested actors data
d1 = df2.merge(df1,on=['title'],how='inner')

# merging d1 with unnested genre data
d1 = d1.merge(df3,on=['title'],how='inner')

# merging d1 with unnested country data
d1 = d1.merge(df4,on=['title'],how='inner')

d1
```

Out[16]:

	title	actor	director	genre	country
0	Dick Johnson Is Dead	nan	Kirsten Johnson	Documentaries	United States
1	Blood & Water	Ama Qamata	nan	International TV Shows	South Africa
2	Blood & Water	Ama Qamata	nan	TV Dramas	South Africa
3	Blood & Water	Ama Qamata	nan	TV Mysteries	South Africa
4	Blood & Water	Khosi Ngema	nan	International TV Shows	South Africa
...	...	...	...	...	...
201986	Zubaan	Anita Shabdish	Mozez Singh	International Movies	India
201987	Zubaan	Anita Shabdish	Mozez Singh	Music & Musicals	India
201988	Zubaan	Chittaranjan Tripathy	Mozez Singh	Dramas	India
201989	Zubaan	Chittaranjan Tripathy	Mozez Singh	International Movies	India
201990	Zubaan	Chittaranjan Tripathy	Mozez Singh	Music & Musicals	India

201991 rows × 5 columns

In [ ]:
```
```

Missing value treatment

In [17]:
```
d1.isna().sum()
```

Out[17]:
```
title       0
actor       0
director    0
genre       0
country     0
dtype: int64
```

Although, there are no null values but we observed that the missing values are filled with 'nan'

In [18]:
```
# filling the missing values (nan) of d1 dataframe

d1['actor'] = np.where(d1['actor']=='nan', 'Unknown', d1['actor'])
d1['director'] = np.where(d1['director']=='nan', 'Unknown', d1['director'])
d1['country'] = np.where(d1['country']=='nan', 'Unknown', d1['country'])
```

In [19]:
```
d1.loc[(d1['actor']=='nan') | (d1['director']=='nan') | (d1['country']=='nan') | (d1['genre']=='nan')]
# so we have replaced all the nan values with 'Unknown'
```

Out[19]:

title	actor	director	genre	country

In [ ]:
```
```

In [20]:
```
# merging our unnested data with the original data
# description column is irrelevant here, so we are not considering that at the moment

data = d1.merge(df[['show_id', 'type', 'title', 'date_added',
       'release_year', 'rating', 'duration']],on=['title'],how='left')
data.head()
```

Out[20]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration
0	Dick Johnson Is Dead	Unknown	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	2020	PG-13	90 min
1	Blood & Water	Ama Qamata	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons
2	Blood & Water	Ama Qamata	Unknown	TV Dramas	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons
3	Blood & Water	Ama Qamata	Unknown	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons
4	Blood & Water	Khosi Ngema	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons

In [21]:
```
# making a copy of data in case we need to retrive it later
data2 = data.copy()
np.shares_memory(data2, data) # data2 is a deep copy of data
```

Out[21]:
```
False
```

In [22]:
```
# checking nulls in the above dataframe
data.isnull().sum()
```

Out[22]:
```
title             0
actor             0
director          0
genre             0
country           0
show_id           0
type              0
date_added      158
release_year      0
rating           67
duration          3
dtype: int64
```

We have only 3 Null in duration, lets check those...

In [23]:
```
# checking nulls in the 'duration' column of above dataframe
data.loc[data['duration'].isnull()]
```

Out[23]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration
126537	Louis C.K. 2017	Louis C.K.	Louis C.K.	Movies	United States	s5542	Movie	April 4, 2017	2017	74 min	NaN
131603	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	Movies	United States	s5795	Movie	September 16, 2016	2010	84 min	NaN
131737	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	Movies	United States	s5814	Movie	August 15, 2016	2015	66 min	NaN

It seems that the values in duration column are present in respective rating column So we will replace Nulls in duration column by respective values in rating column

In [24]:
```
data['duration'].fillna(data['rating'], inplace = True)
```

In [25]:
```
data.loc[126537, ['rating']] = np.nan
data.loc[131603, ['rating']] = np.nan
data.loc[131737, ['rating']] = np.nan
```

In [26]:
```
data.loc[[126537, 131603, 131737]]
# now we can see that the duration values are filled and respective rating values are restored to NaN
```

Out[26]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration
126537	Louis C.K. 2017	Louis C.K.	Louis C.K.	Movies	United States	s5542	Movie	April 4, 2017	2017	NaN	74 min
131603	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	Movies	United States	s5795	Movie	September 16, 2016	2010	NaN	84 min
131737	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	Movies	United States	s5814	Movie	August 15, 2016	2015	NaN	66 min

In [27]:
```
data.isnull().sum()
```

Out[27]:
```
title             0
actor             0
director          0
genre             0
country           0
show_id           0
type              0
date_added      158
release_year      0
rating           70
duration          0
dtype: int64
```

In [28]:
```
# checking if there is any other rating cell with value in 'min'
# If case is set to False, it's case-insensitive.
# na=False --> treating missing values as false

data.loc[data['rating'].str.contains('min', case = False, na=False)]

# there are no such cells now
```

Out[28]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration

In [ ]:
```

```

In [29]:
```
# checking nulls in date_added column

data[data['date_added'].isnull()].head()
```

Out[29]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration
136893	A Young Doctor's Notebook and Other Stories	Daniel Radcliffe	Unknown	British TV Shows	United Kingdom	s6067	TV Show	NaN	2013	TV-MA	2 Seasons
136894	A Young Doctor's Notebook and Other Stories	Daniel Radcliffe	Unknown	TV Comedies	United Kingdom	s6067	TV Show	NaN	2013	TV-MA	2 Seasons
136895	A Young Doctor's Notebook and Other Stories	Daniel Radcliffe	Unknown	TV Dramas	United Kingdom	s6067	TV Show	NaN	2013	TV-MA	2 Seasons
136896	A Young Doctor's Notebook and Other Stories	Jon Hamm	Unknown	British TV Shows	United Kingdom	s6067	TV Show	NaN	2013	TV-MA	2 Seasons
136897	A Young Doctor's Notebook and Other Stories	Jon Hamm	Unknown	TV Comedies	United Kingdom	s6067	TV Show	NaN	2013	TV-MA	2 Seasons

In [30]:
```
# filling the NaN values in date_added column:
# we will filter the rows based on release_year and then impute the mode of date_added
# that mode will be filled in place of NaN present in date_added column

for i in data[data['date_added'].isnull()]['release_year'].unique():
    c = data[data['release_year']==i]['date_added'].mode().values[0]
    data.loc[data['release_year']==i,'date_added']=data.loc[data['release_year']==i,'date_added'].fillna(c)
```

In [31]:
```
data.isna().sum()
```

Out[31]:
```
title           0
actor           0
director        0
genre           0
country         0
show_id         0
type            0
date_added      0
release_year    0
rating         70
duration        0
dtype: int64
```

so, all the NaN values in date_added column are filled

In [ ]:
```

```

In [32]:
```
data[data['country']=='Unknown']['director'].value_counts()
```

Out[32]:
```
Unknown              4927
Hidenori Inoue        153
Suhas Kadav           129
Yoshiyuki Tomino      129
S.S. Rajamouli        120
                     ...
Christopher Guest       1
Chris Howe              1
Paul Dugdale            1
Paul M. Green           1
Storm Theunissen        1
Name: director, Length: 373, dtype: int64
```

In [33]:
```
# just for trial
bb = data[data['country']=='Unknown']['director'].value_counts()
bb
```

Out[33]:
```
Unknown              4927
Hidenori Inoue        153
Suhas Kadav           129
Yoshiyuki Tomino      129
S.S. Rajamouli        120
                     ...
Christopher Guest       1
Chris Howe              1
Paul Dugdale            1
Paul M. Green           1
Storm Theunissen        1
Name: director, Length: 373, dtype: int64
```

In [34]:
```
# just for trial
# droping director='Unknown' from bb
bb.drop('Unknown', inplace=True)
bb
```

Out[34]:
```
Hidenori Inoue       153
Suhas Kadav          129
Yoshiyuki Tomino     129
S.S. Rajamouli       120
Rajiv Chilaka        112
                    ...
Christopher Guest      1
Chris Howe             1
Paul Dugdale           1
Paul M. Green          1
Storm Theunissen       1
Name: director, Length: 372, dtype: int64
```

In [35]:
```
bb.index
```

Out[35]:
```
Index(['Hidenori Inoue', 'Suhas Kadav', 'Yoshiyuki Tomino', 'S.S. Rajamouli',
       'Rajiv Chilaka', 'Yoshikazu Yasuhiko', 'Abhishek Chaubey',
       'Adriano Rudiman', 'Kongkiat Komesiri', 'Rathindran R Prasad',
       ...
       'David Cantolla', 'Todd Biermann', 'Manny Rodriguez',
       'Anthony Giordano', 'Padraic McKinley', 'Christopher Guest',
       'Chris Howe', 'Paul Dugdale', 'Paul M. Green', 'Storm Theunissen'],
      dtype='object', length=372)
```

In [ ]:
```
```

In [36]:
```
# just for trial
cc = data[data['director']=='Storm Theunissen']['country'].value_counts(ascending=False)
cc2 = cc[(cc.index!='Unknown')]
if np.all(cc.index=='Unknown'):
    print('Unknown only')
else:
     print(cc2.index[0])
```

```
Unknown only
```

In [37]:
```
# just for trial
cc
```

Out[37]:
```
Unknown    1
Name: country, dtype: int64
```

In [ ]:
```
```

In [38]:
```
# country is imputed on the basis of director
# here the 'Unknown' entries in country column are replaced by respective mode of valid country for that dire

for i in bb.index:
    aa = data[data['director']==i]['country'].value_counts(ascending=False)
    aa2 = aa[(aa.index!='Unknown')]
    if np.all(aa.index=='Unknown'): # if we dont have any entry for country other than 'Unknown', we keep it
        continue
    else:
        data.loc[(data['director']==i) & (data['country']=='Unknown'), 'country'] = aa2.index[0]
        # a director might have more than one valid country, but we want to fill only 'Unknown'
```

In [39]:
```
len(bb) - len(data[data['country']=='Unknown']['director'].value_counts())
```

Out[39]:
```
117
```

so we have successfully filled the 'Unknown' country with their mode for 117 distinct directors

In [40]:
```
data[data['country']=='Unknown']['director'].value_counts()
```

Out[40]:
```
Unknown               4927
Yoshiyuki Tomino       129
Yoshikazu Yasuhiko      93
Adriano Rudiman         78
Kongkiat Komesiri       75
                      ...
Padraic McKinley         1
Todd Biermann            1
Guillermo Garcia         1
Yoo Byung-jae            1
Storm Theunissen         1
Name: director, Length: 255, dtype: int64
```

In [41]:
```
# Trial:
data[data['director']=='Kongkiat Komesiri']['country'].value_counts()
```

Out[41]:
```
Unknown    75
Name: country, dtype: int64
```

so these 255 directors has all the entries in country column as 'Unknown' only

In [ ]:

In [ ]:

In [42]:
```
bb = data[data['country']=='Unknown']['actor'].value_counts()
bb
```

Out[42]:
```
Unknown              281
Hirotaka Suzuoki      21
Toru Furuya           21
Shuichi Ikeda         21
Fuyumi Shiraishi      21
                    ...
Debi Derryberry        1
Salli Saffioti         1
Larissa Gallagher      1
Jonquil Goode          1
Ketan Kava             1
Name: actor, Length: 3233, dtype: int64
```

In [43]:
```
bb.drop('Unknown', inplace=True)
bb
```

Out[43]:
```
Hirotaka Suzuoki     21
Toru Furuya          21
Shuichi Ikeda        21
Fuyumi Shiraishi     21
Yo Inoue             18
                     ..
Debi Derryberry       1
Salli Saffioti        1
Larissa Gallagher     1
Jonquil Goode         1
Ketan Kava            1
Name: actor, Length: 3232, dtype: int64
```

In [ ]:

In [44]:
```
# country column is imputed on the basis of actor
# here the 'Unknown' entries in country column are replaced by respective mode of country for that actor

for i in bb.index:
    aa = data[data['actor']==i]['country'].value_counts(ascending=False)
    aa2 = aa[(aa.index!='Unknown')]
    if np.all(aa.index=='Unknown'): # if we dont have any entry for country other than 'Unknown', we keep it
        continue
    else:
        data.loc[(data['actor']==i) & (data['country']=='Unknown'), 'country'] = aa2.index[0]
```

In [45]:
```
data[data['country']=='Unknown']['director'].value_counts()
```

Out[45]:
```
Unknown                2814
Kongkiat Komesiri        66
Yoshiyuki Tomino         63
Adriano Rudiman          63
Yoshikazu Yasuhiko       54
                       ...
Sandra Restrepo           1
Kubhaer T. Jethwani       1
Avgousta Zourelidi        1
Vijay S. Bhanushali       1
Storm Theunissen          1
Name: director, Length: 216, dtype: int64
```

now we have only 216 directors who dont have any country assigned.

In [46]:
```
data[data['country']=='Unknown']['actor'].value_counts()
```

Out[46]:
```
Unknown                281
Fuyumi Shiraishi        21
Rumiko Ukai             18
Yo Inoue                18
Toshio Furukawa         18
                      ...
Pinky Pal Rajput         1
İlayda Akdoğan           1
Tommaso Basili           1
Cem Yiğit Üzümoğlu       1
Ketan Singh              1
Name: actor, Length: 1788, dtype: int64
```

now we have only 1788 actors (from 3232 earlier) who dont have any country assigned.

In [ ]:

In [47]:
```
len(data.loc[data['country']=='Unknown'])
```

Out[47]:
```
5030
```

In [48]:
```
len(data2.loc[data2['country']=='Unknown'])
```

Out[48]:
```
11897
```

In [49]:
```
print(f"so we have successfully filled unknown country for: {11897-5030} rows")
```

```
so we have successfully filled unknown country for: 6867 rows
```

In [ ]:

In [ ]:

In [50]:
```
data.sample(7)
```

Out[50]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration
150712	Dragonheart 3: The Sorcerer	Julian Morris	Colin Teague	Sci-Fi & Fantasy	United States	s6643	Movie	January 1, 2020	2015	PG-13	97 min
2170	If I Leave Here Tomorrow: A Film About Lynyrd ...	Bob Burns	Stephen Kijak	Documentaries	United States	s97	Movie	September 7, 2021	2018	TV-MA	97 min
54559	#AnneFrank - Parallel Stories	Gengher Gatti	Sabina Fedeli	Documentaries	Italy	s2305	Movie	July 1, 2020	2019	TV-14	95 min
165336	Kung Fu Magoo	Alyson Stoner	Andrés Couturier	Children & Family Movies	United States	s7241	Movie	August 1, 2018	2010	TV-Y7	79 min
93648	Osmosis	Gaël Kamilindi	Unknown	International TV Shows	France	s3965	TV Show	March 29, 2019	2019	TV-MA	1 Season
71659	Whisky	Jorge Bolani	Juan Pablo Rebella	Dramas	Spain	s3003	Movie	January 22, 2020	2004	TV-MA	98 min
200033	What Still Remains	Siena Goines	Josh Mendoza	Thrillers	United States	s8725	Movie	December 21, 2018	2018	TV-14	91 min

In [ ]:

Other data cleaning

In [51]:
```
data['duration'].value_counts()
```

Out[51]:
```
1 Season     35035
2 Seasons     9559
3 Seasons     5084
94 min        4343
106 min       4040
             ...
3 min            4
5 min            3
11 min           2
8 min            2
9 min            2
Name: duration, Length: 220, dtype: int64
```

In [ ]:
```

```

In [52]:
```
data['duration'].unique()
```

Out[52]:
```
array(['90 min', '2 Seasons', '1 Season', '91 min', '125 min',
       '9 Seasons', '104 min', '127 min', '4 Seasons', '67 min', '94 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 min',
       '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 min',
       '117 min', '128 min', '119 min', '143 min', '114 min', '118 min',
       '108 min', '63 min', '121 min', '142 min', '154 min', '120 min',
       '82 min', '109 min', '101 min', '86 min', '229 min', '76 min',
       '89 min', '156 min', '112 min', '107 min', '129 min', '135 min',
       '136 min', '165 min', '150 min', '133 min', '70 min', '84 min',
       '140 min', '78 min', '7 Seasons', '64 min', '59 min', '139 min',
       '69 min', '148 min', '189 min', '141 min', '130 min', '138 min',
       '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 min',
```

In [53]:
```
data['duration_min']=data['duration'].copy()
```

In [54]:
```
data3 = data.copy()
```

In [ ]:
```

```

In [55]:
```
# removing mins from data
data['duration_min']=data['duration_min'].str.replace(" min","")
data.sample(7)
```

Out[55]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration	duration_mi
187119	The Angry Birds Movie 2	Eugenio Derbez	Thurop Van Orman	Children & Family Movies	Finland	s8191	Movie	February 27, 2020	2019	PG	97 min	9
117526	Fullmetal Alchemist	Kenji Utsumi	Unknown	International TV Shows	Japan	s5096	TV Show	January 1, 2018	2003	TV-14	1 Season	1 Seaso
102575	Aalorukkam	Indrans	V C Abhilash	International Movies	India	s4389	Movie	November 15, 2018	2018	TV-PG	122 min	12
82547	Kids on the Block	Selim Bayraktar	Tugçe Soysop	Comedies	Turkey	s3452	Movie	October 4, 2019	2019	TV-PG	102 min	10
26138	Ricky Zoom	Miranda Pointer	Unknown	Kids' TV	China	s1050	TV Show	April 15, 2021	2019	TV-Y	1 Season	1 Seaso
40834	American Horror Story	Connie Britton	Unknown	TV Mysteries	United States	s1702	TV Show	November 13, 2020	2019	TV-MA	9 Seasons	9 Season
152092	Evan Almighty	Wanda Sykes	Tom Shadyac	Faith & Spirituality	United States	s6711	Movie	April 16, 2019	2007	PG	96 min	9

In [56]:
```
data['duration_min'].dtype
```

Out[56]:
```
dtype('O')
```

In [57]:
```
data['duration_min'] = data['duration_min'].astype('U20')
```

In [58]:
```
data.loc[data['duration_min'].str.contains('Season'),['duration_min']]=0
data['duration_min'] = data['duration_min'].astype('int')
data.head()
```

Out[58]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration	duration_min
0	Dick Johnson Is Dead	Unknown	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	2020	PG-13	90 min	90
1	Blood & Water	Ama Qamata	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0
2	Blood & Water	Ama Qamata	Unknown	TV Dramas	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0
3	Blood & Water	Ama Qamata	Unknown	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0
4	Blood & Water	Khosi Ngema	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0

In [59]:
```
data['duration_min'].dtype
```

Out[59]:
```
dtype('int32')
```

In [ ]:

In [60]:
```
# Analysing duration_min column
data[data['duration_min']!=0]['duration_min'].describe()
```

Out[60]:
```
count    145843.00000
mean        106.85579
std          24.69672
min           3.00000
25%          93.00000
50%         104.00000
75%         119.00000
max         312.00000
Name: duration_min, dtype: float64
```

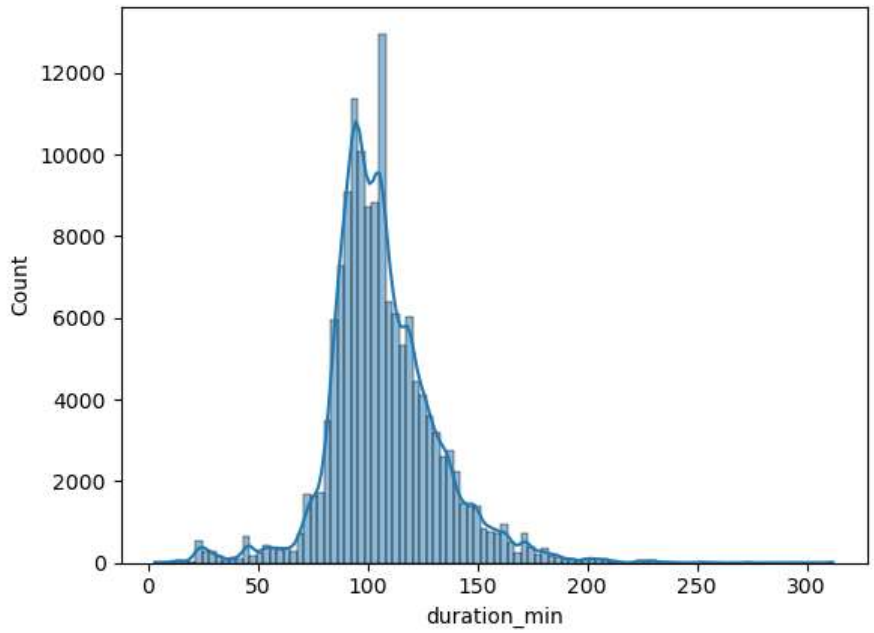
- so we will be dealing with TV show/Movie ranging from 3min to 312 min (excluding seasons duration)
- mean of duration is arround 107 min, while median is 104 min

In [ ]:

In [ ]:

In [61]:
```
sns.histplot(data[data['duration_min']!=0]['duration_min'], bins = 100, kde=True)
```

Out[61]:
```
<Axes: xlabel='duration_min', ylabel='Count'>
```



- duration_min seems to follow normal distribution

In [ ]:
```
```

In [62]:
```
from datetime import datetime
from dateutil.parser import parse
temp = []
for i in data['date_added'].values:
    dt1=parse(i) # parse function is used to convert the date string into a datetime object
    temp.append(dt1.strftime('%Y-%m-%d'))
data['date_added_mod'] = temp
data['date_added_mod'] = pd.to_datetime(data['date_added_mod'])
data['month_added']=data['date_added_mod'].dt.month
data['week_added']=data['date_added_mod'].dt.week
data['year_added']=data['date_added_mod'].dt.year # the year it is added on netflix, different from release_y

data.head()
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_6328\2158172419.py:10: FutureWarning: Series.dt.weekofyear and S
eries.dt.week have been deprecated. Please use Series.dt.isocalendar().week instead.
  data['week_added']=data['date_added_mod'].dt.week
```

Out[62]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration	duration_min	date
0	Dick Johnson Is Dead	Unknown	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	2020	PG-13	90 min	90	
1	Blood & Water	Ama Qamata	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	
2	Blood & Water	Ama Qamata	Unknown	TV Dramas	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	
3	Blood & Water	Ama Qamata	Unknown	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	
4	Blood & Water	Khosi Ngema	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	

In [63]:
```
# Titles like Baahubali(Hindi Version),Baahubali(Tamil Version) are the same movie in different language.
# so we will now remove text enclosed in parentheses (i.e., brackets)

data['title']=data['title'].str.replace(r"\(.*\)","")
data.head()
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_6328\396789613.py:4: FutureWarning: The default value of regex w
ill change from True to False in a future version.
  data['title']=data['title'].str.replace(r"\(.*\)","")
```

Out[63]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration	duration_min	date
0	Dick Johnson Is Dead	Unknown	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	2020	PG-13	90 min	90	
1	Blood & Water	Ama Qamata	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	
2	Blood & Water	Ama Qamata	Unknown	TV Dramas	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	
3	Blood & Water	Ama Qamata	Unknown	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	
4	Blood & Water	Khosi Ngema	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	

In [ ]:

In [65]:
```
data.loc[data['title'].str.contains('Baahubali')]
```

Out[65]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration	duration_
104539	Baahubali: The Beginning	Prabhas	S.S. Rajamouli	Action & Adventure	India	s4482	Movie	October 24, 2018	2015	TV-14	138 min	
104540	Baahubali: The Beginning	Prabhas	S.S. Rajamouli	Dramas	India	s4482	Movie	October 24, 2018	2015	TV-14	138 min	
104541	Baahubali: The Beginning	Prabhas	S.S. Rajamouli	International Movies	India	s4482	Movie	October 24, 2018	2015	TV-14	138 min	
104542	Baahubali: The Beginning	Rana Daggubati	S.S. Rajamouli	Action & Adventure	India	s4482	Movie	October 24, 2018	2015	TV-14	138 min	
104543	Baahubali: The Beginning	Rana Daggubati	S.S. Rajamouli	Dramas	India	s4482	Movie	October 24, 2018	2015	TV-14	138 min	

In [ ]:

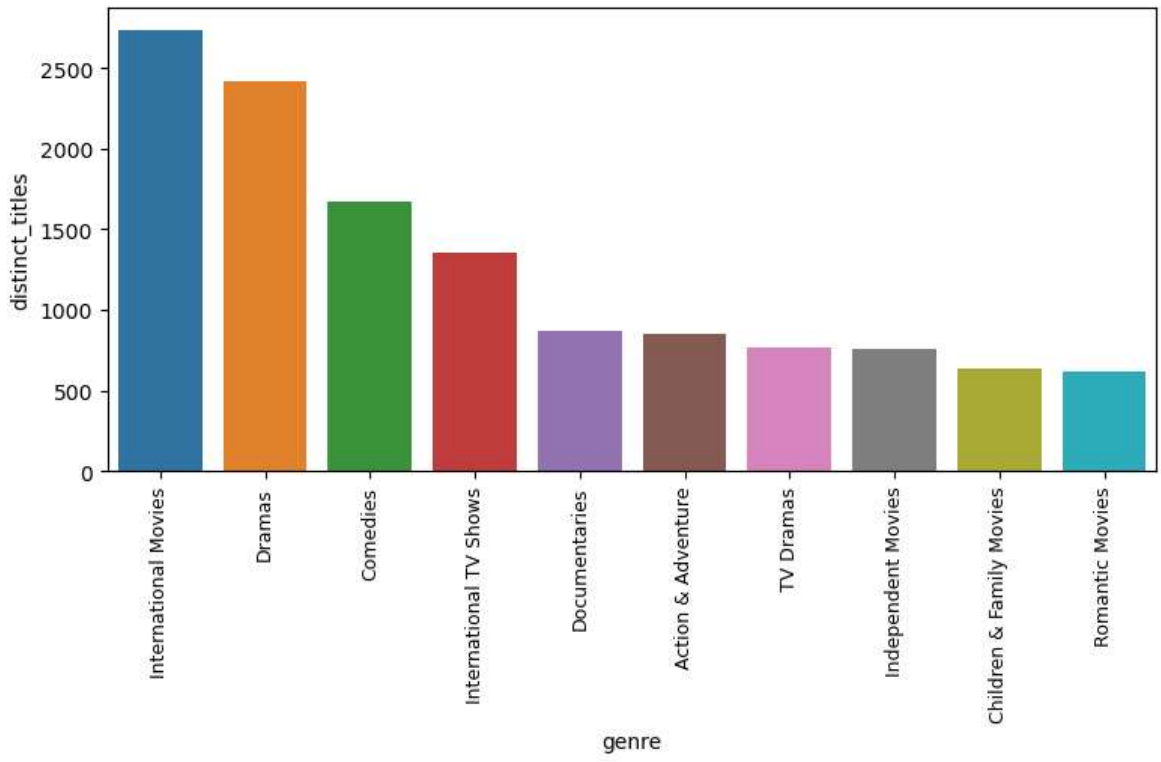
Graphical analysis

In [66]:
```
# number of distinct titles on the basis of genre
gt = data.groupby(['genre']).agg(distinct_titles = ('title','nunique'))
gt.sort_values(by='distinct_titles', ascending = False, inplace=True)
gt.reset_index(inplace=True)
gt
```

Out[66]:

	genre	distinct_titles
0	International Movies	2738
1	Dramas	2418
2	Comedies	1673
3	International TV Shows	1351
4	Documentaries	869
5	Action & Adventure	854
6	TV Dramas	763
7	Independent Movies	756
8	Children & Family Movies	639
9	Romantic Movies	615
10	TV Comedies	581

In [67]:
```
# Plotting top 10 genres
plt.figure(figsize=(9, 4))
sns.barplot(data=gt.head(10), x='genre', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



- International Movies is the top genre followed by Dramas, Comedies, etc

In [68]:
```
# number of distinct titles as per type
tt = data.groupby(['type']).agg(distinct_titles = ('title','nunique'))
tt
```

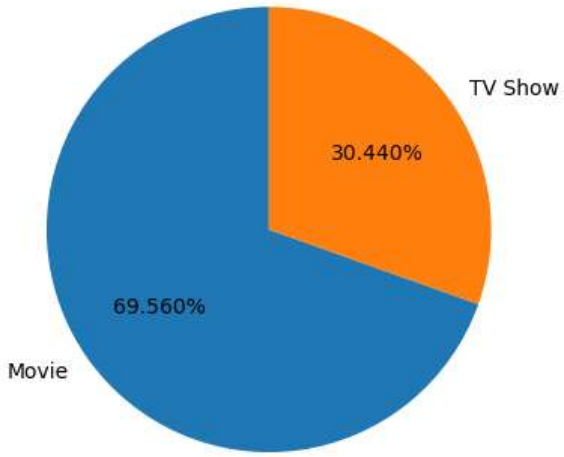
Out[68]:

	distinct_titles
type	
Movie	6115
TV Show	2676

In [69]:
```
tt.values.flatten()
```

Out[69]:
```
array([6115, 2676], dtype=int64)
```

In [70]:
```
plt.pie(tt.values.flatten(), labels=tt.index,
       autopct='%0.3f%%', # to label the wedges with their numeric value
       startangle=90)
plt.show()
```



We have around 70% movies and around 30% TV shows

In [71]:
```
# number of distinct titles as per country
ct = data.groupby(['country']).agg(distinct_titles = ('title','nunique'))
ct.index.values
```

Out[71]:
```
array(['', 'Afghanistan', 'Albania', 'Algeria', 'Angola', 'Argentina',
       'Armenia', 'Australia', 'Austria', 'Azerbaijan', 'Bahamas',
       'Bangladesh', 'Belarus', 'Belgium', 'Bermuda', 'Botswana',
       'Brazil', 'Bulgaria', 'Burkina Faso', 'Cambodia', 'Cambodia,',
       'Cameroon', 'Canada', 'Cayman Islands', 'Chile', 'China',
       'Colombia', 'Croatia', 'Cuba', 'Cyprus', 'Czech Republic',
       'Denmark', 'Dominican Republic', 'East Germany', 'Ecuador',
       'Egypt', 'Ethiopia', 'Finland', 'France', 'Georgia', 'Germany',
       'Ghana', 'Greece', 'Guatemala', 'Hong Kong', 'Hungary', 'Iceland',
       'India', 'Indonesia', 'Iran', 'Iraq', 'Ireland', 'Israel', 'Italy',
       'Jamaica', 'Japan', 'Jordan', 'Kazakhstan', 'Kenya', 'Kuwait',
       'Latvia', 'Lebanon', 'Liechtenstein', 'Lithuania', 'Luxembourg',
       'Malawi', 'Malaysia', 'Malta', 'Mauritius', 'Mexico', 'Mongolia',
       'Montenegro', 'Morocco', 'Mozambique', 'Namibia', 'Nepal',
       'Netherlands', 'New Zealand', 'Nicaragua', 'Nigeria', 'Norway',
       'Pakistan', 'Palestine', 'Panama', 'Paraguay', 'Peru',
       'Philippines', 'Poland', 'Poland,', 'Portugal', 'Puerto Rico',
       'Qatar', 'Romania', 'Russia', 'Samoa', 'Saudi Arabia', 'Senegal',
       'Serbia', 'Singapore', 'Slovakia', 'Slovenia', 'Somalia',
```

after analysing above dataframe, there seems a little flaw where countries like Cambodia and Cambodia, or United States and United States, are shown as different countries

In [72]:
```
data['country'] = data['country'].str.replace(',', '')
data.head()
```

Out[72]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration	duration_min	date
0	Dick Johnson Is Dead	Unknown	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	2020	PG-13	90 min	90	
1	Blood & Water	Ama Qamata	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	
2	Blood & Water	Ama Qamata	Unknown	TV Dramas	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	
3	Blood & Water	Ama Qamata	Unknown	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	
4	Blood & Water	Khosi Ngema	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	0	

In [73]:
```
# number of distinct titles as per country
ct = data.groupby(['country']).agg(distinct_titles = ('title','nunique'))
ct.index.values
```

Out[73]:
```
array(['', 'Afghanistan', 'Albania', 'Algeria', 'Angola', 'Argentina',
       'Armenia', 'Australia', 'Austria', 'Azerbaijan', 'Bahamas',
       'Bangladesh', 'Belarus', 'Belgium', 'Bermuda', 'Botswana',
       'Brazil', 'Bulgaria', 'Burkina Faso', 'Cambodia', 'Cameroon',
       'Canada', 'Cayman Islands', 'Chile', 'China', 'Colombia',
       'Croatia', 'Cuba', 'Cyprus', 'Czech Republic', 'Denmark',
       'Dominican Republic', 'East Germany', 'Ecuador', 'Egypt',
       'Ethiopia', 'Finland', 'France', 'Georgia', 'Germany', 'Ghana',
       'Greece', 'Guatemala', 'Hong Kong', 'Hungary', 'Iceland', 'India',
       'Indonesia', 'Iran', 'Iraq', 'Ireland', 'Israel', 'Italy',
       'Jamaica', 'Japan', 'Jordan', 'Kazakhstan', 'Kenya', 'Kuwait',
       'Latvia', 'Lebanon', 'Liechtenstein', 'Lithuania', 'Luxembourg',
       'Malawi', 'Malaysia', 'Malta', 'Mauritius', 'Mexico', 'Mongolia',
       'Montenegro', 'Morocco', 'Mozambique', 'Namibia', 'Nepal',
       'Netherlands', 'New Zealand', 'Nicaragua', 'Nigeria', 'Norway',
       'Pakistan', 'Palestine', 'Panama', 'Paraguay', 'Peru',
       'Philippines', 'Poland', 'Portugal', 'Puerto Rico', 'Qatar',
       'Romania', 'Russia', 'Samoa', 'Saudi Arabia', 'Senegal', 'Serbia',
       'Singapore', 'Slovakia', 'Slovenia', 'Somalia', 'South Africa',
       'South Korea', 'Soviet Union', 'Spain', 'Sri Lanka', 'Sudan',
       'Sweden', 'Switzerland', 'Syria', 'Taiwan', 'Thailand', 'Turkey',
       'Uganda', 'Ukraine', 'United Arab Emirates', 'United Kingdom',
       'United States', 'Unknown', 'Uruguay', 'Vatican City', 'Venezuela',
       'Vietnam', 'West Germany', 'Zimbabwe'], dtype=object)
```

Now it looks fine

In [74]:
```
ct.sort_values(by='distinct_titles', ascending = False, inplace=True)
ct.reset_index(inplace=True)
ct.head(7)
```

Out[74]:

	country	distinct_titles
0	United States	3874
1	India	1146
2	United Kingdom	847
3	Unknown	568
4	Canada	469
5	France	411
6	Japan	365

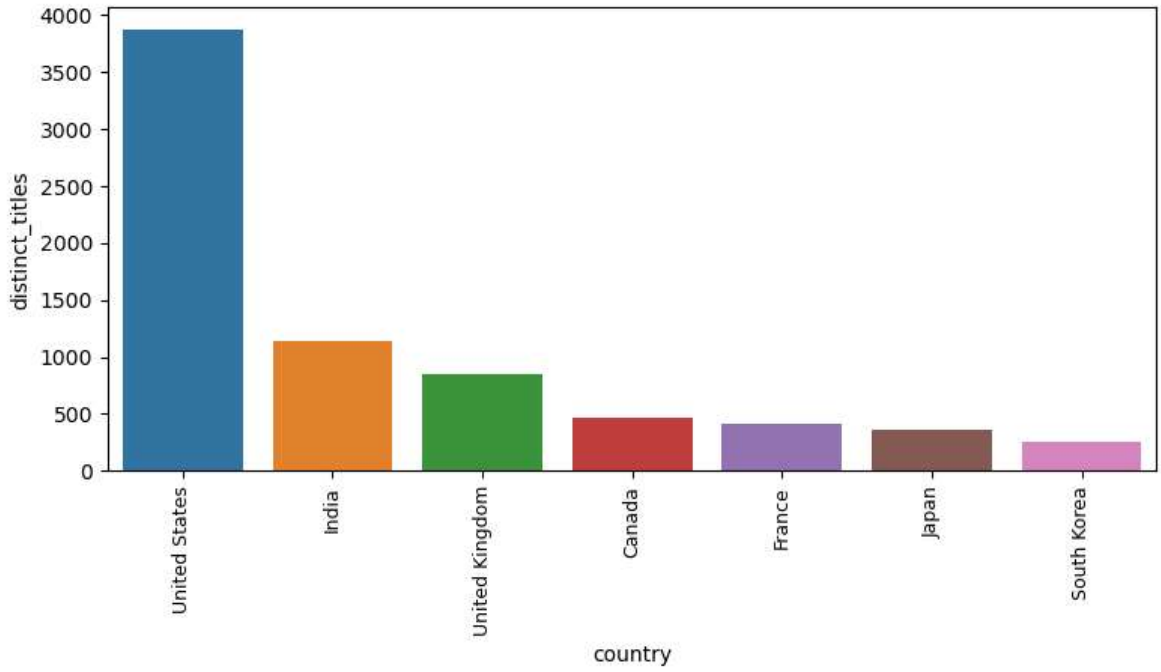
In [75]:
```
ct7 = ct.head(8).drop(3).reset_index(drop=True)
ct7
```

Out[75]:

	country	distinct_titles
0	United States	3874
1	India	1146
2	United Kingdom	847
3	Canada	469
4	France	411
5	Japan	365
6	South Korea	260

- so US has produced most number of titles, followed by India, UK, Canada, France, Japan, etc

In [76]:
```
# Plotting top 7 countries (based on distinct titles produced)
plt.figure(figsize=(9, 4))
sns.barplot(data=ct7, x='country', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



In [ ]:
```
```

In [77]:
```
data[data['title']=='The Hurricane Heist']
```

Out[77]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration	duration_min
191415	The Hurricane Heist	Toby Kebbell	Rob Cohen	Action & Adventure	United Kingdom	s8355	Movie	September 26, 2018	2018	PG-13	103 min	103
191416	The Hurricane Heist	Toby Kebbell	Rob Cohen	Action & Adventure	United States	s8355	Movie	September 26, 2018	2018	PG-13	103 min	103
191417	The Hurricane Heist	Maggie Grace	Rob Cohen	Action & Adventure	United Kingdom	s8355	Movie	September 26, 2018	2018	PG-13	103 min	103
191418	The Hurricane Heist	Maggie Grace	Rob Cohen	Action & Adventure	United States	s8355	Movie	September 26, 2018	2018	PG-13	103 min	103
191419	The Hurricane Heist	Ryan Kwanten	Rob Cohen	Action & Adventure	United Kingdom	s8355	Movie	September 26, 2018	2018	PG-13	103 min	103

In [78]:
```
data.groupby(['title', 'country']).agg(show = ('duration_min', 'mean')).sample(7)
```

Out[78]:

		show
title	country	
Resident Evil: Afterlife	France	97.0
Thomas & Friends: Marvelous Machinery: World of Tomorrow	Unknown	23.0
The Little Nyonya	Unknown	0.0
Move	United States	0.0
Biking Borders	Germany	89.0
Netflix Presents: The Characters	United States	0.0
Colkatay Columbus	India	118.0

In [79]:
```
# eliminating the rows with duration_min=0 or country = 'Unknown'
data11 = data[data['duration_min']!=0]
data11 = data11[data11['country']!='Unknown']
```

In [80]:
```
data11_ct = data11.groupby(['title', 'country']).agg(duration_in_min = ('duration_min', 'mean'))
```

In [81]:
```
data11_ct.reset_index(inplace=True)
```

In [82]:
```
data11_ct
```

Out[82]:

	title	country	duration_in_min
0	#Alive	South Korea	99.0
1	#AnneFrank - Parallel Stories	Italy	95.0
2	#FriendButMarried	Indonesia	102.0
3	#FriendButMarried 2	Indonesia	104.0
4	#Roxy	Canada	105.0
...	...	...	...
7773	Mayurakshi	India	100.0
7774	Kuch Bheege Alfaaz	India	110.0
7775	반드시 잡는다	South Korea	110.0
7776	최강전사 미니특공대 : 영웅의 탄생	South Korea	68.0
7777	최강전사 미니특공대 : 영웅의 탄생	United States	68.0

7778 rows × 3 columns

In [83]:
```
ct7['country'].values
```

Out[83]:
```
array(['United States', 'India', 'United Kingdom', 'Canada', 'France',
       'Japan', 'South Korea'], dtype=object)
```

In [84]:
```
# storing duration data for top7 countries only
data11_ct7 = data11_ct[data11_ct['country'].isin(ct7['country'].values)]
data11_ct7.reset_index(inplace=True, drop=True)
```

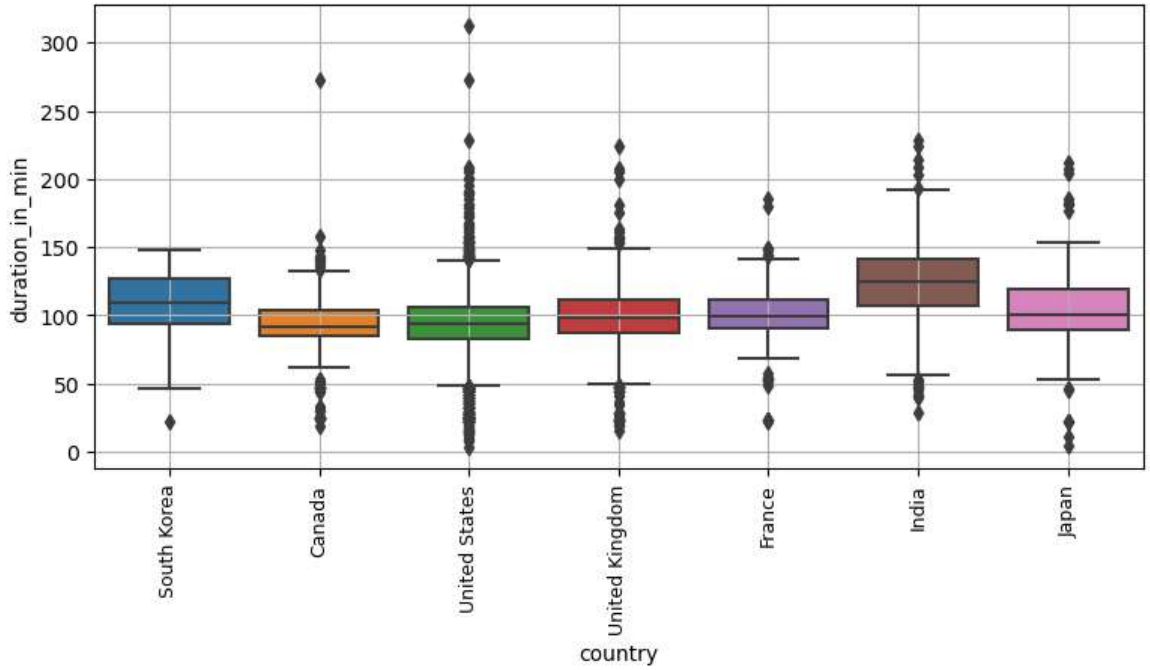
In [85]:
```
data11_ct7
```

Out[85]:

	title	country	duration_in_min
0	#Alive	South Korea	99.0
1	#Roxy	Canada	105.0
2	#Rucker50	United States	56.0
3	#cats_the_mewvie	Canada	90.0
4	#realityhigh	United States	99.0
...	...	...	...
5320	Mayurakshi	India	100.0
5321	Kuch Bheege Alfaaz	India	110.0
5322	반드시 잡는다	South Korea	110.0
5323	최강전사 미니특공대 : 영웅의 탄생	South Korea	68.0
5324	최강전사 미니특공대 : 영웅의 탄생	United States	68.0

5325 rows × 3 columns

In [86]:
```
# visualizing show duration according to country
plt.figure(figsize=(9, 4))
sns.boxplot(data=data11_ct7, x='country', y='duration_in_min')

plt.xticks(rotation=90, fontsize=9)
plt.grid()
plt.show()
```



- Among all the countries, US show duration has many outliers and so is least consistent in show durations
- show duration range of Canada, US, UK, France is quite similar

In [87]:
```
(data11_ct7.groupby(['country']).agg(mean = ('duration_in_min', 'mean'), std = ('duration_in_min', 'std'))).s
```

Out[87]:

country	mean	std
India	123.106858	28.321828
South Korea	107.750000	25.821733
Japan	104.094203	34.609371
France	99.987382	20.585877
United Kingdom	96.815141	28.217522
United States	93.077192	26.048790
Canada	91.636364	24.688723

- India has much higher mean show duration (123 min) compared to other countries

In [ ]:
```
```

In [88]:
```
# number of distinct titles on the basis of director
dt = data.groupby(['director']).agg(distinct_titles = ('title','nunique'))
dt.sort_values(by='distinct_titles', ascending = False, inplace=True)
dt.reset_index(inplace=True)
dt
```

Out[88]:

	director	distinct_titles
0	Unknown	2630
1	Rajiv Chilaka	22
2	Jan Suter	21
3	Raúl Campos	19
4	Marcus Raboy	16
...	...	...
4989	Bradley Walsh	1
4990	Juan Antin	1
4991	Juan Antonio de la Riva	1
4992	Juan Camilo Pinzon	1
4993	María Jose Cuevas	1

4994 rows × 2 columns

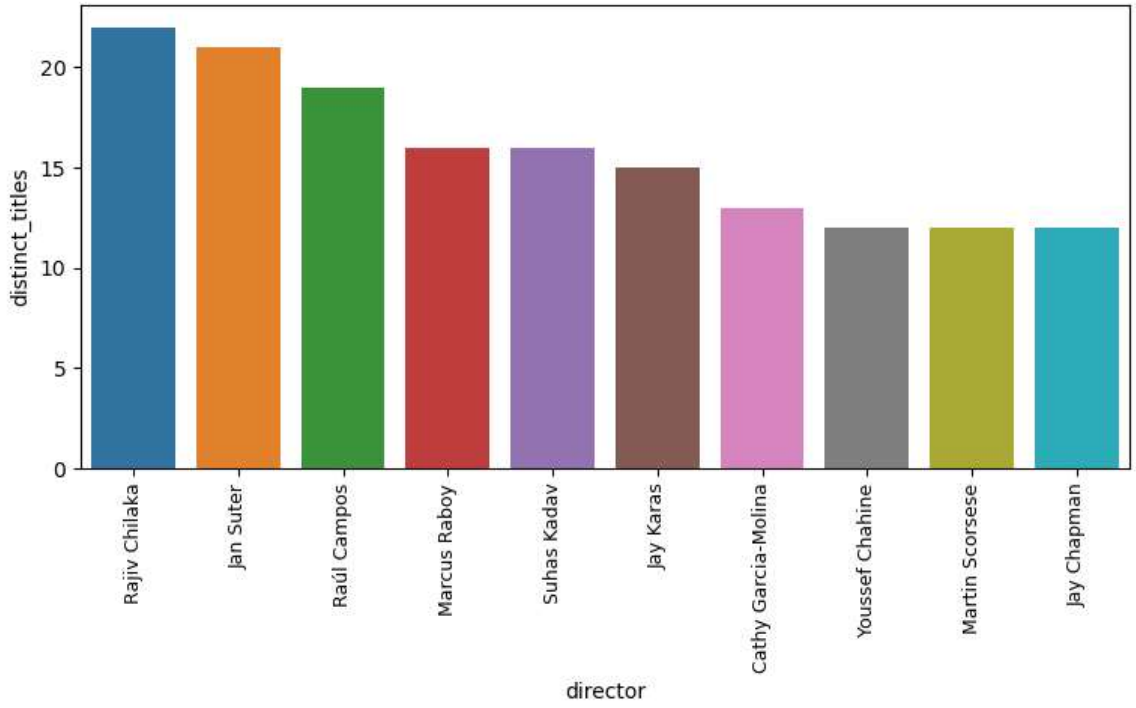
In [89]:
```
dt.drop(0, inplace=True)
dt
```

Out[89]:

	director	distinct_titles
1	Rajiv Chilaka	22
2	Jan Suter	21
3	Raúl Campos	19
4	Marcus Raboy	16
5	Suhas Kadav	16
...	...	...
4989	Bradley Walsh	1
4990	Juan Antin	1
4991	Juan Antonio de la Riva	1
4992	Juan Camilo Pinzon	1
4993	María Jose Cuevas	1

4993 rows × 2 columns

In [90]:
```
# Plotting top 10 directors (based on distinct titles produced)
plt.figure(figsize=(9, 4))
sns.barplot(data=dt.head(10), x='director', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



Rajiv Chilaka, Jan Suter and Raul Campos are the most popular directors across Netflix

In [91]:
```
data
```

Out[91]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration	duration_n
0	Dick Johnson Is Dead	Unknown	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	2020	PG-13	90 min	
1	Blood & Water	Ama Qamata	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	
2	Blood & Water	Ama Qamata	Unknown	TV Dramas	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	
3	Blood & Water	Ama Qamata	Unknown	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	
4	Blood & Water	Khosi Ngema	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	
...	...	...	...	...	...	...	...	...	...	...	...	
201986	Zubaan	Anita Shabdish	Mozez Singh	International Movies	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	
201987	Zubaan	Anita Shabdish	Mozez Singh	Music & Musicals	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	
201988	Zubaan	Chittaranjan Tripathy	Mozez Singh	Dramas	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	
201989	Zubaan	Chittaranjan Tripathy	Mozez Singh	International Movies	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	
201990	Zubaan	Chittaranjan Tripathy	Mozez Singh	Music & Musicals	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	

201991 rows × 16 columns

In [92]:
```
# number of distinct titles released on the basis of release_year
ryt = data.groupby(['release_year']).agg(distinct_titles_released = ('title','nunique'))
ryt.sort_values(by='distinct_titles_released', ascending = False, inplace=True)
ryt.reset_index(inplace=True)
ryt
```

Out[92]:

	release_year	distinct_titles_released
0	2018	1146
1	2017	1030
2	2019	1023
3	2020	953
4	2016	902
...	...	...
69	1959	1
70	1961	1
71	1947	1
72	1966	1
73	1925	1

74 rows × 2 columns

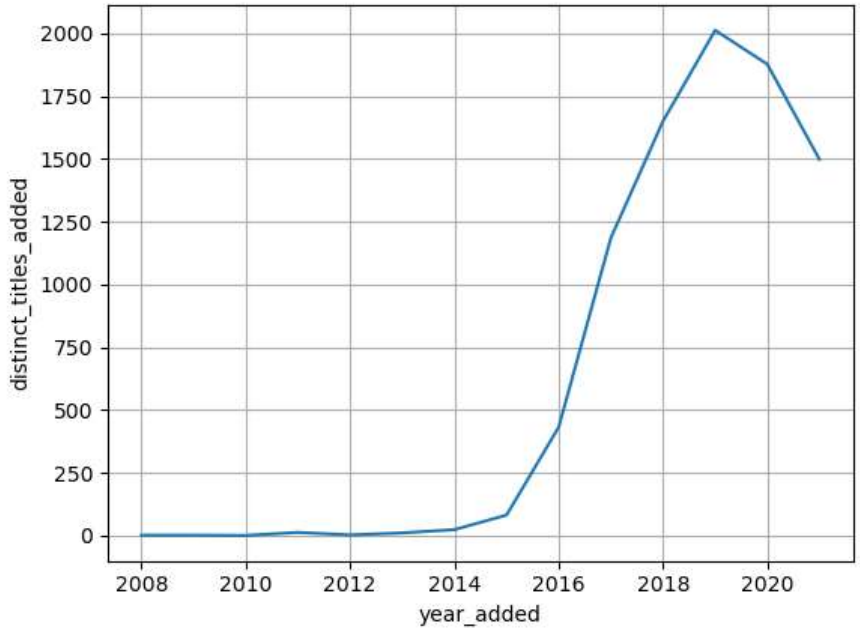
In [ ]:
```
```

In [93]:
```
# number of distinct titles added on the basis of year_added
yat = data.groupby(['year_added']).agg(distinct_titles_added = ('title','nunique'))
yat.sort_values(by='distinct_titles_added', ascending = False, inplace=True)
yat.reset_index(inplace=True)
yat
```

Out[93]:

	year_added	distinct_titles_added
0	2019	2012
1	2020	1877
2	2018	1650
3	2021	1498
4	2017	1185
5	2016	432
6	2015	82
7	2014	24
8	2011	13
9	2013	11
10	2012	3
11	2008	2
12	2009	2
13	2010	1

In [94]:
```
sns.lineplot(yat, x='year_added', y='distinct_titles_added')

plt.grid()
plt.show()
```



- So number of titles added on Netflix took a great pace from 2016 and was on rise till 2019
- from 2019-2021 the number of titles added has declined, this might be due to the corona outbreak

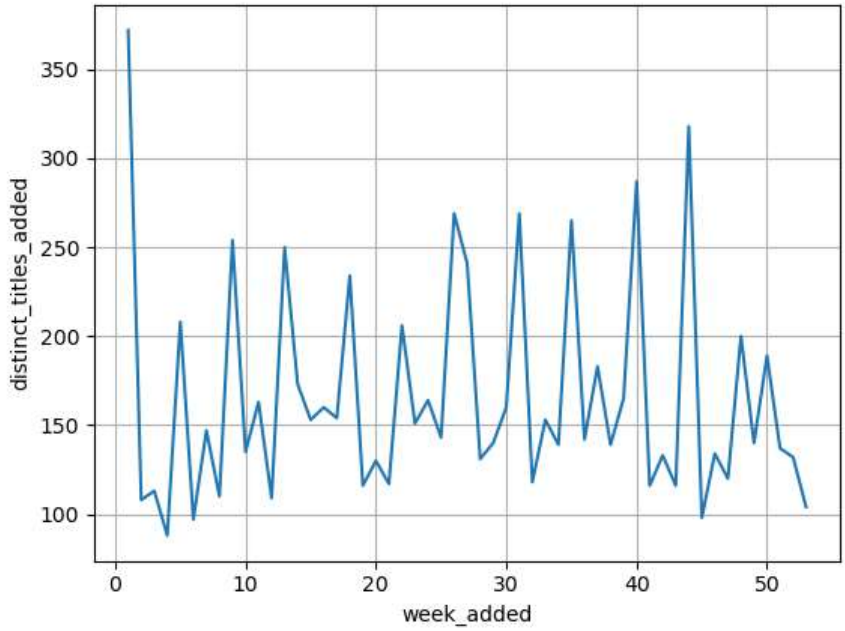
In [ ]:
```
```

In [95]:
```
# number of distinct titles added on the basis of week_added
wat = data.groupby(['week_added']).agg(distinct_titles_added = ('title','nunique'))
wat.sort_values(by='distinct_titles_added', ascending = False, inplace=True)
wat.reset_index(inplace=True)
wat
```

Out[95]:

	week_added	distinct_titles_added
0	1	372
1	44	318
2	40	287
3	31	269
4	26	269
5	35	265
6	9	254
7	13	250
8	27	241
9	18	234
10	5	208

In [96]:
```
sns.lineplot(wat, x='week_added', y='distinct_titles_added')

plt.grid()
plt.show()
```



Most of the titles on Netflix are added in the first week of the year and it follows a cyclical pattern througout the year

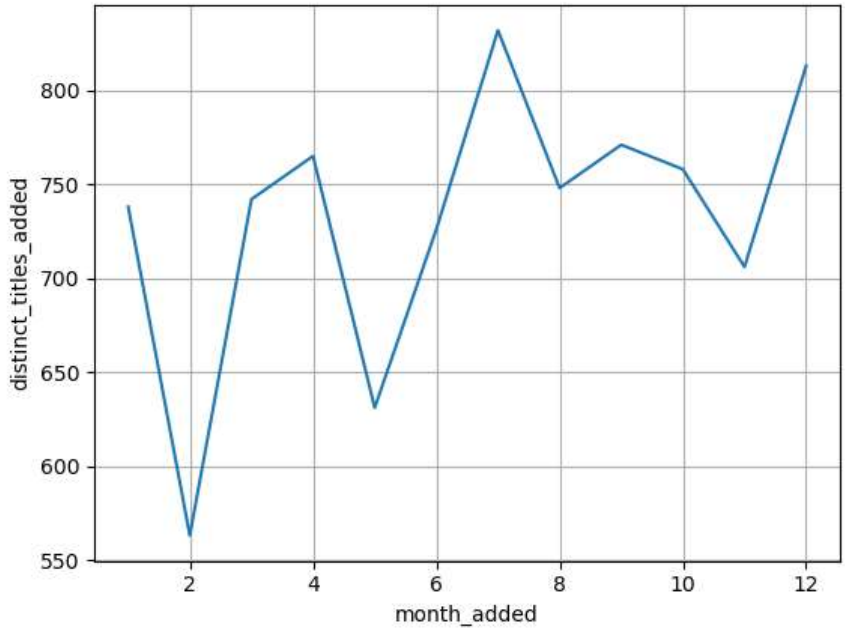
In [ ]:
```
```

In [97]:
```
# number of distinct titles added on the basis of month_added
mat = data.groupby(['month_added']).agg(distinct_titles_added = ('title','nunique'))
mat.sort_values(by='distinct_titles_added', ascending = False, inplace=True)
mat.reset_index(inplace=True)
mat
```

Out[97]:

	month_added	distinct_titles_added
0	7	832
1	12	813
2	9	771
3	4	765
4	10	758
5	8	748
6	3	742
7	1	738
8	6	726
9	11	706
10	5	631
11	2	563

In [98]:
```
sns.lineplot(mat, x='month_added', y='distinct_titles_added')

plt.grid()
plt.show()
```



- Most titles are added in July, December. It might be attributed to summer break in July and X-Mas, corporate holidays in December
- Least titles are added in February

In [ ]:
```
```

Questions to explore

Questions to be Explored Now for Recommendations

1. So this time, the granularity level is country and analysis of TV Shows/Movies the country brings. I am going to consider only the top countries individually for TV Shows and Movies. There are definitely some common countries too which bring out quality content in both TV Shows and Movies.
2. Which Genres do these countries offer and what are the intended audiences(Ratings) which are popular in Netflix?

3)In case of Movies, what is the duration/length of movies which makes them special and depicts attention span?

4)Who are the popular actors/directors across TV Shows and Movies in these countries?

5)In what time of the year, people tend to watch movies and shows in these countries?

6)Popular Actor and Director Combinations in these countries

How are Movie/TV show distributed across top countries?

In [99]:
```
tp = data.groupby(['country', 'type']).agg(distinct_titles = ('title','nunique'))
tp.reset_index(inplace=True)
tp.sort_values(by='distinct_titles', ascending=False, inplace=True)
```

In [100]:
```
tp
```

Out[100]:

	country	type	distinct_titles
175	United States	Movie	2863
67	India	Movie	1045
176	United States	TV Show	1011
173	United Kingdom	Movie	568
178	Unknown	TV Show	339
...	...	...	...
64	Hungary	TV Show	1
46	Ecuador	Movie	1
154	Sri Lanka	Movie	1
155	Sudan	Movie	1
49	Ethiopia	Movie	1

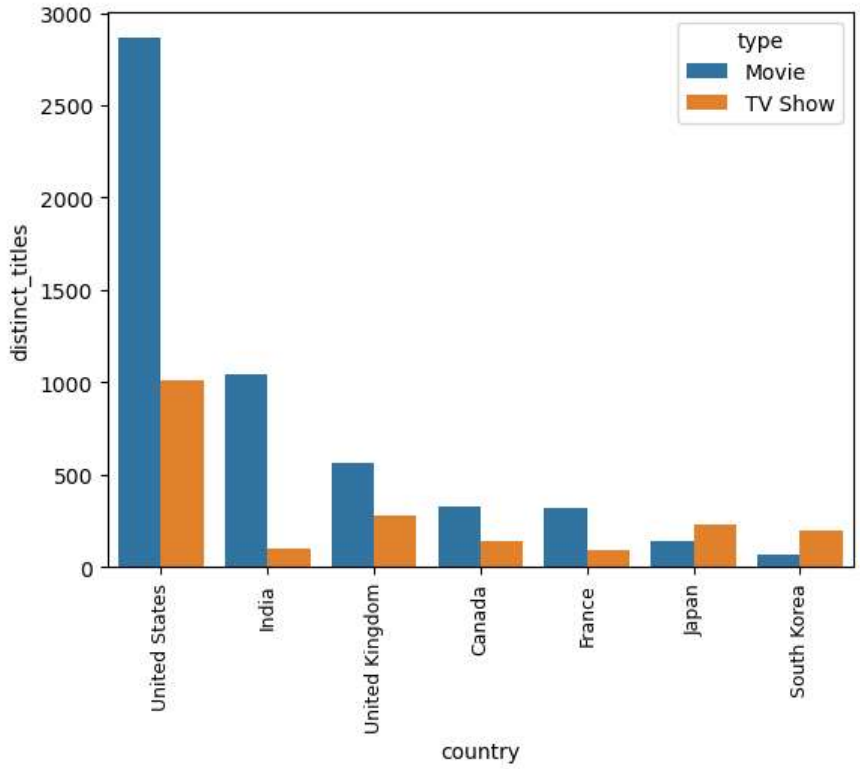
187 rows × 3 columns

In [101]:
```
tp7 = tp[tp['country'].isin(ct7['country'].values)]
tp7
```

Out[101]:

	country	type	distinct_titles
175	United States	Movie	2863
67	India	Movie	1045
176	United States	TV Show	1011
173	United Kingdom	Movie	568
27	Canada	Movie	330
52	France	Movie	317
174	United Kingdom	TV Show	279
81	Japan	TV Show	227
150	South Korea	TV Show	196
28	Canada	TV Show	139
80	Japan	Movie	138
68	India	TV Show	101
53	France	TV Show	94
149	South Korea	Movie	64

In [102]:
```
sns.barplot(data = tp7, x='country', y='distinct_titles', hue='type')
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



- As we can observe, fraction of TV shows in India is significantly lower than other countries
- Japan and South Korea has more TV shows than Movies

In [ ]:

In [ ]:

Who are Popular directors in top countries?

In [103]:
```
ct7
```

Out[103]:

	country	distinct_titles
0	United States	3874
1	India	1146
2	United Kingdom	847
3	Canada	469
4	France	411
5	Japan	365
6	South Korea	260

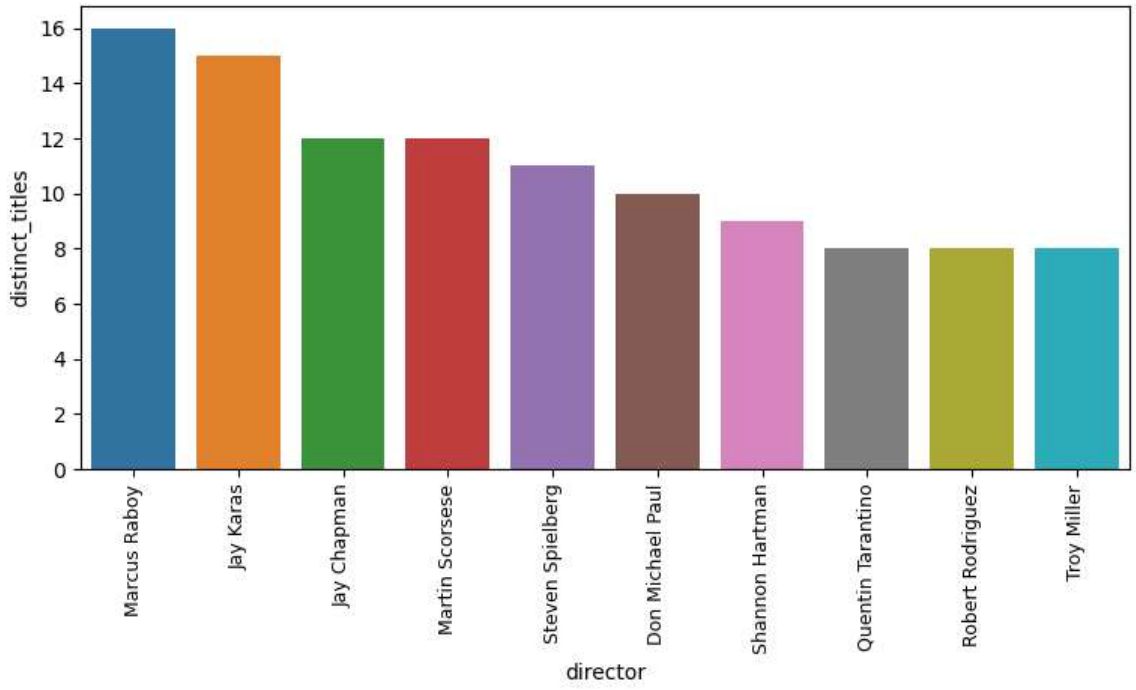
In [104]:
```
# creating a country wise dataframe for US, India, UK, Japan, South Korea
data_US = data.loc[(data['country']=='United States') & (data['director']!='Unknown')]
data_India = data.loc[(data['country']=='India') & (data['director']!='Unknown')]
data_UK = data.loc[(data['country']=='United Kingdom') & (data['director']!='Unknown')]
data_Japan = data.loc[(data['country']=='Japan') & (data['director']!='Unknown')]
data_SK = data.loc[(data['country']=='South Korea') & (data['director']!='Unknown')]
```

In [ ]:

In [ ]:

In [ ]:

In [105]:
```
# top directors from US
# number of distinct titles on the basis of director

dtc = data_US.groupby(['director']).agg(distinct_titles = ('title','nunique'))
dtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
dtc.reset_index(inplace=True)

# Plotting top 10 directors (based on distinct titles produced)
plt.figure(figsize=(9, 4))
sns.barplot(data=dtc.head(10), x='director', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```

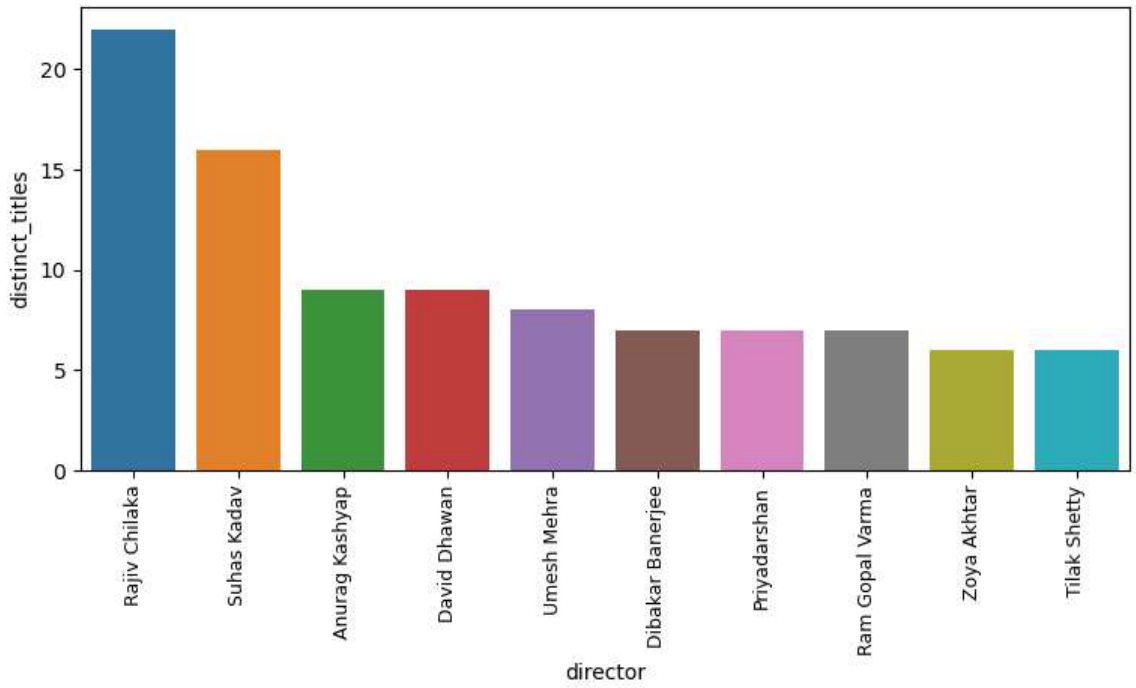


In [106]:
```
print("So, The top 5 directors based on no. of titled produced from US are:")
print("")
for i in range(5):
    print(f"{dtc['director'].values[i]}, ", end="")
```

```
So, The top 5 directors based on no. of titled produced from US are:

Marcus Raboy, Jay Karas, Jay Chapman, Martin Scorsese, Steven Spielberg,
```

In [ ]:

In [107]:
```
# top directors from India
# number of distinct titles on the basis of director

dtc = data_India.groupby(['director']).agg(distinct_titles = ('title','nunique'))
dtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
dtc.reset_index(inplace=True)

# Plotting top 10 directors (based on distinct titles produced)
plt.figure(figsize=(9, 4))
sns.barplot(data=dtc.head(10), x='director', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



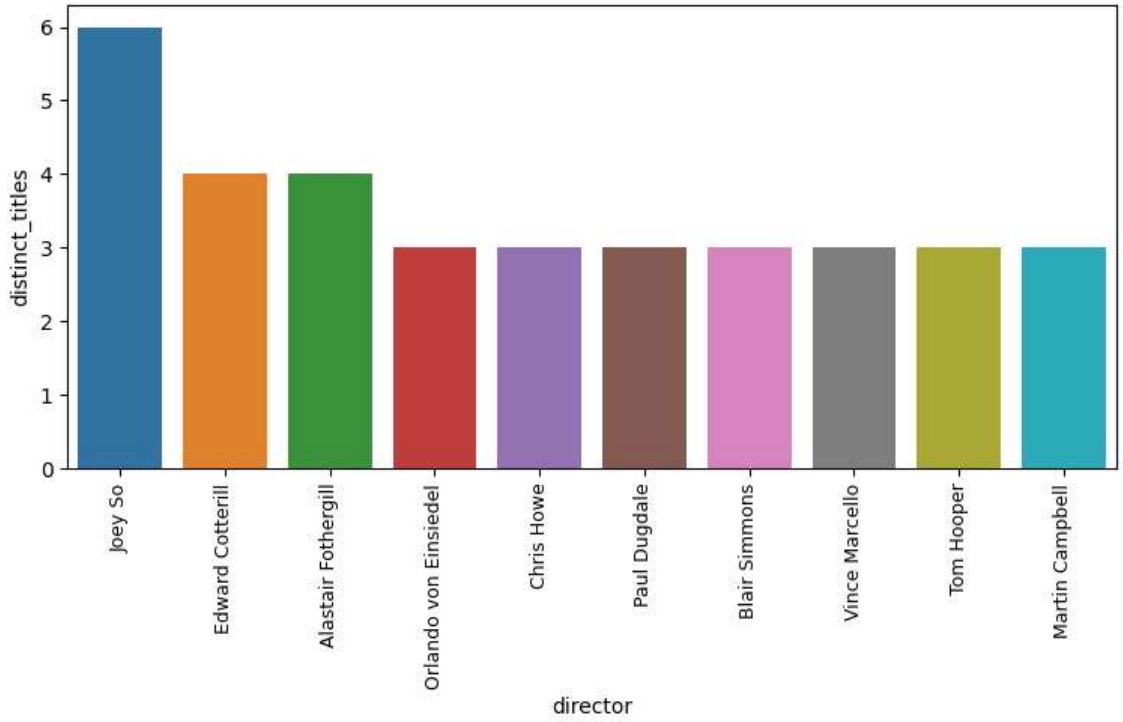
In [108]:
```
print("So, The top 5 directors based on no. of titled produced from India are:")
print("")
for i in range(5):
    print(f"{dtc['director'].values[i]}, ", end="")
```

```
So, The top 5 directors based on no. of titled produced from India are:

Rajiv Chilaka, Suhas Kadav, Anurag Kashyap, David Dhawan, Umesh Mehra,
```

In [ ]:

In [ ]:

In [109]:
```
# top directors from UK
# number of distinct titles on the basis of director

dtc = data_UK.groupby(['director']).agg(distinct_titles = ('title','nunique'))
dtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
dtc.reset_index(inplace=True)

# Plotting top 10 directors (based on distinct titles produced)
plt.figure(figsize=(9, 4))
sns.barplot(data=dtc.head(10), x='director', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



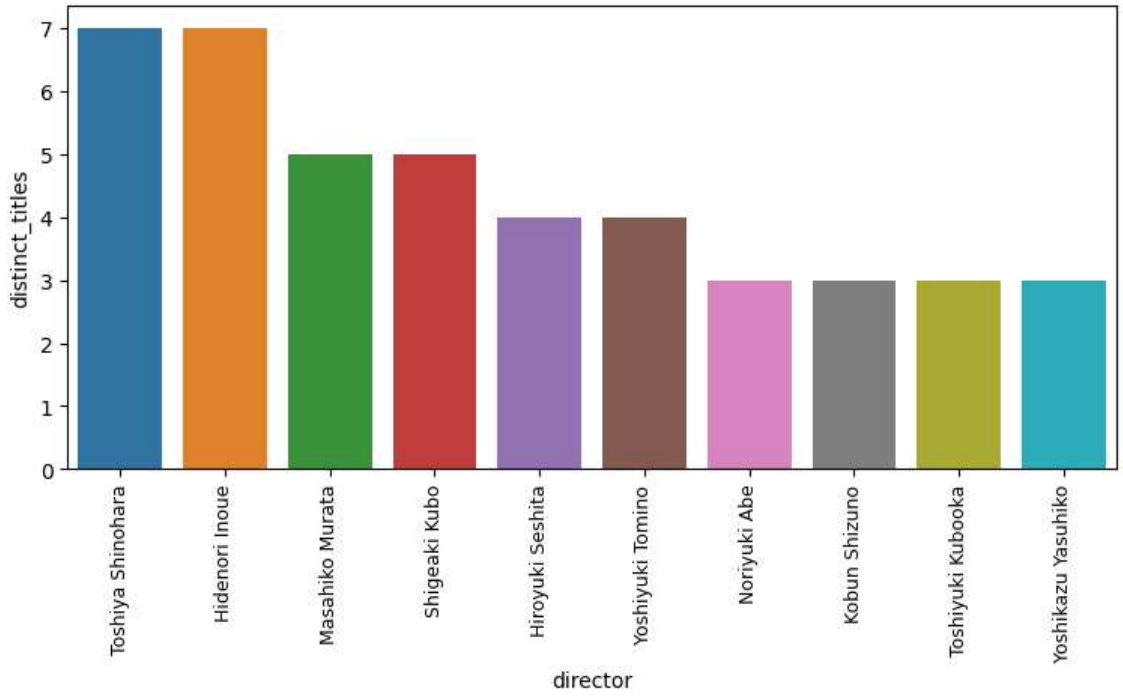
In [110]:
```
print("So, The top 5 directors based on no. of titled produced from UK are:")
print("")
for i in range(5):
    print(f"{dtc['director'].values[i]}, ", end="")
```

```
So, The top 5 directors based on no. of titled produced from UK are:

Joey So, Edward Cotterill, Alastair Fothergill, Orlando von Einsiedel, Chris Howe,
```

In [ ]:

In [ ]:

In [111]:
```
# top directors from Japan
# number of distinct titles on the basis of director

dtc = data_Japan.groupby(['director']).agg(distinct_titles = ('title','nunique'))
dtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
dtc.reset_index(inplace=True)

# Plotting top 10 directors (based on distinct titles produced)
plt.figure(figsize=(9, 4))
sns.barplot(data=dtc.head(10), x='director', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```

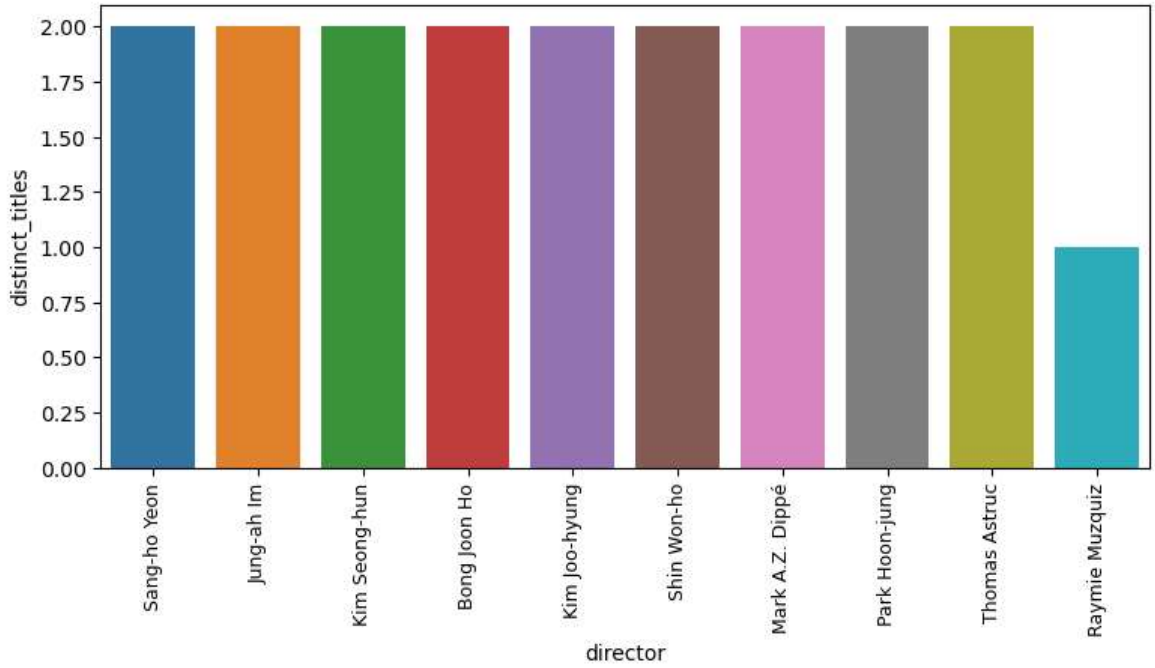


In [112]:
```
print("So, The top 5 directors based on no. of titled produced from Japan are:")
print("")
for i in range(5):
    print(f"{dtc['director'].values[i]}, ", end="")
```

```
So, The top 5 directors based on no. of titled produced from Japan are:

Toshiya Shinohara, Hidenori Inoue, Masahiko Murata, Shigeaki Kubo, Hiroyuki Seshita,
```

In [ ]:

In [113]:
```
# top directors from South Korea
# number of distinct titles on the basis of director

dtc = data_SK.groupby(['director']).agg(distinct_titles = ('title','nunique'))
dtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
dtc.reset_index(inplace=True)

# Plotting top 10 directors (based on distinct titles produced)
plt.figure(figsize=(9, 4))
sns.barplot(data=dtc.head(10), x='director', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



In [114]:
```
print("So, The top directors based on no. of titled produced from South Korea are:")
print("")
for i in range(9):
    print(f"{dtc['director'].values[i]}, ", end="")
```

```
So, The top directors based on no. of titled produced from South Korea are:

Sang-ho Yeon, Jung-ah Im, Kim Seong-hun, Bong Joon Ho, Kim Joo-hyung, Shin Won-ho, Mark A.Z. Dippé, Park Ho
on-jung, Thomas Astruc,
```

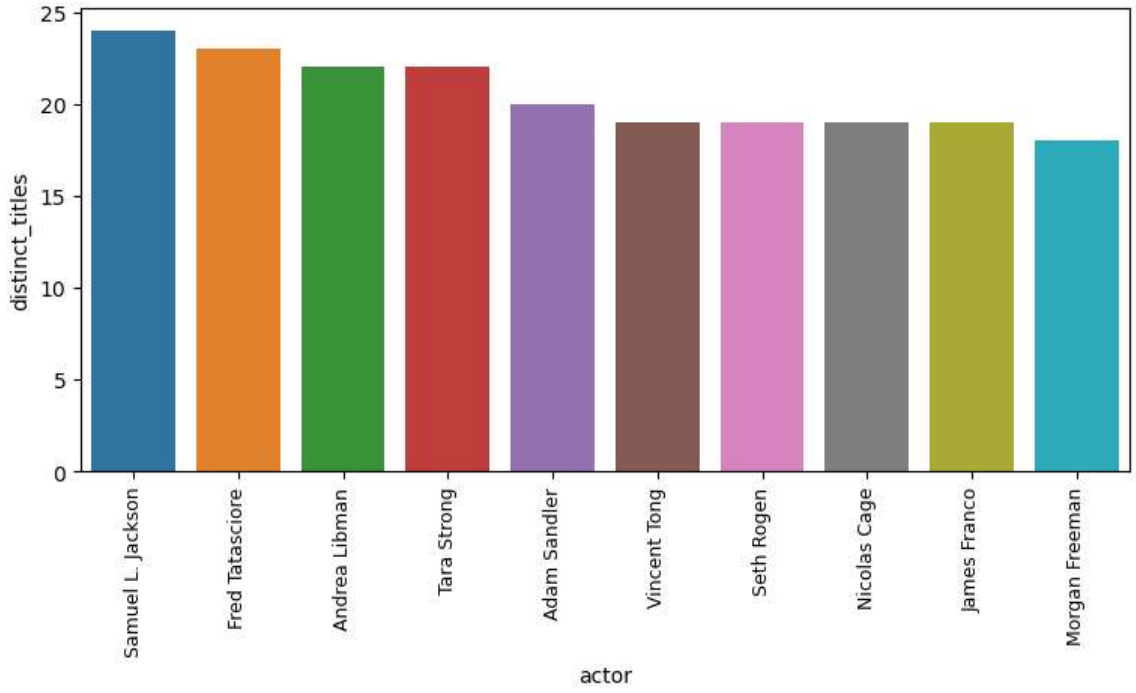
In [ ]:

Who are Popular actors in top countries?

In [115]:
```
# creating a country wise dataframe for US, India, UK, Japan, South Korea
data_US = data.loc[(data['country']=='United States') & (data['actor']!='Unknown')]
data_India = data.loc[(data['country']=='India') & (data['actor']!='Unknown')]
data_UK = data.loc[(data['country']=='United Kingdom') & (data['actor']!='Unknown')]
data_Japan = data.loc[(data['country']=='Japan') & (data['actor']!='Unknown')]
data_SK = data.loc[(data['country']=='South Korea') & (data['actor']!='Unknown')]
```

In [ ]:

In [116]:
```
# top actors from US
# number of distinct titles on the basis of actors

atc = data_US.groupby(['actor']).agg(distinct_titles = ('title','nunique'))
atc.sort_values(by='distinct_titles', ascending = False, inplace=True)
atc.reset_index(inplace=True)

# Plotting top 10 actors (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=atc.head(10), x='actor', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```

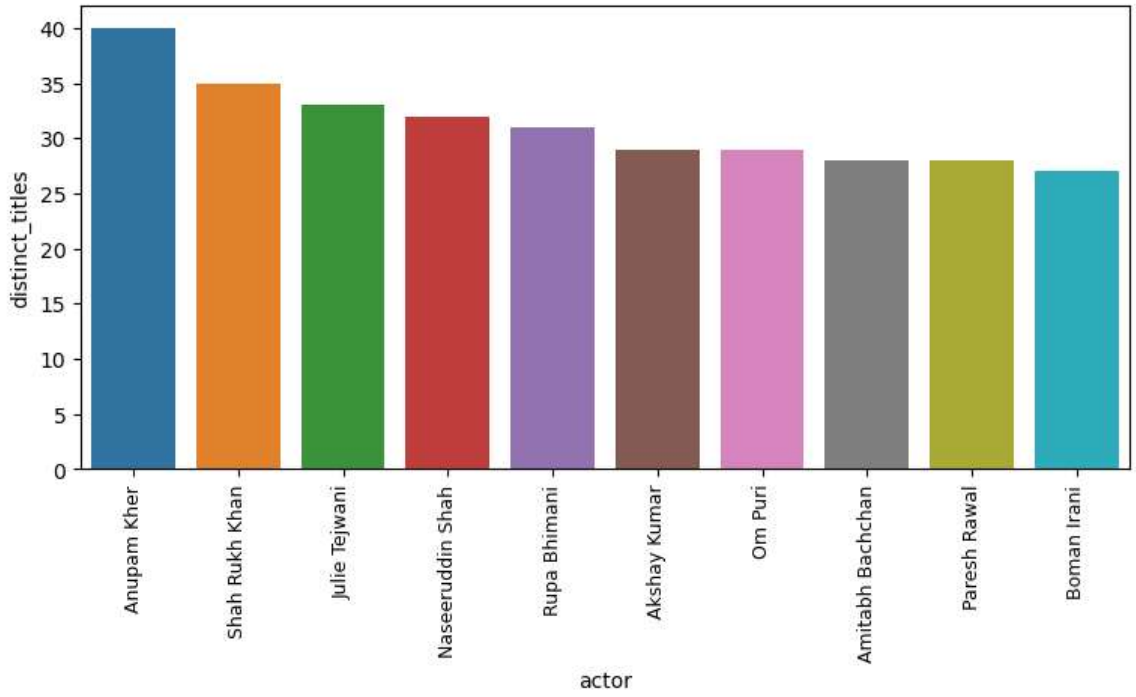


In [117]:
```
print("So, The top 5 actors based on no. of titled from US are:")
print("")
for i in range(5):
    print(f"{atc['actor'].values[i]}, ", end="")
```

```
So, The top 5 actors based on no. of titled from US are:

Samuel L. Jackson, Fred Tatasciore, Andrea Libman, Tara Strong, Adam Sandler,
```

In [ ]:

In [118]:
```
# top actors from India
# number of distinct titles on the basis of actors

atc = data_India.groupby(['actor']).agg(distinct_titles = ('title','nunique'))
atc.sort_values(by='distinct_titles', ascending = False, inplace=True)
atc.reset_index(inplace=True)

# Plotting top 10 actors (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=atc.head(10), x='actor', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```

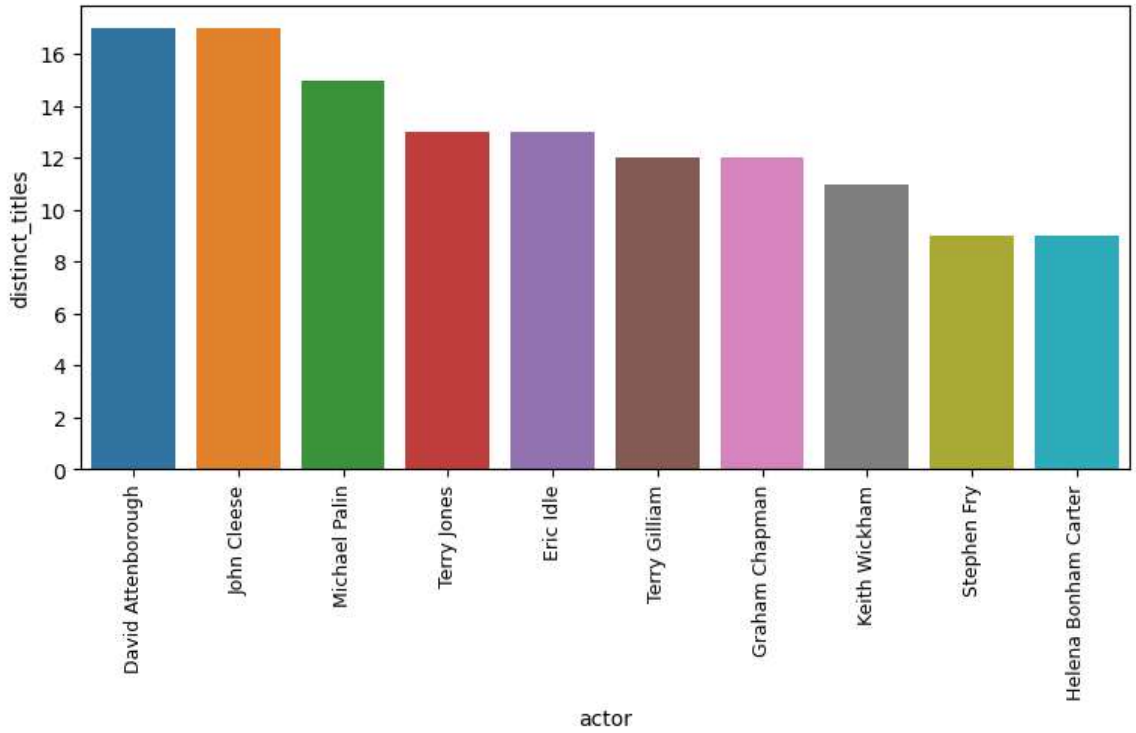


In [119]:
```
print("So, The top 5 actors based on no. of titled from India are:")
print("")
for i in range(5):
    print(f"{atc['actor'].values[i]}, ", end="")
```

```
So, The top 5 actors based on no. of titled from India are:

Anupam Kher, Shah Rukh Khan, Julie Tejwani, Naseeruddin Shah, Rupa Bhimani,
```

In [ ]:

In [120]:
```
# top actors from UK
# number of distinct titles on the basis of actors

atc = data_UK.groupby(['actor']).agg(distinct_titles = ('title','nunique'))
atc.sort_values(by='distinct_titles', ascending = False, inplace=True)
atc.reset_index(inplace=True)

# Plotting top 10 actors (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=atc.head(10), x='actor', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```

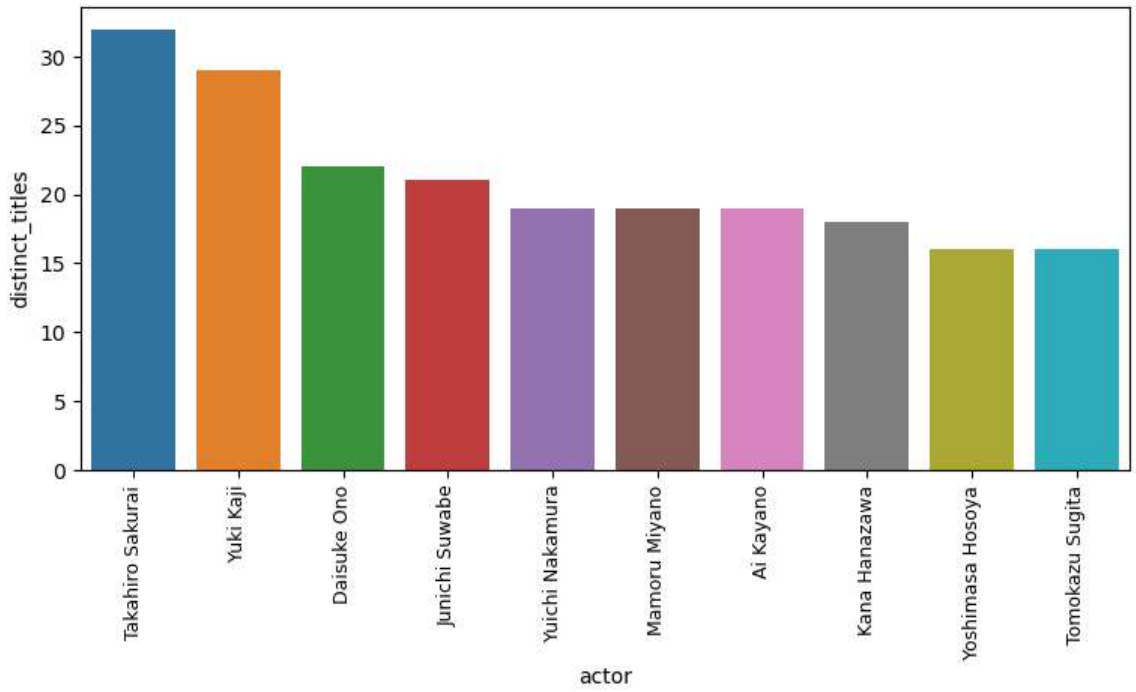


In [121]:
```
print("So, The top 5 actors based on no. of titled from UK are:")
print("")
for i in range(5):
    print(f"{atc['actor'].values[i]}, ", end="")
```

```
So, The top 5 actors based on no. of titled from UK are:

David Attenborough, John Cleese, Michael Palin, Terry Jones, Eric Idle,
```

In [ ]:

In [122]:
```
# top actors from Japan
# number of distinct titles on the basis of actors

atc = data_Japan.groupby(['actor']).agg(distinct_titles = ('title','nunique'))
atc.sort_values(by='distinct_titles', ascending = False, inplace=True)
atc.reset_index(inplace=True)

# Plotting top 10 actors (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=atc.head(10), x='actor', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```

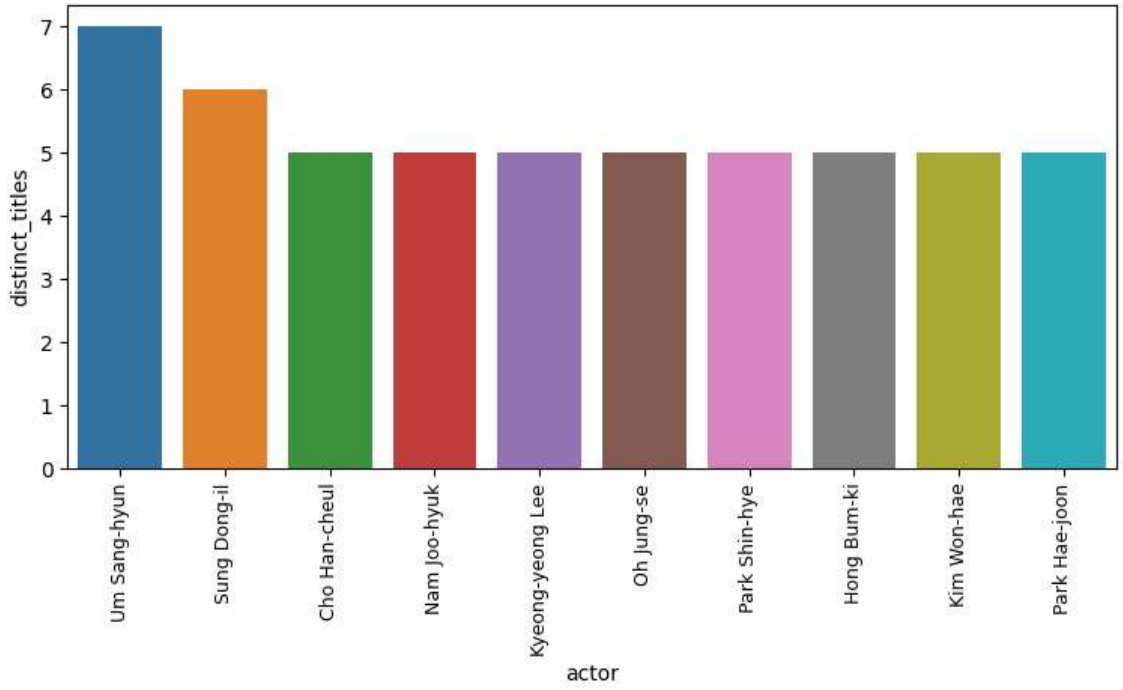


In [123]:
```
print("So, The top 5 actors based on no. of titled from US are:")
print("")
for i in range(5):
    print(f"{atc['actor'].values[i]}, ", end="")
```

```
So, The top 5 actors based on no. of titled from US are:

Takahiro Sakurai, Yuki Kaji, Daisuke Ono, Junichi Suwabe, Yuichi Nakamura,
```

In [ ]:

In [124]:
```
# top actors from South Korea
# number of distinct titles on the basis of actors

atc = data_SK.groupby(['actor']).agg(distinct_titles = ('title','nunique'))
atc.sort_values(by='distinct_titles', ascending = False, inplace=True)
atc.reset_index(inplace=True)

# Plotting top 10 actors (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=atc.head(10), x='actor', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



In [125]:
```
print("So, The top 5 actors based on no. of titled from South Korea are:")
print("")
for i in range(5):
    print(f"{atc['actor'].values[i]}, ", end="")
```

```
So, The top 5 actors based on no. of titled from South Korea are:

Um Sang-hyun, Sung Dong-il, Cho Han-cheul, Nam Joo-hyuk, Kyeong-yeong Lee,
```

In [ ]:

Which ratings are preferred in top countries?

In [128]:
```
data
```

Out[128]:

	title	actor	director	genre	country	show_id	type	date_added	release_year	rating	duration	duration_n
0	Dick Johnson Is Dead	Unknown	Kirsten Johnson	Documentaries	United States	s1	Movie	September 25, 2021	2020	PG-13	90 min	
1	Blood & Water	Ama Qamata	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	
2	Blood & Water	Ama Qamata	Unknown	TV Dramas	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	
3	Blood & Water	Ama Qamata	Unknown	TV Mysteries	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	
4	Blood & Water	Khosi Ngema	Unknown	International TV Shows	South Africa	s2	TV Show	September 24, 2021	2021	TV-MA	2 Seasons	
...	...	...	...	...	...	...	...	...	...	...	...	
201986	Zubaan	Anita Shabdish	Mozez Singh	International Movies	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	
201987	Zubaan	Anita Shabdish	Mozez Singh	Music & Musicals	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	
201988	Zubaan	Chittaranjan Tripathy	Mozez Singh	Dramas	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	
201989	Zubaan	Chittaranjan Tripathy	Mozez Singh	International Movies	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	
201990	Zubaan	Chittaranjan Tripathy	Mozez Singh	Music & Musicals	India	s8807	Movie	March 2, 2019	2015	TV-14	111 min	

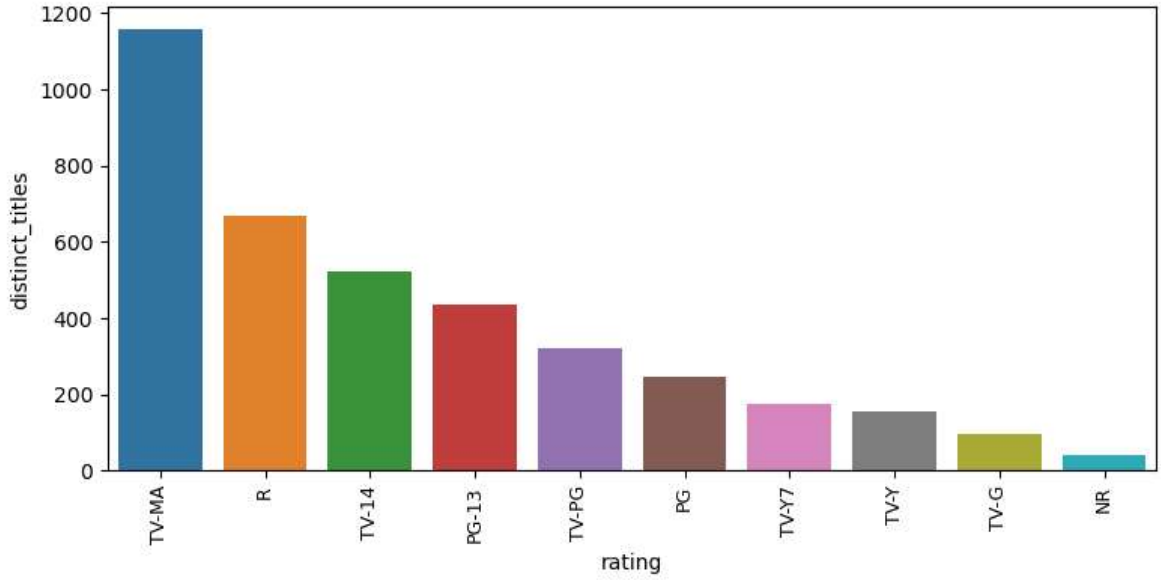
201991 rows × 16 columns

In [ ]:
```
```

In [132]:
```
# creating a country wise dataframe for US, India, UK, Japan, South Korea

data_US = data.loc[(data['country']=='United States') & (data['rating'].isnull()==False)]
data_India = data.loc[(data['country']=='India') & (data['rating'].isnull()==False)]
data_UK = data.loc[(data['country']=='United Kingdom') & (data['rating'].isnull()==False)]
data_Japan = data.loc[(data['country']=='Japan') & (data['rating'].isnull()==False)]
data_SK = data.loc[(data['country']=='South Korea') & (data['rating'].isnull()==False)]
```

In [ ]:
```
```

In [133]:
```
# top ratings from US
# number of distinct titles on the basis of ratings

rtc = data_US.groupby(['rating']).agg(distinct_titles = ('title','nunique'))
rtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
rtc.reset_index(inplace=True)

# Plotting top 10 ratings (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=rtc.head(10), x='rating', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```

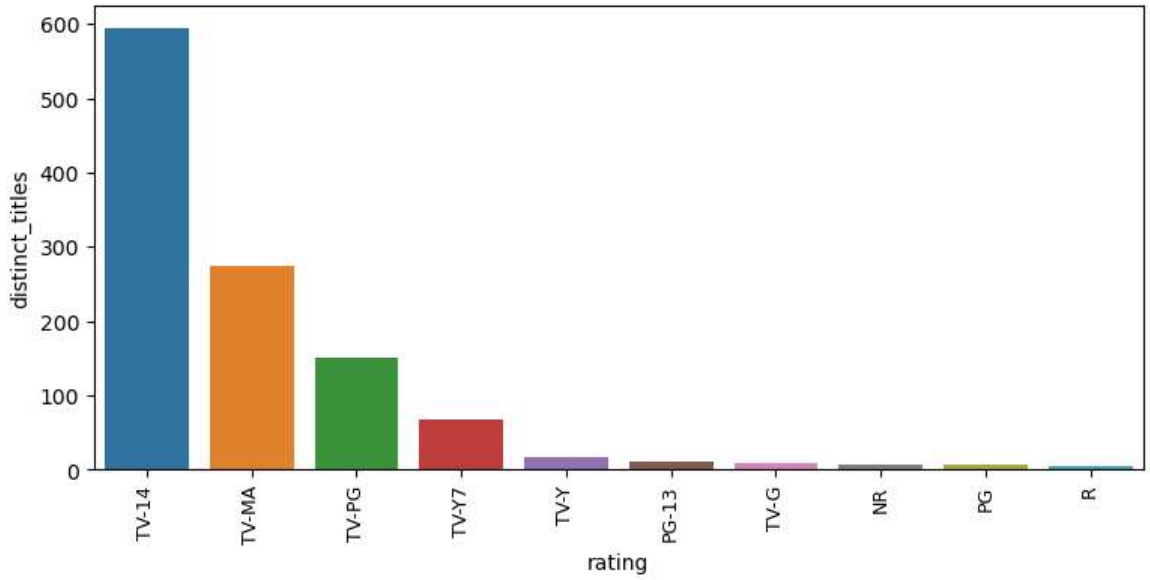


In [134]:
```
print("So, The top 5 ratings based on no. of titled from US are:")
print("")
for i in range(5):
    print(f"{rtc['rating'].values[i]}, ", end="")
```

```
So, The top 5 ratings based on no. of titled from US are:

TV-MA, R, TV-14, PG-13, TV-PG,
```

In [ ]:

In [135]:
```
# top ratings from India
# number of distinct titles on the basis of ratings

rtc = data_India.groupby(['rating']).agg(distinct_titles = ('title','nunique'))
rtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
rtc.reset_index(inplace=True)

# Plotting top 10 ratings (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=rtc.head(10), x='rating', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



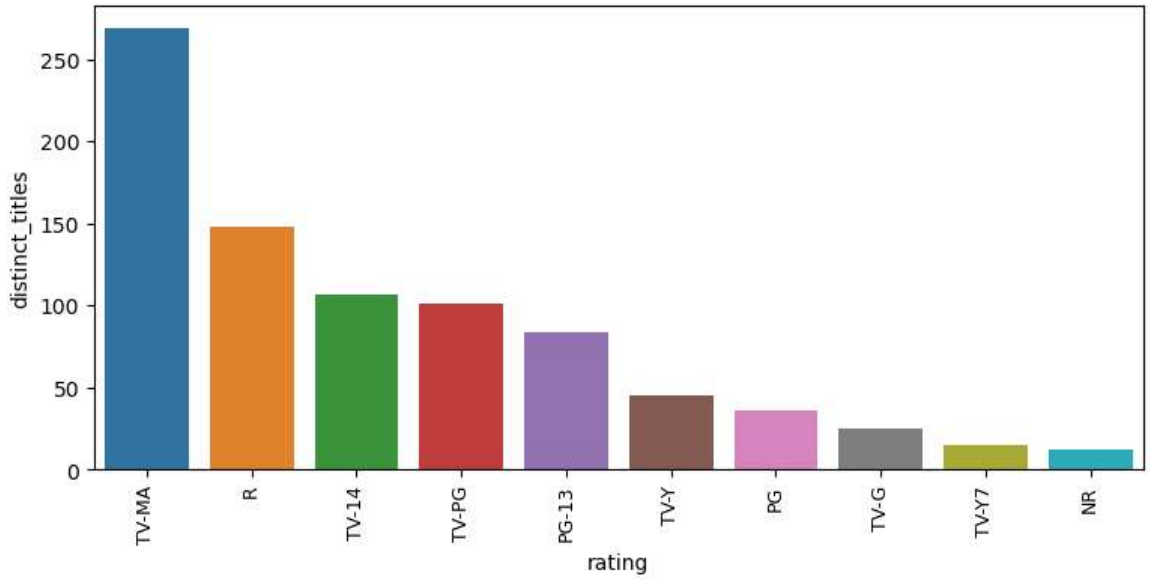
- so major chunk of the movie/shows in India are from TV-14, TV-MA, TV-PG, TV-Y7 ratings
- Also, fraction of R, PG, PG-13 rated movies in India is almost negligible
- Movie/TV show culture in India is quite conservative for ratings.

In [136]:
```
print("So, The top 5 ratings based on no. of titled from India are:")
print("")
for i in range(5):
    print(f"{rtc['rating'].values[i]}, ", end="")
```

```
So, The top 5 ratings based on no. of titled from India are:

TV-14, TV-MA, TV-PG, TV-Y7, TV-Y,
```

In [ ]:

In [137]:
```
# top ratings from UK
# number of distinct titles on the basis of ratings

rtc = data_UK.groupby(['rating']).agg(distinct_titles = ('title','nunique'))
rtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
rtc.reset_index(inplace=True)

# Plotting top 10 ratings (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=rtc.head(10), x='rating', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```

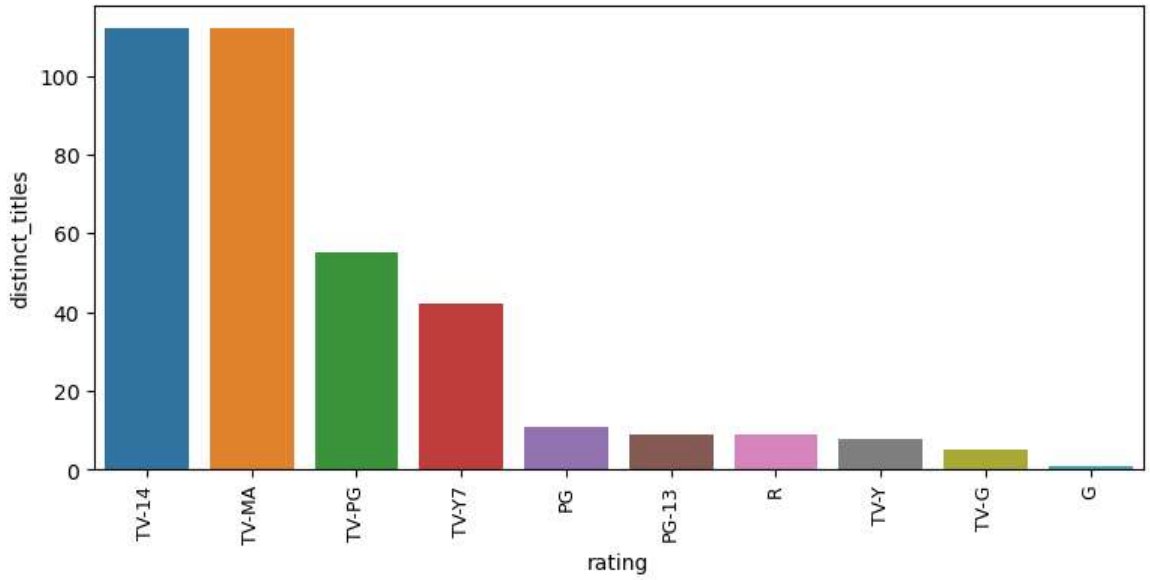


In [138]:
```
print("So, The top 5 ratings based on no. of titled from UK are:")
print("")
for i in range(5):
    print(f"{rtc['rating'].values[i]}, ", end="")
```

```
So, The top 5 ratings based on no. of titled from UK are:

TV-MA, R, TV-14, TV-PG, PG-13,
```

In [ ]:

In [139]:
```
# top ratings from Japan
# number of distinct titles on the basis of ratings

rtc = data_Japan.groupby(['rating']).agg(distinct_titles = ('title','nunique'))
rtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
rtc.reset_index(inplace=True)

# Plotting top 10 ratings (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=rtc.head(10), x='rating', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```

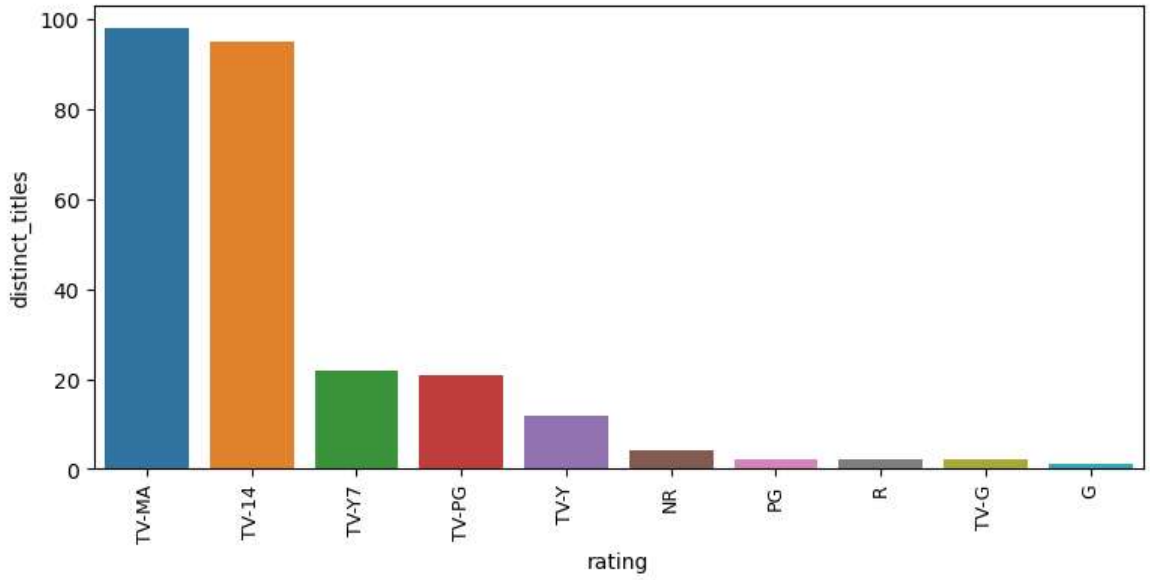


In [140]:
```
print("So, The top 5 ratings based on no. of titled from Japan are:")
print("")
for i in range(5):
    print(f"{rtc['rating'].values[i]}, ", end="")
```

```
So, The top 5 ratings based on no. of titled from Japan are:

TV-14, TV-MA, TV-PG, TV-Y7, PG,
```

In [ ]:

In [141]:
```
# top ratings from South Korea
# number of distinct titles on the basis of ratings

rtc = data_SK.groupby(['rating']).agg(distinct_titles = ('title','nunique'))
rtc.sort_values(by='distinct_titles', ascending = False, inplace=True)
rtc.reset_index(inplace=True)

# Plotting top 10 ratings (based on distinct titles)
plt.figure(figsize=(9, 4))
sns.barplot(data=rtc.head(10), x='rating', y='distinct_titles', estimator=np.sum)
plt.xticks(rotation=90, fontsize=9)
plt.show()
```



In [142]:
```
print("So, The top 5 ratings based on no. of titled from South Korea are:")
print("")
for i in range(5):
    print(f"{rtc['rating'].values[i]}, ", end="")
```

```
So, The top 5 ratings based on no. of titled from South Korea are:

TV-MA, TV-14, TV-Y7, TV-PG, TV-Y,
```

In [ ]:

So from above graphs, we can conclude that:

- US, UK and Japan, South Korea has quite similar distribution of ratings.
- Movie/TV show culture in India, Japan, South Korea is quite conservative for ratings.

In [ ]:

Which is popular director-actor combination in top countries?

In [155]:
```
# creating a country wise dataframe for US, India, UK, Japan, South Korea

data_US = data.loc[(data['country']=='United States') & (data['actor']!='Unknown') & (data['director']!='Unkr
data_India = data.loc[(data['country']=='India') & (data['actor']!='Unknown') & (data['director']!='Unknown')
data_UK = data.loc[(data['country']=='United Kingdom') & (data['actor']!='Unknown') & (data['director']!='Unk
data_Japan = data.loc[(data['country']=='Japan') & (data['actor']!='Unknown') & (data['director']!='Unknown')
data_SK = data.loc[(data['country']=='South Korea') & (data['actor']!='Unknown') & (data['director']!='Unknow
```

In [ ]:

In [161]:
```
# no. of distinct titles based on actor-director pair for US

adt = data_US.groupby(['actor', 'director']).agg(distinct_titles = ('title','nunique'))
adt.sort_values(by='distinct_titles', ascending = False, inplace=True)
adt.reset_index(inplace=True)

adt.head(15)
```

Out[161]:

	actor	director	distinct_titles
0	Kerry Gudjohnsen	Alex Woo	5
1	Paul Killam	Stanley Moore	5
2	Tabitha St. Germain	Ishi Rudell	5
3	Andrea Libman	Ishi Rudell	5
4	Maisie Benson	Alex Woo	5
5	Ashleigh Ball	Ishi Rudell	5
6	Maisie Benson	Stanley Moore	5
7	Alexa PenaVega	Robert Rodriguez	5
8	Samuel L. Jackson	Quentin Tarantino	5
9	Tara Strong	Ishi Rudell	5
10	Rebecca Shoichet	Ishi Rudell	5

- US has many actor-director combination with five distinct titles together

In [ ]:

In [160]:
```
# no. of distinct titles based on actor-director pair for India

adt = data_India.groupby(['actor', 'director']).agg(distinct_titles = ('title','nunique'))
adt.sort_values(by='distinct_titles', ascending = False, inplace=True)
adt.reset_index(inplace=True)

adt.head(15)
```

Out[160]:

	actor	director	distinct_titles
0	Julie Tejwani	Rajiv Chilaka	19
1	Rajesh Kava	Rajiv Chilaka	19
2	Jigna Bhardwaj	Rajiv Chilaka	18
3	Rupa Bhimani	Rajiv Chilaka	18
4	Vatsal Dubey	Rajiv Chilaka	16
5	Swapnil	Rajiv Chilaka	13
6	Mousam	Rajiv Chilaka	13
7	Saurav Chakraborty	Suhas Kadav	8
8	Anupam Kher	David Dhawan	6
9	Smita Malhotra	Tilak Shetty	6
10	Alok Nath	Sooraj R. Barjatya	5

- So In Indian film industry, Julie Tejwani-Rajiv Chilaka & Rajesh Kava-Rajiv Chilaka has done maximum Films/TV shows together with 19 distinct titles

In [ ]:

In [162]:
```
# no. of distinct titles based on actor-director pair for UK

adt = data_UK.groupby(['actor', 'director']).agg(distinct_titles = ('title','nunique'))
adt.sort_values(by='distinct_titles', ascending = False, inplace=True)
adt.reset_index(inplace=True)

adt.head(15)
```

Out[162]:

	actor	director	distinct_titles
0	Keith Wickham	Joey So	5
1	David Attenborough	Alastair Fothergill	4
2	Rob Rackstraw	Joey So	4
3	Jo Wyatt	Blair Simmons	3
4	Michael Murphy	Blair Simmons	3
5	Rachael Stirling	Edward Cotterill	3
6	Simon Greenall	Blair Simmons	3
7	Joey King	Vince Marcello	3
8	Paul Panting	Blair Simmons	3
9	Joel Courtney	Vince Marcello	3
10	Teresa Gallagher	Blair Simmons	3

- So In UK film industry, Keith Wickham-Joey So has done maximum Films/TV shows together with 5 distinct titles

In [ ]:

In [163]:
```
# no. of distinct titles based on actor-director pair for Japan

adt = data_Japan.groupby(['actor', 'director']).agg(distinct_titles = ('title','nunique'))
adt.sort_values(by='distinct_titles', ascending = False, inplace=True)
adt.reset_index(inplace=True)

adt.head(15)
```

Out[163]:

	actor	director	distinct_titles
0	Satsuki Yukino	Toshiya Shinohara	7
1	Kumiko Watanabe	Toshiya Shinohara	7
2	Koji Tsujitani	Toshiya Shinohara	7
3	Houko Kuwashima	Toshiya Shinohara	7
4	Kappei Yamaguchi	Toshiya Shinohara	7
5	Noriko Hidaka	Toshiya Shinohara	5
6	Ken Narita	Toshiya Shinohara	5
7	Yuki Yamada	Shigeaki Kubo	5
8	Takahiro	Shigeaki Kubo	4
9	Kento Hayashi	Shigeaki Kubo	4
10	Junko Takeuchi	Masahiko Murata	4
11	Sho Aoyagi	Shigeaki Kubo	4
12	Akira	Shigeaki Kubo	4
13	Shuichi Ikeda	Yoshiyuki Tomino	4
14	Hiroomi Tosaka	Shigeaki Kubo	4

In [ ]:

In [ ]:

In [164]:
```
# no. of distinct titles based on actor-director pair for South Korea

adt = data_SK.groupby(['actor', 'director']).agg(distinct_titles = ('title','nunique'))
adt.sort_values(by='distinct_titles', ascending = False, inplace=True)
adt.reset_index(inplace=True)

adt.head(15)
```

Out[164]:

	actor	director	distinct_titles
0	Gregg Berger	Mark A.Z. Dippé	2
1	Frank Welker	Mark A.Z. Dippé	2
2	Fred Tatasciore	Mark A.Z. Dippé	2
3	Jennifer Darling	Mark A.Z. Dippé	2
4	Neil Ross	Mark A.Z. Dippé	2
5	Keith Silverstein	Thomas Astruc	2
6	Sung Dong-il	Shin Won-ho	2
7	Wally Wingert	Mark A.Z. Dippé	2
8	Bryce Papenbrook	Thomas Astruc	2
9	Audrey Wasilewski	Mark A.Z. Dippé	2
10	Selah Victor	Thomas Astruc	2
11	Mela Lee	Thomas Astruc	2
12	Stephen Stanton	Mark A.Z. Dippé	2
13	Grant George	Thomas Astruc	2
14	Cristina Vee	Thomas Astruc	2

In [ ]:

In [ ]:

Key Insights and Recommendations

1. We have around 70% movies and around 30% TV shows on Netflix. Netflix need to calibrate this proportion based on how users are consuming Movies/TV Shows.
2. International Movies is the top genre followed by Dramas, Comedies so content falling in that bracket is recommended.
3. US has produced most number of titles, followed by India, UK, Canada, France, Japan, etc
4. Overall Mean of duration is around 107 min and median is 104 min.
5. Among all the countries, US content duration has many outliers and so is least consistent in show durations, It is recommended to avoid this polarity in content duration.
6. India has much higher mean duration (123 min) compared to other countries. Recommended to check if higher show duration is affecting the viewership and take corrective actions if needed.
7. Rajiv Chilaka, Jan Suter and Raul Campos are the most popular directors across Netflix
8. Number of titles added on Netflix took a great pace from 2016 and was on rise till 2019, from 2019-2021 the number of titles added has declined, this might be due to the corona outbreak.
9. Least titles are added in February month. Most titles are added in July, December. It might be attributed to summer break in July and X-Mas, corporate holidays in December. Recommended to add High Budget/High Profile movies in July, Late December and Early January
10. Fraction of TV shows in India is significantly lower than other countries. Netflix needs to gauge feasible length of the Show and ramp up the TV show production in India
11. Japan and South Korea has more TV shows than Movies
12. US-UK and Japan-South Korea has quite similar distribution of ratings.
13. Fraction of R, PG, PG-13 rated contents in India is almost negligible. Movie/TV show culture in India, Japan, South Korea is quite conservative for ratings. Netflix India should try to bring more of R, PG, PG-13 rated contents, it can attract new users.
14. While creating content, it is recommended to capitalize on famous actor-director combination of respective country

In [ ]:

In [ ]:

In [ ]: