Seaborn

Seaborn is a high-level *Python visualization* library that sits on top of *Matplotlib* and integrates tightly with *Pandas*. In a Jupyter notebook, it lets you go from DataFrame to polished, publication-quality graphics with minimal fuss. Here's what you need to know before you start coding:

- **Purpose and Philosophy**
  Seaborn's goal is to simplify common statistical graphics, distribution plots, relational plots, categorical comparisons, heatmaps, regression plots, so you can focus on interpreting your data rather than wrestling with styling details.

- **Data-centric API**
  You almost always pass a pandas DataFrame (or tidy DataFrame) directly to a Seaborn function and refer to columns by name. This makes multi-variable plotting and grouping extremely concise.

- **Built-in Aesthetics**
  It comes with several sensible default themes (e.g. "darkgrid", "whitegrid", "ticks") and color palettes (e.g. "deep", "muted", "colorblind") that you can switch with one function call, no manual tweaking of fonts, line widths or colors.

- **Core Plot Types**

  - **Relational**: scatterplot, lineplot
  - **Distribution**: histplot, kdeplot, violinplot
  - **Categorical**: boxplot, barplot, swarmplot
  - **Regression**: lmplot, regplot
  - **Matrix**: heatmap, clustermap
- **Faceting and Grids**
  With `FacetGrid` and high-level wrappers like `catplot` or `pairplot`, you can easily create multi-panel plots that show subsets of your data side by side.

- **When to Use It**

  - **Exploratory analysis** in notebooks, quickly visualize relationships, distributions, and group comparisons.
  - **Presentation-ready figures**, the defaults are already tuned for clarity and aesthetics.
  - **Statistical insights**, built-in support for fitting and showing regression lines, confidence intervals, and distribution estimates.

Once you import Seaborn in your notebook and set your preferred style and palette, you'll be ready to produce clear, attractive visualizations with just a single function call per chart.

Install Necessary Libraries

In [1]:
```
# Uncommnet the following lines to install the required packages
# if not already installed

# ! pip install numpy
# ! pip install pandas
# ! pip install matplotlib
# ! pip install seaborn
```

Setup

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
%reload_ext autoreload
%autoreload 2
```

Import Data

In [3]:
```
# Check for built-in datasets in seaborn
sns.get_dataset_names()
```

Out[3]:
```
['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 [4]:
```
# Load 'car_crashes' dataset from seaborn
car_crashes = sns.load_dataset('car_crashes')

# Display the first few rows of the dataset
car_crashes.head()
```

Out[4]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
0	18.8	7.332	5.640	18.048	15.040	784.55	145.08	AL
1	18.1	7.421	4.525	16.290	17.014	1053.48	133.93	AK
2	18.6	6.510	5.208	15.624	17.856	899.47	110.35	AZ
3	22.4	4.032	5.824	21.056	21.280	827.34	142.39	AR
4	12.0	4.200	3.360	10.920	10.680	878.41	165.63	CA

In [5]:
```
# Load 'tips' dataset from seaborn
tips = sns.load_dataset('tips')

# Display the first few rows of the dataset
tips.head()
```

Out[5]:

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

In [6]:
```
# Load 'flights' dataset from seaborn
flights = sns.load_dataset('flights')

# Display the first few rows of the dataset
flights.head()
```

Out[6]:

	year	month	passengers
0	1949	Jan	112
1	1949	Feb	118
2	1949	Mar	132
3	1949	Apr	129
4	1949	May	121

In [7]:
```
# Load 'iris' dataset from seaborn
iris = sns.load_dataset('iris')

# Display the first few rows of the dataset
iris.head()
```

Out[7]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [8]:
```
# Load 'attention' dataset from seaborn
attention = sns.load_dataset('attention')

# Display the first few rows of the dataset
attention.head()
```

Out[8]:

	Unnamed: 0	subject	attention	solutions	score
0	0	1	divided	1	2.0
1	1	2	divided	1	3.0
2	2	3	divided	1	3.0
3	3	4	divided	1	5.0
4	4	5	divided	1	4.0

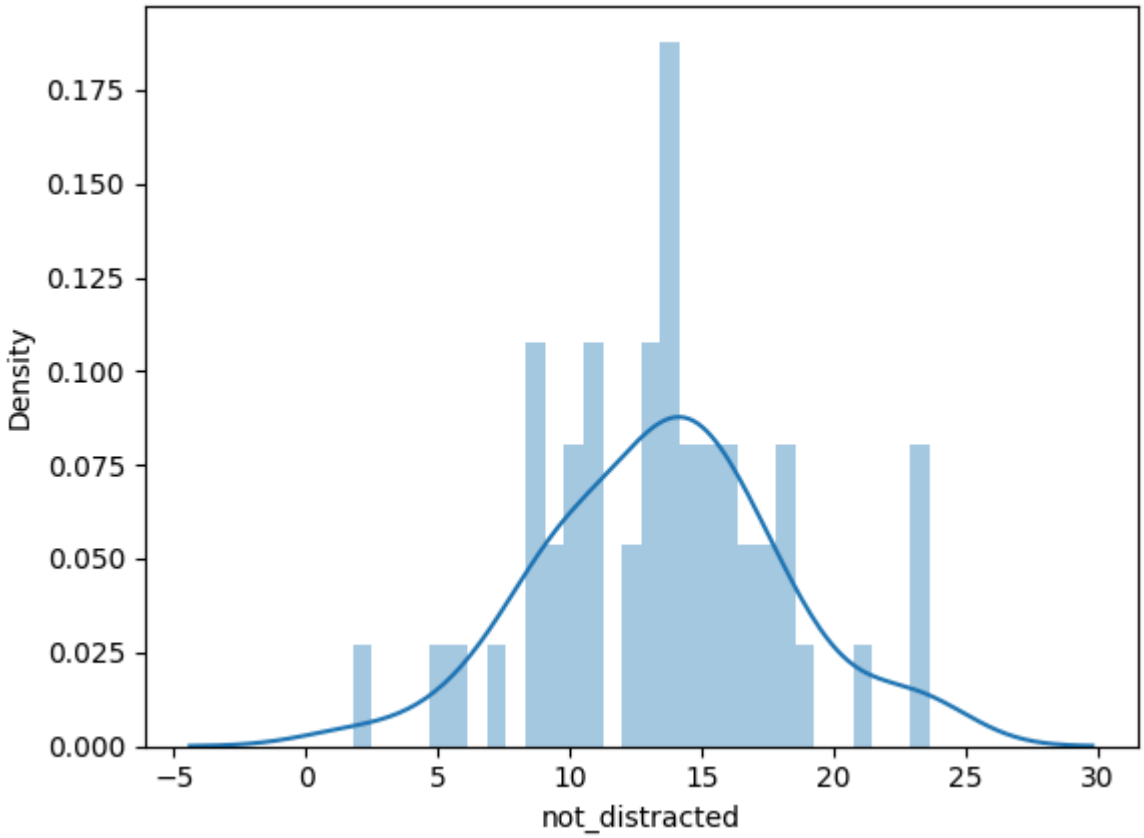
Distribution Plots

Distrubition Plot

In [9]:
```
sns.distplot(car_crashes['not_distracted'], kde=True, bins=30)
```

```
C:\Users\kusha\AppData\Local\Temp\ipykernel_16000\3630582742.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(car_crashes['not_distracted'], kde=True, bins=30)
```

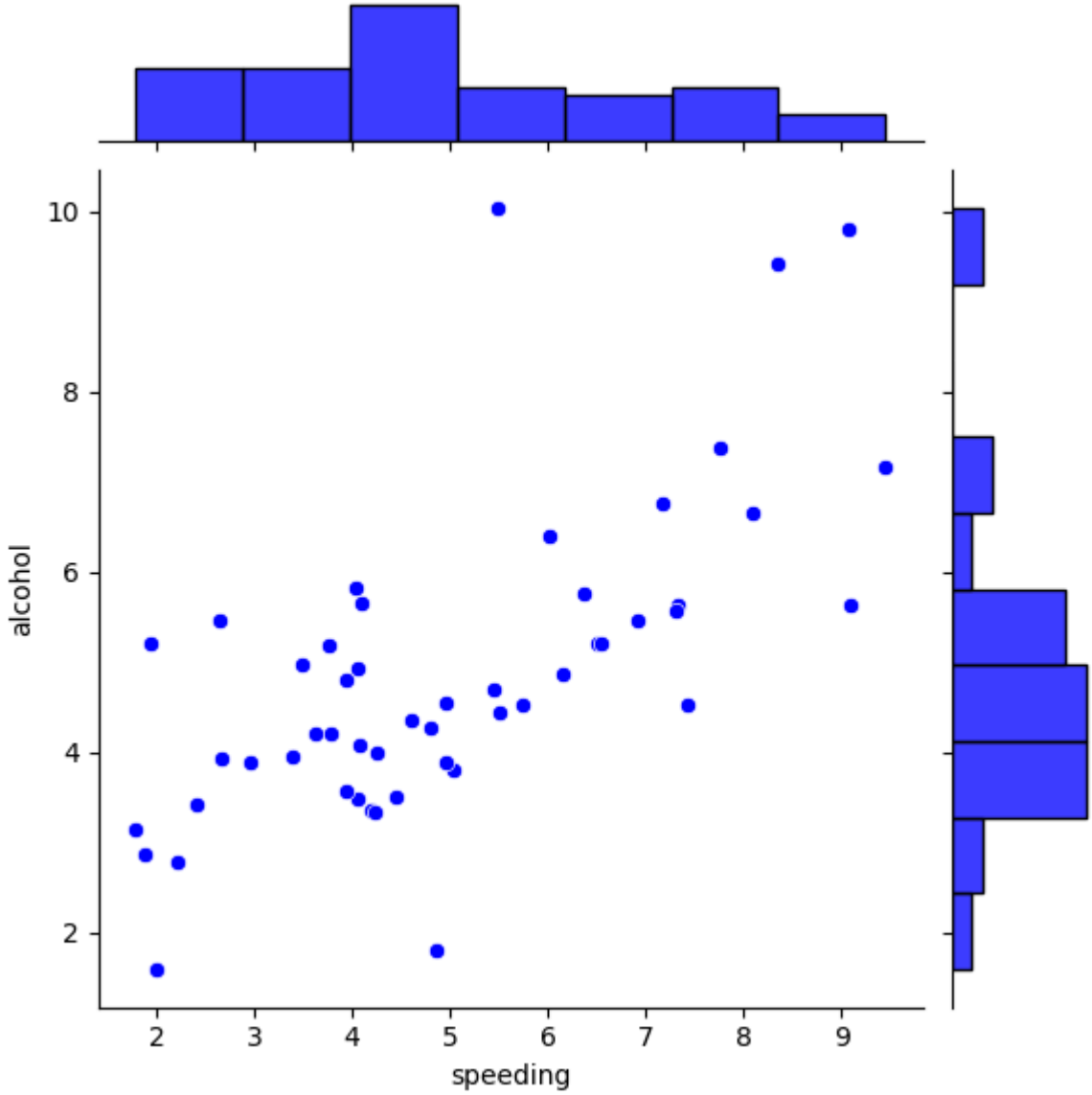
Out[9]:
```
<Axes: xlabel='not_distracted', ylabel='Density'>
```



Joint Plot

In [10]:
```
sns.jointplot(x='speeding', y='alcohol', data=car_crashes,
              kind='scatter', color='blue')
```

