Working on Dataset 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 five 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

Summary Satatistics

```
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 No.of Rows and Colums

```
df.shape
```

```
(244, 7)
```

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

```
244
```

```
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)
```

Checking Missing Vaules

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

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

Checking Unique Values

```
df.day.unique()
```

```
['Sun', 'Sat', 'Thur', 'Fri']
Categories (4, object): ['Thur', 'Fri', 'Sat', 'Sun']
```

Remove specific columns

```
df1=df.drop(["day","time"],axis=1)
df1
```

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

244 rows × 5 columns

Grouping

```
df1.groupby(["sex"]).mean()
```

```
<ipython-input-39-753766b12dbc>:1: FutureWarning: The default value of numeric_only in D
  df1.groupby(["sex"]).mean()
```

sex	total_bill	tip	size
Male	20.744076	3.089618	2.630573
Female	18.056897	2.833448	2.459770

```
df1.groupby(["time", "sex"]).mean()
```

```
  File "<ipython-input-44-d85d00d269c6>", line 1
    df1.groupby(["time", "sex"]).mean(:
                                      ^
SyntaxError: invalid syntax
```

SEARCH STACK OVERFLOW

```
df1[df1["size"]<18].groupby(["sex", "class"]).mean()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-38-992c5ed5a534> in <cell line: 1>()
----> 1 df1[df1["size"]<18].groupby(["sex", "class"]).mean()

                                  2 frames
/usr/local/lib/python3.10/dist-packages/pandas/core/groupby/grouper.py in
get_grouper(obj, key, axis, level, sort, observed, mutated, validate, dropna)
    886                 in_axis, level, gpr = False, gpr, None
    887             else:
--> 888                 raise KeyError(gpr)
    889         elif isinstance(gpr, Grouper) and gpr.key is not None:
    890             # Add key to exclusions

KeyError: 'class'
```

SEARCH STACK OVERFLOW

Colab paid products - Cancel contracts here

0s completed at 7:19 PM