

Introdução

Dados 🧩 são informações. Pense neles como peças de LEGO que, quando montadas corretamente, criam algo significativo. Pode ser qualquer coisa: números, palavras, medições ou observações. Nos dias de hoje, dados são super importantes, pois ajudam a tomar decisões em diversas áreas, como negócios, ciência e tecnologia.

O que é Python? 🐍

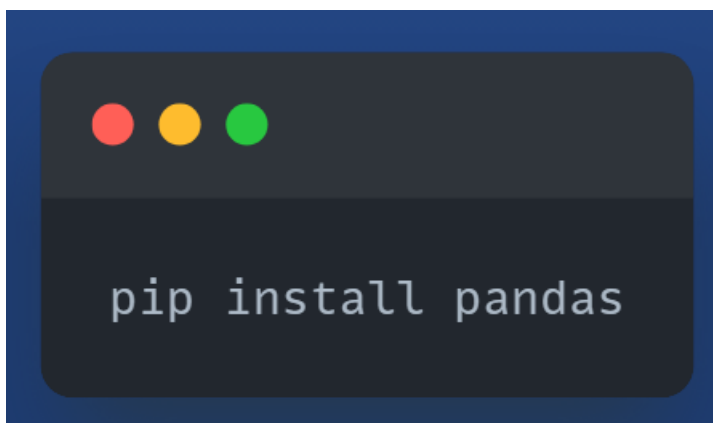
Python é uma linguagem de programação super popular e fácil de aprender. Imagine Python como aquele amigo prestativo que está sempre pronto para ajudar. Ele é usado para várias coisas, desde desenvolvimento web até ciência de dados. Com Python, você pode fazer praticamente qualquer coisa no mundo da programação.

O que é Pandas 🐼

Pandas é uma biblioteca do Python que facilita a vida de quem trabalha com dados. Pense no Pandas como uma caixa de ferramentas poderosa para mexer em suas peças de LEGO (dados). Com Pandas, você pode organizar, analisar e visualizar dados de forma eficiente e prática.

Como instalar a biblioteca

Instalar o Pandas é fácil. Você só precisa do Python instalado. Depois, abra o terminal ou o prompt de comando e digite:

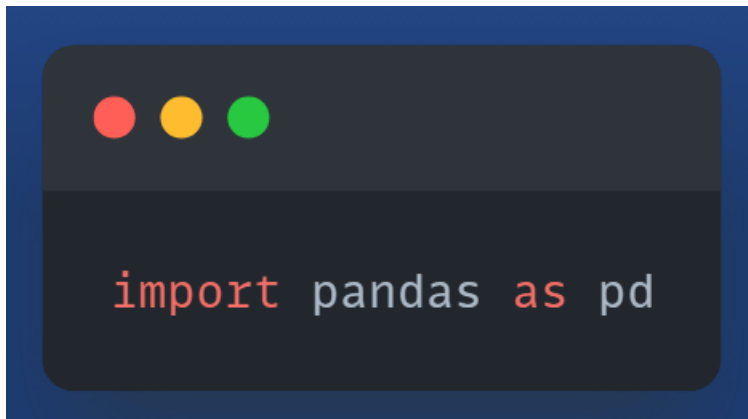
A screenshot of a terminal window with a dark blue background. The window has three colored window control buttons (red, yellow, green) in the top left corner. The text 'pip install pandas' is displayed in a light gray monospace font in the center of the terminal.

Pronto! Agora você tem a ferramenta certa para começar a trabalhar com dados no Python.

Como manipular esses dados para iniciantes?

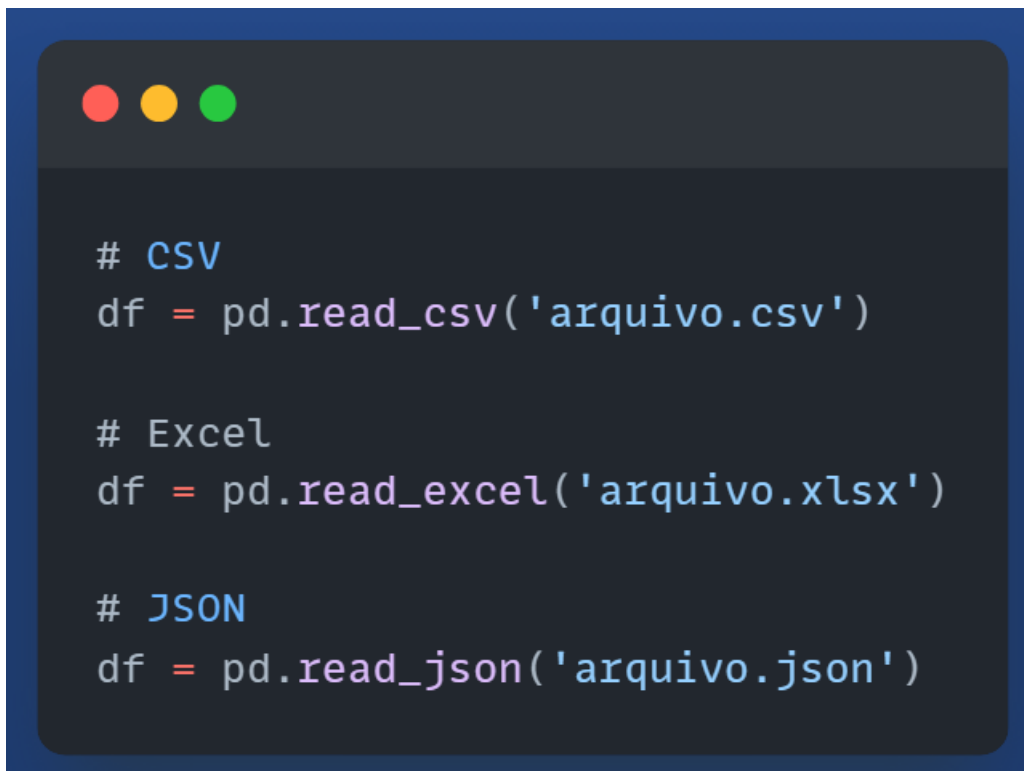
Manipular dados com Pandas é como brincar com LEGO. Primeiro, você precisa importar a biblioteca e carregar seus dados. Você pode usar um arquivo CSV, por exemplo. Aqui vai um exemplo básico:

Importar a biblioteca:

A terminal window with a dark blue background and a light gray title bar. The title bar has three colored circles (red, yellow, green) on the left. The terminal displays the command `import pandas as pd` in a monospaced font, with `import` in red, `pandas` in gray, `as` in red, and `pd` in gray.

```
import pandas as pd
```

Ler arquivos:

A terminal window with a dark blue background and a light gray title bar. The title bar has three colored circles (red, yellow, green) on the left. The terminal displays three code blocks for reading data from different file formats: CSV, Excel, and JSON. Each block starts with a comment line in blue, followed by a command to read the file into a DataFrame named 'df'.

```
# CSV
df = pd.read_csv('arquivo.csv')

# Excel
df = pd.read_excel('arquivo.xlsx')

# JSON
df = pd.read_json('arquivo.json')
```

Estes comandos carregam dados de diferentes formatos de arquivo para um DataFrame.

Criar um DataFrame:

```
data = {'Nome': ['Ana', 'João', 'Maria'],  
        'Idade': [25, 30, 28]}  
df = pd.DataFrame(data)
```

Cria um DataFrame a partir de um dicionário.

Informações do df:

```
# Tamanho da Base  
df.shape  
# Verificar o Tipo De Dados Da Coluna  
df.dtypes
```

Visualizar dados:

```
# Primeiras 5 linhas  
print(df.head())  
  
# Últimas 5 linhas  
print(df.tail())  
  
# Informações sobre o DataFrame  
print(df.info())  
  
# Estatísticas descritivas  
print(df.describe())
```

Estes comandos ajudam a entender a estrutura e o conteúdo do DataFrame.

Selecionar dados:

```
# Adicionar coluna
df['Nova_Coluna'] = [1, 2, 3]

# Remover coluna
df = df.drop('Nova_Coluna', axis=1)
```

Adicionar ou remover colunas:

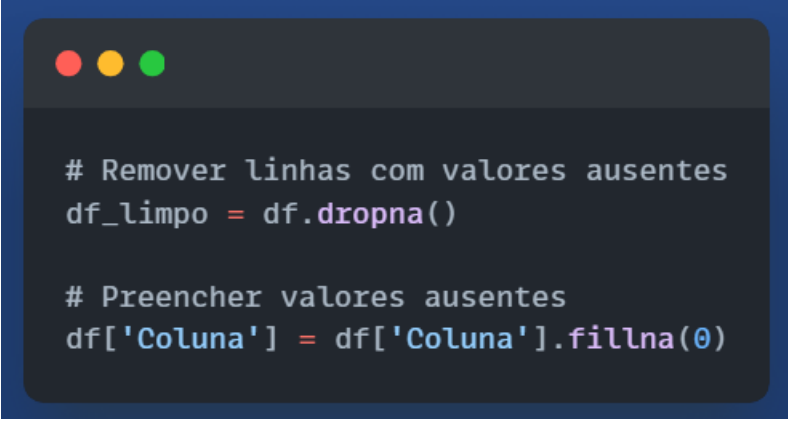
```
# Selecionar uma coluna
idade = df['Idade']

# Selecionar múltiplas colunas
subset = df[['Nome', 'Idade']]

# Selecionar linhas por índice
primeira_linha = df.iloc[0]

# Selecionar linhas por condição
adultos = df[df['Idade'] ≥ 18]
```

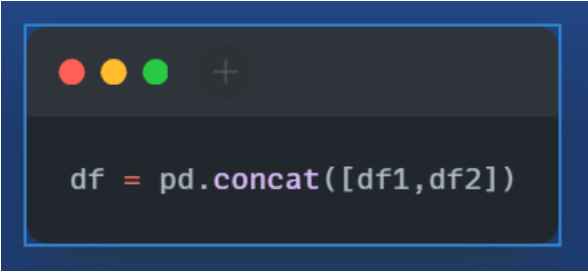
Lidar com valores ausente

A code editor window with a dark background and three colored window control buttons (red, yellow, green) in the top left corner. It contains two lines of Python code: a comment and a function call for dropping missing values, followed by another comment and a function call for filling missing values with zero.

```
# Remover linhas com valores ausentes  
df_limpo = df.dropna()  
  
# Preencher valores ausentes  
df['Coluna'] = df['Coluna'].fillna(0)
```

s:

Concatenar múltiplos DataFrames em um único DataFrame:

A code editor window with a dark background and four colored window control buttons (red, yellow, green, and a plus sign) in the top left corner. It contains a single line of Python code using the pd.concat function to concatenate two DataFrames.

```
df = pd.concat([df1, df2])
```

Conclusão

Trabalhar com dados pode parecer complicado, mas com as ferramentas certas, como Python e Pandas, fica muito mais fácil e divertido. Pensar nos dados como peças de LEGO e no Pandas como a caixa de ferramentas ajuda a entender como podemos manipular e analisar informações de maneira eficiente.

#DataScience #Python #Pandas