In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [2]:
```
tips=sns.load_dataset('tips')
```

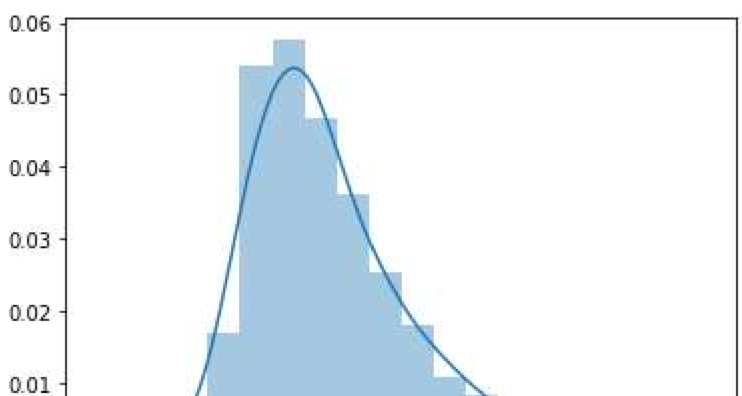
In [3]:
```
tips.head()
```

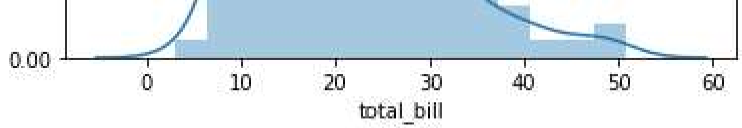
Out[3]:

	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

In [4]:
```
sns.distplot(tips['total_bill'])
```

Out[4]: <matplotlib.axes._subplots.AxesSubplot at 0x29d18045a88>



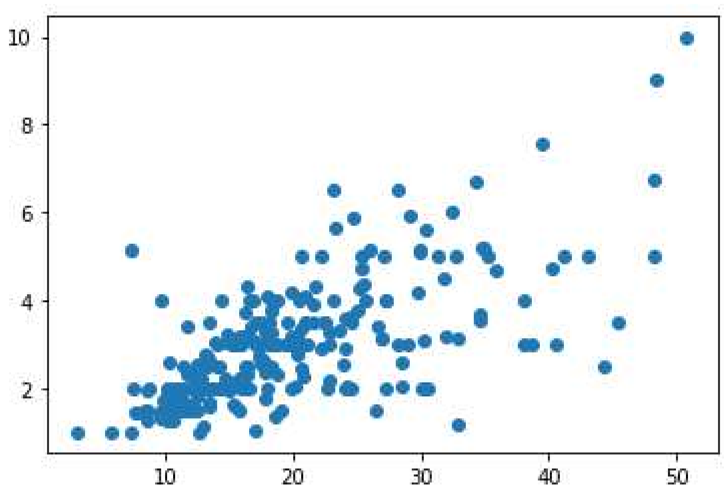


In [5]:
```
tips.info()
```

```
<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: 7.3 KB
```

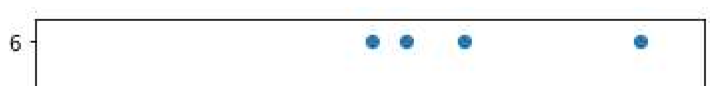
In [6]:
```
plt.scatter(tips['total_bill'], tips ['tip'])
```

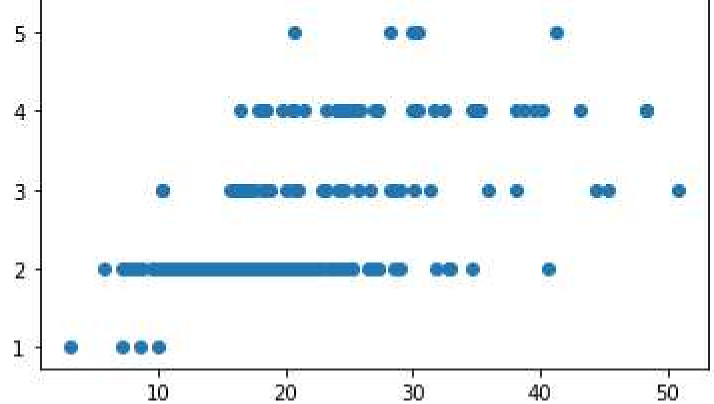
Out[6]:
```
<matplotlib.collections.PathCollection at 0x29d181a1f48>
```



In [7]:
```
plt.scatter(tips['total_bill'], tips ['size'])
```

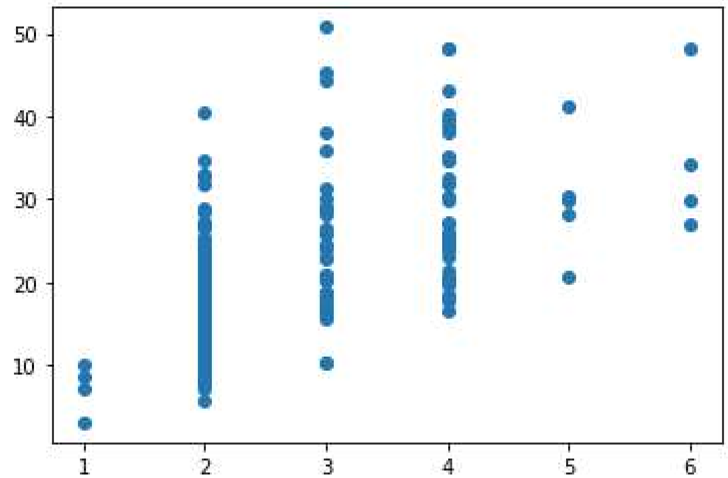
Out[7]:
```
<matplotlib.collections.PathCollection at 0x29d182176c8>
```





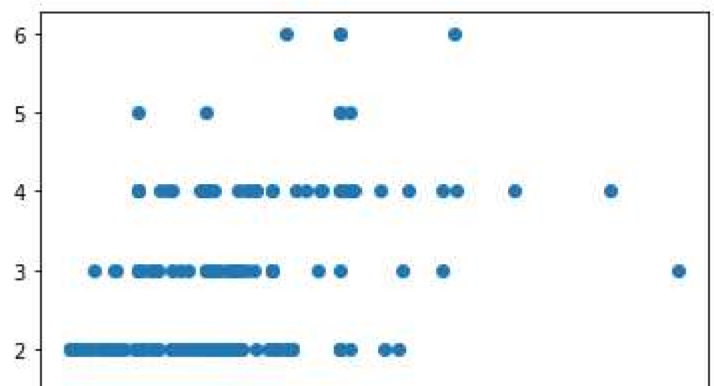

```
In [8]: plt.scatter(tips['size'], tips ['total_bill'])
```

Out[8]: <matplotlib.collections.PathCollection at 0x29d1827a408>



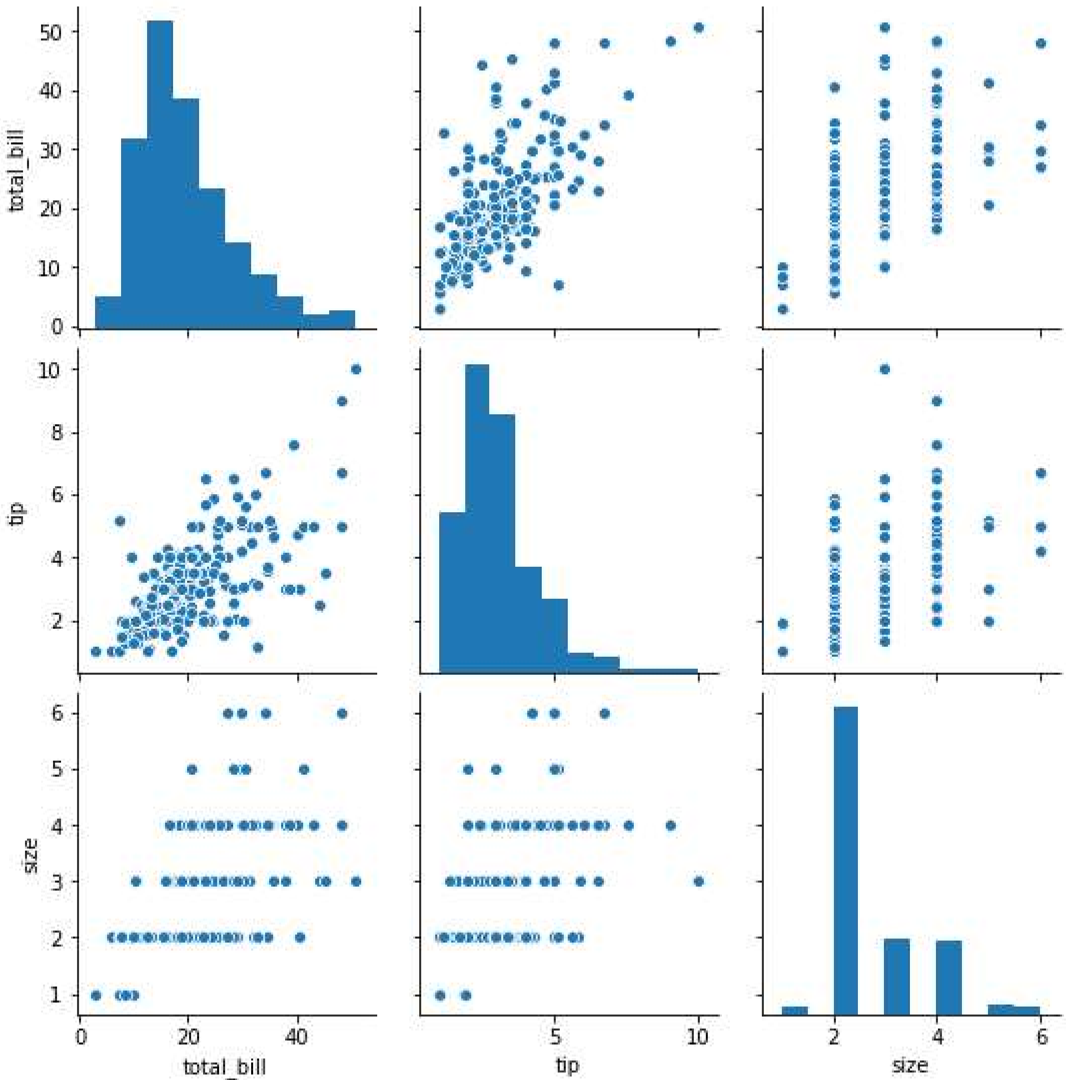

```
In [9]: plt.scatter(tips['tip'], tips ['size'])
```

Out[9]: <matplotlib.collections.PathCollection at 0x29d182d6c08>



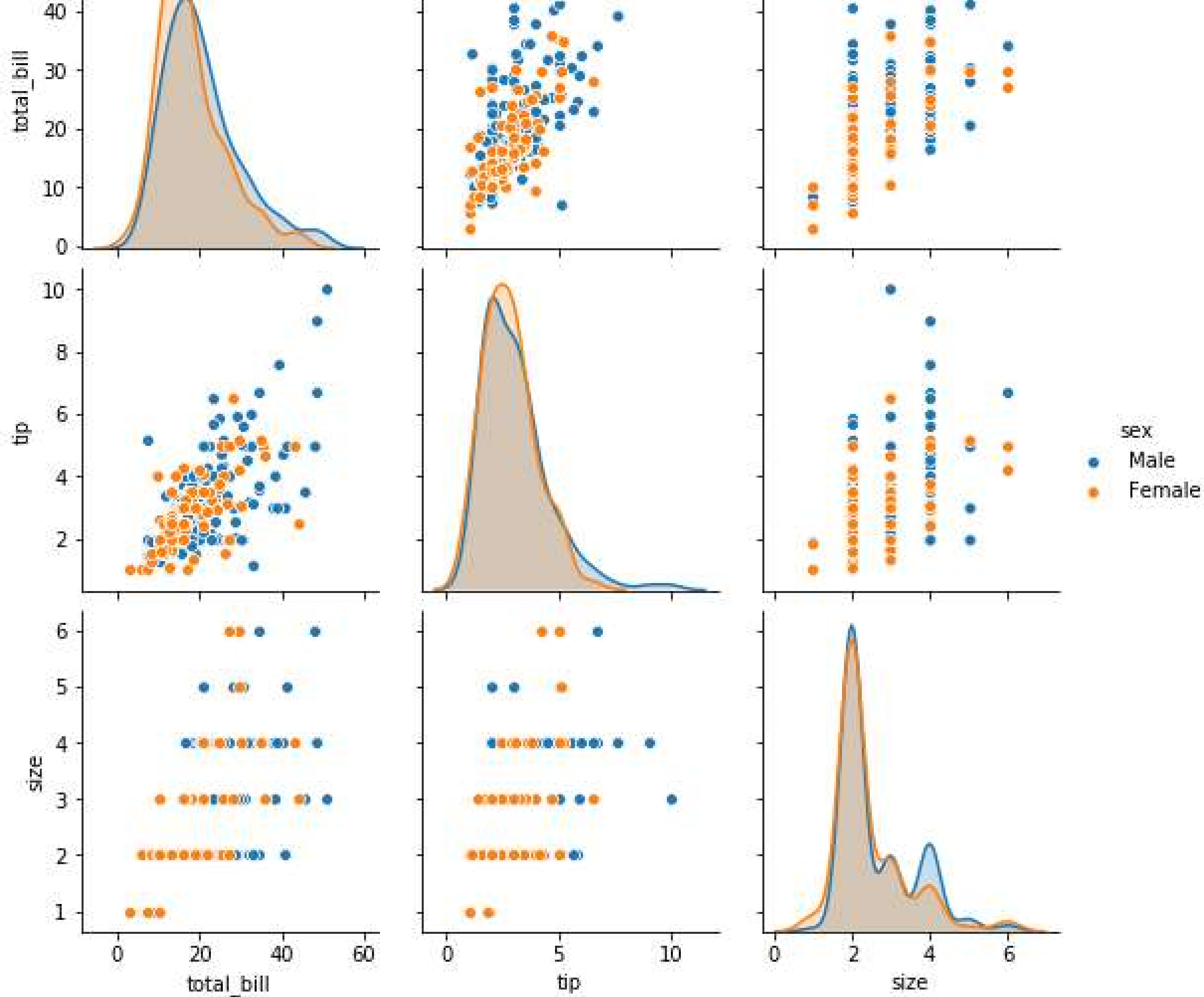
In [10]:
```
sns.pairplot(tips)
```

Out[10]: <seaborn.axisgrid.PairGrid at 0x29d18315ac8>



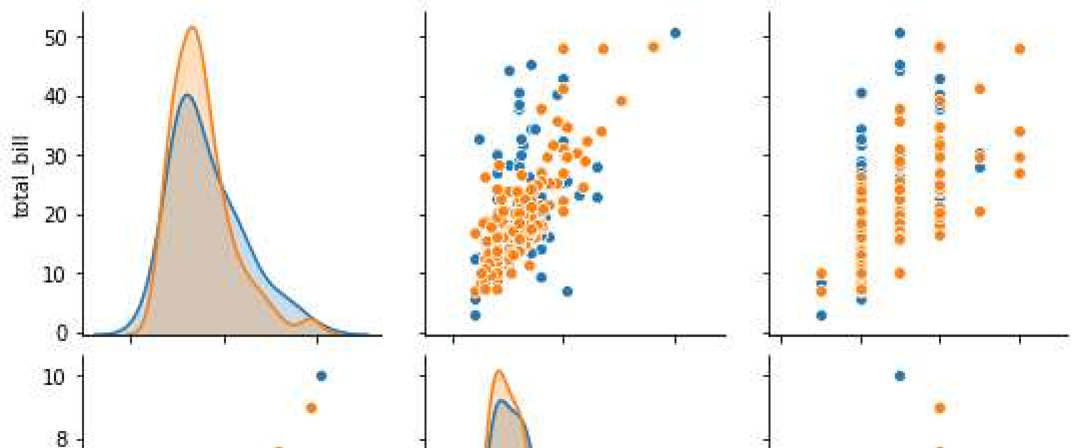
In [11]:
```
sns.pairplot(tips,hue='sex')   # hue is a legend used for splitting categorical data
```

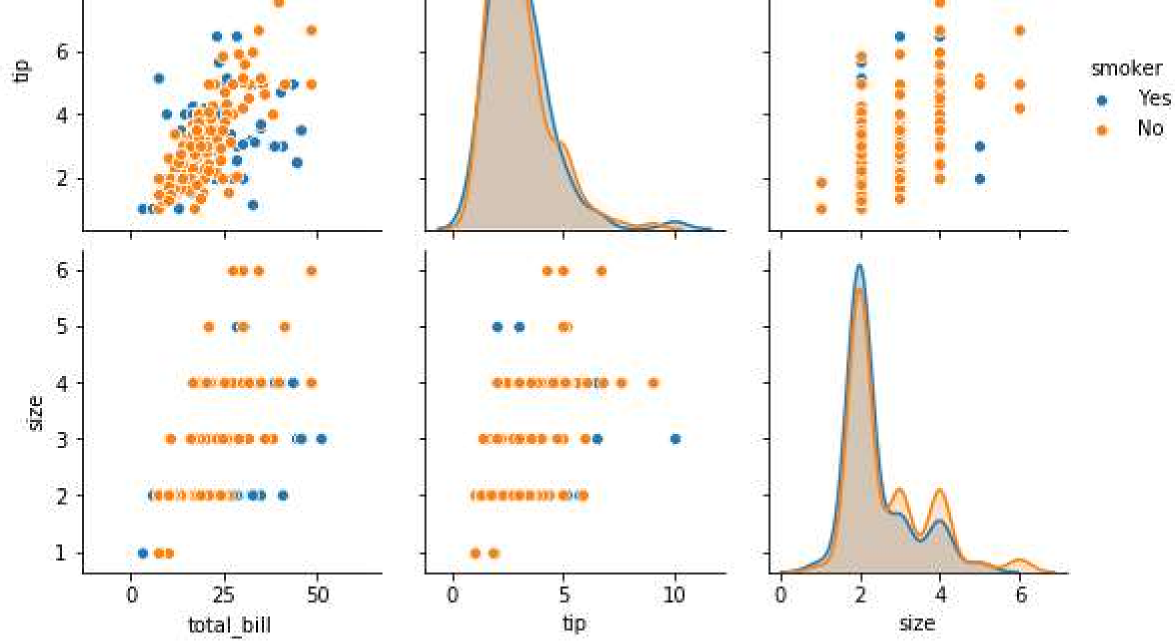
Out[11]: <seaborn.axisgrid.PairGrid at 0x29d188554c8>



In [14]:
```
sns.pairplot(tips,hue='smoker')
```

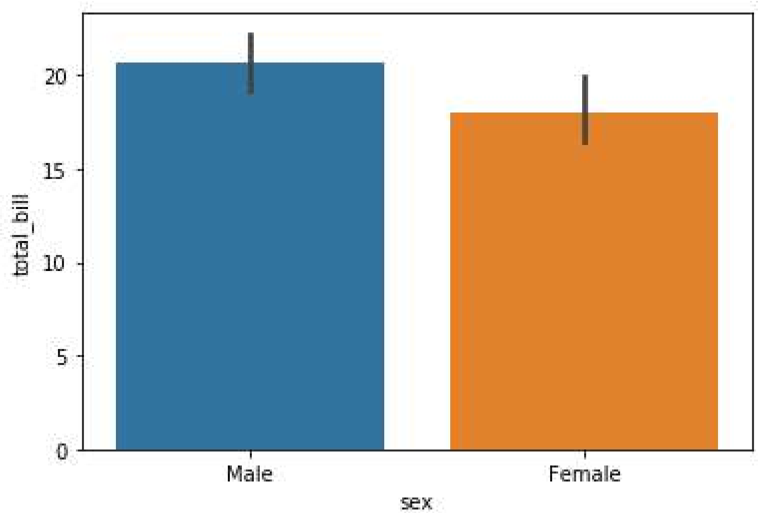
Out[14]: <seaborn.axisgrid.PairGrid at 0x29d19083bc8>



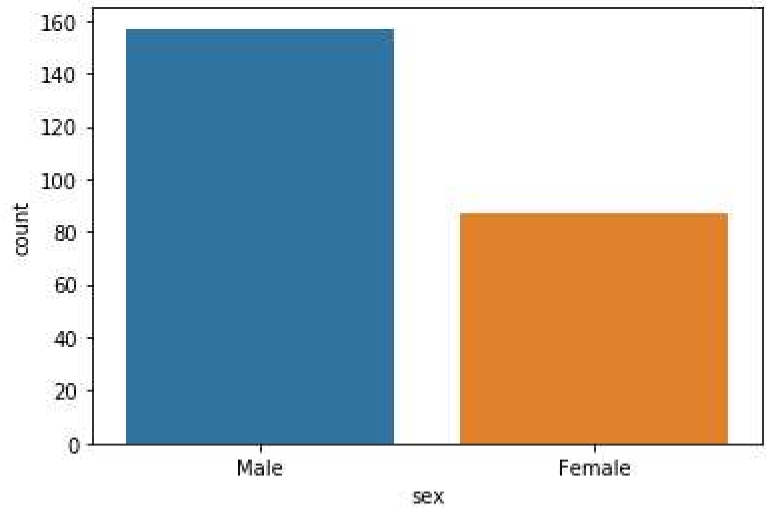


Bar Plot

In [15]:
```
sns.barplot(x='sex', y='total_bill', data=tips)    # black line is average
```
Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x29d1b25afc8>

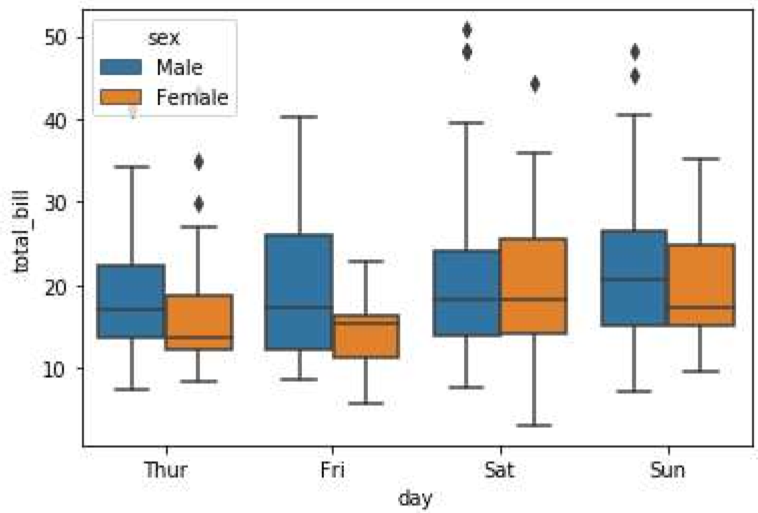


In [17]:
```
sns.countplot(x='sex', data=tips)       # count of categorical variables is done by countplot
```
Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x29d1b68d308>



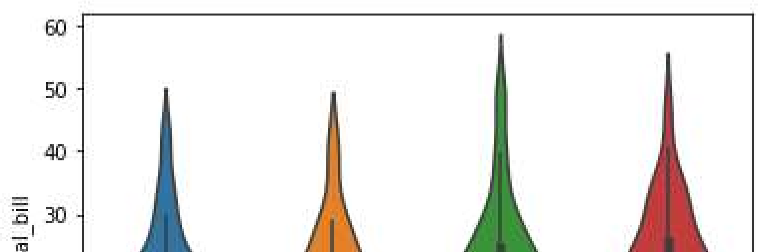
In [22]:
```
sns.boxplot(x='day', y='total_bill', data=tips, hue = 'sex')
```

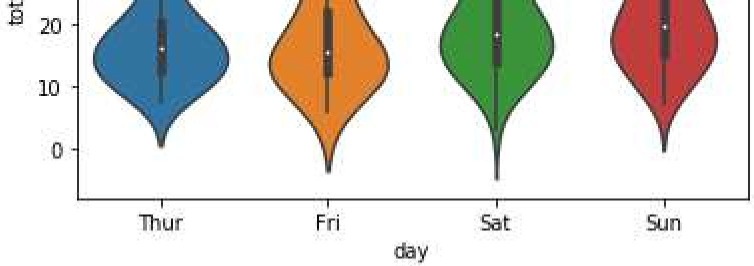
Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x29d1b8d5bc8>



In [20]:
```
sns.violinplot(x='day', y='total_bill', data=tips) #violin plot gives distribution along with boxplot
```

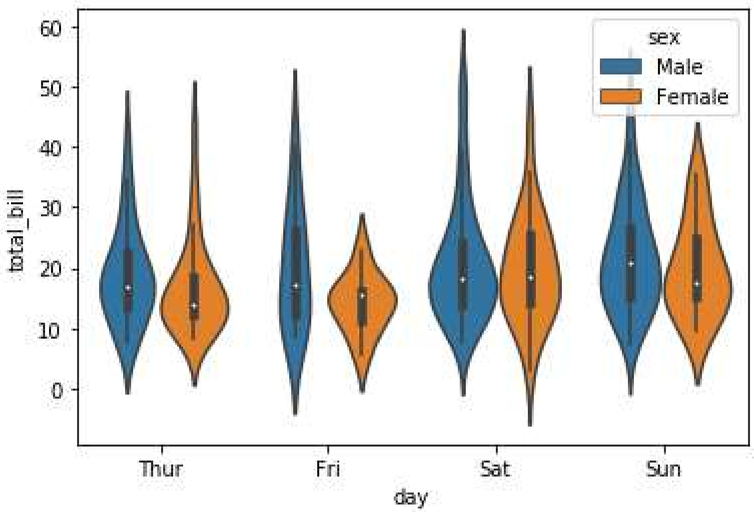
Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x29d1b79ae08>





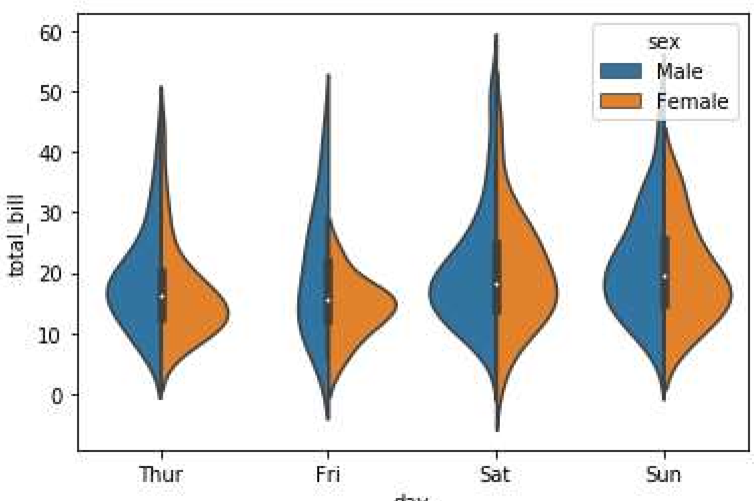

```
In [21]: sns.violinplot(x='day', y='total_bill', data=tips, hue = 'sex')
```

Out[21]: <matplotlib.axes._subplots.AxesSubplot at 0x29d1b167c88>




```
In [23]: sns.violinplot(x='day', y='total_bill', data=tips, hue = 'sex', split = True)
```

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x29d1b9e42c8>



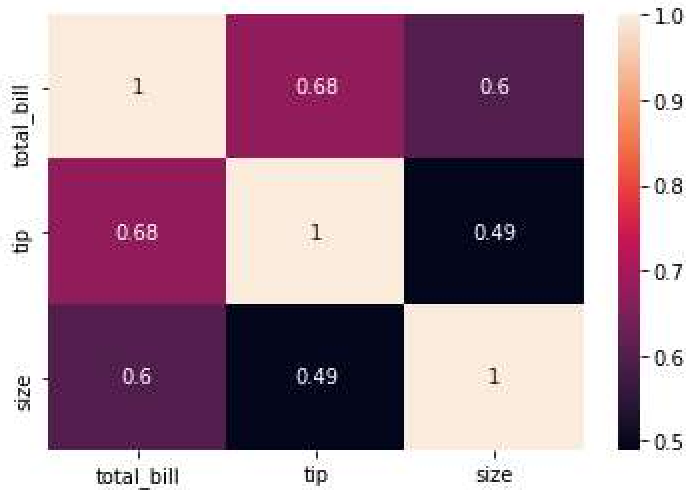
In [26]:

```
h=tips.corr()
```

In [28]:

```
sns.heatmap(h,annot=True)  ##Heat map Works for data which is in metrics format;
```

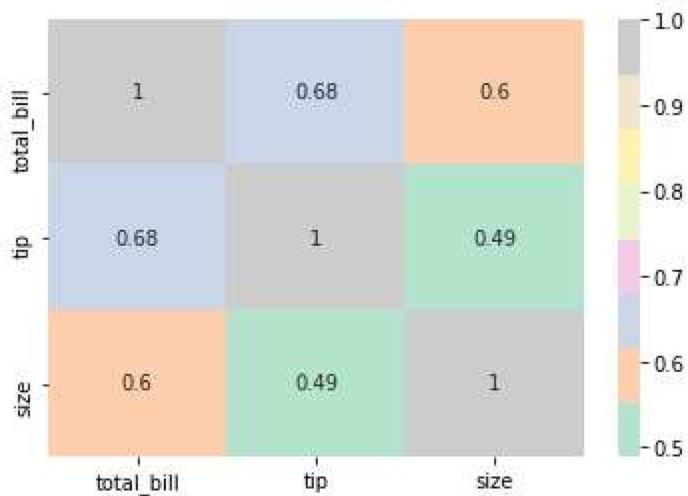
Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x29d19412f08>



In [33]:

```
sns.heatmap(h,annot=True, cmap='Pastel2')
```

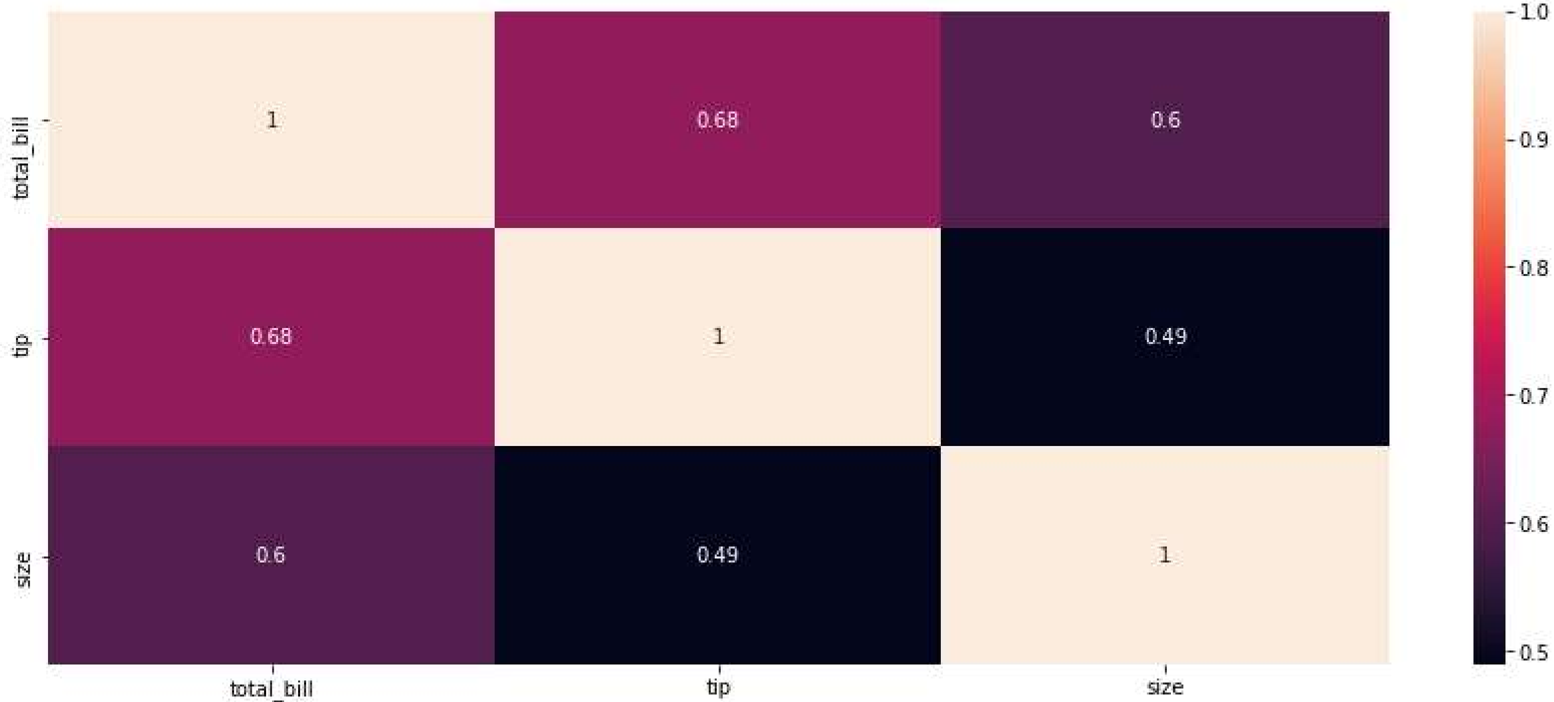
Out[33]: <matplotlib.axes._subplots.AxesSubplot at 0x29d1922cfc8>



In [34]:

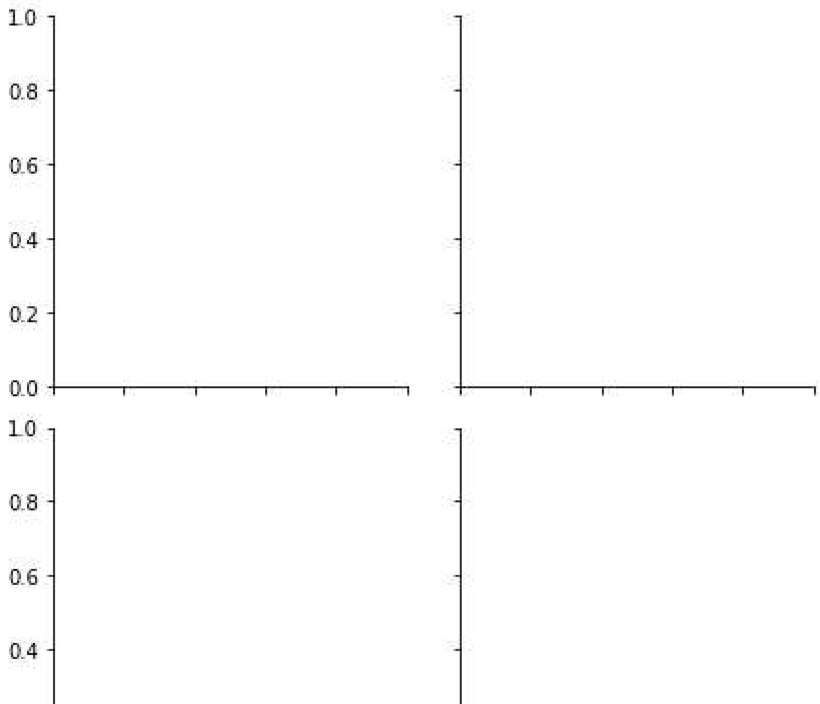
```
plt.figure(figsize=(15,6))
sns.heatmap(data=tips.corr(),annot=True)
```

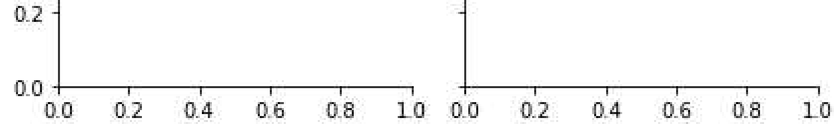
Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x29d18e67288>



In [37]:
```
sns.FacetGrid(data=tips, row='smoker', col='time')
```

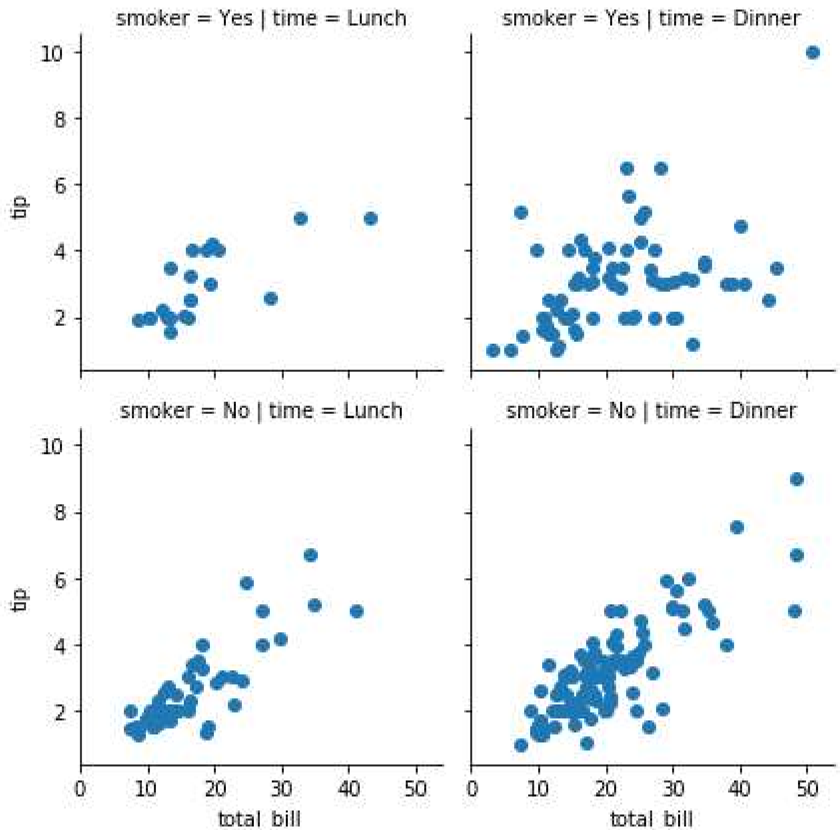
Out[37]: <seaborn.axisgrid.FacetGrid at 0x29d197009c8>





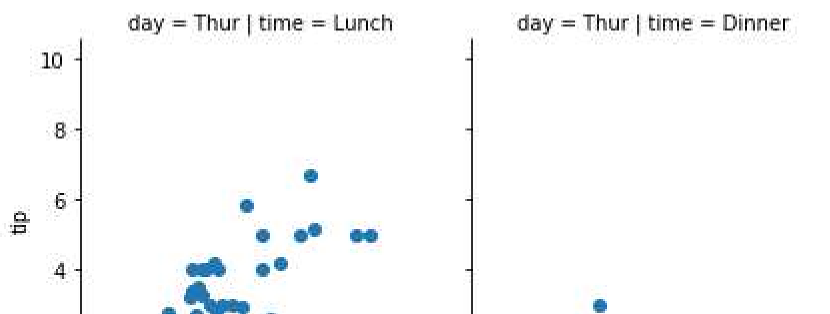
In [38]:
```
sns.FacetGrid(data=tips, row='smoker', col='time').map(plt.scatter,'total_bill','tip')
```

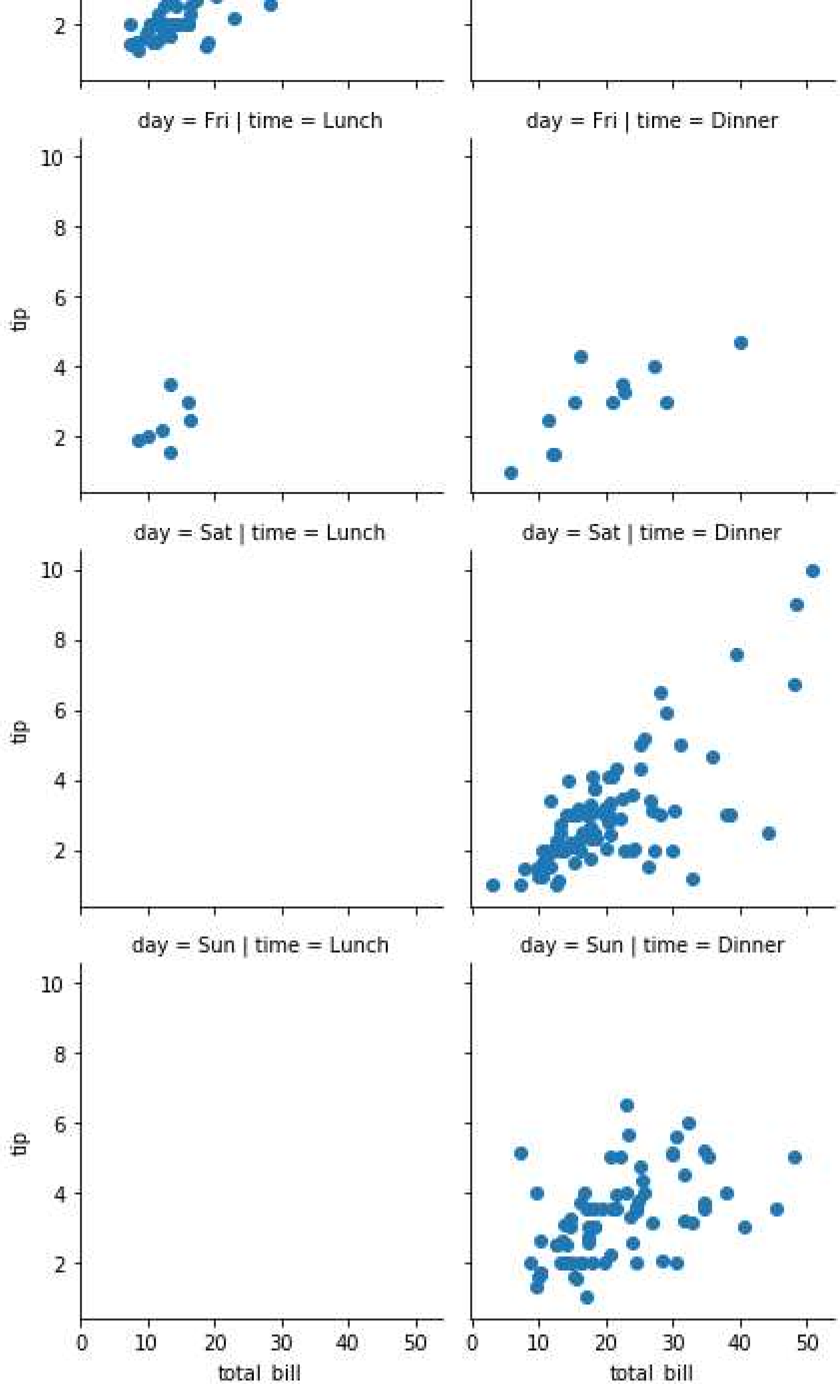
Out[38]: <seaborn.axisgrid.FacetGrid at 0x29d1bbb2888>



In [39]:
```
sns.FacetGrid(data=tips, row='day', col='time').map(plt.scatter,'total_bill','tip')
```

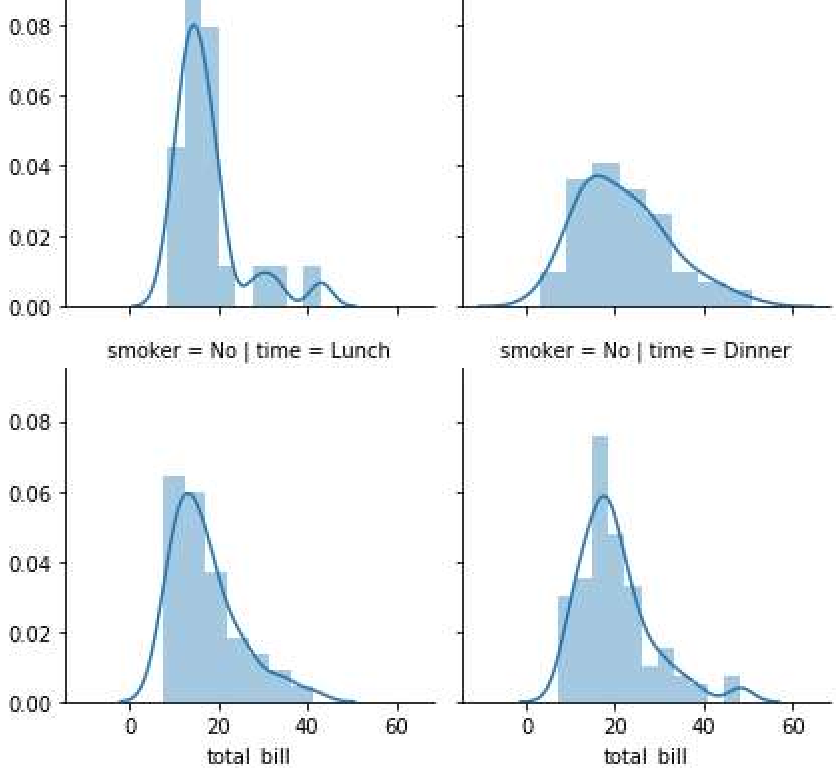
Out[39]: <seaborn.axisgrid.FacetGrid at 0x29d1bd34988>






```
In [42]: sns.FacetGrid(data=tips, row='smoker', col='time').map(sns.distplot,'total_bill')
```

```
Out[42]: <seaborn.axisgrid.FacetGrid at 0x29d19ab4788>
```



EDS - Exploratory Data Analysis

1. Variable Identification
2. Univariate Analysis
3. Biovariate Analysis
4. Missing Value/Data Treatment
5. Outlier Treatment
6. Variable Creation/Creation of dummy variables
7. Variable Transformation

Step 1 - Variable Identification

In [45]:
```
eda=pd.read_excel('EDA_DataSet.xlsx')
```

In [46]:
```
eda
```

Out[46]:

	Name	Age	Gender	Education	Salary	AppraisedValue	Location	Landacres	HouseSizesqrft	Rooms	Baths	Garage

	Name	Age	Gender	Education	Salary	AppraisedValue	Location	Landacres	HouseSizesqrft	Rooms	Baths	Garage
0	Tony	25	M	Grad	50	700.0	GlenCove	0.2297	2448	8.0	3.5	2
1	Harret	52	F	PostGrad	95	364.0	GlenCove	0.2192	1942	7.0	2.5	1
2	Jane	26	F	PostGrad	65	600.0	GlenCove	0.1630	2073	7.0	3.0	2
3	Rose	45	F	Grad	99	548.4	LongBeach	0.4608	2707	8.0	2.5	1
4	John	42	M	Grad	77	405.9	LongBeach	0.2549	2042	NaN	1.5	1
5	Mark	62	M	PostGrad	118	374.1	GlenCove	0.2290	2089	7.0	2.0	0
6	Bruce	51	M	Grad	101	600.0	GlenCove	0.1714	1344	8.0	1.0	0
7	Steve	43	M	Grad	108	299.0	Roslyn	0.1750	1120	5.0	1.5	0
8	Carol	24	F	PostGrad	51	471.0	Roslyn	0.2130	1817	6.0	2.0	0
9	Henry	25	M	PostGrad	68	510.7	Roslyn	0.1377	2496	NaN	2.0	1
10	Donald	41	M	Grad	86	517.7	LongBeach	0.2497	1615	7.0	2.0	1
11	Maria	51	F	Grad	122	1200.0	LongBeach	0.4116	4067	9.0	4.0	1
12	Janet	49	F	PostGrad	112	700.0	Roslyn	0.3372	3130	8.0	3.0	1
13	Sophia	32	F	Grad	85	374.8	Roslyn	0.1503	1423	NaN	2.0	0
14	Jeffery	37	M	Grad	90	543.0	LongBeach	0.2348	1799	6.0	2.5	1

In [47]:
```
eda.head()
```

Out[47]:

	Name	Age	Gender	Education	Salary	AppraisedValue	Location	Landacres	HouseSizesqrft	Rooms	Baths	Garage
0	Tony	25	M	Grad	50	700.0	GlenCove	0.2297	2448	8.0	3.5	2
1	Harret	52	F	PostGrad	95	364.0	GlenCove	0.2192	1942	7.0	2.5	1
2	Jane	26	F	PostGrad	65	600.0	GlenCove	0.1630	2073	7.0	3.0	2
3	Rose	45	F	Grad	99	548.4	LongBeach	0.4608	2707	8.0	2.5	1
4	John	42	M	Grad	77	405.9	LongBeach	0.2549	2042	NaN	1.5	1

In [48]:
```
eda.tail()
```

Out[48]:

	Name	Age	Gender	Education	Salary	AppraisedValue	Location	Landacres	HouseSizesqrft	Rooms	Baths	Garage
10	Donald	41	M	Grad	86	517.7	LongBeach	0.2497	1615	7.0	2.0	1
11	Maria	51	F	Grad	122	1200.0	LongBeach	0.4116	4067	9.0	4.0	1
12	Janet	49	F	PostGrad	112	700.0	Roslyn	0.3372	3130	8.0	3.0	1
13	Sophia	32	F	Grad	85	374.8	Roslyn	0.1503	1423	NaN	2.0	0

14	Jeffery	37	M	Grad	90	543.0	LongBeach	0.2348	1799	6.0	2.5	1

In [49]:
```
eda.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Name            15 non-null     object
 1   Age             15 non-null     int64
 2   Gender          15 non-null     object
 3   Education       15 non-null     object
 4   Salary          15 non-null     int64
 5   AppraisedValue  15 non-null     float64
 6   Location        15 non-null     object
 7   Landacres       15 non-null     float64
 8   HouseSizesqrft  15 non-null     int64
 9   Rooms           12 non-null     float64
 10  Baths           15 non-null     float64
 11  Garage          15 non-null     int64
dtypes: float64(4), int64(4), object(4)
memory usage: 1.5+ KB
```

In [51]:
```
eda.describe()
```

Out[51]:

	Age	Salary	AppraisedValue	Landacres	HouseSizesqrft	Rooms	Baths	Garage
count	15.000000	15.000000	15.000000	15.000000	15.000000	12.000000	15.000000	15.000000
mean	40.333333	88.466667	547.240000	0.242487	2140.800000	7.166667	2.333333	0.800000
std	11.842217	22.752917	217.331829	0.093602	754.829517	1.114641	0.794325	0.676123
min	24.000000	50.000000	299.000000	0.137700	1120.000000	5.000000	1.000000	0.000000
25%	29.000000	72.500000	390.350000	0.173200	1707.000000	6.750000	2.000000	0.000000
50%	42.000000	90.000000	517.700000	0.229000	2042.000000	7.000000	2.000000	1.000000
75%	50.000000	104.500000	600.000000	0.252300	2472.000000	8.000000	2.750000	1.000000
max	62.000000	122.000000	1200.000000	0.460800	4067.000000	9.000000	4.000000	2.000000

3 steps

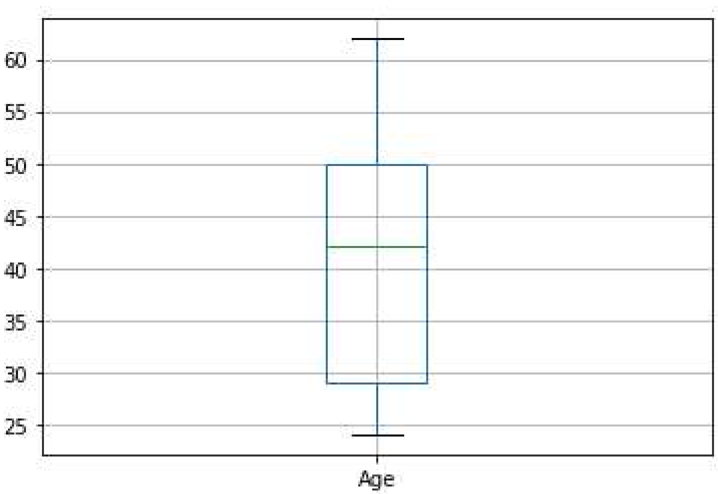
1. Identify variable (dependent (apprised value) & independent (rest all))
2. Broader data types (numeric & categorical) - Name, Gender, Education, Location are categorical and rest all are numberic

3. Python data types using info () command

Step - 2: Univariate Analysis - Pick each individual variable and study it and identify the insights from it

In [52]:
```
eda.boxplot('Age')
```

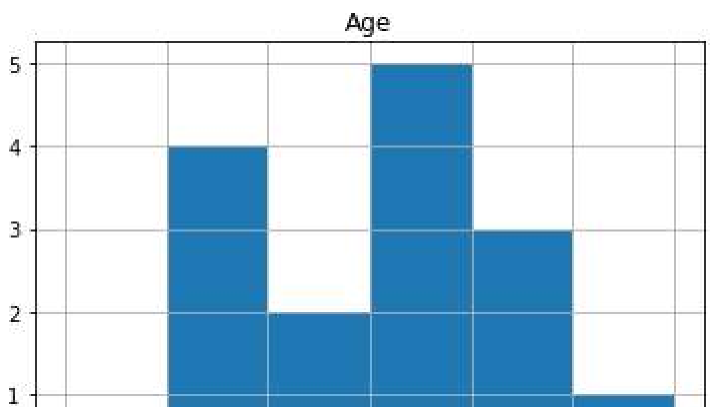
Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0x29d1dadde88>

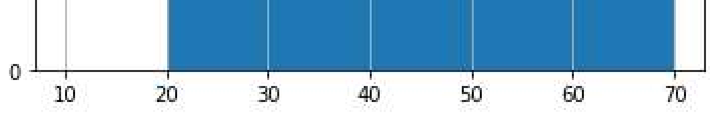


Observation - No outliers in variable 'age'

In [57]:
```
eda.hist('Age', bins=[10,20,30,40,50,60,70])
```

Out[57]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x0000029D1DD1FA88>]],
      dtype=object)
```

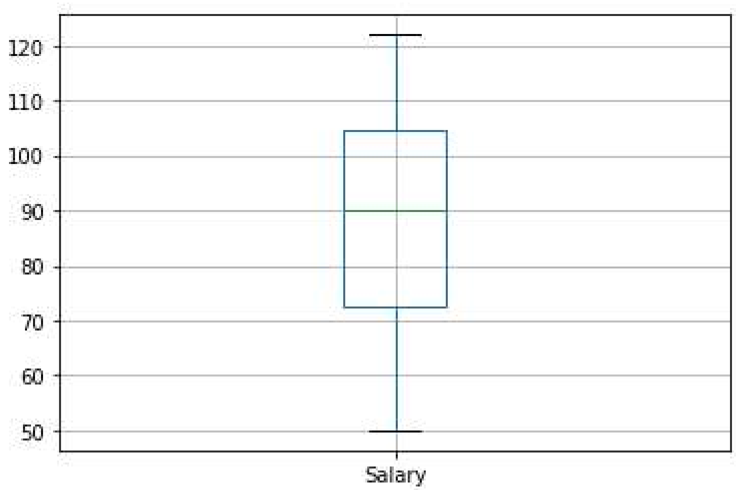




Observation - Most of the people are in between age 40 & 50

In [58]:
```
eda.boxplot('Salary')
```

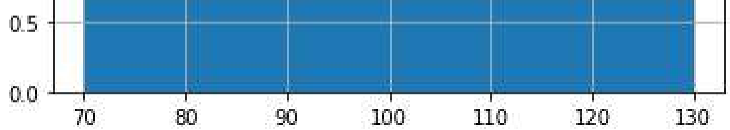
Out[58]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x29d1dd9cb48>
```



Observation - No outliers

In [63]:
```
eda.hist('Salary', bins=[70,80,90,100,110,120,130])
```

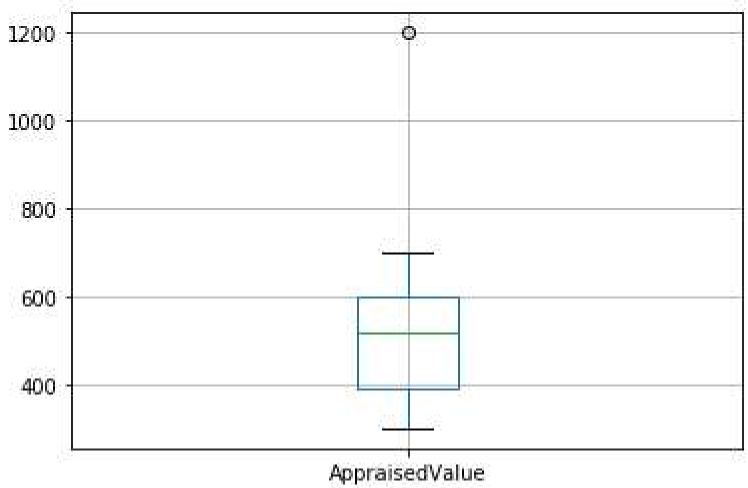
Out[63]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x0000029D1E00D608>]],
      dtype=object)
```



Observation - Highest salary in the range of 90 & 100

In [61]:
```
eda.boxplot('AppraisedValue')
```

Out[61]: <matplotlib.axes._subplots.AxesSubplot at 0x29d1df26708>



Observation - 1 outlier