Exploratory Data Analysis (EDA)

1. Understand Data
2. Clean Data
3. Find a Relationship bw Data

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

In [ ]:
```
# load dataset
Ks= sns.load_dataset("titanic")
```

In [ ]:
```
# save it as csv
Ks.to_csv("Titanic.csv")
```

In [ ]:
```
#
Ks.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
```

In [ ]:
```
#check rows and col of dataset
Ks.shape
```

Out[ ]:
```
(891, 15)
```

In [ ]:
```
#tail() shows last 5 rows of dataset
Ks.tail()
```

Out[ ]:

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

In [ ]:

```
#head() shows first five rows of data
Ks.head()
```

Out[ ]:

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

In [ ]:

```
#  describe function shows details of columns containing numerics values
Ks.describe()
```

Out[ ]:

	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 [ ]:

```
# find unique value in all coulmns of dataset
Ks.nunique()
```

Out[ ]:
```
survived         2
pclass           3
sex              2
age             88
sibsp            7
parch            7
fare           248
embarked         3
class            3
who              3
adult_male       2
deck             7
embark_town      3
alive            2
alone            2
dtype: int64
```

In [ ]:
```
#check col name
Ks.columns
```

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

In [ ]:
```
# will show the unique value of column:'sex'
Ks["sex"].unique()
```

Out[ ]:
```
array(['male', 'female'], dtype=object)
```

In [ ]:
```
#check unique value of two or more columns
pd.DataFrame(Ks,columns=['who','adult_male'])
```

Out[ ]:

	who	adult_male
0	man	True
1	woman	False
2	woman	False
3	woman	False
4	man	True
...	...	...
886	man	True
887	woman	False
888	woman	False
889	man	True
890	man	True

891 rows × 2 columns

In [ ]:
```
# cleaning & filtering the data
#  firstly find missing values in dataset
Ks.isnull()
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	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 [ ]:
```
Ks.isnull().sum()
# age has 177 missing/NaN values & deck has 688 NaN so we will remove deck column first
```

Out[ ]:
```
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
```

In [ ]:
```
#removing missing value data or cleaning data
Kclean= pd.DataFrame(Ks.drop(["deck"], axis =1))
Kclean.head()
```

Out[ ]:

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

In [ ]:
```
Kclean.isnull().sum()
```

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

In [ ]:
```
Kclean.shape
Kclean.head()
```

Out[ ]:

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

In [ ]:
```
#dropna() method allows the user to analyze and drop Rows with Null values
#we use it here to drop missing values from AGE column

Kclean = Kclean.dropna()
```

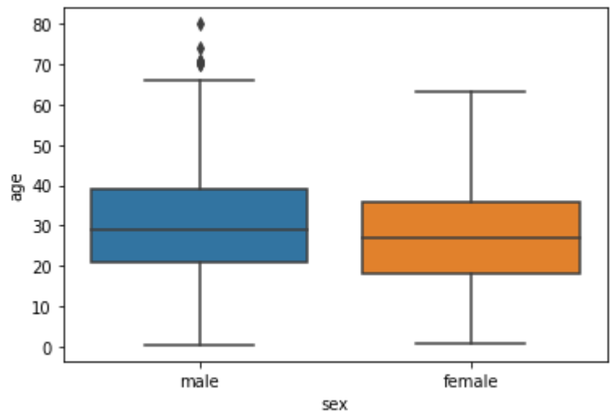
In [ ]:
```
Kclean.describe()
```

Out[ ]:

	survived	pclass	age	sibsp	parch	fare
count	712.000000	712.000000	712.000000	712.000000	712.000000	712.000000
mean	0.404494	2.240169	29.642093	0.514045	0.432584	34.567251
std	0.491139	0.836854	14.492933	0.930692	0.854181	52.938648
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	1.000000	20.000000	0.000000	0.000000	8.050000
50%	0.000000	2.000000	28.000000	0.000000	0.000000	15.645850
75%	1.000000	3.000000	38.000000	1.000000	1.000000	33.000000
max	1.000000	3.000000	80.000000	5.000000	6.000000	512.329200

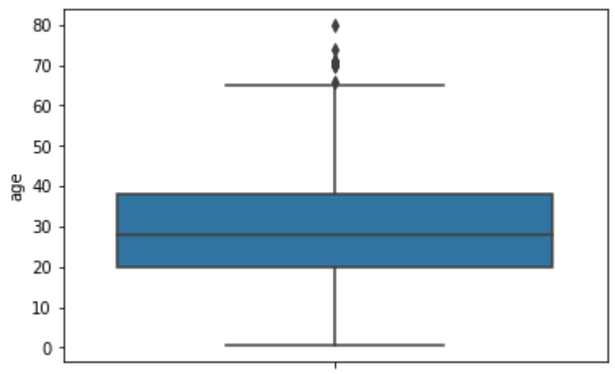
In [ ]:
```
sns.boxplot(x= "sex", y="age", data= Kclean)
```

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



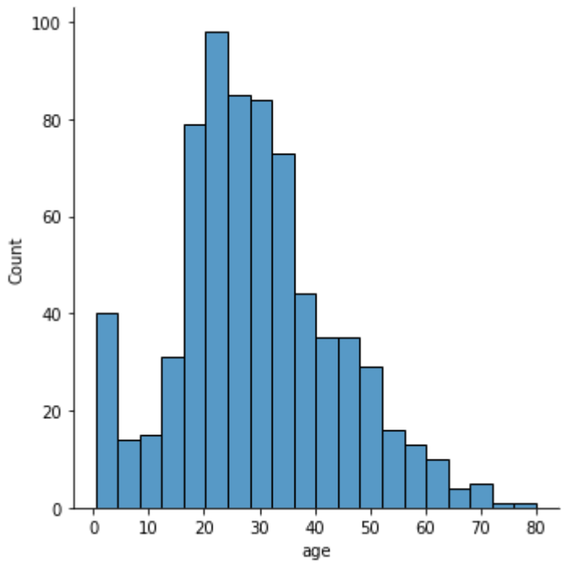
In [ ]:
```
sns.boxplot( y="age", data= Kclean)# outliar exists in age columns
```

Out[ ]:
```
<AxesSubplot:ylabel='age'>
```



In [ ]:
```
sns.displot(Kclean["age"])
```

Out[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x24d089f9510>
```



In [ ]:
```
#out liars removal
Kclean["age"].mean()
```

Out[ ]:
```
29.64209269662921
```

In [ ]:
```
# as we can see from above boxplot out liars are above 70
Kclean= Kclean[Kclean['age'] < 68]
Kclean.head()
```

Out[ ]:

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

In [ ]:
```
Kclean.shape# we can see outliars have been removed
```

Out[ ]:
```
(705, 14)
```

In [ ]:
```
sns.boxplot( y="age", data= Kclean)# outliar have been removed from boxplot
```

Out[ ]:
```
<AxesSubplot:ylabel='age'>
```