```
# importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from scipy.stats import anderson
from matplotlib import cm
from matplotlib import colors

warnings.filterwarnings('ignore')
```

```
# Abro el df

df = pd.read_csv(r'https://raw.githubusercontent.com/IT-Academy-BCN/Data-Science/main/Data-sources/tips.

df.head(3)
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3

Nivell 1

- Exercici 1 Realitza la pràctica del notebook a GitHub "03 EXAMINING DATA" amb seaborn i el dataset "tips".

```
df.columns
```

```
Index(['total_bill', 'tip', 'sex', 'smoker', 'day', 'time', 'size'], dtype='object')
```

```
print('La media de total_bill es: ', round(df.total_bill.mean(),2), '€')
```

```
La media de total_bill es:  19.79 €
```

```
df.time.unique()
```

```
array(['Dinner', 'Lunch'], dtype=object)
```

```
df.describe().round(2)
```

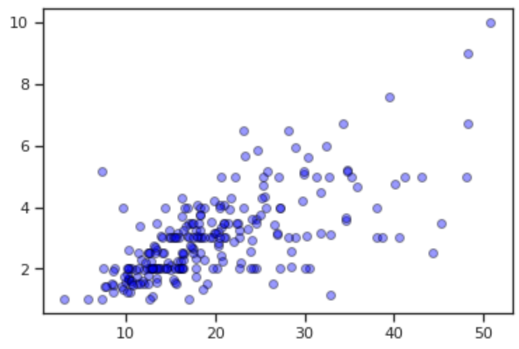
	total_bill	tip	size
count	244.00	244.00	244.00
mean	19.79	3.00	2.57
std	8.90	1.38	0.95
min	3.07	1.00	1.00
25%	13.35	2.00	2.00
50%	17.80	2.90	2.00
75%	24.13	3.56	3.00
max	50.81	10.00	6.00

Scatter plot

```
# Plot
x= df['total_bill']
y= df['tip']
colors = 'Blue'
```

```

plt.scatter(x, y, c=colors, alpha=0.4, edgecolors='black')
```

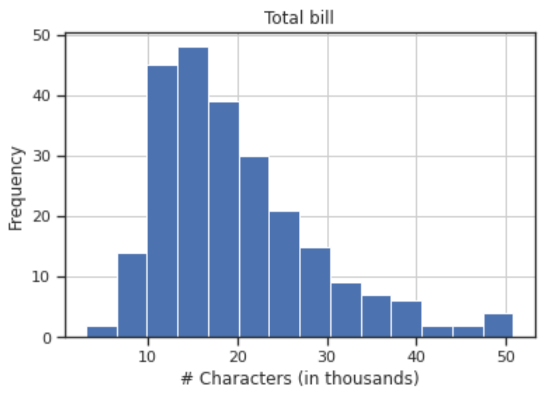
```
<matplotlib.collections.PathCollection at 0x7fe04610ca90>
```



Histograma

```
df.hist(['total_bill'], bins=14)
plt.title('Total bill')
plt.ylabel('Frequency')
plt.xlabel('# Characters (in thousands)')
```

```
Text(0.5, 0, '# Characters (in thousands)')
```



SUMMARY STATISTICS

```
print('aqui tienes todos los estadisticos de la base de datos continuos (No discretos)')
df.describe().round(2)
```

```
aqui tienes todos los estadisticos de la base de datos continuos (No discretos)
```

	total_bill	tip	size
count	244.00	244.00	244.00
mean	19.79	3.00	2.57
std	8.90	1.38	0.95
min	3.07	1.00	1.00
25%	13.35	2.00	2.00
50%	17.80	2.90	2.00
75%	24.13	3.56	3.00
max	50.81	10.00	6.00

```
# Calulo el valor de la desv Std y miro si la distribución es normal
desvStd = df.total_bill.std()
print ('desviacion estandar', round(desvStd,3))

p_Value =anderson(df['total_bill'])
```

```
print('la distribucion es normal. P-value: ',  p_Value[1][2])
```

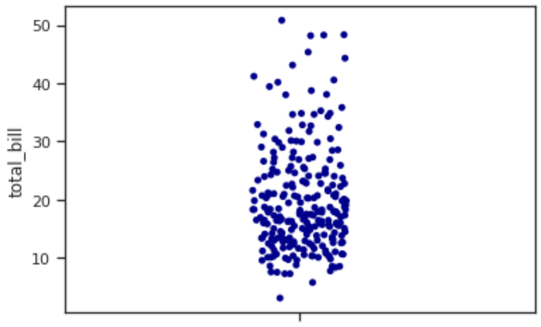
```
desviacion estandar 8.902
la distribucion es normal. P-value:  0.775
```

```
# Calculo los cuartiles del BoxPlot

print('Para saber el valor de Q1, Q2, IQR:')
(df['total_bill']).describe()
```

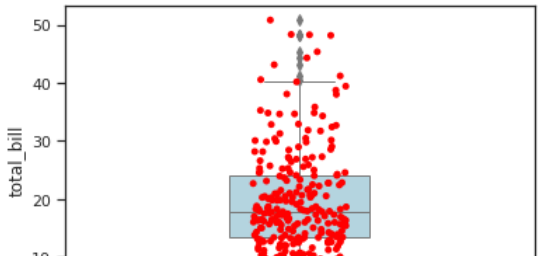
```
Para saber el valor de Q1, Q2, IQR:
count    244.000000
mean      19.785943
std        8.902412
min        3.070000
25%       13.347500
50%       17.795000
75%       24.127500
max       50.810000
Name: total_bill, dtype: float64
```

```
sns.stripplot(y=df["total_bill"], orient='v', color='darkblue')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe04615a290>
```



```
ax = sns.boxplot(y=df["total_bill"], data=df,  color='lightblue', fliersize=5,  orient='v', linewidth=1
ax = sns.stripplot(y=df["total_bill"], orient='v', color='red')
```



EXERCISE - 3.8

What percent of the data fall between Q1 and the median? What percent is between the median and Q3?

EXERCISE - 3.9

estimate the following values for num_char in the dataset:

a).- Q1

b).- Q3 and

c).- IQR

```
df
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

```python
# calcular el rango intercuartílico
q3, q1, mediana = np. percentile (df['total_bill'], [75, 25, 50]).round(2)
iqr = q3 - q1
print('Q1: ', q1)
print('Q3: ', q3)
print('mediana: ', mediana)
print('IQR: ', iqr)
```

```
Q1:  13.35
Q3:  24.13
mediana:  17.8
IQR:  10.78
```

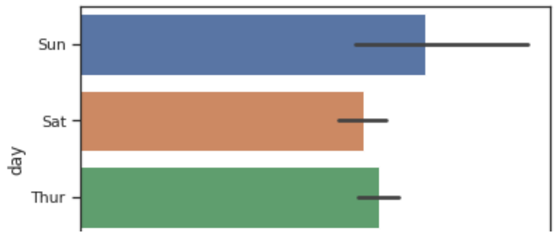
```python
tips = df.copy()
tips.tip
```

```
0      1.01
1      1.66
2      3.50
3      3.31
4      3.61
       ...
239    5.92
240    2.00
241    2.00
242    1.75
243    3.00
Name: tip, Length: 244, dtype: float64
```

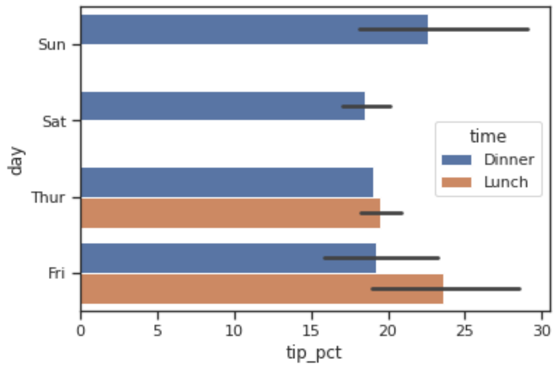
```python

tips['tip_pct'] = round((tips['tip'] / (tips['total_bill'] - tips['tip']))*100, 2)
tips.head()
```

	total_bill	tip	sex	smoker	day	time	size	tip_pct
0	16.99	1.01	Female	No	Sun	Dinner	2	6.32
1	10.34	1.66	Male	No	Sun	Dinner	3	19.12
2	21.01	3.50	Male	No	Sun	Dinner	3	19.99
3	23.68	3.31	Male	No	Sun	Dinner	2	16.25
4	24.59	3.61	Female	No	Sun	Dinner	4	17.21

```python
sns.barplot(x='tip_pct', y='day', data=tips, orient="h")
plt.show()
```



```
sns.barplot(x='tip_pct', y='day', hue='time', data=tips, orient='h')
plt.show()
sns.set(style="darkgrid")
```



```
round(tips.describe(include='all'), 1)
```

	total_bill	tip	sex	smoker	day	time	size	tip_pct
count	244.0	244.0	244	244	244	244	244.0	244.0
unique	NaN	NaN	2	2	4	2	NaN	NaN
top	NaN	NaN	Male	No	Sat	Dinner	NaN	NaN
freq	NaN	NaN	157	151	87	176	NaN	NaN
mean	19.8	3.0	NaN	NaN	NaN	NaN	2.6	20.2
std	8.9	1.4	NaN	NaN	NaN	NaN	1.0	16.3
min	3.1	1.0	NaN	NaN	NaN	NaN	1.0	3.7
25%	13.3	2.0	NaN	NaN	NaN	NaN	2.0	14.8
50%	17.8	2.9	NaN	NaN	NaN	NaN	2.0	18.3
75%	24.1	3.6	NaN	NaN	NaN	NaN	3.0	23.7
max	50.8	10.0	NaN	NaN	NaN	NaN	6.0	245.2

```
tips.isnull().sum()/len(tips)
```

```
total_bill    0.0
tip           0.0
sex           0.0
smoker        0.0
day           0.0
time          0.0
size          0.0
tip_pct       0.0
dtype: float64
```

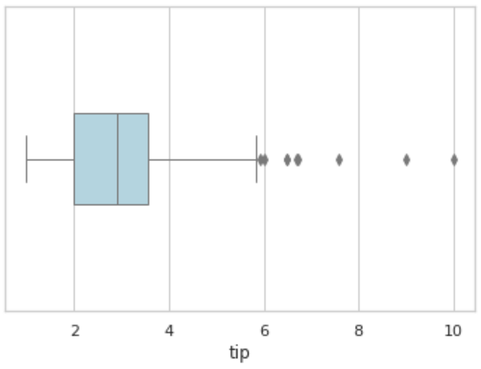
```
round((tips['tip']).describe(), 3)
```

```
count    244.000
mean       2.998
std        1.384
min        1.000
25%        2.000
50%        2.900
75%        3.562
max       10.000
Name: tip, dtype: float64
```

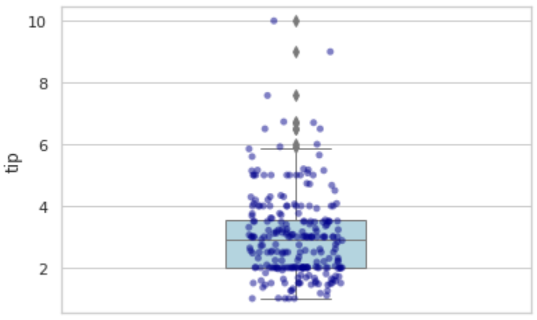
```
(tips['tip']).median()
```

```
2.9
```

```
sns.set(style="whitegrid")

ax = sns.boxplot(x = tips['tip'], color='lightblue', fliersize=5, orient='v', linewidth=1, width=0.3)
plt.show()
```



```
ax = sns.boxplot(y="tip", data=tips,  color='lightblue', fliersize=5,  orient='v', linewidth=1 , width=0
ax = sns.stripplot(y=tips["tip"], orient='v', color='darkblue', alpha= 0.5)

```

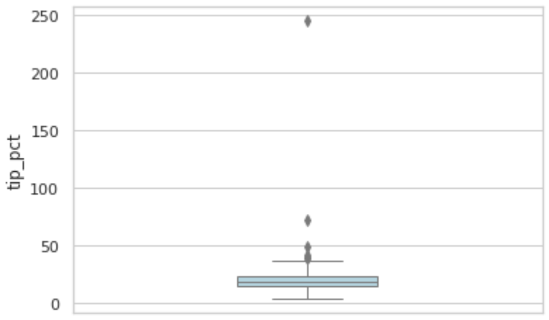


Una Variable: 1 Numérica = 'tip_pct'

```
tips.dtypes
```

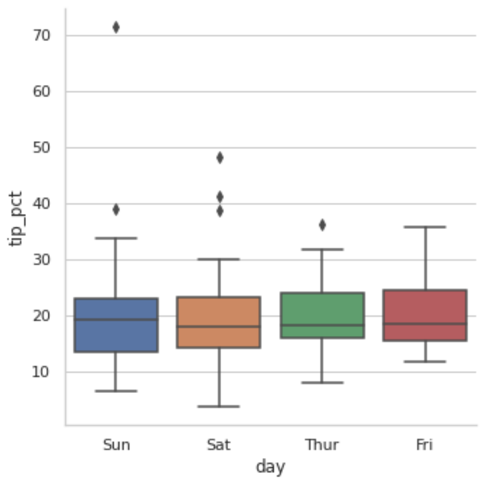
```
total_bill    float64
tip           float64
sex            object
smoker         object
day            object
time           object
size            int64
tip_pct       float64
dtype: object
```

```
sns.boxplot(y="tip_pct", data=tips[tips.tip < 10],  color='lightblue', fliersize=5,  orient='v', linewid
```

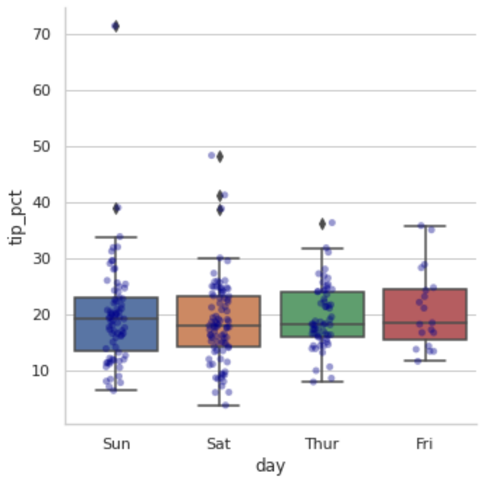


Dos Variables: 1 Categórica = 'day', 1 Numérica = 'tip_pct'

```
# añadimos variable categorica 'day' en x:
ax = sns.catplot(x='day', y='tip_pct', kind='box',
                 data=tips[tips.tip_pct < 245]);
```

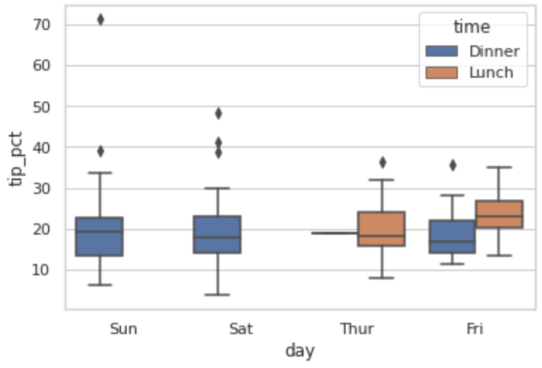


```
## añadimos variable categorica 'day' en x:
ax = sns.catplot(x='day', y='tip_pct', kind='box',
                 data=tips[tips.tip_pct < 245]);

ax = sns.stripplot(x='day', y='tip_pct', data=tips[tips.tip_pct < 245], orient='v', color='darkblue', al
```

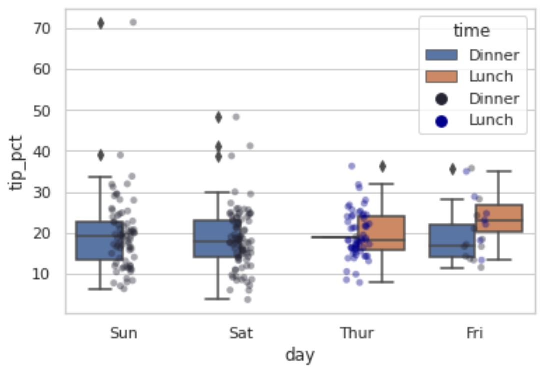


Tres Variables : 2 Categóricas = ('day', 'time'), 1 Numérica = 'tip_pct'

```
sns.boxplot(x='day', y='tip_pct', hue='time',
               data=tips[tips.tip_pct < 245]);
```



```
sns.boxplot(x='day', y='tip_pct', hue = 'time',
               data=tips[tips.tip_pct < 245]);
ax = sns.stripplot(x='day', y='tip_pct', hue='time', data=tips[tips.tip_pct < 245], orient='v', color='c
```



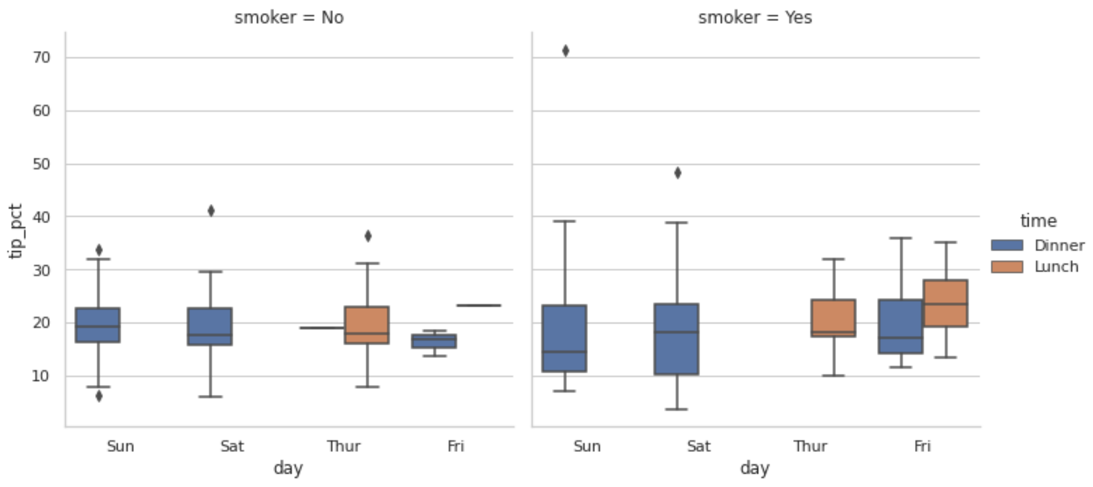
Facet Grids y Categorical DataFrame

Nos permite profundizar todavía más en el analysis, añadiendo una variable categórica adicional.

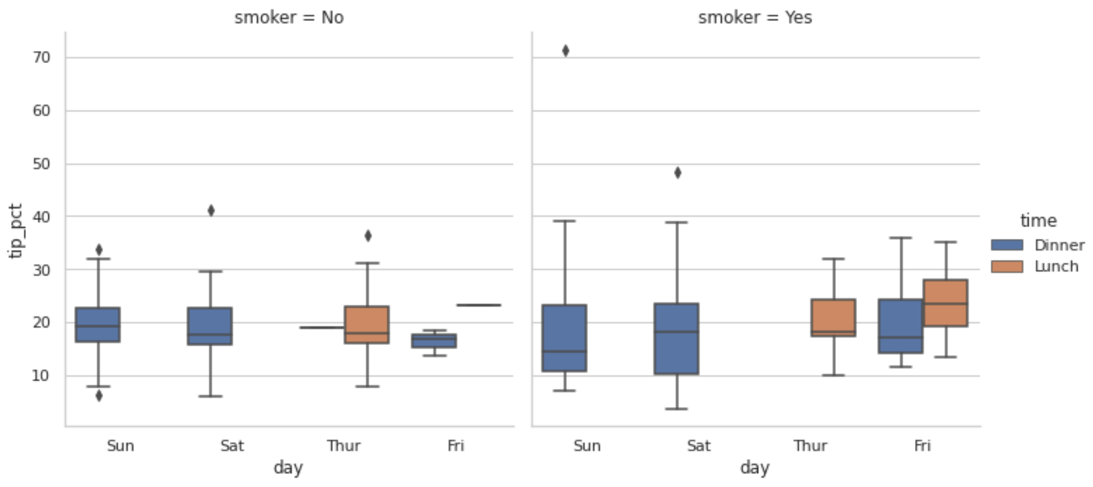
Usando el método factorplot( ) de "Facet Grid" :

Cuatro Variables **texto en negrita** : 3 Categoricas = ('day', 'time', 'smoker'), 1 Numérica = 'tip_pct'

```

sns.catplot(x='day', y='tip_pct', hue='time', col='smoker',
               kind='box', data=tips[tips.tip_pct < 245]);
```



```

sns.catplot(x='day', y='tip_pct', hue='time', col='smoker',
               kind='box', data=tips[tips.tip_pct < 245]);
```



Nivell 2

Exercici 2

Repeteix l'exercici 1 amb el dataset que disposem en el repositori de GitHub PRE-PROCESSING-DATA movies.dat

```python
# Abro el fichero

titulo = ['Id', 'Titulo', 'Tipo']
df = pd.read_csv(r'https://raw.githubusercontent.com/IT-Academy-BCN/Data-Science/main/Pre-processing-dat

#df[['A', 'B']] = pd.DataFrame(df['Titulo Año'].str.split(pat='(', expand= True))

#df = pd.DataFrame(df['Titulo Año'].str.split('(',1).tolist())
df.head(4)
```

```
   Id                Titulo                          Tipo
0   1      Toy Story (1995)   Animation|Children's|Comedy
1   2        Jumanji (1995)  Adventure|Children's|Fantasy
2   3  Grumpier Old Men (1995)             Comedy|Romance
```

```python
# Miro el tamaño del df

df.shape
df['year'] = df["Titulo"].str[-5:-1]
```

```python
# Divido la fecha

df['Titulo'].str.split('(', expand= True)
df['Anyo'] = df['Titulo'].str[-5:-1]
#df['lonfitud'] = df['Titulo'].len()
df['Anyo'] = df['Anyo'].astype('float64')




df
```

```
        Id                          Titulo                          Tipo  year    Anyo
0        1                Toy Story (1995)   Animation|Children's|Comedy  1995  1995.0
1        2                  Jumanji (1995)  Adventure|Children's|Fantasy  1995  1995.0
2        3         Grumpier Old Men (1995)                Comedy|Romance  1995  1995.0
3        4        Waiting to Exhale (1995)                  Comedy|Drama  1995  1995.0
4        5  Father of the Bride Part II (1995)                  Comedy  1995  1995.0
...    ...                             ...                           ...   ...     ...
3878  3948          Meet the Parents (2000)                       Comedy  2000  2000.0
3879  3949       Requiem for a Dream (2000)                        Drama  2000  2000.0
3880  3950                Tigerland (2000)                         Drama  2000  2000.0
3881  3951         Two Family House (2000)                         Drama  2000  2000.0
3882  3952          Contender, The (2000)                 Drama|Thriller  2000  2000.0

3883 rows × 5 columns
```

```python
df['Anyo']= pd.DataFrame(df['Anyo'].unique())
print()
df= df.sort_values('Anyo')
df.describe()
df
```

	Id	Titulo	Tipo	year	Anyo
77	78	Crossing Guard, The (1995)	Drama	1995	1919.0
79	80	White Balloon, The (Badkonake Sefid) (1995)	Drama	1995	1920.0
80	81	Things to Do in Denver when You're Dead (1995)	Crime|Drama|Romance	1995	1921.0
70	71	Fair Game (1995)	Action	1995	1922.0
75	76	Screamers (1995)	Sci-Fi|Thriller	1995	1923.0
...	...	...	...	...	...
3878	3948	Meet the Parents (2000)	Comedy	2000	NaN
3879	3949	Requiem for a Dream (2000)	Drama	2000	NaN
3880	3950	Tigerland (2000)	Drama	2000	NaN
3881	3951	Two Family House (2000)	Drama	2000	NaN
3882	3952	Contender, The (2000)	Drama|Thriller	2000	NaN

```python
genres = [
    "Action",
    "Adventure",
    "Animation",
    "Children's",
    "Comedy",
    "Crime",
    "Documentary",
    "Drama",
    "Fantasy",
    "Film-Noir",
    "Horror",
    "Musical",
    "Mystery",
    "Romance",
    "Sci-Fi",
    "Thriller",
    "War",
    "Western",
]
```

```python
# Creo columnas con el tipo de pelicula que es

for genre in genres:
    df[genre] = df["Tipo"].apply(
        lambda values: int(genre in values.split("|"))
    )
df.head(1)
df3= df.copy()
```

```python
estilos =pd.DataFrame(df.sum())

# Quito las columnas primeras porque no son numericas
estilos = estilos[2:].copy()
print('\n\nVeo que tengo que quitar las filas year y Anyo')
estilos
```

```
Veo que tengo que quitar las filas year y Anyo
```

	0
Tipo	DramaDramaCrime|Drama|RomanceActionSci-Fi|Thri...
year	19951995199519951995199619951995199619951995 19...
Anyo	158755.0
Action	503
Adventure	283
Animation	105
Children's	251
Comedy	1200
Crime	211
Documentary	127
Drama	1603
Fantasy	68
Film-Noir	44

```
estilos.shape

estilos = estilos.rename_axis('genero').reset_index()
print(estilos)
```

```
         genero                                                  0
0          Tipo  DramaDramaCrime|Drama|RomanceActionSci-Fi|Thri...
1          year  19951995199519951995199619951995199619951995 19...
2          Anyo                                           158755.0
3        Action                                                503
4     Adventure                                                283
5     Animation                                                105
6    Children's                                                251
7        Comedy                                               1200
8         Crime                                                211
9   Documentary                                                127
10        Drama                                               1603
11      Fantasy                                                 68
12    Film-Noir                                                 44
13       Horror                                                343
14      Musical                                                114
15      Mystery                                                106
16      Romance                                                471
17       Sci-Fi                                                276
18     Thriller                                                492
19          War                                                143
20      Western                                                 68
```

```
df3 = df.copy()
```

```
# Cambio el nombre de la columna cantidad

df3 = estilos.rename(columns ={0:'cantidad'})
# Quito las primeras filas que no dicen nada
df3 = df3[3:]
df3 = df3.sort_values('cantidad', ascending = False)
print(df3.columns)
```

```
Index(['genero', 'cantidad'], dtype='object')
```

```
# Modelo adecuado


df3
```

	genero	cantidad
10	Drama	1603
7	Comedy	1200
3	Action	503
18	Thriller	492
16	Romance	471
13	Horror	343
4	Adventure	283
17	Sci-Fi	276
6	Children's	251
8	Crime	211
19	War	143
9	Documentary	127
14	Musical	114
15	Mystery	106
5	Animation	105
20	Western	68

```
# Hago estadistica de las variables numéricas

df3['cantidad'] = df3['cantidad'].astype('float64')
df3['cantidad'].describe().round(1)

```

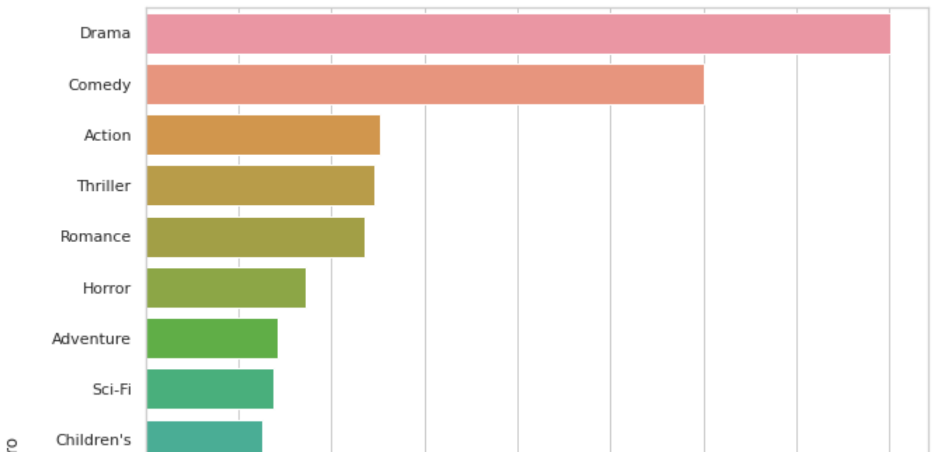
```
count      18.0
mean      356.0
std       413.7
min        44.0
25%       108.0
50%       231.0
75%       439.0
max      1603.0
Name: cantidad, dtype: float64
```

Nivell 3

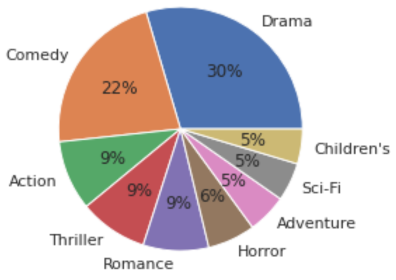
- Exercici 3 En aquest exercici no us donarem gaires indicacions perquè volem que ens mostreu la vostra creativitat. Sorprèn-me amb gràfiques i interpretacions del dataset "movies.dat" del exercici anterior.

```
# DIbujo un diagrama de barra

fig1, ax1 = plt.subplots(figsize=(10,12))
sns.barplot(x='cantidad', y='genero', data=df3)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe0467157d0>
```



```
#creo pie chart de los 9 estilos más importantes

df33=df3[0:9]
plt.pie(df33['cantidad'], labels = df33['genero'], autopct='%.0f%%')
plt.show()
print('Principales generos')
```



```
Principales generos
```

```
# Obtener los años del fichero:

dfAnyos = df['Anyo'].unique()
dfAnyos
```

```
array([1919., 1920., 1921., 1922., 1923., 1925., 1926., 1927., 1928.,
       1929., 1930., 1931., 1932., 1933., 1934., 1935., 1936., 1937.,
       1938., 1939., 1940., 1941., 1942., 1943., 1944., 1945., 1946.,
       1947., 1948., 1949., 1950., 1951., 1952., 1953., 1954., 1955.,
       1956., 1957., 1958., 1959., 1960., 1961., 1962., 1963., 1964.,
       1965., 1966., 1967., 1968., 1969., 1970., 1971., 1972., 1973.,
       1974., 1975., 1976., 1977., 1978., 1979., 1980., 1981., 1982.,
       1983., 1984., 1985., 1986., 1987., 1988., 1989., 1990., 1991.,
       1992., 1993., 1994., 1995., 1996., 1997., 1998., 1999., 2000.,
         nan])
```

```
# Listo los generos de las películas
genres

```

```
['Action',
 'Adventure',
 'Animation',
 "Children's",
 'Comedy',
 'Crime',
 'Documentary',
 'Drama',
 'Fantasy',
 'Film-Noir',
 'Horror',
 'Musical',
 'Mystery',
 'Romance',
 'Sci-Fi',
 'Thriller',
    'War',
```

Listo por años

```
df6 =pd.DataFrame(df.groupby(by =['Anyo']).sum().reset_index())
df6.columns
df6.head(3)
```

	Anyo	Id	Action	Adventure	Animation	Children's	Comedy	Crime	Documentary	Drama	Fantasy	Film-Noir	Horror	Musical
0	1919.0	78	0	0	0	0	0	0	0	1	0	0	0	0
1	1920.0	80	0	0	0	0	0	0	0	1	0	0	0	0

```
# Para cada genero calculo cuantas pelicas han producido a lo largo de los años

total=0
for x in genres:
    print(x, df[x].sum())
    total+=df[x].sum()
```

```
Action 503
Adventure 283
Animation 105
Children's 251
Comedy 1200
Crime 211
Documentary 127
Drama 1603
Fantasy 68
Film-Noir 44
Horror 343
Musical 114
Mystery 106
Romance 471
Sci-Fi 276
Thriller 492
War 143
Western 68
```

Sumo por géneros

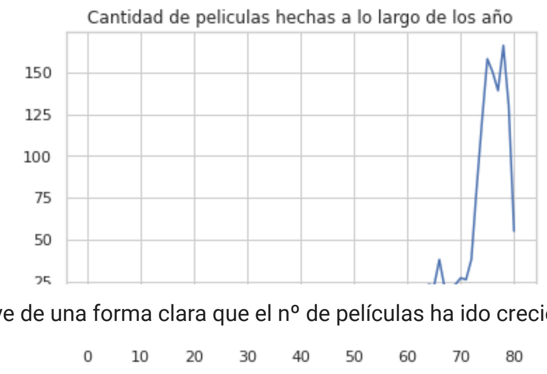
Convertir el índice en una columna y cuento por año:

```
genres_year = df.iloc[:, 3:].groupby("year").sum().loc[:,:]

genres_year = genres_year.reset_index()
genres_year.head(3)

```

	year	Anyo	Action	Adventure	Animation	Children's	Comedy	Crime	Documentary	Drama	Fantasy	Film-Noir	Horror	Musical
0	1919	0.0	1	1	0	0	1	0	0	2	0	0	0	0
1	1920	0.0	0	0	0	0	2	0	0	0	0	0	0	0

```
# El gráfico para un solo estilo podría ser este:

ax= genres_year['Drama'].plot()
ax.set_title('Cantidad de peliculas hechas a lo largo de los año')
```

```
Text(0.5, 1.0, 'Cantidad de peliculas hechas a lo largo de los año')
```



Se ve de una forma clara que el nº de películas ha ido creciendo a lo largo de los años.

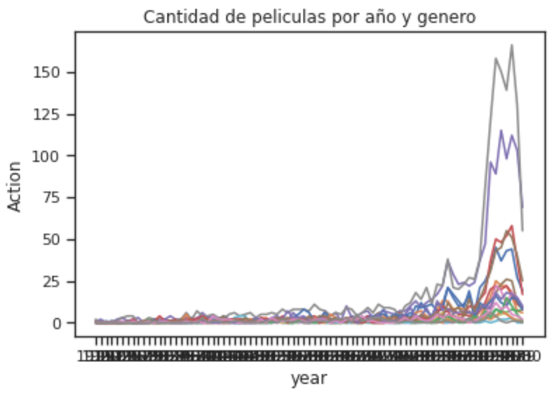
Dibujo la evolucion de todos los estilos

```
genres_year
```

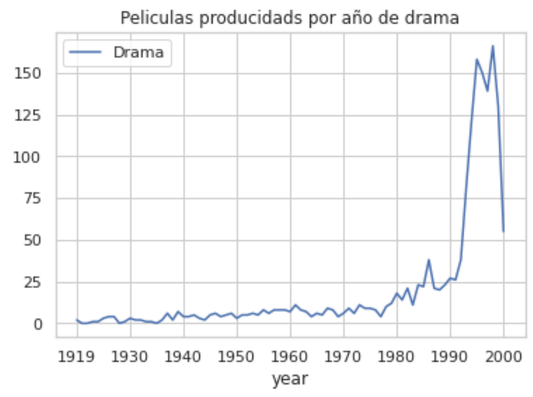
	year	Anyo	Action	Adventure	Animation	Children's	Comedy	Crime	Documentary	Drama	Fantasy	Film-Noir	Horror	Music
0	1919	0.0	1	1	0	0	1	0	0	2	0	0	0	
1	1920	0.0	0	0	0	0	2	0	0	0	0	0	0	
2	1921	0.0	1	0	0	0	0	0	0	0	0	0	0	
3	1922	0.0	0	0	0	0	0	0	0	1	0	0	1	
4	1923	0.0	0	0	0	0	2	0	0	1	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
76	1996	17610.0	37	22	7	20	115	23	19	150	5	2	12	
77	1997	0.0	43	22	6	22	98	26	11	139	6	2	10	
78	1998	0.0	44	16	8	18	112	25	18	166	2	3	15	
79	1999	0.0	27	7	7	11	103	12	15	130	2	0	14	
80	2000	0.0	19	6	8	9	69	8	8	55	1	0	8	

```
# El dibujo a lo largo de todos los años y todos los estilos
sns.figsize = ( 145, 128)
figsize = (190,80)
for x, generoA in enumerate ( genres):

  sns.lineplot(x = 'year', y = generoA, data = genres_year )


plt.title('Cantidad de peliculas por año y genero')

print('\n\nVemos que en general el nº de peliculas ha aumentado a lo largo de los años\n')
```

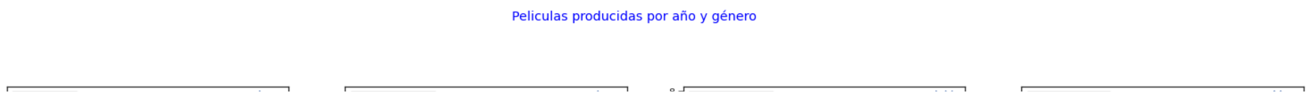
```
Vemos que en general el nº de peliculas ha aumentado
```



```
genres_year .plot('year', 'Drama')
plt.title('Peliculas producidads por año de drama')
plt.show()
```



```
# Creamos un subplot para cada estilo

fig, ax =  plt.subplots( 5, 4,
                         figsize = ( 26, 12), sharex=True)

fig.suptitle("Peliculas producidas por año y género", color='blue')
sns.set(style ='ticks')


for x, generoA in enumerate ( genres):
  axe = ax.ravel()

  genres_year.plot('year', generoA, ax=axe[x])

```



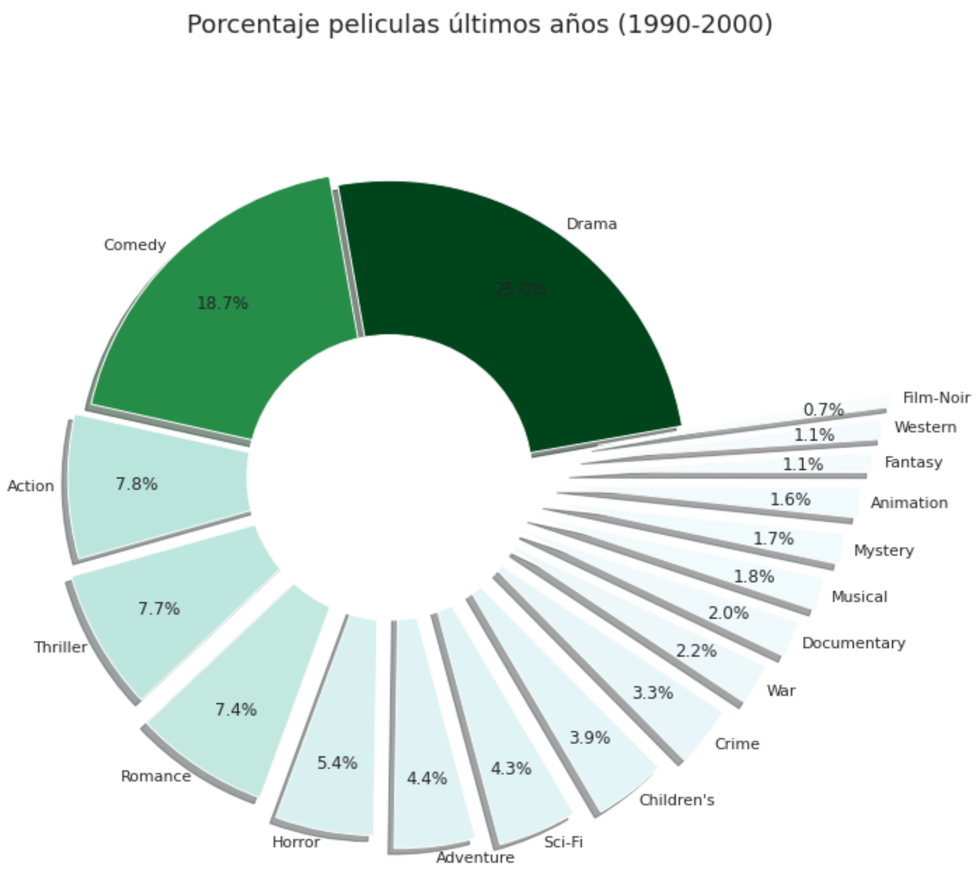
```
# Me quedo solo con los ultimos años

df2= genres_year[70:]
df2.head(2)
df10anyos = df.iloc[:, 4:].sum().sort_values(ascending=False)
df10anyos= df10anyos[1:] # Para quitarme el año como contador

print('Cantidad de películas hechas en los ultimos años')

df10anyos

```

```
Cantidad de películas hechas en los ultimos años
Drama          1603.0
Comedy         1200.0
Action          503.0
Thriller        492.0
Romance         471.0
Horror          343.0
Adventure       283.0
Sci-Fi          276.0
Children's      251.0
Crime           211.0
War             143.0
Documentary     127.0
Musical         114.0
Mystery         106.0
Animation       105.0
Fantasy          68.0
Western          68.0
Film-Noir        44.0
dtype: float64
```

```
import matplotlib as mpl
```

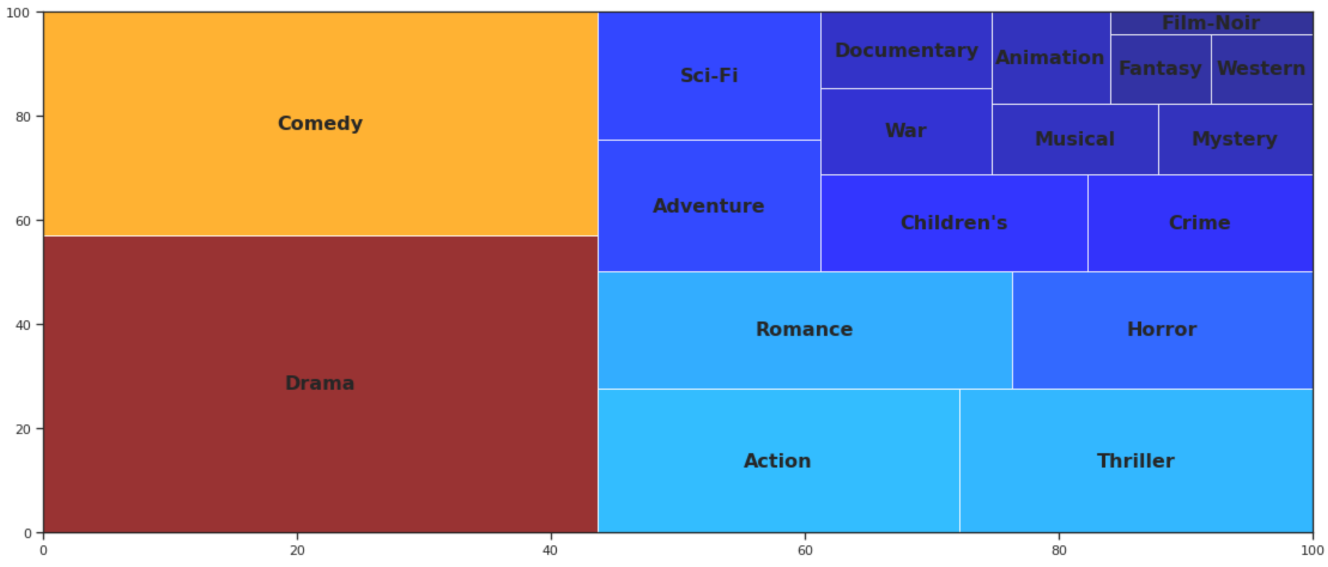
```
# Creo Pie Chart para ver distribucion por generos en los ultimo s años

etiquetas = df10anyos.index
explode = [i/24 for i in range(0,len(df10anyos))]


fig1, ax1 = plt.subplots(figsize=(10,10))

normdata = mpl.colors.Normalize(min(df10anyos), max(df10anyos))
colormap = mpl.cm.BuGn         # Esta linea nos permite cambiar el color
colors =colormap(normdata(df10anyos))


ax1.pie(df10anyos, explode=explode, labels=etiquetas, autopct='%.1f%%',
        pctdistance=0.77,
        labeldistance=1.04,
        shadow=True,
        startangle=10,
        colors=colors)
ax1.set_title("Porcentaje peliculas últimos años (1990-2000)",fontsize=18)

ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

#draw inner circle
centre_circle = plt.Circle((0,0),.48,fc='white')

fig = plt.gcf()

fig.gca().add_artist(centre_circle)

plt.tight_layout()
```

```
#plt.legend(title = "Genres:",labels=labels,loc="right",ncol=3)
plt.show()
```



```
# Dibujo un Treemap

import squarify
import matplotlib as mpl

normdata = mpl.colors.Normalize(min(df10anyos.values), max(df10anyos.values))
colormap = mpl.cm.jet
colors =colormap(normdata(df10anyos.values))

fig = plt.gcf()
ax = fig.add_subplot()
fig.set_size_inches(19, 8)

squarify.plot(sizes=df10anyos.values, alpha=0.8, label=df10anyos.index,color = colors,
              text_kwargs={'fontsize':16, 'fontname':"helvetica",'weight':'bold'})



```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe04b493590>findfont: Font family ['helvetica'] not found. Falling back to Deja
```



✓ 5 s completado a las 7:04