Importing Libraries

In [1]:

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

Importing Dataset

In [2]:

```
df = pd.read_csv("../datasets/Titanic-Dataset.csv")
df.head()
```

Out[2]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cal
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	N
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	N
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	N

In [3]:

```
df.shape
```

Out[3]:

```
(891, 12)
```

In [4]:

```
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
```

Handlingf null values

In [5]:

```
df.isnull().any()
```

Out[5]:

```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age             True
SibSp          False
Parch          False
Ticket         False
Fare           False
Cabin           True
Embarked        True
dtype: bool
```

In [6]:

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

Out[6]:

```
PassengerId      0
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 [7]:

```
df.describe()
```

Out[7]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [8]:

```
df['Age'].median()
```

Out[8]:

```
28.0
```

In [9]:

```
df['Age'] = df['Age'].fillna(df['Age'].median())
```

In [10]:

```
df['Age'].isnull().any()
```

Out[10]:

```
False
```

In [11]:

```
df['Cabin'].mode()
```

Out[11]:

```
0        B96 B98
1    C23 C25 C27
2             G6
Name: Cabin, dtype: object
```

In [12]:

```
df['Cabin'].mode()[0][0:3]
```

Out[12]:

```
'B96'
```

In [13]:

```
df['Cabin'] = df['Cabin'].fillna(method = 'bfill')
```

In [14]:

```
df['Embarked'].mode()
```

Out[14]:

```
0    S
Name: Embarked, dtype: object
```

In [15]:

```
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])
```

In [16]:

```
df.head()
```

Out[16]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cal
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	C
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	C1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	E

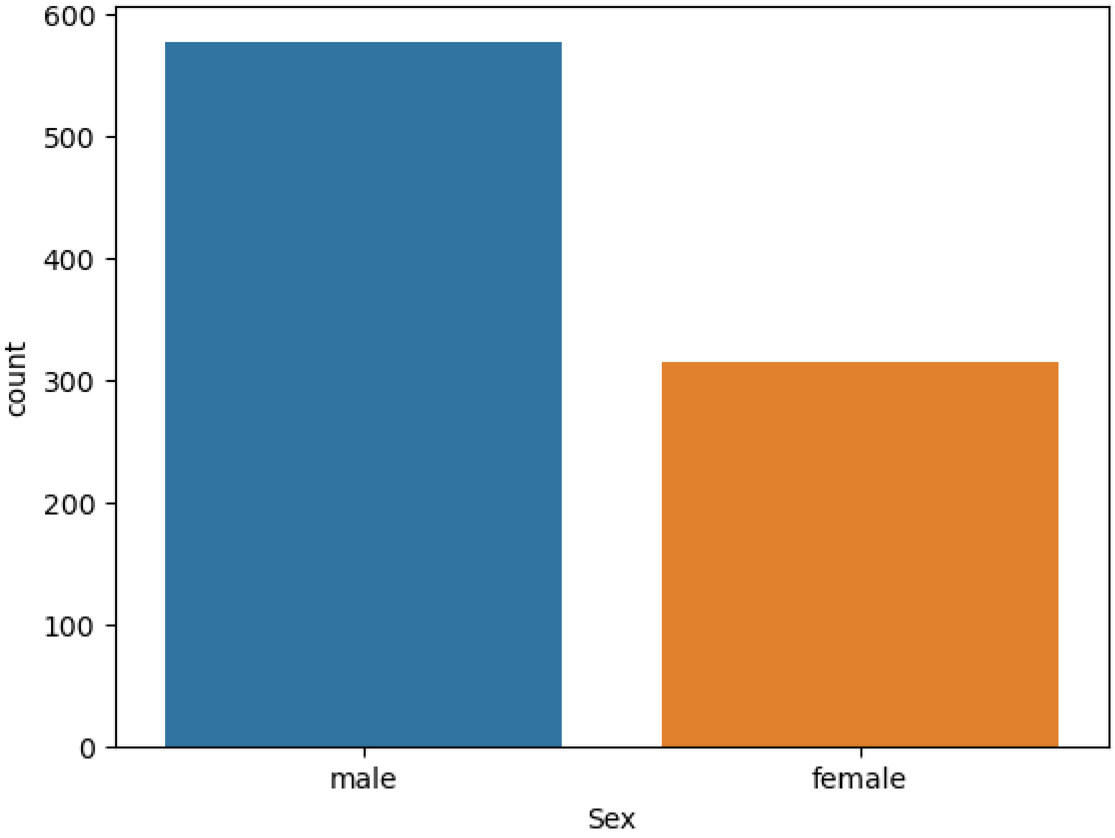
Data Visualisation

In [17]:

```
sns.countplot(x = 'Sex', data = df)
```

Out[17]:

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



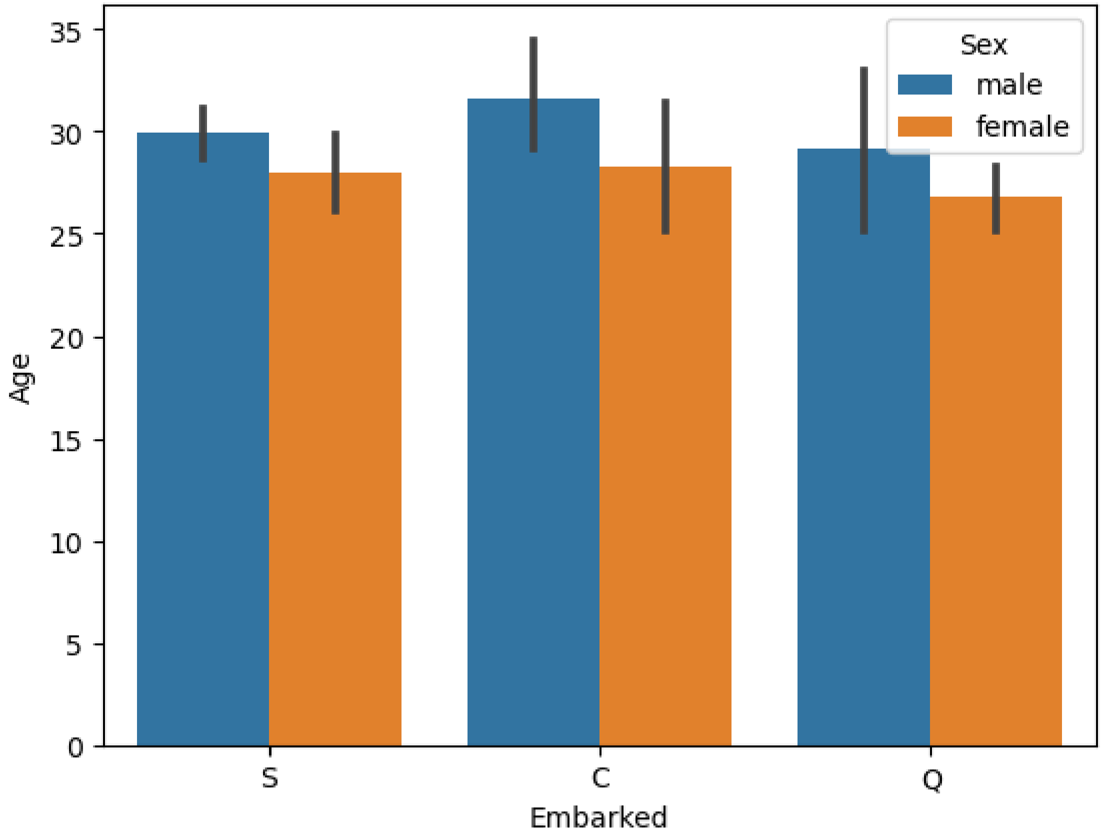
Inference: From this graph we can see the difference between the count of different Gender

In [18]:

```
sns.barplot(data=df,x="Embarked",y="Age",hue="Sex")
```

Out[18]:

```
<Axes: xlabel='Embarked', ylabel='Age'>
```



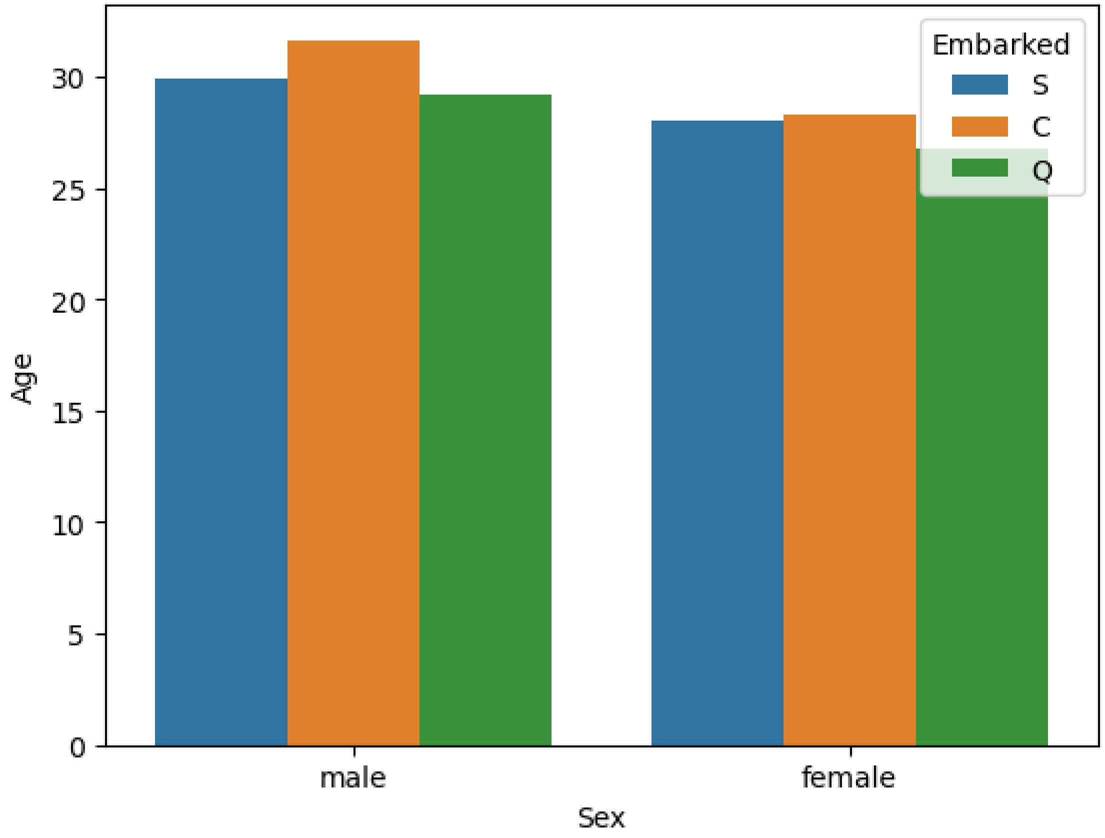
inference: From the above graph we can see the variation in the age of different embarkment zone and the gender

In [19]:

```
sns.barplot(data=df,x="Sex",y="Age",hue="Embarked", errorbar = None)
```

Out[19]:

```
<Axes: xlabel='Sex', ylabel='Age'>
```



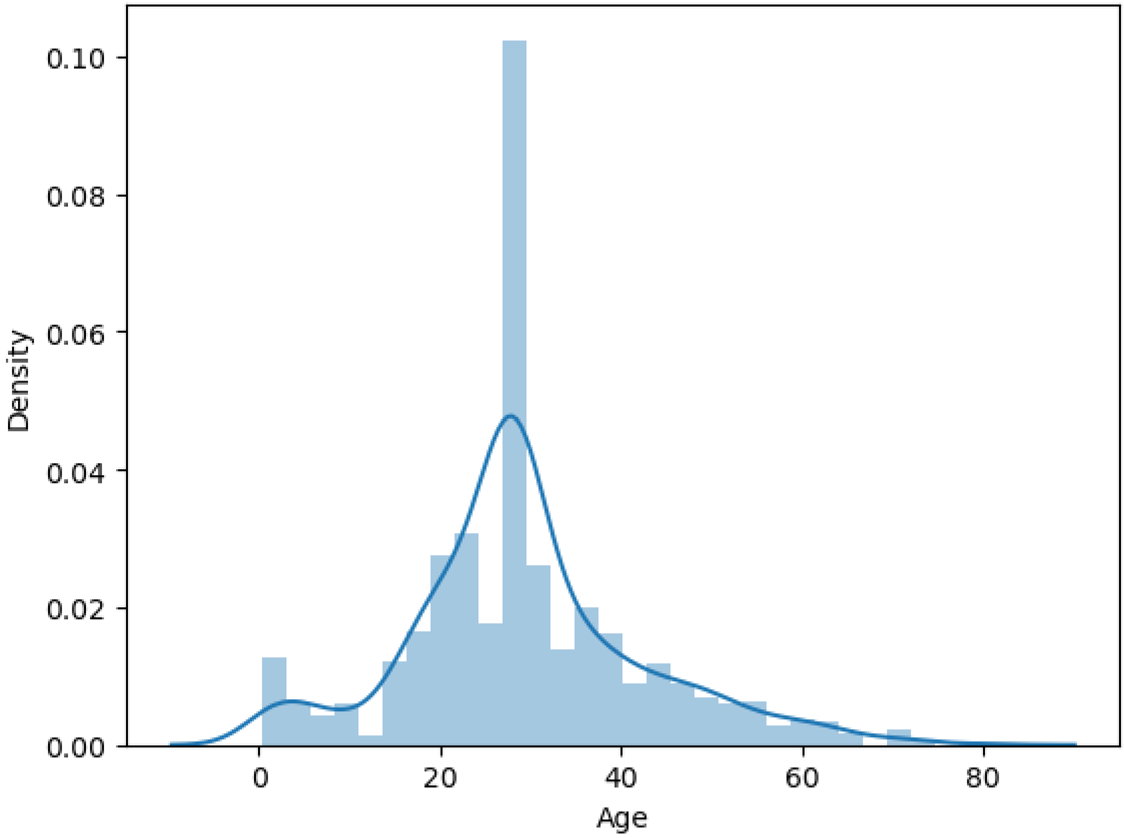
In [20]:

```
sns.distplot(df['Age'])
```

```
/var/folders/m8/dg41v9m11bdcfq4q15h80_140000gn/T/ipykernel_95392/32558
28239.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.
14.0.

Please adapt your code to use either `displot` (a figure-level functio
n with
similar flexibility) or `histplot` (an axes-level function for histogr
ams).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (http
s://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  sns.distplot(df['Age'])
```

Out[20]:

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



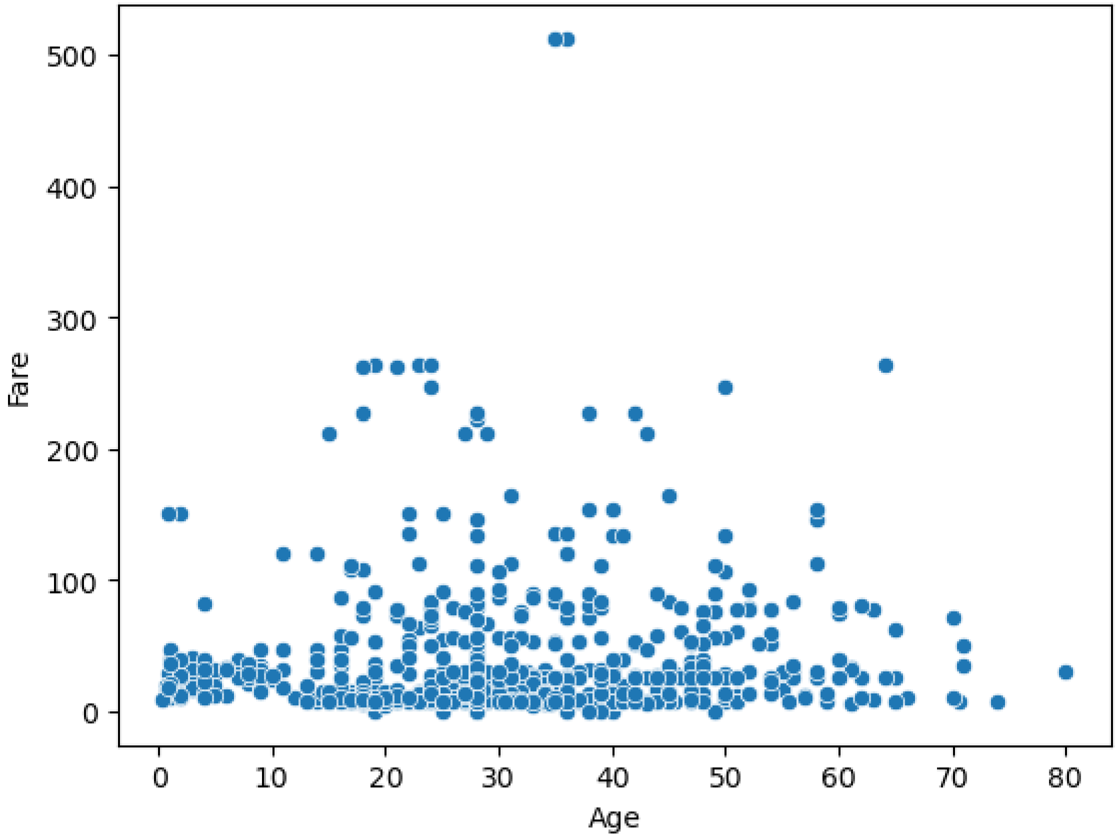
inference: From the above gaph we can the Normal Distribution of age

In [21]:

```
sns.scatterplot(x = 'Age', y = 'Fare', data = df)
```

Out[21]:

```
<Axes: xlabel='Age', ylabel='Fare'>
```



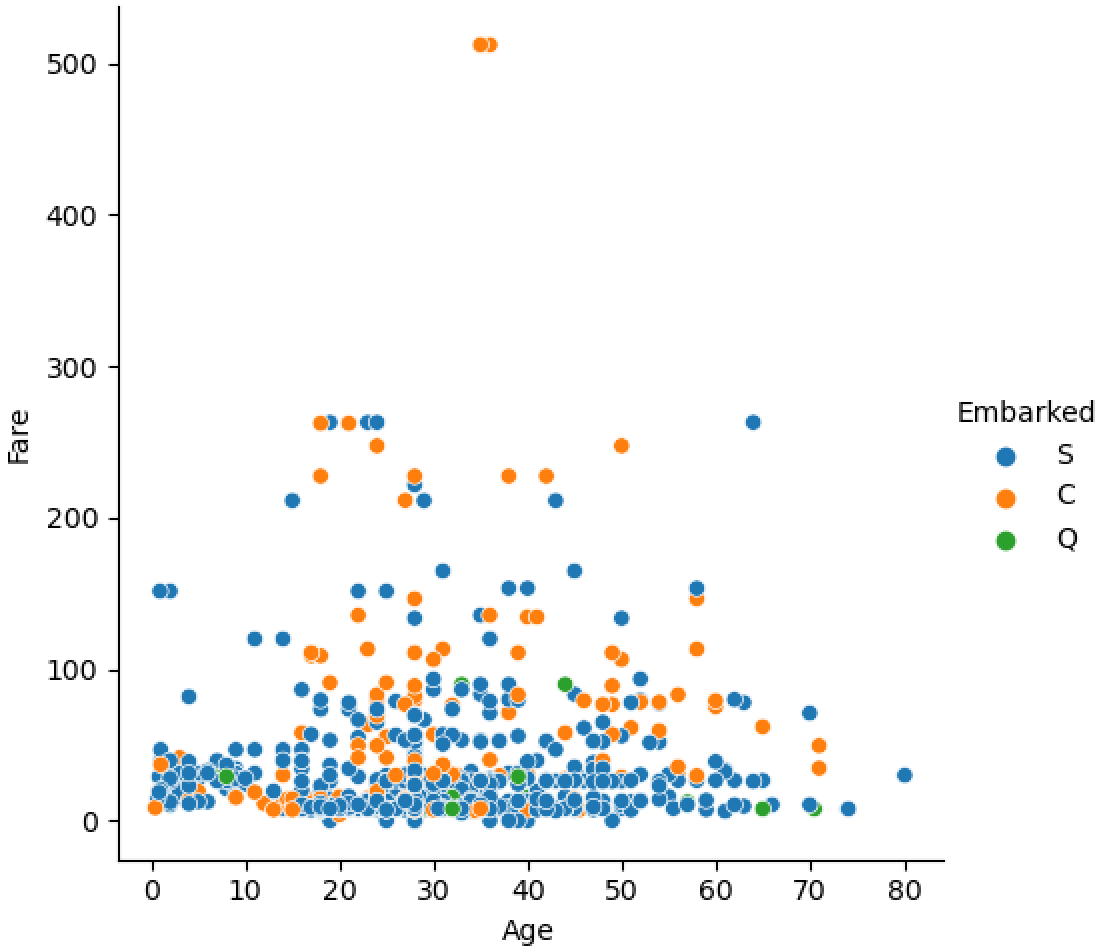
Inference: From the above graph we can see the fair that was charged based on the age of passenger

```
In [22]:
```

```
sns.relplot(x = 'Age', y= 'Fare', data = df, hue = 'Embarked')
```

```
Out[22]:
```

```
<seaborn.axisgrid.FacetGrid at 0x142014890>
```



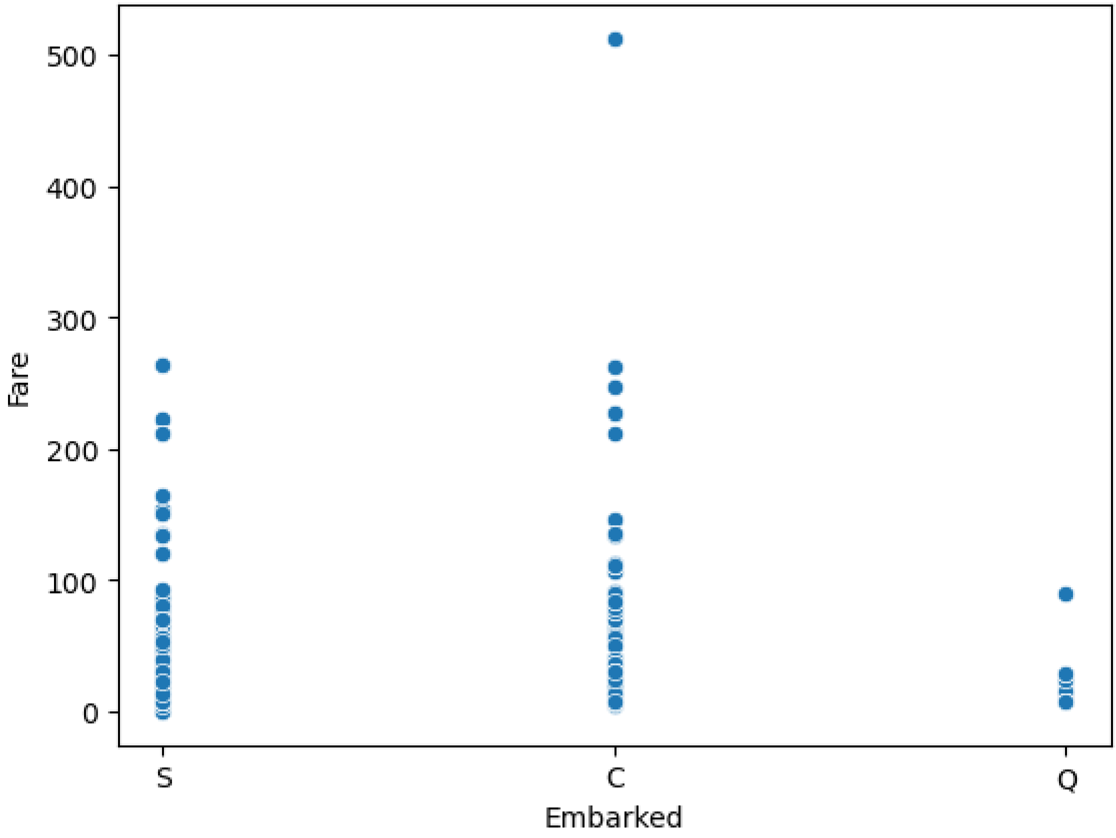
inference: the above graph shows the fare chared for the passengers in different embarment zones and compares with their age

In [23]:

```
sns.scatterplot(x = 'Embarked',y = 'Fare', data = df)
```

Out[23]:

```
<Axes: xlabel='Embarked', ylabel='Fare'>
```



Inference: the above graph shows the fare chared for the passengers in different embarment zones

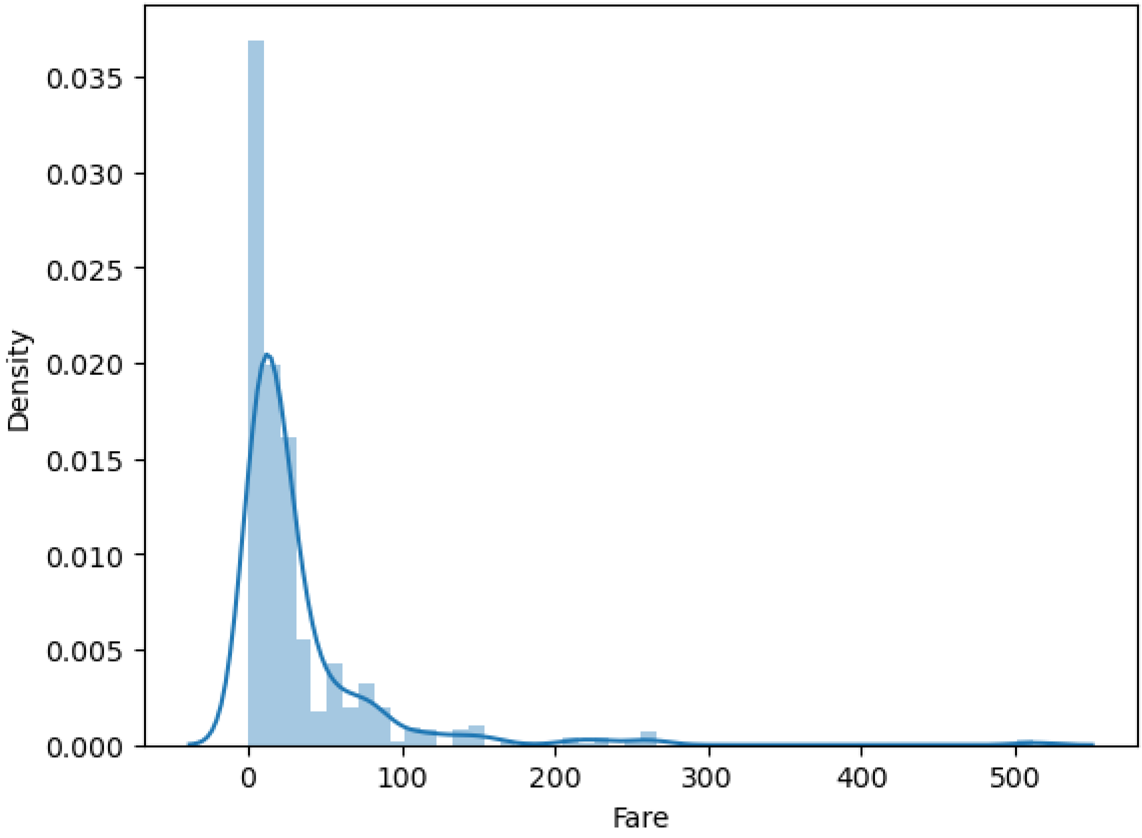
In [24]:

```
sns.distplot(df.Fare)
```

```
/var/folders/m8/dg41v9m11bdcfq4q15h80_140000gn/T/ipykernel_95392/34021
12601.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.
14.0.

Please adapt your code to use either `displot` (a figure-level functio
n with
similar flexibility) or `histplot` (an axes-level function for histogr
ams).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (http
s://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  sns.distplot(df.Fare)
```

Out[24]:

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



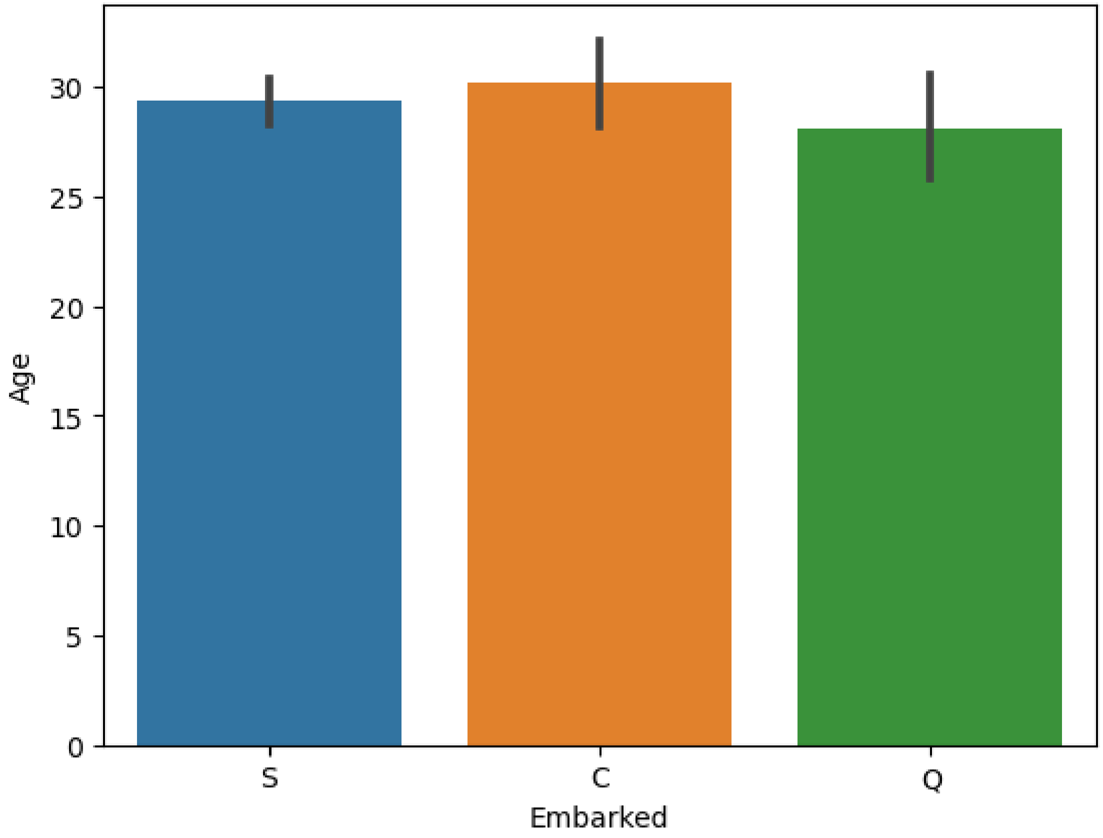
inference: from the above graph we can see the distribution of the fare

In [25]:

```
sns.barplot(x = 'Embarked', y ='Age', data = df)
```

Out[25]:

```
<Axes: xlabel='Embarked', ylabel='Age'>
```

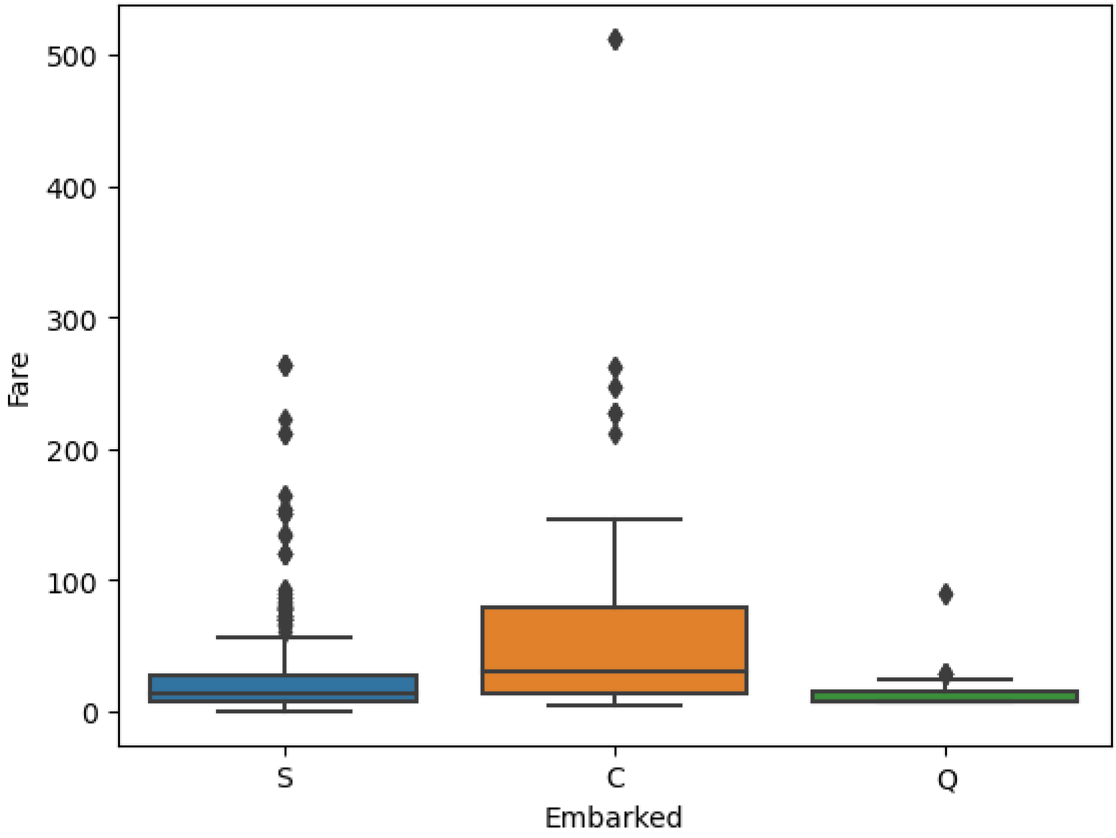


In [26]:

```
sns.boxplot(x = 'Embarked', y = 'Fare', data = df)
```

Out[26]:

```
<Axes: xlabel='Embarked', ylabel='Fare'>
```

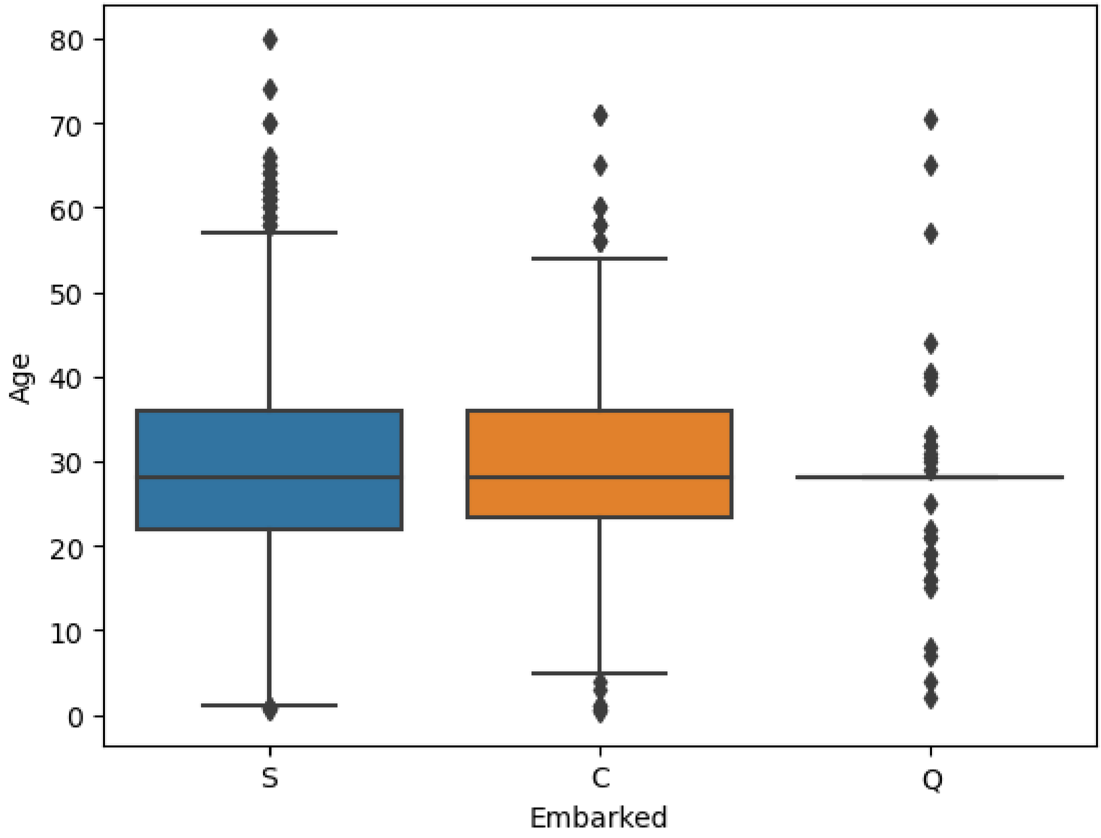


In [27]:

```
sns.boxplot(x = 'Embarked', y = 'Age', data = df)
```

Out[27]:

```
<Axes: xlabel='Embarked', ylabel='Age'>
```



In [28]:

```
corr = df.corr()
corr
```

```
/var/folders/m8/dg41v9m11bdcfq4q15h80_140000gn/T/ipykernel_95392/24380
84875.py:1: FutureWarning: The default value of numeric_only in DataFr
ame.corr is deprecated. In a future version, it will default to False.
Select only valid columns or specify the value of numeric_only to sile
nce this warning.
  corr = df.corr()
```

Out[28]:

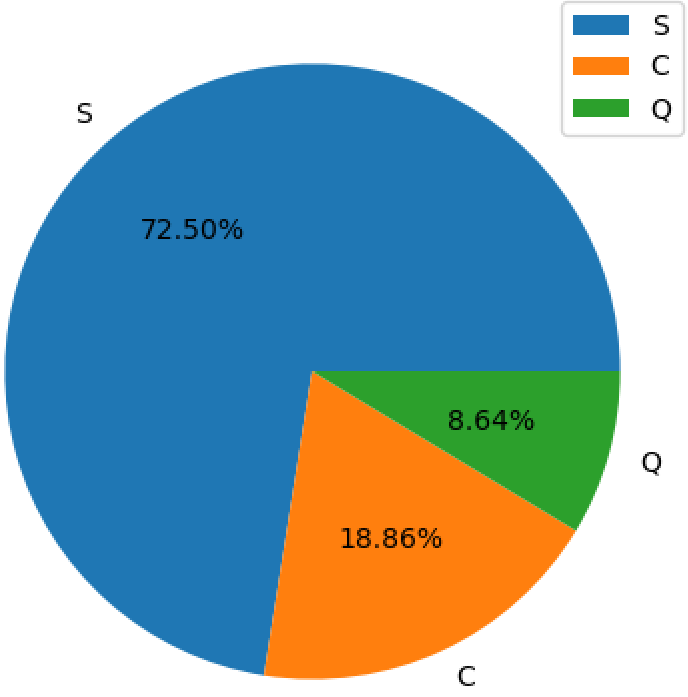
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.034212	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.064910	-0.035322	0.081629	0.257307
Pclass	-0.035144	-0.338481	1.000000	-0.339898	0.083081	0.018443	-0.549500
Age	0.034212	-0.064910	-0.339898	1.000000	-0.233296	-0.172482	0.096688
SibSp	-0.057527	-0.035322	0.083081	-0.233296	1.000000	0.414838	0.159651
Parch	-0.001652	0.081629	0.018443	-0.172482	0.414838	1.000000	0.216225
Fare	0.012658	0.257307	-0.549500	0.096688	0.159651	0.216225	1.000000

In [29]:

```
fig=plt.figure()
axes1=fig.add_axes([0.1,0.1,0.8,0.8])  #[left,bottom,width,height]
axes1.pie(df['Embarked'].value_counts(),labels = ['S', 'C','Q'],autopct="%0.2f%%")
axes1.legend()
```

Out[29]:

```
<matplotlib.legend.Legend at 0x1423decd0>
```



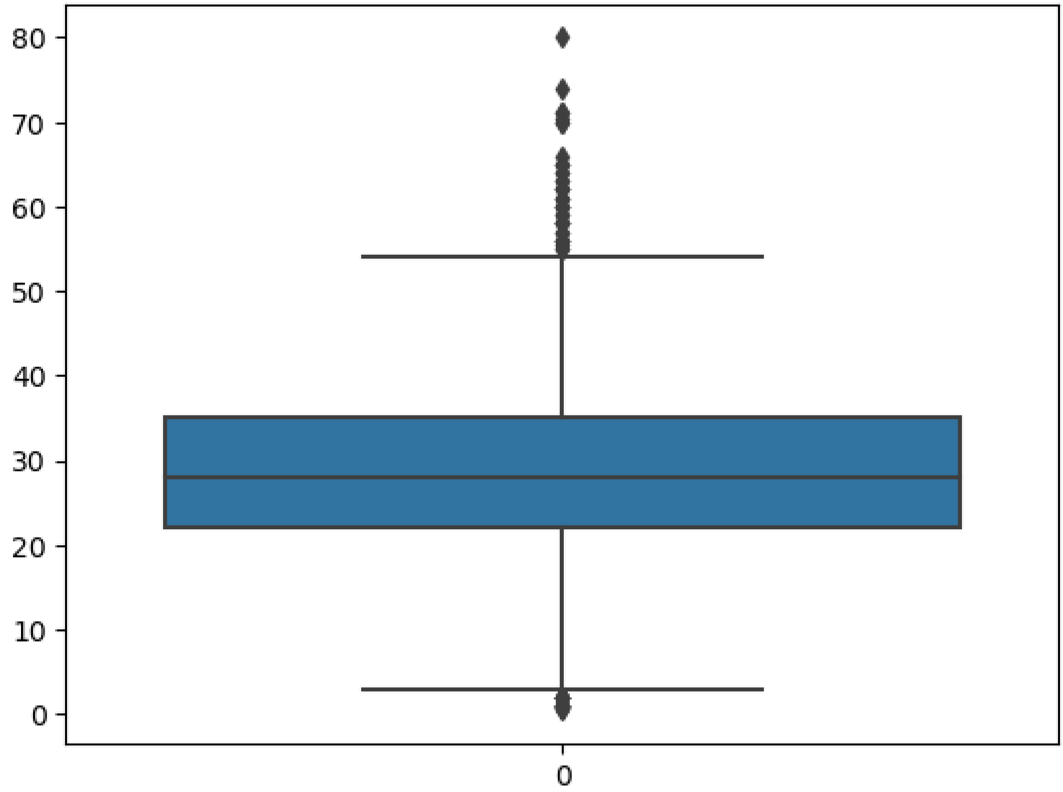
Outlier detection

In [30]:

