DATA VISULIZATION

STEP1

```
import seaborn as sns
import matplotlib.pyplot as plt
```

step 2

Double-click (or enter) to edit

load dataset

```
titanic= sns.load_dataset("titanic")
titanic.head()
```

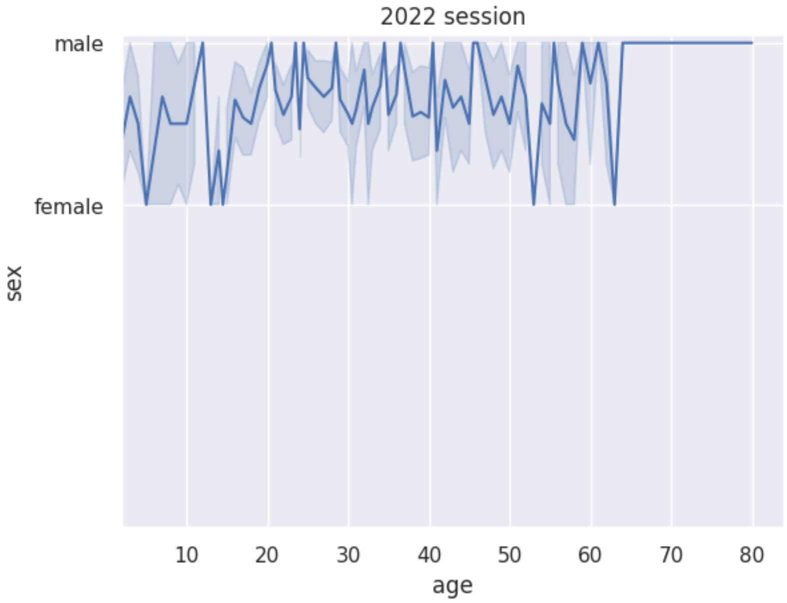
Saving...

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	a
				.0	1	0	7.2500	S	Third	man	
1	1	1	female	38.0	1	0	71.2833	C	First	woman	
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	
3	1	1	female	35.0	1	0	53.1000	S	First	woman	
4	0	3	male	35.0	0	0	8.0500	S	Third	man	

step3 PLOT A GRAPH

```
sns.lineplot(x="age",y="sex",data=titanic)
plt.xlim(2)
plt.ylim(3)
plt.title("2022 session")
```

