Creado por:

Isabel Maniega

Titanic Dataset - Predicción

Para competir en Kaggle será necesario descargar de esta página los csv de: train.csv, test.csv, gender_submission.csv

https://www.kaggle.com/competitions/titanic/data

In [1]:
```
# pip install seaborn
```

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Importamos el dataset train.csv

In [3]:
```
df = pd.read_csv("./data/train.csv")
df.head()
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	

Borramos la columna de PassengerId

In [4]:
```
df = df.drop("PassengerId", axis=1)
df
```

Out[4]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cab
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	Na
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	Na
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C1
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	Na
...	...	...	...	...	...	...	...	...	...	
886	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	Na
887	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B
888	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	Na
889	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C1
890	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	Na

891 rows × 11 columns

Exploratory Data Analysis (EDA)

In [5]:
```
df.tail()
```

Out[5]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	E
886	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	
887	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	
888	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	
889	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	
890	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	

In [6]:
```
len(df)
```

Out[6]: 891

In [7]:
```
df.shape
```

Out[7]: (891, 11)

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

Out[8]:

	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Conclusiones:

- Existen columnas de "missing values" (Valores que faltan)

In [9]:
```
# y aqui vemos cuantas columnas tiene valores que faltan
df.isnull().sum()
```

Out[9]:
```
Survived      0
Pclass        0
Name          0
Sex           0
Age         177
SibSp         0
Parch         0
Ticket        0
Fare          0
Cabin       687
Embarked      2
dtype: int64
```

In [10]:
```
df.Cabin.value_counts()
```

Out[10]:
```
Cabin
B96 B98        4
G6             4
C23 C25 C27    4
C22 C26        3
F33            3
              ..
E34            1
C7             1
C54            1
E36            1
C148           1
Name: count, Length: 147, dtype: int64
```

In [11]:
```
for cabina in df.Cabin:
    print(cabina)
```

```
nan
C85
nan
C123
nan
nan
E46
nan
nan
nan
G6
C103
nan
nan
nan
nan
nan
nan
nan
nan
nan
D56
nan
A6
nan
nan
nan
C23 C25 C27
nan
nan
nan
B78
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
D33
nan
B30
C52
nan
nan
nan
nan
```

```
nan
B28
C83
nan
nan
nan
F33
nan
nan
nan
nan
nan
nan
nan
nan
F G73
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
C23 C25 C27
nan
nan
nan
E31
nan
nan
nan
A5
D10 D12
nan
nan
nan
nan
D26
nan
nan
nan
nan
nan
nan
nan
C110
nan
nan
nan
nan
nan
nan
nan
B58 B60
nan
```

```
nan
nan
nan
E101
D26
nan
nan
nan
F E69
nan
nan
nan
nan
nan
nan
nan
D47
C123
nan
B86
nan
nan
nan
nan
nan
nan
nan
nan
F2
nan
nan
C2
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
E33
nan
nan
nan
B19
nan
nan
nan
A7
nan
nan
C49
nan
nan
```

```
nan
nan
nan
F4
nan
A32
nan
nan
nan
nan
nan
nan
nan
nan
F2
B4
B80
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
G6
nan
nan
nan
A31
nan
nan
nan
nan
nan
nan
D36
nan
nan
D15
nan
nan
nan
nan
nan
nan
C93
nan
nan
nan
nan
nan
nan
C83
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
```

```
nan
nan
nan
nan
nan
C78
nan
nan
D35
nan
nan
G6
C87
nan
nan
nan
nan
B77
nan
nan
nan
nan
E67
B94
nan
nan
nan
nan
C125
C99
nan
nan
nan
C118
nan
D7
nan
nan
nan
nan
nan
nan
nan
nan
A19
nan
nan
nan
nan
nan
nan
nan
B49
D
nan
nan
nan
nan
C22 C26
C106
B58 B60
```

```
nan
nan
nan
E101
nan
C22 C26
nan
C65
nan
E36
C54
B57 B59 B63 B66
nan
nan
nan
nan
nan
nan
C7
E34
nan
nan
nan
nan
nan
C32
nan
D
nan
B18
nan
C124
C91
nan
nan
nan
C2
E40
nan
T
F2
C23 C25 C27
nan
nan
nan
F33
nan
nan
nan
nan
nan
C128
nan
nan
nan
nan
E33
nan
nan
nan
```

```
nan
nan
nan
nan
nan
nan
D37
nan
nan
B35
E50
nan
nan
nan
nan
nan
nan
C82
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
B96 B98
nan
nan
D36
G6
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
C78
nan
nan
nan
nan
nan
nan
nan
```

```
nan
nan
nan
nan
nan
nan
nan
nan
nan
E10
C52
nan
nan
nan
E44
B96 B98
nan
nan
C23 C25 C27
nan
nan
nan
nan
nan
nan
nan
A34
nan
nan
nan
C104
nan
nan
C111
C92
nan
nan
E38
D21
nan
nan
E12
nan
E63
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
D
nan
A14
nan
nan
nan
nan
```

```
nan
nan
nan
nan
B49
nan
C93
B37
nan
nan
nan
nan
C30
nan
nan
nan
D20
nan
C22 C26
nan
nan
nan
nan
nan
B79
C65
nan
nan
nan
nan
nan
nan
E25
nan
nan
D46
F33
nan
nan
nan
B73
nan
nan
B18
nan
nan
nan
C95
nan
nan
nan
nan
nan
nan
nan
nan
nan
B38
nan
nan
B39
```

```
B22
nan
nan
nan
C86
nan
nan
nan
nan
nan
C70
nan
nan
nan
nan
nan
A16
nan
E67
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
C101
E25
nan
nan
nan
nan
E44
nan
nan
nan
C68
nan
A10
nan
E68
nan
B41
nan
nan
nan
D20
nan
nan
nan
nan
nan
nan
nan
A20
```

```
nan
nan
nan
nan
nan
nan
nan
nan
nan
C125
nan
nan
nan
nan
nan
nan
nan
nan
F4
nan
nan
D19
nan
nan
nan
D50
nan
D9
nan
nan
A23
nan
B50
nan
nan
nan
nan
nan
nan
nan
nan
B35
nan
nan
nan
D33
nan
A26
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
nan
D48
```

```
nan
nan
E58
nan
nan
nan
nan
nan
nan
C126
nan
B71
nan
nan
nan
nan
nan
nan
nan
B51 B53 B55
nan
D49
nan
nan
nan
nan
nan
nan
nan
B5
B20
nan
nan
nan
nan
nan
nan
nan
C68
F G63
C62 C64
E24
nan
nan
nan
nan
nan
E24
nan
nan
C90
C124
C126
nan
nan
F G73
C45
E101
nan
nan
```

```
nan
nan
nan
nan
E8
nan
nan
nan
nan
nan
B5
nan
nan
nan
nan
nan
nan
B101
nan
nan
D45
C46
B57 B59 B63 B66
nan
nan
B22
nan
nan
D30
nan
nan
E121
nan
nan
nan
nan
nan
nan
nan
B77
nan
nan
nan
B96 B98
nan
D11
nan
nan
nan
nan
nan
nan
E77
nan
nan
nan
F38
nan
nan
B3
```

```
nan
B20
D6
nan
nan
nan
nan
nan
nan
B82 B84
nan
nan
nan
nan
nan
nan
D17
nan
nan
nan
nan
nan
B96 B98
nan
nan
nan
A36
nan
nan
E8
nan
nan
nan
nan
nan
B102
nan
nan
nan
nan
B69
nan
nan
E121
nan
nan
nan
nan
nan
B28
nan
nan
nan
nan
nan
E49
nan
nan
nan
C47
```

```
nan
nan
nan
nan
nan
nan
nan
nan
nan
C92
nan
nan
nan
D28
nan
nan
nan
E17
nan
nan
nan
nan
D17
nan
nan
nan
nan
A24
nan
nan
nan
D35
B51 B53 B55
nan
nan
nan
nan
nan
nan
C50
nan
nan
nan
nan
nan
nan
nan
nan
B42
nan
C148
nan
```

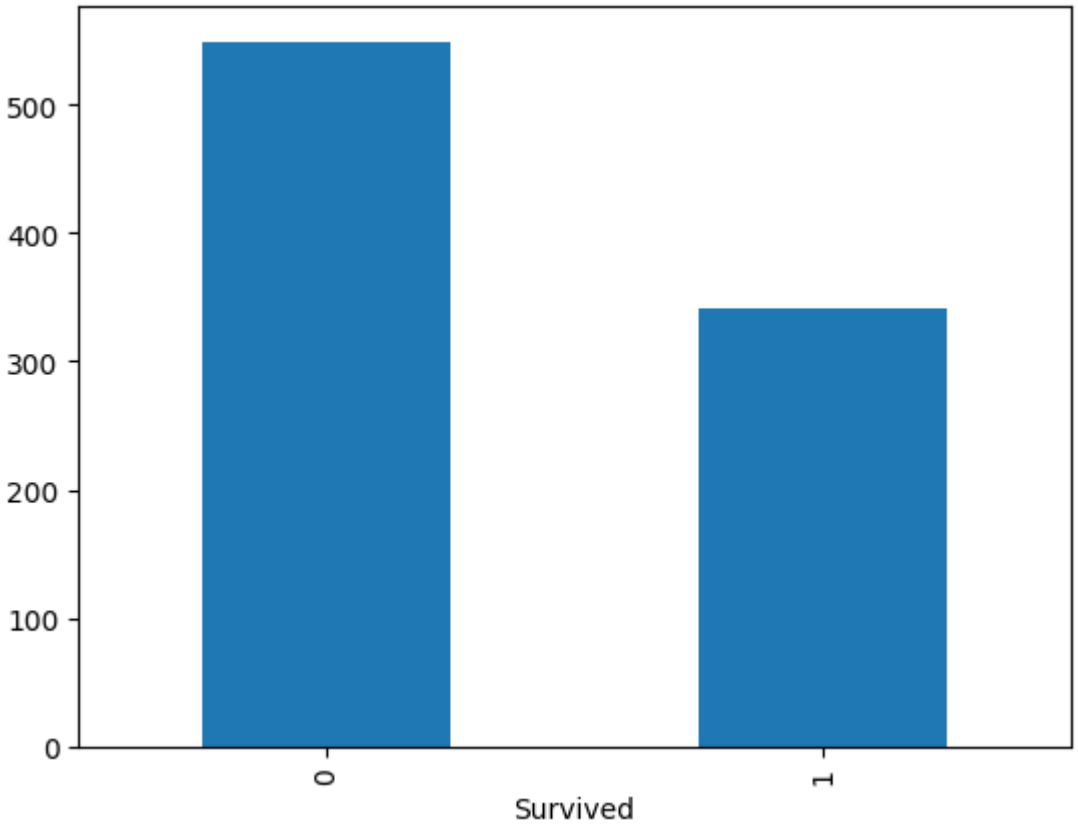
In [12]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Survived  891 non-null    int64  
 1   Pclass    891 non-null    int64  
 2   Name      891 non-null    object 
 3   Sex       891 non-null    object 
 4   Age       714 non-null    float64
 5   SibSp     891 non-null    int64  
 6   Parch     891 non-null    int64  
 7   Ticket    891 non-null    object 
 8   Fare      891 non-null    float64
 9   Cabin     204 non-null    object 
 10  Embarked  889 non-null    object 
dtypes: float64(2), int64(4), object(5)
memory usage: 76.7+ KB
```

In [13]:
```
df.Survived.value_counts()
```

Out[13]:
```
Survived
0    549
1    342
Name: count, dtype: int64
```

In [14]:
```
df.Survived.value_counts().plot(kind="bar")
plt.show()
```



¿Cómo seleccionar información concreta de nuestro dataset?

Forma 1

In [15]:
```
df["Age"].head()
```

Out[15]:
```
0    22.0
1    38.0
2    26.0
3    35.0
4    35.0
Name: Age, dtype: float64
```

Forma 2

In [16]:
```
df.Age.head()
```

Out[16]:
```
0    22.0
1    38.0
2    26.0
3    35.0
4    35.0
Name: Age, dtype: float64
```

Forma 3

In [17]:
```
df[["Age"]].head()
```

Out[17]:

	Age
0	22.0
1	38.0
2	26.0
3	35.0
4	35.0

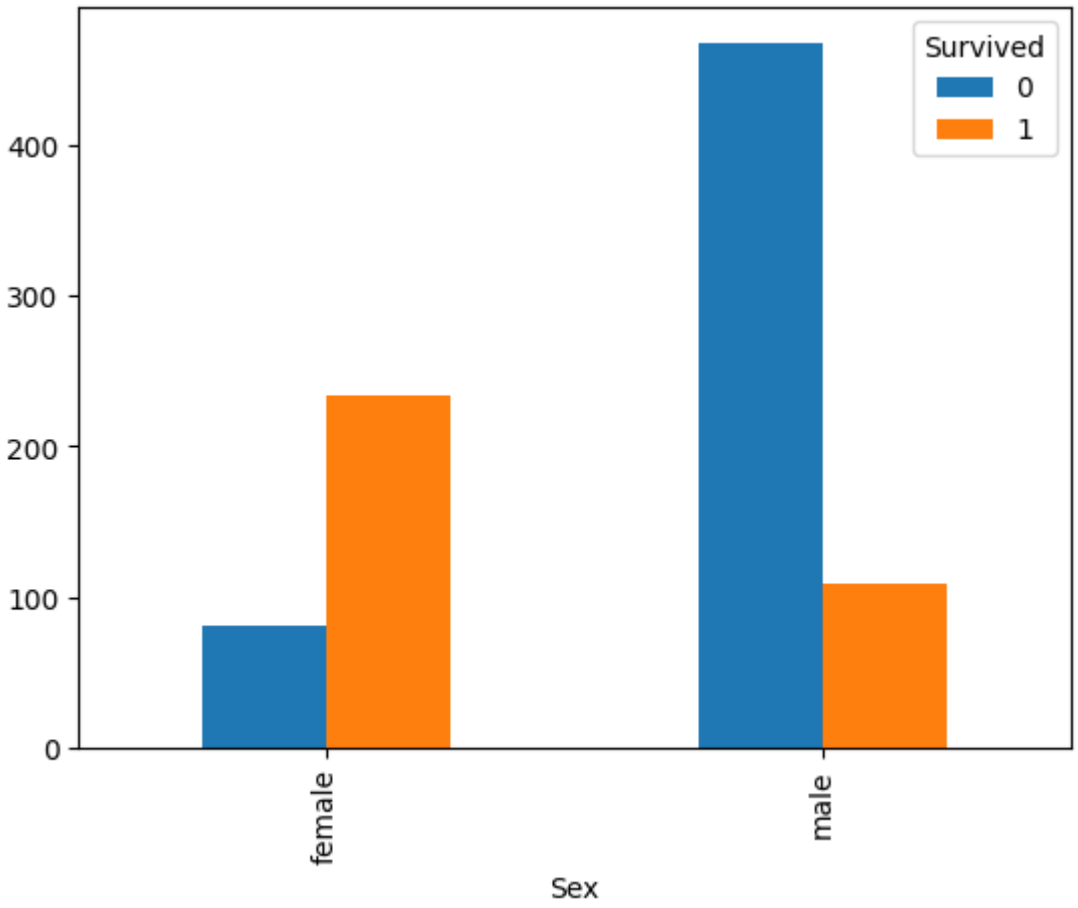
Crosstab

In [18]:
```
pd.crosstab(df.Sex, df.Survived)
```

Out[18]:

Survived	0	1
Sex		
female	81	233
male	468	109

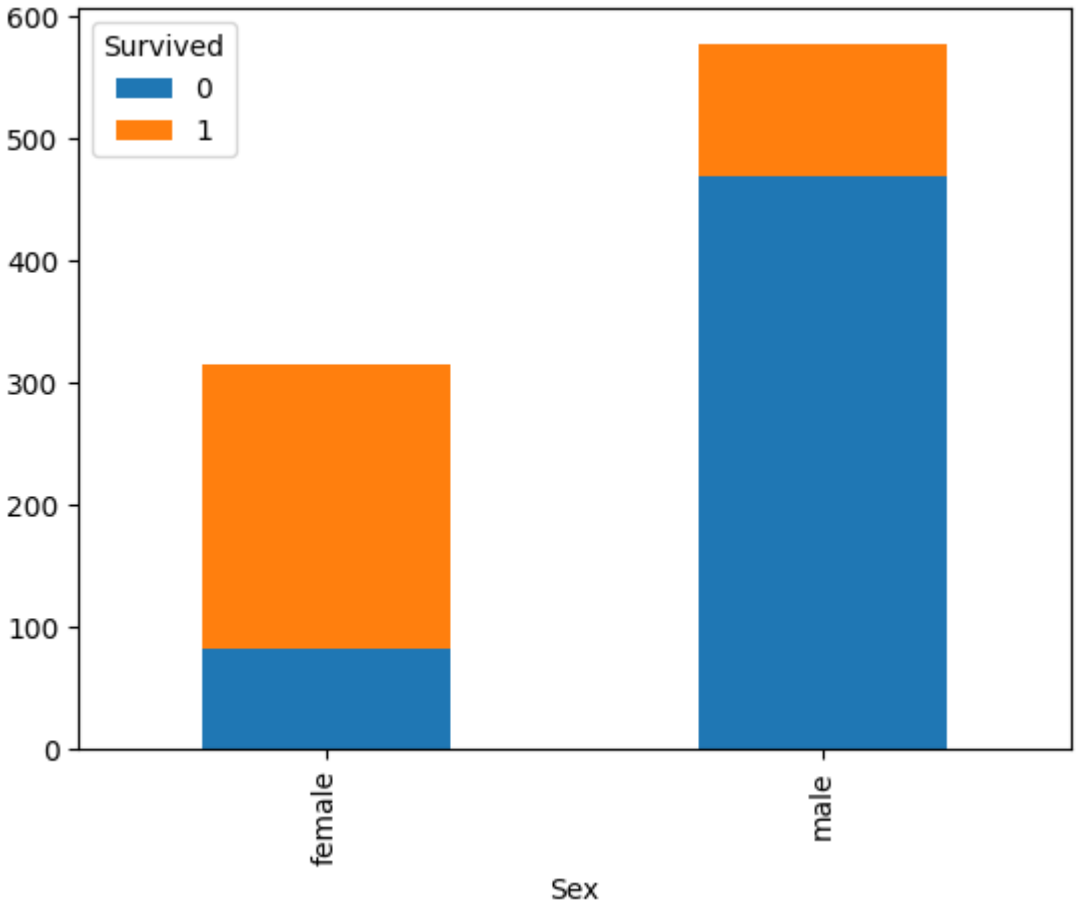
In [19]:
```
pd.crosstab(df.Sex, df.Survived).plot(kind="bar")
plt.show()
```



Conclusión:

- La mayoría de las mujeres sobreviven.
- La mayoría de los hombres NO sobrevivieron

In [20]:
```
pd.crosstab(df.Sex, df.Survived).plot(kind="bar", stacked=True)
plt.show()
```



Conclusión:

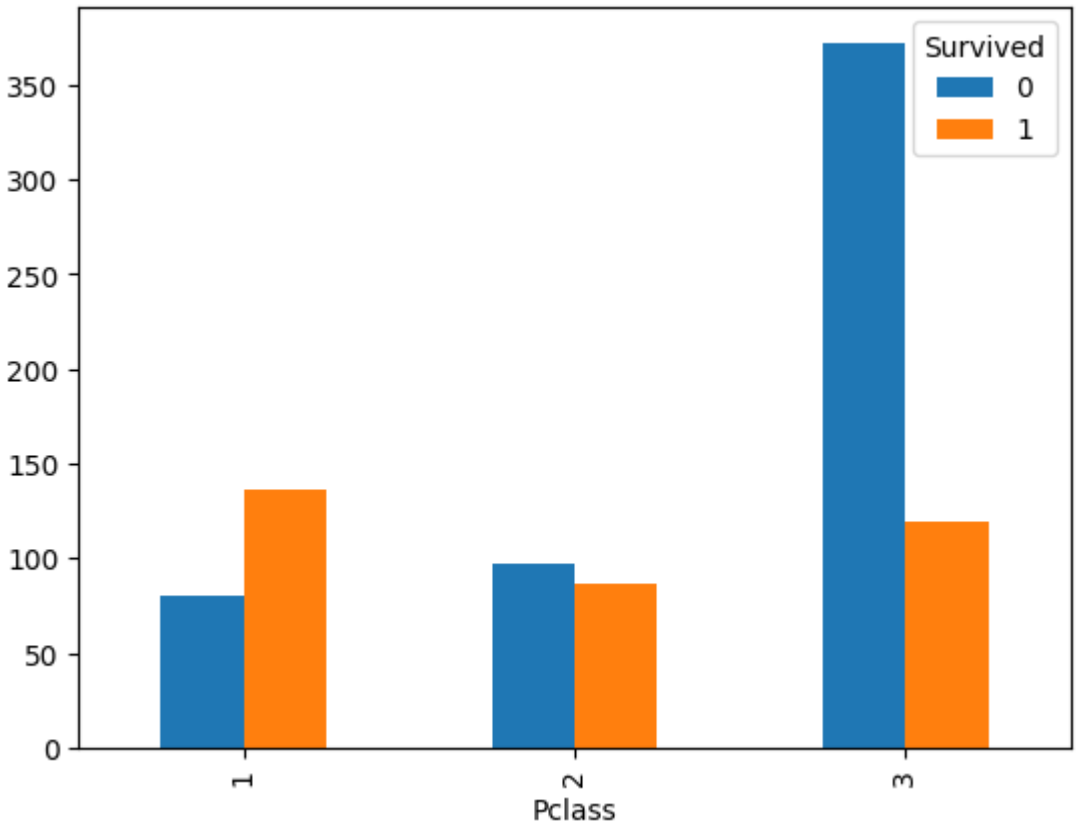
- Hay más hombres que mujeres, es casi el doble.

In [21]:
```
pd.crosstab(df.Pclass, df.Survived)
```

Out[21]:

Survived	0	1
Pclass		
1	80	136
2	97	87
3	372	119

In [22]:
```
pd.crosstab(df.Pclass, df.Survived).plot(kind="bar")
plt.show()
```



Conclusión:

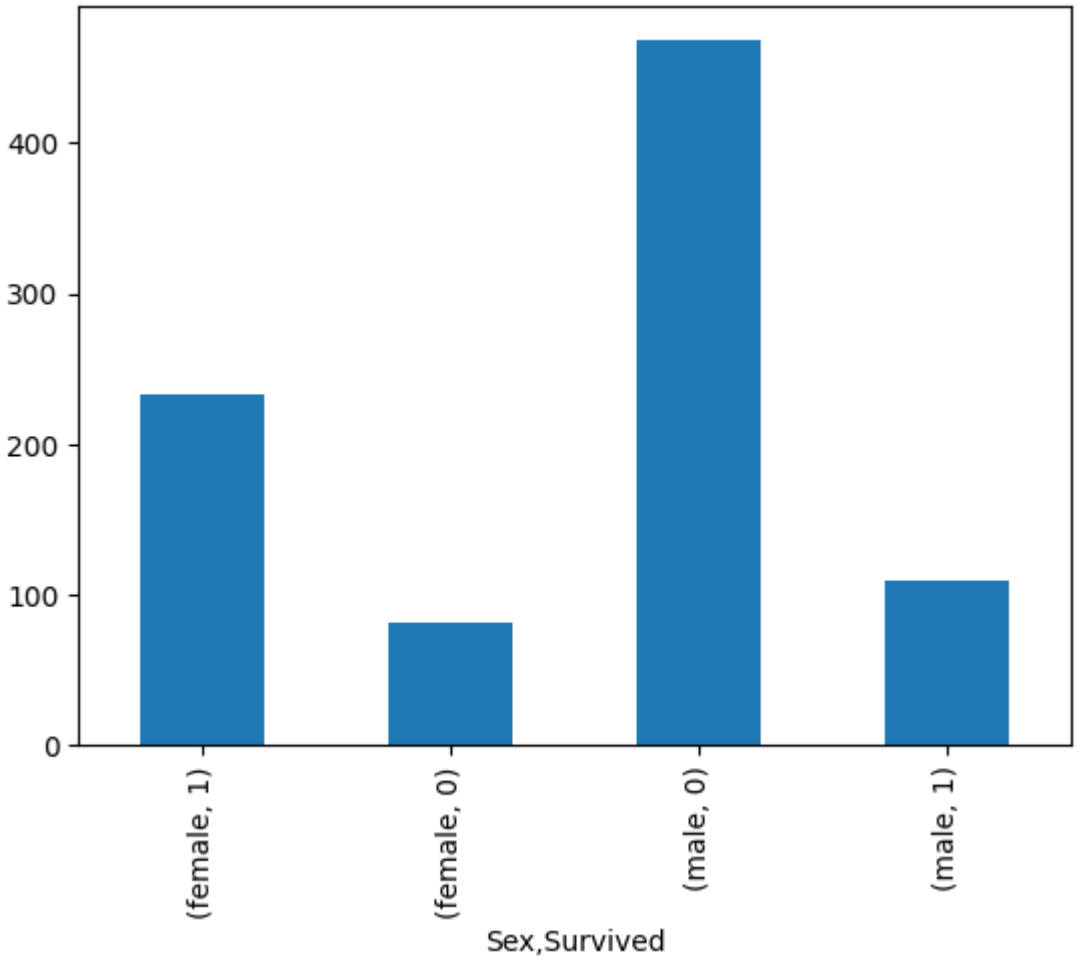
- La mayoría de los que NO sobrevivieron eran de la 3ª clase

groupby

In [23]:
```
df.groupby("Sex").Survived.value_counts()
```

Out[23]:
```
Sex     Survived
female  1           233
        0            81
male    0           468
        1           109
Name: count, dtype: int64
```

In [24]:
```
df.groupby("Sex").Survived.value_counts().plot(kind="bar")
plt.show()
```



por filtrado

- Selecciono aquellas filas donde Pclass == 1
- Me creo un dataframe de la misma forma que tenía antes

In [ ]:
```
# Una forma...
```

In [25]:
```
df_sex_uno = df[df.Pclass == 1]
df_sex_uno.head()
```

Out[25]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
6	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46
11	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C103
23	1	1	Sloper, Mr. William Thompson	male	28.0	0	0	113788	35.5000	A6

In [ ]:
```
# Otra forma...
```

In [26]:
```
df_sex_crosstab = df[df.Pclass == 1]["Survived"]
df_sex_crosstab.head()
```

Out[26]:
```
1     1
3     1
6     0
11    1
23    1
Name: Survived, dtype: int64
```

Ejemplos de creación de dataframes

In [27]:
```
df_sobreviven_todos = df[df["Survived"] == 1]
df_sobreviven_ninguno = df[df["Survived"] == 0]
hombres_sobrevivieron = df[(df["Survived"] == 1) & (df["Sex"] == "male")]
hombres__no_sobrevivieron = df[(df["Survived"] == 0) & (df["Sex"] == "mal
mujeres_sobrevivieron = df[(df["Survived"] == 1) & (df["Sex"] == "female"
mujeres_no_sobrevivieron = df[(df["Survived"] == 0) & (df["Sex"] == "fema
```

In [28]:
```
df_sobreviven_todos.head()
```

Out[28]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
8	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN
9	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN

In [29]:
```
df_sobreviven_todos.Survived.value_counts(3)
```

Out[29]:
```
Survived
1    1.0
Name: proportion, dtype: float64
```

In [30]:
```
df_sobreviven_ninguno.head()
```

Out[30]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Em
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	
5	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	
6	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	
7	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	

In [31]:
```
df_sobreviven_ninguno.Survived.value_counts(3)
```

Out[31]:
```
Survived
0    1.0
Name: proportion, dtype: float64
```

In [32]:
```
hombres_sobrevivieron.head()
```

Out[32]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	E
17	1	2	Williams, Mr. Charles Eugene	male	NaN	0	0	244373	13.0000	NaN	
21	1	2	Beesley, Mr. Lawrence	male	34.0	0	0	248698	13.0000	D56	
23	1	1	Sloper, Mr. William Thompson	male	28.0	0	0	113788	35.5000	A6	
36	1	3	Mamee, Mr. Hanna	male	NaN	0	0	2677	7.2292	NaN	
55	1	1	Woolner, Mr. Hugh	male	NaN	0	0	19947	35.5000	C52	

In [33]:
```
hombres__no_sobrevivieron.head()
```

Out[33]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Em
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	
5	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	
6	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	
7	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	

In [34]:
```
mujeres_sobrevivieron.head()
```

Out[34]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
8	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN
9	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN

In [35]:
```
mujeres_no_sobrevivieron.head()
```

Out[35]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
14	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14.0	0	0	350406	7.8542	NaN
18	0	3	Vander Planke, Mrs. Julius (Emelia Maria Vande...	female	31.0	1	0	345763	18.0000	NaN
24	0	3	Palsson, Miss. Torborg Danira	female	8.0	3	1	349909	21.0750	NaN
38	0	3	Vander Planke, Miss. Augusta Maria	female	18.0	2	0	345764	18.0000	NaN
40	0	3	Ahlin, Mrs. Johan (Johanna Persdotter Larsson)	female	40.0	1	0	7546	9.4750	NaN

Obtenemos información de los gráficos

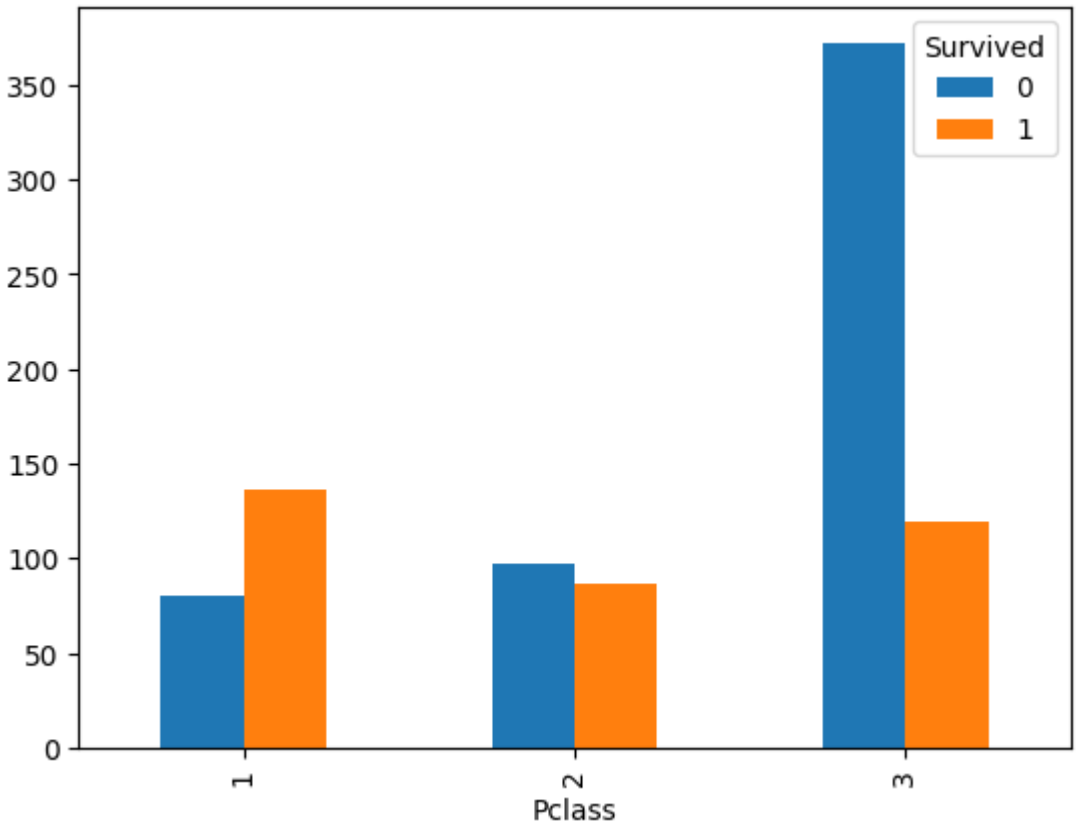
In [36]:
```
df.head()
```

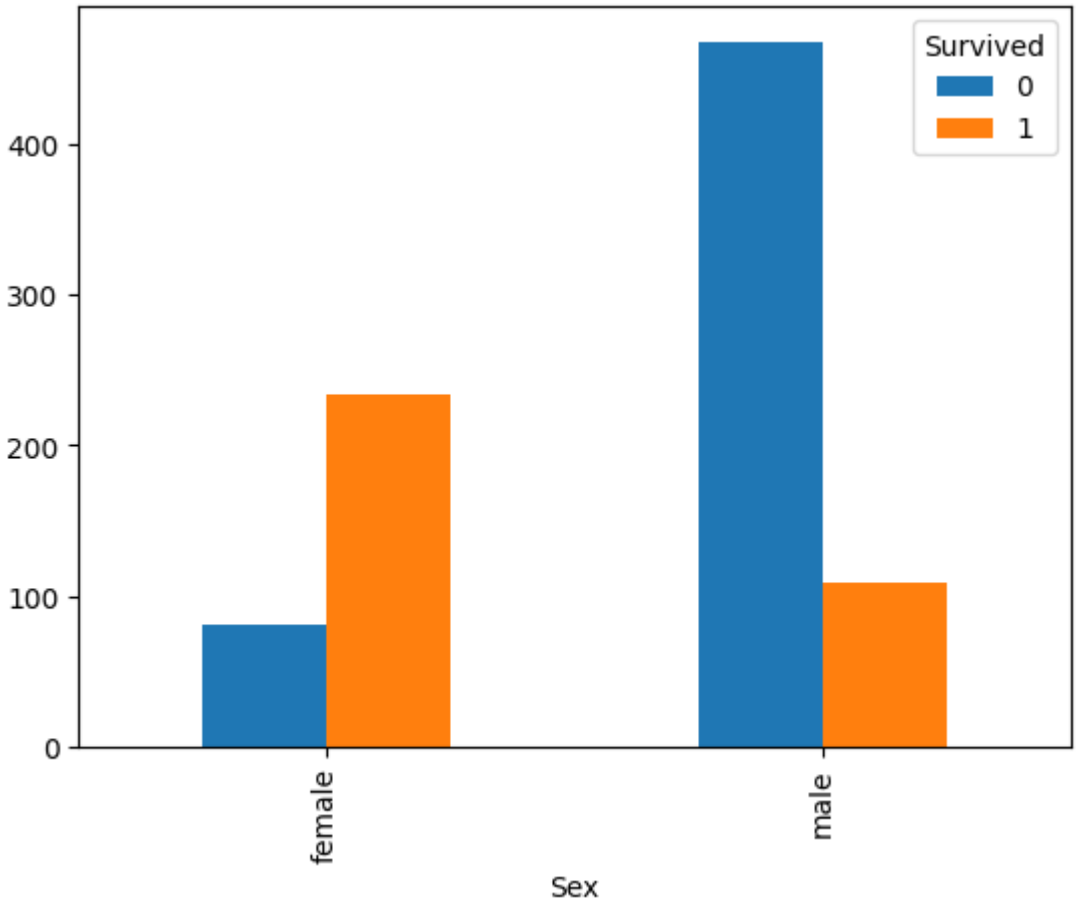
Out[36]:

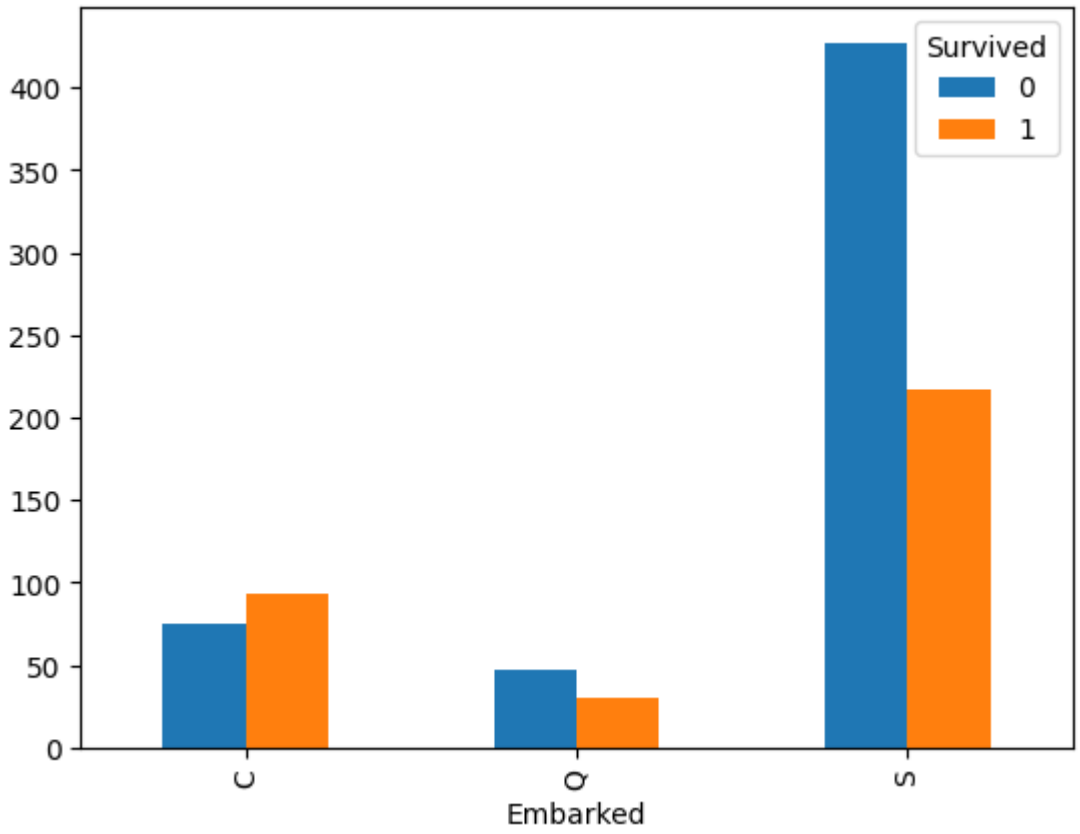
	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN

In [37]:
```
opciones = ["Pclass", "Sex", "Embarked"]

for opcion in opciones:
    pd.crosstab(df[opcion], df.Survived).plot(kind="bar")
    plt.show()
```

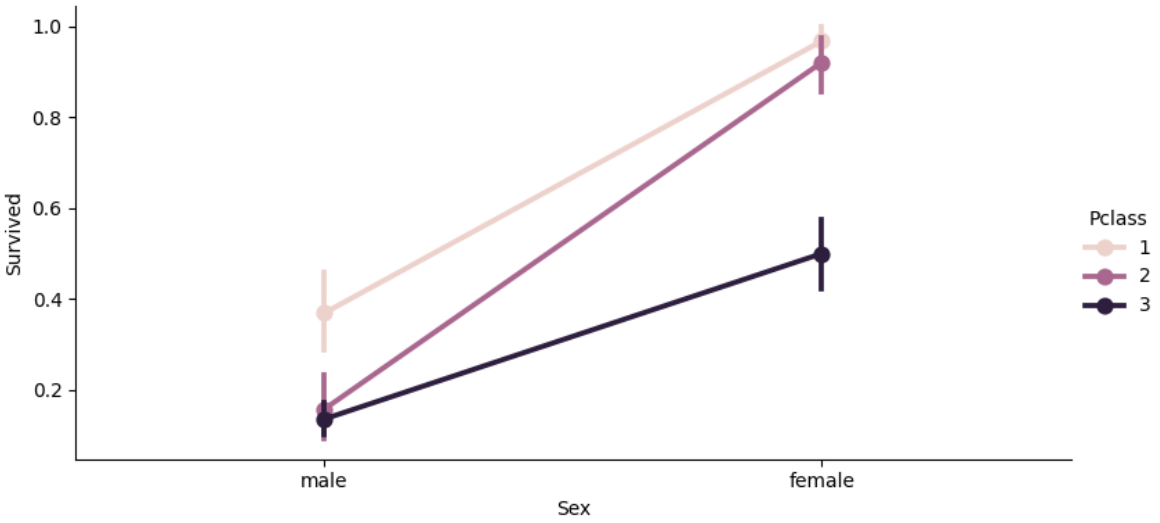

Survived
0
1
350
300
250
200
150
100
50
0
1
2
3
Pclass


Survived
0
1
400
300
200
100
0
female
male
Sex

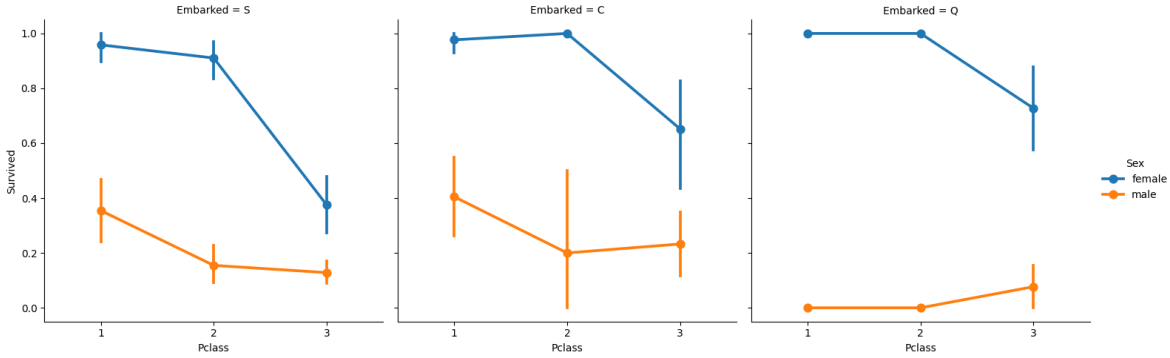


SEABORN

In [38]:
```
sns.catplot(x="Sex", y="Survived", hue="Pclass", kind="point", height=4,
plt.show()
```



In [39]:
```
sns.catplot(x="Pclass", y="Survived", hue="Sex", kind="point", col="Embar
plt.show()
```



Algunas conclusiones:

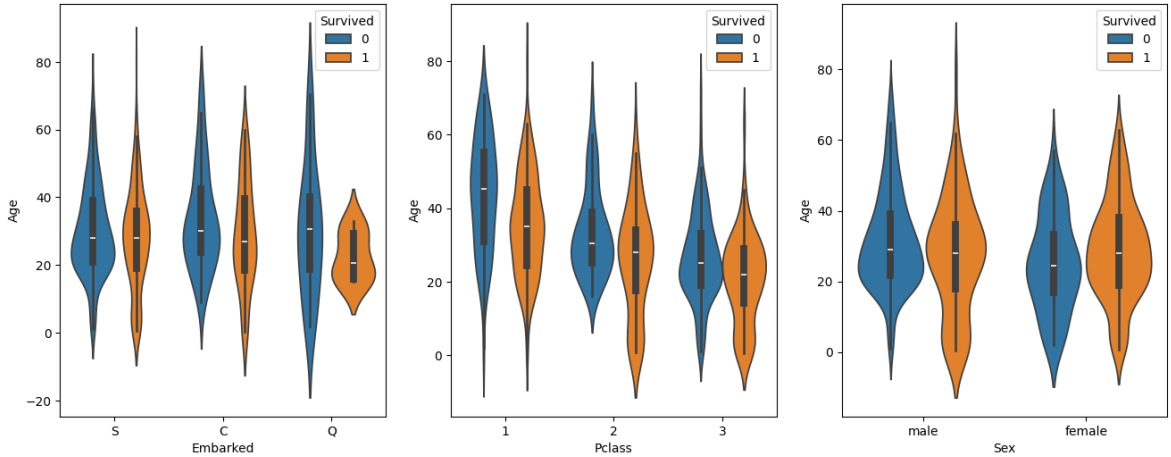
- Nos fijamos en la gráfica de la izquierda, embarked="S" --> las mujeres de 3 clase que embarcaron en "S" fallecieron muchas en comparación con 1 y 2 clase, pese a ello sobrevivieron algo más que los hombres de 1 clase del mismo puerto.
- los hombres con mayor porcentaje e supervivencia embarcaron en "C"
- Los hombres con menor porcentaje de supervivencia embarcaron en "Q"
- Vemos nuevamente como la mayoría de las mujeres sobreviveron, pero no los hombres.

Edad y supervivencia

In [40]:
```
# me creo una figura
fig = plt.figure(figsize=(16,6))
# 3 subplots
# 1 fila 3 columnas - gráfica 1
ax1 = fig.add_subplot(131)
# 1 fila 3 columnas - gráfica 2
ax2 = fig.add_subplot(132)
# 1 fila 3 columnas - gráfica 3
ax3 = fig.add_subplot(133)

# violinplot
sns.violinplot(x="Embarked", y="Age", hue="Survived", data=df, ax=ax1)
sns.violinplot(x="Pclass", y="Age", hue="Survived", data=df, ax=ax2)
sns.violinplot(x="Sex", y="Age", hue="Survived", data=df, ax=ax3)

plt.show()
```

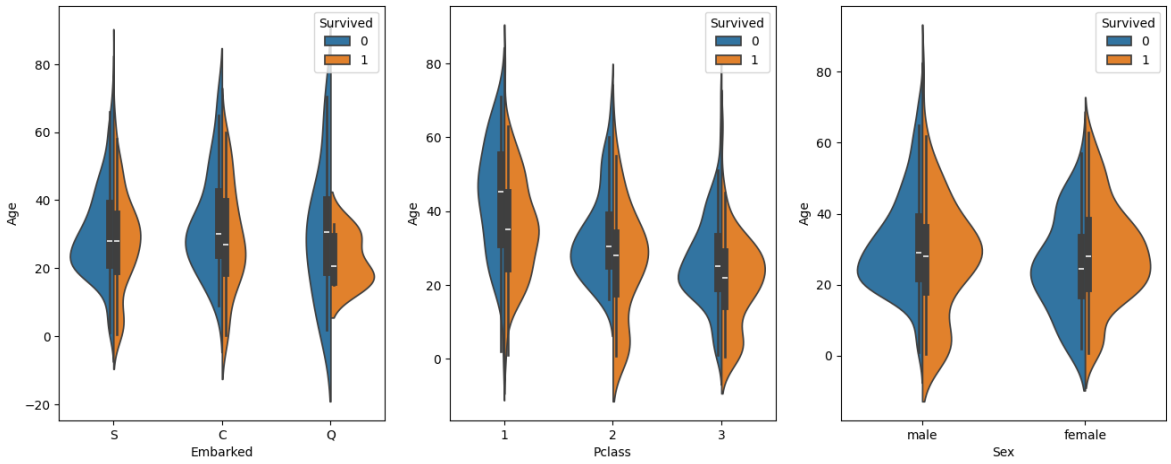


Hago un split== True, para mostrarlo más visual

In [41]:
```
# me creo una figura
fig = plt.figure(figsize=(16,6))
# 3 subplots
# 1 fila 3 columnas - gráfica 1
ax1 = fig.add_subplot(131)
# 1 fila 3 columnas - gráfica 2
ax2 = fig.add_subplot(132)
# 1 fila 3 columnas - gráfica 3
ax3 = fig.add_subplot(133)

# violinplot
sns.violinplot(x="Embarked", y="Age", hue="Survived", data=df, split=True
sns.violinplot(x="Pclass", y="Age", hue="Survived", data=df, split=True,
sns.violinplot(x="Sex", y="Age", hue="Survived", data=df, split=True, ax=

plt.show()
```



Conclusiones:

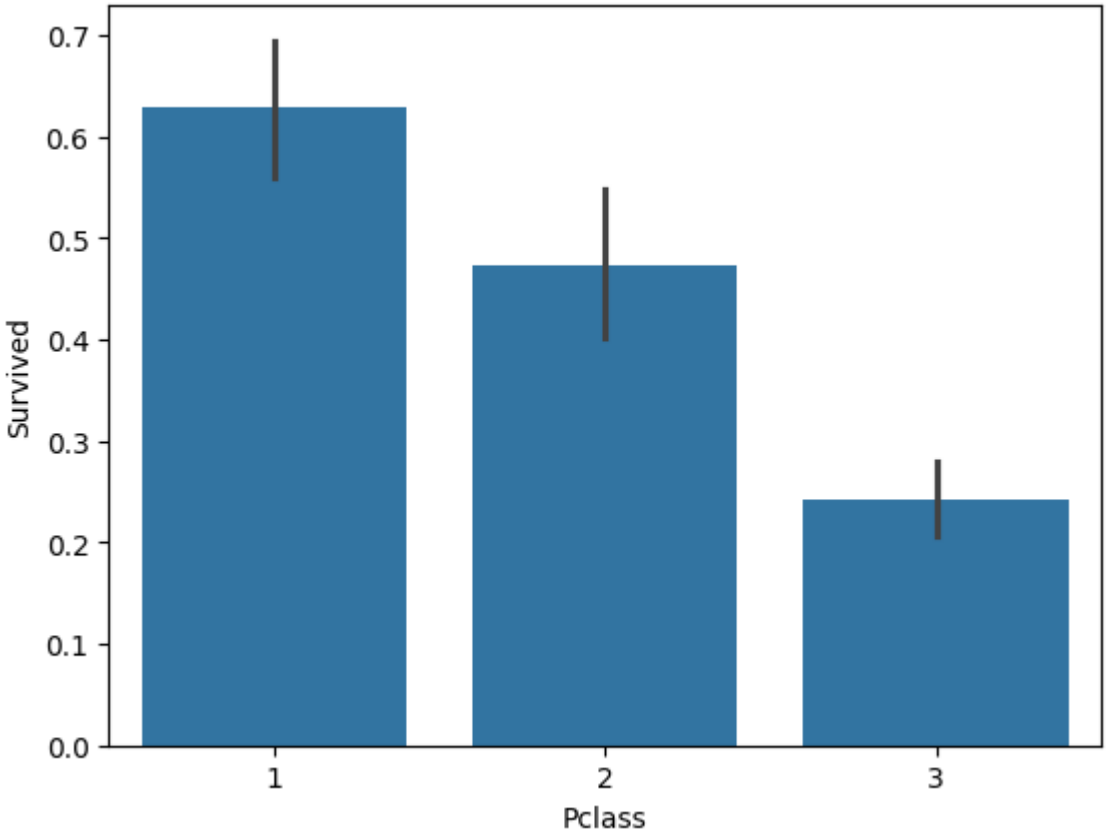
- EMBARKED y Age:
  - La gente de unos 18-35 años de Q SI sobrevivieron mayoritariamente,(no todos)
  - no hay porcentajes mayoritarios significativos en las otras 2 embarcaciones
  - En Q embarcaron bastantes niños los cuales no sobrevivieron.
- PCLASS y Age:
  - De la 2ª clase sobre todo y la 3 sobrevivieron la mayoria de sus niños
- Sex y Age:
  - Hay mas ancianos que ancianas
  - Los jovenes (varón) menores de 20 años en general sobrevivieron pero no las mujeres