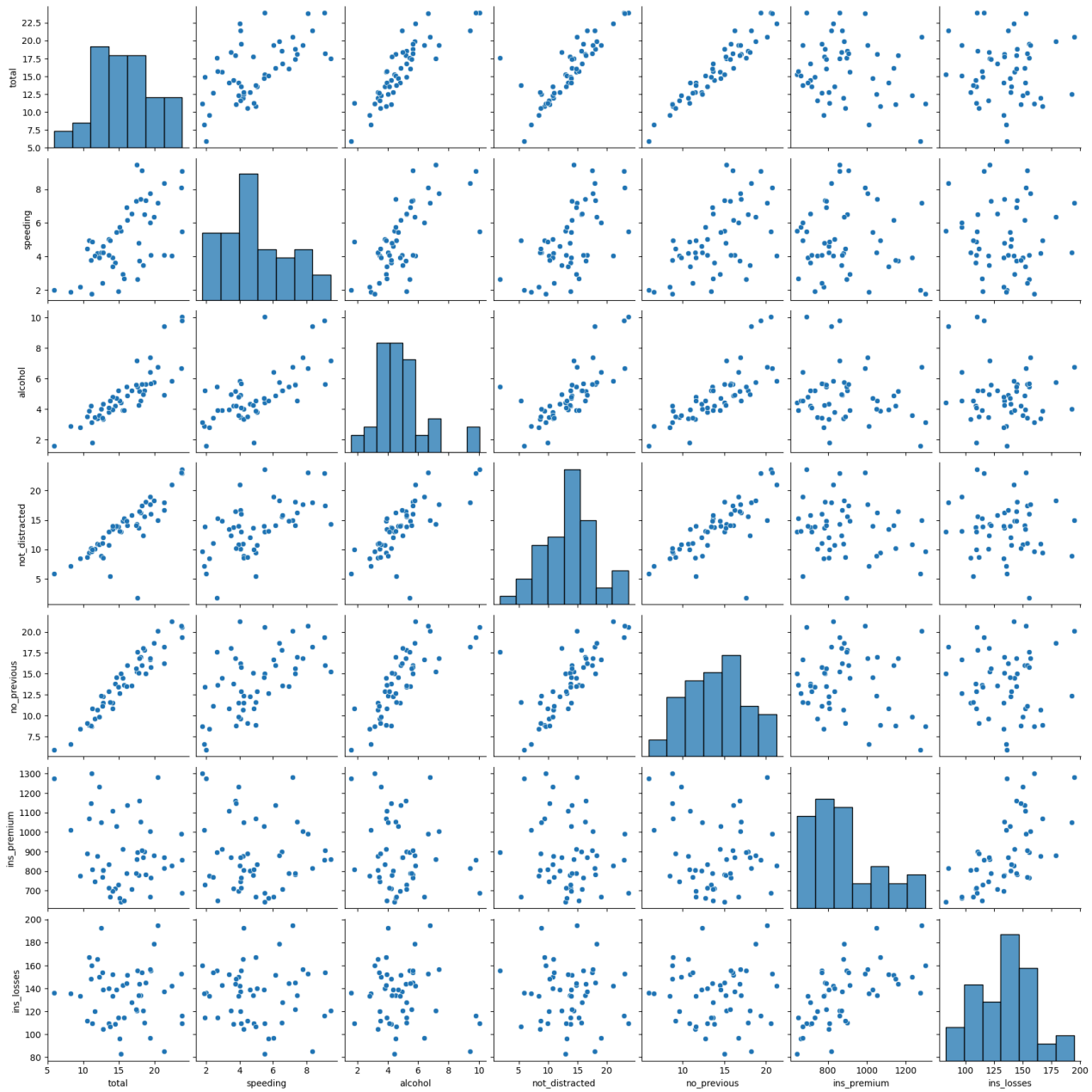
Out[10]:
```
<seaborn.axisgrid.JointGrid at 0x1d71a02a890>
```



Pair Plots

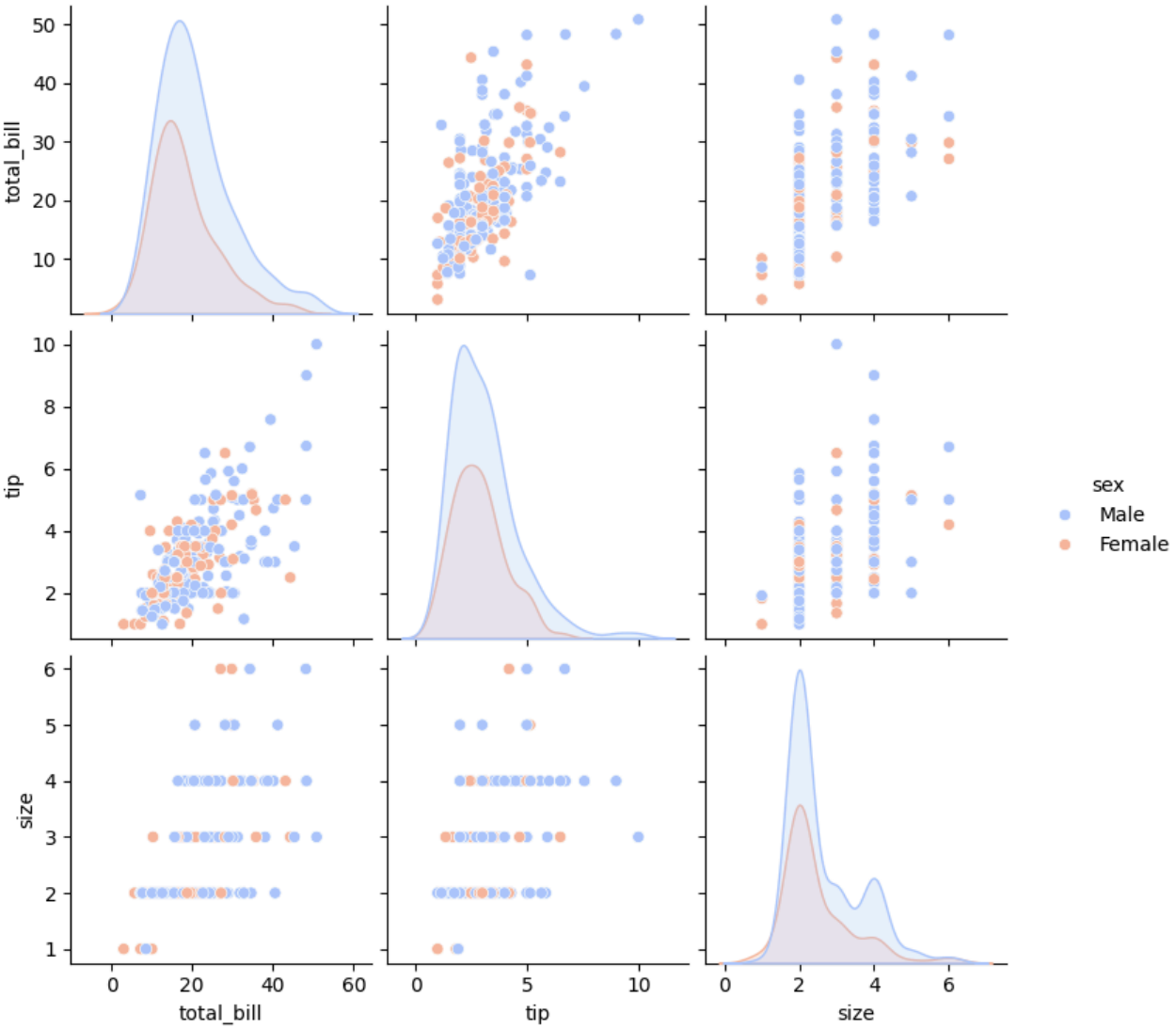
In [11]:
```
sns.pairplot(car_crashes)
```

Out[11]:
```
<seaborn.axisgrid.PairGrid at 0x1d6f113e0e0>
```



In [12]:
```
sns.pairplot(tips, hue='sex', palette='coolwarm')
```

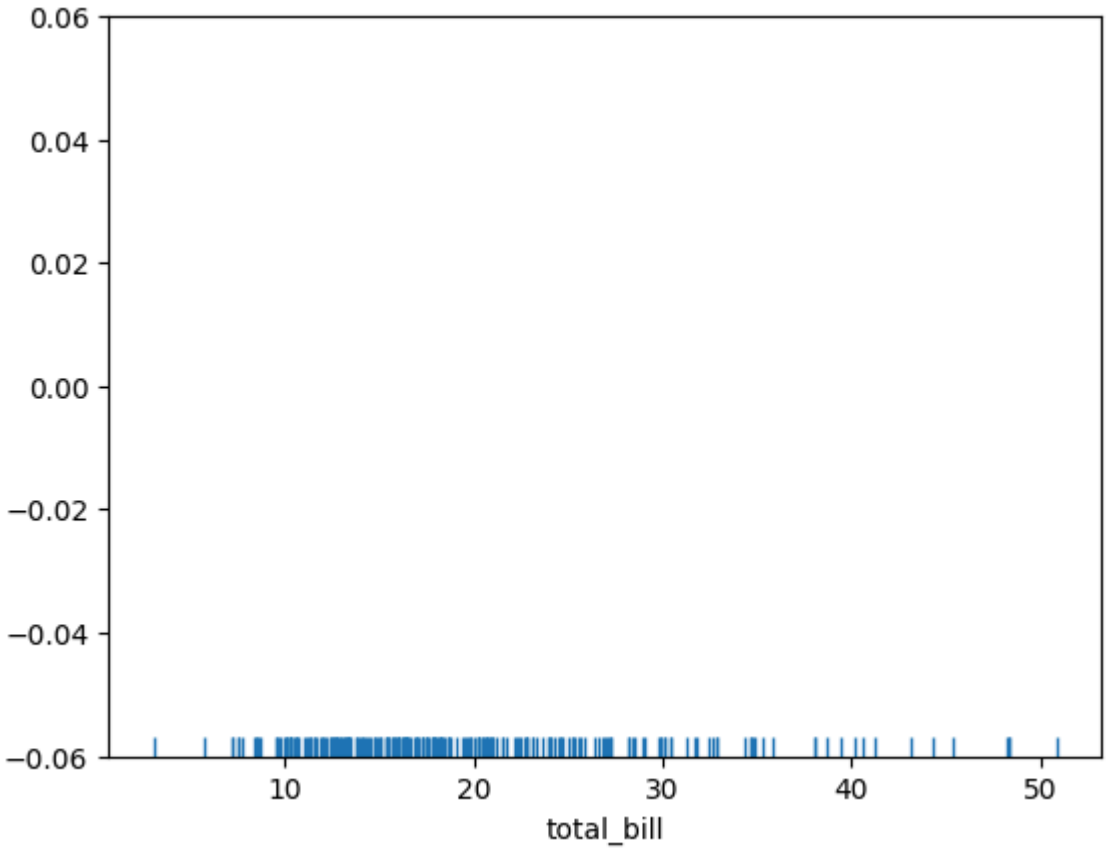
Out[12]:
```
<seaborn.axisgrid.PairGrid at 0x1d72065a320>
```



Rug Plot

In [13]:
```
sns.rugplot(tips['total_bill'])
```

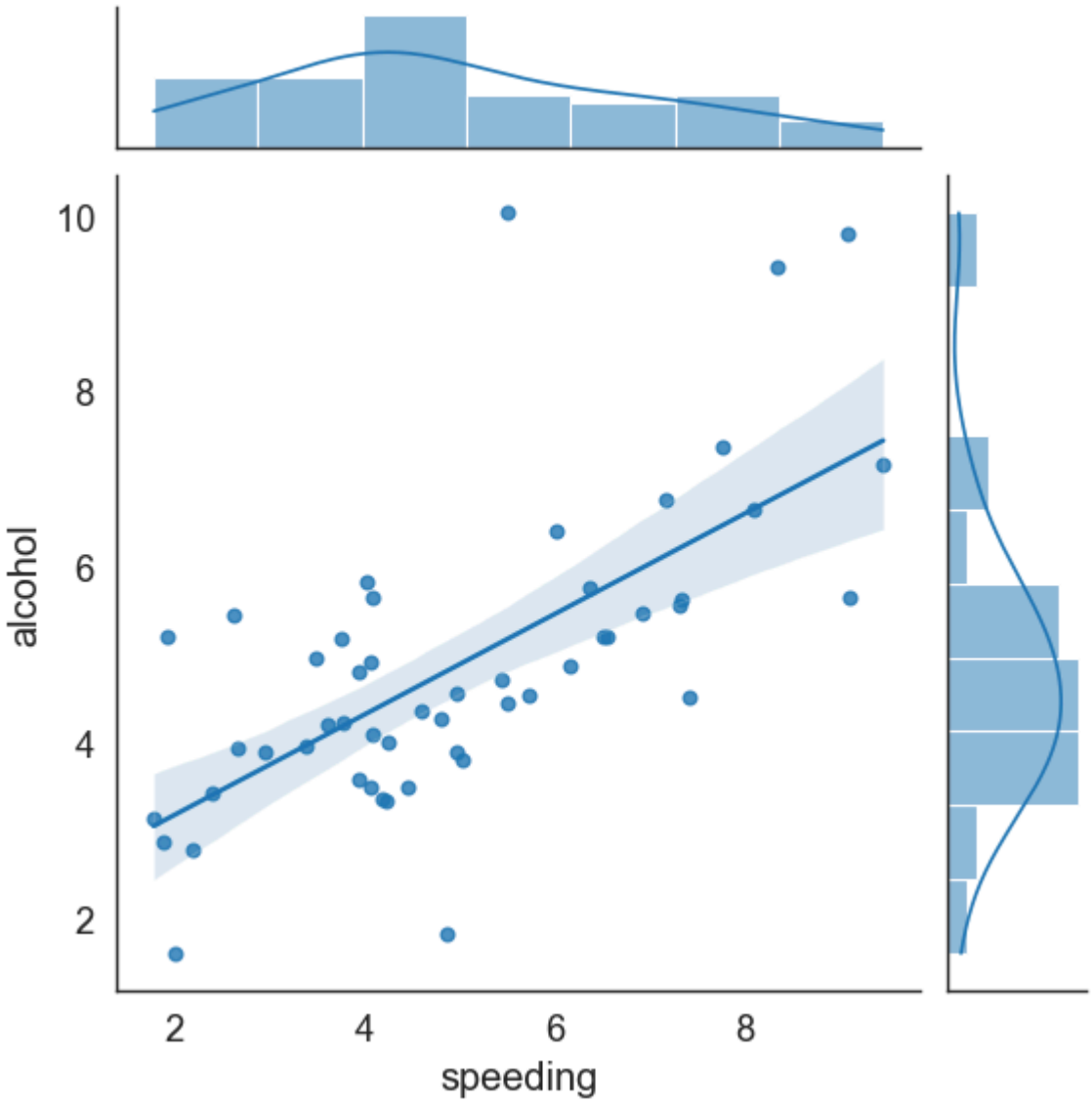
Out[13]:
```
<Axes: xlabel='total_bill'>
```



Styling Plots

In [14]:
```
sns.set_style('white') # white, dark, whitegrid, darkgrid, ticks

plt.figure(figsize=(10, 6)) # Set figure size

sns.set_context('paper', font_scale=1.5) # paper, notebook, talk, poster
sns.jointplot(x='speeding', y='alcohol', data=car_crashes, kind='reg')

sns.despine(right=True) # Remove the right spine of the plot
```

```
<Figure size 1000x600 with 0 Axes>
```

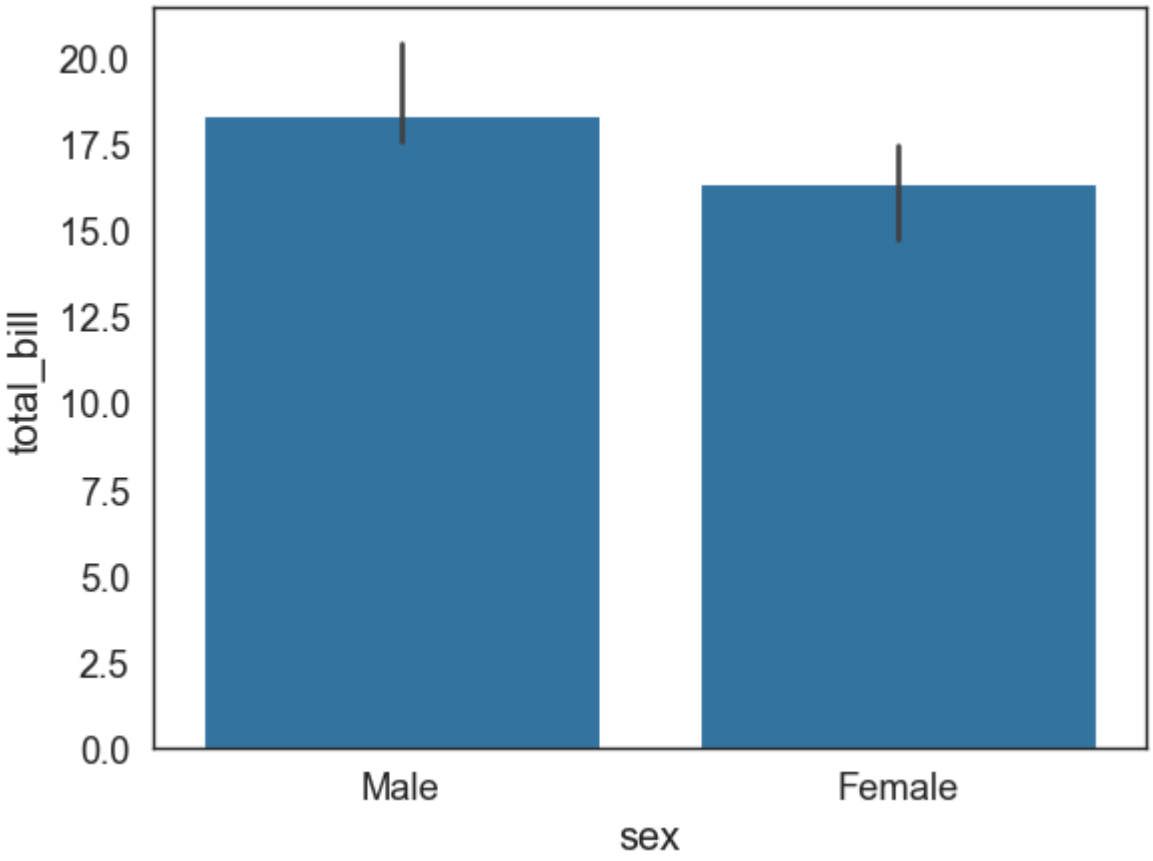


Categorical Plots

Bar Plots

In [15]:
```
sns.barplot(x='sex', y='total_bill', data=tips, estimator=np.median)
```

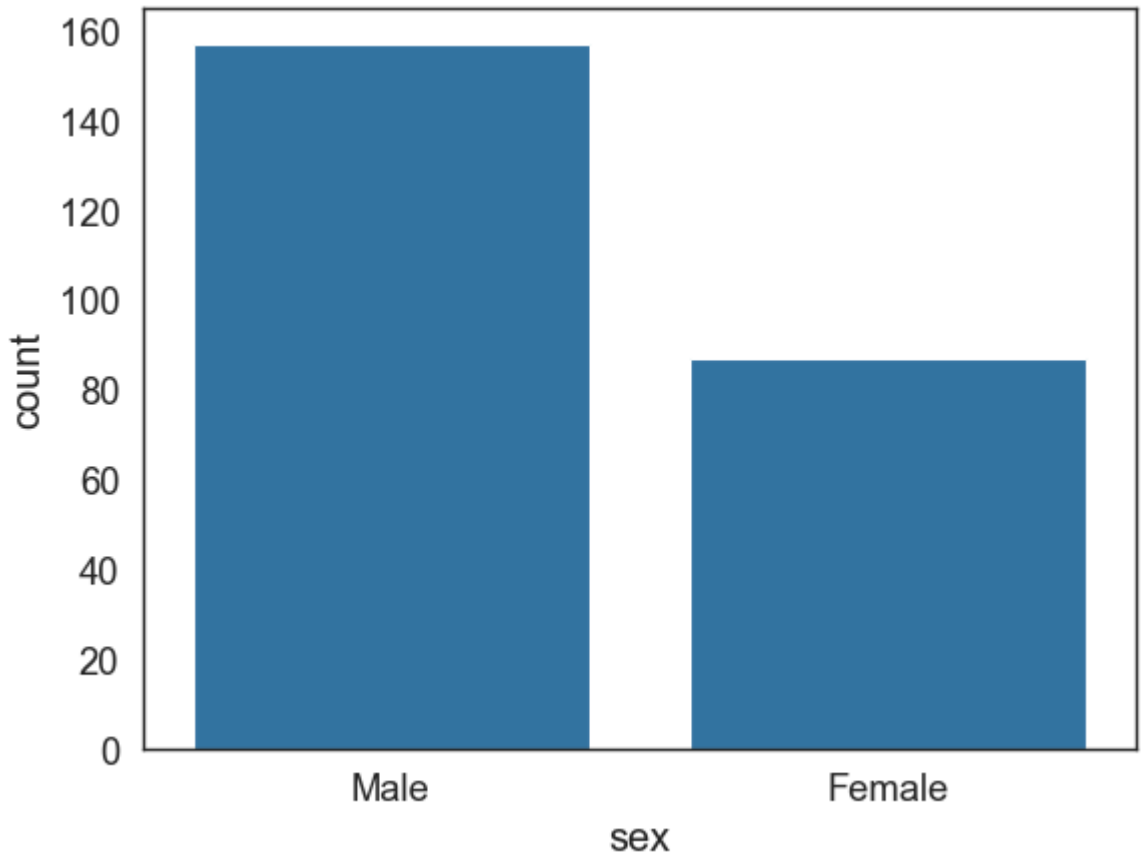
Out[15]:
```
<Axes: xlabel='sex', ylabel='total_bill'>
```



Count Plot

In [16]:
```
sns.countplot(x='sex', data=tips)
```

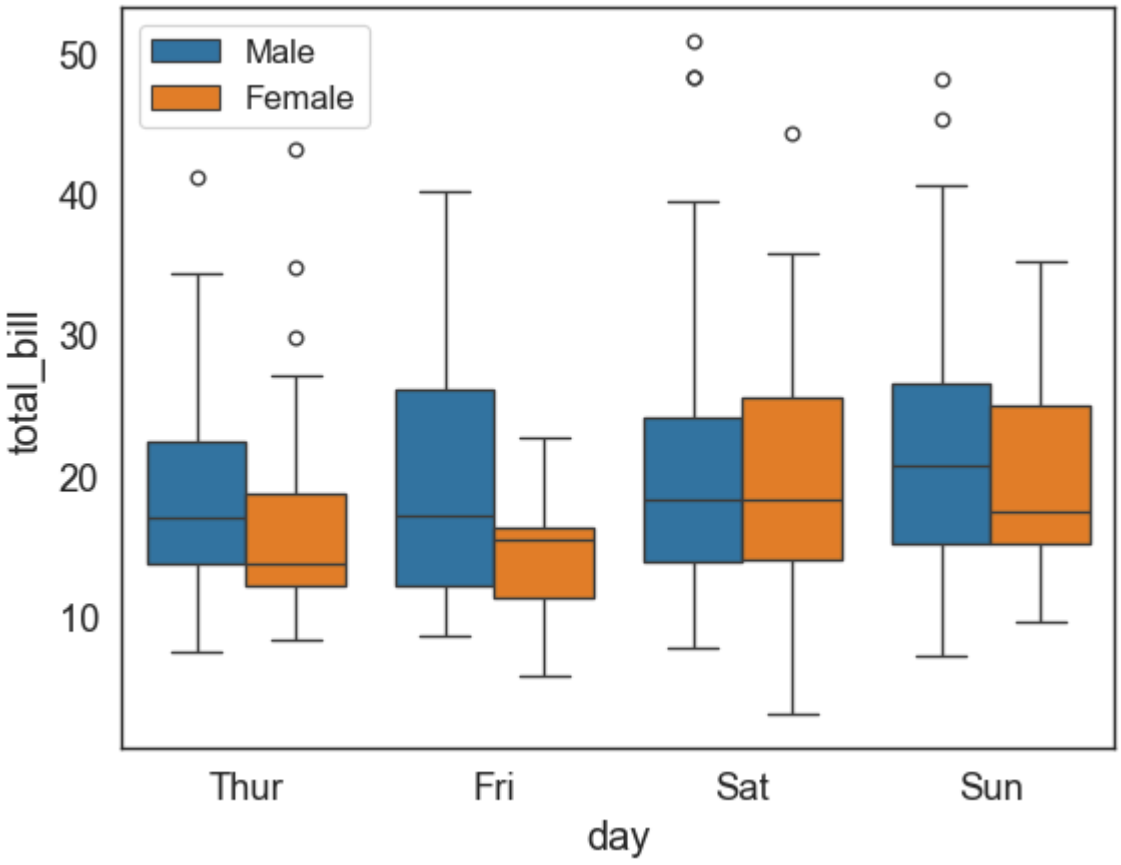
Out[16]: <Axes: xlabel='sex', ylabel='count'>



Box Plot

In [17]:
```
sns.boxplot(x='day', y='total_bill', data=tips, hue='sex')
plt.legend(loc='upper left', fontsize='small')
```

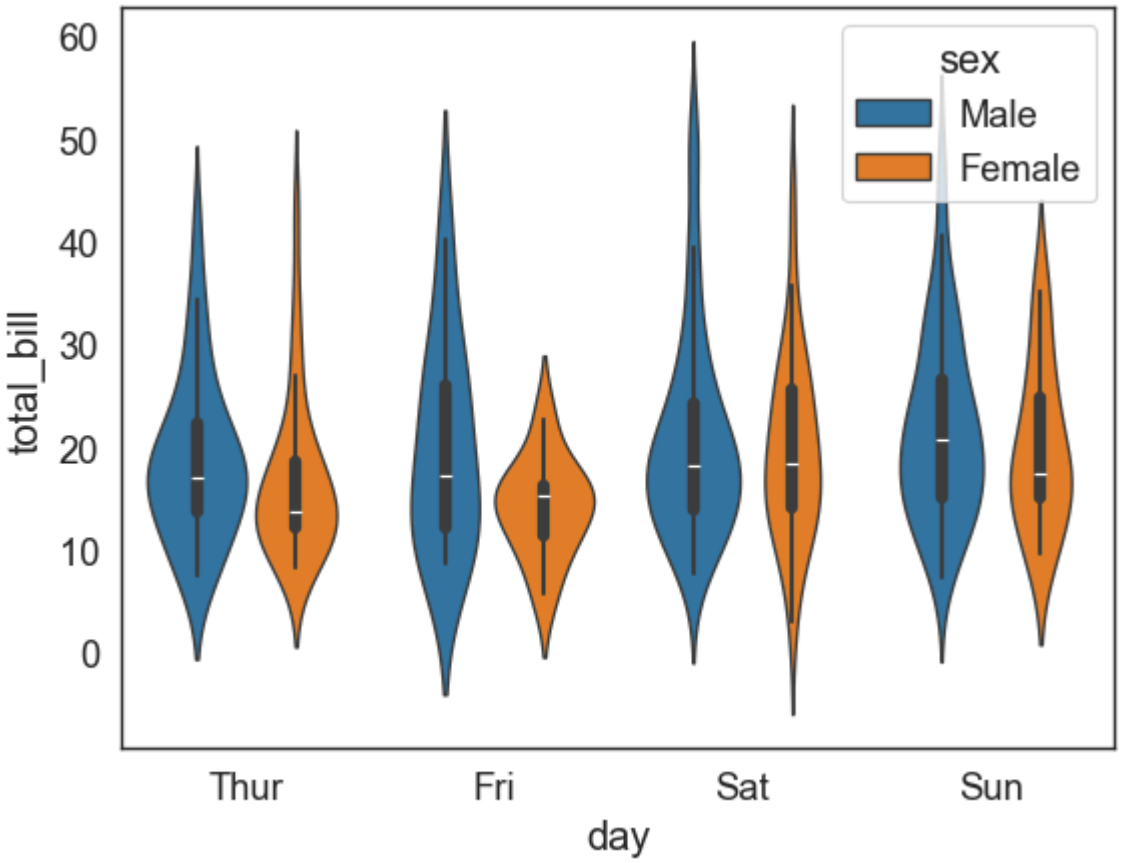
Out[17]:
```
<matplotlib.legend.Legend at 0x1d720fb4640>
```



Violin Plot

In [18]:
```
sns.violinplot(x='day', y='total_bill', data=tips, hue='sex')
```

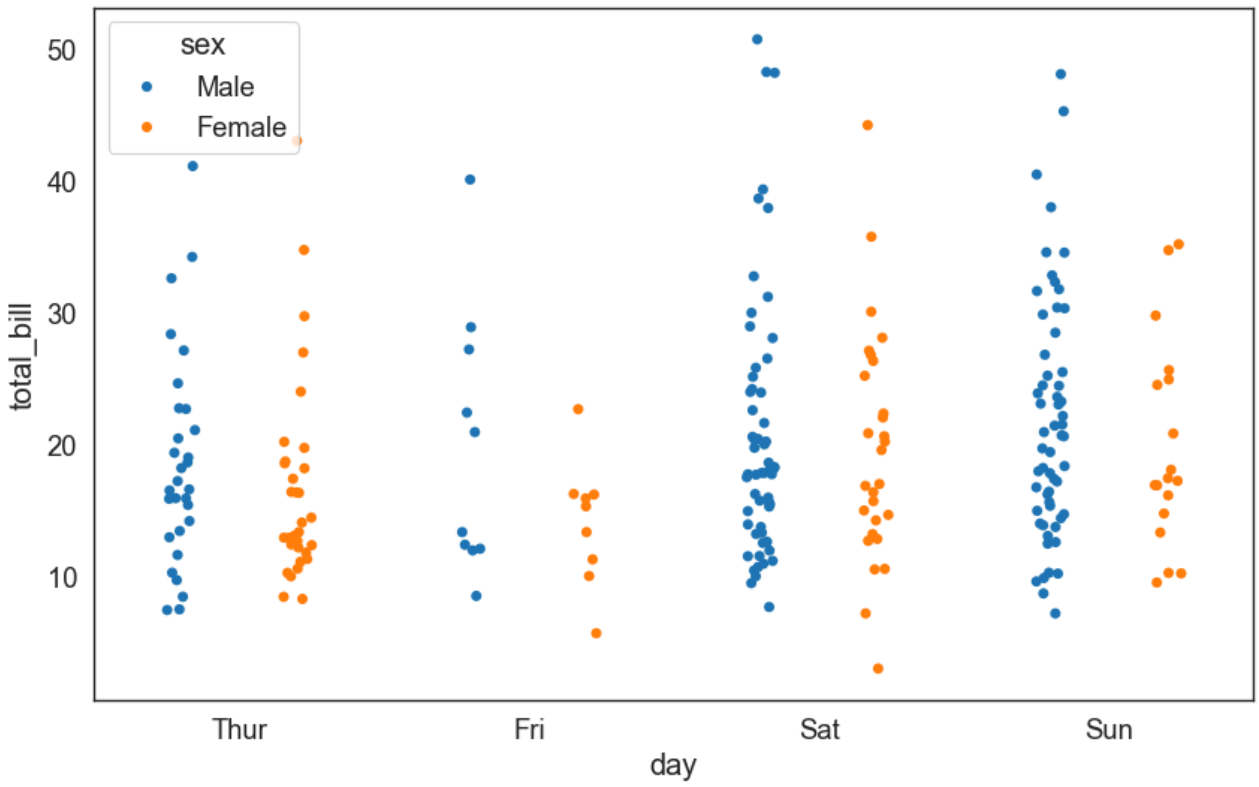
Out[18]:
```
<Axes: xlabel='day', ylabel='total_bill'>
```



Strip Plot

In [19]:
```
plt.figure(figsize=(10, 6)) # Adjust figure size
sns.stripplot(x='day', y='total_bill', data=tips,
              jitter=True, hue='sex', dodge=True)
```

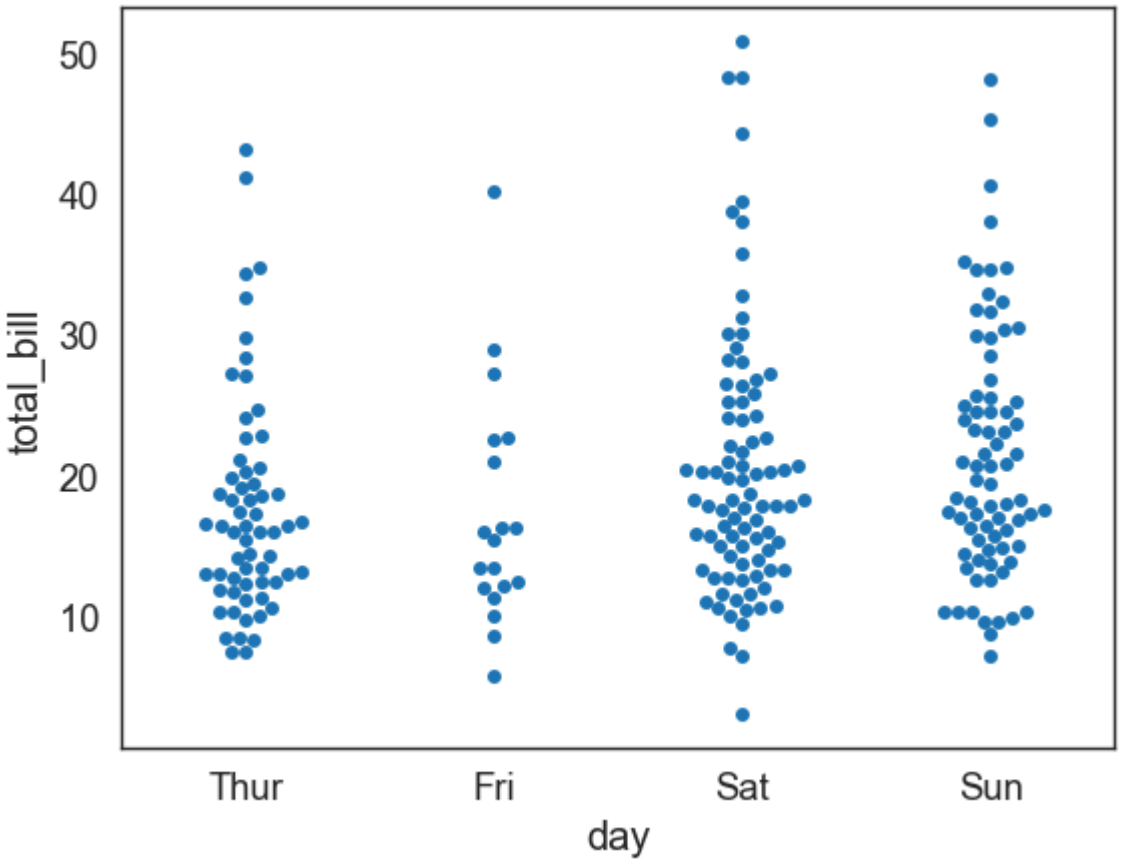
Out[19]: <Axes: xlabel='day', ylabel='total_bill'>



