## Tópico 2 – Manipulando dados

**OBJETIVOS**

* Analisar os benefícios em utilizar uma linguagem de programação para realizar análise de dados;
* Praticar e utilizar as estruturas de dados da [biblioteca](https://leadfortaleza.com.br/ead/glossary/Biblioteca) [pandas](https://leadfortaleza.com.br/ead/glossary/pandas);
* Empregar conjuntos de dados e realizar análises preliminares.

Vamos à prática!

Para analisar grandes conjuntos de dados, é necessário utilizar alguma ferramenta computacional que facilite a sua manipulação. Esta ferramenta deve permitir que você faça consultas, selecione subconjuntos, calcule medidas estatísticas, dentre outras operações, para responder perguntas sobre dados.

Neste tópico, você irá conhecer uma das melhores e mais utilizadas ferramentas para trabalhar com dados: a linguagem Python com a biblioteca pandas! Para você já ir se familiarizando, pandas é uma biblioteca Open Source, com diversas funções e operações eficientes para manipular dados tabulares.

Mas por que Python e pandas? Bem, existem diversas ferramentas para auxiliar profissionais a analisar dados, cada uma com suas vantagens e desvantagens. Embora isso não seja uma regra, em geral, cientistas de dados optam por utilizar linguagens de programação, como Python, ao invés de utilizar softwares prontos, como Excel, LibreOffice e Weka. Isso se deve por alguns motivos enumerados a seguir:

1. **Eficiência e escalabilidade**: devido às implementações internas otimizadas, pandas tem um desempenho muito melhor para lidar com grandes conjuntos de dados, o que é essencial quando se trabalha com Big Data;
2. **Flexibilidade e customização**: enquanto softwares prontos estão limitados a funções implementadas pelos seus desenvolvedores, quando você programa, você pode criar suas próprias funções para resolver problemas específicos;
3. **Automação**: tarefas repetitivas podem ser automatizadas criando scripts, o que poupa tempo e otimiza o trabalho. Um código que realiza análise em um conjunto de dados pode ser facilmente reutilizado para fazer a mesma análise em outros dados, sem muitos problemas;
4. **Reprodutibilidade**: compartilhando o código, você pode mostrar os passos tomados para se chegar em um resultado, permitindo que outras pessoas entendam, corrijam ou reutilizem seu trabalho;
5. **Integração com outros sistemas**: seu código pode ser incorporado em sistemas já existentes, incrementando as suas funcionalidades;
6. **Aprendizado de máquina**: por fim, utilizar Python permite que você utilize todo o poder de aprendizado de máquina em suas análises.

#### Ícone Você sabia?

Note que isso não quer dizer que programar seja sempre a melhor opção para todos os casos. Dependendo do problema e da tarefa, ferramentas como Excel ou LibreOffice podem ser perfeitamente adequadas. Além disso, existem outras linguagens voltadas para análise de dados que também são muito utilizadas, como a linguagem R. Embora neste curso iremos utilizar principalmente Python, encorajo você a sempre estudar e conhecer novas ferramentas, que tal?

Agora, sem mais delongas, é hora de começar a programar! Você precisará ter o Python 3 e o pandas instalados na sua máquina. Os exemplos e atividades serão desenvolvidos utilizando o ambiente **Jupyter Notebook**. A forma mais simples de instalar todos os componentes utilizados no curso é através do **Anaconda**, uma distribuição do Python que já vem com diversas bibliotecas e ferramentas de ciência de dados instaladas por padrão. O download do instalador do Anaconda para Linux, Windows e MacOS pode ser feito neste link:

<https://www.anaconda.com/products/individual>

Caso você utilize outra distribuição do Python, pode instalar cada componente individualmente através de linha de comando com o instalador de pacotes pip. Por exemplo, para instalar o pandas, abra uma janela do terminal e entre com o comando:

pip install pandas

Uma vez que a biblioteca esteja instalada, podemos importá-la e utilizá-la como qualquer outra biblioteca do Python através da declaração import. Para seguir os exemplos da aula, crie um Jupyter Notebook e copie os códigos utilizados nos exemplos. Aproveite para explorar e testar ideias. Para importar o pandas, utilize o comando a seguir:

In[1]: import pandas as pd

Se nenhum erro for retornado, isso significa que a [biblioteca](https://leadfortaleza.com.br/ead/glossary/Biblioteca) está corretamente instalada. Note que o trecho “as pd” está atribuindo à [biblioteca](https://leadfortaleza.com.br/ead/glossary/Biblioteca) [pandas](https://leadfortaleza.com.br/ead/glossary/pandas) o pseudônimo pd. Isso significa que sempre que você quiser se referir ao [pandas](https://leadfortaleza.com.br/ead/glossary/pandas) no código, você deverá escrever pd ao invés de [pandas](https://leadfortaleza.com.br/ead/glossary/pandas). Você deve estar se perguntando qual o motivo disso. Calma! Utilizar o pseudônimo não é estritamente necessário, mas é uma convenção que a comunidade de cientistas de dados utiliza para deixar o código mais compacto e legível. Embora não seja obrigatório, é recomendável que você utilize essa convenção, pois facilitará que você entenda o código de outras pessoas, e, assim, que outras pessoas entendam o seu código, tudo bem?

A [biblioteca](https://leadfortaleza.com.br/ead/glossary/Biblioteca) [pandas](https://leadfortaleza.com.br/ead/glossary/pandas) utiliza duas estruturas para armazenar e realizar operações com dados: [Series](https://leadfortaleza.com.br/ead/glossary/Series) e [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame" \t "_blank). Essa é um vetor unidimensional, similar às listas nativas de Python. Ou seja, uma lista de valores. Portanto, a [Series](https://leadfortaleza.com.br/ead/glossary/Series) possui um índice, chamado index que rotula cada elemento, e permite que ele seja acessado através desse índice. Confira, no exemplo a seguir, como criar uma [Series](https://leadfortaleza.com.br/ead/glossary/Series) de inteiros a partir de uma lista:

1 In[2]: s = pd.[Series](https://leadfortaleza.com.br/ead/glossary/Series)([1,2,3,4,5])

2 In[3]: s

3 Out[3]:

4 0 1

5 1 2

6 2 3

7 3 4

8 4 5

9 dtype: int64

Download do código sem numeração no link a seguir: [Download código](https://leadfortaleza.com.br/ead2pcd/conteudo/tmp/myopenolat_1_102915155002521/aula/code/code1.txt)

Examinando o código, observe que foi utilizado o método [Series](https://leadfortaleza.com.br/ead/glossary/Series) da biblioteca pandas através do pseudônimo pd, passando como parâmetro do método uma lista de inteiros de 1 a 5. Esse objeto [Series](https://leadfortaleza.com.br/ead/glossary/Series) foi atribuído à variável s. No exemplo, s foi chamada e exibiu como saída o índice (coluna esquerda) e os valores armazenados (coluna direita) da [Series](https://leadfortaleza.com.br/ead/glossary/Series).

Com isso, a [Series](https://leadfortaleza.com.br/ead/glossary/Series) possuem dois atributos principais: um índice, chamado index, que rotula cada elemento e permite que ele seja acessado através desse índice; e values, que contém os valores propriamente ditos. Esses atributos podem ser acessados separadamente. Para exibir apenas os valores, basta chamar o atributo values do objeto, como no exemplo a seguir:

In[4]: s.values

Out[4]: array([1, 2, 3, 4, 5])

Note que o valor retornado é um array (vetor), de inteiros com os valores que foram passados para o objeto. O index pode ser acessado dessa mesma forma. Assim, como mencionado, podemos acessar cada elemento especificamente através do seu index. Por exemplo, caso você precise acessar o quarto elemento da [Series](https://leadfortaleza.com.br/ead/glossary/Series), escreva:

In[5]: s[3]

Out[5]: 4

O valor retornado é o quarto elemento, que corresponde ao valor quatro. Note que essa forma de acessar elementos é idêntica às listas tradicionais de Python. Portanto, lembre-se que o índice começa com o 0!

Se o index não for especificado durante a criação da [Series](https://leadfortaleza.com.br/ead/glossary/Series), o pandas utilizará o padrão que é uma lista crescente de inteiros, começando do zero. Pode ser útil especificar um index diferente. Isso pode ser feito atribuindo uma lista para o argumento index durante a chamada do método. Agora crie uma [Series](https://leadfortaleza.com.br/ead/glossary/Series) de dados fictícios de quantidade de acidentes automobilísticos ao longo de um ano, cujo index são estes meses:

1 In[6]: s = pd.Series([15,20,12,16,10,11,12,15,9,10,12,17], index = ['jan','fev', 'mar',

2 'abr', 'mai', 'jun',

3 'jul', 'ago', 'set',

4 'out','nov', 'dez'])

5

6 Out[6]:

7 jan 15

8 fev 20

9 mar 12

10 abr 16

11 mai 10

12 jun 11

13 jul 12

14 ago 15

15 set 9

16 out 10

17 nov 12

18 dez 17

19 dtype: int64

Download do código sem numeração no link a seguir: [Download código](https://leadfortaleza.com.br/ead2pcd/conteudo/tmp/myopenolat_1_102915155002521/aula/code/code2.txt)

Ademais, a [Series](https://leadfortaleza.com.br/ead/glossary/Series) dá suporte a diversos tipos de operações sobre os dados. Algumas operações úteis são medidas estatísticas que ajudam a descrever os dados como um todo. Você já deve estar familiarizado com conceitos de média e desvio padrão. A média é uma medida de tendência central, obtida somando o valor de todos os elementos e dividindo pela quantidade. O desvio padrão é uma medida de dispersão, que aumenta conforme os valores dos dados se afastam da média. Por isso, nós podemos calcular a média e o desvio padrão de uma Series através dos métodos mean() e std(), e também calcular os valores máximo e mínimo de uma [Series](https://leadfortaleza.com.br/ead/glossary/Series), através dos métodos max() e min(). Confira nos exemplos a seguir:

In[9]: s.mean()

Out[9]: 13.25

In[10]: s.std()

Out[10]: 3.333712099692528

In[11]: s.max()

Out[11]: 20

In[12]: s.min()

Out[12]: 9

As informações obtidas indicam que, em média, ocorreram 13,25 acidentes por mês naquele ano. O desvio padrão de 3.33 indica que a quantidade de acidentes em cada mês não aparenta ter uma grande variação em relação a média. O máximo de acidentes naquele ano foi 20 em um mês e o mínimo foi de 9. Note que já começamos a ter uma percepção melhor sobre os dados. Porém, chamar esses métodos um a um pode ser cansativo. Por isso o pandas dispõe de um método bastante útil que calcula e exibe diversas medidas estatísticas de uma [Series](https://leadfortaleza.com.br/ead/glossary/Series) ao mesmo tempo, chamado describe(). Para calcular todas essas estatísticas de uma vez, faça de acordo com o que é apresentado a seguir:

1 In[13]: s.describe()

2 Out[13]:

3 count 12.000000

4 mean 13.250000

5 std 3.333712

6 min 9.000000

7 25% 10.750000

8 50% 12.000000

9 75% 15.250000

10 max 20.000000

11 dtype: float64

Download do código sem numeração no link a seguir: [Download código](https://leadfortaleza.com.br/ead2pcd/conteudo/tmp/myopenolat_1_102915155002521/aula/code/code4.txt)

Além da média, desvio padrão, máximo e mínimo, o método describe() também calcula a contagem de elementos. Nesse caso, os meses e os **quartis**, indicados pelos valores 25%, 50% e 75%. Quando os dados são ordenados de forma crescente, o quartil é um valor que os divide em quatro partes iguais. O primeiro quartil, por exemplo, é o valor para o qual abaixo dele estão 25% (¼) dos dados. O segundo quartil, também chamado mediana, divide os dados em quantidades iguais, e abaixo do terceiro quartil estão 75% (¾) dos dados. Observando o resultado dos quartis, percebemos que, em um quarto dos meses, o número de acidentes por mês foi inferior a 11; em metade dos meses foi inferior ou igual a 12; e em três quartos dos meses foi inferior a 16. Isso nos dá uma percepção ainda mais clara sobre como os acidentes estão distribuídos. Dessa forma, observa-se que meses em que ocorreram mais de 16 acidentes foram incomuns e devem ser motivo de atenção. Isso é muito útil quando se trabalha com grandes conjuntos de dados. Confira esses dados novamente.

1 In[13]: s.describe()

2 Out[13]:

3 count 12.000000

4 mean 13.250000

5 std 3.333712

6 min 9.000000

7 25% 10.750000

8 50% 12.000000

9 75% 15.250000

10 max 20.000000

11 dtype: float64

Download do código sem numeração no link a seguir: [Download código](https://leadfortaleza.com.br/ead2pcd/conteudo/tmp/myopenolat_1_102915155002521/aula/code/code5.txt)

E se você quiser saber o total de acidentes que ocorreram naquele ano? A seguir, confira como isso pode ser feito.

Para fazer esse somatório dos acidentes ocorridos durante o ano, utiliza-se o método sum(), como nesse exemplo:

In[14]: s.sum()

Out[14]: 159

O método nos diz que naquele ano ocorreram 159 acidentes, somando todos os meses. Um detalhe, que é importante notar, é que o resultado da seleção de uma [Series](https://leadfortaleza.com.br/ead/glossary/Series) também é uma [Series](https://leadfortaleza.com.br/ead/glossary/Series)! Portanto, os mesmos métodos podem ser aplicados. Você entendeu o que isso significa? Por exemplo, caso você queira saber a média dos acidentes no segundo semestre, você pode selecionar os últimos 6 meses e aplicar o método mean(). Dessa forma:

In[15]: s[6:].mean()

Out[15]: 12.5

Observe que foi feita uma seleção a partir do sexto elemento (junho) até o fim e, nessa seleção, que é uma [Series](https://leadfortaleza.com.br/ead/glossary/Series), foi aplicado o método mean(). Com isso, cobrimos o básico sobre [Series](https://leadfortaleza.com.br/ead/glossary/Series). Você entendeu até agora? Tente fazer alguns testes, fazendo seleções e aplicando métodos. Tente responder a algumas perguntas sobre os dados. Por exemplo, qual o total de acidentes no primeiro trimestre? E qual a média desse período? São muitas as possibilidades de exercício. Ah! Falando em exercício, que tal fechar esse assunto exercitando o que aprendeu? A seguir, pratique um pouco, depois conheça o [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame).

## 2.1 DataFrame

A segunda estrutura de dados existente no pandas é o [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame). Estas são estruturas bidimensionais no formato de tabela, perfeitas para lidar com os dados tabulares que você estudou anteriormente. Então, assim como foi estudado, no [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame), cada linha representa uma observação; e cada coluna, uma variável dessa observação. Existem diversas formas de criar DataFrames. A mais simples é passando um dicionário de Python, utilizando o mapeamento chave-valor. A chave é o nome da variável, e o valor passado é uma coleção de valores. No exemplo a seguir, será criado e exibido um [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame) representando a Tabela 1 que utilizamos como exemplo anteriormente. Verifique:

1 In[16]: df = pd.DataFrame({'Aluno' : ['Marina','Felipe','Cleyton','Isabel'],

2 'Créditos cursados': [20,64,32,24],

3 'Rendimento acadêmico' : [8.55,7.88,8.17,9.04],

4 'Mês de nascimento' : ['Novembro','Setembro','Janeiro','Julho'],

5 'Curso': ['Computação','Estatística','Computação','Matemática']})

6

7 In[17]:df

8 Out[17]:

Download do código sem numeração no link a seguir: [Download código](https://leadfortaleza.com.br/ead2pcd/conteudo/tmp/myopenolat_1_102915155002521/aula/code/code6.txt)

|  | **Aluno** | **Créditos cursados** | **Rendimento acadêmico** | **Mês de nascimento** | **Curso** |
| --- | --- | --- | --- | --- | --- |
| **0** | Marina | 20 | 8.55 | Novembro | Computação |
| **1** | Felipe | 64 | 7.88 | Setembro | Estatística |
| **2** | Cleyton | 32 | 8.17 | Janeiro | Computação |
| **3** | Isabel | 24 | 09.04 | Julho | Matemática |

Você notou algo diferente entre a tabela exibida, chamando a variável df e a Tabela 1? No [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame), parece haver uma coluna a mais, com valores indo de 0 a 3. Na verdade, isso não é uma coluna, e sim o index do [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame), que é análogo ao index da Series. Podemos utilizar o index para acessar linhas específicas, entretanto devemos chamar o método iloc(). No exemplo a seguir, será acessada a linha 2, que retorna um objeto com as variáveis da linha:

1 In[18]:df.iloc[2]

2 Out[18]:

3 Aluno Cleyton

4 Créditos cursados 32

5 Rendimento acadêmico 8.17

6 Mês de nascimento Janeiro

7 Curso Computação

8 Name: 2, dtype: object

Download do código sem numeração no link a seguir: [Download código](https://leadfortaleza.com.br/ead2pcd/conteudo/tmp/myopenolat_1_102915155002521/aula/code/code7.txt)

Um detalhe importante sobre DataFrames é que as colunas são formadas por [Series](https://leadfortaleza.com.br/ead/glossary/Series). Isso significa que você pode selecionar as colunas individualmente e aplicar sobre elas os mesmos métodos utilizados em [Series](https://leadfortaleza.com.br/ead/glossary/Series)! Para isso, seleciona-se uma coluna através do nome dela entre colchetes, de forma idêntica ao conceito chave-valor dos dicionários de Python. Confira nos exemplos a seguir como selecionar a coluna **Rendimento acadêmico** e calcular a sua média:

1 In[19]: df['Rendimento acadêmico']

2 Out[19]:

3 0 8.55

4 1 7.88

5 2 8.17

6 3 9.04

7 Name: Rendimento acadêmico, dtype: float64

Download do código sem numeração no link a seguir: [Download código](https://leadfortaleza.com.br/ead2pcd/conteudo/tmp/myopenolat_1_102915155002521/aula/code/code8.txt)

O retorno dessa chamada é uma [Series](https://leadfortaleza.com.br/ead/glossary/Series). Lembre que a coluna da esquerda é o index e a da direita os values.

In[20]: df['Rendimento acadêmico'].mean()

Out[20]: 8.41

Em média, o rendimento da turma foi de 8.41. Os outros métodos como std(), max(), min(), count() também podem ser usados em uma coluna, ou mesmo no [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame) todo.

Você pode ainda utilizar o método describe() da mesma maneira que na [Series](https://leadfortaleza.com.br/ead/glossary/Series), para calcular diferentes medidas estatísticas em todas as variáveis quantitativas de uma só vez. Confira a seguir:

In[21]: df.describe()

Out[21]:

|  | **Créditos cursados** | **Rendimento acadêmico** |
| --- | --- | --- |
| **count** | 4.000.000 | 4.000.000 |
| **mean** | 35.000.000 | 8.410.000 |
| **std** | 19.966.639 | 501.664 |
| **min** | 20.000.000 | 7.880.000 |
| **25%** | 23.000.000 | 8.097.500 |
| **50%** | 28.000.000 | 8.360.000 |
| **75%** | 40.000.000 | 8.672.500 |
| **max** | 64.000.000 | 9.040.000 |

Você notou que somente foram calculadas medidas das variáveis Rendimento acadêmico e Créditos cursados? Isso ocorre porque, nessa tabela, essas duas são as únicas variáveis quantitativas. Todas as outras são categóricas e não é possível fazer operações aritméticas sobre elas. Conseguiu entender até aqui? Podemos prosseguir? Então vamos lá!

Até o momento, você criou algumas estruturas de dados de forma manual. Esse conhecimento é bastante útil quando você precisar criar os seus próprios datasets, mas, na maior parte do tempo, estaremos analisando dados já existentes. Para isso, o pandas dispõe diversos métodos de leitura de arquivos de dados tabulares. Os métodos variam para cada formato de arquivo. Um formato aberto bastante utilizado é CSV, arquivos de texto com valores separados por vírgula. Outros formatos existentes são XLSX (formado de planilhas do Microsoft Excel), JSON, HTML, e muitos outros.

Para ler um arquivo CSV, utilize o método read\_csv, passando como argumento o caminho do arquivo na máquina. Caso os dados estejam disponíveis na Web, você pode passar como argumento a URL dos dados.

No exemplo a seguir, será lido um [dataset](https://leadfortaleza.com.br/ead/glossary/Dataset" \t "_blank) CSV disponível online. Os dados, em inglês, são referentes ao consumo de álcool por pessoa em diferentes países, coletados pela Organização Mundial da Saúde no ano de 2010. Confira o exemplo e, em seguida, verifique uma informação relevante no box Fique atento!

In[22]: df=pd.read\_csv('https://raw.githubusercontent.com/fivethirtyeight/data/master/alcohol-consumption/drinks.csv')

#### Ícone Fique Atento!

Alternativamente, você pode baixar os dados para a sua máquina, selecionando a opção **Salvar como**. Nesse caso, o argumento passado para o método é o nome do arquivo se ele for salvo na mesma pasta que o Jupyter Notebook ou caminho completo para o arquivo, se ele for salvo em uma pasta diferente. Para isso, tente salvar o arquivo na mesma pasta e carregá-lo chamando df=pd.read\_csv('drinks.csv').

Se o arquivo for lido corretamente, o [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame" \t "_blank) pode ser exibido chamando o nome da variável df, como você já foi feito anteriormente, mas, e se o [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame" \t "_blank) possuir muitas linhas? Verifique a seguir!

Caso o [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame) tenha muitas linhas, como no exemplo a seguir, a tabela não será exibida inteiramente. Um método muito utilizado é o head(), que recebe como parâmetro um número inteiro e exibe apenas essa quantidade do topo das linhas do [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame). Por exemplo, se você quiser exibir as 6 primeiras linhas do [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame), você pode fazer assim:

In[23]: df.head(6)

Out[23]:

|  | **country** | **beer\_servings** | **spirit\_servings** | **wine\_servings** | **total\_litres\_of\_pure\_alcohol** |
| --- | --- | --- | --- | --- | --- |
| **0** | Afghanistan | 0 | 0 | 0 | 0.0 |
| **1** | Albania | 89 | 132 | 54 | 4.9 |
| **2** | Algeria | 25 | 0 | 14 | 0.7 |
| **3** | Andorra | 245 | 138 | 312 | 12.4 |
| **4** | Angola | 217 | 57 | 45 | 5.9 |
| **5** | Antigua & Barbuda | 102 | 128 | 45 | 4.9 |

O [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame) exibido corresponde às seis primeiras linhas do [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame) completo. Esse método é útil para ter uma noção geral dos dados, quais são as colunas e que tipo de variáveis elas contém, sem precisar analisar todas as linhas. Você consegue entender este [dataset](https://leadfortaleza.com.br/ead/glossary/Dataset)? Ele possui 5 colunas: uma variável categórica, o nome do país e quatro variáveis quantitativas que são quantidade de latas de cerveja, doses de licor, taças de vinho e a quantidade total de litros de álcool consumido por pessoa em cada país em determinado ano.

Para responder perguntas em grandes datasets, pode ser importante selecionar dados que atendam a condições específicas. Para isso, é utilizada a chamada **indexação booleana**. Você utiliza como chave a condição desejada entre os colchetes. Para ilustrar essa sintaxe, imagine que você queira saber os dados a respeito de consumo de álcool no Brasil. O exemplo a seguir mostra esse código:

In[24]: df[df['country'] == 'Brazil']

Out[24]:

|  | **country** | **beer\_servings** | **spirit\_servings** | **wine\_servings** | **total\_litres\_of\_pure\_alcohol** |
| --- | --- | --- | --- | --- | --- |
| **23** | Brazil | 245 | 145 | 16 | 7.2 |

Perceba que essa sintaxe corresponde a dizer: “me dê todos os dados, em que a coluna ‘country’ tem valor igual a ‘Brazil’”. Outras condições possíveis são comparações numéricas. Por exemplo, se você quiser saber os países cujo consumo total de álcool é maior ou igual a 12 litros por pessoa, faça assim:

In[25]: df[df['total\_litres\_of\_pure\_alcohol'] >= 12]

Out[25]:

|  | **country** | **beer\_servings** | **spirit\_servings** | **wine\_servings** | **total\_litres\_of\_pure\_alcohol** |
| --- | --- | --- | --- | --- | --- |
| **3** | Andorra | 245 | 138 | 312 | 12.4 |
| **15** | Belarus | 142 | 373 | 42 | 14.4 |
| **98** | Lithuania | 343 | 244 | 56 | 12.9 |

Novamente, a sintaxe corresponde a: “me dê todos os dados onde o valor da coluna 'total\_litres\_of\_pure\_alcohol' é maior ou igual a 12.

Você percebeu que o resultado da seleção de um [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame) também é um [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame)? Muitos métodos aplicados sobre DataFrames possuem essa característica, como o já utilizado método head(). Isso significa que você pode aplicar, sobre o resultado desses métodos, todos aqueles que se aplicam em [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame), assim como ocorre com a [Series](https://leadfortaleza.com.br/ead/glossary/Series). Por exemplo, se você quiser exibir apenas os 5 primeiros resultados da seleção dos países nos quais o consumo de álcool é igual a 0, você pode chamar o método head() após a seleção. Confira:

In[26]: df[df['total\_litres\_of\_pure\_alcohol'] == 0].head()

Out[26]:

|  | **country** | **beer\_servings** | **spirit\_servings** | **wine\_servings** | **total\_litres\_of\_pure\_alcohol** |
| --- | --- | --- | --- | --- | --- |
| **0** | Afghanistan | 0 | 0 | 0 | 0.0 |
| **13** | Bangladesh | 0 | 0 | 0 | 0.0 |
| **23** | North Korea | 0 | 0 | 0 | 0.0 |
| **23** | Iran | 0 | 0 | 0 | 0.0 |
| **23** | Kuwait | 0 | 0 | 0 | 0.0 |

Neste exemplo, o resultado de df[df['total\_litres\_of\_pure\_alcohol'] == 0] é um [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame) contendo todos os países nos quais o consumo de álcool é zero. E sobre este [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame), é chamado o método head(), exibindo apenas os primeiros 5 elementos. Agora, que tal conferir uma dica bem interessante no box a seguir?

#### Ícone Fique Atento!

Quando estiver explorando conjuntos de dados, tenha em mente que nem sempre eles serão totalmente fidedignos à realidade. Coleta de dados está sempre suscetível a limitações, como erros de medição e dados faltantes. Por exemplo, neste [dataset](https://leadfortaleza.com.br/ead/glossary/Dataset), existem vários países nos quais o consumo total de álcool é igual a zero, mas isso pode significar apenas que não haviam dados disponíveis naquele país. Ou talvez o álcool seja proibido e consumido clandestinamente. Note que, além de compreender o que os dados dizem, é preciso compreender o **domínio** dos dados, e como foi feita a coleta para evitar chegar a conclusões precipitadas.

Outra operação muito útil é ordenar os dados a partir de uma variável do maior para o menor, ou do menor para o maior. No pandas, isso é feito pelo método sort\_values(). Este método recebe, como argumento, a variável pela qual os dados serão ordenados, através do argumento by, e se a ordem é ascendente ou descendente, atribuindo True ou False ao argumento ascending. O default de ascending é True. Ou seja, ordenar do menor para o maior. Confira como exibir os 5 países com maior consumo de vinho, do maior para o menor:

In[27]: df.sort\_values(by='wine\_servings', ascending = False).head()

Out[27]:

|  | **country** | **beer\_servings** | **spirit\_servings** | **wine\_servings** | **total\_litres\_of\_pure\_alcohol** |
| --- | --- | --- | --- | --- | --- |
| **61** | France | 127 | 151 | 370 | 11.8 |
| **136** | Portugal | 194 | 67 | 339 | 11.0 |
| **3** | Andorra | 245 | 138 | 312 | 12.4 |
| **166** | Switzerland | 185 | 100 | 280 | 10.2 |
| **48** | Denmark | 224 | 81 | 278 | 10.4 |

Observe que o resultado de df.sort\_values(by ='wine\_servings', ascending = False) é um [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame) contendo todos os países, ordenados do maior para o menor consumo de álcool. Chamando o método head() neste [DataFrame](https://leadfortaleza.com.br/ead/glossary/DataFrame), apenas os 5 primeiros resultados são obtidos, que correspondem aos 5 países com maior consumo. Interessante, não é? Então, o que você achou desse resultado? Esse é só um exemplo de muitos que você pode pesquisar e usar para estudar e tirar suas dúvidas.

Agora que você aprendeu o básico sobre carregar e manipular dados, pode tentar aplicar esses conhecimentos em outros datasets. Existem milhares de datasets públicos disponíveis na Web, a maioria em inglês. Confira a seguir links de portais que contém vários datasets interessantes:

#### Ícone Saiba Mais

<http://www.dados.gov.br/>

<http://www.portaldatransparencia.gov.br/>

<https://www.kaggle.com/>

<https://www.reddit.com/r/datascience>

<https://github.com/BuzzFeedNews>

<https://github.com/fivethirtyeight/data>

Infelizmente, encerramos por aqui esta aula de exploração de dados. Todavia, não é só isso que você pode explorar. O que você estudou nesta aula foram apenas algumas das operações básicas. O [pandas](https://leadfortaleza.com.br/ead/glossary/pandas) dispõe de muitos outros métodos para selecionar, modificar, agrupar, combinar e limpar dados. Ao longo dos seus estudos, certamente você utilizará algumas dessas funções, que são muito importantes para o trabalho com dados. Mas não se assuste! Você não precisa necessariamente decorar todos os métodos possíveis. O mais importante é que você compreenda, conceitualmente, o que é necessário fazer com os dados para solucionar problemas, e essa habilidade virá gradualmente com estudo e prática. Ah! Não esqueça de conferir o glossário da aula. Até a próxima!

### Referências

* Data Never Sleeps 5.0, <https://www.domo.com/learn/data-never-sleeps-5> - Último acesso em: Maio de 2020.
* Data Scientist: The Sexiest Job of the 21st Century, [https://hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century](https://www.domo.com/learn/data-never-sleeps-5https:/hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century) - Último acesso em Abril de 2020.
* Dear Mona Followup: Where Do People Drink The Most Beer, Wine And Spirits?, <https://fivethirtyeight.com/features/dear-mona-followup-where-do-people-drink-the-most-beer-wine-and-spirits/> - Último acesso em: Abril de 2020.
* Gomes; Pimenta; Schneider (23 de outubro de 2019). Data Mining in Information Science Research: Challenges And Opportunities. doi:10.5281/zenodo.3521038
* How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read, <https://www.forbes.com/sites/forbestechcouncil/2019/11/15/data-is-the-new-oil-and-thats-a-good-thing/#78bb16b87304> - Último acesso em: Abril de 2020.
* Pandas,<https://pandas.pydata.org/> - Último acesso em: Abril de 2020.
* Reis, E.A.; Reis, I.A. (2001). Análise Descritiva de Dados - Tabelas e Gráficos. Relatório Técnico RTE-04/2001, Depto Estatística-UFMG.