In [78]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [6]:
```
df=pd.read_csv('train.csv')
```

In [7]:
```
df.head()
```

Out[7]:

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

Why do EDA

Model building

Analysis and reporting

Validate assumptions

Handling missing values

Feature engineering

detecting outliers

In [14]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Column Types

Numerical : Age, Fare, PassengereId

Categorical : Survived, Pclass, Sex, SibSp, ParCh, Embarked

Mixed :Name, Ticket, Cabin

Univariate Analysis

Univariate analysis focuses on analyzing each feature in the dataset independently.

Distribution analysis : The distribution of each feature is examined to identify its shape, central tendency, and dispersion.

Identifying potential issues : Univariate analysis helps in identifying potential problems with the data such as outliers, skewness, and missing values

The shape of a data distribution refers to its overall pattern or form as it is represented on a graph. Some common shapes of data distributions include:

Normal Distribution: A symmetrical and bell-shaped distribution where the mean, median, and mode are equal and the majority of the data falls in the middle of the distribution with gradually decreasing frequencies towards the tails.

Skewed Distribution: A distribution that is not symmetrical, with one tail being longer than the other. It can be either positively skewed (right-skewed) or negatively skewed (left-skewed).

Bimodal Distribution: A distribution with two peaks or modes.

Uniform Distribution: A distribution where all values have an equal chance of occurring.

The shape of the data distribution is important in identifying the presence of outliers, skewness, and the type of statistical tests and models that can be used for further analysis.

Dispersion is a statistical term used to describe the spread or variability of a set of data. It measures how far the values in a data set are spread out from the central tendency (mean, median, or mode) of the data.

There are several measures of dispersion, including:

Range: The difference between the largest and smallest values in a data set.

Variance: The average of the squared deviations of each value from the mean of the data set.

Standard Deviation: The square root of the variance. It provides a measure of the spread of the data that is in the same units as the original data.

Interquartile range (IQR): The range between the first quartile (25th percentile) and the third quartile (75th percentile) of the data.

Dispersion helps to describe the spread of the data, which can help to identify the presence of outliers and skewness in the data.

Steps of doing Univariate Analysis on Numerical columns

Descriptive Statistics: Compute basic summary statistics for the column, such as mean, median, mode, standard deviation, range, and quartiles. These statistics give a general understanding of the distribution of the data and can help identify skewness or outliers.

Visualizations: Create visualizations to explore the distribution of the data. Some common visualizations for numerical data include histograms, box plots, and density plots. These visualizations provide a visual representation of the distribution of the data and can help identify skewness an outliers.

Identifying Outliers: Identify and examine any outliers in the data. Outliers can be identified using visualizations. It is important to determine whether the outliers are due to measurement errors, data entry errors, or legitimate differences in the data, and to decide whether to include or exclude them from the analysis.

Skewness: Check for skewness in the data and consider transforming the data or using robust statistical methods that are less sensitive to skewness, if necessary.

Conclusion: Summarize the findings of the EDA and make decisions about how to proceed with further analysis.

In [ ]:

Age

conclusions

Age is normally(almost) distributed

20% of the values are missing

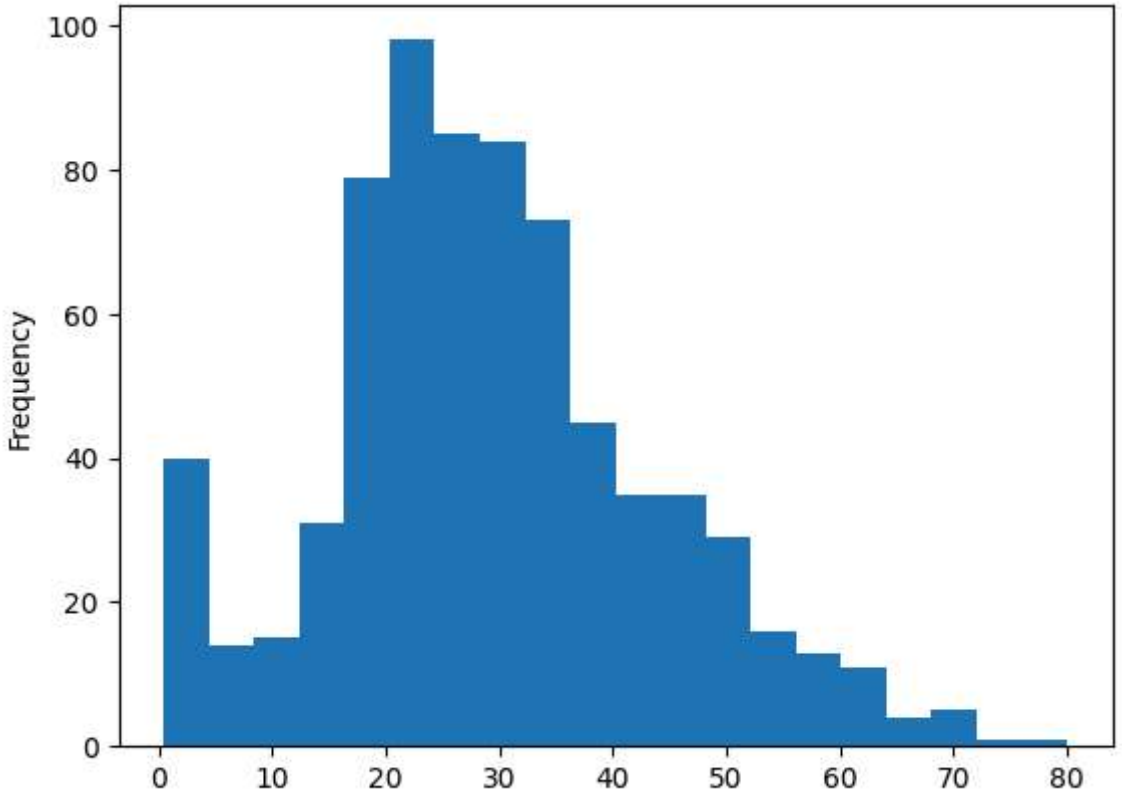
There are some outliers

In [15]:
```
df['Age'].describe()
```

Out[15]:
```
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
```

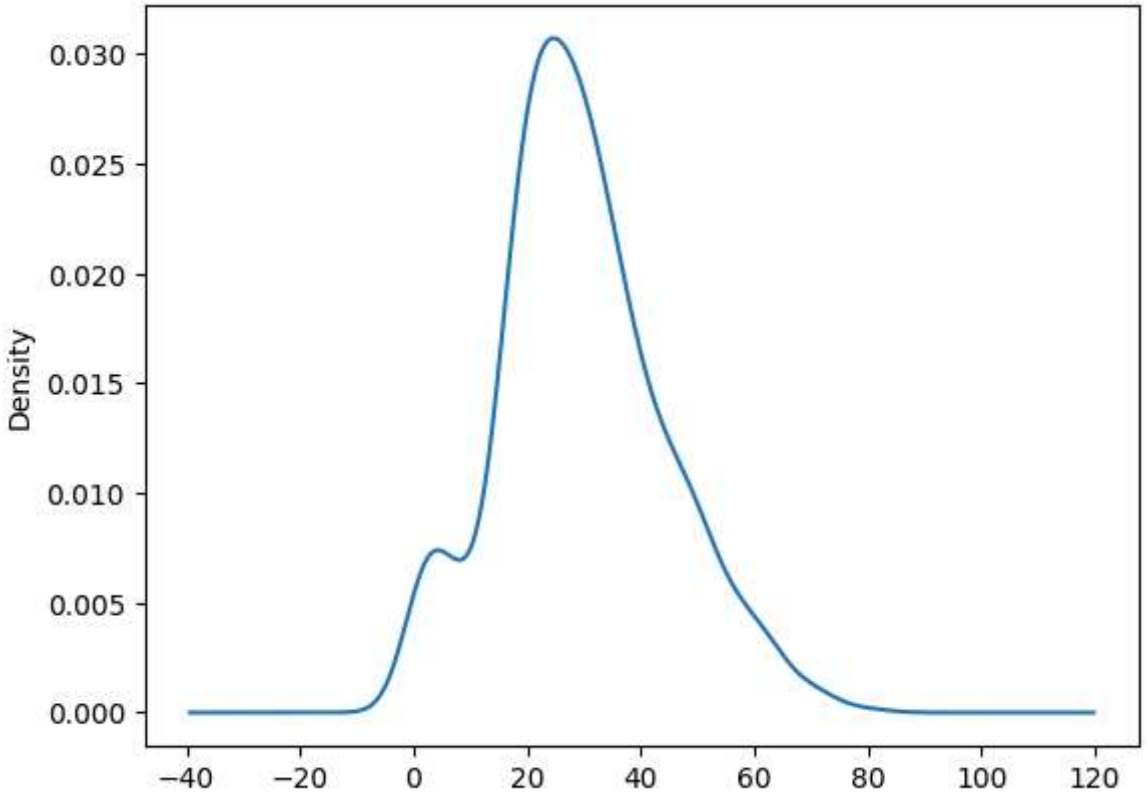
In [20]:
```
df['Age'].plot(kind='hist', bins=20)
```

Out[20]:
```
<Axes: ylabel='Frequency'>
```



In [21]:
```
df['Age'].plot(kind='kde')
```

Out[21]:
```
<Axes: ylabel='Density'>
```

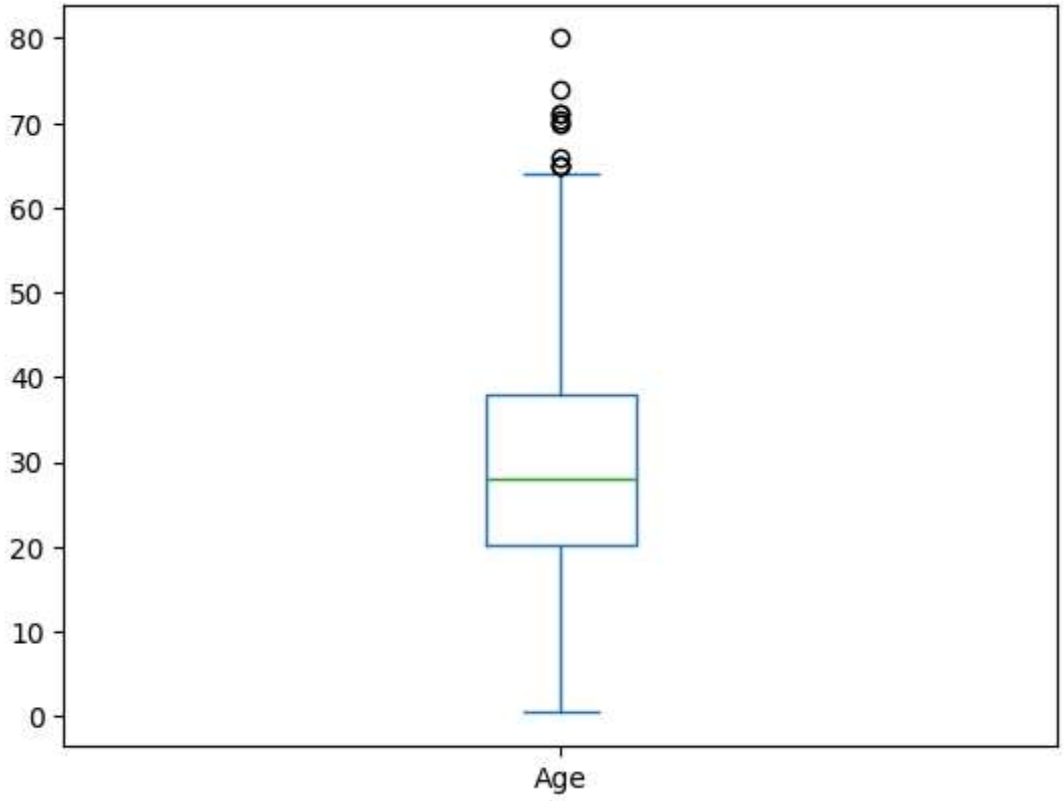


In [22]:
```
df['Age'].skew()
```

Out[22]:
```
0.38910778230082704
```

In [25]:
```
df['Age'].plot(kind='box')
```

Out[25]:
```
<Axes: >
```



In [27]:
```
df[df['Age']>65]
```

Out[27]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
33	34	0	2	Wheadon, Mr. Edward H	male	66.0	0	0	C.A. 24579	10.5000	NaN	S
96	97	0	1	Goldschmidt, Mr. George B	male	71.0	0	0	PC 17754	34.6542	A5	C
116	117	0	3	Connors, Mr. Patrick	male	70.5	0	0	370369	7.7500	NaN	Q
493	494	0	1	Artagaveytia, Mr. Ramon	male	71.0	0	0	PC 17609	49.5042	NaN	C
630	631	1	1	Barkworth, Mr. Algernon Henry Wilson	male	80.0	0	0	27042	30.0000	A23	S
672	673	0	2	Mitchell, Mr. Henry Michael	male	70.0	0	0	C.A. 24580	10.5000	NaN	S
745	746	0	1	Crosby, Capt. Edward Gifford	male	70.0	1	1	WE/P 5735	71.0000	B22	S
851	852	0	3	Svensson, Mr. Johan	male	74.0	0	0	347060	7.7750	NaN	S

In [28]:
```
df['Age'].isnull().sum()
```

Out[28]:
```
177
```

In [33]:
```
df['Age'].isnull().sum()/len(df['Age'])
```

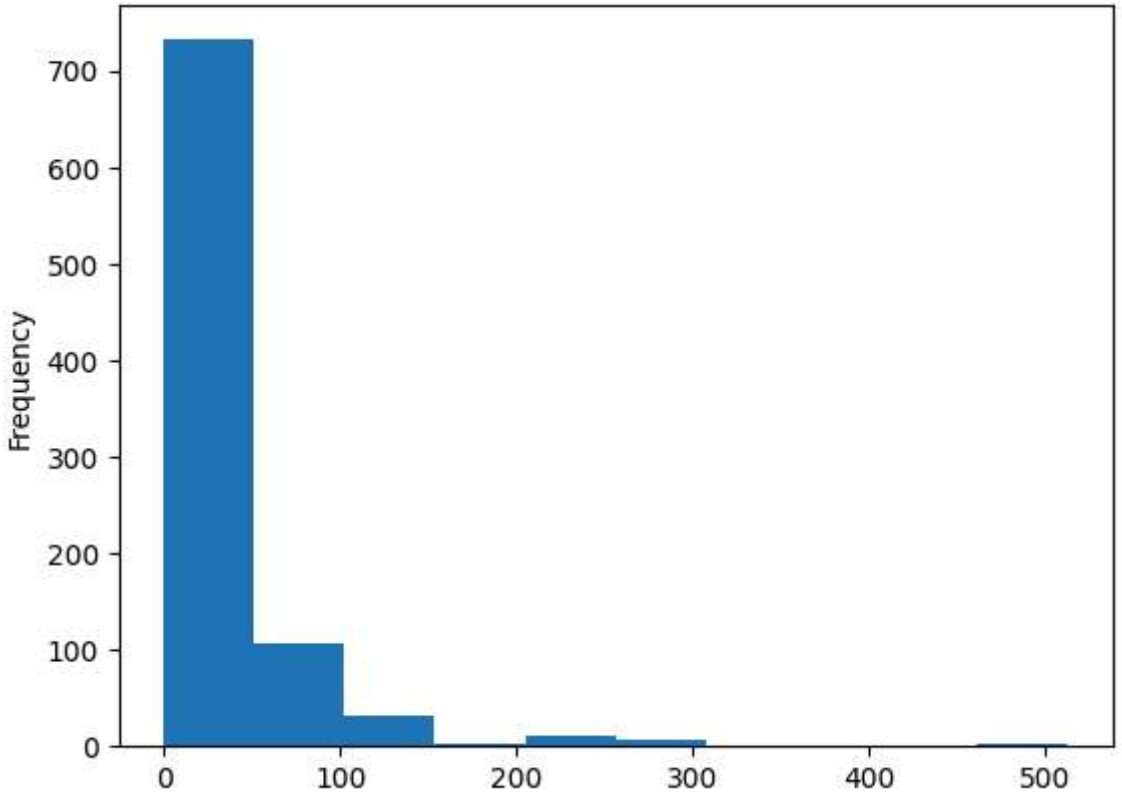
Out[33]:
```
0.19865319865319866
```

In [34]:
```
df['Fare'].describe()
```

Out[34]:
```
count    891.000000
mean      32.204208
std       49.693429
min        0.000000
25%        7.910400
50%       14.454200
75%       31.000000
max      512.329200
Name: Fare, dtype: float64
```

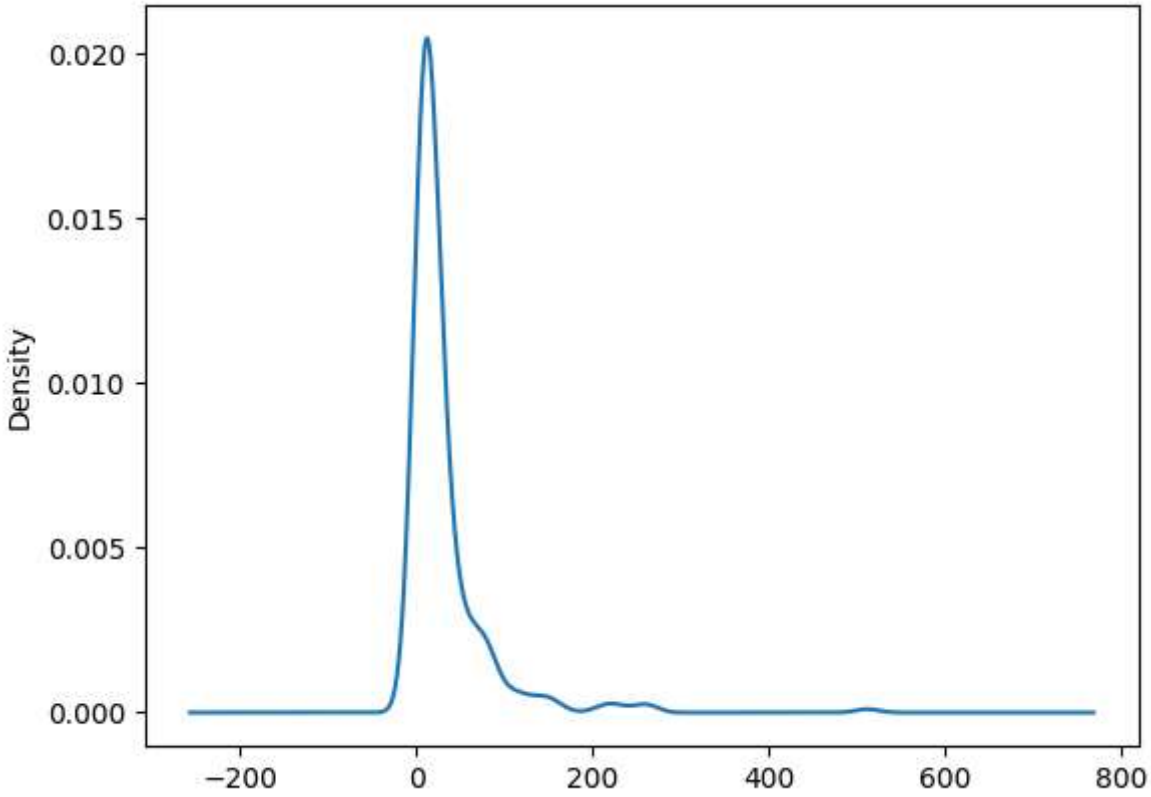
In [35]:
```
df['Fare'].plot(kind='hist')
```

Out[35]:
```
<Axes: ylabel='Frequency'>
```



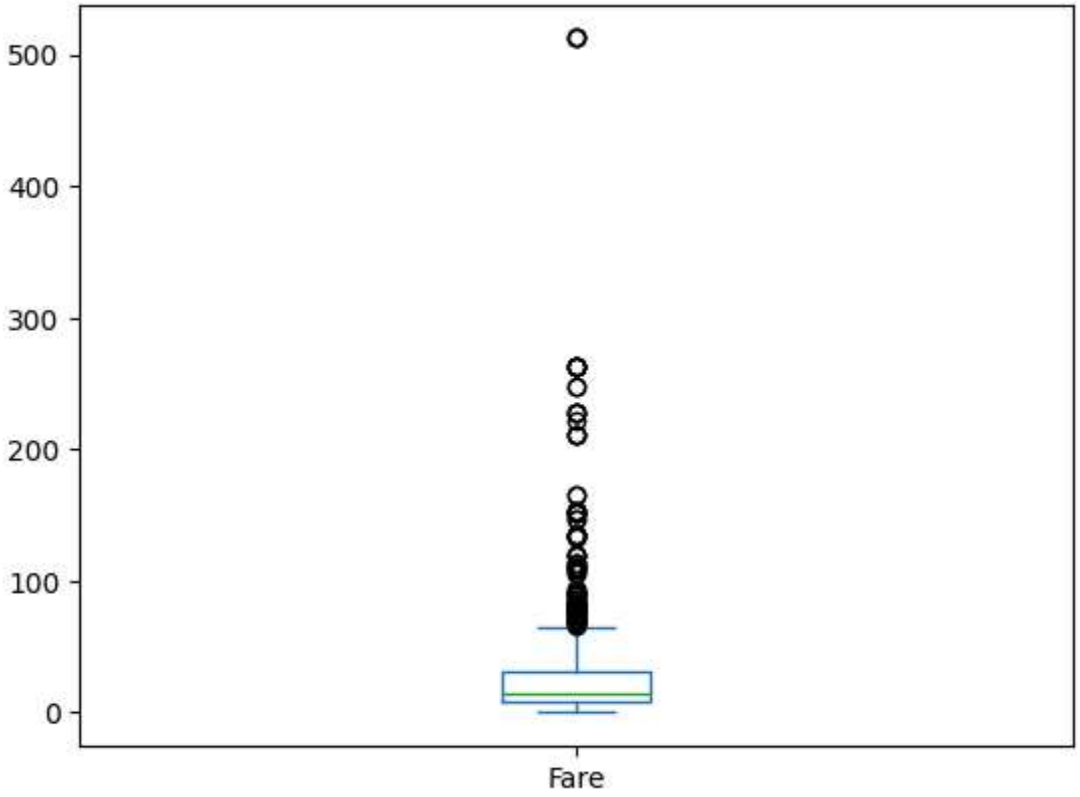
In [36]:
```
df['Fare'].plot(kind='kde')
```

Out[36]:
```
<Axes: ylabel='Density'>
```



In [37]:
```
df['Fare'].plot(kind='box')
```

Out[37]:
```
<Axes: >
```



In [39]:
```
df['Fare'].skew()
```

Out[39]:
```
4.787316519674893
```

In [42]:
```
df[df['Fare']>250]
```

Out[42]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
27	28	0	1	Fortune, Mr. Charles Alexander	male	19.0	3	2	19950	263.0000	C23 C25 C27	S
88	89	1	1	Fortune, Miss. Mabel Helen	female	23.0	3	2	19950	263.0000	C23 C25 C27	S
258	259	1	1	Ward, Miss. Anna	female	35.0	0	0	PC 17755	512.3292	NaN	C
311	312	1	1	Ryerson, Miss. Emily Borie	female	18.0	2	2	PC 17608	262.3750	B57 B59 B63 B66	C
341	342	1	1	Fortune, Miss. Alice Elizabeth	female	24.0	3	2	19950	263.0000	C23 C25 C27	S
438	439	0	1	Fortune, Mr. Mark	male	64.0	1	4	19950	263.0000	C23 C25 C27	S
679	680	1	1	Cardeza, Mr. Thomas Drake Martinez	male	36.0	0	1	PC 17755	512.3292	B51 B53 B55	C
737	738	1	1	Lesurer, Mr. Gustave J	male	35.0	0	0	PC 17755	512.3292	B101	C
742	743	1	1	Ryerson, Miss. Susan Parker "Suzette"	female	21.0	2	2	PC 17608	262.3750	B57 B59 B63 B66	C

Fare

Conclusions

The data is highly(positively) skewed

Fare col actually contains the group fare and not the individual fare (This might be an issue)

We need to create a new col called individual fare

In [ ]:

Steps of doing Univariate Analysis on Categorical columns

Descriptive Statistics: Compute the frequency distribution of the categories in the column. This will give a general understanding of the distribution of the categories and their relative frequencies.

Visualizations: Create visualizations to explore the distribution of the categories. Some common visualizations for categorical data include count plots and pie charts. These visualizations provide a visual representation of the distribution of the categories and can help identify any patterns or anomalies in the data.

Missing Values: Check for missing values in the data and decide how to handle them. Missing values can be imputed or excluded from the analysis, depending on the research question and the data set.

Conclusion: Summarize the findings of the EDA and make decisions about how to proceed with further analysis.

Survived

conclusions

Parch and SibSp cols can be merged to form a new col call family_size

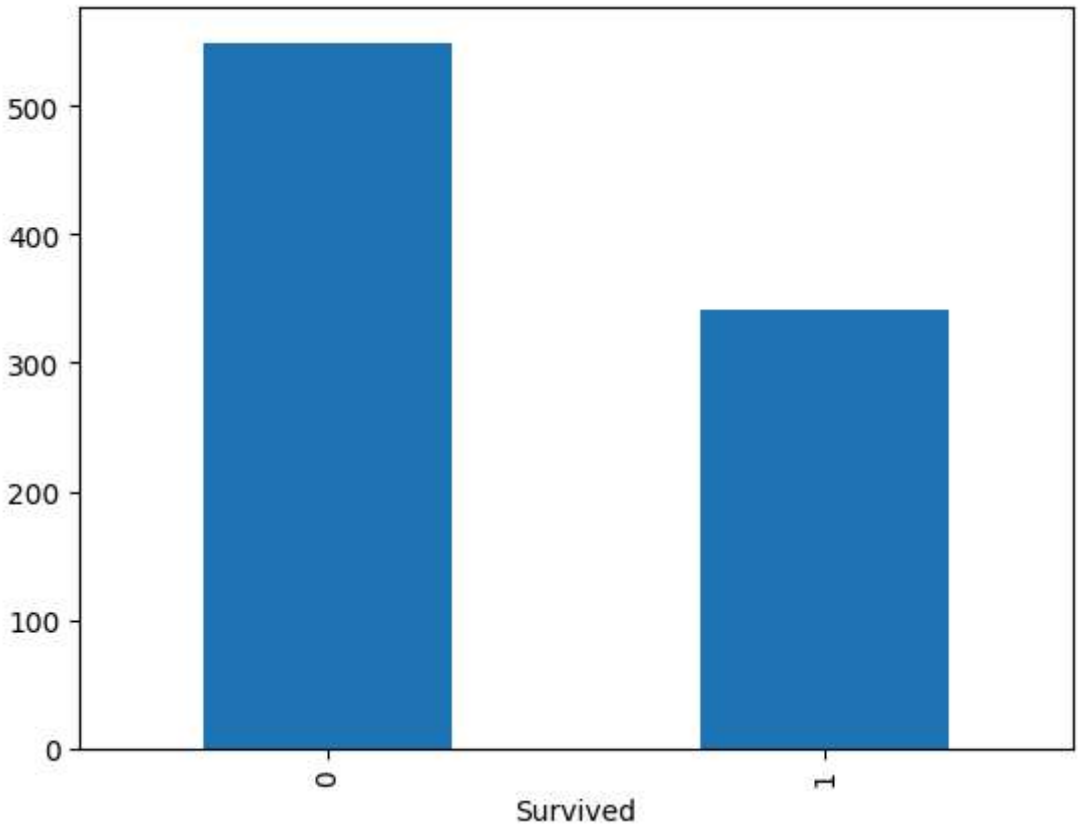
Create a new col called is_alone

In [44]:
```
df['Survived'].value_counts()
```

Out[44]:
```
Survived
0    549
1    342
Name: count, dtype: int64
```

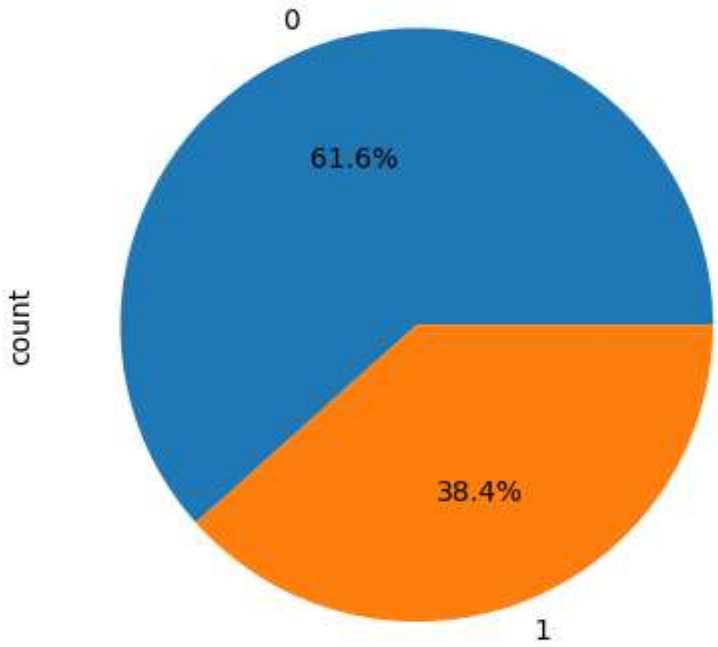
In [45]:
```
df['Survived'].value_counts().plot(kind='bar')
```

Out[45]:
```
<Axes: xlabel='Survived'>
```



In [50]:
```
df['Survived'].value_counts().plot(kind='pie', autopct='%0.1f%%')
```

Out[50]:
```
<Axes: ylabel='count'>
```



In [51]:
```
df['Survived'].isnull().sum()
```

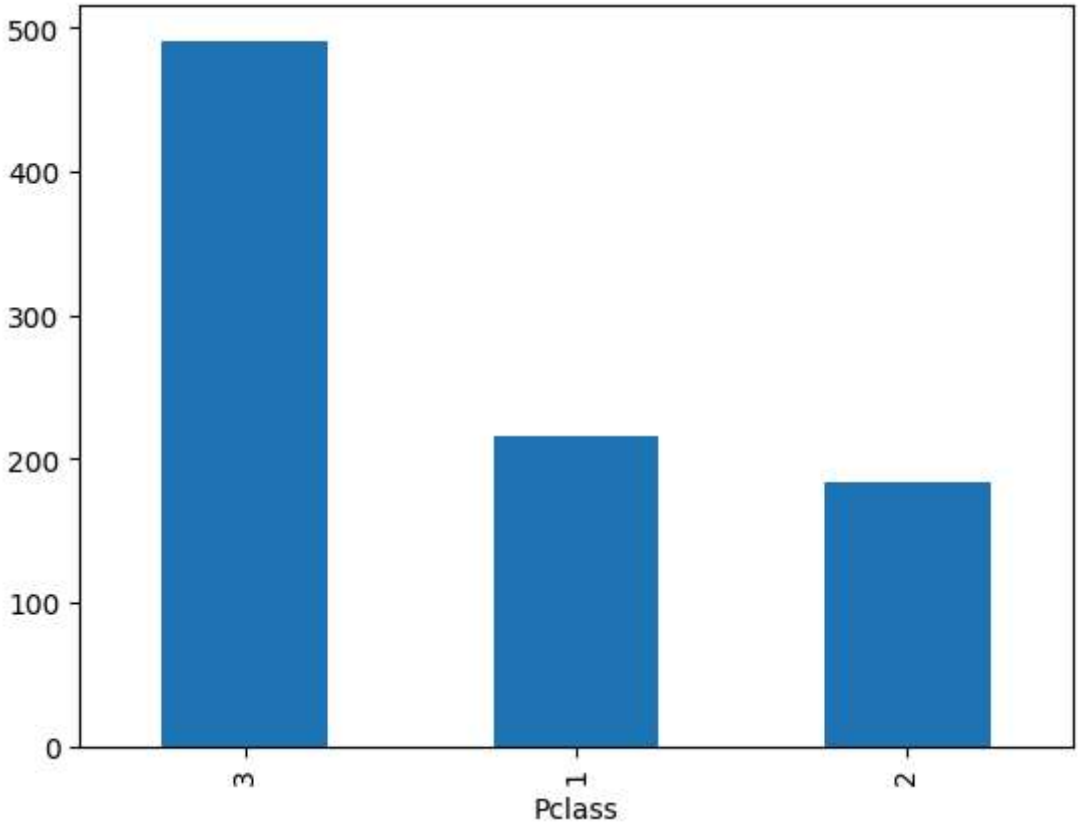
Out[51]:
```
0
```

In [52]:
```
df['Pclass'].value_counts()
```

Out[52]:
```
Pclass
3    491
1    216
2    184
Name: count, dtype: int64
```

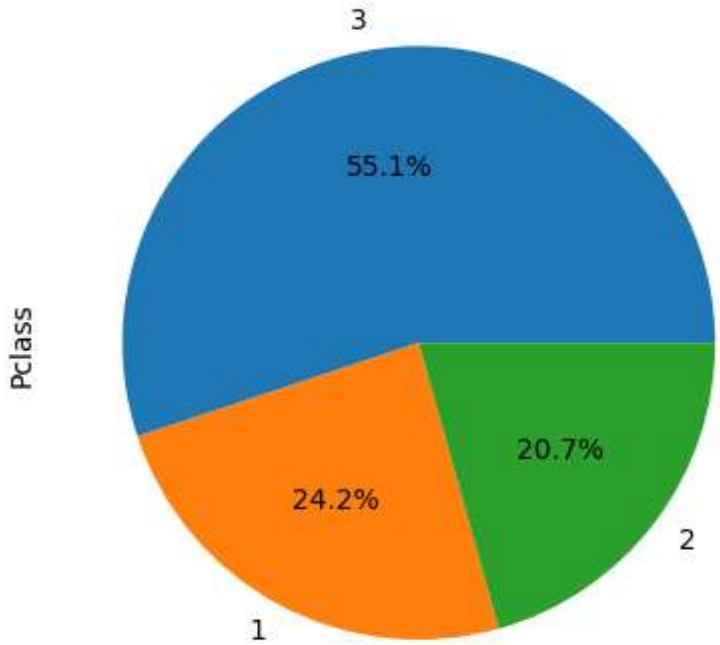
In [53]:
```
df['Pclass'].value_counts().plot(kind='bar')
```

Out[53]:
```
<Axes: xlabel='Pclass'>
```



In [57]:
```
df['Pclass'].value_counts().plot(kind='pie',autopct='%0.1f%%',label='Pclass')
```

Out[57]:
```
<Axes: ylabel='Pclass'>
```



In [60]:
```
df['SibSp'].value_counts()
```

Out[60]:
```
SibSp
0    608
1    209
2     28
4     18
3     16
8      7
5      5
Name: count, dtype: int64
```

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

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

In [62]:
```
df['Parch'].value_counts()
```

Out[62]:
```
Parch
0    678
1    118
2     80
5      5
3      5
4      4
6      1
Name: count, dtype: int64
```

In [ ]:

Steps of doing Bivariate Analysis

Select 2 cols

Understand type of relationship

Numerical - Numerical

a. You can plot graphs like scatterplot(regression plots), 2D histplot, 2D KDEplots

b. Check correlation coefficent to check linear relationship

Numerical - Categorical - create visualizations that compare the distribution of the numerical data across different categories of the categorical data.

a. You can plot graphs like barplot, boxplot, kdeplot violinplot even scatterplots

Categorical - Categorical

a. You can create cross-tabulations or contingency tables that show the distribution of values in one categorical column, grouped by the values in the other categorical column.

b. You can plots like heatmap, stacked barplots, treemaps

Write your conclusions

In [63]:
```
df.head()
```

Out[63]:

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

In [64]:
```
pd.crosstab(df['Survived'],df['Pclass'])
```

Out[64]:

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

In [65]:
```
pd.crosstab(df['Survived'],df['Pclass'],normalize='columns')
```

Out[65]:

Pclass	1	2	3
Survived			
0	0.37037	0.527174	0.757637
1	0.62963	0.472826	0.242363

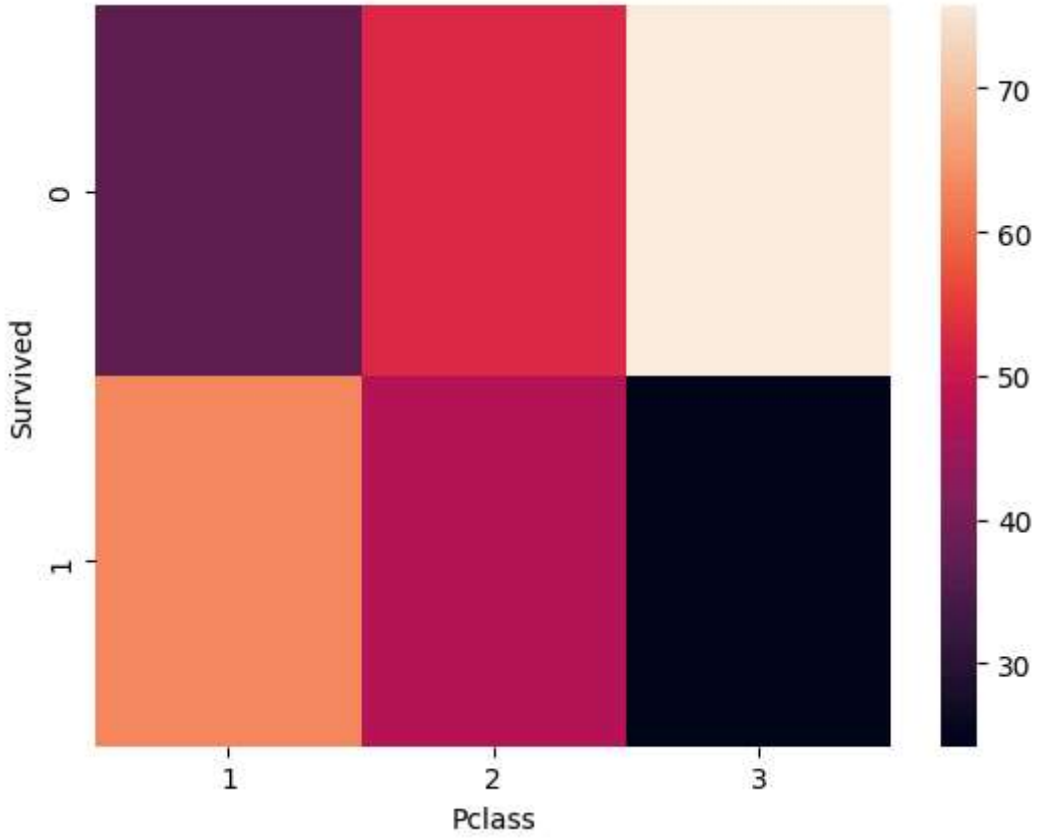
In [66]:
```
pd.crosstab(df['Survived'],df['Pclass'],normalize='columns')*100
```

Out[66]:

Pclass	1	2	3
Survived			
0	37.037037	52.717391	75.763747
1	62.962963	47.282609	24.236253

In [67]:
```
sns.heatmap(pd.crosstab(df['Survived'],df['Pclass'],normalize='columns')*100)
```

Out[67]:
```
<Axes: xlabel='Pclass', ylabel='Survived'>
```



In [68]:
```
pd.crosstab(df['Survived'],df['Sex'],normalize='columns')*100
```

Out[68]:

Sex	female	male
Survived		
0	25.796178	81.109185
1	74.203822	18.890815

In [70]:
```
sns.heatmap(pd.crosstab(df['Survived'],df['Sex'],normalize='columns')*100)
```

Out[70]:
```
<Axes: xlabel='Sex', ylabel='Survived'>
```

In [71]:
```
pd.crosstab(df['Survived'],df['Embarked'],normalize='columns')*100
```

Out[71]:

Embarked	C	Q	S
Survived			
0	44.642857	61.038961	66.304348
1	55.357143	38.961039	33.695652

In [72]:
```
pd.crosstab(df['Sex'],df['Embarked'],normalize='columns')*100
```

Out[72]:

Embarked	C	Q	S
Sex			
female	43.452381	46.753247	31.521739
male	56.547619	53.246753	68.478261

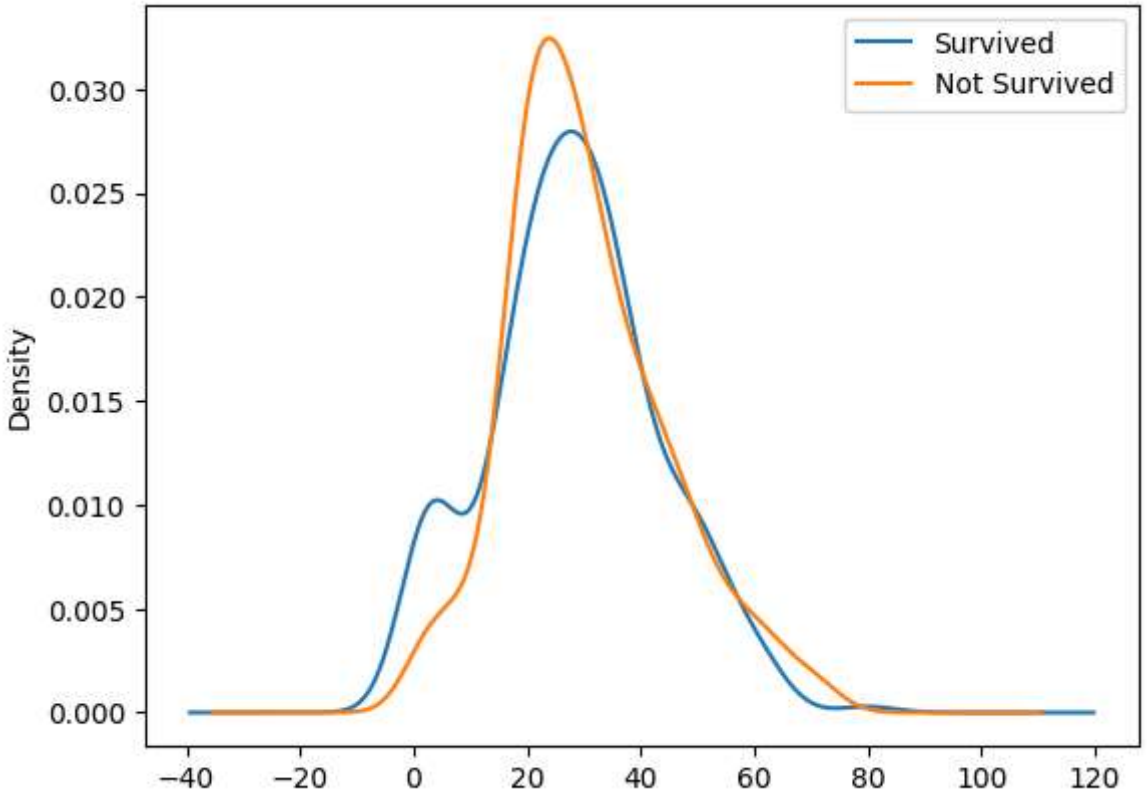
In [73]:
```
pd.crosstab(df['Pclass'],df['Embarked'],normalize='columns')*100
```

Out[73]:

Embarked	C	Q	S
Pclass			
1	50.595238	2.597403	19.720497
2	10.119048	3.896104	25.465839
3	39.285714	93.506494	54.813665

In [79]:
```
# survived and age
df[df['Survived']==1]['Age'].plot(kind='kde',label='Survived')
df[df['Survived']==0]['Age'].plot(kind='kde',label='Not Survived')

plt.legend()
plt.show()
```



In [80]:
```
df[df['Pclass']==1]['Age'].mean()
```

Out[80]:
```
38.233440860215055
```

In [81]:
```
df['SibSp'].value_counts()
```

Out[81]:
```
SibSp
0    608
1    209
2     28
4     18
3     16
8      7
5      5
Name: count, dtype: int64
```

In [82]:
```
df[df['SibSp']==8]
```

Out[82]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
159	160	0	3	Sage, Master. Thomas Henry	male	NaN	8	2	CA. 2343	69.55	NaN	S
180	181	0	3	Sage, Miss. Constance Gladys	female	NaN	8	2	CA. 2343	69.55	NaN	S
201	202	0	3	Sage, Mr. Frederick	male	NaN	8	2	CA. 2343	69.55	NaN	S
324	325	0	3	Sage, Mr. George John Jr	male	NaN	8	2	CA. 2343	69.55	NaN	S
792	793	0	3	Sage, Miss. Stella Anna	female	NaN	8	2	CA. 2343	69.55	NaN	S
846	847	0	3	Sage, Mr. Douglas Bullen	male	NaN	8	2	CA. 2343	69.55	NaN	S
863	864	0	3	Sage, Miss. Dorothy Edith "Dolly"	female	NaN	8	2	CA. 2343	69.55	NaN	S

In [83]:
```
df[df['Ticket']=='CA. 2343']
```

Out[83]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
159	160	0	3	Sage, Master. Thomas Henry	male	NaN	8	2	CA. 2343	69.55	NaN	S
180	181	0	3	Sage, Miss. Constance Gladys	female	NaN	8	2	CA. 2343	69.55	NaN	S
201	202	0	3	Sage, Mr. Frederick	male	NaN	8	2	CA. 2343	69.55	NaN	S
324	325	0	3	Sage, Mr. George John Jr	male	NaN	8	2	CA. 2343	69.55	NaN	S
792	793	0	3	Sage, Miss. Stella Anna	female	NaN	8	2	CA. 2343	69.55	NaN	S
846	847	0	3	Sage, Mr. Douglas Bullen	male	NaN	8	2	CA. 2343	69.55	NaN	S
863	864	0	3	Sage, Miss. Dorothy Edith "Dolly"	female	NaN	8	2	CA. 2343	69.55	NaN	S

In [84]:
```
df1=pd.read_csv('test.csv')
```

In [85]:
```
df1.head()
```

Out[85]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

In [88]:
```
df= pd.concat([df,df1])
```

In [89]:
```
df
```

Out[89]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1.0	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1.0	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0.0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
413	1305	NaN	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	1306	NaN	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	NaN	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	NaN	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	NaN	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

1309 rows × 12 columns

In [90]:
```
df[df['Ticket']=='CA. 2343']
```

Out[90]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
159	160	0.0	3	Sage, Master. Thomas Henry	male	NaN	8	2	CA. 2343	69.55	NaN	S
180	181	0.0	3	Sage, Miss. Constance Gladys	female	NaN	8	2	CA. 2343	69.55	NaN	S
201	202	0.0	3	Sage, Mr. Frederick	male	NaN	8	2	CA. 2343	69.55	NaN	S
324	325	0.0	3	Sage, Mr. George John Jr	male	NaN	8	2	CA. 2343	69.55	NaN	S
792	793	0.0	3	Sage, Miss. Stella Anna	female	NaN	8	2	CA. 2343	69.55	NaN	S
846	847	0.0	3	Sage, Mr. Douglas Bullen	male	NaN	8	2	CA. 2343	69.55	NaN	S
863	864	0.0	3	Sage, Miss. Dorothy Edith "Dolly"	female	NaN	8	2	CA. 2343	69.55	NaN	S
188	1080	NaN	3	Sage, Miss. Ada	female	NaN	8	2	CA. 2343	69.55	NaN	S
342	1234	NaN	3	Sage, Mr. John George	male	NaN	1	9	CA. 2343	69.55	NaN	S
360	1252	NaN	3	Sage, Master. William Henry	male	14.5	8	2	CA. 2343	69.55	NaN	S
365	1257	NaN	3	Sage, Mrs. John (Annie Bullen)	female	NaN	1	9	CA. 2343	69.55	NaN	S

In [92]:
```
df['Ticket'].value_counts()
```

Out[92]:
```
Ticket
CA. 2343        11
CA 2144          8
1601             8
PC 17608         7
S.O.C. 14879     7
                ..
113792           1
36209            1
323592           1
315089           1
359309           1
Name: count, Length: 929, dtype: int64
```

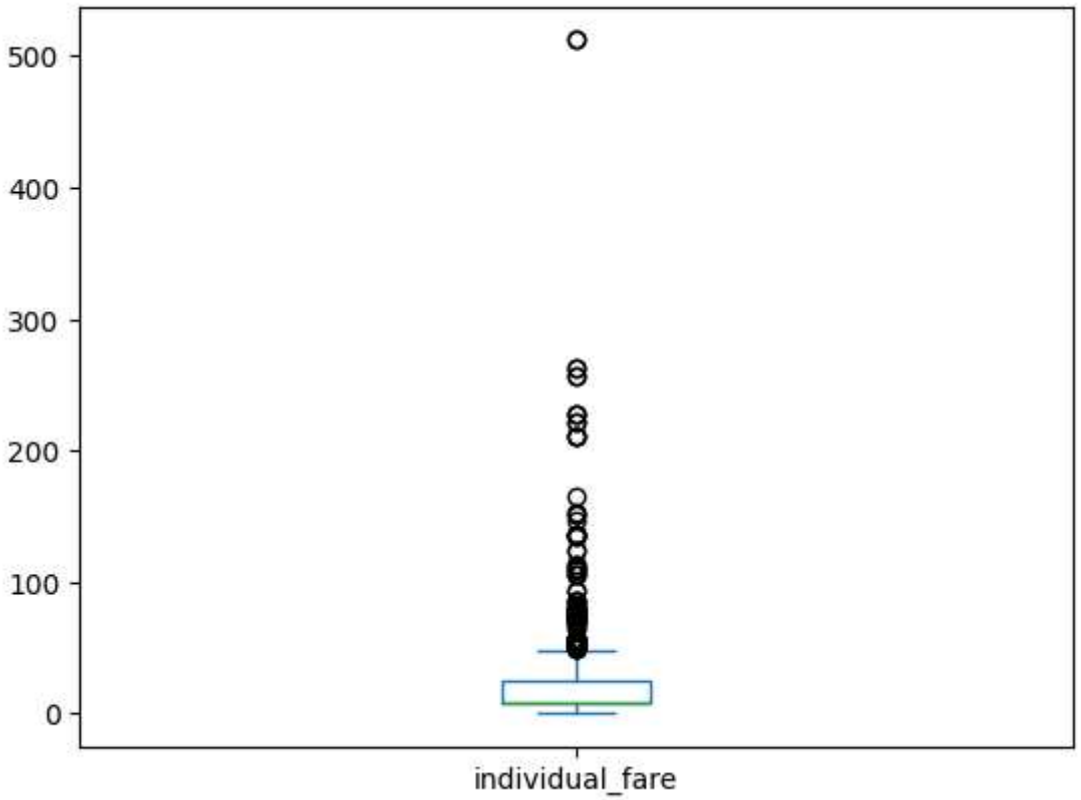
In [97]:
```
df['individual_fare']=df['Fare']/(df['SibSp'] + df['Parch'] + 1)
```

In [98]:
```
df['individual_fare']
```

Out[98]:
```
0        3.625000
1       35.641650
2        7.925000
3       26.550000
4        8.050000
          ...
413      8.050000
414    108.900000
415      7.250000
416      8.050000
417      7.452767
Name: individual_fare, Length: 1309, dtype: float64
```

In [99]:
```
df['individual_fare'].plot(kind='box')
```

Out[99]:
```
<Axes: >
```



In [100…
```
df['individual_fare'].describe()
```

Out[100]:
```
count    1308.000000
mean       20.518215
std        35.774337
min         0.000000
25%         7.452767
50%         8.512483
75%        24.237500
max       512.329200
Name: individual_fare, dtype: float64
```

In [101…
```
df[['individual_fare','Fare']].describe()
```

Out[101]:

	individual_fare	Fare
count	1308.000000	1308.000000
mean	20.518215	33.295479
std	35.774337	51.758668
min	0.000000	0.000000
25%	7.452767	7.895800
50%	8.512483	14.454200
75%	24.237500	31.275000
max	512.329200	512.329200

In [103…
```
df['family_size']=df['SibSp'] + df['Parch'] + 1
```

In [105…
```
df
```

Out[105]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	individual_fare	family_si
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	3.625000	
1	2	1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	35.641650	
2	3	1.0	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	7.925000	
3	4	1.0	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	26.550000	
4	5	0.0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	8.050000	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
413	1305	NaN	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S	8.050000	
414	1306	NaN	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C	108.900000	
415	1307	NaN	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S	7.250000	
416	1308	NaN	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S	8.050000	
417	1309	NaN	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C	7.452767	

1309 rows × 14 columns

In [107…
```python
# family type
# 1 -> alone
# 2-4 -> small
# >5 -> large
def transform_family_size(num):
    if num==1:
        return "alone"
    elif num>1 and num<5:
        return "small"
    else:
        return "large"
```

In [109…
```python
df['family_type'] = df['family_size'].apply(transform_family_size)
```

In [110…
```python
df
```

Out[110]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	individual_fare	family_si
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	3.625000	
1	2	1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	35.641650	
2	3	1.0	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	7.925000	
3	4	1.0	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	26.550000	
4	5	0.0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	8.050000	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
413	1305	NaN	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S	8.050000	
414	1306	NaN	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C	108.900000	
415	1307	NaN	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S	7.250000	
416	1308	NaN	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S	8.050000	
417	1309	NaN	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C	7.452767	

1309 rows × 15 columns

In [111…
```
pd.crosstab(df['Survived'], df['family_type'])
```

Out[111]:

Survived	alone	large	small
0.0	374	52	123
1.0	163	10	169

In [113…
```
pd.crosstab(df['Survived'], df['family_type'], normalize='columns')*100
```

Out[113]:

Survived	alone	large	small
0.0	69.646182	83.870968	42.123288
1.0	30.353818	16.129032	57.876712

In [115…
```
df['Name'].str.split(',').str.get(0)
```

Out[115]:
```
0             Braund
1            Cumings
2          Heikkinen
3           Futrelle
4              Allen
            ...
413          Spector
414    Oliva y Ocana
415          Saether
416             Ware
417            Peter
Name: Name, Length: 1309, dtype: object
```

In [116…
```
df['surname'] = df['Name'].str.split(',').str.get(0)
```

In [117…
```
df
```

Out[117]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	individual_fare	family_si
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	3.625000	
1	2	1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	35.641650	
2	3	1.0	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	7.925000	
3	4	1.0	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	26.550000	
4	5	0.0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	8.050000	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
413	1305	NaN	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S	8.050000	
414	1306	NaN	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C	108.900000	
415	1307	NaN	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S	7.250000	
416	1308	NaN	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S	8.050000	
417	1309	NaN	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C	7.452767	

1309 rows × 16 columns

In [122…
```
df['Name'].str.split(',').str.get(1).str.strip().str.split(' ').str.get(0)
```

Out[122]:
```
0          Mr.
1         Mrs.
2        Miss.
3         Mrs.
4          Mr.
        ...
413        Mr.
414      Dona.
415        Mr.
416        Mr.
417    Master.
Name: Name, Length: 1309, dtype: object
```

In [123…
```
df['title']=df['Name'].str.split(',').str.get(1).str.strip().str.split(' ').str.get(0)
```

In [125…
```
df['title'].value_counts()
```

Out[125]:
```
title
Mr.          757
Miss.        260
Mrs.         197
Master.       61
Rev.           8
Dr.            8
Col.           4
Mlle.          2
Major.         2
Ms.            2
Lady.          1
Sir.           1
Mme.           1
Don.           1
Capt.          1
the            1
Jonkheer.      1
Dona.          1
Name: count, dtype: int64
```

In [126…
```
df
```

Out[126]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	individual_fare	family_si
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	3.625000	
1	2	1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	35.641650	
2	3	1.0	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	7.925000	
3	4	1.0	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	26.550000	
4	5	0.0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	8.050000	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
413	1305	NaN	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S	8.050000	
414	1306	NaN	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C	108.900000	
415	1307	NaN	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S	7.250000	
416	1308	NaN	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S	8.050000	
417	1309	NaN	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C	7.452767	

1309 rows × 17 columns

In [127…
```
df['Cabin'].isnull().sum()
```

Out[127]:
```
1014
```

In [129…
```
df['Cabin'].fillna('M', inplace=True)
```

In [130…
```
df['Cabin'].value_counts
```

Out[130]:
```
<bound method IndexOpsMixin.value_counts of 0         M
1       C85
2         M
3      C123
4         M
       ...
413       M
414    C105
415       M
416       M
417       M
Name: Cabin, Length: 1309, dtype: object>
```

In [131…
```
df['deck']=df['Cabin'].str[0]
```

In [134…
```
df['deck'].value_counts()
```

Out[134]:
```
deck
M    1014
C      94
B      65
D      46
E      41
A      22
F      21
G       5
T       1
Name: count, dtype: int64
```

In [135…
```
pd.crosstab(df['deck'],df['Pclass'])
```

Out[135]:

Pclass	1	2	3
deck			
A	22	0	0
B	65	0	0
C	94	0	0
D	40	6	0
E	34	4	3
F	0	13	8
G	0	0	5
M	67	254	693
T	1	0	0

In [137…

```
pd.crosstab(df['Survived'],df['deck'], normalize='columns')*100
```

Out[137]:

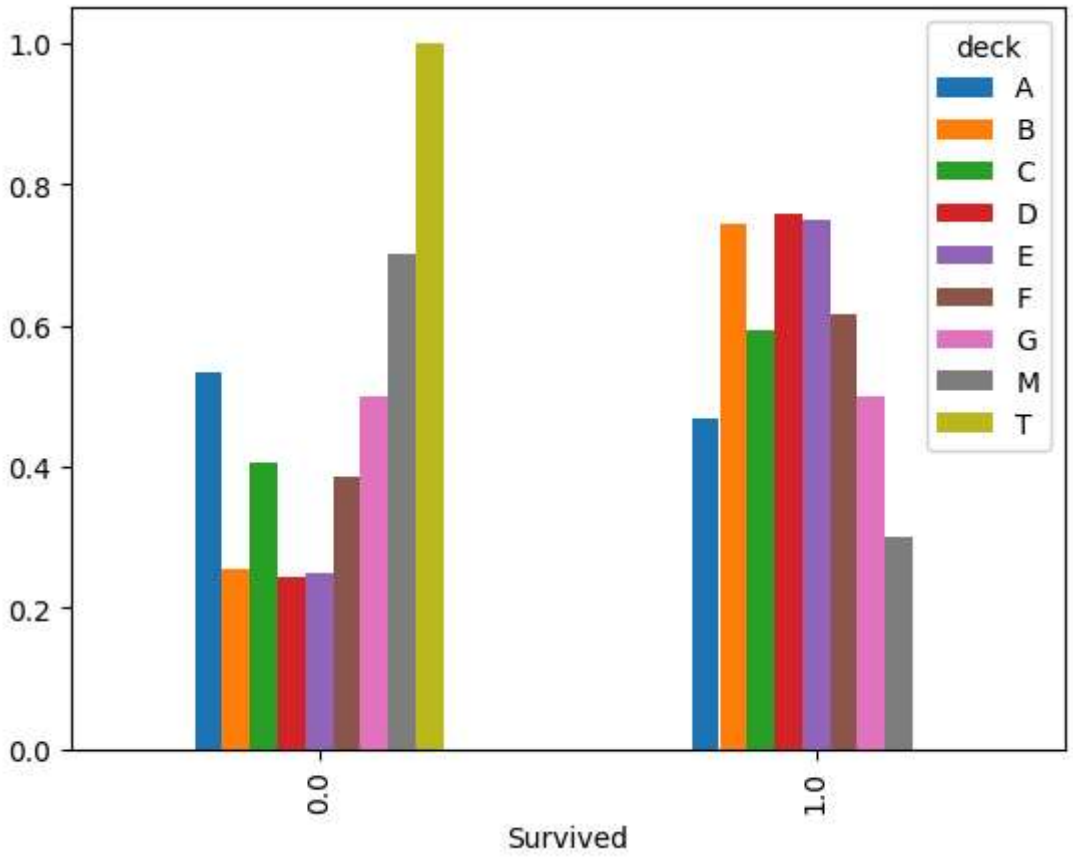
deck	A	B	C	D	E	F	G	M	T
Survived									
0.0	53.333333	25.531915	40.677966	24.242424	25.0	38.461538	50.0	70.014556	100.0
1.0	46.666667	74.468085	59.322034	75.757576	75.0	61.538462	50.0	29.985444	0.0

In [138…

```
pd.crosstab(df['Survived'],df['deck'], normalize='columns').plot(kind='bar')
```

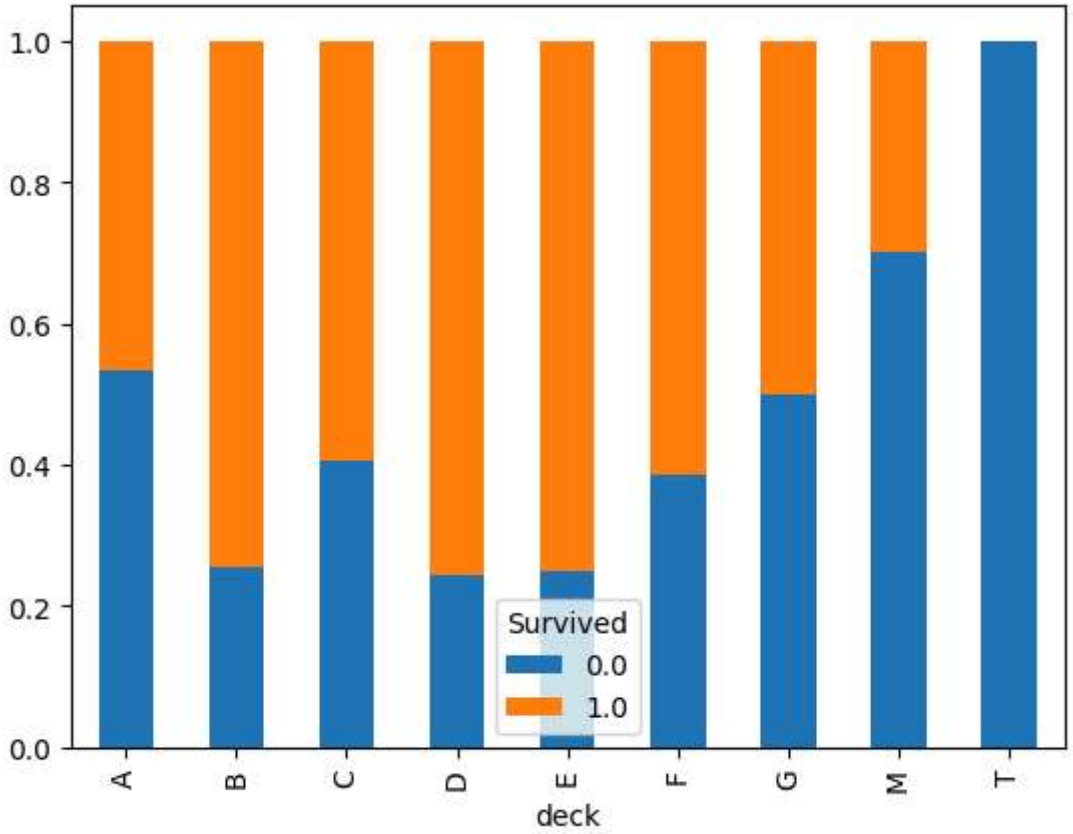
Out[138]:

```
<Axes: xlabel='Survived'>
```



In [140…
```
pd.crosstab(df['deck'],df['Survived'], normalize='index').plot(kind='bar', stacked=True)
```

Out[140]:
```
<Axes: xlabel='deck'>
```



In [143…

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21380\3086603635.py in ?()
----> 1 df.drop(df['Name']).corr()

~\anaconda3\envs\ML_project\Lib\site-packages\pandas\core\frame.py in ?(self, labels, axis, index, columns, level, inplace, errors)
   5343                 weight  250.0   150.0
   5344         falcon  speed   320.0   250.0
   5345                 weight  1.0     0.8
   5346         """
-> 5347         return super().drop(
   5348             labels=labels,
   5349             axis=axis,
   5350             index=index,

~\anaconda3\envs\ML_project\Lib\site-packages\pandas\core\generic.py in ?(self, labels, axis, index, columns, level, inplace, errors)
   4707         obj = self
   4708
   4709         for axis, labels in axes.items():
   4710             if labels is not None:
-> 4711                 obj = obj._drop_axis(labels, axis, level=level, errors=errors)
   4712
   4713         if inplace:
   4714             self._update_inplace(obj)

~\anaconda3\envs\ML_project\Lib\site-packages\pandas\core\generic.py in ?(self, labels, axis, level, errors, only_slice)
   4778                 mask = ~axis.isin(labels)
   4779                 # Check if label doesn't exist along axis
   4780                 labels_missing = (axis.get_indexer_for(labels) == -1).any()
   4781                 if errors == "raise" and labels_missing:
-> 4782                     raise KeyError(f"{labels} not found in axis")
   4783
   4784             if isinstance(mask.dtype, ExtensionDtype):
   4785                 # GH#45860

KeyError: "['Braund, Mr. Owen Harris'\n 'Cumings, Mrs. John Bradley (Florence Briggs Thayer)'\n 'Heikkinen, Miss. Laina' ... 'Saether, Mr. Simon Sivertsen'\n 'Ware, Mr. Frederick' 'Peter, Master. Michael J'] not found in axis"
```

In [144…

