**Para saber mais: expressões regulares**

**Expressões regulares**, também chamadas de [regex](https://www.alura.com.br/artigos/principais-casos-uso-regex-para-tratamento-dados" \t "_blank), são combinações de caracteres que podemos usar para representar partes de uma string. Elas são bem úteis para procurar padrões de escrita em textos ou validar informações.

As expressões regulares podem conter caracteres especiais e comuns. A maioria dos caracteres comuns, como '3', 'a', ou 'palavra', são as expressões regulares mais simples. Elas simplesmente servem para procurar uma expressão ou palavra específica em um texto.

Já os caracteres especiais são úteis para representar classes de caracteres comuns para identificar padrões nas strings. Alguns caracteres especiais são:

| **Caractere especial** | **Função** |
| --- | --- |
| . (Ponto) | Corresponde a qualquer caractere |
| \* (Asterisco) | Faz com que o regex corresponda a 0 ou mais repetições da expressão anterior |
| ? (Interrogação) | Faz com que o regex corresponda a 0 ou 1 repetição da expressão anterior |
| {m} | Indica que exatamente m cópias da expressão anterior devem corresponder na string. |
| {m, n} | Indica que m a n cópias da expressão anterior devem corresponder na string. |
| [] | Usado para indicar um conjunto de caracteres. [amk] corresponde a 'a', 'm' ou 'k'. O '-' pode ser usado como intervalo |
| \d | Corresponde a um dígito decimal de 0 a 9. Equivalente a [0-9] |
| \w | Corresponde a caracteres de palavras Unicode. Equivalente a [a-zA-Z0-9] |

Caso queira saber mais a respeito da sintaxe das expressões regulares, pode conferir a documentação da linguagem Python na parte de [sintaxe das expressões regulares](https://docs.python.org/3/library/re.html).

O método [replace()](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.replace.html" \t "_blank) é utilizado para substituir valores do DataFrame e permitem o uso de expressões regulares, bastando utilizar o parâmetro regex = True.

**Para saber mais: métodos de junção**

A biblioteca pandas oferece várias ferramentas para trabalhar com DataFrames, incluindo a capacidade de combinar, juntar e mesclá-los. Essas operações são úteis para combinar dados de diferentes fontes em um único DataFrame. Existem três principais métodos de união de DataFrames em pandas: **concat**, **join** e **merge**.

**Concatenação (concat)**

A concatenação é um método simples para combinar DataFrames, no qual suas colunas e índices são mantidos e os valores são simplesmente concatenados uns sobre os outros. O método [concat()](https://pandas.pydata.org/docs/reference/api/pandas.concat.html" \l "pandas.concat" \t "_blank) permite que você concatene DataFrames verticalmente (um acima do outro) ou horizontalmente (lado a lado).

A sintaxe básica para usar o método concat é a seguinte:

pd.concat([dataframe1, dataframe2, ...], axis=0/1)

No qual, axis é o eixo do qual a concatenação vai acontecer. Se axis=0, a concatenação ocorre ao longo do eixo vertical (empilhamento). Se axis=1, a concatenação ocorre ao longo do eixo horizontal (lado a lado).

## Join

O método [join](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.join.html" \t "_blank) é usado para unir dois DataFrames com base em suas colunas de índice ou uma coluna de nome especificada. Ele pode ser usado para unir DataFrames com as mesmas colunas ou com colunas diferentes.

Os dois DataFrames precisam ter uma coluna com nome em comum ou possuírem os índices contendo o mesmo tipo de informação, caso a união ocorra pelo índice e não usando alguma coluna. A sintaxe básica para usar o método join é a seguinte:

dataframe1.**join**(dataframe2, **on**='nome\_coluna', how='left')

No qual on é o nome da coluna em comum entre os dois DataFrames usada como chave de junção. Caso não seja utilizado nenhum valor, a união será feita com base nos índices dos DataFrames. O parâmetro how é o tipo de junção a ser realizada, podendo ser 'left', 'right', 'inner', 'outer'.

## Merge

O método [merge](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.merge.html) é o método mais completo para unir dois DataFrames com base em seus índices ou em colunas de ligação. A sintaxe básica para usar o método merge é a seguinte:

dataframe1.merge(dataframe2, **on**=None, how='inner', left\_on='nome\_coluna\_dataframe1', right\_on='nome\_coluna\_dataframe2')

Nela, o on é o nome da coluna nos dois DataFrames, caso seja o mesmo nome. O parâmetro how é o tipo de junção a ser realizada, podendo ser 'left', 'right', 'inner', 'outer' e 'cross'. Os parâmetros left\_on e right\_on são respectivamente os nomes das colunas do DataFrame 1 e DataFrame 2, caso tenham nomes diferentes.