Case – Engenharia de Dados

Uma solução para o controle de sentimento para as mídias sociais

Gustavo Guimarães Brandão

# Propósito do Case

Intuito desse case é apresentar uma forma de armazenar arquivos em um Data Lake. Foi utilizado um ambiente Cloud para a viabilização dessa solução.

Principal proposta é conseguirmos realizar o download de vídeos de mídias sociais, transcrever esses áudios e armazená-los em um ADLS para utilização posterior em análises de sentimento e impacto dos vídeos em cima da imagem do Santander.

# Arquitetura do Projeto

Diagrama

Descrição gerada automaticamente

Imagem 1 – Arquitetura do Projeto

Acima vemos a arquitetura do projeto, nela esta separada em diversos retângulos cujo cada um tem seu proposito e finalidade.

No primeiro retângulo estão todos os objetos que remetem a configuração do ambiente, instalação dos ambientes e utilização de dependência e credencias, tais como instalação do codec “ffmpeg”, instalação de driver de som “pulseaudio-equalizer” para que a transcrição funcione conforme previsto e importe de todas as credenciais para ter acesso ao Storage Account e ter as devidas permissões para rodar os Jobs do databricks.

No segundo retângulo temos todas as tecnologias que poderão ser visadas pela tecnologia, como tiktok, youtube e tiktok. Para essa primeira versão foi desenvolvido apenas para o ambiente do youtube.

No terceiro retângulo temos um dos motores principais. Ele quem fará o trabalho de coletar todos os vídeos do youtube, na stage – que no nosso caso será o ambiente do databricks – e depois de baixar os arquivos .webm, convertê-los para o formato .wav, formato que a transcrição de áudio aceita.

Nos objetos acima, temos a apresentação da camada bronze, que no nosso caso é um storage account na Azure, um container no ADLS.

No quarto e último retângulo temos a própria transcrição de áudio, ela fará importe de todos os arquivos wav do storage account e fará a transcrição deles, após isso é aplicado uma leve harmonização dos dados, realizado a coleta de algumas informações do Youtube – como quantidade de views, url etc – e inserido na tabela “audt.audio-transcription” que é representada como a nossa camada Silver. Vale ressaltar que também é viável a realização de alguns processos como “Lematização” para auxiliar e melhorar a qualidade de uma futura analise se sentimento aplicada em cima dessas informações.

Por fim temos a apresentação da camada Gold, uma view montada em cima da camada Silver. Ela é nomeada como “g\_audt.videos\_with\_most\_view”, a funcionalidade dela é principalmente ser uma representante da camada Gold e demonstrar todo o fluxo de governança e fluxo dos dados. Acima dela também tem uma conexão com alguma ferramenta de BI, tal como Power BI, Qlick etc.

# Realizando a instalação

O projeto é composto por diversos objetos da Azure, sendo eles:

* Azure Data Lake Storage Gen 2
* Azure Key Vault
* Azure Databricks

É essencial termos a criação de todos esses objetos no portal para que o script funcione conforme esperado.

## Azure Data Lake Storage Gen 2

Necessário fazer a criação de uma instancia para criamos 2 containers. Esses 2 containers serão essenciais para que nosso script funcione pois eles serão responsáveis por armazenas os arquivos .wav e a tabela Silver. Sendo os containers:

* audio-transcription-files (Camada Bronze/Raw) – Container responsável por armazenar todos os arquivos wav para futura utilização do script
* s-audio-transcription-files (Camada Silver) – Container responsável por armazenar a tabela “audt.audio-transcription”, tabela responsável por armezar todas as informações de transcrição, url, views, data de envio etc.

Se atentar de ter todas as informações do storage account para nos passos posteriores vincular com o databricks pelas configurações do spark.

## Azure Key Vault

Responsável por armazenar todas as credenciais utilizada para acessar o ADLS e para execução dos Jobs do Databricks, possibilitando a paralelização

## Azure Databricks

Nele ficará todo o coração do script, desde os notebooks que farão a execução, os scopes para acessar a Key Vault e coletar as credenciais até a tabela de camada Gold.

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Notebooks**

* main – Responsável por toda a orquestração dos objetos e chama dos notebooks auxiliares. Nela também está armazenada o código para anonimizar as informações, esse script fará a exclusão das linhas de código de vídeos que não se encontram mais no Youtube, deduzindo que o usuário não quer disponibilizar mais aquelas informações na internet
* adls\_manipulation – Responsável por fazer a ligação com o Storage Account, enviando os .wav para o ADLS e fazendo o “mount” do repositório do ADLS
* audio\_transcriber – Ele será um notebook vinculado ao job para transcrição dos arquivos wav. Ele receberá o caminho do arquivo, validará se a linguagem é válida, se o arquivo não está corrompido e, caso todas essas informações estejam validas ele executará o script para transcrição. Após o sucesso da execução da transcrição, algumas informações como a quantidade de views e data de envio.
* audio\_transcriber\_starter – responsável em instanciar o notebook “audio\_transcriber”. Principal ferramenta para que a paralelização das tarefas ocorra com sucesso.
* config\_and\_setup – Arquivo contendo todas as variáveis de ambiente e programas que necessitam a instalação, como “pulseaudio-equalizer” e “ffmpeg”
* DDL – Arquivo responsável por criar o database audt caso não exista, a tabela “audt.audio-transcription” no ADLS e a view “g\_audt.videos\_with\_most\_view”. Vale ressaltar que a “g\_audt.videos\_with\_most\_view” será nossa tabela Gold e ela ficará repousada nas dependências do databricks pois facilita na criação de view e ela já está no ambiente que serão realizadas as análises pelos cientistas de dados e pelos engenheiros de machine learning.
* video\_downloader\_factory – Responsável por orquestrar todas as informações recebidas pelo notebook main, ela quem instanciará o ambiente e qual tarefa será necessária de acordo com ambiente, facilitando a futuras melhorias e fazendo com o usuário tenha mais flexibilidade.
* youtube\_folder/youtube\_video\_downloader – Principal notebook para realizar o download dos vídeos do youtube e fazer a conversão de .webm para .wav.

**Tabelas**

* g\_audt.videos\_with\_most\_view (Camada Gold) – View baseada na tabela “audt.audio-transcription”. Ela é repousada nas dependências do databricks e responsável por gerar visões de negócio para os vídeos com mais visualização da identificada pelo script

**Scope**

* akv-audio-transcription – Conexão direta com o Azure Key Vault. Será ele quem trará as credenciais de token do databricks (“databricks-user-token”) e a credencial do app utilizado no acesso da Storage Account (“storage-app-password”)

**Cluster**

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Foi criado um cluster simples para execução dos Jobs em paralelo, nele contem todas as informações de conexões do spark. Seguem elas abaixo:

* *spark.databricks.cluster.profile singleNode*
* *spark.hadoop.fs.azure.account.oauth2.client.endpoint.staaudiotranscripter.dfs.core.windows.net https://login.microsoftonline.com/02589359-ab33-4ac8-a14d-396cc39943ae/oauth2/token spark.master local[\*, 4]*
* *spark.hadoop.fs.azure.account.oauth2.client.secret.staaudiotranscripter.dfs.core.windows.net {{secrets/akv-audio-transcription/storage-app-password}}*
* *spark.hadoop.fs.azure.account.oauth.provider.type.staaudiotranscripter.dfs.core.windows.net org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider*
* *spark.hadoop.fs.azure.account.auth.type.staaudiotranscripter.dfs.core.windows.net OAuth*
* *spark.hadoop.fs.azure.account.oauth2.client.id.staaudiotranscripter.dfs.core.windows.net 64e72c3a-2237-4645-9e29-c7af44dc446a*

**Job**

* audio\_transcriber – Job vinculado com o notebook “audio\_transcriber”. Ele quem fará as execuções em paralelo para transcrição dos áudios. Ele recebe valores como:
  + Runs - Maximum concurrent runs – 1000
  + Tasks - Notebook parameters - audio\_source
* youtube\_video\_downloader – Job vinculado com o notebook “youtube\_folder/youtube\_video\_downloader”. Ele quem fará as execuções em paralelo para o download de vídeo de maneira mais agilizada. Ele recebe valores como:
  + Runs - Maximum concurrent runs – 1000
  + Tasks - Notebook parameters – URL do vídeo do youtube

Após realizar a criação de todos esses objetos, configuração do cluster, do scope, inserção das secrets no Key Vault, basta fazer a instalação de todos os notebooks dentro do ambiente do Databricks. Após isso realizar a criação dos Jobs e vincular aos respectivos notebooks mencionados na documentação acima.