# Educação Profissional Paulista

## Técnico em **Ciência de Dados**

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO

# Bibliotecas: Pandas, NumPy, SciPy, Matplotlib e Seaborn

## Pandas: combinando *DataFrames*

Aula 3

Código da aula: [DADOS]ANO1C2B4S26A3

# Mapa da Unidade 5 Componente 3

**Bibliotecas: Pandas, NumPy, SciPy, Matplotlib e Seaborn**

semana 23

Pandas: acesso e seleção

semana 24

Pandas: ler e escrever

semana 25

Pandas: transformando DataFrame

**Você está aqui!**

semana 26

Pandas: combinando DataFrames

**Bibliotecas: Pandas, NumPy, SciPy, Matplotlib e Seaborn**

**Mapa da Unidade 5 Componente 3**

# Você está aqui!

26

Pandas: combinando DataFrames

**Aula 3**

Código da aula: [DADOS]ANO1C2B4S26A3

## Objetivos da Aula

- Combinar DataFrame com métodos do Pandas Python.

## Recursos Didáticos

- Recurso audiovisual para exibição de vídeos e imagens;
- Acesso ao laboratório de informática e/ou internet;
- Software Anaconda/Jupyter Notebook instalado ou similar.

## Duração da Aula

50 minutos.

## Competências Técnicas

- Ser proficiente em linguagens de programação para manipular e analisar grandes conjuntos de dados;
- Usar técnicas para explorar e analisar dados, aplicar modelos estatísticos, identificar padrões, realizar inferências e tomar decisões baseadas em evidências.

## Competências Socioemocionais

- Colaborar efetivamente com outros profissionais, como cientistas de dados e engenheiros de dados;
- Trabalhar em equipes multifuncionais, colaborando com colegas, gestores e clientes.

Construindo o **conceito**

# Combinando *DataFrames*

### Método *merge*()

O método *merge*() é usado para combinar DataFrames com base em colunas comuns.

Ele permite especificar o tipo de junção (*inner*, *outer*, *left* ou *right*) e a coluna pela qual os DataFrames devem ser unidos.

### Método *concat*()

O método *concat*() é usado para concatenar (juntar) DataFrames ao longo de um eixo (linhas ou colunas). Ele não requer colunas em comum e simplesmente empilha os DataFrames.

Construindo o **conceito**

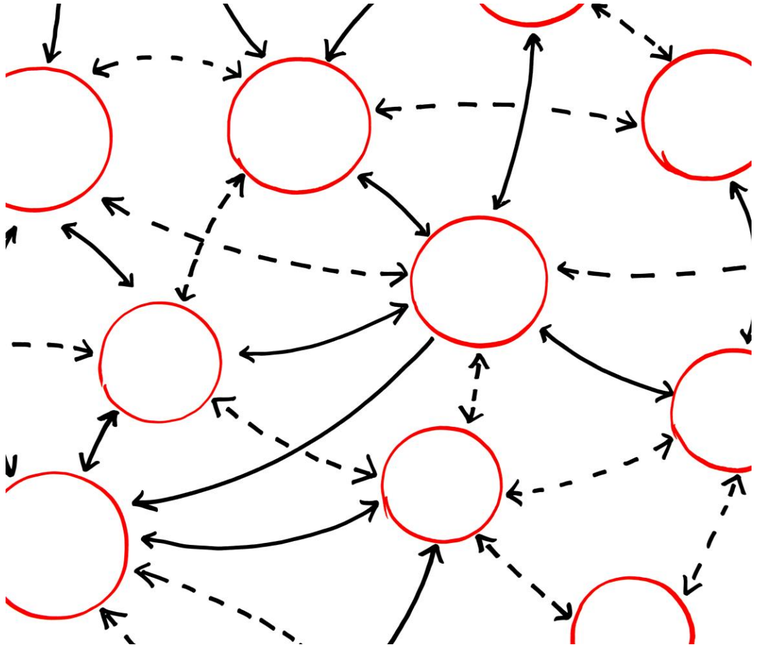
# *MERGE* – Métodos de junção

***Inner Join***

Método de junção que retorna apenas as linhas que possuem valores correspondentes em ambos os DataFrames, com base nas colunas em comum.

***Left Join***

Método de junção que retorna todas as linhas do DataFrame da esquerda e as linhas da direita que possuem valores correspondentes nas colunas em comum.



Construindo
o **conceito**

# *MERGE* – Métodos de junção

***Right Join***

O *Right Join* é um método de junção que retorna todas as linhas do DataFrame da direita e as linhas correspondentes do DataFrame da esquerda que possuem valores correspondentes nas colunas em comum.

