SMART BRIDGE EXTERNSHIP IN

APPLIED DATA SCIENCE

ASSIGNMENT 2

NAME : JEYASRI VARTHINI B

REG.NO. : 20MID0049

BRANCH : Integrated M.Tech. Computer Science with specialization in Data Science

CAMPUS : VIT VELLORE

EMAIL : jeyasrivarthini.b2020@vitstudent.ac.in

DATE : 27-05-2023

jupyter JEYASRI_VARTHINI_B_ADS_ASSIGNMENT_2 Last Checkpoint: a few seconds ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Checkpoint created: 03:22:27 Trusted Kernel

Run Code

TITANIC SHIP CASE STUDY

In [1]:
```
# Importing required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

2. Loading the dataset

In [2]:
```
data=pd.read_csv('titanic.csv')
```

In [3]:
```
data.head()
```

Out[3]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alon
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	Fals
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	Fals
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	Tru
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	Fals
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	Tru

In [4]:
```
data.tail()
```

Out[4]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alo
886	0	2	male	27.0	0	0	13.00	S	Second	man	True	NaN	Southampton	no	T
887	1	1	female	19.0	0	0	30.00	S	First	woman	False	B	Southampton	yes	T
888	0	3	female	NaN	1	2	23.45	S	Third	woman	False	NaN	Southampton	no	Fa
889	1	1	male	26.0	0	0	30.00	C	First	man	True	C	Cherbourg	yes	T
890	0	3	male	32.0	0	0	7.75	Q	Third	man	True	NaN	Queenstown	no	T

In [5]:
```
data.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    object
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    object
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), float64(2), int64(4), object(7)
memory usage: 92.4+ KB
```

In [6]:
```
data.columns
```

Out[6]:
```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'deck', 'embark_town',
       'alive', 'alone'],
      dtype='object')
```

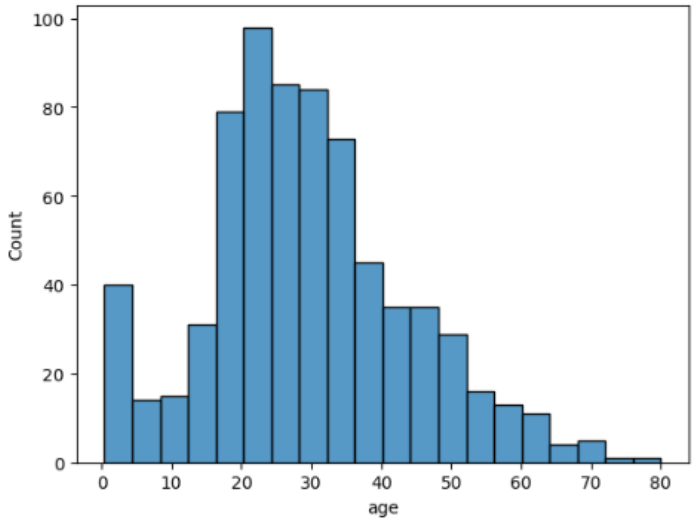
In [7]:
```
data.rename({'class':'class_name'},axis=1,inplace=True)
data.columns
```

Out[7]:
```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class_name', 'who', 'adult_male', 'deck', 'embark_town',
       'alive', 'alone'],
      dtype='object')
```

3. Visualization - Univariate Analysis

In [8]:
```
# Histogram
sns.histplot(data['age'])
```

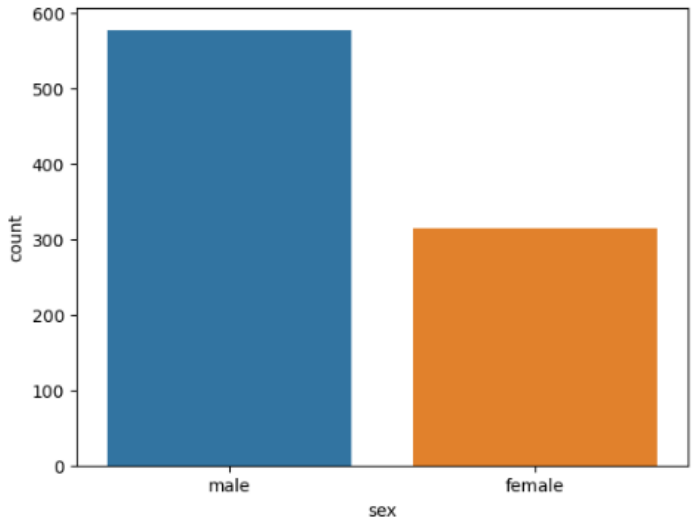
Out[8]:
```
<AxesSubplot:xlabel='age', ylabel='Count'>
```



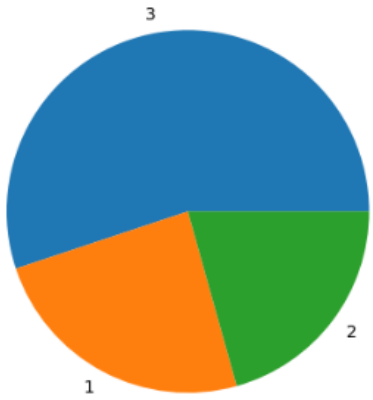
In [9]:
```
# Bar Chart
sns.countplot(data['sex'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

Out[9]:
```
<AxesSubplot:xlabel='sex', ylabel='count'>
```



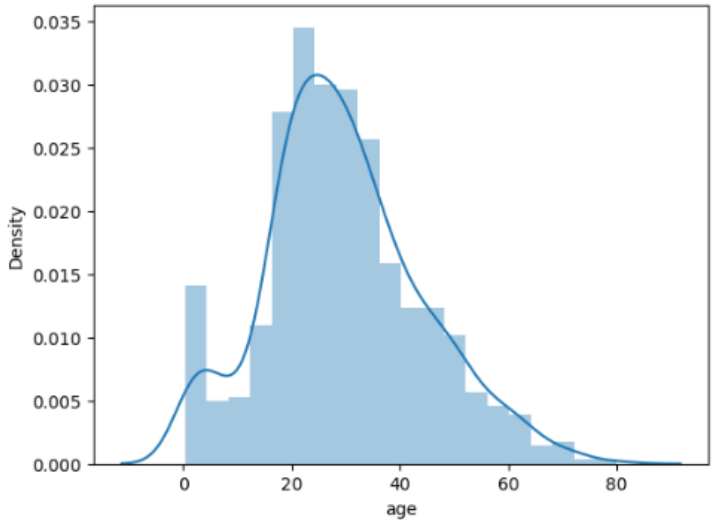
In [10]:
```
# Pie Chart
x = data['pclass'].value_counts()
plt.pie(x.values,labels=x.index)
plt.show()
```



In [11]:
```
# Distance Plot
sns.distplot(data.age)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[11]: <AxesSubplot:xlabel='age', ylabel='Density'>

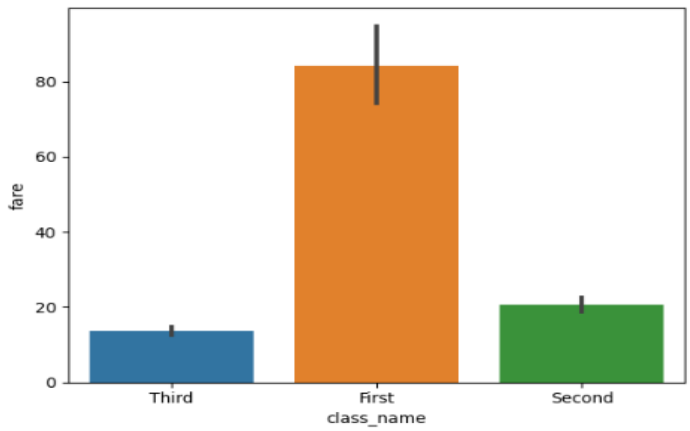


3. Visualization - Bivariate Analysis

Categorical VS Numerical

In [12]:
```
# Bar Plot
sns.barplot(x=data['class_name'], y=data['fare'])
```

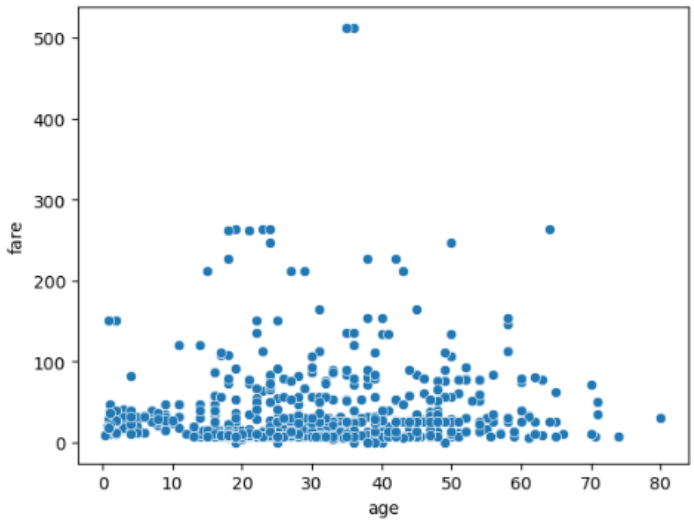
Out[12]: <AxesSubplot:xlabel='class_name', ylabel='fare'>



Numerical VS Numerical

In [13]:
```
# Scatter Plot
sns.scatterplot(x=data['age'],y=data['fare'])
```

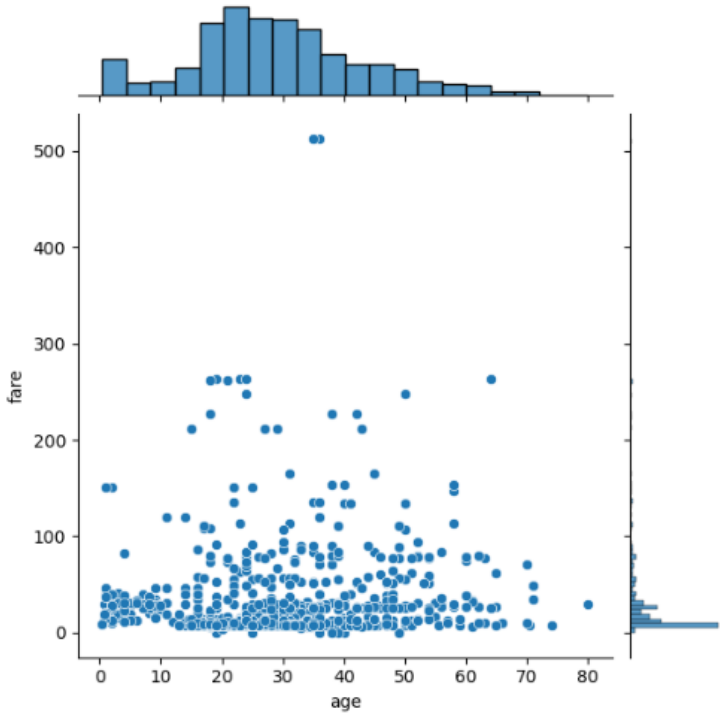
Out[13]:
```
<AxesSubplot:xlabel='age', ylabel='fare'>
```



In [14]:
```
# Join Plot
sns.jointplot(data.age,data.fare)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword ar
gs: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit
keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[14]:
```
<seaborn.axisgrid.JointGrid at 0x2686995fa90>
```

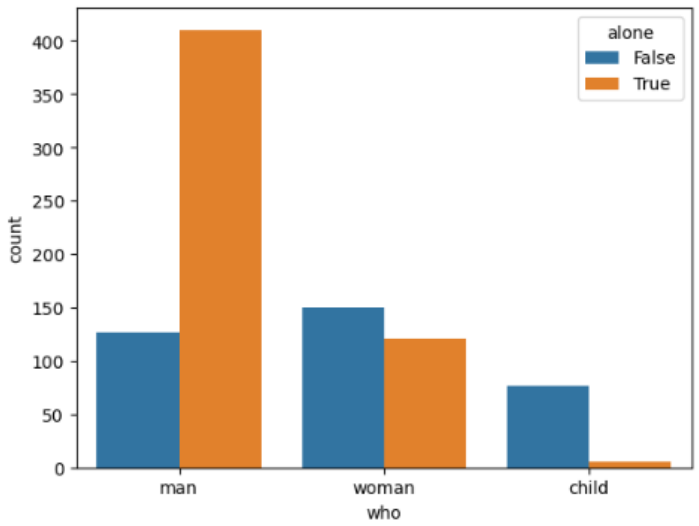


Categorical VS Categorical

In [15]:
```
# Count PLot
sns.countplot(data['who'],hue=data['alone'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

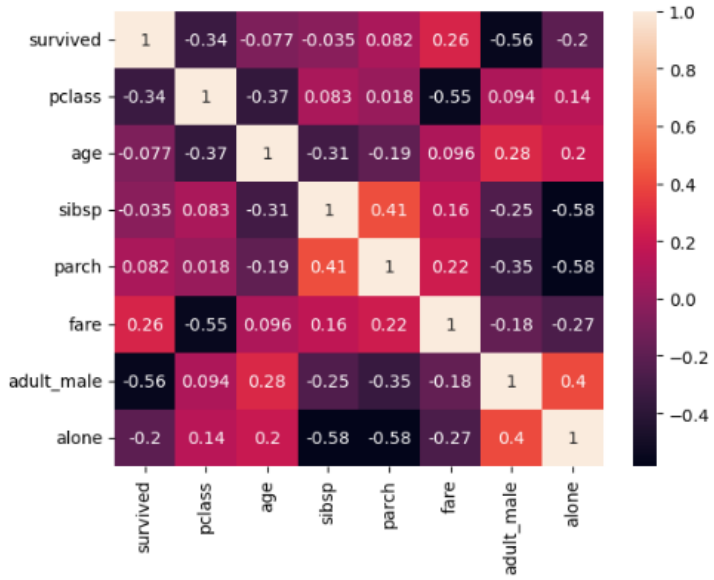
Out[15]: <AxesSubplot:xlabel='who', ylabel='count'>



3. Visualization - Multivariate Analysis ¶

In [16]:
```
# Heat Map
sns.heatmap(data.corr(), annot=True)
```

Out[16]: <AxesSubplot:>



4. Descriptive Statistics

In [17]:
```
data.describe()
```

Out[17]:

	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

In [18]:
```
data.mean()
```

```
C:\Users\JEYASRI VARTHINI\AppData\Local\Temp\ipykernel_21572\531903386.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  data.mean()
```

Out[18]:
```
survived       0.383838
pclass         2.308642
age           29.699118
sibsp          0.523008
parch          0.381594
fare          32.204208
adult_male     0.602694
alone          0.602694
dtype: float64
```

In [19]:
```
data.median()
```

```
C:\Users\JEYASRI VARTHINI\AppData\Local\Temp\ipykernel_21572\4184645713.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  data.median()
```

Out[19]:
```
survived       0.0000
pclass         3.0000
age           28.0000
sibsp          0.0000
parch          0.0000
fare          14.4542
adult_male     1.0000
alone          1.0000
dtype: float64
```

In [20]:
```
data.mode()
```

Out[20]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class_name	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	24.0	0	0	8.05	S	Third	man	True	C	Southampton	no	True

In [21]:
```
data.var()
```

```
C:\Users\JEYASRI VARTHINI\AppData\Local\Temp\ipykernel_21572\445316826.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  data.var()
```

Out[21]:
```
survived         0.236772
pclass           0.699015
age            211.019125
sibsp            1.216043
parch            0.649728
fare          2469.436846
adult_male       0.239723
alone            0.239723
dtype: float64
```

In [22]:
```
data.std()
```

```
C:\Users\JEYASRI VARTHINI\AppData\Local\Temp\ipykernel_21572\2723740006.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  data.std()
```

Out[22]:
```
survived       0.486592
pclass         0.836071
age           14.526497
sibsp          1.102743
parch          0.806057
fare          49.693429
adult_male     0.489615
alone          0.489615
dtype: float64
```

5. Handling Missing Values

In [23]:
```
data.isna()
```

Out[23]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class_name	who	adult_male	deck	embark_town	alive	alone
0	False	False	False	False	False	False	False	False	False	False	False	True	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	True	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	True	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	False	True	False	False	False
887	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	True	False	False	False	False	False	False	False	True	False	False	False
889	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	False	True	False	False	False

891 rows × 15 columns

In [24]:
```
data.isnull().any()
```

Out[24]:
```
survived       False
pclass         False
sex            False
age             True
sibsp          False
parch          False
fare           False
embarked        True
class_name     False
who            False
adult_male     False
deck            True
embark_town     True
alive          False
alone          False
dtype: bool
```

In [25]:
```
data.isnull().sum()
```

Out[25]:
```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class_name       0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

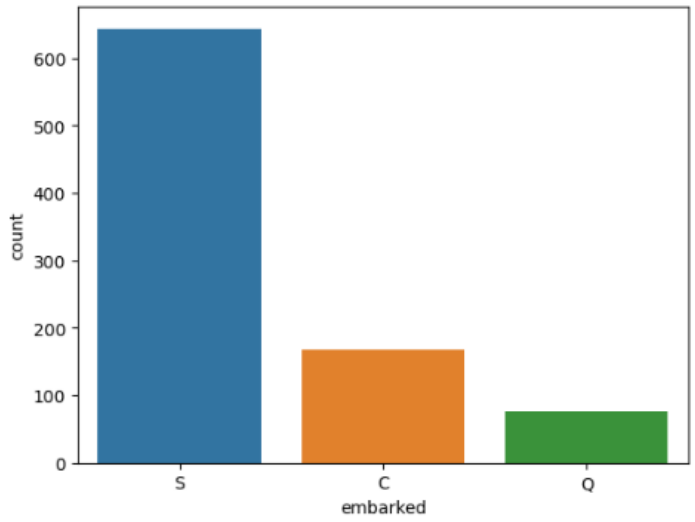
In [26]:
```
# Replacing Missing Values of age (Numerical attribute) with its Mean
data['age'].fillna(data['age'].mean(),inplace=True)
data['age'].isnull().sum()
```

Out[26]:
```
0
```

In [27]:
```
# Frequently occuring category of embarked (Categorical attribute)
sns.countplot(data['embarked'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

Out[27]: <AxesSubplot:xlabel='embarked', ylabel='count'>



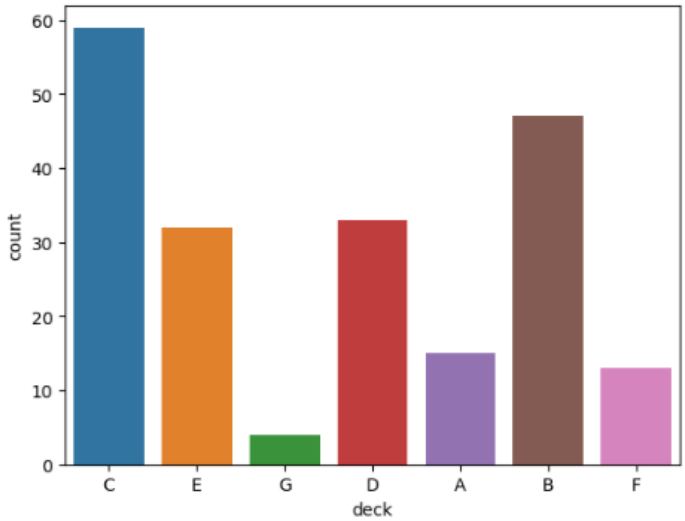
In [28]:
```
# Replacing Missing Values of embarked (Categorical attribute) with frequently occuring category
data['embarked'].fillna('S',inplace=True)
data['embarked'].isnull().sum()
```

Out[28]: 0

In [29]:
```
# Frequently occuring category of deck (Categorical attribute)
sns.countplot(data['deck'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

Out[29]: <AxesSubplot:xlabel='deck', ylabel='count'>



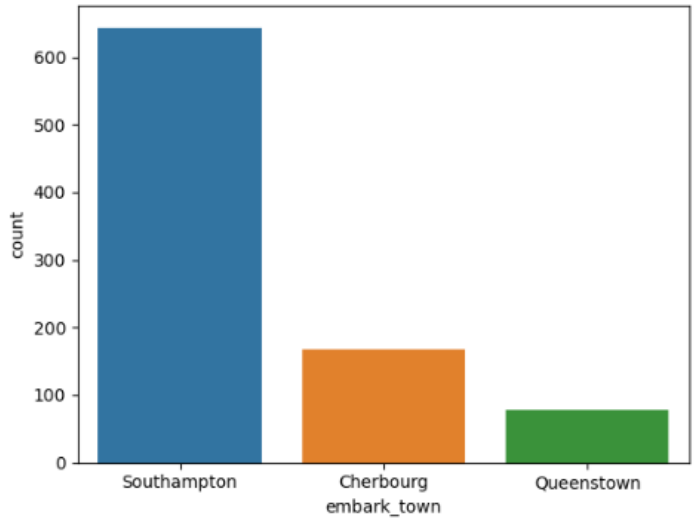
In [30]:
```
# Replacing Missing Values of deck (Categorical attribute) with frequently occuring category
data['deck'].fillna('C',inplace=True)
data['deck'].isnull().sum()
```

Out[30]: 0

In [31]:
```
# Frequently occuring category of embark_town (Categorical attribute)
sns.countplot(data['embark_town'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

Out[31]: <AxesSubplot:xlabel='embark_town', ylabel='count'>



In [32]:
```
# Replacing Missing Values of embark_town (Categorical attribute) with frequently occuring category
data['embark_town'].fillna('Southampton',inplace=True)
data['embark_town'].isnull().sum()
```

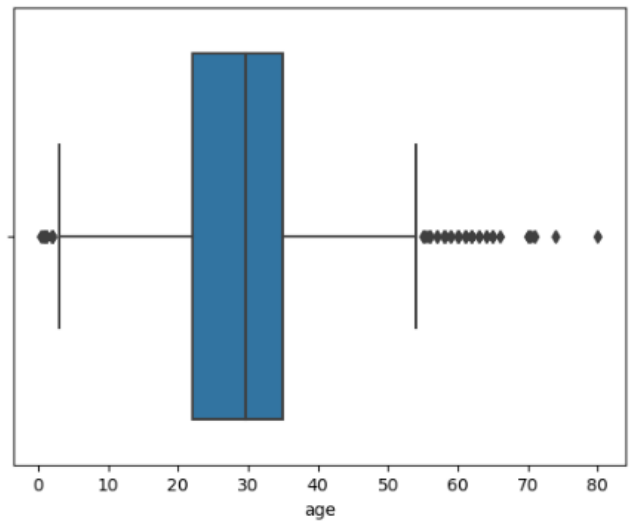
Out[32]: 0

6. Finding and replacing the outliers

In [33]:
```
# age (Numerical attribute)
sns.boxplot(data.age)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

Out[33]: <AxesSubplot:xlabel='age'>



In [34]:
```
max_threshold = data.age.quantile(0.85)
max_threshold
```

Out[34]: 42.0

In [35]:
```
min_threshold = data.age.quantile(0.20)
min_threshold
```

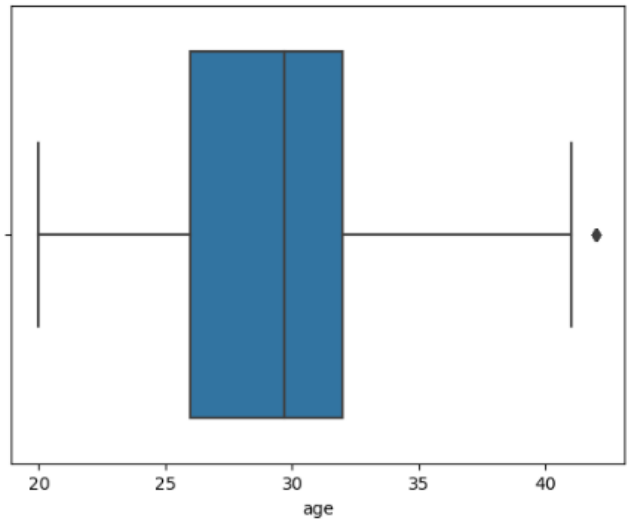
Out[35]: 20.0

In [36]:
```
data = data[data.age<=max_threshold]
data = data[data.age>=min_threshold]
```

In [37]:
```
sns.boxplot(data.age)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

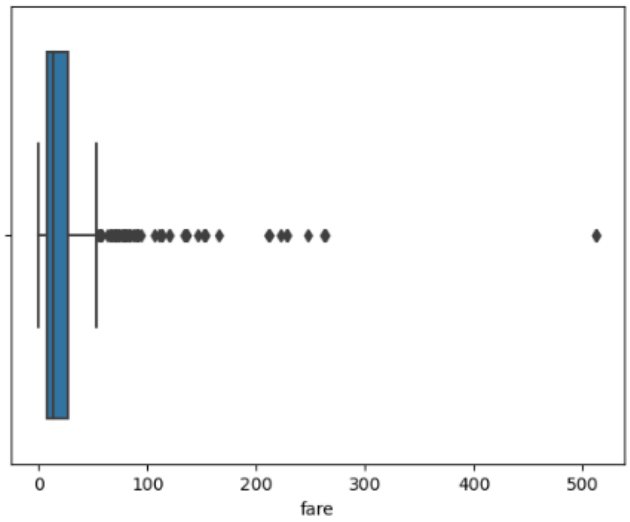
Out[37]: <AxesSubplot:xlabel='age'>



In [38]:
```
# fare (Numerical attribute)
sns.boxplot(data.fare)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

Out[38]: <AxesSubplot:xlabel='fare'>



In [39]:
```
max_threshold = data.fare.quantile(0.80)
max_threshold
```

Out[39]: 31.3425

In [40]:
```
min_threshold = data.fare.quantile(0.20)
min_threshold
```

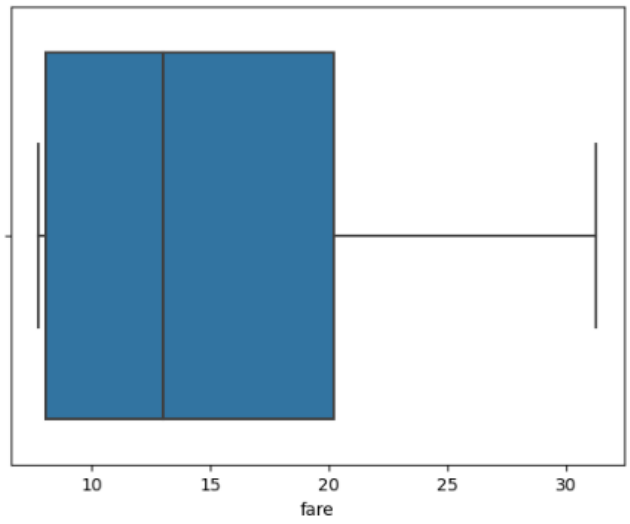
Out[40]: 7.775

In [41]:
```
data = data[data.fare<=max_threshold]
data = data[data.fare>=min_threshold]
```

In [42]:
```
sns.boxplot(data.fare)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

Out[42]: <AxesSubplot:xlabel='fare'>



7. Encoding - Label Encoding

In [43]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data.sex = le.fit_transform(data.sex)
data.embarked = le.fit_transform(data.embarked)
data.class_name = le.fit_transform(data.class_name)
data.adult_male = le.fit_transform(data.adult_male)
data.deck = le.fit_transform(data.deck)
data.embark_town = le.fit_transform(data.embark_town)
data.alive = le.fit_transform(data.alive)
data.alone = le.fit_transform(data.alone)
```

In [44]:
```
data.head()
```

Out[44]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class_name	who	adult_male	deck	embark_town	alive	alone
2	1	3	0	26.000000	0	0	7.9250	2	2	woman	0	2	2	1	1
4	0	3	1	35.000000	0	0	8.0500	2	2	man	1	2	2	0	1
5	0	3	1	29.699118	0	0	8.4583	1	2	man	1	2	1	0	1
8	1	3	0	27.000000	0	2	11.1333	2	2	woman	0	2	2	1	0
12	0	3	1	20.000000	0	0	8.0500	2	2	man	1	2	2	0	1

7. Encoding - One Hot Encoding

In [45]:
```
data = pd.get_dummies(data,columns=['who'])
```

In [46]:
```
data.head()
```

Out[46]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class_name	adult_male	deck	embark_town	alive	alone	who_man	who_woman
2	1	3	0	26.000000	0	0	7.9250	2	2	0	2	2	1	1	0	1
4	0	3	1	35.000000	0	0	8.0500	2	2	1	2	2	0	1	1	0
5	0	3	1	29.699118	0	0	8.4583	1	2	1	2	1	0	1	1	0
8	1	3	0	27.000000	0	2	11.1333	2	2	0	2	2	1	0	0	1
12	0	3	1	20.000000	0	0	8.0500	2	2	1	2	2	0	1	1	0

8. Splitting data into dependent and independent variables

In [47]:
```
# Dependent/Target variable y
y = data['survived']
y.head()
```

Out[47]:
```
2     1
4     0
5     0
8     1
12    0
Name: survived, dtype: int64
```

In [48]:
```
# Independent/Predictor variable x
x = data.drop(columns=['survived'],axis=1)
x.head()
```

Out[48]:

	pclass	sex	age	sibsp	parch	fare	embarked	class_name	adult_male	deck	embark_town	alive	alone	who_man	who_woman
2	3	0	26.000000	0	0	7.9250	2	2	0	2	2	1	1	0	1
4	3	1	35.000000	0	0	8.0500	2	2	1	2	2	0	1	1	0
5	3	1	29.699118	0	0	8.4583	1	2	1	2	1	0	1	1	0
8	3	0	27.000000	0	2	11.1333	2	2	0	2	2	1	0	0	1
12	3	1	20.000000	0	0	8.0500	2	2	1	2	2	0	1	1	0

9. Scaling the independent variables

In [49]:
```
# Minmax Scaling (Scaling values between 0 and 1)
name = x.columns
name
```

Out[49]:
```
Index(['pclass', 'sex', 'age', 'sibsp', 'parch', 'fare', 'embarked',
       'class_name', 'adult_male', 'deck', 'embark_town', 'alive', 'alone',
       'who_man', 'who_woman'],
      dtype='object')
```

In [50]:
```
from sklearn.preprocessing import MinMaxScaler
scale = MinMaxScaler()
x_scaled = scale.fit_transform(x)
x_scaled
```

Out[50]:
```
array([[1.        , 0.        , 0.27272727, ..., 1.        , 0.        ,
        1.        ],
       [1.        , 1.        , 0.68181818, ..., 1.        , 1.        ,
        0.        ],
       [1.        , 1.        , 0.44086898, ..., 1.        , 1.        ,
        0.        ],
       ...,
       [0.5       , 1.        , 0.31818182, ..., 1.        , 1.        ,
        0.        ],
       [1.        , 0.        , 0.44086898, ..., 0.        , 0.        ,
        1.        ],
       [0.        , 1.        , 0.27272727, ..., 1.        , 1.        ,
        0.        ]])
```

In [51]:
```
x = pd.DataFrame(x_scaled,columns=name)
x
```

Out[51]:

	pclass	sex	age	sibsp	parch	fare	embarked	class_name	adult_male	deck	embark_town	alive	alone	who_man	who_woman
0	1.0	0.0	0.272727	0.000000	0.0	0.006383	1.0	1.0	0.0	0.333333	1.0	1.0	1.0	0.0	1.0
1	1.0	1.0	0.681818	0.000000	0.0	0.011702	1.0	1.0	1.0	0.333333	1.0	0.0	1.0	1.0	0.0
2	1.0	1.0	0.440869	0.000000	0.0	0.029077	0.5	1.0	1.0	0.333333	0.5	0.0	1.0	1.0	0.0
3	1.0	0.0	0.318182	0.000000	0.4	0.142906	1.0	1.0	0.0	0.333333	1.0	1.0	0.0	0.0	1.0
4	1.0	1.0	0.000000	0.000000	0.0	0.011702	1.0	1.0	1.0	0.333333	1.0	0.0	1.0	1.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
360	0.5	1.0	0.363636	0.000000	0.0	0.115957	1.0	0.5	1.0	0.333333	1.0	0.0	1.0	1.0	0.0
361	1.0	0.0	0.863636	0.000000	1.0	0.908511	0.5	1.0	0.0	0.333333	0.5	0.0	0.0	0.0	1.0
362	0.5	1.0	0.318182	0.000000	0.0	0.222340	1.0	0.5	1.0	0.333333	1.0	0.0	1.0	1.0	0.0
363	1.0	0.0	0.440869	0.333333	0.4	0.667021	1.0	1.0	0.0	0.333333	1.0	0.0	0.0	0.0	1.0
364	0.0	1.0	0.272727	0.000000	0.0	0.945745	0.0	0.0	1.0	0.333333	0.0	1.0	1.0	1.0	0.0

365 rows × 15 columns

10. Splitting data into Training and Testing data

In [52]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=0)
```

In [53]:
```
x_train.head()
```

Out[53]:

	pclass	sex	age	sibsp	parch	fare	embarked	class_name	adult_male	deck	embark_town	alive	alone	who_man	who_woman
295	1.0	1.0	0.409091	0.0	0.0	0.072694	1.0	1.0	1.0	0.333333	1.0	0.0	1.0	1.0	0.0
63	0.5	1.0	0.136364	0.0	0.0	0.309396	0.0	0.5	1.0	0.333333	0.0	0.0	1.0	1.0	0.0
241	1.0	0.0	0.409091	0.0	0.8	0.565957	1.0	1.0	0.0	0.333333	1.0	0.0	0.0	0.0	1.0
306	1.0	1.0	0.386364	0.0	0.0	0.354255	1.0	1.0	1.0	0.333333	1.0	0.0	1.0	1.0	0.0
317	1.0	1.0	0.440869	0.0	0.0	0.286170	1.0	1.0	1.0	0.333333	1.0	0.0	1.0	1.0	0.0

In [54]:
```
x_test.head()
```

Out[54]:

	pclass	sex	age	sibsp	parch	fare	embarked	class_name	adult_male	deck	embark_town	alive	alone	who_man	who_woman
106	1.0	0.0	0.409091	0.333333	0.2	0.114362	1.0	1.0	0.0	1.000000	1.0	0.0	0.0	0.0	1.0
259	1.0	0.0	0.863636	0.333333	1.0	1.000000	1.0	1.0	0.0	0.333333	1.0	0.0	0.0	0.0	1.0
45	1.0	1.0	0.440869	0.000000	0.0	0.005140	1.0	1.0	1.0	0.333333	1.0	0.0	1.0	1.0	0.0
26	1.0	1.0	0.272727	0.666667	0.0	0.037766	1.0	1.0	1.0	0.333333	1.0	0.0	0.0	1.0	0.0
78	0.5	1.0	0.440869	0.000000	0.0	0.309574	0.0	0.5	1.0	0.333333	0.0	0.0	1.0	1.0	0.0

In [55]:
```
y_train
```

Out[55]:
```
713    0
135    0
567    0
735    0
760    0
      ..
793    0
444    1
284    0
104    0
404    0
Name: survived, Length: 292, dtype: int64
```

In [56]:
```
y_test
```

Out[56]:
```
251    0
610    0
101    0
69     0
181    0
      ..
516    1
384    0
168    0
402    0
273    0
Name: survived, Length: 73, dtype: int64
```

JUPITER NOTEBOOK (.ipynb file) IN GITHUB REPOSITORY:

https://github.com/JeyasriVarthiniB/Smart-Bridge-Externship-in-Applied-Data-Science/blob/main/JEYASRI_VARTHINI_B_ADS_ASSIGNMENT_2.ipynb