DOCUMENTAÇÃO

OPERAÇÕES EM PYSPARK/SPARKSQL: ‘ACIDENTES’

1- Instalação do Pyspark;

2- Importando diversas funções para possível utilização;

Fizemos a importação de várias funções para melhorar a fluidez das operações sem que precisássemos voltar ao início para instalar alguma função que tivesse ficado pendente

3- Integração com a GCP para futura importação dos Datasets;

Cumprindo o requisito de armazenar os datasets originais no ‘bucket’ da GCP, esse passo é essencial para que possamos conectar a sessão do Colab com a conta administradora da GCP, fornecendo, assim, permissão para que possamos importar para o ‘notebook’ o dataset desejado.

4- Copiando os arquivos do ‘Google Storage’ para a pasta ‘tmp’ no Colab;

5- Iniciando a sessão Spark;

6- Fazendo a leitura do dataset (já tratado) localizado na pasta tmp;

Fazendo a importação do Dataset previamente tratado em Pandas para que os insights e pesquisas sejam feitos em cima de uma base de dados de maior qualidade. Usamos na importação as opções ‘inferschema’ para que o Spark automaticamente atribua quais são os tipos de dados presentes em cada coluna. Adicionalmente, ‘header = true’, para que os nomes das colunas sejam propriamente selecionados (ao invés do cabeçalho genérico ‘c0, c1….’ e, por fim, definimos o separador como sendo ‘,’ para garantir a leitura correta do dataframe, visto que o mesmo está em formato csv.

7- Mostrando o Schema da construção do dataframe;

Usar o comando ‘df.printSchema’ é de grande valia para analisarmos o tipo dos dados após a importação, garantindo que todos estejam na formatação adequada para as consultas e operações a seguir.

8- Alterando o tipo de dado da ‘Data’ para o formato adequado;

Com o intuito de facilitar futuras operações neste dataset, optamos por converter o tipo de dado da coluna ‘Data’ para ‘date’ (o formato correto para manipulação de datas).

9- Verificando modificação;

10- Criando nova coluna para agrupar os estados por região do Brasil;

Visando o agrupamento dos estados por região do Brasil, usamos o comando 'withColumn' para criar a coluna das regiões, juntamente com o 'F.when' para quando o estado for da respectiva região, ele ser adicionado à nova coluna.

11- Removendo o índice por não apresentar utilidade para o trabalho;

O índice '_c0' não representa utilidade para o trabalho. Optamos por removê-lo nesta etapa.

12- Criando nova coluna com somente o mês;

Seguindo a mesma lógica do passo 10, criamos uma coluna para o mês.

13- Criando nova coluna com somente o ano;

Seguindo a mesma lógica do passo 10, criamos uma coluna para o ano.

14- Filtrando pela data de início da pandemia;

15- Contando o número de acidentes distintos:

16- Fazendo a média geral de idade dos envolvidos nos acidentes:

Observações interessantes: uso da função 'round' em conjunto com a 'avg' para que pudéssemos limitar o número de casas decimais obtidas no resultado da 'avg' (média). Paralelamente, usamos também 'alias' para renomear a coluna com o resultado da média.

18- Fazendo a média de idade por estado;

19- Salvando o arquivo normalizado e consultado em Pyspark na GCP;

20- Carregando o arquivo para operações em SparkSQL;

21- Agrupa os acidentes por ano;

22- Agrupa os acidentes por mês no período de JAN até SET em 2019 e 2020.