```
sns.boxplot(df.Age)
```

Out[30]:

```
<Axes: >
```



In [31]:

```
q1 = df.Age.quantile(0.25)
q3 = df.Age.quantile(0.75)
```

In [32]:

```
IQR =q3-q1
```

In [33]:

```
upper_limit =q3+1.5*IQR
```

In [34]:

```
upper_limit
```

Out[34]:

```
54.5
```

In [35]:

```
lower_limit = q1-1.5*IQR
lower_limit
```

Out[35]:

```
2.5
```

In [36]:

```
df.median()
```

```
/var/folders/m8/dg41v9m11bdcfq4q15h80_l40000gn/T/ipykernel_95392/53005
1474.py:1: FutureWarning: The default value of numeric_only in DataFra
me.median is deprecated. In a future version, it will default to Fals
e. In addition, specifying 'numeric_only=None' is deprecated. Select o
nly valid columns or specify the value of numeric_only to silence this
warning.
  df.median()
```

Out[36]:

```
PassengerId    446.0000
Survived         0.0000
Pclass           3.0000
Age             28.0000
SibSp            0.0000
Parch            0.0000
Fare            14.4542
dtype: float64
```

In [37]:

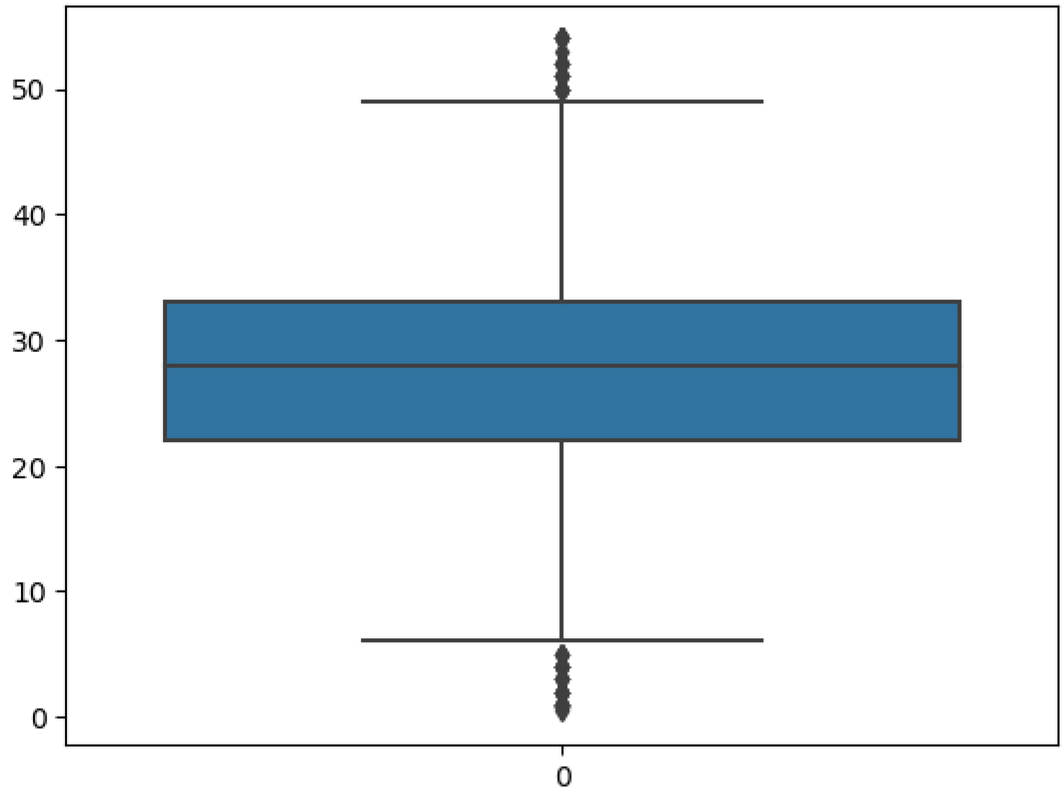
```
df = df[df.Age<upper_limit]
```

In [38]:

```
sns.boxplot(df["Age"])
```

Out[38]:

```
<Axes: >
```

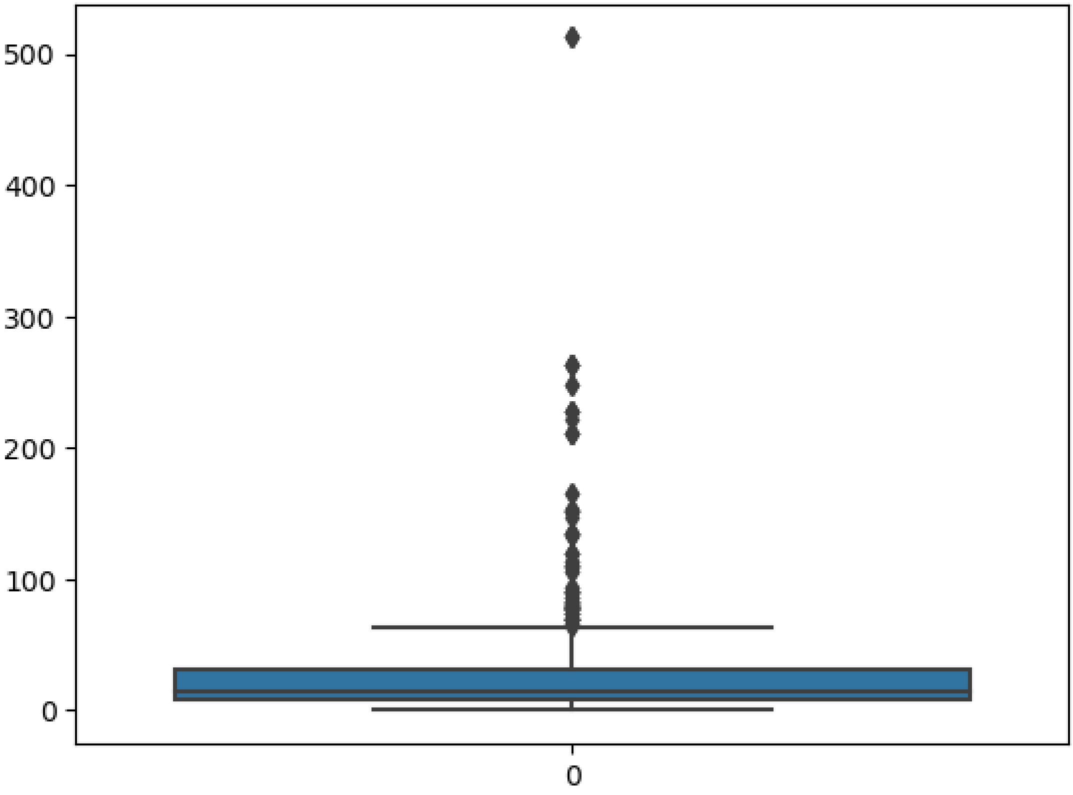


In [39]:

```
sns.boxplot(df.Fare)
```

Out[39]:

```
<Axes: >
```



In [40]:

```
q1 = df.Fare.quantile(0.25)
q3 = df.Fare.quantile(0.75)
```

In [41]:

```
IQR =q3-q1
```

In [42]:

```
upper_limit =q3+1.5*IQR
```

In [43]:

```
upper_limit
```

Out[43]:

```
64.4063
```

In [44]:

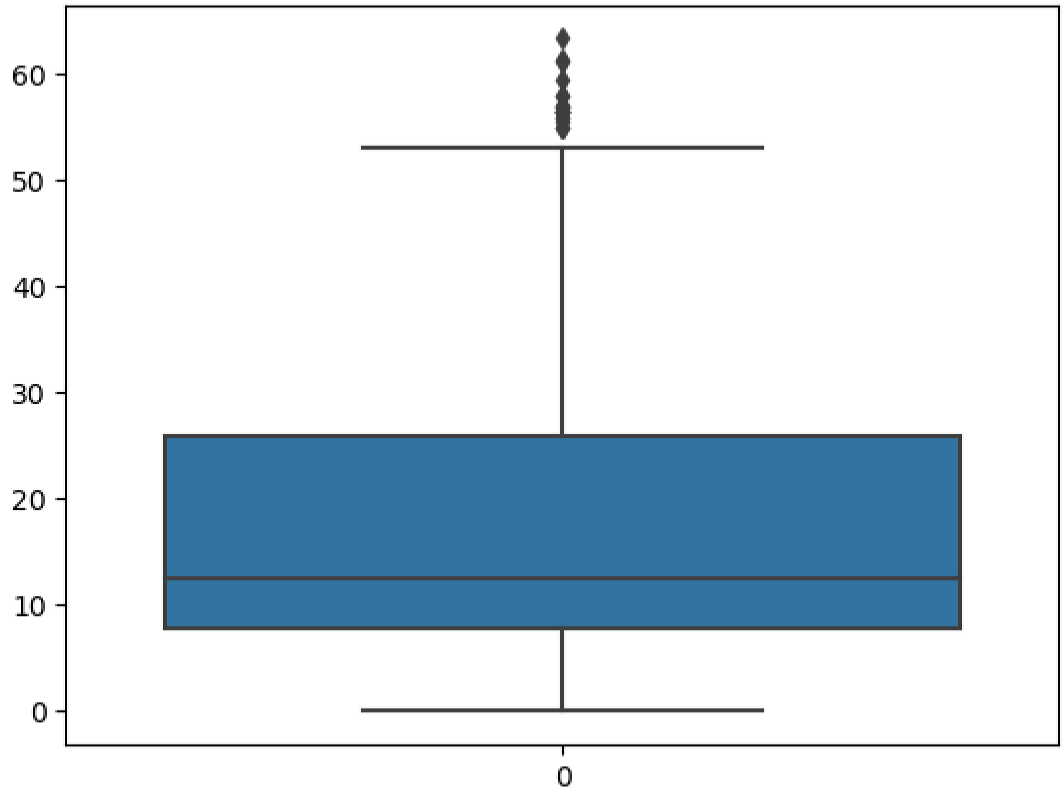
```
df = df[df.Fare<upper_limit]
```

In [45]:

```
sns.boxplot(df.Fare)
```

Out[45]:

```
<Axes: >
```



Splitting Dependent and Independent variables

In [46]:

```
df.head()
```

Out[46]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cal
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	C1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	E
5	6	0	3	Moran, Mr. James	male	28.0	0	0	330877	8.4583	E

In [ ]:

```

```

In [47]:

```
x = df.loc[:,['Survived', 'Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare']]
y = df.Embarked
```

In [48]:

```
x.head()
```

Out[48]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare
0	0	3	male	22.0	1	0	7.2500
2	1	3	female	26.0	0	0	7.9250
3	1	1	female	35.0	1	0	53.1000
4	0	3	male	35.0	0	0	8.0500
5	0	3	male	28.0	0	0	8.4583

In [49]:

```
y.head()
```

Out[49]:

```
0    S
2    S
3    S
4    S
5    Q
Name: Embarked, dtype: object
```

Encoding

In [50]:

```
from sklearn.preprocessing import LabelEncoder
```

In [51]:

```
le=LabelEncoder()
```

In [52]:

```
x["Sex"]=le.fit_transform(x["Sex"])
```

In [53]:

```
x['Sex']
```

Out[53]:

```
0      1
2      0
3      0
4      1
5      1
      ..
886    1
887    0
888    0
889    1
890    1
Name: Sex, Length: 741, dtype: int64
```

In [54]:

```
x["Sex"].value_counts()
```

Out[54]:

```
1    503
0    238
Name: Sex, dtype: int64
```

In [55]:

```
x['Sex'].nunique()
```

Out[55]:

```
2
```

In [56]:

```
y = le.fit_transform(y)
```

Feature Scalling

In [57]:

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

In [58]:

```
x = sc.fit_transform(x)
```

In [59]:

```
x
```

Out[59]:

```
array([[-0.72435582,  0.6809705 ,  0.68786702, ...,  0.59768849,
        -0.43803498, -0.7675324 ],
       [ 1.38053698,  0.6809705 , -1.45376937, ..., -0.49586008,
        -0.43803498, -0.71687899],
       [ 1.38053698, -2.13016677, -1.45376937, ...,  0.59768849,
        -0.43803498,  2.67314775],
       ...,
       [-0.72435582,  0.6809705 , -1.45376937, ...,  0.59768849,
         2.07811944,  0.44814957],
       [ 1.38053698, -2.13016677,  0.68786702, ..., -0.49586008,
        -0.43803498,  0.93967531],
       [-0.72435582,  0.6809705 ,  0.68786702, ..., -0.49586008,
        -0.43803498, -0.73001135]])
```

Train and Test Split

In [60]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=0)
```

In [61]:

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

Out[61]:

```
((518, 7), (223, 7), (518,), (223,))
```