```
df
```

Out[144]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	individual_fare	family_si
0	1	0.0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	M	S	3.625000	
1	2	1.0	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	35.641650	
2	3	1.0	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	M	S	7.925000	
3	4	1.0	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	26.550000	
4	5	0.0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	M	S	8.050000	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
413	1305	NaN	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	M	S	8.050000	
414	1306	NaN	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C	108.900000	
415	1307	NaN	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	M	S	7.250000	
416	1308	NaN	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	M	S	8.050000	
417	1309	NaN	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	M	C	7.452767	

1309 rows × 18 columns

In [147…
```
df = df[['PassengerId','Survived','Pclass','Age','SibSp','Parch','Fare','individual_fare','family_size']]
```

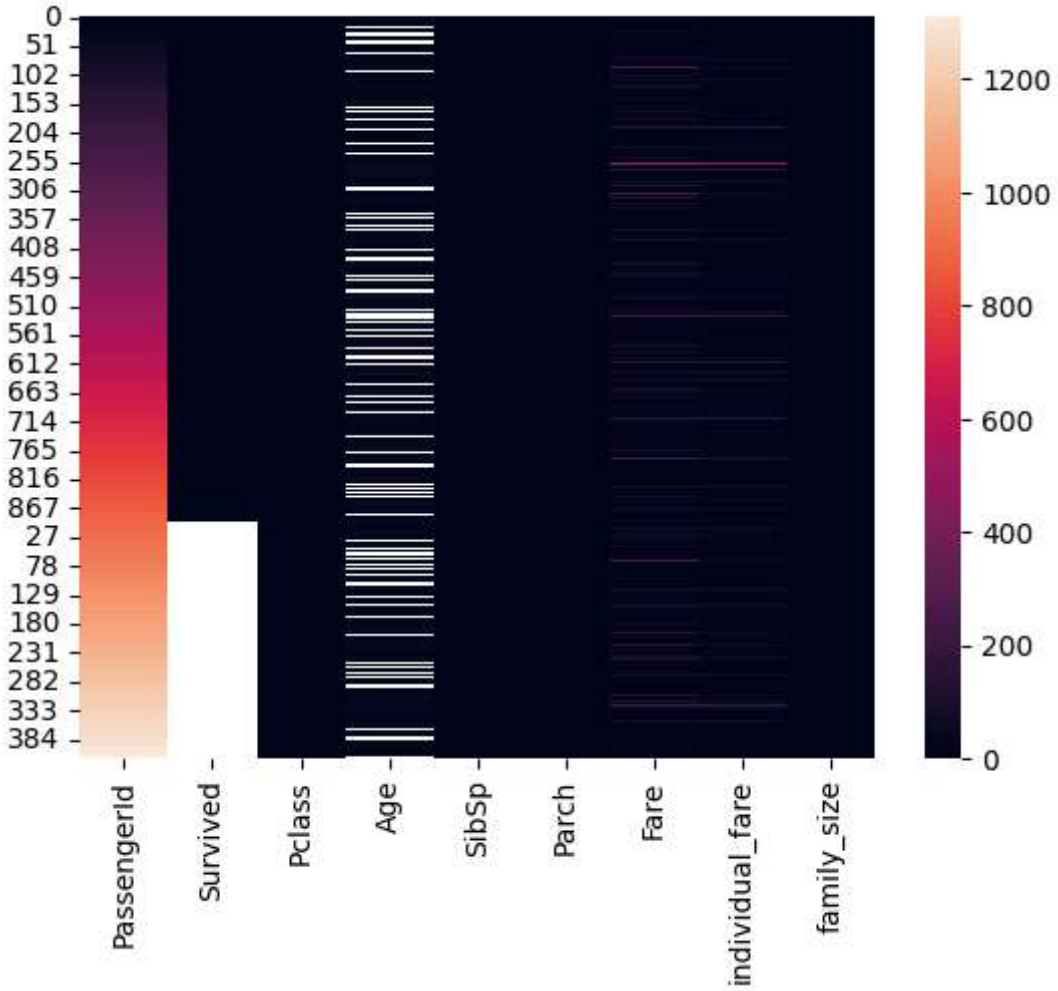
In [148…
```
df.corr()
```

Out[148]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	individual_fare	family_size
PassengerId	1.000000	-0.005007	-0.038354	0.028814	-0.055224	0.008942	0.031428	0.035365	-0.031437
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307	0.221600	0.016639
Pclass	-0.038354	-0.338481	1.000000	-0.408106	0.060832	0.018322	-0.558629	-0.504270	0.050027
Age	0.028814	-0.077221	-0.408106	1.000000	-0.243699	-0.150917	0.178740	0.193545	-0.240229
SibSp	-0.055224	-0.035322	0.060832	-0.243699	1.000000	0.373587	0.160238	-0.089807	0.861952
Parch	0.008942	0.081629	0.018322	-0.150917	0.373587	1.000000	0.221539	-0.065498	0.792296
Fare	0.031428	0.257307	-0.558629	0.178740	0.160238	0.221539	1.000000	0.832029	0.226492
individual_fare	0.035365	0.221600	-0.504270	0.193545	-0.089807	-0.065498	0.832029	1.000000	-0.094874
family_size	-0.031437	0.016639	0.050027	-0.240229	0.861952	0.792296	0.226492	-0.094874	1.000000

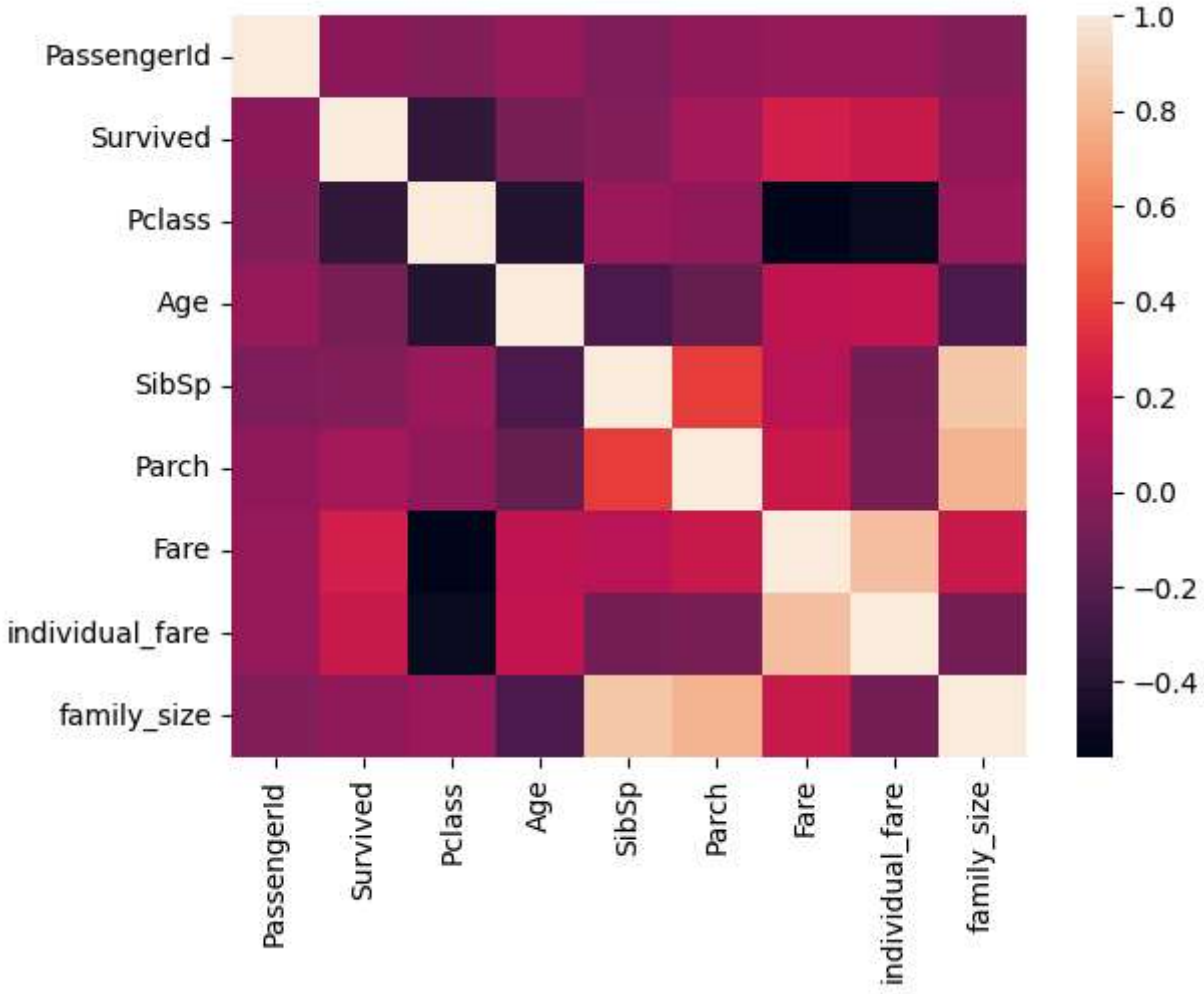
In [149…
```
sns.heatmap(df)
```

Out[149]:
```
<Axes: >
```



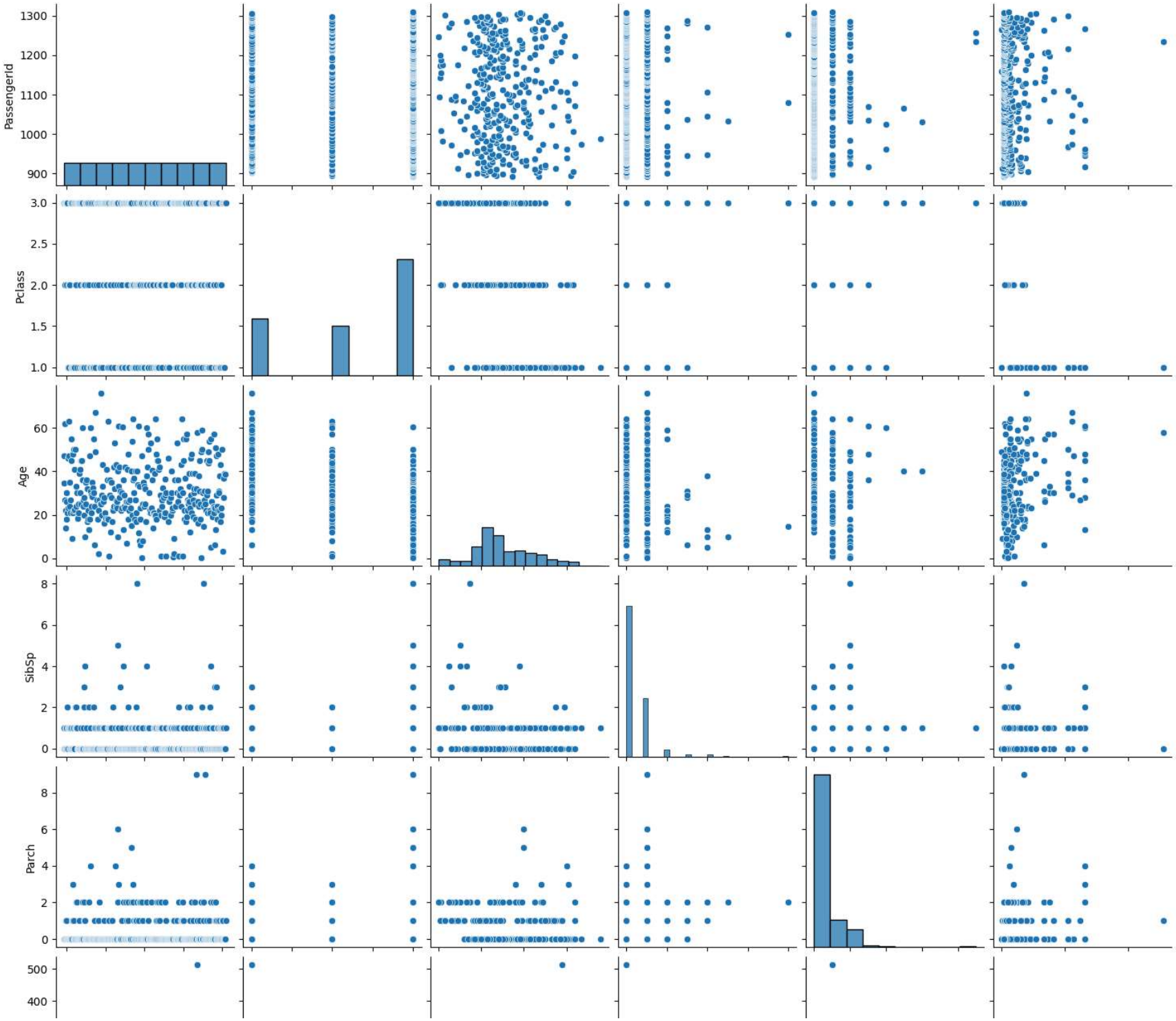
In [150…
```
sns.heatmap(df.corr())
```

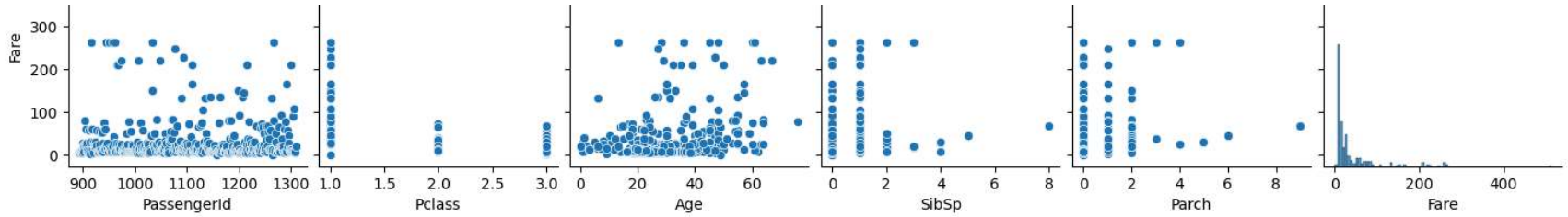
Out[150]: <Axes: >



In [152…
```
sns.pairplot(df1)
```

Out[152]:
```
<seaborn.axisgrid.PairGrid at 0x1e700edd510>
```


PassengerId
Pclass
Age
SibSp
Parch



In [4]:
```
print('python'[1+])
```

```
y
```

In [ ]:
```

```