🧭 Seaborn Cheatsheet – When to Use Which Plot

💎 Importance of Seaborn in Data Science

Seaborn is one of the most powerful Python libraries for data visualization.

It helps data scientists explore, understand, and communicate insights effectively through beautiful and informative charts.

🔥 Why Seaborn Matters:

🎨 **Beautiful & Informative:** Creates professional, publication-quality visualizations with minimal code.

📊 **Built on Matplotlib:** Easier syntax but more powerful styling and color control.

🧠 **Statistical Awareness:** Automatically computes and visualizes statistical relationships (like confidence intervals, regression lines, distributions).

⚡ **Fast EDA (Exploratory Data Analysis):** Quickly reveals patterns, outliers, and trends in data.

🔗 **Integrated with Pandas:** Works directly with DataFrames — no need to convert data manually.

🔍 **Supports Complex Plots:** From simple histograms to multi-dimensional plots like pairplot and heatmap.

📈 **Enhances Storytelling:** Turns raw data into visuals that make insights easy to explain to non-technical audiences.

🎯 Purpose:

Choose the right plot based on our data type and what we want to explore — distribution, relationship, or comparison.

📈 **Numerical vs Numerical:** scatterplot, lineplot, jointplot

🧩 **Numerical vs Categorical:** boxplot, violinplot, barplot

🧮 **Single Variable:** histplot, kdeplot

🔍 **Multiple Variables:** pairplot, heatmap

🧠 1. Distribution Plots (for understanding single variable)

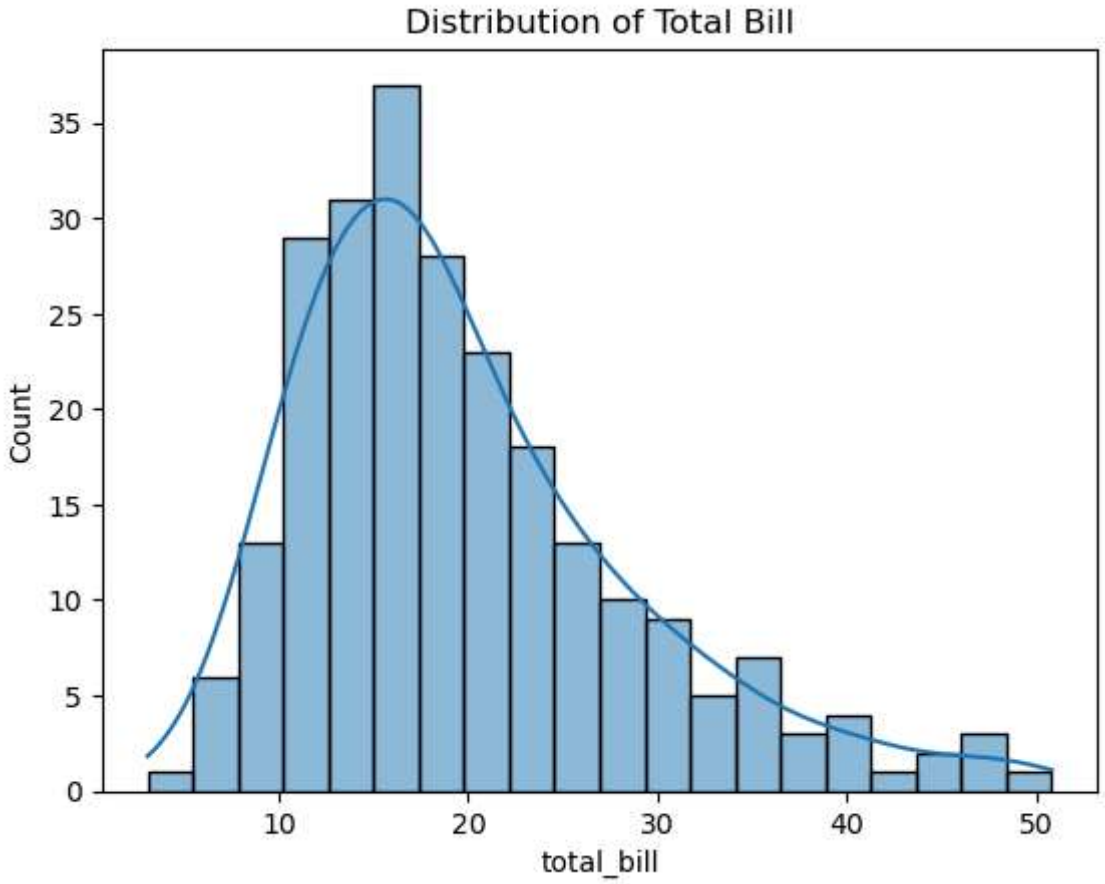
a. Histogram – sns.histplot()

When to use: To see the distribution (shape) of a numerical variable.

Why: Helps identify skewness, outliers, and spread.

In [1]:
```
import seaborn as sns
import matplotlib.pyplot as plt
tips = sns.load_dataset("tips")

sns.histplot(data=tips, x="total_bill", bins=20, kde=True)
plt.title("Distribution of Total Bill")
plt.show()
```

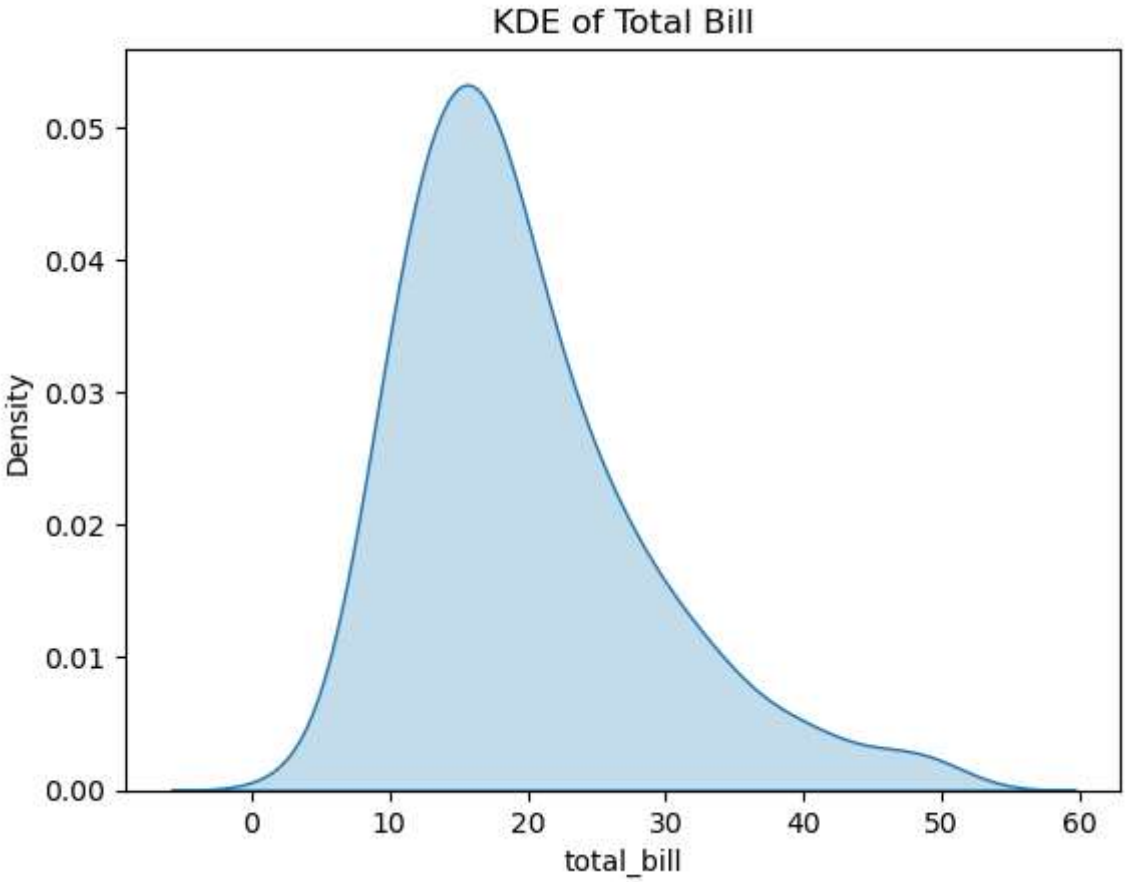


b. KDE Plot – sns.kdeplot()

When to use: To visualize the probability density (smooth curve).

Why: Good for seeing distribution trends clearly (especially for continuous data).

In [3]:
```
sns.kdeplot(data=tips, x="total_bill", fill=True)
plt.title("KDE of Total Bill")
plt.show()
```

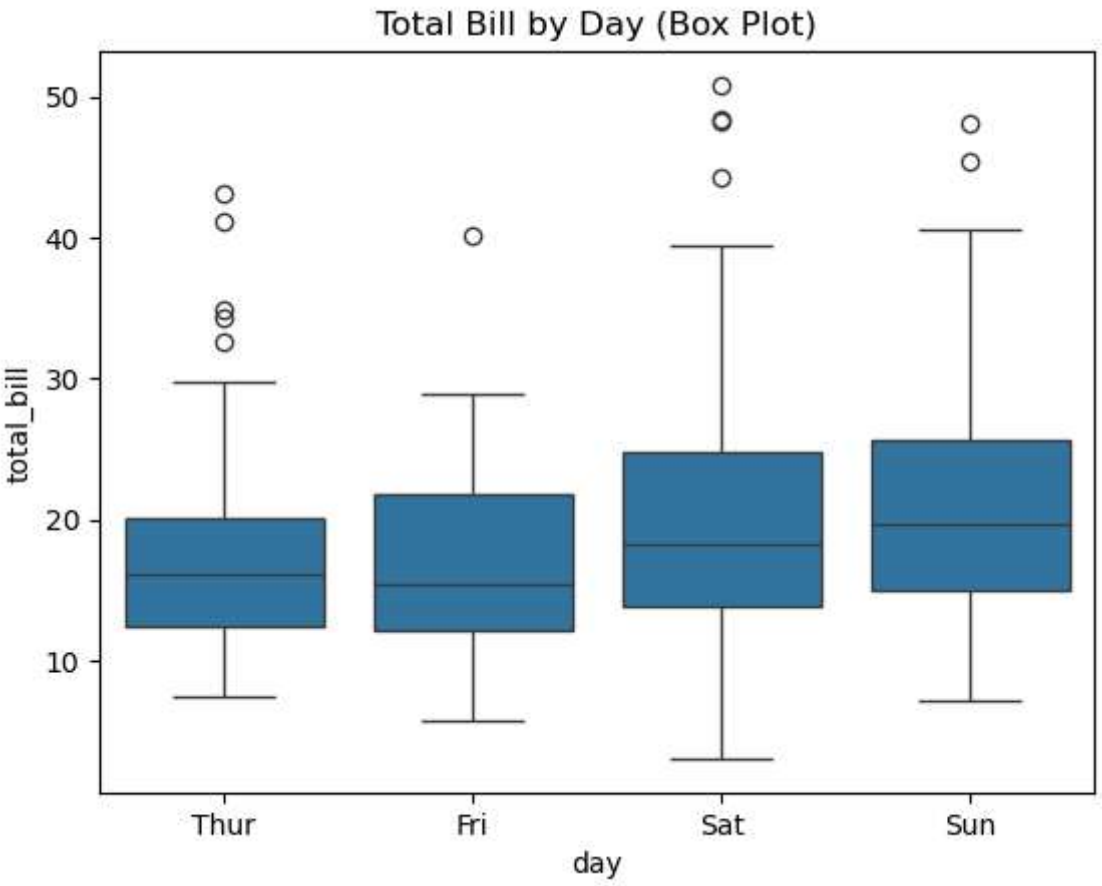


c. Box Plot – sns.boxplot()

When to use: To show summary statistics (median, quartiles, outliers).

Why: Great for comparing distributions across categories.

In [4]:
```
sns.boxplot(data=tips, x="day", y="total_bill")
plt.title("Total Bill by Day (Box Plot)")
plt.show()
```

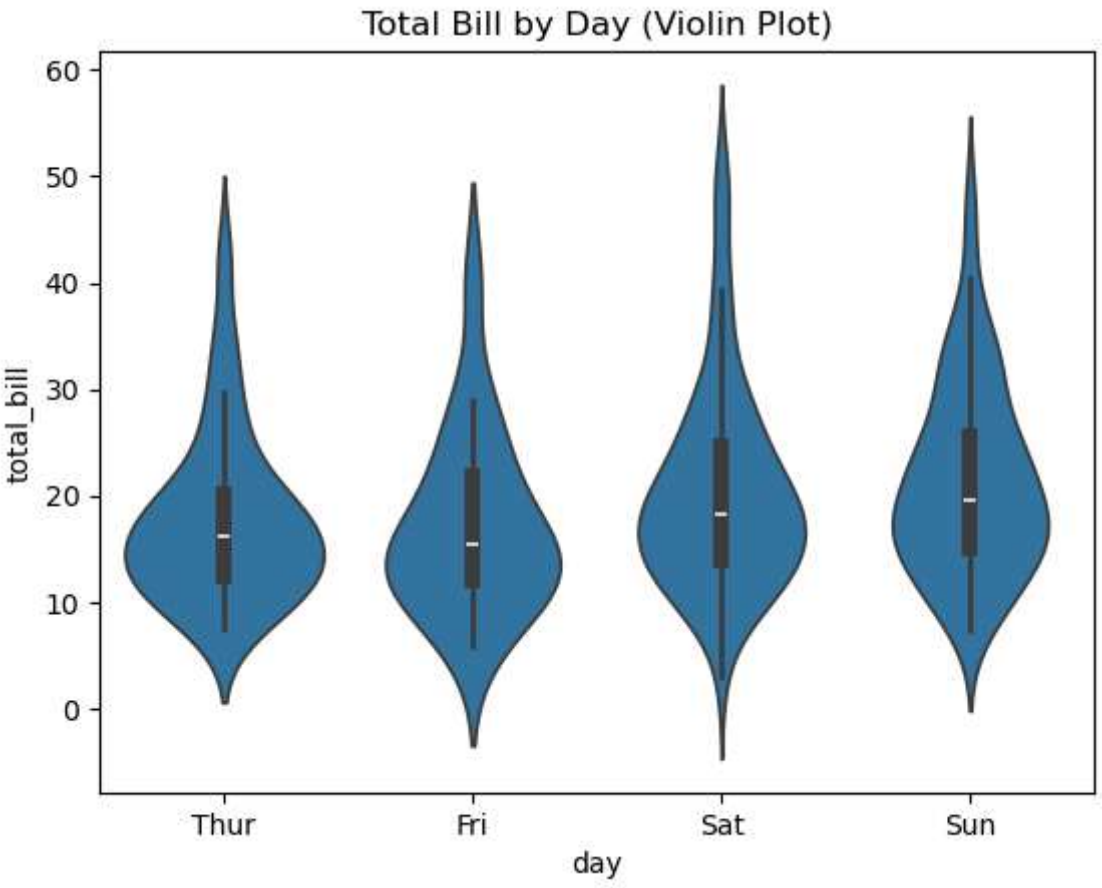


d. Violin Plot – sns.violinplot()

When to use: Combination of boxplot + KDE.

Why: Shows distribution shape and summary stats together.

In [5]:
```
sns.violinplot(data=tips, x="day", y="total_bill")
plt.title("Total Bill by Day (Violin Plot)")
plt.show()
```



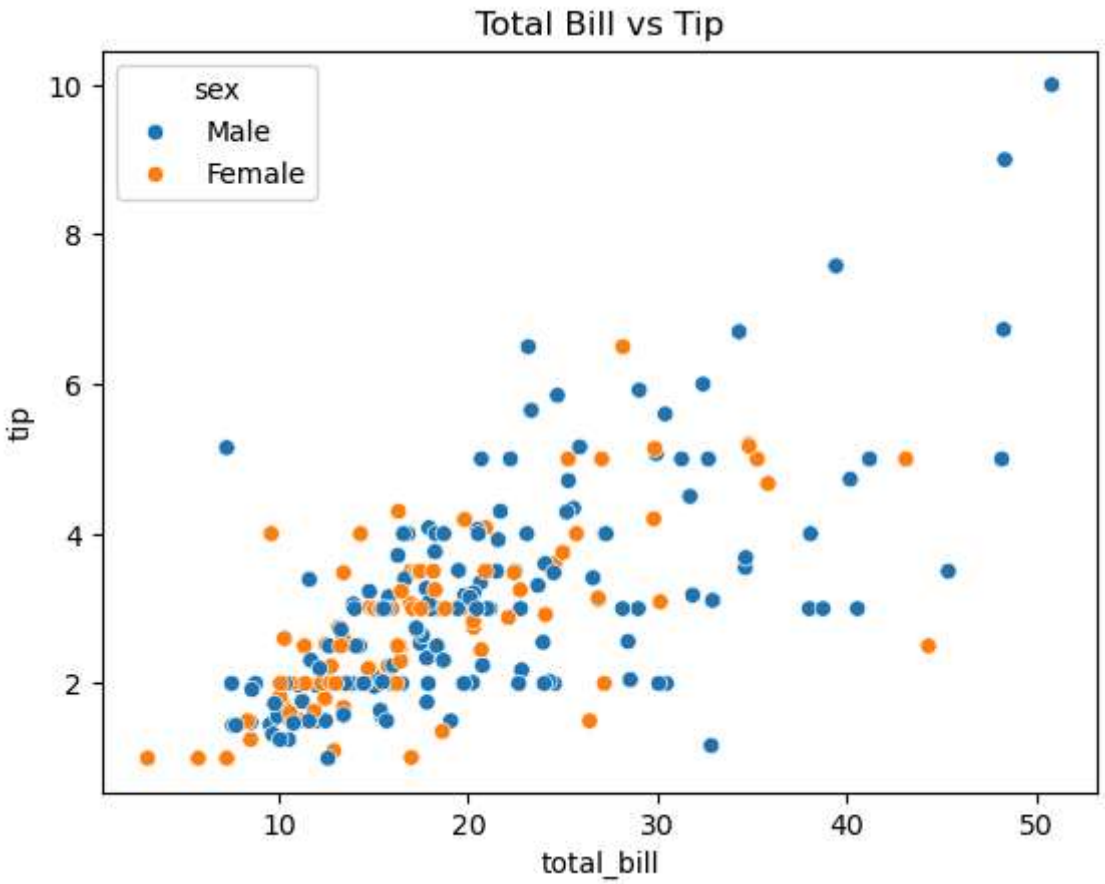
📊 2. Relationship Plots (for two variables)

a. Scatter Plot – sns.scatterplot()

When to use: To check relationship between two continuous variables.

Why: Helps detect correlation or clusters.

In [6]:
```
sns.scatterplot(data=tips, x="total_bill", y="tip", hue="sex")
plt.title("Total Bill vs Tip")
plt.show()
```

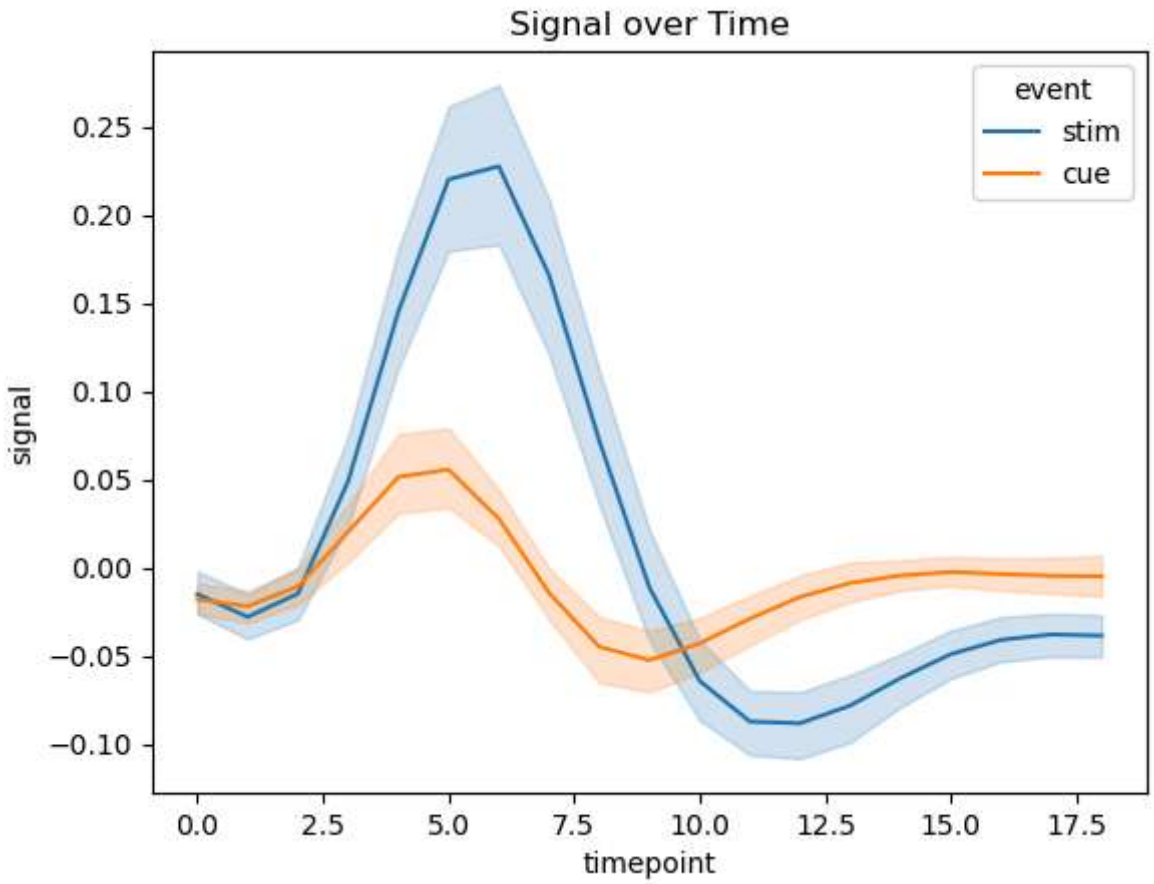


b. Line Plot – sns.lineplot()

When to use: For time series or ordered data.

Why: Shows trend over time or sequential order.

In [7]:
```
fmri = sns.load_dataset("fmri")
sns.lineplot(data=fmri, x="timepoint", y="signal", hue="event")
plt.title("Signal over Time")
plt.show()
```

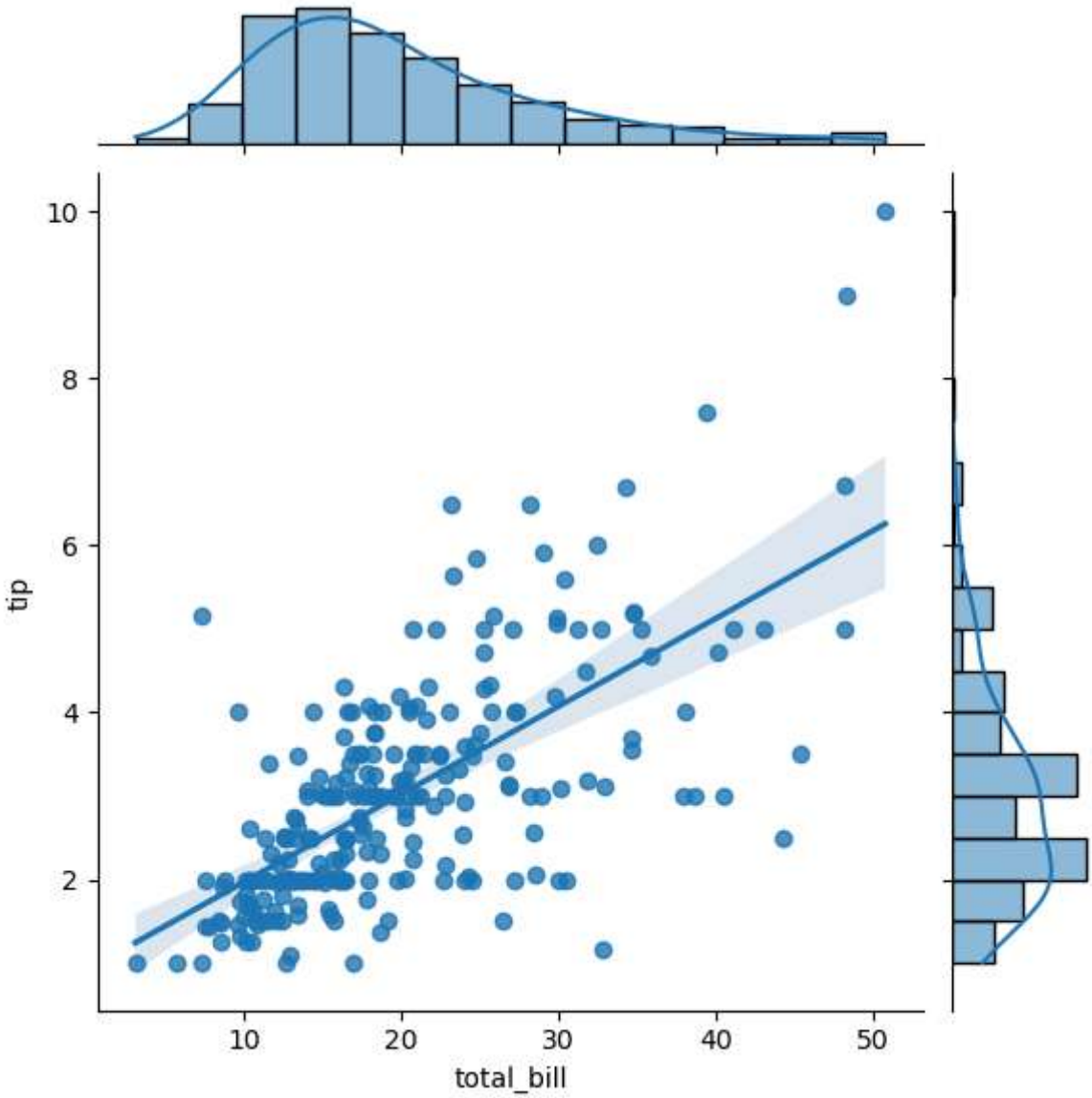


c. Joint Plot – sns.jointplot()

When to use: To show scatter + marginal distributions.

Why: Useful for exploring bivariate relationships deeply.

In [9]:
```
sns.jointplot(data=tips, x="total_bill", y="tip", kind="reg")
plt.show()
```



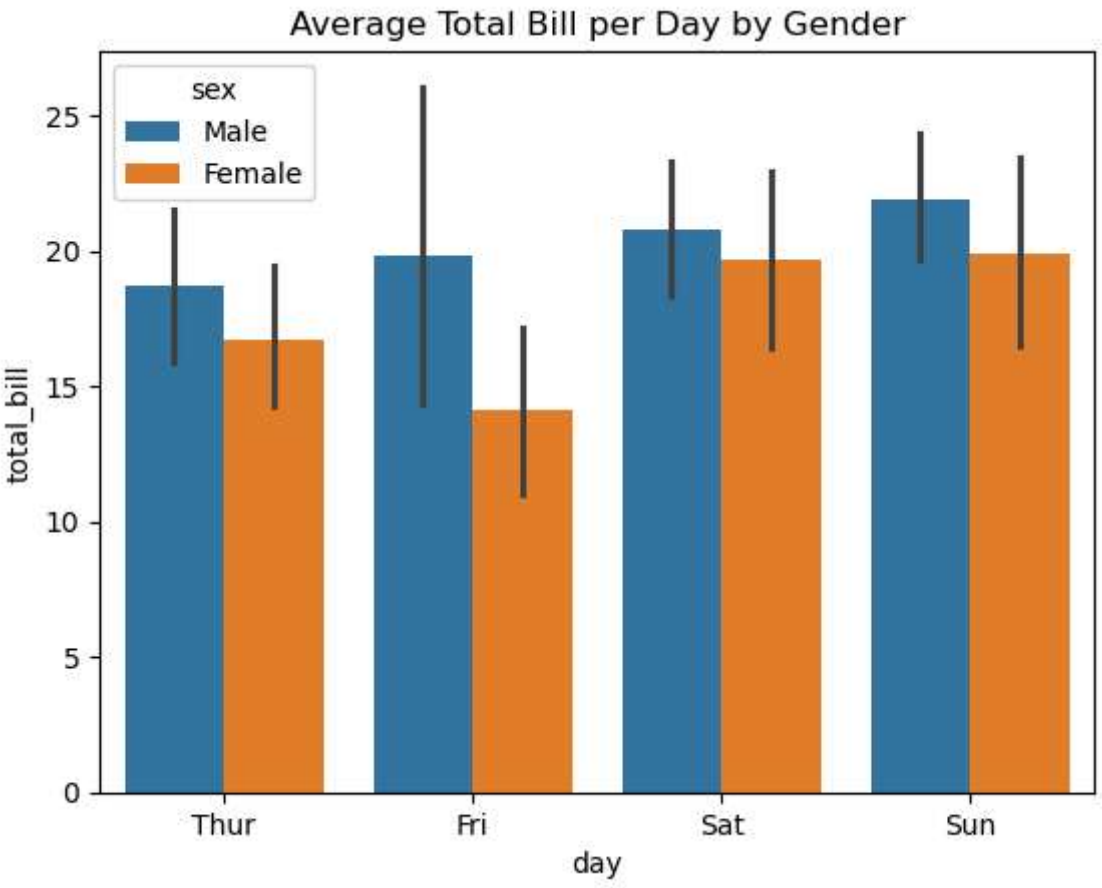
🔁 3. Categorical Plots (comparing categories)

a. Bar Plot – sns.barplot()

When to use: To compare mean values of a numeric variable across categories.

Why: Summarized comparison with confidence intervals.

In [10]:
```
sns.barplot(data=tips, x="day", y="total_bill", hue="sex")
plt.title("Average Total Bill per Day by Gender")
plt.show()
```

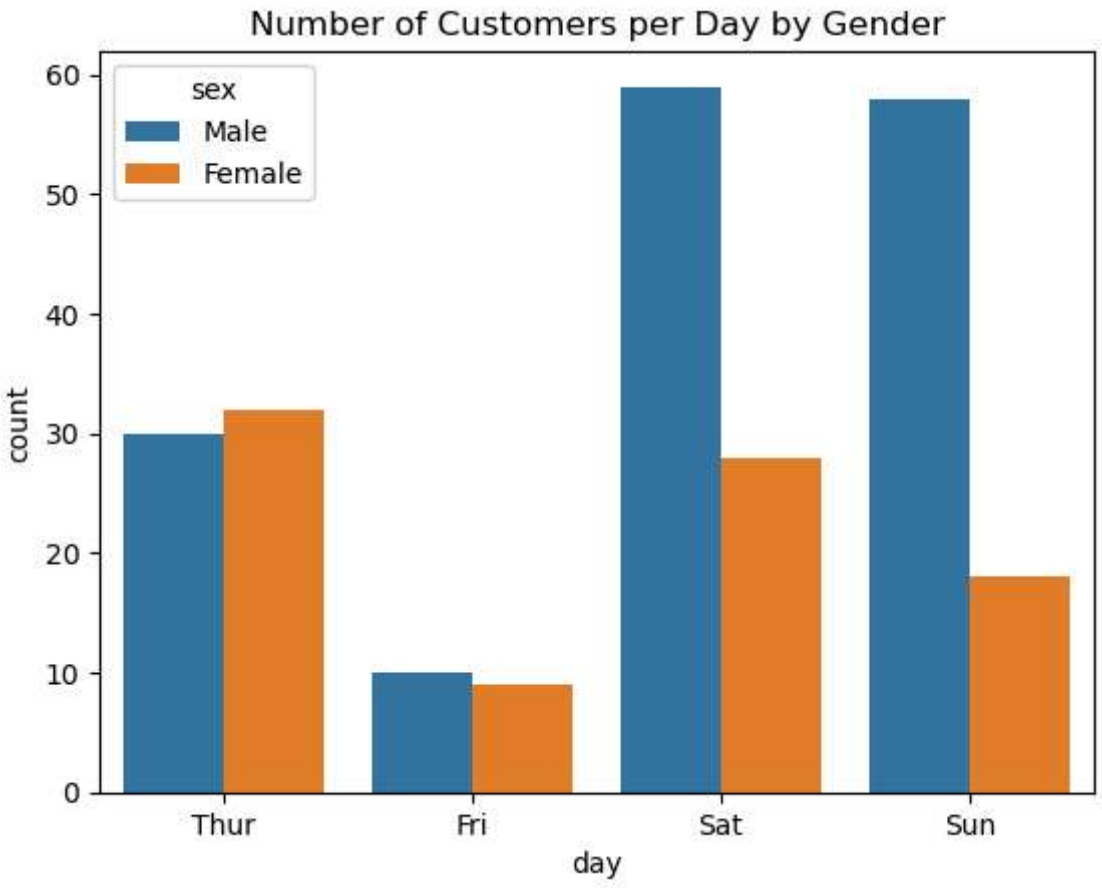


b. Count Plot – sns.countplot()

When to use: To count occurrences of each category.

Why: Good for categorical frequency analysis.

In [11]:
```
sns.countplot(data=tips, x="day", hue="sex")
plt.title("Number of Customers per Day by Gender")
plt.show()
```

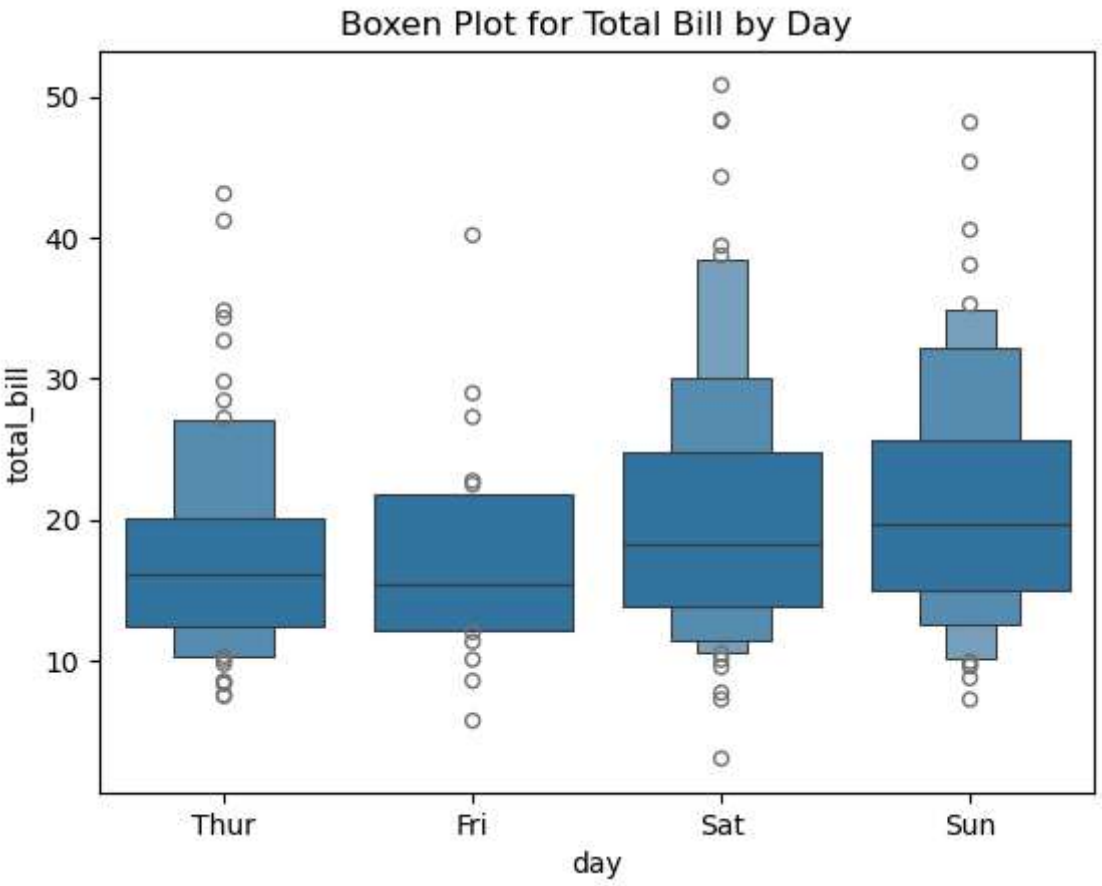


c. Boxen Plot – sns.boxenplot()

When to use: For large datasets, better than boxplot.

Why: Shows detailed distribution of large samples.

In [12]:
```
sns.boxenplot(data=tips, x="day", y="total_bill")
plt.title("Boxen Plot for Total Bill by Day")
plt.show()
```



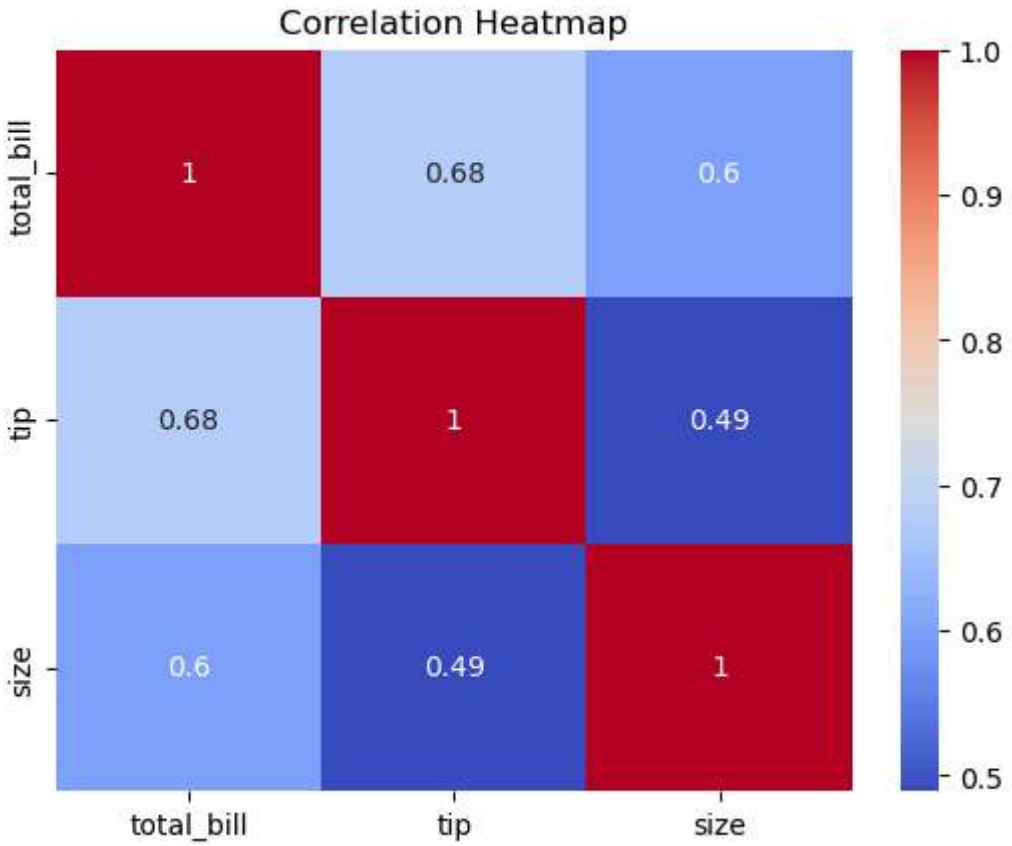
🌐 4. Correlation & Heatmaps

a. Heatmap – sns.heatmap()

When to use: To visualize correlation between multiple numerical variables.

Why: Easy to see which variables are related.

In [13]:
```
corr = tips.corr(numeric_only=True)
sns.heatmap(corr, annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()
```

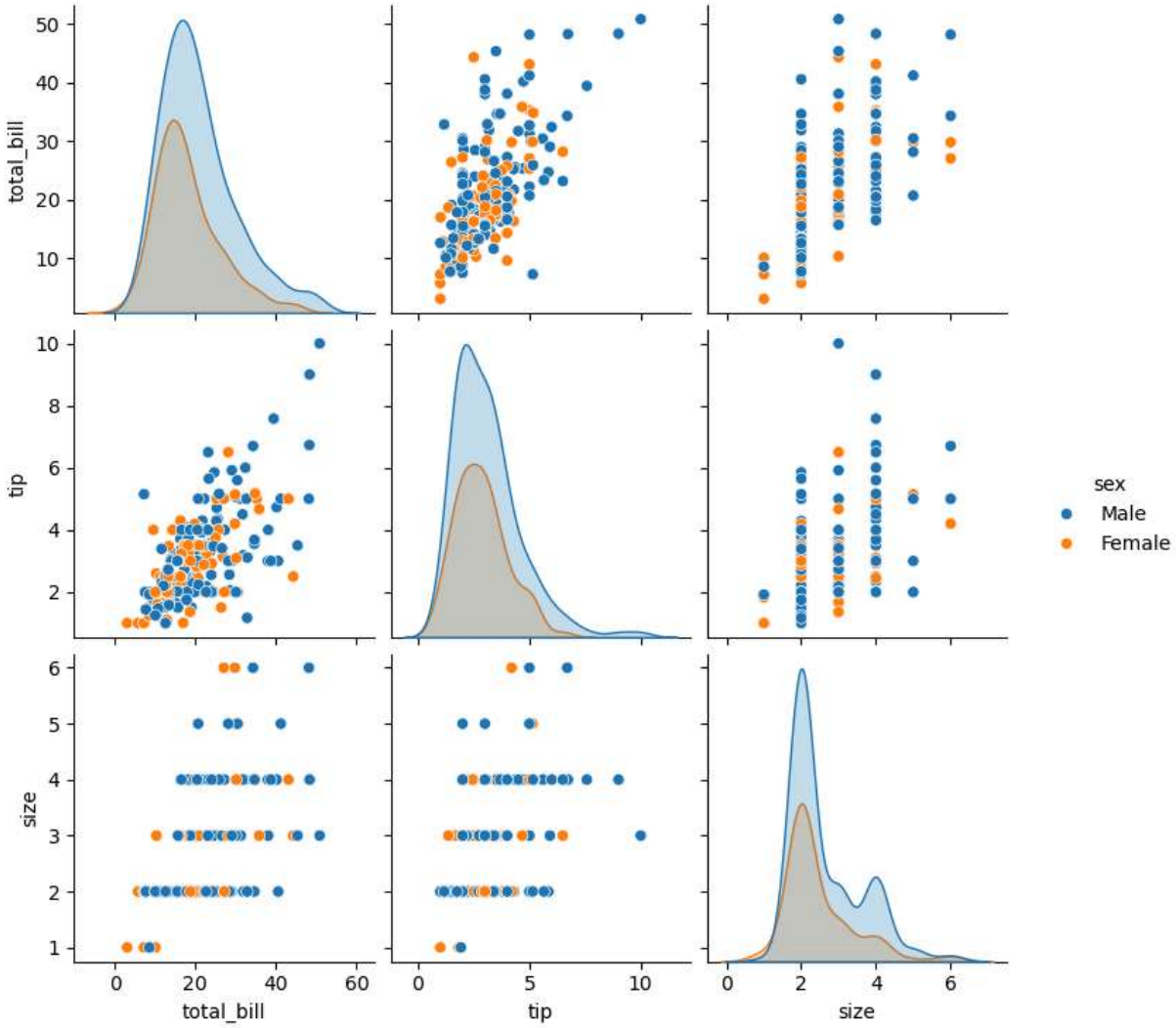


🧩 5. Pair Plot – sns.pairplot()

When to use: To visualize pairwise relationships across multiple numerical variables.

Why: Quick overview of patterns and relationships.

In [14]:
```
sns.pairplot(data=tips, hue="sex")
plt.show()
```


total_bill
tip
size
sex
Male
Female
0
20
40
60
0
5
10
0
2
4
6
1
2
3
4
5
6
2
4
6
8
10
10
20
30
40
50

In [ ]: