INICIANDO O PROJETO

Olá! Nós somos o time de pessoas Engenheiras de Dados responsável por lidar com os dados da fusão de duas grandes empresas do **varejo**, que vendem vários tipos de produtos como miojo, sabão em pó, dentre outros.

Essas duas grandes empresas se uniram, passando a fazer parte da mesma liderança. A fusão ocorreu há um mês e elas precisam entender como foi o **impacto dessa mudança nas vendas**. Então, precisam reunir os dados tanto da Empresa A quanto da Empresa B e combiná-los.

Vamos unir esses dados para fornecer ao time de *Analytics* e ao time de *Business Intelligence (BI)*, os quais vão construir relatórios para a liderança monitorar. Desejamos gerar *insights* sobre o total de vendas de produtos de determinada categoria no último mês, o que é muito importante para a liderança.

Nosso papel como pessoas engenheiras de dados é coletar os dados das duas empresas, combiná-los e realizar as **transformações necessárias** para que sejam compatíveis.

Como estamos lidando com duas empresas diferentes, teremos que trabalhar com diferenças nos nomes dos campos, na quantidade de colunas e em muitos outros aspectos. Nossa responsabilidade é lidar com essas diferenças, consolidar todas essas informações em um único arquivo e fornecer a solução para o time de *Analytics*.

Outro ponto importante é que essa demanda será recorrente. Nos próximos meses, será necessário **repetir a fusão desses dois arquivos** e disponibilizá-los. Por isso, ao construirmos essa solução, precisamos levar em consideração que ela deve ser **reprodutível** para os outros meses.

Quais ambientes podemos usar para este projeto?

Estou muito animado para montar esse projeto com você, mas nosso primeiro passo é **criar esse ambiente**. Nossa equipe conta com várias pessoas trabalhando nesse projeto e precisamos criar um ambiente que seja igual para todas. Existem diferentes maneiras de atingir esse objetivo.

Vamos optar por um ambiente que nos permita conhecer várias ferramentas: o Linux. Para isso, vamos usar o *Windows Subsystem for Linux (WSL)*, que é uma máquina virtual onde conseguimos utilizar a distribuição do sistema operacional Linux no nosso sistema Windows.

Se você possui uma máquina com Windows, poderá usar esse sistema operacional. Se você não tem acesso ao WSL, mas possui um Linux, pode trabalhar no projeto diretamente no Linux. No entanto, se não tem acesso a uma máquina Linux e não quer ou não pode instalar o WSL, pode usar o *Google Colab*, que é uma solução na nuvem bastante prática.

O *Colab* permitirá que você execute a maioria das etapas do projeto. Contudo, algumas etapas, como a **construção da estrutura** do nosso projeto, que é o que faremos agora, não serão possíveis de serem realizadas lá. Ainda assim, vale a pena testar os comandos que usaremos aqui no *Colab*, já que ele roda uma máquina Linux por trás.

Se você consegue acompanhar o passo a passo comigo, eu sugiro que execute a atividade de preparação do ambiente e crie esse ambiente Linux na sua máquina. Nesse caso, instale o **WSL**.

Vamos partir do princípio que você já instalou o WSL, e vamos começar a montar a estrutura desse projeto que precisa ser **reprodutível** para todas as pessoas da equipe.

Vamos lá?

Iniciando a construção do ambiente

Nosso primeiro passo é abrir o WSL, que nos apresentará uma tela de terminal. Aqui, podemos utilizar alguns comandos Linux. Caso você não esteja familiarizado, não se preocupe. Vamos reforçar esses conceitos de comandos Linux com atividades e apresentá-los alguns. Por exemplo, o `ls`, um comando que **mostra todas as pastas que temos no diretório** em que estamos atualmente.

```
ls
```

Observem que nada foi exibido ao rodarmos o comando `ls`, isso acontece porque acabamos de criar o sistema operacional e não inserimos nada ainda.

Criando estrutura de pastas

Para começar o projeto, vamos criar algumas **estruturas de pastas** que serão a nossa estrutura do projeto. Para fazer isso, vamos digitar o comando `mkdir`, que é de **criar um diretório ou pasta**. Podemos chamar a nossa pasta de `Documentos`, por exemplo.

```
mkdir Documentos
```

Teclamos "Enter" e não obtivemos retorno. Mas ao repetirmos o comando `ls`, obtemos a pasta:

```
Documentos
```

Para acessá-la, utilizamos o comando `cd` (*change directory*) passando o nome da pasta.

```
cd Documentos
```

Dentro de `Documentos`, desejamos criar uma nova pasta que se relacione ao nosso projeto. Essa pasta será importante e desejamos que toda a equipe tenha

essa mesma estrutura. Para ela, escolhemos o nome `pipeline_dados` e usamos o comando `mkdir`.

```
mkdir pipeline_dados
```

Teclamos "Enter" e nada retornará. Podemos usar o comando `ls` ou `dir` para visualizarmos a pasta.

```
dir
```

Obtemos dentro da pasta `Documentos`:

```
pipeline_dados
```

Para entrar nessa pasta, digitamos `cd pipeline_dados`.

```
cd pipeline_dados
```

Estamos agora dentro da pasta `pipeline_dados`.

Criando a pasta de dados brutos

Dentro do diretório `pipeline_dados`, vamos agora construir uma estrutura de pastas que faça sentido para a maioria dos projetos de Engenharia de Dados. Primeiro, precisamos de um local para guardar os **dados brutos**.

Para isso, vamos usar o comando `mkdir` novamente e criar a pasta `data_raw`, onde ficarão armazenados os dados na forma que eles vieram da origem, como uma espécie de *backup*.

```
mkdir data_raw
```

Ao usarmos o comando `dir`, obtemos a pasta `data_raw`.

Criando uma pasta para os dados transformados

Onde vamos salvar os dados depois que forem **transformados**? Podemos criar uma pasta para isso, chamada `data_processed`, onde ficarão os dados transformados e trabalhados.

```
mkdir data_processed
```

Dessa forma, temos duas pastas em nosso diretório que atende ao projeto. Quais são as outras etapas importantes?

Além disso, teremos a etapa de **explorar** os nossos dados e conhecê-los. Para isso, também precisamos de uma pasta, onde guardaremos o código que utilizaremos para essa exploração de dados. Costumamos fazer essa exploração de dados por meio de *Jupyter Notebooks*, que são ferramentas que nos permitem rodar tanto o código *Python* quanto gerar *Markdowns*.

Criando pasta de exploração de dados

Para conseguirmos organizar um documento que tenha código e descrição do que aquele código está fazendo, podemos criar uma pasta para nossos *notebooks*. Então digitaremos `mkdir notebooks` e apertamos "Enter".

```
mkdir notebooks
```

Criando pasta de *scripts*

A última pasta que pensamos em construir para nossa estrutura será a pasta de *scripts*. Mas por quê?

Depois que concluímos a etapa de exploração de nossos dados e definimos como queremos tratar e disponibilizar esses dados, vamos transformar toda essa lógica de *Python* (serpente) em um *script*. Com um único comando, podemos executar esse *script* e ele executará todo o processo.

Então, vamos criar uma pasta para nossos *scripts* digitando `mkdir scripts`, e apertamos "Enter".

```
mkdir scripts
```

Agora, podemos usar o comando `ls` e visualizamos nossas quatro pastas:

```
data_processed

data_raw

notebooks

scripts
```

Essas pastas formarão a estrutura do nosso projeto, composta por `data_processed`, `data_raw`, `notebooks` e `scripts`.

Inserindo informações na pasta `data_raw`

Agora, precisamos começar a alimentar essas pastas. Elas estão criadas, mas não têm nenhuma informação dentro delas. Como podemos preenchê-las? Vamos começar pelo `data_raw`.

Nossos dados estão disponíveis no *GitHub*. Terá um *link* nas atividades com o GitHub. Nele, teremos dois arquivos: um em *JSON* (javascript object notation) e um arquivo *CSV* (comma-separated values).

- dados_empresaA.json
- dados_empresaB.csv

Cada empresa, dada suas estruturas, determinou que seus dados seriam salvos em um formato. Então, a empresa A salva seus dados em *JSON*. Provavelmente,

porque esses dados vêm de uma API (*Application Programming Interface*). Uma solução que você navega na internet e consegue extrair esses dados.

Então, o padrão é retornar esses arquivos em *JSON*. Já o arquivo *CSV* da empresa B, provavelmente vem de um banco de dados. Eles trabalharam com um banco de dados simples e exportaram esses dados desse mês em um formato *CSV* para nos fornecer.

Aqui, já conseguimos perceber uma diferença que talvez precisemos trabalhar. Esses dados estão estruturados e salvos de maneira diferente. Mas já nos preocupamos com isso. Primeiro, queremos fazer o *download* desses dados.

Como podemos fazer o *download* desses dados e trazê-los para nossa máquina WSL (Windows Subsystem for Linux)? Voltando ao nosso terminal, vamos rodar um comando para isso. Primeiro, precisamos entrar na pasta onde queremos salvar os dados. Então, vamos usar `cd` e procurar a pasta que queremos, que é a `data_raw`.

```
cd data_raw
```

Entramos na pasta. Aqui, nosso terminal já está bem poluído, difícil de visualizar. Podemos apertar no teclado "Ctrl + L" e ele limpará esse terminal. Na verdade, dará um *scroll* para baixo e fará sumir todas as informações que tínhamos.

Podemos usar o comando chamado `wget`. O comando `wget` do Linux é capaz de **acessar uma requisição e fazer o *download* desses dados**. Então, vamos passar o comando `wget` e a URL onde estão nossos dados.

```
wget https://github.com/alura-cursos/Pipeline-de-dados-combinando-Python-e-orientcao-a-objeto/blob/main/data_raw/dados_empresaA.json
```

Vamos copiar a URL onde está nosso arquivo e podemos fazer o comando *download*. Nossos arquivos não são tão grandes. Esse *download* será bem rápido. Podemos verificar se está tudo certo apertando o `dir` e visualizamos que dentro da nossa pasta, temos nosso arquivo *JSON*.

Mas vamos focar agora no próximo arquivo. Fizemos o *download* do arquivo *JSON* da empresa A e agora desejamos o outro arquivo da empresa B, do tipo *CSV*. Para isso, vamos repetir o mesmo comando, `wget`, e colar nossa URL e apertar "Enter".

Novamente, ele fará o *download* bem rápido. Podemos apertar "Ctrl + L" para limpar o terminal, e `dir` para exibir os arquivos dentro de nosso terminal. Então, temos os dois arquivos, um *JSON* e um *CSV*.

- dados_empresaA.json
- dados_empresaB.csv

Com esses arquivos, populamos nossa primeira pasta da nossa estrutura.