***Outer Join***

O *Outer Join* é um método de junção que retorna todas as linhas de ambos os DataFrames e preenche com NaN as células vazias onde não há correspondência entre as colunas.

Construindo
o **conceito**

# MERGE

Observe o seguinte DataFrame:

```
import pandas as pd

# DataFrame 1
df1 = pd.DataFrame({
    'Nome': ['João', 'João', 'Pedro', 'Caio'],
    'Telefone': ['12121', '343434', '565656', '787878'],
    'Carros': ['azul', 'preto', 'verde', 'amarelo']
})

# DataFrame 2
df2 = pd.DataFrame({
    'Nome': ['João', 'Marcelo', 'Thiago', 'Caio'],
    'Irmãos': ['1', '3', '2', '2']
})
```

```
df1
```

| | Nome | Telefone | Carros |
|---|---|---|---|
| **0** | João | 12121 | azul |
| **1** | João | 343434 | preto |
| **2** | Pedro | 565656 | verde |
| **3** | Caio | 787878 | amarelo |

```
df2
```

| | Nome | Irmãos |
|---|---|---|
| **0** | João | 1 |
| **1** | Marcelo | 3 |
| **2** | Thiago | 2 |
| **3** | Caio | 2 |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

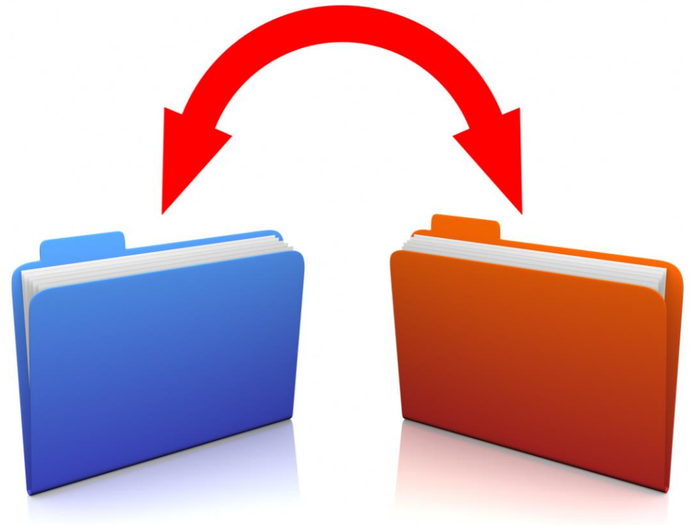
# *MERGE*

Vamos combinar os DataFrames com base na coluna "**Nome**" e usando o método de junção "**inner**".

```
# Merge com base na coluna 'Nome'
df_combinado = pd.merge(df1, df2, how='inner', on='Nome')
df_combinado
```

| | Nome | Telefone | Carros | Irmãos |
|---|---|---|---|---|
| **0** | João | 12121 | azul | 1 |
| **1** | João | 343434 | preto | 1 |
| **2** | Caio | 787878 | amarelo | 2 |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.



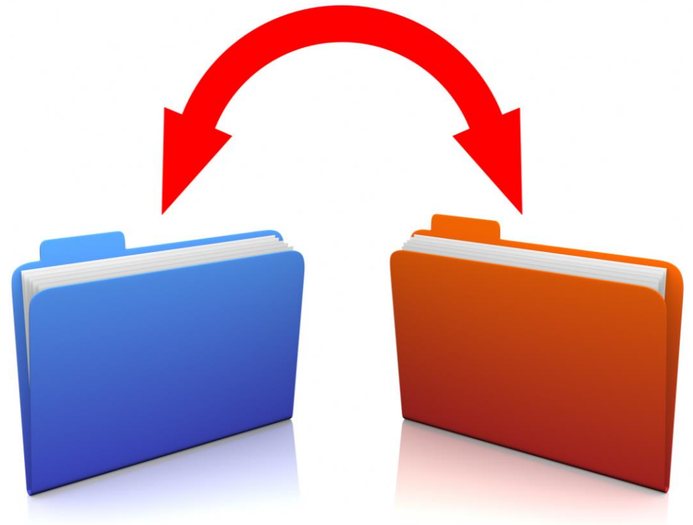
Construindo o **conceito**

# *Concat*

Vamos combinar os DataFrames com o método *concat*() ao longo do eixo das colunas (axis=1).

```
df3 = pd.concat([df1, df2], axis=1)
df3
```

| | Nome | Telefone | Carros | Nome | Irmãos |
|---|---|---|---|---|---|
| **0** | João | 12121 | azul | João | 1 |
| **1** | João | 343434 | preto | Marcelo | 3 |
| **2** | Pedro | 565656 | verde | Thiago | 2 |
| **3** | Caio | 787878 | amarelo | Caio | 2 |



Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

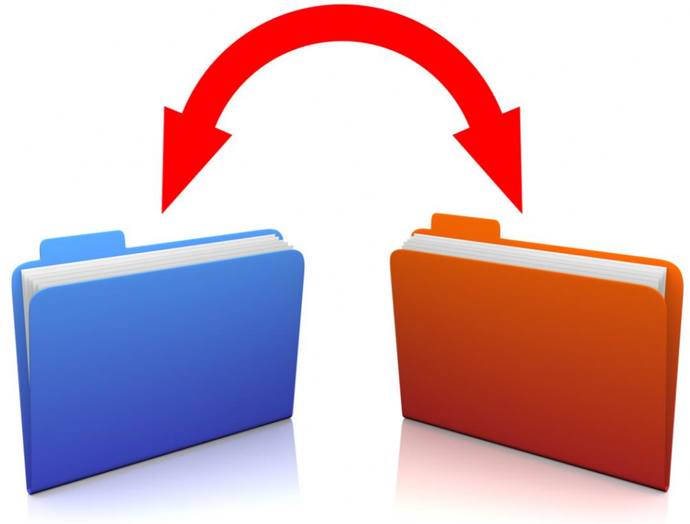
Construindo o **conceito**

# *Concat*

Vamos combinar os DataFrames com o método *concat*() ao longo do eixo das linhas (axis=0).

```
df4 = pd.concat([df1, df2], axis=0)
df4
```

| | Nome | Telefone | Carros | Irmãos |
|---|---|---|---|---|
| **0** | João | 12121 | azul | NaN |
| **1** | João | 343434 | preto | NaN |
| **2** | Pedro | 565656 | verde | NaN |
| **3** | Caio | 787878 | amarelo | NaN |
| **0** | João | NaN | NaN | 1 |
| **1** | Marcelo | NaN | NaN | 3 |
| **2** | Thiago | NaN | NaN | 2 |
| **3** | Caio | NaN | NaN | 2 |





Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo
o **conceito**

# Exemplo – Unindo *data frames* por colunas comuns (*Merge*)

**Cenário:** Você possui dois DataFrames com informações relacionadas, mas armazenadas em colunas diferentes. Deseja combinar os dados em um único DataFrame, com base em uma coluna comum.

```python
import pandas as pd

# Criando DataFrames de exemplo
clientes = pd.DataFrame(
    {"ID_cliente": [1, 2, 3], "nome": ["João", "Maria", "Pedro"], "idade": [30, 25, 22]}
)
pedidos = pd.DataFrame(
    {
        "ID_cliente": [1, 1, 2, 3],
        "produto": ["Camisa", "Tenis", "Celular", "Notebook"],
        "valor": [50, 120, 800, 2500],
    }
)
```

```
pedidos
```

| | ID_cliente | produto | valor |
|---|---|---|---|
| **0** | 1 | Camisa | 50 |
| **1** | 1 | Tenis | 120 |
| **2** | 2 | Celular | 800 |
| **3** | 3 | Notebook | 2500 |

```
clientes
```

| | ID_cliente | nome | idade |
|---|---|---|---|
| **0** | 1 | João | 30 |
| **1** | 2 | Maria | 25 |
| **2** | 3 | Pedro | 22 |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo
o **conceito**

# Exemplo – Unindo *DataFrames* por colunas comuns (*Merge*)

**Objetivo:** Combinar os DataFrame para obter um DataFrame com informações de clientes e seus pedidos, unindo por ID_cliente.

```
# Realizando o merge por 'ID_cliente'
df_merge = clientes.merge(pedidos, on="ID_cliente")

# Mostrando o DataFrame resultante
df_merge
```

| | ID_cliente | nome | idade | produto | valor |
|---|---|---|---|---|---|
| **0** | 1 | João | 30 | Camisa | 50 |
| **1** | 1 | João | 30 | Tenis | 120 |
| **2** | 2 | Maria | 25 | Celular | 800 |
| **3** | 3 | Pedro | 22 | Notebook | 2500 |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

# Exemplo

**Cenário:** Você deseja incluir todas as linhas de ambos os DataFrames no resultado do *merge*, mesmo que não haja correspondência nas colunas de junção.

```python
import pandas as pd

# Criando DataFrames de exemplo
clientes = pd.DataFrame(
    {"ID_cliente": [1, 2, 3], "nome": ["João", "Maria", "Pedro"], "idade": [30, 25, 22]}
)
pedidos = pd.DataFrame(
    {
        "ID_cliente": [1, 1, 2, 4],
        "produto": ["Camisa", "Tenis", "Celular", "Tablet"],
        "valor": [50, 120, 800, 750],
        "quantidade": [10, 20, 5, 1],
    }
)
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

# Exemplo

**Cenário:** Você deseja incluir todas as linhas de ambos os DataFrames no resultado do *merge*, mesmo que não haja correspondência nas colunas de junção.

```
clientes
```

| | ID_cliente | nome | idade |
|---|---|---|---|
| **0** | 1 | João | 30 |
| **1** | 2 | Maria | 25 |
| **2** | 3 | Pedro | 22 |

```
pedidos
```

| | ID_cliente | produto | valor | quantidade |
|---|---|---|---|---|
| **0** | 1 | Camisa | 50 | 10 |
| **1** | 1 | Tenis | 120 | 20 |
| **2** | 2 | Celular | 800 | 5 |
| **3** | 4 | Tablet | 750 | 1 |

```
# Realizando o merge externo completo
df_merge = clientes.merge(pedidos, on="ID_cliente", how="outer")

# Mostrando o DataFrame resultante
df_merge
```

| | ID_cliente | nome | idade | produto | valor | quantidade |
|---|---|---|---|---|---|---|
| **0** | 1 | João | 30.0 | Camisa | 50.0 | 10.0 |
| **1** | 1 | João | 30.0 | Tenis | 120.0 | 20.0 |
| **2** | 2 | Maria | 25.0 | Celular | 800.0 | 5.0 |
| **3** | 3 | Pedro | 22.0 | NaN | NaN | NaN |
| **4** | 4 | NaN | NaN | Tablet | 750.0 | 1.0 |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

# Exemplo

**Cenário:** Você deseja realizar um *merge* à esquerda, mas incluir apenas linhas do DataFrame principal que atendem a um determinado critério.

```python
import pandas as pd

# Criando DataFrames de exemplo
clientes = pd.DataFrame(
    {
        "ID_cliente": [1, 2, 3, 4],
        "nome": ["João", "Maria", "Pedro", "Ana"],
        "idade": [30, 25, 22, 32],
    }
)
pedidos = pd.DataFrame(
    {
        "ID_cliente": [1, 1, 2, 3, 5],
        "produto": ["Camisa", "Tenis", "Celular", "Notebook", "Tablet"],
        "valor": [50, 120, 800, 2500, 750],
        "quantidade": [10, 20, 5, 2, 1],
    }
)
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

# Exemplo

**Cenário:** Você deseja realizar um *merge* à esquerda, mas incluir apenas linhas do DataFrame principal que atendem a um determinado critério.

```
clientes
```

| | ID_cliente | nome | idade |
|---|---|---|---|
| **0** | 1 | João | 30 |
| **1** | 2 | Maria | 25 |
| **2** | 3 | Pedro | 22 |
| **3** | 4 | Ana | 32 |

```
pedidos
```

| | ID_cliente | produto | valor | quantidade |
|---|---|---|---|---|
| **0** | 1 | Camisa | 50 | 10 |
| **1** | 1 | Tenis | 120 | 20 |
| **2** | 2 | Celular | 800 | 5 |
| **3** | 3 | Notebook | 2500 | 2 |
| **4** | 5 | Tablet | 750 | 1 |

```
# Realizando o merge à esquerda com filtro
df_merge = clientes[clientes["idade"] > 25].merge(pedidos, on="ID_cliente", how="left")

# Mostrando o DataFrame resultante
df_merge
```

| | ID_cliente | nome | idade | produto | valor | quantidade |
|---|---|---|---|---|---|---|
| **0** | 1 | João | 30 | Camisa | 50.0 | 10.0 |
| **1** | 1 | João | 30 | Tenis | 120.0 | 20.0 |
| **2** | 4 | Ana | 32 | NaN | NaN | NaN |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

# Exemplo

**Cenário:** Você deseja concatenar dois DataFrames ao longo do eixo x.

Observe que o padrão **axis=0** e veja a diferença entre ignore_index=True e ignore_index=False.

```python
import pandas as pd

# Criando DataFrames de exemplo
df1 = pd.DataFrame({'col1': [1, 2, 3], 'col2': ['A', 'B', 'C']})
df2 = pd.DataFrame({'col1': [3, 4, 5], 'col2': ['C', 'D', 'E']})
```

```
df1
```

| | col1 | col2 |
|---|---|---|
| **0** | 1 | A |
| **1** | 2 | B |
| **2** | 3 | C |

```
df2
```

| | col1 | col2 |
|---|---|---|
| **0** | 3 | C |
| **1** | 4 | D |
| **2** | 5 | E |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

# Exemplo

**Cenário:** Você deseja concatenar dois DataFrame ao longo do eixo x.

Observe que o padrão axis=0 e veja a **diferença entre ignore_index=True e ignore_index=False.**

```
# Realizando a concatenação (pode haver duplicatas)
df_concat = pd.concat([df1, df2], ignore_index=True, axis=0)

# Mostrando o DataFrame resultante
df_concat
```

| | col1 | col2 |
|---|---|---|
| **0** | 1 | A |
| **1** | 2 | B |
| **2** | 3 | C |
| **3** | 3 | C |
| **4** | 4 | D |
| **5** | 5 | E |

```
# Realizando a concatenação
df_concat = pd.concat([df1, df2], ignore_index=False)

# Mostrando o DataFrame resultante
df_concat
```

| | col1 | col2 |
|---|---|---|
| **0** | 1 | A |
| **1** | 2 | B |
| **2** | 3 | C |
| **0** | 3 | C |
| **1** | 4 | D |
| **2** | 5 | E |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

# Exemplo

**Cenário:** Imagine que você possui dois DataFrames com informações relacionadas, mas armazenadas em colunas separadas:

- **DataFrame 1:** Alunos (ID, nome, curso);
- **DataFrame 2:** Notas (ID_aluno, nota1, nota2).

```python
import pandas as pd

# Criando DataFrames de exemplo
alunos = pd.DataFrame(
    {
        "ID": [1, 2, 3],
        "nome": ["João", "Maria", "Pedro"],
        "curso": ["Engenharia", "Medicina", "Direito"],
    }
)
notas = pd.DataFrame(
    {
        "ID_aluno": [1, 1, 2, 3],
        "nota1": [8.5, 9.2, 7.8, 9.1],
        "nota2": [9.0, 8.8, 8.2, 9.4],
    }
)
```

```
alunos
```

| | ID | nome | curso |
|---|---|---|---|
| **0** | 1 | João | Engenharia |
| **1** | 2 | Maria | Medicina |
| **2** | 3 | Pedro | Direito |

```
notas
```

| | ID_aluno | nota1 | nota2 |
|---|---|---|---|
| **0** | 1 | 8.5 | 9.0 |
| **1** | 1 | 9.2 | 8.8 |
| **2** | 2 | 7.8 | 8.2 |
| **3** | 3 | 9.1 | 9.4 |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Construindo o **conceito**

# Exemplo

## Objetivo:

Combinar os DataFrames em um único DataFrame que contenha as informações de alunos e suas notas lado a lado.

Observe a diferença entre ***concat*** e ***merge***.

```
1 # Realizando a concatenação horizontal
2 df_completo = pd.concat([alunos, notas], axis=1)
3
4 # Mostrando o DataFrame resultante
5 df_completo
```

| | ID | nome | curso | ID_aluno | nota1 | nota2 |
|---|---|---|---|---|---|---|
| **0** | 1.0 | João | Engenharia | 1 | 8.5 | 9.0 |
| **1** | 2.0 | Maria | Medicina | 1 | 9.2 | 8.8 |
| **2** | 3.0 | Pedro | Direito | 2 | 7.8 | 8.2 |
| **3** | NaN | NaN | NaN | 3 | 9.1 | 9.4 |

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

Diálogo

© Getty Images

O que nós **aprendemos hoje?**

# Nuvem de palavras

© Getty Images

O que nós **aprendemos hoje?**

# Então ficamos assim...

1. Para combinar informações de tabelas diferentes, podemos usar a biblioteca Pandas do Python.

2. A biblioteca possui o método *merge* que combina dois DataFrames em um só a partir de, pelo menos, uma coluna (chave).

3. Para juntar DataFrames ao longo do eixo x ou eixo y, podemos usar o método *concat*.

# Saiba mais

Já abriu um arquivo do tipo json no Pandas? No capítulo “Compreendendo os Dados” você vai descobrir como.

ALURA. *Pandas*: transformação e manipulação de dados, [s. d.]. Disponível em: https://www.alura.com.br/conteudo/pandas-transformacao-manipulacao-dados. Acesso em: 19 jul. 2024.

# Referências da aula

MCKINNEY, W. *Python para análise de dados*: tratamento de dados com Pandas, NumPy & Jupyter. São Paulo: Novatec, 2023.

PANDAS. *Pandas documentation*, 10 abr. 2024. Disponível em: https://pandas.pydata.org/docs/. Acesso em: 19 jul. 2024.

# Educação Profissional Paulista

## Técnico em **Ciência de Dados**

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO