

## 07 - Seaborn

### Docupedia Export

Author:Lima Queila (CtP/ETS)

Date:24-Feb-2026 18:08

## Table of Contents

### 1 TREINAMENTO DE PYTHON (SEABORN)

**4**

1.1 Plotando dados categóricos

15

1.2 Regressão linear

28

1.3 Desafio

32



## 1

## TREINAMENTO DE PYTHON (SEABORN)



Seaborn é uma biblioteca para geração de gráficos estatísticos em Python. Tem como base o Matplotlib e possui forte integração com as estruturas de dados do pandas.

O grande objetivo do seaborn é facilitar a visualização e entendimento de dados. Serão abordados:

- Visualizando relações estatísticas.
- Plotando com categorias de dados.
- Visualizando a distribuição de um conjunto de dados.
- Visualizando relações lineares.
- Construindo redes multi-plot estruturadas.

O modo mais comum de chamar a biblioteca é usando o alias **sns**.

```
import seaborn as sns
```

Análise estatística é um processo onde procuramos entender como variáveis em um conjunto de dados se relacionam e como esses relacionamentos dependem de outras variáveis.

Na biblioteca Seaborn utiliza-se a função **relplot()** para explorar estes relacionamentos. Dois tipos de gráficos são usados: dispersão de pontos (scatterplot) e de linhas (lineplot).

Para a aula de hoje estaremos usando o dataset **tips.csv**. Ele consiste em dados de um restaurante, contendo o valor de uma compra (*total\_bill*), o valor da gorjeta (*tip*), informações sobre o cliente (*sex*, *smoker*), o dia e horário da compra (*day*, *time*) e a quantidade de pratos pedidos (*servings*).

```
import pandas as pd
```

```
tips = pd.read_csv('data/tips.csv', sep=';')
tips.head()
```

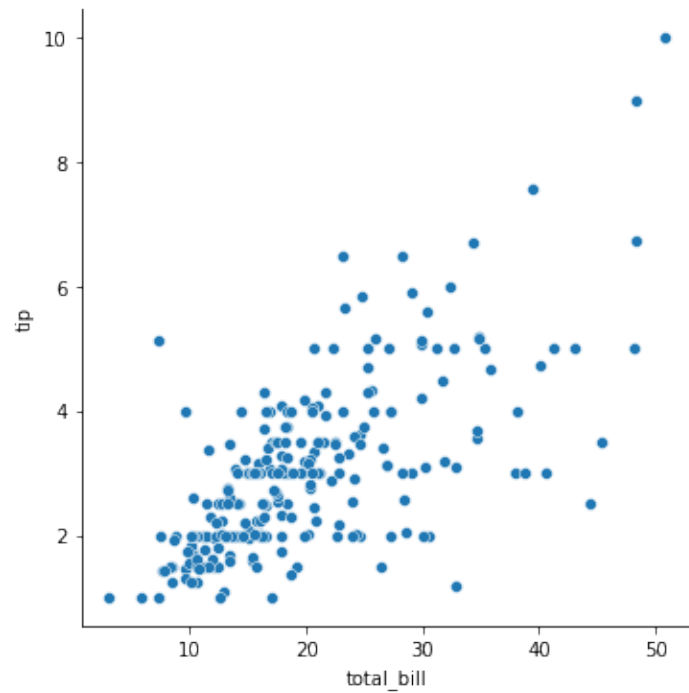
	total_bill	tip	sex	smoker	day	time	servings
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

```
tips.shape
```

(244, 7)

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

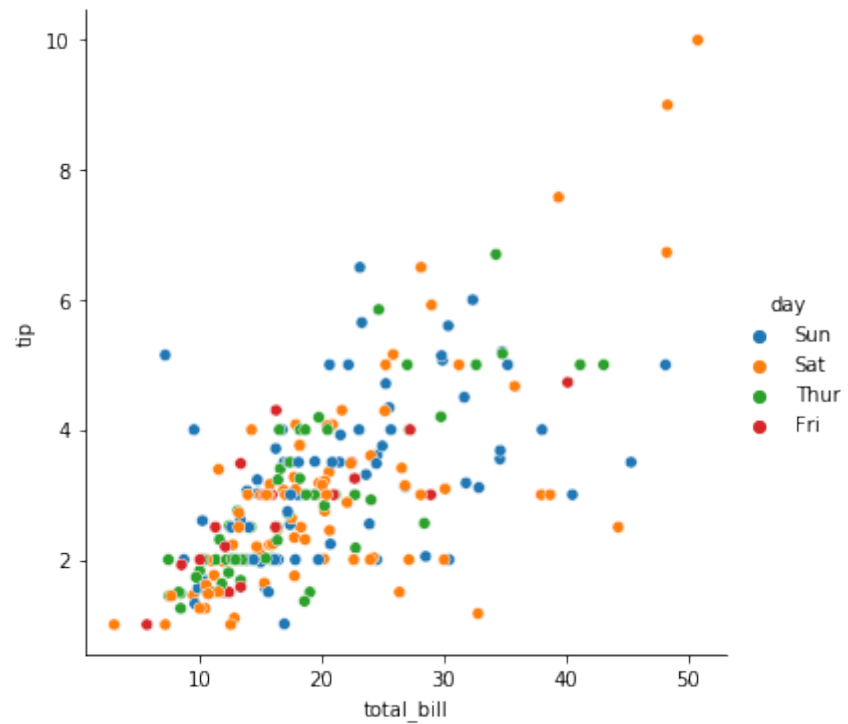
<seaborn.axisgrid.FacetGrid at 0x228c88258c8>



Os pontos são plotados em duas dimensões (x, y), mas podemos adicionar outra dimensão ao mudar a cor dos pontos de acordo com uma terceira variável. Para isso utilizamos o parâmetro `hue`. Nela passamos o nome da variável que queremos mudar. `hue="nome_da_variável"`.

```
sns.relplot(x='total_bill', y='tip', hue = 'day', data = tips)
```

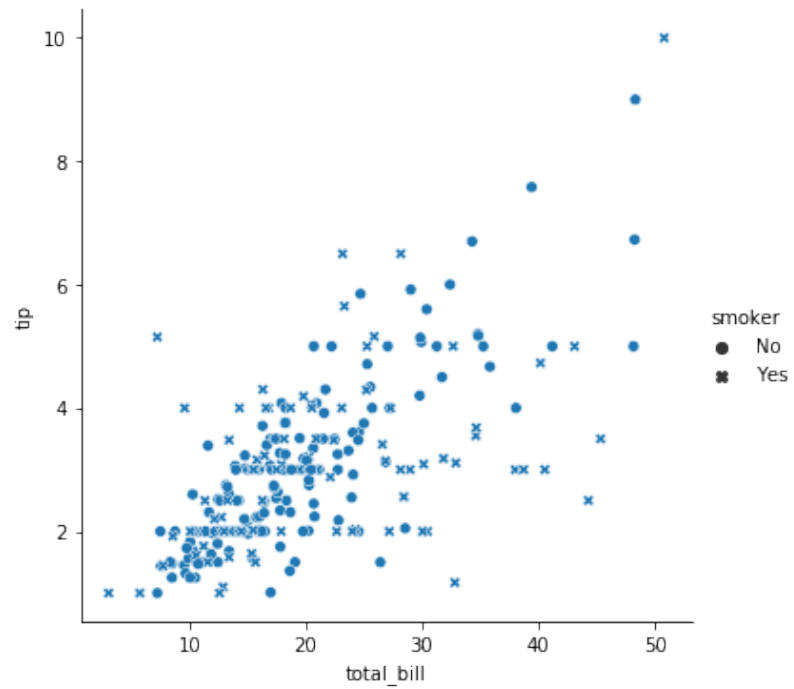
<seaborn.axisgrid.FacetGrid at 0x1cac8c9b608>



É possível também alterar o estilo e o tamanho dos pontos.

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

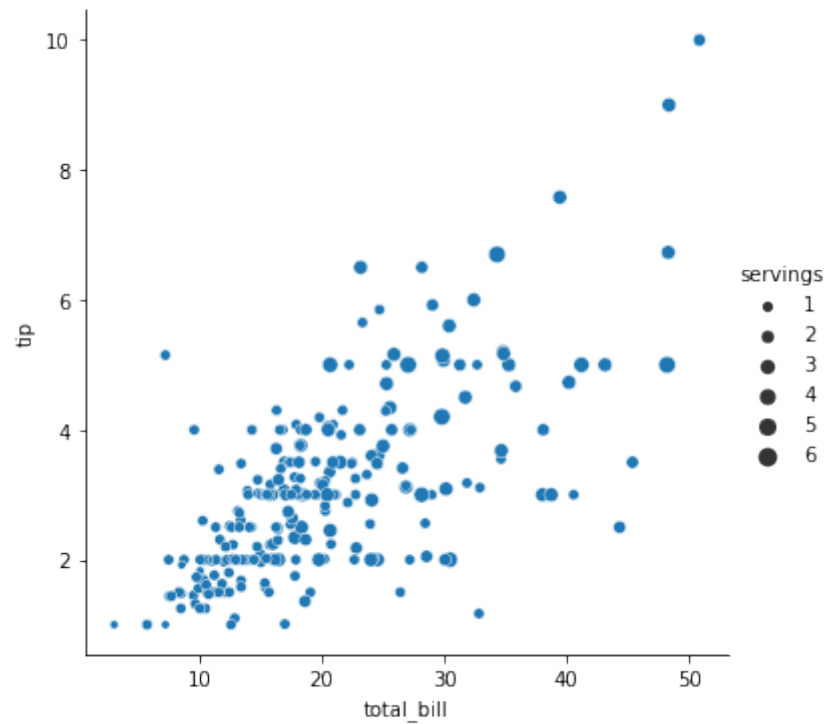
<seaborn.axisgrid.FacetGrid at 0x1cac8c8afc8>



```
sns.relplot(x='total_bill', y='tip', size = 'servings', data = tips)
```

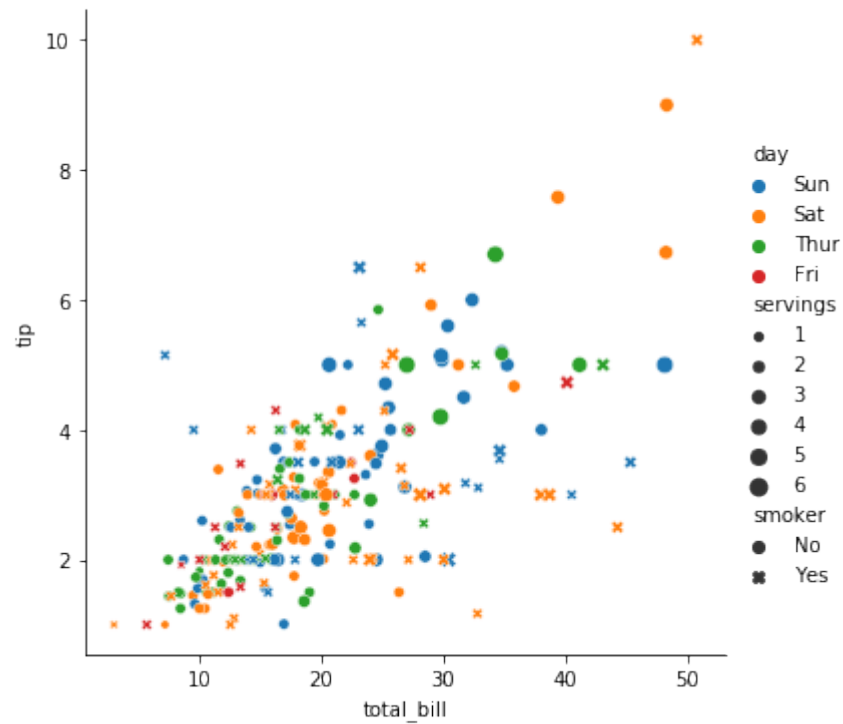
<seaborn.axisgrid.FacetGrid at 0x1cac8d78bc8>





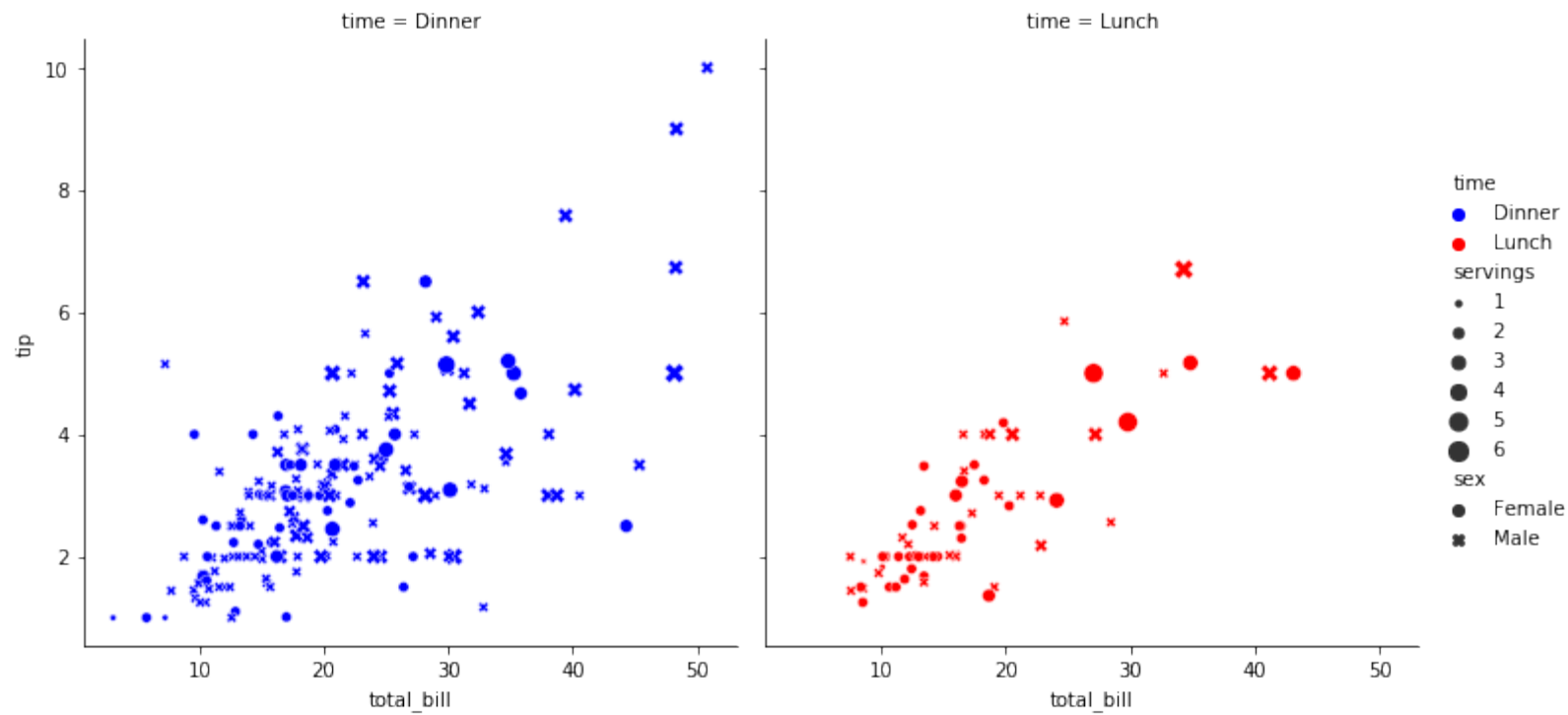
```
sns.relplot(x='total_bill', y='tip', style = 'smoker', hue = 'day', size = 'servings', data = tips)
```

```
<seaborn.axisgrid.FacetGrid at 0x1cac8d7e388>
```



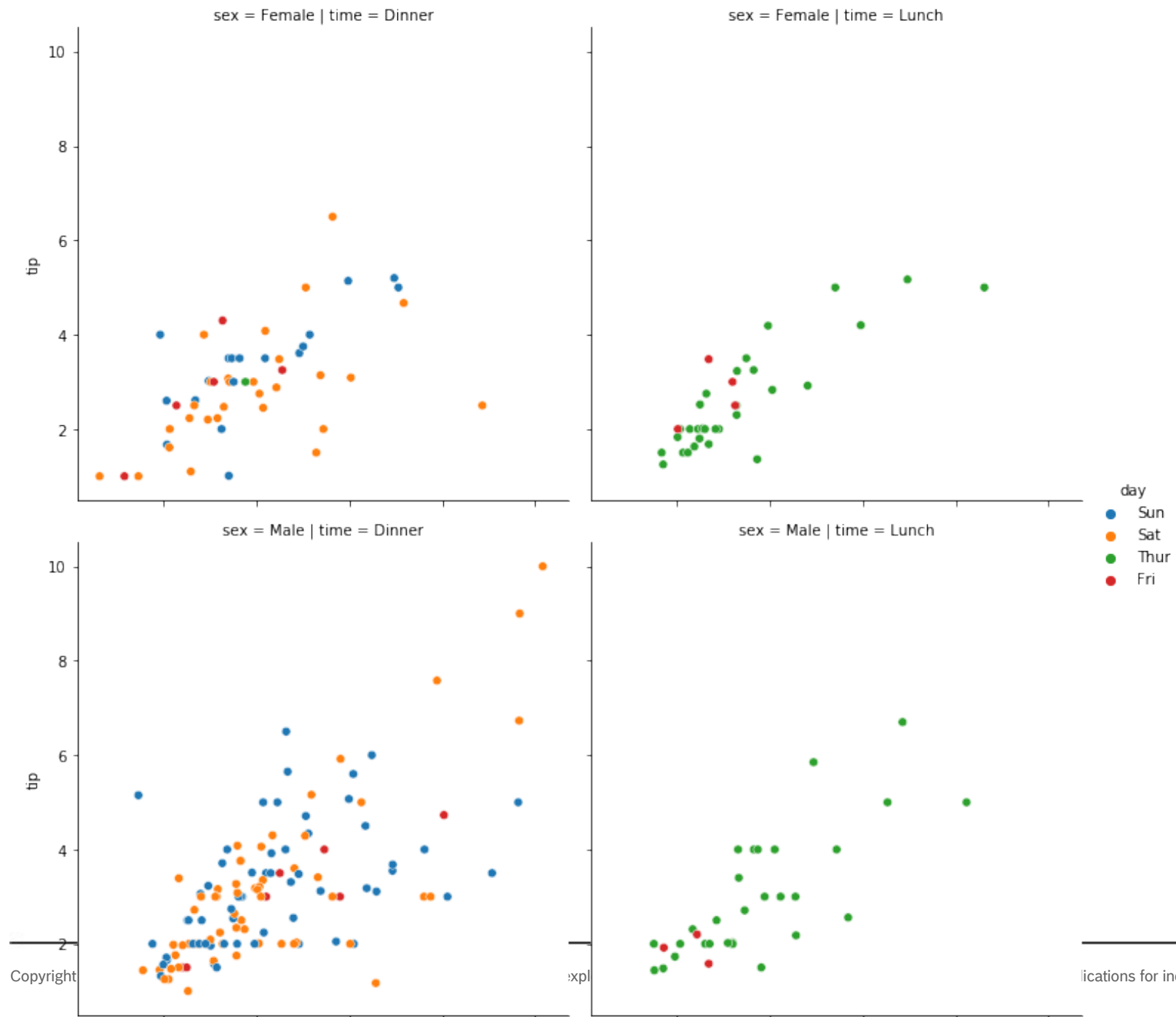
Podemos separar em colunas e linhas, de acordo com um variável:

```
sns.relplot(x="total_bill", y="tip", col="time",  
            hue="time", size="servings", style="sex",  
            palette=["b", "r"], sizes=(10, 100), data=tips)
```



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

<seaborn.axisgrid.FacetGrid at 0x228ca2a8888>

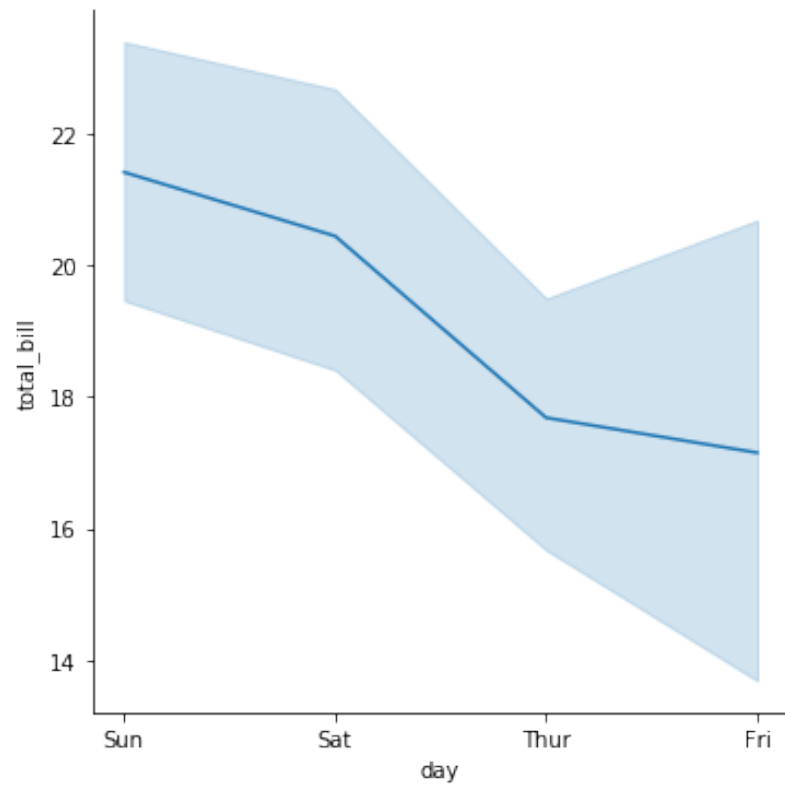


Em alguns conjuntos de dados é interessante entender as mudanças de uma variável ao longo do tempo, ou alguma outra variável contínua. Em situações assim, uma boa escolha é enfatizar a continuidade usando gráficos de linhas.

```
k = ['Thur', 'Fri', 'Sat', 'Sun']

dayWeek = sorted(tips.day, key=k.index) # para o gráfico ficar na ordem certa
                                           # caso contrário, a ordem é a em que os itens aparecem no dataset
sns.relplot(x='day', y='total_bill', data=tips, kind='line')
```

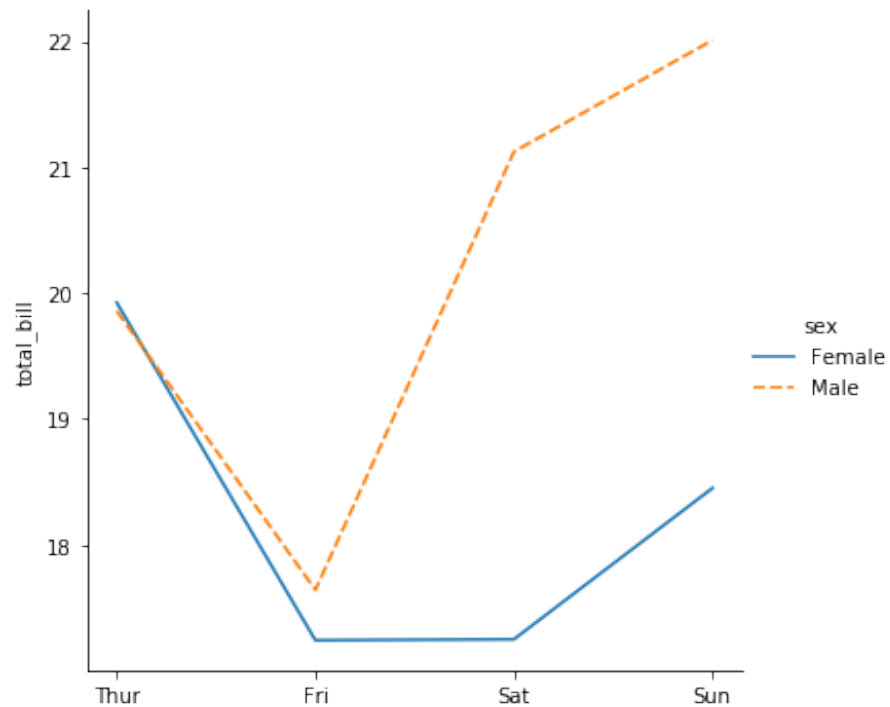
<seaborn.axisgrid.FacetGrid at 0x1726f5f45c8>



Podemos adicionar outras variáveis, assim como no gráfico de dispersão.

```
sns.relplot(x=dayWeek,y='total_bill',data=tips,kind='line',hue='sex',style='sex')
```

<seaborn.axisgrid.FacetGrid at 0x228cae03d08>



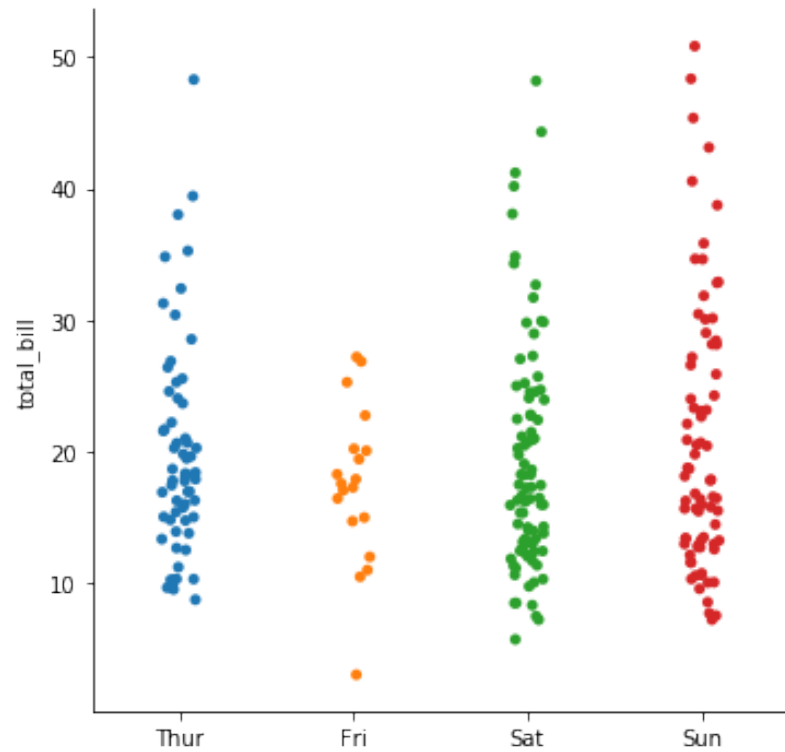
## 1.1 Plotando dados categóricos

Em estatística, uma variável categórica é uma variável que pode assumir apenas um número limitado, e geralmente fixo, de valores possíveis, atribuindo cada indivíduo ou outra unidade de observação a um determinado grupo ou categoria nominal com base em alguma propriedade qualitativa. Existem diversos tipos de gráficos possíveis, cada um se encaixa melhor para certas aplicações. No seaborn utilizaremos a função **catplot**, o tipo de gráfico é setado pelo argumento "kind".

O padrão (se nada for passado ao kind) é o strip plot. Mudando o argumento **kind** mudamos o tipo de gráfico categórico.

```
sns.catplot(x=dayWeek, y='total_bill', data=tips)
```

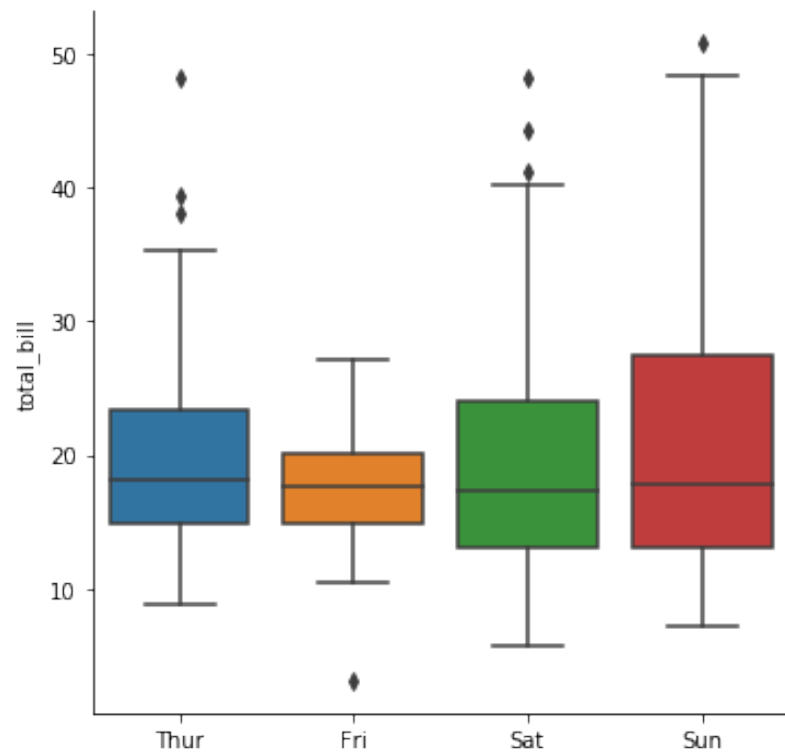
<seaborn.axisgrid.FacetGrid at 0x228ca99da08>



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

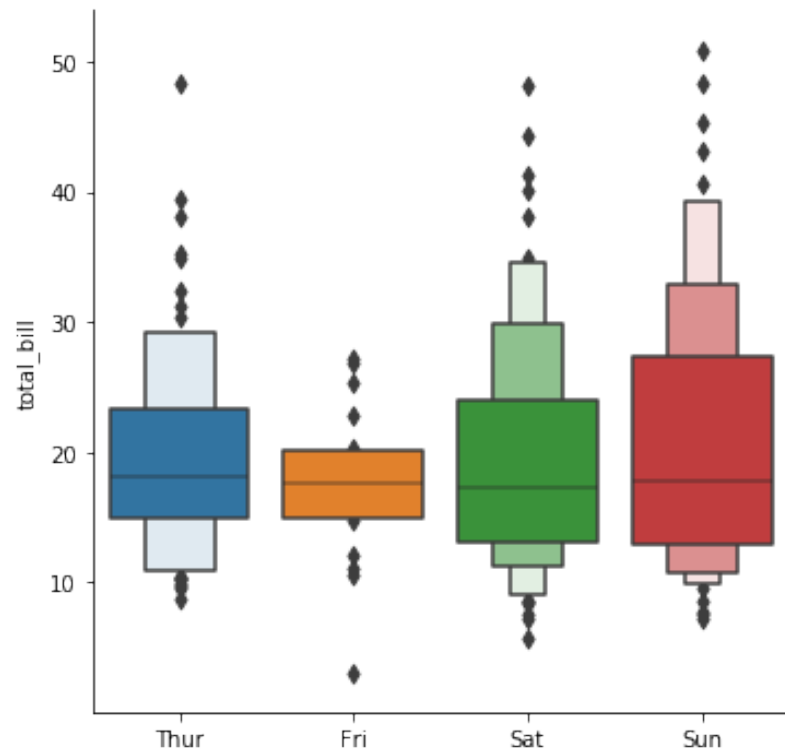
<seaborn.axisgrid.FacetGrid at 0x228caf08a88>





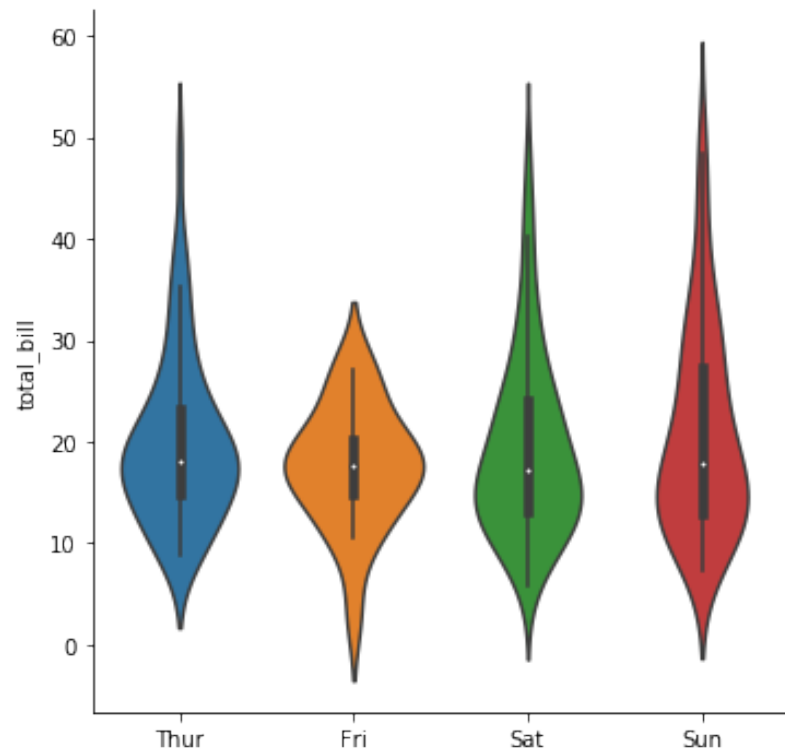
```
sns.catplot(x=dayWeek, y='total_bill', data=tips, kind='boxen')
```

<seaborn.axisgrid.FacetGrid at 0x228caf89d88>



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

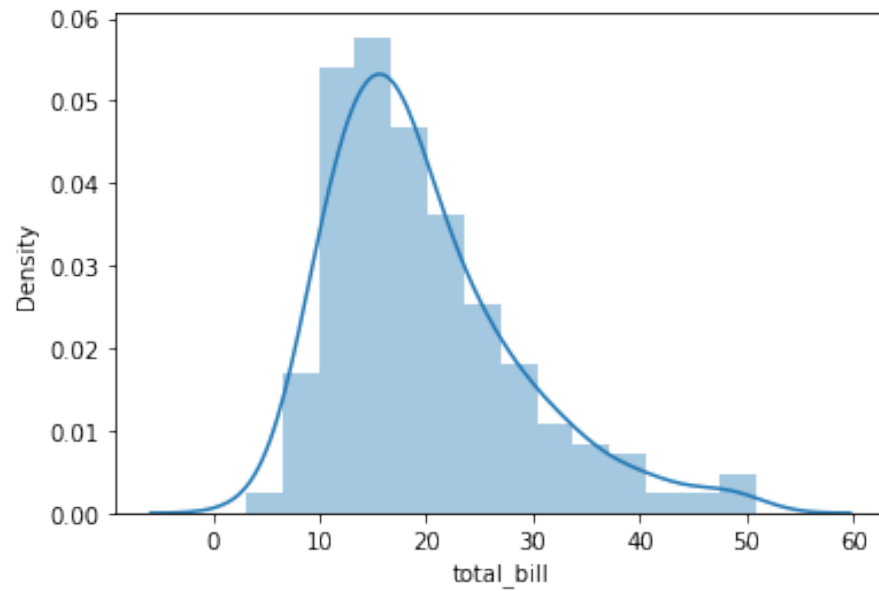
<seaborn.axisgrid.FacetGrid at 0x228cbfe2ac8>



Ao lidar com dados, é interessante ter uma ideia de como as variáveis estão distribuídas. A forma mais fácil de se fazer isso, é utilizando a função `distplot()`. Serão plotados um histograma e uma curva chamada Kernel Density Estimation, KDE.

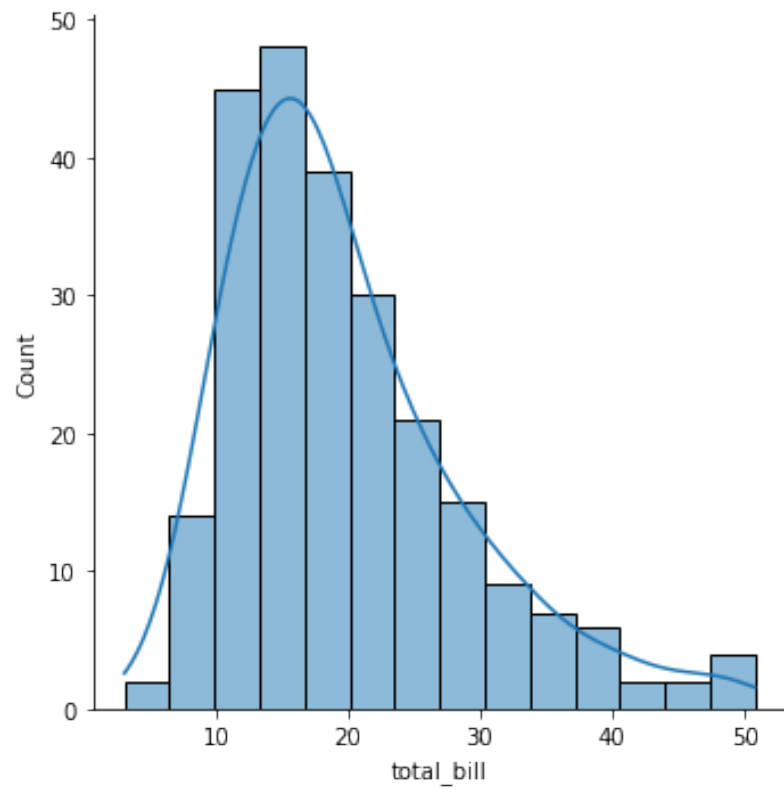
```
sns.distplot(tips['total_bill'])
```

<matplotlib.axes.\_subplots.AxesSubplot at 0x1cacc924cc8>



```
sns.displot(tips['total_bill'], kde=True)
```

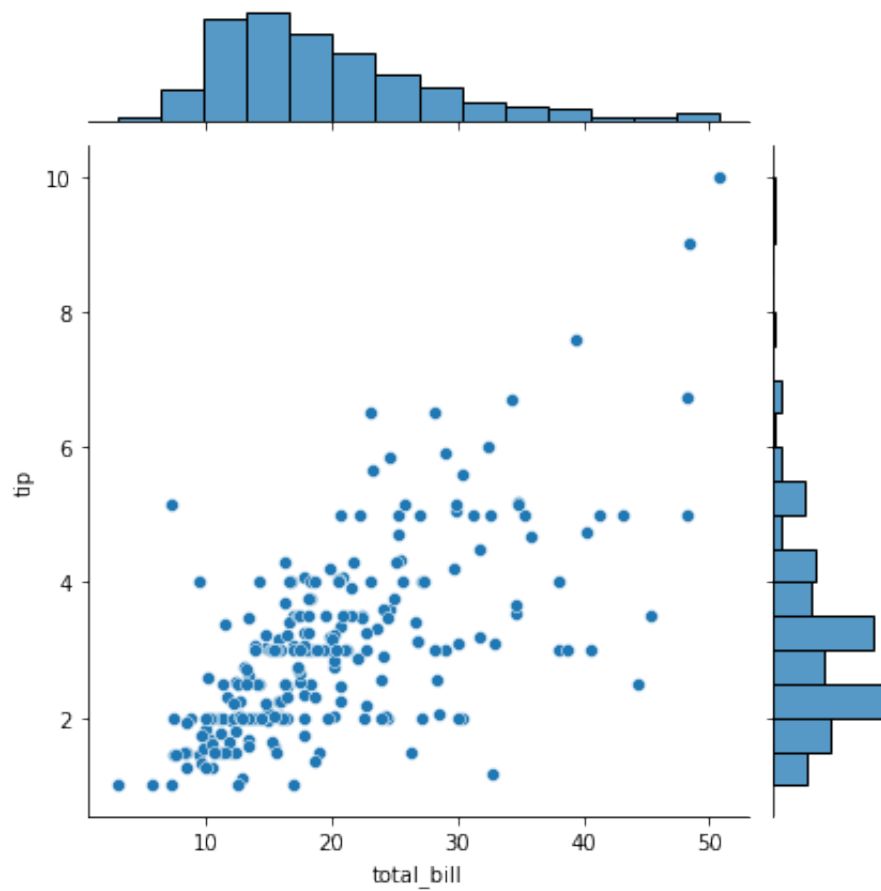
<seaborn.axisgrid.FacetGrid at 0x1cacc6de688>



Para visualizar a distribuição de duas variáveis, a função `jointplot()` cria um painel para mostrar o relacionamento conjunto.

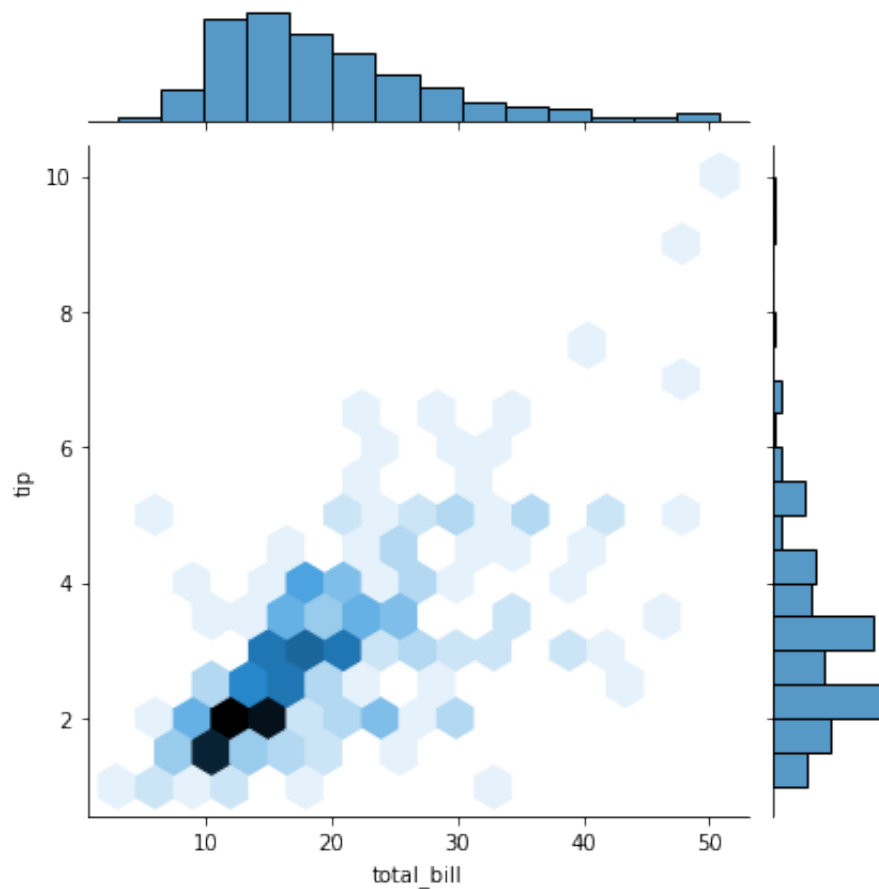
```
sns.jointplot(x='total_bill',y='tip',data=tips)
```

<seaborn.axisgrid.JointGrid at 0x1cacc01b388>



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

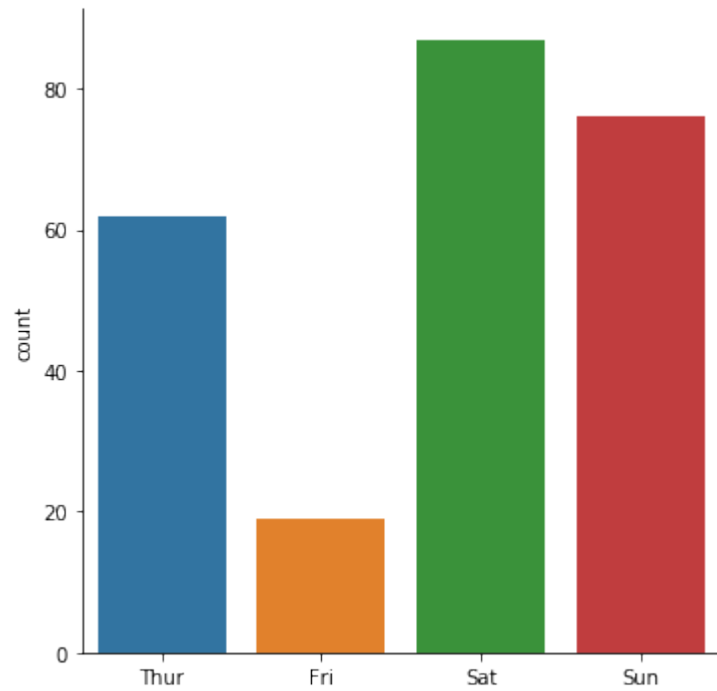
<seaborn.axisgrid.JointGrid at 0x1cacc160bc8>



O count é parecido com o histograma, mas com variáveis categóricas, ele mostra a frequência de cada valor categórico dentro do dataset, podendo mostrar relações com outras variáveis.

```
sns.catplot(x=dayWeek, data=tips, kind='count')
```

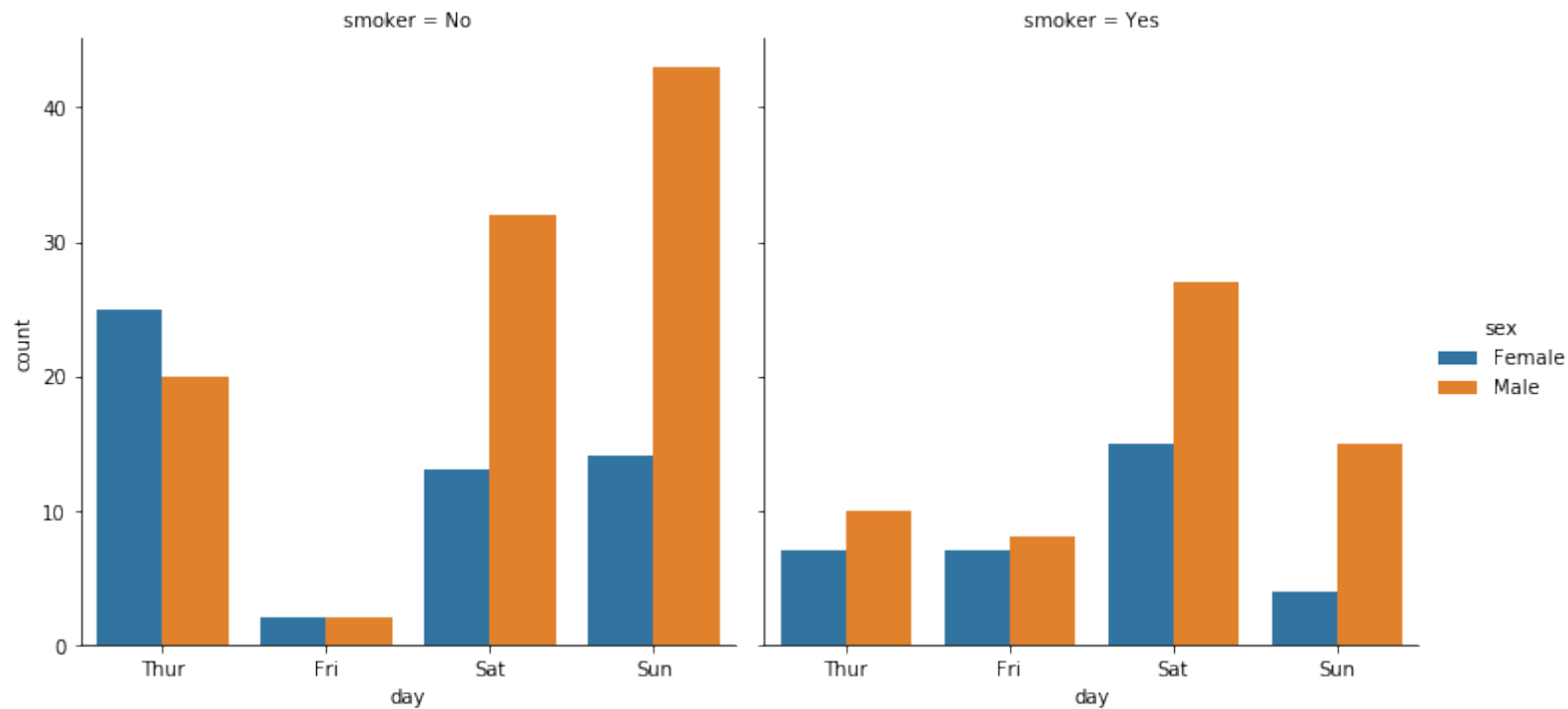
<seaborn.axisgrid.FacetGrid at 0x228cc3dafc8>



```
sns.catplot(x='day', hue='sex', data=tips, col='smoker', kind='count', order=k)
```



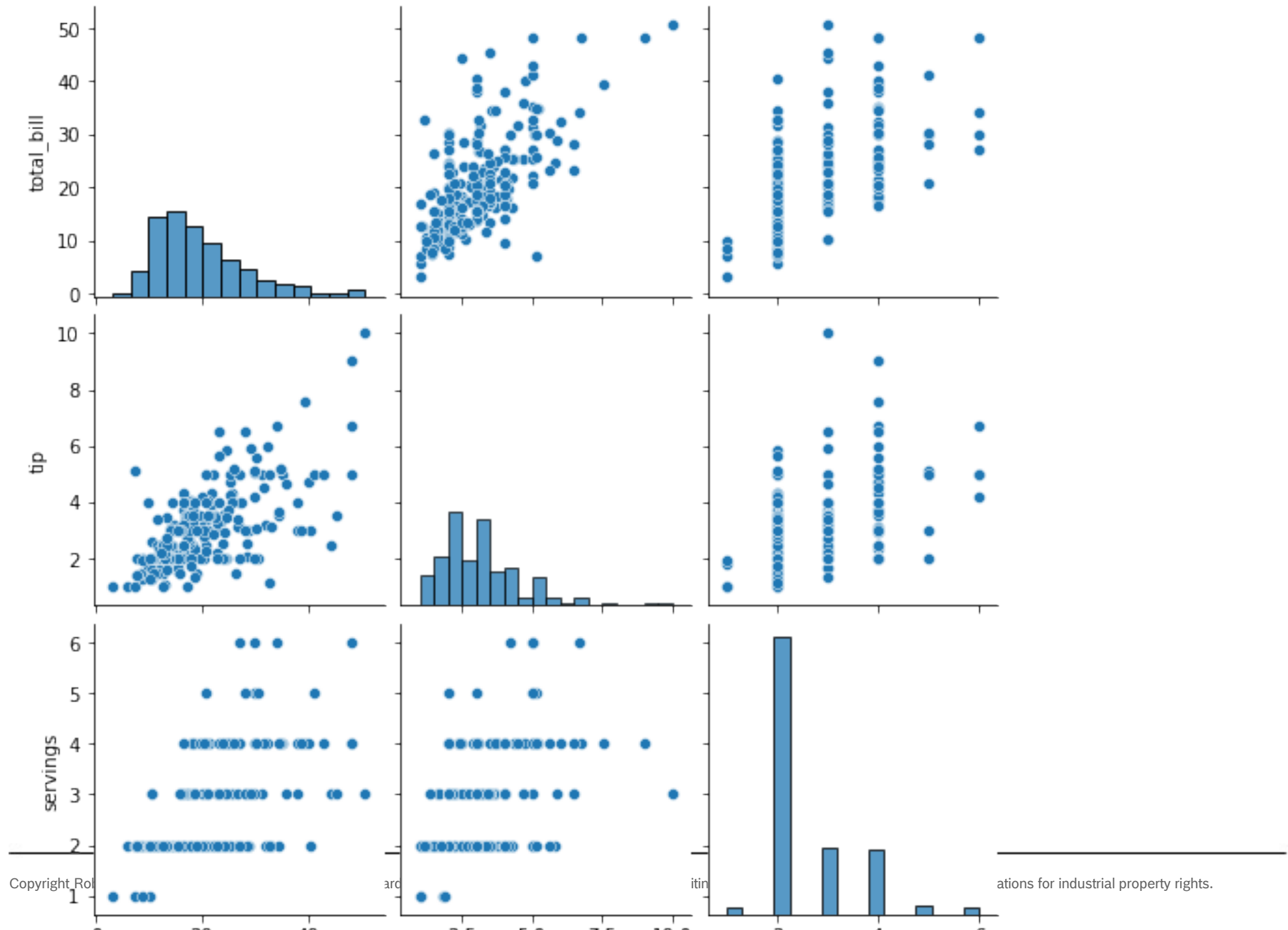
<seaborn.axisgrid.FacetGrid at 0x1726f45a4c8>



Para visualizar a correlação entre todas as variáveis do dataset, podemos utilizar a função `pairplot` do seaborn, ela mostrará a correlação entre variáveis em gráficos de dispersão, além dos histogramas mostrando as distribuições de cada variável.

```
sns.pairplot(tips)
```

<seaborn.axisgrid.PairGrid at 0x1cacdaf8948>



## 1.2

## Regressão linear

A análise de regressão estuda a relação entre uma variável chamada dependente e outras variáveis chamadas independentes.

A relação pode ser representada por um modelo matemático que associa as variáveis entre si. Caso o modelo defina uma relação linear entre a variável dependente e uma variável independente, este é então chamado de modelo de regressão linear simples.



Existência de correlação  
positiva (em média, quanto maior for  
a altura maior será o peso)

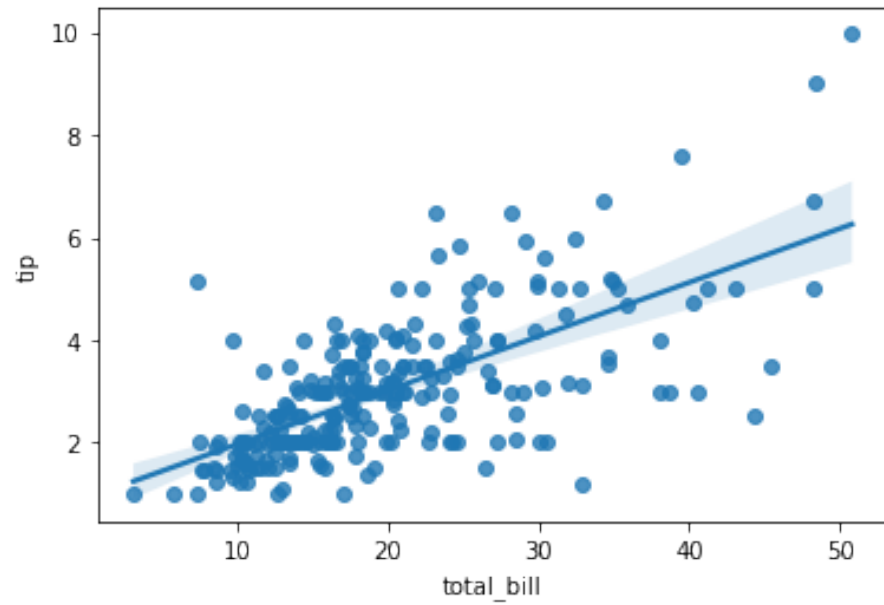


Existência de correlação  
negativa (em média, quanto maior for  
for a colheita menor será o preço)

Muitos conjuntos de dados contém múltiplas variáveis quantitativas e frequentemente o objetivo de uma análise é relacionar as variáveis umas com as outras. Assim, podemos realizar uma regressão linear para melhor visualização. As funções `regplot()` e `lmplot()` realizam esta tarefa.

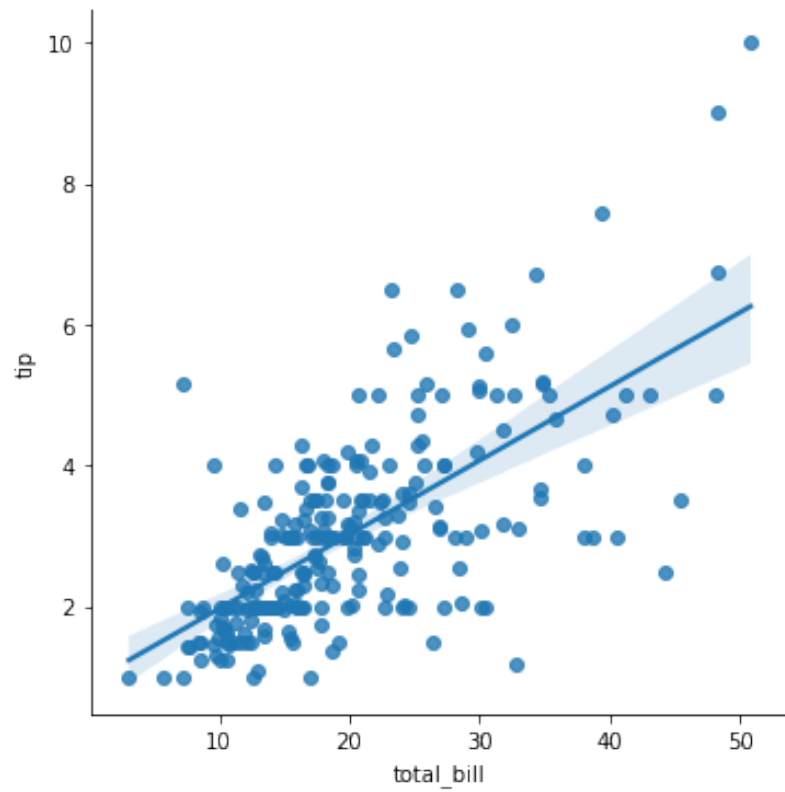
```
sns.regplot(x='total_bill',y='tip',data=tips)
```

<matplotlib.axes.\_subplots.AxesSubplot at 0x1cacdfc9508>

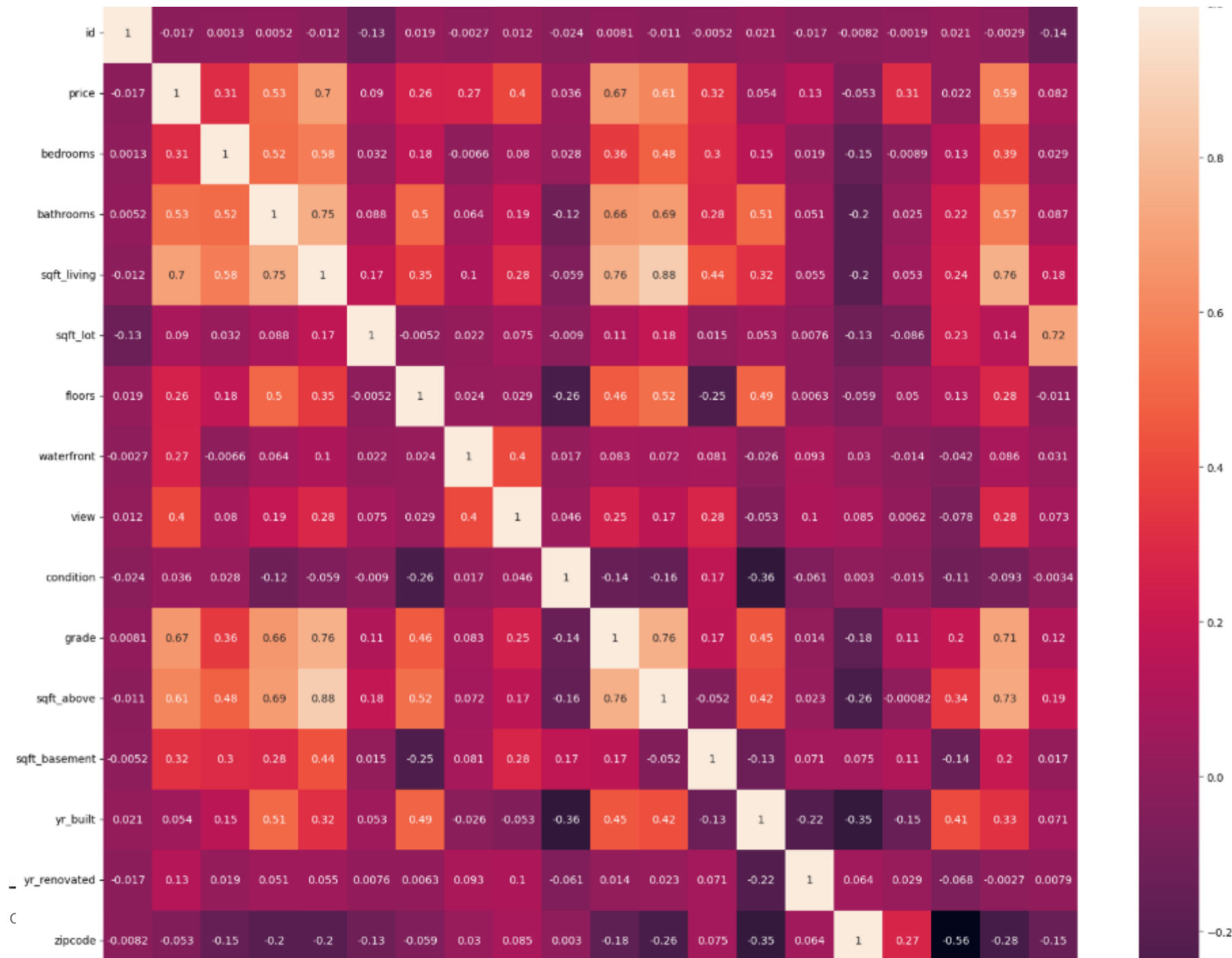


```
sns.lmplot(x='total_bill',y='tip',data=tips)
```

<seaborn.axisgrid.FacetGrid at 0x1cace2010c8>



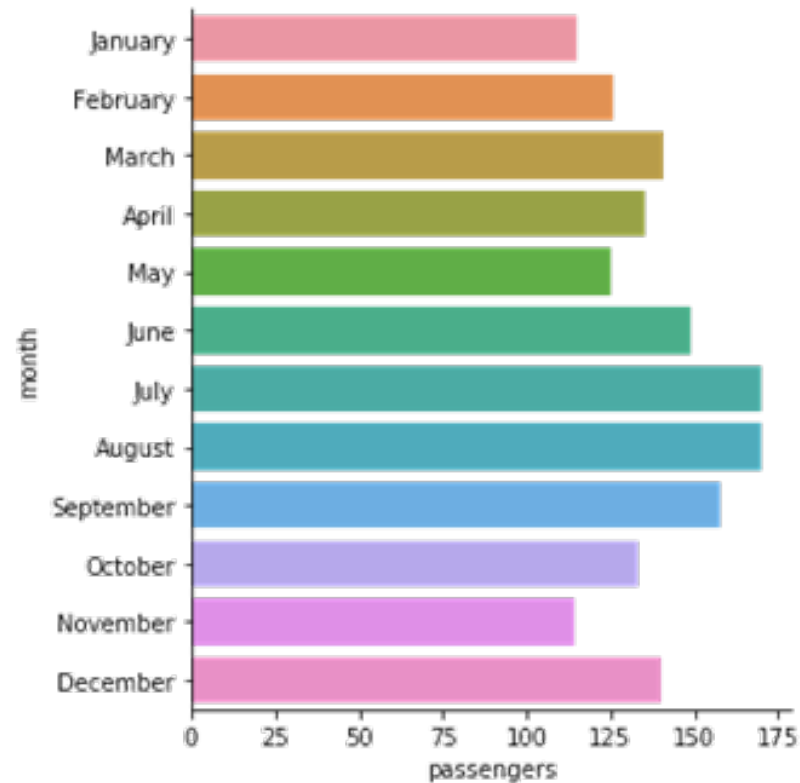
```
base_casas = pd.read_csv('house_prices.csv')
base_casas.drop('date', axis=1, inplace = True)
figura = plt.figure(figsize=(20,20))
sns.heatmap(base_casas.corr(), annot=True);
```



## 1.3

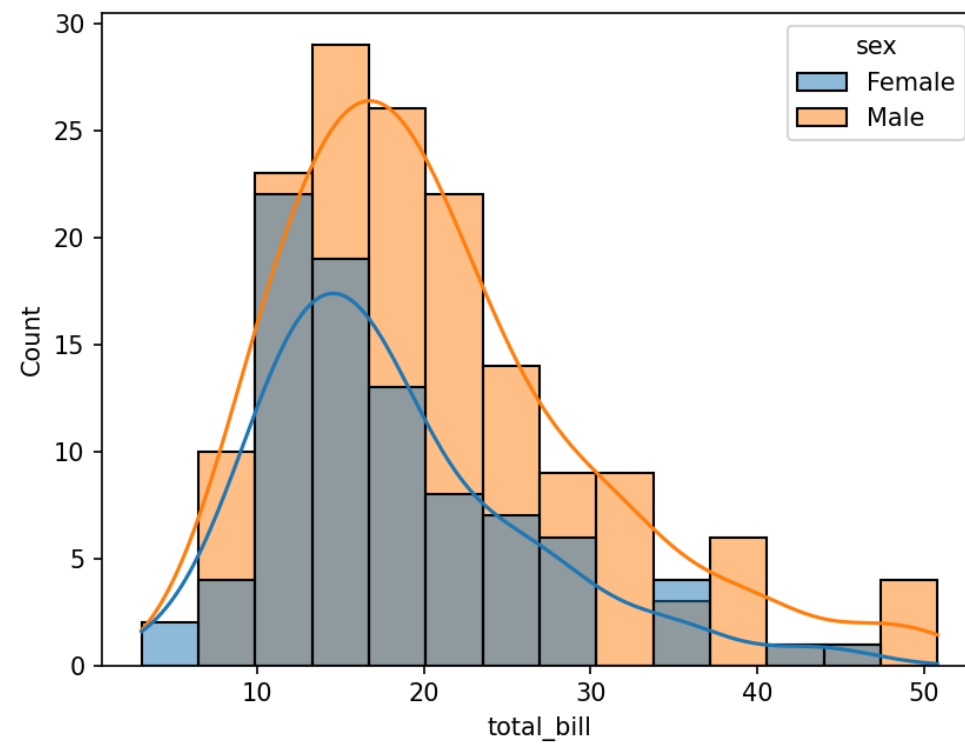
### Desafio

1. A partir da planilha “flights.csv”, plote um gráfico através da biblioteca seaborn do tipo barras horizontais, em que apresente os meses (eixo Y) e a quantidade de passageiro em cada mês do ano de 1950 (eixo X). A paleta de cores é opcional.

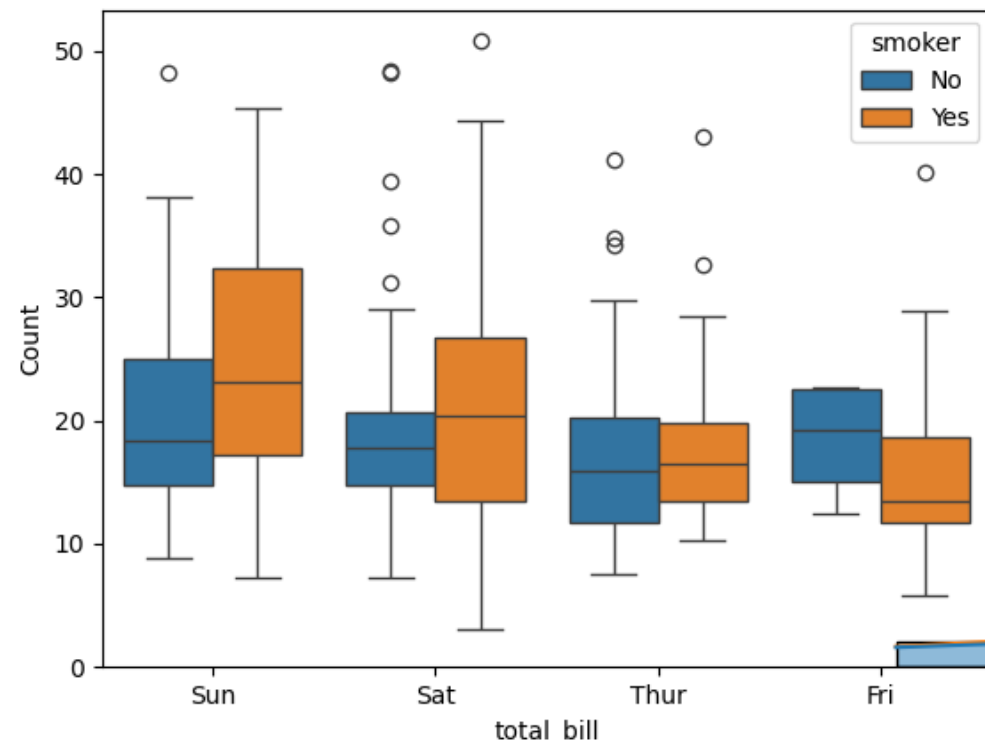


2. Crie um histograma (`histplot`) da variável `total_bill`, separe por `sex`, adicione KDE. Por fim interprete: Existe diferença entre os gastos de homens e mulheres?

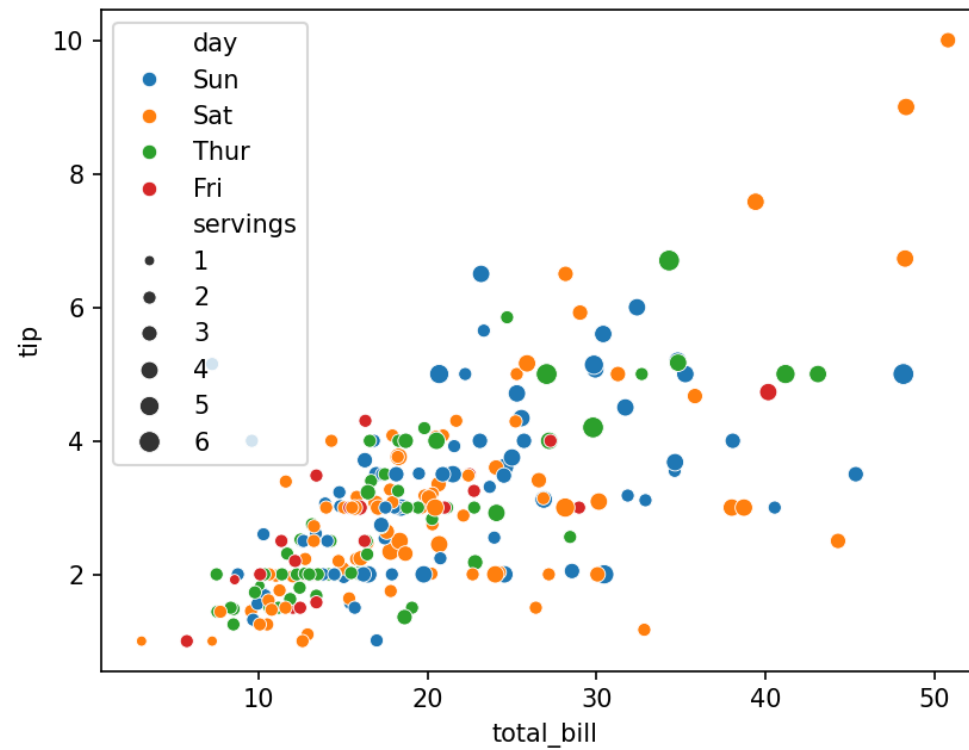




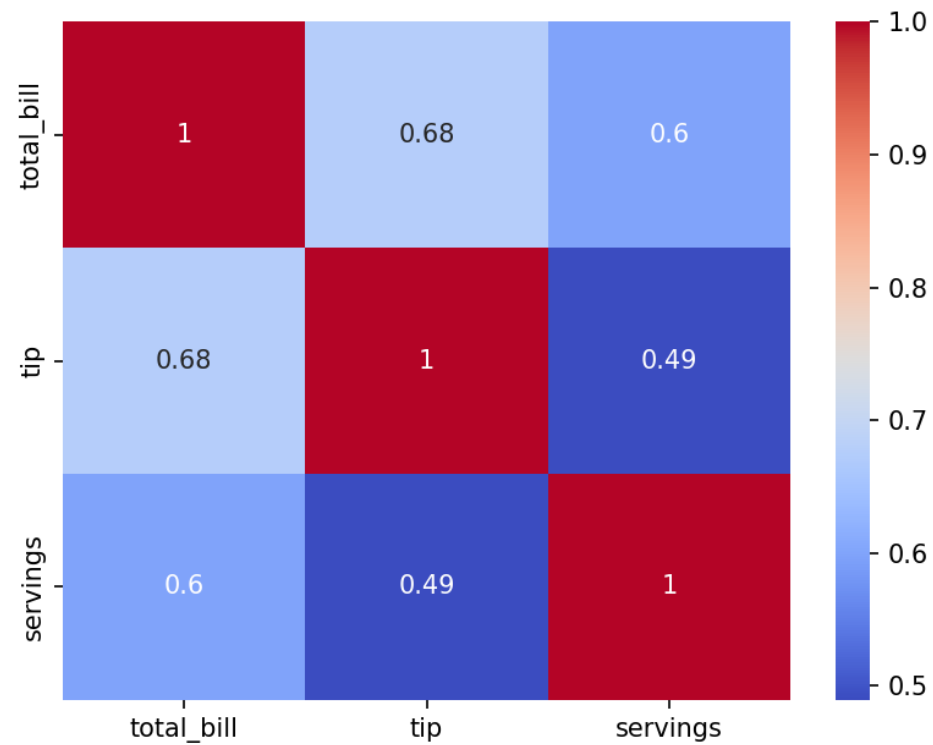
3. Com o dataset `tips`, crie um `boxplot` de `total_bill` por `day`, adicione `hue="smoker"`. Identifique: Qual dia tem mais outliers? Fumantes gastam mais?



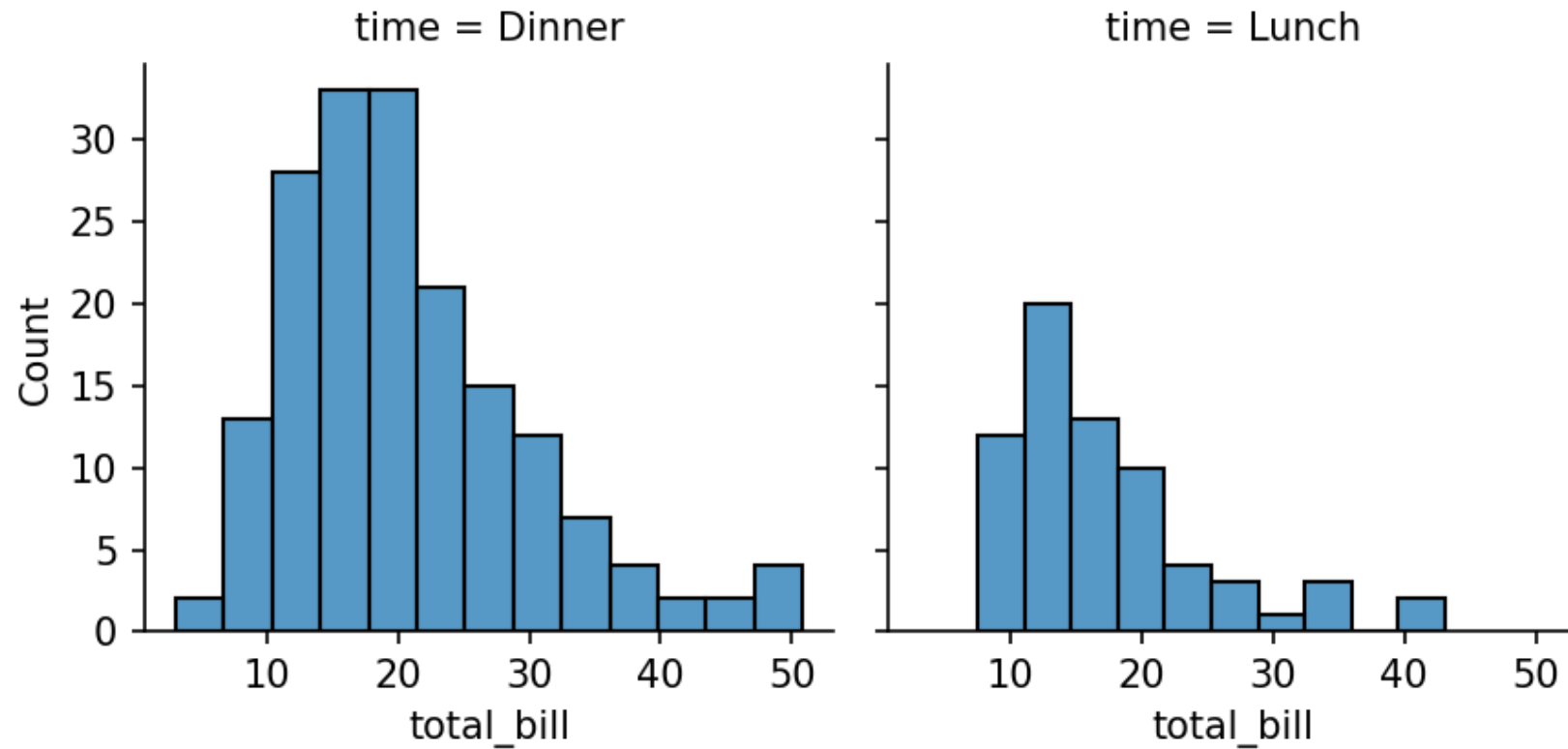
4. Com `tips`, crie um `scatterplot` de `total_bill` vs `tip`, mude a cor de acordo com os dias e o tamanho de acordo com a quantidade de pratos. Responda: Existe correlação? Grupos maiores dão mais gorjeta proporcionalmente?



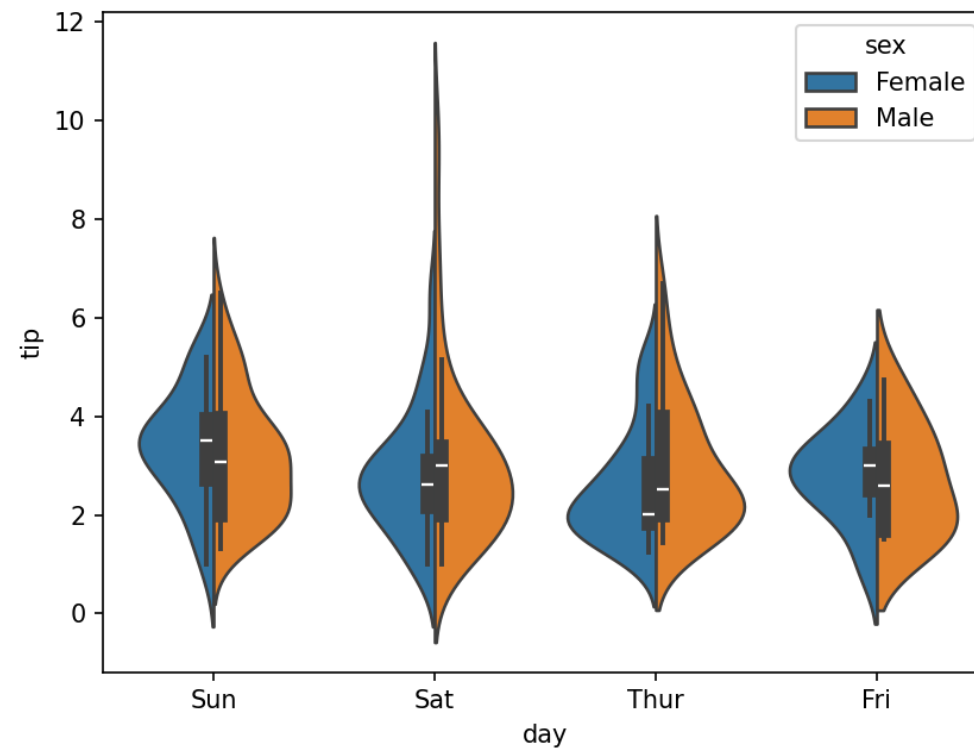
5. Com o dataset "tips.csv" calcule a matriz de correlação, plote com `heatmap` (com `annot=True`). Responda: Qual variável mais influencia a gorjeta?



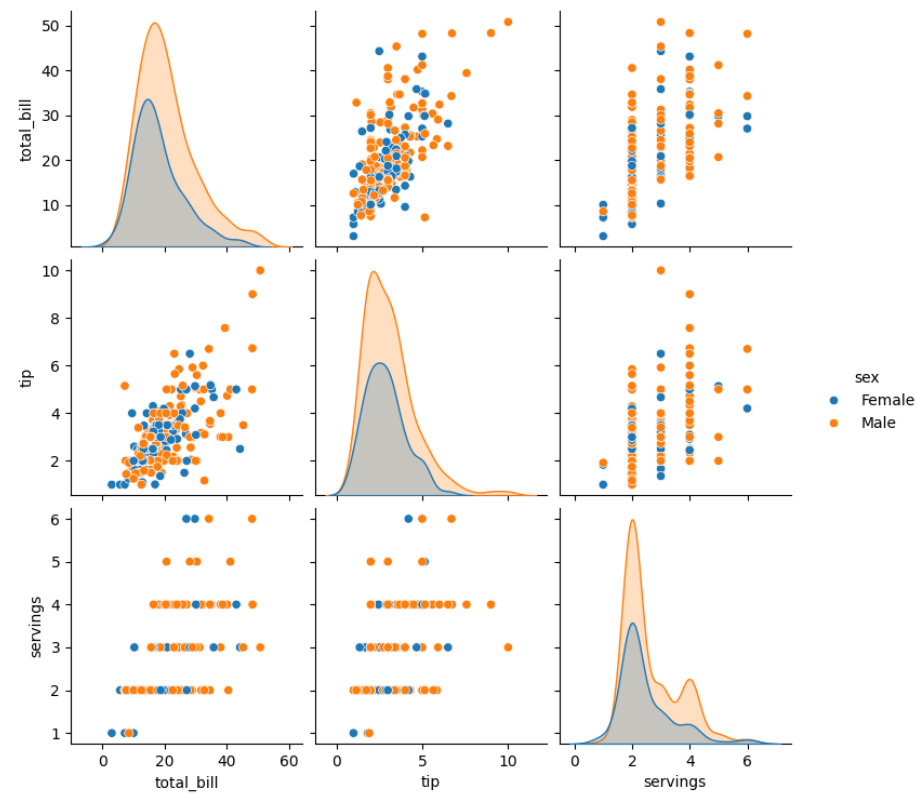
6. Use `FacetGrid`, crie gráficos separados por `time` (Lunch/Dinner), dentro de cada gráfico, plote `histplot` de `total_bill`. Interprete: Há diferença clara entre almoço e jantar?



7. Crie um `violinplot` de `tip` por `day`, diferencie o gênero pela cor. Responda: Em quais dias há maior variação nas gorjetas?



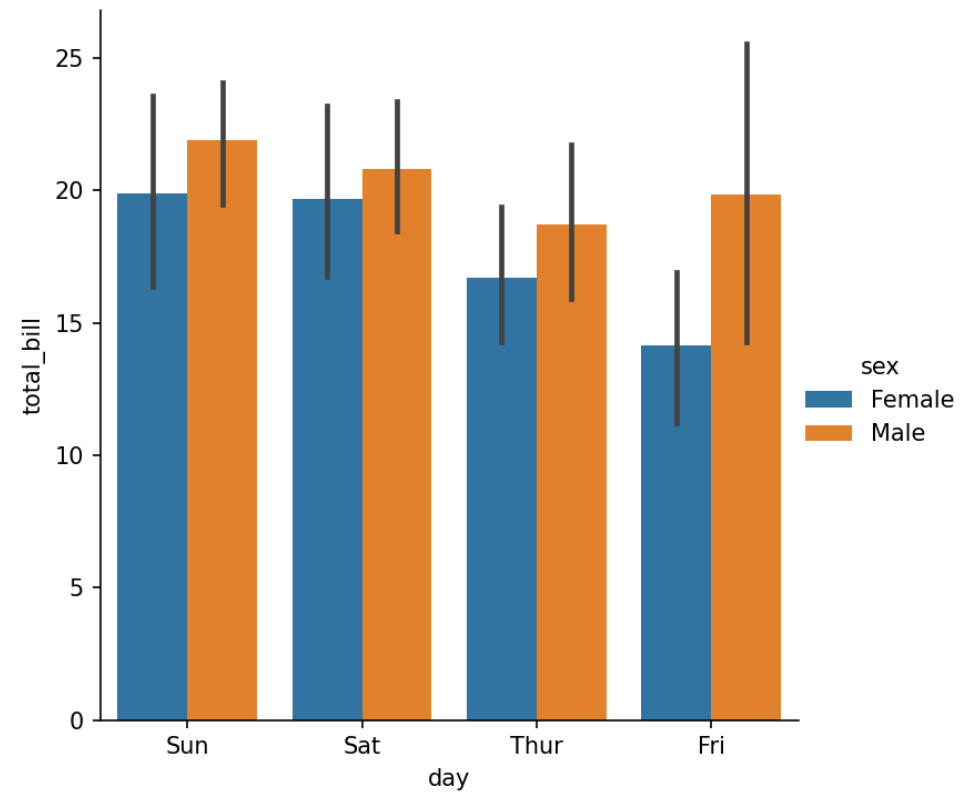
8. Use `pairplot` com `tips`, use `hue="sex"`. Analise: Quais variáveis parecem mais correlacionadas? Há clusters visíveis?



9. Crie um `catplot` :

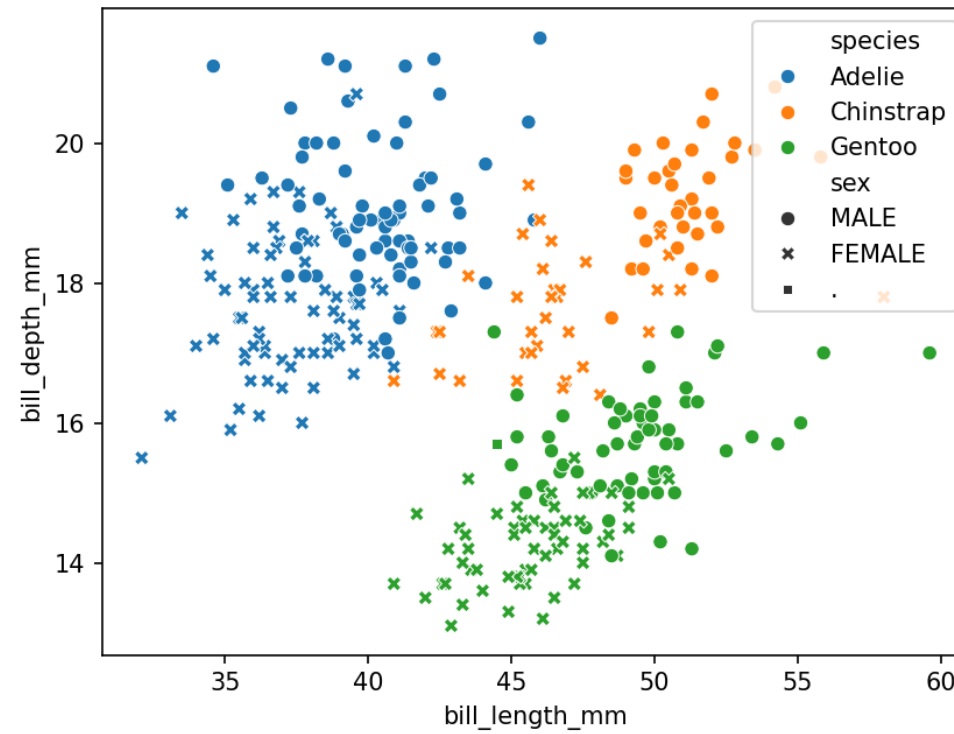
- Tipo: `bar`
- `x="day"`, `y="total_bill"`
- `hue="sex"`

Interprete: Quem gasta mais em média por dia?



10. Com o arquivo "penguins.csv", crie um gráfico como o da imagem abaixo. Interprete.





11. Escolha um dataset no Kaggle, faça a limpeza e explore os dados com gráficos (histograma, barras, scatter, boxplot e heatmap). Por fim, interprete os resultados e apresente insights relevantes.