```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
```

```
warnings.filterwarnings('ignore')
```

```
df = sns.load_dataset('titanic')
```

```
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	emba
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Sout
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Ch
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Sout
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Sout
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Sout

```
df.shape
```

```
(891, 15)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype   
---  ------       --------------  -----   
 0   survived     891 non-null    int64   
 1   pclass       891 non-null    int64   
 2   sex          891 non-null    object  
 3   age          714 non-null    float64 
 4   sibsp        891 non-null    int64   
 5   parch        891 non-null    int64   
 6   fare         891 non-null    float64 
 7   embarked     889 non-null    object  
 8   class        891 non-null    category
 9   who          891 non-null    object  
 10  adult_male   891 non-null    bool    
 11  deck         203 non-null    category
 12  embark_town  889 non-null    object  
 13  alive        891 non-null    object  
 14  alone        891 non-null    bool    
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
```

```
df.isnull().sum()
```

```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

```
df.describe()
```

	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

```
df['age'].fillna(df['age'].mean(), inplace=True)
```

```
df.isnull().sum()
```

```
survived         0
pclass           0
sex              0
age              0
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

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

```
S    644
C    168
Q     77
Name: embarked, dtype: int64
```

```
df['embarked'].fillna('S', inplace=True)
```

```
df.isnull().sum()
```

```
survived         0
pclass           0
sex              0
age              0
sibsp            0
parch            0
fare             0
embarked         0
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

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

```
C    59
B    47
D    33
E    32
A    15
F    13
G     4
Name: deck, dtype: int64
```

```
df['deck'].fillna(method='ffill', inplace=True)
```

```
df.isnull().sum()
```

```
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
class          0
who            0
adult_male     0
deck           1
embark_town    2
alive          0
alone          0
dtype: int64
```

```
df['deck'].fillna(method='bfill', inplace=True)
```

```
df.isnull().sum()
```

```
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
class          0
who            0
adult_male     0
deck           0
embark_town    2
alive          0
alone          0
dtype: int64
```

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

```
Southampton    644
Cherbourg      168
Queenstown      77
Name: embark_town, dtype: int64
```

```
df['embark_town'].fillna('Southampton', inplace=True)
```

```
df.isnull().sum()
```

```
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
class          0
who            0
adult_male     0
deck           0
embark_town    0
alive          0
alone          0
dtype: int64
```

EDA (Exploratory Data Analysis)

Use the inbuilt dataset 'titanic'. The dataset contains 891 rows and contains information about the passengers who boarded the unfortunate Titanic ship. Use the Seaborn library to see if we can find any patterns in the data.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype   
---  ------       --------------  -----   
 0   survived     891 non-null    int64   
 1   pclass       891 non-null    int64   
 2   sex          891 non-null    object  
 3   age          891 non-null    float64 
 4   sibsp        891 non-null    int64   
 5   parch        891 non-null    int64   
 6   fare         891 non-null    float64 
 7   embarked     891 non-null    object  
 8   class        891 non-null    category
 9   who          891 non-null    object  
 10  adult_male   891 non-null    bool    
 11  deck         891 non-null    category
 12  embark_town  891 non-null    object  
 13  alive        891 non-null    object  
 14  alone        891 non-null    bool    
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
```

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

```
0    549
1    342
Name: survived, dtype: int64
```

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

```
3    491
1    216
2    184
Name: pclass, dtype: int64
```

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

```
male      577
female    314
Name: sex, dtype: int64
```

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

```
29.699118    177
24.000000     30
22.000000     27
18.000000     26
28.000000     25
            ... 
36.500000      1
55.500000      1
0.920000       1
23.500000      1
74.000000      1
Name: age, Length: 89, dtype: int64
```

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

```
0    608
1    209
2     28
4     18
3     16
8      7
5      5
Name: sibsp, dtype: int64
```

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

```
0    678
1    118
2     80
5      5
3      5
4      4
6      1
Name: parch, dtype: int64
```

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

     8.0500     43
     13.0000    42
     7.8958     38
     7.7500     34
     26.0000    31
                ..
     35.0000     1
     28.5000     1
     6.2375      1
     14.0000     1
     10.5167     1
     Name: fare, Length: 248, dtype: int64
```

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

     S    646
     C    168
     Q     77
     Name: embarked, dtype: int64
```

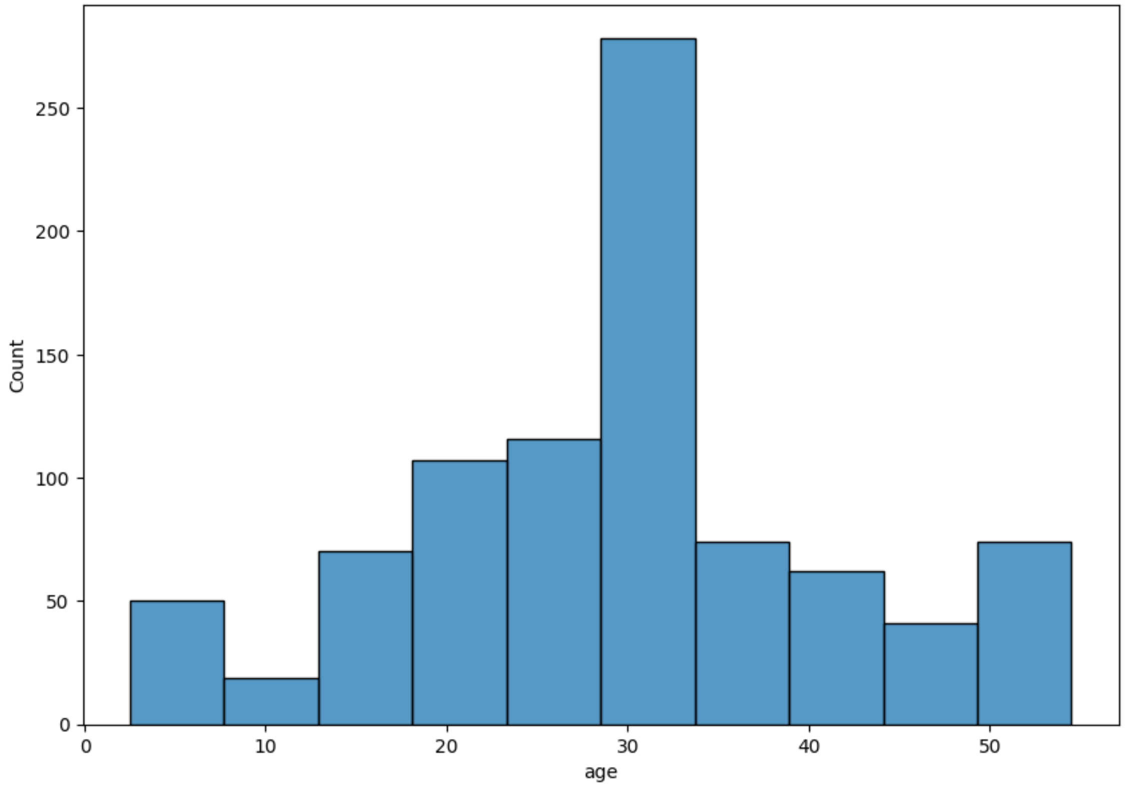
```
df['who'].value_counts()

     man      537
     woman    271
     child     83
     Name: who, dtype: int64
```

Write a code to check how the price of the ticket (column name: 'fare') for each passenger is distributed by plotting a histogram.

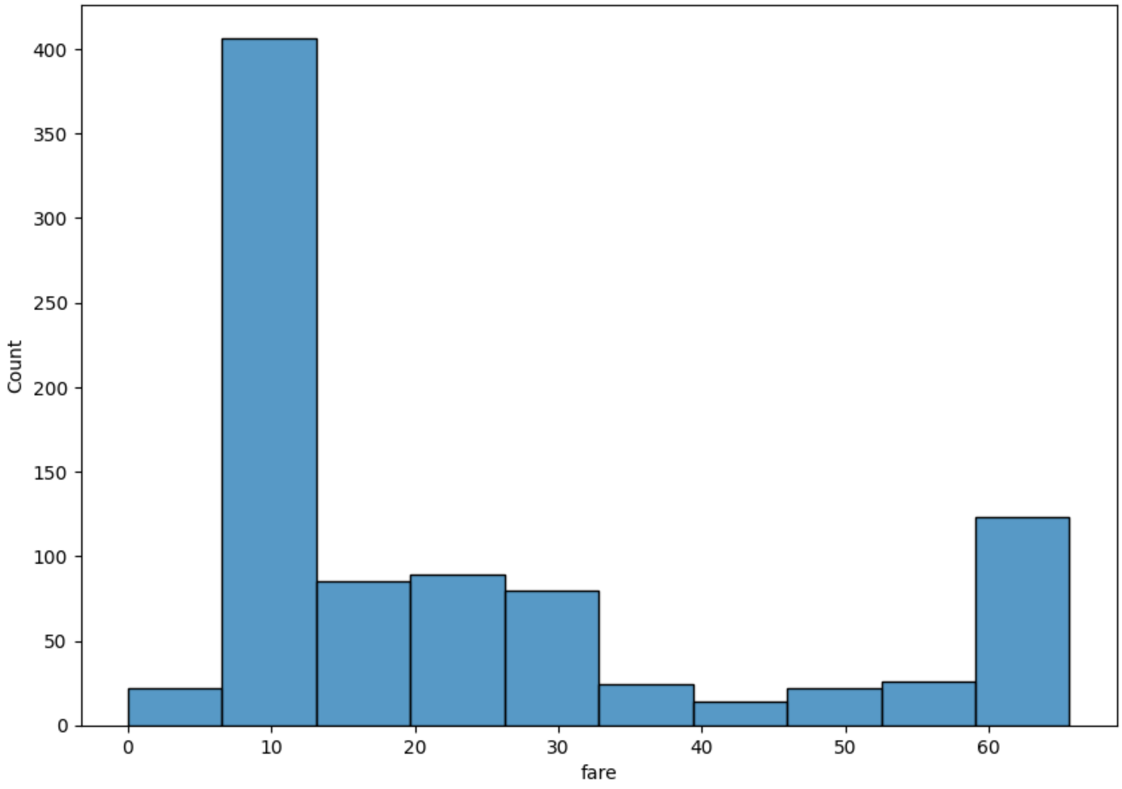
```
plt.figure(figsize = (10,7))
sns.histplot(df['age'],bins=10)

     <Axes: xlabel='age', ylabel='Count'>
```



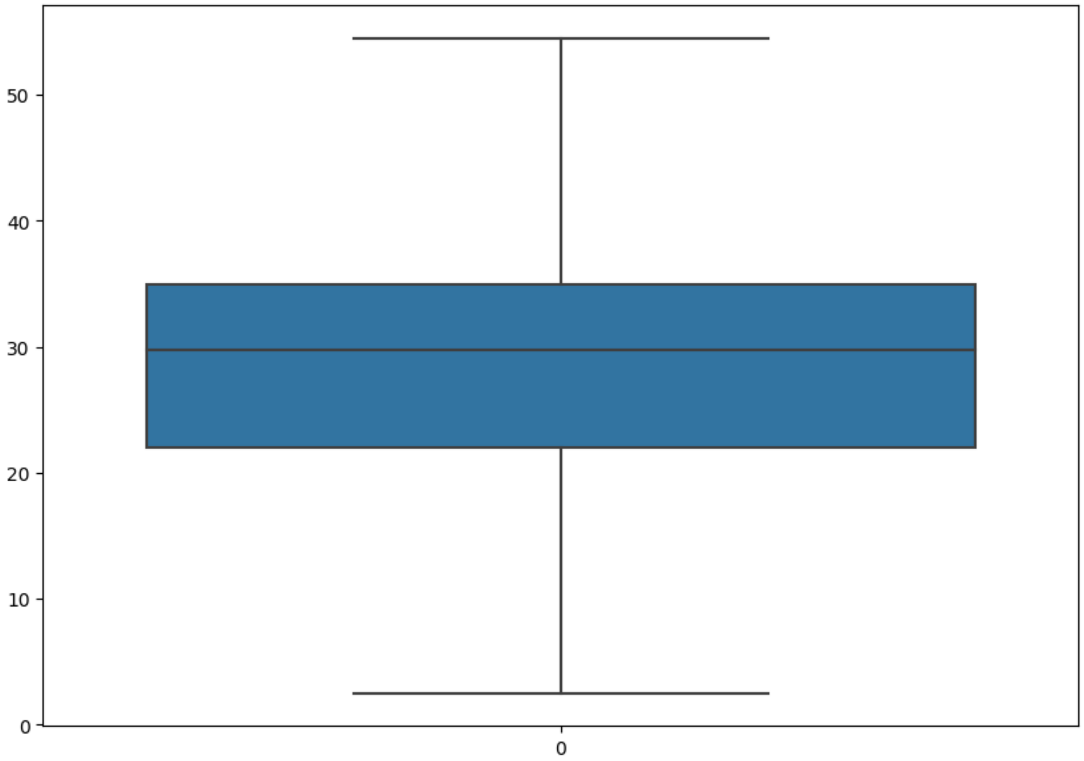
```
plt.figure(figsize = (10,7))
sns.histplot(df['fare'],bins = 10)
```

```
<Axes: xlabel='fare', ylabel='Count'>
```



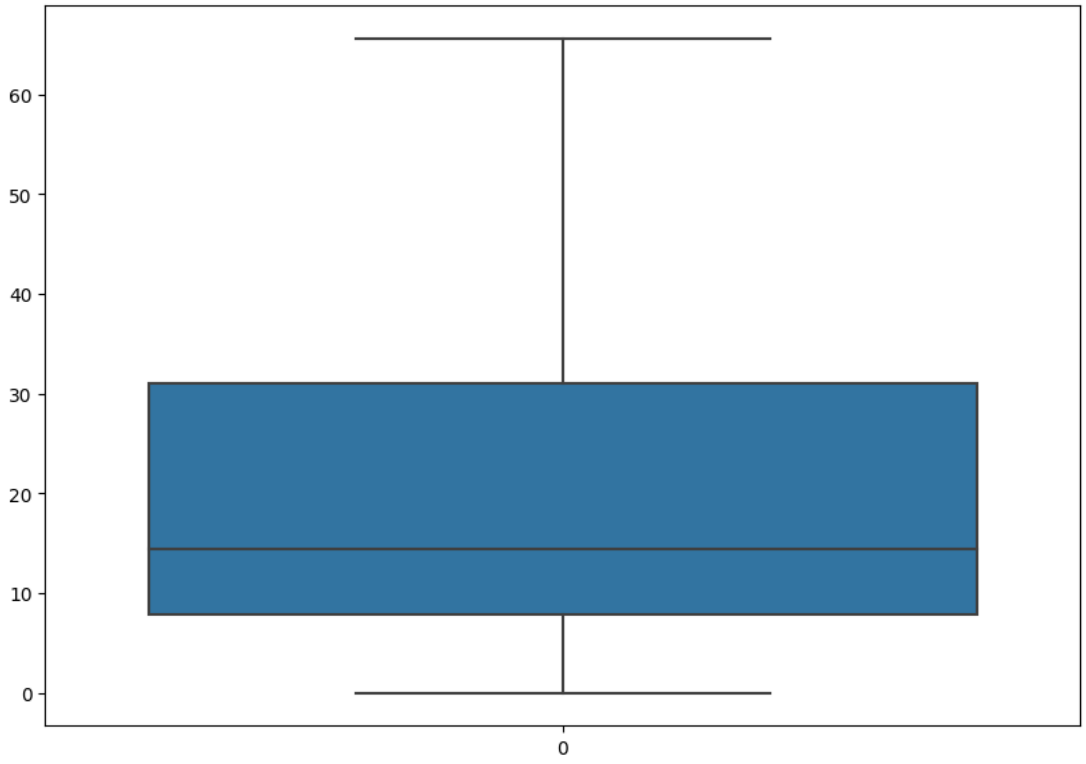
```
plt.figure(figsize=(10,7))
sns.boxplot(df['age'])
```

```
<Axes: >
```



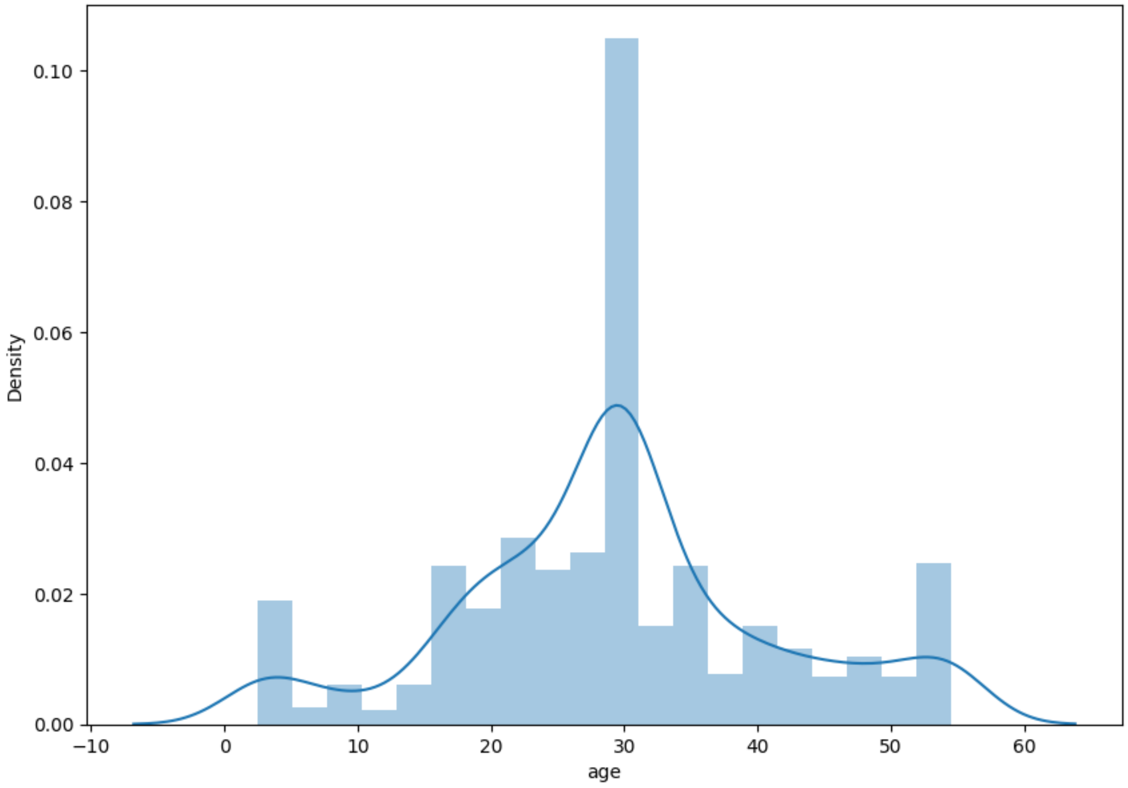
```
plt.figure(figsize=(10,7))
sns.boxplot(df['fare'])
```

```
<Axes: >
```



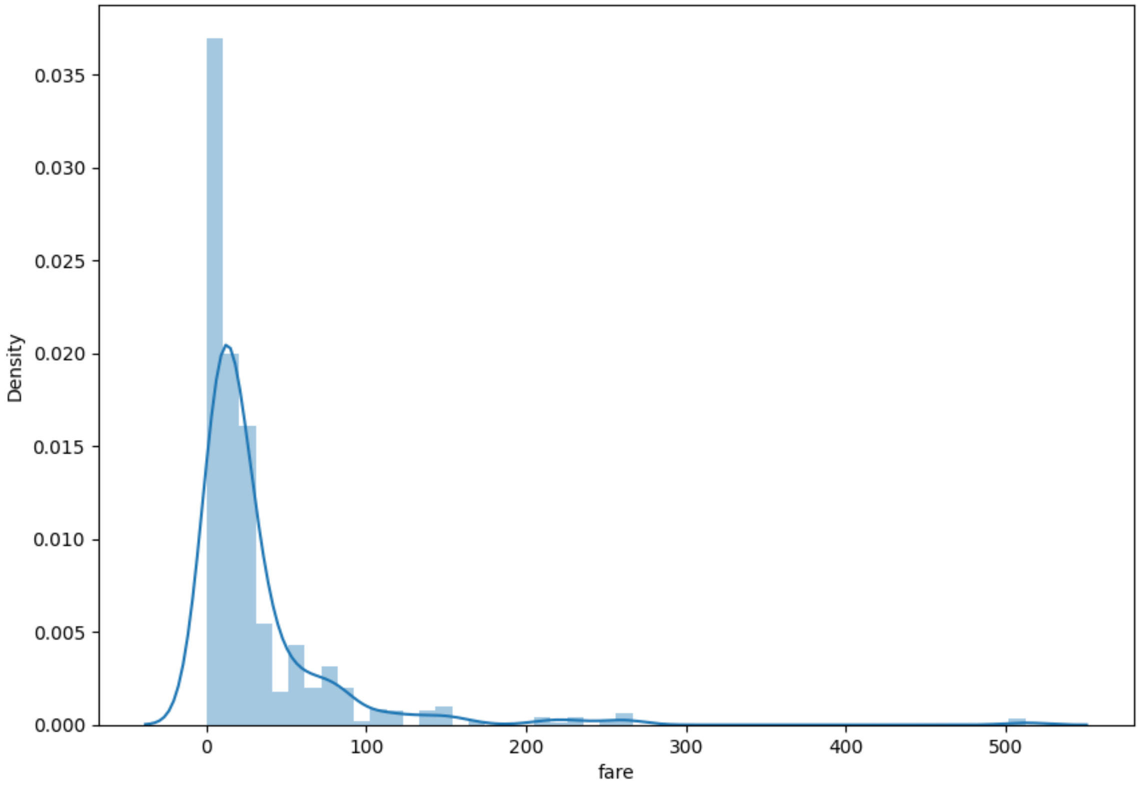
```
plt.figure(figsize=(10,7))
sns.distplot(df['age'])
```

```
<Axes: xlabel='age', ylabel='Density'>
```



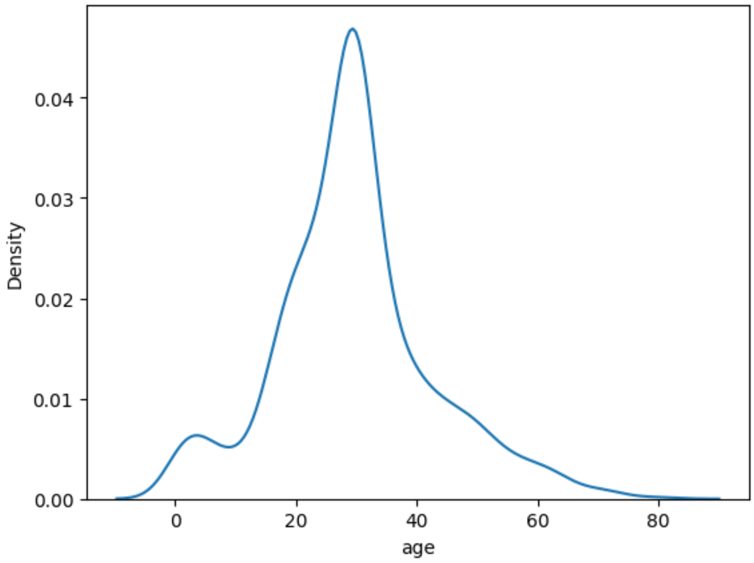
```
plt.figure(figsize=(10,7))
sns.distplot(df['fare'])
```

```
<Axes: xlabel='fare', ylabel='Density'>
```



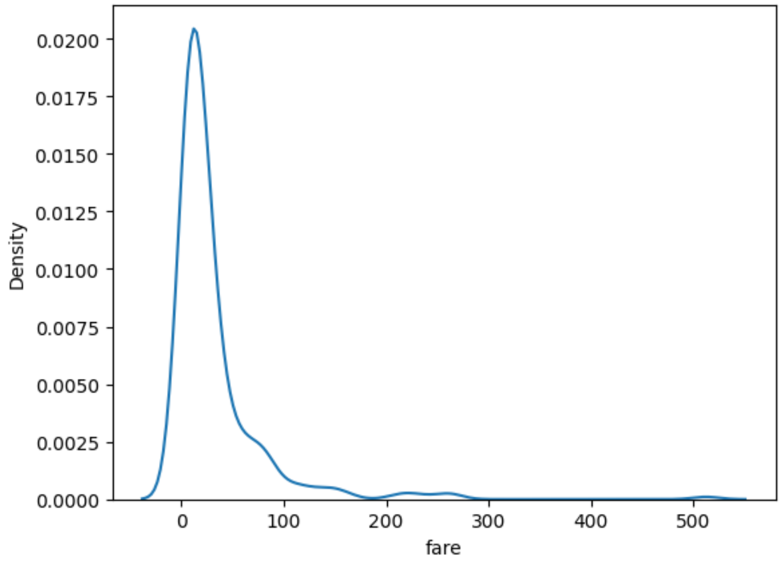
```
sns.kdeplot(df['age'])
```

```
<Axes: xlabel='age', ylabel='Density'>
```



```
sns.kdeplot(df['fare'])
```

```
<Axes: xlabel='fare', ylabel='Density'>
```



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype   
---  ------       --------------  -----   
 0   survived     891 non-null    int64   
 1   pclass       891 non-null    int64   
 2   sex          891 non-null    object  
 3   age          891 non-null    float64 
 4   sibsp        891 non-null    int64   
 5   parch        891 non-null    int64   
 6   fare         891 non-null    float64 
 7   embarked     891 non-null    object  
 8   class        891 non-null    category
 9   who          891 non-null    object  
 10  adult_male   891 non-null    bool    
 11  deck         891 non-null    category
 12  embark_town  891 non-null    object  
 13  alive        891 non-null    object  
 14  alone        891 non-null    bool    
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
```

```
df['age'].skew()
```

```
0.4344880940129925
```

```
df['fare'].skew()
```

```
4.787316519674893
```

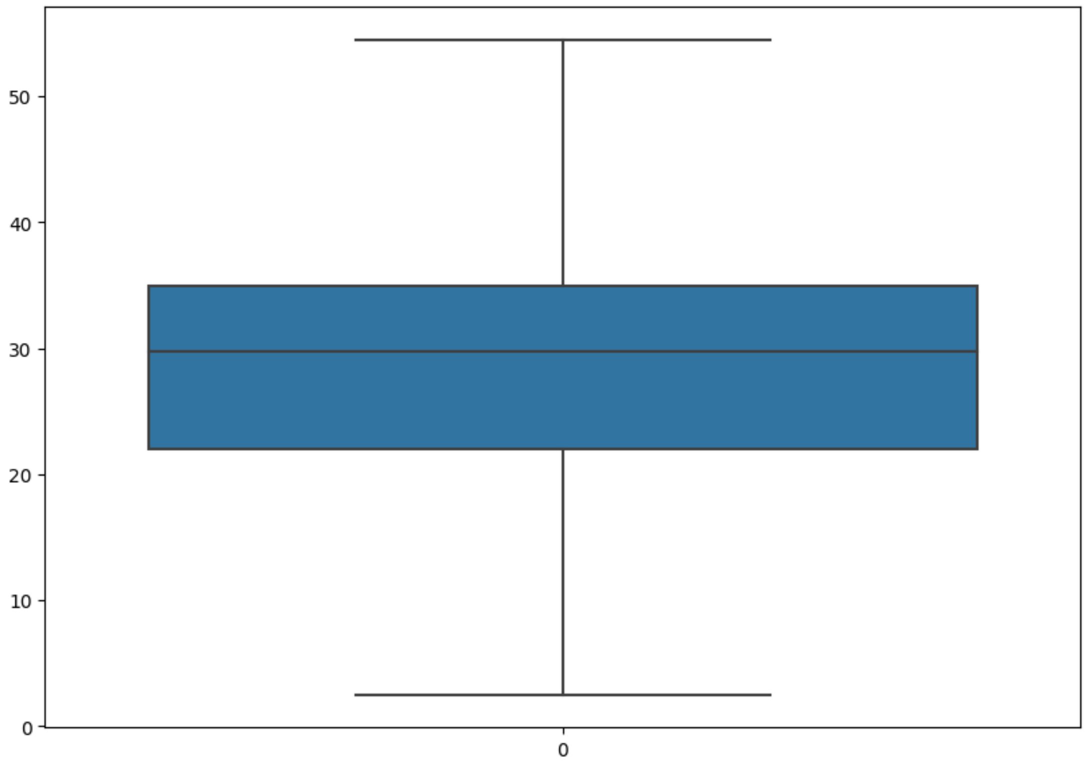
```
df[df['fare']>300]
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
258	1	1	female	35.0	0	0	512.3292	C	First	woman	False	B	Cherbourg	yes	True
679	1	1	male	36.0	0	1	512.3292	C	First	man	True	B	Cherbourg	yes	False
737	1	1	male	35.0	0	0	512.3292	C	First	man	True	B	Cherbourg	yes	True

```
# Defining function for Outliers Treatment
def Outlier_Treatment(col):
  Q1 = df[col].quantile(0.25)
  Q3 = df[col].quantile(0.75)
  IQR = Q3 - Q1
  upper = Q3 + (1.5 * IQR)
  lower = Q1 - (1.5 * IQR)
  np.clip(df[col], lower, upper, inplace = True)
```

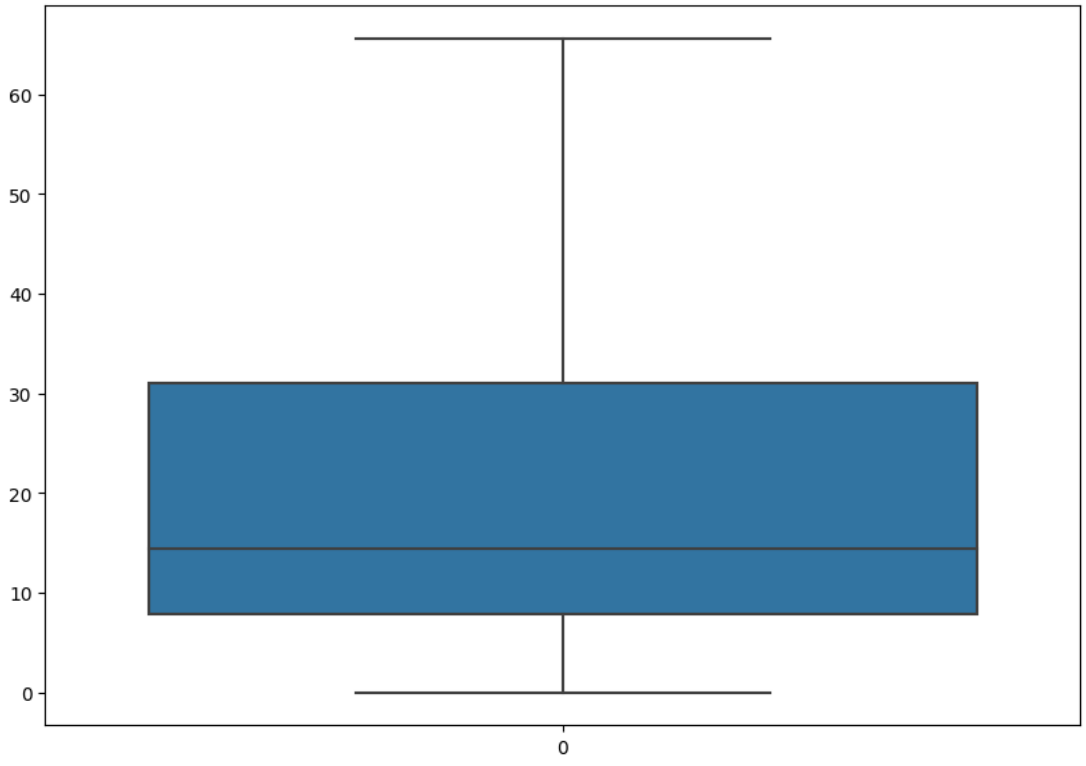
```
Outlier_Treatment('age')
plt.figure(figsize = (10,7))
sns.boxplot(df['age'])
```

```
<Axes: >
```



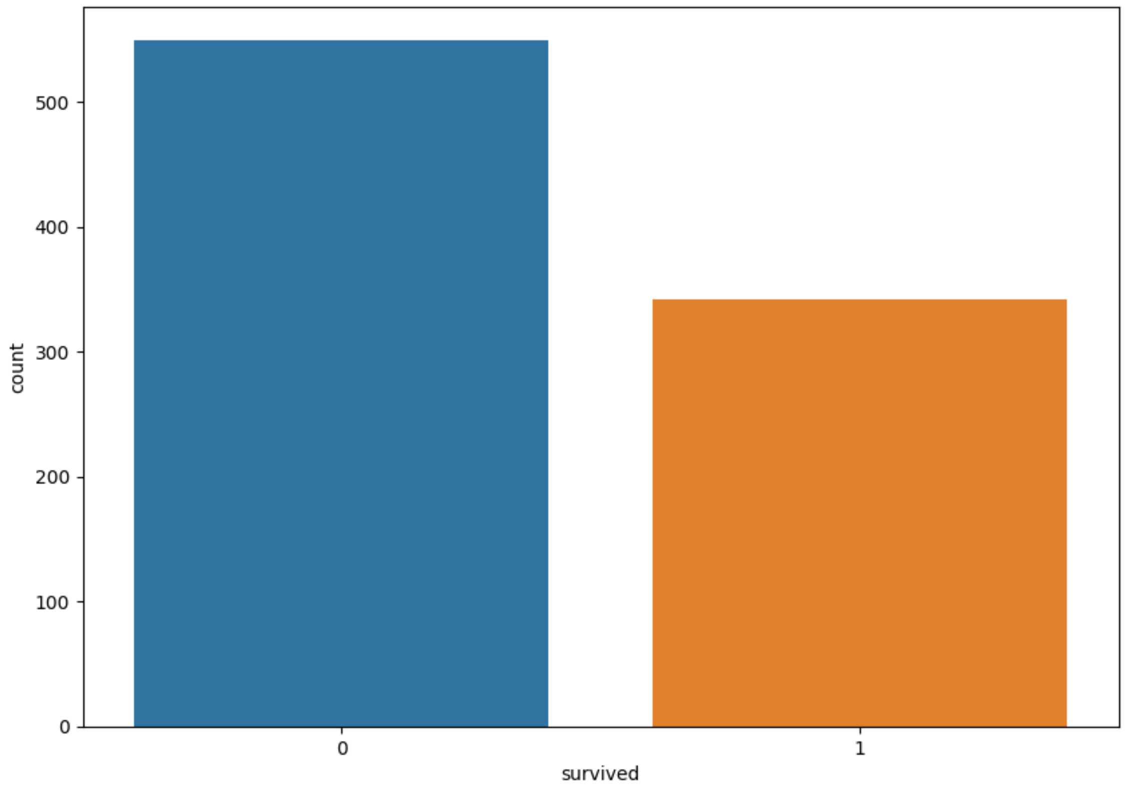
```
Outlier_Treatment('fare')
plt.figure(figsize = (10,7))
sns.boxplot(df['fare'])
```

```
<Axes: >
```



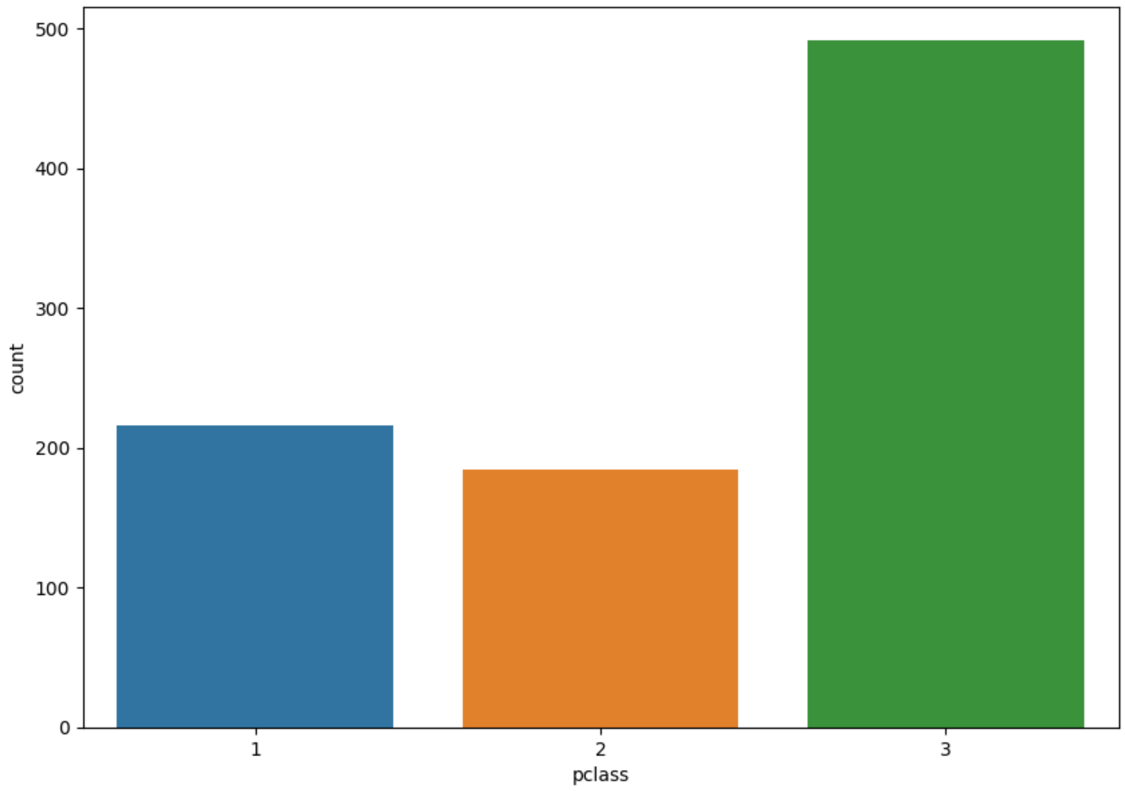
```
plt.figure(figsize = (10,7))
sns.countplot(data = df, x ='survived')
```

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



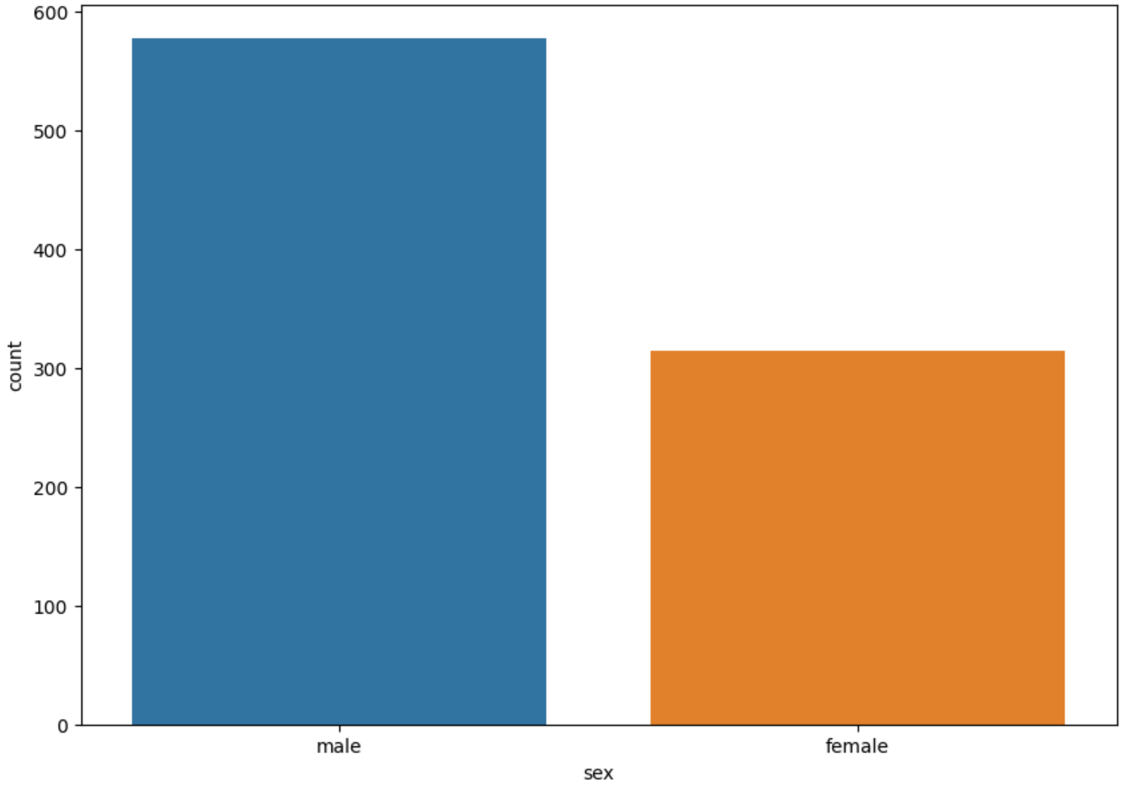
```
plt.figure(figsize = (10,7))
sns.countplot(data = df, x ='pclass')
```

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



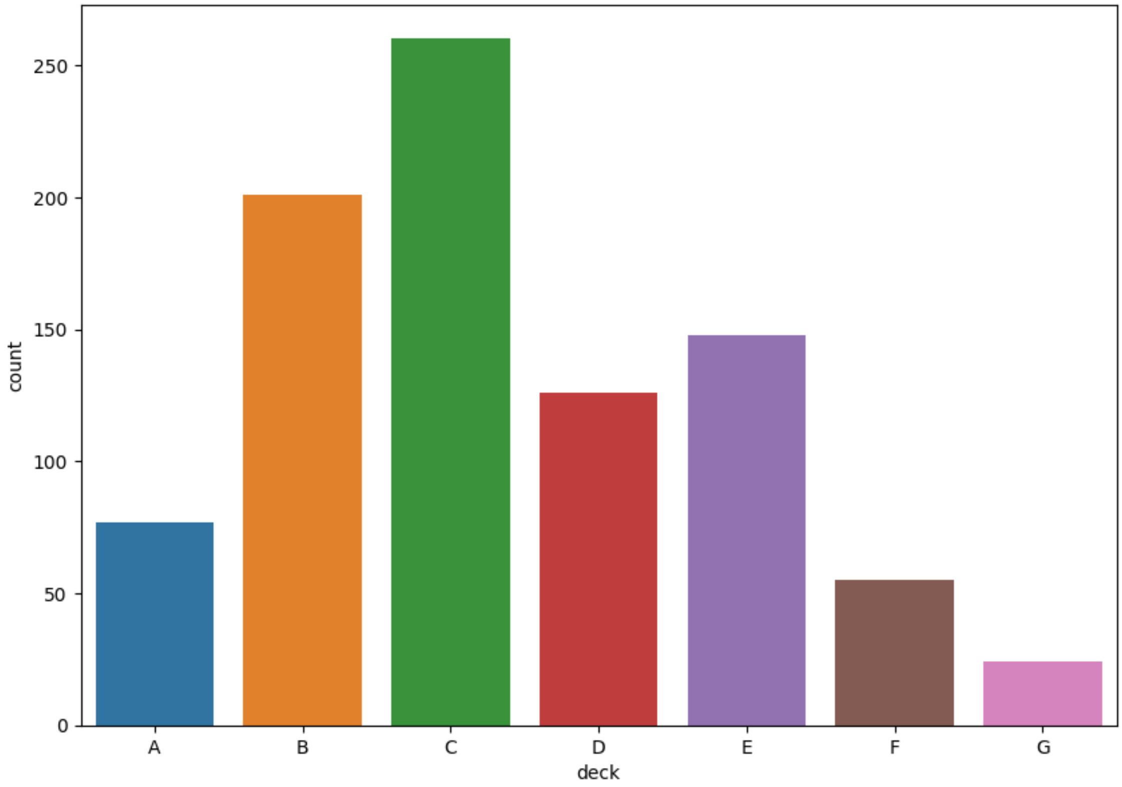
```
plt.figure(figsize = (10,7))
sns.countplot(data = df, x ='sex')
```

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



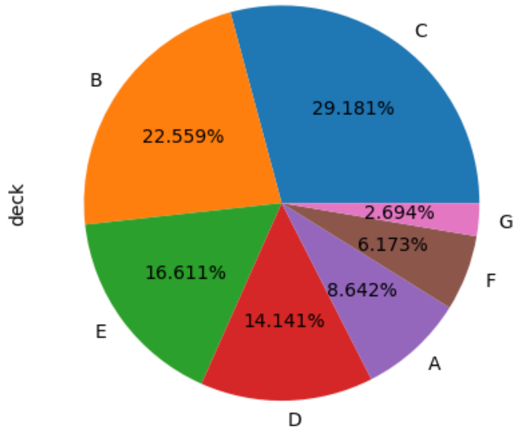
```
plt.figure(figsize = (10,7))
sns.countplot(data = df, x ='deck')
```

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



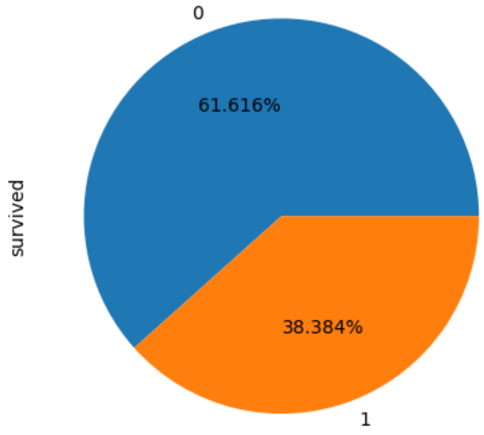
```
df['deck'].value_counts().plot(kind='pie',autopct='%.3f%%')
```

```
<Axes: ylabel='deck'>
```



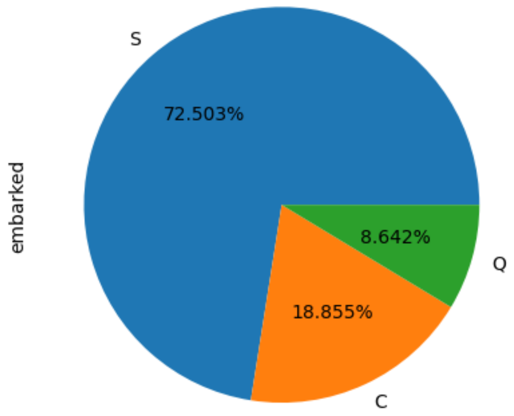
```
df['survived'].value_counts().plot(kind='pie',autopct='%.3f%%')
```

```
<Axes: ylabel='survived'>
```

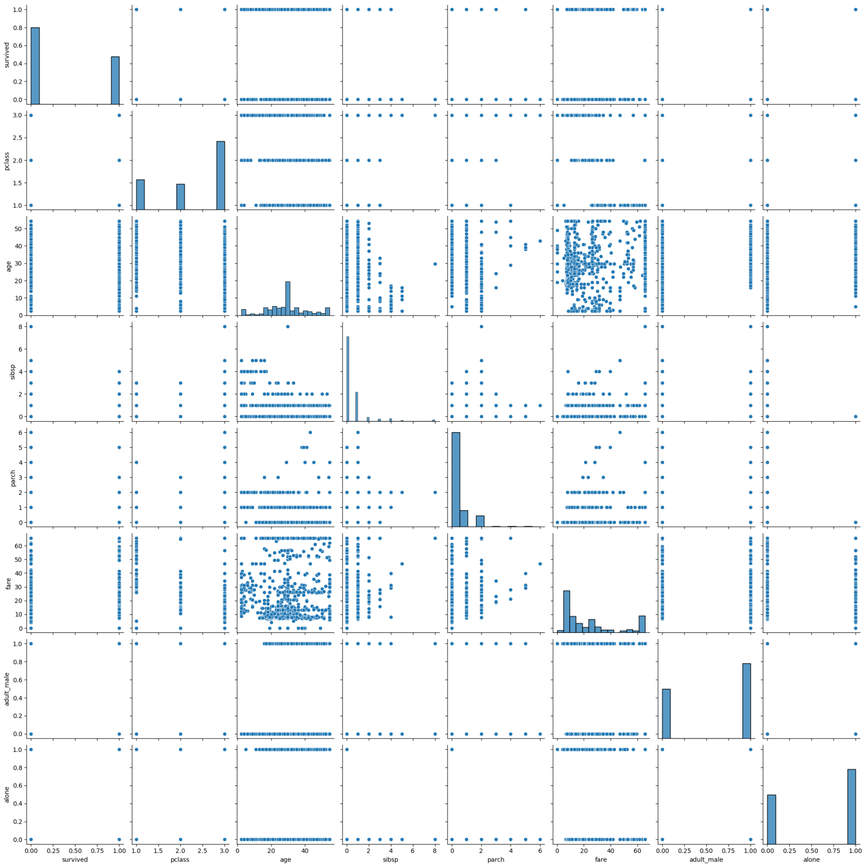


```
df['embarked'].value_counts().plot(kind='pie',autopct='%.3f%%')
```

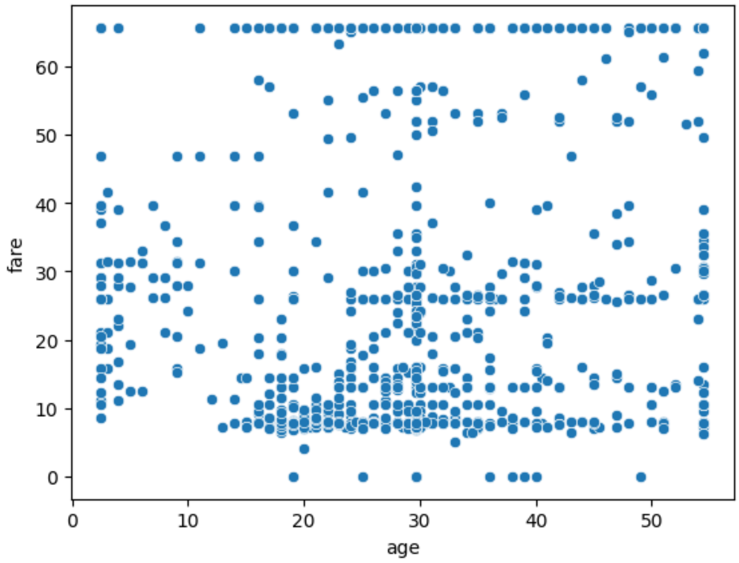
```
<Axes: ylabel='embarked'>
```



```
sns.pairplot(data=df)
plt.show()
```

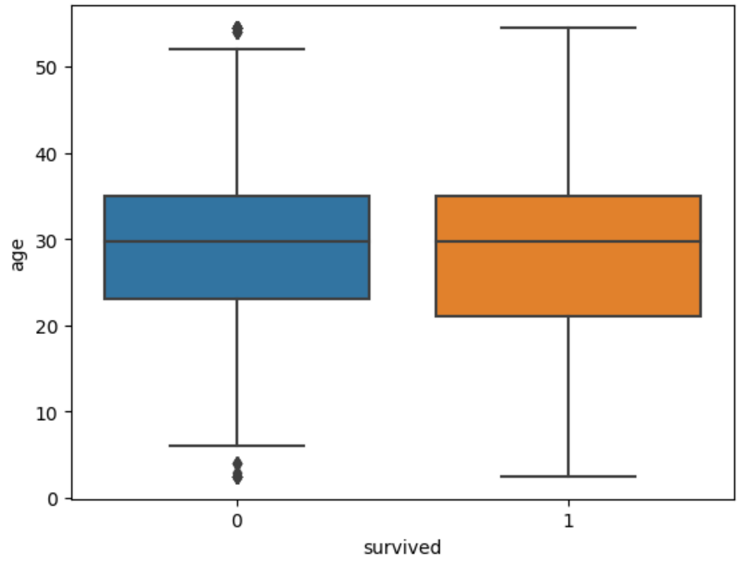

survived
pclass
age
sibsp
parch
fare
adult_male
alone
0.00
0.25
0.50
0.75
1.00
1.0
1.5
2.0
2.5
3.0
0
20
40
0
2
4
6
8
0
2
4
6
0
20
40
60

```
sns.scatterplot(x='age', y='fare', data=df)
plt.show()
```



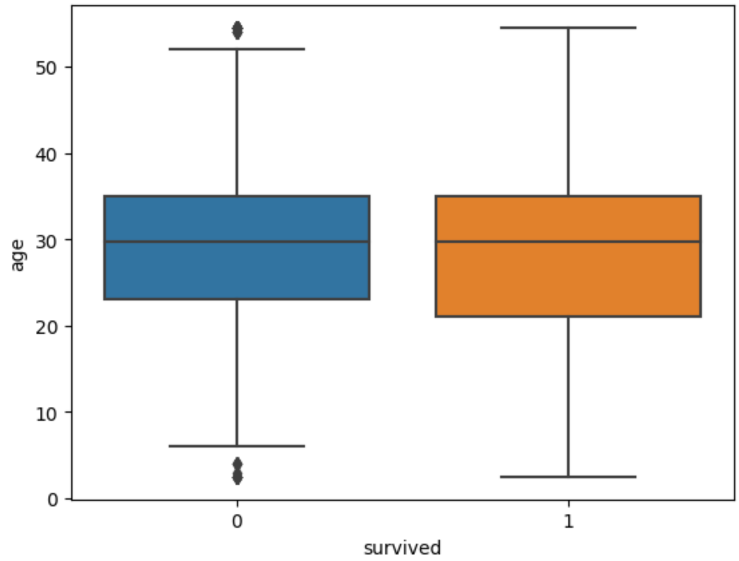
```
sns.boxplot(x='survived', y='age',data=df)
```

```
<Axes: xlabel='survived', ylabel='age'>
```



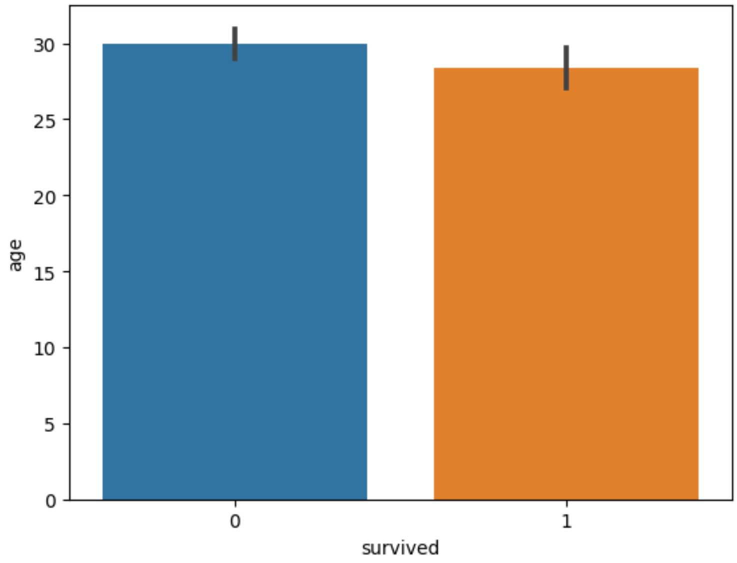
```
sns.boxplot(x='survived', y='age',data=df)
```

```
<Axes: xlabel='survived', ylabel='age'>
```



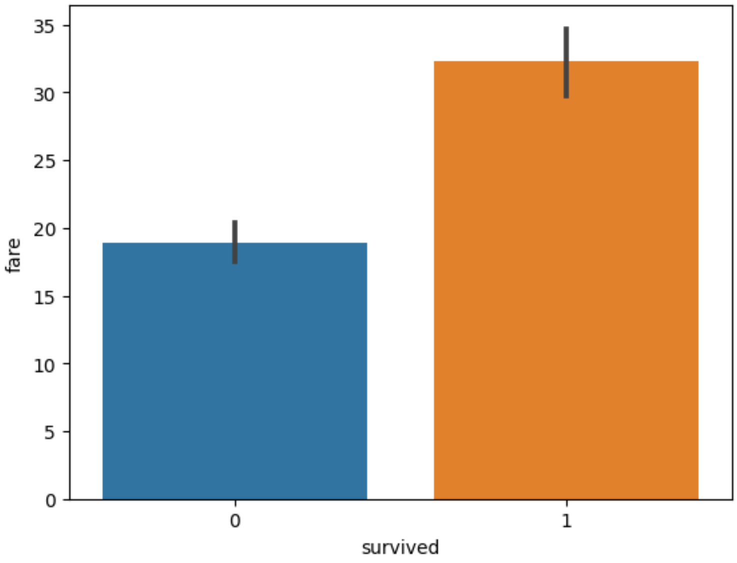
```
sns.barplot(x='survived', y='age',data=df)
```

```
<Axes: xlabel='survived', ylabel='age'>
```



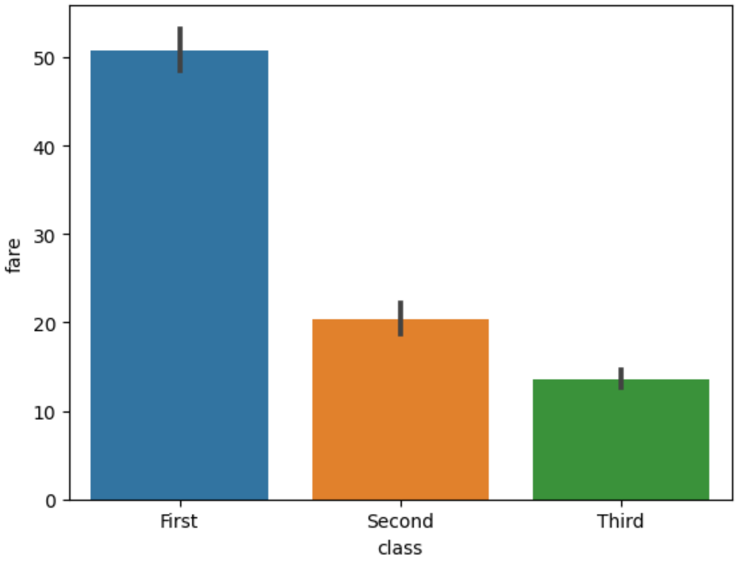
```
sns.barplot(x='survived', y='fare',data=df)
```

```
<Axes: xlabel='survived', ylabel='fare'>
```



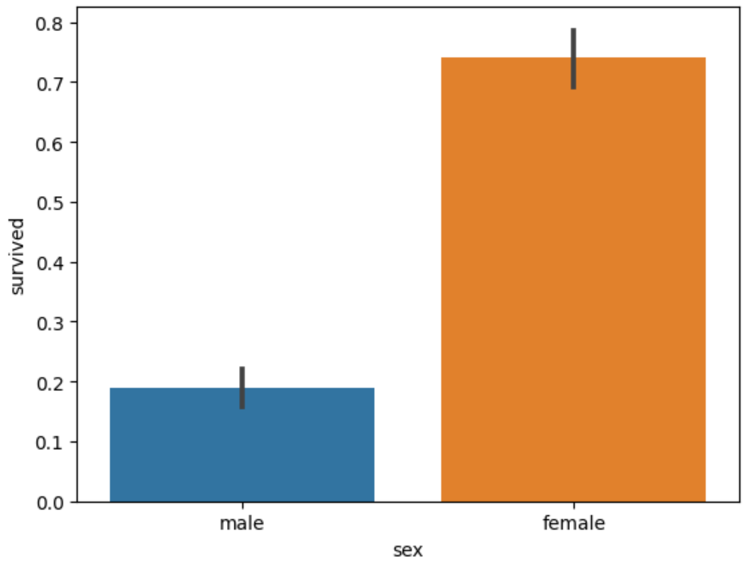
```
sns.barplot(x='class', y='fare',data=df)
```

```
<Axes: xlabel='class', ylabel='fare'>
```



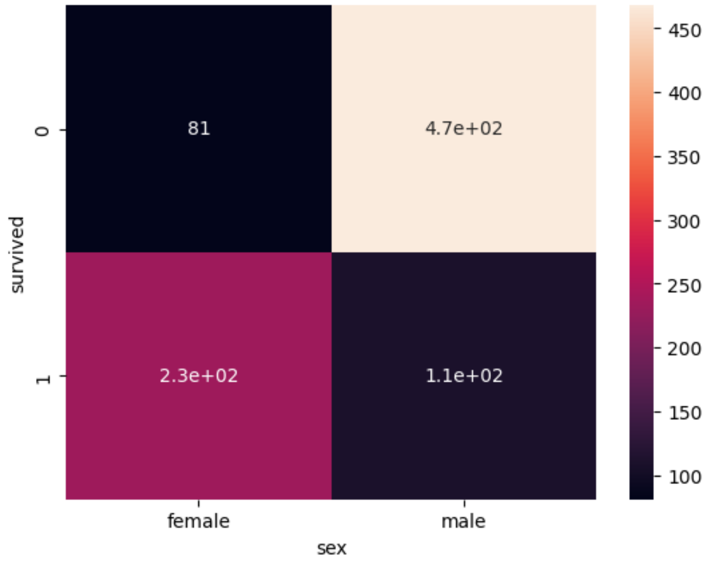
```
sns.barplot(x='sex', y='survived',data=df)
```

```
<Axes: xlabel='sex', ylabel='survived'>
```



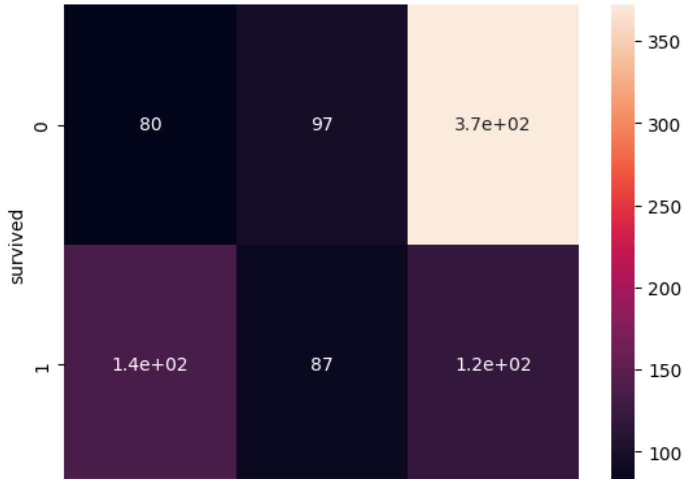
```
sns.heatmap(pd.crosstab(df['survived'],df['sex']), annot=True)
```

```
<Axes: xlabel='sex', ylabel='survived'>
```



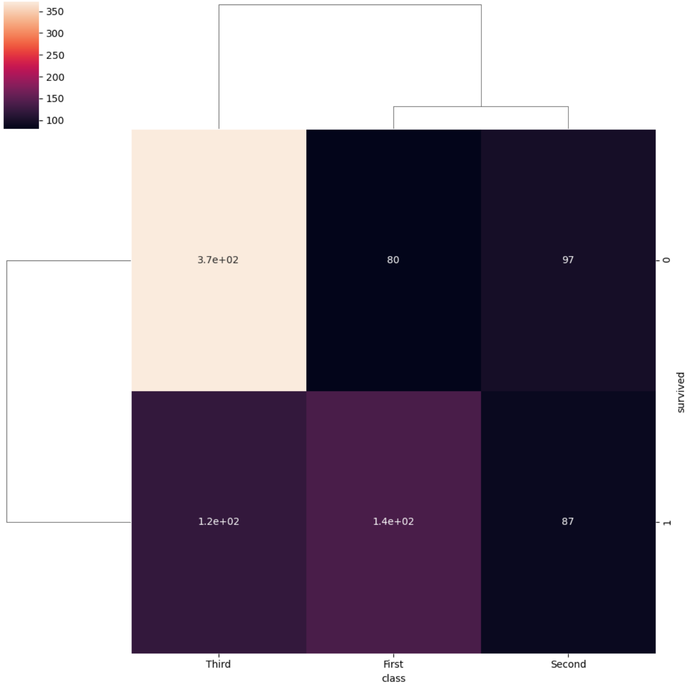
```
sns.heatmap(pd.crosstab(df['survived'],df['class']), annot=True)
```

```
<Axes: xlabel='class', ylabel='survived'>
```



```
sns.clustermap(pd.crosstab(df['survived'],df['class']), annot=True)
```

```
<seaborn.matrix.ClusterGrid at 0x7ffa2de72fb0>
```


3.7e+02
1.2e+02
1.4e+02
survived
Third
First
Second
class