Seaborn

When ever we get Dataset into ML and DL we divide our features into independent and dependent features

Dataset f1 f2 f3 -> f4(o/p)

(f1) when ever we are comparing two features is called Univarient Analysis

(f1,f2) Bivariant

Distribution plots

** displot--> ** joinplot--> ** pairplot-->

In [2]:

```
import seaborn as sns
```

In [3]:

```
print(sns.__version__)
```

```
0.11.2
```

In [5]:

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

In [6]:

```
df.head()
```

Out[6]:

	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 [7]:

```
df.dtypes
```

Out[7]:

```
total_bill     float64
tip            float64
sex           category
smoker        category
day           category
time          category
size             int64
dtype: object
```

Correlation with Heatmap

A correlation heatmap uses colored cells, to show a 2D correlation matrix(table) between two discrete dimensions or event types. it is very important in Feature Selection

In [5]:

```
df.corr()
```

Out[5]:

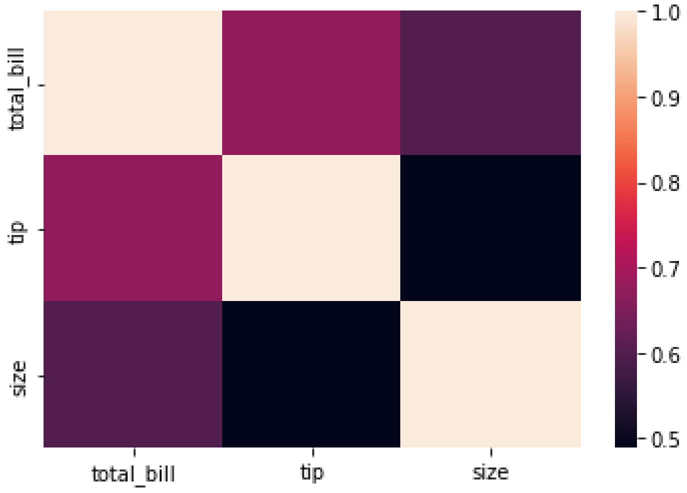
	total_bill	tip	size
total_bill	1.000000	0.675734	0.598315
tip	0.675734	1.000000	0.489299
size	0.598315	0.489299	1.000000

In [6]:

```
sns.heatmap(df.corr())
```

Out[6]:

```
<AxesSubplot:>
```



Join Plot

A join plot allows to study the relationship between 2 numerica variables. The centeral chart display their correction, it is usually a scatterplot, ahexbin plot, a 20 histogram or a 20 density plot.

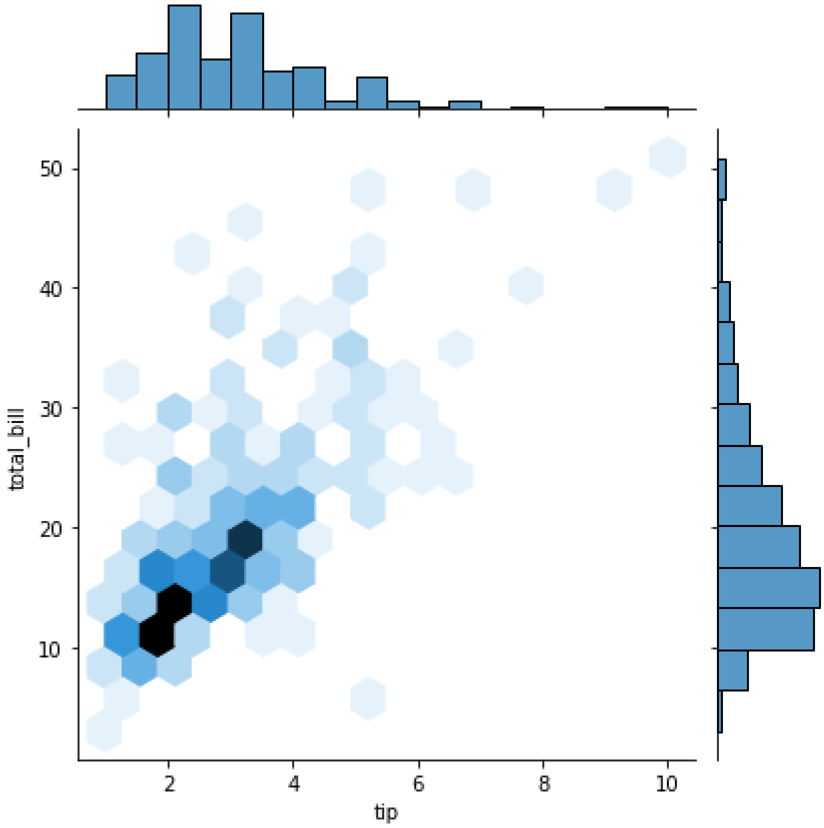
Univariate Analysis

In [7]:

```
sns.jointplot(x='tip',y='total_bill',data=df,kind='hex')
```

Out[7]:

```
<seaborn.axisgrid.JointGrid at 0x27e0a841a58>
```

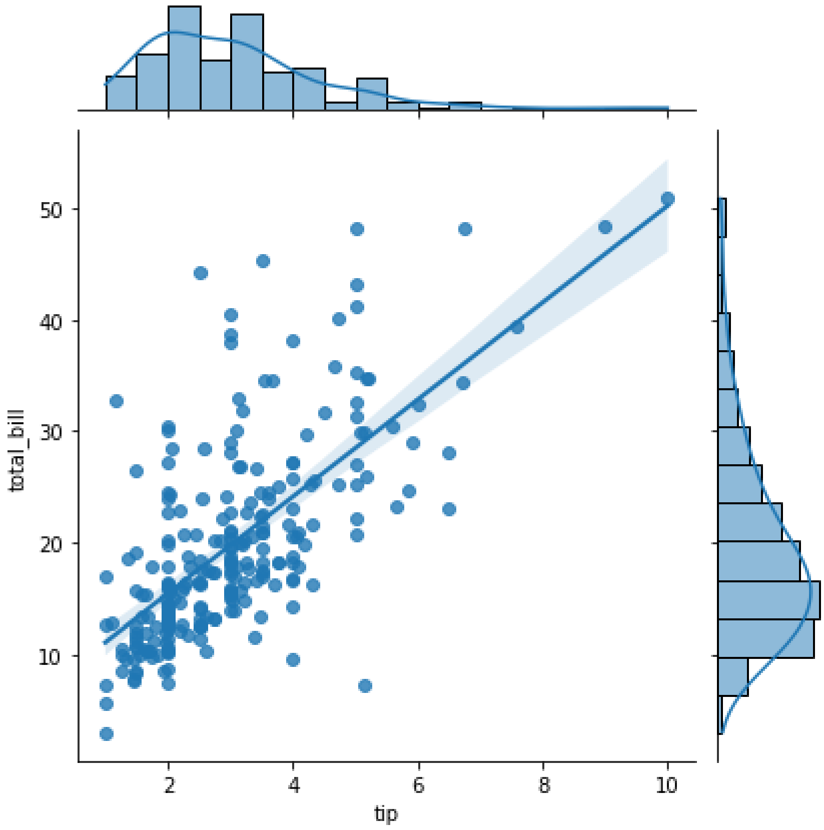


In [20]:

```
sns.jointplot(x='tip',y='total_bill',data=df,kind='reg')
```

Out[20]:

```
<seaborn.axisgrid.JointGrid at 0x1ad399e2d30>
```



Pair Plot

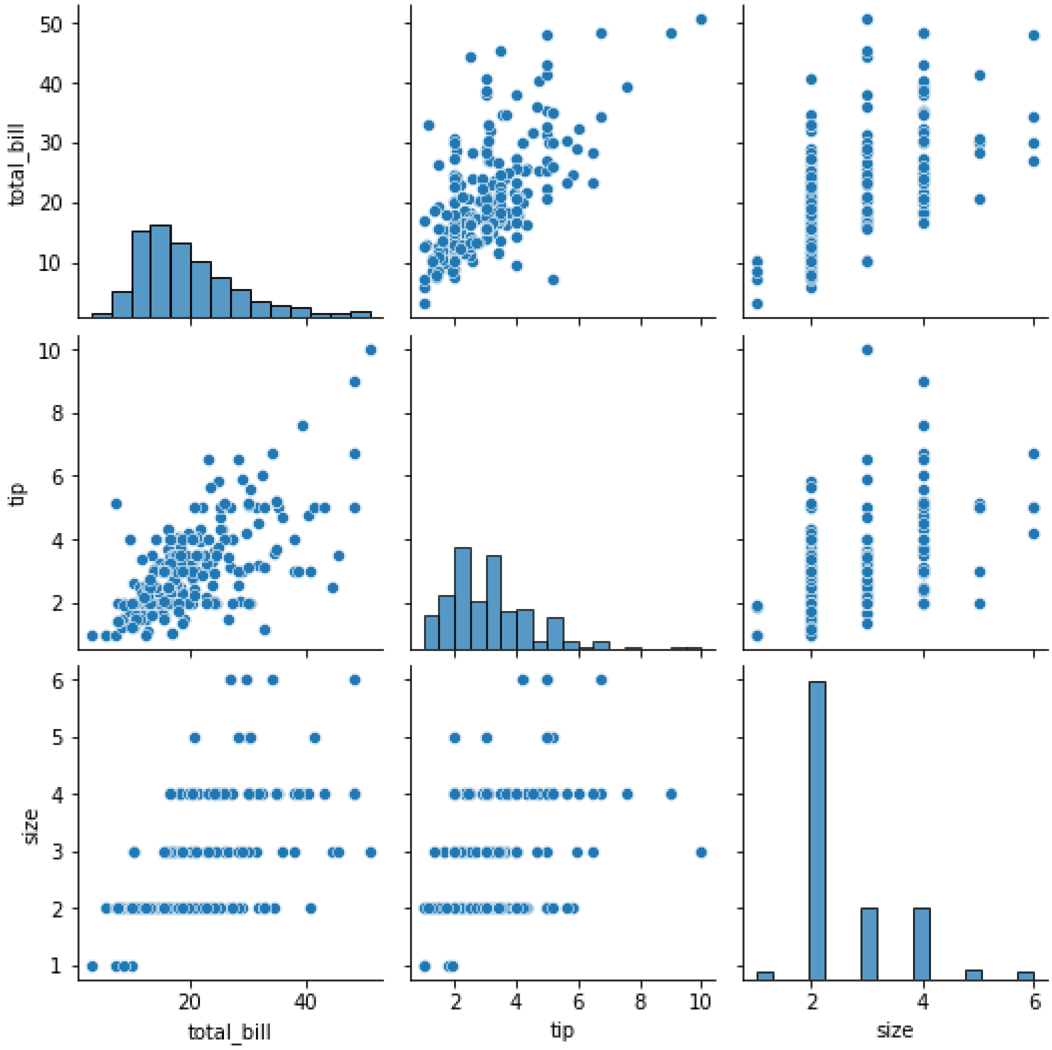
A 'Pairs plot' is also known as a scatterplot, in which one variable in the same data row is matched with another variables value, like this: Pairs points are just elaborations on this, showing at variables paired with all the other variables

In [8]:

```
sns.pairplot(df)
```

Out[8]:

```
<seaborn.axisgrid.PairGrid at 0x27e0aa95978>
```

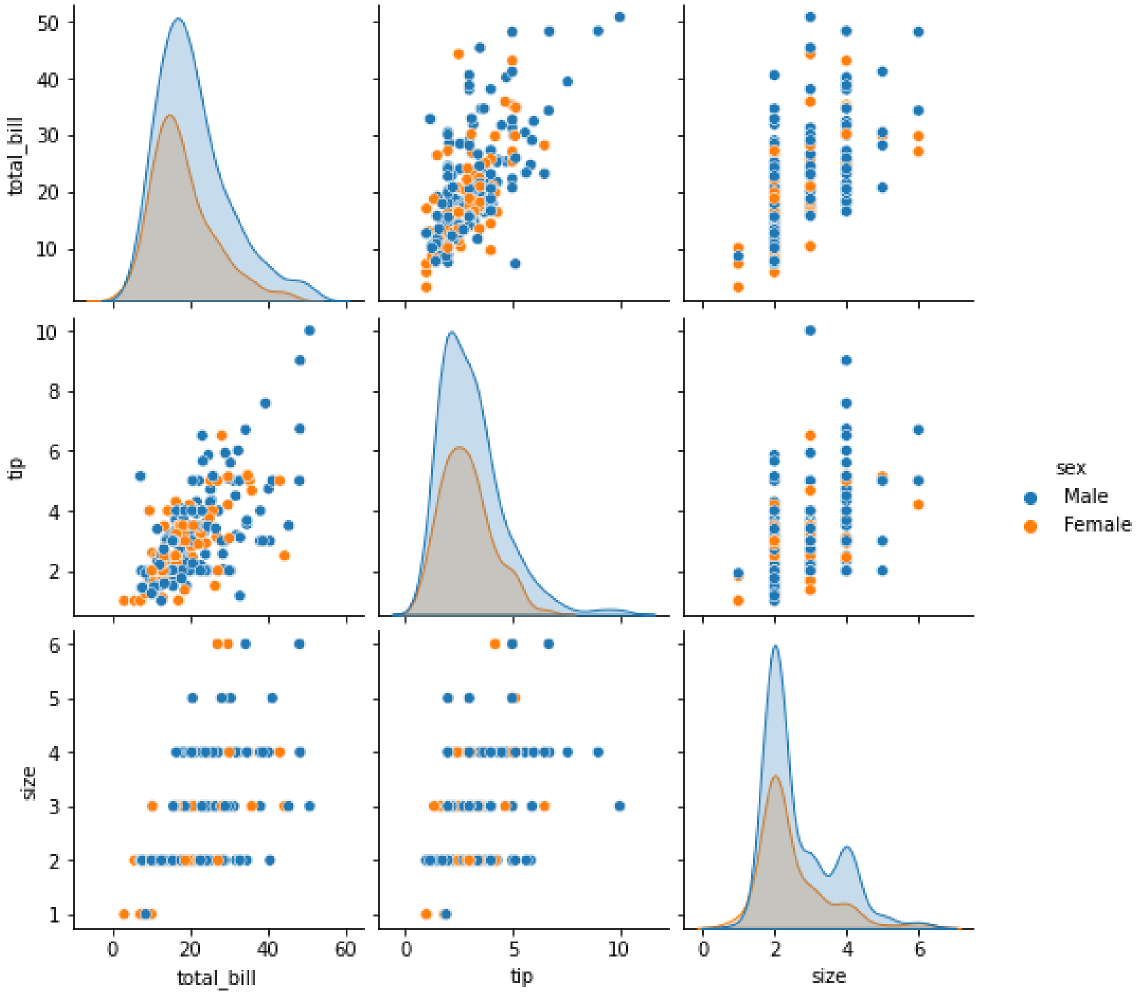


In [22]:

```
sns.pairplot(df,hue='sex')
```

Out[22]:

```
<seaborn.axisgrid.PairGrid at 0x1ad3a0c2e80>
```



In [26]:

```
df['smoker'].value_counts()
```

Out[26]:

```
No     151
Yes     93
Name: smoker, dtype: int64
```

Dist plot

Dist plot helps us to check the distributions of the columns features

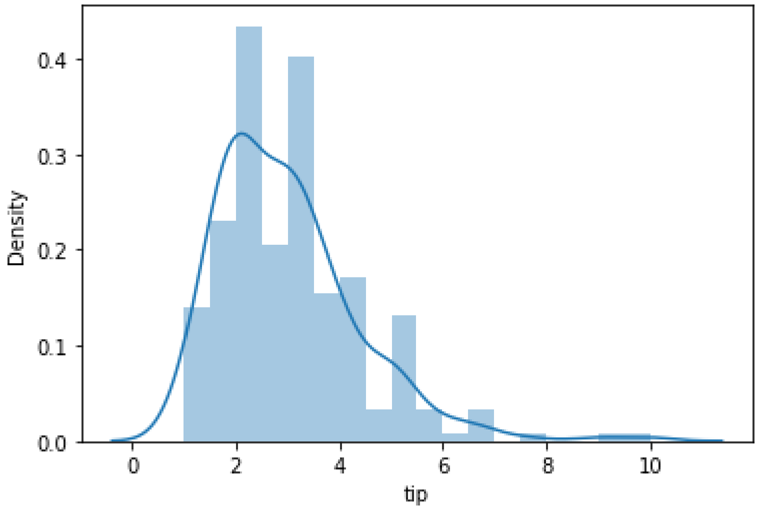
In [9]:

```
sns.distplot(df['tip'])
```

```
c:\users\dell\appdata\local\programs\python\python36\lib\site-packages\sea
born\distributions.py:2619: FutureWarning: `distplot` is a deprecated func
tion and will be removed in a future version. Please adapt your code to us
e either `displot` (a figure-level function with similar flexibility) or `
histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[9]:

```
<AxesSubplot:xlabel='tip', ylabel='Density'>
```

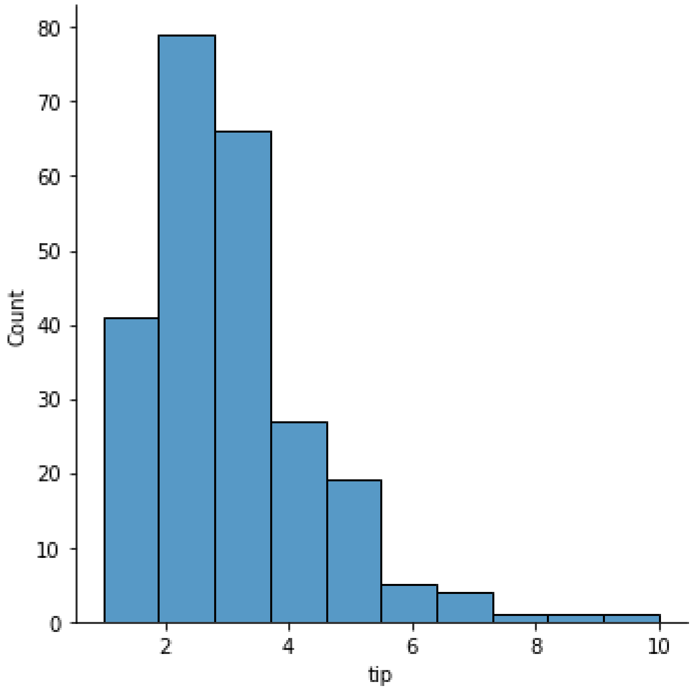


In [28]:

```
sns.displot(df['tip'],kde=False,bins=10)
```

Out[28]:

```
<seaborn.axisgrid.FacetGrid at 0x1ad3bcb9160>
```



Categorical Plots

Seaborn also head us in doing the analysis on Categorical Data points.

- boxplot
- viollnplot
- countplot
- bar plot

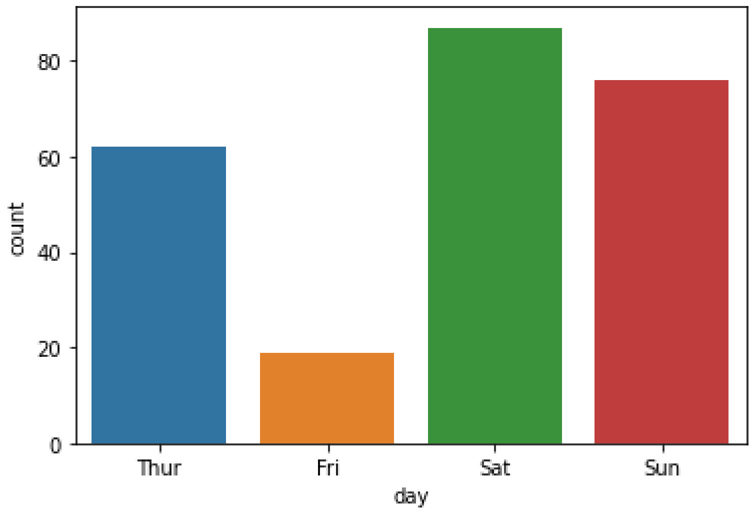
In [32]:

```
## count plot
sns.countplot('day',data=df)

```

Out[32]:

```
<AxesSubplot:xlabel='day', ylabel='count'>
```

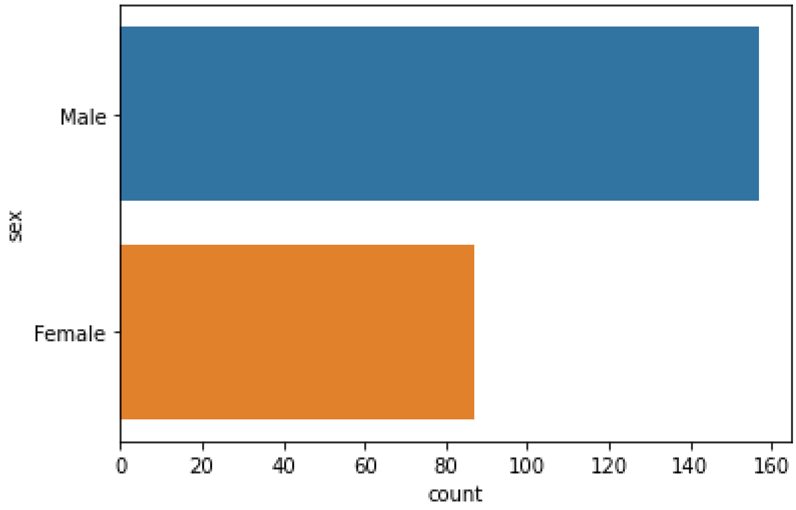


In [33]:

```
sns.countplot(y='sex',data=df)
```

Out[33]:

```
<AxesSubplot:xlabel='count', ylabel='sex'>
```

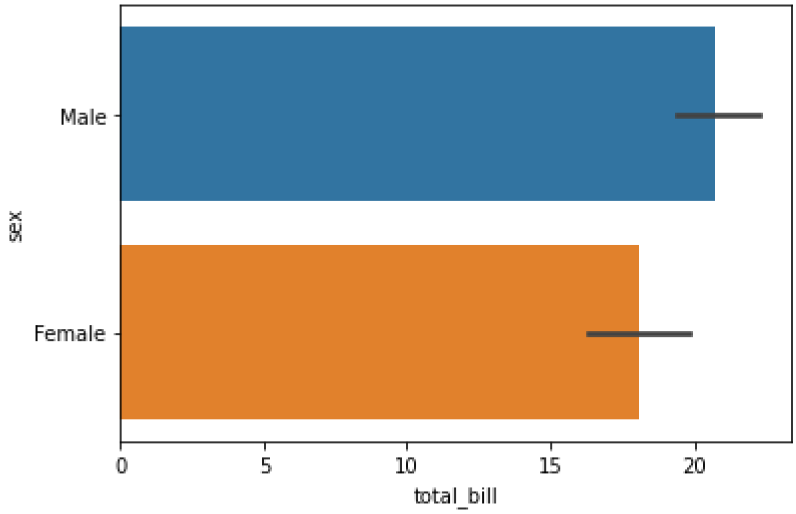


In [35]:

```
## Bar plot
sns.barplot(x='total_bill',y='sex',data=df)
```

Out[35]:

```
<AxesSubplot:xlabel='total_bill', ylabel='sex'>
```



In [36]:

```
df.head()
```

Out[36]:

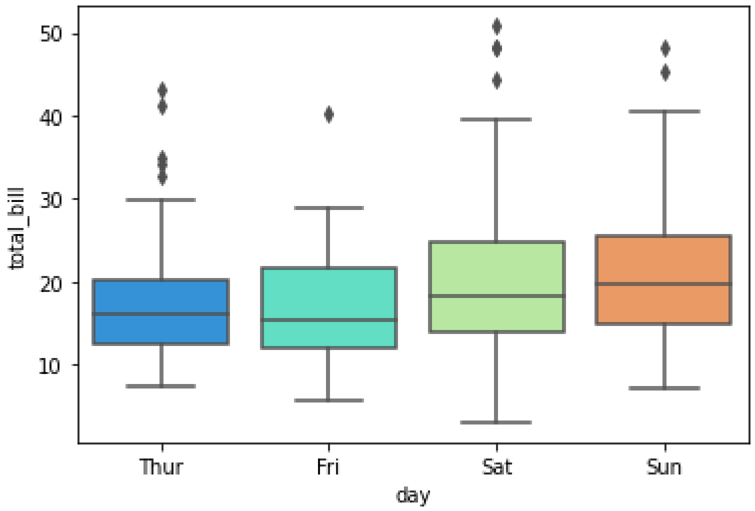
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

In [37]:

```
sns.boxplot(x='day',y='total_bill',data=df,palette='rainbow')
```

Out[37]:

```
<AxesSubplot:xlabel='day', ylabel='total_bill'>
```

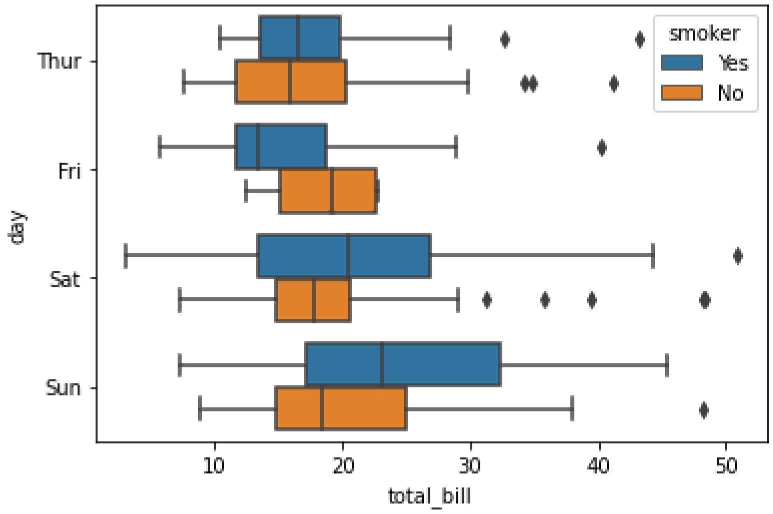


In [40]:

```
sns.boxplot(x='total_bill',y='day', hue='smoker', data=df)
```

Out[40]:

```
<AxesSubplot:xlabel='total_bill', ylabel='day'>
```



Viollin Plot

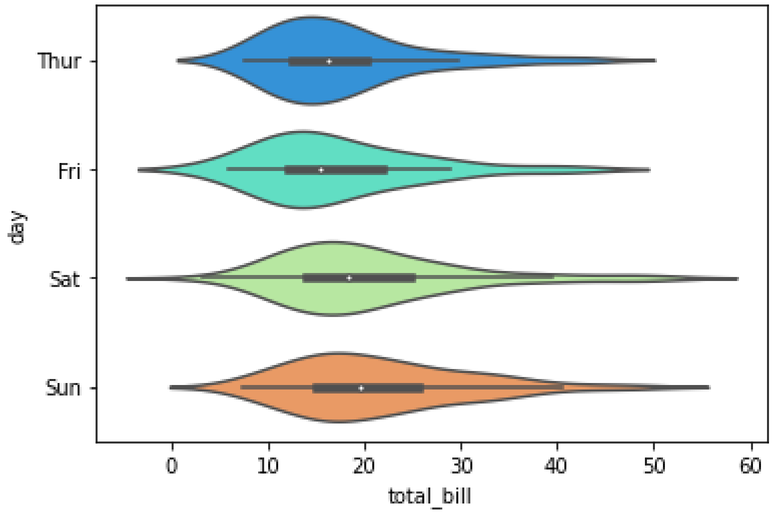
Violin plot helps us to see the both the distribution of data in terms of kernel density estimation and the box plot

In [41]:

```
sns.violinplot(x='total_bill',y='day', data=df,palette='rainbow')
```

Out[41]:

```
<AxesSubplot:xlabel='total_bill', ylabel='day'>
```



In [ ]:

In [ ]:

In [ ]: