In [1]:
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.style.use('ggplot')
```

Categorical Plots

Categorical Scatter Plot

- Stripplot
- Swarmplot

Categorical Distribution Plots

- Boxplot
- Violinplot

Categorical Estimate Plot -> for central tendency

- Barplot
- Pointplot
- Countplot

Figure level function -> `catplot`

In [2]:
```
# import datasets

tips = sns.load_dataset('tips')
iris = sns.load_dataset('iris')
```

In [3]:
```
tips.sample(2)
```

Out[3]:

	total_bill	tip	sex	smoker	day	time	size
94	22.75	3.25	Female	No	Fri	Dinner	2
88	24.71	5.85	Male	No	Thur	Lunch	2

In [4]:
```
iris.sample(2)
```

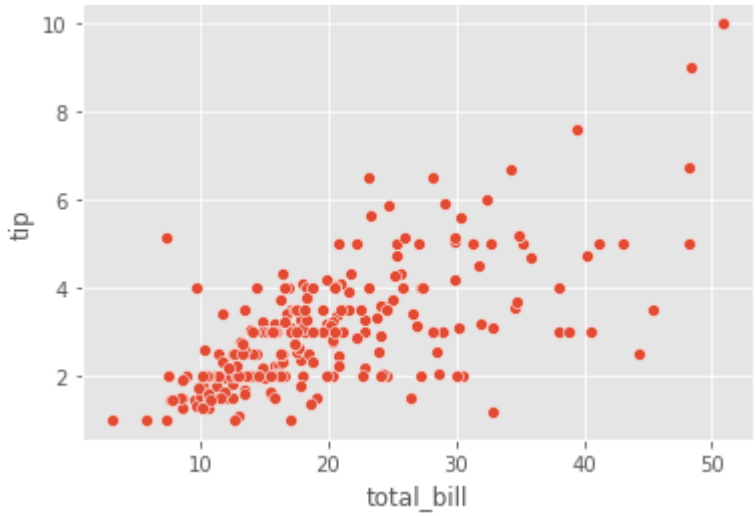
Out[4]:

	sepal_length	sepal_width	petal_length	petal_width	species
53	5.5	2.3	4.0	1.3	versicolor
103	6.3	2.9	5.6	1.8	virginica

Categorical Scatter Plots

In [5]:
```
# ON Numerical data

sns.scatterplot(data=tips, x='total_bill',y='tip')
```

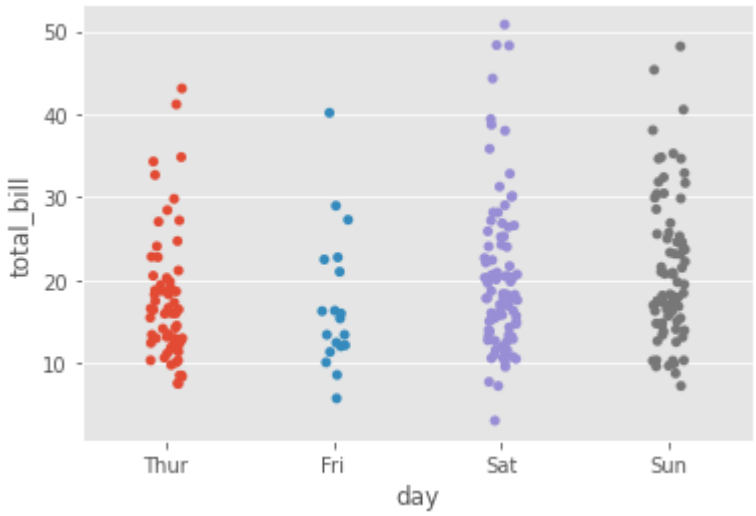
Out[5]:
```
<AxesSubplot:xlabel='total_bill', ylabel='tip'>
```



1.strip plot

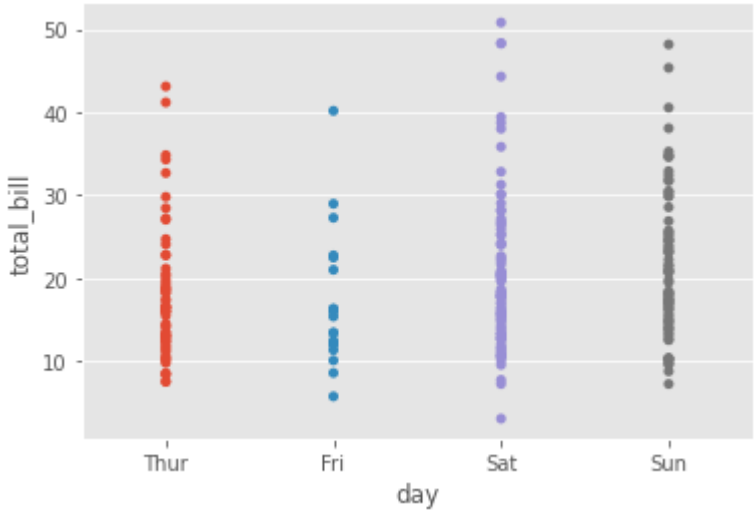
In [6]:
```
# axes level function

# On categorical data

sns.stripplot(data=tips,x='day',y='total_bill')
```

Out[6]: <AxesSubplot:xlabel='day', ylabel='total_bill'>



In [7]:
```
# jitter

sns.stripplot(data=tips,x='day',y='total_bill', jitter=False)
```

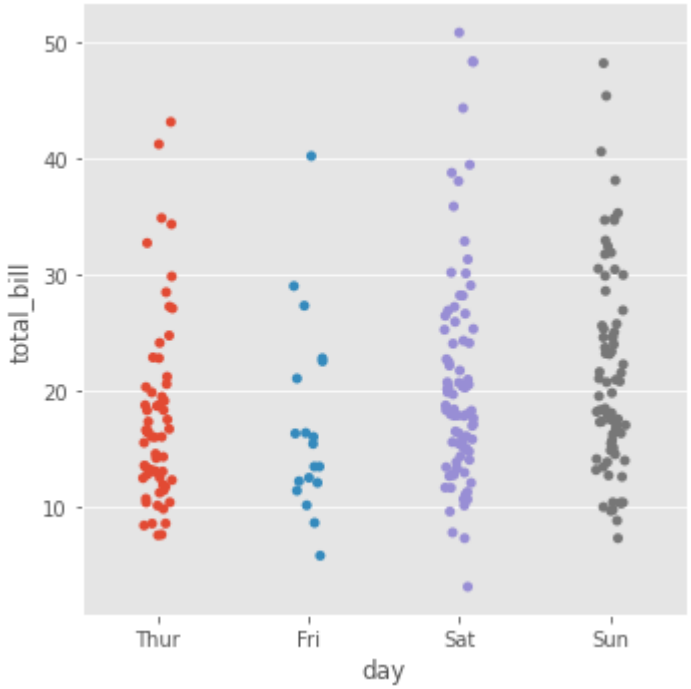
Out[7]: <AxesSubplot:xlabel='day', ylabel='total_bill'>



catplot

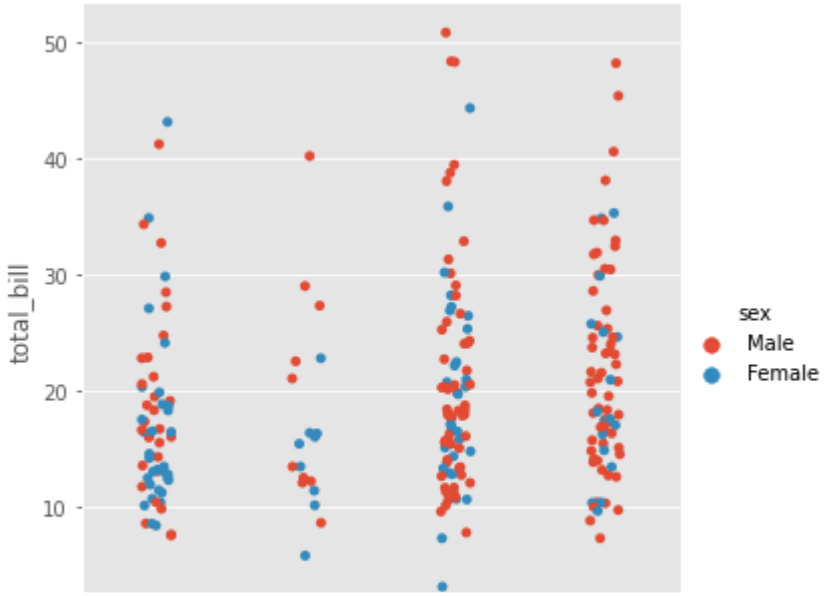
In [8]:
```
# catplot -> Categorical plot

# Figure Level functions

sns.catplot(data=tips,x='day',y='total_bill',kind = 'strip')
```

Out[8]:
```
<seaborn.axisgrid.FacetGrid at 0x1e85ea0ed90>
```



In [9]:
```
# hue

sns.catplot(data=tips,x='day',y='total_bill',kind = 'strip',
            hue ='sex')
```

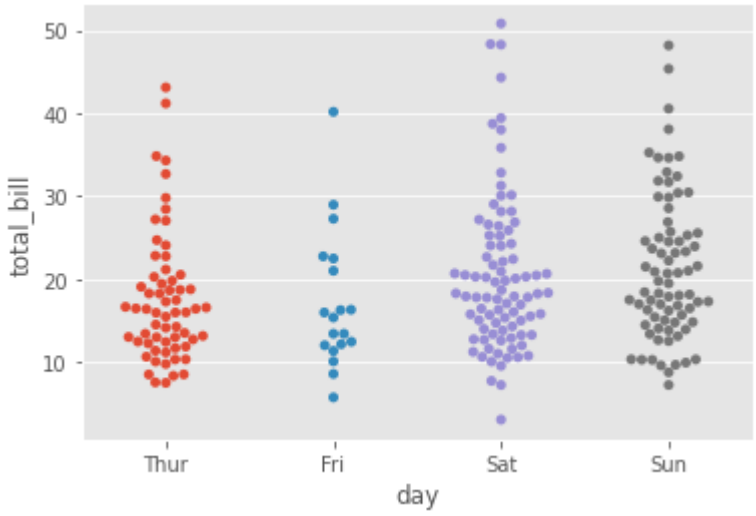
Out[9]: <seaborn.axisgrid.FacetGrid at 0x1e85f320e50>



2 .Swarm plot

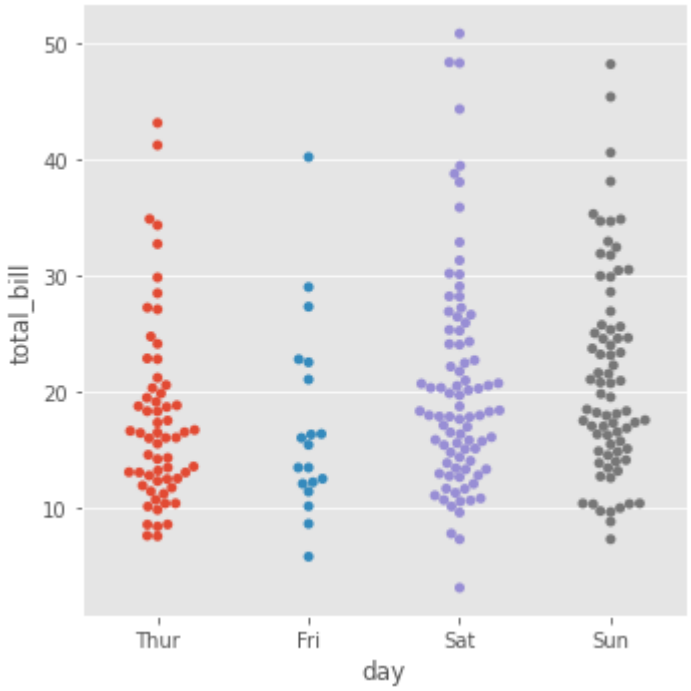
In [10]:
```
sns.swarmplot(data=tips,x ='day', y='total_bill')
```

Out[10]: <AxesSubplot:xlabel='day', ylabel='total_bill'>



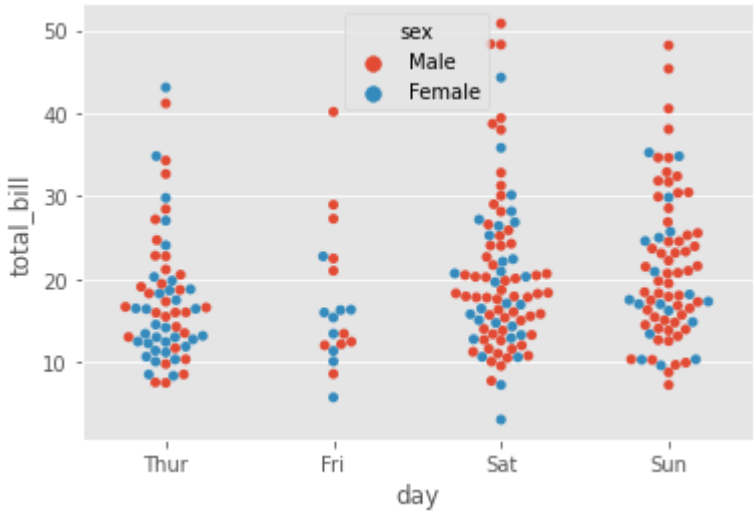
In [11]:
```
# use by catplot

sns.catplot(data=tips,x ='day', y='total_bill',kind ='swarm')
```

Out[11]:
```
<seaborn.axisgrid.FacetGrid at 0x1e85f433850>
```



In [12]:
```
# hue

sns.swarmplot(data=tips,x ='day', y='total_bill',hue='sex')
```

Out[12]:
```
<AxesSubplot:xlabel='day', ylabel='total_bill'>
```

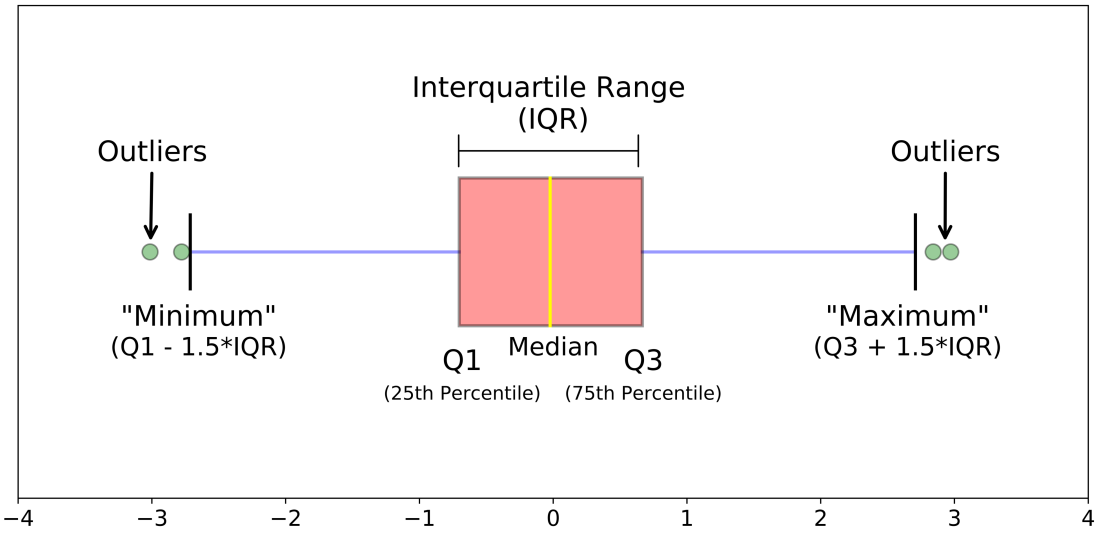


Categorical Distribution Plots

1. Boxplot

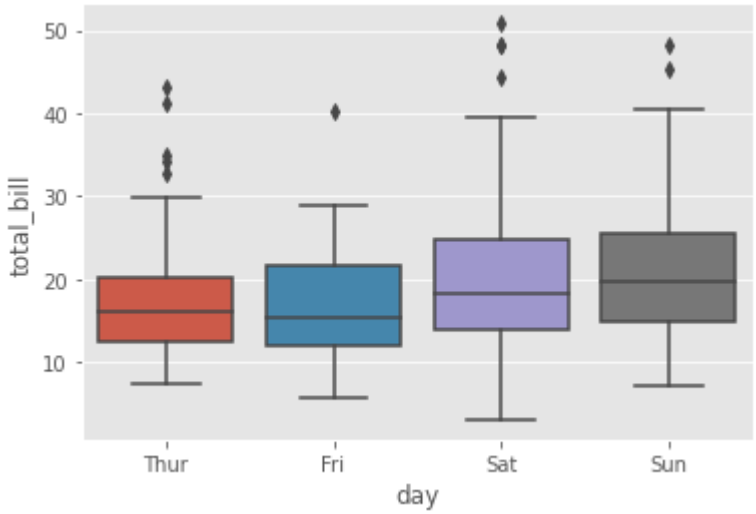
A boxplot is a standardized way of displaying the distribution of data based on a five number summary **("minimum", first quartile [Q1], median, third quartile [Q3] and "maximum")**.

- It can tell you about your outliers and what their values are.
- Boxplots can also tell you if your data is symmetrical,
- how tightly your data is grouped and if and how your data is skewed.



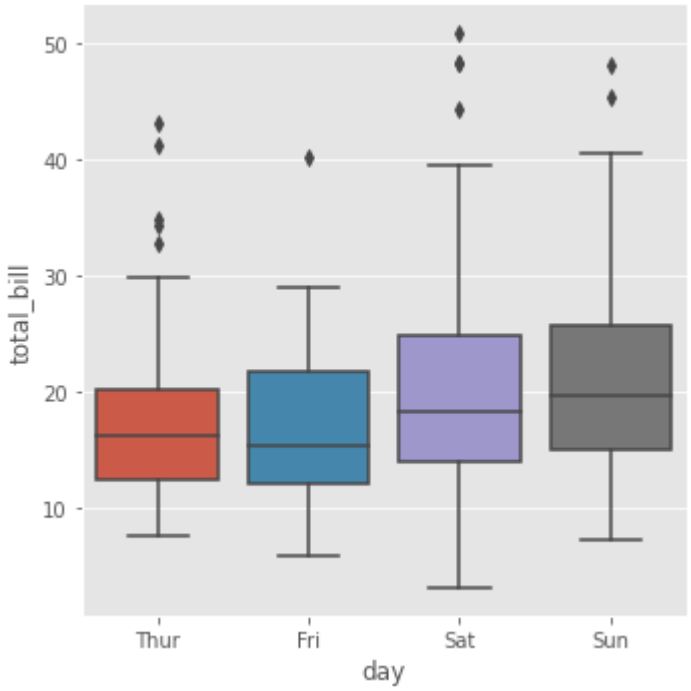
In [13]:
```
# Box plot
sns.boxplot(data=tips,x='day',y='total_bill')
```

Out[13]:
```
<AxesSubplot:xlabel='day', ylabel='total_bill'>
```



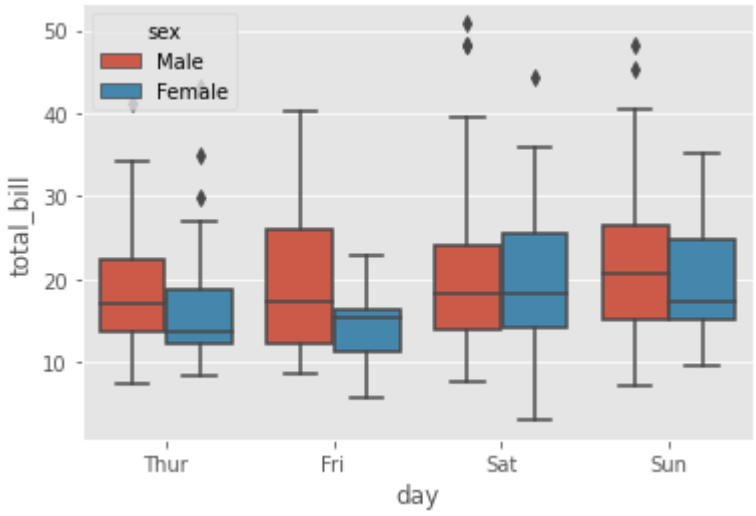
In [14]:
```
# Using catplot - Figure level

sns.catplot(data=tips,x='day',y='total_bill',kind='box')
```

Out[14]:
```
<seaborn.axisgrid.FacetGrid at 0x1e85f3b5dc0>
```



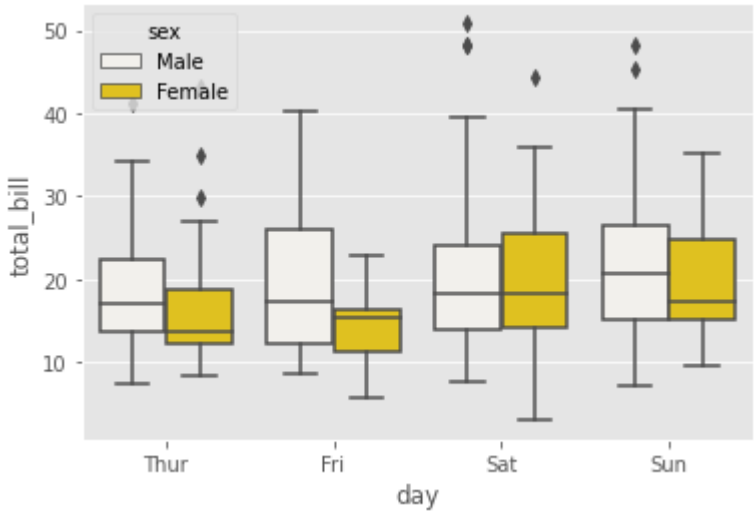
In [15]:
```
# hue

sns.boxplot(data=tips,x='day',y='total_bill',hue='sex')
```

Out[15]:
```
<AxesSubplot:xlabel='day', ylabel='total_bill'>
```



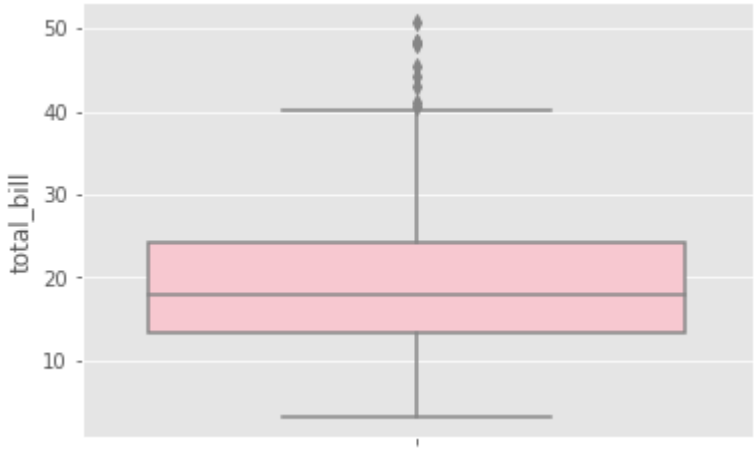
In [16]:
```
# color

sns.boxplot(data=tips,x='day',y='total_bill',hue='sex', color = 'gold')
```

Out[16]: <AxesSubplot:xlabel='day', ylabel='total_bill'>



In [17]:
```
# single boxplot -> numerical col

sns.boxplot(data=tips , y='total_bill', color ='pink')
```

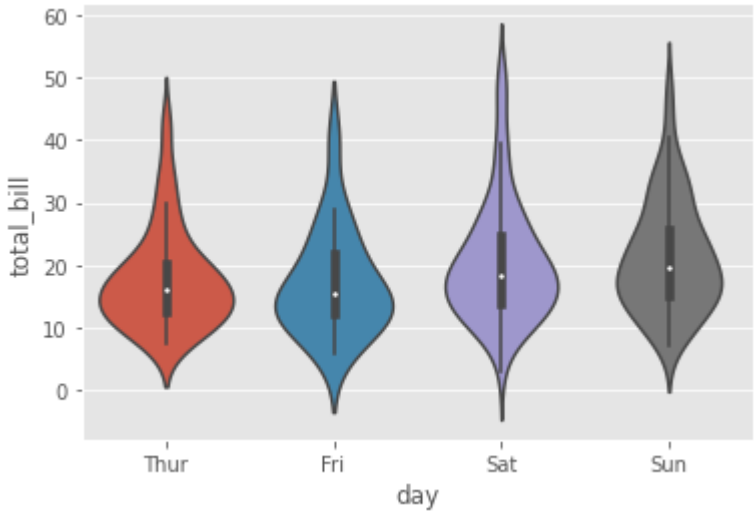
Out[17]: <AxesSubplot:ylabel='total_bill'>



2 . Voilin Plot

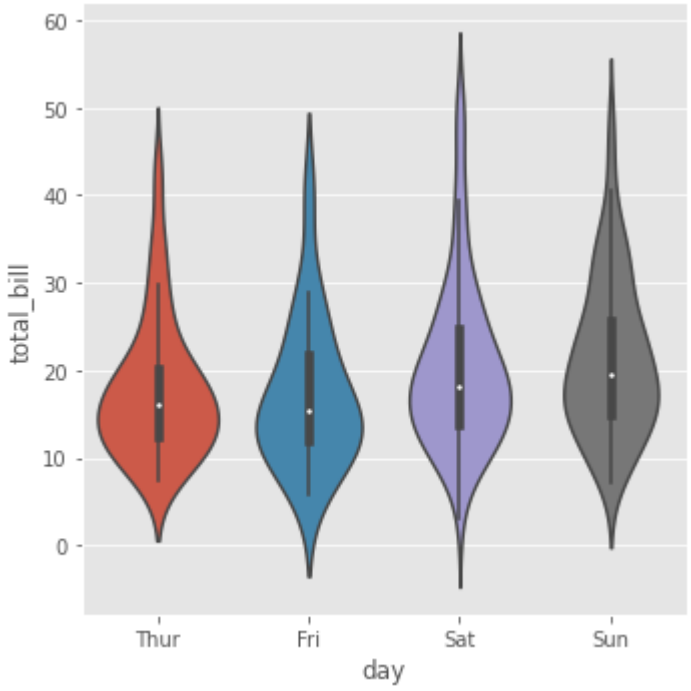
In [18]:
```
# Violinplot = (Boxplot + KDEplot)

sns.violinplot(data=tips,x='day',y='total_bill')
```

Out[18]:
```
<AxesSubplot:xlabel='day', ylabel='total_bill'>
```



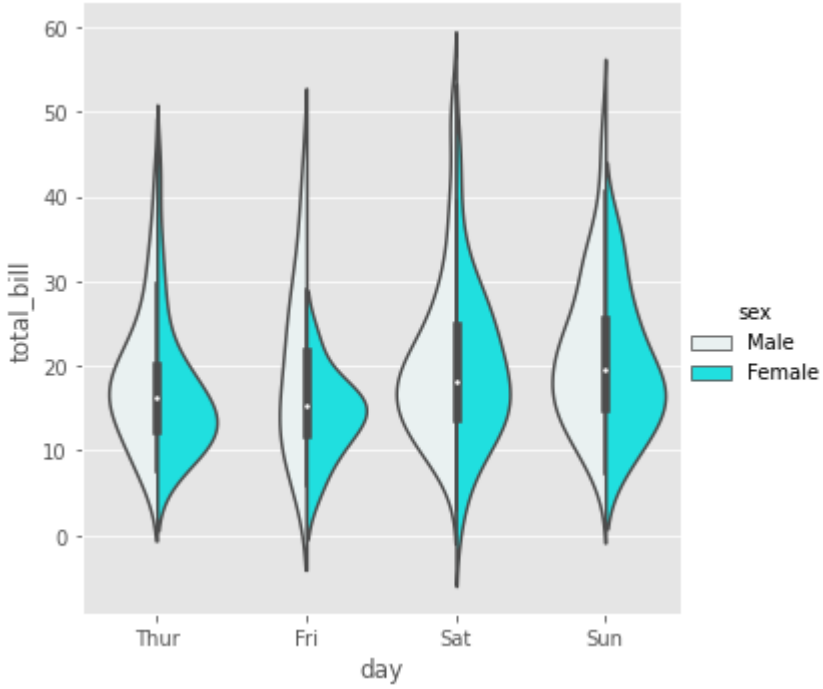
In [19]:
```
# catplot -figure level function

sns.catplot(data=tips,x='day',y='total_bill',kind ='violin')
```

Out[19]:
```
<seaborn.axisgrid.FacetGrid at 0x1e85f61a490>
```



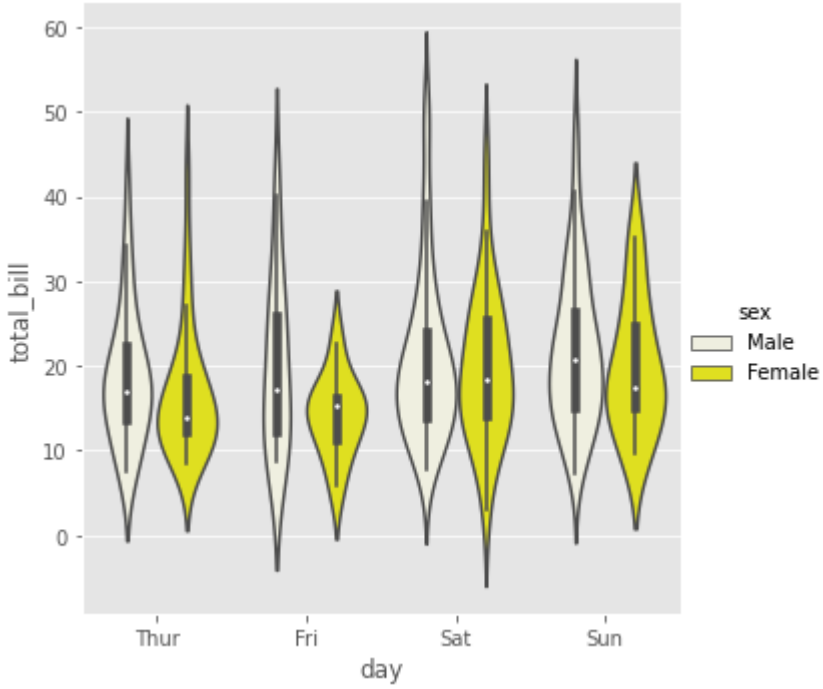
In [20]:
```
# Color

sns.catplot(data=tips,x='day',y='total_bill',
            kind ='violin', hue ='sex', split =True, color='cyan')
```

Out[20]: <seaborn.axisgrid.FacetGrid at 0x1e85f5b2580>



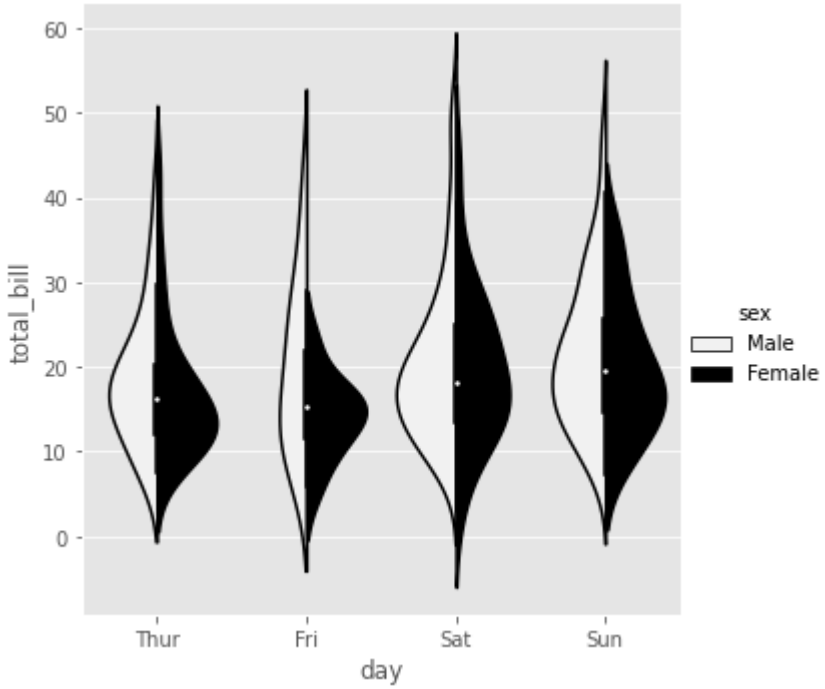
In [21]:
```
# hue

sns.catplot(data=tips,x='day',y='total_bill',kind ='violin',
            hue ='sex' , color ='yellow')
```

Out[21]: <seaborn.axisgrid.FacetGrid at 0x1e860a5da60>



In [22]:
```
# split

sns.catplot(data=tips,x='day',y='total_bill',
            kind ='violin', hue ='sex',
            color ='black', split =True)
```

Out[22]: <seaborn.axisgrid.FacetGrid at 0x1e860a0f580>



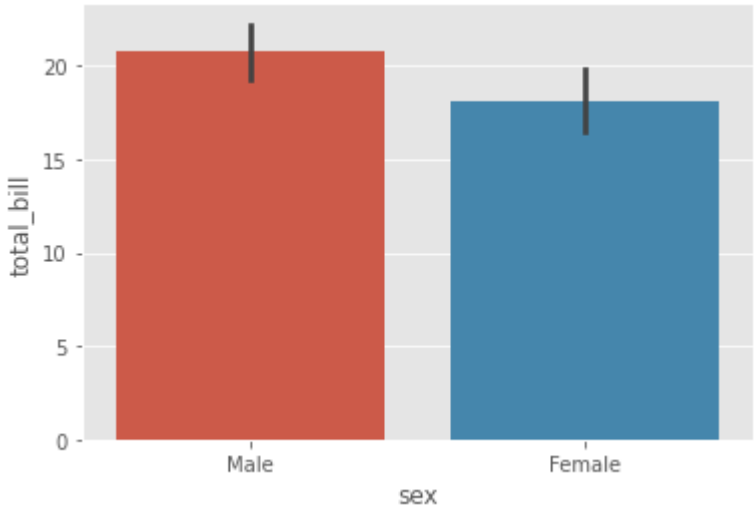
Categorical Estimate Plot

1.barplot

- When there are multiple observations in each category, it also uses bootstrapping to compute a confidence interval around the estimate, which is plotted using **error bars**

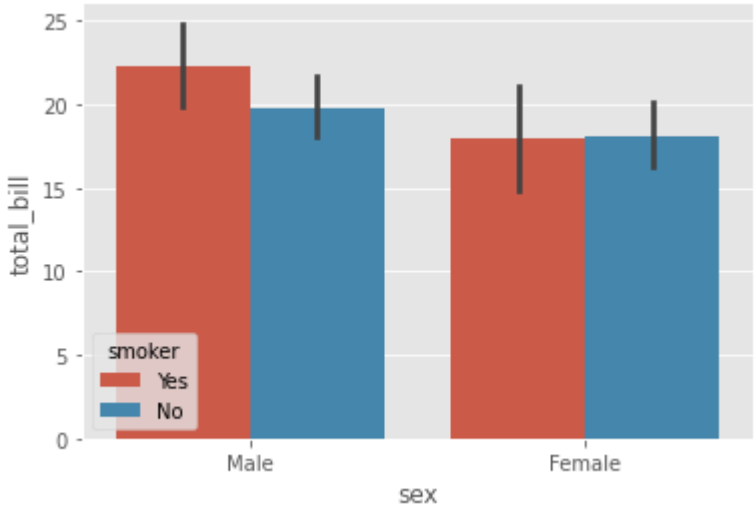
In [23]:
```
sns.barplot(data=tips, x='sex', y='total_bill')
```

Out[23]: <AxesSubplot:xlabel='sex', ylabel='total_bill'>



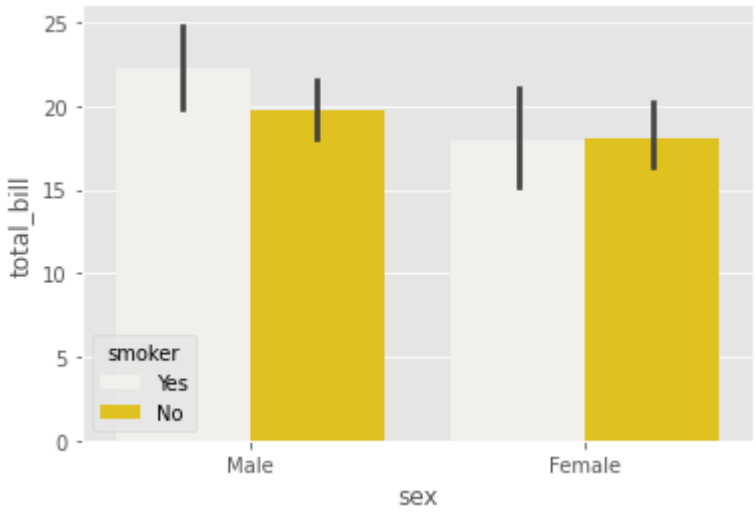
In [24]:
```
# hue

sns.barplot(data=tips, x='sex', y='total_bill',hue='smoker')
```

Out[24]: <AxesSubplot:xlabel='sex', ylabel='total_bill'>



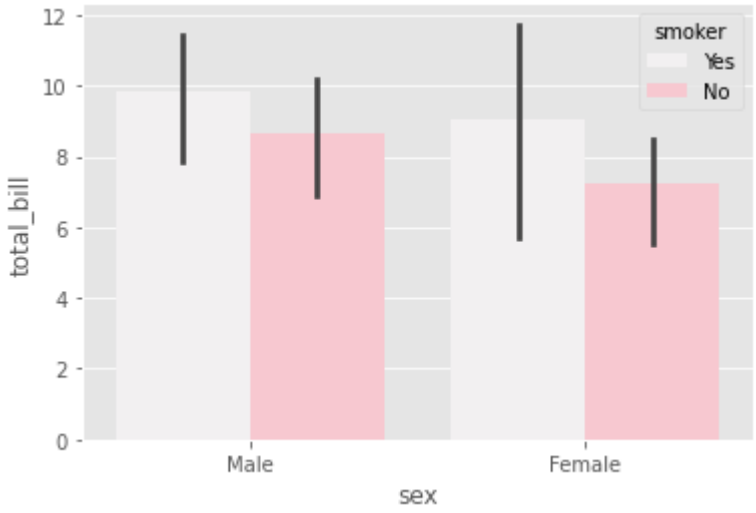
In [25]:
```
# color

sns.barplot(data=tips, x='sex', y='total_bill',hue='smoker',
              color ='gold')
```

Out[25]: <AxesSubplot:xlabel='sex', ylabel='total_bill'>



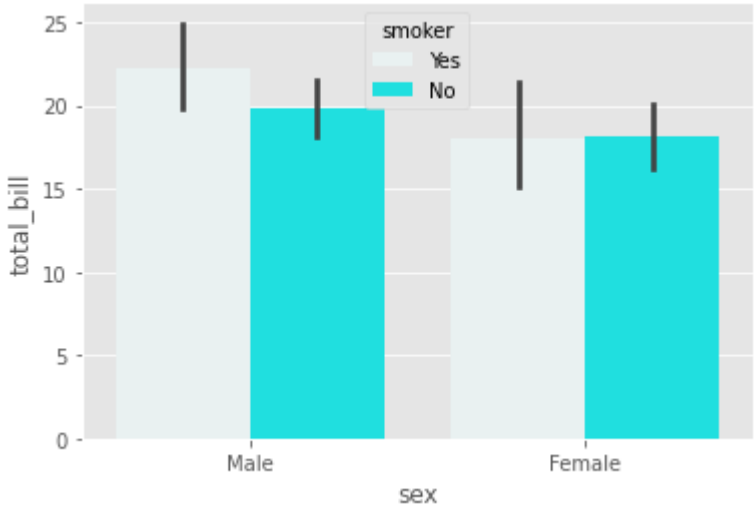
In [26]:
```
# estimator

import numpy as np

sns.barplot(data=tips, x='sex', y='total_bill',hue='smoker', color ='pink',
            estimator=np.std)
```

Out[26]: <AxesSubplot:xlabel='sex', ylabel='total_bill'>



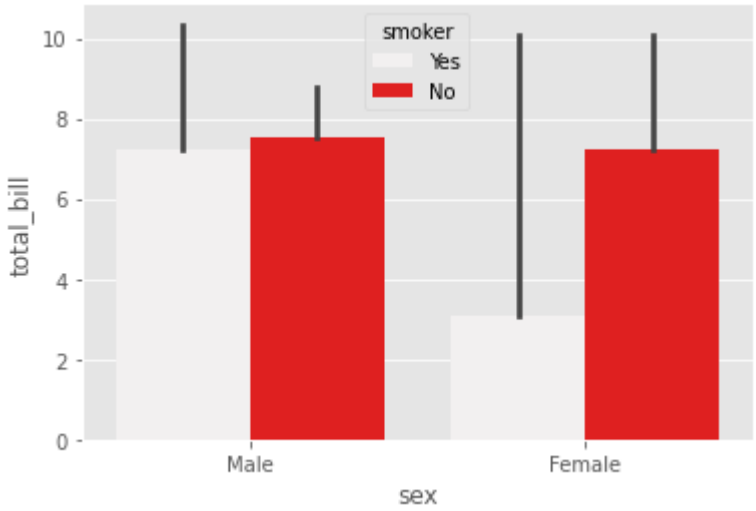
In [27]:
```
# estimator ( mean)


sns.barplot(data=tips, x='sex', y='total_bill',hue='smoker',color ='cyan',
            estimator=np.mean)
```

Out[27]: <AxesSubplot:xlabel='sex', ylabel='total_bill'>



In [28]:
```
# estimator ( min)


sns.barplot(data=tips, x='sex', y='total_bill',hue='smoker',color ='red',
            estimator=np.min)
```

Out[28]: <AxesSubplot:xlabel='sex', ylabel='total_bill'>

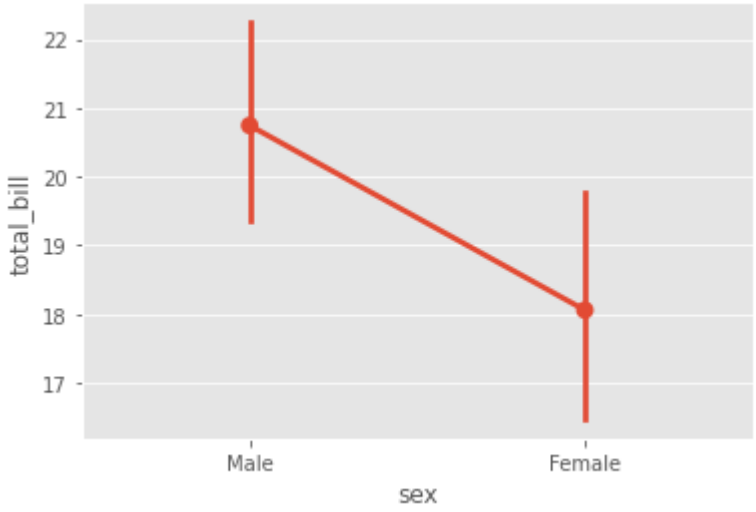


2. Point plot

A point plot represents an estimate of central tendency for a numeric variable by the position of

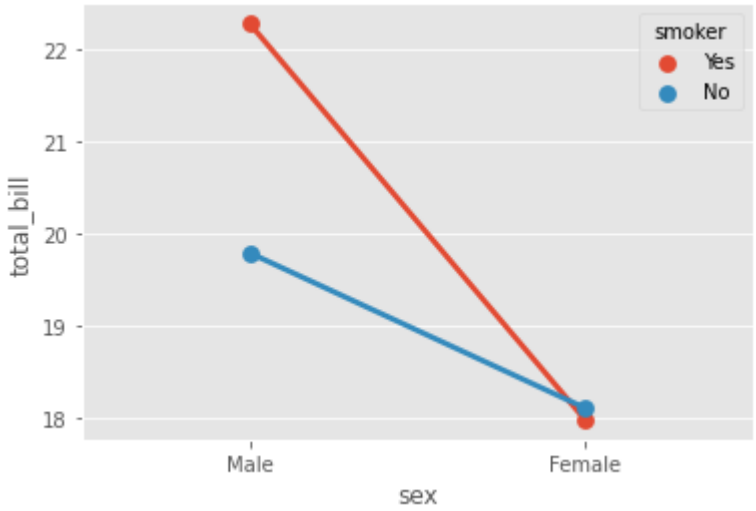
In [29]:
```
sns.pointplot(data=tips, x='sex', y='total_bill')
```

Out[29]: <AxesSubplot:xlabel='sex', ylabel='total_bill'>



In [30]:
```
# CI ---> for removing error bars

sns.pointplot(data=tips, x='sex', y='total_bill' ,
              hue ='smoker', ci =None)
```

Out[30]: <AxesSubplot:xlabel='sex', ylabel='total_bill'>

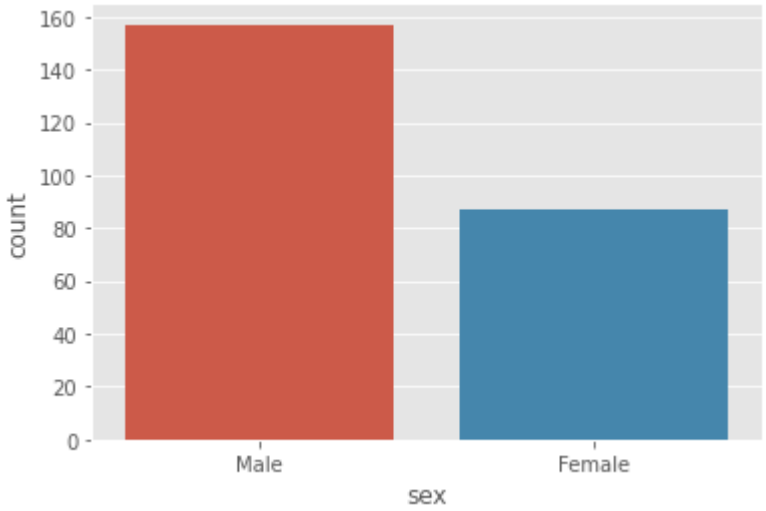


3.Count Plot

A special case for the bar plot is when you want to show the number of observations in each category rather than computing a statistic for a second variable. This is similar to a histogram over a categorical, rather than quantitative, variable

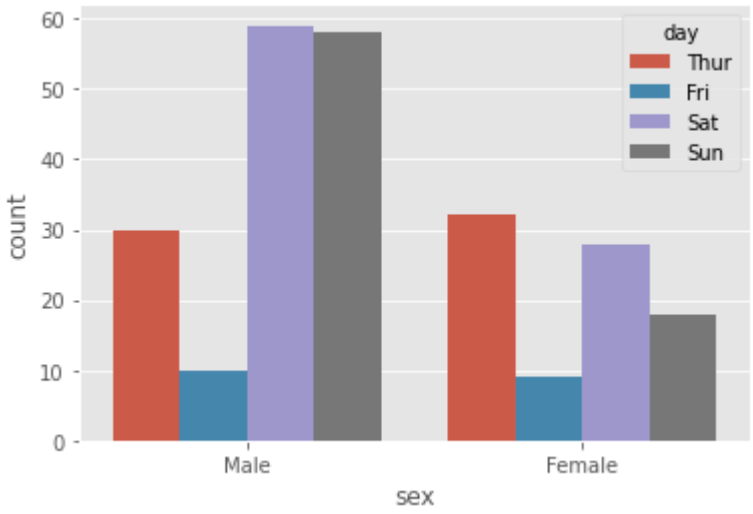
In [31]:
```
# countplot

sns.countplot(data=tips,x='sex')
```

Out[31]: <AxesSubplot:xlabel='sex', ylabel='count'>



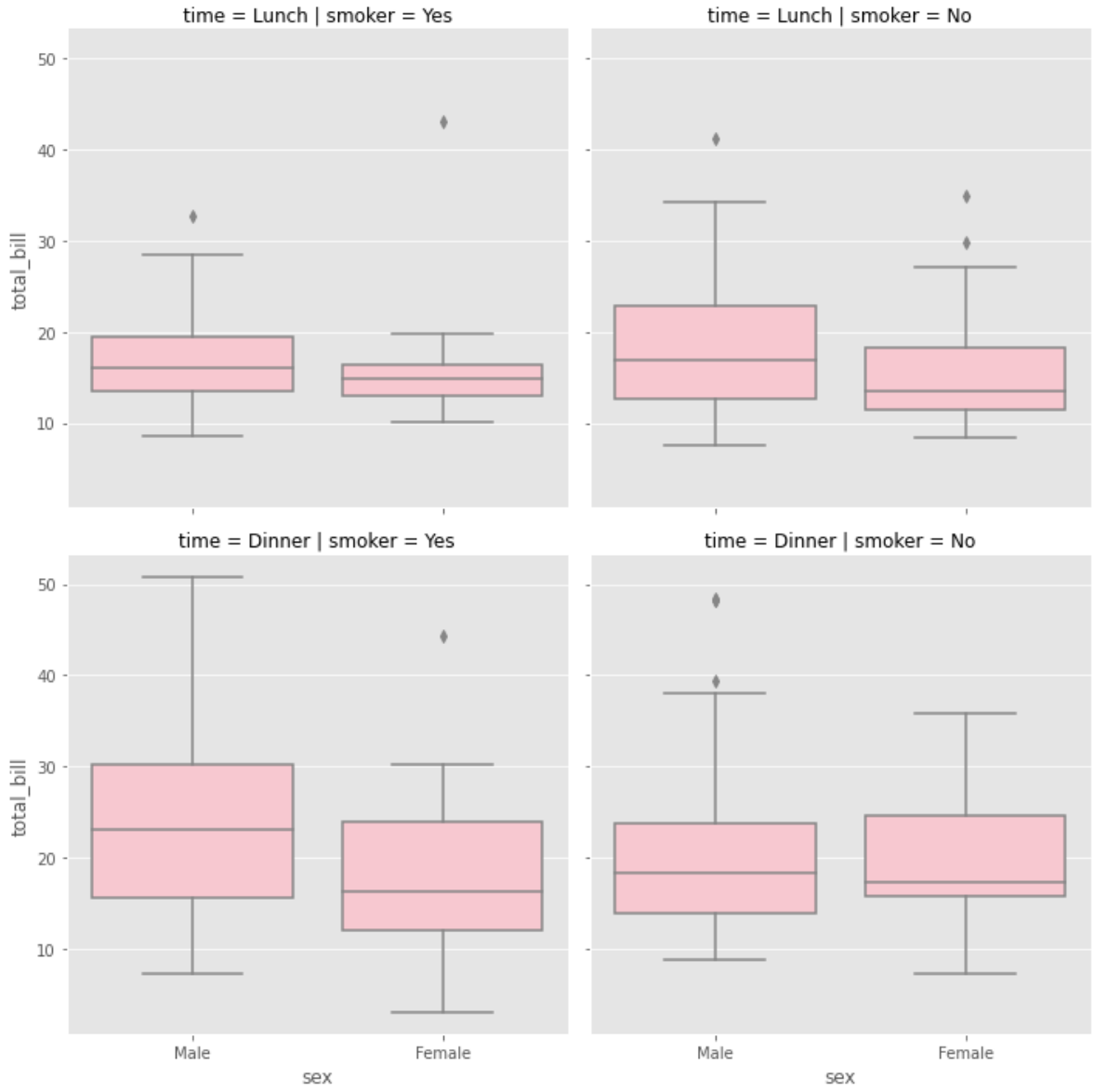
In [32]:
```
sns.countplot(data=tips,x='sex',hue='day')
```

Out[32]: <AxesSubplot:xlabel='sex', ylabel='count'>



In [33]:
```
# faceting using catplot

sns.catplot(data=tips, x='sex',y='total_bill',
            col='smoker',kind='box',row='time',color ='pink')
```

Out[33]:
```
<seaborn.axisgrid.FacetGrid at 0x1e860f46a00>
```



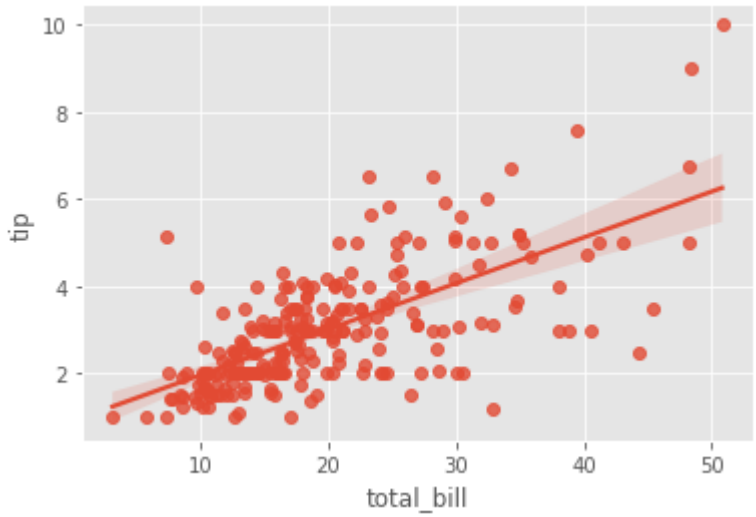
Regression Plots

- regplot
- lmplot

In the simplest invocation, both functions draw a scatterplot of two variables, x and y, and then fit the regression model y ~ x and plot the resulting regression line and a 95% confidence interval for that regression.

In [34]:
```
# axes level

# hue parameter is not available

sns.regplot(data=tips,x='total_bill',y='tip')
```

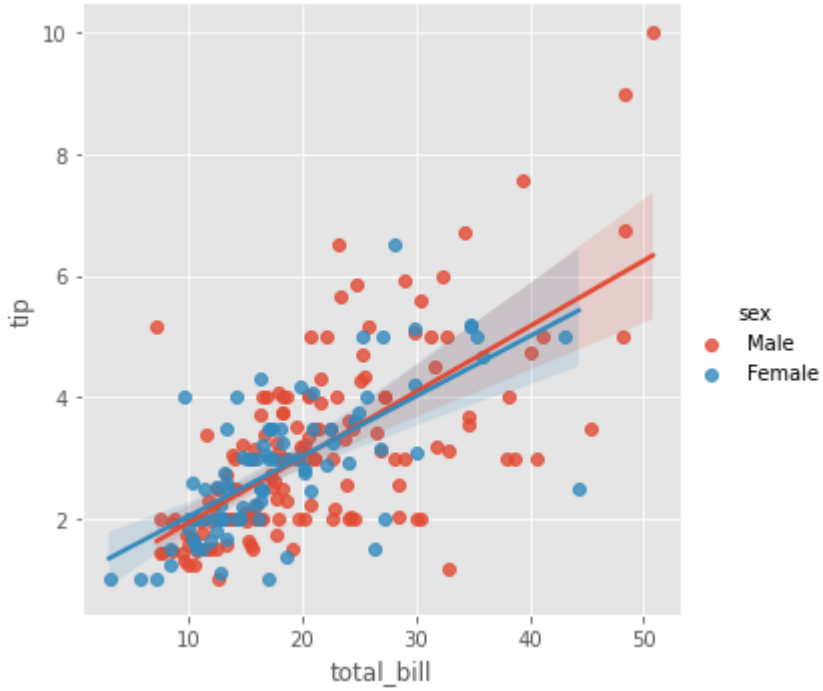
Out[34]: <AxesSubplot:xlabel='total_bill', ylabel='tip'>



Lmplot

In [35]:
```
sns.lmplot(data=tips,x='total_bill',y='tip', hue ='sex')
```

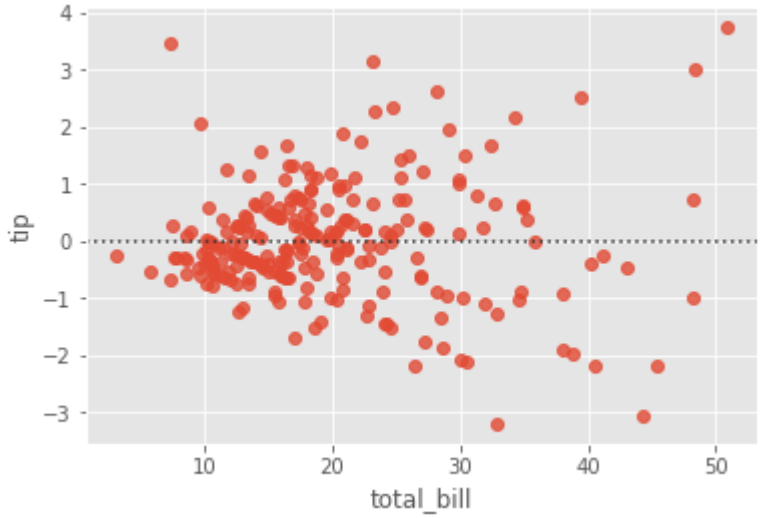
Out[35]: <seaborn.axisgrid.FacetGrid at 0x1e862649a00>



residplot

In [36]:
```
# residual plot

sns.residplot(data=tips,x='total_bill',y='tip')
```

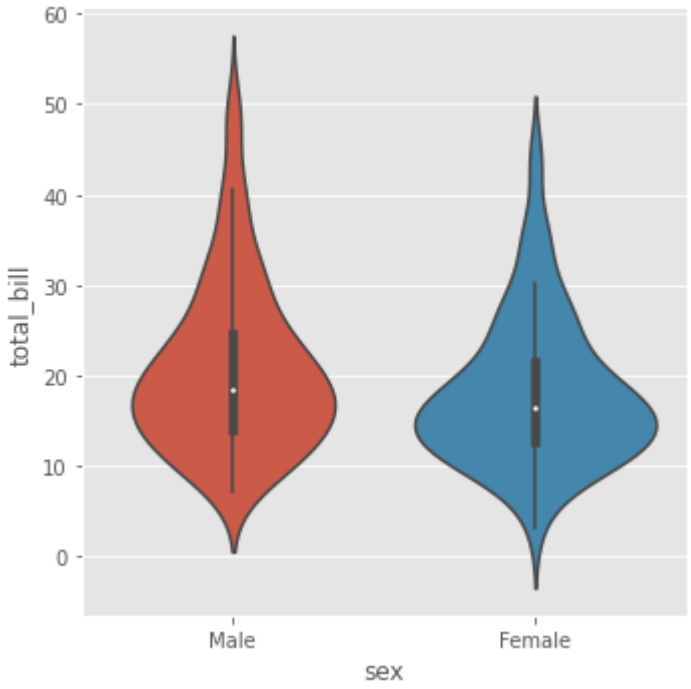
Out[36]:
```
<AxesSubplot:xlabel='total_bill', ylabel='tip'>
```



A second way to plot Facet plots -> FacetGrid

In [37]:
```
# figure level -> relplot -> displot -> catplot -> lmplot

sns.catplot(data=tips,x='sex',y='total_bill',
            kind='violin')
```

Out[37]: <seaborn.axisgrid.FacetGrid at 0x1e85e854190>

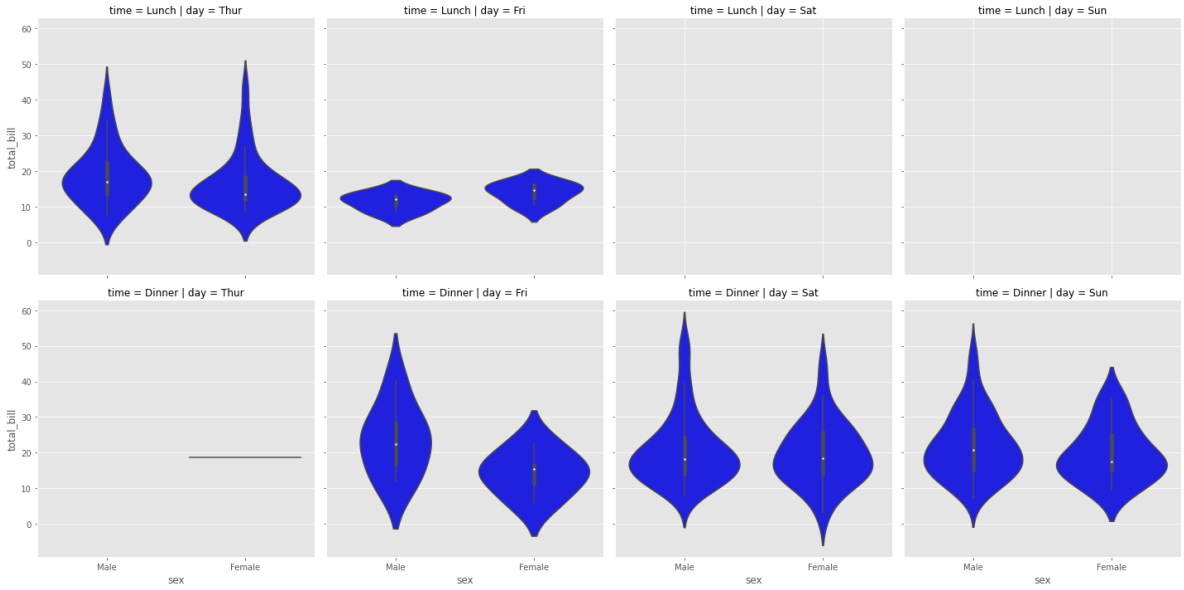


In [41]:
```
# Facet Plot - columns

sns.catplot(data=tips,x='sex',y='total_bill', color = 'gold',
            kind='violin',col='day')
```

Out[41]: <seaborn.axisgrid.FacetGrid at 0x1e862e98fd0>

In [50]:
```
# Facet Plot - column + rows

sns.catplot(data=tips,x='sex',y='total_bill',color ='blue' ,
            kind='violin',col='day',row='time')
```

Out[50]: <seaborn.axisgrid.FacetGrid at 0x1e86965dbe0>



Facetgrid

In [48]:
```
# Second Method - Facetgrid

# it is the lower level code of catplot

secondmethod = sns.FacetGrid(data =tips ,col ='day' , row ='time')

secondmethod.map(sns.violinplot,'sex','total_bill')
```

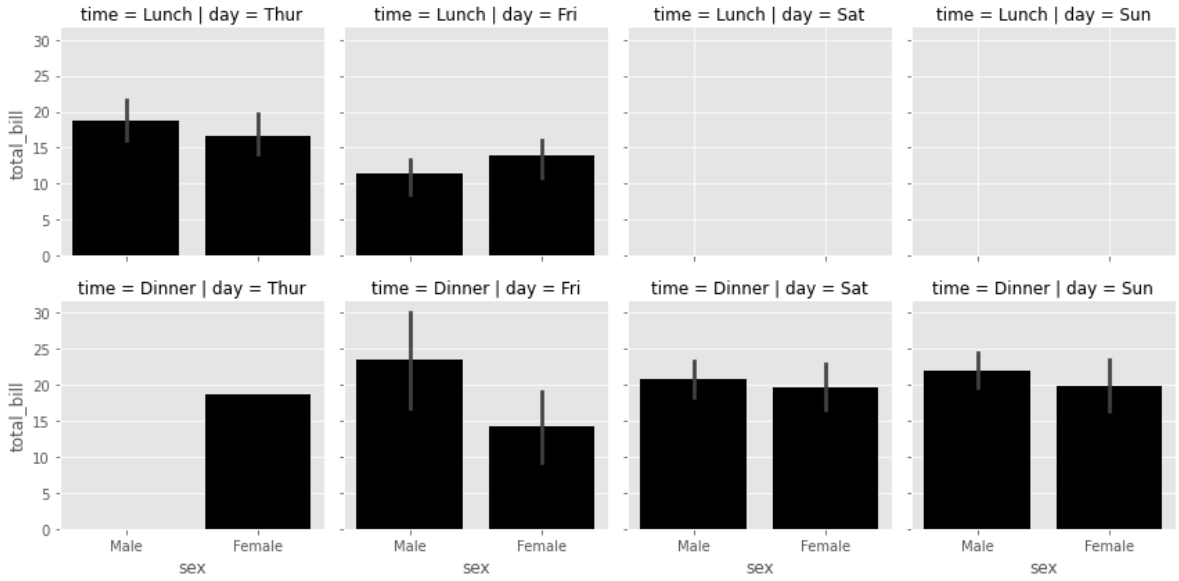
```
C:\Users\user\anaconda3\lib\site-packages\seaborn\axisgrid.py:670: UserWarnin
g: Using the violinplot function without specifying `order` is likely to prod
uce an incorrect plot.
  warnings.warn(warning)
```

Out[48]: <seaborn.axisgrid.FacetGrid at 0x1e868e51580>

In [53]:
```
secondmethod = sns.FacetGrid(data =tips ,col ='day' , row ='time' )

secondmethod.map(sns.barplot,'sex','total_bill',color='black')
```

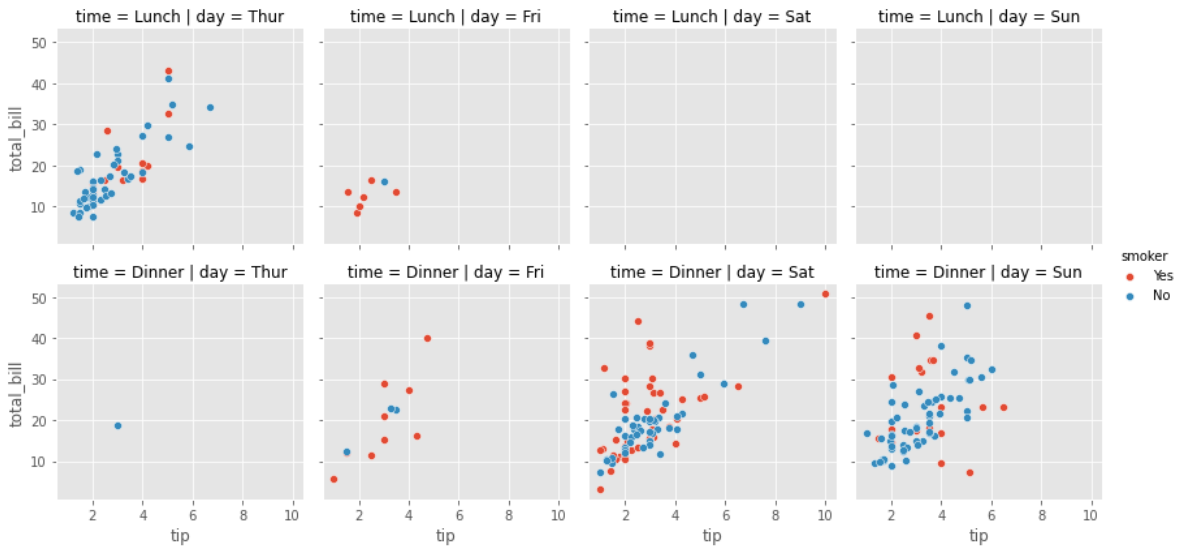
```
C:\Users\user\anaconda3\lib\site-packages\seaborn\axisgrid.py:670: UserWarnin
g: Using the barplot function without specifying `order` is likely to produce
an incorrect plot.
  warnings.warn(warning)
```

Out[53]: <seaborn.axisgrid.FacetGrid at 0x1e867b9caf0>



In [57]:
```
# Adding Legend

secondmethod = sns.FacetGrid(data=tips,col='day',row='time',hue='smoker')

secondmethod.map(sns.scatterplot,'tip','total_bill')

secondmethod.add_legend()
```

Out[57]: <seaborn.axisgrid.FacetGrid at 0x1e86bf707f0>



Plotting Pairwise Relationship (PairGrid Vs Pairplot)

Here's a concise explanation:

- **Pairplot**: It is a function in the seaborn library that **creates a grid of scatterplots**, showing the relationships between variables in a dataset. Each variable is plotted against every other variable, providing a visual overview of their correlations.
- **PairGrid**: It is a class in seaborn that allows for **more customization** in creating a grid of subplots. With PairGrid, you can manually specify the plots to be displayed on the grid, including scatterplots, histograms, kernel density plots, etc. It provides more flexibility in creating complex visualizations compared to pairplot.

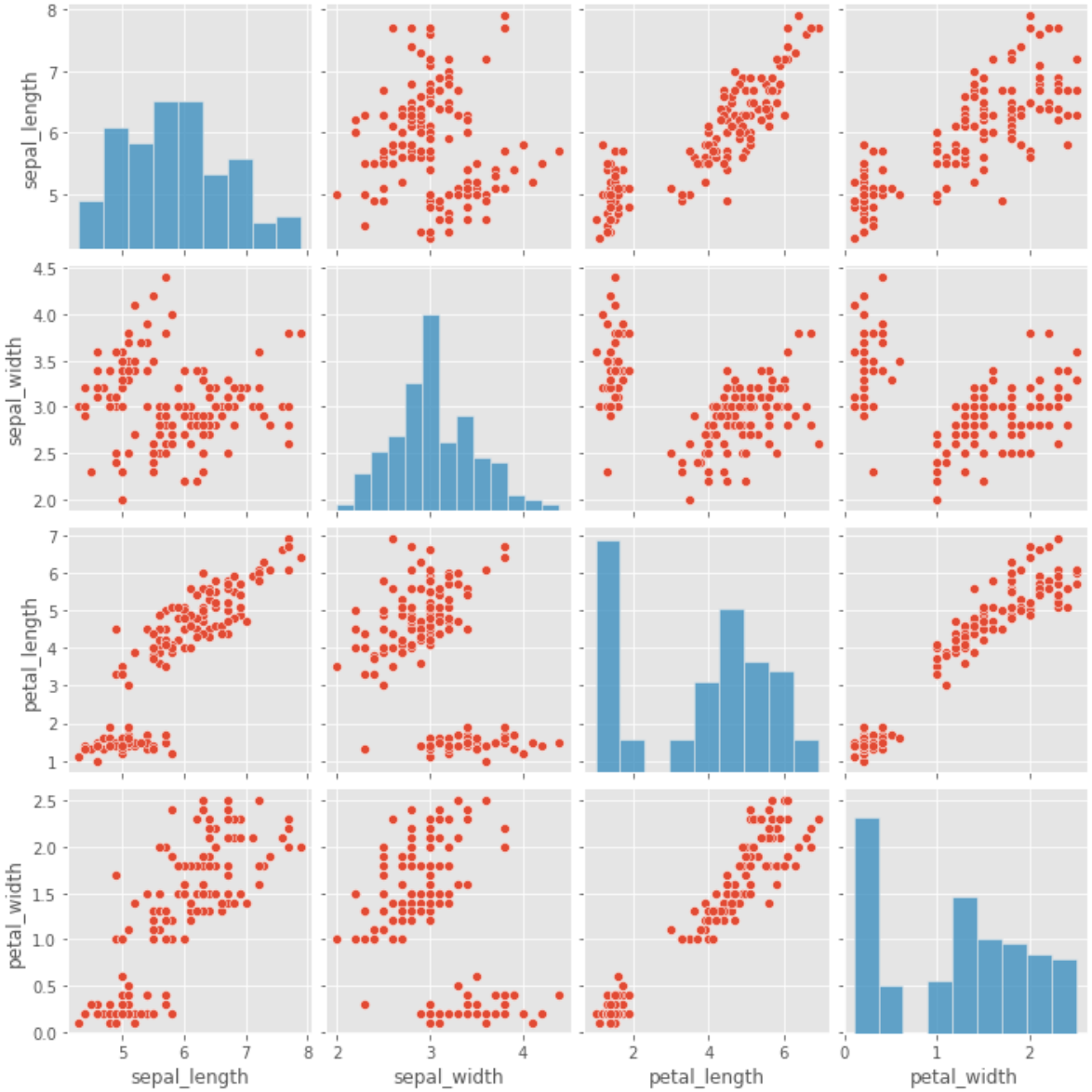
In [59]:
```
iris.sample()
```

Out[59]:

	sepal_length	sepal_width	petal_length	petal_width	species
53	5.5	2.3	4.0	1.3	versicolor

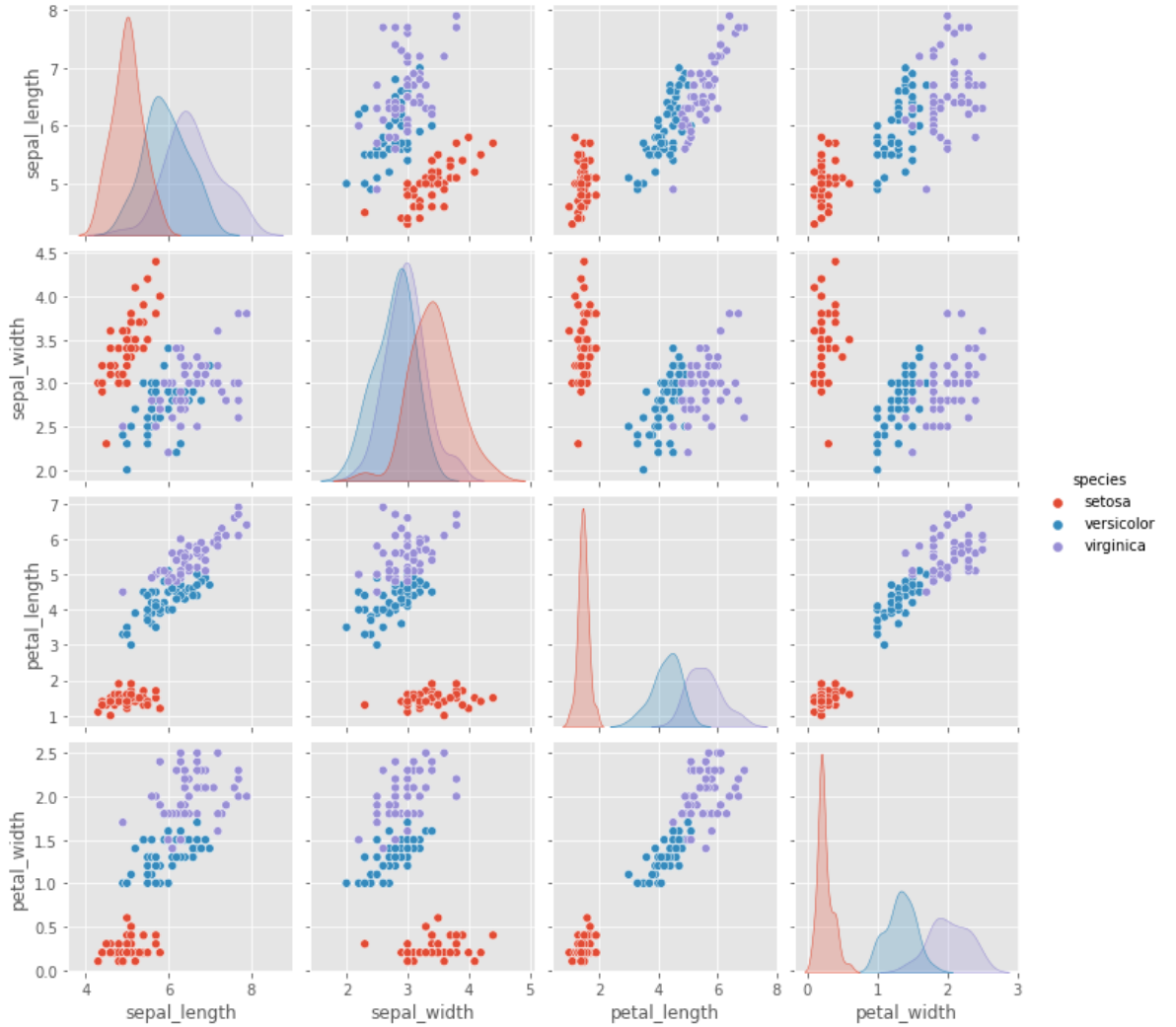
In [60]:
```
# Pairplot

sns.pairplot(iris)
```

Out[60]: <seaborn.axisgrid.PairGrid at 0x1e86c618880>



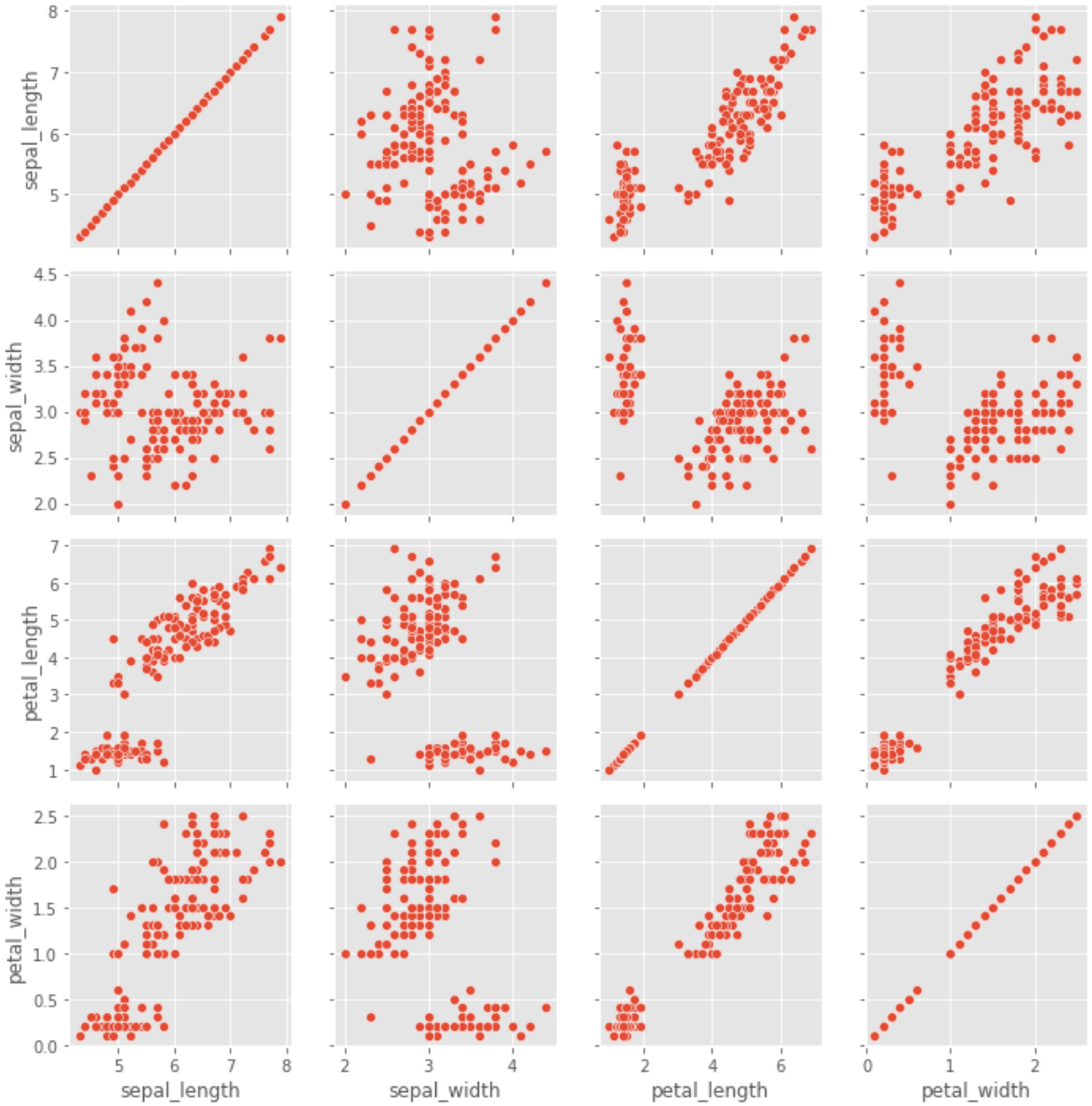
In [61]:
```
# hue

sns.pairplot(iris,hue='species')
```

Out[61]: <seaborn.axisgrid.PairGrid at 0x1e86da65460>



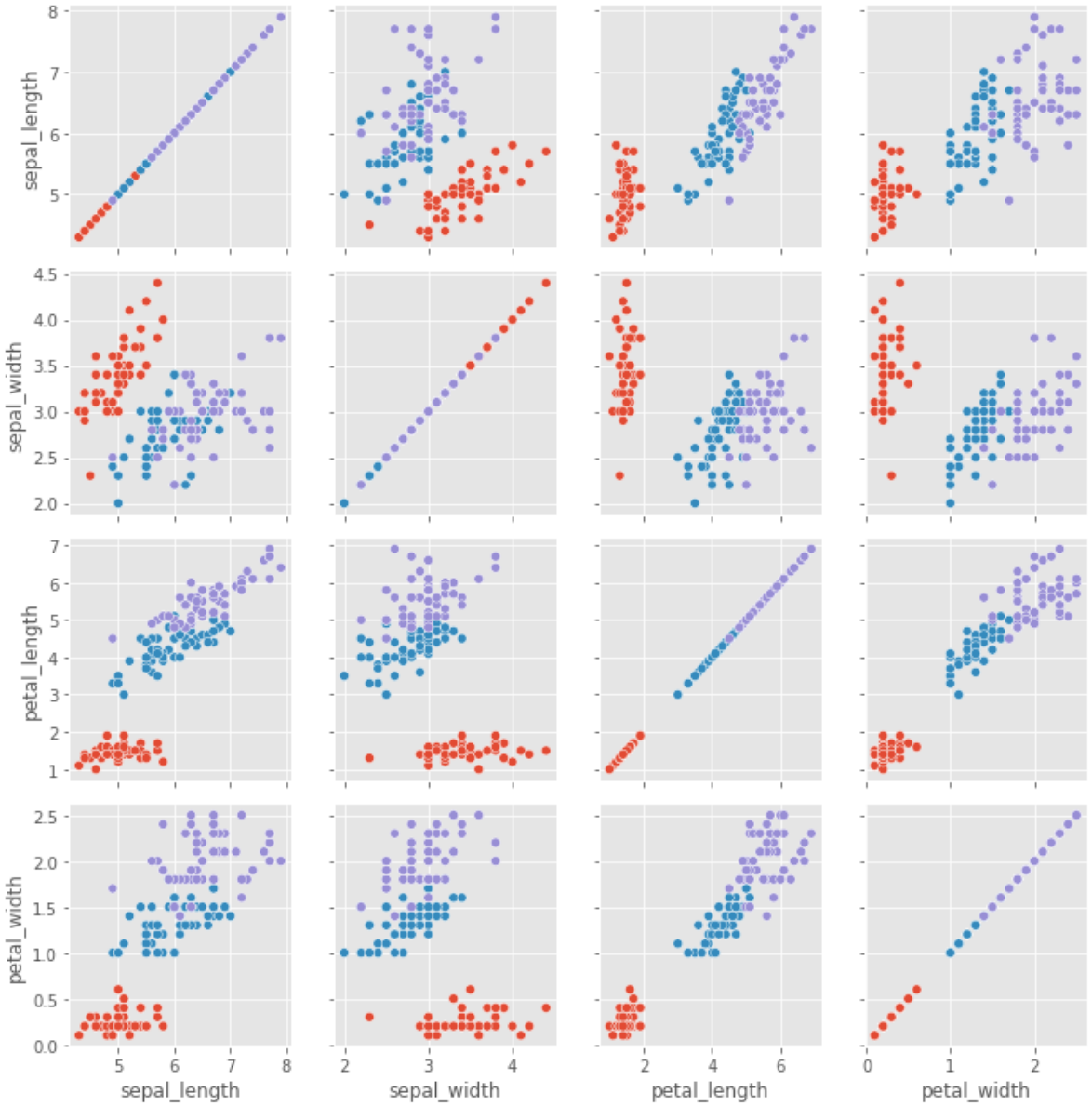
In [63]:
```
# pair grid

g = sns.PairGrid(data=iris )

# g.map

g.map(sns.scatterplot)
```

Out[63]: <seaborn.axisgrid.PairGrid at 0x1e8707bed90>



In [64]:
```
# hue

g = sns.PairGrid(data=iris,hue='species')

# g.map
g.map(sns.scatterplot)
```

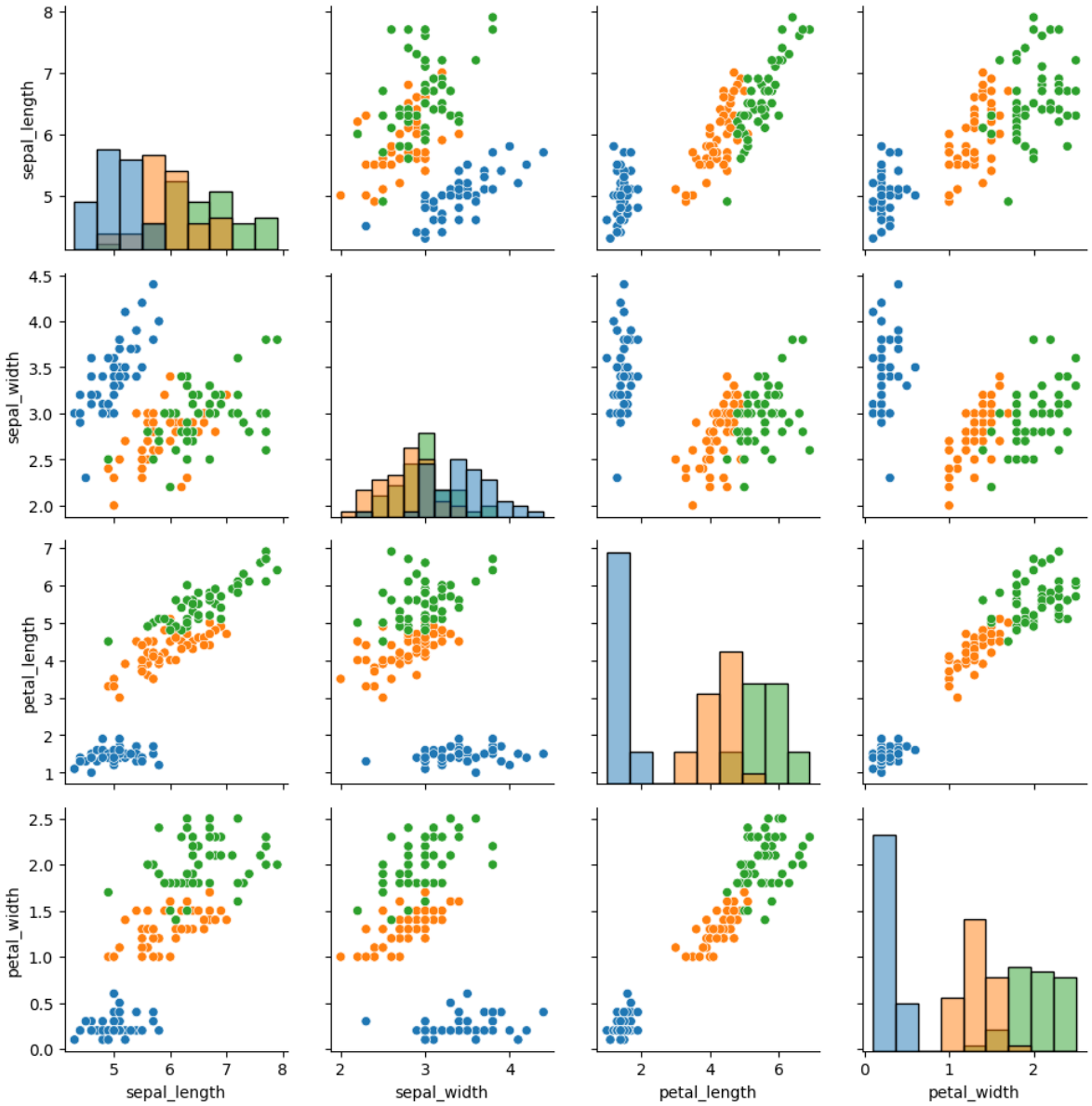
Out[64]: <seaborn.axisgrid.PairGrid at 0x1e870873640>



In [66]:
```
# style

plt.style.use('default')
```

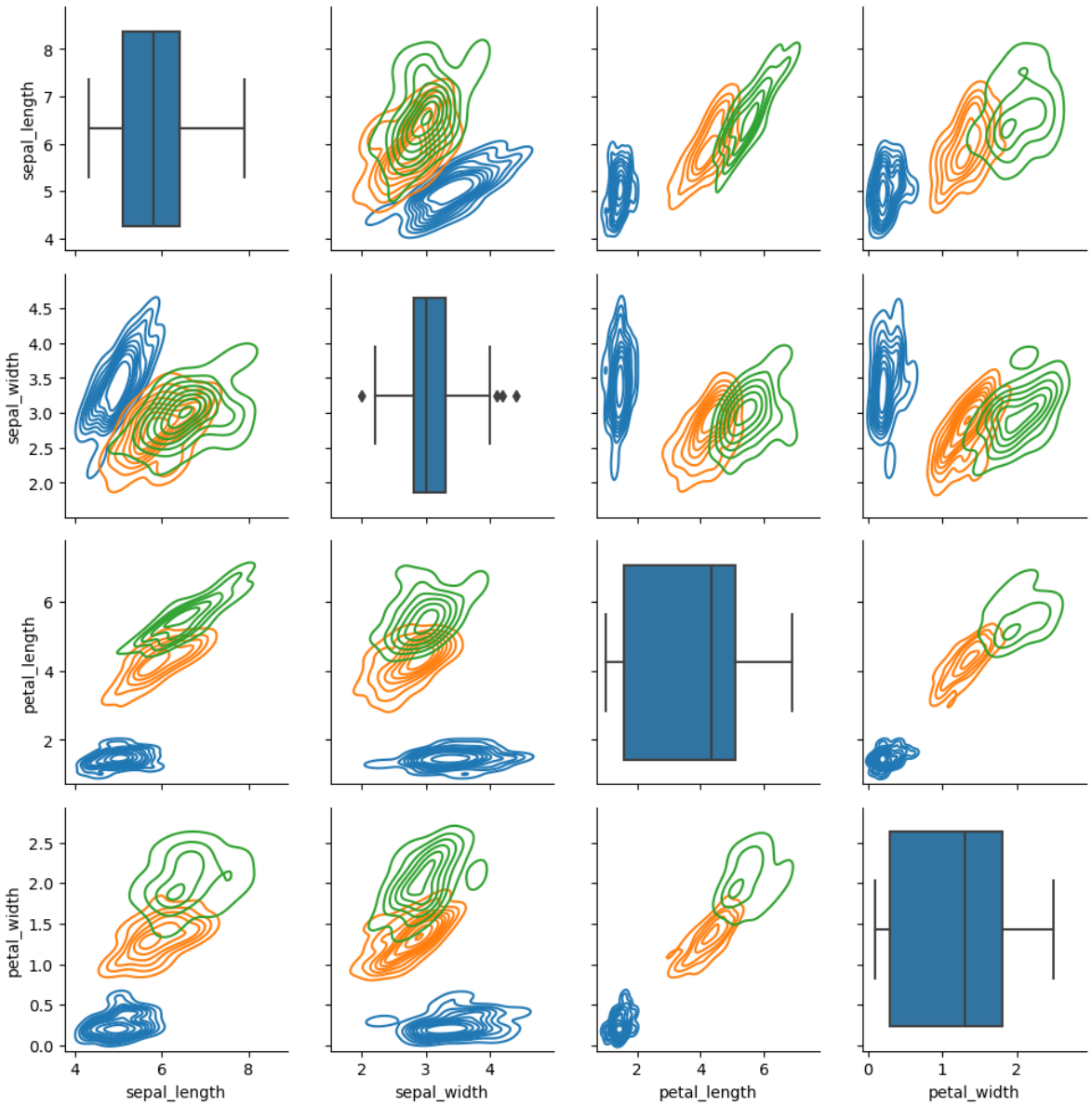
In [68]:
```
# map_diag -> map_offdiag

g = sns.PairGrid(data=iris,hue='species')

g.map_diag(sns. histplot)

g.map_offdiag(sns.scatterplot)
```

Out[68]: <seaborn.axisgrid.PairGrid at 0x1e8759bbc10>



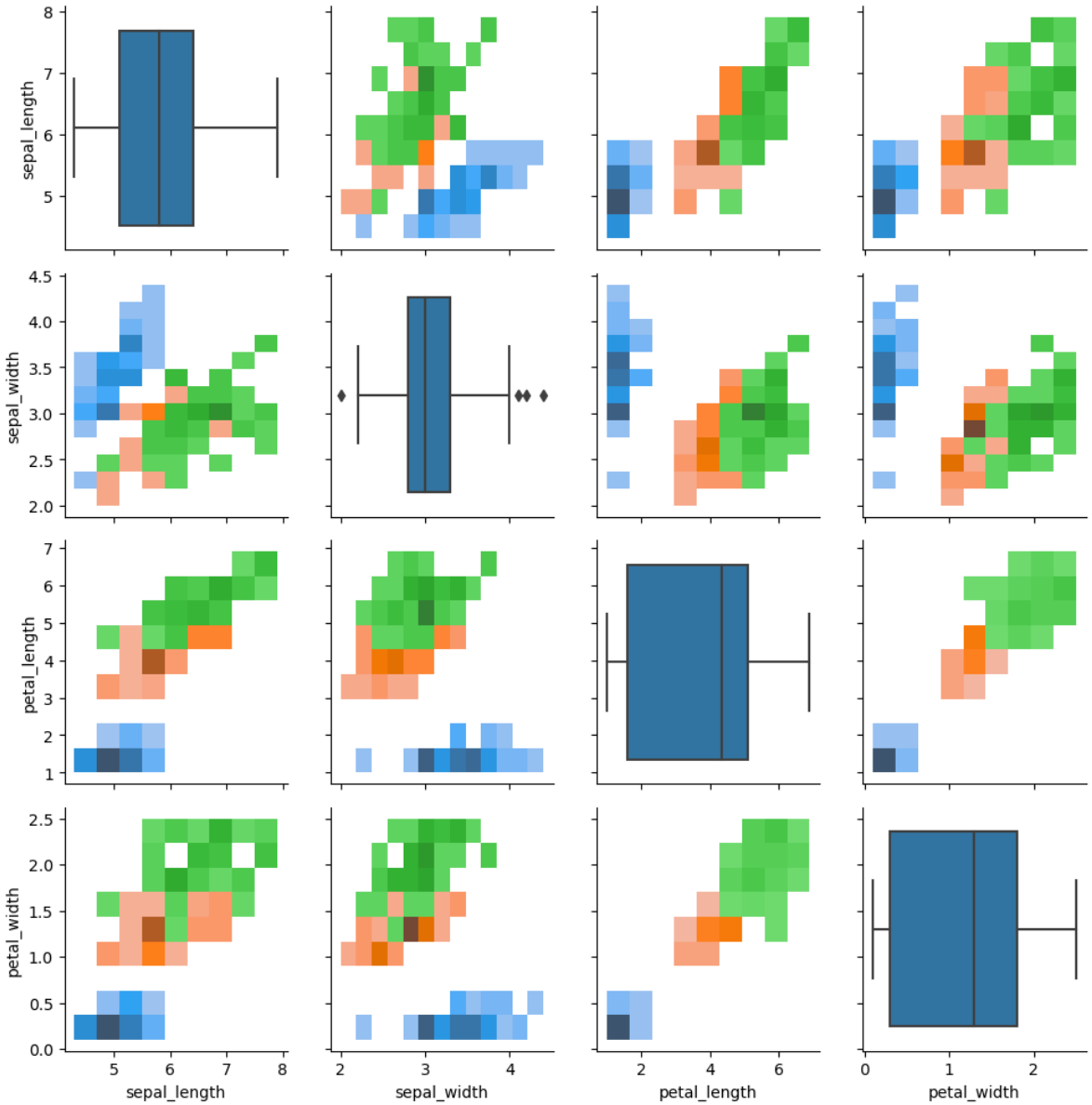
In [69]:
```
g = sns.PairGrid(data=iris,hue='species')

g.map_diag(sns.boxplot)

g.map_offdiag(sns.kdeplot)
```

Out[69]: <seaborn.axisgrid.PairGrid at 0x1e87684fca0>



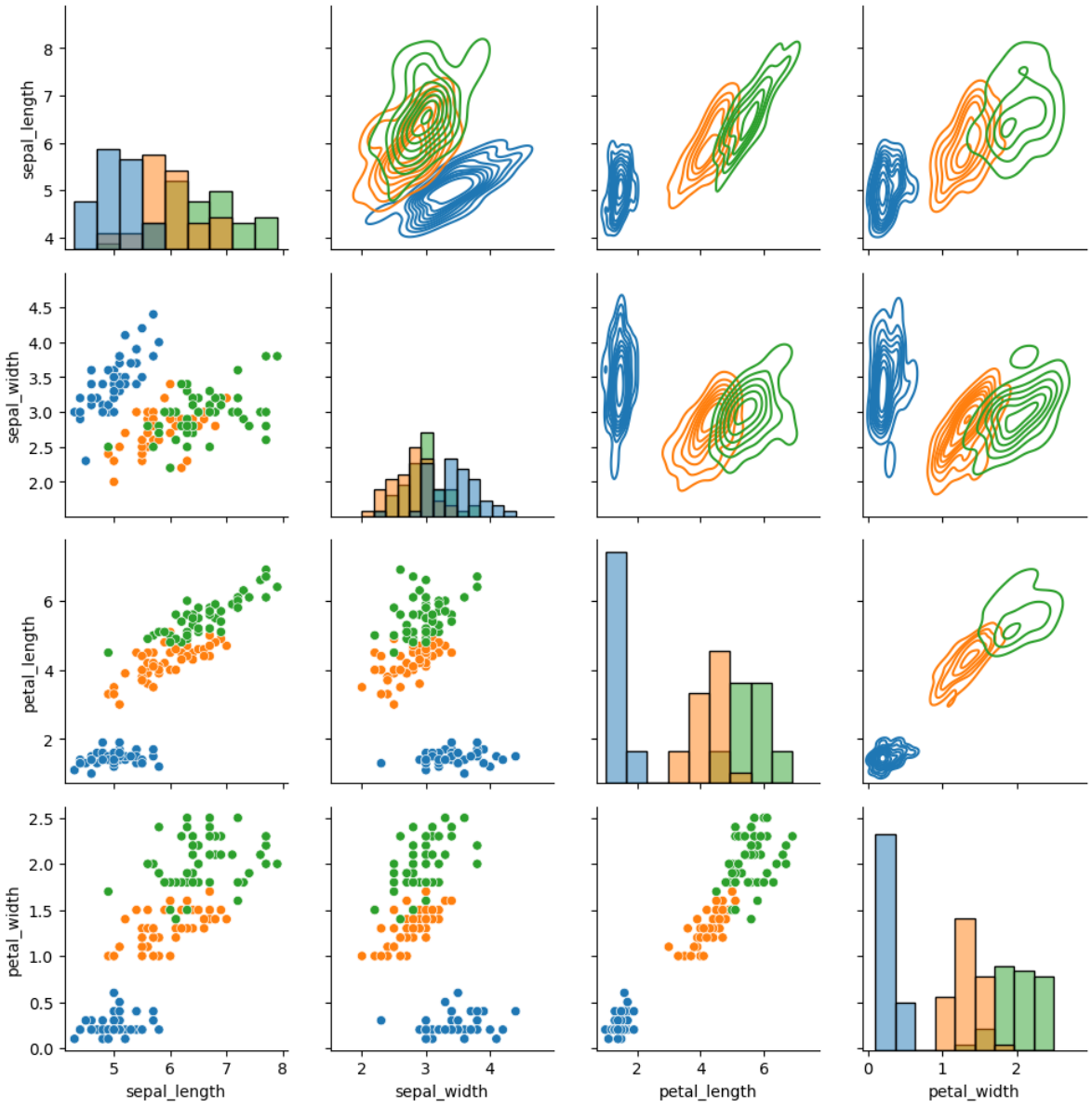
In [71]:
```
g = sns.PairGrid(data=iris,hue='species')

g.map_diag(sns.boxplot)

g.map_offdiag(sns.histplot)
```

Out[71]: <seaborn.axisgrid.PairGrid at 0x1e8788cdb20>



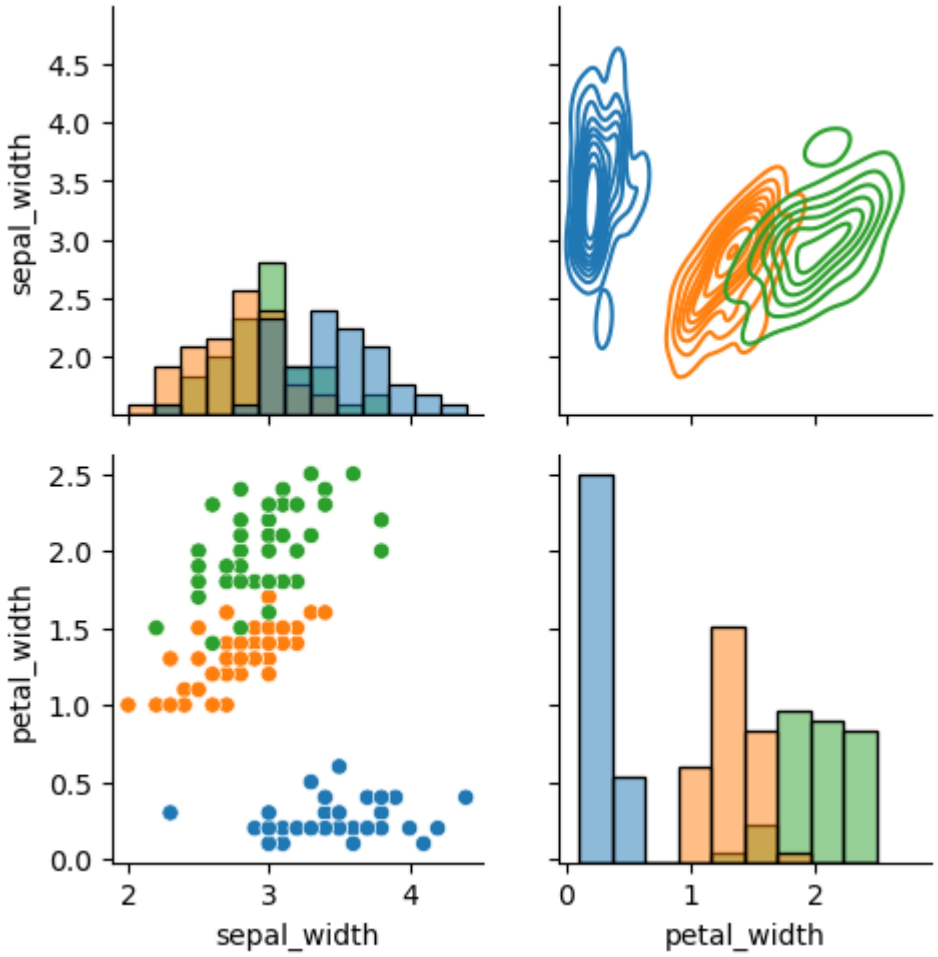
In [72]:
```
# map_diag -> map_upper -> map_lower

g = sns.PairGrid(data=iris,hue='species')

g.map_diag(sns.histplot) # diagonal

g.map_upper(sns.kdeplot) # upper

g.map_lower(sns.scatterplot) # lower
```

Out[72]: <seaborn.axisgrid.PairGrid at 0x1e8788de760>



In [73]:
```
# vars - working upon the desired columns

g = sns.PairGrid(data=iris,hue='species',vars=['sepal_width','petal_width'])

g.map_diag(sns.histplot)

g.map_upper(sns.kdeplot)

g.map_lower(sns.scatterplot)
```

Out[73]: <seaborn.axisgrid.PairGrid at 0x1e87b757520>



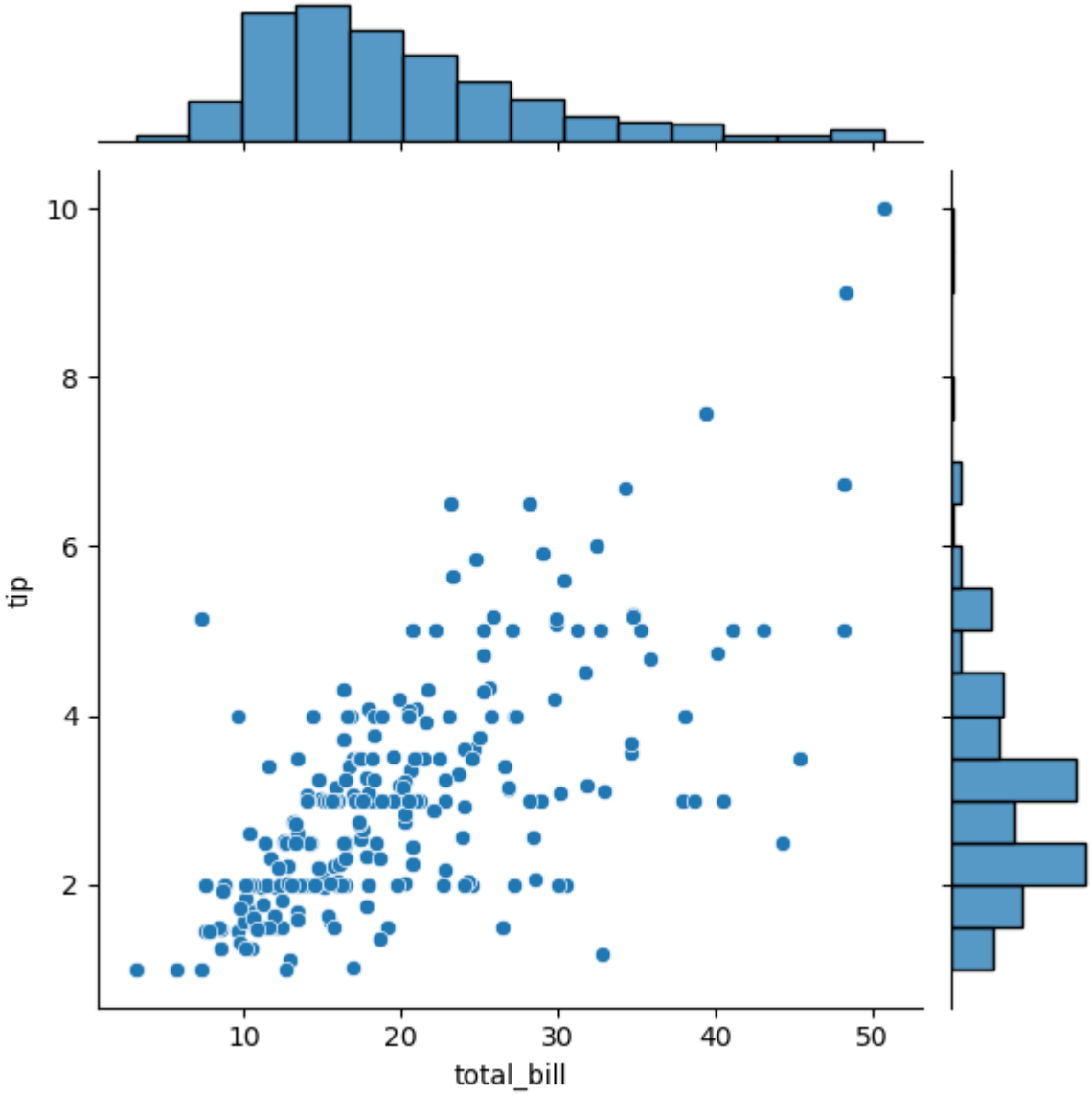
JointGrid Vs Jointplot

Here's a concise explanation:

- **Jointplot** : It is a function in seaborn that combines scatterplot and histograms (or kernel density plots) to visualize the **relationship between two variables**. It shows both the individual distributions and the correlation between the variables.
- JointGrid: It is a class in seaborn that provides a framework for creating **custom joint plots**. With JointGrid, you have more control over the layout and types of plots used in a

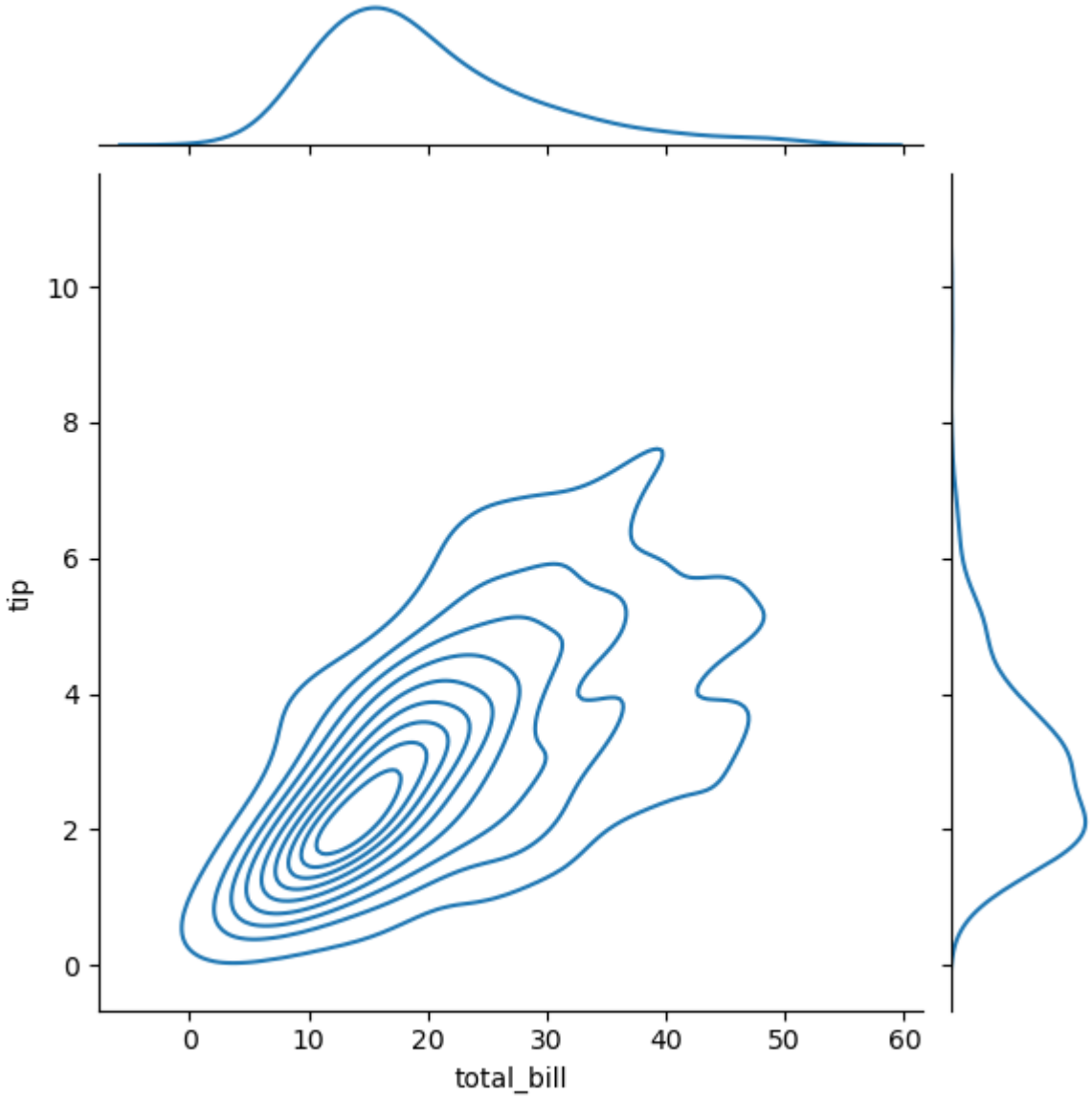
In [74]:
```
sns.jointplot(data=tips,x='total_bill',y='tip')
```

Out[74]:
```
<seaborn.axisgrid.JointGrid at 0x1e87b78ad30>
```



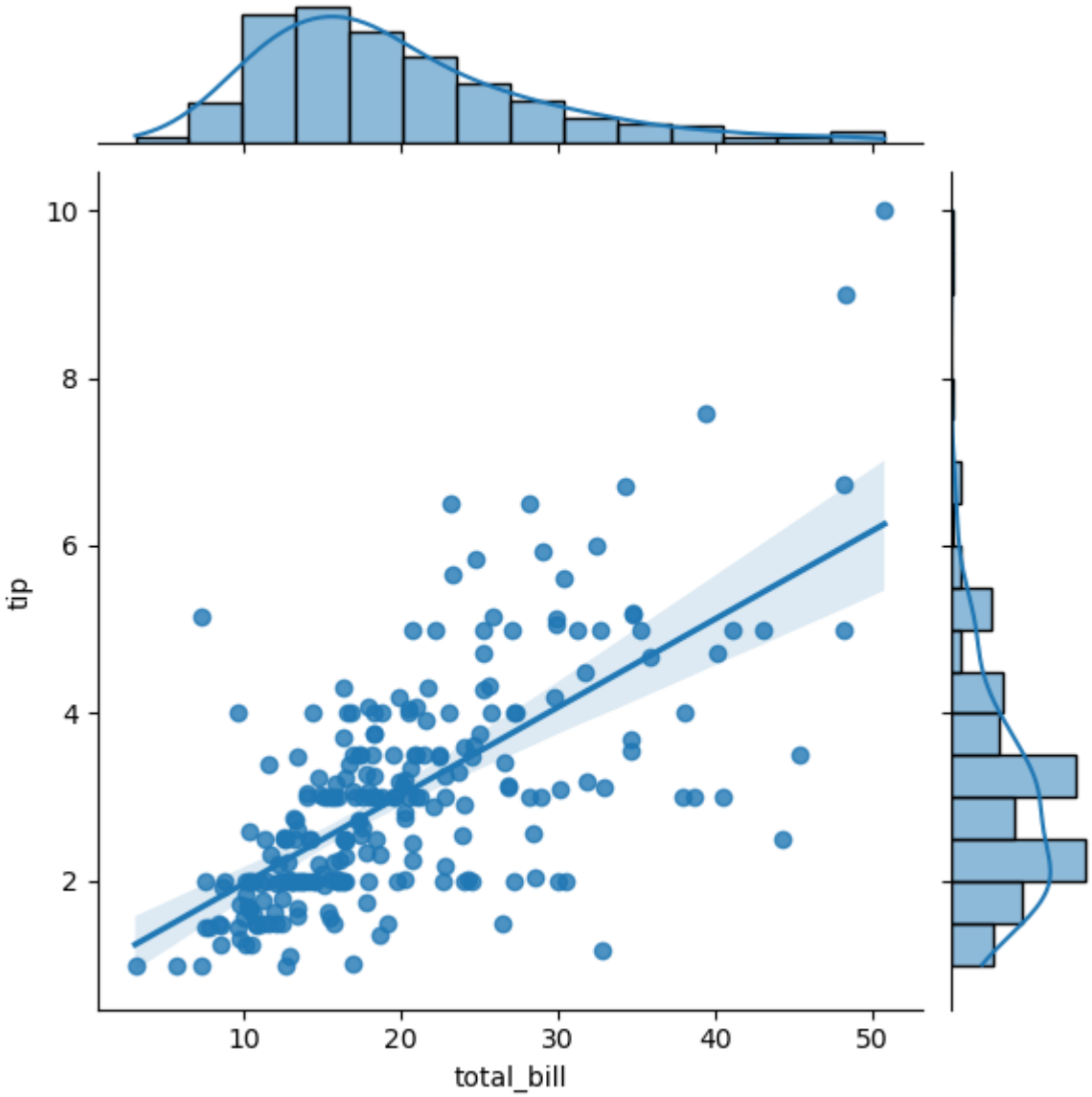
In [78]:
```
# kind

sns.jointplot(data=tips,x='total_bill',y='tip',kind= 'kde' )
```

Out[78]: <seaborn.axisgrid.JointGrid at 0x1e8768a06a0>



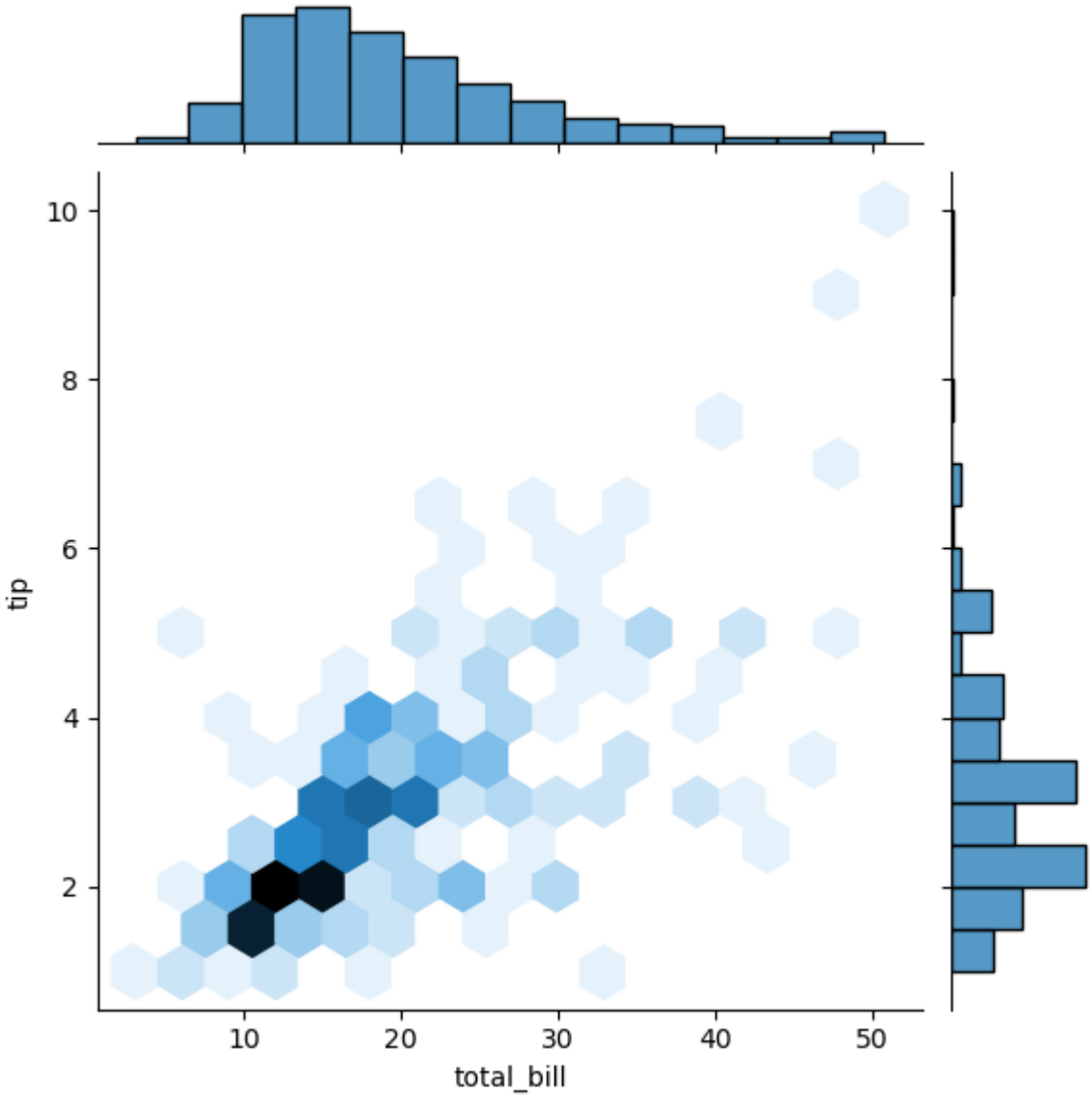
In [83]:
```
#regression line

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

Out[83]:
```
<seaborn.axisgrid.JointGrid at 0x1e87de2cac0>
```



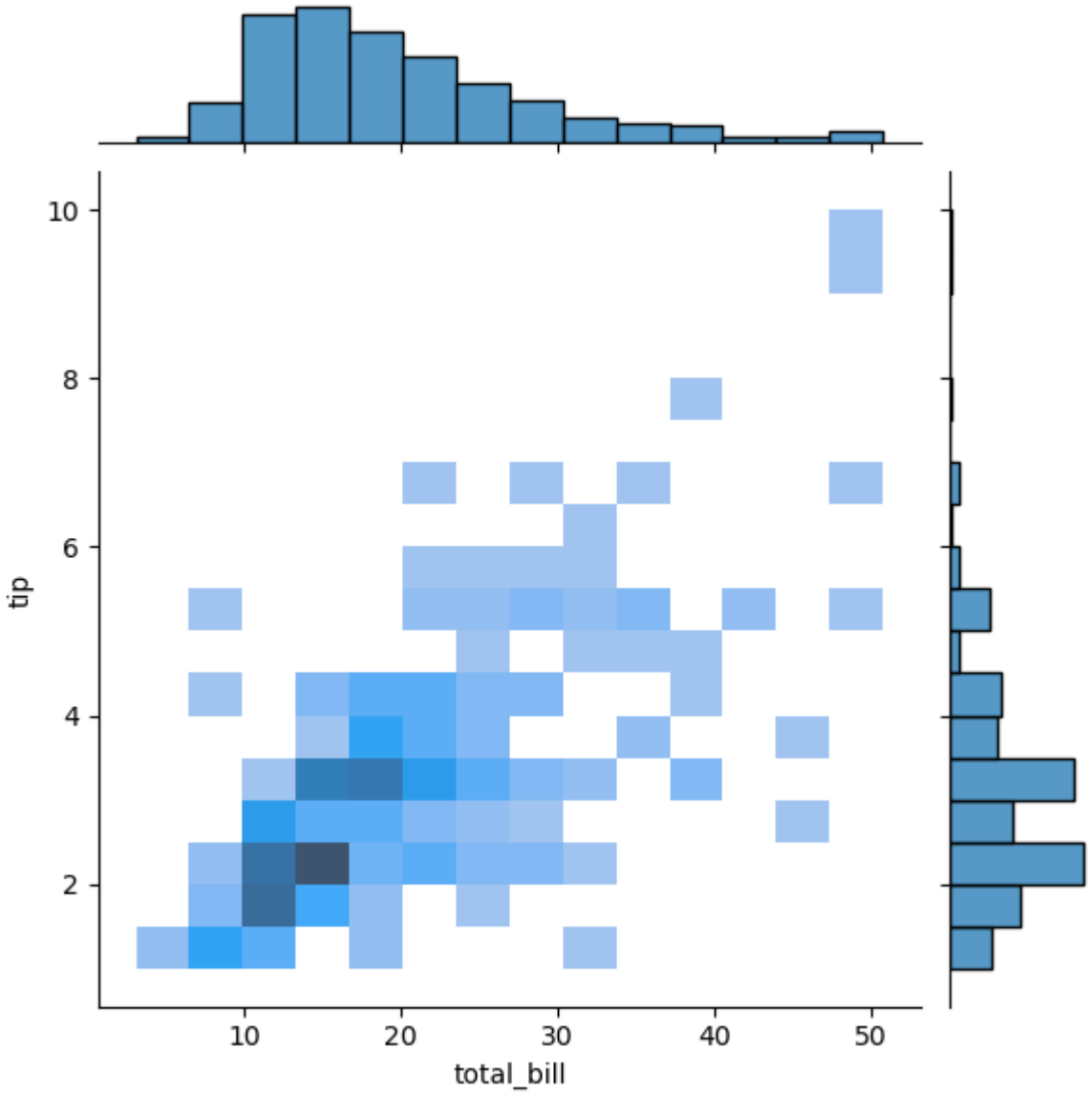
In [82]:
```
# hex

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

Out[82]: <seaborn.axisgrid.JointGrid at 0x1e87d97b370>



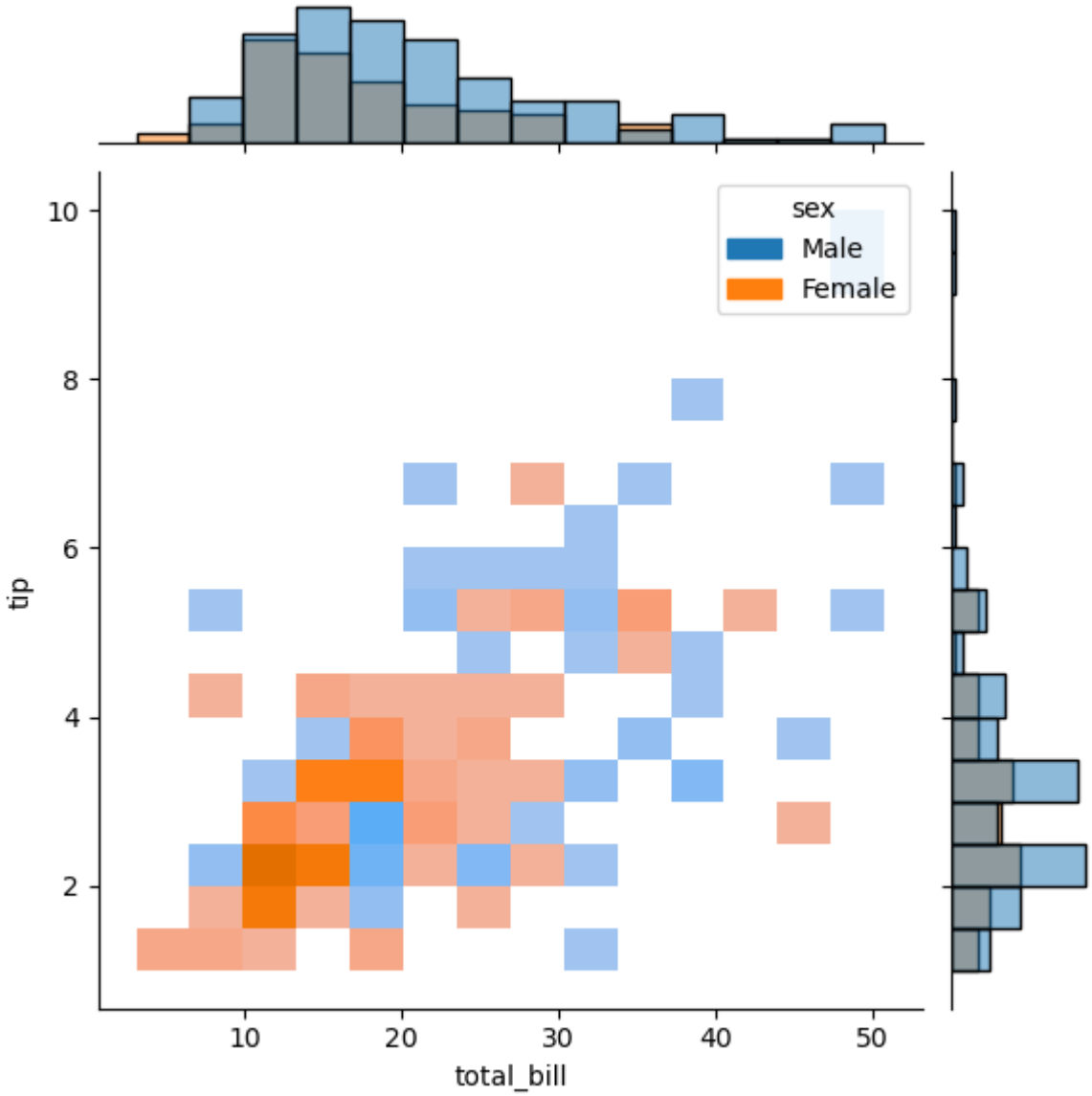
In [79]:
```
# kind -2d histogram + 1 histogram

sns.jointplot(data=tips,x='total_bill',y='tip',kind='hist' )
```

Out[79]: <seaborn.axisgrid.JointGrid at 0x1e877fe9250>



In [76]:
```
# hue

sns.jointplot(data=tips,x='total_bill',y='tip',kind='hist',hue='sex')
```

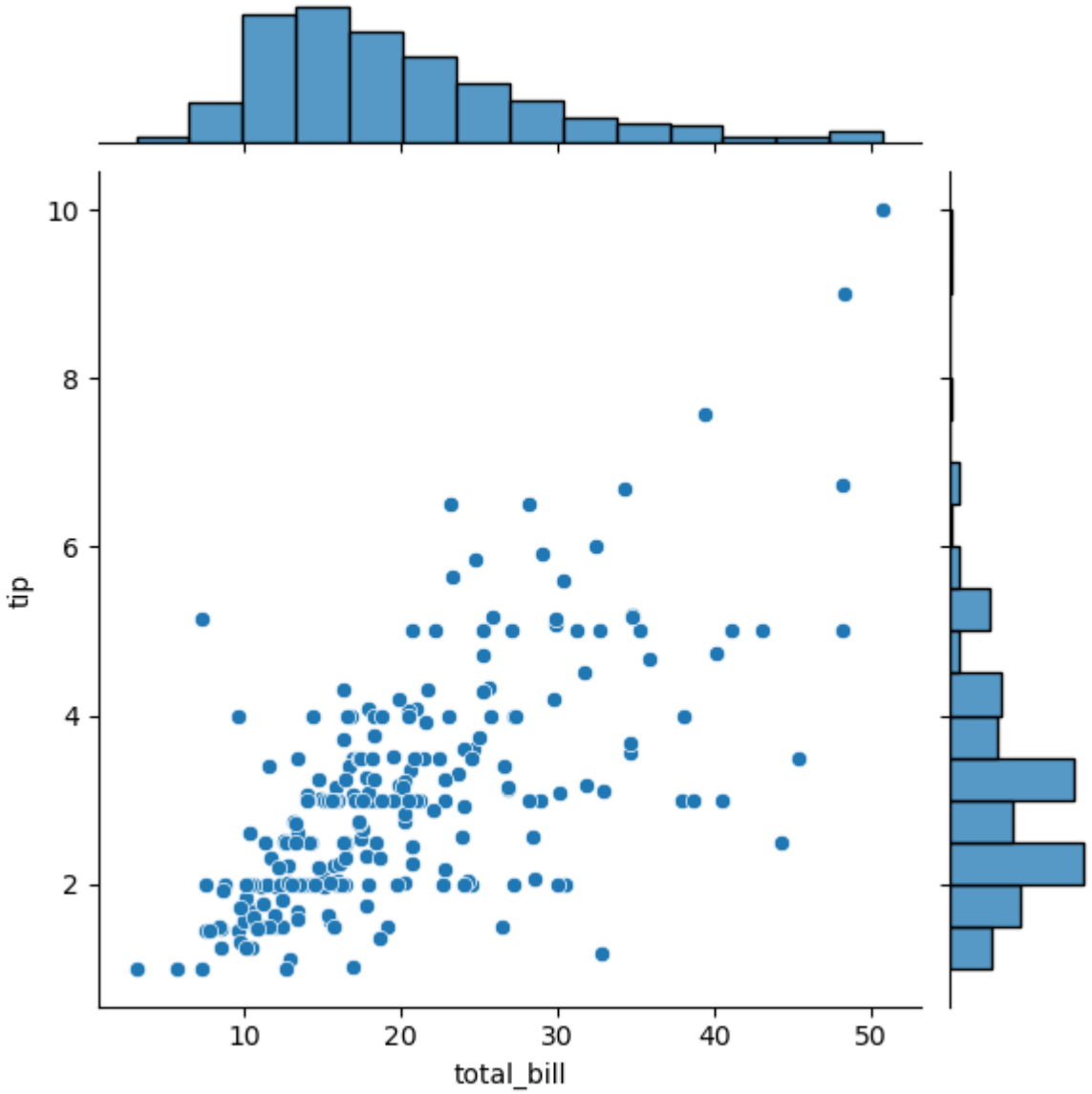
Out[76]:
```
<seaborn.axisgrid.JointGrid at 0x1e878736550>
```



Jointgrid

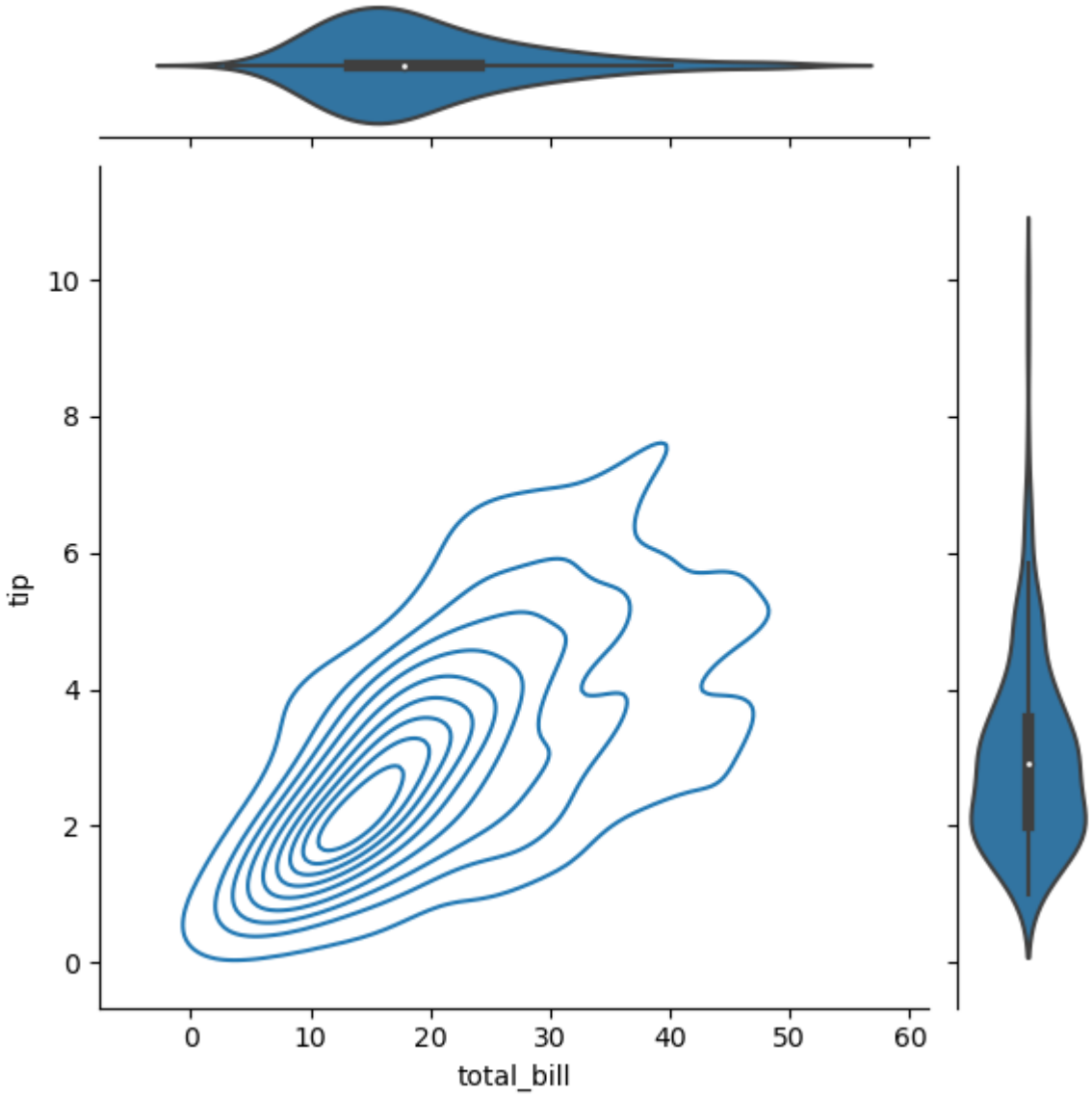
In [84]:
```
# we can change plots as per our requirements

g = sns.JointGrid(data=tips,x='total_bill',y='tip')

g.plot(sns.scatterplot,sns.histplot)
```

Out[84]:
```
<seaborn.axisgrid.JointGrid at 0x1e87e238850>
```



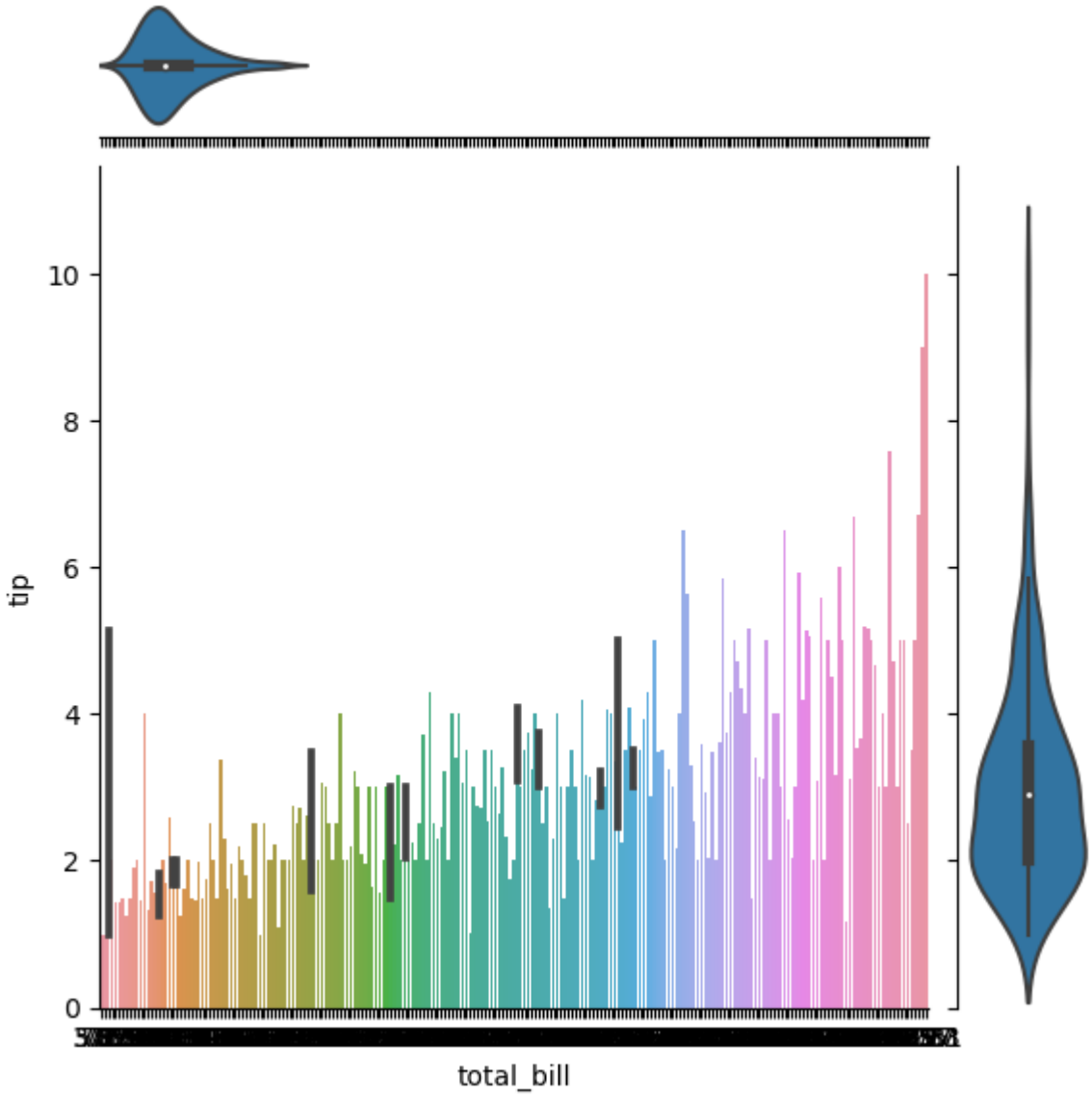
In [85]:
```
g = sns.JointGrid(data=tips,x='total_bill',y='tip')

g.plot(sns.kdeplot,sns.violinplot)
```

Out[85]:
```
<seaborn.axisgrid.JointGrid at 0x1e87e7d5a90>
```



In [87]:
```
g = sns.JointGrid(data=tips,x='total_bill',y='tip')

g.plot(sns.barplot,sns.violinplot)
```

Out[87]: <seaborn.axisgrid.JointGrid at 0x1e87ec37040>



Utility Functions

In [88]:
```
# get dataset names

sns.get_dataset_names()
```

Out[88]:
```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'dowjones',
 'exercise',
 'flights',
 'fmri',
 'geyser',
 'glue',
 'healthexp',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'seaice',
 'taxis',
 'tips',
 'titanic']
```

In [91]:
```
# load dataset
sns.load_dataset('planets')
```

Out[91]:

	method	number	orbital_period	mass	distance	year
0	Radial Velocity	1	269.300000	7.10	77.40	2006
1	Radial Velocity	1	874.774000	2.21	56.95	2008
2	Radial Velocity	1	763.000000	2.60	19.84	2011
3	Radial Velocity	1	326.030000	19.40	110.62	2007
4	Radial Velocity	1	516.220000	10.50	119.47	2009
...	...	...	...	...	...	...
1030	Transit	1	3.941507	NaN	172.00	2006
1031	Transit	1	2.615864	NaN	148.00	2007
1032	Transit	1	3.191524	NaN	174.00	2007
1033	Transit	1	4.125083	NaN	293.00	2008
1034	Transit	1	4.187757	NaN	260.00	2008

1035 rows × 6 columns

In [92]:
```
sns.load_dataset('diamonds')
```

Out[92]:

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
...	...	...	...	...	...	...	...	...	...	...
53935	0.72	Ideal	D	SI1	60.8	57.0	2757	5.75	5.76	3.50
53936	0.72	Good	D	SI1	63.1	55.0	2757	5.69	5.75	3.61
53937	0.70	Very Good	D	SI1	62.8	60.0	2757	5.66	5.68	3.56
53938	0.86	Premium	H	SI2	61.0	58.0	2757	6.15	6.12	3.74
53939	0.75	Ideal	D	SI2	62.2	55.0	2757	5.83	5.87	3.64

53940 rows × 10 columns

In [93]:
```
sns.load_dataset('healthexp')
```

Out[93]:

	Year	Country	Spending_USD	Life_Expectancy
0	1970	Germany	252.311	70.6
1	1970	France	192.143	72.2
2	1970	Great Britain	123.993	71.9
3	1970	Japan	150.437	72.0
4	1970	USA	326.961	70.9
...	...	...	...	...
269	2020	Germany	6938.983	81.1
270	2020	France	5468.418	82.3
271	2020	Great Britain	5018.700	80.4
272	2020	Japan	4665.641	84.7
273	2020	USA	11859.179	77.0

274 rows × 4 columns

In [ ]:
```

```