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

```
df=sns.load_dataset("titanic")
df.tail()
```

	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	13.000	S	Third	man	True	NaN	Southampton	no	True
887	1	1	female	19.0	0	0	30.000	S	First	woman	False	B	Cherbourg	yes	False
888	0	3	female	29.0	1	2	23.450	S	Third	woman	False	NaN	Southampton	yes	False
889	1	1	male	26.0	0	0	30.000	C	First	man	True	C	Cherbourg	yes	True
890	0	3	male	32.0	0	0	7.750	Q	Third	man	True	NaN	Queenstown	no	True

891 rows × 15 columns

```
df=sns.load_dataset("titanic")
df.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: 88.7+ KB
```

```
# 1. Use filled dataset and counts plot of passenger class
sns.countplot('pclass',data=df)
plt.show()
```

```
C:\Users\user\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the ext
ra arguments will be keyword-only. In the future, passing more arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(f"
```

```
# 2. Kde-plot heatmap to see relationship between different features and survived.
sns.heatmap(df.corr())
```

```
<ipykernel_launch_ref_1>:1: FutureWarning: Dropping of nuisance columns in DataFrame.corr() is deprecated; in a future version this will raise TypeError.
  sns.heatmap(df.corr())
```

```
df=sns.load_dataset("tips")
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	Male	Yes	Sun	Dinner	2
241	22.67	2.00	Male	Yes	Sun	Dinner	2
242	17.82	1.75	Male	No	Sun	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

```
df=sns.load_dataset("tips")
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
```

```
# 3.Insert tips dataset and also see data distribution of feature total_bill(use histogram).
sns.distplot(df['total_bill'],kde=False)
plt.show()
```

```
C:\Users\user\anaconda3\lib\site-packages\seaborn\distplot.py:2619: FutureWarning: distplot is a deprecated function and will be removed in a future ve
rsion. Please adapt your code to use either displot (a figure-level function with similar flexibility) or histplot (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

```
df=sns.load_dataset("iris")
df.tail()
```

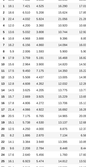
	sepal_length	sepal_width	petal_length	petal_width	species
145	6.7	3.3	5.7	2.5	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

```
df=sns.load_dataset("iris")
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   sepal_length 150 non-null    float64
 1   sepal_width  150 non-null    float64
 2   petal_length 150 non-null    float64
 3   petal_width  150 non-null    float64
 4   species      150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0 KB
```

```
# 4.Plot a scatterplot and scatter plot for iris dataset feature sepal_length and sepal_width.
sns.jointplot(x='sepal_length',y='sepal_width',data=df[iris])
plt.show()
```



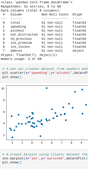

```
df=sns.load_dataset("car_crashes")
df.tail
```

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
44	23.9	7.182	6.695	16.530	15.340	734.55	145.08	AL
45	19.1	7.831	8.925	14.294	17.014	804.85	120.75	WAR
46	23.6	4.010	5.308	19.024	17.464	899.47	110.35	AZ
47	22.4	4.032	5.824	15.456	21.280	827.34	142.39	AR
48	12.0	4.200	3.360	10.920	10.680	679.41	108.63	CA
49	13.8	3.036	2.208	10.764	12.420	828.289	137.13	COR
50	16.9	4.563	2.873	9.396	8.450	1268.73	157.32	CT
51	15.2	3.800	3.344	10.944	10.336	1137.87	167.63	DE
52	9.9	2.376	1.782	9.603	9.900	1273.89	136.05	DC
53	17.8	5.700	5.251	14.440	16.020	1305.21	144.18	FL
54	19.4	2.964	3.883	14.920	14.550	912.19	142.80	GA
55	21.9	7.878	7.227	14.391	15.225	953.20	139.60	ID
56	19.4	5.006	4.657	13.193	14.958	911.49	51.70	IL
57	22.1	5.746	5.083	19.227	17.238	1044.18	161.27	IN
58	14.7	3.528	4.339	12.779	12.779	743.48	108.10	IA
59	20.1	5.726	4.020	15.127	12.884	1011.80	96.87	KS
60	12.9	4.386	4.899	9.675	12.375	1048.78	160.76	KY
61	8.2	1.968	2.870	7.134	8.560	1351.14	135.63	MN
62	14.1	3.384	2.820	11.985	10.857	1115.61	152.28	MS
63	9.6	2.208	2.784	8.448	8.448	777.18	133.35	MN
64	17.6	4.400	3.168	1.760	17.600	896.07	155.77	MO
65	23.1	3.003	5.478	19.017	13.629	758.82	144.48	MO
66	19.6	4.508	4.116	15.092	15.092	1004.75	137.30	MT
67	14.9	1.937	3.131	12.427	12.143	730.28	114.82	NE
68	14.7	3.528	4.704	12.495	14.553	1029.87	138.71	NV
69	11.3	2.486	2.486	10.962	9.605	746.54	135.11	NH
70	12.2	1.793	3.176	9.522	8.784	1301.52	158.85	NJ
71	18.1	3.986	4.349	12.135	16.832	1053.48	120.75	NM
72	12.3	3.936	2.583	10.824	9.840	1234.31	159.01	NY
73	16.8	5.208	3.192	15.792	13.608	799.24	117.62	NC
74	23.9	5.497	10.036	23.661	20.554	688.75	109.72	ND
75	19.1	3.440	7.456	12.038	13.384	967.79	123.82	OH
76	18.1	5.793	5.072	16.109	16.835	1004.75	115.70	OK
77	19.6	4.228	3.920	9.576	11.368	804.71	116.29	OR
78	15.7	5.024	5.342	12.717	14.915	1148.99	148.65	PA
79	18.2	3.276	3.458	15.470	14.196	1137.05	116.29	RI
80	20.4	7.956	7.272	17.004	16.575	1304.75	148.65	SC
81	12.3	3.936	3.198	10.947	10.455	869.18	132.28	SD
82	15.4	3.014	3.542	10.012	10.934	889.91	90.87	TN
83	14.3	4.006	3.146	11.869	11.726	790.90	143.19	TX
84	15.9	2.703	2.703	12.564	14.679	1004.75	148.65	TX
85	19.4	5.050	4.074	12.806	15.334	992.61	132.98	UT
86	11.3	2.599	2.486	10.096	12.995	745.28	108.61	VT
87	13.7	3.425	3.151	11.648	11.376	768.95	133.73	VA
88	19.0	4.452	3.800	8.892	9.114	890.03	111.62	WA
89	19.6	4.900	4.900	15.092	15.092	992.61	152.88	WV
90	19.4	4.850	4.654	5.820	11.082	670.31	106.62	WI
91	17.8	7.300	5.500	14.094	15.600	791.14	133.06	WY

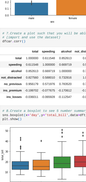
```
df=sns.load_dataset("car_crashes")
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51 entries, 0 to 50
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   total         51 non-null     float64
 1   speeding      51 non-null     float64
 2   alcohol       51 non-null     float64
 3   not_distracted 51 non-null    float64
 4   no_previous   51 non-null     float64
 5   ins_premium   51 non-null     float64
 6   ins_losses    51 non-null     float64
 7   abbrev        51 non-null     object
dtypes: float64(7), object(1)
memory usage: 3.3+ KB
```

```
# 5.Use car_crashes dataset from seaborn and find relationship between speeding and alcohol.
plt.scatter(x='speeding',y='alcohol',data=df)
plt.show()
```



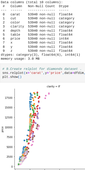

```
# 6.Create barplot using titanic dataset the survivors of the titanic crash based on sex.
sns.countplot('survived',data=df[sex])
plt.show()
```




```
# 7.Create a plot such that you will be able to find which features are highly correlated for car_crashes.
# Compare and use the dataset.
df=sns.load_dataset("car_crashes")
df.corr()
```

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses
total	1.000000	0.911048	0.852613	0.827935	0.872274	0.662068	0.636045
speeding	0.911048	1.000000	0.875632	0.805423	0.723191	0.571976	0.489848
alcohol	0.852613	0.875632	1.000000	0.732810	0.784659	0.548022	0.512147
not_distracted	0.827935	0.805423	0.732810	1.000000	0.730307	0.807042	0.723807
no_previous	0.872274	0.723191	0.784659	0.730307	1.000000	0.524845	0.657688
ins_premium	0.662068	0.571976	0.548022	0.807042	0.524845	1.000000	0.925318
ins_losses	0.636045	0.489848	0.512147	0.723807	0.657688	0.925318	1.000000

```
# 8.Create a boxplot to see 3 number summary for iris dataset.
sns.boxplot(x='day',y='total_bill',data=df[tip])
plt.show()
```




```
df=sns.load_dataset("diamonds")
df.tail
```

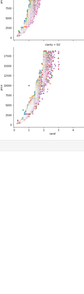
	carat	cut	color	clarity	depth	table	price	x	y	z
53937	0.71	Good	E	SI2	63.5	55.0	2757	5.69	5.75	3.61
53938	0.70	Premium	E	SI1	59.8	62.0	2757	5.69	5.64	3.39
53939	0.86	Fair	E	SI2	55.8	60.0	2757	6.15	6.12	3.74
53940	0.75	Premium	I	VS2	62.4	58.0	2757	5.83	5.77	3.60
53941	0.50	Good	I	SI2	63.5	56.0	2757	5.90	5.85	3.71

53940 rows × 10 columns

```
df=sns.load_dataset("diamonds")
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 10 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   carat   53940 non-null  float64
 1   cut     53940 non-null  category
 2   color   53940 non-null  category
 3   clarity 53940 non-null  category
 4   depth   53940 non-null  float64
 5   table   53940 non-null  float64
 6   price   53940 non-null  int64
 7   x       53940 non-null  float64
 8   y       53940 non-null  float64
 9   z       53940 non-null  float64
dtypes: category(3), float64(6), int64(1)
memory usage: 3.9 MB
```

```
# 9.Create a bigger plot for diamonds dataset.
sns.relplot(x='carat',y='price',data=df[kind,hue='color',style='clarity',size='depth',col='clarity',col_wrap=2)
plt.show()
```




```
# 10.Create a violin plot for diamonds dataset.
```