Agora que já sabemos como criar um dataframe vamos entender algumas funções básicas para a manipulação dos dados.

Importando a biblioteca que contém as funções que precisamos:

from pyspark.sql.functions import \*

Para pegar a média dos valores da coluna “Age”:

df.select(mean(df.age))

Para pegar o maior valor da coluna “Age”:

df.select(max(df.age))

Para pegar o menor valor da coluna “Age”:

df.select(min(df.age))

Adicionar uma nova coluna à tabela utilizamos:

df = df.withColumn(“nova\_coluna”, df[“coluna\_1”] \*df[ “coluna\_2”])

Caso utiliza como parâmetro do nome da coluna uma coluna já existente ele irá sobrescrever seus valores pelos passados como segundo parâmetro.

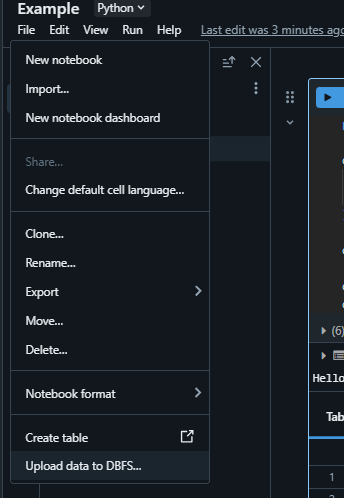
Lembrando que neste caso para realizarmos operações entre duas colunas é necessário que ambas sejam do tipo “inteiro” (int).

Remover uma coluna da tabela:

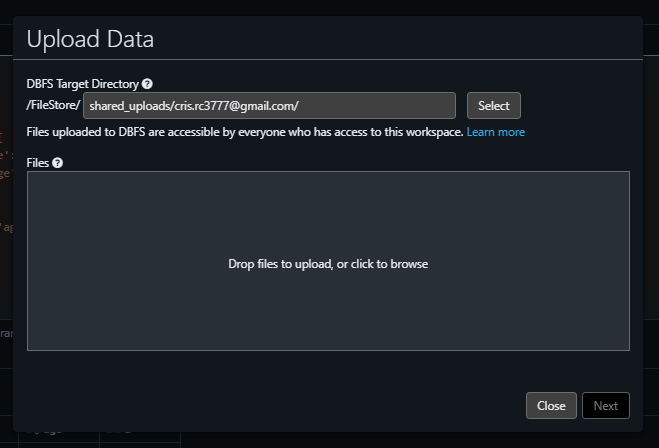
df = df.drop(“nome\_da\_coluna”)

É importante lembrar que, tanto para adicionar colunas quanto para remover, os métodos retornam um novo dataframe, e para salvarmos esse dataframe temos que atribuir no mesmo dataframe ou em um novo.

Importando arquivos CSV

 Para isso, dentro de um notebook iremos clicar em “File” e clicar em Upload “Data to DBFS”

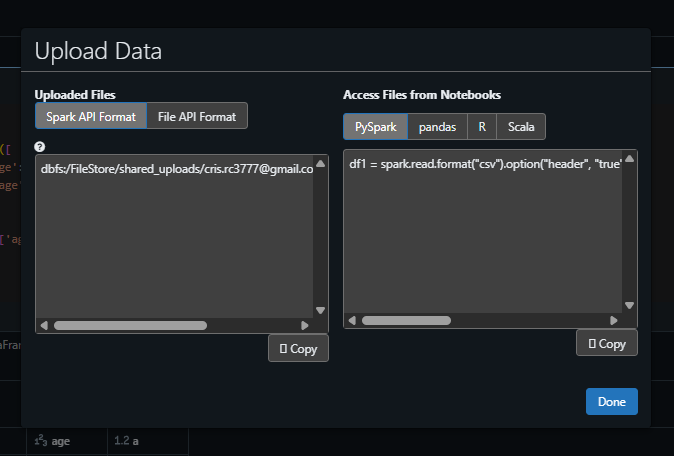
Irá abrir o seguinte modal:



Onde podemos arrastar nosso arquivo, ou clicar e selecionar.

Após selecionar o arquivo, vamos clicar no botão Next.

A próxima tela exibida será a seguinte, onde podemos copiar o path, ou já copiar a variável com o dataframe, vamos clicar no copy com a variável do dataframe.



df = spark.read.format("csv").option("header", "true").load("dbfs:/FileStore/shared\_uploads/cristian.cardoso2@br.bosch.com/marcas\_duplicadas.csv")

display(df.show(80))

Caso tenha perdido este link podemos visualizar com o seguinte comando:

display(dbutils.fs.ls('dbfs:/FileStore/shared\_uploads/{seu e-mail}'))

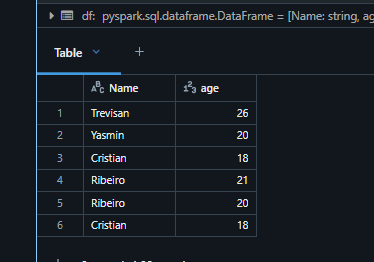
Limpando os dados

Para analisar um dataframe é essencial que ele esteja limpo, ou seja, que tenham sido retirados os dados desnecessários, ou a duplicação de dados em casos que isso não é relevante. A limpeza de dados é relativa para cada caso de análise, você deve perceber quais são as informações úteis para sua análise e quais não são.

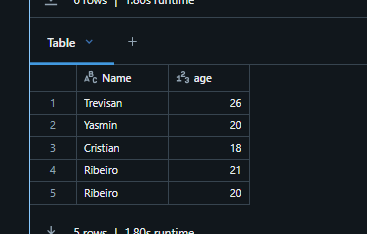
Para retirar os dados que são completamente duplicados:

df\_sem\_dados\_duplicados = df.distinc()

Então se aplicarmos no dataframe:



Teremos como retorno:



Para remover linhas de dados que possuem alguma das colunas com valor do TIPO “null”:

df = df.dropna()

Podemos realizar consultas semelhantes a consultas SQL da seguinte maneira:

consulta = df.where(

    (col('Age') == 27)

)

Operador AND:

consulta = df.where(

        (col(“Age”) == 20) **&**

        (col(“Name”) == ”Yasmin”)

)

Operador OR:

consulta = df.where(

        (col(“Age”) == 20) **|**

        (col(“Name”) == ”Yasmin”)

)

Operador NOT:

consulta = df.where(

**~**(col(“Name”) == ”Yasmin”)

)

Junção:

consulta\_resultado = df.where(

    (col('Age') == 26) &          # operador AND

    (

        (col(“Name”) == “Trevisan”) |    # operador OR

        ~(col(“Role”) == “Instructor”)     # operador NOT

    )

)

Para trabalharmos rapidamente com os dados de uma única coluna podemos utilizar “.agg”.

df = df.agg(sum("Age"))

Isso irá atribuir à df uma tabela de único valor com sua coluna sendo chamada de “sum(Age), é possível renomear da seguinte forma:

df = df.agg(sum("Age").alias(“Média das idades”))

Para ordenar uma tabela com base em um valor:

Decrescente:

df = df.sort("Age", ascending=False)

Crescente:

df = df.sort("Age", ascending=True)

Filtrando dados:

df = df.filter(df["Name"].isNotNull())

df = df.filter(df["Name"] == “Donathan”)

Agrupamento de dados

df = df.groupBy(“Name”)

Após isso podemos trabalhar com as colunas agrupadas separadamente, como por exemplo:

df = df.groupBy(“Name”).agg(avg("Age").alias("Média das idades"))

Para editar os valores contidos em uma coluna:

df = df.withColumn("Wage", regexp\_replace(col("Wage"), "R\\$", ""))

Para realizar operações entre colunas é necessário que ambas sejam de tipos inteiros, como já afirmado anteriormente, vamos ver então como fazer isso: