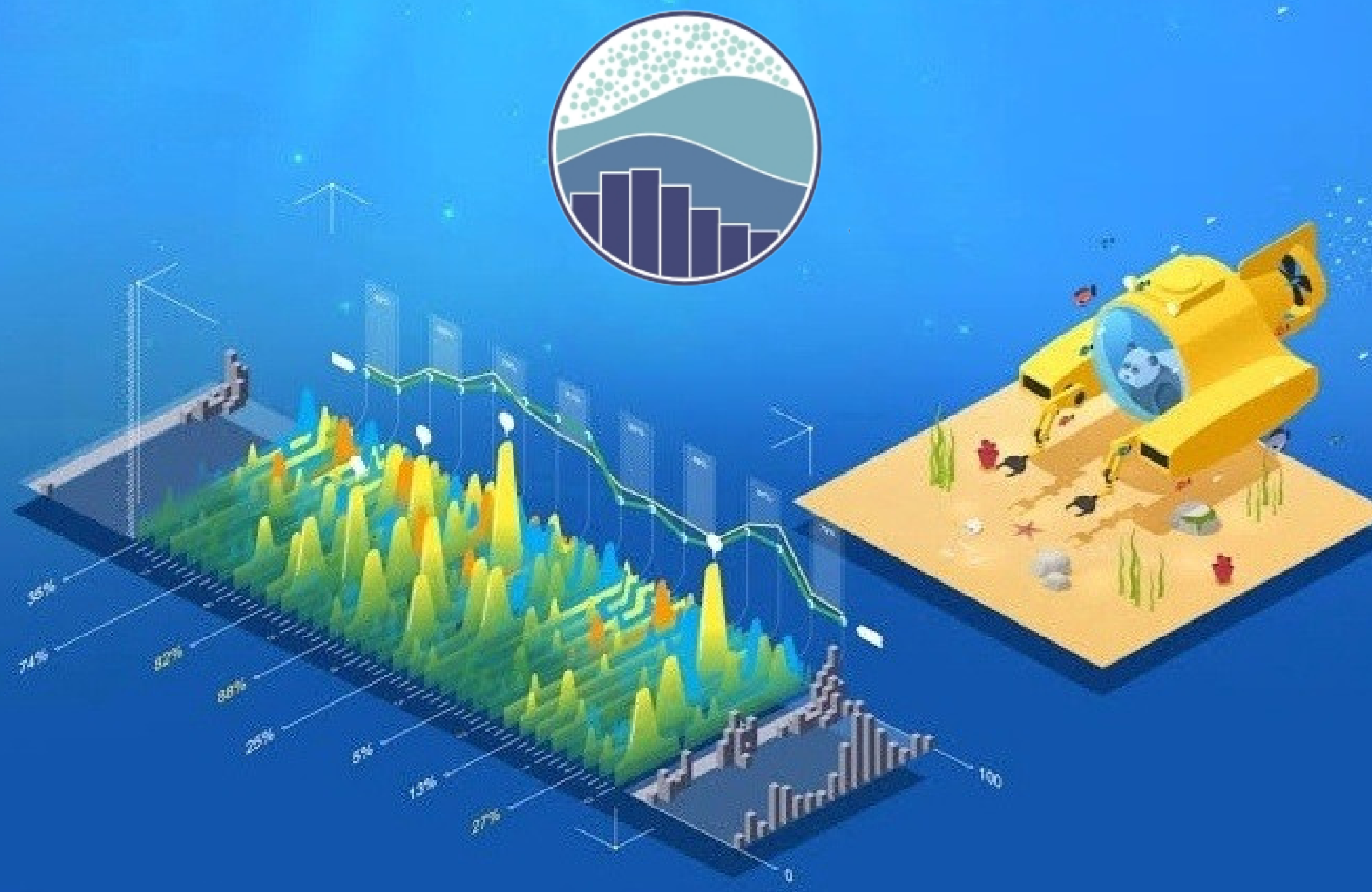
SEABORN



BEGINNER'S CODE GUIDE

Ehsan Paydar
EPSOFT · he/him

EPSOFT
Everywhere
epsoft98@gmail.com
www.epsoft.ir
@epsoft98
in/ehsanpaydar
epsoft
epsoft98

seaborn

-Stats visuals in Python
-Matplotlib + pandas enhancer

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# creating Data
x = [1,2,3]
y = [4,5,6]
```

```
# Seaborn Line Graph
sns.lineplot(x=x,y=y)

plt.show()
```

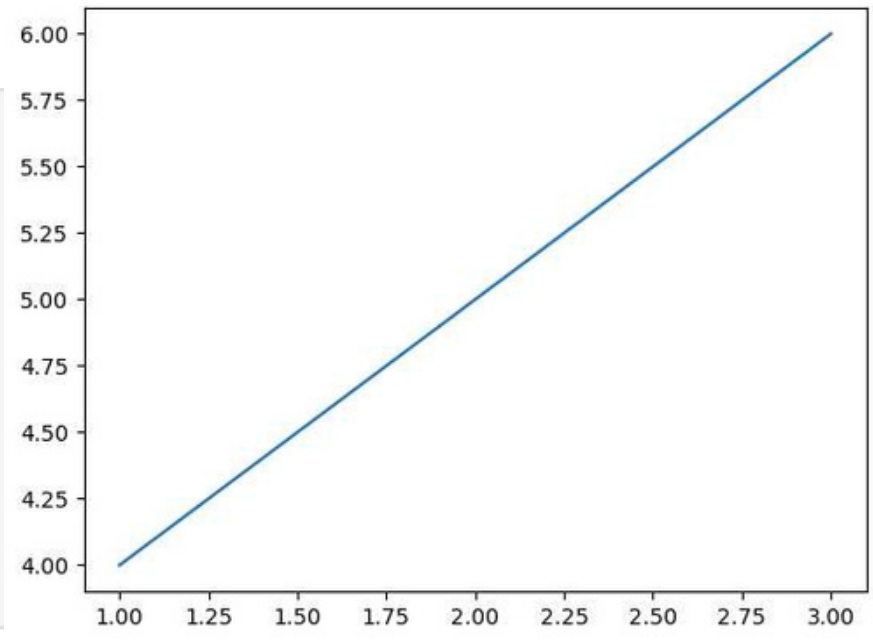
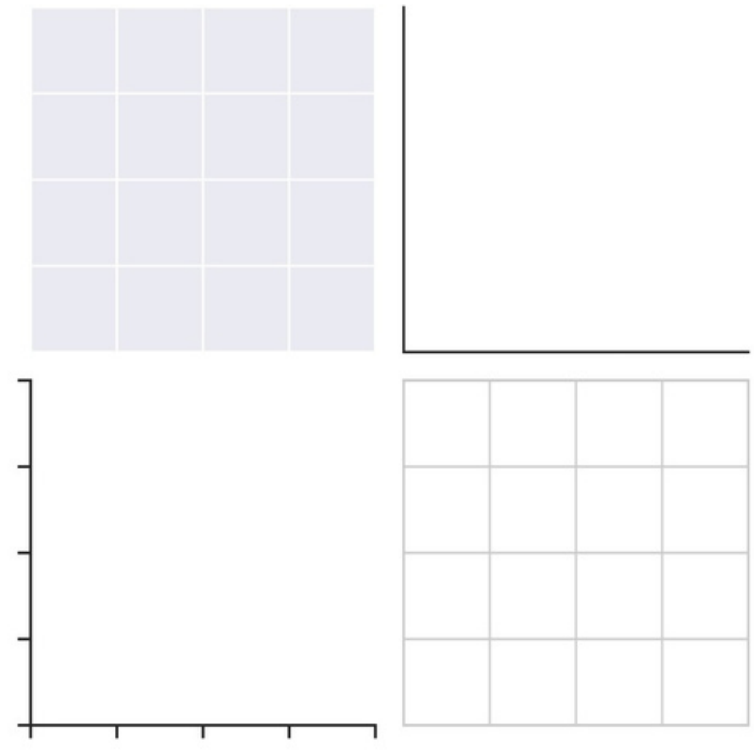


Figure Styles

5 Themes :

-Darkgrid
-Whitegrid
-Dark
-White
-Ticks



Darkgrid

Function set(), set_theme() and set_style("darkgrid") act same

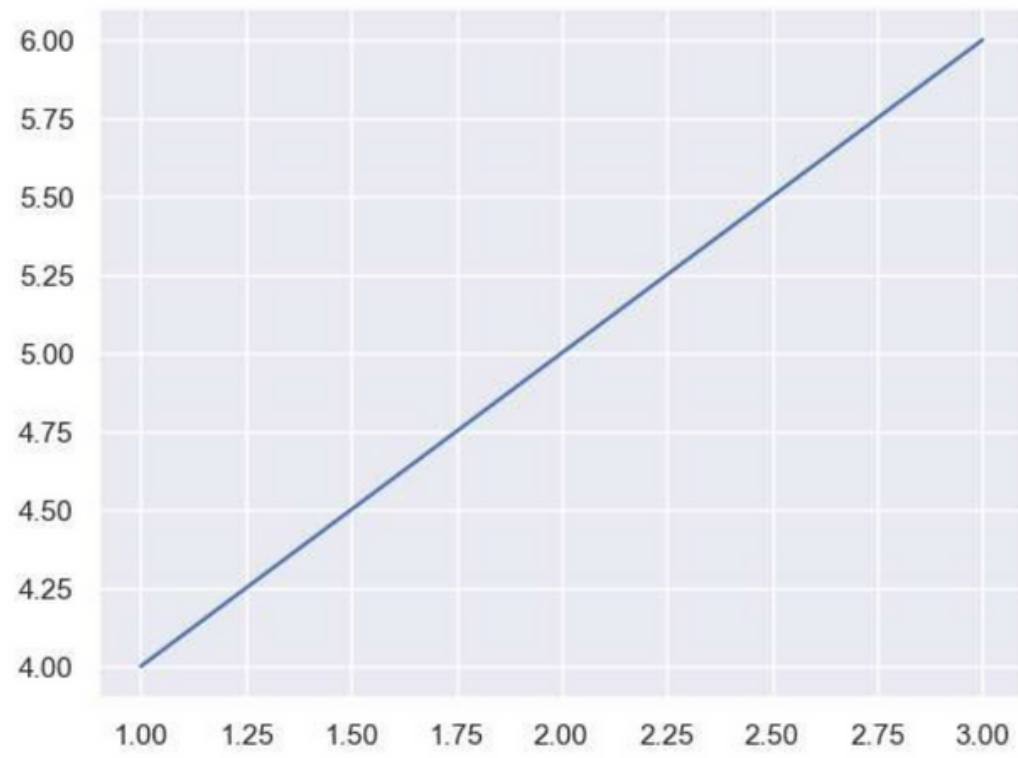
```
sns.set()

sns.lineplot(x=x,y=y)
```

or

```
sns.set_theme()
```

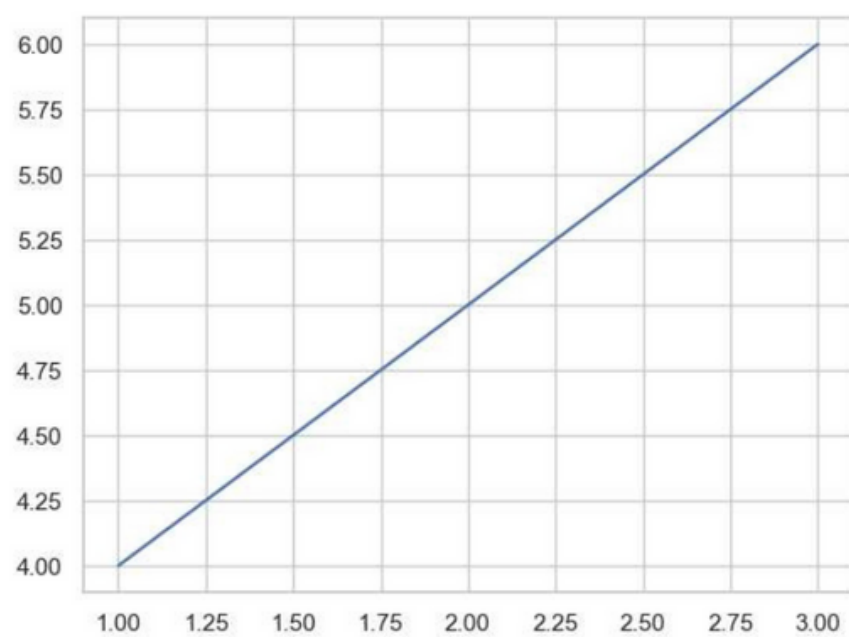
or

```
sns.set_style("darkgrid")
```



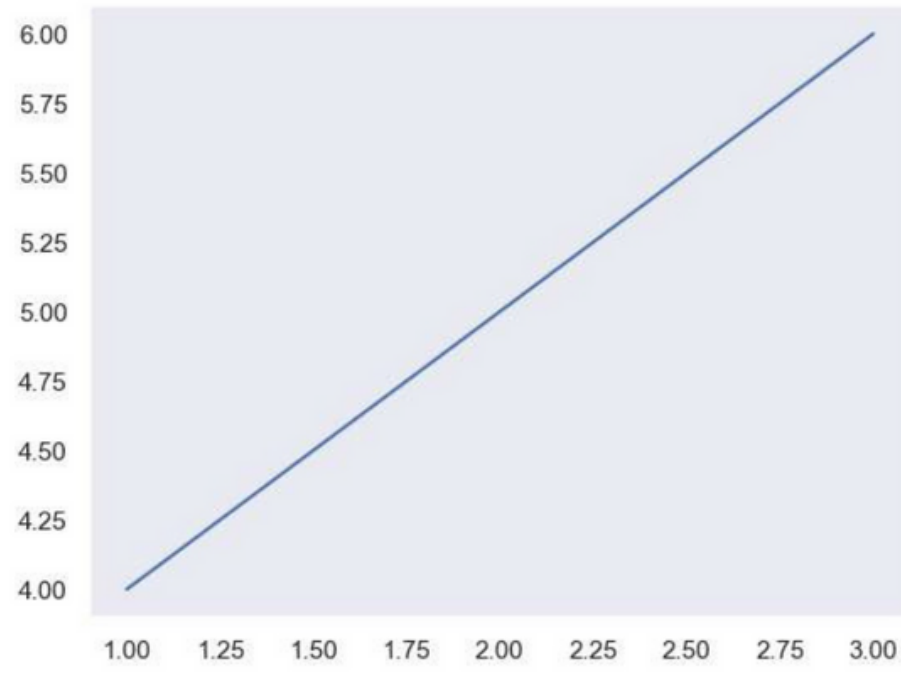
Whitegrid

