```
from google.colab import files
uploaded = files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving netflix.csv to netflix (1).csv
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt


# Load the data
df = pd.read_csv('netflix.csv')

# Let's look at the sample rows
df.sample(5)
```

```
{"repr_error":"0","type":"dataframe"}
```

```
# 1. No. of rows and columns
print(df.shape)

# 2. Basic Info: Column names, column data type, number of non-null values
print(df.info())
```

```
(8807, 12)
<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
None
```

```
# 3. Percentage of null values
df.isnull().sum() * 100 / len(df)
```

```
show_id          0.000000
type             0.000000
title            0.000000
director        29.908028
cast             9.367549
country          9.435676
date_added       0.113546
release_year     0.000000
rating           0.045418
duration         0.034064
listed_in        0.000000
description      0.000000
dtype: float64
```

```
df['type'].value_counts()
```

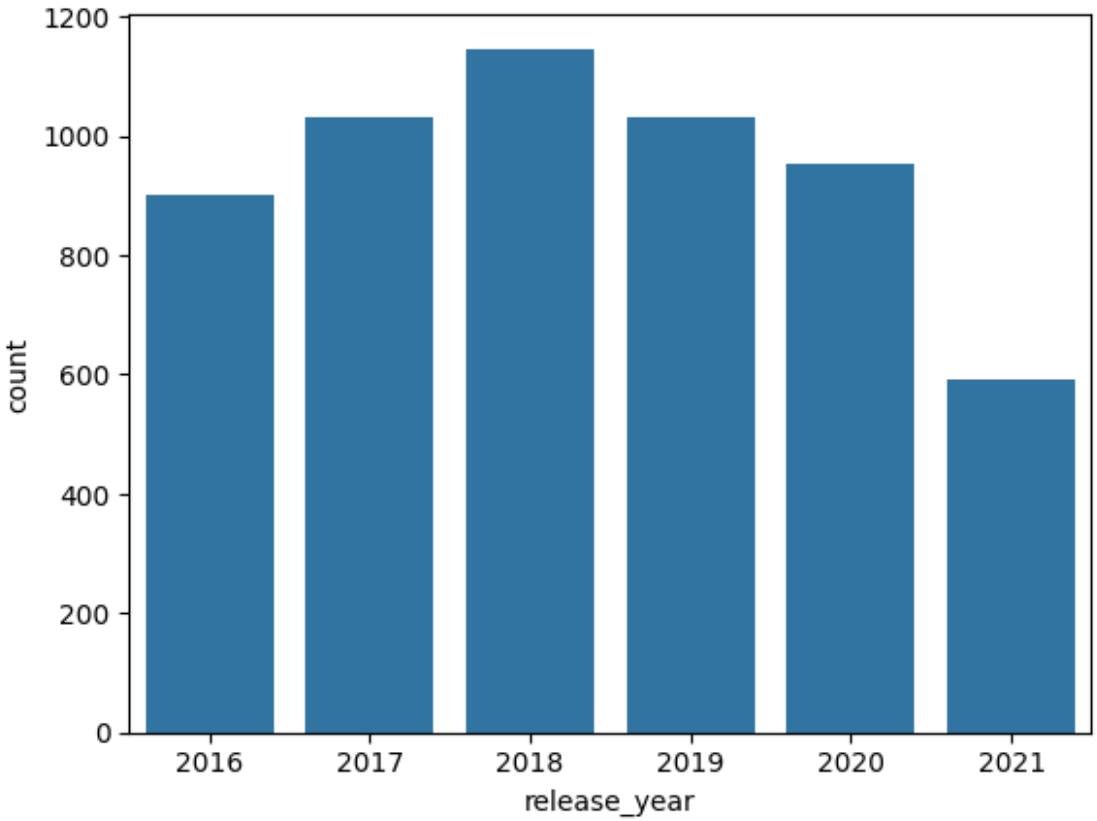
```
type
Movie      6131
TV Show    2676
Name: count, dtype: int64
```

#Insights:

```
df['listed_in'] = df['listed_in'].astype(str).apply(lambda x:
x.split(','))

df.explode('listed_in')['listed_in'].value_counts()
```

```
listed_in
 ' " \' International Movies\']"]']    1786
['["[\'Dramas\'"'                      1462
['["[\'Comedies\'"'                    1100
 ' " \' International Movies\'"'        838
['["[\'International TV Shows\'"'       772
                                       ...
['["[\'TV Action & Adventure\']"]']       1
['["[\'Sports Movies\']"]']               1
['["[\'Sci-Fi & Fantasy\']"]']            1
['["[\'Independent Movies\']"]']          1
['["[\'Anime Features\']"]']              1
Name: count, Length: 124, dtype: int64
```

```
'country', 'TV Shows', 'Movies'
'US', 12, 25

('US', 12, 25)

import seaborn as sns # Univariate
sns.countplot(data=df[df['release_year'] > 2015],x='release_year')

<Axes: xlabel='release_year', ylabel='count'>
```




```
# Movie and TV Show
df_movie = df[df['type'] == 'Movie']
df_movie.head()
```


```
{"summary":"{\n  \"name\": \"df_movie\",\n  \"rows\": 6131,\n
\"fields\": [\n    {\n      \"column\": \"show_id\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 6131,\n        \"samples\": [\n
\"s6904\",\n          \"s2720\",\n          \"s7150\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"type\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 1,\n
\"samples\": [\n          \"Movie\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"title\",\n      \"properties\": {\
n        \"dtype\": \"string\",\n        \"num_unique_values\": 6131,\
n        \"samples\": [\n          \"H\\u00e9roes\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"director\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\":
4354,\n        \"samples\": [\n          \"Dean Parisot\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"cast\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\":
```

```
5445,\n        \"samples\": [\n          \"Sunny Pawar, Chandan Roy
Sanyal, Gautam Sarkar, Sumeet Thakur, Mala Mukherjee, Masood Akhtar,
Veer Rajwant Singh, Joyraj Bhattacharya\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"country\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
651,\n        \"samples\": [\n          \"Canada, United States,
India, United Kingdom\"\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"date_added\",\n      \"properties\": {\n
\"dtype\": \"object\",\n        \"num_unique_values\": 1533,\n
\"samples\": [\n          \"May 2, 2018\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"release_year\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
9,\n        \"min\": 1942,\n        \"max\": 2021,\n
\"num_unique_values\": 73,\n        \"samples\": [\n          1998\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"rating\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
17,\n        \"samples\": [\n          \"PG-13\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"duration\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
205,\n        \"samples\": [\n          \"110 min\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"listed_in\",\n
\"properties\": {\n        \"dtype\": \"object\",\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"description\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 6105,\n        \"samples\": [\n
\"Seemingly simple but deceptively complex, the game of \\\"Go\\\"
serves as the backdrop for this battle between artificial intelligence
and man.\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df_movie"}
```


```
df_movie['duration_int'] = df_movie['duration'].dropna().apply(lambda
x : x[ : -4]).astype('int')
```

```
<ipython-input-75-603805007d4a>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df_movie['duration_int'] =
df_movie['duration'].dropna().apply(lambda x : x[ : -4]).astype('int')
```

```
df_movie.head()
```


```
{"summary":"{\n  \"name\": \"df_movie\",\n  \"rows\": 6131,\n
\"fields\": [\n    {\n      \"column\": \"show_id\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 6131,\n        \"samples\": [\n
\"s6904\",\n          \"s2720\",\n          \"s7150\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"type\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 1,\n
\"samples\": [\n          \"Movie\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"title\",\n      \"properties\": {\
n        \"dtype\": \"string\",\n        \"num_unique_values\": 6131,\
n        \"samples\": [\n          \"H\\u00e9roes\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"director\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\":
4354,\n        \"samples\": [\n          \"Dean Parisot\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"cast\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\":
5445,\n        \"samples\": [\n          \"Sunny Pawar, Chandan Roy
Sanyal, Gautam Sarkar, Sumeet Thakur, Mala Mukherjee, Masood Akhtar,
Veer Rajwant Singh, Joyraj Bhattacharya\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"country\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
651,\n        \"samples\": [\n          \"Canada, United States,
India, United Kingdom\"\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"date_added\",\n      \"properties\": {\n
\"dtype\": \"object\",\n        \"num_unique_values\": 1533,\n
\"samples\": [\n          \"May 2, 2018\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"release_year\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
9,\n        \"min\": 1942,\n        \"max\": 2021,\n
\"num_unique_values\": 73,\n        \"samples\": [\n          1998\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"rating\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
17,\n        \"samples\": [\n          \"PG-13\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"duration\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
205,\n        \"samples\": [\n          \"110 min\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"listed_in\",\n
\"properties\": {\n        \"dtype\": \"object\",\n
```

\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"description\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 6105,\n   \"samples\": [\n
\"Seemingly simple but deceptively complex, the game of \\\"Go\\\"
serves as the backdrop for this battle between artificial intelligence
and man.\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"duration_int\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 28.290593447417347,\n   \"min\":
3.0,\n   \"max\": 312.0,\n   \"num_unique_values\": 205,\n
\"samples\": [\n   110.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"df_movie"}

```
df_movie.groupby('country')['duration_int'].describe()
```

{"summary":"{\n \"name\": \"df_movie\",\n \"rows\": 651,\n
\"fields\": [\n   {\n   \"column\": \"country\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 651,\n   \"samples\": [\n
\"United States, Uruguay\",\n   \"Israel, Sweden, Germany,
Netherlands\",\n   \"Ireland, United States, France\"\
n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"count\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 88.72722314833469,\n   \"min\": 1.0,\n   \"max\":
2055.0,\n   \"num_unique_values\": 41,\n   \"samples\": [\n
77.0,\n   28.0,\n   50.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"mean\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 21.14619703401173,\n
\"min\": 13.0,\n   \"max\": 208.0,\n
\"num_unique_values\": 205,\n   \"samples\": [\n   72.0,\n
113.0,\n   103.6\n   ],\n   \"semantic_type\": \"\",\
n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"std\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 13.956721762144653,\n   \"min\":
0.0,\n   \"max\": 90.50966799187809,\n
\"num_unique_values\": 135,\n   \"samples\": [\n
2.8284271247461903,\n   24.99599967994879,\n
32.526911934581186\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"min\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 26.327043373852234,\n   \"min\": 3.0,\n   \"max\":
208.0,\n   \"num_unique_values\": 123,\n   \"samples\": [\n
95.0,\n   106.0,\n   99.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"25%\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 21.48059372069142,\n

\"min\": 13.0,\n   \"max\": 208.0,\n
\"num_unique_values\": 171,\n   \"samples\": [\n
109.75,\n   88.0,\n   102.75\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"50%\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 21.102319504159496,\n
\"min\": 13.0,\n   \"max\": 208.0,\n
\"num_unique_values\": 136,\n   \"samples\": [\n   95.5,\n
94.5,\n   114.0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"75%\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 21.809913290428927,\n   \"min\": 13.0,\n   \"max\":
208.0,\n   \"num_unique_values\": 167,\n   \"samples\": [\n
53.0,\n   29.0,\n   24.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"max\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 27.744401059409075,\n
\"min\": 13.0,\n   \"max\": 312.0,\n
\"num_unique_values\": 123,\n   \"samples\": [\n   95.0,\n
86.0,\n   104.0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe"}

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

```
array(['United States', 'South Africa', nan, 'India',
       'United States, Ghana, Burkina Faso, United Kingdom, Germany,
Ethiopia',
       'United Kingdom', 'Germany, Czech Republic', 'Mexico',
'Turkey',
       '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
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',
       '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
States',
       '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 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', 'Malaysia',
       'Israel', 'Australia, New Zealand', 'United Kingdom, Germany',
       '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', 'Philippines',
       'Iceland', 'Denmark', '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 States, France', 'United States, Bulgaria',
       'United Kingdom, France, Germany, United States',
       'Norway, Denmark', '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 States', 'United Kingdom, United States,
France',
       'United Kingdom,', 'United States, India, United Kingdom',
'Kenya',
       'Spain, Argentina', 'India, United Kingdom, France, Qatar',
       'Belgium, France', 'Argentina, Chile', 'United States,
Thailand',
       'Chile, Brazil', 'United States, Colombia',
       'Canada, United States, United Kingdom', 'Uruguay',
'Luxembourg',
       'United States, Cambodia, Romania', 'Bangladesh',
       'Spain, Belgium, United States',
       'United Kingdom, United States, Australia',
       'Canada, United States, France', 'Portugal, United States',
       'Portugal, Spain', 'India, United States',
       'United Kingdom, Ireland', 'United Kingdom, Spain, United
States',
       'Hungary, United States', 'United States, South Korea',
       'Canada, United States, Cayman Islands', 'India, France',
       'France, Canada', 'Canada, Hungary, United States', 'Norway',
       'Canada, United Kingdom, United States',
       'United Kingdom, Germany, France, United States',
       'Denmark, United States', 'Senegal', 'France, Algeria',
       'United Kingdom, Finland, Germany, United States, Australia,
Japan, France, Ireland',
       'Philippines, Canada, United Kingdom, United States',
       'Ireland, France, Iceland, United States, Mexico, Belgium,
United Kingdom, Hong Kong',
       'Singapore', 'Kuwait', 'United States, France, Serbia',
       'United States, Italy', 'Spain, Italy',
       'United States, Ireland, United Kingdom, India',
       'United Kingdom, Singapore', 'Hong Kong, United States',
       'United States, Malta, France, United Kingdom',
       'United States, China, Canada', 'Canada, United States,
```

```
Ireland',
       'Lebanon, Canada, France', 'Japan, Canada, United States',
       'Spain, France, Canada',
       'Denmark, Singapore, Canada, United States',
       'United States, France, Denmark', 'United States, China,
Colombia',
       'Spain, Thailand, United States', 'Mexico, Spain',
       'Ireland, Luxembourg, Belgium', 'China, United States',
       'Canada, Belgium', 'Canada, United Kingdom',
       'Lebanon, United Arab Emirates, France, Switzerland, Germany',
       'France, Belgium, Italy',
       'Lebanon, United States, United Arab Emirates', 'Lebanon,
France',
       'France, Lebanon', 'France, Lebanon, United Kingdom',
       'France, Norway, Lebanon, Belgium',
       'Sweden, Czech Republic, United Kingdom, Denmark, Netherlands',
       'United States, United Kingdom, India', 'Indonesia,
Netherlands',
       'Turkey, South Korea', 'Serbia, United States', 'Namibia',
       'United Kingdom, Kenya', 'United Kingdom, France, Germany,
Spain',
       'United Kingdom, France, United States, Belgium, Luxembourg,
China, Germany',
       'Thailand, United States',
       'United States, France, Canada, Belgium', 'United Kingdom,
China',
       'Germany, China, United Kingdom',
       'Australia, New Zealand, United States',
       'Hong Kong, Iceland, United States', 'France, Australia,
Germany',
       'United States, Belgium, Canada, France', 'South Africa,
Angola',
       'United States, Philippines',
       'United States, United Kingdom, Canada, China',
       'United States, Canada, United Kingdom', 'Turkey, United
States',
       'Peru, Germany, Norway', 'Mozambique', 'Brazil, France',
       'China, Spain, South Korea, United States', 'Spain, Germany',
       'Hong Kong, China', 'France, Belgium, Luxembourg, Cambodia,',
       'United Kingdom, Australia', 'Belarus',
       'Indonesia, United Kingdom',
       'Switzerland, France, Belgium, United States', 'Ghana',
       'Spain, France, Canada, United States', 'Chile, Italy',
       'United Kingdom, Nigeria', 'Chile', 'France, Egypt',
       'Egypt, France', 'France, Brazil, Spain, Belgium',
       'Egypt, Algeria', 'Canada, South Korea, United States',
       'Nigeria, United Kingdom', 'United States, France, Canada',
       'Poland, United States',
       'United Arab Emirates, Jordan, Lebanon, Saudi Arabia',
```

```
       'United States, Mexico, Spain, Malta',
       'Saudi Arabia, United Arab Emirates', 'Zimbabwe',
       'United Kingdom, Germany, United Arab Emirates, New Zealand',
       'Romania, United States', 'Canada, Nigeria',
       'Saudi Arabia, Netherlands, Germany, Jordan, United Arab
Emirates, United States',
       'United Kingdom, Spain', 'Finland, France',
       'United Kingdom, Germany, United States, France',
       'India, United Kingdom, China, Canada, Japan, South Korea,
United States',
       'Italy, United Kingdom, France', 'United States, Mexico,
Colombia',
       'Turkey, India', 'Italy, Turkey',
       'United Kingdom, United States, Japan',
       'France, Belgium, United States',
       'Puerto Rico, United States, Colombia', 'Uruguay, Argentina',
       'United States, United Kingdom, Japan', 'United States,
Argentina',
       'United Kingdom, Italy', 'Ireland, United Kingdom',
       'United Kingdom, France, Belgium, Canada, United States',
       'Netherlands, Germany, Denmark, United Kingdom', 'Hungary',
       'Austria, Germany', 'Taiwan, China',
       'United Kingdom, United States, Ireland',
       'South Korea, United States', 'Brazil, United Kingdom',
       'Pakistan, United States', 'Romania, France, Switzerland,
Germany',
       'Romania, United Kingdom', 'France, Malta, United States',
       'Cyprus',
       'United Kingdom, France, Belgium, Ireland, United States',
       'United States, Norway, Canada', 'Kenya, United States',
       'France, South Korea, Japan, United States', 'Taiwan,
Malaysia',
       'Uruguay, Argentina, Germany, Spain',
       'United States, United Kingdom, France, Germany, Japan',
       'United States, France, Japan',
       'United Kingdom, France, United States',
       'Spain, France, United States',
       'Indonesia, South Korea, Singapore', 'United States, Spain',
       'Netherlands, Germany, Italy, Canada',
       'Spain, Germany, Denmark, United States', 'Norway, Sweden',
       'South Korea, Canada, United States, China',
       'Argentina, Uruguay, Serbia', 'France, Japan',
       'Mauritius, South Africa', 'United States, Poland',
       'United Kingdom, United States, Germany, Denmark, Belgium,
Japan',
       'India, Germany', 'India, United Kingdom, Canada, United
States',
       'Philippines, United States', 'Romania, Bulgaria, Hungary',
       'Uruguay, Guatemala', 'France, Senegal, Belgium',
```

```
       'United Kingdom, Canada', 'Mexico, United States, Spain,
Colombia',
       'Canada, Norway', 'Singapore, United States',
       'Finland, Germany, Belgium', 'United Kingdom, France',
       'United States, Chile', 'United Kingdom, Japan, United States',
       'Spain, United Kingdom', 'Argentina, United States, Mexico',
       'United States, South Korea, Japan', 'Canada, Australia',
       'United Kingdom, Hungary, Australia', 'Italy, Belgium',
       'United States, United Kingdom, Germany', 'Switzerland',
       'Singapore, Malaysia',
       'France, Belgium, Luxembourg, Romania, Canada, United States',
       'South Africa, Nigeria', 'Spain, France',
       'United Kingdom, Hong Kong', 'Pakistan', 'Brazil, United
States',
       'Denmark, Brazil, France, Portugal, Sweden', 'India, Turkey',
       'Malaysia, Singapore, Hong Kong', 'Philippines, Singapore',
       'Australia, Canada', 'Taiwan, China, France, United States',
       'Germany, Italy', 'Colombia, Peru, United Kingdom',
       'Thailand, China, United States', 'Argentina, United States',
       'Sweden, United States', 'Uruguay, Spain, Mexico',
       'France, Luxembourg, Canada', 'Denmark, Spain', 'Chile,
Argentina',
       'United Kingdom, Belgium, Sweden', 'Canada, Brazil',
       'Italy, France', 'Canada, Germany',
       'Pakistan, United Arab Emirates', 'Ghana, United States',
       'Mexico, Finland', 'United Arab Emirates, United Kingdom,
India',
       'Netherlands, Belgium', 'United States, Taiwan',
       'Austria, Iraq, United States', 'United Kingdom, Malawi',
       'Paraguay, Argentina', 'United Kingdom, Russia, United States',
       'India, Pakistan', 'Indonesia, Singapore', 'Spain, Belgium',
       'Iceland, Sweden, Belgium', 'Croatia', 'Uruguay, Argentina,
Spain',
       'United Kingdom, Ireland, United States',
       'Canada, Germany, France, United States', 'United Kingdom,
Japan',
       'Norway, Denmark, Netherlands, Sweden',
       'Hong Kong, China, United States', 'Ireland, Canada',
       'Italy, Switzerland, France, Germany', 'Mexico, Netherlands',
       'United States, Sweden', 'Germany, France, Russia',
       'France, Iran, United States', 'United Kingdom, India',
       'Russia, Poland, Serbia', 'Spain, Portugal', 'Peru',
       'Mexico, Argentina',
       'United Kingdom, Canada, United States, Cayman Islands',
       'Indonesia, United States',
       'United States, Israel, United Kingdom, Canada',
       'Norway, Iceland, United States', 'Czech Republic, United
States',
       'United Kingdom, India, United States',
```

```
       'United Kingdom, West Germany', 'India, Australia',
       'United States,', 'Belgium, United Kingdom, United States',
       'India, Germany, Austria',
       'United States, Brazil, South Korea, Mexico, Japan, Germany',
       'Spain, Mexico', 'China, Japan', 'Argentina, France',
       'China, United States, United Kingdom',
       'France, Luxembourg, United States',
       'China, United States, Australia', 'Colombia, Mexico',
       'United States, Canada, Ireland', 'Chile, Peru',
       'Argentina, Italy', 'Canada, Japan, United States',
       'United Kingdom, Canada, United States, Germany',
       'Italy, Switzerland, Albania, Poland',
       'United States, Japan, Canada', 'Cambodia',
       'Italy, United States, Argentina',
       'Saudi Arabia, Syria, Egypt, Lebanon, Kuwait',
       'United States, Canada, Indonesia, United Kingdom, China,
Singapore',
       'Spain, Colombia',
       'United Kingdom, South Africa, Australia, United States',
       'Bulgaria', 'Argentina, Brazil, France, Poland, Germany,
Denmark',
       'United Kingdom, Spain, United States, Germany',
       'Philippines, Qatar', 'Netherlands, Belgium, Germany, Jordan',
       'United Arab Emirates, United States', 'Norway, Germany,
Sweden',
       'South Korea, China', 'Georgia', 'Soviet Union, India',
       'Australia, United Arab Emirates', 'Canada, Germany, South
Africa',
       'South Korea, China, United States', 'India, Soviet Union',
       'India, Mexico', 'Georgia, Germany, France',
       'United Arab Emirates, Romania', 'India, Malaysia',
       'Germany, Jordan, Netherlands', 'Turkey, France, Germany,
Poland',
       'Greece, United States', 'France, United Kingdom, United
States',
       'Norway, Germany', 'France, Morocco', 'Cambodia, United
States',
       'United States, Denmark', 'United States, Colombia, Mexico',
       'United Kingdom, Italy, Israel, Peru, United States',
       'Argentina, Uruguay, Spain, France',
       'United Kingdom, France, United States, Belgium',
       'France, Canada, China, Cambodia',
       'United Kingdom, France, Belgium, United States', 'Chile,
France',
       'Netherlands, United States', 'France, United Kingdom, India',
       'Czech Republic, Slovakia', 'Singapore, France',
       'Spain, Switzerland', 'United States, Australia, China',
       'South Africa, United States, Germany',
       'United States, United Kingdom, Australia',
```

```
       'Spain, Italy, Argentina', 'Chile, Spain, Argentina, Germany',
       'West Germany', 'Austria, Czech Republic', 'Lebanon, Qatar',
       'United Kingdom, Jordan, Qatar, Iran',
       'France, South Korea, Japan', 'Israel, Germany, France',
       'Canada, Japan, Netherlands', 'United States, Hungary',
       'France, Germany', 'France, Qatar',
       'United Kingdom, Germany, Canada', 'Ireland, South Africa',
       'Chile, United States, France', 'Belgium, France, Netherlands',
       'United Kingdom, Ukraine, United States',
       'Germany, Australia, France, China', 'Norway, United States',
       'United States, Bermuda, Ecuador',
       'United States, Hungary, Ireland, Canada',
       'United Kingdom, Egypt, United States',
       'United States, France, United Kingdom', 'Spain, Mexico,
France',
       'United States, South Africa', 'Hong Kong, China, Singapore',
       'South Africa, China, United States', 'Denmark, France,
Poland',
       'New Zealand, United Kingdom',
       'Netherlands, Denmark, South Africa', 'Iran, France',
       'United Kingdom, United States, France, Germany',
       'Australia, France', 'Ireland, United Kingdom, United States',
       'United Kingdom, France, Germany', 'Canada, Luxembourg',
       'Brazil, Netherlands, United States, Colombia, Austria,
Germany',
       'France, Canada, Belgium', 'Canada, France',
       'Bulgaria, United States, Spain, Canada', 'Sweden,
Netherlands',
       'France, United States, Mexico',
       'Australia, United Kingdom, United Arab Emirates, Canada',
       'Australia, Armenia, Japan, Jordan, Mexico, Mongolia, New
Zealand, Philippines, South Africa, Sweden, United States, Uruguay',
       'India, Iran', 'France, Belgium, Spain',
       'Denmark, Sweden, Israel, United States', 'United States,
Iceland',
       'United Kingdom, Russia',
       'United States, Israel, Italy, South Africa',
       'Netherlands, Denmark, France, Germany', 'South Korea, Japan',
       'United Kingdom, Pakistan', 'France, New Zealand',
       'United Kingdom, Czech Republic, United States, Germany,
Bahamas',
       'China, Germany, India, United States', 'Germany, Sri Lanka',
       'United States, India, Bangladesh',
       'United States, Canada, France', 'Brazil, France, Germany',
       'Germany, United States, Hong Kong, Singapore',
       'France, Germany, Switzerland',
       'Germany, France, Luxembourg, United Kingdom, United States',
       'United Kingdom, Canada, Italy', 'Czech Republic, France',
       'Taiwan, Hong Kong, United States, China', 'Germany,
```

```
Australia',
       'United Kingdom, Poland, United States', 'Denmark, Zimbabwe',
       'United Kingdom, South Africa',
       'Finland, Sweden, Norway, Latvia, Germany',
       'South Africa, United States, New Zealand, Canada',
       'United States, Italy, United Kingdom, Liechtenstein',
       'Denmark, France, Belgium, Italy, Netherlands, United States,
United Kingdom',
       'United States, Australia, Mexico',
       'United Kingdom, Czech Republic, Germany, United States',
       'France, China, Japan, United States',
       'United States, South Korea, China', 'Germany, Belgium',
       'Pakistan, Norway, United States',
       'United States, Canada, Belgium, United Kingdom', 'Venezuela',
       'Canada, France, Italy, Morocco, United States',
       'Canada, Spain, France', 'United States, Indonesia',
       'Spain, France, Italy',
       'United Arab Emirates, United States, United Kingdom',
       'United Kingdom, Israel, Russia', 'Spain, Cuba',
       'United States, Brazil', 'United States, France, Mexico',
       'United States, Nicaragua',
       'United Kingdom, United States, Spain, Germany, Greece,
Canada',
       'Italy, Canada, France',
       'United Kingdom, Denmark, Canada, Croatia', 'Italy, Germany',
       'United States, France, United Kingdom, Japan',
       'United States, United Kingdom, Denmark, Sweden',
       'United States, United Kingdom, Italy',
       'United States, France, Canada, Spain',
       'Russia, United States, China', 'United States, Canada,
Germany',
       'Ireland, United States', 'United States, United Arab
Emirates',
       'United States, Ireland',
       'Ireland, United Kingdom, Italy, United States', 'Poland,',
       'Slovenia, Croatia, Germany, Czech Republic, Qatar',
       'Canada, United Kingdom, Netherlands',
       'United States, Spain, Germany', 'India, Japan',
       'China, South Korea, United States',
       'United Kingdom, France, Belgium',
       'Canada, Ireland, United States',
       'United Kingdom, United States, Dominican Republic',
       'United States, Senegal', 'Germany, United Kingdom, United
States',
       'South Africa, Germany, Netherlands, France',
       'Canada, United States, United Kingdom, France, Luxembourg',
       'Ireland, United States, France', 'Germany, United States,
Canada',
       'United Kingdom, Germany, Canada, United States',
```

```
       'United States, France, Canada, Lebanon, Qatar',
       'Netherlands, Belgium, United Kingdom, United States',
       'France, Belgium, China, United States',
       'United States, Chile, Israel',
       'United Kingdom, Norway, Denmark, Germany, Sweden',
       'Norway, Denmark, Sweden', 'China, India, Nepal',
       'Colombia, Mexico, United States', 'United Kingdom, South
Korea',
       'Denmark, China', 'United States, Greece, Brazil',
       'South Korea, France',
       'United States, Australia, Samoa, United Kingdom',
       'Germany, United Kingdom', 'Argentina, Chile, Peru',
       'Turkey, Azerbaijan', 'Poland, West Germany',
       'Germany, United States, Sweden', 'Canada, Spain',
       'United States, Cambodia', 'United States, Greece',
       'Norway, United Kingdom, France, Ireland',
       'United Kingdom, Poland', 'Israel, Sweden, Germany,
Netherlands',
       'Switzerland, France', 'Italy, India', 'United States,
Botswana',
       'Chile, Argentina, France, Spain, United States',
       'United States, India, South Korea, China',
       'Denmark, Germany, Belgium, United Kingdom, France',
       'Denmark, Germany, Belgium, United Kingdom, France, Sweden',
       'France, Switzerland, Spain, United States, United Arab
Emirates',
       'Brazil, India, China, United States',
       'Denmark, France, United States, Sweden', 'Australia, Iraq',
       'China, Morocco, Hong Kong', 'Canada, United States, Germany',
       'United Kingdom, Thailand', 'Venezuela, Colombia',
       'Colombia, United States',
       'France, Germany, Czech Republic, Belgium',
       'Switzerland, Vatican City, Italy, Germany, France',
       'Portugal, France, Poland, United States',
       'United States, New Zealand, Japan',
       'United States, Netherlands, Japan, France', 'India,
Switzerland',
       'Canada, India', 'United States, Morocco',
       'Singapore, Japan, France',
       'Canada, Mexico, Germany, South Africa',
       'United Kingdom, United States, Canada',
       'Germany, France, United States, Canada, United Kingdom',
       'United States, Uruguay', 'India, Canada',
       'Ireland, Canada, United Kingdom, United States',
       'United States, Germany, Australia', 'Australia, France,
Ireland',
       'Australia, India', 'United States, United Kingdom, Canada,
Japan',
       'Sweden, United Kingdom, Finland', 'Hong Kong, Taiwan',
```

```
       'United States, United Kingdom, Spain, South Korea',
'Guatemala',
       'Ukraine',
       'Italy, South Africa, West Germany, Australia, United States',
       'United States, Germany, United Kingdom, Australia',
       'Italy, France, Switzerland', 'Canada, France, United States',
       'Switzerland, United States', 'Thailand, Canada, United
States',
       'China, Hong Kong, United States', 'United Kingdom, New
Zealand',
       'Czech Republic, United Kingdom, France',
       'Australia, United Kingdom, Canada', 'Jamaica, United States',
       'Australia, United Kingdom, United States, New Zealand, Italy,
France',
       'France, United States, Canada',
       'United Kingdom, France, Canada, Belgium, United States',
       'Denmark, United Kingdom, Sweden', 'United States, Hong Kong',
       '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,
China',
       'Ireland, Canada, Luxembourg, United States, United Kingdom,
Philippines, India',
       'United States, Czech Republic, United Kingdom', 'Israel,
Germany',
       'Mexico, France',
       'Israel, Germany, Poland, Luxembourg, Belgium, France, United
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, Israel,
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,
Slovenia',
       '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
States',
       '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 Emirates',
       '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)
```

```
q3 = df_movie['duration_int'].quantile(0.75)
q1 = df_movie['duration_int'].quantile(0.25)

iqr = q3-q1

ub = q3 + 1.5*iqr
lb = q1 - 1.5*iqr


df_movie[(df_movie['duration_int'] > lb) & (df_movie['duration_int'] <
ub)]['duration_int'].describe()
```

```
count    5678.000000
mean       99.962487
std        21.094071
min        47.000000
25%        88.000000
50%        98.500000
75%       113.000000
```

```
max       154.000000
Name: duration_int, dtype: float64
```

```
import pandas as pd

# Example data
data = {'City': ['New York', 'London', 'Tokyo', 'New York', 'Tokyo',
'London']}
df = pd.DataFrame(data)

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   City    6 non-null      object
dtypes: object(1)
memory usage: 180.0+ bytes
```

```
df['City'].astype('category')
```

```
0    New York
1      London
2       Tokyo
3    New York
4       Tokyo
5      London
Name: City, dtype: category
Categories (3, object): ['London', 'New York', 'Tokyo']
```

```
df['Size'] = ['Small', 'Medium', 'Large', 'Small', 'Medium', 'Large']

df # small < medium < larged
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 6,\n  \"fields\": [\n
{\n      \"column\": \"City\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 3,\n
\"samples\": [\n          \"New York\",\n          \"London\",\n
\"Tokyo\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Size\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n
\"Small\",\n          \"Medium\",\n          \"Large\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```

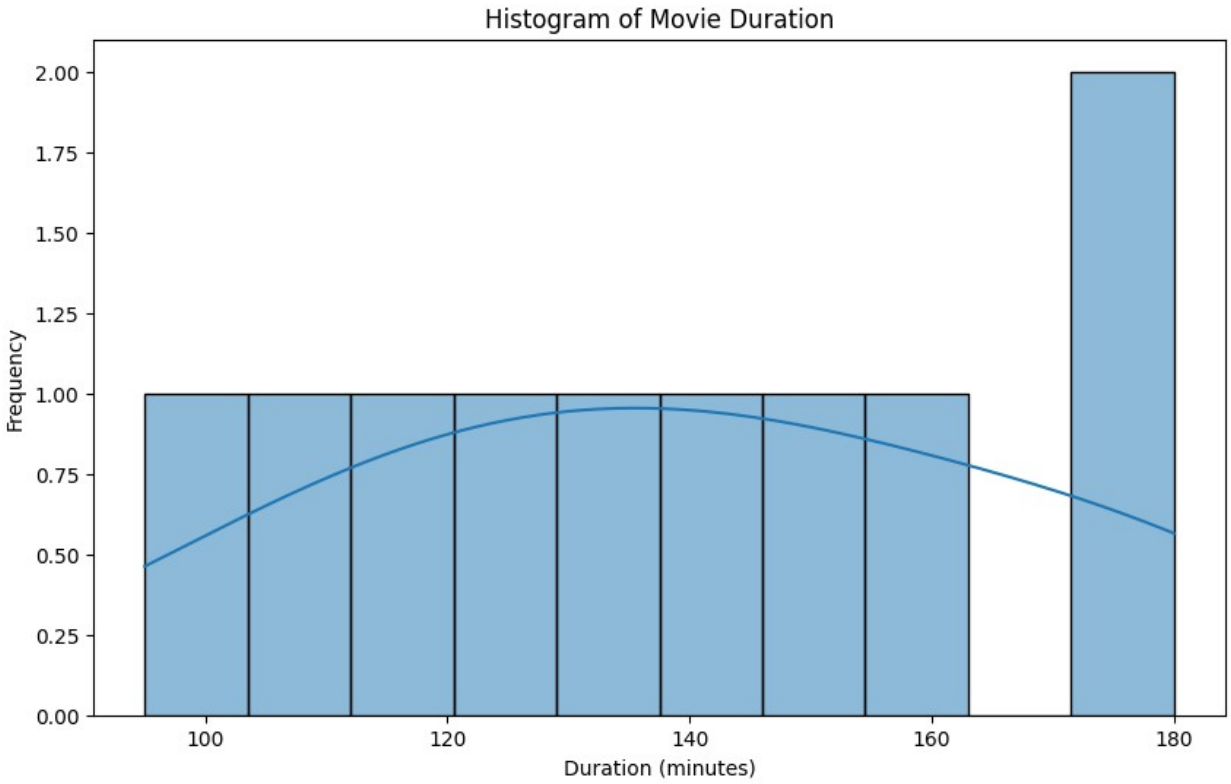

```
df['Size'] =
df['Size'].astype('category').cat.set_categories(['Small', 'Medium',
'Large'], ordered=True)
```

```
df[df['Size'] > 'Small']
```


```
{"summary":"{\n  \"name\": \"df[df['Size'] > 'Small']\",\n  \"rows\":
4,\n  \"fields\": [\n    {\n      \"column\": \"City\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n
\"Tokyo\",\n          \"London\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Size\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"Large\",\n          \"Medium\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```

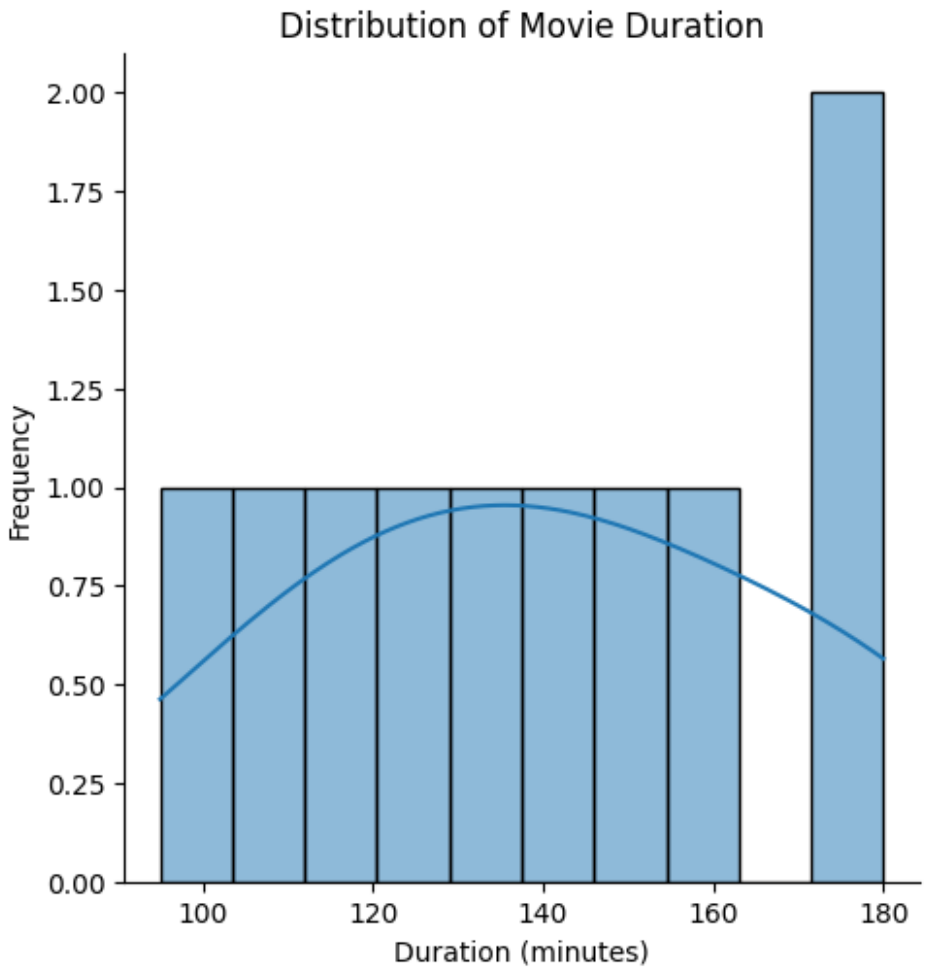

For continuous variable(s): Distplot, countplot, histogram for univariate analysis

```
data = {
    'Movie': ['Movie1', 'Movie2', 'Movie3', 'Movie4', 'Movie5',
'Movie6', 'Movie7', 'Movie8', 'Movie9', 'Movie10'],
    'Duration': [120, 95, 150, 180, 110, 135, 145, 125, 160, 180]
}

df = pd.DataFrame(data)

plt.figure(figsize=(10, 6))
sns.histplot(df['Duration'], kde=True, bins=10)
plt.title('Histogram of Movie Duration')
plt.xlabel('Duration (minutes)')
plt.ylabel('Frequency')
plt.show()
```

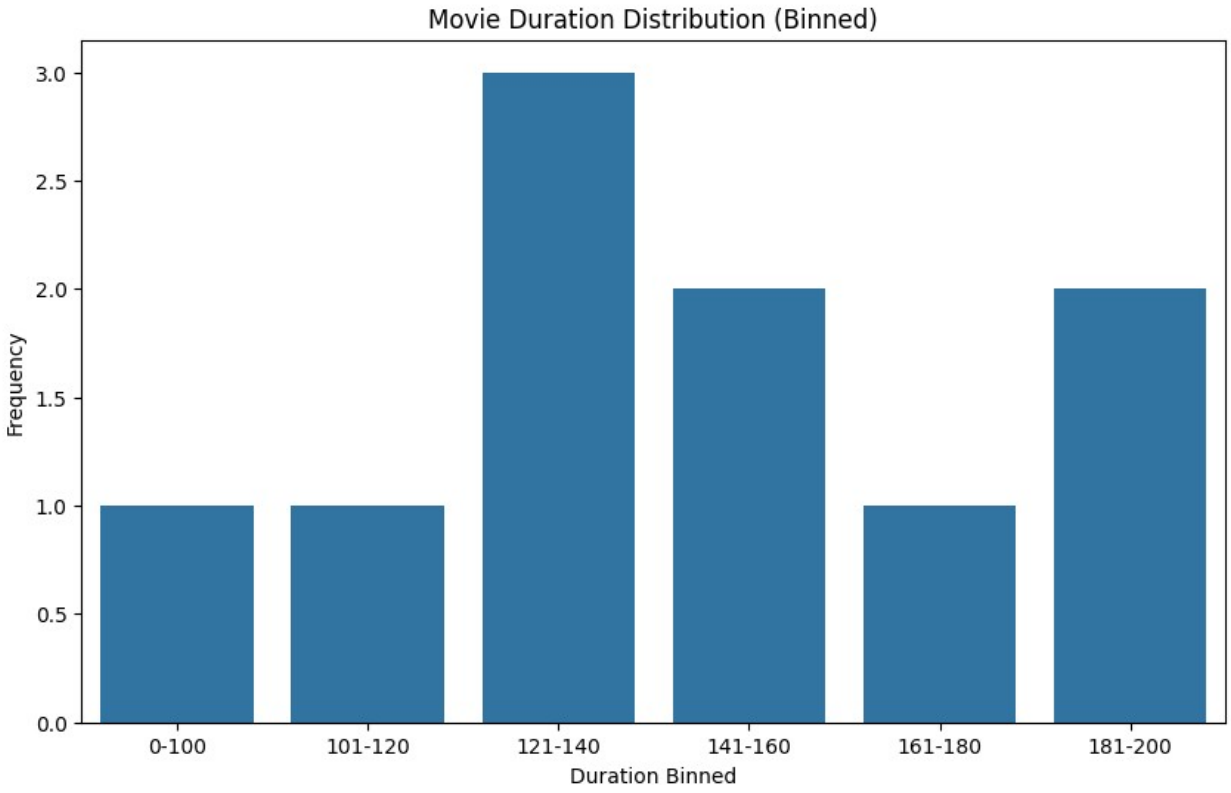



```
plt.figure(figsize=(10, 6))
sns.displot(df['Duration'], kde=True, bins=10)  # KDE curve and
histogram in one
plt.title('Distribution of Movie Duration')
plt.xlabel('Duration (minutes)')
plt.ylabel('Frequency')
plt.show()
```

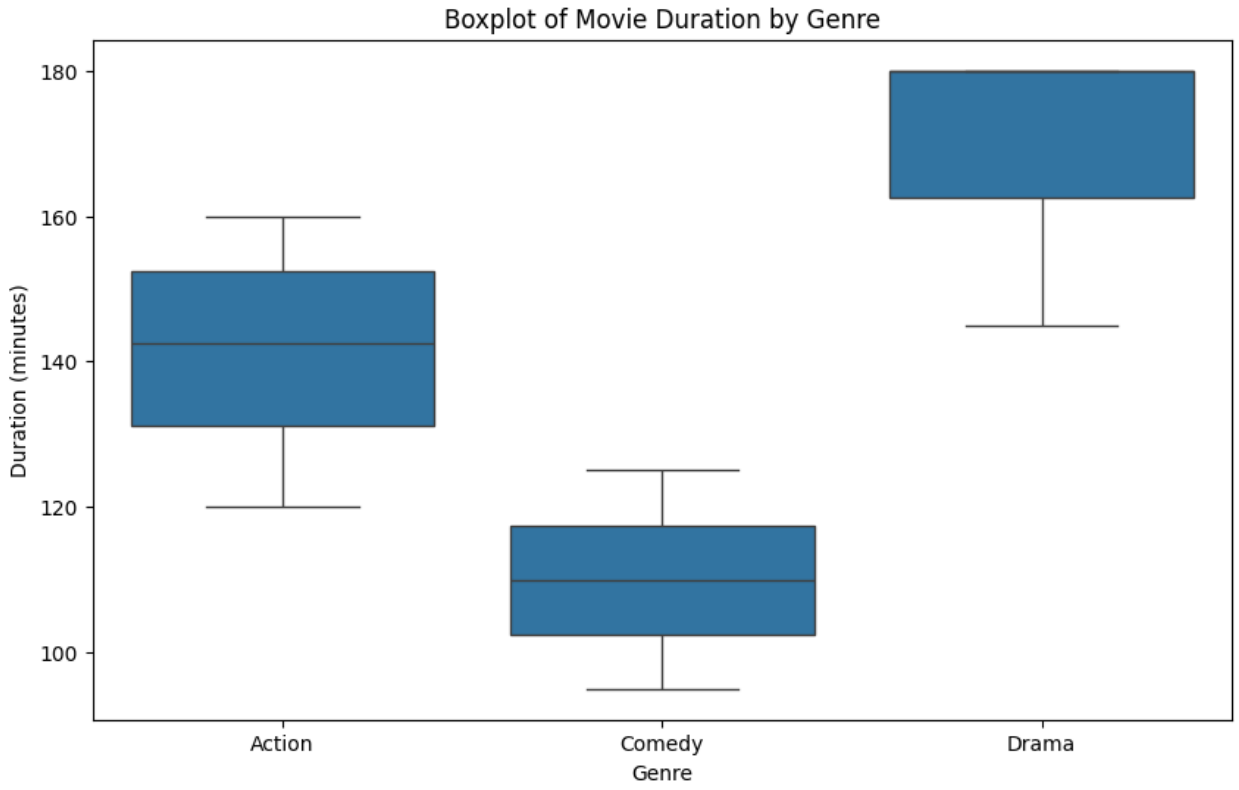
```
<Figure size 1000x600 with 0 Axes>
```



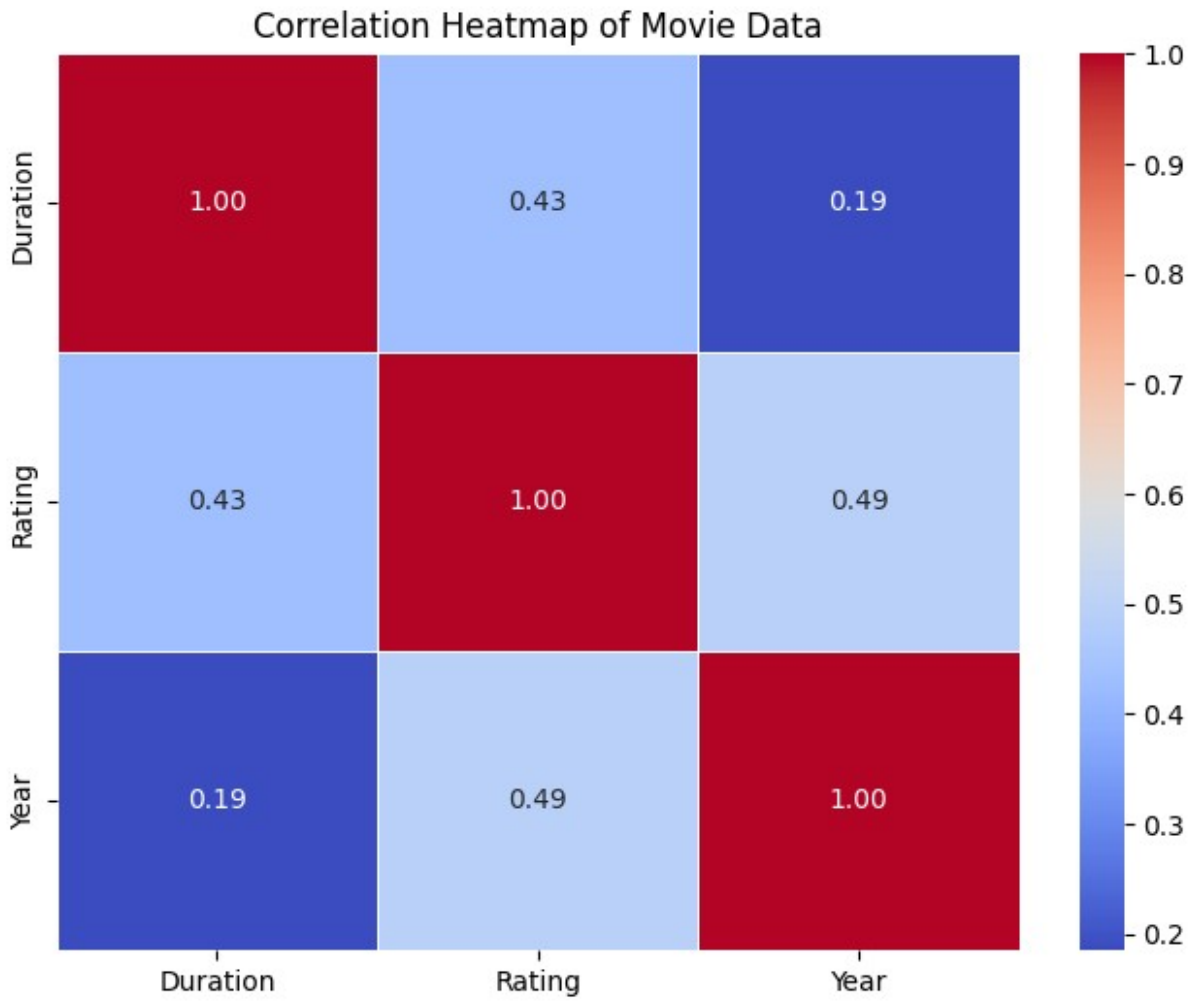

```
bins = [0, 100, 120, 140, 160, 180, 200]  # Custom bins
labels = ['0-100', '101-120', '121-140', '141-160', '161-180', '181-
200']

# Create a new column for binned categories
df['Duration_Binned'] = pd.cut(df['Duration'], bins=bins,
labels=labels, right=False)

# Plotting countplot for binned data
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Duration_Binned')
plt.title('Movie Duration Distribution (Binned)')
plt.xlabel('Duration Binned')
plt.ylabel('Frequency')
plt.show()
```



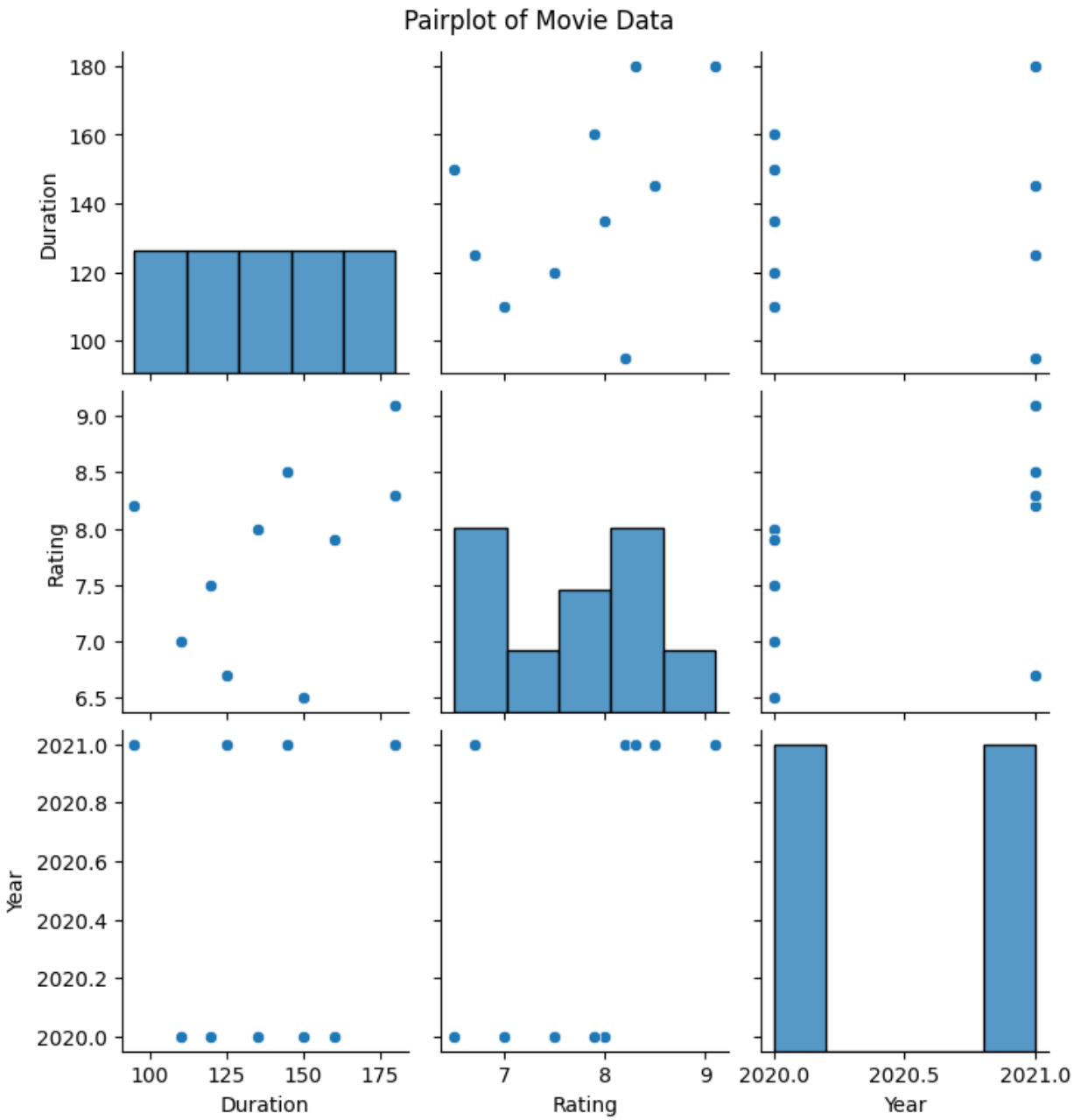

```
#For categorical variable(s): Boxplot
data = {
    'Movie': ['Movie1', 'Movie2', 'Movie3', 'Movie4', 'Movie5',
'Movie6', 'Movie7', 'Movie8', 'Movie9', 'Movie10'],
    'Genre': ['Action', 'Comedy', 'Action', 'Drama', 'Comedy',
'Action', 'Drama', 'Comedy', 'Action', 'Drama'],
    'Duration': [120, 95, 150, 180, 110, 135, 145, 125, 160, 180]
}
df = pd.DataFrame(data)

# Boxplot for 'Duration' across different 'Genre'
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='Genre', y='Duration')
plt.title('Boxplot of Movie Duration by Genre')
plt.xlabel('Genre')
plt.ylabel('Duration (minutes)')
plt.show()
```




```
#For correlation: Heatmaps, Pairplots
data = {
    'Movie': ['Movie1', 'Movie2', 'Movie3', 'Movie4', 'Movie5',
'Movie6', 'Movie7', 'Movie8', 'Movie9', 'Movie10'],
    'Duration': [120, 95, 150, 180, 110, 135, 145, 125, 160, 180],
    'Rating': [7.5, 8.2, 6.5, 9.1, 7.0, 8.0, 8.5, 6.7, 7.9, 8.3],
    'Year': [2020, 2021, 2020, 2021, 2020, 2020, 2021, 2021, 2020,
2021]
}

# Create DataFrame
df = pd.DataFrame(data)

# Compute the correlation matrix
correlation_matrix = df[['Duration', 'Rating', 'Year']].corr()

# Create the heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
fmt='.2f', linewidths=0.5)
plt.title('Correlation Heatmap of Movie Data')
plt.show()
```




```
sns.pairplot(df[['Duration', 'Rating', 'Year']])
plt.suptitle('Pairplot of Movie Data', y=1.02)  # Adjust title
position
plt.show()
```




```
df=pd.read_csv('netflix.csv')

missing_values = df.isnull().sum()
print("Missing values in each column:")
print(missing_values)
```

```
Missing values in each column:
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
```

```
data_types = df.dtypes
unique_values = df.nunique()
print("Data types:")
print(data_types)
print("Unique values per column:")
print(unique_values)
```

```
Data types:
show_id         object
type            object
title           object
director        object
cast            object
country         object
date_added      object
release_year     int64
rating          object
duration        object
listed_in       object
description     object
dtype: object
Unique values per column:
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
```

```
summary_statistics = df.describe()
print("Summary statistics for numerical columns:")
print(summary_statistics)
```

```
Summary statistics for numerical columns:
       release_year
count   8807.000000
mean    2014.180198
std        8.819312
min     1925.000000
25%     2013.000000
50%     2017.000000
75%     2019.000000
max     2021.000000
```

Business Insights (10 Points) - Should include patterns observed in the data along with what can u infer from it

1. Most Popular Genres Pattern: Based on the frequency distribution (e.g., using .value_counts() or countplot), genres like Drama, Action, Comedy, and Documentary may emerge as the most common genres in the dataset.

Business Insight:

Targeted Content Creation: Netflix can focus more on producing movies and TV shows in these popular genres to meet user demand.

Marketing Strategy: Use the most popular genres in campaigns to attract more viewers (e.g., "Top Comedy Movies" or "Best Action Movies").

1. Top Countries Producing Content Pattern: Countries like the United States, India, and United Kingdom are likely to have the highest number of shows and movies.

Business Insight:

Localization: Netflix can prioritize content from these countries to expand viewership in those regions. Additionally, they can enhance regional offerings for countries with growing user bases.

International Growth: Emphasize partnerships and content creation in emerging markets where Netflix's presence is growing.

1. Distribution of Movie Duration Pattern: Most movies might have a typical runtime of 90-150 minutes based on the duration column (observed through boxplots or histograms).

Business Insight:

Content Strategy: Movies in the 90-120 minute range tend to be more popular, so Netflix may consider producing more movies in this duration to cater to users' attention spans.

Optimizing User Experience: Given that users may prefer shorter content (e.g., 90-minute movies), Netflix can focus on curating a collection of films that fit this runtime.

1. Rating Distribution by Genre Pattern: The average ratings across genres could show that genres like Drama and Thriller have higher ratings compared to genres like Action or Sci-Fi.

Business Insight:

Content Investment: Invest more in high-rated genres like Drama and Thriller, which tend to attract a more engaged audience.

Recommendations: Netflix can recommend higher-rated genres to users based on their viewing history, improving customer satisfaction and retention.

1. Release Year Trends Pattern: Analyzing the release year column can show if there has been an increase in content production in recent years, or if certain periods (e.g., post-2010) are witnessing a surge in movie and TV series releases.

Business Insight:

Content Evolution: If more recent years show higher content production, this might indicate a strategy shift toward creating more original content, reflecting the growing demand for fresh content.

User Trends: Tailor the content recommendations to specific years or periods based on users' viewing preferences for certain types of movies (e.g., nostalgia for older classics or preference for modern releases).

1. User Rating Behavior Pattern: If most ratings are clustered around a narrow range (e.g., 7-8), it suggests that users generally provide moderate-to-high ratings for the content they watch.

Business Insight:

Content Quality: Netflix could investigate why most movies or TV shows are getting middle-range ratings and look for ways to improve content quality in highly-rated areas.

User Engagement: Encourage more user ratings and reviews by providing incentives, such as offering personalized recommendations based on their ratings.

1. Content Type Popularity (Movies vs TV Shows) Pattern: The dataset might show a higher volume of movies compared to TV shows, or vice versa.

Business Insight:

Content Strategy: If movies dominate, Netflix could expand its offerings of TV shows, particularly serialized content that encourages long-term subscription retention.

Subscription Model: For a market where viewers prefer more episodic content, Netflix could focus on creating more original series rather than standalone movies.

1. Geographic Distribution of Content Pattern: Analyzing the country distribution of the content could show that certain countries (like the U.S.) have a higher concentration of Netflix-produced or original content.

Business Insight:

Expansion in International Markets: Netflix can focus on acquiring or producing more content from emerging markets to cater to global tastes and to appeal to diverse international audiences.

Localized Content: Offering more localized content (e.g., sub-titled or dubbed content in regional languages) could enhance user engagement in non-English speaking countries.

1. Content Popularity Based on Release Year Pattern: Movies or shows released in certain years (e.g., in the 2000s or 2010s) might have a higher number of views or ratings than older ones.

Business Insight:

Data-Driven Recommendations: Netflix can improve its recommendation algorithm by promoting newer releases more aggressively or focusing on users' preferences for older shows (nostalgic content).

Content Lifecycle: Content from older years may still have a loyal following, so a revival strategy (e.g., re-releases, remakes, or spin-offs) could be explored.

1. Outliers in Ratings (Extremely High or Low Ratings) Pattern: Some content might have very high ratings (e.g., 9+), which could represent critical hits, while others might have very low ratings (e.g., below 4), indicating poorly received content.

Business Insight:

Curating High-Quality Content: Netflix should analyze the factors behind highly-rated content (such as acting, storyline, or directing) to replicate these aspects in future productions.

Content Removal or Reworking: For poorly-rated content, Netflix may consider either removing it from recommendations, improving it with additional seasons/episodes, or investing in better content.

#Recommendations (10 Points) - Actionable items for business. No technical jargon. No complications. Simple action items that everyone can understand.

1. Produce More Content in Popular Genres Action: Focus on creating more content in genres like Drama, Comedy, and Action since they are the most popular.

Reason: Viewers are more likely to watch and engage with these genres, which can lead to higher subscriptions.

1. Increase Investment in International Content Action: Invest in content from countries with growing viewer bases, especially India, South Korea, and Brazil.

Reason: Global content attracts diverse audiences and helps Netflix grow in international markets.

1. Create Shorter Films and TV Shows Action: Produce more content that's around 90-120 minutes in length, as this is the preferred viewing time for many users.

Reason: Shorter content aligns with the typical attention span of viewers and may lead to higher engagement.

1. Offer Personalized Recommendations Based on Ratings Action: Use highly rated content in popular genres to recommend similar shows or movies to users.

Reason: Tailored recommendations help increase user satisfaction and engagement by suggesting content viewers are more likely to enjoy.

1. Focus on Content with High Ratings Action: Pay attention to content with high ratings (8+) and replicate successful elements (like acting, storytelling, etc.) in new content.

Reason: Users are more likely to subscribe and stay engaged with content that is highly rated and receives good feedback.

1. Introduce More TV Shows and Series Action: If movies dominate, consider expanding the variety of TV shows and series available to users.

Reason: TV series keep users engaged for longer periods and encourage binge-watching, leading to longer subscription periods.

1. Prioritize Localized Content for Different Regions Action: Increase the production and availability of localized content in different languages, tailored to each region's culture.

Reason: Providing content in users' native languages increases accessibility, appeal, and engagement in local markets.

1. Revive Older Content or Remake Classics Action: Consider reviving older, nostalgic content (e.g., movies or series from the 90s or early 2000s) or producing remakes and spin-offs.

Reason: Nostalgic content resonates with long-time viewers and can attract new subscribers who are interested in rewatching classics.

1. Promote Content with High User Engagement Action: Highlight and promote content that has high user engagement, such as user ratings, comments, or shares.

Reason: Popular content with lots of engagement can be an effective marketing tool and drive more subscribers to the platform.

1. Consider Content Removal Based on Low Ratings Action: Remove or downplay poorly rated content from recommendations and focus on promoting higher-rated content.

Reason: Highlighting better-rated content will improve user satisfaction and encourage users to stay subscribed for a better viewing experience.