Swarm Plot

In [20]:
```
sns.swarmplot(x='day', y='total_bill', data=tips)
```

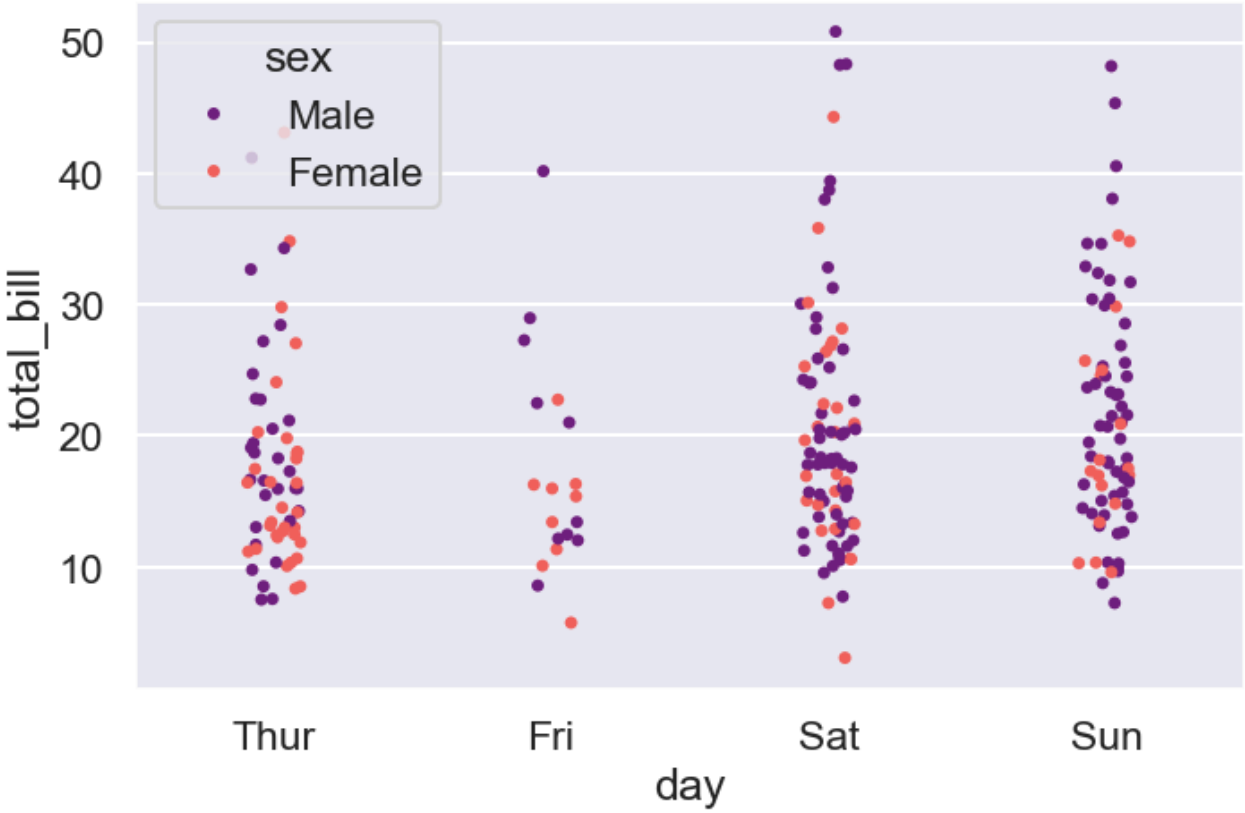
Out[20]:
```
<Axes: xlabel='day', ylabel='total_bill'>
```



Palettes

In [21]:
```
plt.figure(figsize=(8, 5))
sns.set_style('darkgrid')
sns.set_context('talk')
sns.stripplot(x='day', y='total_bill', data=tips,
              jitter=True, hue='sex', palette='magma')
```

Out[21]:
```
<Axes: xlabel='day', ylabel='total_bill'>
```

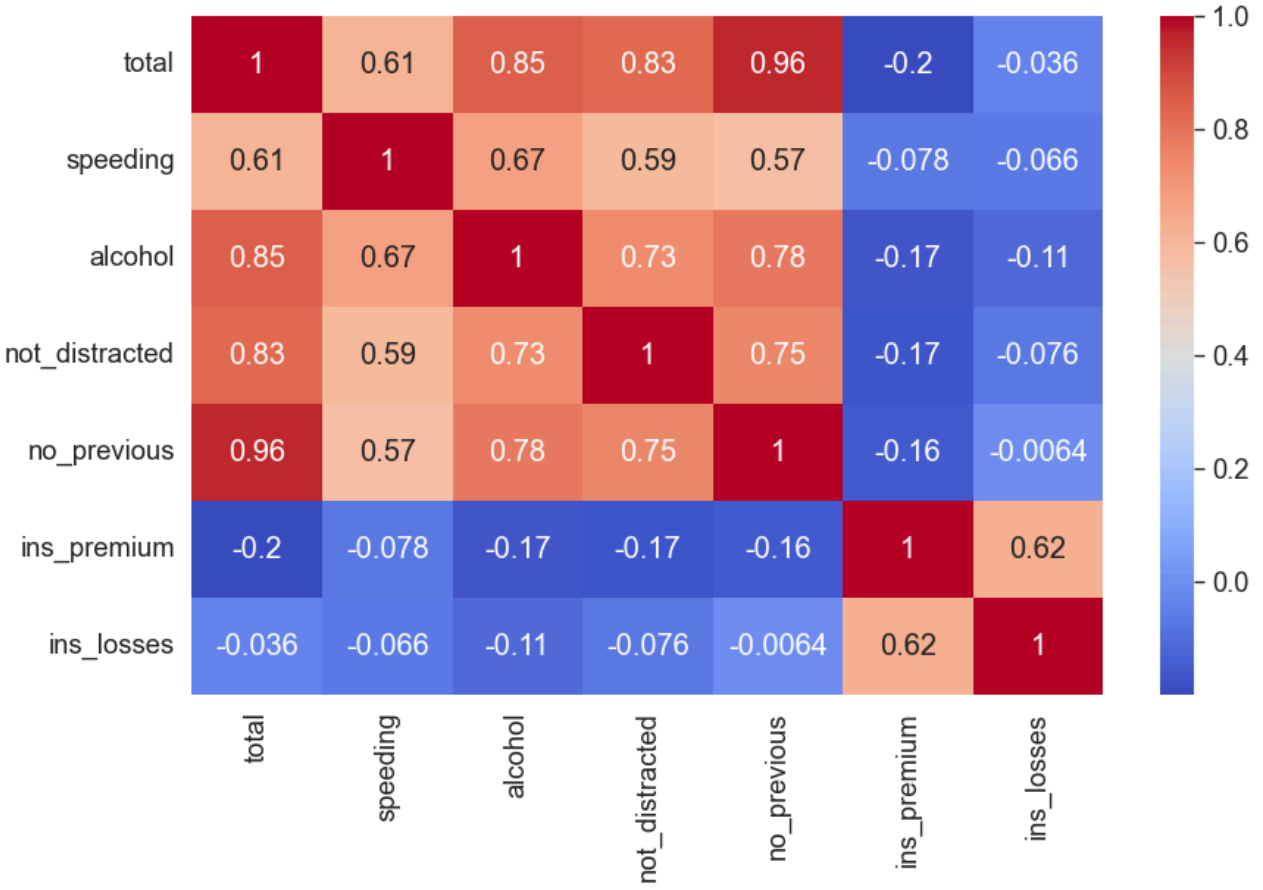


Matrix Plots

Heatmaps

In [22]:
```
plt.figure(figsize=(10, 6))
sns.set_context('paper', font_scale=1.5)

# Get only numerical data
car_crashes_numerical = car_crashes.drop(columns=['abbrev'])
# Find the correlation matrix
crash_matrix = car_crashes_numerical.corr()
# Plot heatmap
sns.heatmap(crash_matrix, annot=True, cmap='coolwarm')
```

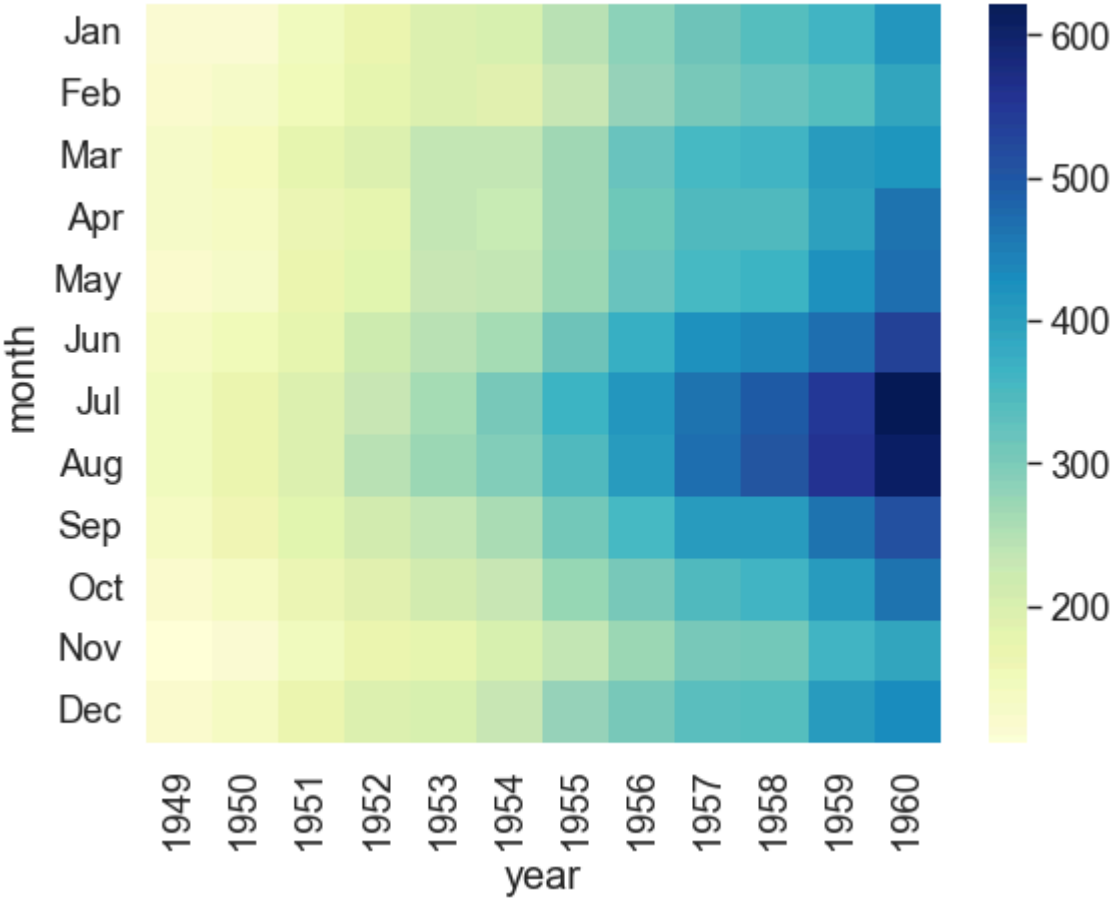
Out[22]: <Axes: >



In [23]:
```
# Create a pivot table for flights dataset
flights_pivot = flights.pivot_table(index='month', columns='year', values='passengers')
# Plot heatmap
sns.heatmap(flights_pivot, cmap='YlGnBu')
```

```
C:\Users\kusha\AppData\Local\Temp\ipykernel_16000\1799231264.py:2: FutureWarning: The default value of observed=False is deprecated and will change to observed=True in a future version of pandas. Specify observed=False to silence this warning and retain the current behavior
  flights_pivot = flights.pivot_table(index='month', columns='year', values='passengers')
```

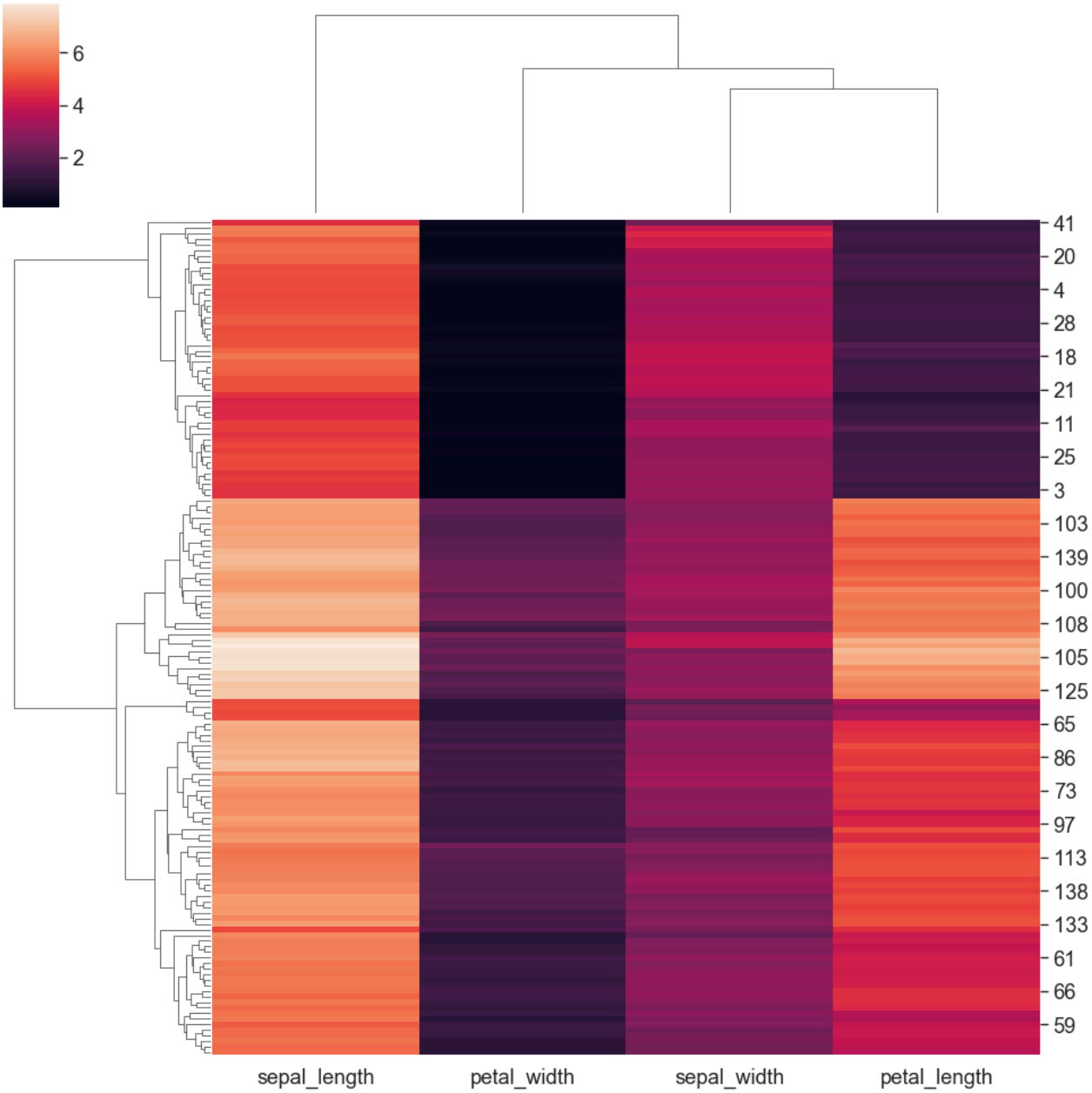
Out[23]: <Axes: xlabel='year', ylabel='month'>



Cluster Map

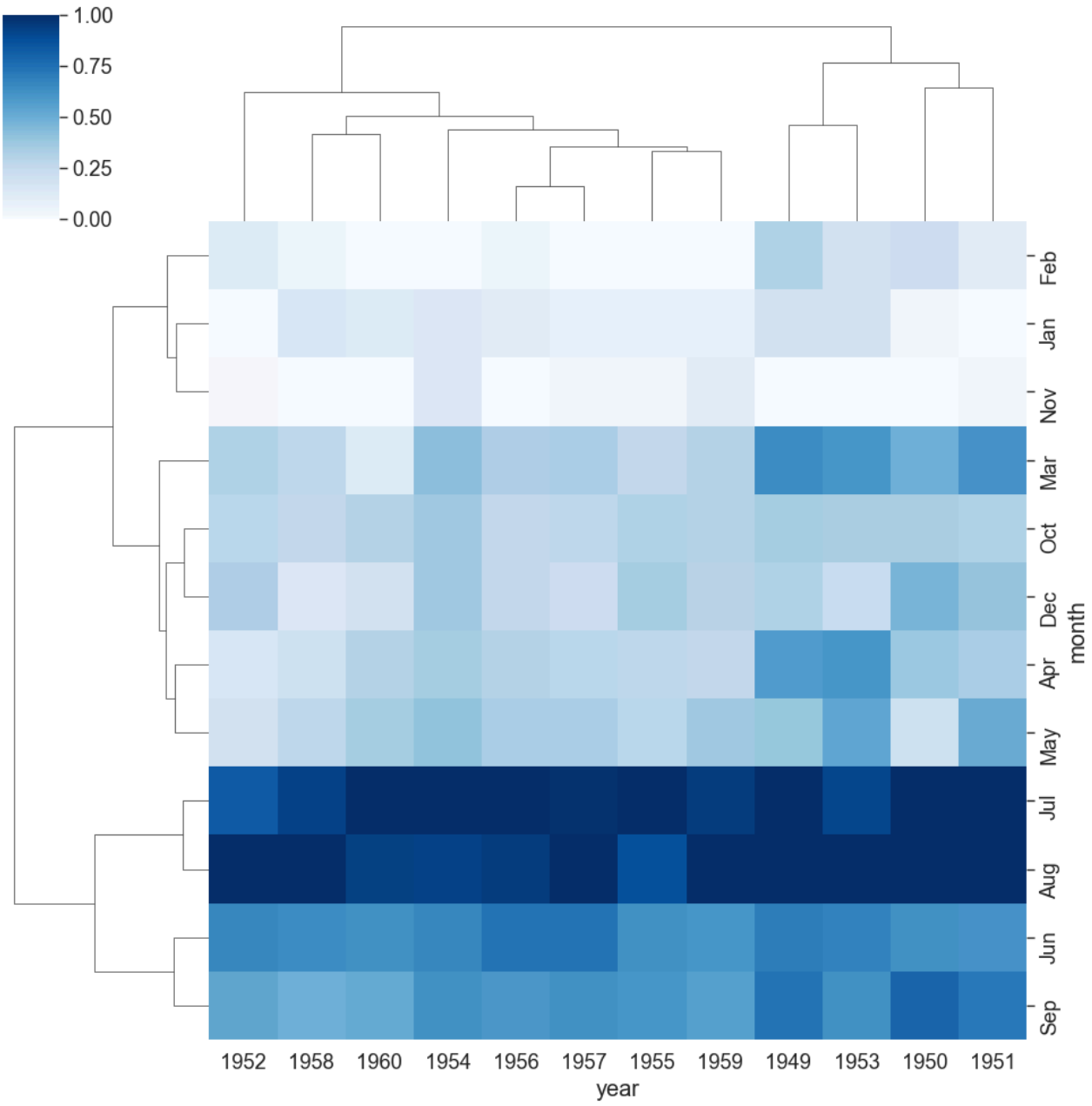
In [24]:
```
species = iris.pop('species')
sns.clustermap(iris)
```

Out[24]:
```
<seaborn.matrix.ClusterGrid at 0x1d725372320>
```



In [25]:
```
sns.clustermap(flights_pivot, cmap='Blues', standard_scale=1)
```

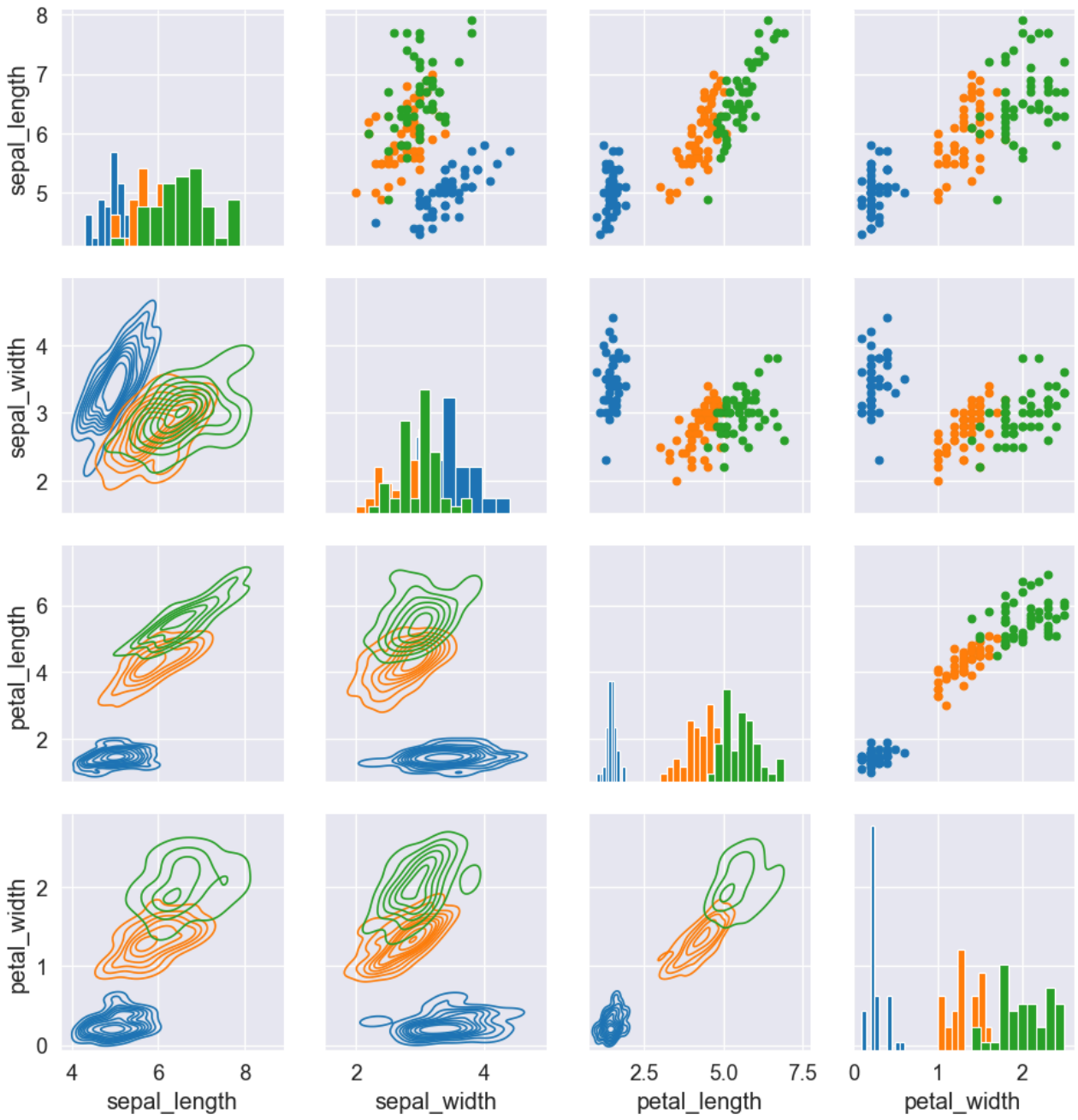
Out[25]:
```
<seaborn.matrix.ClusterGrid at 0x1d72558b0a0>
```



PairGrid

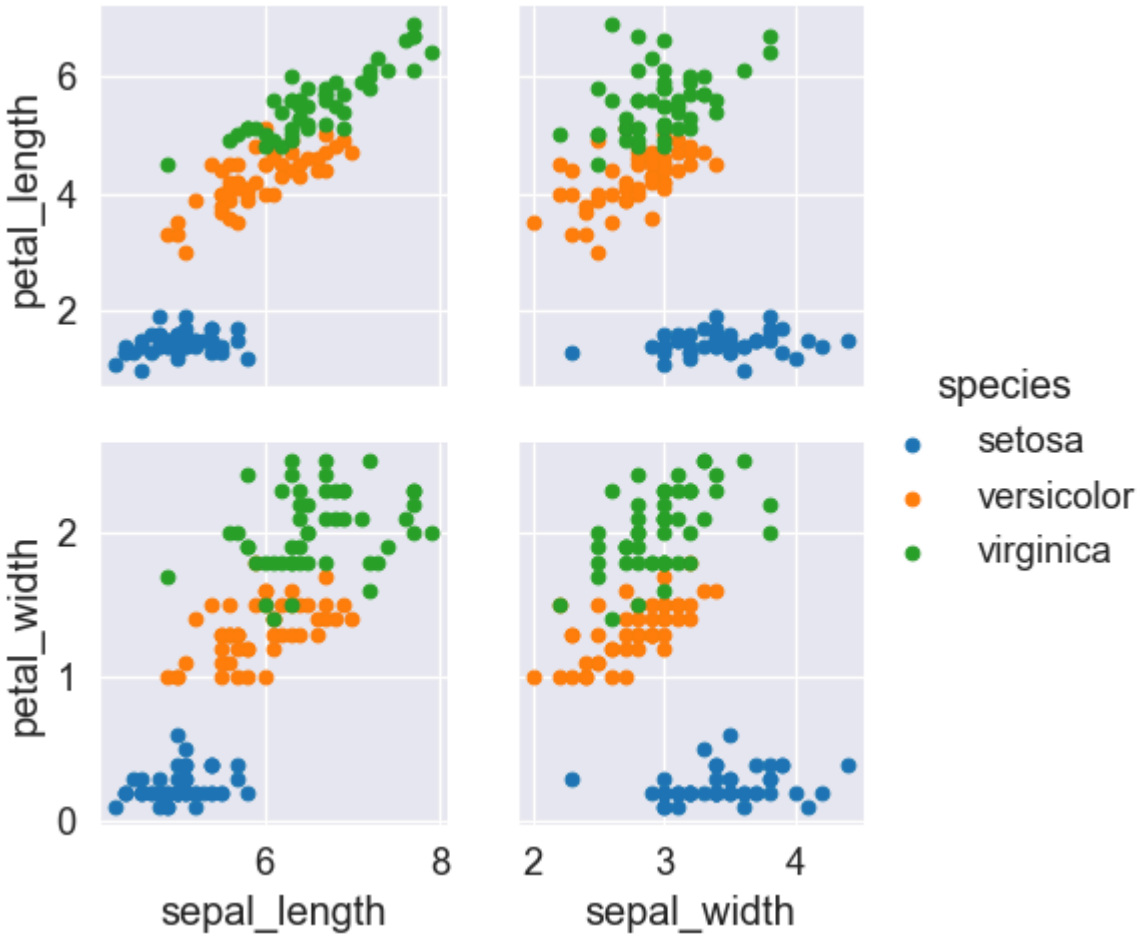
In [26]:
```
iris['species'] = species # Set species back to iris dataset

iris_grid = sns.PairGrid(iris, hue='species')
# iris_grid.map(plt.scatter)
iris_grid.map_diag(plt.hist)
# iris_grid.map_offdiag(plt.scatter)

iris_grid.map_upper(plt.scatter)
iris_grid.map_lower(sns.kdeplot)
```

Out[26]:
```
<seaborn.axisgrid.PairGrid at 0x1d7255a7f70>
```



In [27]:
```
iris_grid_2 = sns.PairGrid(iris, hue='species',
                           x_vars=['sepal_length', 'sepal_width'],
                           y_vars=['petal_length', 'petal_width'])
iris_grid_2.map(plt.scatter)
iris_grid_2.add_legend()
```

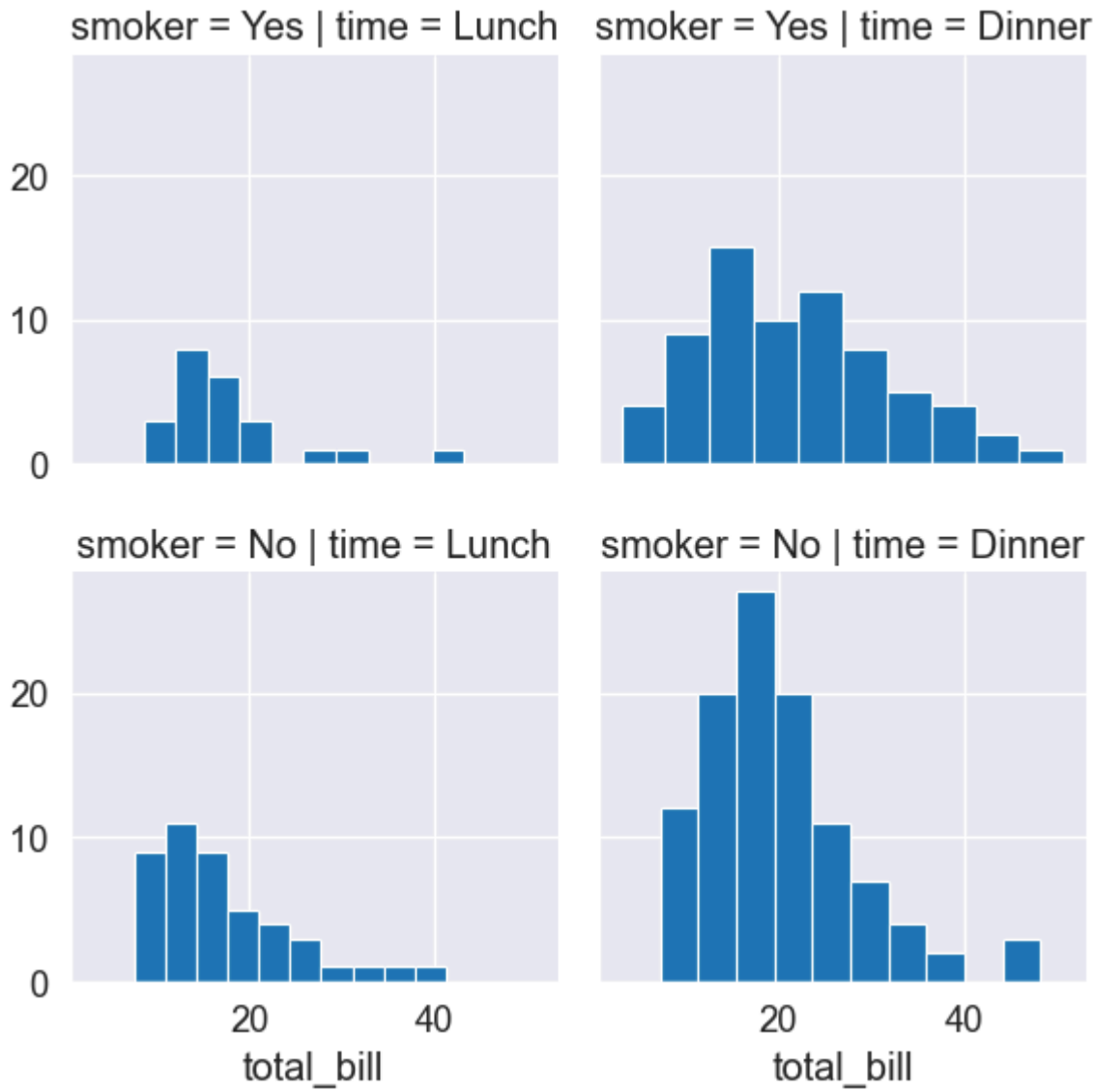
Out[27]: <seaborn.axisgrid.PairGrid at 0x1d725de77c0>



Facet Grid

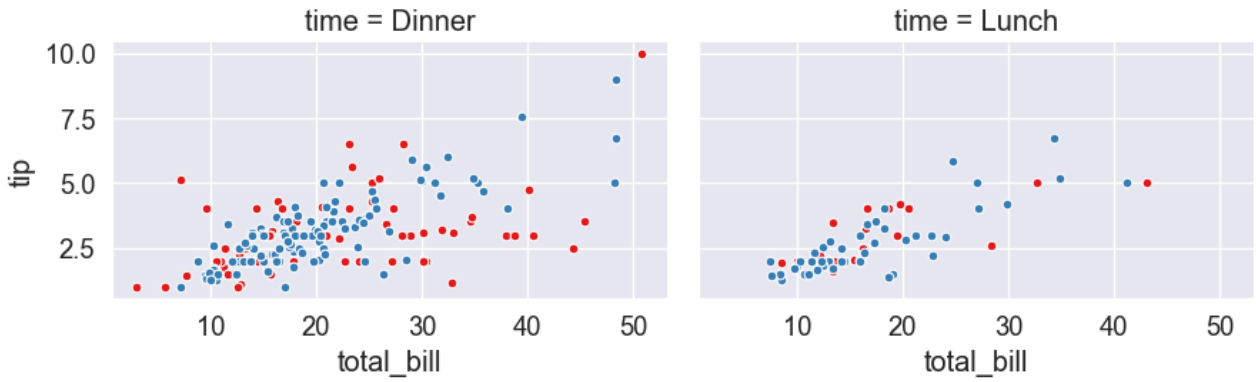
In [28]:
```
tips_fg = sns.FacetGrid(tips, col='time', row='smoker')
tips_fg.map(plt.hist, 'total_bill', bins=10)
```

Out[28]:
```
<seaborn.axisgrid.FacetGrid at 0x1d725fdb3d0>
```



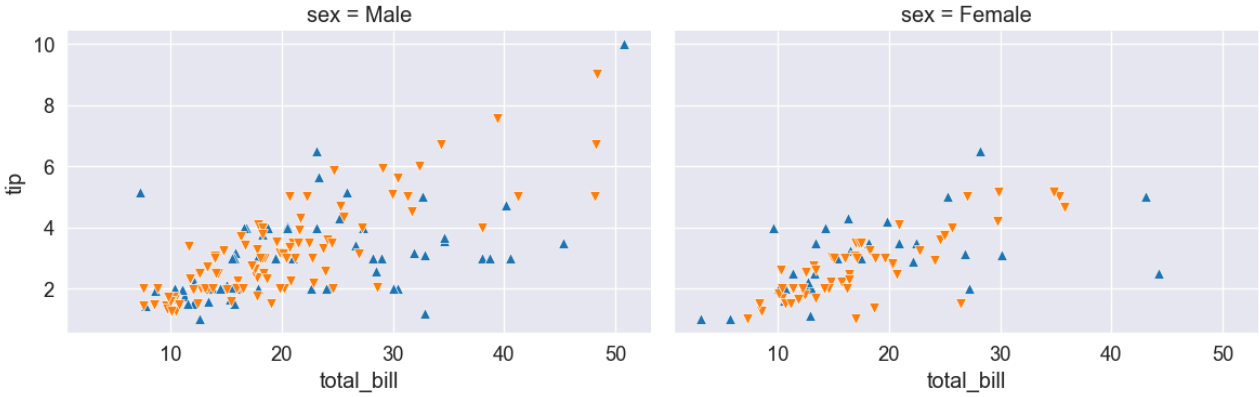
In [29]:
```
tips_fg_2 = sns.FacetGrid(tips, col='time', hue='smoker', height=3,
                          aspect=1.5, col_order=['Dinner', 'Lunch'],
                          palette='Set1')
tips_fg_2.map(plt.scatter, 'total_bill', 'tip', edgecolor='w')
```

Out[29]: <seaborn.axisgrid.FacetGrid at 0x1d725fcb9a0>



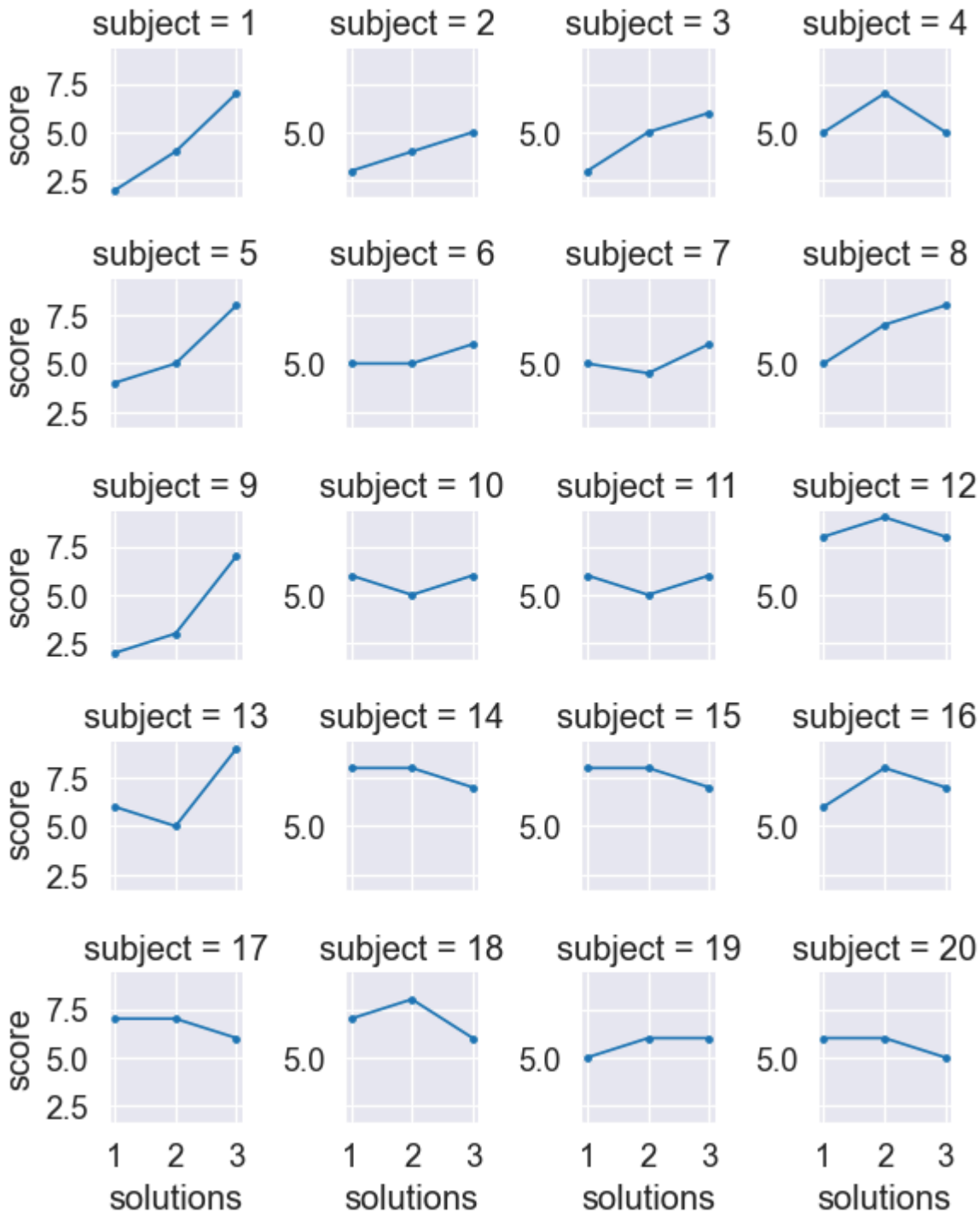
In [30]:
```
kws = dict(s=50, linewidth=.5, edgecolor='w')
tips_fg_3 = sns.FacetGrid(tips, col='sex', hue='smoker', height=4, aspect=1.5,
                          hue_order=['Yes', 'No'], hue_kws=dict(marker=['^', 'v']))
tips_fg_3.map(plt.scatter, 'total_bill', 'tip', **kws)
```

Out[30]: <seaborn.axisgrid.FacetGrid at 0x1d725e08340>



In [31]:
```
attention_fg = sns.FacetGrid(attention, col='subject', col_wrap=4, height=1.5)
attention_fg.map(plt.plot, 'solutions', 'score', marker='.')
```

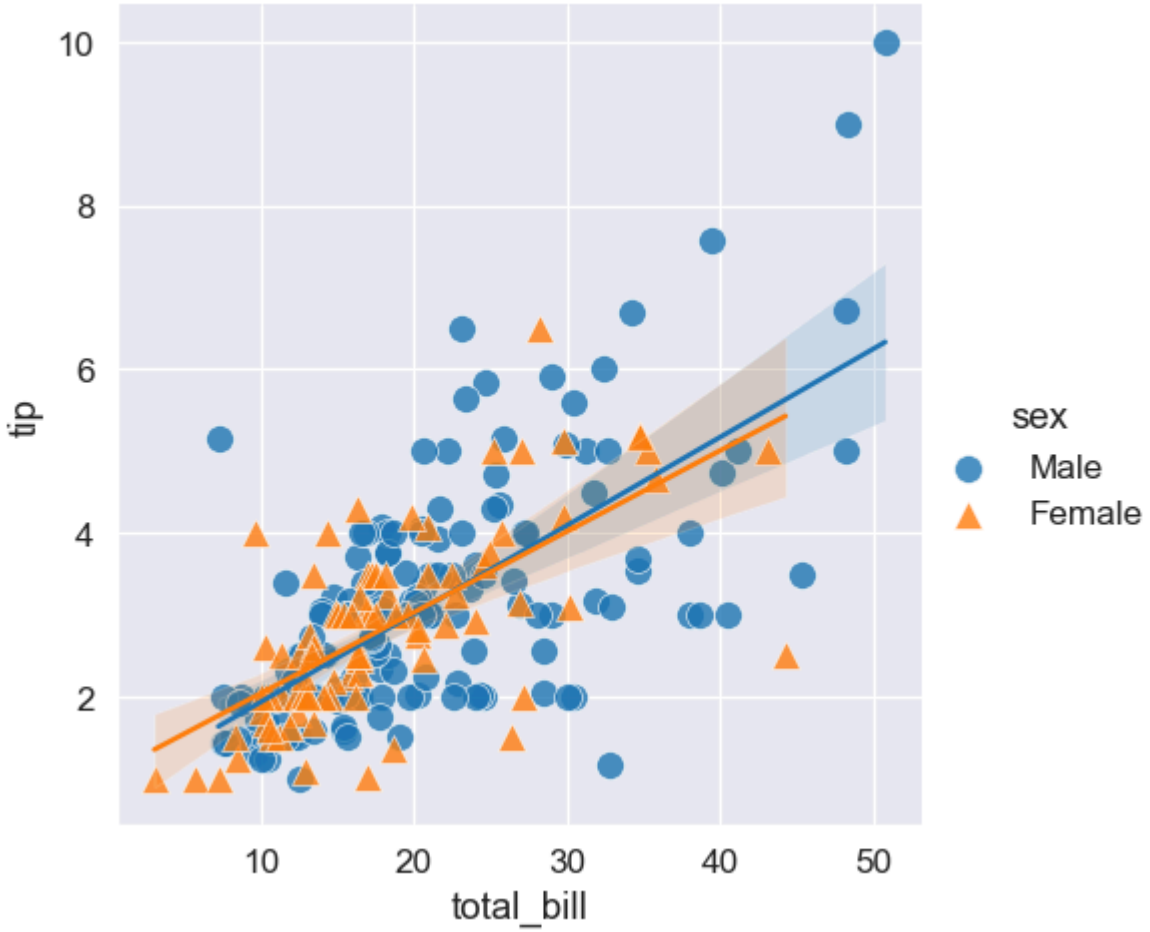
Out[31]: <seaborn.axisgrid.FacetGrid at 0x1d7292bf400>



Regression Plots

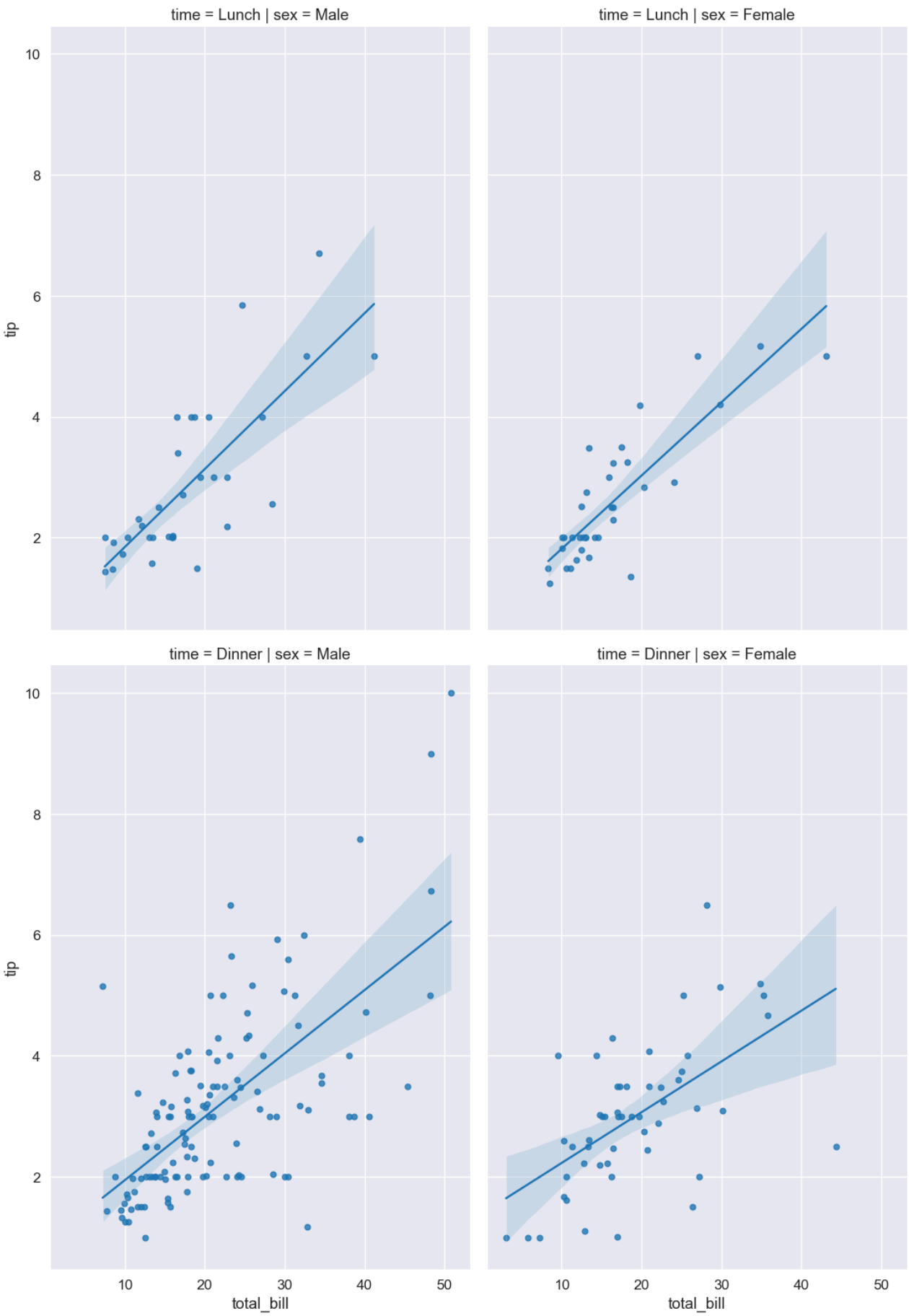
In [32]:
```
plt.figure(figsize=(8, 6))
sns.set_context('paper', font_scale=1.4)
sns.lmplot(x='total_bill', y='tip', data=tips, hue='sex', markers=['o', '^'],
           scatter_kws={'s': 100, 'linewidths': 0.5, 'edgecolor': 'w'})
```

Out[32]:
```
<seaborn.axisgrid.FacetGrid at 0x1d725f86e90>
<Figure size 800x600 with 0 Axes>
```



In [33]:
```
sns.lmplot(x='total_bill', y='tip', col='sex', row='time', data=tips,
                  height=8, aspect=0.7,)
```

Out[33]:
```
<seaborn.axisgrid.FacetGrid at 0x1d725fc9330>
```


time = Lunch | sex = Male
time = Lunch | sex = Female
time = Dinner | sex = Male
time = Dinner | sex = Female
tip
total_bill
2
4
6
8
10
10
20
30
40
50