In [1]:

```
#S04 T02: Visualització gràfica de Múltiples variables
```

In [2]:

```
#Nivell 1
```

In [3]:

```
#Exercici 1
#Realitza la pràctica del notebook a GitHub "03 EXAMINING DATA" amb seaborn i el datase
t "tips".
#Llibreries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
import statistics
import warnings

warnings.filterwarnings('ignore') #Desactivació de Warnings
```

```
The history saving thread hit an unexpected error (OperationalError('disk
I/O error')).History will not be written to the database.
```

In [4]:

```
#Accès a les dades del fitxer tips.csv
tips_df=sns.load_dataset("tips")
tips_df.head(n=10)
```

Out[4]:

	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
5	25.29	4.71	Male	No	Sun	Dinner	4
6	8.77	2.00	Male	No	Sun	Dinner	2
7	26.88	3.12	Male	No	Sun	Dinner	4
8	15.04	1.96	Male	No	Sun	Dinner	2
9	14.78	3.23	Male	No	Sun	Dinner	2

In [5]:

```
#Dimensions
tips_df.shape
```

Out[5]:

```
(244, 7)
```

In [6]:

```
#Tipus de columnes
tips_df.dtypes
```

Out[6]:

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

In [7]:

```
tips_df['day'].unique
```

Out[7]:

```
<bound method Series.unique of 0       Sun
1       Sun
2       Sun
3       Sun
4       Sun
       ...
239     Sat
240     Sat
241     Sat
242     Sat
243    Thur
Name: day, Length: 244, dtype: category
Categories (4, object): [Thur, Fri, Sat, Sun]>
```

In [8]:

```
tips_df.dtypes['day']
```

Out[8]:

```
CategoricalDtype(categories=['Thur', 'Fri', 'Sat', 'Sun'], ordered=False)
```

In [9]:

```
#Columnes del dataframe
tips_df.columns
```

Out[9]:

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

In [10]:

```
#Array de valors del dataframe
tips_df.values
```

Out[10]:

```
array([[16.99, 1.01, 'Female', ..., 'Sun', 'Dinner', 2],
       [10.34, 1.66, 'Male', ..., 'Sun', 'Dinner', 3],
       [21.01, 3.5, 'Male', ..., 'Sun', 'Dinner', 3],
       ...,
       [22.67, 2.0, 'Male', ..., 'Sat', 'Dinner', 2],
       [17.82, 1.75, 'Male', ..., 'Sat', 'Dinner', 2],
       [18.78, 3.0, 'Female', ..., 'Thur', 'Dinner', 2]], dtype=object)
```

In [11]:

```
#Array de valors d'una fila del dataframe
tips_df.values[10]
```

Out[11]:

```
array([10.27, 1.71, 'Male', 'No', 'Sun', 'Dinner', 2], dtype=object)
```

In [12]:

```
#Llista d'axes del dataframe
tips_df.axes
```

Out[12]:

```
[RangeIndex(start=0, stop=244, step=1),
 Index(['total_bill', 'tip', 'sex', 'smoker', 'day', 'time', 'size'], dtyp
e='object')]
```

In [13]:

```
tips_df.describe()
```

Out[13]:

	total_bill	tip	size
count	244.000000	244.000000	244.000000
mean	19.785943	2.998279	2.569672
std	8.902412	1.383638	0.951100
min	3.070000	1.000000	1.000000
25%	13.347500	2.000000	2.000000
50%	17.795000	2.900000	2.000000
75%	24.127500	3.562500	3.000000
max	50.810000	10.000000	6.000000

In [14]:

```
#Indexes
tips_df.index
```

Out[14]:

```
RangeIndex(start=0, stop=244, step=1)
```

In [15]:

```
#Number of dimensions
tips_df.ndim
```

Out[15]:

```
2
```

In [16]:

```
#Size of dimensions
tips_df.size
```

Out[16]:

```
1708
```

In [17]:

```
#Dimensió de la memoria
tips_df.memory_usage(index=True, deep=False)
```

Out[17]:

```
Index          128
total_bill    1952
tip           1952
sex            340
smoker         340
day            436
time           340
size          1952
dtype: int64
```

In [18]:

```
tips_df.memory_usage(index=True, deep=True)
```

Out[18]:

```
Index          128
total_bill    1952
tip           1952
sex            448
smoker         443
day            645
time           449
size          1952
dtype: int64
```

In [19]:

```
print("Value of mean",tips_df['total_bill'].mean())
```

```
Value of mean 19.785942622950824
```

In [20]:

```
#Desviació standard de la columna total_bill
statistics.pstdev(tips_df['total_bill']) # llibreria statistics
```

Out[20]:

```
8.884150577771132
```

In [21]:

```
#Desviació standard de la columna total_bill
np.std(tips_df['total_bill']) # llibreria numpy
```

Out[21]:

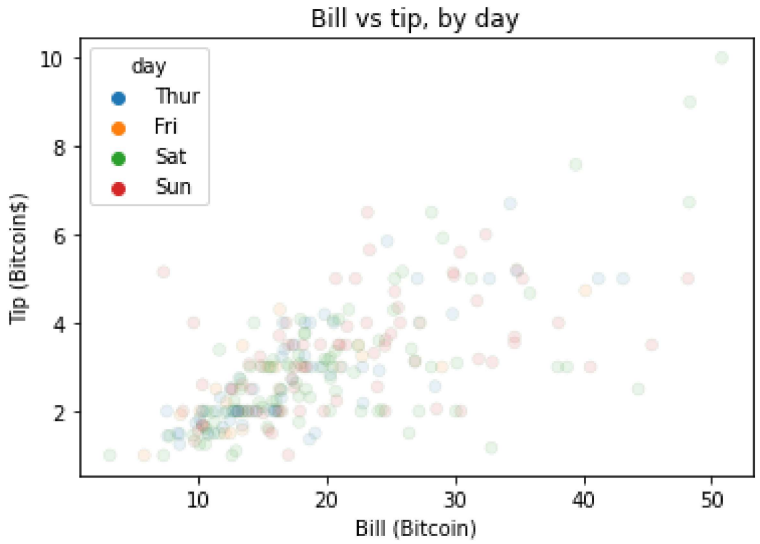
```
8.88415057777113
```

In [22]:

```
#Serie [seaborn] scatterplot: by day
fig, ax=plt.subplots()
sct=sns.scatterplot(data=tips_df, x='total_bill', y='tip', hue='day', ax=ax, edgecolor=
'darkgreen'
                    ,alpha=.10)
ax.set_title('Bill vs tip, by day')
ax.set_xlabel('Bill (Bitcoin)')
ax.set_ylabel('Tip (Bitcoin$)')
```

Out[22]:

```
Text(0, 0.5, 'Tip (Bitcoin$)')
```

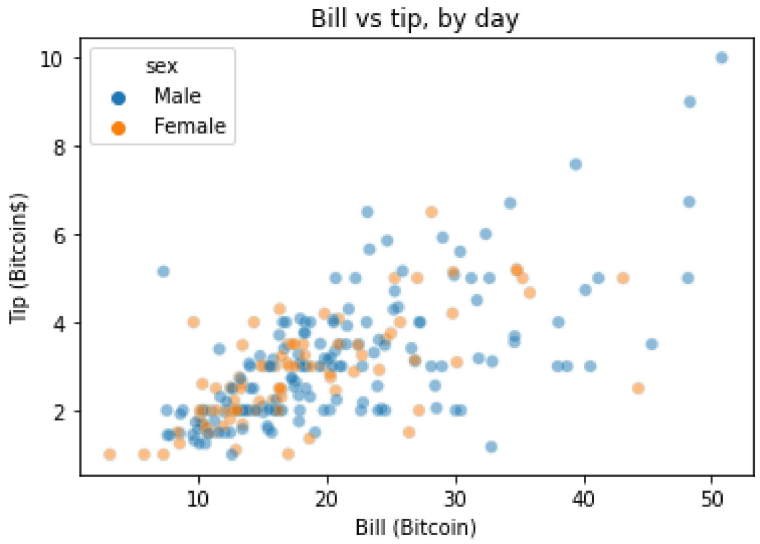


In [23]:

```
#Serie [seaborn] scatterplot: by sex
fig, ax=plt.subplots()
sct=sns.scatterplot(data=tips_df, x='total_bill', y='tip', hue='sex', ax=ax, edgecolor=
'lightblue'
                    ,alpha=.5)
ax.set_title('Bill vs tip, by day')
ax.set_xlabel('Bill (Bitcoin)')
ax.set_ylabel('Tip (Bitcoin$)')
```

Out[23]:

```
Text(0, 0.5, 'Tip (Bitcoin$)')
```

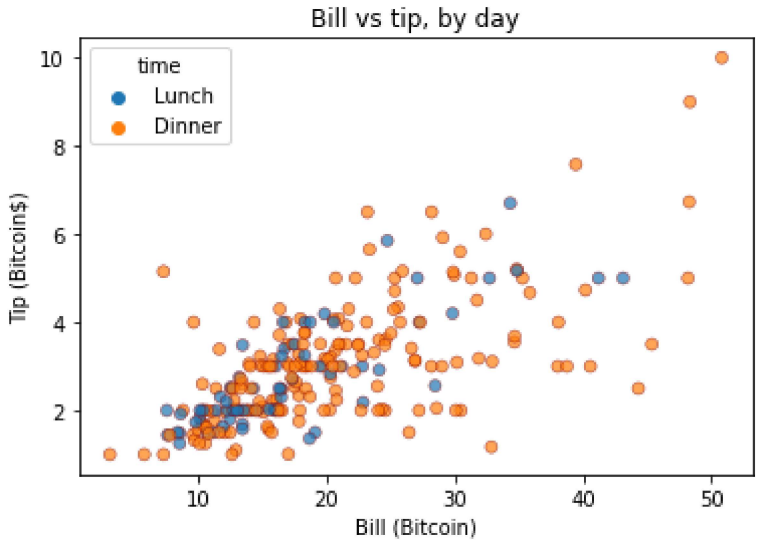


In [24]:

```
#Serie [seaborn] scatterplot: by time
fig, ax=plt.subplots()
sct=sns.scatterplot(data=tips_df, x='total_bill', y='tip', hue='time', ax=ax, edgecolor
='darkred'
                ,alpha=.7)
ax.set_title('Bill vs tip, by day')
ax.set_xlabel('Bill (Bitcoin)')
ax.set_ylabel('Tip (Bitcoin$)')
```

Out[24]:

```
Text(0, 0.5, 'Tip (Bitcoin$)')
```

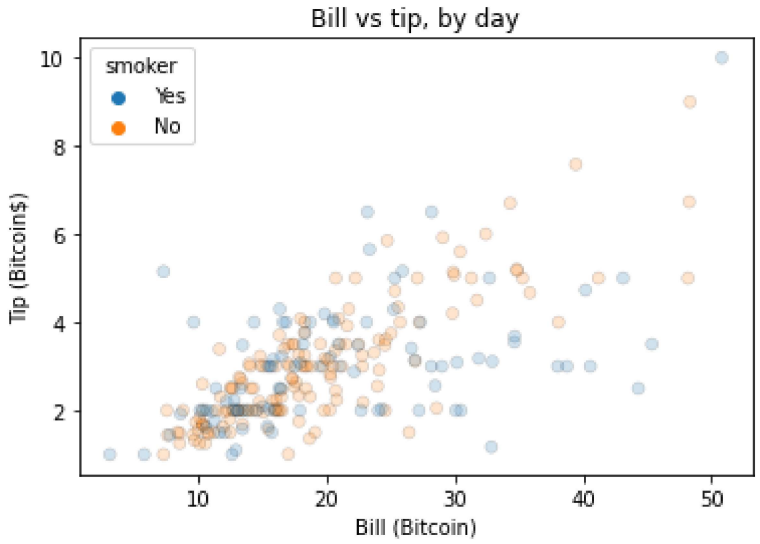


In [25]:

```
#Serie [seaborn] scatterplot: by smoker
fig, ax=plt.subplots()
sct=sns.scatterplot(data=tips_df, x='total_bill', y='tip', hue='smoker', ax=ax, edgecol
or='black'
                ,alpha=.2)
ax.set_title('Bill vs tip, by day')
ax.set_xlabel('Bill (Bitcoin)')
ax.set_ylabel('Tip (Bitcoin$)')
```

Out[25]:

```
Text(0, 0.5, 'Tip (Bitcoin$)')
```

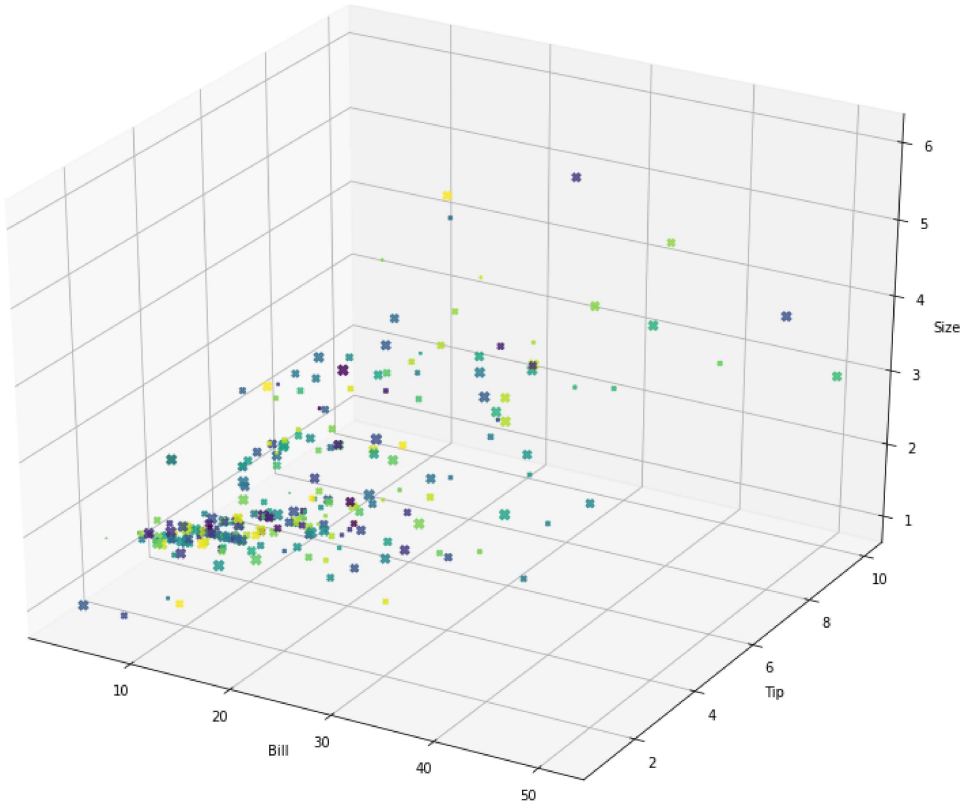


In [26]:

```
#Serie [matplotlib] scatterplot:
N=tips_df.shape[0]
colors = np.random.rand(N)
area = (30 * np.random.rand(N))*2

fig = plt.figure(figsize=(15,12))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(tips_df['total_bill'],
           tips_df['tip'],
           tips_df['size'],
           s=area,
           marker='X',
           c=colors,alpha=.8)

ax.set_xlabel('Bill')
ax.set_ylabel('Tip')
ax.set_zlabel('Size')
plt.show()
```

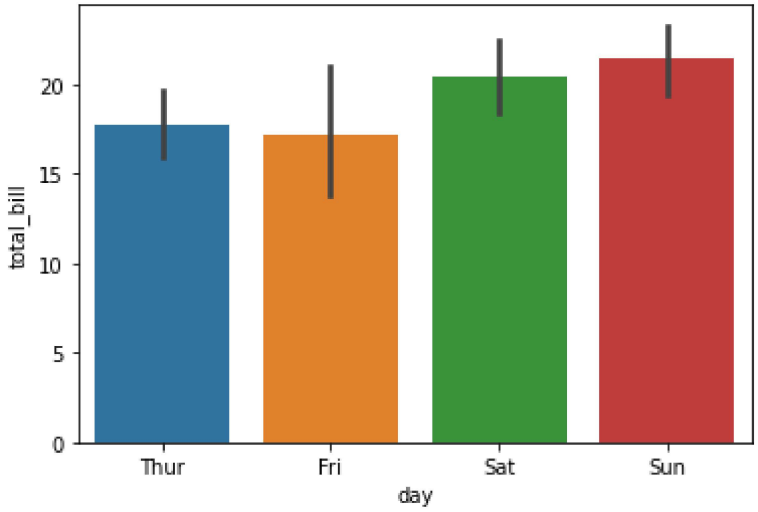


In [27]:

```
#Serie [seaborn] barplot: by day
tips_df=sns.load_dataset("tips")
sns.barplot(data=tips_df, x="day", y="total_bill")
```

Out[27]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2562ab93d60>
```

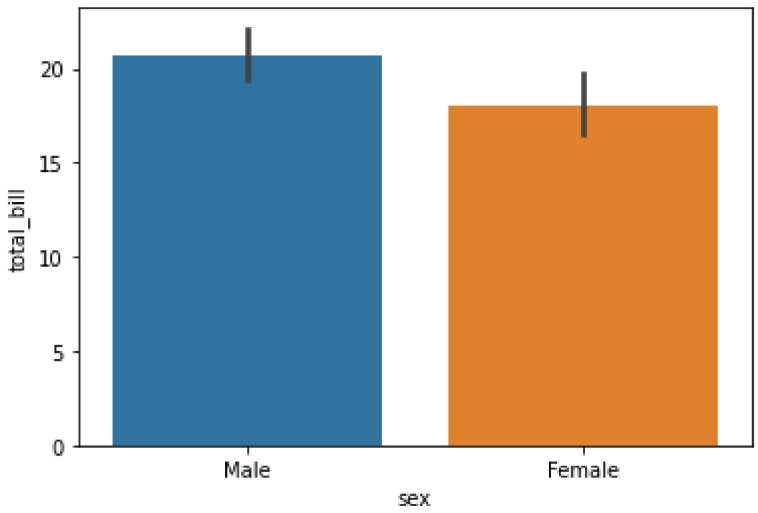


In [28]:

```
#Serie [seaborn] barplot: by sex
tips_df=sns.load_dataset("tips")
sns.barplot(data=tips_df, x="sex", y="total_bill")
```

Out[28]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2562ac71a60>
```

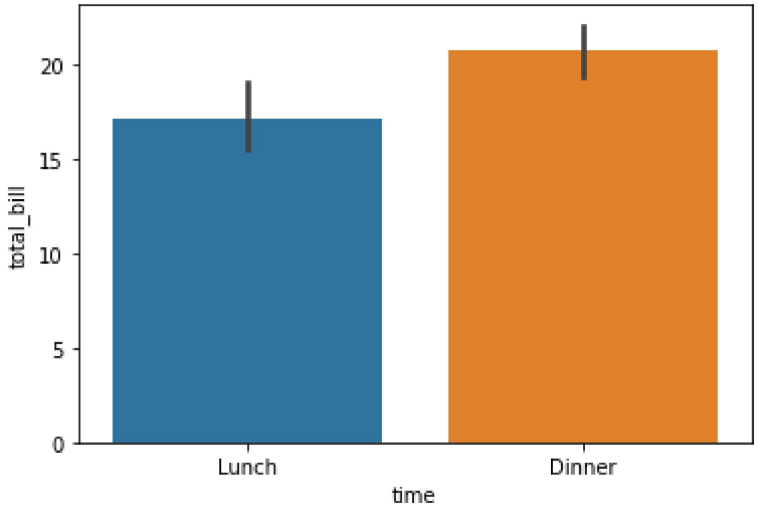


In [29]:

```
#Serie [seaborn] barplot: by time
tips_df=sns.load_dataset("tips")
sns.barplot(data=tips_df, x="time", y="total_bill")
```

Out[29]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2562acb66a0>
```

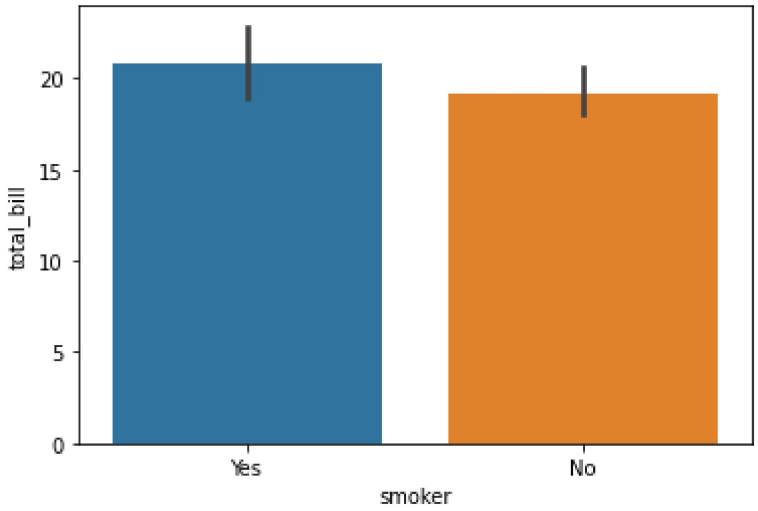


In [30]:

```
#Serie [seaborn] barplot: by smoker
tips_df=sns.load_dataset("tips")
sns.barplot(data=tips_df, x="smoker", y="total_bill")
```

Out[30]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2562ad00e50>
```



In [31]:

```
#Nivell 2
```

In [32]:

```
#Exercici 2
#Repeteix l'exercici 1 amb el dataset que disposem en el repositori de GitHub PRE-PROCE
SSING-DATA, movies.dat
#Llibreries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

warnings.filterwarnings('ignore') #Desactivació de Warnings
```

In [33]:

```
#Accès a les dades del fitxer movies.dat
movies_df=pd.read_csv("Python/movies.dat", sep="::")
movies_df.head(n=8)
```

Out[33]:

	n	film (year)	category
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
3	4	Waiting to Exhale (1995)	Comedy|Drama
4	5	Father of the Bride Part II (1995)	Comedy
5	6	Heat (1995)	Action|Crime|Thriller
6	7	Sabrina (1995)	Comedy|Romance
7	8	Tom and Huck (1995)	Adventure|Children's

In [34]:

```
#Tipus de columnes
movies_df.dtypes
```

Out[34]:

```
n               int64
film (year)    object
category       object
dtype: object
```

In [35]:

```
#Columna 'film (year)' té 2 dades: film & year. Es recomana tractar-ho individualment.
#Mitjançant 2 sentencies de Python es pot obtenir 2 columnes diferenciades ('film' i 'y
ear').
movies_df[['film', 'year']] = movies_df['film (year)'].str.split('(', 1, expand=True)
movies_df['year'] = movies_df['year'].str.split(')', 1, expand=True)
```

In [58]:

```
#Columna 'category' té fins a 3 valors: film & year. Es recomana tractar aquests valors individualment.
#Mitjançant 2 sentencies de Python es pot obtenir 3 columnes diferenciades ('genre1','genre2' i 'genre3').
movies_df[['genre1', 'genre2','genre3']] = movies_df['category'].str.split('|', 2, expand=True)
```

In [59]:

```
#Tipus de columnes ampliat
movies_df.dtypes
```

Out[59]:

```
n            int64
category    object
film        object
year        object
genre1      object
genre2      object
genre3      object
dtype: object
```

In [38]:

```
movies_df['film'].head(n=20)
```

Out[38]:

```
0                           Toy Story
1                             Jumanji
2                    Grumpier Old Men
3                   Waiting to Exhale
4         Father of the Bride Part II
5                                Heat
6                             Sabrina
7                        Tom and Huck
8                        Sudden Death
9                            GoldenEye
10            American President, The
11       Dracula: Dead and Loving It
12                               Balto
13                               Nixon
14                  Cutthroat Island
15                             Casino
16              Sense and Sensibility
17                          Four Rooms
18    Ace Ventura: When Nature Calls
19                          Money Train
Name: film, dtype: object
```

In [39]:

```
movies_df['year'].tail(n=10)
```

Out[39]:

```
3873    2000
3874    2000
3875    2000
3876    2000
3877    1971
3878    2000
3879    2000
3880    2000
3881    2000
3882    2000
Name: year, dtype: object
```

In [40]:

```
#Converteixo l'Object 'year' en string
movies_df['year'] = movies_df['year'].astype(str)
```

In [41]:

```
movies_df.dtypes
```

Out[41]:

```
n               int64
film (year)    object
category       object
film           object
year           object
genre1         object
genre2         object
dtype: object
```

In [61]:

```
movies_df['genre1'].head(n=6)
```

Out[61]:

```
0    Animation
1    Adventure
2       Comedy
3       Comedy
4       Comedy
5       Action
Name: genre1, dtype: object
```

In [62]:

```
movies_df['genre2'].head(n=6)
```

Out[62]:

```
0    Children's
1    Children's
2       Romance
3         Drama
4          None
5         Crime
Name: genre2, dtype: object
```

In [63]:

```
movies_df['genre3'].head(n=6)
```

Out[63]:

```
0      Comedy
1     Fantasy
2        None
3        None
4        None
5    Thriller
Name: genre3, dtype: object
```

In [42]:

```
#Dimensions
movies_df.shape
```

Out[42]:

```
(3883, 7)
```

In [43]:

```
#La columna 'film (year)' la podem esborrar del dataframe
movies_df.drop ('film (year)', axis=1, inplace=True)
```

In [126]:

```
#La columna 'category' la podem esborrar del dataframe
movies_df.drop ('category', axis=1, inplace=True)
```

In [127]:

```
#Tipus de columnes actualitzat
movies_df.dtypes
```

Out[127]:

```
n          int64
film      object
year      object
genre1    object
genre2    object
genre3    object
dtype: object
```

In [45]:

```
#Columnes del dataframe
movies_df.columns
```

Out[45]:

```
Index(['n', 'category', 'film', 'year', 'genre1', 'genre2'], dtype='objec
t')
```

In [46]:

```
#Array de valors del dataframe
movies_df.values
```

Out[46]:

```
array([[1, "Animation|Children's|Comedy", 'Toy Story ', '1995',
        'Animation', "Children's|Comedy"],
       [2, "Adventure|Children's|Fantasy", 'Jumanji ', '1995',
        'Adventure', "Children's|Fantasy"],
       [3, 'Comedy|Romance', 'Grumpier Old Men ', '1995', 'Comedy',
        'Romance'],
       ...,
       [3950, 'Drama', 'Tigerland ', '2000', 'Drama', None],
       [3951, 'Drama', 'Two Family House ', '2000', 'Drama', None],
       [3952, 'Drama|Thriller', 'Contender, The ', '2000', 'Drama',
        'Thriller']], dtype=object)
```

In [64]:

```
#Llista d'axes del dataframe
movies_df.axes
```

Out[64]:

```
[RangeIndex(start=0, stop=3883, step=1),
 Index(['n', 'category', 'film', 'year', 'genre1', 'genre2', 'genre3'], dt
ype='object')]
```

In [49]:

```
movies_df.describe().round(2)
```

Out[49]:

	n
count	3883.00
mean	1986.05
std	1146.78
min	1.00
25%	982.50
50%	2010.00
75%	2980.50
max	3952.00

In [50]:

```
#Indexes
movies_df.index
```

Out[50]:

```
RangeIndex(start=0, stop=3883, step=1)
```

In [51]:

```
#Number of dimensions
movies_df.ndim
```

Out[51]:

```
2
```

In [52]:

```
#Size of dimensions
movies_df.size
```

Out[52]:

```
23298
```

In [80]:

```
#Dimensió de la memoria
movies_df.memory_usage(index=False, deep=True)
```

Out[80]:

```
n            31064
category    283297
film        286827
year        242125
genre1      264073
genre2      166965
genre3      115419
dtype: int64
```

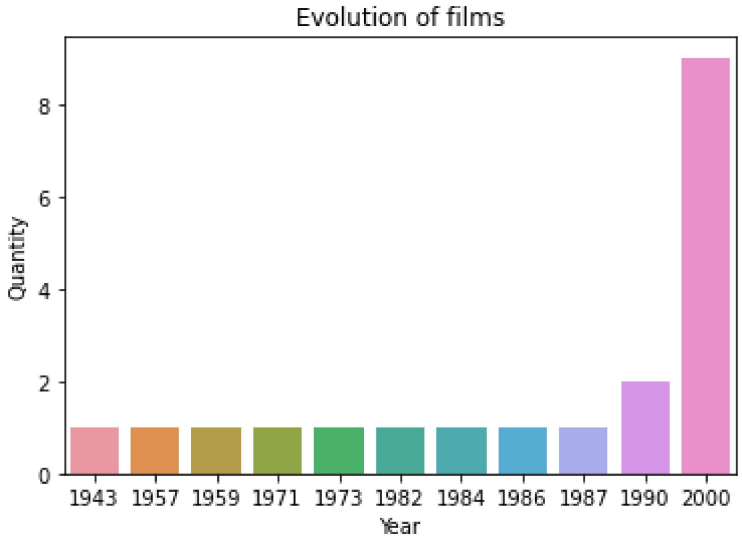
In [124]:

```
#Serie [seaborn] countplot: by year

fig, ax=plt.subplots()
sns.countplot(data=movies_df.tail(n=20).sort_values(by='year'), x='year')
ax.set_title('Evolution of films')
ax.set_xlabel('Year')
ax.set_ylabel('Quantity')
```

Out[124]:

```
Text(0, 0.5, 'Quantity')
```



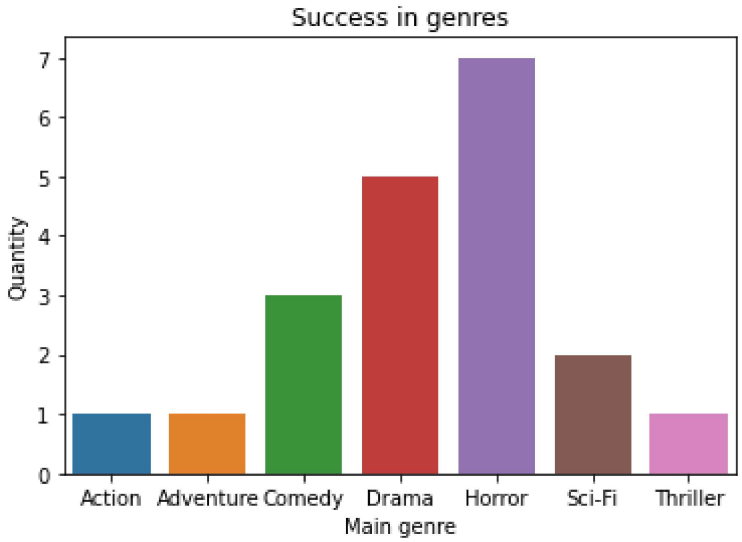
In [125]:

```
#Serie [seaborn] countplot: by genre1

fig, ax=plt.subplots()
sns.countplot(data=movies_df.tail(n=20).sort_values(by='genre1'), x='genre1')
ax.set_title('Success in genres')
ax.set_xlabel('Main genre')
ax.set_ylabel('Quantity')
```

Out[125]:

```
Text(0, 0.5, 'Quantity')
```



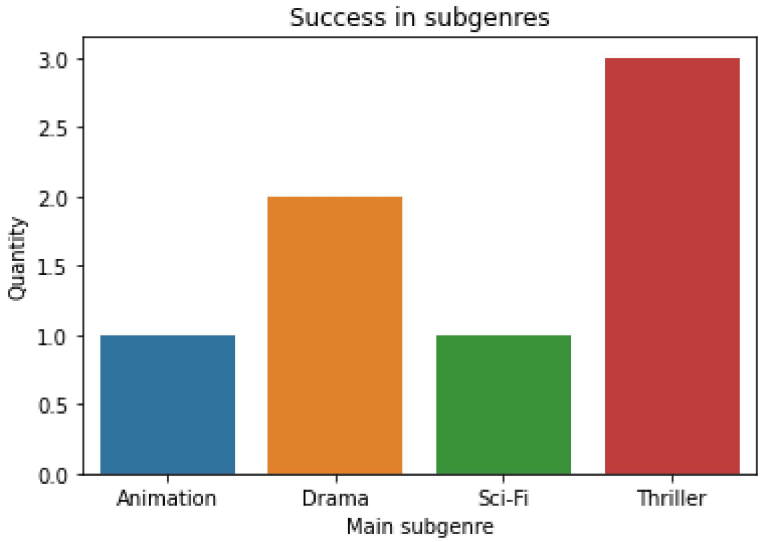
In [128]:

```
#Serie [seaborn] countplot: by genre2

fig, ax=plt.subplots()
sns.countplot(data=movies_df.tail(n=20).sort_values(by='genre2'), x='genre2')
ax.set_title('Success in subgenres')
ax.set_xlabel('Main subgenre')
ax.set_ylabel('Quantity')
```

Out[128]:

```
Text(0, 0.5, 'Quantity')
```



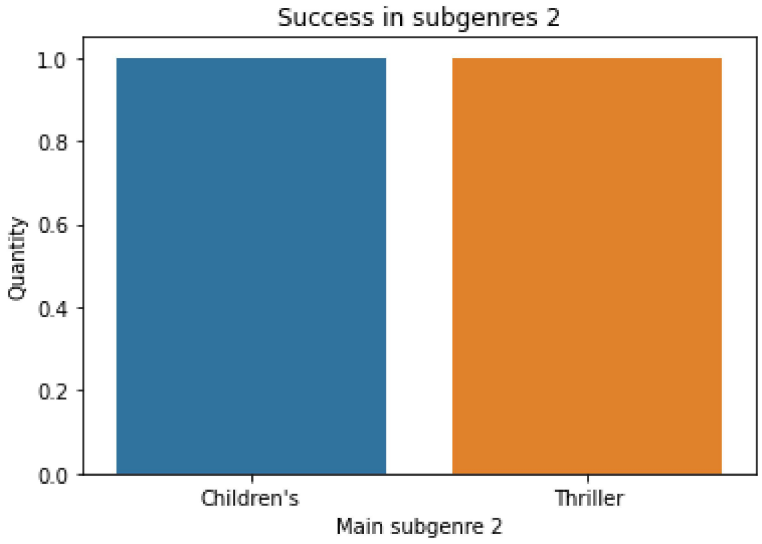
In [129]:

```
#Serie [seaborn] countplot: by genre3

fig, ax=plt.subplots()
sns.countplot(data=movies_df.tail(n=20).sort_values(by='genre3'), x='genre3')
ax.set_title('Success in subgenres 2')
ax.set_xlabel('Main subgenre 2')
ax.set_ylabel('Quantity')
```

Out[129]:

```
Text(0, 0.5, 'Quantity')
```



In [123]:

```
#Nivell 3
```

In [ ]:

```
#Exercici 3
#En aquest exercici no us donarem gaires indicacions perquè volem que ens mostreu la vo
stra creativitat.
#Sorprèn-me amb gràfiques i interpretacions del dataset "movies.dat" del exercici anter
ior.
#Llibreries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

warnings.filterwarnings('ignore') #Desactivació de Warnings
```

In [40]:

```
#Accès a les dades del fitxer movies.dat
movies=pd.read_csv("Python/movies.dat", sep="::", engine='python')
movies.head(n=14)
```

Out[40]:

	n	film (year)	category
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
3	4	Waiting to Exhale (1995)	Comedy|Drama
4	5	Father of the Bride Part II (1995)	Comedy
5	6	Heat (1995)	Action|Crime|Thriller
6	7	Sabrina (1995)	Comedy|Romance
7	8	Tom and Huck (1995)	Adventure|Children's
8	9	Sudden Death (1995)	Action
9	10	GoldenEye (1995)	Action|Adventure|Thriller
10	11	American President, The (1995)	Comedy|Drama|Romance
11	12	Dracula: Dead and Loving It (1995)	Comedy|Horror
12	13	Balto (1995)	Animation|Children's
13	14	Nixon (1995)	Drama

In [41]:

```
#Dimensions
movies.shape
```

Out[41]:

```
(3883, 3)
```

In [42]:

```
#Columnes del dataframe
movies.columns
```

Out[42]:

```
Index(['n', 'film (year)', 'category'], dtype='object')
```

In [43]:

```
#Array de valors del dataframe
movies.values
```

Out[43]:

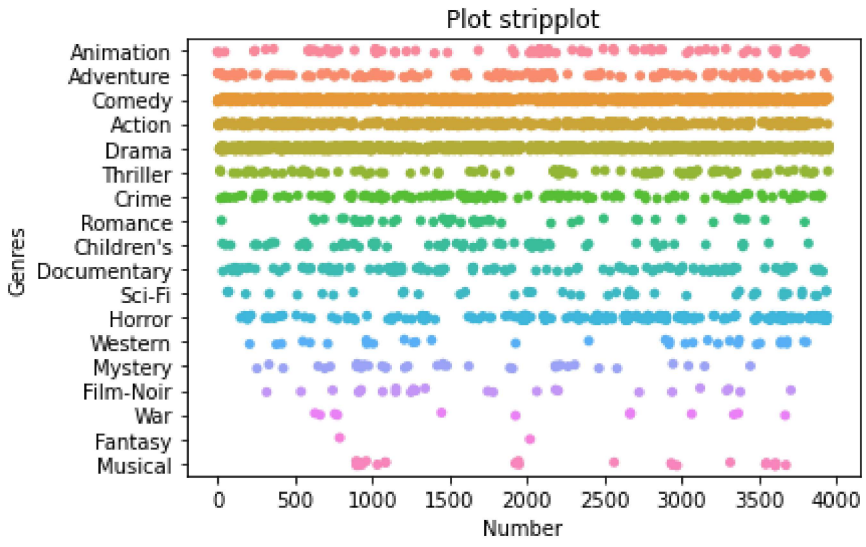
```
array([[1, 'Toy Story (1995)', "Animation|Children's|Comedy"],
       [2, 'Jumanji (1995)', "Adventure|Children's|Fantasy"],
       [3, 'Grumpier Old Men (1995)', 'Comedy|Romance'],
       ...,
       [3950, 'Tigerland (2000)', 'Drama'],
       [3951, 'Two Family House (2000)', 'Drama'],
       [3952, 'Contender, The (2000)', 'Drama|Thriller']], dtype=object)
```

In [64]:

```
#Capturem el primer item de la columna 'Category'. Exemple: 'Animation|Children's|Comed
y' => 'Animation'
movies[['genre','genre_rest']] = movies['category'].str.split('|', 1, expand=True)
```

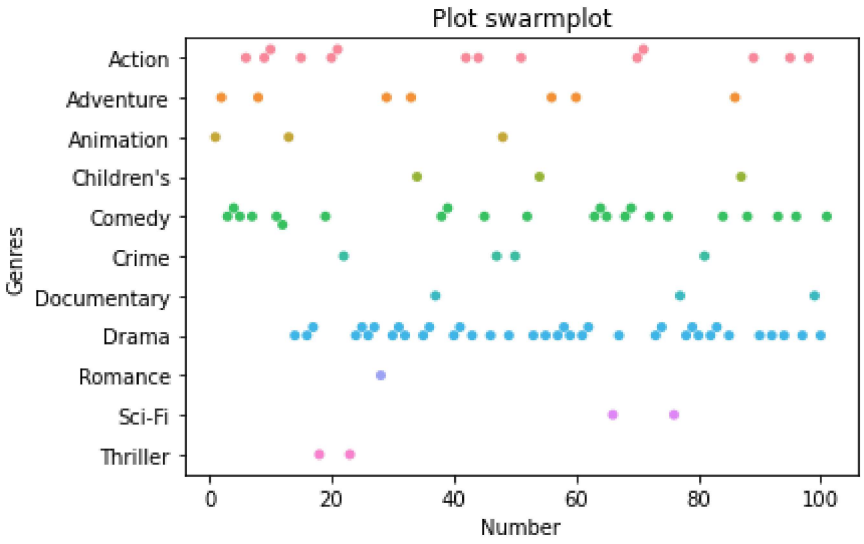
In [98]:

```
#Serie (seaborn) stripplot: by genre
fig, ax=plt.subplots()
sns.stripplot(data=movies, x="n", y="genre")
ax.set_title('Plot stripplot')
ax.set_xlabel('Number')
ax.set_ylabel('Genres')
ax.height=150
ax.width=150
```



In [99]:

```
#Serie (seaborn) swarmplot: by genre
fig, ax=plt.subplots()
sns.swarmplot(data=movies.head(n=100).sort_values(by='category'), x="n", y="genre")
ax.set_title('Plot swarmplot')
ax.set_xlabel('Number')
ax.set_ylabel('Genres')
ax.height=150
ax.width=150
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: