Documentação

Atividade de análise de voos

|  |  |
| --- | --- |
|  | Grupo 02:  Diego- 04241019  Ivan- 04241013  Gustavo- 04241034  Gisele- 04241052  Luis- 04241047  Vitor- 04141059 |

**2CCO – São Paulo Tech Schoool**

***[Sumário]***

Tratamento

Particionamento

Armazenamento

Link para github

**Tratamento**

**1 - Preparação e Carregamento dos Dados:**

**Importação de Bibliotecas:** O script importa bibliotecas essenciais para manipulação de arquivos, dados e conexão com a AWS. As principais são:

* **pandas** para manipulação e análise de dados em DataFrames.
* **os** para interagir com o sistema de arquivos local.
* **boto3** para se conectar e gerenciar recursos na AWS.
* **win32com**.**client** para automação de tarefas com arquivos Microsoft (neste caso, Excel).

**Conversão de Arquivo XLS:** A função converter\_xls\_para\_xlsx usa a biblioteca win32com para abrir um arquivo .xls e salvá-lo como .xlsx. Isso é necessário para que a biblioteca pandas possa ler o arquivo, já que o formato .xls é mais antigo e pode causar problemas de compatibilidade.

**2. Tratamento e Padronização de Dados:**

**Tratamento do dataset de Aeroportos e Empresas Aéreas:**

* Os arquivos aerodromospublicos.xlsx e Empresas Aereas.xlsx são carregados em DataFrames.
* **Limpeza de Nomes de Colunas**: Os nomes das colunas são padronizados para ficarem em letras maiúsculas, sem acentos e com espaços substituídos por sublinhados (\_), por exemplo, "NOME FANTASIA" se torna "NOME\_FANTASIA".
* **Limpeza de Dados Textuais**: A função remover\_acentos é aplicada a todas as colunas de texto (object), garantindo que o conteúdo também seja padronizado.
* **Conversão de Coordenadas**: A função extrair\_dms utiliza uma expressão regular (re) para extrair os valores de graus, minutos, segundos e a direção (N, S, W, E) de uma string de coordenadas, como 8° 20' 55'' S. Em seguida, a função dms\_para\_dd converte esses valores para o formato de graus decimais, que é mais fácil de ser usado para cálculos geográficos.
* **Filtro por Tipo de Voo**: Os dados de empresas aéreas são filtrados para incluir apenas as que operam voos domésticos.

**Tratamento do dataset de Reclamações:**

* **O arquivo dadosconsumidor2024**.csv é lido, usando ponto e vírgula como separador.
* **Limpeza de Colunas e Dados**: A função limpar\_texto é aplicada para padronizar os nomes das colunas e o conteúdo textual.
* **Conversão de Datas**: As colunas de data são identificadas e convertidas para o formato dd/mm/yyyy ou dd/mm/yyyy hh:mm:ss, garantindo que sejam reconhecidas corretamente.
* **Filtro por Assunto**: Os dados são filtrados para manter apenas as reclamações relacionadas a "TRANSPORTES" com o "ASSUNTO" "AEREO".

**Tratamento do dataset de Voos:**

* **O arquivo VRA\_2024.csv é lido**. O script limita a leitura a 100.000 linhas para otimizar o processamento e a demonstração.
* **Limpeza e Conversão de Tipos**: O mesmo processo de padronização de colunas e dados é aplicado. As colunas de datas e horas são convertidas para o formato dd/mm/yyyy hh:mm:ss.
* **Filtragem de Voos Domésticos:** Os dados são filtrados para incluir apenas voos domésticos. Isso é feito verificando se os códigos ICAO (SIGLA\_ICAO\_AEROPORTO\_ORIGEM e SIGLA\_ICAO\_AEROPORTO\_DESTINO) começam com 'SB', que é o padrão para aeroportos brasileiros.

**Particionamento**

* **Converte a coluna de data** para um tipo de dado datetime, o que permite a extração de informações como ano e mês de forma programática.
* **Cria uma nova coluna ANO\_MES** com o ano e o mês (por exemplo, 202405 para maio de 2024), que serve como chave para o particionamento.
* **Salva os dados** de cada mês em um arquivo .csv separado, garantindo que os dados fiquem organizados em uma estrutura de diretórios eficiente. Por exemplo, os dados de janeiro de 2024 de reclamações seriam salvos em dados/trusted/reclamacoes/reclamacao\_202401.csv.

**Armazenamento**

**1. Estrutura de Armazenamento**

A base da arquitetura é a organização dos dados em três buckets S3 na AWS, cada um correspondendo a um estágio de tratamento:

* **Raw**: Contém os dados brutos, exatamente como foram obtidos da fonte original.
* **Trusted**: Armazena os dados que foram limpos, padronizados e estão prontos para uso.
* **Refined**: Destinado a dados que passaram por processamento analítico, como agregações ou transformações que os tornam prontos para dashboards e relatórios.

A infraestrutura para esses buckets e suas "pastas" virtuais foi criada usando **Terraform** e está disponível através do arquivo **main.tf.**

**2. Envio de Dados**

* **Envio para bucket Raw**: O script **envio\_bucket.py** realiza o carregamento de todos os dados locais para o bucket raw da AWS para futuro tratamento ao mesmo tempo que preserva a estrutura original dos dados
* **Envio para bucket Trusted**: Já o envio para o bucket Trusted é feito pelo mesmo script onde ocorre o tratamento de dados, o script **tratamento.py**, usamos o mesmo arquivo para possível futuro armazenamento e automatização dos tratamentos usando um lambda ou Trigger.

**Código completo**: https://github.com/Ivanrangelpm/analise-de-voos