Seaborn

Seaborn is a library that uses Matplotlib underneath to plot graphs. It will be used to visualize random distributions.

Why Seaborn?

- provides a layer of abstraction hence simpler to use
- better aesthetics
- more graphs included

Seaborn Roadmap

Types of Functions

- Axes Level
- Figure Level

In the context of data visualization, "axes level" and "figure level" refer to different levels of control and customization available to you when creating plots and charts using libraries like Matplotlib or Seaborn.

1. **Axes Level**: The axes level refers to the level of control and customization at the individual plot level. In this level, you manipulate the properties of the axes objects directly. An axes object represents an individual plot within a figure and includes elements such as the x-axis, y-axis, data points, labels, and legends. By accessing and modifying these properties, you can customize the appearance and behavior of the specific plot.

Axes level refers to customizing individual plots within a figure, such as changing colors, labels, and limits.

2. **Figure Level**: The figure level refers to the level of control and customization at the overall figure level. A figure represents the entire window or canvas where one or more plots are displayed. It includes one or more axes objects. In this level, you manipulate the properties of the figure itself.

Figure level refers to customizing the overall layout of the entire figure, including size, arrangement of subplots, and global titles.

By distinguishing between axes level and figure level customization, you can have fine-grained control over individual plots while also being able to modify the global properties of the entire figure. This allows you to create complex and customized visualizations to effectively communicate your data.

Main Classification

- Relational Plot
- Distribution Plot
- Categorical Plot
- Regression Plot
- Matrix Plot
- Multiplots

1. Relational Plot

- to see the statistical relation between 2 or more variables.
- Bivariate Analysis

Plots under this section

- scatterplot
- lineplot

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

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

In [3]:
```
tips
```

Out[3]:

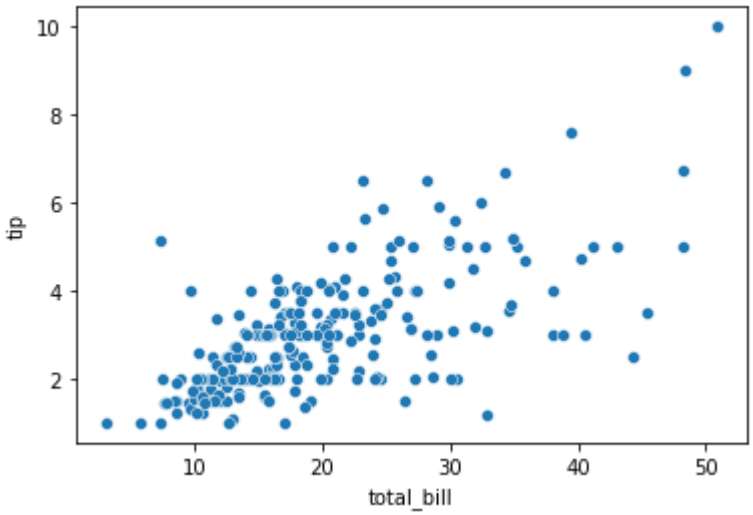
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	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

Scatter plot (Axes level)

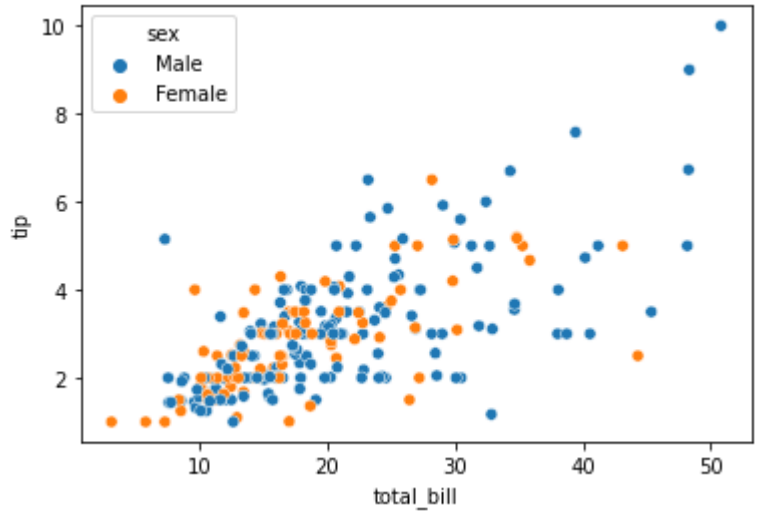
In [4]:
```
# Axes Level Function

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

Out[4]:
```
<AxesSubplot:xlabel='total_bill', ylabel='tip'>
```



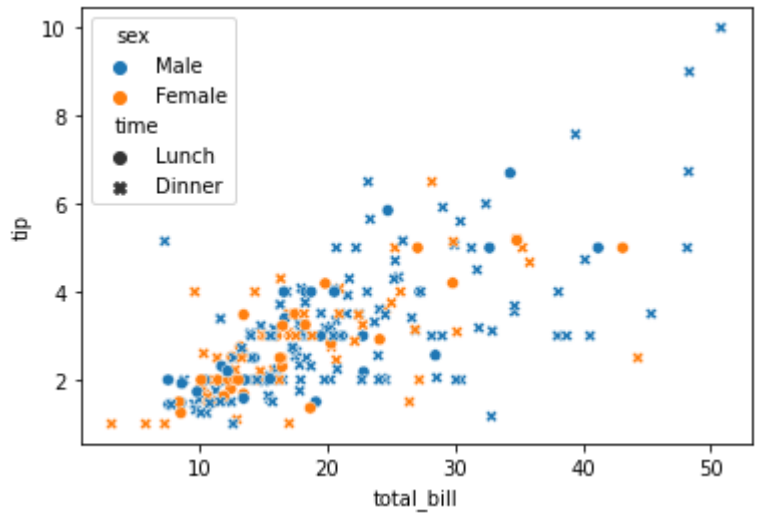
In [5]:
```
# hue

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

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



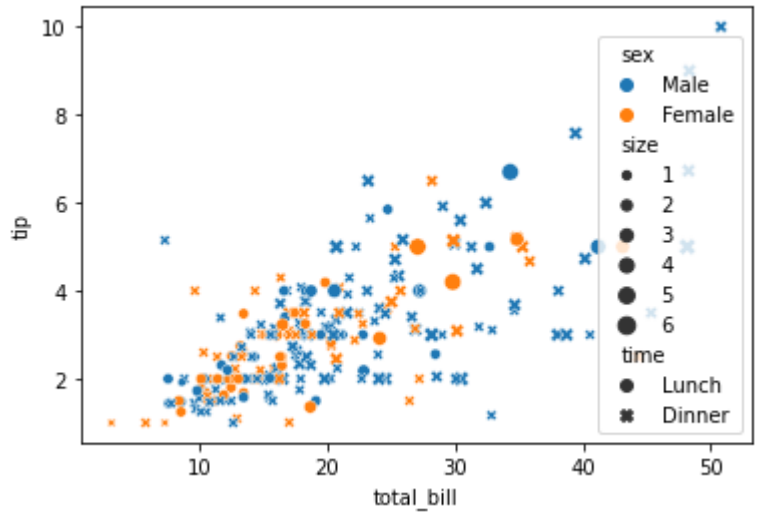
In [6]:
```
# Style

sns.scatterplot(data=tips , x = 'total_bill' ,
                y ='tip', hue ='sex', style ='time')
```

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



In [7]:
```
# Size

sns.scatterplot(data=tips , x = 'total_bill' ,
                y ='tip', hue ='sex', style ='time',
                size ='size')
```

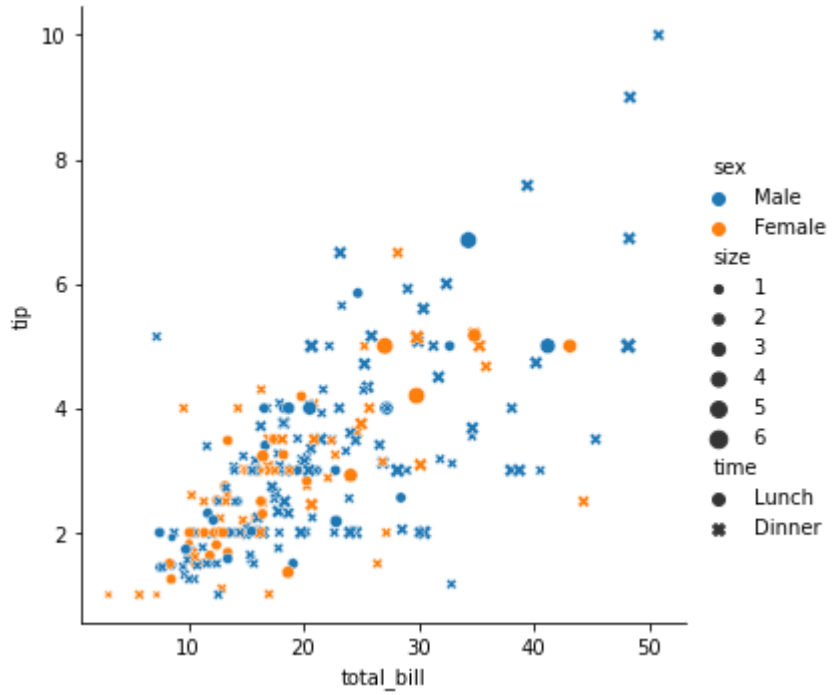
Out[7]: <AxesSubplot:xlabel='total_bill', ylabel='tip'>



relplot (figure level)

In [8]:
```
## Figure level function --- Square Shape

sns.relplot(data=tips , x = 'total_bill' ,y ='tip', kind ='scatter',
            hue ='sex', style ='time',
                size ='size')
```

Out[8]: <seaborn.axisgrid.FacetGrid at 0x261e410de80>



Line plot

In [9]:
```
gap =px.data.gapminder()
```

In [10]:
```
gap
```

Out[10]:

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_num
0	Afghanistan	Asia	1952	28.801	8425333	779.445314	AFG	4
1	Afghanistan	Asia	1957	30.332	9240934	820.853030	AFG	4
2	Afghanistan	Asia	1962	31.997	10267083	853.100710	AFG	4
3	Afghanistan	Asia	1967	34.020	11537966	836.197138	AFG	4
4	Afghanistan	Asia	1972	36.088	13079460	739.981106	AFG	4
...	...	...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306	ZWE	716
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786	ZWE	716
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960	ZWE	716
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623	ZWE	716
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298	ZWE	716

1704 rows × 8 columns

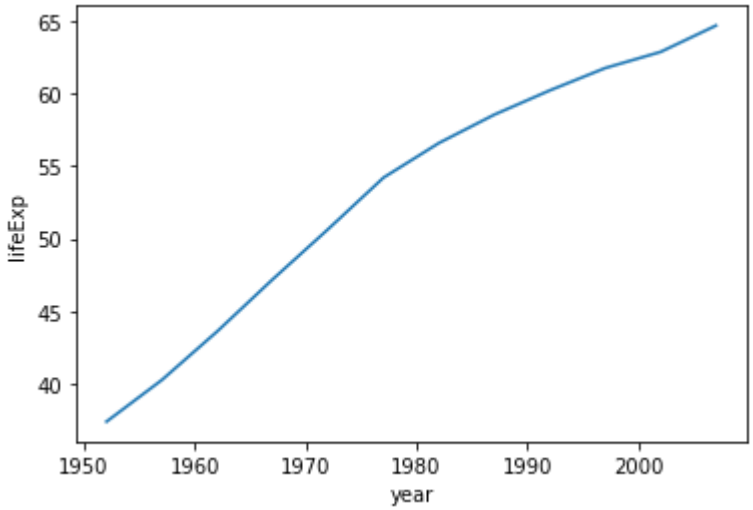
In [16]:
```
temp_df = gap[gap['country']=='India']
temp_df
```

Out[16]:

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_num
696	India	Asia	1952	37.373	372000000	546.565749	IND	356
697	India	Asia	1957	40.249	409000000	590.061996	IND	356
698	India	Asia	1962	43.605	454000000	658.347151	IND	356
699	India	Asia	1967	47.193	506000000	700.770611	IND	356
700	India	Asia	1972	50.651	567000000	724.032527	IND	356
701	India	Asia	1977	54.208	634000000	813.337323	IND	356
702	India	Asia	1982	56.596	708000000	855.723538	IND	356
703	India	Asia	1987	58.553	788000000	976.512676	IND	356
704	India	Asia	1992	60.223	872000000	1164.406809	IND	356
705	India	Asia	1997	61.765	959000000	1458.817442	IND	356
706	India	Asia	2002	62.879	1034172547	1746.769454	IND	356
707	India	Asia	2007	64.698	1110396331	2452.210407	IND	356

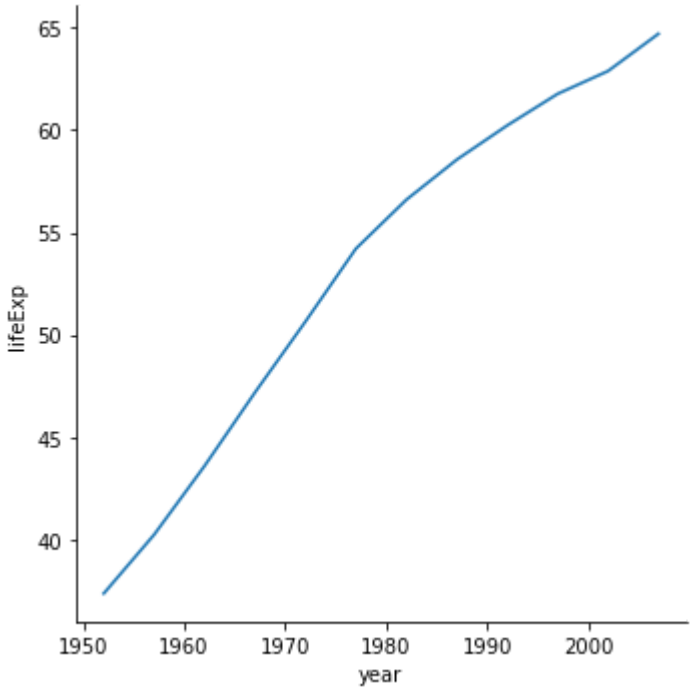
In [17]:
```
# Axes Level Function

sns.lineplot(data =temp_df ,x ='year' ,y='lifeExp')
```

Out[17]:
```
<AxesSubplot:xlabel='year', ylabel='lifeExp'>
```



In [20]:
```
# Figure Level function --> Square

sns.relplot(data =temp_df ,x ='year' ,y='lifeExp' , kind ='line')
```

Out[20]:
```
<seaborn.axisgrid.FacetGrid at 0x261e42b75b0>
```



In [22]:
```
# Hue

df = gap[gap['country'].isin (['India','Pakistan','China'])]
```

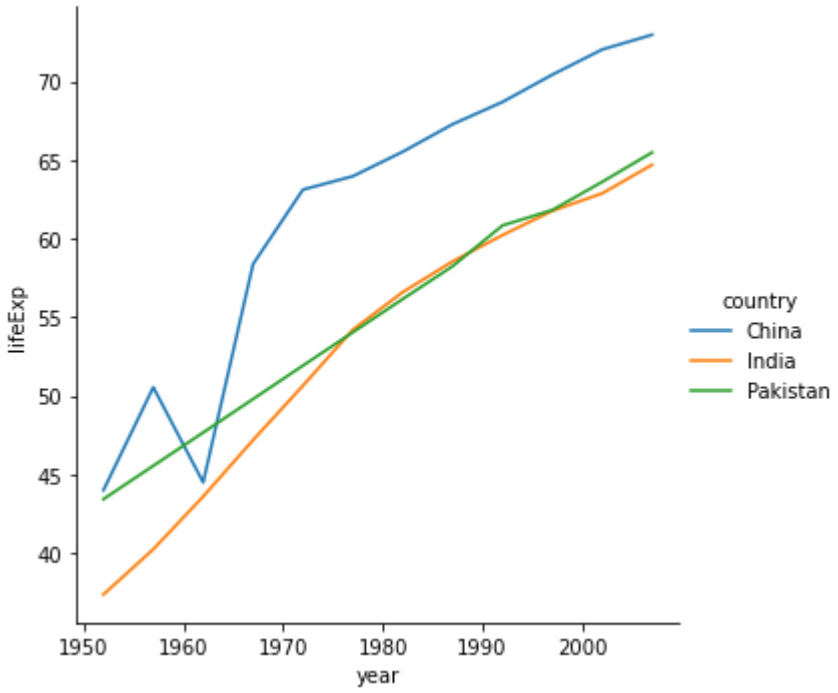
In [25]:
```
df.sample(5)
```

Out[25]:

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_num
702	India	Asia	1982	56.59600	708000000	855.723538	IND	356
698	India	Asia	1962	43.60500	454000000	658.347151	IND	356
289	China	Asia	1957	50.54896	637408000	575.987001	CHN	156
1174	Pakistan	Asia	2002	63.61000	153403524	2092.712441	PAK	586
706	India	Asia	2002	62.87900	1034172547	1746.769454	IND	356

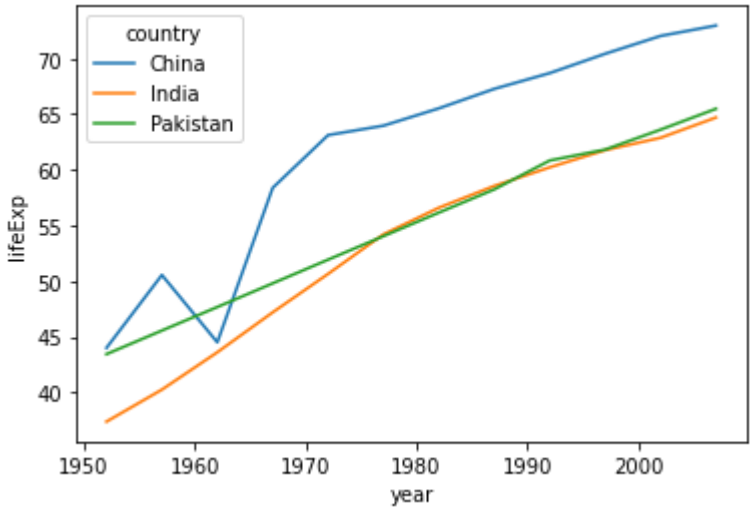
In [27]:
```
# Figure level function

sns.relplot(data=df , x ='year' , y ='lifeExp' ,
            hue='country' , kind ='line')
```

Out[27]:
```
<seaborn.axisgrid.FacetGrid at 0x261e650b040>
```



In [29]:
```
# axes level function

sns.lineplot(data=df , x ='year' , y ='lifeExp' , hue='country' )
```

Out[29]:
```
<AxesSubplot:xlabel='year', ylabel='lifeExp'>
```



In [31]:
```
df1 = gap[gap['country'].isin (['India','Brazil','Germany'])]
```

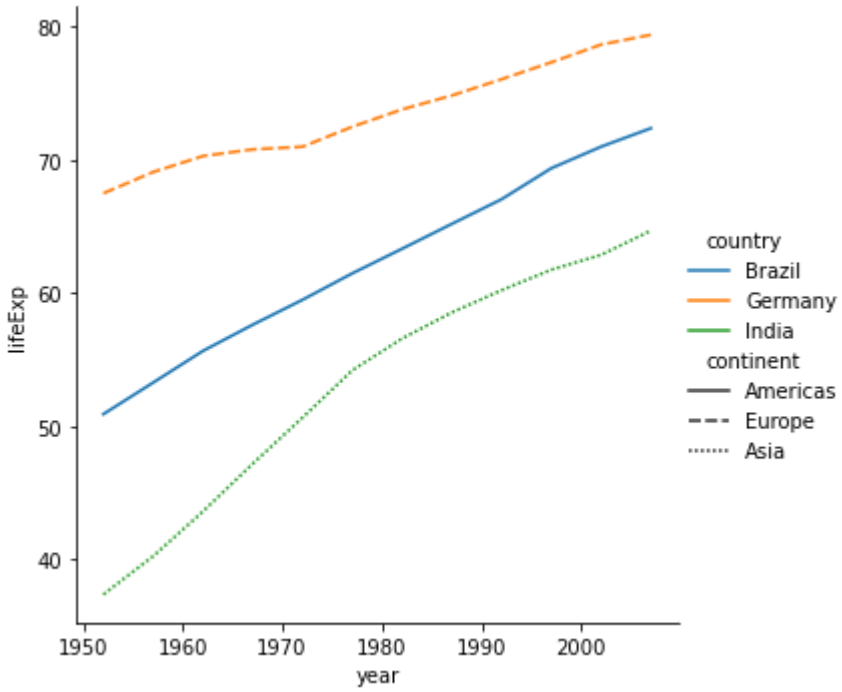
In [33]:
```
df1.sample(5)
```

Out[33]:

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_num
707	India	Asia	2007	64.698	1110396331	2452.210407	IND	356
175	Brazil	Americas	1987	65.205	142938076	7807.095818	BRA	76
696	India	Asia	1952	37.373	372000000	546.565749	IND	356
173	Brazil	Americas	1977	61.489	114313951	6660.118654	BRA	76
571	Germany	Europe	1987	74.847	77718298	24639.185660	DEU	276

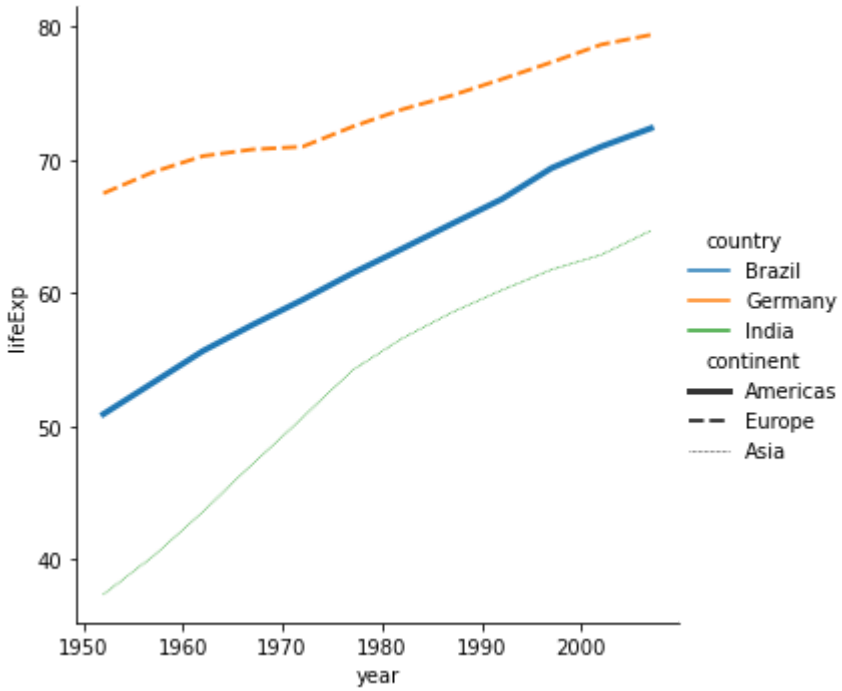
In [37]:
```
# Style

sns.relplot(data=df , x ='year' , y ='lifeExp' ,
            hue='country' , kind ='line' ,style='continent')
```

Out[37]: <seaborn.axisgrid.FacetGrid at 0x261e72ab790>



In [38]:
```
# Size

sns.relplot(data=df , x ='year' , y ='lifeExp' ,
            hue='country' , kind ='line' ,
            style='continent',size ='continent')
```

Out[38]:
```
<seaborn.axisgrid.FacetGrid at 0x261e70a1700>
```



facet plot

- facet plot - figure level function -> work with relplot
- it will not work with scatterplot and lineplot

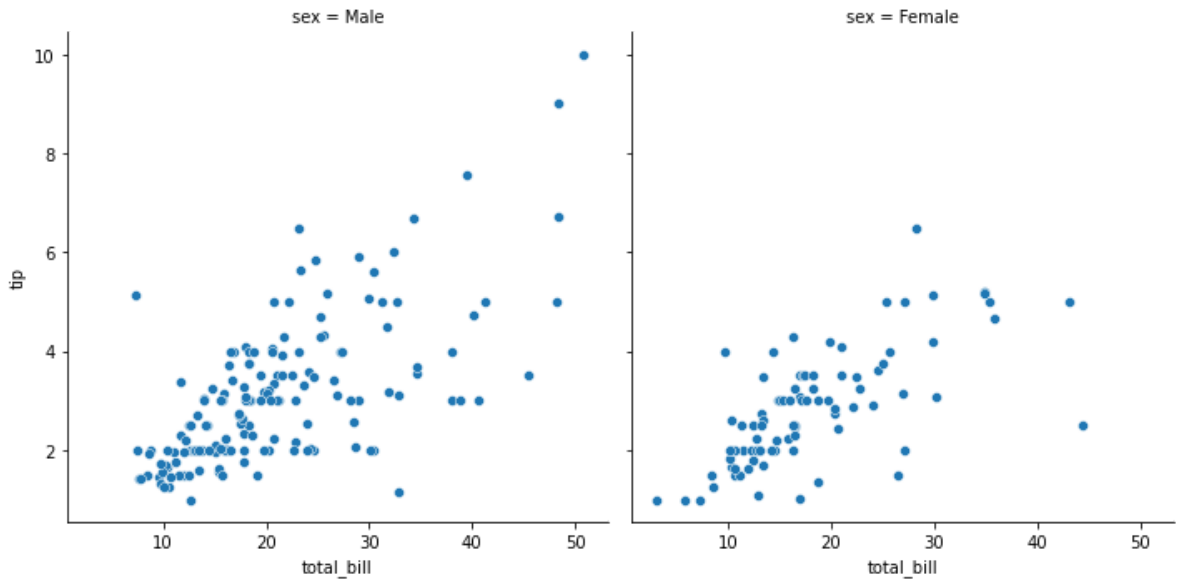
In [41]:
```
tips.sample(2)
```

Out[41]:

	total_bill	tip	sex	smoker	day	time	size
145	8.35	1.5	Female	No	Thur	Lunch	2
36	16.31	2.0	Male	No	Sat	Dinner	3

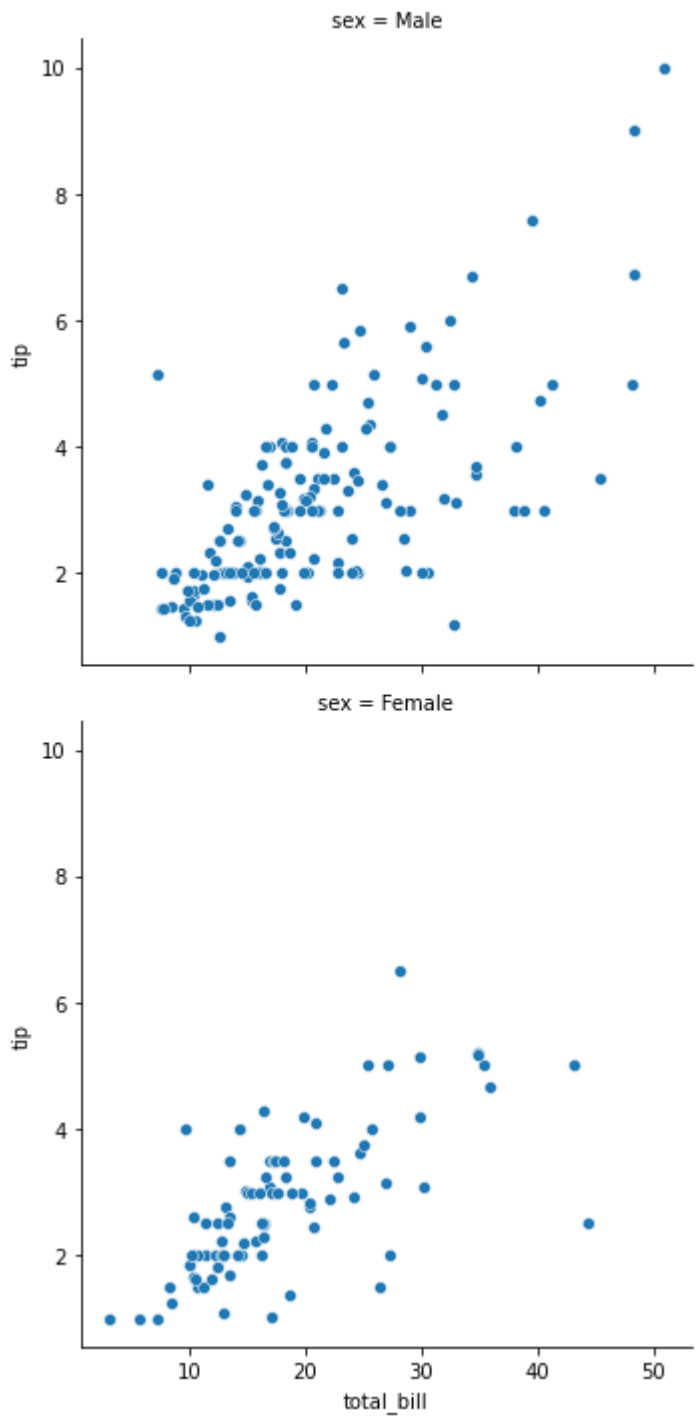
In [42]:
```
# For particular category , we can plot graphs

sns.relplot(data=tips , x = 'total_bill' ,y ='tip', col ='sex') # col
```

Out[42]:
```
<seaborn.axisgrid.FacetGrid at 0x261e74c5f10>
```



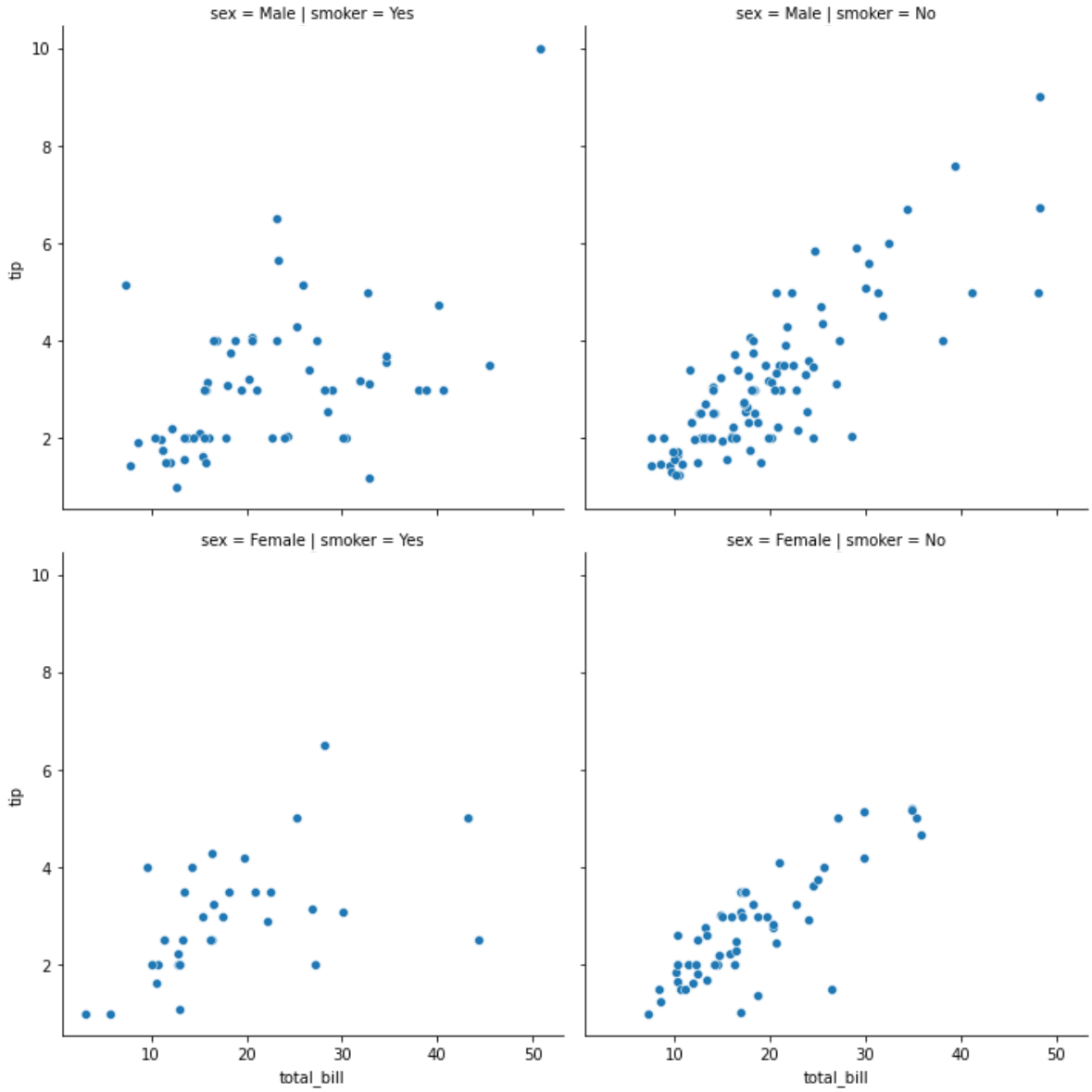
In [43]:
```
# row

sns.relplot(data=tips , x = 'total_bill' ,y ='tip', row='sex')
```

Out[43]: <seaborn.axisgrid.FacetGrid at 0x261e64ffe50>



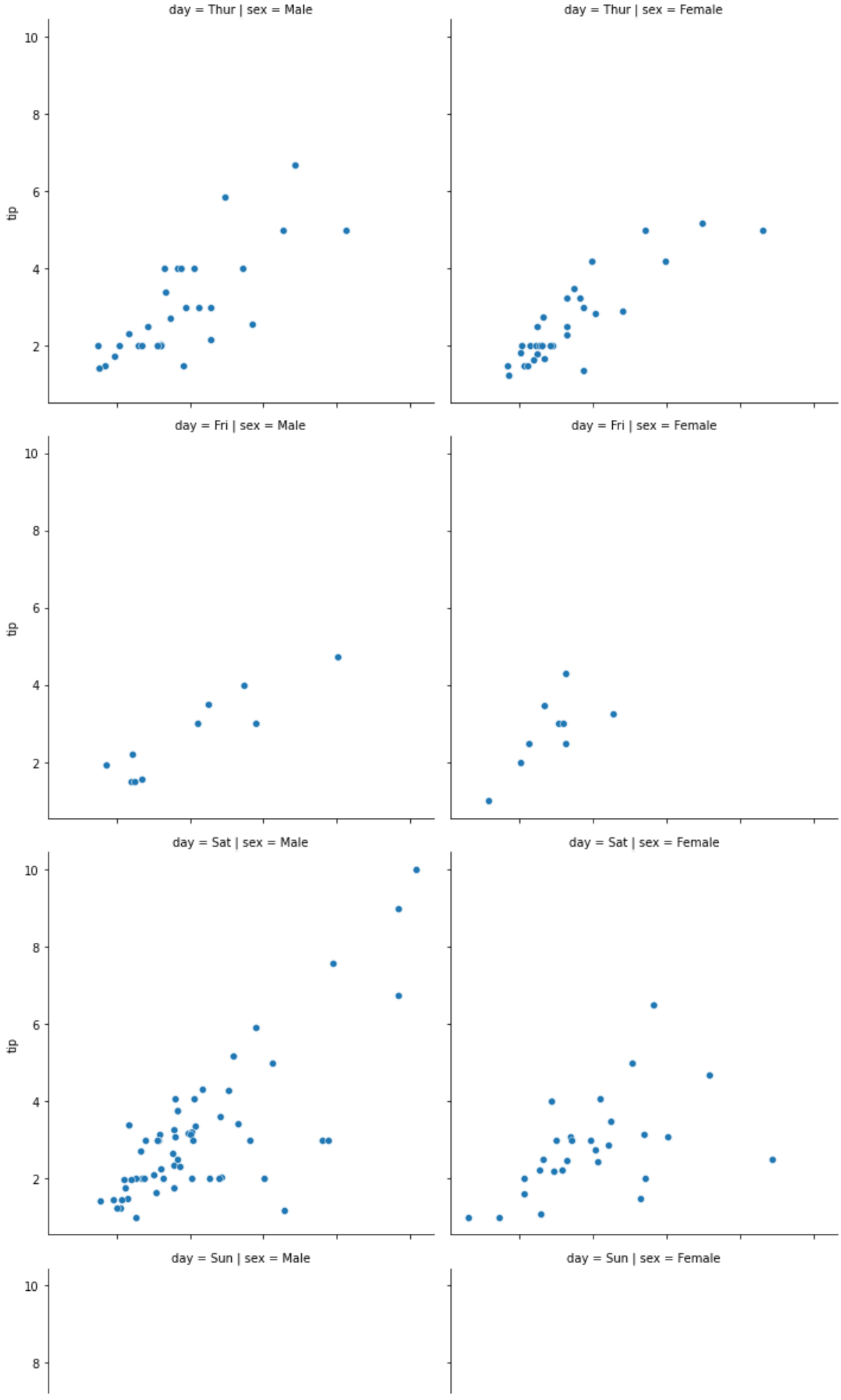
In [45]:
```
# Both row , col

sns.relplot(data=tips , x = 'total_bill' ,y ='tip',
            col ='smoker', row='sex')
```

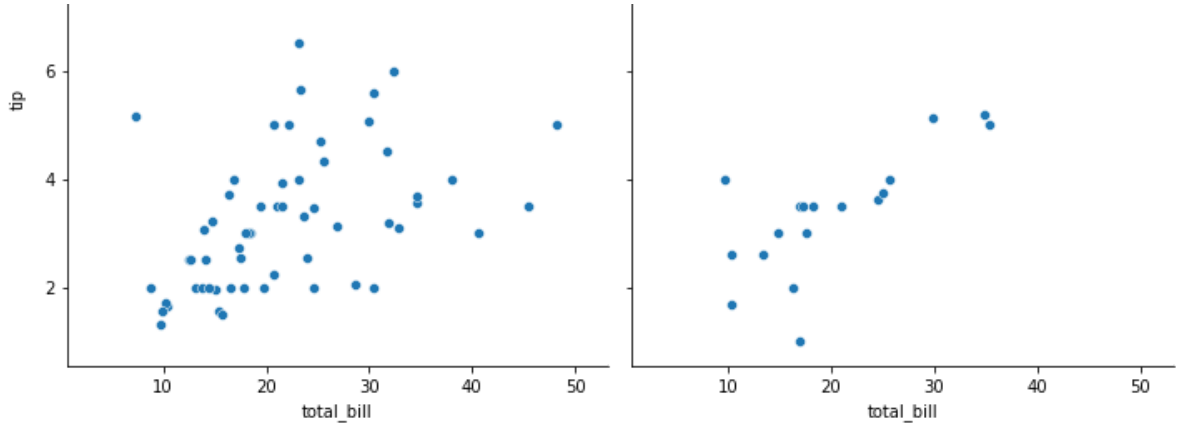
Out[45]: <seaborn.axisgrid.FacetGrid at 0x261e84e0040>



In [47]:
```
sns.relplot(data=tips , x = 'total_bill' ,y ='tip',
            col ='sex', row='day')
```

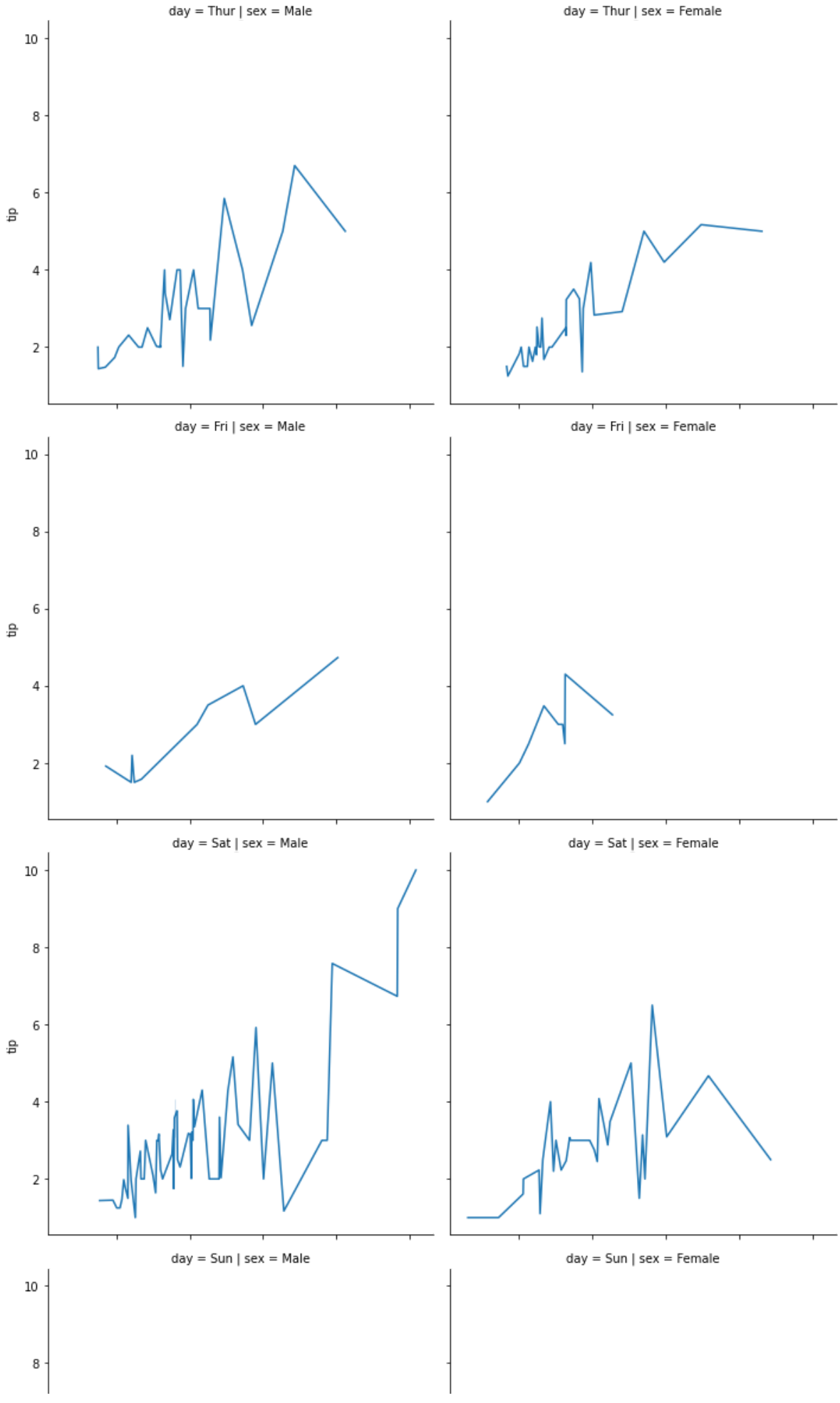
Out[47]:
```
<seaborn.axisgrid.FacetGrid at 0x261e85bd850>
```


day = Thur | sex = Male
day = Thur | sex = Female
day = Fri | sex = Male
day = Fri | sex = Female
day = Sat | sex = Male
day = Sat | sex = Female
day = Sun | sex = Male
day = Sun | sex = Female
tip
10
8
6
4
2


tip
6
4
2
10
20
30
40
50
total_bill
10
20
30
40
50
total_bill

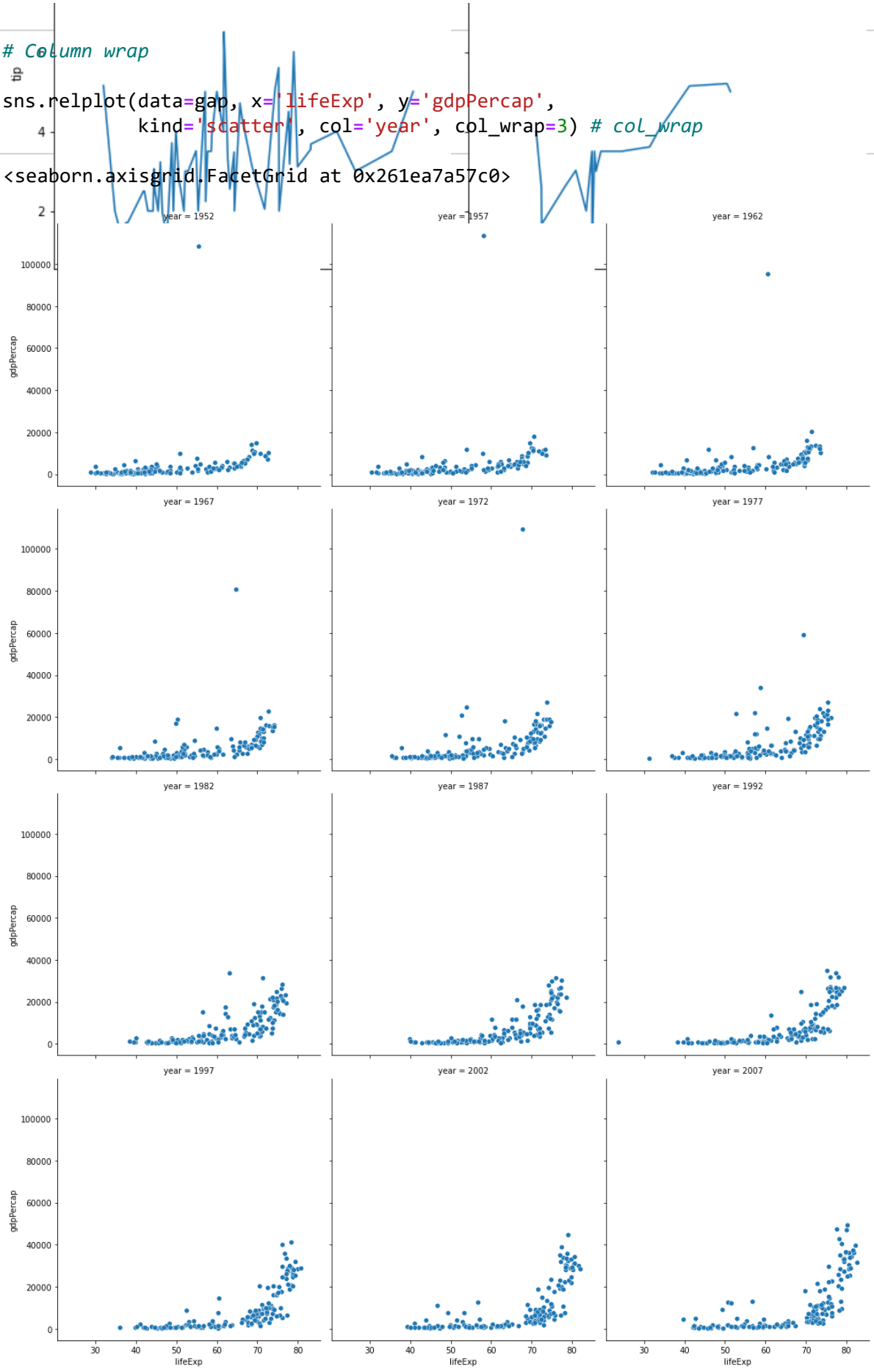
In [48]:
```
# facet on line plot

sns.relplot(data=tips , x = 'total_bill' ,y ='tip',
            col ='sex', row='day', kind ='line')
```

Out[48]:
```
<seaborn.axisgrid.FacetGrid at 0x261e6182460>
```


day = Thur | sex = Male
day = Thur | sex = Female
day = Fri | sex = Male
day = Fri | sex = Female
day = Sat | sex = Male
day = Sat | sex = Female
day = Sun | sex = Male
day = Sun | sex = Female
tip
10
8
6
4
2

In [49]:
```
# Column wrap

sns.relplot(data=gap, x='lifeExp', y='gdpPercap',
            kind='scatter', col='year', col_wrap=3) # col_wrap
```

Out[49]:
```
<seaborn.axisgrid.FacetGrid at 0x261ea7a57c0>
```



2. Distribution Plots

- used for univariate analysis
- used to find out the distribution
- Range of the observation
- Central Tendency
- is the data bimodal?
- Are there outliers?

Plots under distribution plot

- histplot
- kdeplot
- rugplot

In [50]:
```
# figure level -> displot
# axes level -> histplot -> kdeplot -> rugplot
```

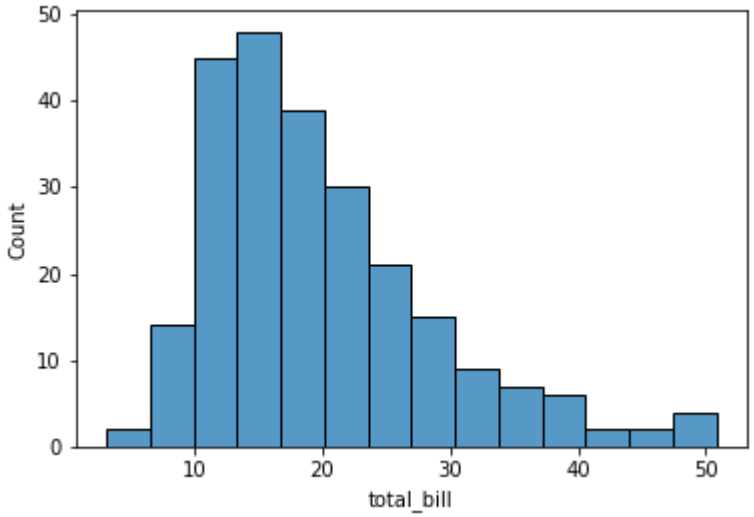
In [53]:
```
tips.sample(2)
```

Out[53]:

	total_bill	tip	sex	smoker	day	time	size
107	25.21	4.29	Male	Yes	Sat	Dinner	2
116	29.93	5.07	Male	No	Sun	Dinner	4

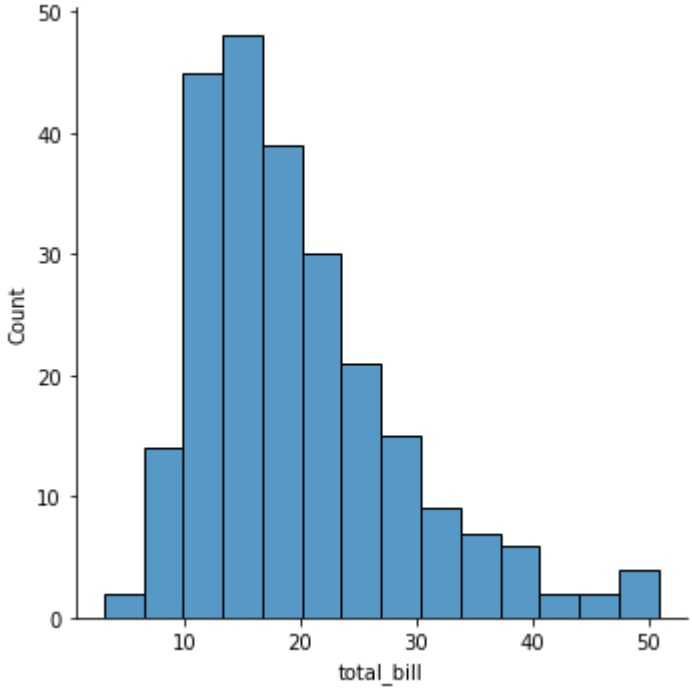
In [54]:
```
# Plotting on univarient Histogram

# Axes Level Function

sns.histplot(data =tips, x ='total_bill')
```

Out[54]:
```
<AxesSubplot:xlabel='total_bill', ylabel='Count'>
```



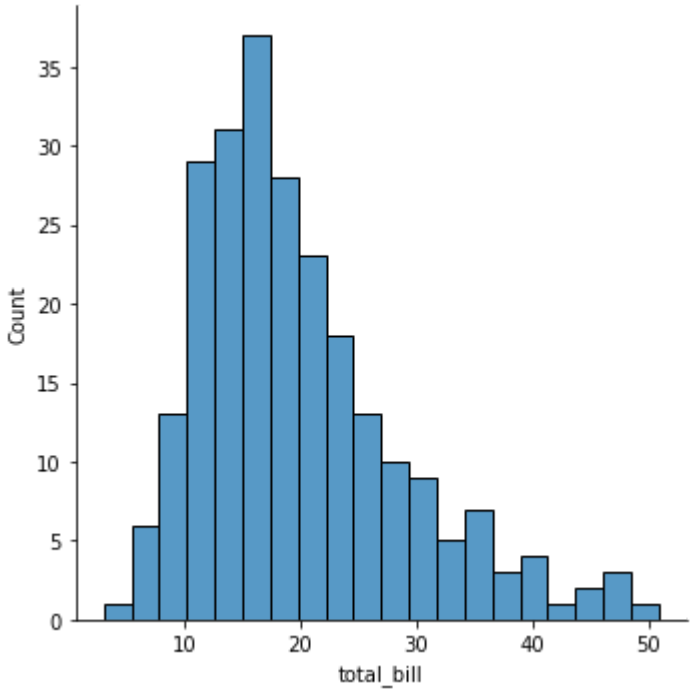
In [59]:
```
# Figure level Function

sns.displot(data =tips,x ='total_bill',kind ='hist')
```

Out[59]: <seaborn.axisgrid.FacetGrid at 0x261ecffbaf0>



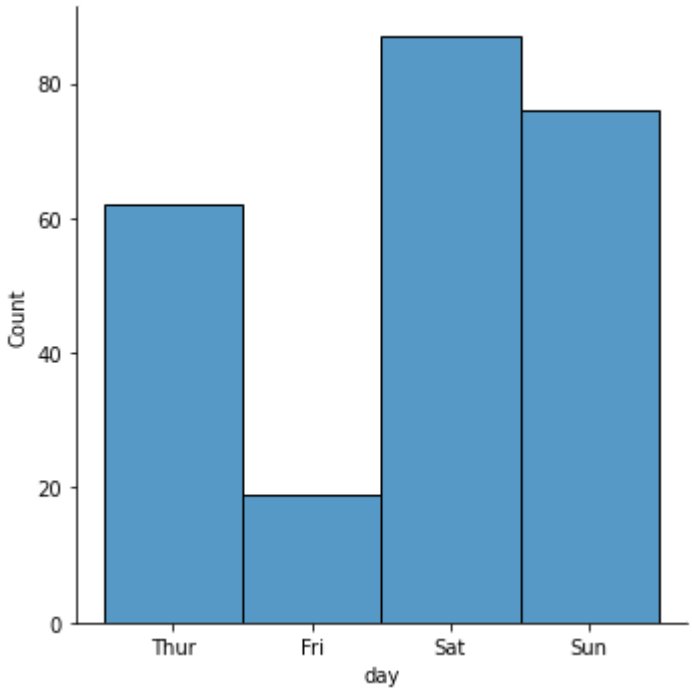
In [60]:
```
# Bins Parameter

sns.displot(data =tips,x ='total_bill',
            kind ='hist',bins =20)
```

Out[60]: <seaborn.axisgrid.FacetGrid at 0x261ed4732b0>



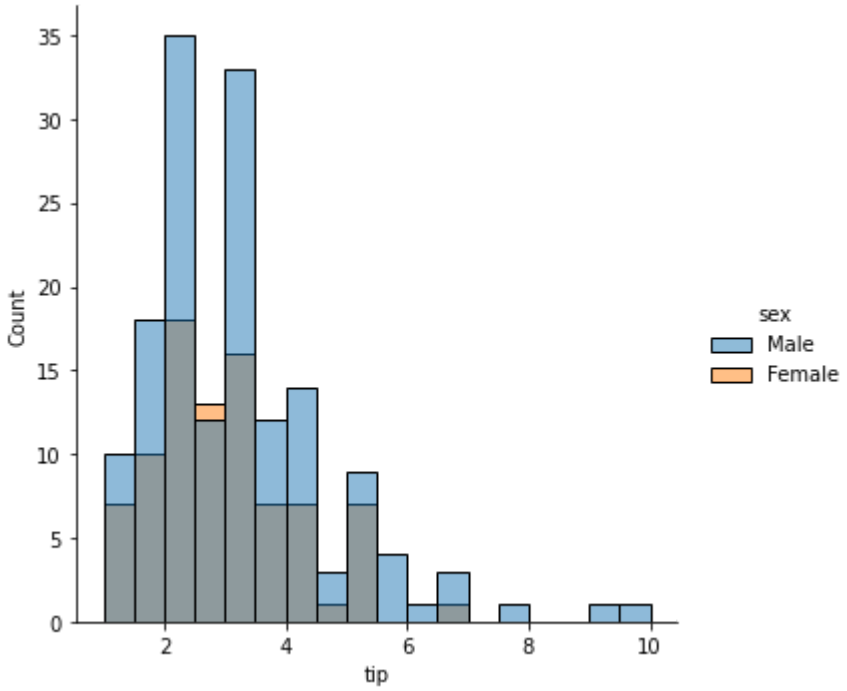
In [61]:
```
# It’s also possible to visualize the distribution of a categorical variable u

# Discrete bins are automatically set for categorical variables

# countplot
sns.displot(data=tips, x='day', kind='hist')
```

Out[61]: <seaborn.axisgrid.FacetGrid at 0x261ed4dbe20>



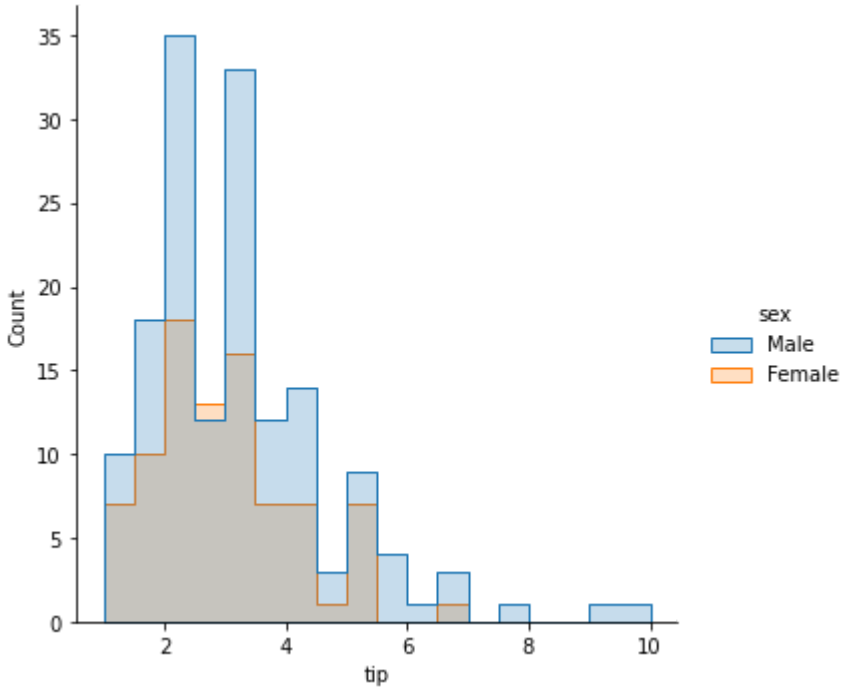
In [62]:
```
# hue parameter

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

Out[62]: <seaborn.axisgrid.FacetGrid at 0x261ec5c7040>



In [63]:
```
# element -> step

sns.displot(data=tips, x='tip', kind='hist',
            hue='sex',element='step')
```

Out[63]: <seaborn.axisgrid.FacetGrid at 0x261ed3065e0>



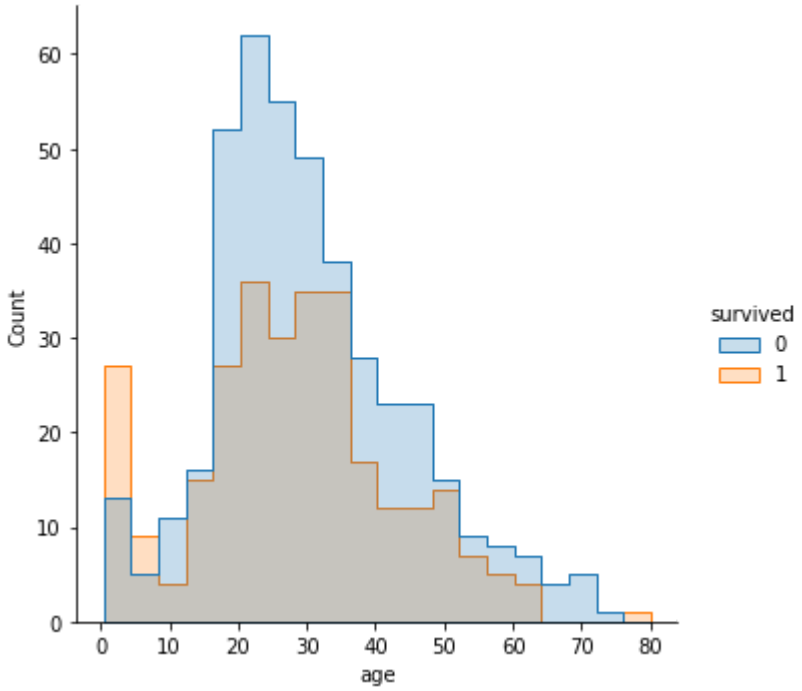
In [65]:
```
titanic = sns.load_dataset('titanic')
titanic.sample(2)
```

Out[65]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	dec
773	0	3	male	NaN	0	0	7.225	C	Third	man	True	Na
83	0	1	male	28.0	0	0	47.100	S	First	man	True	Na

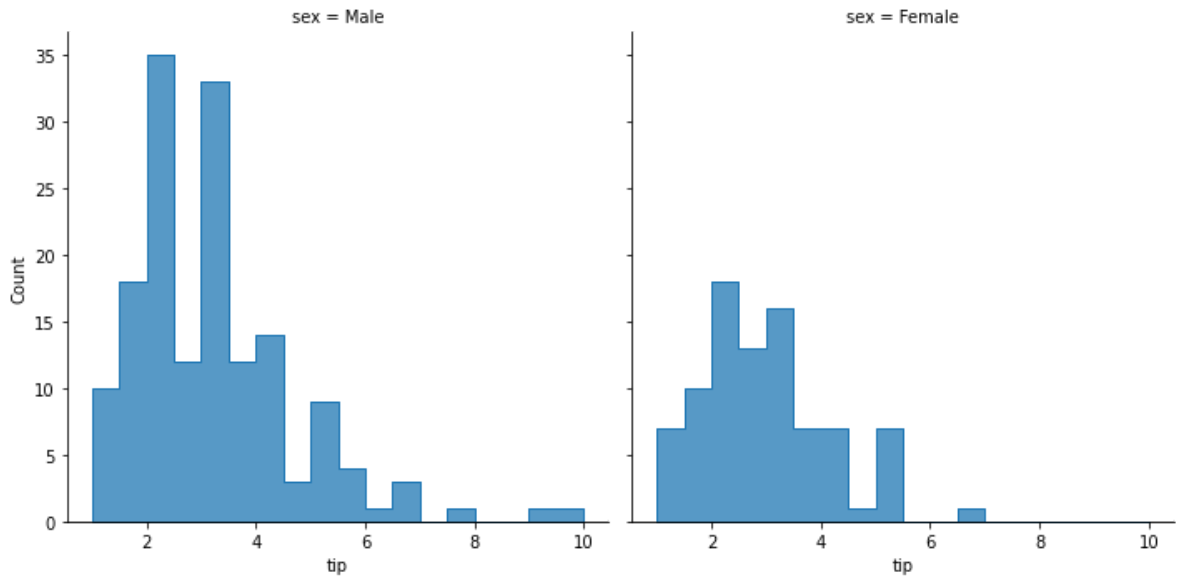
In [69]:
```
sns.displot(data =titanic,x ='age' , kind ='hist',
            element ='step' , hue ='survived')
```

Out[69]: <seaborn.axisgrid.FacetGrid at 0x261ecf182b0>



In [70]:
```
# faceting using col and row -> not work on histplot function because(axes lev

sns.displot(data=tips, x='tip', kind='hist',col='sex',element='step')
```

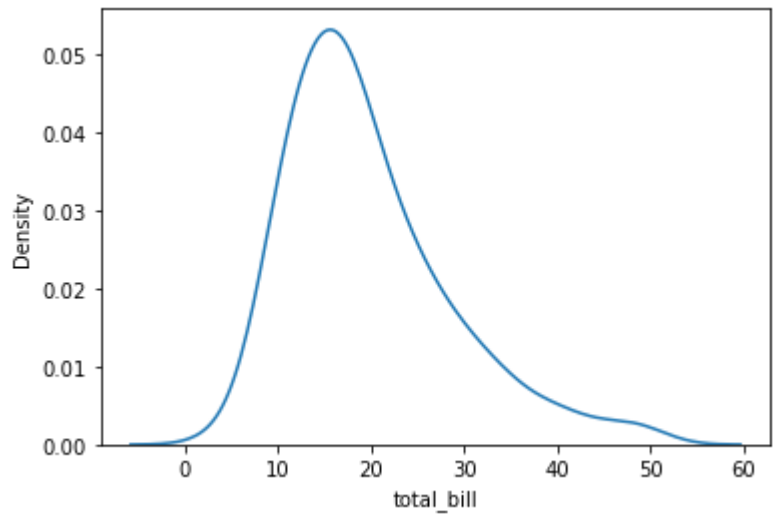
Out[70]: <seaborn.axisgrid.FacetGrid at 0x261ecf488e0>



kdeplot (kernel density Estimation)

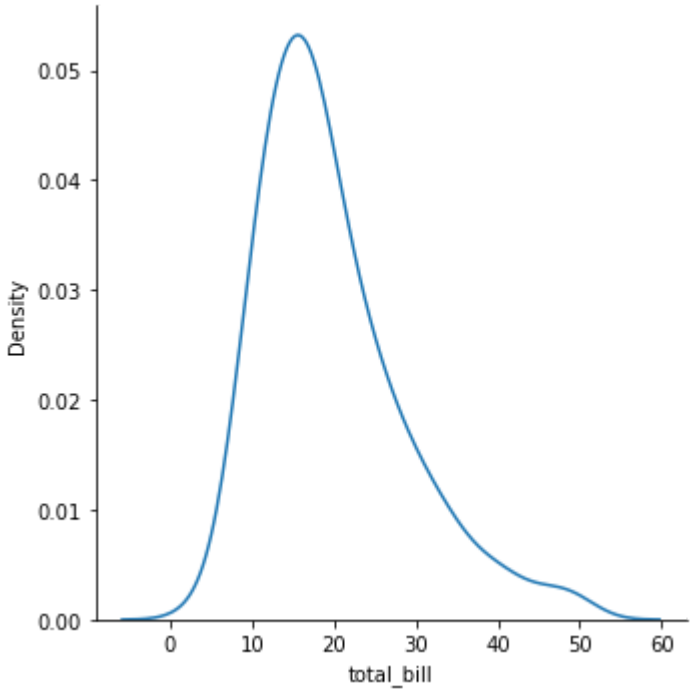
In [71]:
```
# Rather than using discrete bins, a KDE plot smooths the observations
#with a Gaussian kernel, producing a continuous density estimate

sns.kdeplot(data=tips,x='total_bill')
```

Out[71]: <AxesSubplot:xlabel='total_bill', ylabel='Density'>



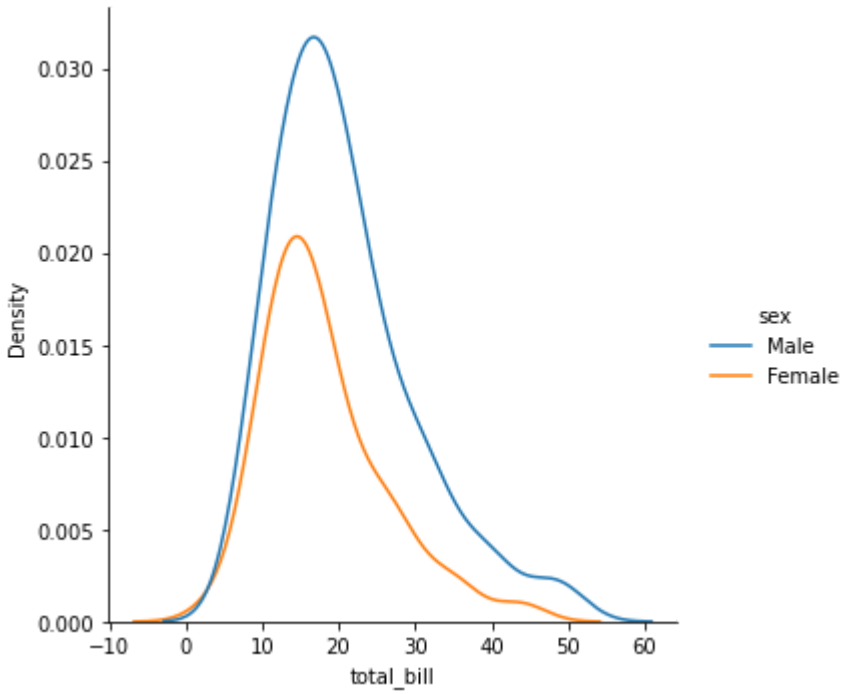
In [72]:
```
# Figure level Function

sns.displot(data = tips , x ='total_bill' , kind ='kde')
```

Out[72]: <seaborn.axisgrid.FacetGrid at 0x261ed1384c0>



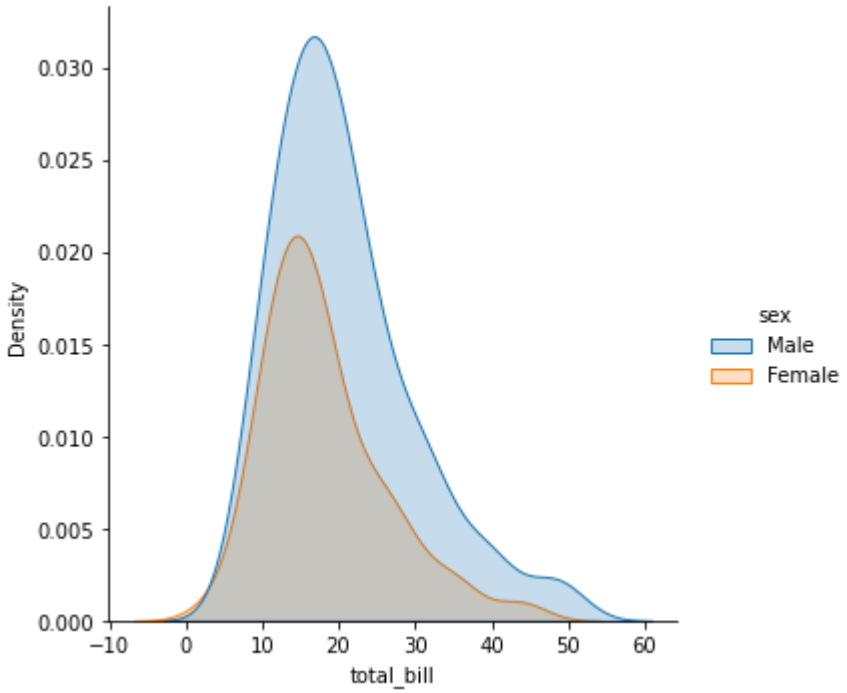
In [73]:
```
# Hue

sns.displot(data = tips , x ='total_bill' , kind ='kde' , hue ='sex')
```

Out[73]: <seaborn.axisgrid.FacetGrid at 0x261ee5c08e0>



In [74]:
```
# fill

sns.displot(data = tips , x ='total_bill' , kind ='kde',
            hue ='sex',fill =True) # fill
```

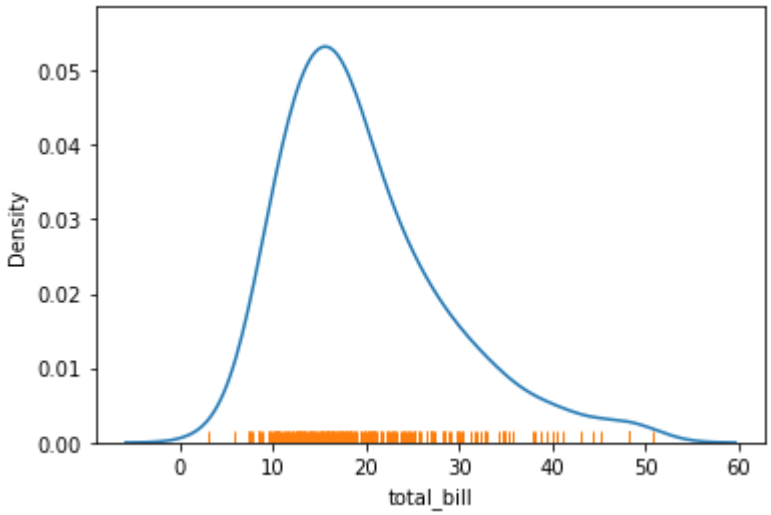
Out[74]: <seaborn.axisgrid.FacetGrid at 0x261ee604820>



Rugplot

In [75]:
```
# Plot marginal distributions by drawing ticks along the x and y axes.

# This function is intended to complement other plots by showing the location
# of individual observations in an unobtrusive way.

sns.kdeplot(data=tips,x='total_bill')

sns.rugplot(data=tips,x='total_bill') # rug plot
```

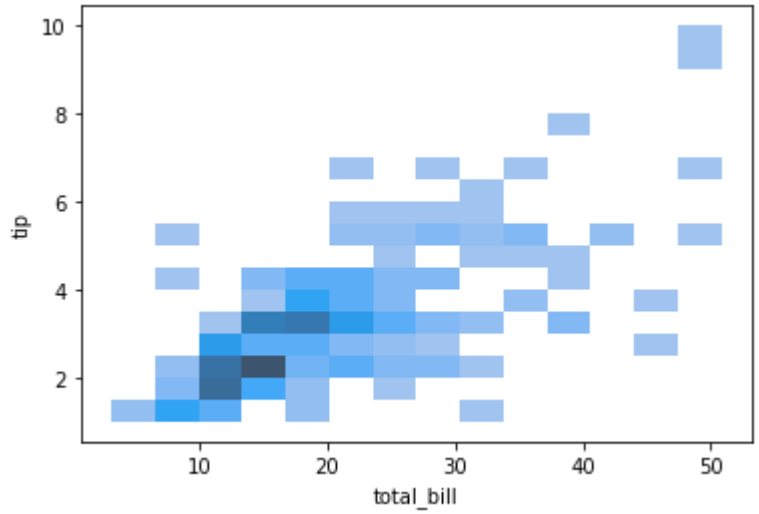
Out[75]: <AxesSubplot:xlabel='total_bill', ylabel='Density'>



Bivariate Histogram

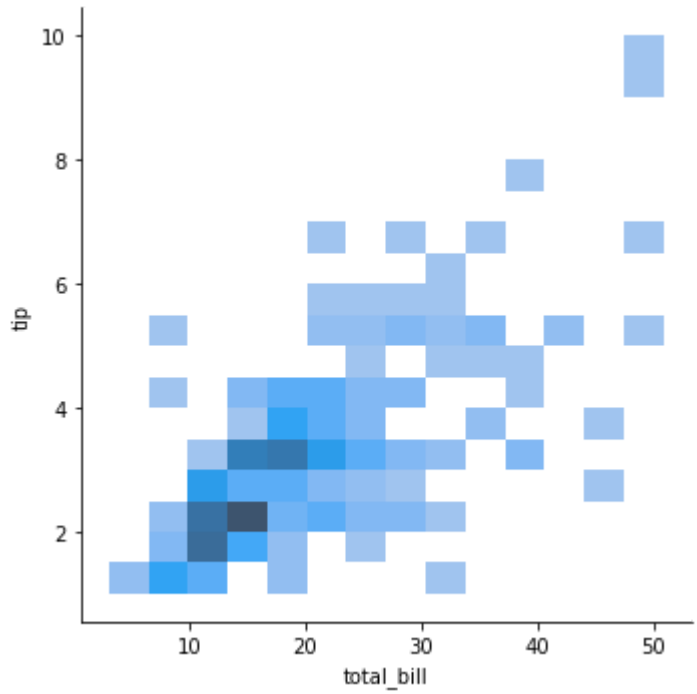
In [77]:
```
# A bivariate histogram bins the data within rectangles that tile the plot

# and then shows the count of observations within each rectangle
# with the fill color

sns.histplot(data=tips, x='total_bill', y='tip') # Axes level
```

Out[77]:
```
<AxesSubplot:xlabel='total_bill', ylabel='tip'>
```



In [78]:
```
# figure level

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

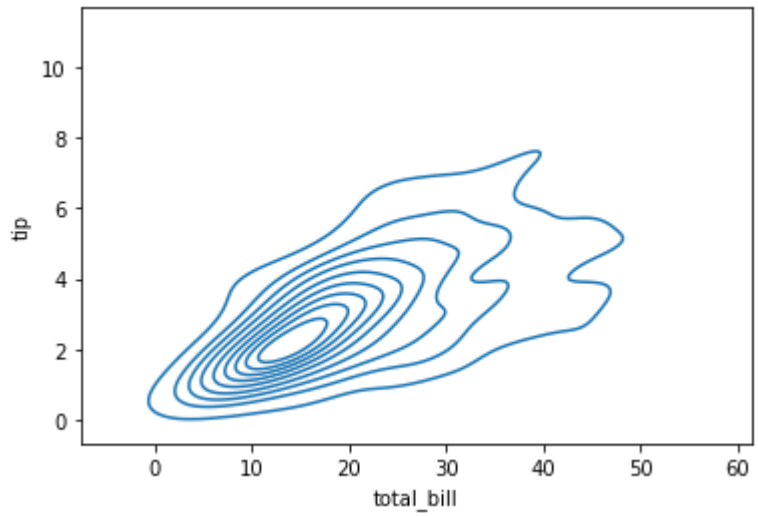
Out[78]:
```
<seaborn.axisgrid.FacetGrid at 0x261ee61e9a0>
```



Bivariate Kde Plot

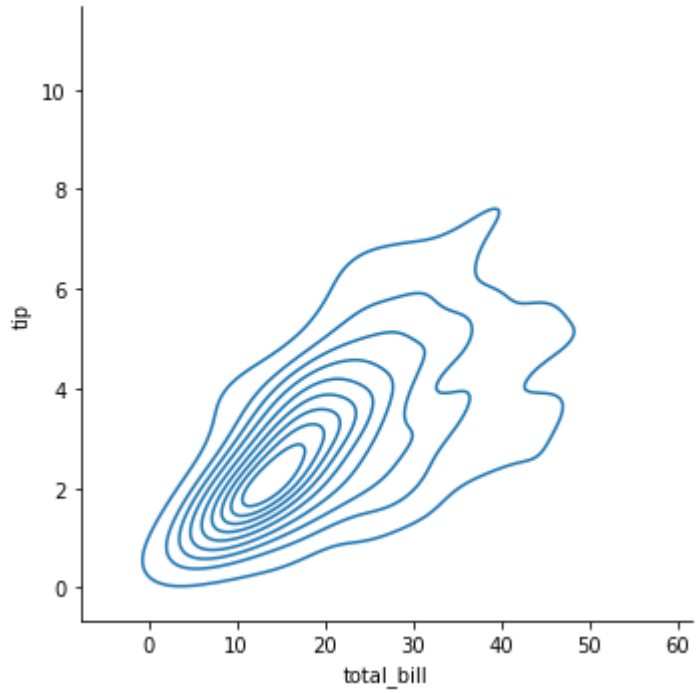
In [79]:
```
# a bivariate KDE plot smoothes the (x, y) observations with a 2D Gaussian

sns.kdeplot(data=tips, x='total_bill', y='tip') # Axes level
```

Out[79]: <AxesSubplot:xlabel='total_bill', ylabel='tip'>



In [81]:
```
# Figure level Function

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

Out[81]: <seaborn.axisgrid.FacetGrid at 0x261ee8bddf0>



3. Matrix Plot

- Heatmap
- Clustermap

In [83]:
```
# Heatmap

# Plot rectangular data as a color-encoded matrix

gap
```

Out[83]:

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_num
0	Afghanistan	Asia	1952	28.801	8425333	779.445314	AFG	4
1	Afghanistan	Asia	1957	30.332	9240934	820.853030	AFG	4
2	Afghanistan	Asia	1962	31.997	10267083	853.100710	AFG	4
3	Afghanistan	Asia	1967	34.020	11537966	836.197138	AFG	4
4	Afghanistan	Asia	1972	36.088	13079460	739.981106	AFG	4
...	...	...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306	ZWE	716
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786	ZWE	716
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960	ZWE	716
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623	ZWE	716
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298	ZWE	716

1704 rows × 8 columns

In [85]:
```
# Converting long data to wide data

df = gap.pivot(index = 'country' , columns ='year',values ='lifeExp')
```

In [86]:
```
df
```

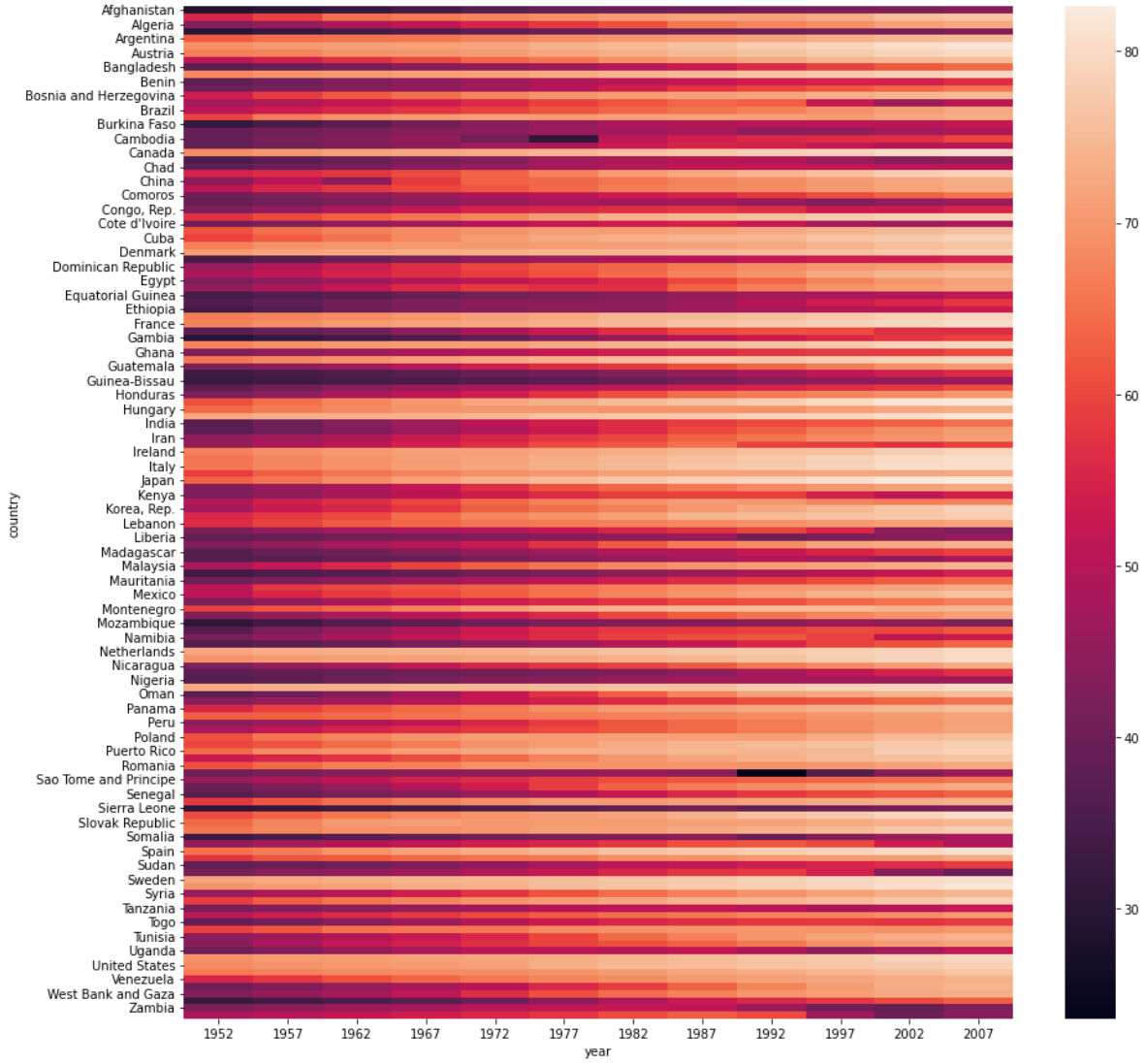
Out[86]:

year	1952	1957	1962	1967	1972	1977	1982	1987	1992	1997	20
country											
Afghanistan	28.801	30.332	31.997	34.020	36.088	38.438	39.854	40.822	41.674	41.763	42.1
Albania	55.230	59.280	64.820	66.220	67.690	68.930	70.420	72.000	71.581	72.950	75.6
Algeria	43.077	45.685	48.303	51.407	54.518	58.014	61.368	65.799	67.744	69.152	70.9
Angola	30.015	31.999	34.000	35.985	37.928	39.483	39.942	39.906	40.647	40.963	41.0
Argentina	62.485	64.399	65.142	65.634	67.065	68.481	69.942	70.774	71.868	73.275	74.3
...	...	...	...	...	...	...	...	...	...	...	
Vietnam	40.412	42.887	45.363	47.838	50.254	55.764	58.816	62.820	67.662	70.672	73.0
West Bank and Gaza	43.160	45.671	48.127	51.631	56.532	60.765	64.406	67.046	69.718	71.096	72.3
Yemen, Rep.	32.548	33.970	35.180	36.984	39.848	44.175	49.113	52.922	55.599	58.020	60.3
Zambia	42.038	44.077	46.023	47.768	50.107	51.386	51.821	50.821	46.100	40.238	39.1
Zimbabwe	48.451	50.469	52.358	53.995	55.635	57.674	60.363	62.351	60.377	46.809	39.9

142 rows × 12 columns

In [95]:
```
# Axes level function

plt.figure(figsize =(15,15))

sns.heatmap(data =df)
```

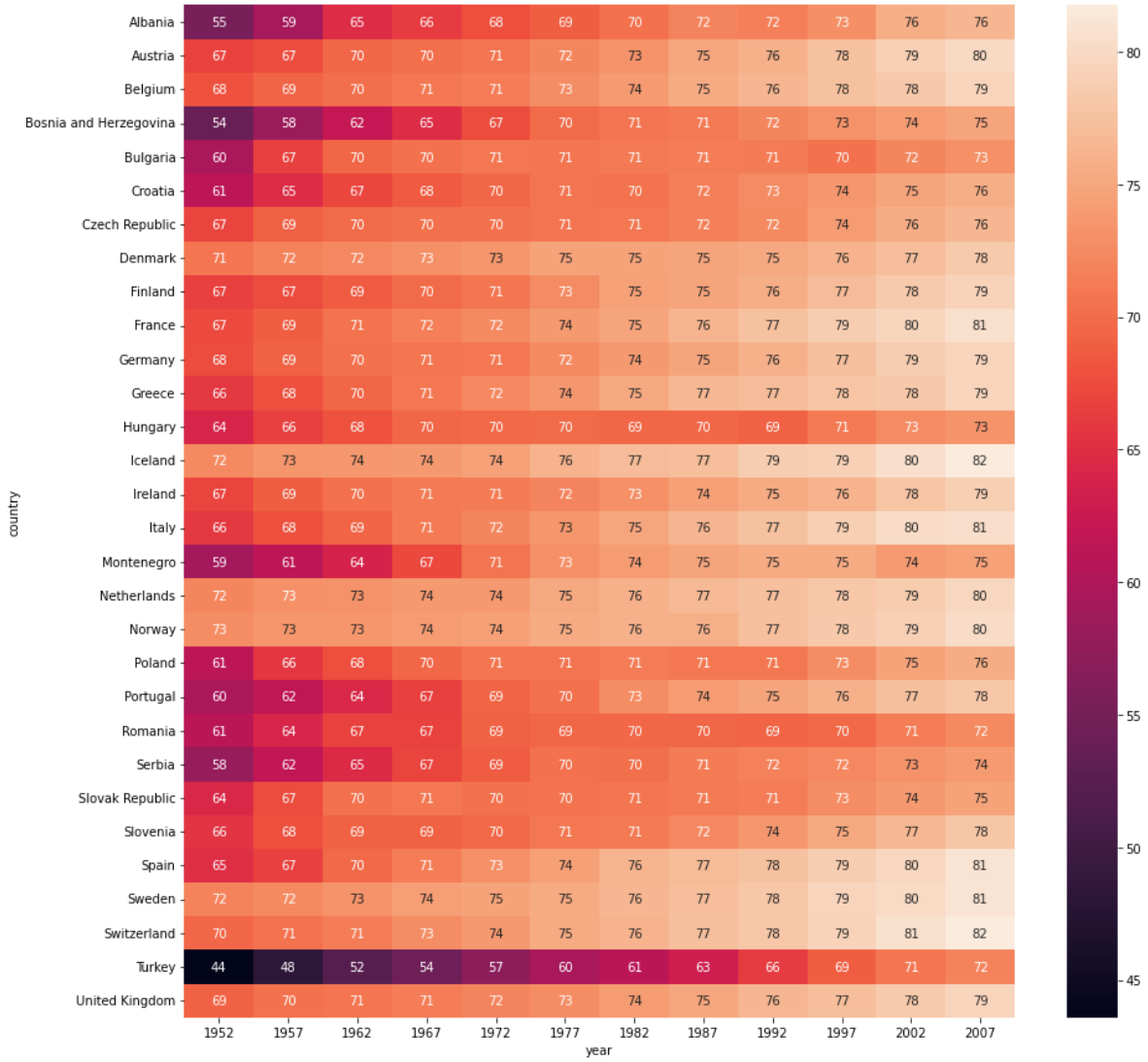
Out[95]: <AxesSubplot:xlabel='year', ylabel='country'>



annot

In [104]:
```
df.eur = gap[gap['continent']=='Europe'].pivot(index = 'country',
                                               columns ='year',values ='lifeExp')

plt.figure(figsize =(15,15))

sns.heatmap(data=df.eur , annot =True)
```

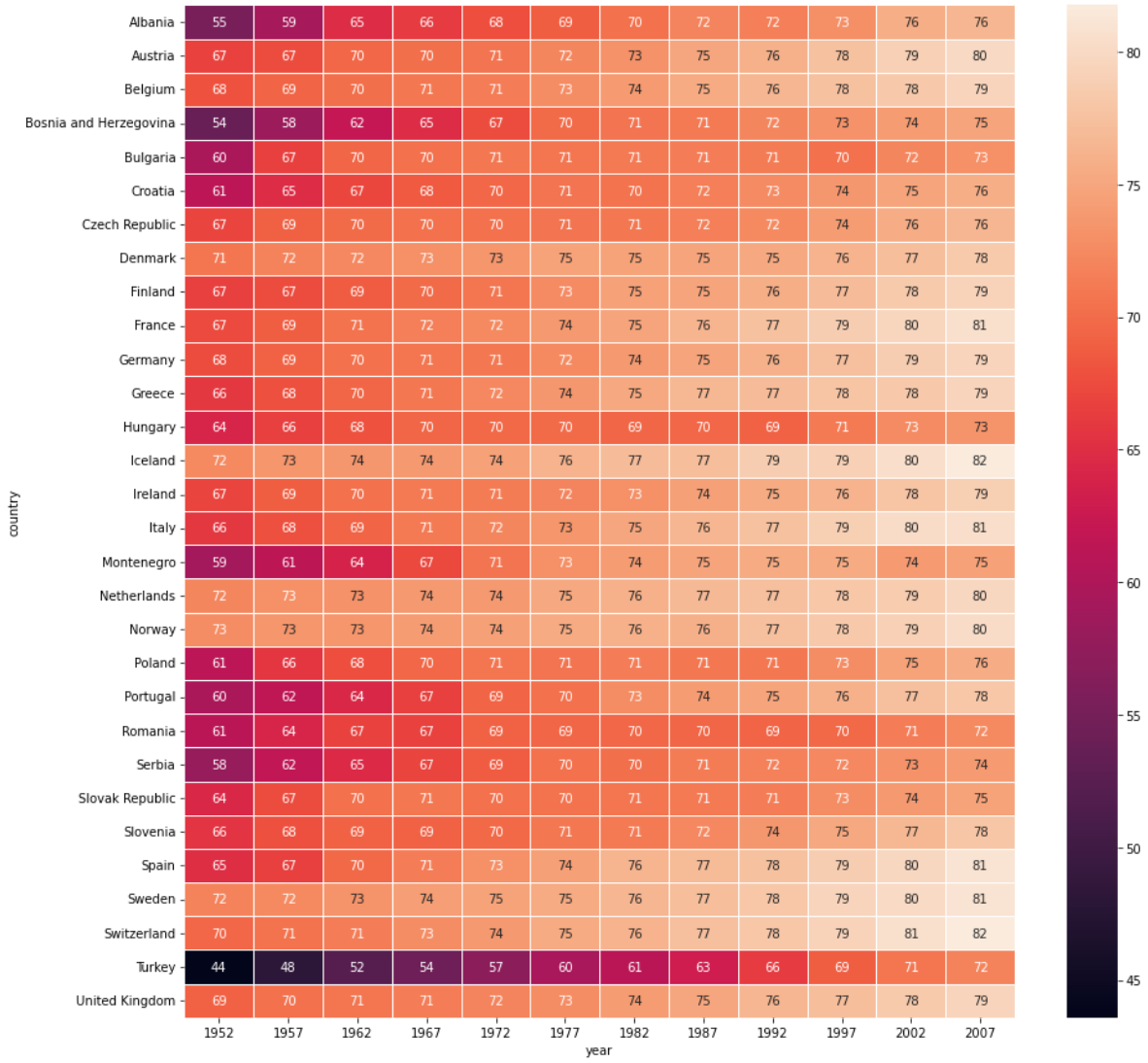
Out[104]: <AxesSubplot:xlabel='year', ylabel='country'>



linewidth

In [105]:
```
plt.figure(figsize =(15,15))

sns.heatmap(data=df.eur , annot =True , linewidth = 0.5)
```

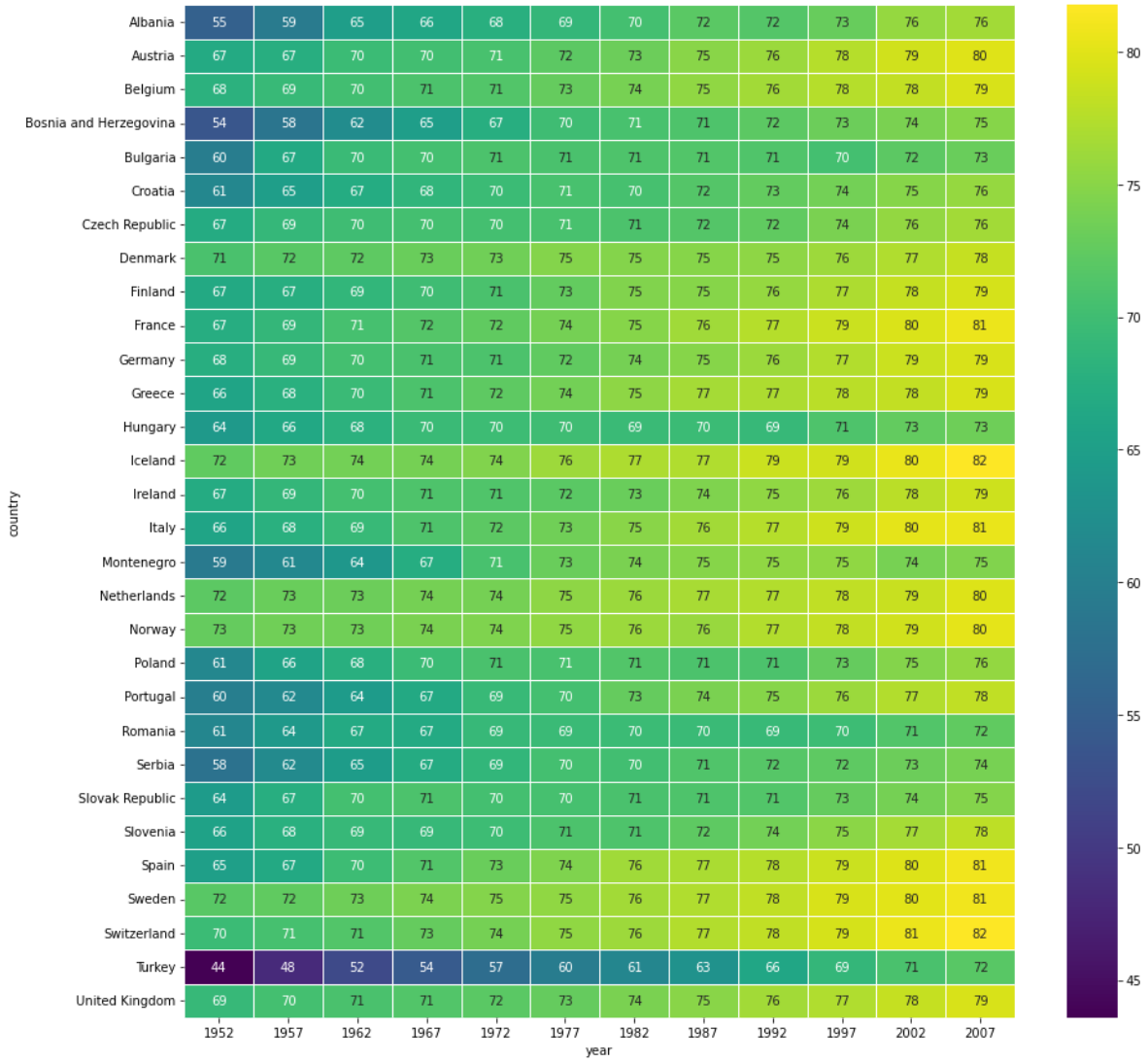
Out[105]: <AxesSubplot:xlabel='year', ylabel='country'>



colors

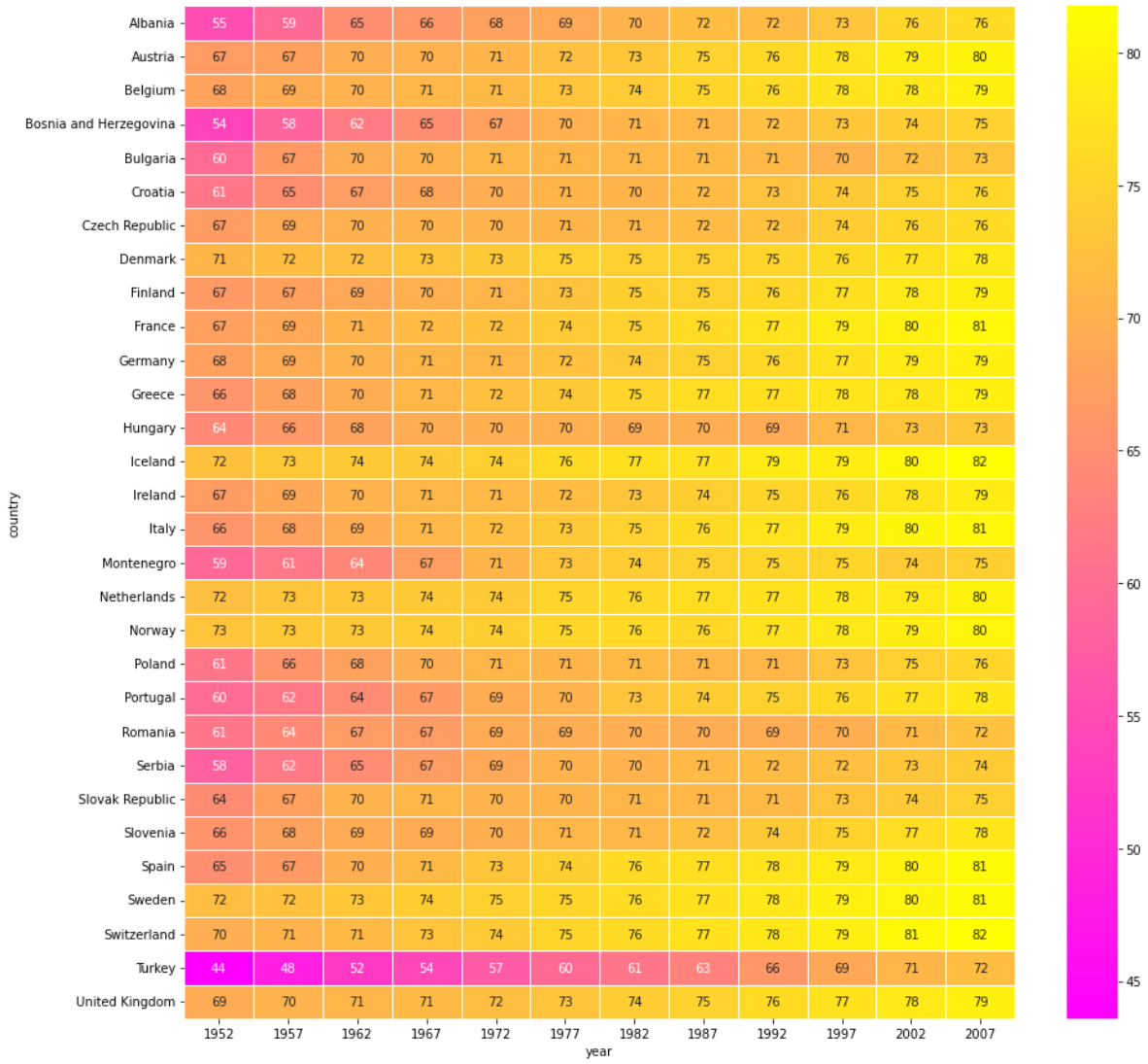
In [106]:
```
plt.figure(figsize =(15,15))

sns.heatmap(data=df.eur , annot =True , linewidth = 0.5 , cmap ='viridis')
```

Out[106]: <AxesSubplot:xlabel='year', ylabel='country'>



In [111]:
```
plt.figure(figsize =(15,15))

sns.heatmap(data=df.eur , annot =True , linewidth = 0.5 , cmap ='spring')
```

Out[111]: <AxesSubplot:xlabel='year', ylabel='country'>



Cluster map

In [113]:
```
# Plot a matrix dataset as a hierarchically-clustered heatmap. (similarity)

# This function requires scipy to be available.

iris = px.data.iris()
iris .sample()
```

Out[113]:

	sepal_length	sepal_width	petal_length	petal_width	species	species_id
16	5.4	3.9	1.3	0.4	setosa	1

In [115]:
```
sns.clustermap(iris.iloc[:,[0,1,2,3]])
```

Out[115]:
```
<seaborn.matrix.ClusterGrid at 0x261f290c310>
```

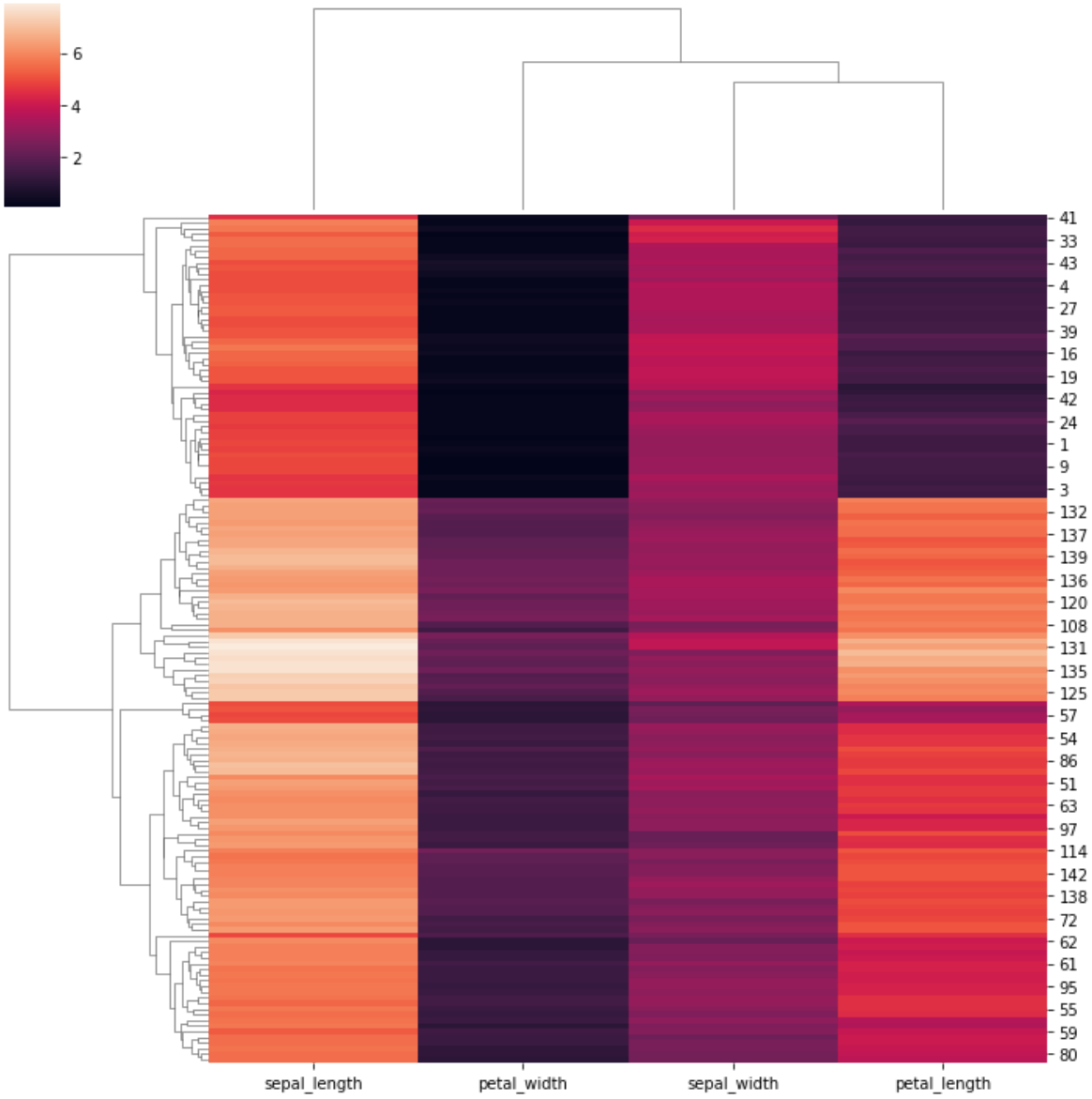
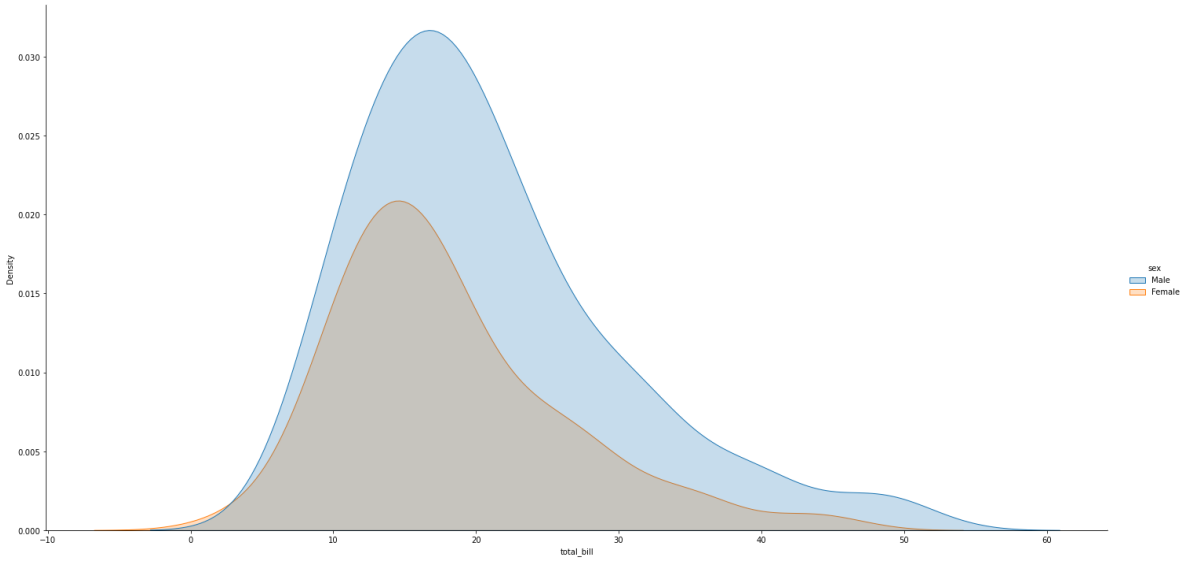


Figure level (Height)

In [116]:
```
# height , aspect

sns.displot(data=tips,x='total_bill',kind='kde',
            hue='sex',fill=True,
            height=10,aspect=2)
```

Out[116]: <seaborn.axisgrid.FacetGrid at 0x261f3a46a30>



In [ ]:

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)
```

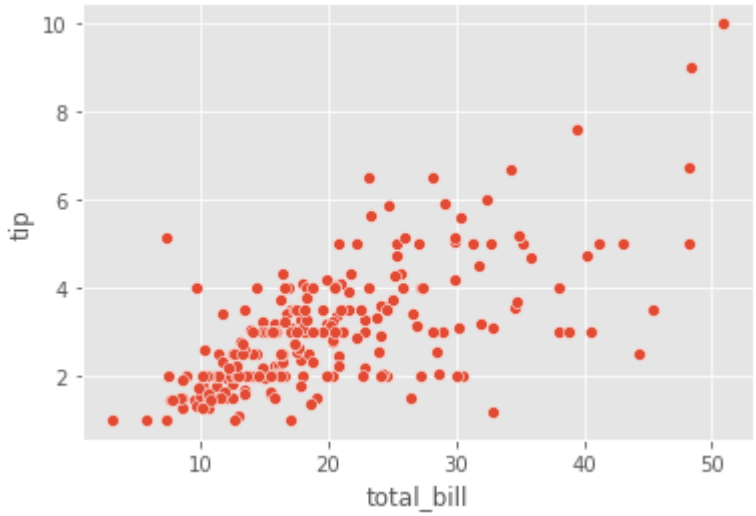
Out[4]:

	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')
```

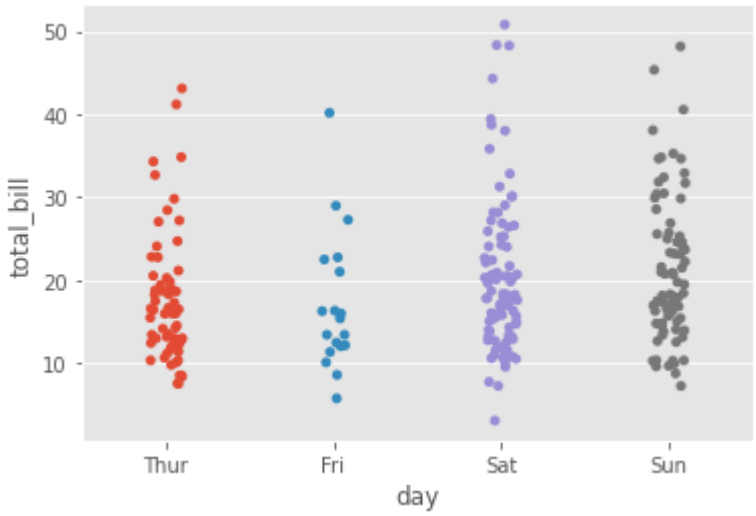
Out[5]:
```
<AxesSubplot:xlabel='total_bill', ylabel='tip'>
```



1.strip plot

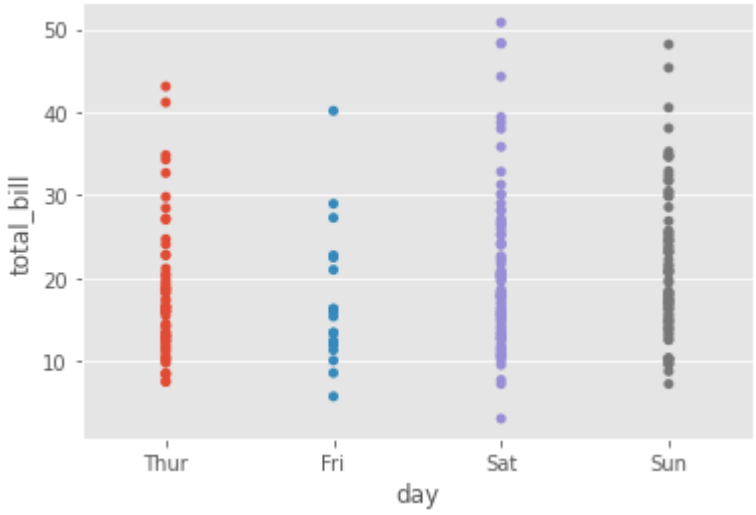
In [6]:
```
# axes level function

# 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)
```

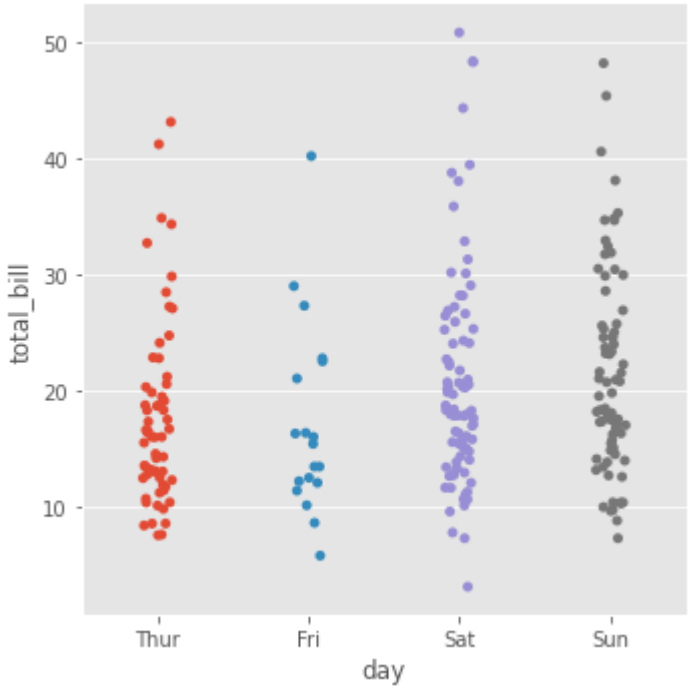
Out[7]: <AxesSubplot:xlabel='day', ylabel='total_bill'>



catplot

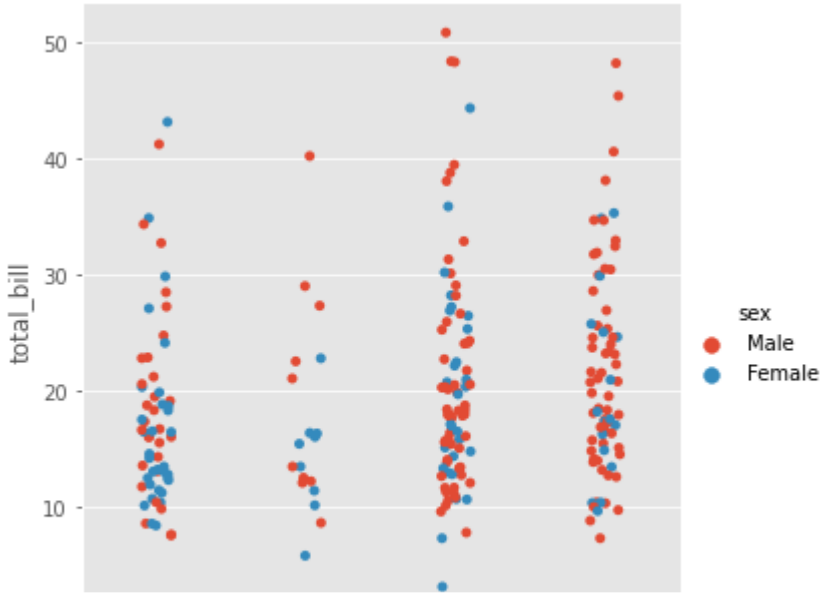
In [8]:
```
# catplot -> Categorical plot

# 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')
```

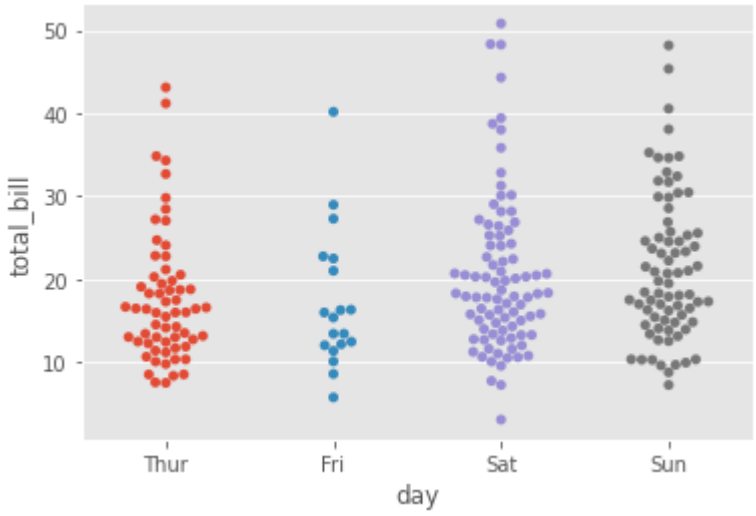
Out[9]: <seaborn.axisgrid.FacetGrid at 0x1e85f320e50>



2 .Swarm plot

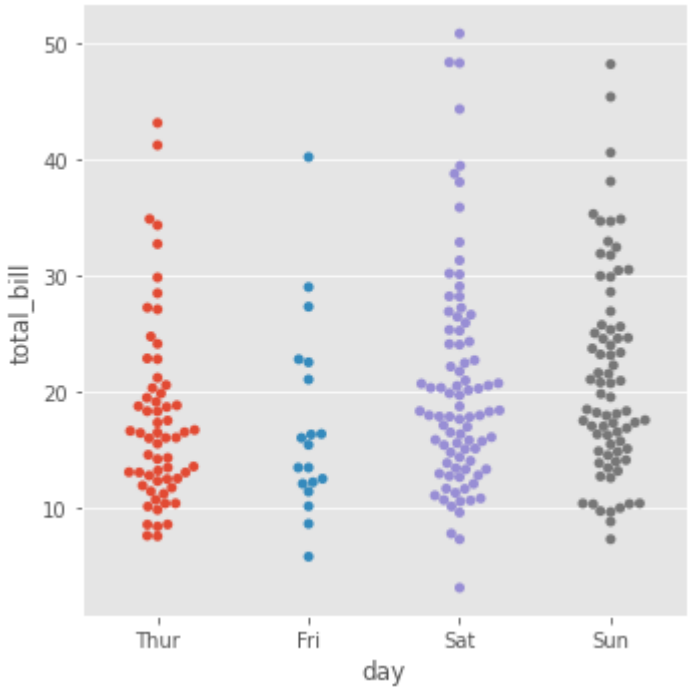
In [10]:
```
sns.swarmplot(data=tips,x ='day', y='total_bill')
```

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



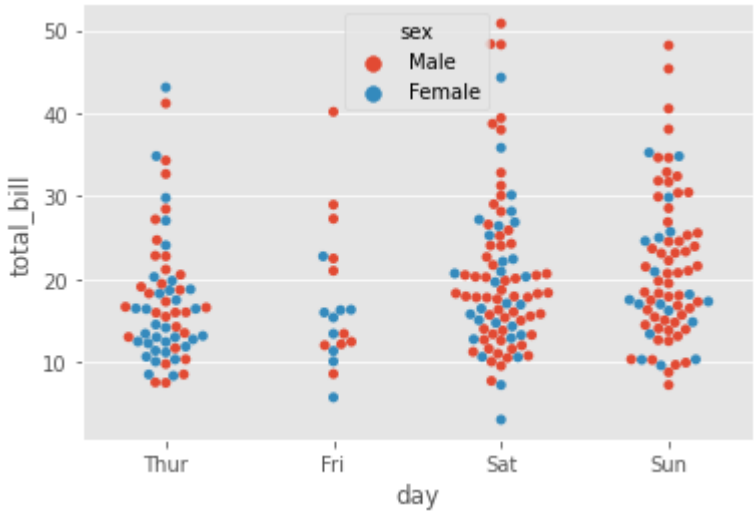
In [11]:
```
# use by catplot

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'>
```

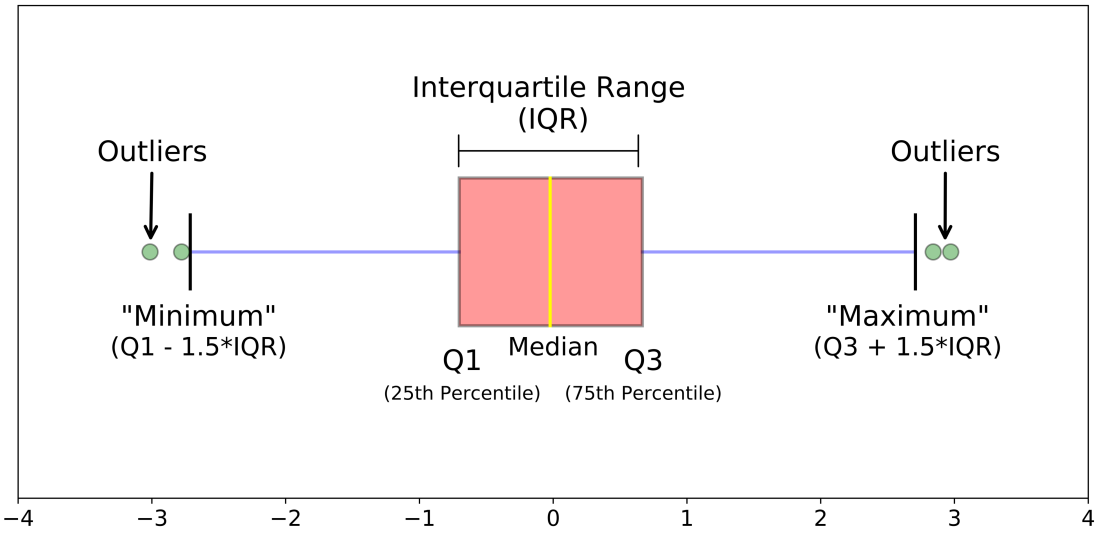


Categorical Distribution Plots

1. Boxplot

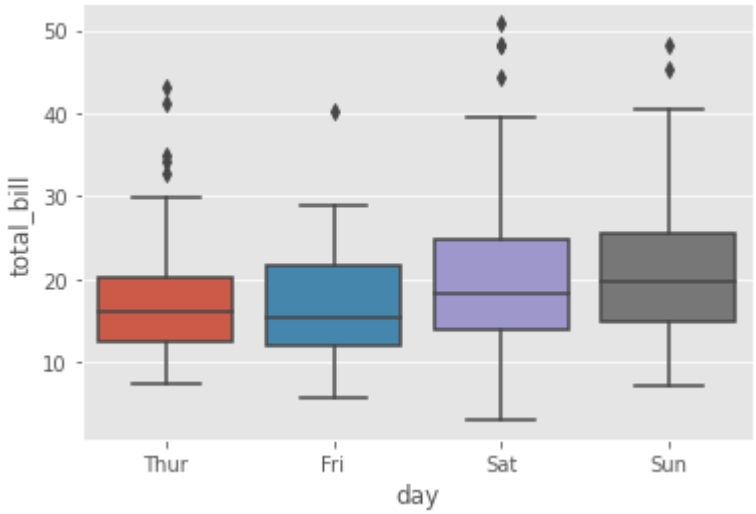
A boxplot is a standardized way of displaying the distribution of data based on a five number summary **("minimum", first quartile [Q1], median, third quartile [Q3] and "maximum")**.

- 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.



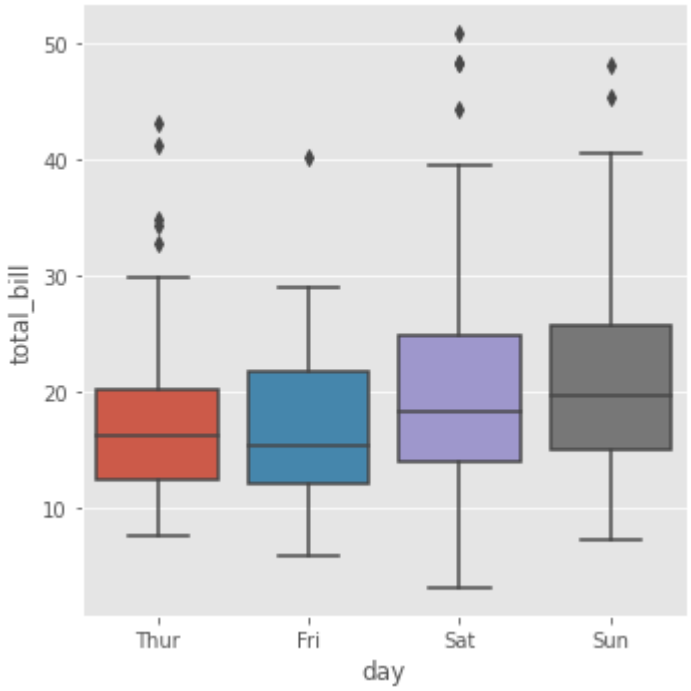
In [13]:
```
# Box plot
sns.boxplot(data=tips,x='day',y='total_bill')
```

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



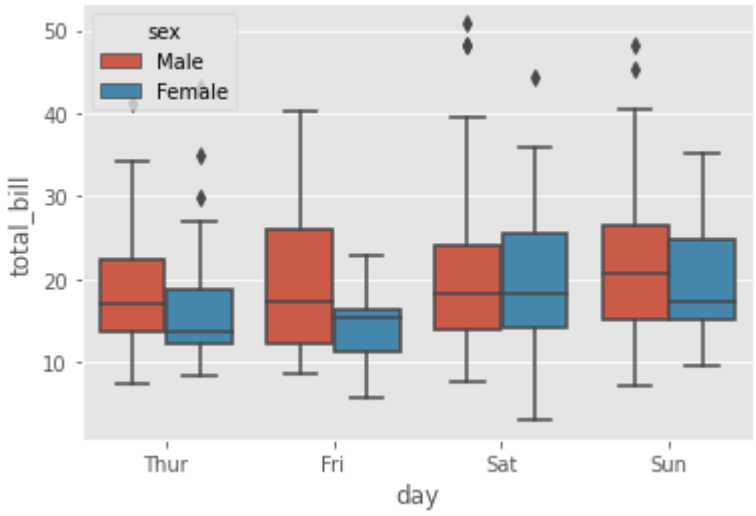
In [14]:
```
# Using catplot - Figure level

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

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



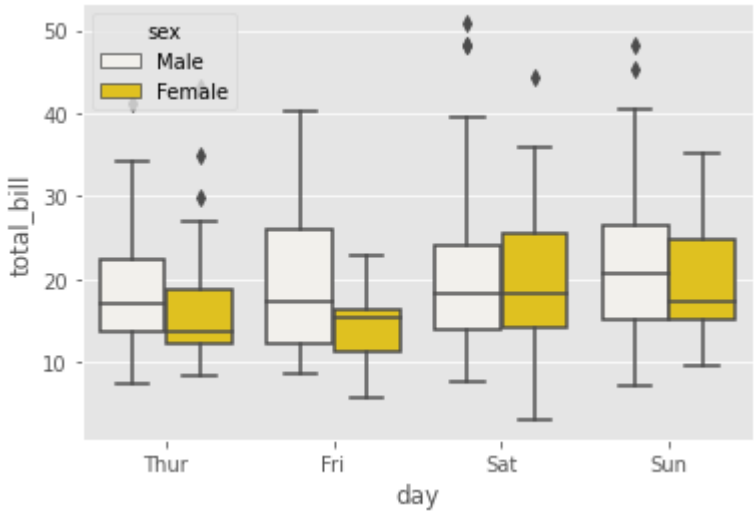
In [15]:
```
# hue

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

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



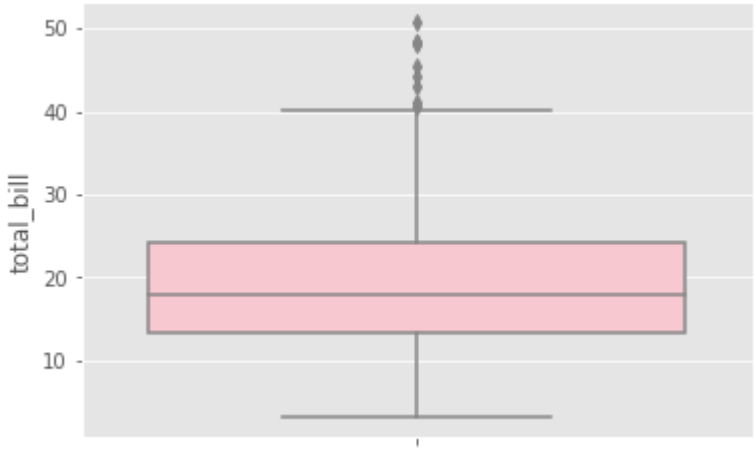
In [16]:
```
# color

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')
```

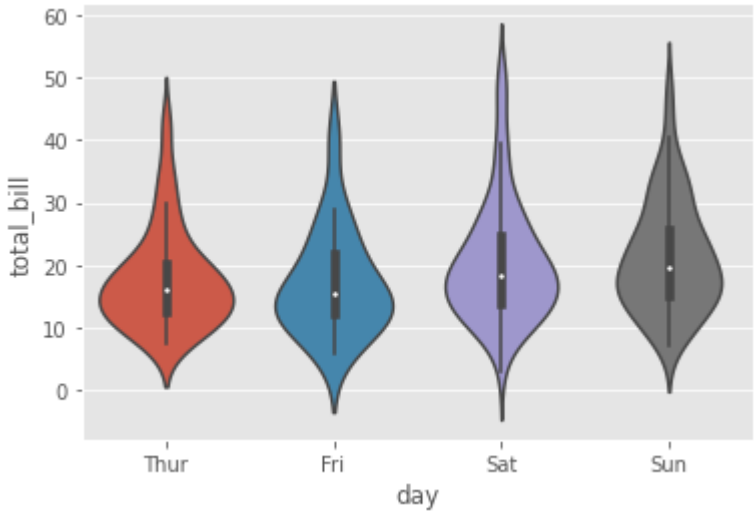
Out[17]: <AxesSubplot:ylabel='total_bill'>



2 . Voilin Plot

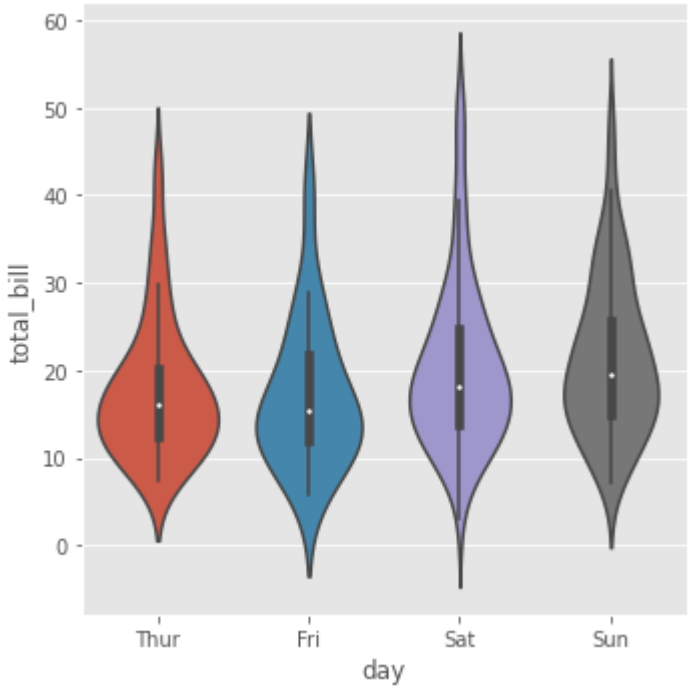
In [18]:
```
# Violinplot = (Boxplot + KDEplot)

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

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



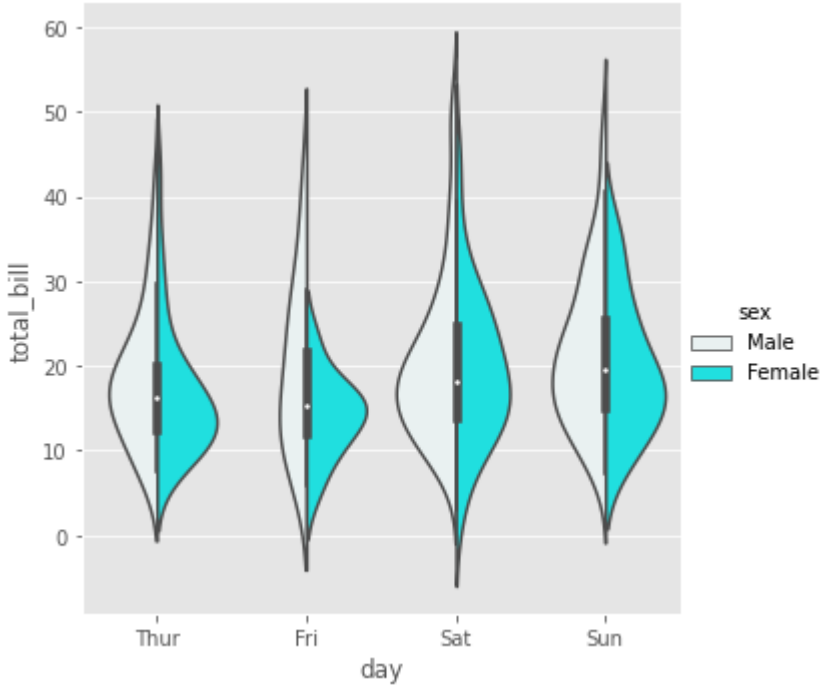
In [19]:
```
# catplot -figure level function

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

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



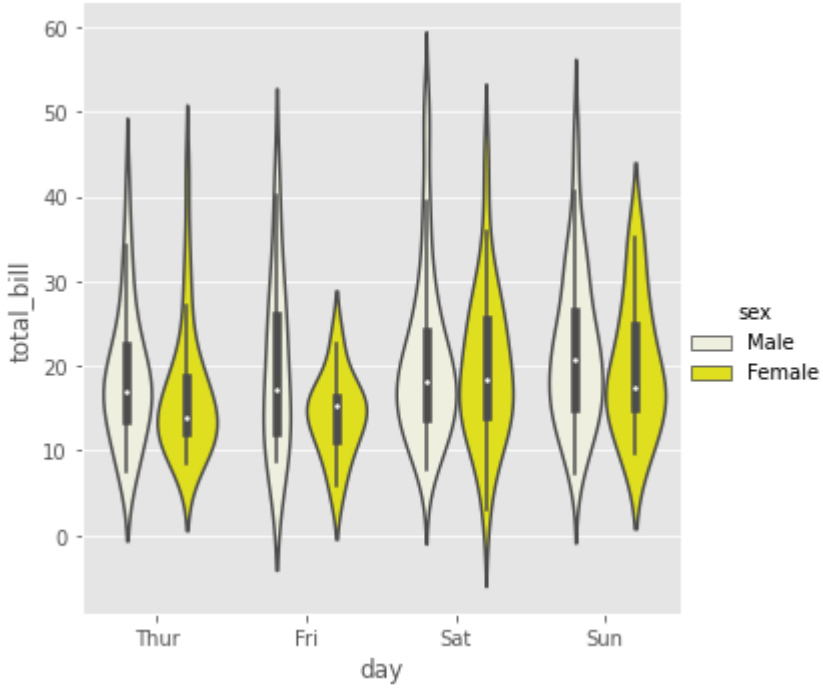
In [20]:
```
# Color

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

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



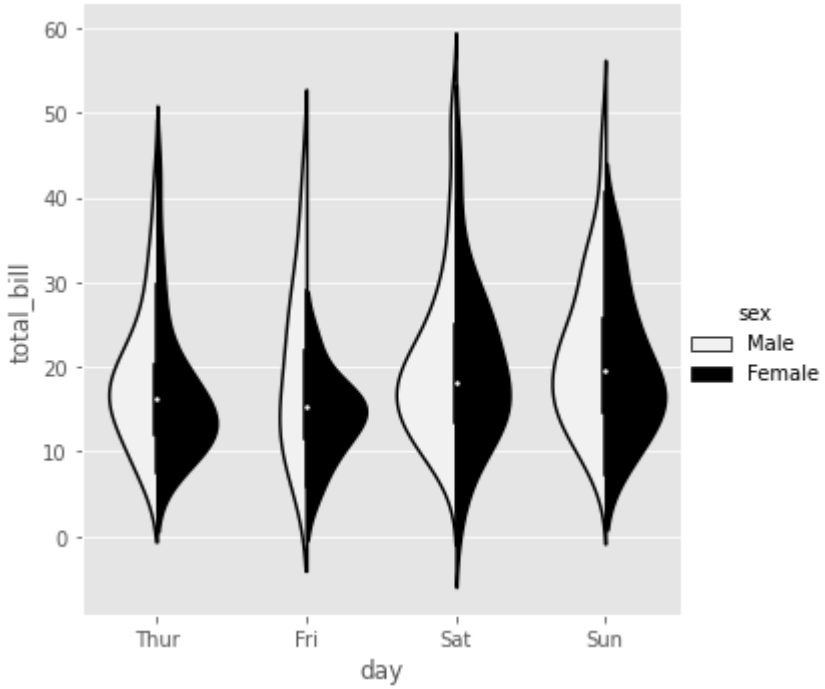
In [21]:
```
# hue

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>



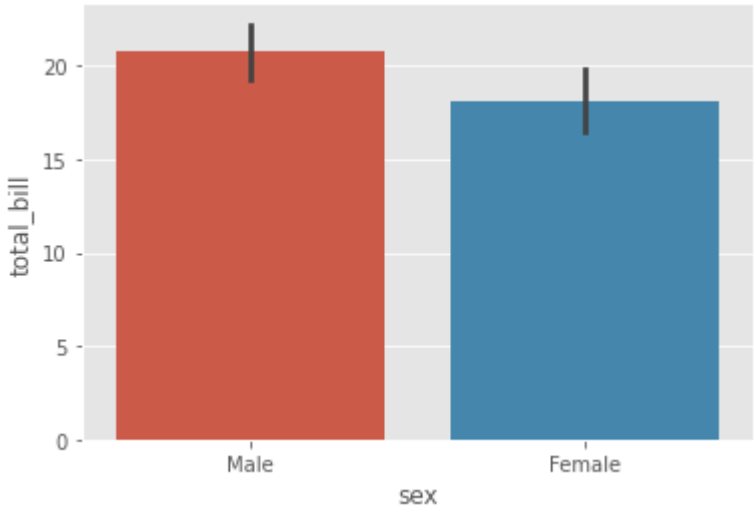
Categorical Estimate Plot

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**

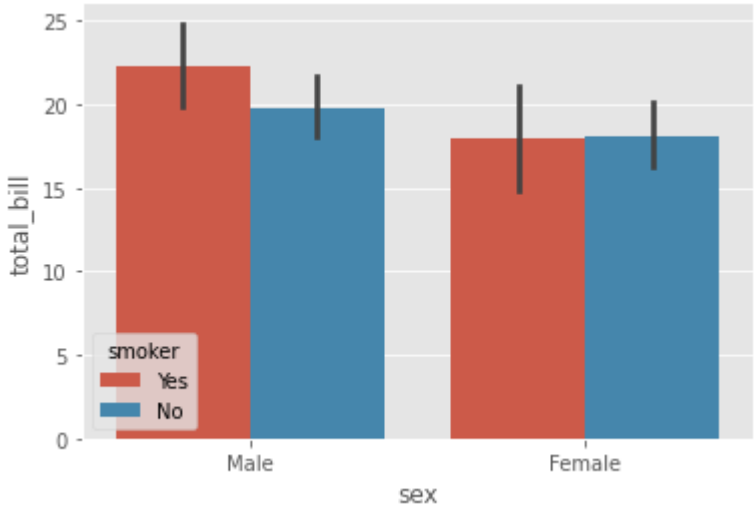
In [23]:
```
sns.barplot(data=tips, x='sex', y='total_bill')
```

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



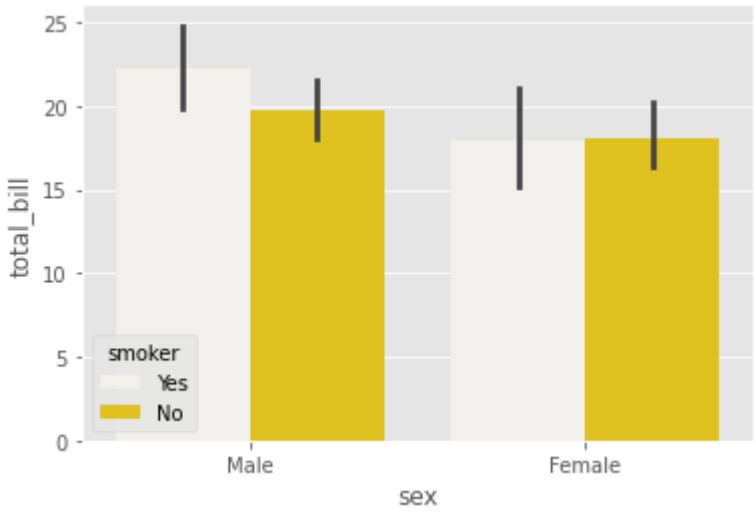
In [24]:
```
# hue

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

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



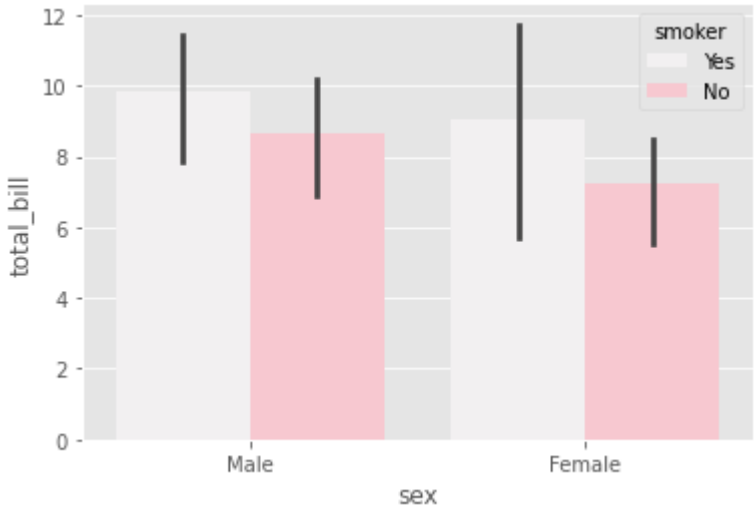
In [25]:
```
# color

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

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



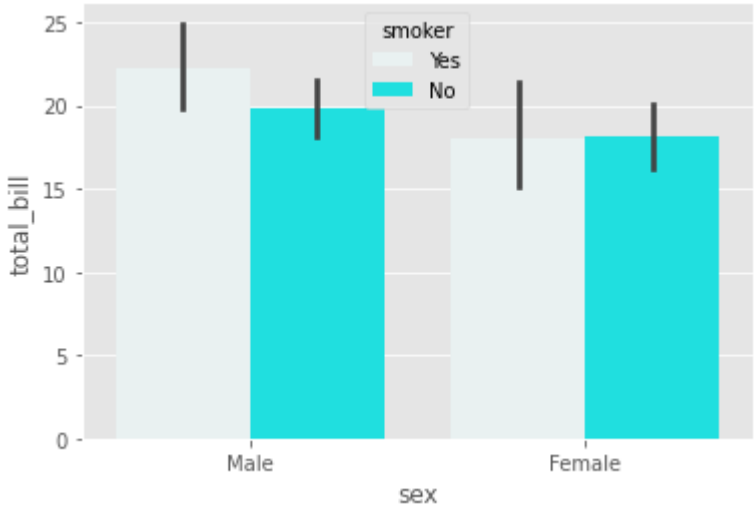
In [26]:
```
# estimator

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'>



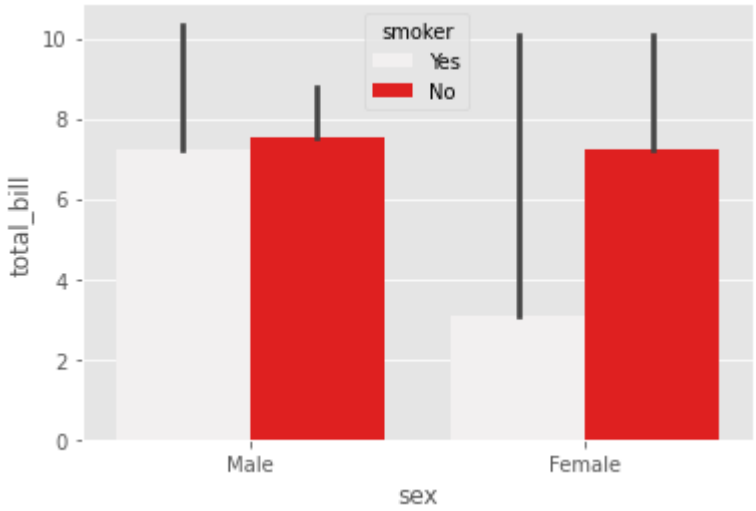
In [27]:
```
# estimator ( mean)


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'>

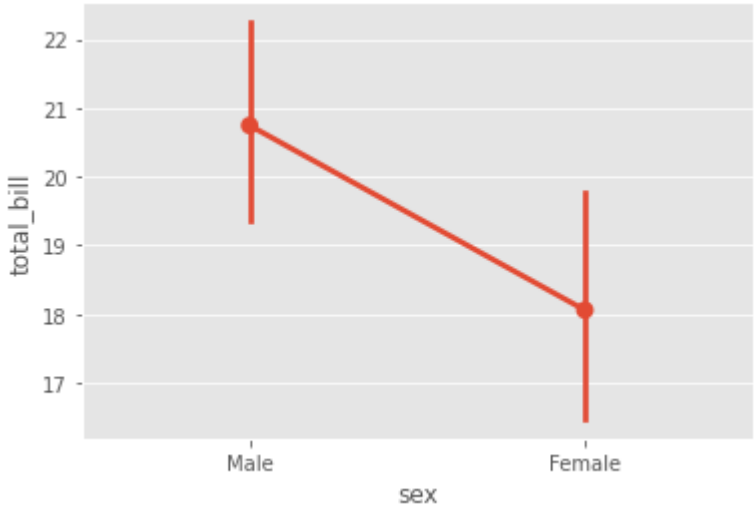


2. Point plot

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

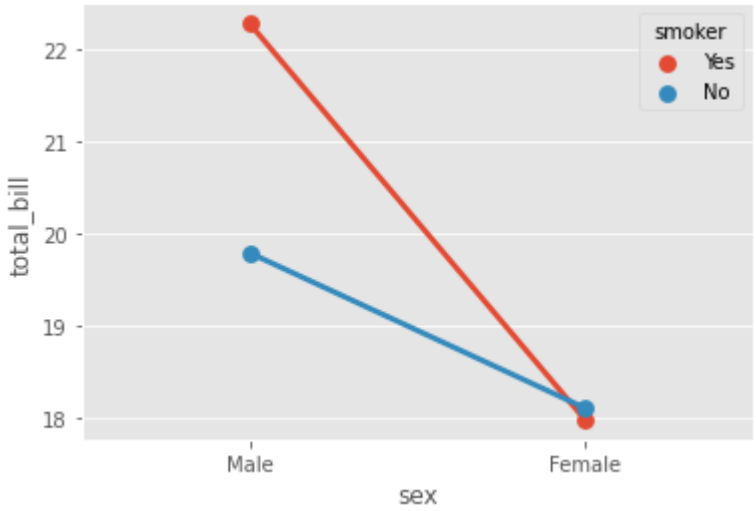
In [29]:
```
sns.pointplot(data=tips, x='sex', y='total_bill')
```

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'>

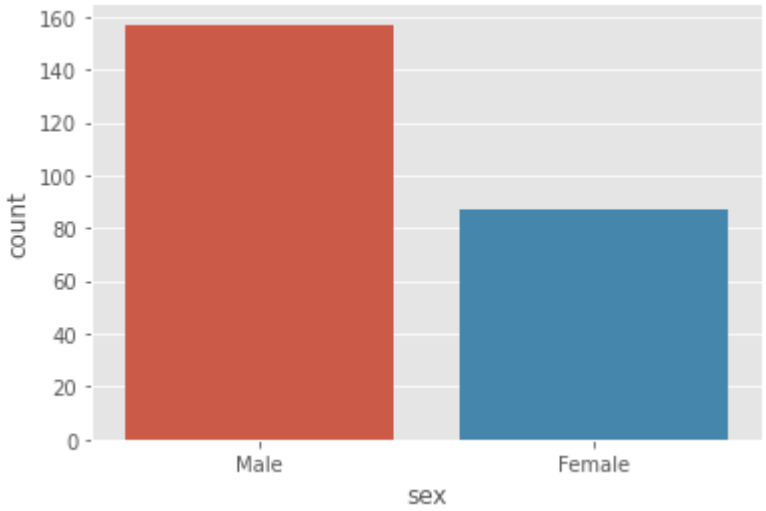


3.Count Plot

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

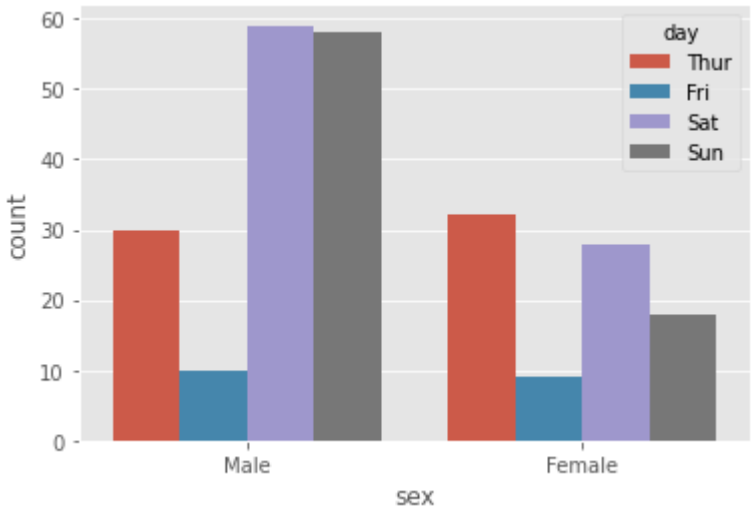
In [31]:
```
# countplot

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

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



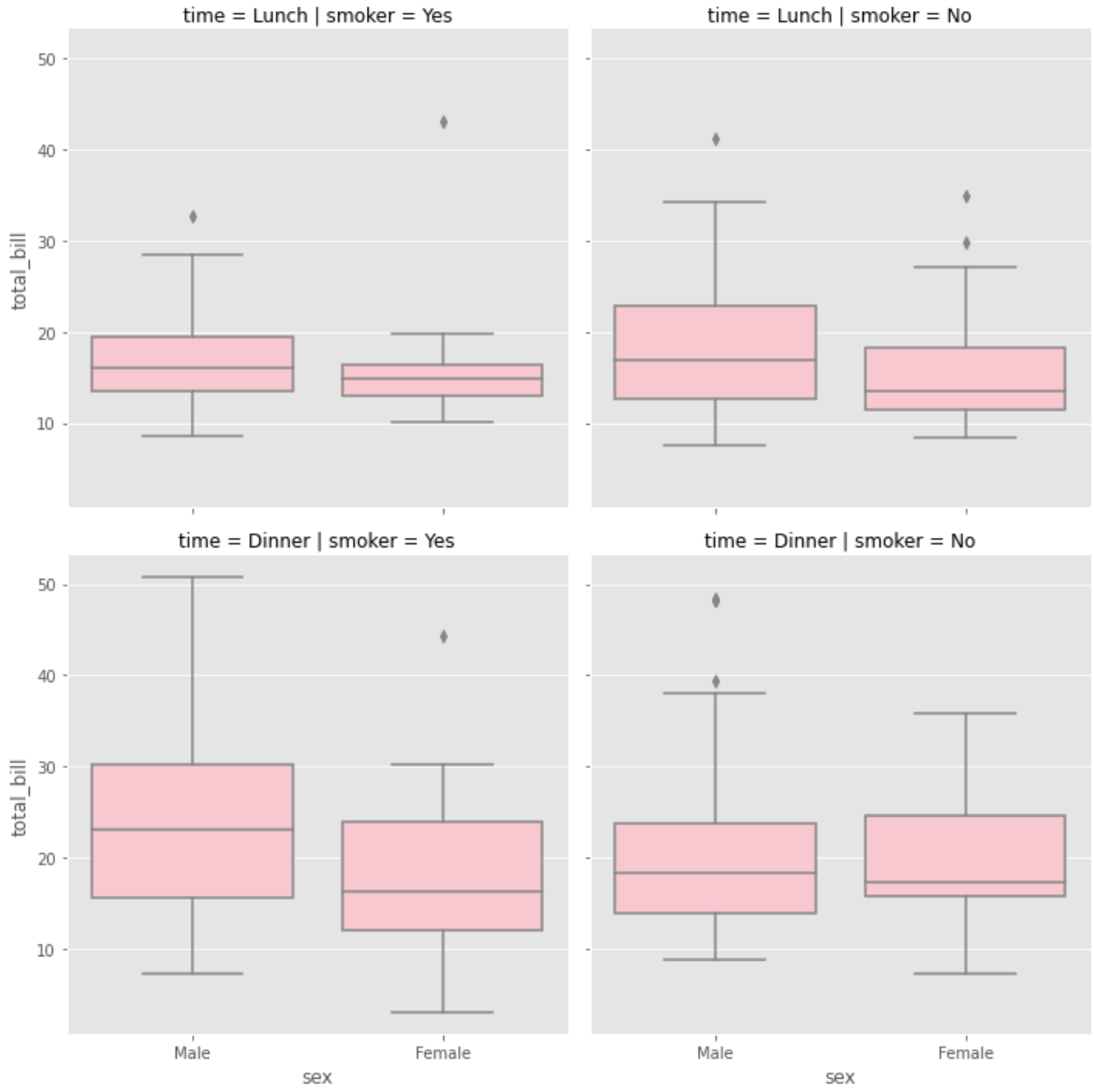
In [32]:
```
sns.countplot(data=tips,x='sex',hue='day')
```

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>
```



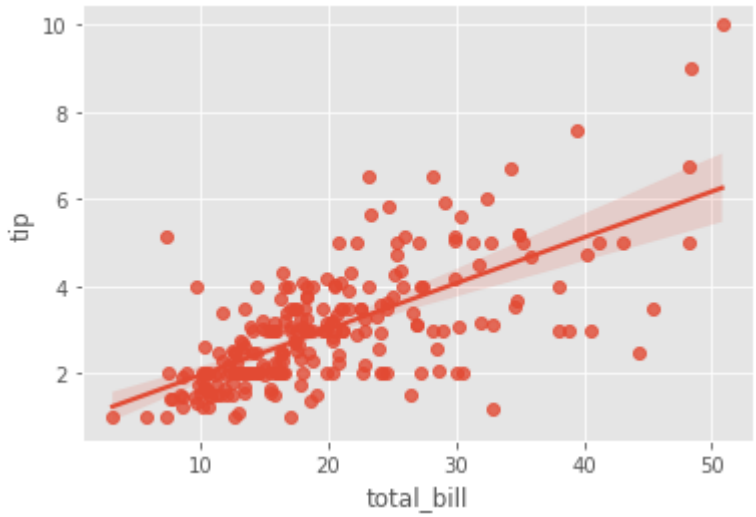
Regression Plots

- 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')
```

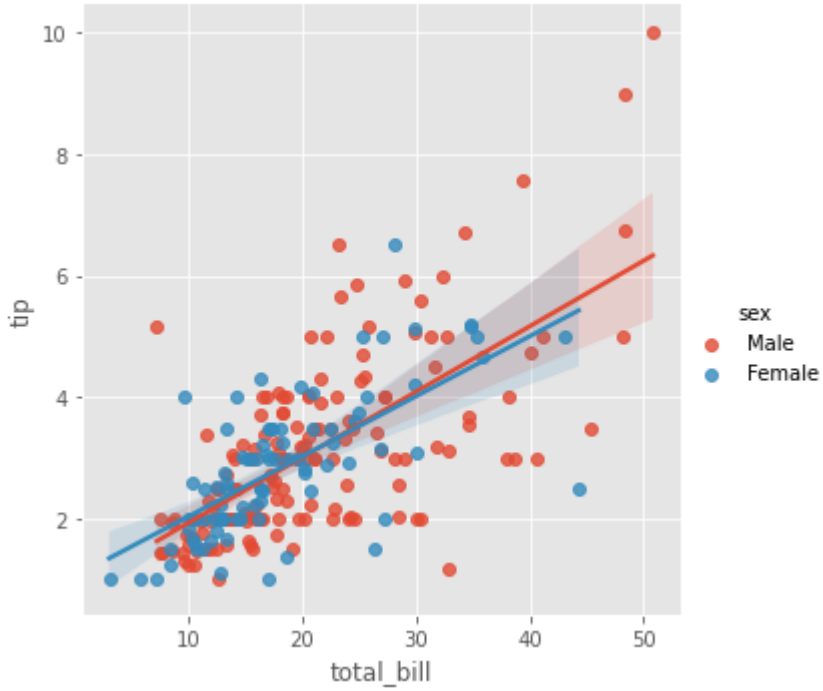
Out[34]: <AxesSubplot:xlabel='total_bill', ylabel='tip'>



Lmplot

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

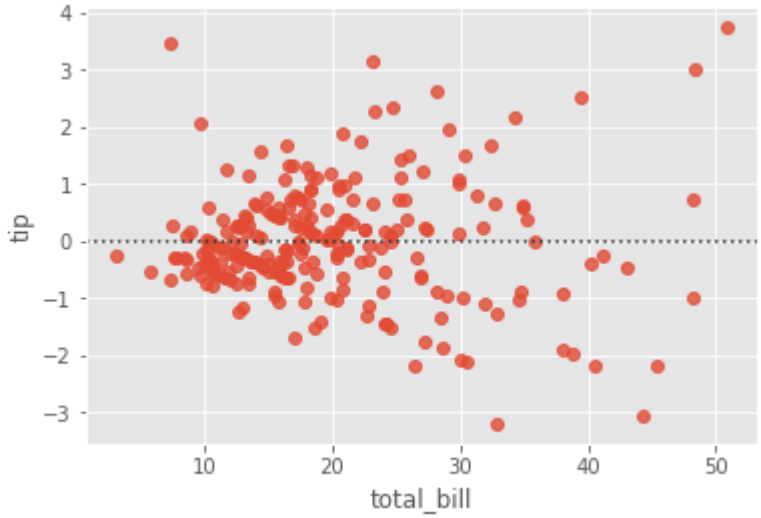
Out[35]: <seaborn.axisgrid.FacetGrid at 0x1e862649a00>



residplot

In [36]:
```
# residual plot

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

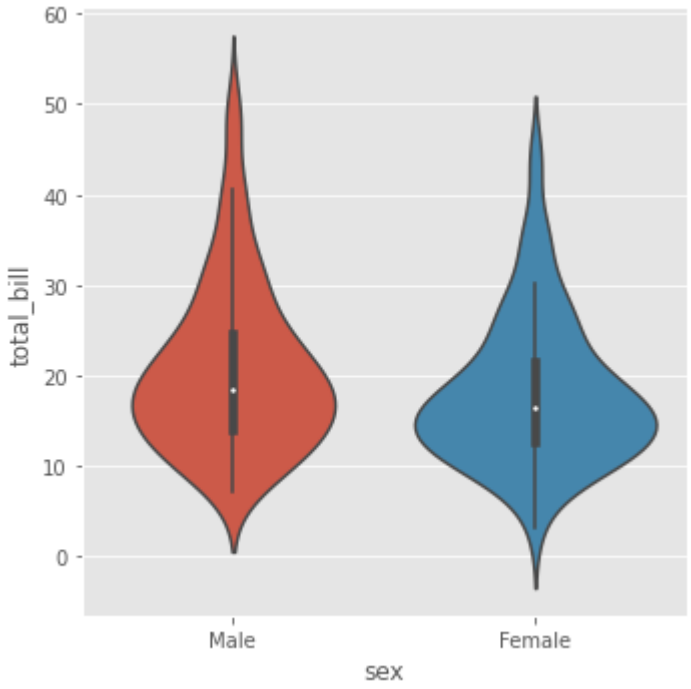
Out[36]:
```
<AxesSubplot:xlabel='total_bill', ylabel='tip'>
```



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>

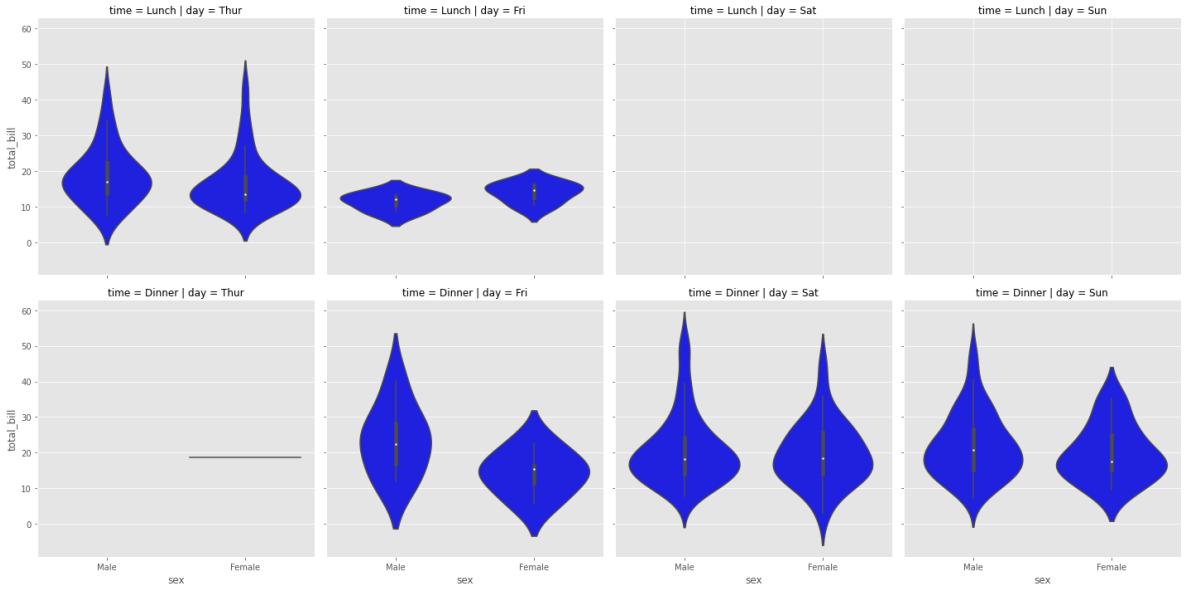


In [41]:
```
# Facet Plot - columns

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')
```

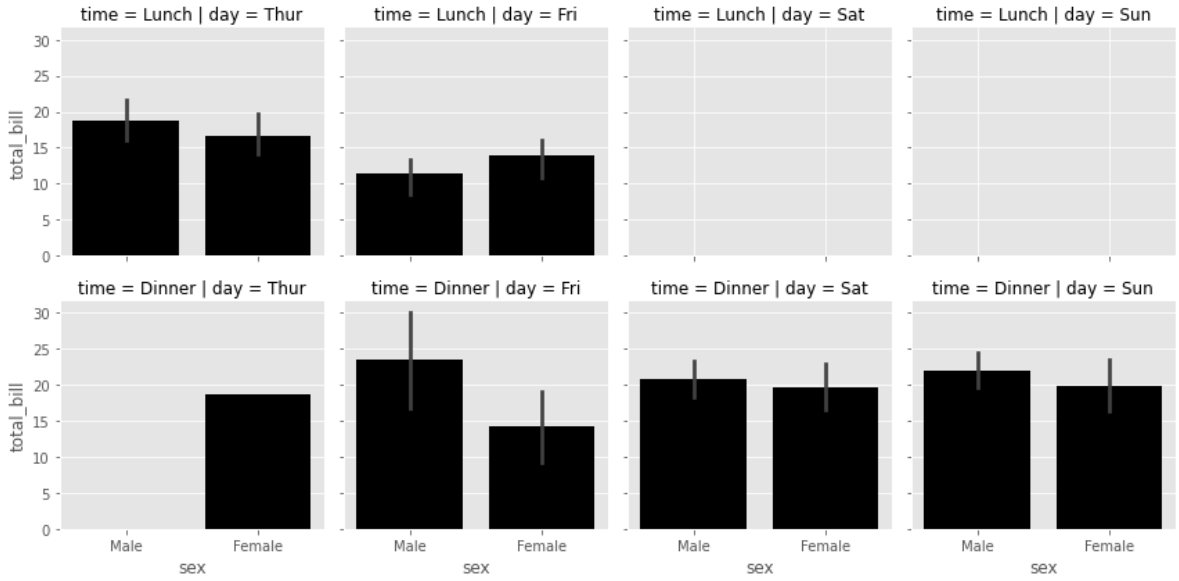
```
C:\Users\user\anaconda3\lib\site-packages\seaborn\axisgrid.py:670: UserWarnin
g: Using the violinplot function without specifying `order` is likely to prod
uce an incorrect plot.
  warnings.warn(warning)
```

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')
```

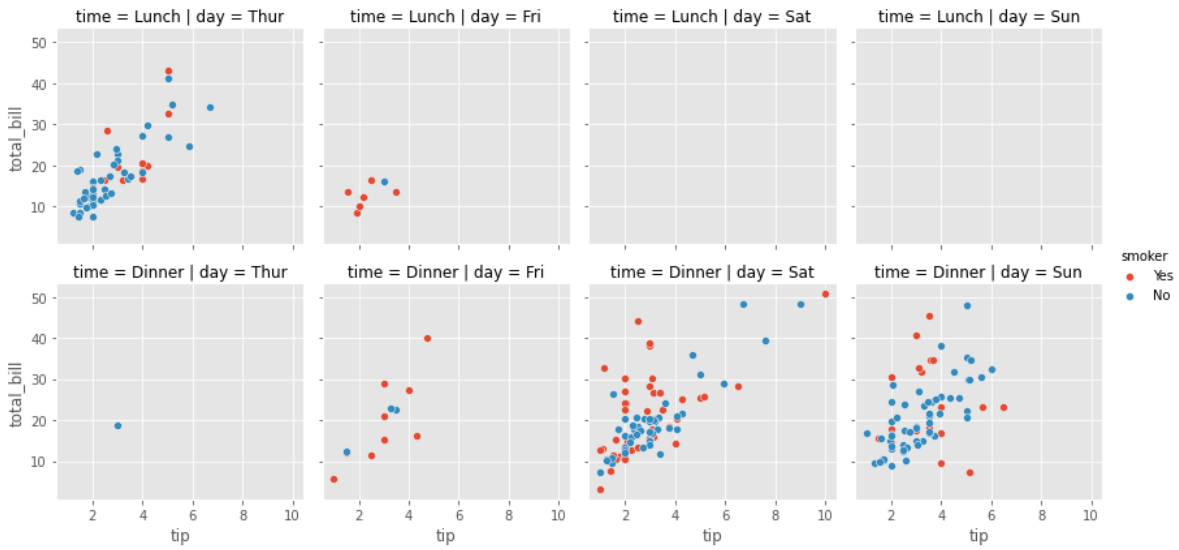
```
C:\Users\user\anaconda3\lib\site-packages\seaborn\axisgrid.py:670: UserWarnin
g: Using the barplot function without specifying `order` is likely to produce
an incorrect plot.
  warnings.warn(warning)
```

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.

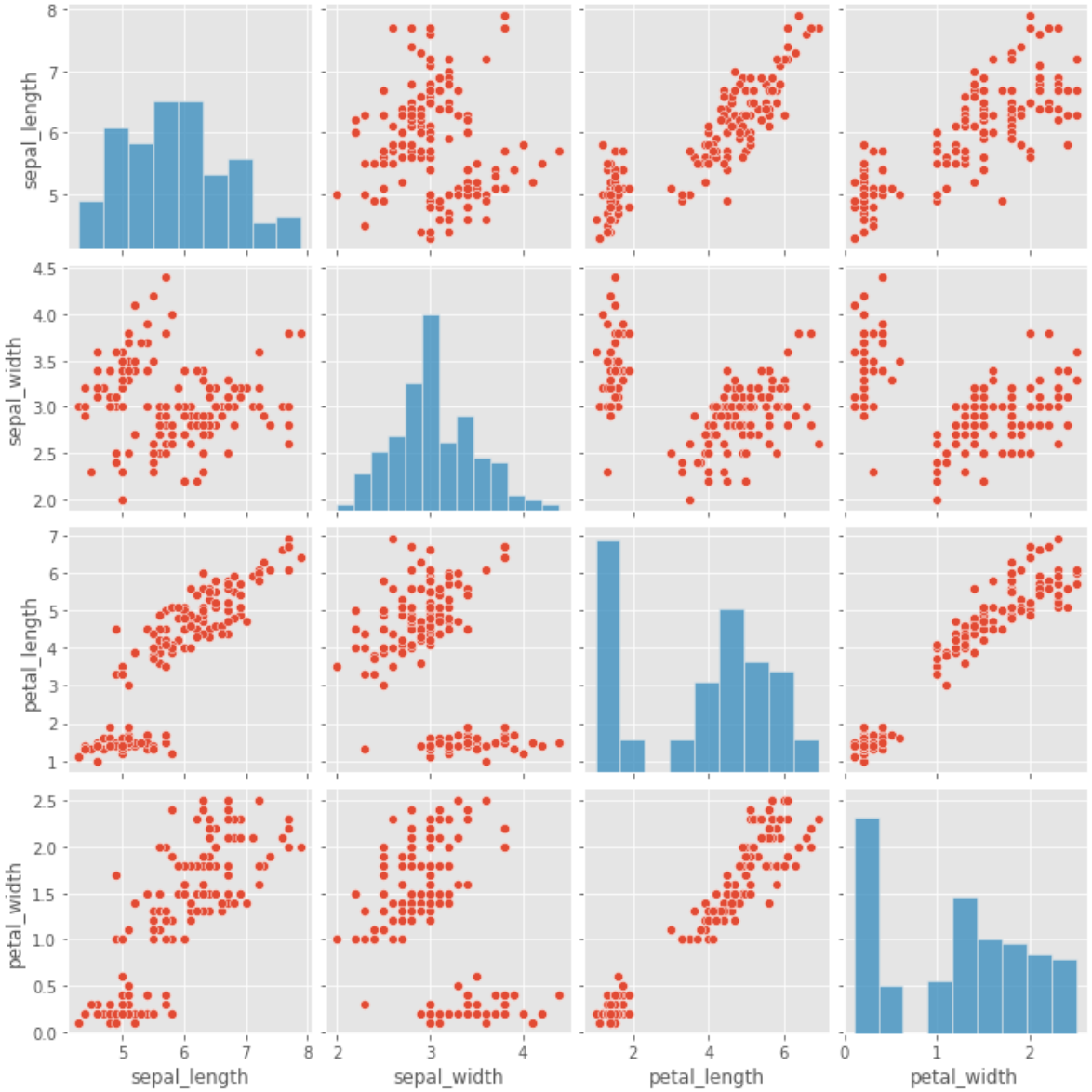
In [59]:
```
iris.sample()
```

Out[59]:

	sepal_length	sepal_width	petal_length	petal_width	species
53	5.5	2.3	4.0	1.3	versicolor

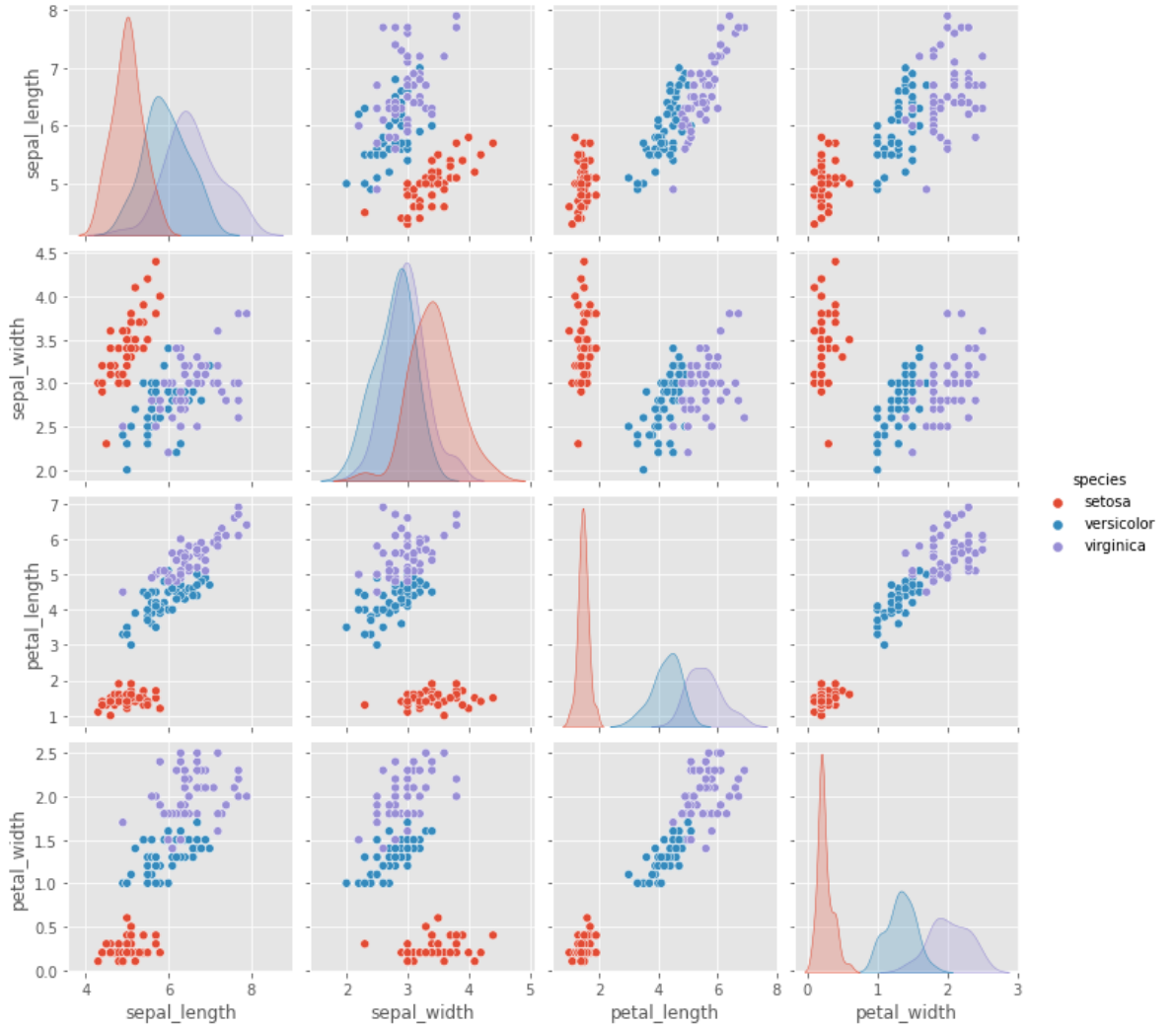
In [60]:
```
# Pairplot

sns.pairplot(iris)
```

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



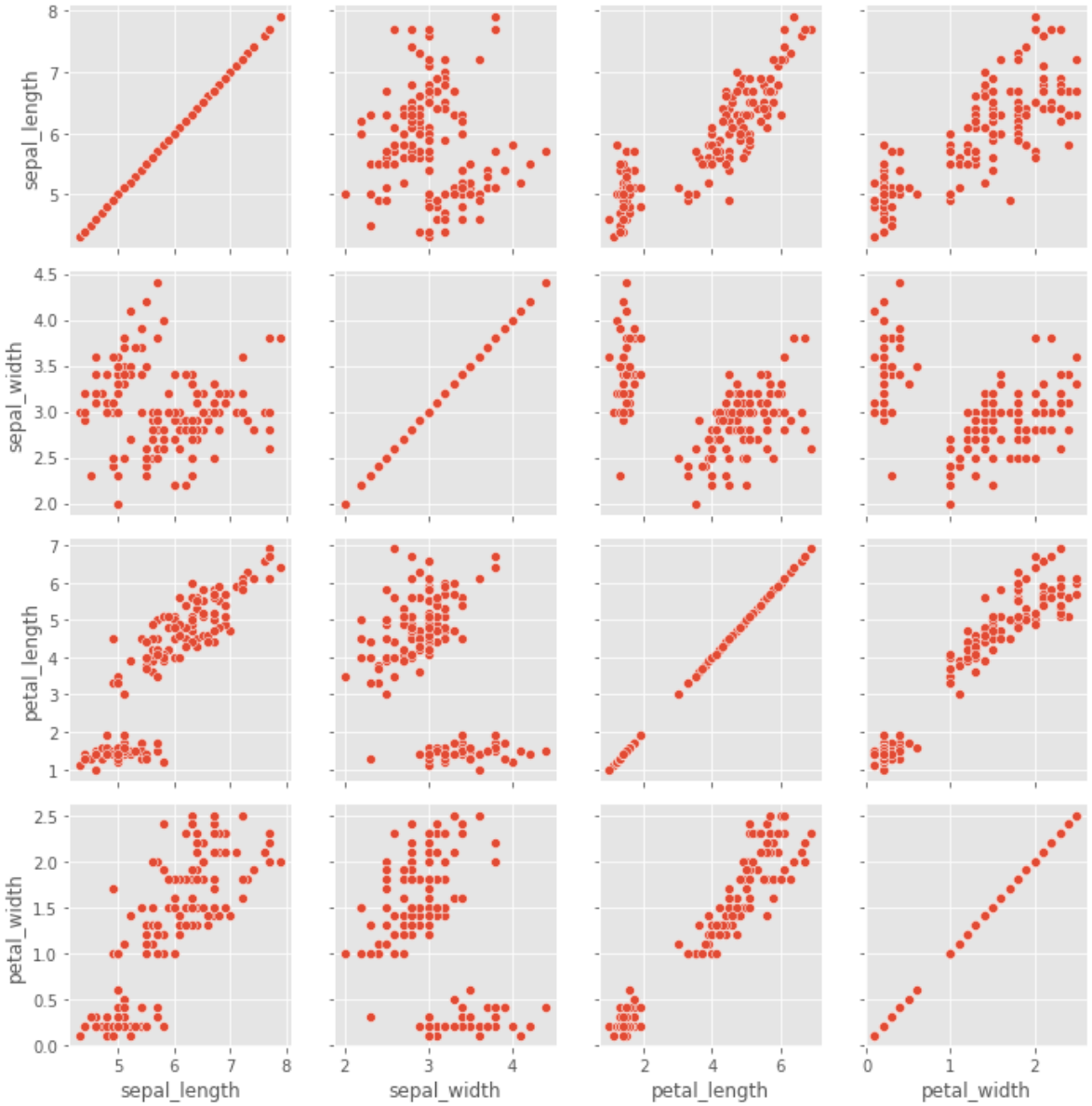
In [61]:
```
# hue

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

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



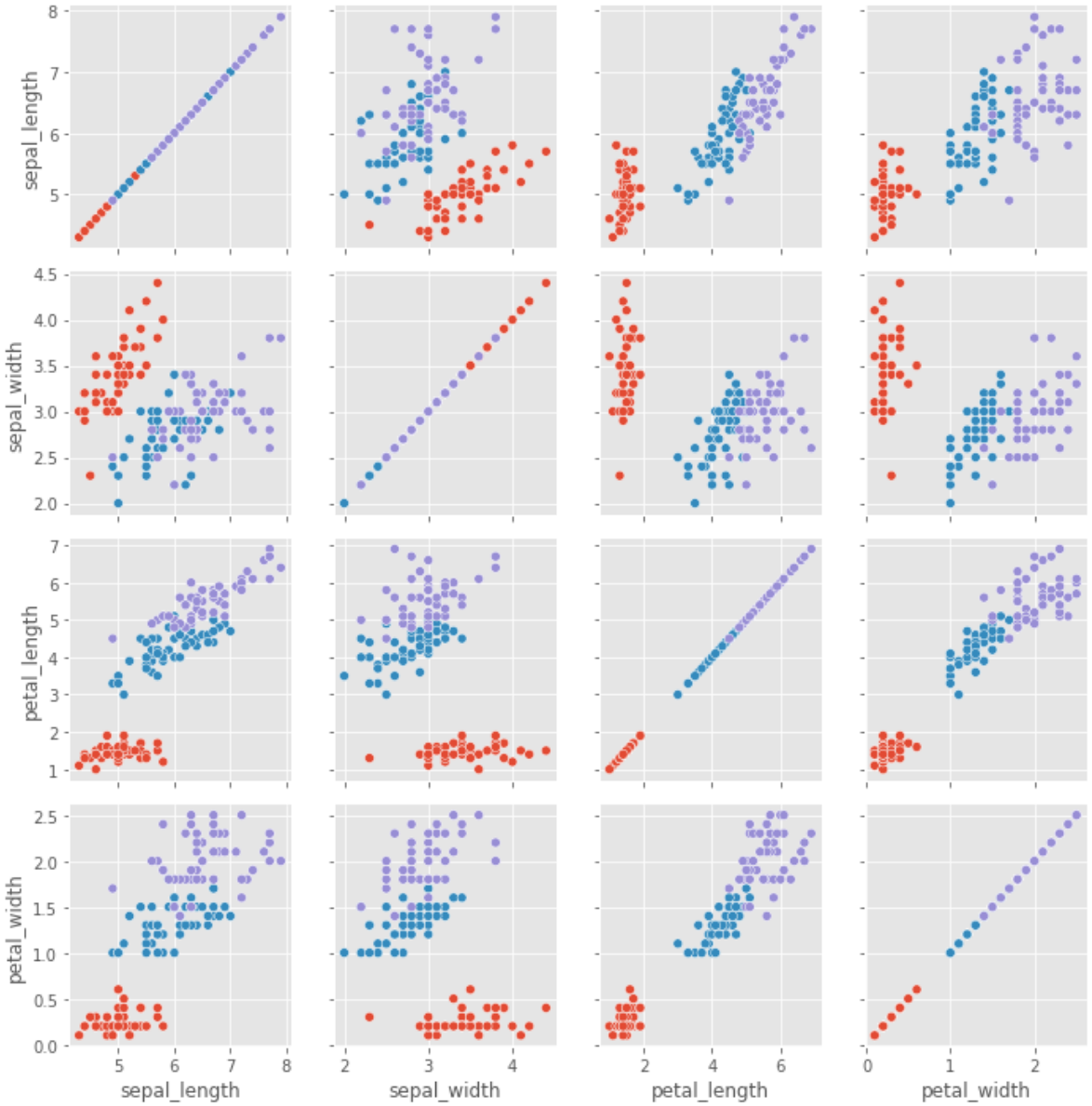
In [63]:
```
# pair grid

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)
```

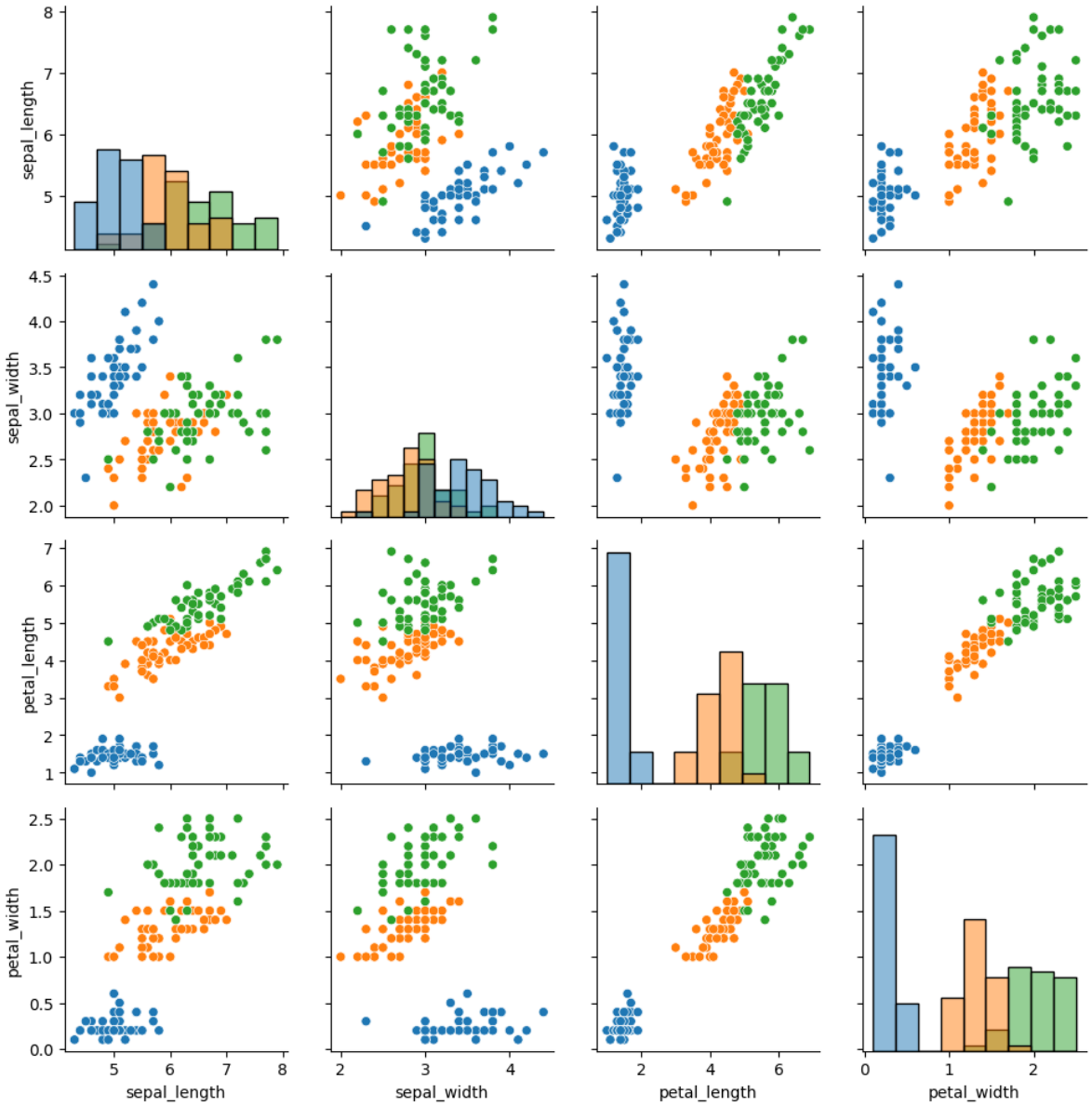
Out[64]: <seaborn.axisgrid.PairGrid at 0x1e870873640>



In [66]:
```
# style

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

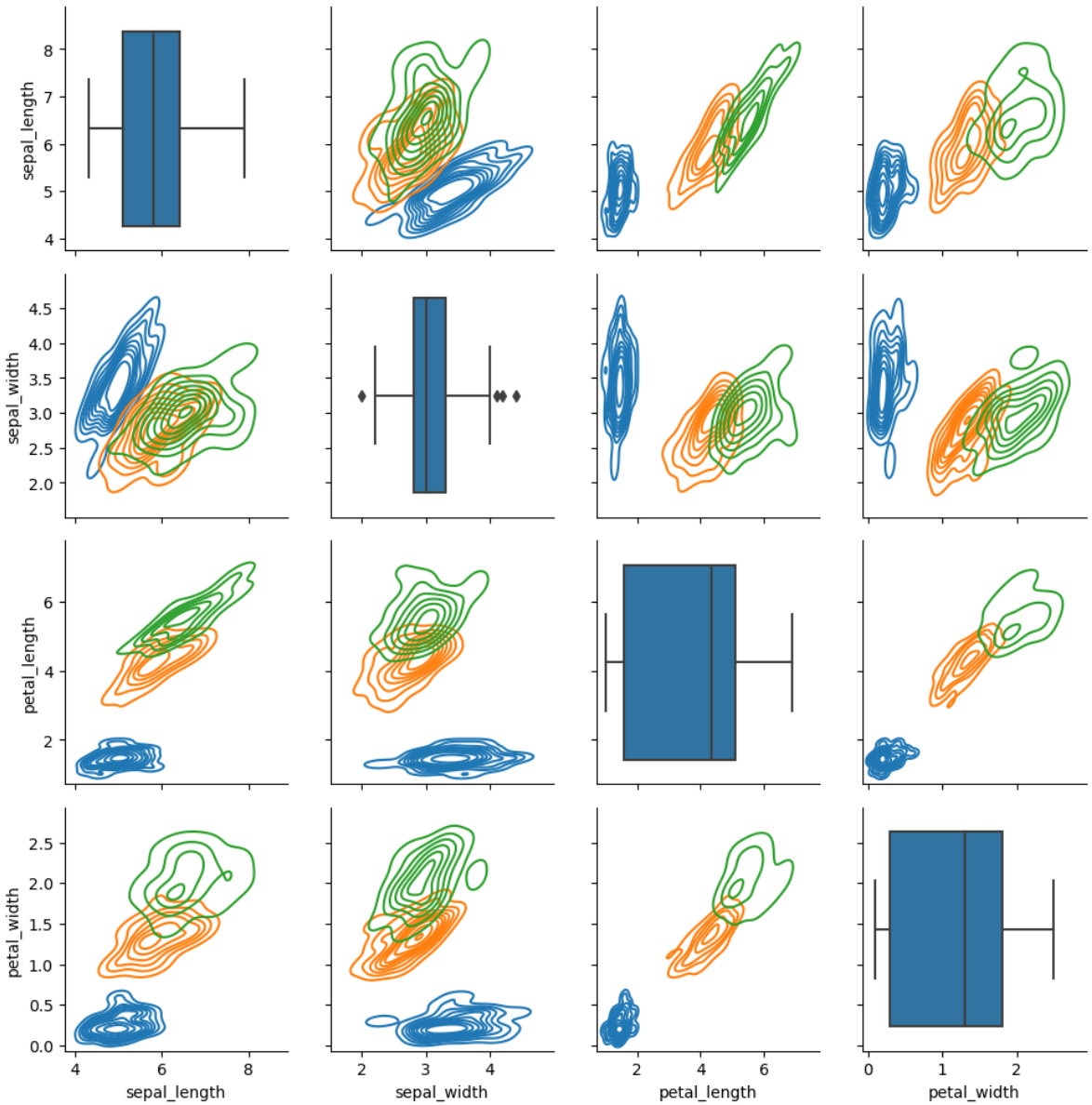
In [68]:
```
# map_diag -> map_offdiag

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

g.map_diag(sns. histplot)

g.map_offdiag(sns.scatterplot)
```

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



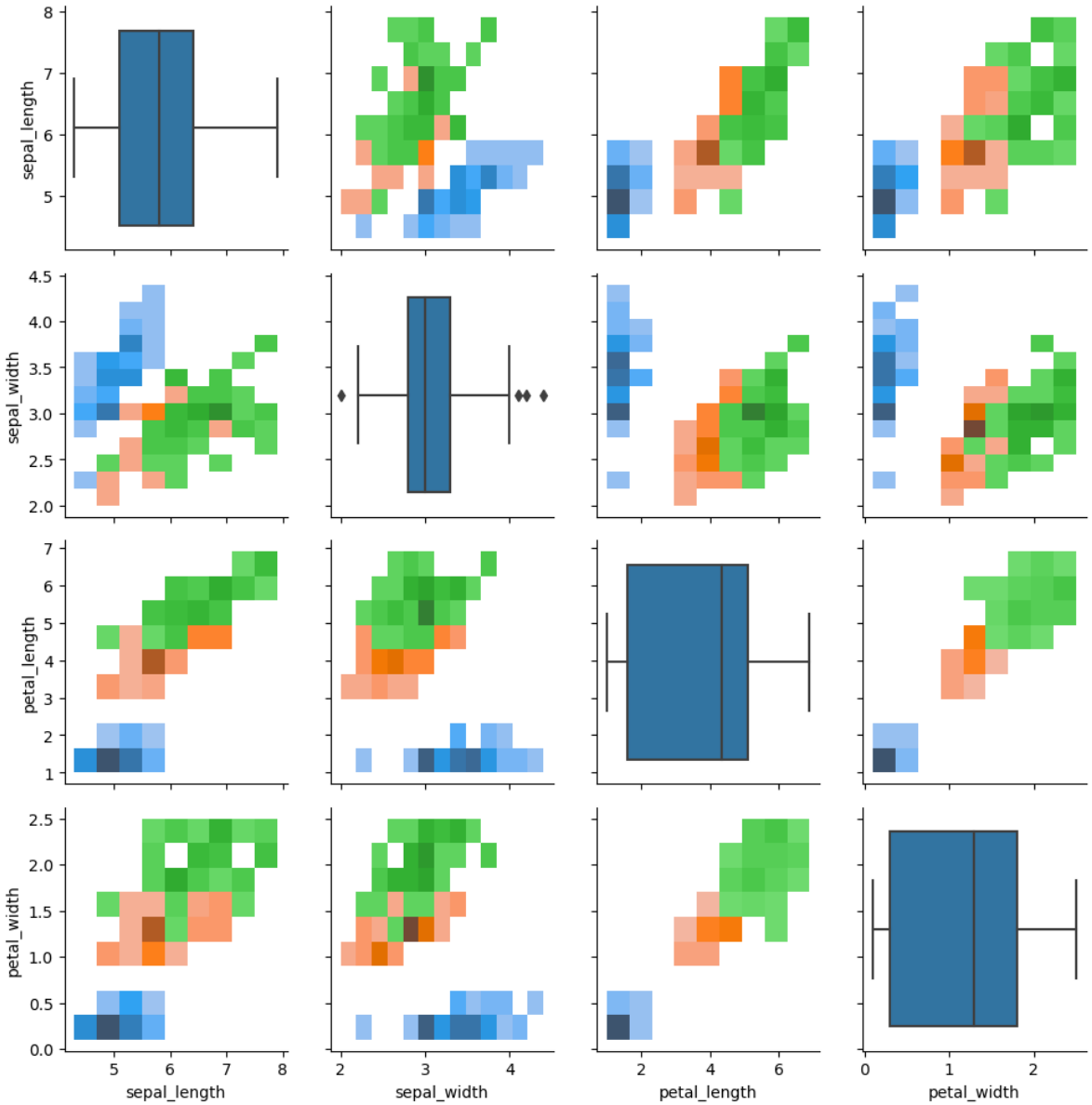
In [69]:
```
g = sns.PairGrid(data=iris,hue='species')

g.map_diag(sns.boxplot)

g.map_offdiag(sns.kdeplot)
```

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



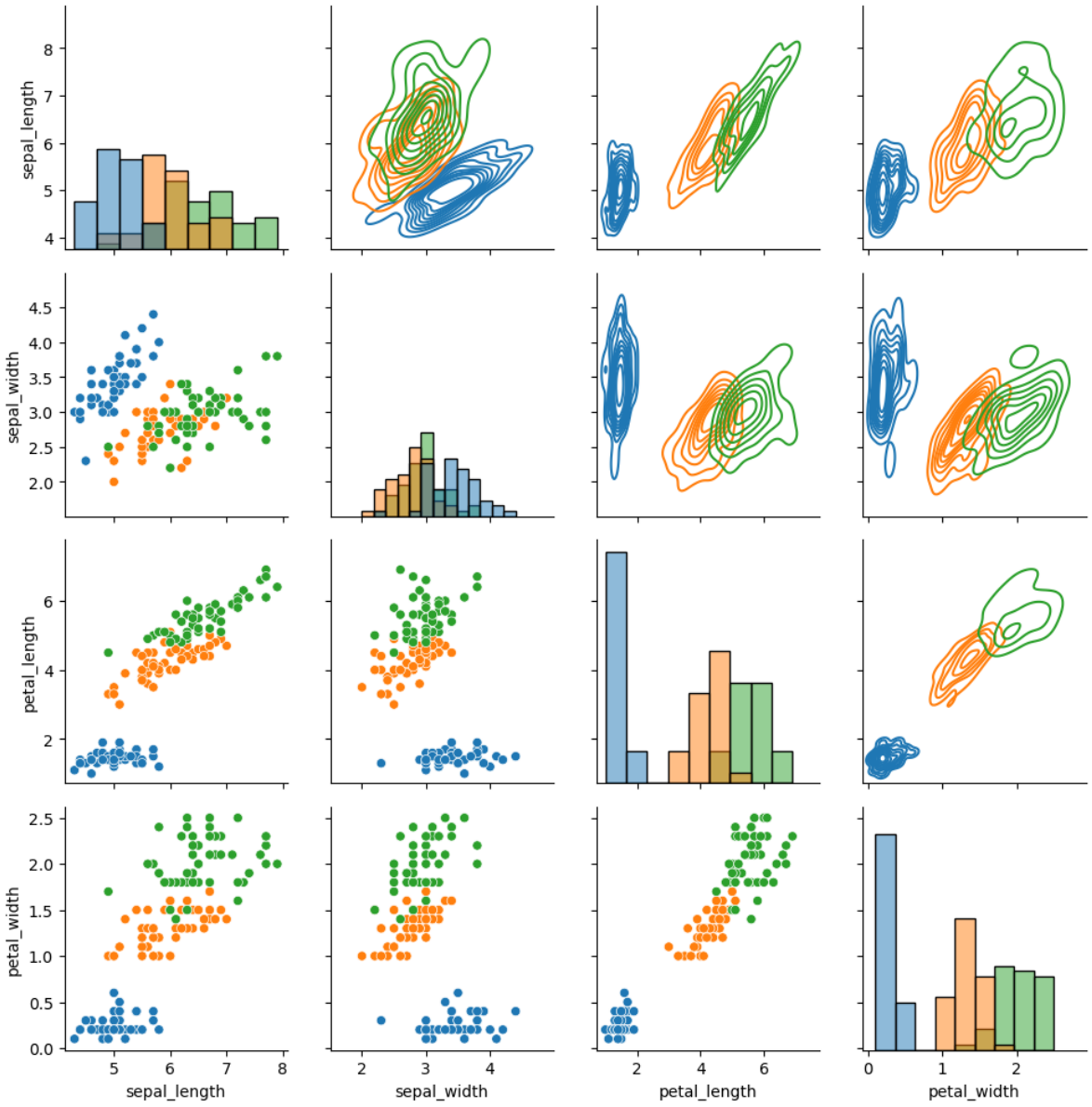
In [71]:
```
g = sns.PairGrid(data=iris,hue='species')

g.map_diag(sns.boxplot)

g.map_offdiag(sns.histplot)
```

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



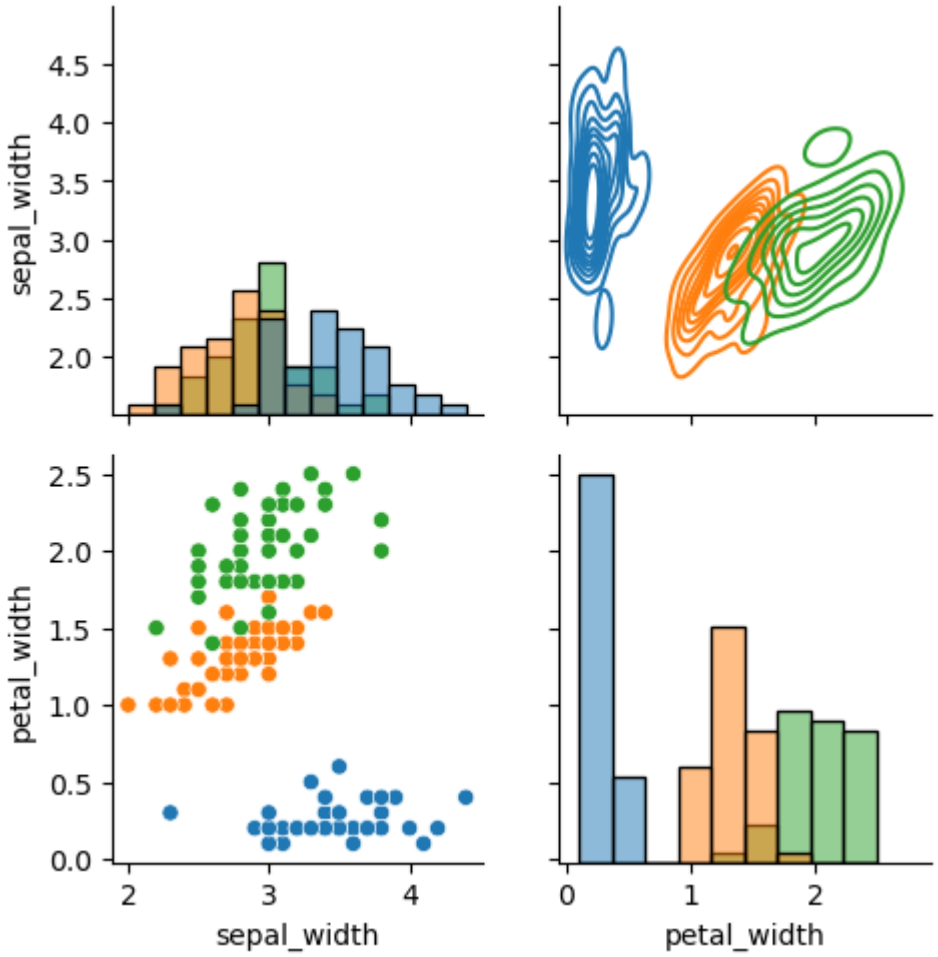
In [72]:
```
# map_diag -> map_upper -> map_lower

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>



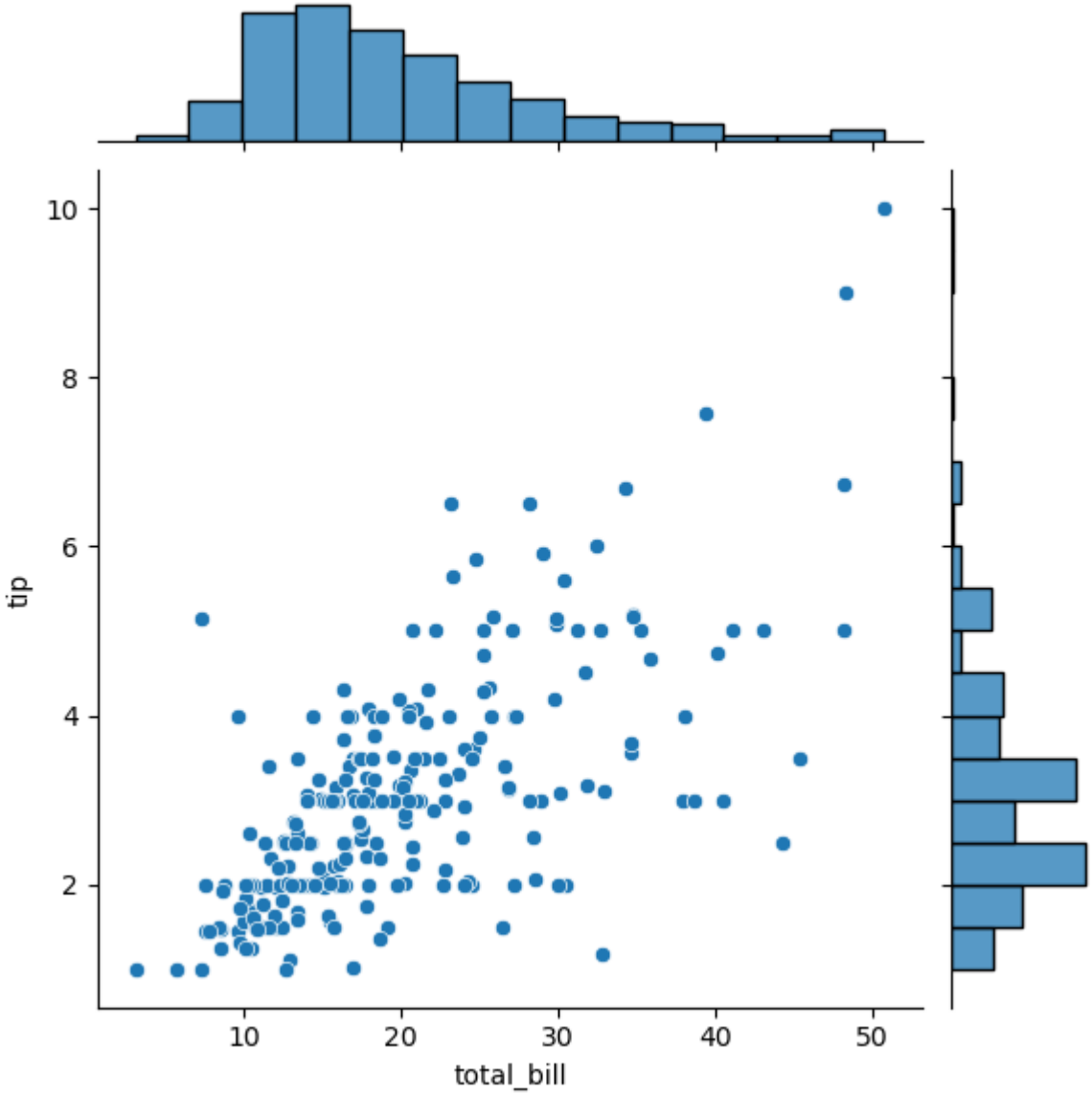
JointGrid Vs Jointplot

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

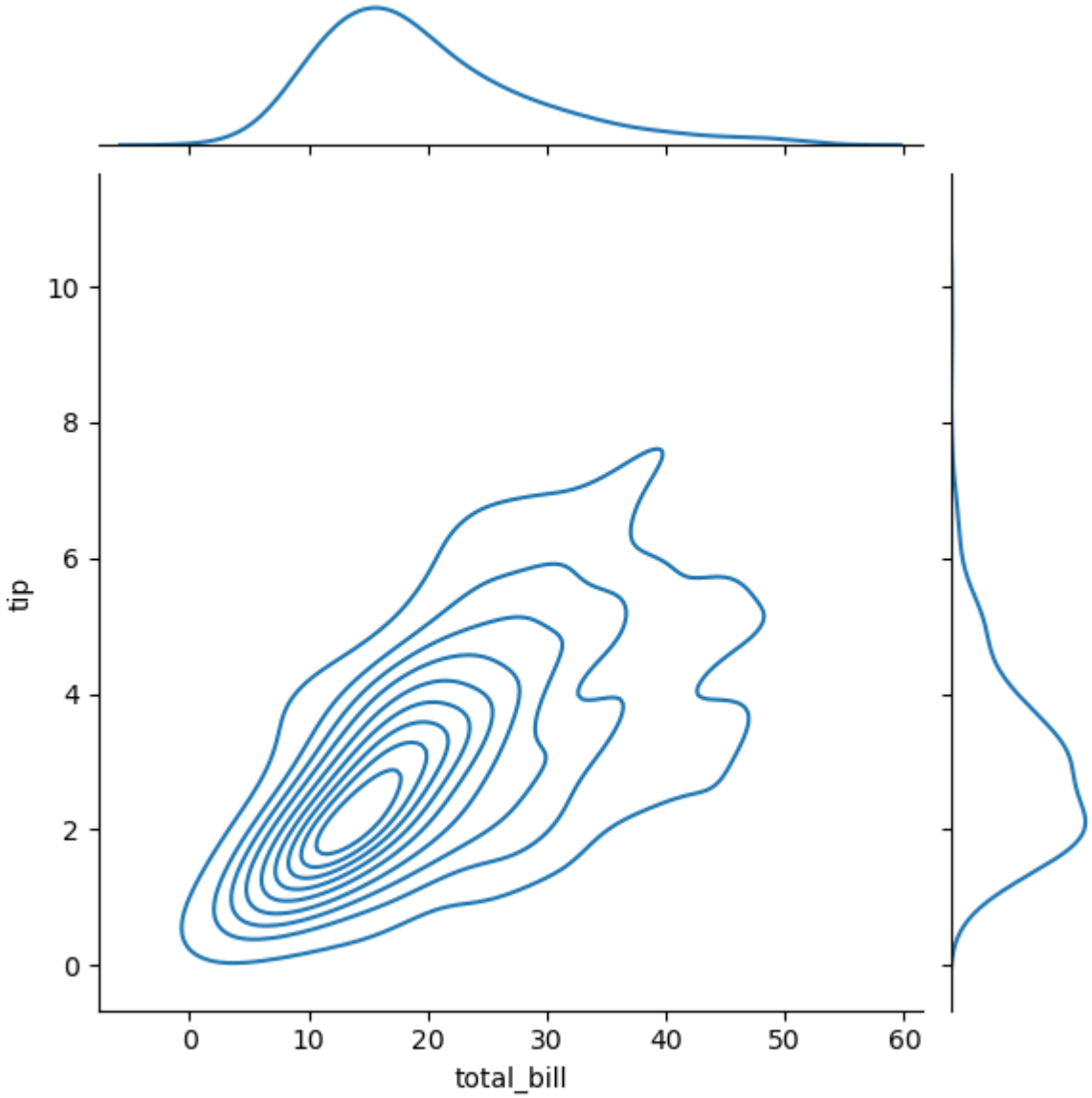
In [74]:
```
sns.jointplot(data=tips,x='total_bill',y='tip')
```

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



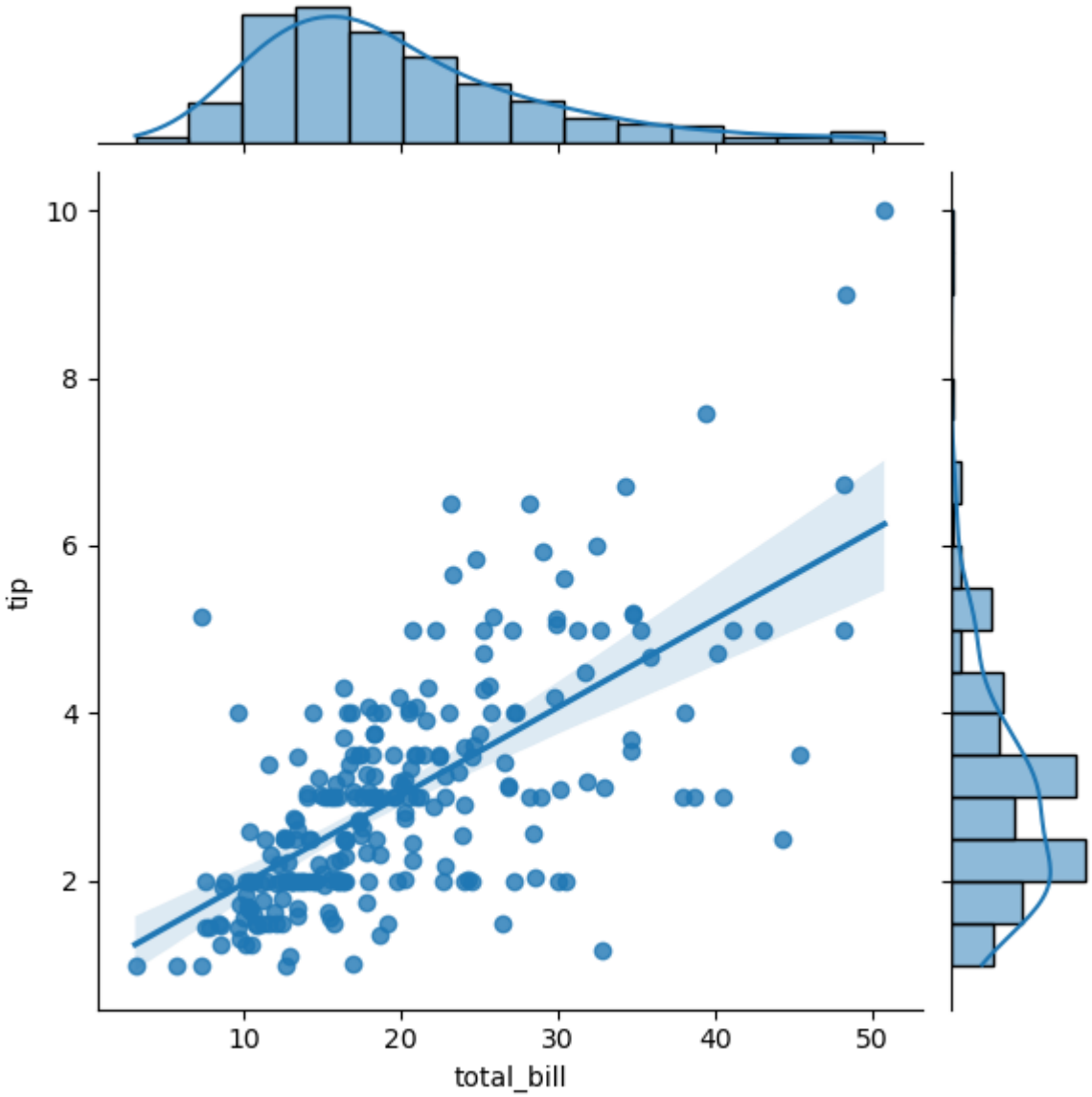
In [78]:
```
# kind

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

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



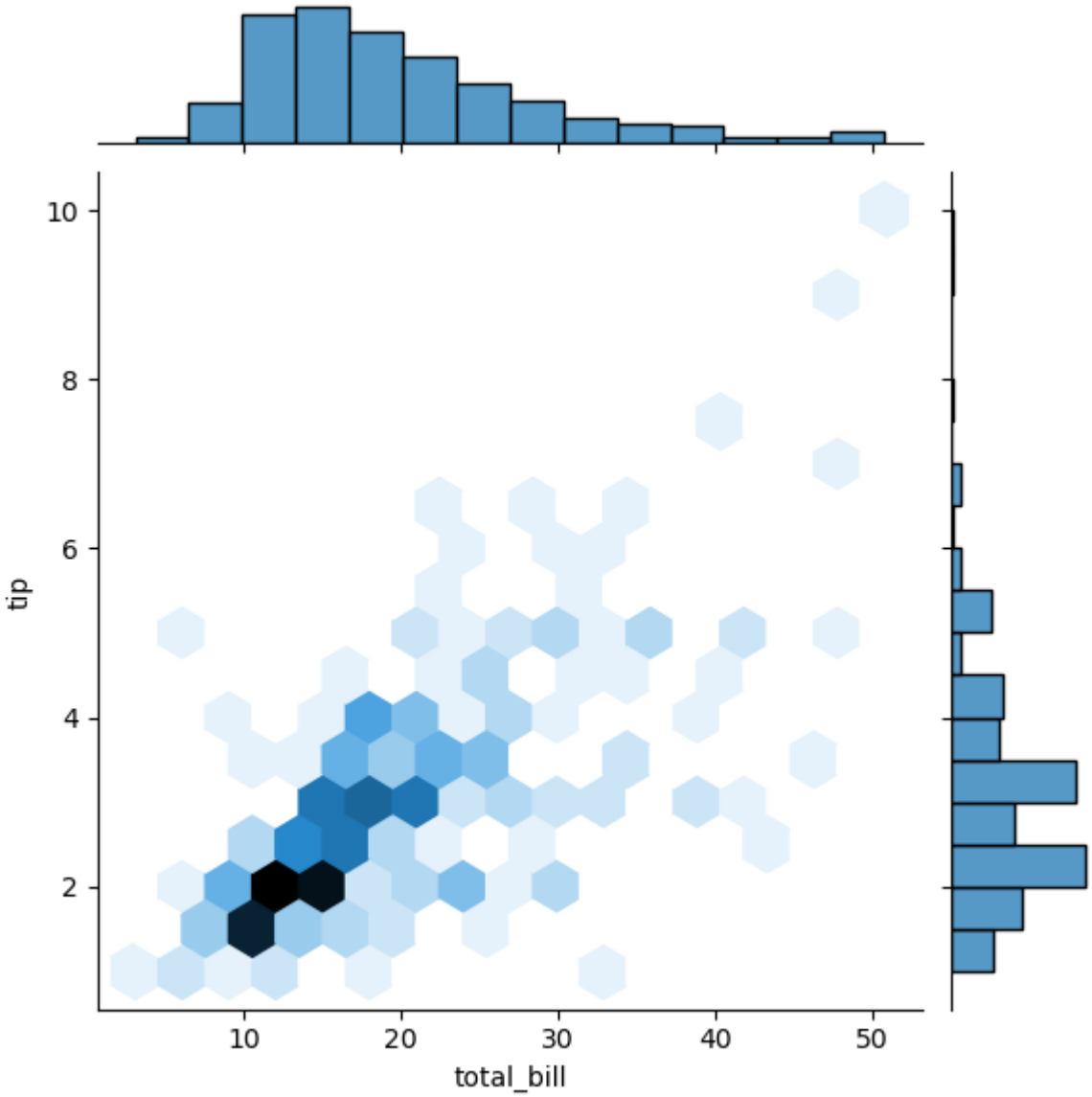
In [83]:
```
#regression line

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

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



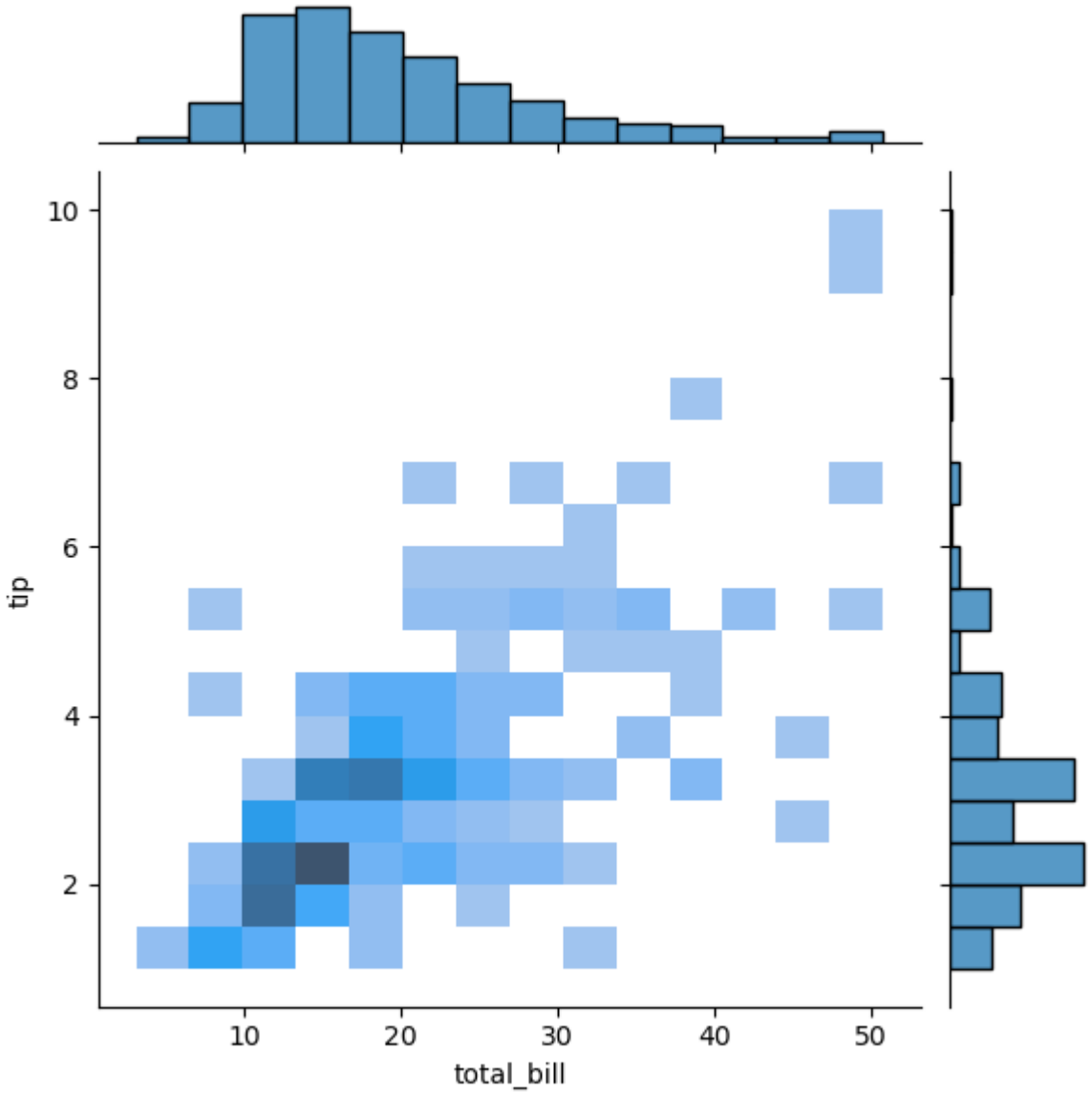
In [82]:
```
# hex

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

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



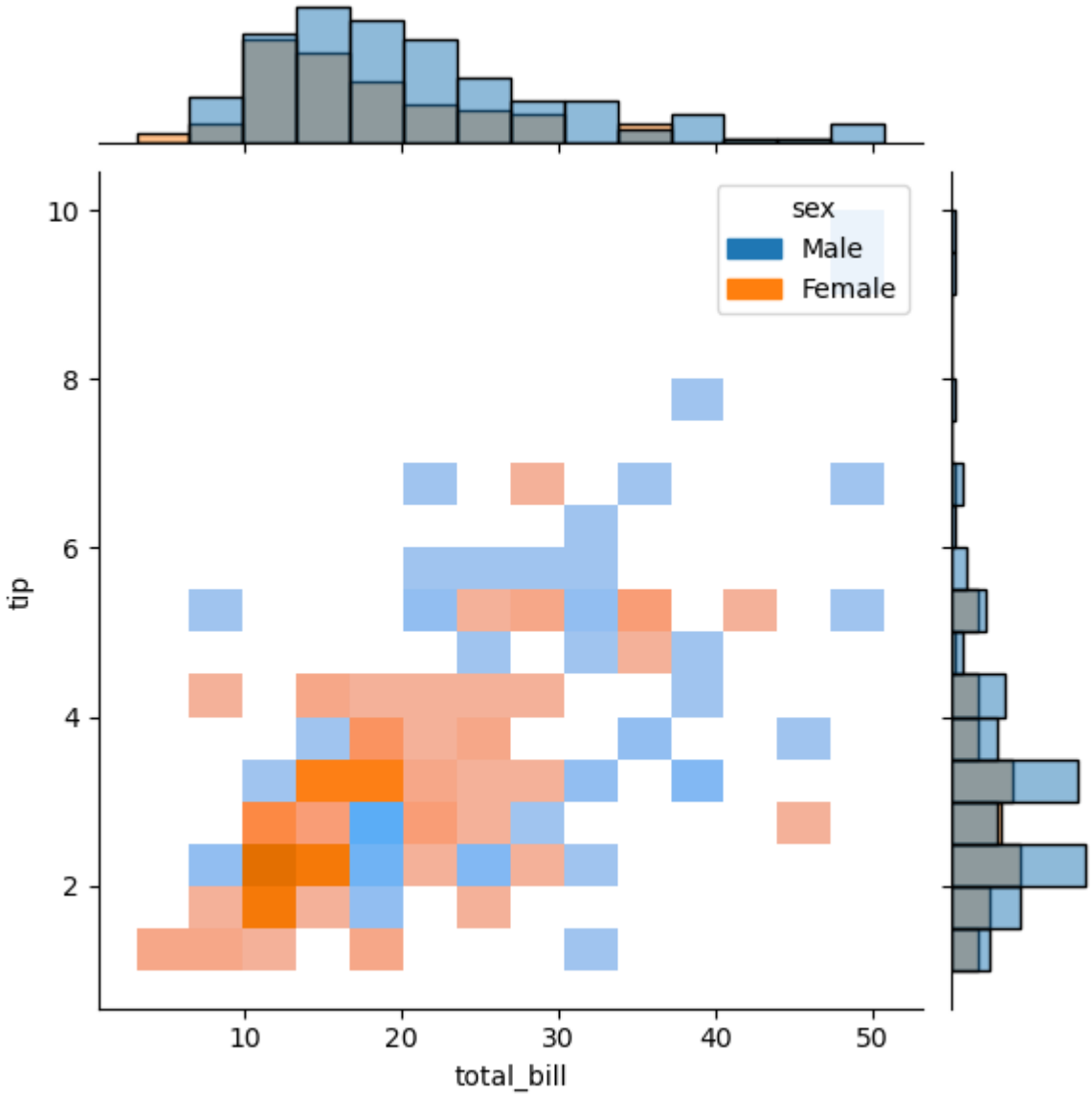
In [79]:
```
# kind -2d histogram + 1 histogram

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')
```

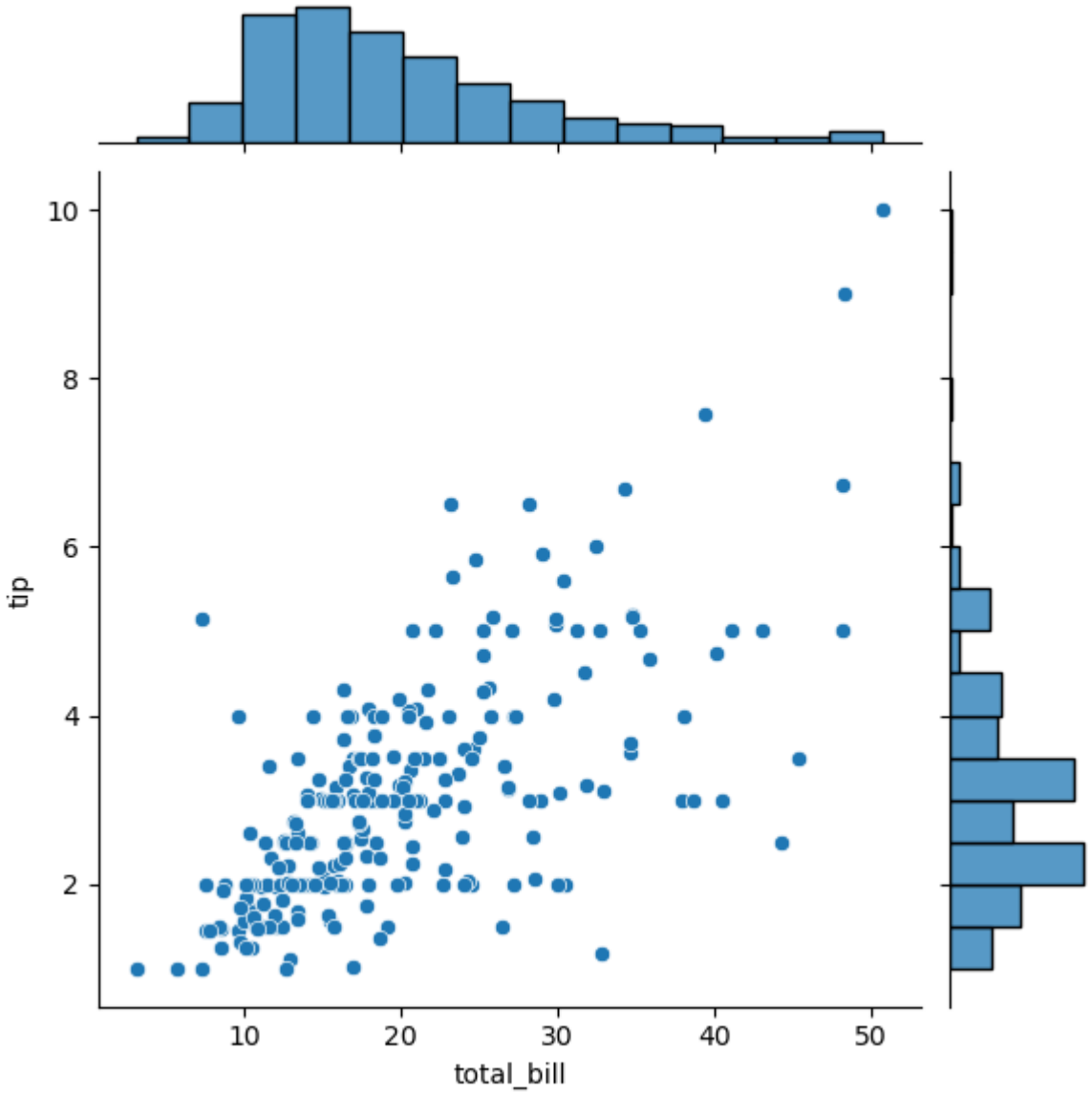
Out[76]:
```
<seaborn.axisgrid.JointGrid at 0x1e878736550>
```



Jointgrid

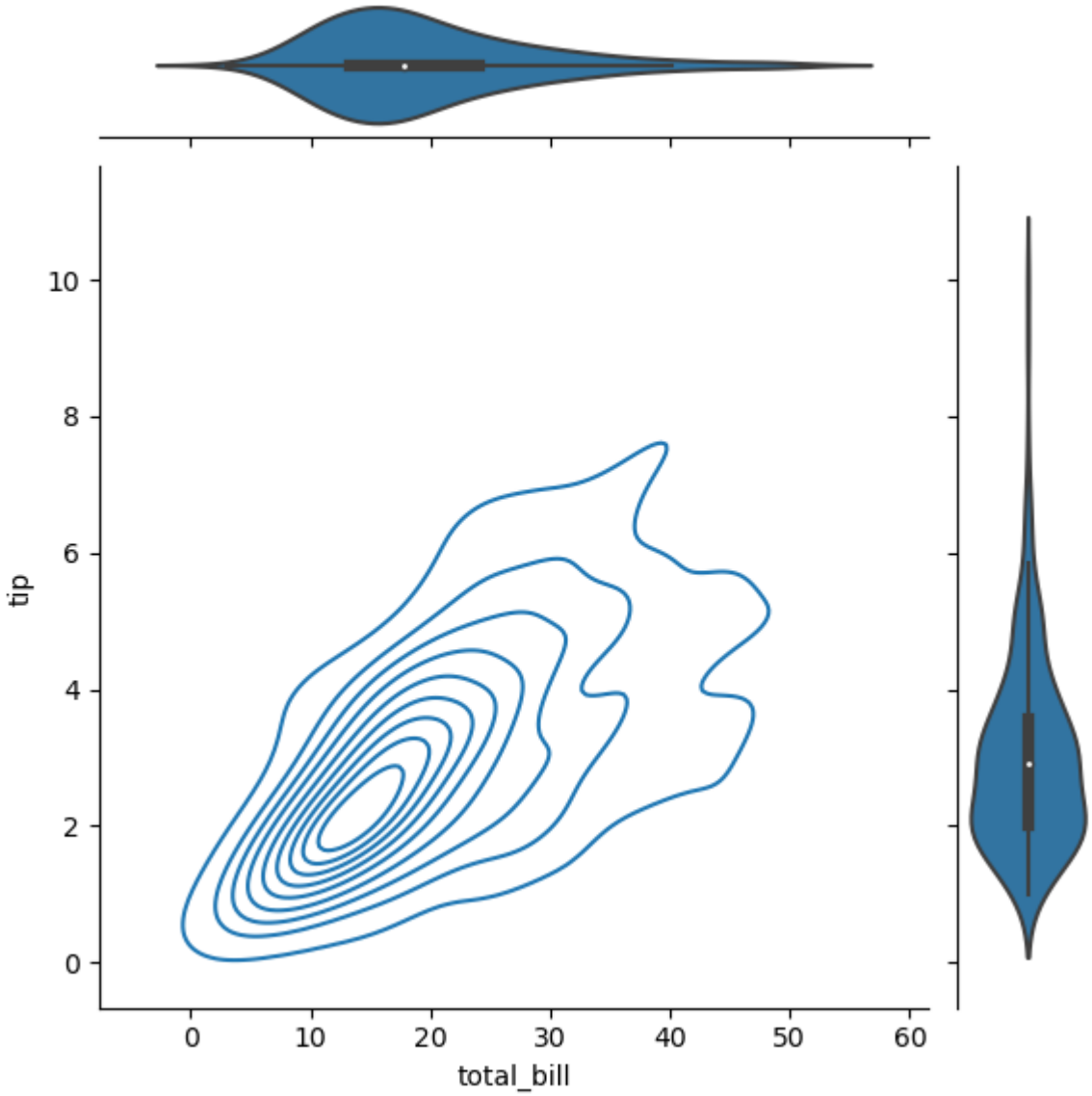
In [84]:
```
# we can change plots as per our requirements

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

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

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



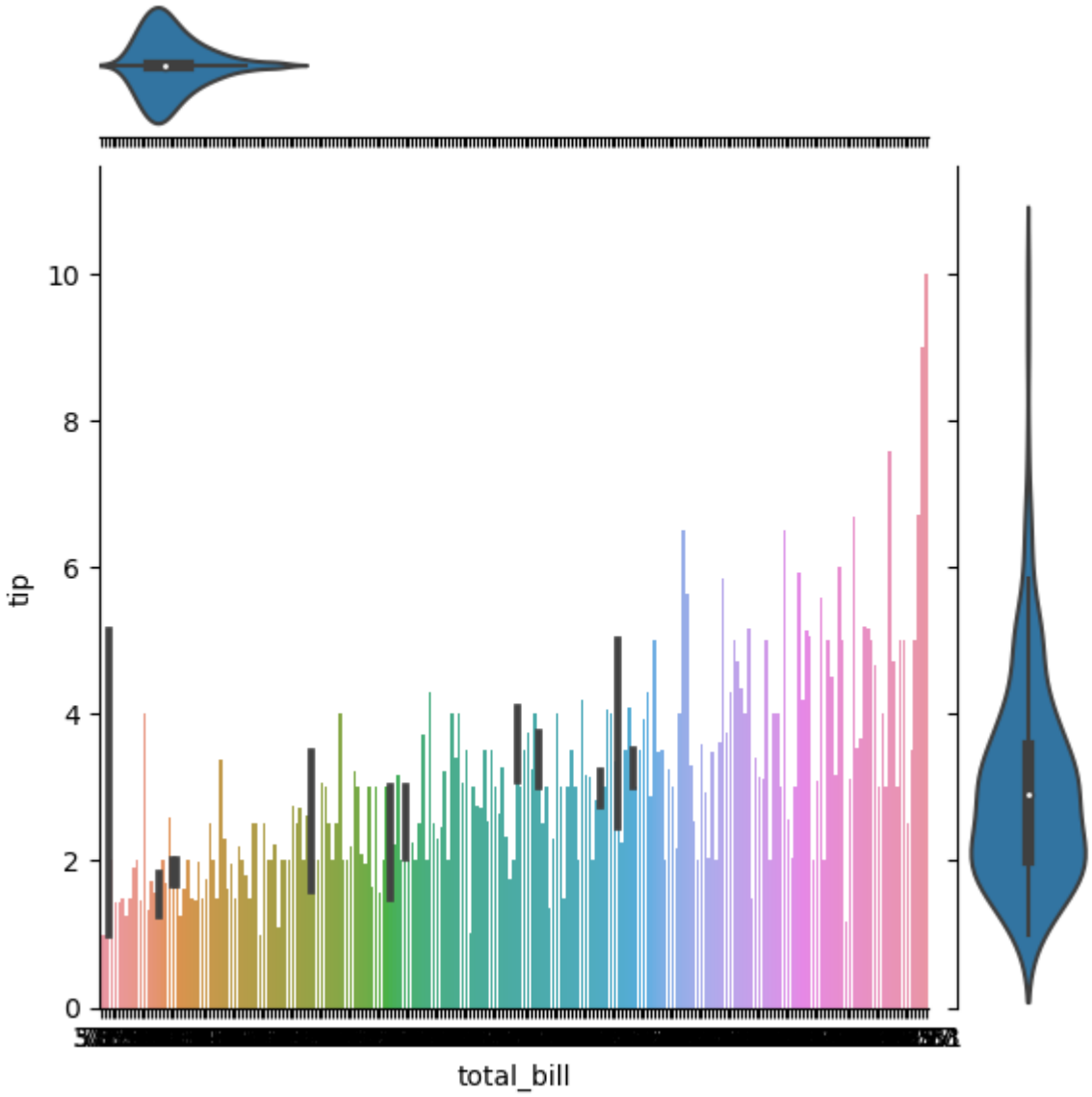
In [85]:
```
g = sns.JointGrid(data=tips,x='total_bill',y='tip')

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 [ ]:
```

```