Python Chilla Pandas Assignment

> **Title= "Mr"\ Name= "Ali Nawaz"\ email = "nawazktk99@gmail.com"\ whatsapp = "03358043653"\ Artificial Intelligence Engineer at NUST\ Education : Master in Software Engineering**

Start of Chilla with ploting

Importing libraries

In [ ]:

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

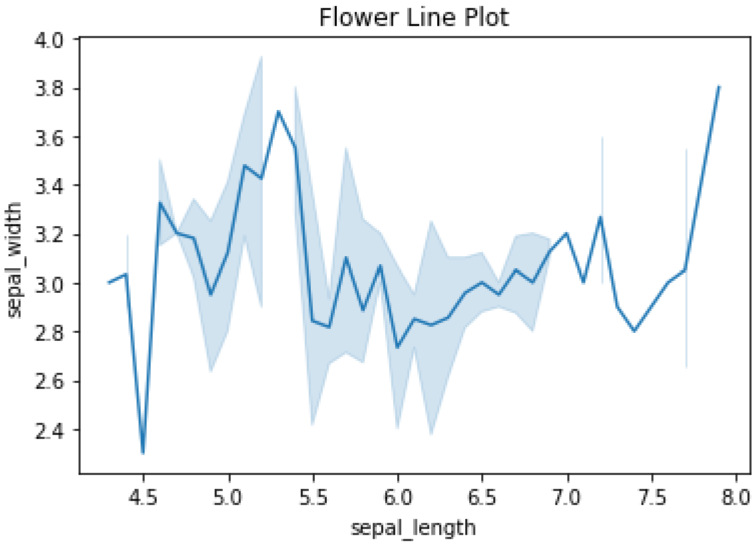
In [ ]:

```
pholl = sns.load_dataset("iris")
pholl.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [ ]:

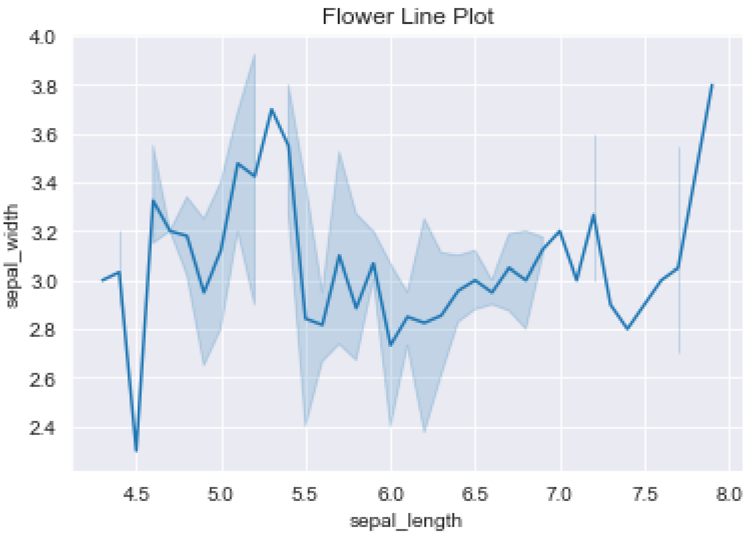
```
sns.lineplot(x='sepal_length', y = "sepal_width", data=pholl)
plt.title("Flower Line Plot")
plt.show()
```



How to change the background color of the graph

In [ ]:

```
# Use the seaborn.set() Function to Change the Background Color of Seaborn Plots in Pyt
# Use the seaborn.set_style() Function to Change the Background Color of Seaborn Plots
#  white, dark, whitegrid, darkgrid, ticks
sns.set_style("darkgrid")
sns.lineplot(x='sepal_length', y = "sepal_width", data=pholl)
plt.title("Flower Line Plot")
plt.show()
```



Different hue

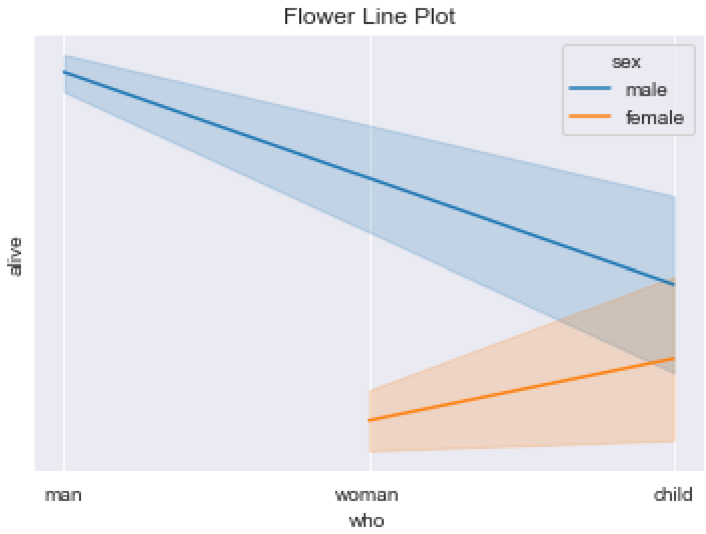
In [ ]:

```
kashti = sns.load_dataset("titanic")
kashti.to_csv("titanic.csv")
kashti.head(2)
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	e
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	S
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	

In [ ]:

```
kashti = sns.load_dataset("titanic")
sns.set_style("darkgrid")
sns.lineplot(x='who', y = "alive", hue= 'sex', data=kashti)
plt.title("Flower Line Plot")
plt.show()
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
import plotly.express as px
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
import seaborn as sns
```

In [ ]:

```
# adding all col and make mean in new col
df = pd.DataFrame(np.random.randn(10,4), columns=list('ABCD'))
df['average'] = df.mean(axis=1)
# other method
# df['average'] = df[['col1', 'col2']].mean(axis=1)
# col = df.loc[: , "col1":"col3"]
# df['mean'] = col.mean(axis=1)
df
```

Out[ ]:

	A	B	C	D	average
0	0.394691	-0.099034	1.221874	-0.034684	0.370712
1	0.618873	-1.062932	1.421820	0.947714	0.481369
2	-0.035818	0.714579	-0.759874	0.402847	0.080434
3	0.034415	-0.931410	0.049617	-0.153396	-0.250194
4	-0.312452	0.553099	1.468710	-0.653885	0.263868
5	-0.086214	-2.664909	-1.171882	-1.606185	-1.382297
6	-0.563718	-0.888630	0.337625	1.152865	0.009536
7	0.544186	-1.112088	0.047404	0.248578	-0.067980
8	0.044452	-0.103955	-0.215611	0.680726	0.101403
9	0.820109	-0.677758	-0.080284	-0.367010	-0.076236

Pakistan vs India Cereals, total production Data Plots and variation

In [ ]:

```
pak = pd.read_csv("D:/Python ka Chilla/python_chilla/data/production_faost_data_pak.csv
ind = pd.read_csv("D:/Python ka Chilla/python_chilla/data/production_faost_data_india.c
pak.head(3)
```

Out[ ]:

	Domain Code	Domain	Area Code	Area	Element Code	Element	Item Code	Item	Year Code	Year	Unit	Value
0	QCL	Crops and livestock products	165	Pakistan	5312	Area harvested	1717	Cereals, Total	1961	1961	ha	7858558
1	QCL	Crops and livestock products	165	Pakistan	5419	Yield	1717	Cereals, Total	1961	1961	hg/ha	8564
2	QCL	Crops and livestock products	165	Pakistan	5510	Production	1717	Cereals, Total	1961	1961	tonnes	6729680

Pakistan Data Analysis

In [ ]:

```
print(pak.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   Domain Code       180 non-null    object
 1   Domain            180 non-null    object
 2   Area Code         180 non-null    int64 
 3   Area              180 non-null    object
 4   Element Code      180 non-null    int64 
 5   Element           180 non-null    object
 6   Item Code         180 non-null    int64 
 7   Item              180 non-null    object
 8   Year Code         180 non-null    int64 
 9   Year              180 non-null    int64 
 10  Unit              180 non-null    object
 11  Value             180 non-null    int64 
 12  Flag              180 non-null    object
 13  Flag Description  180 non-null    object
dtypes: int64(6), object(8)
memory usage: 19.8+ KB
None
```

In [ ]:

```
# drop all rows with Nan values
pak = pak.dropna()
pak.head()
```

Out[ ]:

	Domain Code	Domain	Area Code	Area	Element Code	Element	Item Code	Item	Year Code	Year	Unit	Value
0	QCL	Crops and livestock products	165	Pakistan	5312	Area harvested	1717	Cereals, Total	1961	1961	ha	7858558
1	QCL	Crops and livestock products	165	Pakistan	5419	Yield	1717	Cereals, Total	1961	1961	hg/ha	8564
2	QCL	Crops and livestock products	165	Pakistan	5510	Production	1717	Cereals, Total	1961	1961	tonnes	6729680
3	QCL	Crops and livestock products	165	Pakistan	5312	Area harvested	1717	Cereals, Total	1962	1962	ha	8090856
4	QCL	Crops and livestock products	165	Pakistan	5419	Yield	1717	Cereals, Total	1962	1962	hg/ha	8580

In [ ]:

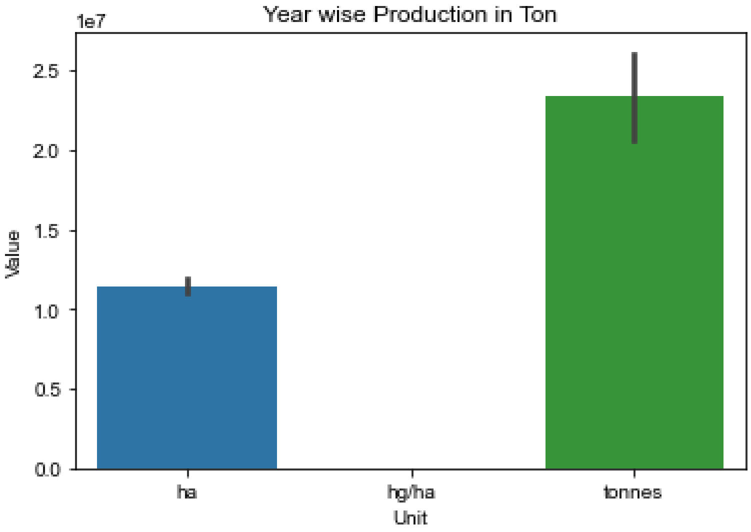
```
pak[pak['Value']>9000000].groupby(['Area', 'Item']).mean()
```

Out[ ]:

		Area Code	Element Code	Item Code	Year Code	Year	Value
Area	**Item**						
Pakistan	**Cereals, Total**	165.0	5410.074766	1717.0	1993.747664	1993.747664	1.856807e+07

In [ ]:

```
sns.barplot(x='Unit', y = "Value", data=pak, saturation=0.8)
sns.set_style('dark')
plt.title("Year wise Production in Ton")
plt.show()
```



In [ ]:

```
fig = px.pie(pak, values='Year', names='Flag', title='Pie Chart for The Crop Production
fig.show()
```

In [ ]:

```
pak.head(1)
```

Out[ ]:

	Domain Code	Domain	Area Code	Area	Element Code	Element	Item Code	Item	Year Code	Year	Unit	Value
0	QCL	Crops and livestock products	165	Pakistan	5312	Area harvested	1717	Cereals, Total	1961	1961.0	ha	7858558.0

In [ ]:

```
fig = px.sunburst(pak, path=['Area', 'Item'], values='Value',
                  color='Year', hover_data=['Unit'])
fig.show()
```

In [ ]:

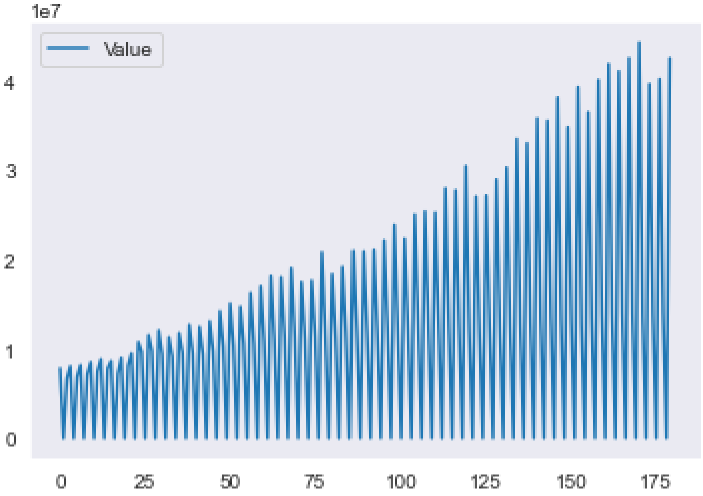
```
fig = px.bar(pak, x="Element", y="Value", color="Unit",
                 pattern_shape="Unit", pattern_shape_sequence=[".", "x", "+"])
fig.show()
```

In [ ]:

```
# plots for individual crops
pak['Value'] = pak['Value'].astype(float)
pak['Year'] = pak['Year'].astype(float)
pak['Value'].plot()

plt.legend(loc='upper left')
```

Out[ ]:

```
<matplotlib.legend.Legend at 0x116d3efc7f0>
```



In [ ]:

```
# checking columns of dataframe
print(pak.columns)
```

```
Index(['Domain Code', 'Domain', 'Area Code', 'Area', 'Element Code', 'Element',
       'Item Code', 'Item', 'Year Code', 'Year', 'Unit', 'Value', 'Flag',
       'Flag Description'],
      dtype='object')
```

Kashti Dataset Usecase

In [ ]:

```
df = sns.load_dataset('titanic')
df.head(2)
```

Out[ ]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	e
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	S
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	

In [ ]:

```
df.to_csv('D:/Python ka Chilla/python_chilla/data/titanic', index=False)
```

In [ ]:

```
dff = df.drop(['sibsp', 'embarked'], axis=1)
```

```
dff.head()
```

Out[ ]:

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

In [ ]:

```
dff.describe()
```

Out[ ]:

	survived	pclass	age	parch	fare
count	891.000000	891.000000	714.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.381594	32.204208
std	0.486592	0.836071	14.526497	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000
25%	0.000000	2.000000	20.125000	0.000000	7.910400
50%	0.000000	3.000000	28.000000	0.000000	14.454200
75%	1.000000	3.000000	38.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	6.000000	512.329200

In [ ]:

```
dff.mean()
```

Out[ ]:

```
survived       0.383838
pclass         2.308642
age           29.699118
parch          0.381594
fare          32.204208
adult_male     0.602694
alone          0.602694
dtype: float64
```

In [ ]:

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

Out[ ]:

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

In [ ]:

```
# dff.groupby(['sex', 'class']).mean()
dff.groupby(['sex']).mean()
```

Out[ ]:

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000001F8078D9C18>
```

In [ ]:

```
dff[dff['age']>18].groupby(['sex', 'class']).mean()
```

Out[ ]:

		survived	pclass	age	parch	fare	adult_male	alone
sex	**class**							
female	**First**	0.972973	1.0	37.500000	0.418919	105.043469	0.0	0.418919
	Second	0.900000	2.0	33.158333	0.500000	21.224653	0.0	0.466667
	Third	0.423729	3.0	30.161017	0.983051	14.785453	0.0	0.440678
male	**First**	0.375000	1.0	42.901042	0.270833	68.877389	1.0	0.562500
	Second	0.071429	2.0	34.750000	0.154762	20.219593	1.0	0.678571
	Third	0.133663	3.0	30.366337	0.099010	10.022624	1.0	0.851485

In [ ]:

Python Chilla Data Cleaning Notebook

> **Ali Nawaz\ Artificial Intelligence Engineer at NUST\ Education : Master in Software Engineering**

In [ ]:

```
import plotly.express as px
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv('D:/Python ka Chilla/python_chilla/data/cleaned_chilla_data.csv')
df.head(2)
```

	sex	location	age_limit	qaulification	subject	purpose	employment	blood	SIM_company	si
0	Male	Pakistan	36-40	Masters	Natural Sciences	to boost my skill set	Unemplyed	B+	U-fone	Prepa
1	Male	Pakistan	26-30	Bachelors	IT	to boost my skill set	Student	B+	U-fone	Prepa

2 rows × 23 columns

Data Cleaning and Analyzing

In [ ]:

```
# # rename_col_name
# df.rename(columns={'Qualification_completed': 'Qaulification', 'field_of_study': 'Sub
# 'Purpose_for_chilla': 'purpose','What are you?': 'Employment','Blood group ':'Blood',
# 'Your favorite programming language?':'Programming_language','Marital Status?':'Marit
# 'Where do you live?':'Living_place','Research/Working experience (Float/Int) years':'
# 'Your Weight in kg? (float)':'Weight','Height in cm? Freelancer- (Float)':'Height','H
# 'Light kitni der band hti hy? int':'Loadsheeding'}, inplace = True)
```

In [ ]:

```
# df = df.replace({'Age' : { 36-40 : 38, 26-30 : 28, 31-35 : 33, 21-25 : 23, 16-20 : 16
# df['Age'] = df['Age'].str.replace('36-40','38') other way to change
# df = df.replace({'marital_status' : { 'Yes' : 1, 'No' : 0}})
# df.housing.map(dict(yes=1, no=0))

df['experience'] = df['experience'].astype(float)#.apply(pd.to_numeric)
# df['experience'] = pd.to_numeric(df['experience'], downcast='float')
df['age'] = df['age'].astype(float)
df['weight'] = df['weight'].astype(float)
df['height'] = df['height'].astype(float)
df['coding_duration'] = df['coding_duration'].astype(float)
df['loadsheeding'] = df['loadsheeding'].astype(float)
# df.drop('age_limit', axis=1, inplace=True)
df.to_csv("D:/Python ka Chilla/python_chilla/data/cleaned_chilla_data.csv", index=False
```

In [ ]:

```
df.head(5)
```

	sex	location	age_limit	qaulification	subject	purpose	employment	blood	SIM_company
0	Male	Pakistan	36-40	Masters	Natural Sciences	to boost my skill set	Unemplyed	B+	U-fone
1	Male	Pakistan	26-30	Bachelors	IT	to boost my skill set	Student	B+	U-fone
2	Male	Pakistan	31-35	Masters	Enginnering	Switch my field of study	Employed	B+	Zong
3	Female	Pakistan	31-35	Masters	IT	to boost my skill set	Employed	O+	U-fone
4	Female	Pakistan	26-30	Masters	Enginnering	to boost my skill set	Student	A-	Mobilink

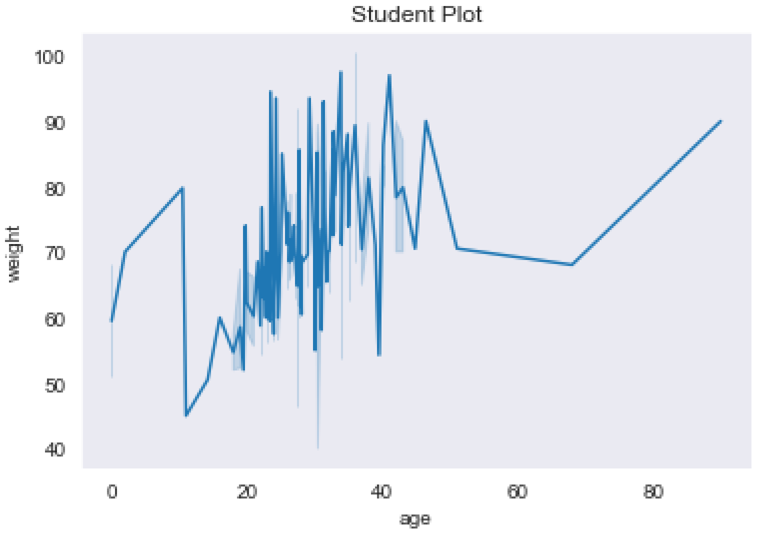
5 rows × 23 columns

In [ ]:

```
fig = px.ecdf(df, x="coding_duration", color="sex")
fig.show()
```

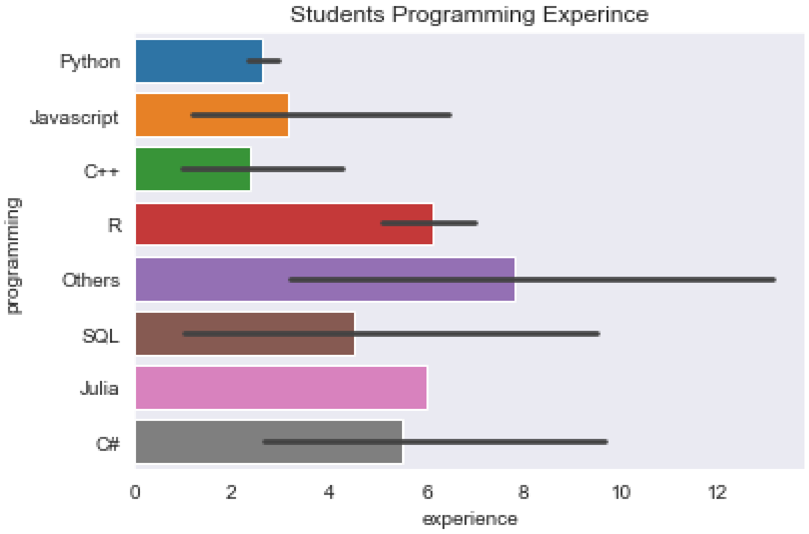
In [ ]:

```
sns.lineplot(x='age', y = "weight", data=df)
plt.title("Student Plot")
plt.show()
```



In [ ]:

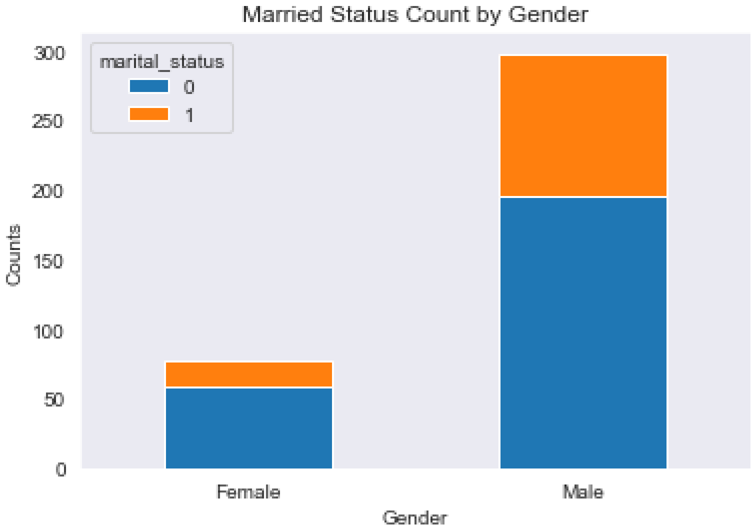
```
sns.barplot(x='experience', y = "programming", data=df, saturation=0.8)
sns.set_style('dark')
plt.title("Students Programming Experince")
plt.show()
```



In [ ]:

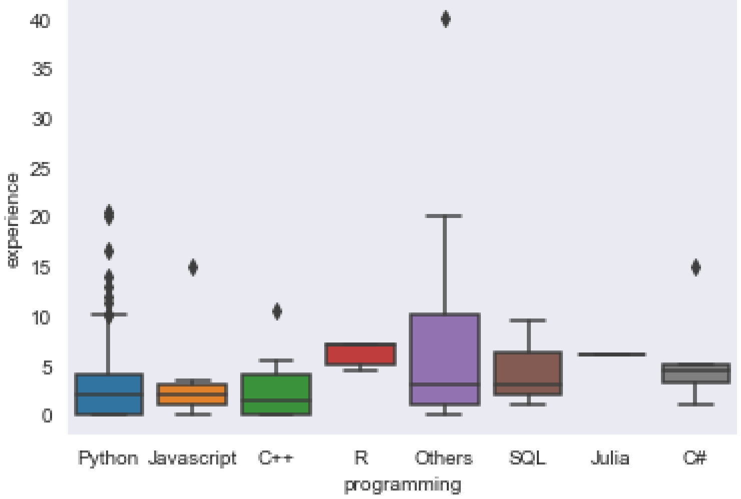
```
dff = df[['sex', 'marital_status']]

# create a pivot table
dfp = dff.pivot_table(index='sex', columns=['marital_status'], aggfunc=len)

# plot the dataframe
dfp.plot(kind='bar', stacked=True, ylabel='Counts', xlabel='Gender',
         title='Married Status Count by Gender', rot=0)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a8242737f0>
```



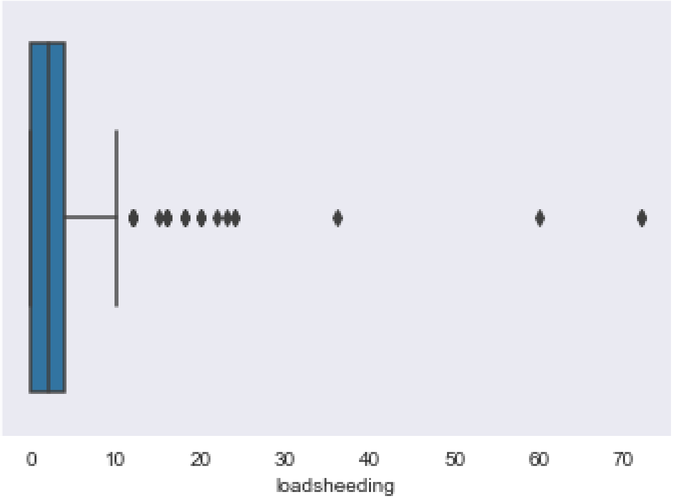
```
In [ ]: sns.boxplot(x='programming', y = "experience", data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a821dbde48>
```



```
In [ ]: sns.boxplot(x=df['loadsheeding'])
```

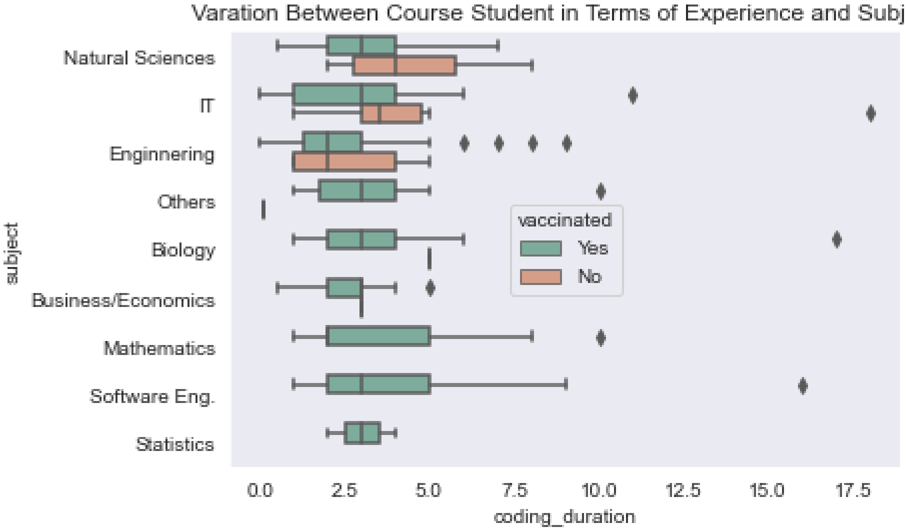
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a821cee0f0>
```



In [ ]:

```
sns.boxplot(x='coding_duration', y = "subject", data=df, hue='vaccinated', palette= 'Se
plt.title("Varation Between Course Student in Terms of Experience and Subject")
```

```
Text(0.5, 1.0, 'Varation Between Course Student in Terms of Experience and Subject')
```



In [ ]:

```
fig = px.scatter(df, x="experience", y="coding_duration", color="living_place", margina
           marginal_x="box", trendline="ols", template="simple_white")
fig.show()
```

In [ ]:

```
# df = df.query("weight == 178.0").query("living_place == 'Urban'")
# df.loc[df['experience'] < 2.0, 'programming'] = 'employment' # Represent only large c
fig = px.pie(df, values='experience', names='programming', title='Experience in Program
fig.show()
```

In [ ]:

```
fig = px.sunburst(df, path=['employment', 'qaulification'], values='coding_duration',
                  color='experience', hover_data=['location'])
fig.show()
```

In [ ]:

```
fig = px.violin(df, y="experience", x="vaccinated", color="sex", box=True, points="all"
fig.show()
```

In [ ]:

```
fig = px.scatter(df, x="weight", y="height", color="SIM_company")
fig.show()
```

In [ ]:

```
fig = px.bar(df, x="subject", y="experience", color="pc",
             pattern_shape="pc", pattern_shape_sequence=[".", "x", "+"])
fig.show()
```

In [ ]:
```
fig = px.parallel_categories(df, color="age", color_continuous_scale=px.colors.sequenti
fig.show()
```

In [ ]:
```
fig = px.bar_polar(df, r="age_limit", theta="subject", color="age_limit", template="plo
            color_discrete_sequence= px.colors.sequential.Plasma_r)
fig.show()
```

In [ ]:
```
fig = px.line(df, x='experience', y='age', color='subject', markers=True)
fig.show()
```

In [ ]:
```
fig = px.scatter_3d(df, x='age', y='experience', z='coding_duration',
              color='subject')
fig.show()
```

Data Wrangling Notebook

> **Steps**
>
> - Data collection
> - handling missing val
> - data formating
> - data normalization (scaling, centring)
> - Data binnin (for group of data)
> - making dummies of catagorical data nurmerical data
> - Clean the Data
> - Find a Relationship between data
> - analayize data
> -

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

In [ ]:
```
df = sns.load_dataset('titanic')
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	e
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	S

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

In [ ]:

```
# ere we will convert the age into days instrad of year
df['age']= df['age']*365
# assignment to remove the zeros
# df['age'] = df['age'].astype('int64')
df.dtypes
```

```
survived           int64
pclass             int64
sex                int64
age              float64
sibsp              int64
parch              int64
fare             float64
embarked          object
class           category
who               object
adult_male          bool
deck            category
embark_town       object
alive             object
alone               bool
dtype: object
```

In [ ]:

```
# two ways
# df_gender = pd.get_dummies(df['sex'])
# df_new = pd.concat([df, df_gender], axis=1)
df['sex'] = df['sex'].map({'male': 1, 'female': 0})

df.head()
```

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

Binning

> grouping of value into smaller no of val\ convert numeric into categories (1-15)(15-30) etc\ to have better understaing\

In [ ]:

```
pd.qcut(
    df.age,                         # Column to bin
    3,                         # Number of quantiles
    labels=None,               # List of labels to include
    retbins=False,             # Whether to return the bins/labels or not
    precision=3,               # The precision to store and display the bins labels
    duplicates='raise'         # If bin edges are not unique, raise a ValueError
)
```

```
0       (0.419, 23.0]
1        (34.0, 80.0]
2        (23.0, 34.0]
3        (34.0, 80.0]
4        (34.0, 80.0]
            ...
886      (23.0, 34.0]
887     (0.419, 23.0]
888               NaN
889      (23.0, 34.0]
890      (23.0, 34.0]
Name: age, Length: 891, dtype: category
Categories (3, interval[float64]): [(0.419, 23.0] < (23.0, 34.0] < (34.0, 80.0]]
```

In [ ]:

```
df['Age Groups'] = pd.qcut(df['age'], 4)
df.head()
```

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

In [ ]:

```
df['Age Groups'] = pd.qcut(
   df['age'],
   [0, 0.25, 0.5, 0.75, 1],
   labels=['0-25%', '26-49%', '51-75%', '76-100%']
)
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	e

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

In [ ]:

EDA in Python

> **Steps**
>
> - Understand the Data
> - Clean the Data
> - Fimd a Relationship between data

In [ ]:

```
import plotly.express as px
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:

```
df = sns.load_dataset('titanic')
# df = pd.read_csv('/asdf/asdf/titanic.csv')
df.head(5)
```

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

In [ ]:

```
df.describe()
```

	survived	pclass	age	sibsp	parch	fare

	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 [ ]:

```
df.shape
```

```
(891, 15)
```

In [ ]:

```
# unique values checking in data
df.nunique()
```

```
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 [ ]:

```
# col names
df.columns
```

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

In [ ]:

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

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

In [ ]:

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

```
array([22.  , 38.  , 26.  , 35.  ,   nan, 54.  ,  2.  , 27.  , 14.  ,
        4.  , 58.  , 20.  , 39.  , 55.  , 31.  , 34.  , 15.  , 28.  ,
        8.  , 19.  , 40.  , 66.  , 42.  , 21.  , 18.  ,  3.  ,  7.  ,
       49.  , 29.  , 65.  , 28.5 ,  5.  , 11.  , 45.  , 17.  , 32.  ,
       16.  , 25.  ,  0.83, 30.  , 33.  , 23.  , 24.  , 46.  , 59.  ,
       71.  , 37.  , 47.  , 14.5 , 70.5 , 32.5 , 12.  ,  9.  , 36.5 ,
       51.  , 55.5 , 40.5 , 44.  ,  1.  , 61.  , 56.  , 50.  , 36.  ,
       45.5 , 20.5 , 62.  , 41.  , 52.  , 63.  , 23.5 ,  0.92, 43.  ,
       60.  , 10.  , 64.  , 13.  , 48.  ,  0.75, 53.  , 57.  , 80.  ,
       70.  , 24.5 ,  6.  ,  0.67, 30.5 ,  0.42, 34.5 , 74.  ])
```

In [ ]:

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

```
array(['man', 'woman', 'child'], dtype=object)
```

In [ ]:

```
# Assignment
pd.unique(df[['sex', 'who' , 'age', 'fare']].values.ravel('K'))
```

```
array(['male', 'female', 'man', 'woman', 'child', 22.0, 38.0, 26.0, 35.0,
       nan, 54.0, 2.0, 27.0, 14.0, 4.0, 58.0, 20.0, 39.0, 55.0, 31.0,
       34.0, 15.0, 28.0, 8.0, 19.0, 40.0, 66.0, 42.0, 21.0, 18.0, 3.0,
       7.0, 49.0, 29.0, 65.0, 28.5, 5.0, 11.0, 45.0, 17.0, 32.0, 16.0,
       25.0, 0.83, 30.0, 33.0, 23.0, 24.0, 46.0, 59.0, 71.0, 37.0, 47.0,
       14.5, 70.5, 32.5, 12.0, 9.0, 36.5, 51.0, 55.5, 40.5, 44.0, 1.0,
       61.0, 56.0, 50.0, 36.0, 45.5, 20.5, 62.0, 41.0, 52.0, 63.0, 23.5,
       0.92, 43.0, 60.0, 10.0, 64.0, 13.0, 48.0, 0.75, 53.0, 57.0, 80.0,
       70.0, 24.5, 6.0, 0.67, 30.5, 0.42, 34.5, 74.0, 7.25, 71.2833,
       7.925, 53.1, 8.05, 8.4583, 51.8625, 21.075, 11.1333, 30.0708, 16.7,
       26.55, 31.275, 7.8542, 29.125, 7.225, 8.0292, 35.5, 31.3875, 263.0,
       7.8792, 7.8958, 27.7208, 146.5208, 7.75, 10.5, 82.1708, 7.2292,
       11.2417, 9.475, 41.5792, 15.5, 21.6792, 17.8, 39.6875, 7.8,
       76.7292, 61.9792, 27.75, 46.9, 83.475, 27.9, 15.2458, 8.1583,
       8.6625, 73.5, 14.4542, 56.4958, 7.65, 12.475, 9.5, 7.7875, 47.1,
       15.85, 34.375, 61.175, 20.575, 34.6542, 63.3583, 77.2875, 8.6542,
       7.775, 24.15, 9.825, 14.4583, 247.5208, 7.1417, 22.3583, 6.975,
       7.05, 15.0458, 26.2833, 9.2167, 79.2, 6.75, 11.5, 36.75, 7.7958,
       12.525, 66.6, 7.3125, 61.3792, 7.7333, 69.55, 16.1, 15.75, 20.525,
       25.925, 33.5, 30.6958, 25.4667, 28.7125, 0.0, 15.05, 22.025,
       8.4042, 6.4958, 10.4625, 18.7875, 113.275, 76.2917, 90.0, 9.35,
       13.5, 7.55, 26.25, 12.275, 7.125, 52.5542, 20.2125, 86.5, 512.3292,
       79.65, 153.4625, 135.6333, 19.5, 29.7, 77.9583, 20.25, 78.85,
       91.0792, 12.875, 8.85, 151.55, 23.25, 12.35, 110.8833, 108.9,
       56.9292, 83.1583, 262.375, 164.8667, 134.5, 6.2375, 57.9792,
       133.65, 15.9, 9.225, 75.25, 69.3, 55.4417, 211.5, 4.0125, 227.525,
       15.7417, 7.7292, 120.0, 12.65, 18.75, 6.8583, 7.875, 14.4, 55.9,
       8.1125, 81.8583, 19.2583, 19.9667, 89.1042, 38.5, 7.725, 13.7917,
       9.8375, 7.0458, 7.5208, 12.2875, 9.5875, 49.5042, 78.2667, 15.1,
       7.6292, 22.525, 26.2875, 59.4, 7.4958, 34.0208, 93.5, 221.7792,
       106.425, 49.5, 13.8625, 7.8292, 39.6, 17.4, 51.4792, 26.3875,
       40.125, 8.7125, 42.4, 15.55, 32.3208, 7.0542, 8.4333, 25.5875,
       9.8417, 8.1375, 10.1708, 211.3375, 13.4167, 7.7417, 9.4833, 7.7375,
       8.3625, 23.45, 25.9292, 8.6833, 8.5167, 7.8875, 37.0042, 6.45,
       6.95, 8.3, 6.4375, 39.4, 14.1083, 13.8583, 50.4958, 9.8458,
       10.5167], dtype=object)
```

Cleaning and Filtering the Data

> Finding missing value Findnig

In [ ]:

```
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
```

In [ ]:

```
# droping the col
dff = df.drop(['deck'], axis= 1)
dff.head()
```

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

In [ ]:

```
dff = dff.dropna()
dff.head(2)
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	embark_
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	Southar
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	Cherl

In [ ]:

```
dff.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
embark_town    0
alive          0
```

```
alone         0
dtype: int64
```

In [ ]:

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

```
male      453
female    259
Name: sex, dtype: int64
```

In [ ]:

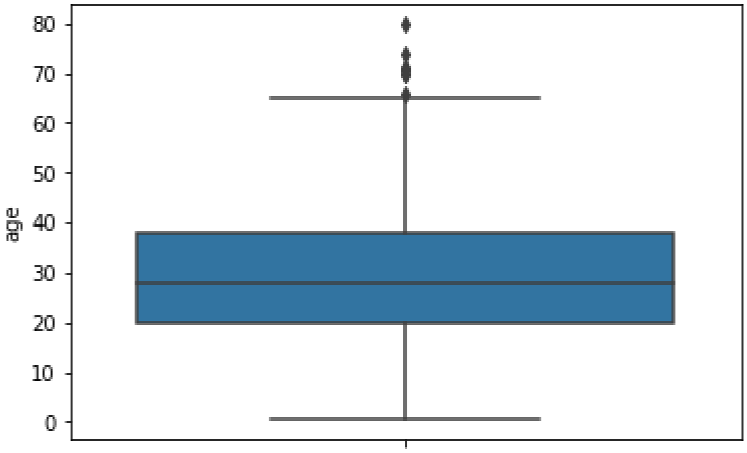
```
dff.describe()
```

	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 [ ]:

```
# out lier finding
sns.boxplot( y = 'age', data = dff)#x = 'sex',
```

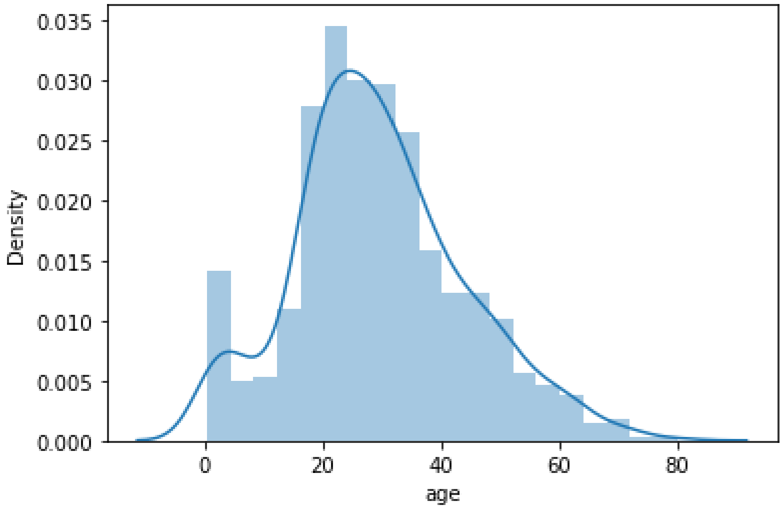
```
<matplotlib.axes._subplots.AxesSubplot at 0x20bd2890ac8>
```



In [ ]:

```
sns.distplot(df['age'])# normality check or disperstion zaida hy so for ferfactly data
```

```
C:\Users\Ali\anaconda3\envs\python-chilla\lib\site-packages\seaborn\distributions.py:261
9: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).

<matplotlib.axes._subplots.AxesSubplot at 0x20bd3006400>
```



In [ ]:

```
dff['age'].mean()
```

```
29.64209269662921
```

In [ ]:

```
dff =  dff[dff['age']< 68]
dff.head()
```

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

In [ ]:

```
print(dff.shape)
dff.head(2)
```

```
(705, 14)
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	embark_
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	Southar
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	Cherl

In [ ]:

```
dff.age.value_counts()
```

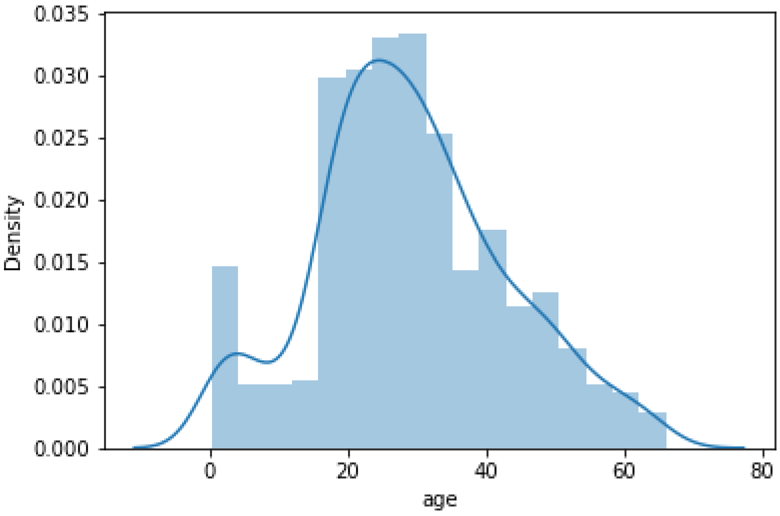
```
24.00    30
22.00    27
18.00    26
19.00    25
28.00    25
         ..
```

```
55.50     1
36.50     1
12.00     1
14.50     1
0.42      1
Name: age, Length: 83, dtype: int64
```

In [ ]:

```
sns.distplot( dff['age'])
```

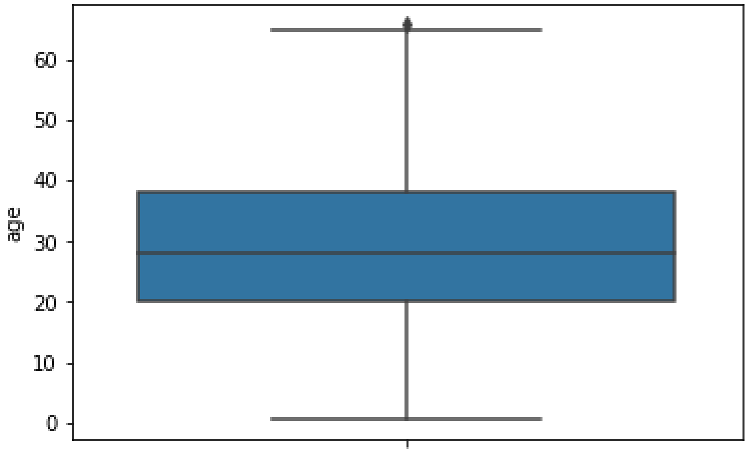
```
C:\Users\Ali\anaconda3\envs\python-chilla\lib\site-packages\seaborn\distributions.py:261
9: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).

<matplotlib.axes._subplots.AxesSubplot at 0x20bd30c6278>
```



In [ ]:

```
sns.boxplot(y= 'age', data= dff)
```

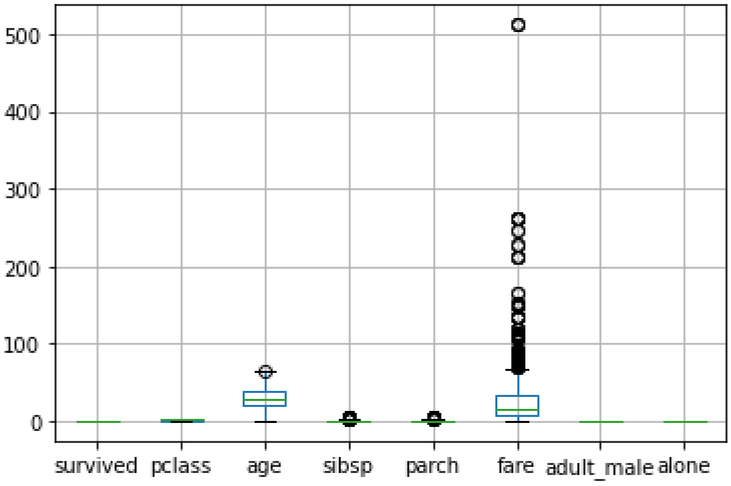
```
<matplotlib.axes._subplots.AxesSubplot at 0x20bd31694e0>
```



In [ ]:

```
dff.boxplot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x20bd31f5e48>
```



In [ ]:

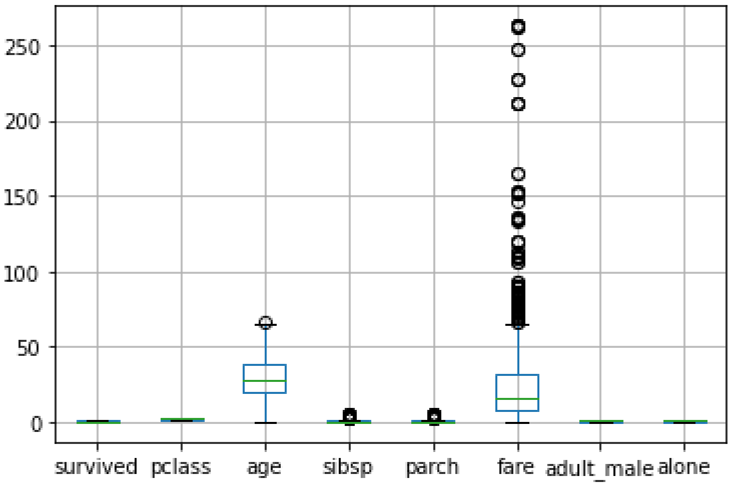
```
dff = dff[dff['fare']< 300]
dff.head()
```

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

In [ ]:

```
dff.boxplot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x20bd32de128>
```

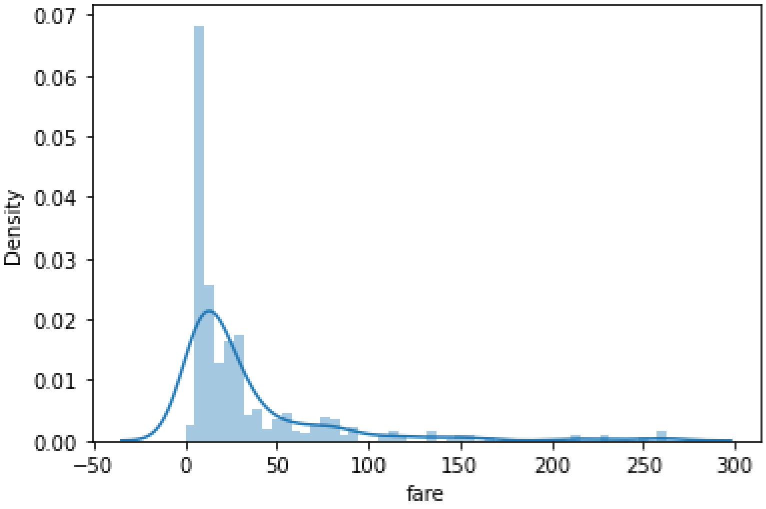


In [ ]:

```
sns.distplot(dff['fare'])
```

```
C:\Users\Ali\anaconda3\envs\python-chilla\lib\site-packages\seaborn\distributions.py:261
9: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adap
```

```
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).

<matplotlib.axes._subplots.AxesSubplot at 0x20bd342c240>
```



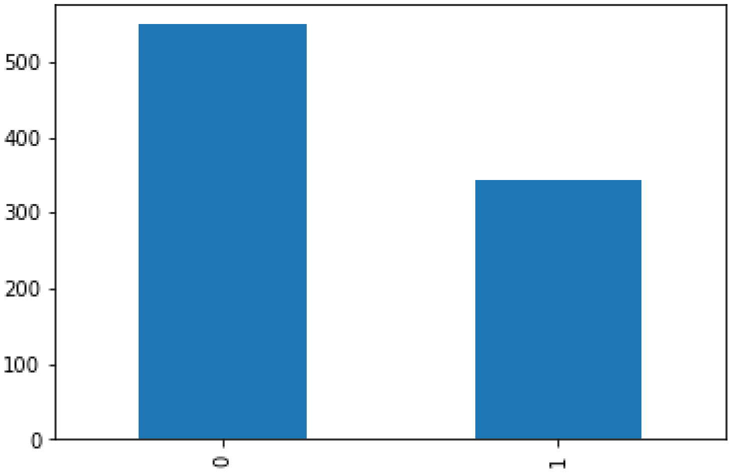
In [ ]:

```
dff.hist()
```

In [ ]:

```
pd.value_counts(df['survived']).plot.bar()
```

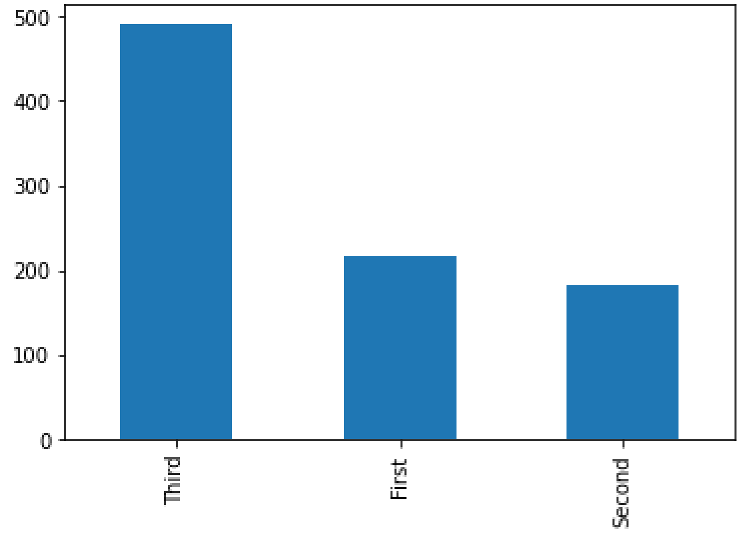
```
<matplotlib.axes._subplots.AxesSubplot at 0x20bd59c66a0>
```



In [ ]:

```
pd.value_counts(df['class']).plot.bar()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x20bd5a3a588>
```



In [ ]:

```
dff.groupby(['sex']).mean()
```

	survived	pclass	age	sibsp	parch	fare	adult_male	alone
sex								
female	0.751938	2.077519	27.717054	0.647287	0.717054	45.530120	0.00000	0.375969
male	0.202703	2.351351	30.048806	0.445946	0.272523	25.038155	0.90991	0.668919

In [ ]:

```
dff.groupby(['sex', 'class']).mean()
```

		survived	pclass	age	sibsp	parch	fare	adult_male	alone
sex	**class**								
female	**First**	0.963415	1.0	34.231707	0.560976	0.512195	103.696393	0.000000	0.353659
	Second	0.918919	2.0	28.722973	0.500000	0.621622	21.951070	0.000000	0.405405
	Third	0.460784	3.0	21.750000	0.823529	0.950980	15.875369	0.000000	0.372549
male	**First**	0.389474	1.0	40.067579	0.389474	0.336842	62.901096	0.968421	0.526316
	Second	0.153061	2.0	30.340102	0.377551	0.244898	21.221429	0.908163	0.632653
	Third	0.151394	3.0	26.143108	0.494024	0.258964	12.197757	0.888446	0.737052

Relationship or Correlation

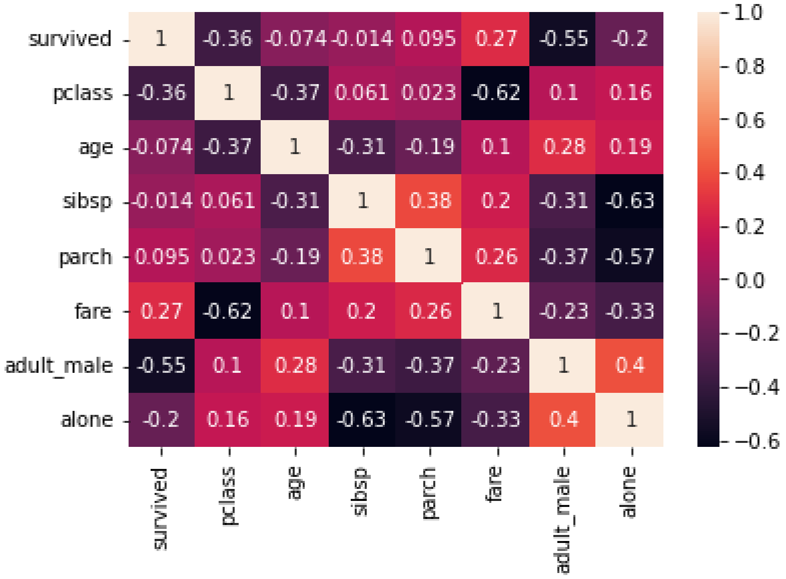
In [ ]:

```
cor = dff.corr() #do variable ka relation k interactioon ak k bharny say dosra bhar rah
```

In [ ]:

```
sns.heatmap(cor, annot = True)#only numerical data corelation can be find
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x20bd5d1e7b8>
```


survived
pclass
age
sibsp
parch
fare
adult_male
alone
1 -0.36 -0.074 -0.014 0.095 0.27 -0.55 -0.2
-0.36 1 -0.37 0.061 0.023 -0.62 0.1 0.16
-0.074 -0.37 1 -0.31 -0.19 0.1 0.28 0.19
-0.014 0.061 -0.31 1 0.38 0.2 -0.31 -0.63
0.095 0.023 -0.19 0.38 1 0.26 -0.37 -0.57
0.27 -0.62 0.1 0.2 0.26 1 -0.23 -0.33
-0.55 0.1 0.28 -0.31 -0.37 -0.23 1 0.4
-0.2 0.16 0.19 -0.63 -0.57 -0.33 0.4 1
survived
pclass
age
sibsp
parch
fare
adult_male
alone
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

```
In [ ]:
```