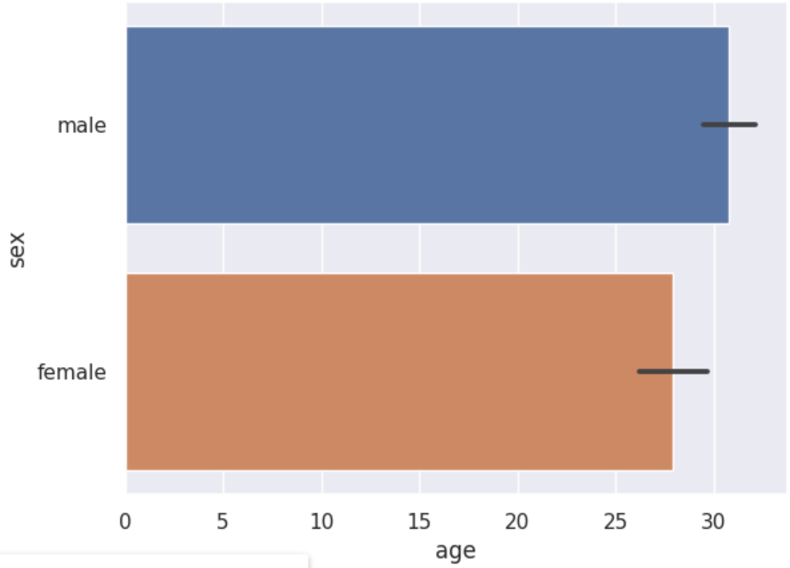
```
Text(0.5, 1.0, '2022 session')
```



BAR PLOT

```
sns.barplot(x="age",y="sex",data=titanic)
```

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

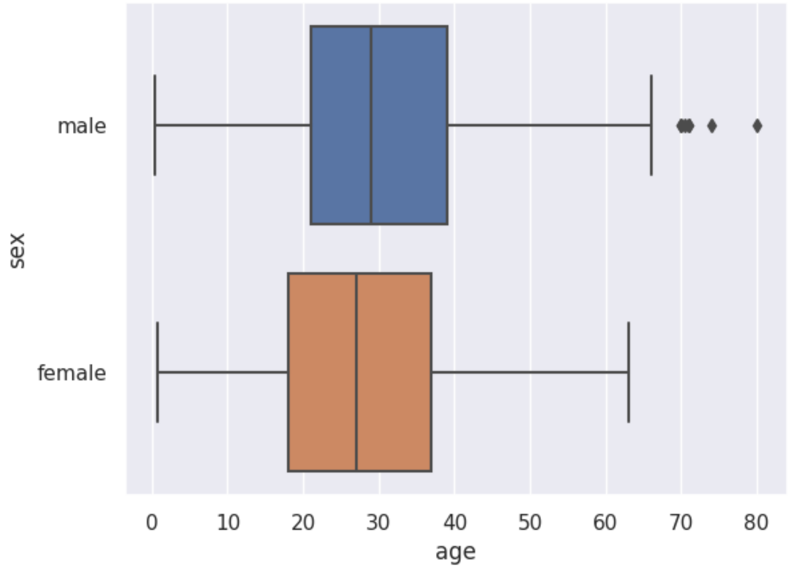


Saving...

BOX PLOT

```
sns.boxplot(x="age",y="sex",data=titanic)
```

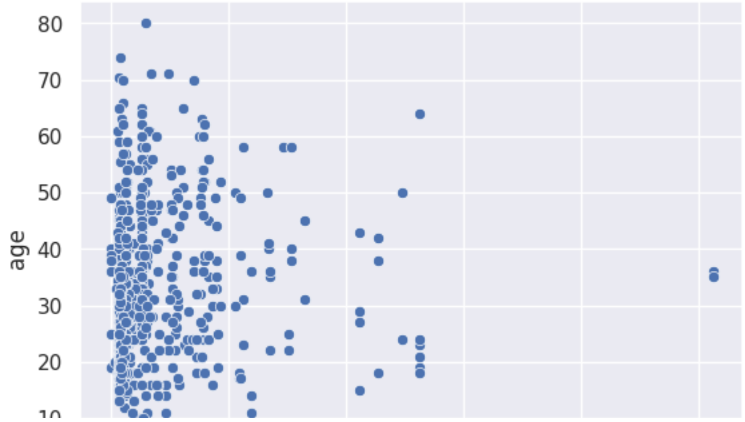
```
<Axes: xlabel='age', ylabel='sex'>
```



scatter plot

```
sns.scatterplot(x="fare",y="age",data=titanic)
```

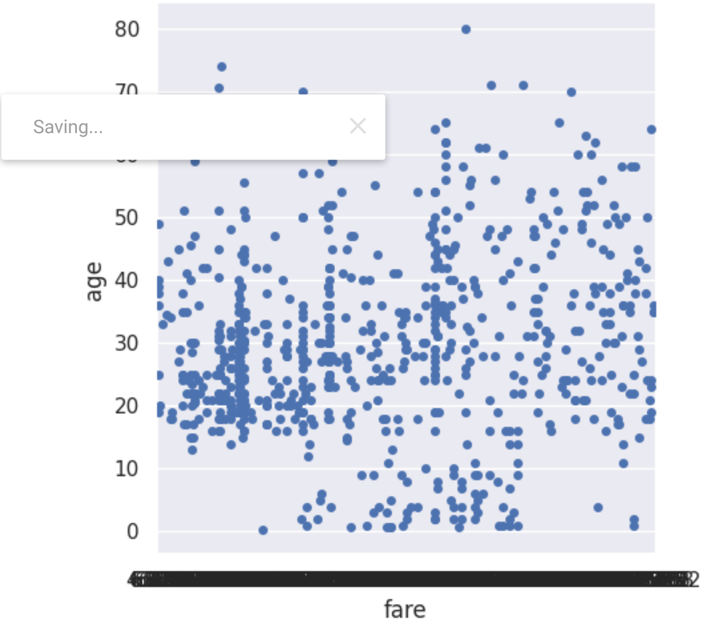
```
<Axes: xlabel='fare', ylabel='age'>
```



CATPLOT

```
sns.catplot(x="fare",y="age",data=titanic)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fc63623ba60>
```

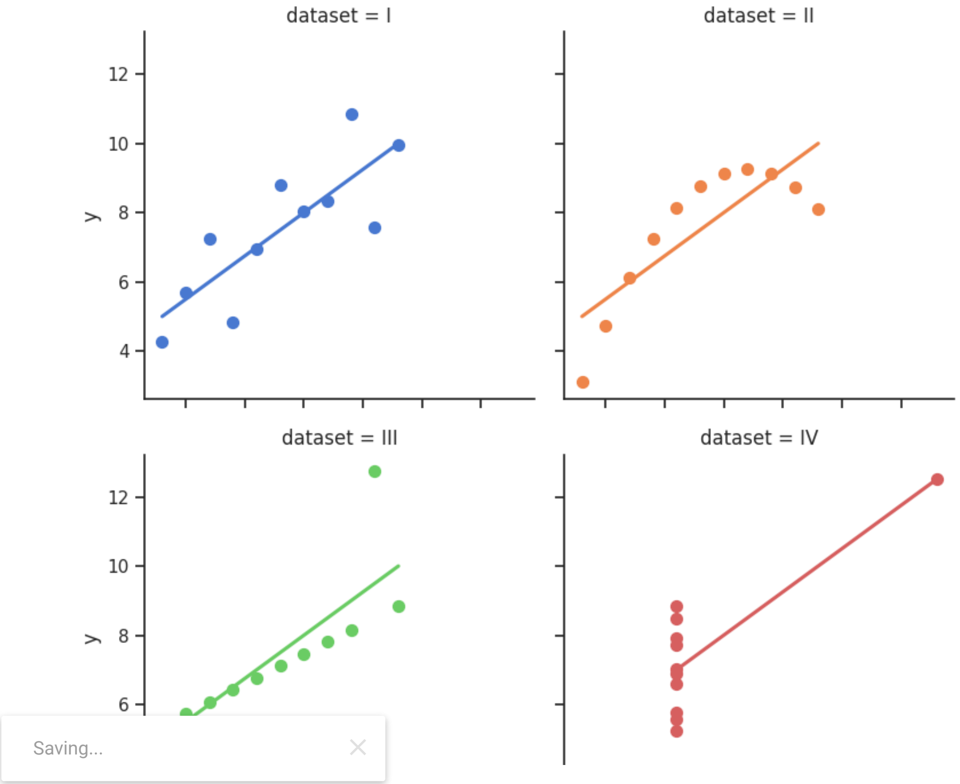


```
import seaborn as sns
sns.set_theme(style="ticks")

# Load the example dataset for Anscombe's quartet
df = sns.load_dataset("anscombe")

# Show the results of a linear regression within each dataset
sns.lmplot(
    data=df, x="x", y="y", col="dataset", hue="dataset",
    col_wrap=2, palette="muted", ci=None,
    height=4, scatter_kws={"s": 50, "alpha": 1}
)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fc636179ff0>
```

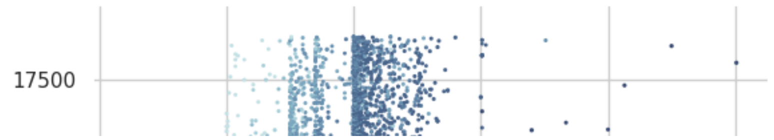


```
import seaborn as sns
import matplotlib.pyplot as plt
sns.set_theme(style="whitegrid")

# Load the example diamonds dataset
diamonds = sns.load_dataset("diamonds")

# Draw a scatter plot while assigning point colors and sizes to different
# variables in the dataset
f, ax = plt.subplots(figsize=(6.5, 6.5))
sns.despine(f, left=True, bottom=True)
clarity_ranking = ["I1", "SI2", "SI1", "VS2", "VS1", "VVS2", "VVS1", "IF"]
sns.scatterplot(x="carat", y="price",
                hue="clarity", size="depth",
                palette="ch:r=-.2,d=.3_r",
                hue_order=clarity_ranking,
                sizes=(1, 8), linewidth=0,
                data=diamonds, ax=ax)
```

```
<Axes: xlabel='carat', ylabel='price'>
```



HEAT MAP

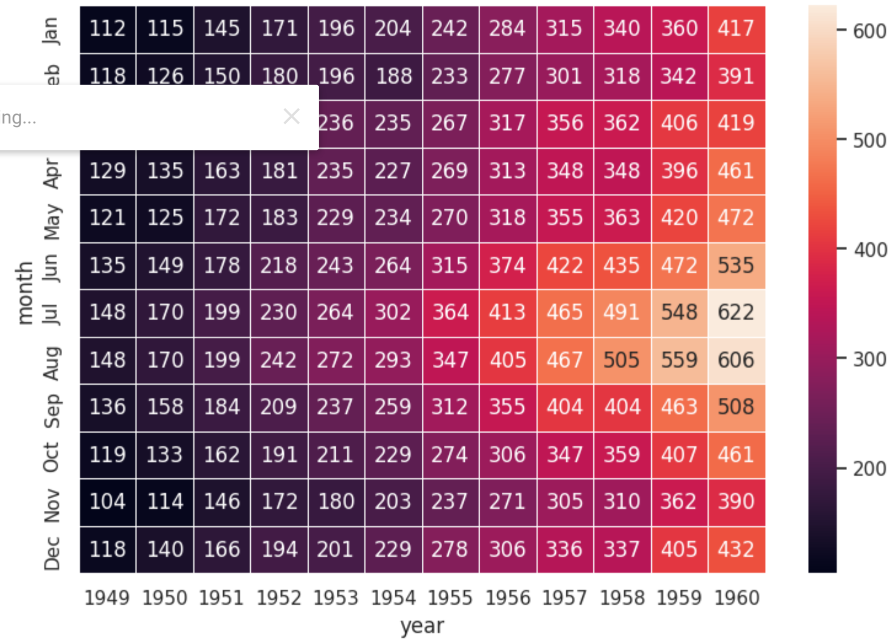
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_theme()

# Load the example flights dataset and convert to long-form
flights_long = sns.load_dataset("flights")
flights = flights_long.pivot("month", "year", "passengers")

# Draw a heatmap with the numeric values in each cell
f, ax = plt.subplots(figsize=(9, 6))
sns.heatmap(flights, annot=True, fmt="d", linewidths=.5, ax=ax)
```

```
<ipython-input-23-fd553bdfde69>:7: FutureWarning: In a future version of pandas all argu
  flights = flights_long.pivot("month", "year", "passengers")
<Axes: xlabel='year', ylabel='month'>
```

Saving...



	1949	1950	1951	1952	1953	1954	1955	1956	1957	1958	1959	1960
Jan	112	115	145	171	196	204	242	284	315	340	360	417
eb	118	126	150	180	196	188	233	277	301	318	342	391
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	236	235	267	317	356	362	406	419
Apr	129	135	163	181	235	227	269	313	348	348	396	461
May	121	125	172	183	229	234	270	318	355	363	420	472
Jun	135	149	178	218	243	264	315	374	422	435	472	535
Jul	148	170	199	230	264	302	364	413	465	491	548	622
Aug	148	170	199	242	272	293	347	405	467	505	559	606
Sep	136	158	184	209	237	259	312	355	404	404	463	508
Oct	119	133	162	191	211	229	274	306	347	359	407	461
Nov	104	114	146	172	180	203	237	271	305	310	362	390
Dec	118	140	166	194	201	229	278	306	336	337	405	432

✓ 8s completed at 11:00 PM ● ✕

Saving... ✕

working on data set from seaborn library

```
import seaborn as sns
df = sns.load_dataset("tips")
df
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

checking information about data

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype   
---  ------      --------------  -----   
 0   total_bill  244 non-null    float64 
 1   tip         244 non-null    float64 
 2   sex         244 non-null    category
 3   smoker      244 non-null    category
 4   day         244 non-null    category
 5   time        244 non-null    category
 6   size        244 non-null    int64   
dtypes: category(4), float64(2), int64(1)
memory usage: 7.4 KB
```

checking first 5 entries

```
df.head()
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

checking last 5 entries

```
df.tail()
```

	total_bill	tip	sex	smoker	day	time	size
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2

summary statistics

```
df.describe()
```

	total_bill	tip	size
count	244.000000	244.000000	244.000000
mean	19.785943	2.998279	2.569672
std	8.902412	1.383638	0.951100
min	3.070000	1.000000	1.000000
25%	13.347500	2.000000	2.000000
50%	17.795000	2.900000	2.000000
75%	24.127500	3.562500	3.000000
max	50.810000	10.000000	6.000000

checking number of rows and columns

```
df.shape
```

```
(244, 7)
```

checking number of rows

```
df.shape[0]
```

```
244
```

checking number of columns

```
df.shape[1]
```

```
7
```

checking column name

```
df.columns
```

```
Index(['total_bill', 'tip', 'sex', 'smoker', 'day', 'time', 'size'], dtype='object')
```

checking row heading

```
df.index
```

```
RangeIndex(start=0, stop=244, step=1)
```

removing specific columns

```
df1=df.drop(["tip","smoker"],axis=1)
df1
```

	total_bill	sex	day	time	size
0	16.99	Female	Sun	Dinner	2
1	10.34	Male	Sun	Dinner	3
2	21.01	Male	Sun	Dinner	3
3	23.68	Male	Sun	Dinner	2
4	24.59	Female	Sun	Dinner	4
...	...	...	...	...	...
239	29.03	Male	Sat	Dinner	3
240	27.18	Female	Sat	Dinner	2
241	22.67	Male	Sat	Dinner	2
242	17.82	Male	Sat	Dinner	2
243	18.78	Female	Thur	Dinner	2

244 rows × 5 columns

checking missig values

```
df.isnull
```

```
<bound method DataFrame.isnull of      total_bill   tip     sex smoker   day    time  size
0         16.99  1.01  Female     No   Sun  Dinner     2
1         10.34  1.66    Male     No   Sun  Dinner     3
2         21.01  3.50    Male     No   Sun  Dinner     3
3         23.68  3.31    Male     No   Sun  Dinner     2
4         24.59  3.61  Female     No   Sun  Dinner     4
..          ...   ...     ...    ...   ...     ...   ...
239       29.03  5.92    Male     No   Sat  Dinner     3
240       27.18  2.00  Female    Yes   Sat  Dinner     2
241       22.67  2.00    Male    Yes   Sat  Dinner     2
242       17.82  1.75    Male     No   Sat  Dinner     2
243       18.78  3.00  Female     No  Thur  Dinner     2

[244 rows x 7 columns]>
```

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

```
total_bill    0
tip           0
sex           0
smoker        0
day           0
time          0
size          0
dtype: int64
```

checking unique values

```
df.sex.unique()
```

```
['Female', 'Male']
Categories (2, object): ['Male', 'Female']
```

```
df.smoker.unique()
```

```
['No', 'Yes']
Categories (2, object): ['Yes', 'No']
```

```
df.time.unique()
```

```
['Dinner', 'Lunch']
Categories (2, object): ['Lunch', 'Dinner']
```

count specific values

```
df.groupby("sex").count()
```

	total_bill	tip	smoker	day	time	size
sex						
Male	157	157	157	157	157	157
Female	87	87	87	87	87	87

```
df.groupby("time").count()
```

	total_bill	tip	sex	smoker	day	size
time						
Lunch	68	68	68	68	68	68
Dinner	176	176	176	176	176	176