

Manipulação de Dados com Pandas

Transforme dados brutos em informação estruturada através de operações poderosas de filtragem, junção e agregação.



Eduardo Ogasawara

eduardo.ogasawara@cefet-rj.br

<https://eic.cefet-rj.br/~eogasawara>

Criando uma Tabela Básica

Criamos colunas numéricas e categóricas usando vetores do NumPy. Esses vetores são combinados em uma estrutura tabular chamada DataFrame, que permite operações de filtragem e análise.

```
import numpy as np
import pandas as pd

weight = np.array([60, 72, 57, 90, 95, 72])
height = np.array([1.75, 1.80, 1.65, 1.90, 1.74, 1.91])
subject = np.array(["A", "B", "C", "D", "E", "F"])

d = pd.DataFrame({
    "weight": weight,
    "height": height,
    "subject": subject
})
```

Estrutura Tabular de Dados



Colunas

Cada coluna representa uma variável



Linhas

Cada linha representa um indivíduo



Operações

Permite análises estatísticas e seleção

```
print(d.dtypes)
```

```
weight int64
```

```
height float64
```

```
subject object
```

Criando uma Tabela Auxiliar

Criamos uma segunda tabela com dados complementares que associa cada sujeito a um estado. Ela também é representada como um DataFrame e será integrada à tabela principal.

```
subject = np.array(["A", "B", "C", "D", "E", "F"])
state = np.array(["RJ", "SP", "MG", "RJ", "SP", "MG"])

ds = pd.DataFrame({
    "subject": subject,
    "state": state
})
```

subject	state
A	RJ
B	SP
C	MG
D	RJ
E	SP
F	MG

Tabela Principal e Tabela Auxiliar



Tabela Principal

Dados físicos: peso e altura



Chave Comum

Coluna `subject`



Tabela Auxiliar

Dados geográficos: estados

Ambas as tabelas compartilham a chave `subject`, que permitirá a junção entre elas.

Integração de Dados por Junção

01

Identificar chave

Coluna `subject`

02

Combinar linhas

Junção das tabelas

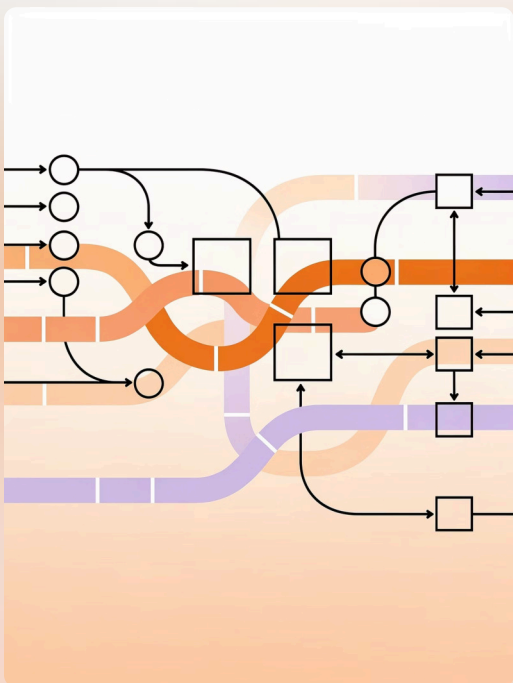
03

Resultado completo

Todas as colunas reunidas

```
dsm = pd.merge(d, ds, on="subject")
```

subject	weight	height	state
A	60	1.75	RJ
B	72	1.80	SP
C	57	1.65	MG
D	90	1.90	RJ
E	95	1.74	SP



O Poder das Transformações

1

Entrada

Dados brutos

2

Transformação

Operações encadeadas

3

Saída


Informação estruturada

Um conjunto de dados pode ser transformado passo a passo. Cada operação recebe dados e produz novos dados, criando um pipeline claro e organizado.

Filtragem de Linhas

Filtrar significa selecionar apenas algumas linhas baseado em uma condição lógica. Aqui selecionamos pessoas com altura maior que 1.7m.

```
filtered = dsm[dsm["height"] > 1.7]
```

 O resultado é um subconjunto do DataFrame original contendo apenas as linhas que atendem à condição.

subject	weight	height	state
A	60	1.75	RJ
B	72	1.80	SP
D	90	1.90	RJ
E	95	1.74	SP
F	72	1.91	MG

Seleção de Colunas

1

Antes

subject, weight, height, state

2

Depois

subject, weight, height

Selecionar colunas define quais variáveis queremos manter. Isso reduz a tabela ao que é relevante para a análise.

```
selected = filtered[["subject", "weight", "height"]]
```

subject	weight	height
A	60	1.75
B	72	1.80
D	90	1.90
E	95	1.74
F	72	1.91

ORGANIZAÇÃO

Ordenação de Dados

Ordenar significa reorganizar as linhas segundo um critério. Aqui usamos a altura como critério, com os menores valores aparecendo primeiro.

```
ordered = selected.sort_values(  
    by="height"  
)
```

subject	weight	height
E	95	1.74
A	60	1.75
B	72	1.80
D	90	1.90
F	72	1.91

Agregação por Grupos

Agrupar significa dividir os dados em categorias. Cada grupo pode ser analisado separadamente, permitindo comparar regiões.

```
grouped = dsm.groupby("state").agg(  
    count=("subject", "count"),  
    height=("height", "mean")  
)
```

MG

2 sujeitos

Altura média: 1.78m

RJ

2 sujeitos

Altura média: 1.83m

SP

2 sujeitos

Altura média: 1.77m

Interpretação dos Resultados

3

Estados

Grupos analisados

2

Sujeitos

Por estado

1.83m

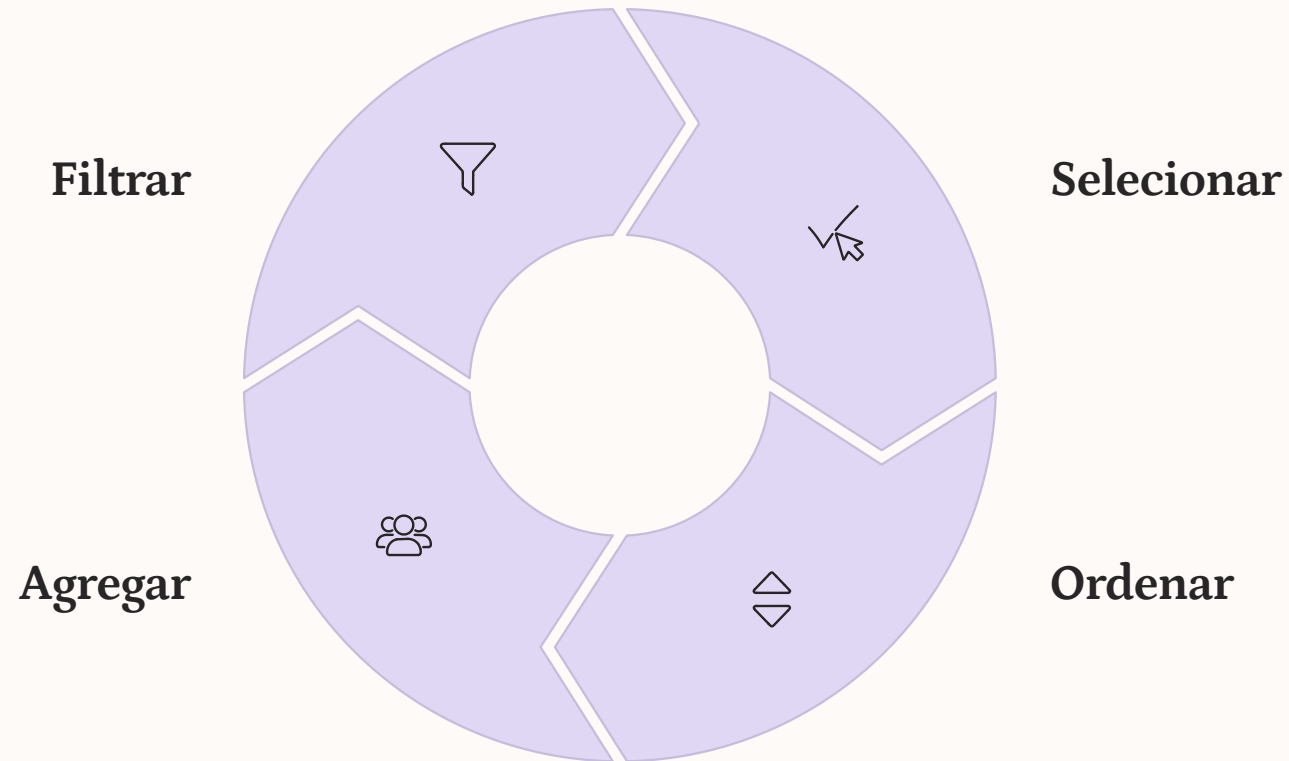
Maior Média

Estado do RJ

Cada linha representa um estado. A coluna `count` mostra quantos sujeitos existem e `height` mostra a altura média.

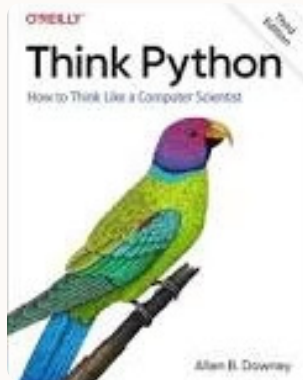
state	count	height
MG	2	1.780
RJ	2	1.825
SP	2	1.770

Dados Prontos para Análise



Após aplicar todas as transformações, os dados estão estruturados e prontos para análises avançadas e visualizações. A manipulação transforma dados brutos em informação valiosa.

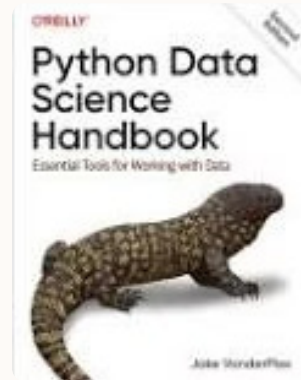
Referências



Think Python

Downey, A. *Think Python: How to Think Like a Computer Scientist*. O'Reilly Media.

An essential introduction to programming fundamentals and computational thinking using Python.



Python Data Science Handbook

VanderPlas, J. *Python Data Science Handbook*. O'Reilly Media.

A comprehensive guide to essential tools for working with data in Python, including NumPy, Pandas, and visualization libraries.



Data Science from Scratch

Grus, J. *Data Science from Scratch*. O'Reilly Media.

Learn data science fundamentals by building algorithms and tools from the ground up using Python.