In [42]:
```
df.Age.describe()
```

Out[42]:
```
count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
75%       38.000000
max       80.000000
Name: Age, dtype: float64
```

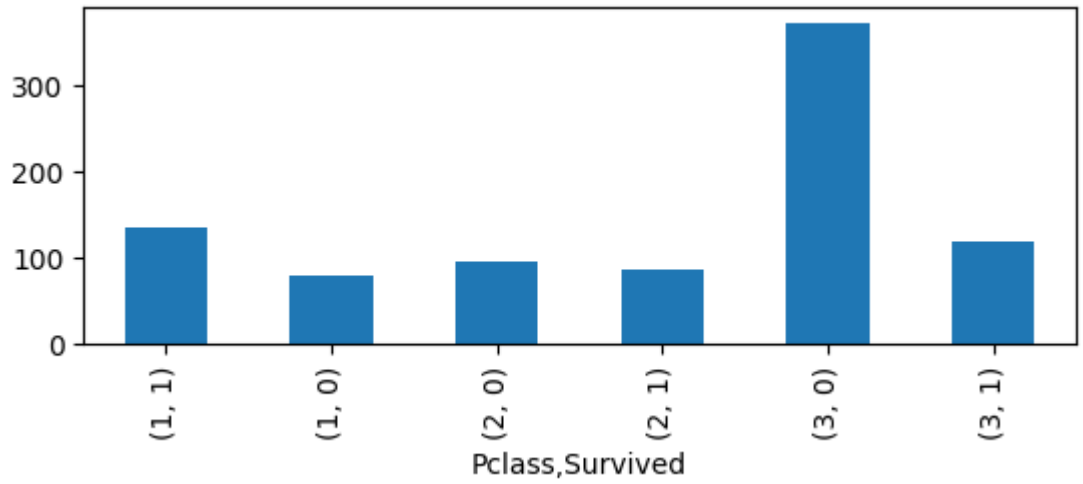
barplot

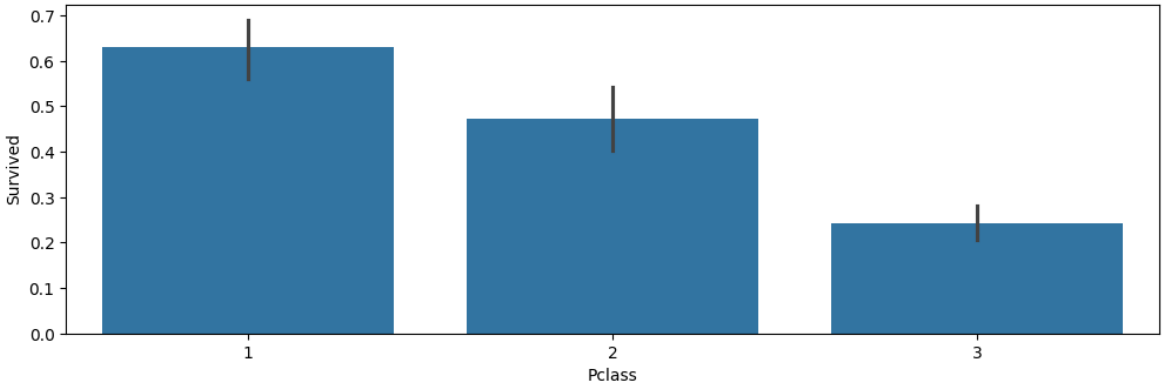
In [44]:
```
sns.barplot(x="Pclass", y="Survived", data=df)
plt.show()
```



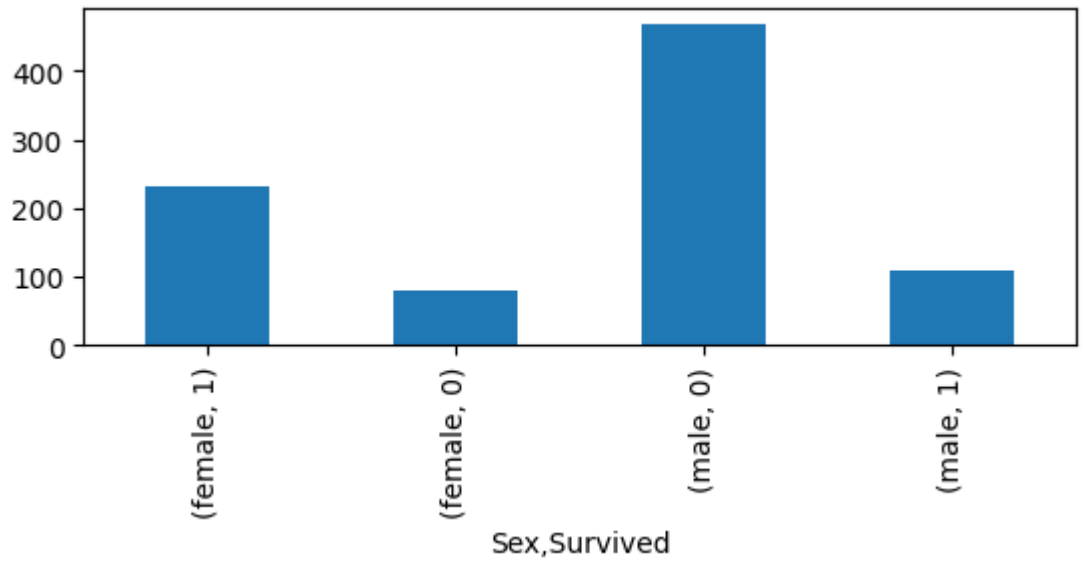
In [45]:
```
def funcion_graficas(feat):
    plt.subplot(2, 1, 1)
    df.groupby(feat).Survived.value_counts().plot(kind="bar")
    plt.figure(figsize=(12,8))
    plt.subplot(2, 1, 2)
    sns.barplot(x=feat, y="Survived", data=df)
    plt.show()
```

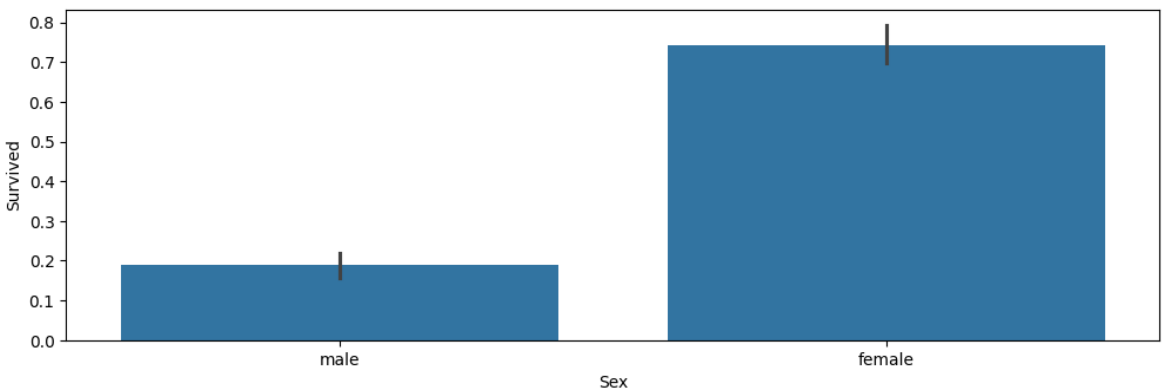
In [46]:
```
funcion_graficas("Pclass")
```



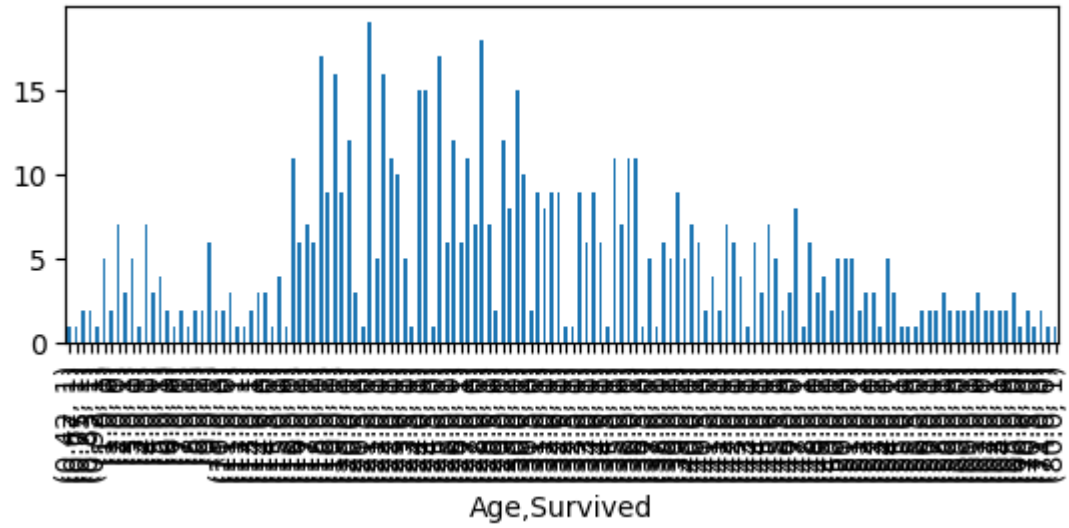


In [47]:
```
funcion_graficas("Sex")
```

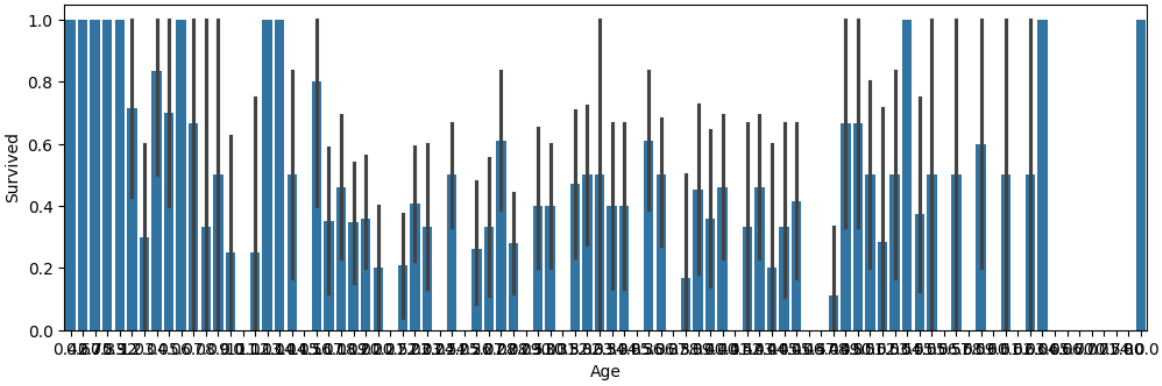




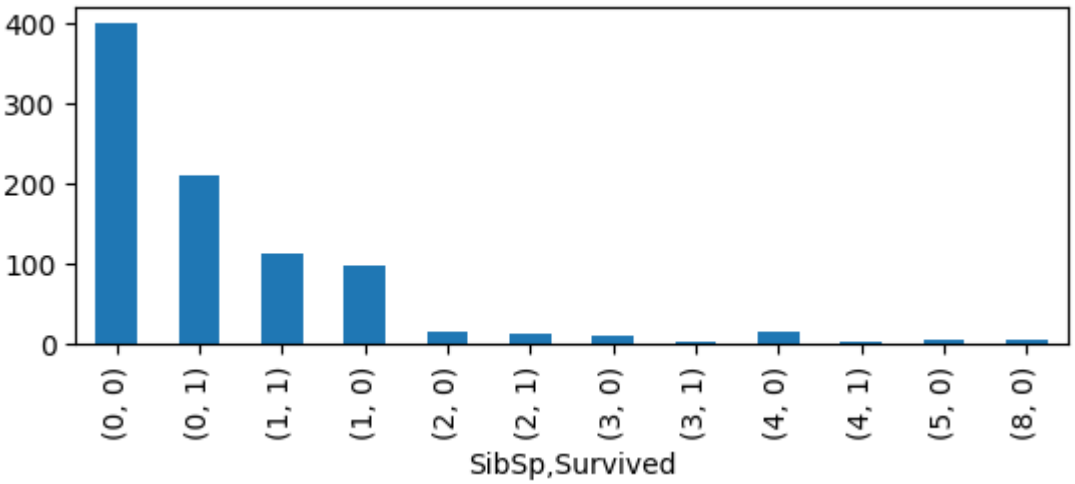
In [48]:
```
funcion_graficas("Age")
```



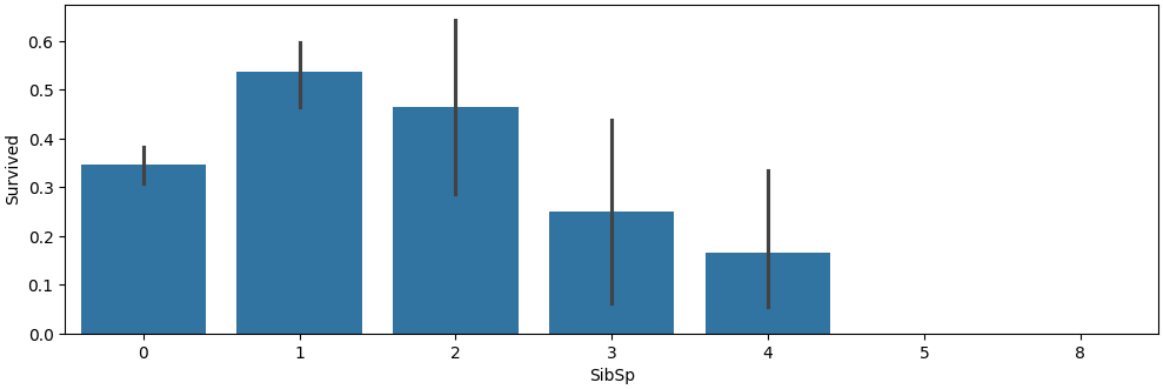
Age,Survived



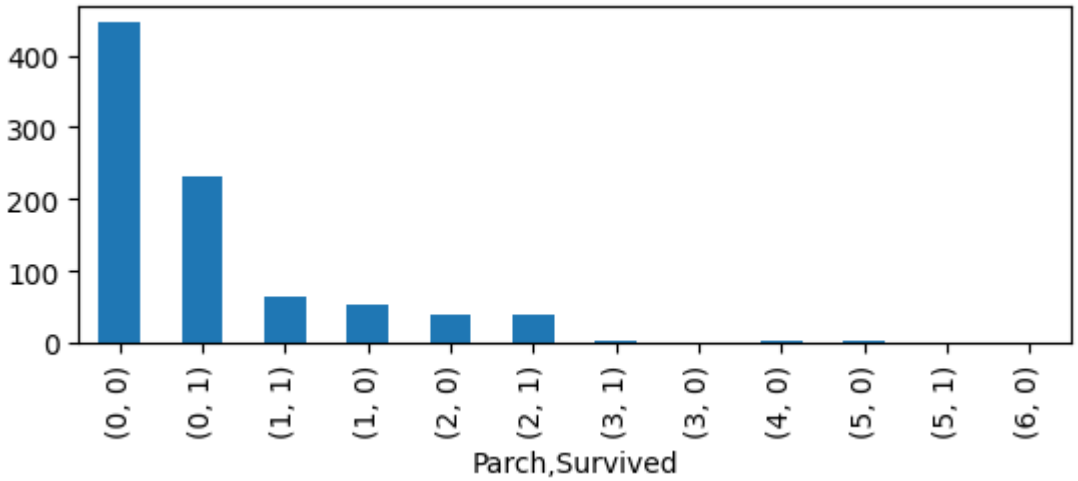
In [49]:
```
funcion_graficas("SibSp")
```

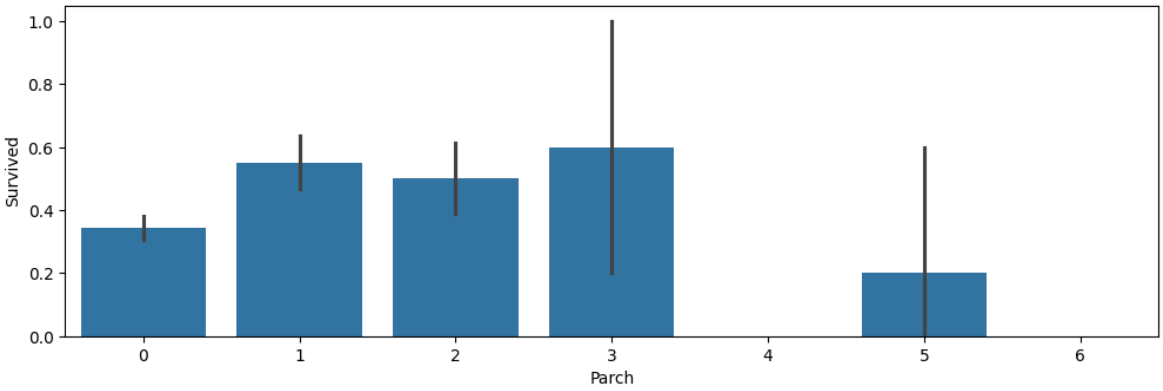


SibSp,Survived

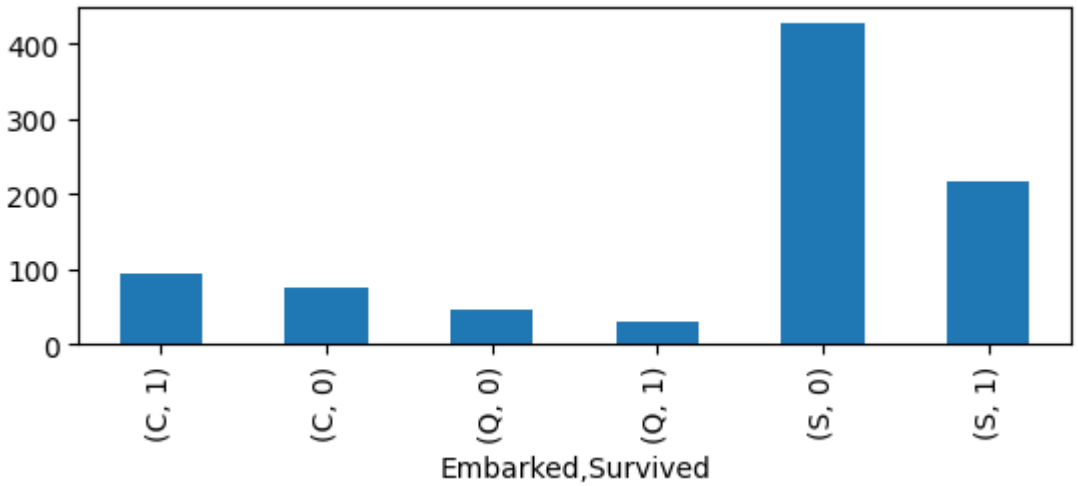


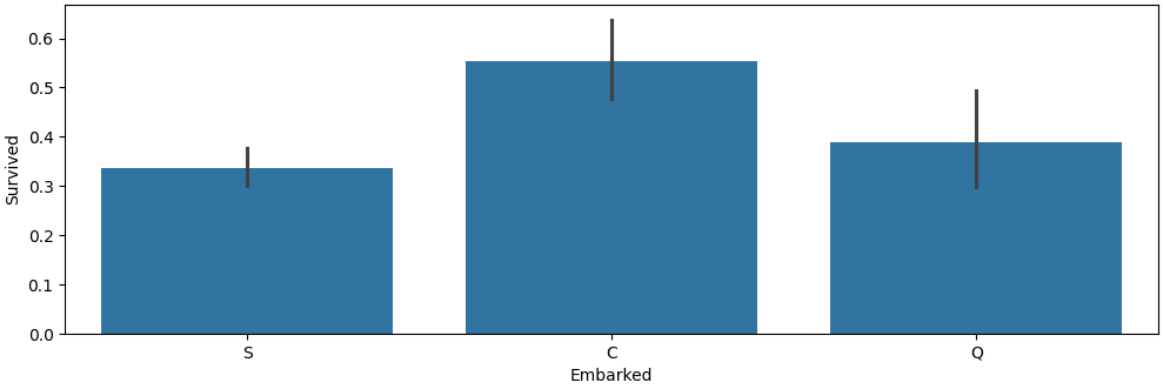
In [50]:
```
funcion_graficas("Parch")
```





In [51]:
```
funcion_graficas("Embarked")
```





Feature Engineering

En esta parte podemos hacer uso de la información obtenida y conclusiones.

Para hacerlo lo más simple posible, lo que haremos será elegir solamente algunas columnas.

In [52]:
```
df.head()
```

Out[52]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN

In [53]:
```
df.isnull().sum()
```

Out[53]:
```
Survived      0
Pclass        0
Name          0
Sex           0
Age         177
SibSp         0
Parch         0
Ticket        0
Fare          0
Cabin       687
Embarked      2
dtype: int64
```

-1-Name- no lo tendremos en cuenta por simplificar

df["Name"] = df["Name"].str.extract("([A-Za-z]+)", expand=False)

seria posible una posible forma de analizar la columna Name, pero no lo haremos.

-2-Age- Usamos el valor promedio de la columna para rellernar los valores que faltan

In [54]:
```
df.Age.isnull().sum()
```

Out[54]:
```
177
```

In [55]:
```
df.Age = df.Age.fillna(df.Age.mean())
```

In [56]:
```
df.Age.isnull().sum()
```

Out[56]:
```
0
```

In [57]:
```
df
```

Out[57]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.000000	1	0	PC 17599	71.2833
2	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000
4	0	3	Allen, Mr. William Henry	male	35.000000	0	0	373450	8.0500
...	...	...	...	...	...	...	...	...	...
886	0	2	Montvila, Rev. Juozas	male	27.000000	0	0	211536	13.0000
887	1	1	Graham, Miss. Margaret Edith	female	19.000000	0	0	112053	30.0000
888	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	29.699118	1	2	W./C. 6607	23.4500
889	1	1	Behr, Mr. Karl Howell	male	26.000000	0	0	111369	30.0000
890	0	3	Dooley, Mr. Patrick	male	32.000000	0	0	370376	7.7500

891 rows × 11 columns

-3-Ticket- No la tendremos en cuenta por simplificar

In [58]:
```
df.Ticket.value_counts()
```

Out[58]:
```
Ticket
347082      7
CA. 2343    7
1601        7
3101295     6
CA 2144     6
           ..
9234        1
19988       1
2693        1
PC 17612    1
370376      1
Name: count, Length: 681, dtype: int64
```

-4-Cabin- No la tendremos en cuenta por falta de información

In [59]:
```
df.Cabin.isnull().sum(), len(df)
```

Out[59]:
```
(687, 891)
```

-5-Embarked

In [60]:
```
df.Embarked.isnull().sum()
```

Out[60]:
```
2
```

In [61]:
```
df.Embarked.value_counts()
```

Out[61]:
```
Embarked
S    644
C    168
Q     77
Name: count, dtype: int64
```

In [62]:
```
df["Embarked"] = df["Embarked"].fillna("S")
```

In [63]:
```
df.Embarked.value_counts()
```

Out[63]:
```
Embarked
S    646
C    168
Q     77
Name: count, dtype: int64
```

In [64]:
```
df.Embarked.isnull().sum()
```

Out[64]:
```
0
```

BORRAMOS del dataframe las columnas antes mencionadas

In [65]:
```
df.head(2)
```

Out[65]:

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	En
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	

In [66]:
```
df = df.drop(["Name", "Ticket", "Cabin"], axis=1)
df.head(2)
```

Out[66]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.0	1	0	7.2500	S
1	1	1	female	38.0	1	0	71.2833	C

Concepto de **datos categóricos**:

- columnas con strings hombre/mujer por ejemplo.
- columnas con strings con 3 opciones ("Embarked")
- en el caso de Pclass 3 hace referencia a "tercera clase"
- y 3 no vale, más que 1, y más en este caso, cuya probabilidad de supervivencia es más baja.

In [ ]:
```
# pd.get_dummies()
```

In [67]:
```
df = pd.get_dummies(df, columns=["Sex", "Pclass", "Embarked"], drop_first
df.head()
```

Out[67]:

	Survived	Age	SibSp	Parch	Fare	Sex_male	Pclass_2	Pclass_3	Embarked_Q
0	0	22.0	1	0	7.2500	True	False	True	False
1	1	38.0	1	0	71.2833	False	False	False	False
2	1	26.0	0	0	7.9250	False	False	True	False
3	1	35.0	1	0	53.1000	False	False	False	False
4	0	35.0	0	0	8.0500	True	False	True	False

Escalado de los datos

Existen varias formas de hacer el escalado de datos. Normalmente no hay diferencias significativas, pero algunas veces sí.

Por abreviar, trataremos de mencionar 2 tipos (sklearn):

- StandardScaler
- MinMaxScaler

En nuestro caso, no daremos importancia a cuál es el mejor en este caso concreto.
(Preporcesamiento)

In [ ]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing
# https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing
```

In [68]:
```
# StandardScaler

# x-mean(x) / std(x)

df.Age = (df.Age - np.mean(df.Age, axis=0)) / (np.std(df.Age, axis=0))
df.Fare = (df.Fare - np.mean(df.Fare, axis=0)) / (np.std(df.Fare, axis=0)
df.head()
```

Out[68]:

	Survived	Age	SibSp	Parch	Fare	Sex_male	Pclass_2	Pclass_3	Embar
0	0	-0.592481	1	0	-0.502445	True	False	True	
1	1	0.638789	1	0	0.786845	False	False	False	
2	1	-0.284663	0	0	-0.488854	False	False	True	
3	1	0.407926	1	0	0.420730	False	False	False	
4	0	0.407926	0	0	-0.486337	True	False	True	

Obtención de X, y

In [69]:
```
X = df.drop("Survived", axis=1)
X.head()
```

Out[69]:

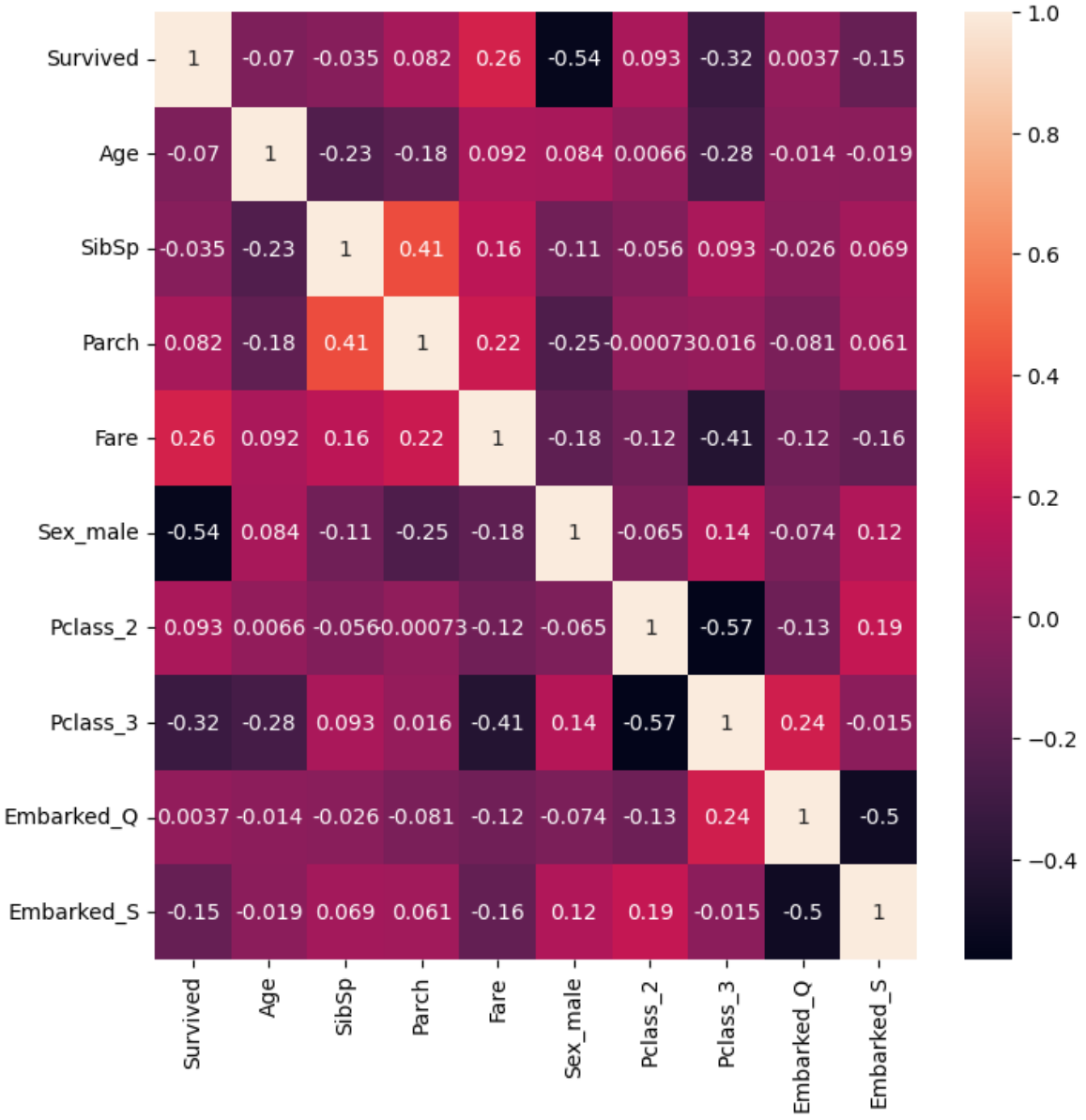
	Age	SibSp	Parch	Fare	Sex_male	Pclass_2	Pclass_3	Embarked_Q	Em
0	-0.592481	1	0	-0.502445	True	False	True	False	
1	0.638789	1	0	0.786845	False	False	False	False	
2	-0.284663	0	0	-0.488854	False	False	True	False	
3	0.407926	1	0	0.420730	False	False	False	False	
4	0.407926	0	0	-0.486337	True	False	True	False	

In [70]:
```
y = df["Survived"]
y.head()
```

Out[70]:
```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

Heapmap

In [71]:
```
plt.figure(figsize=(8,8))
sns.heatmap(df.corr(), annot=True)
plt.show()
```



Creado por:

Isabel Maniega