```
sns.set_style("whitegrid")

sns.lineplot(x=x,y=y)
```



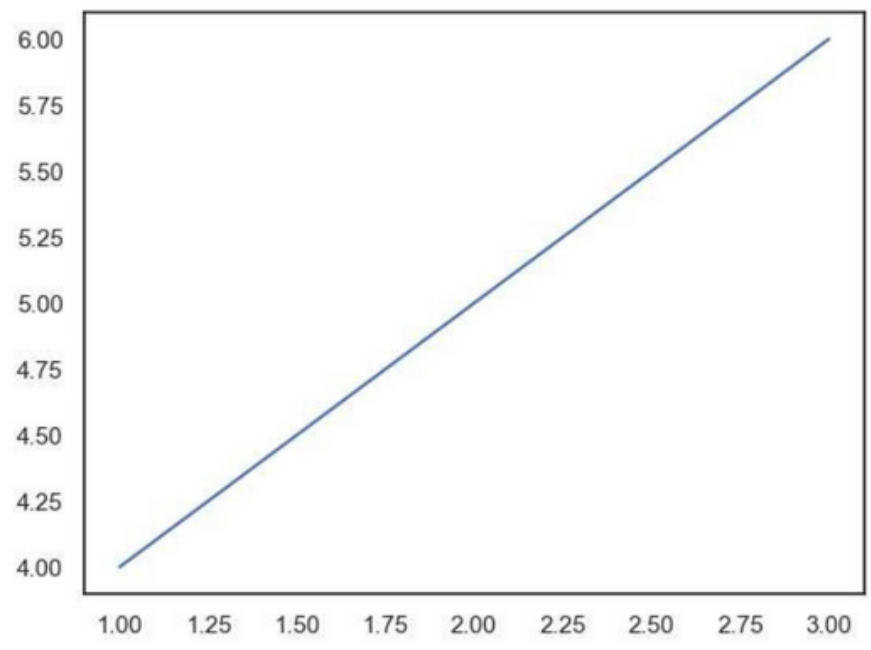
Dark

```
sns.set_style("dark")

sns.lineplot(x=x,y=y)
```



White

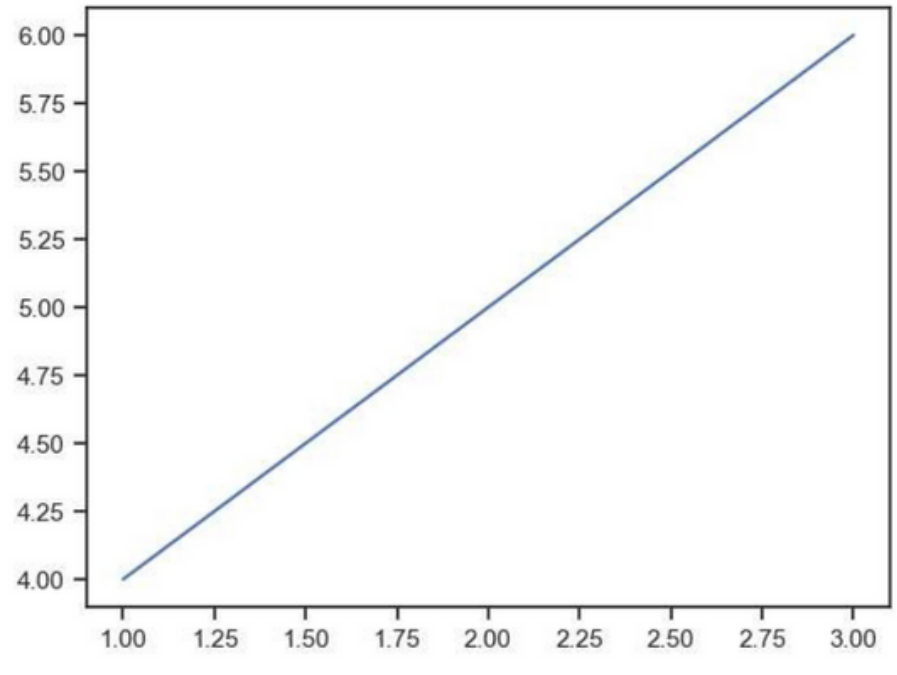
```
sns.set_style("white")

sns.lineplot(x=x,y=y)
```



Ticks

Extra Structure
on x-y-scale

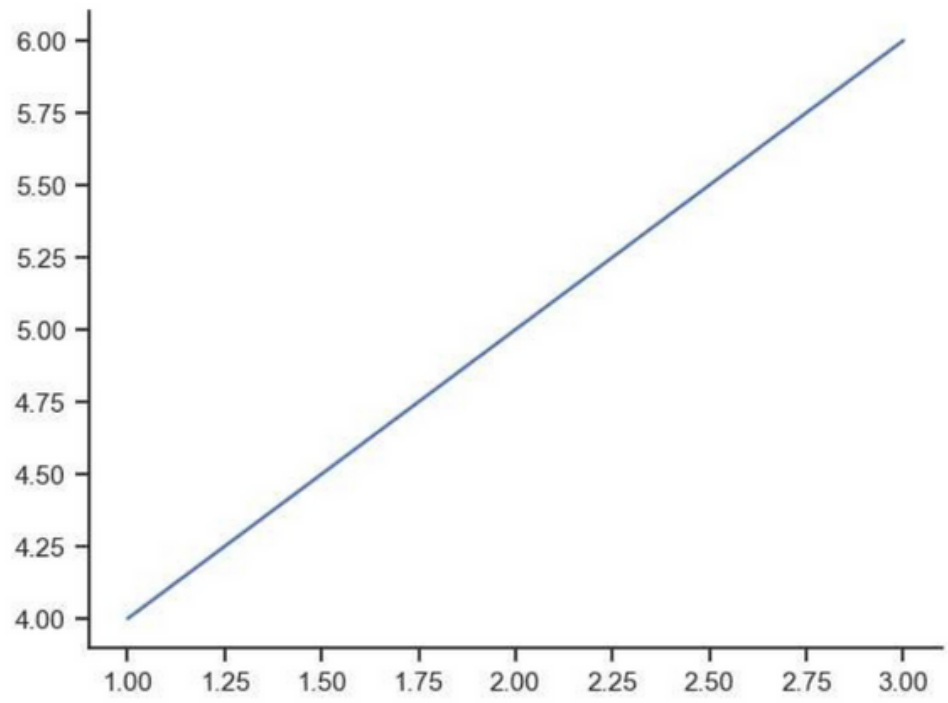
```
sns.set_style("ticks")

sns.lineplot(x=x,y=y)
```



despine()

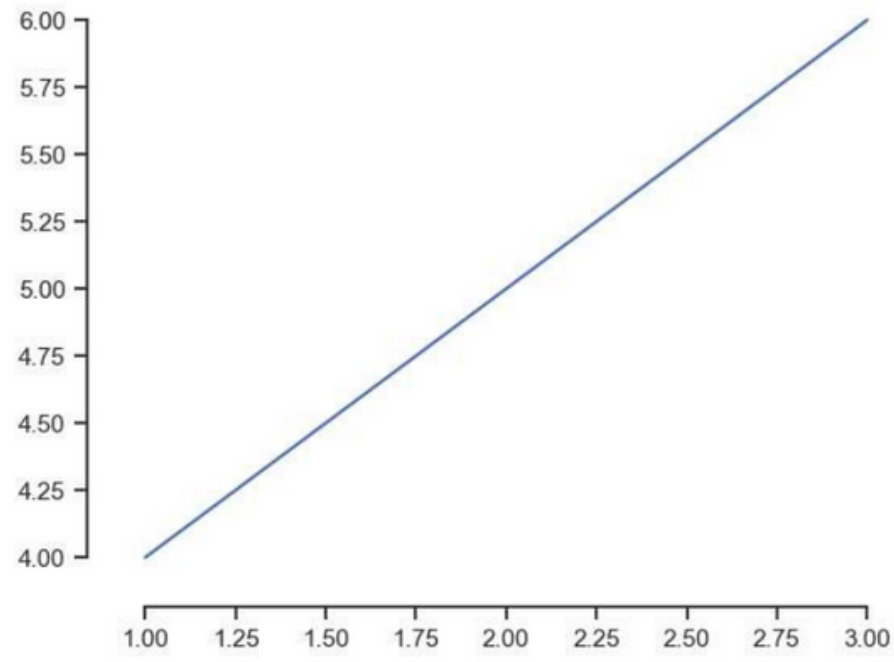
removing the top & right spines

```
sns.lineplot(x=x,y=y)

sns.despine()
```



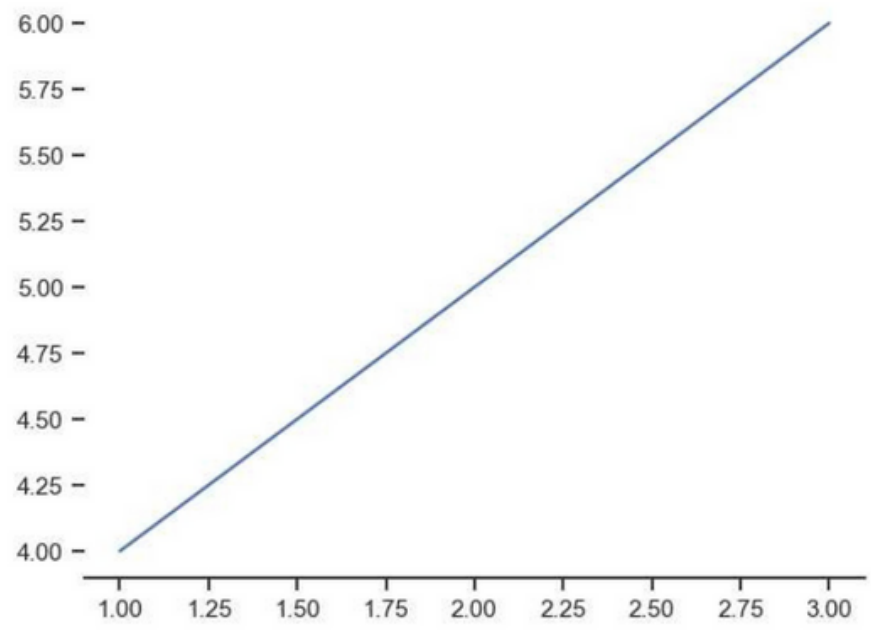
limit the range of the surviving spines

```
sns.lineplot(x=x,y=y)

sns.despine(offset=10,
            trim=True)
```



control which spines are removed

```
sns.lineplot(x=x,y=y)

sns.despine(left=True)
```



axes_style()

to view all parameters of set_style

```
sns.axes_style()
```

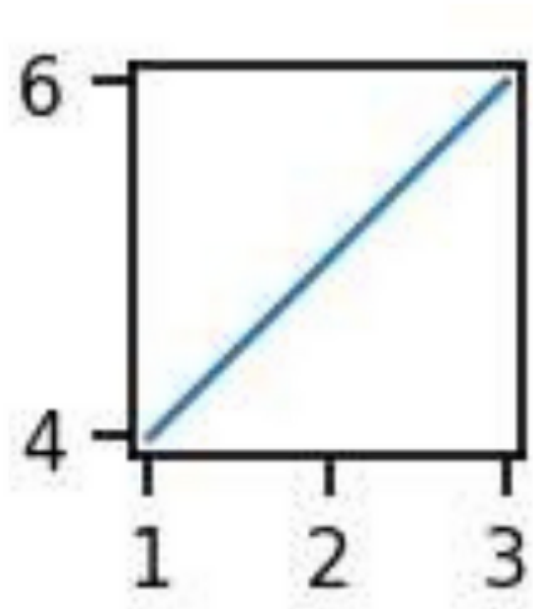
Scaling Plot Elements set_context()

to control on the plot elements and the scale of plot

four preset templates : Paper, Notebook, Talk, Poster

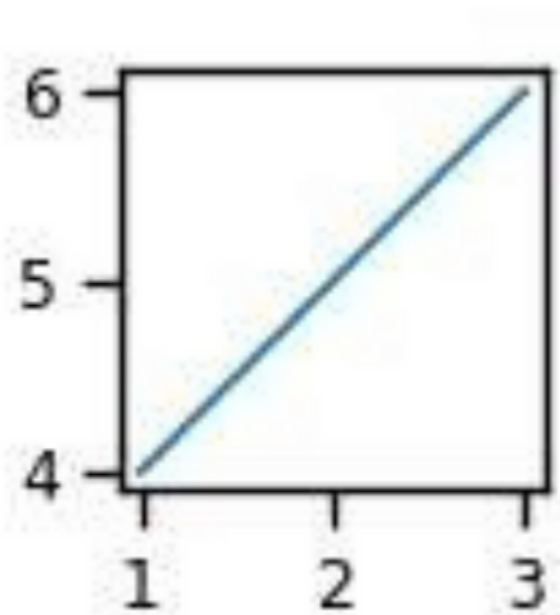
Notebook

```
sns.set_context("notebook")

sns.lineplot(x=x,y=y)
```



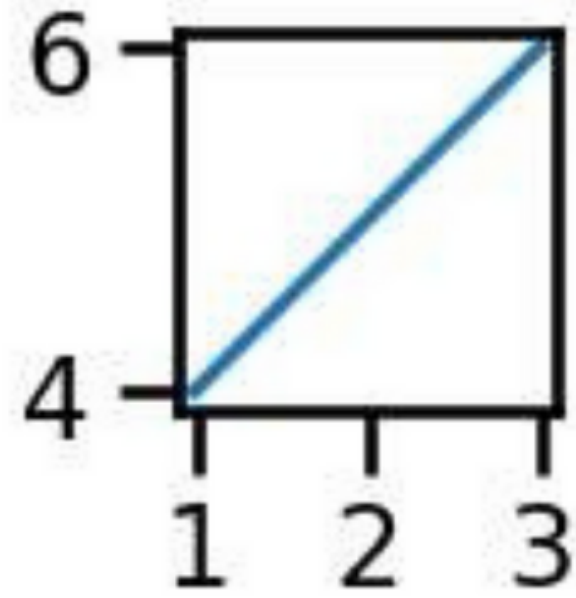
Paper

```
sns.set_context("paper")

sns.lineplot(x=x,y=y)
```



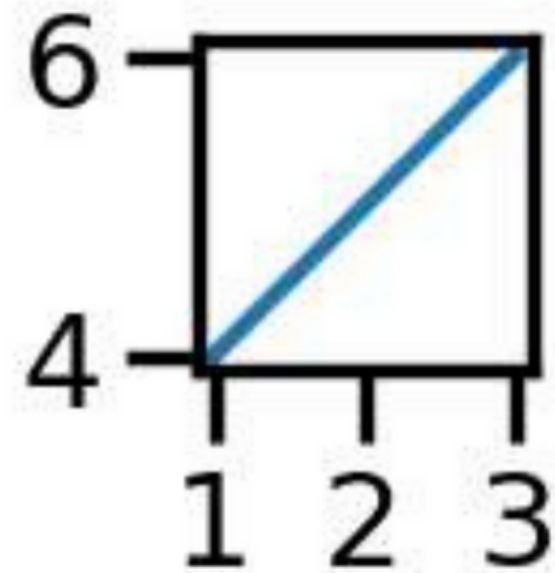
Talk

```
sns.set_context("talk")

sns.lineplot(x=x,y=y)
```



Poster

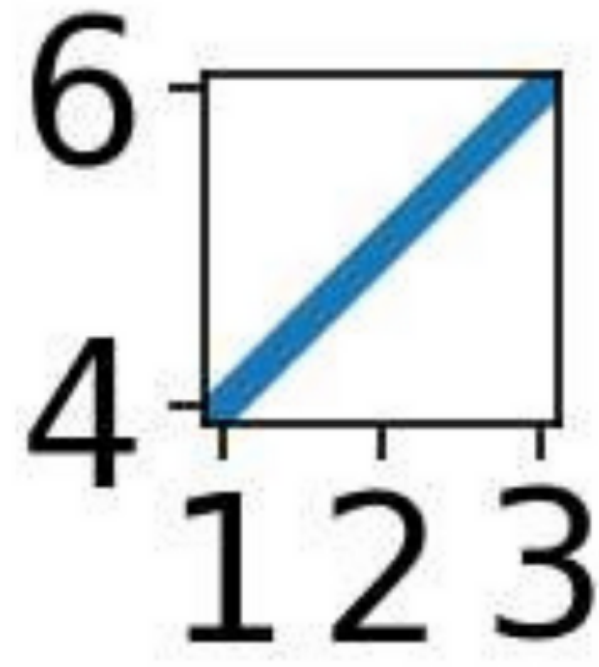
```
sns.set_context("poster")

sns.lineplot(x=x,y=y)
```



more parameters in set_context()

fontscale, rc

```
sns.set_context("notebook",
font_scale=2.9,
rc={"lines.linewidth": 6})

sns.lineplot(x=x, y=y)
```



Color palettes color_palette()

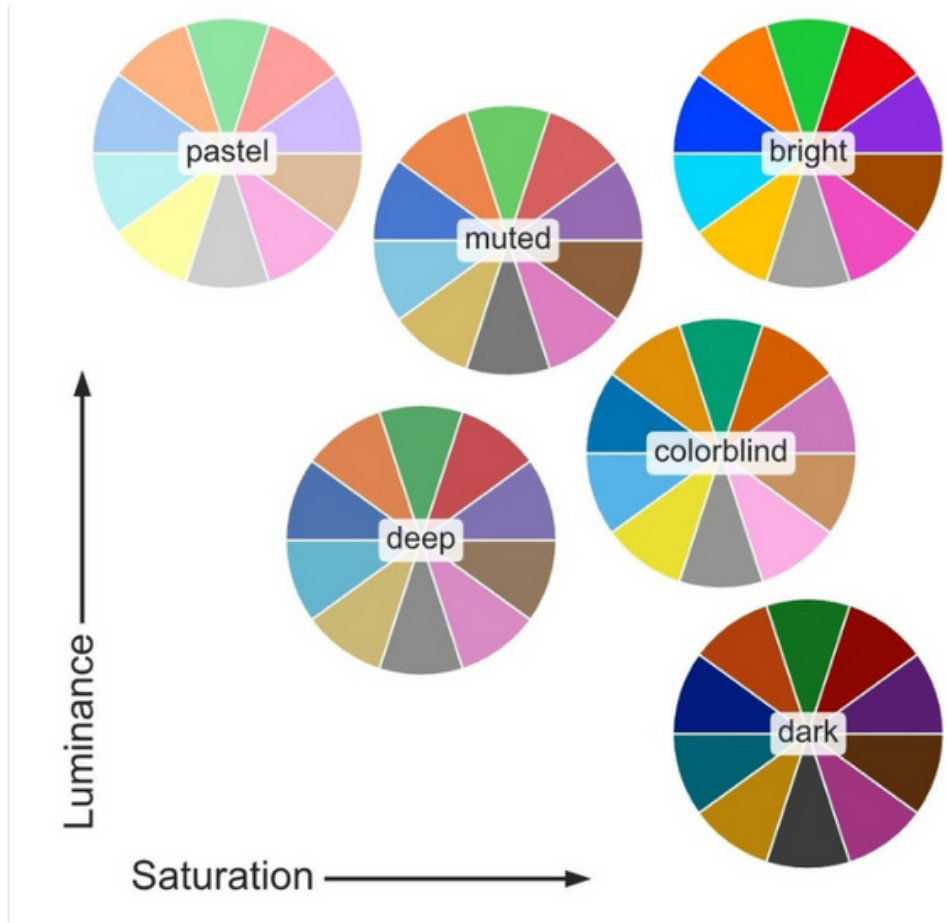
used to give colors to plots

```
sns.color_palette()
```

6 palette themes

deep, muted, pastel, bright, dark, and colorblind

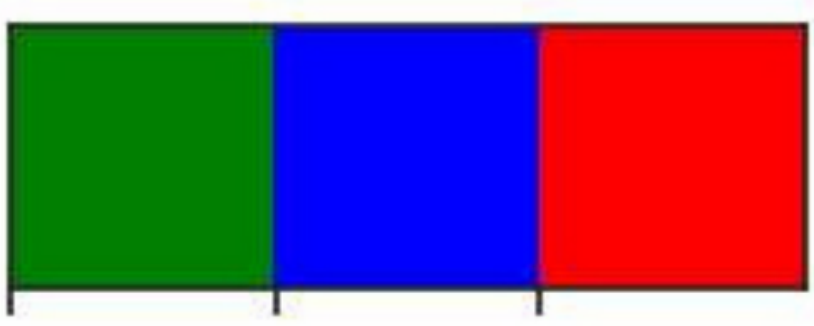
```
sns.color_palette('bright')
```



palplot() horizontal bar of colors

```
sns.palplot(sns.color_palette())
```

```
sns.palplot(["green","blue","red"]
```



Key Palette Groups

1. **Qualitative Palettes** used for categorical data

   it provide distinct colors for each category

   typically used for bar plots, scatter plots

   Palettes: Set1, Set2, Set3, Pastel1, and Dark2

```
sns.color_palette("Set2")
```

2. Sequential Palettes

used for data that has a natural order varies from low to high

useful for heatmap and line plot

Palettes: Greens, Blues, Reds, viridis, cubehelix

```
sns.color_palette("Greens")
```

3. Diverging Palettes

used for data that has center point from negative to positive values

it use two distinct colors. for -1 to 1 data, -1 to 0 takes one color and 0 to +1 takes another color.

Palettes: coolwarm, BrBG, RdBu_r and PuOr

```
sns.color_palette("coolwarm",7)
```

set_palette() Define default palette

```
sns.set_palette("Set1")
```

Importing Datasets load_dataset()

Seaborn comes with inbuilt datasets like 'tips'

This dataset loads as Pandas DataFrame by default

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

df.head()
```

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

get_dataset_names()

to view more all the available data sets in the Seaborn

```
sns.get_dataset_names()
```

Types of Plot in Seaborn

Each category is designed for a specific type of data and serves a different purpose

choose most appropriate visualization technique for data analysis depending on the nature of data and the insights you want to extract

1. Relational Plots relpolt()

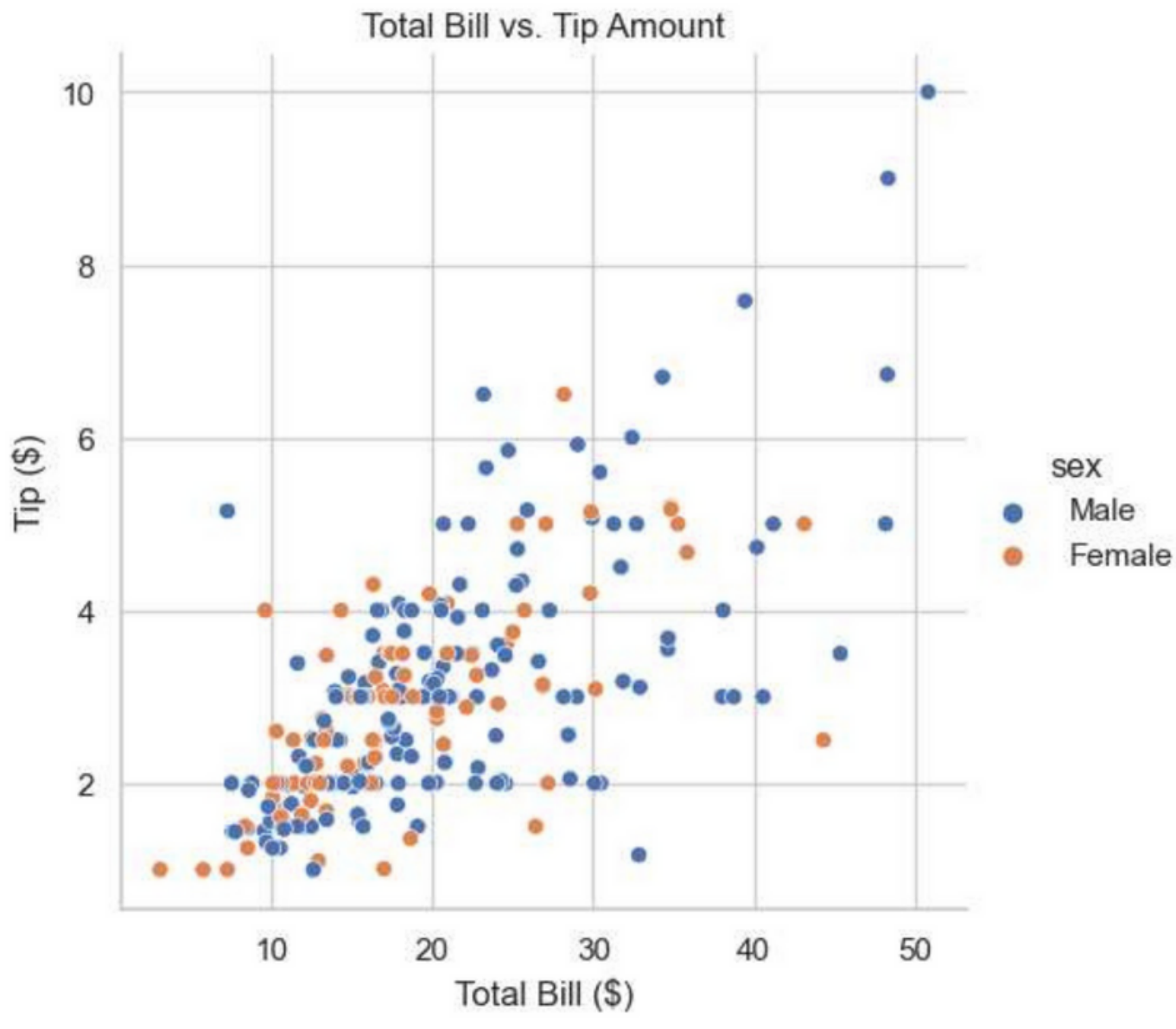
used to visualize the relationships between two or more variables, to explore how variables relate to each other.

Plots: Scatter and Line

Scatter plot using relplot()

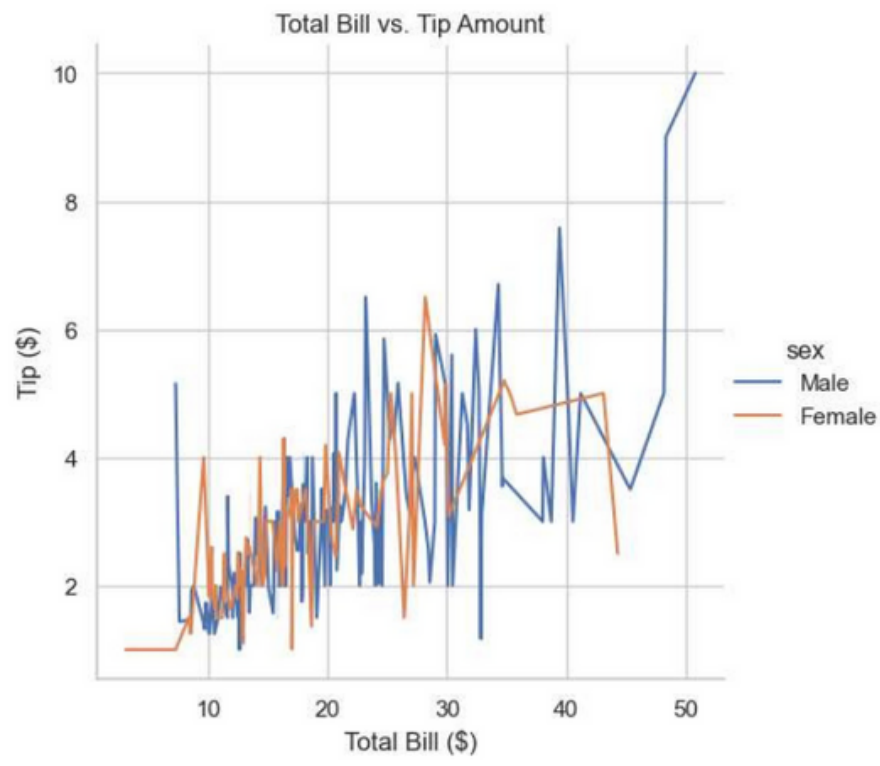
```
sns.set(style="whitegrid")

# in-built dataset 'tips'
tips = sns.load_dataset("tips")
```

```
g = sns.relplot(x="total_bill",
    y="tip", hue="sex",
    data=tips, kind="scatter")

# Customize the plot
g.set(title="Total Bill vs. Tip",
      xlabel="Total Bill ($)",
      ylabel="Tip ($)")

plt.show()
```



Lineplot in replot()

```
sns.relplot(x="total_bill",
            y="tip", hue="sex",
            data=tips, kind="line"
```



2. Distributional Plots displot()

to visualize variable distributions (normal, skewed, uniform, bimodal), data spread (range), central tendencies(mean median mode), and patterns in datasets, aiding in data exploration before advanced analysis.

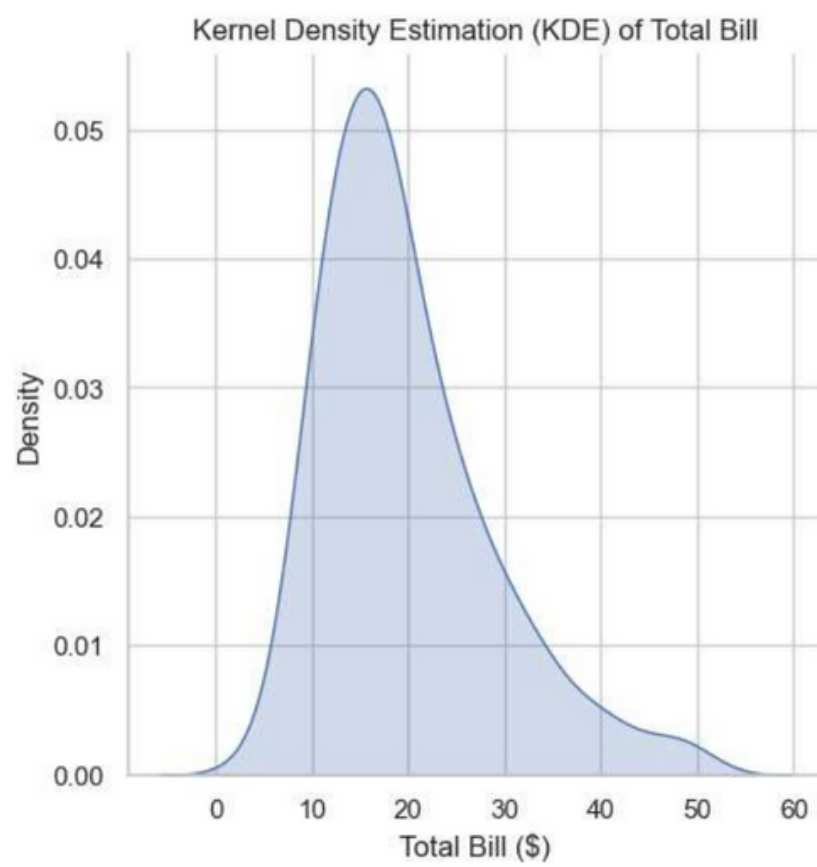
Plots : Histogram, Kernel Density Estimation (KDE), and Empirical Cumulative Distribution Function (ECDF)

Kernel Density Estimation (KDE) Plot

```
sns.displot(tips["total_bill"],
            kind="kde", fill=True)
```

or

```
# kdeplot()
sns.kdeplot(tips["total_bill"],
                fill=True)
```

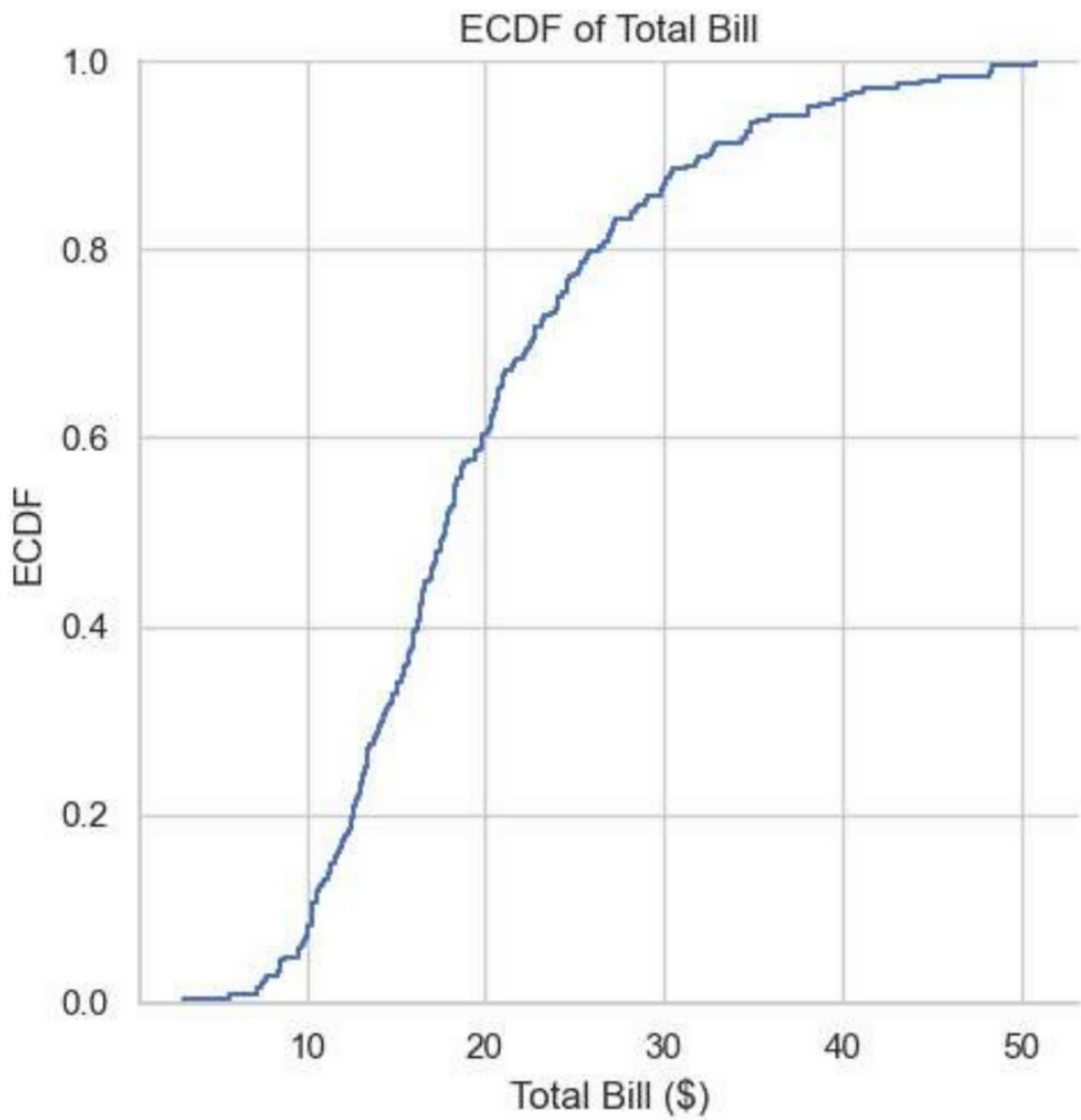


Empirical Cumulative Distribution Function (ECDF) Plot

```
sns.displot(tips["total_bill"],
            kind="ecdf")
```

or

```
# ecdfplot()
sns.ecdfplot(tips["total_bill"])
```

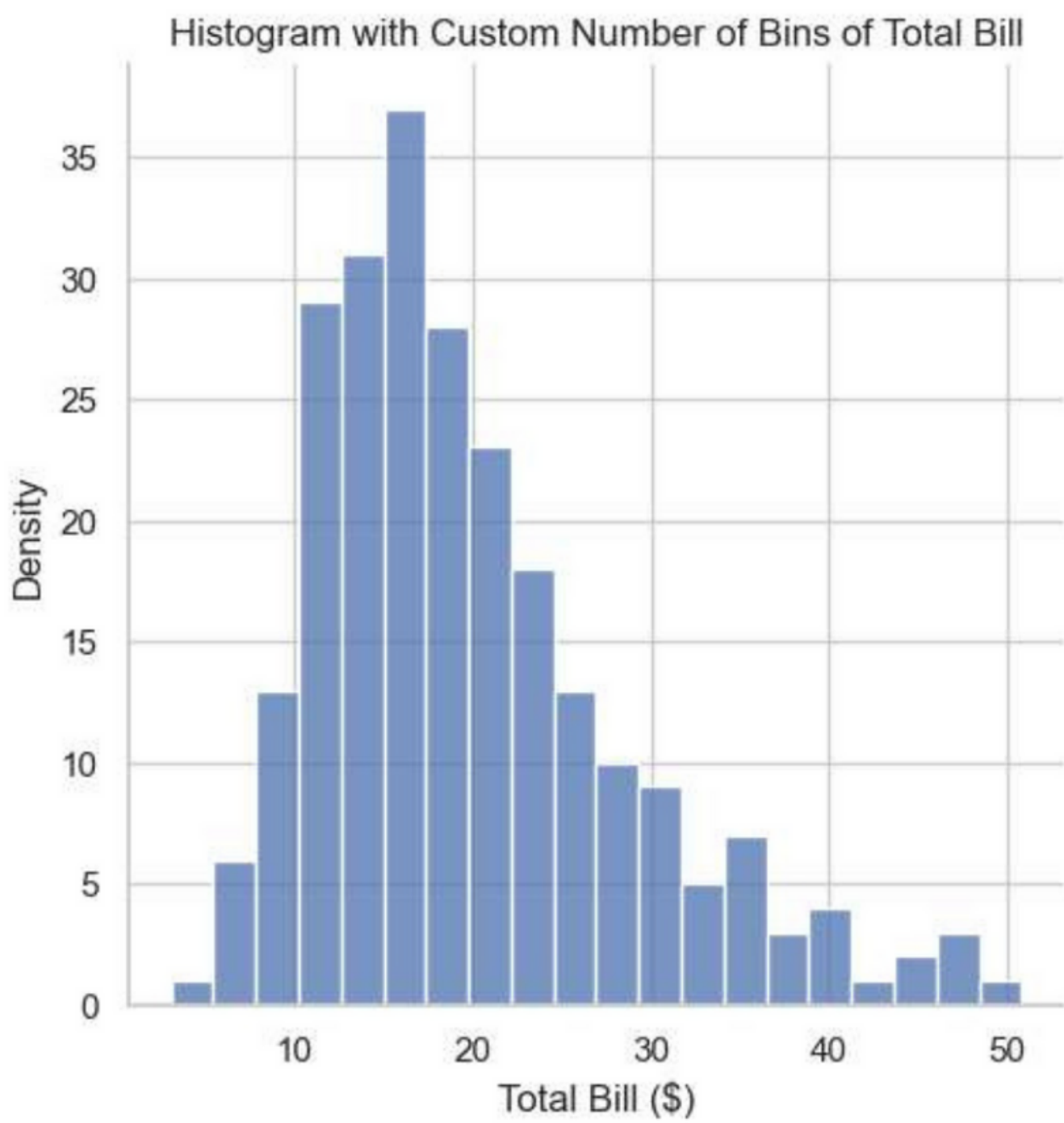


Histogram Plot

```
sns.displot(tips["total_bill"],
            kind="hist",
            bins=20)
```

or

```
# histplot()
sns.histplot(tips["total_bill"],
             bins=20)
```



3. Categorical Plots catplot()

explore relationships between categorical and numerical data.

Categorical Data (e.g. weeks, fruit types), Numerical Data (e.g. age, price)

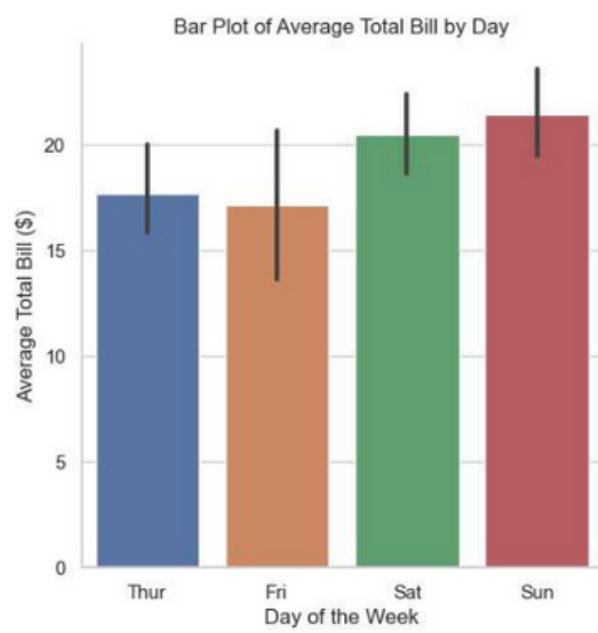
Plots : Bar, Box, Violin, Point, Strip, Swarm

Bar Plot()

```
sns.catplot(x="day",
            y="total_bill",
            data=tips,
            kind="bar")
```

or

```
# barplot()
sns.barplot(x="day",
            y="total_bill",
            data=tips)
```

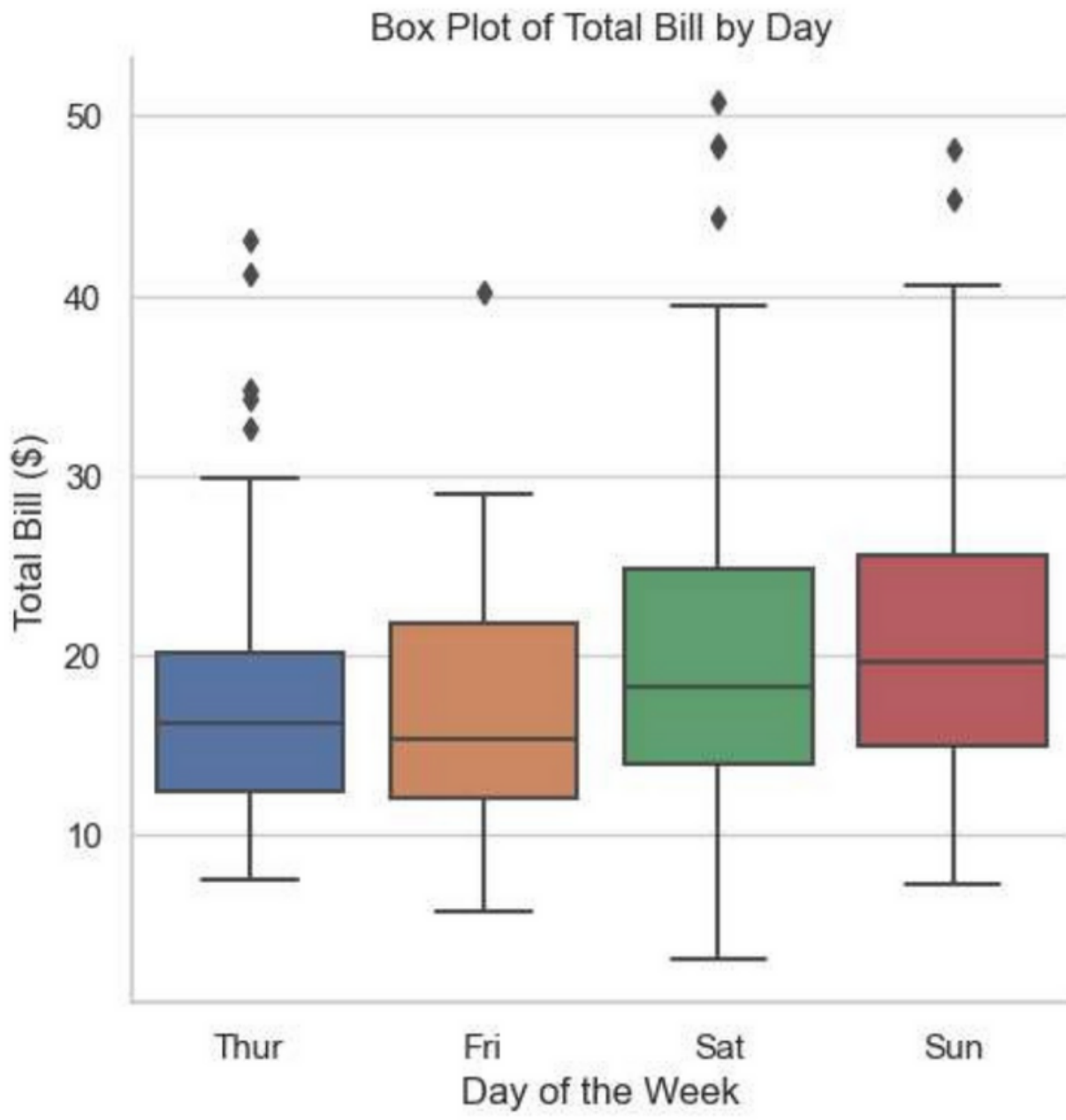


Box Plot

```
sns.catplot(x="day",
                y="total_bill",
                data=tips)
```

or

```
# boxplot()
sns.boxplot(x="day",
                y="total_bill",
                data=tips)
```

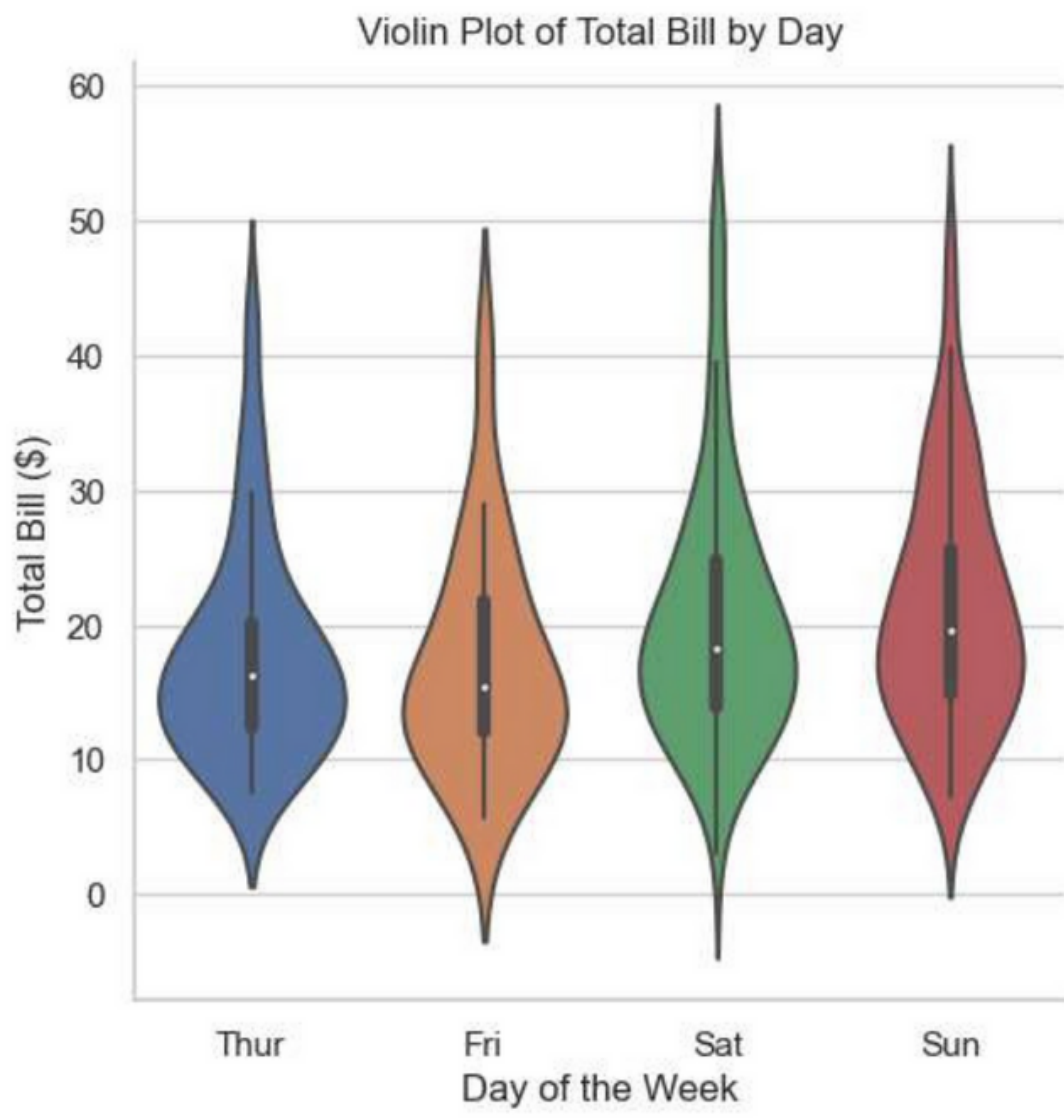


Violin Plot

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

or

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

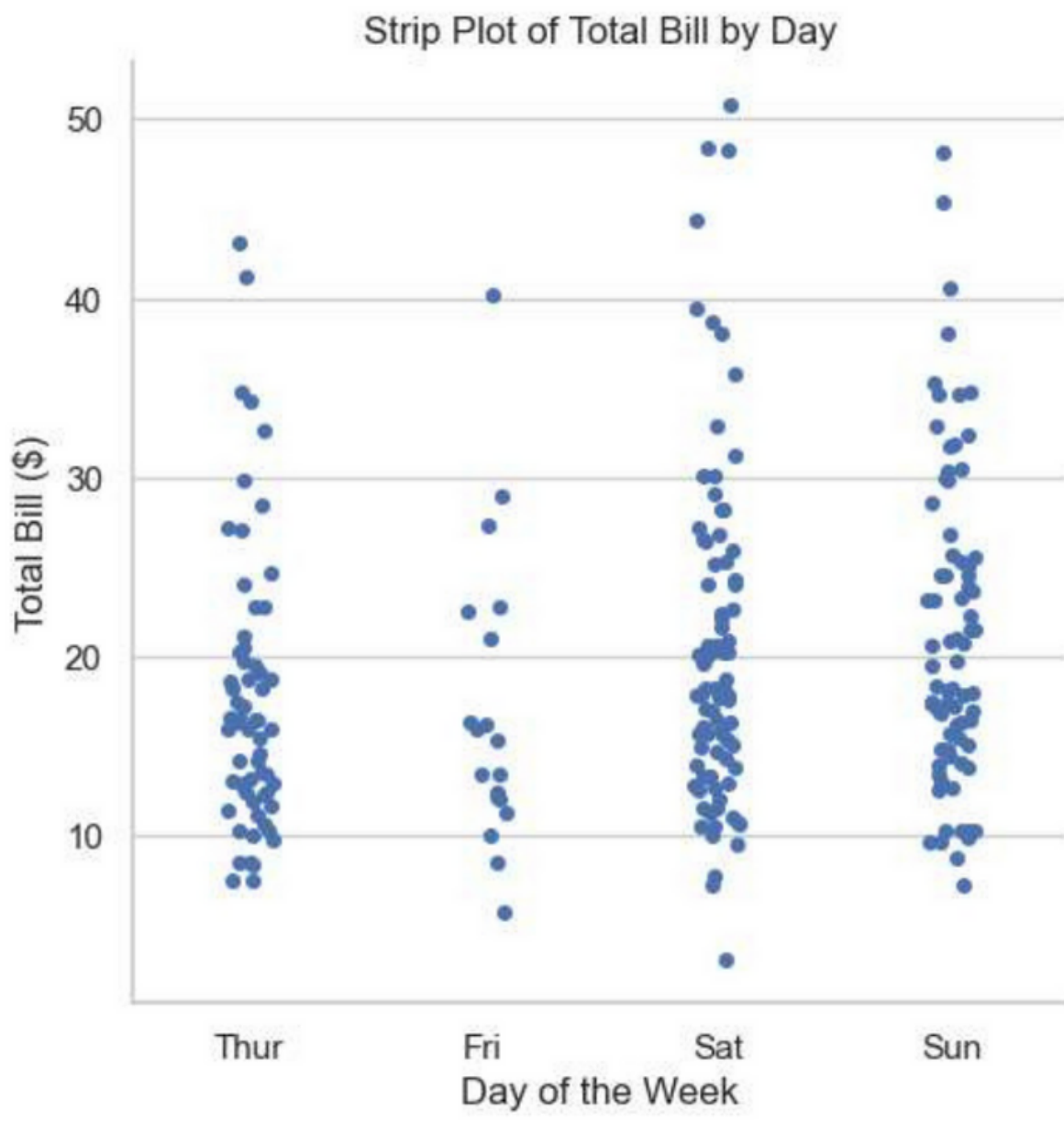


Strip Plot

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

or

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

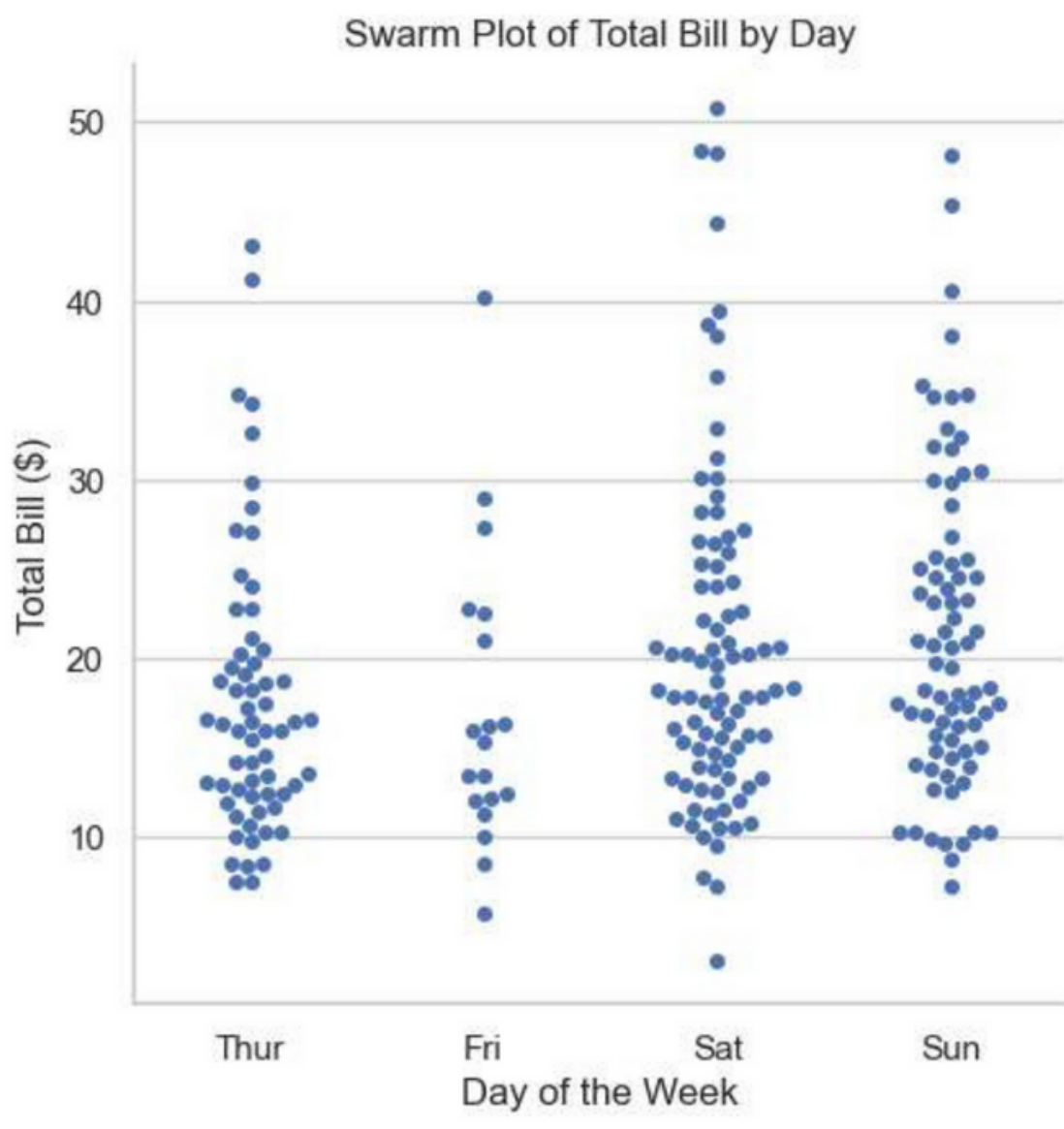


Swarm Plot

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

or

```
# swarmplot()
sns.swarmplot(x="day",
              y="total_bill",
              data=tips)
```

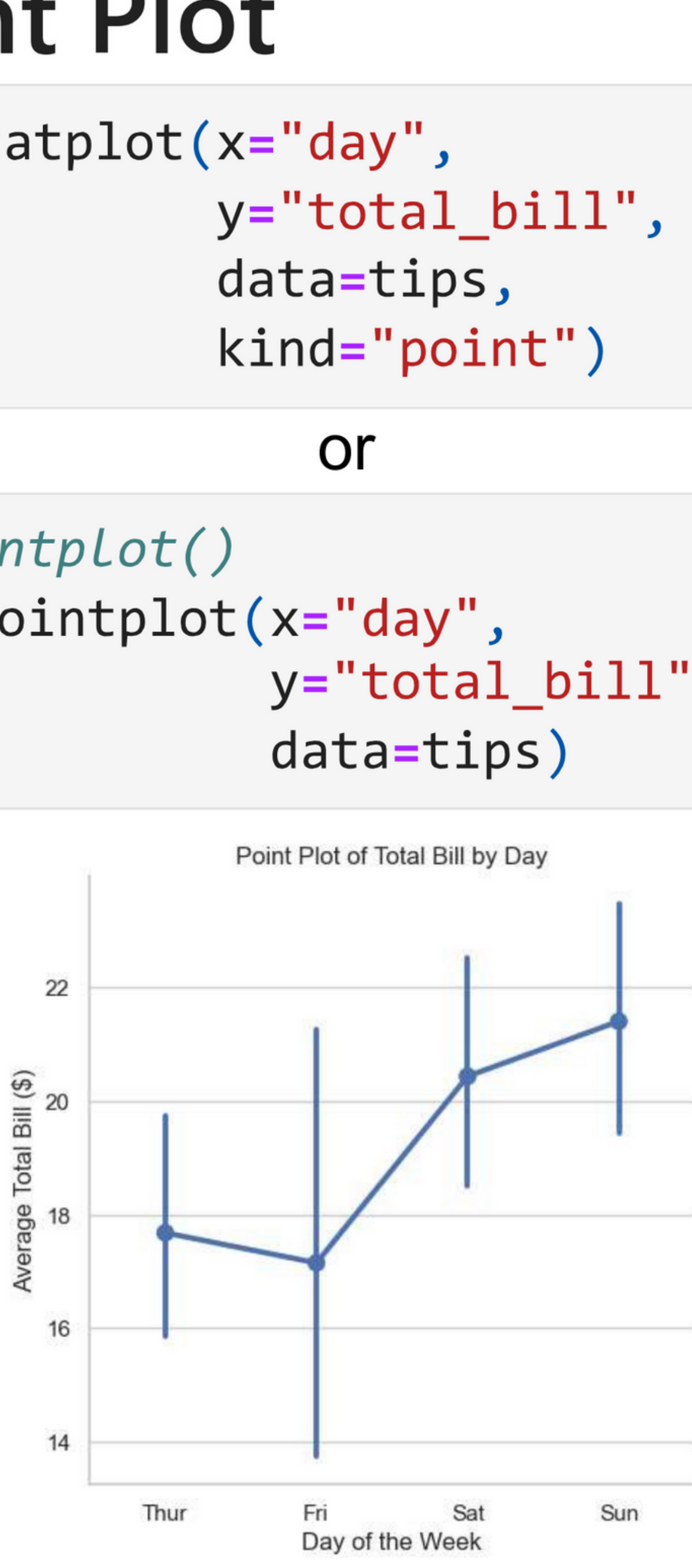


Point Plot

```
sns.catplot(x="day",
                y="total_bill",
                data=tips,
                kind="point")
```

or

```
# pointplot()
sns.pointplot(x="day",
                  y="total_bill",
                  data=tips)
```



4. Matrix Plot

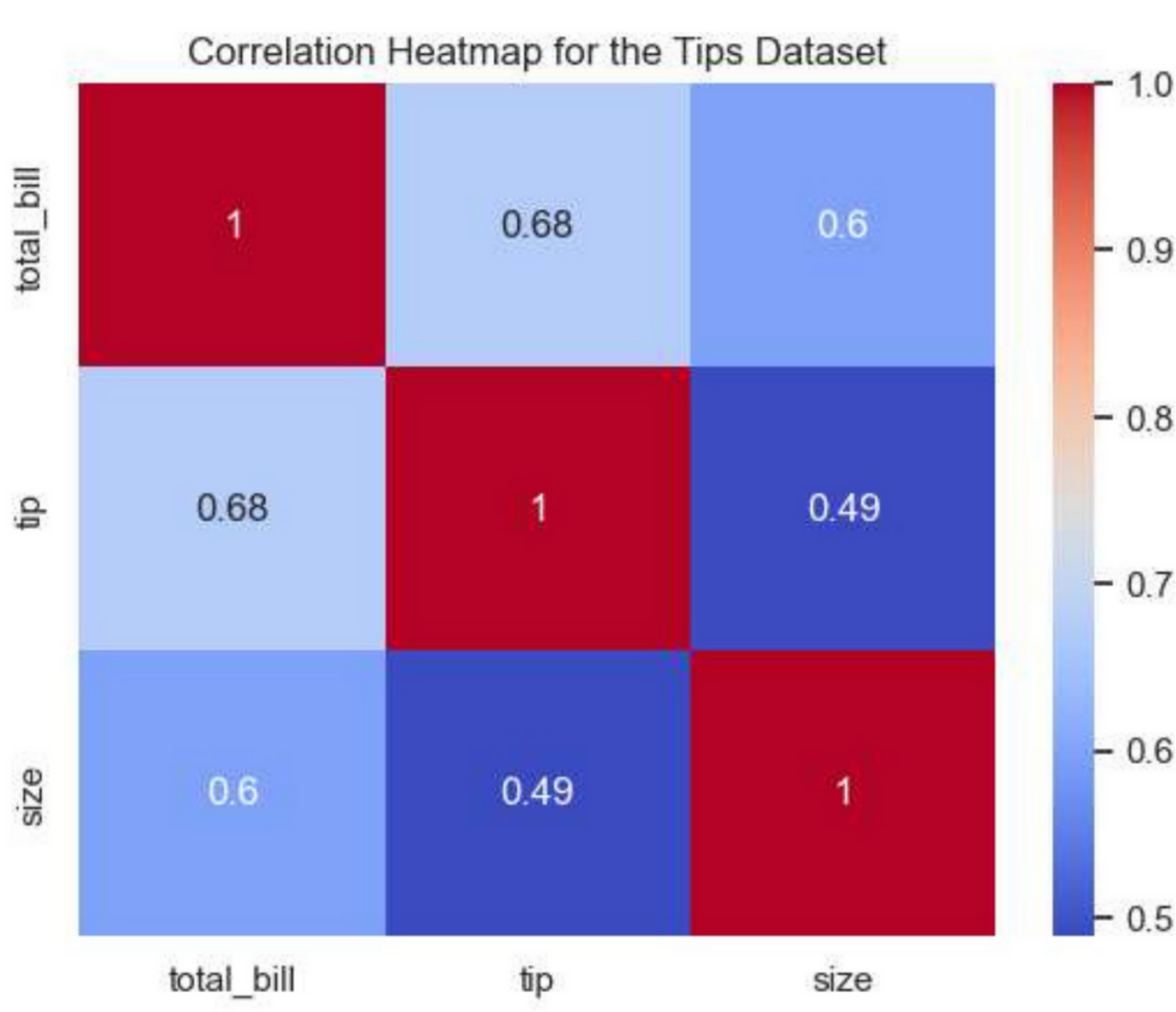
visualize the relationships and patterns between multiple variables

explore correlations between pairs of variables in a dataset

Plots: Heatmap, Clustermap, Pair

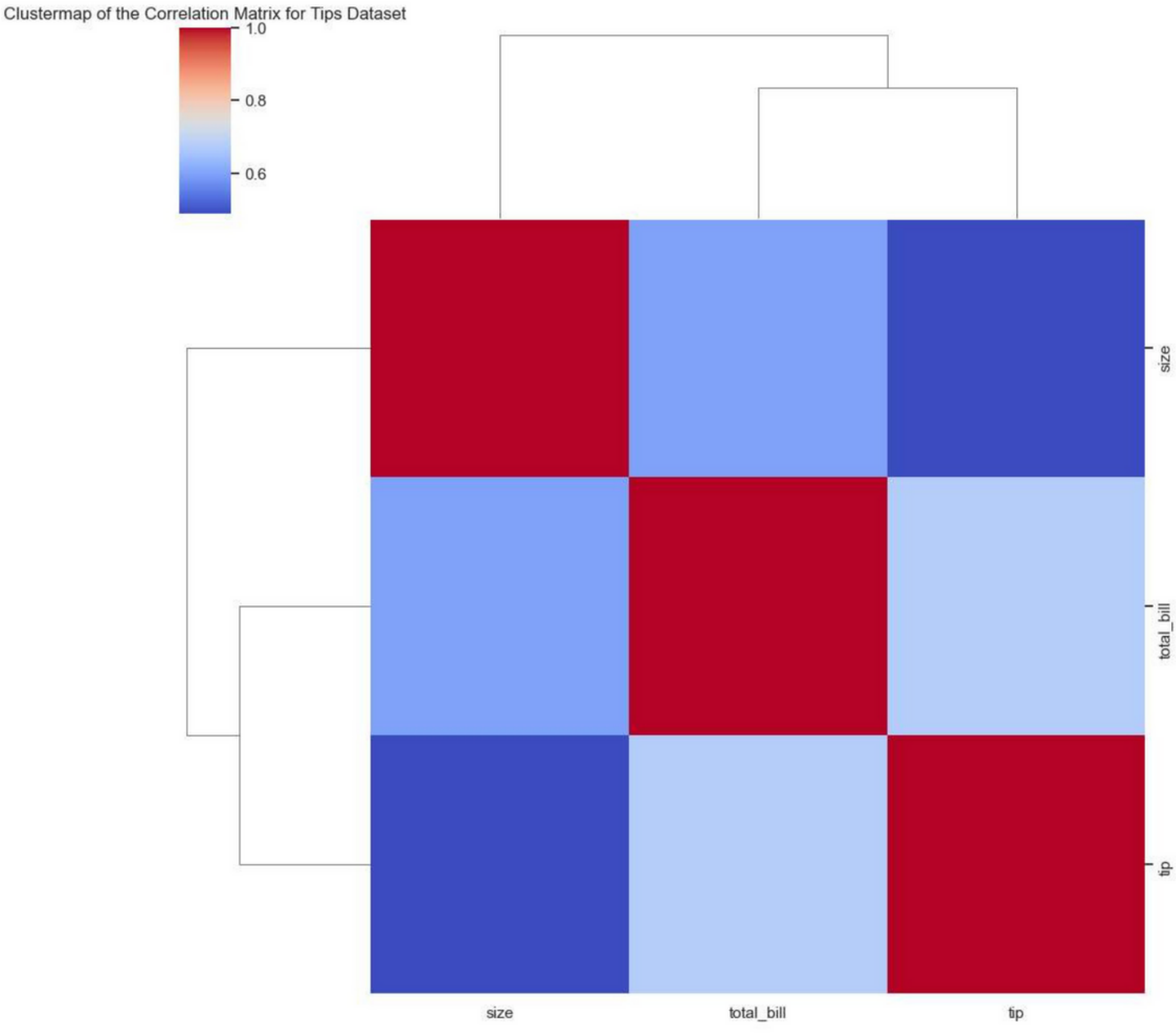
Heatmap

```
sns.heatmap(tips.corr(),annot=True
                    cmap="coolwarm")
```



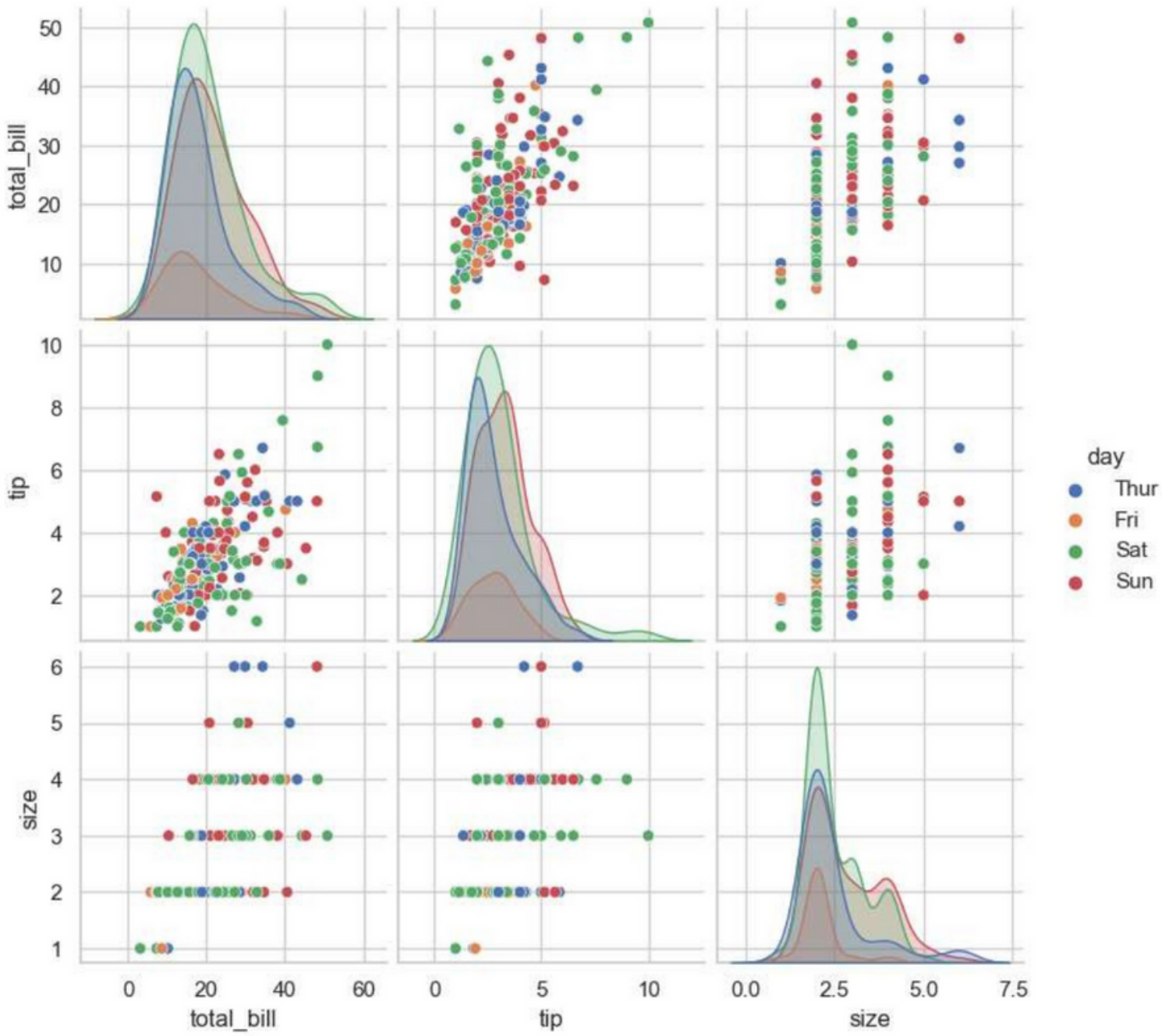
Clustermap

```
sns.clustermap(tips.corr(),
               cmap="coolwarm")
```



Pair plot

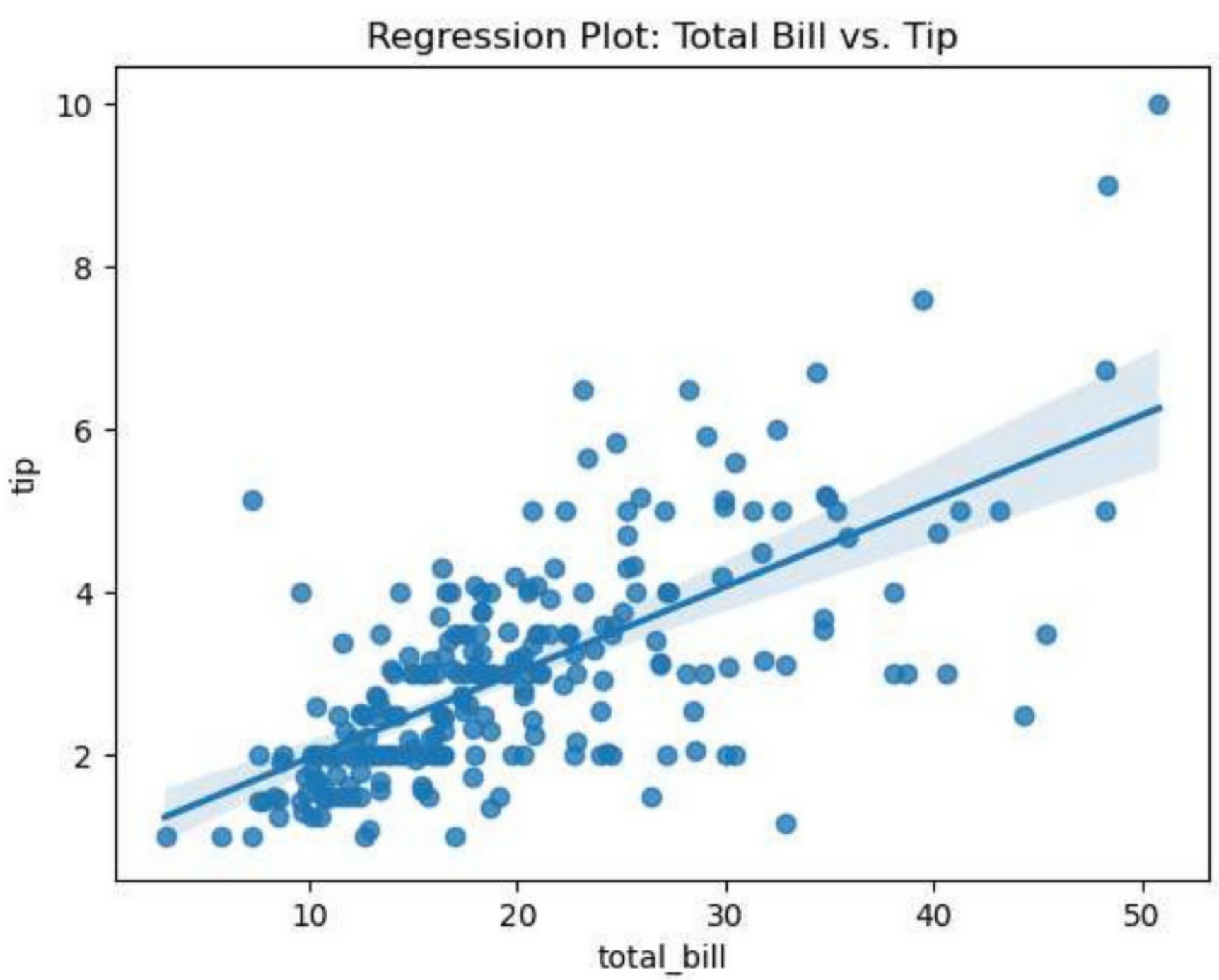
```
sns.pairplot(tips, hue="day")
```



5. Regression Plot

regplot()

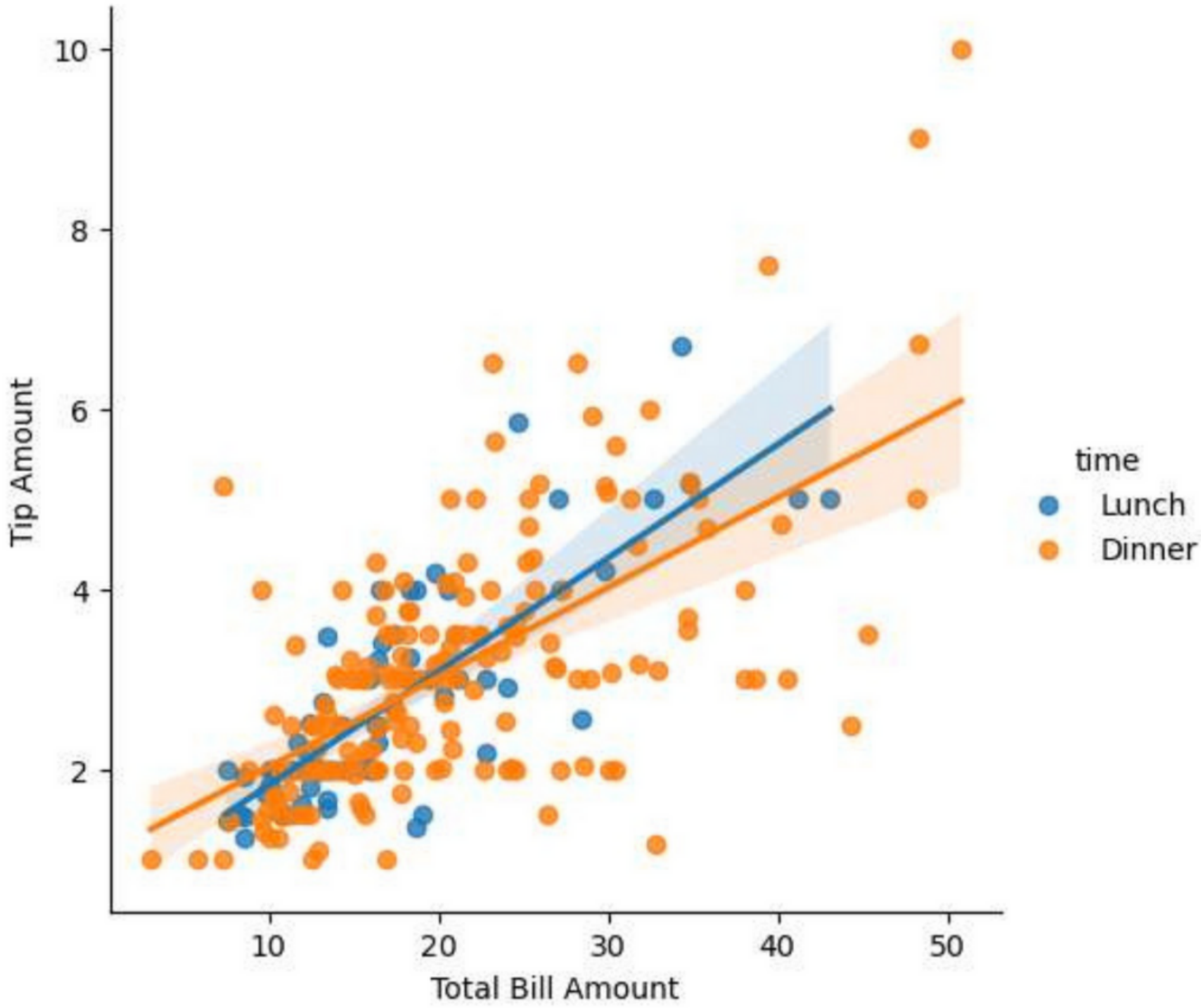
```
sns.regplot(x="total_bill",
            y="tip",data=tips)
```



lmplot()

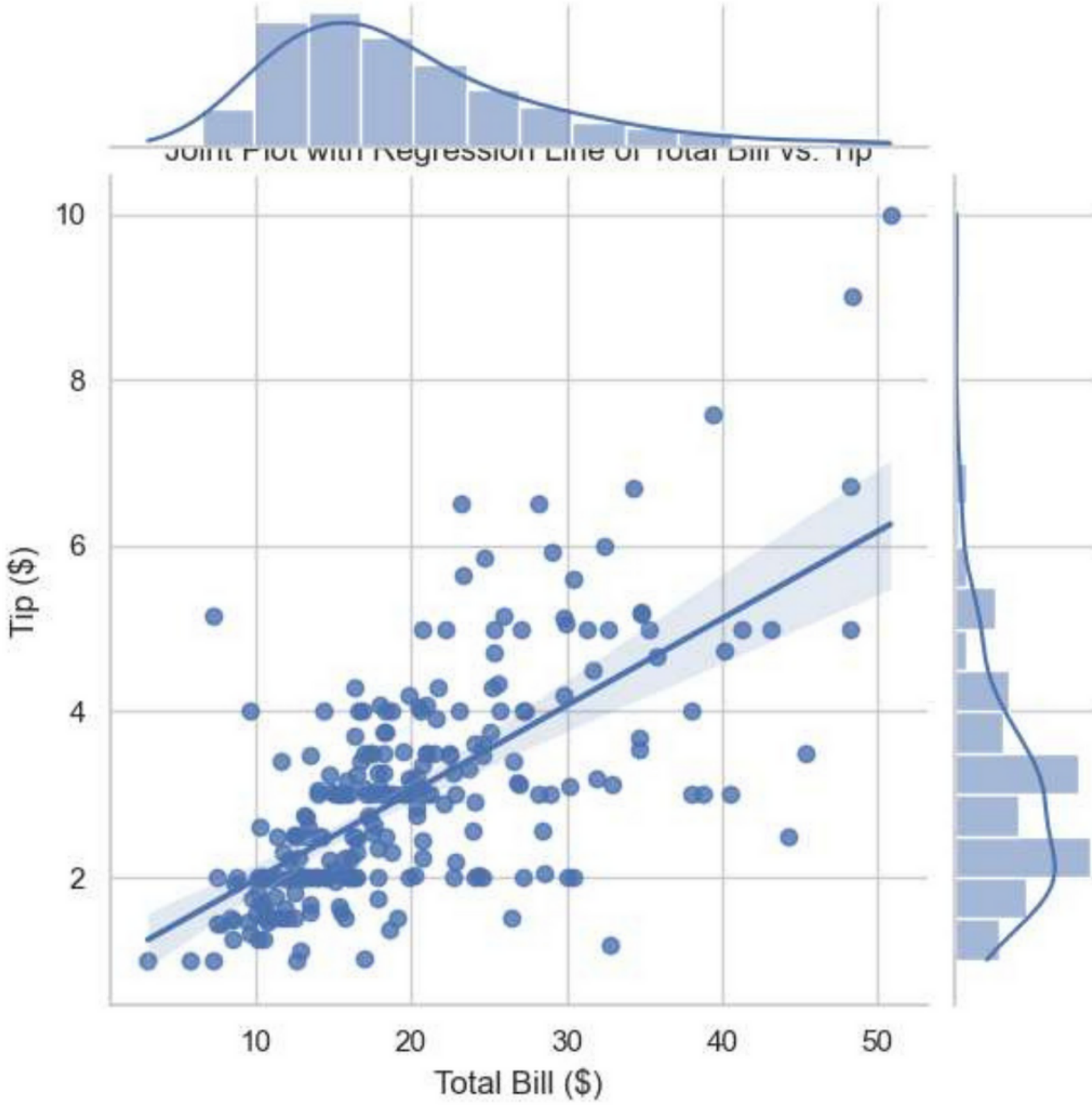
Create a linear model plot (lmplot) with facets by time category

```
sns.lmplot(x="total_bill",
           y="tip",hue="time",
           data=tips)
```



joinplot()

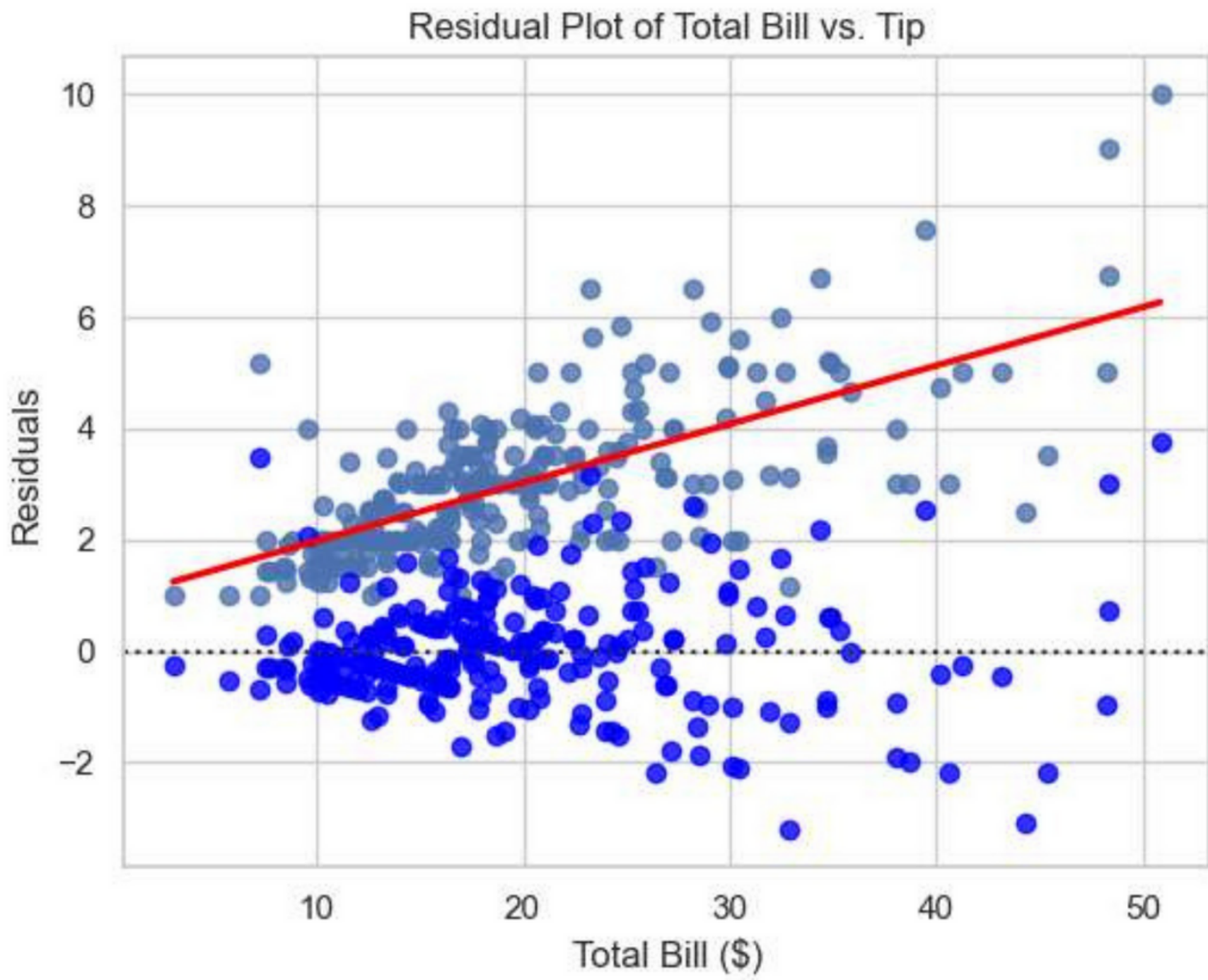
```
sns.jointplot(x="total_bill",
              y="tip", data=tips,
              kind="reg")
```



residplot()

```
# Fit a linear regression model
sns.regplot(x='total_bill',
    y='tip', data=tips,ci=None,
    line_kws={'color': 'red'})
```

```
# Create a residual plot
sns.residplot(x='total_bill',y='ti
              data=tips, color='blue
```

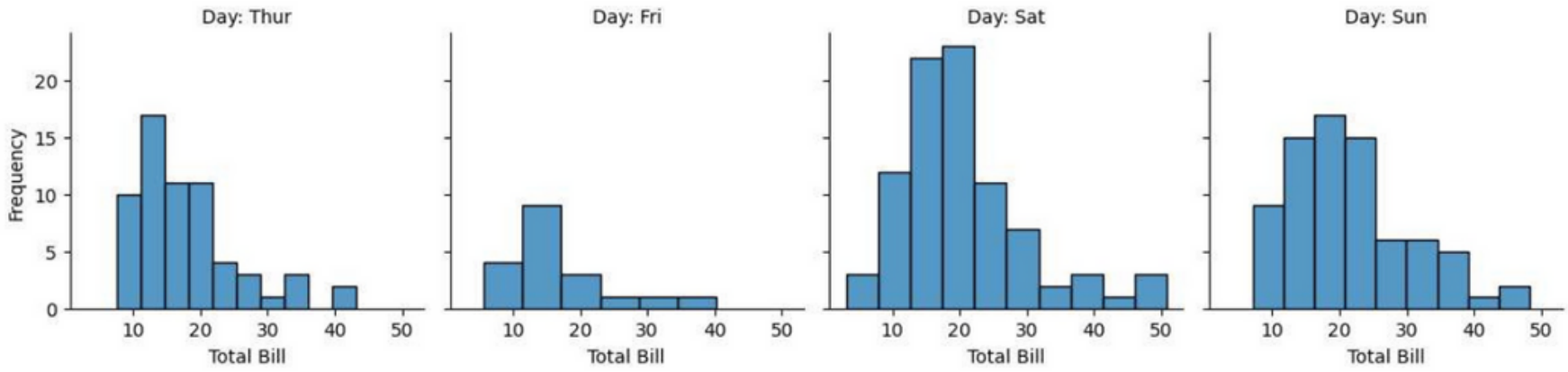


Multiple Plots in Seaborn

FacetGrid

Create a FacetGrid for multiple plots based on the 'day' variable

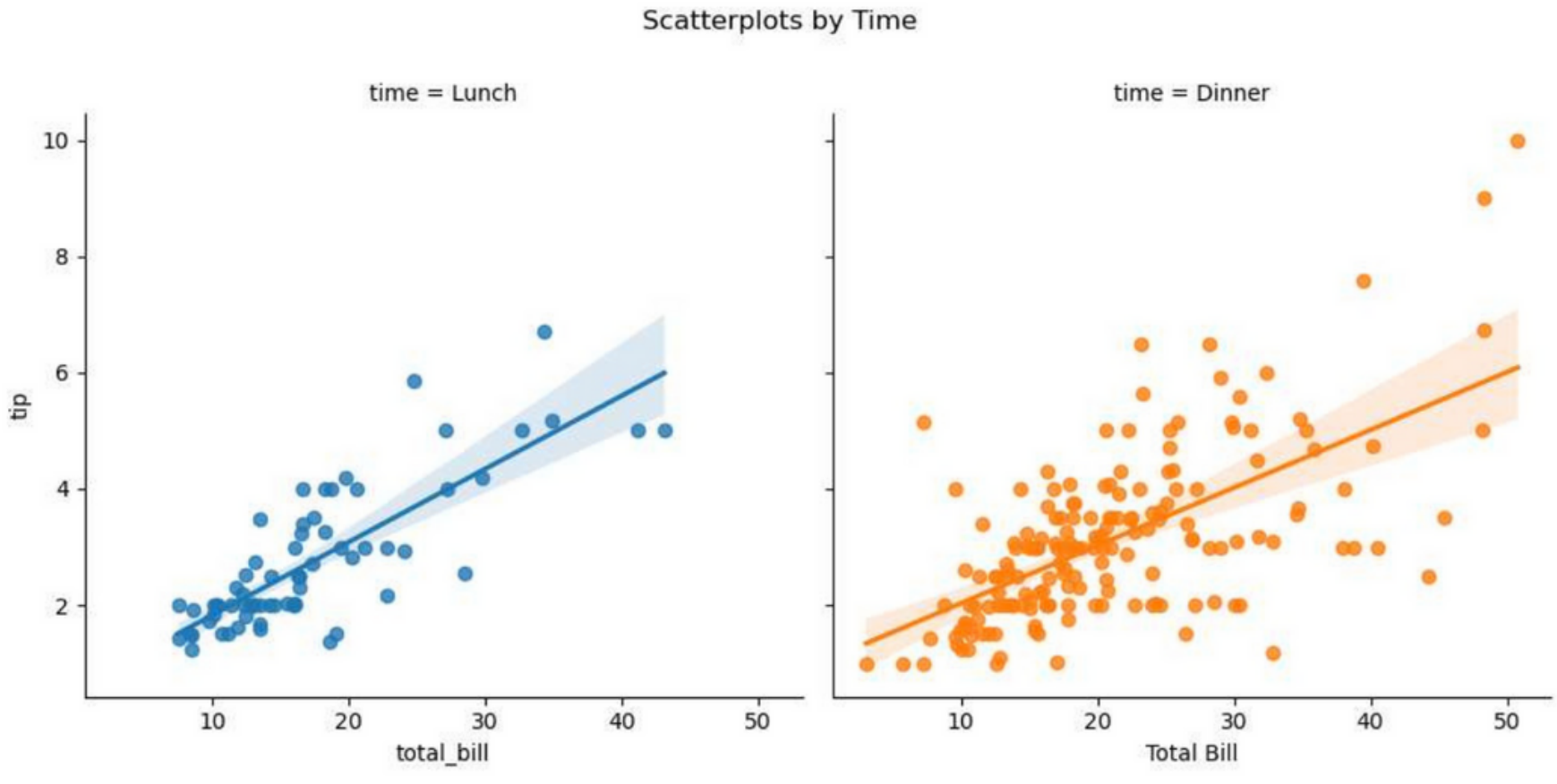
```
g = sns.FacetGrid(tips, col="day")
g.map(sns.histplot, "total_bill")
```



Implot

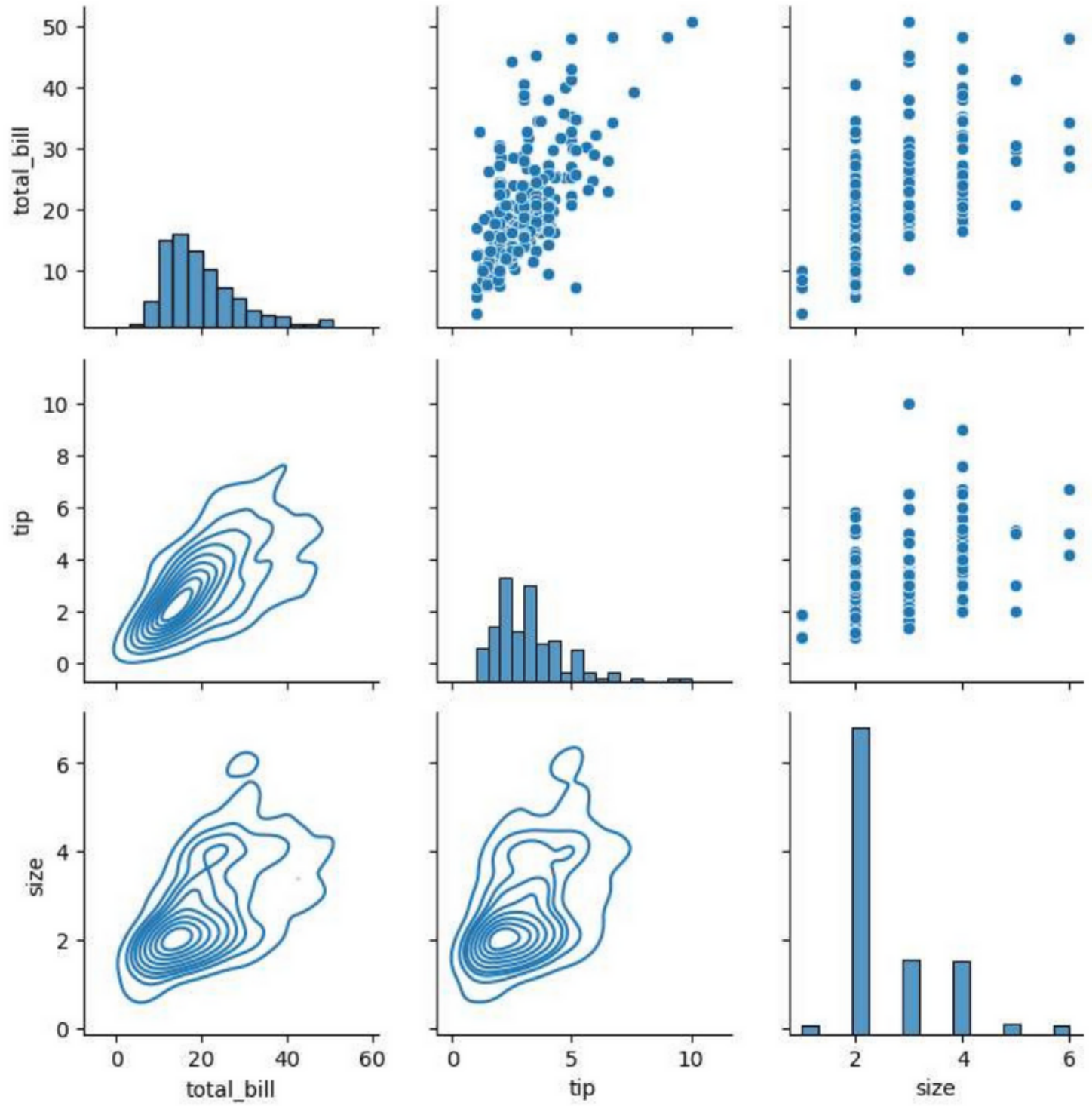
Create a linear model plot with facets based on 'time' variable

```
sns.lmplot(x="total_bill", y="tip"
           hue="time", data=tips,
           col="time")
```



PairGrid

```
# pairwise scatterplots
g = sns.PairGrid(tips)

# Customize the appearance
g.map_upper(sns.scatterplot)
g.map_diag(sns.histplot)
g.map_lower(sns.kdeplot)
```



thank you