**Para saber mais: método query**

O método query() do Pandas permite filtrar dados de um DataFrame utilizando uma **expressão de consulta** semelhante à linguagem SQL. Ele permite que você selecione linhas com base em condições específicas definidas na expressão de consulta, facilitando a filtragem dos dados. Isso é bastante útil ao lidar com grandes conjuntos de dados e condições de filtragem simples e complexas.

Ele aceita um texto (string) contendo a expressão de consulta e retorna um DataFrame contendo apenas as linhas que atendem às condições especificadas. A expressão de consulta pode incluir operadores lógicos como and, or e not, bem como operadores de comparação como ==, <, >, <= e >=.

Vamos ver alguns exemplos?

**Query para filtrar dados em uma coluna**

Vamos utilizar aqui, em nossos exemplos, a base de dados da Serenatto, que você pode baixar por meio deste [link](https://cdn3.gnarususercontent.com.br/3675-data-science-analise-exploratoria/Projeto/Base_Atividades/serenatto_2sem_2023.csv). Primeiro, vamos trazer o método query() aplicado a filtragem de apenas uma coluna:

**import** pandas **as** pd

vendas = pd.read\_csv("serenatto\_2sem\_2023.csv")

vendas.query('produto == "Tiramisù" ').head()

Neste exemplo, estamos filtrando o DataFrame para selecionar apenas as linhas em que o produto seja “Tiramisù”. Para efeito de visualização colocamos o head() para aparecer apenas os 5 primeiros valores.

### Query para filtrar dados de duas ou mais colunas

Este método também pode ser utilizado para filtrar dados em duas ou mais colunas, combinando as condições com operadores lógicos.

vendas.query('valor > 10 and metodo\_pagamento != "Dinheiro" ')

Neste outro exemplo, estamos filtrando o DataFrame para selecionar apenas as linhas em que o valor do produto é maior que 10 e o método de pagamento é qualquer um, exceto “Dinheiro”.

### Query com o operador @

Além disso, o método query() permite referenciar variáveis externas dentro da expressão de consulta usando o operador @. Sendo bastante útil para criar consultas dinâmicas com base em variáveis definidas previamente.

# Definindo uma variável externa

produtos = ['Café au lait', 'Espresso', 'Cappuccino']

vendas.query('produto in @produtos and metodo\_pagamento == "PIX" ')

Neste exemplo, estamos filtrando o DataFrame para selecionar apenas as linhas em que tenham os produtos passados pela lista produtos e que o método de pagamento seja em PIX.

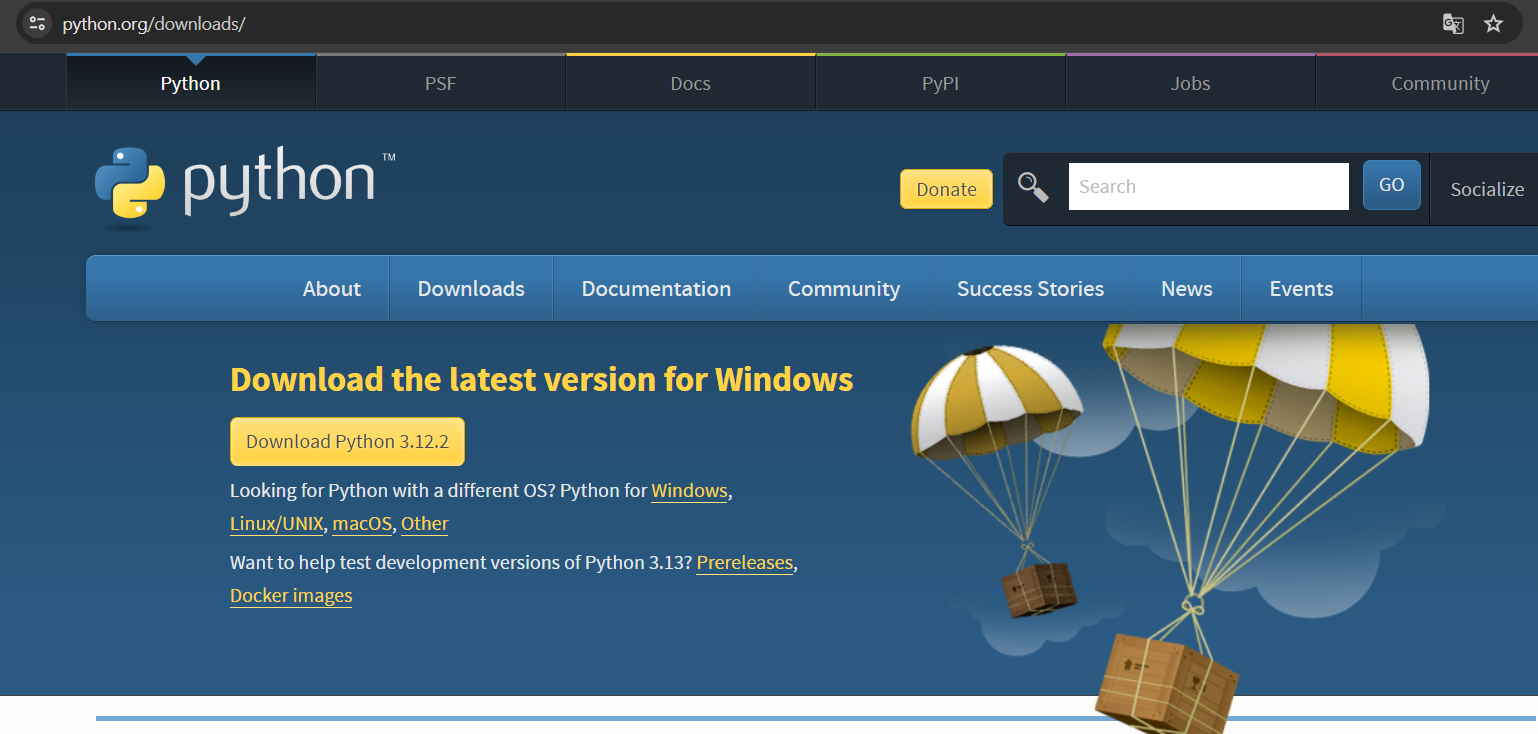
Para que você aprenda bastante sobre o método query() é importante recorrer também a sua [documentação](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.query.html).

# Para saber mais: utilizando notebooks localmente com Jupyter

O ambiente do Google Colab não é a única opção para trabalharmos com análise de dados e com notebooks interativos. Se quisermos um ambiente mais personalizado e criar códigos localmente em nosso computador, podemos recorrer ao Jupyter Notebook, uma ferramenta poderosa e versátil para trabalhar com dados.

O primeiro passo para a utilização do Jupyter é ter o Python instalado no computador. Para isso, precisamos ir no site oficial da linguagem Python e fazer o download da versão mais atualizada para o sistema operacional de preferência, seja Windows, MacOS ou ambientes Linux. Você pode acessar o site oficial a partir do link abaixo:

* [Página de downloads do Python](https://www.python.org/downloads/)



Depois de realizar o download do arquivo, basta prosseguir com a instalação. No Windows e MacOS, basta executar o programa e seguir os passos avançando as etapas. No Linux, basta executar o comando sudo apt-get install python3 no terminal.

A instalação do Python no computador vai habilitar um gerenciador de pacotes diretamente a partir do terminal ou prompt do computador. O nome desse gerenciador de pacotes é **pip**. A partir dele, será possível instalar o Jupyter Notebook, bem como as bibliotecas de análise de dados do Python.

Para usar o **pip** é muito simples, basta abrir o terminal ou prompt do computador e iniciar o comando com a palavra pip. Na [documentação do Jupyter Notebook](https://jupyter.org/install), podemos checar a maneira de instalá-lo, a partir do comando:

pip install notebook

Ao término do comando, o Jupyter Notebook será instalado no computador e para acessá-lo também é muito simples, basta usar novamente o terminal ou prompt do computador, digitar o comando abaixo e pressionar enter:

jupyter notebook

A partir disso, o Jupyter Notebook já poderá ser usado. A interface deste ambiente se assemelha muito ao ambiente do Google Colab. Porém, diferentemente do Colab, o Jupyter Notebook não vem com as bibliotecas de análise de dados instaladas. Para instalar, podemos utilizar o gerenciador de pacotes pip. Podemos utilizar diretamente do terminal ou prompt do computador ou em um Jupyter Notebook já criado. Assim como no Colab, é necessário utilizar um sinal de exclamação ( ! ) antes do código. Porém, no terminal não é necessário utilizar este sinal.

Para instalar o pandas no terminal ou prompt:

pip install pandas==1.5.3

Para instalar o pandas no Jupyter Notebook:

!pip install pandas==1.5.3

Para instalar o seaborn no terminal ou prompt:

pip install seaborn==0.13.1

Para instalar o seaborn no Jupyter Notebook:

!pip install seaborn==0.13.1

