Teste Técnico – Enjoei
Este documento tem o intuito de promover uma solução para um fluxo automatizado de dados de coleta da Fake Store API (https://fakestoreapi.com/docs) ou qualquer outra API que tenha contexto similar.

O código proposto como solução funciona para a estrutura deste case, pois tratam-se de pouquíssimos dados e tudo pode ser entregue em um único script e processado localmente. Porém, em um contexto real, o cenário poderia ser bem diferente. Para esta solução, consideraremos os seguintes pontos:

- Volumetria alta de dados, por se tratar de transações feitas por clientes de uma grande marca, como a própria Enjoei;
- Uso de estrutura de cloud para o processamento dos dados;
- Aplicação de ferramentas visando automatização do processo;
- Disponibilização dos dados para equipes de Analytics e Data Science.

Arquitetura de solução
A parte mais importante desta entrega consiste em preparar um processo de ETL (*Extract, Transform and Load*) eficaz, sucinto e que tenha camadas de armazenamento com possibilidade de rastreabilidade. O modelo de armazenamento escolhido tem base no conceito de Arquitetura Medalhão (*https://statplace.com.br/blog/arquitetura-de-medalhao/*), contendo as camadas Bronze, Prata e Ouro.

Após a definição do modelo de armazenamento, é importante definir quais stacks serão utilizadas. Para este caso, foram definidas as seguintes ferramentas:

- **Cloud:** A Cloud escolhida será a Google Cloud Platform (GCP), onde serão armazenados os dados das camadas bronze e prata no *Cloud Storage*. Também serão utilizados os recursos de cluster da GCP e o Dataproc para gerenciamento dos jobs;

- **Banco de dados:** O tipo de banco escolhido será o BigQuery para a camada ouro;

- **Framework para execução dos scripts**: Spark com Python (Pyspark). O Spark é um dos frameworks de processamento distribuído mais utilizados atualmente devido a sua ótima performance em Big Data, fácil aprendizado e suporte a linguagens de programação como Python, R, Scala e SQL;

- **Orquestrador dos Scripts**: As estapas do processo de ETL serão orquestradas por uma DAG do Apache Airflow. Através desta stack, será determinada a sequência de execução dos scripts e também um horário específico para que o trigger ocorra;

- **Ferramenta de BI**: As análises serão feitas utilizando o Power BI, sendo este atualmente a ferramenta mais completa para este propósito, oferencendo uma gama maior de recursos do que seus concorrentes, além de ser muito intuitivo.

Agendamento do processo
No Airflow, o trigger pode ser feito através de um evento específico, um horário previamente determinado ou manualmente. Para este caso, o processo será automaticamente iniciado às 19h, visando coletar todas as transações feitas durante as 24 horas anteriores e disponibilizá-las para análises descritivas e preditivas até a manhã do dia seguinte.

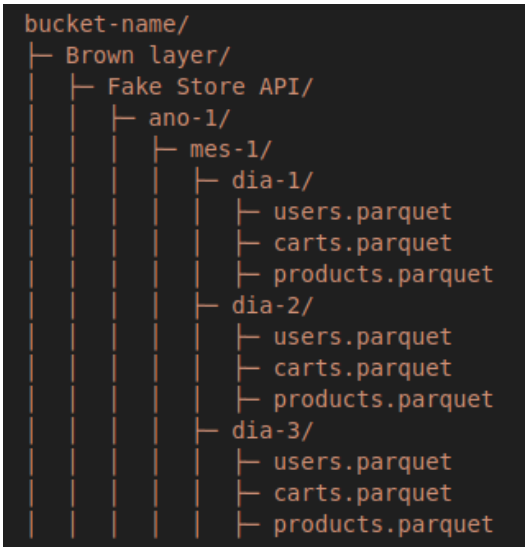
Precauções
Devido ao trigger ser a noite, é possível colocar uma configuração na DAG para que se algo falhar, o dono do processo receba uma notificação via chat do Google e possa então verificar o ocorrido. Esta é uma boa prática para que não seja necessário assistir a execução o tempo todo.

Também serão programadas mensagens de sucesso ao final do processo, indicando que tudo deu certo.

Na sequência do documento, serão detalhadas cada uma das etapas do ETL.

Extração dos dados
Devido a volumetria alta de dados, as execuções serão divididas em mais de um script. O primeiro deles será a coleta dos dados da API e o posterior armazenamento deles em formato parquet (*https://www.databricks.com/br/glossary/what-is-parquet*) em um diretório do Cloud Storage. Os dados serão armazenados em seus formatos brutos, visando termos rastreabilidade de tudo que foi obtido nessa fase.

O armazenamento seguirá a estrutura de **ano** < **mês** < **dia**, conforme ilustração abaixo:



É importante refinar o período de dados a ser mantido no bucket, visando otimização de custos com storage. Porém se a empresa optar, todo o histórico poderá ser mantido e não haverá uma etapa de expurgo de dados, havendo então somente adição de arquivos.

É muito importante também ter uma documentação de como os dados serão organizados, facilitando as etapas seguintes e visando evitar que este diretório se torne um Data Swamp (*Pântano de dados*).

Todo este contexto resume a **Camada Bronze** da arquitetura**.**

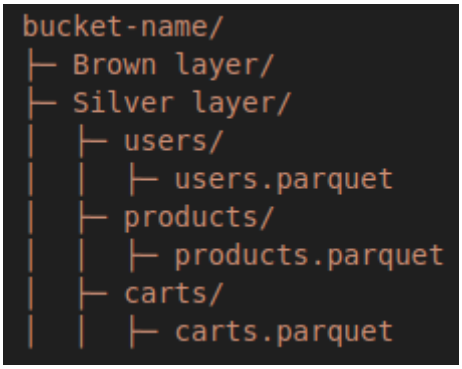
Tranformações
Esta etapa é composta de três scripts:

- users.py
- carts.py
- products.py

Aqui os dados brutos recebem tratamentos como a eliminação de duplicidades, tratamento de valores nulos, tipos corretos de colunas, normalizações e enriquecimentos através de regras de negócio mapeadas.

Sobre a volumetria que percorre os scripts de transformações, será adotada a premissa de leitura com base no dia atual, ou seja, haverá um filtro para as pastas da camada bronze para um único dia, por exemplo, 01/11/2024 e assim sucessivamente. Com isso, espera-se que o processamento seja mais ágil, contribuindo para um melhor custo com uso de recursos de cluster.

Cada um dos scripts armazenará dados em um diretório específico do bucket, porém neste caso, o modo de gravação será por ***overwrite***, ou seja, só será guardado o que estiver sendo processado, visto que o dado bruto está guardado na camada bronze.



Todo este contexto resume a **Camada Prata** da arquitetura**.**

Carregamento dos dados em banco
Nesta etapa os dados são transferidos para o banco BigQuery em um Data Warehouse, representando a **Camada Ouro** da arquitetura, seguindo as seguintes premissas:

Carrinho de compras

- Hard Delete dos dados do banco através da coluna ***date*** para o período selecionado, por exemplo, 01/11/2024, caso existam;
- Insert de dados para o mesmo período.

Usuários

- Conceito de UPSERT, ou seja, se um usuário já existe no banco, mas seus dados mudaram, a linha cujo aquele userId estiver será atualizada. Caso não exista, será adicionado como novo registro.

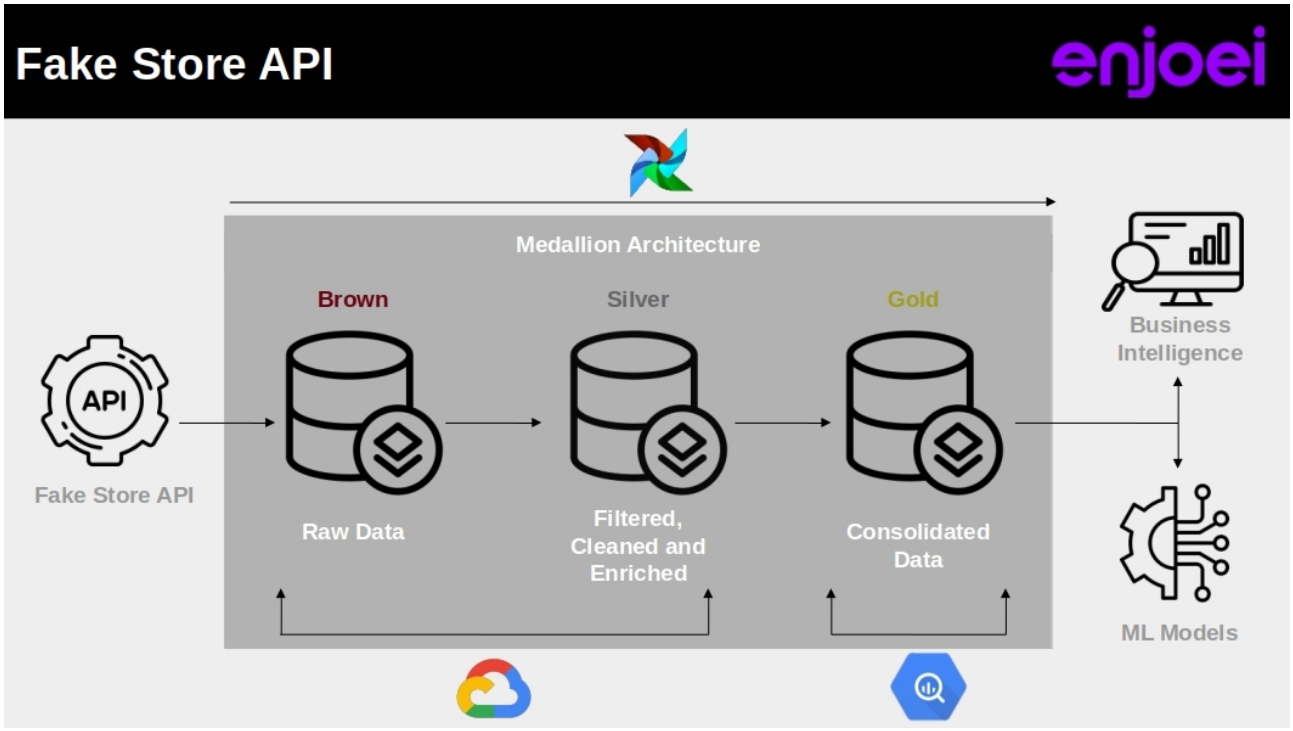
Produtos

- Conceito de UPSERT, ou seja, se um produto já existe no banco, mas seus dados mudaram, a linha cujo aquele product_Id estiver será atualizada. Caso não exista, será adicionado como novo registro.

Sobre a escolha de um banco de dados para a camada ouro, destacam-se os pontos:

- Recursos do banco como Constraints e Foreign Keys que ajudam na verificação de possíveis erros no processo de ETL;
- Fácil consulta através da linguagem SQL;
- Compartilhamento com stakeholders;
- Facilidade de conexão com o Power BI.

Diagrama de arquitetura do processo



Análises descritivas dos dados

As perguntas feitas no teste, dentro do contexto proposto neste relatório podem ser respondidas através de consultas SQL ou dashboards em BI. Neste caso, facilmente se chega a respostas como:

- Qual a última data que um usuário adicionou compras ao carrinho?
- Qual a categoria de compras preferida do usuário?
- Quais são os dados de cadastro dos usuários.

E muitas outras respostas.

Segurança dos dados durante disponibilização
Tão importante quanto o processo citado acima é também garantir que os dados estejam seguros. Neste contexto, ficam definidas as seguintes premissas:

- Os usuários que consomem os dados não terão acesso as pastas do Cloud Storage onde as informações estão. Isso pode ser feito através de um controle de acesso feito na própria Cloud, dando permissão somente a usuários específicos do processo de ETL.

- Para o banco de dados, os usuários não terão acesso direto as tabelas do DW, mas sim a Materialized Views geradas pelo time de engenharia após entendimento do que é necessário para a construção dos dashboards no Power BI. Esta premissa traz vantagens como:

    1. Segurança para os dados do DW, onde não haverão riscos de informações serem perdidas devido drops, truncates, deletes, etc;

    2. Somente usuários autorizados do time de Analytics e Data Science terão acesso as materialized views;

    3. Melhor eficiência para o Power BI, pois tudo será entregue nas materialized views, não havendo necessidade alguma de tratamentos adicionais no Power Query. Isso tornará mais rápido o carregamento dos painéis;

    4. Aumento de eficiência para o time de Analytics, pois precisarão apenas se preocupar com a geração de valor através dos dados, uma vez que o ETL estará pronto.

Conclusão
A proposta acima deve entregar o esperado, garantindo qualidade, boa performance e um custo eficiente de infraestrutura.

Alguns pontos podem ser discutidos pelo time de engenharia, como por exemplo o processamento apenas dos dados extraídos de períodos maiores, caso o volume de dados seja baixo, ou de repente, o uso de outras ferramentas. Em ambos os casos, é necessário avaliar os cenários antes de aplicar as mudanças.