Relatório do projeto Mercury

Um resumo do Mercury, meu primeiro projeto de Data Eng

Motivação

Após conseguir um estágio na área de Business Intelligence, fui introduzido ao mundo de dados. Inicialmente eu tinha uma certa aversão a SQL e Banco de Dados por ser “simples demais”, uma ignorância minha. No meu dia a dia como estagiário, SQL e Banco de Dados era o que eu mais via. Conversando com um outro estagiário, ele me apresentou alguns cursos que estava realizando para melhorar suas habilidades e consequentemente ser um melhor estagiário, dessa forma que eu conheci a Data Science Academy. Nosso setor é chamado de BI, mas na prática realizamos o trabalho de Engenheiro de Dados, e então comecei a fazer o curso Fundamentos de Engenharia de Dados da DSA, e aprendi bastante, o que me ajudou muito a entender o que a equipe de BI faz. Nem cheguei a terminar o primeiro curso e já comprei a Formação Engenheiro de Dados, que era um curso completo para me tornar um Engenheiro de Dados, e o primeiro módulo era: Design e Implementação de Data Warehouses. Esse curso foi o match perfeito, envolvia modelagem de Banco de Dados que eu aprendia na faculdade (Sistemas de Informação - UFF), e tudo o que eu aprendia no dia seguinte eu verificava como foi implementado em uma empresa real com centenas de clientes consumindo os dados todos os dias. Somando todo esse conhecimento, senti que era necessário fazer o meu próprio DW, eu já tinha visto como é na empresa que eu trabalho e no projeto do curso, então chegou a minha vez. Com isso, eu criei o Mercury, um projeto prático de Data Warehouse, onde faço tudo, do início ao fim dos dados.

Projeto

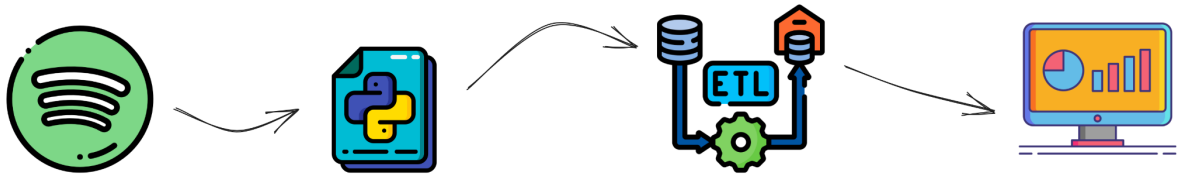
O desenvolvimento do projeto durou 3 semanas, onde foram feitas cargas diárias, históricas, além de muito select, drop e truncate. Detalhes técnicos estão nos códigos no GitHub.

Requisitos

Não foram estabelecidos requisitos formais, a ideia era poder ver meus dados de uso do Spotify para poder entender e analisar as minhas escutas.

Fluxo de Dados

A fonte de dados (Spotify) é uma API, com diversos endpoints nos quais eram extraídos diferentes tipos de dados todos em lotes. Os dados passam por uma filtragem inicial, para não serem armazenados dados que não seriam utilizados na análise. Essa filtragem consiste em apenas extrair os dados necessários da fonte e gravar na staging area no banco de dados. Após isso os dados são transformados dentro do banco de dados, sendo modelados multidimensionalmente, no modelo star schema. A ferramenta de visualização de dados se conecta no esquema com os dados organizados, limpos e transformados.



Arquitetura e ferramentas

Overview

Para esse projeto foi utilizado um ambiente on-premises com docker. Foi criado um stack com dois bancos de dados PostgreSQL, um para o Metabase e outro para o DW. Além desses existe também um container para o Metabase. Todos os containers possuem volumes e estão na mesma rede, mercury_net. Infelizmente o script ETL não está sendo executado dentro do stack, pois a autenticação do Spotify exige que seja aberta uma sessão no navegador, onde o usuário se conecta na sua conta para autorizar o uso dos dados, com isso o script está sendo executado na minha máquina, agendado pelo cron.

Explicações

Foi utilizado o PostgreSQL pois foi o banco que eu tenho mais familiaridade por conta do curso da DSA. A implementação foi on-premises pois estou aprendendo e não quero arriscar gastos indesejados para realizar o meu primeiro projeto, a ideia é depois que eu estiver sabendo o que estou fazendo, posso explorar opções em cloud, e isso me deu uma boa oportunidade para aprender e praticar Docker e Linux. Foi criado um script em Python agendado no cron pois julguei ser o adequado para minha demanda, pra mim não faz sentido ter um Airflow com uma grande capacidade e consumindo recursos da minha máquina para realizar um simples script Python, compensei a falta de monitoramento com uma boa estrutura de código. Escolhi o Metabase como ferramenta de Dataviz por ser open source, permite query SQL, possui conectores e possui uma imagem no Docker Hub.

Segurança e backup

Como medida de segurança, tanto o Metabase quanto os Bancos de Dados estão disponíveis apenas na rede interna (mercury_net), e além disso para acessar cada base é necessário se autenticar com login e senha. Criei um usuário para cada aplicação, um usuário para o Metabase com permissão apenas de visualização em tabelas do DW. PyCharm por outro lado é um usuário com mais permissões, como uso de sequências, select e insert. Esse usuário é o que o script etl.py utiliza. Por fim, existe o superusuário MercuryDBA. É realizado um backup diário dos dois bancos, através de um shell script agendado no cron.

Modelagem Dimensional

Extraindo dados de diferentes endpoints da API do Spotify, é possível obter informações detalhadas sobre os fatos (escutas), como dados relativos ao artista, faixa e álbum. Utilizei

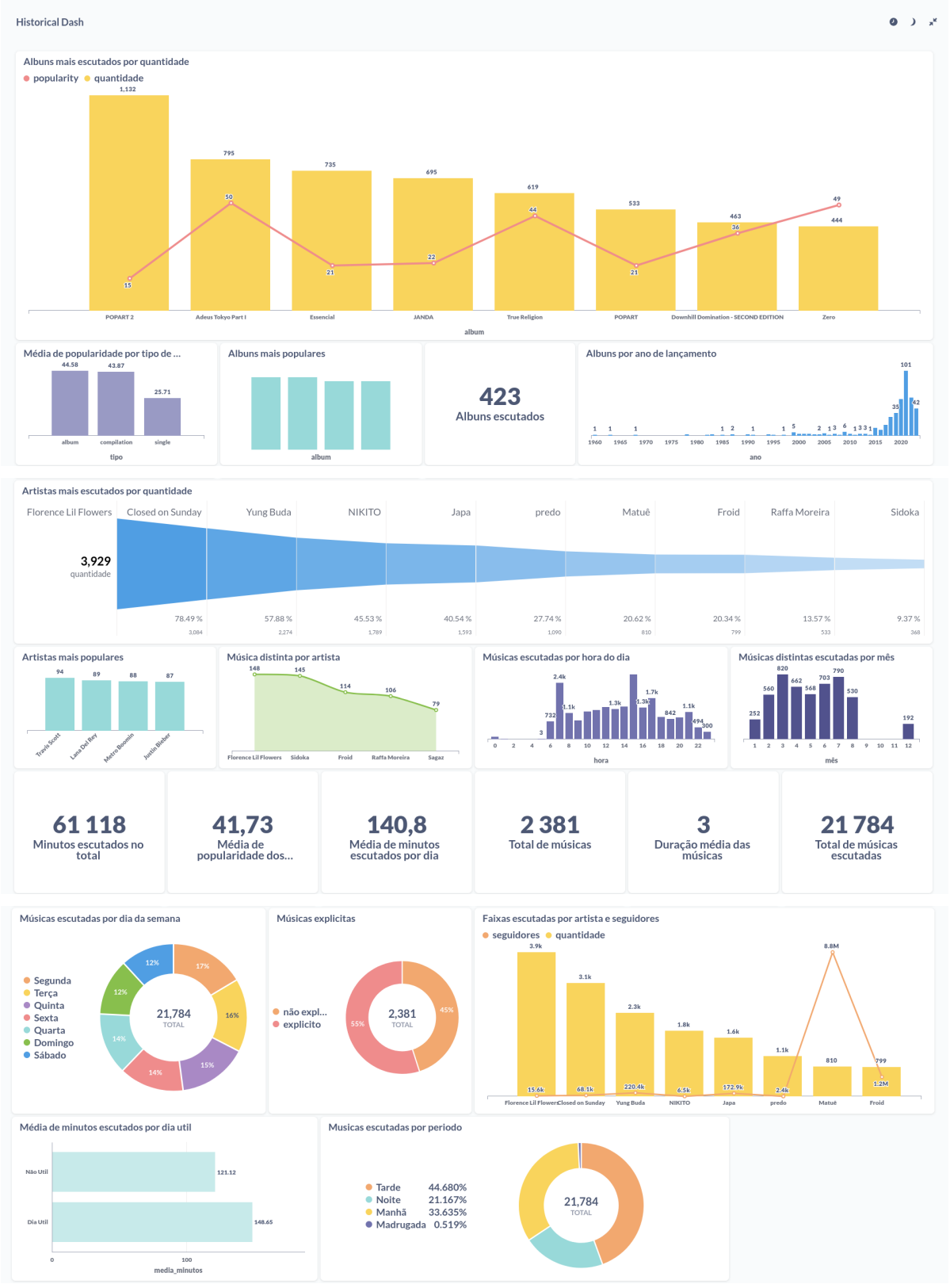
o modelo Star Schema por ser mais simples, sendo ideal para o meu caso. Com isso modelei meu DW da forma abaixo:

Visualização dos Dados

Para visualizar os dados, conforme mencionado anteriormente, utilizei o Metabase. Nele eu optei por escrever queries, para praticar SQL e poder ter uma maior customização.

Dashboard histórico

Dashboard com o olhar histórico, analisando todos os dados nos Data Warehouse sem filtro de data. Destaco que todos os gráficos são interativos, quando o cursor passa por determinado gráfico, mais dados são exibidos.



Dashboard single

Dashboard com o olhar para um dia em específico, compara os dados com o dia anterior.