UNIVERSIDADE ESTADUAL DA PARAÍBA

CAMPUS I – CAMPINA GRANDE

CENTRO DE CIÊNCIAS E TECNOLOGIA

DEPARTAMENTO DE COMPUTAÇÃO

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

RENAN REY COSTA RODRIGUES

DESENVOLVIMENTO DE ARQUITETURA DE DADOS PARA SUPRIMIR DEMANDA DE INFORMAÇÃO DE UMA ORGANIZAÇÃO COM DIVERSOS SISTEMAS USANDO O APACHE AIRFLOW

CAMPINA GRANDE- PB
2023

RENAN REY COSTA RODRIGUES

DESENVOLVIMENTO DE ARQUITETURA DE DADOS PARA SUPRIMIR DEMANDA DE INFORMAÇÃO DE UMA ORGANIZAÇÃO COM DIVERSOS SISTEMAS USANDO O APACHE AIRFLOW

Trabalho de Conclusão de Curso de Gradiação em Ciência da Computação da Universidade Estadual da Paraíba, como requisito à obtenção do título de Bacharel em Ciência da Computação.

Área de concentração: Engenharia de Dados.

Orientador: Prof. Dr. Fábio Luiz Leite Júnior

CAMPINA GRANDE- PB
2023

R696d Rodrigues, Renan Rey Costa.

Desenvolvimento de arquitetura de dados para suprimir demanda de informação de uma organização com diversos sistemas utilizando o *Apache Airflow* [manuscrito] / Renan Rey Costa Rodrigues. - 2023.

65 p. : il. colorido.

Digitado.

Trabalho de Conclusão de Curso (Graduação em Computação) - Universidade Estadual da Paraíba, Centro de Ciências e Tecnologia, 2023.

"Orientação : Prof. Dr. Fábio Luiz Leite Júnior, Coordenação do Curso de Computação - CCT. "

1. Engenharia de dados. 2. Gestão de dados. 3. Tomada de decisão. I. Título

21. ed. CDD 658.05

Elaborada por Talita M. A. Tavares - CRB - CRB 15/971 BC/UEPB

RENAN REY COSTA RODRIGUES

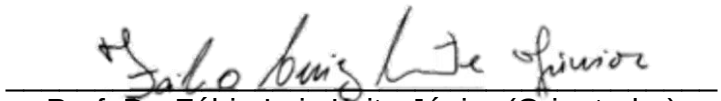
DESENVOLVIMENTO DE ARQUITETURA DE DADOS PARA SUPRIMIR DEMANDA DE INFORMAÇÃO DE UMA ORGANIZAÇÃO COM DIVERSOS SISTEMAS USANDO O APACHE AIRFLOW

Trabalho de Conclusão de Curso de Gradiação em Ciência da Computação da Universidade Estadual da Paraíba, como requisito à obtenção do título de Bacharel em Ciência da Computação.

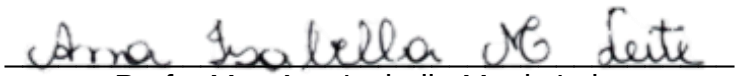
Área de concentração: Engenharia de Dados.

Aprovada em 10 de Julho de 2023.

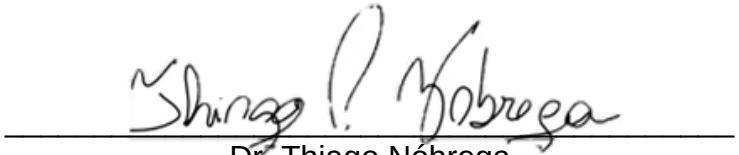
BANCA EXAMINADORA



Prof. Dr. Fábio Luiz Leite Júnior (Orientador)
Universidade Estadual da Paraíba (UEPB)



Profa. Msc Ana Isabella Muniz Leite
Universidade Estadual da Paraíba (UEPB)



Dr. Thiago Nóbrega
Universidade Federal de Campina Grande (UFCG)

AGRADECIMENTOS

A Deus, pois todo conhecimento, graça e sabedoria vem Dele, por meio Dele e para Ele.

A minha noiva, Beatriz Brito, cujo seu apoio incondicional e presença constante, mesmo para momentos de descanso, foram fundamentais na minha jornada até aqui. Você me inspira.

A minha família, em especial a minha mãe Helionalda e minha irmã Renaly, que, como guerreiras, me ajudaram e me inspiraram a passar por todas as dificuldades.

Ao professor e orientador, Fábio Luiz Leite Júnior, por ter contribuído na minha carreira acadêmica e profissional com seu vasto conhecimento e comprometimento com o curso.

A minha querida banca, por ter aceitado o convite e por toda ajuda, disponibilidade e atenção no desenvolvimento deste trabalho.

Aos professores do curso de Ciência da Computação, por todo conhecimento e contribuição para o meu crescimento pessoal e profissional.

Aos meus amigos, pois, os momentos de felicidades, vitórias e até dificuldades só se tornam reais quando são compartilhados.

“Uma lição indolor não tem sentido. Aquele que nada sacrifica, nada é capaz de obter” Edward Elric

RESUMO

O ambiente empresarial atual é caracterizado por dinamismo e multidisciplinaridade, o que torna a gestão de dados um desafio. A evolução constante desse ambiente exige o acompanhamento eficaz por meio da disponibilização de informações íntegras, de qualidade e de fácil acesso. Diante desse contexto, é necessário desenvolver instrumentos que extraiam informações de diversos sistemas, promovendo a transformação de dados em informações e conhecimento, a fim de fomentar o processo de tomada de decisão nas organizações. O objetivo deste trabalho é desenvolver uma arquitetura eficaz e escalável capaz de validar, anonimizar, processar e integrar dados provenientes de sistemas diversos. Para automatizar esses processos, utiliza-se o Apache Airflow, com a linguagem Python, e os serviços da Microsoft Azure. O escopo do trabalho envolveu a utilização de dados do Health Insurance Marketplace, que são publicamente disponibilizados e contêm informações sobre planos de saúde e odontológicos nos Estados Unidos. A arquitetura desenvolvida demonstrou eficácia no escalonamento do projeto, permitindo a adição de novas bases de dados ao processamento dos dados sem a necessidade de modificações. Isso tem implicações significativas para as organizações, pois proporciona uma gestão de dados mais eficiente, escalável e constante, fornecendo uma base sólida para a tomada de decisões fundamentada nas informações dos resultados obtidos.

Palavras-chave: engenharia de dados**;** processamento de dados**;** Apache Airflow.

ABSTRACT

The current business environment is characterized by dynamism and multidisciplinary, which makes data management a challenge. The constant evolution of this environment requires effective monitoring through the provision of reliable, quality, and easily accessible information. In this context, it is necessary to develop tools that extract information from various systems, promoting the transformation of data into insights and knowledge, to foster the decision-making process in organizations. The objective of this work is to develop an effective and scalable architecture capable of validating, anonymizing, processing, and integrating data from diverse systems. To automate these processes, Apache Airflow is used, along with the Python language and Microsoft Azure services. The scope of the work involved using data from the Health Insurance Marketplace, which is publicly available and contains information about health and dental plans in the United States. The developed architecture demonstrated effectiveness in scaling the project, allowing for the addition of new databases to the data processing without the need for modifications. This has significant implications for organizations as it provides more efficient, scalable, and consistent data management, providing a solid foundation for informed decision-making based on the obtained results.

Keywords: data Engineering**;** data Processing**;** Apache Airflow.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

ABFS	SISTEMA DE ARQUIVOS BLOB DO AZURE
ADLS GEN 2	AZURE DATA LAKE STORAGE GEN 2
AWS	AMAZON WEB SERVICES
BDM	BIG DATA MANAGEMENT
BENCS-PUF	BENEFICIOS COST SHARING
BR-PUF	BUSINESS RULES
CDE	ENTREGA CONTÍNUA
CI	INTEGRAÇÃO CONTÍNUA
CMS	CENTRO DE SERVIÇOS MEDICARE E MEDICAID
CSV	COMMA SEPARATED VALUES
CW-PUF	CROSSWALK
DAG	DIRECTED ACYCLIC GRAPH
DDP	PIPELINE DE PROCESSAMENTO DE DADOS
DI	DATA INTEGRATION
DL	DATA LAKE
DP	DATA PRIVACY
DS	DATA CONSUMPTION
DW	DATA WAREHOUSE
ERP	ENTERPRISE RESOURCE PLANNING
ETL	EXTRACT, TRANSFORM, LOAD
HDFS	HADOOP DISTRIBUTED FILE SYSTEM
IDS	INTEGRATED DATA STORAGE
JSON	JAVASCRIPT OBJECT NOTATION
KPIs	INDICADORES-CHAVE DE DESEMPENHO
NTWRK-PUF	NETWORK
ODS	OPERATIONAL DATA STORE
PDF	PORTABLE DOCUMENT FORMAT
PDS	PROCESSED DATA STORAGE

RDFS	RAW DATA FILE SYSTEM
RDS	RAW DATA STORAGE
SA-PUF	SERVICE AREA
SGBD	SISTEMA DE GERENCIAMENTO DE BANCO DE DADOS

SUMÁRIO

1 INTRODUÇÃO

O ambiente empresarial caracteriza-se pela sua natureza dinâmica e multidisciplinar. Nesse contexto, a disponibilidade de informações íntegras, de qualidade e de fácil acesso desempenha um papel essencial na fundamentação de tomadas de decisões embasadas em dados confiáveis. Contudo, apesar dos avanços alcançados ao longo dos anos, a garantia de uma disponibilização eficiente de dados e informações relevantes para o tomada de decisões continua sendo um dos principais desafios enfrentados pelas organizações (REGO, 2013; LEVINS, 2019).

Relembrando que o valor dos dados não se resume apenas à sua quantidade, mas reside na sua transformação progressiva em direção ao conhecimento (dados → informações → conhecimento). Essa progressão é de suma importância para a inteligência competitiva, o aumento da produtividade e a melhoria da qualidade nas organizações (RAUTENBERG, 2019; ESCOVEDO, 2020). Dados que não passam por esse processo de evolução podem se tornar inúteis e até mesmo prejudiciais para as empresas, como destacado por Oliveira (2022).

É importante ressaltar que essa evolução não se restringe apenas aos dados estruturados, ou seja, aqueles armazenados em bancos de dados ou arquivos de formato tabular. Estima-se que cerca de 80% dos dados das organizações estejam armazenados em formas não estruturadas. Muitos desses dados são volumosos e exigem uma abordagem especial para sua utilização efetiva (REGO, 2013; TAYEFI, 2021).

Desenvolver instrumentos para extrair informações de diversos sistemas, a fim de fomentar o conhecimento no processo de tomada de decisões, é de incontestável relevância na gestão dos dados. A complexidade em combinar dados na coleta de dados provenientes de múltiplas fontes (integração dos dados) e a união de bases históricas onde se encontra oscilação da sua estrutura ao longo do tempo, visando à realização de análises que apoiem a tomada de decisões, utilizando geralmente grandes quantidades de informação, de forma sistematizada ainda é algo desafiador e custoso para a maioria das empresas. Assim, uma boa gestão dos dados busca atender de forma ágil e eficiente às necessidades das diversas áreas de negócio como ciência, análise e visualização de dados (ESCOVEDO, 2020; TAYEFI, 2021).

Nesse sentido, o desenvolvimento de uma arquitetura de dados eficaz no escalonamento do projeto e capaz de adicionar novas base de dados ao processamento de dados sem a necessidade de modificações se torna crucial, com capacidade para validar, anonimizar, processar e integrar dados provenientes de diversos sistemas. Essa arquitetura deve ser flexível

o suficiente para permitir a inclusão ou remoção de unidades de sistema sem a necessidade de grandes modificações no fluxo, impulsionando a transformação de dados em conhecimento. Por meio da ferramenta de gerenciamento de fluxo de processos de engenharia de dados, o Apache Airflow, que possibilita a orquestração de automações e a execução de transformações complexas em nível programático, necessário para construção da ferramenta prosposta, e utilizando o Azure Data Lake Storage Gen2 (ADLS GEN2) como solução de armazenamento para grandes volumes de dados, foi possível desenvolver uma solução que integra dados de diferentes fontes, gerando informações úteis e agregando valor às empresas (HAEFFNER, 2022).

1.1 Objetivos

O objetivo deste trabalho é contribuir por meio de uma pesquisa exploratória, a utilização de conceitos estabelecidos na literatura para proporcionar uma maior compreensão da aplicação dos princiípios de ingestão, extração, limpeza e integradação de dados. Realizando a implementação de um processamento de dados utilizando serviços de Big Data, simulando a experiência de um projeto real, com dados fictícios provenientes de um e-commerce especializado em seguros de saúde.

1.1.1 Objetivos específicos

- Identificar os desafios e limitações do uso de dados de uma base de diversos fontes.
- Fornecer uma arquitetura de dados com etapas de validação, anonimização, processamento e integração dos dados provenientes de diversos sistemas.
- Construir um processamento de dados utilizando Apache Airflow e o Azure Data Lake Storage Gen2.
- Analisar os resultados da pipeline para aferir a eficácia da solução proposta.

2 FUNDAMENTAÇÃO TEÓRICA

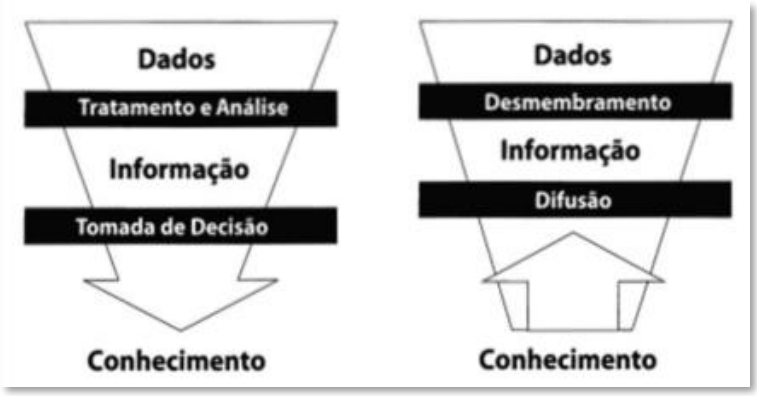
O objetivo deste capítulo é explorar conceitos fundamentais, assim como serviços para manipulação e armazenamento desses dados. A compreensão desses termos é essencial para contextualizar as definições e necessidades da solução proposta.

2.1 Gestão dos dados

Segundo a Oracle (2023), a gestão de dados consiste na prática de coletar, armazenar e utilizar dados de maneira segura, eficiente e econômica. Essa prática visa auxiliar pessoas, organizações e sistemas a consumir o uso de dados dentro dos limites legais, facilitando a tomada de decisões e ações que maximizem os benefícios para a organização. Reconhecer os dados como um dos ativos mais importantes nas empresas e a necessidade de gerenciá-los de forma eficaz, por meio de uma estratégia sólida, têm se tornado cada vez mais relevantes, uma vez que as empresas dependem cada vez mais de ativos intangíveis para criar valor (REGO, 2013; ORACLE, 2023).

A evolução de dados, informação e conhecimento, como exemplificado na Figura 1 é essencial para o objetivo de toda empresa: utilizar o conhecimento das informações para tomar decisões rápidas e precisas. O conceito "orientado a dados" refere-se a processos organizacionais que se baseiam não apenas em experiências, mas também na orientação pelos dados disponíveis (REGO, 2013; FÁVERO, 2017).

Figura 1 – Hierarquia entre dados, informações e conhecimento



Fonte: FÁVERO, 2017.

Considerando a crescente disponibilidade de grandes volumes de dados estruturados e não estruturados, torna-se essencial para os processos decisórios a validação dos dados, bem como sua rastreabilidade ao sistema origem (contextualização) e a capacidade de gerar *insights* embasados em informações precisas. A análise de dados possibilita uma compreensão mais

profunda e precisa, identificando padrões e correlações que sustentam a vantagem competitiva. Esse processo é denominado por Provost e Fawcett (2013) como Tomada de Decisão Guiada por Dados.

Apesar do conhecimento sobre esses aspectos ser antigo, a capacidade tecnológica para acompanhar o crescimento do ambiente empresarial, cada vez mais evoluído e dinâmico, ainda é limitada (REGO, 2013). Diante desse contexto, a gestão de dados enfrenta diversos desafios. De acordo com a Oracle (2023), esses desafios podem ser categorizados da seguinte forma:

1. Ausência de *insights* de dados: embora haja um aumento na quantidade de dados provenientes de diferentes fontes, eles não são visíveis se a organização não souber identificá-los, localizá-los e utilizá-los adequadamente.
2. Dificuldade em manter o desempenho no processamento de dados: à medida que as organizações coletam, armazenam e utilizam cada vez mais dados, é essencial garantir que o processamento ocorra dentro de um prazo adequado, levando em consideração a vida útil dos dados em sistemas de Big Data.
3. Desafios na consolidação de dados em constante mudança: é natural que as bases de dados, seus metadados, valores e regulamentações estejam em constante modificação, o que torna difícil a tarefa de consolidá-los de forma precisa e atualizada.
4. Necessidade de processamento e conversão eficientes dos dados: simplesmente coletar e identificar dados não é suficiente para agregar valor. É necessário processá-los e convertê-los em um formato adequado para análise, antes que percam seu potencial valor.
5. Constante necessidade de armazenamento eficiente dos dados: as organizações enfrentam o desafio de armazenar dados de forma eficiente, utilizando diferentes sistemas, data warehouses e data lakes.

Esses desafios refletem a complexidade e a importância da gestão de dados em um ambiente empresarial em constante transformação.

2.2 Formas dos dados

Os dados estruturados são informações organizadas e formatadas de acordo com um esquema predefinidio, o que facilita sua compreensão e manipulação. Os dados estruturados têm tipicamente estrutura de recursos repetitivos. A única diferença entre uma ocorrência de dados e outra está no conteúdo dos dados. Assim, seus registros são fáceis de lidar dentro de um sistema de gerenciamento de banco de dados (SGBDs), com definição clara dos seus

atributos, tipos de dados e relacionamentos. No entanto, os registros estruturados normalmente representam apenas uma pequena fração dos dados encontrados em uma corporação (FATIMA, 2016; LEVINS, 2019; TAYEFI et al., 2021).

Os dados não estruturados diferem dos dados estruturados pela ausência de uma organização predefinida, podendo abranger uma ampla gama de elementos, tais como fluxo de dados oriundos de equipamentos e sensores, informações provenientes de sites, dados de monitoramento, redes sociais, jogos, entre outros (FATIMA, 2016; LEVINS, 2019). Esses dados podem ser caracterizados por sua natureza diversa e formatos variados, incluindo vídeos, imagens, áudios e outros tipos de mídia. A crescente disponibilidade e a rápida geração de dados não estruturados têm desafiado as abordagens tradicionais de análise e armazenamento, exigindo o desenvolvimento de novas soluções e técnicas para lidar com sua complexidade e extrair valor dessas fontes de informação (SANTOS, 2022).

Há muita discussão do quanto os dados de uma corporação são estruturados ou não estruturados. Conforme mencionado por Mary Levins em seu livro "Data Architecture" (2019) e Dijcks (2013), há uma ampla gama de estimativas, variando de 2% a 20%. Essa variação substancial depende da natureza do negócio e da própria natureza dos dados utilizados para realizar tal apuração.

Figura 2 – Dados estruturados representam uma pequena parte dos dados gerados

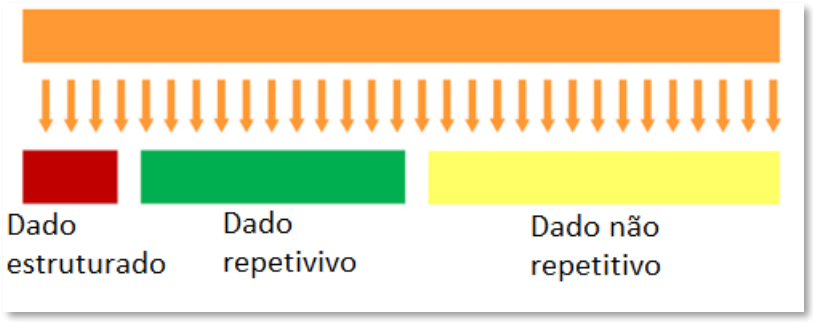
Fonte: LEVINS, 2019.

2.1.1 Dados repetitivos e não repetitivos não estruturados

Dentro de uma corporação, é possível identificar dois tipos básicos de dados não estruturados: dados repetitivos não estruturados e dados não repetitivos não estruturados. Um exemplo de dados repetitivos não estruturados pode ser encontrado em disparos de e-mails agendados por determinada função da empresa, enquanto e-mails trocados entre colaboradores são um exemplo de dados não repetitivos não estruturados. Cada e-mail pode apresentar diferentes características, como extensão variável, estar redigido em idiomas específicos e ser redigido de acordo com a vontade do autor. Essa natureza irregular dos dados não estruturados

dificulta sua adaptação e armazenamento em SGBDs, como mencionado por Levins (2019) em sua obra.

Figura 3 – Dados repetitivos e não repetitivos

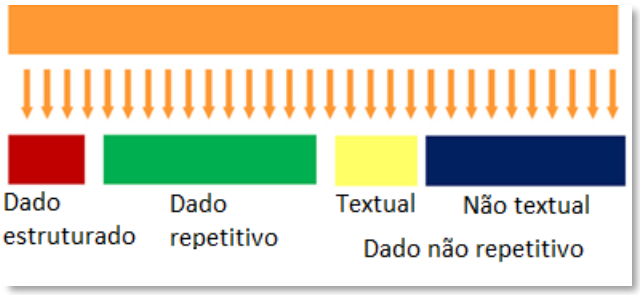


Fonte: LEVINS, 2019.

2.1.2 Dados textuais e não textuais

Os dados não repetitivos não estruturados ainda podem ser classificados em dados textuais e dados não textuais. Os dados textuais referem-se àqueles apresentados na forma de texto, como e-mails ou contratos judiricos, por exemplo. Por outro lado, os dados não textuais englobam elementos como imagens de perfil de usuários em uma plataforma ou plantas e diagramas (LEVINS, 2019).

Figura 4 – Dados textuais e não textuais



Fonte: LEVINS, 2019.

2.1.3 Importância do dado pelo seu formato

Existem várias perspectivas para considerar esses diferentes tipos de dados, mas a que se destaca é a relação com o valor comercial. Há diversos indícios que demonstram o alto valor comercial dos dados estruturados. Para o sucesso de um negócio, é crucial ter um saldo correto do fluxo de caixa, informações dos clientes e seus históricos de interações com a empresa. Em suma, cada ação do cliente é valiosa e desempenha um papel fundamental no empreendimento, e isso está diretamente relacionado à estruturação adequada dos dados, conforme ilustrado na Figura 5 abaixo:

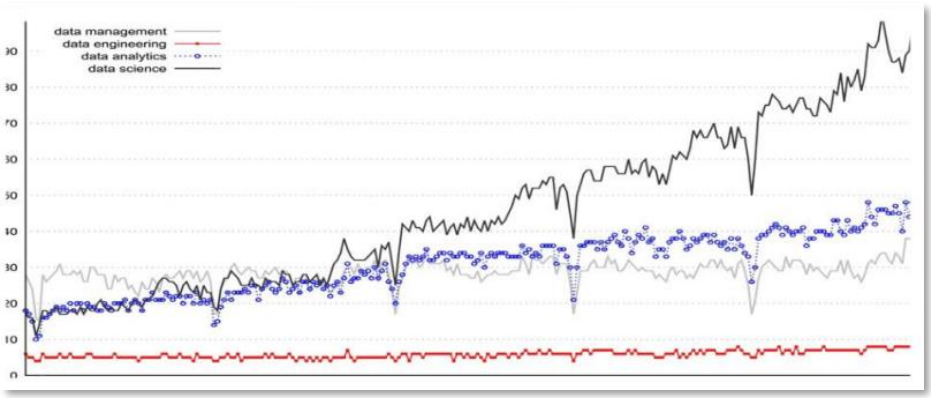
Figura 5 – Variação do valor de negócio para os diferentes tipos de dados

Fonte: LEVINS, 2019.

2.3 Extração, transformação e carregamento

A ciência de dados tem como objetivo extrair conhecimento e insights a partir de dados em diferentes formatos (estruturados, semiestruturados e não estruturados), utilizando algoritmos e ferramentas de análise (DHAR, 2013). Nesse contexto, os engenheiros de dados desempenham um papel fundamental, construindo ecossistemas que viabilizam o trabalho dos cientistas de dados, gerenciando o ciclo de vida dos dados, desde a coleta, integração e persistência, garantindo qualidade, eficiência e segurança. No entanto, apesar da importância destacada dos engenheiros de dados, o artigo de Romero, Wrembel e Song (2020) identifica uma baixa popularidade desses profissionais e das tecnologias relacionadas entre os usuários da internet, conforme ilustrado na Figura 6 abaixo:

Figura 6 – Popularidade dos tópicos de gerenciamento de dados

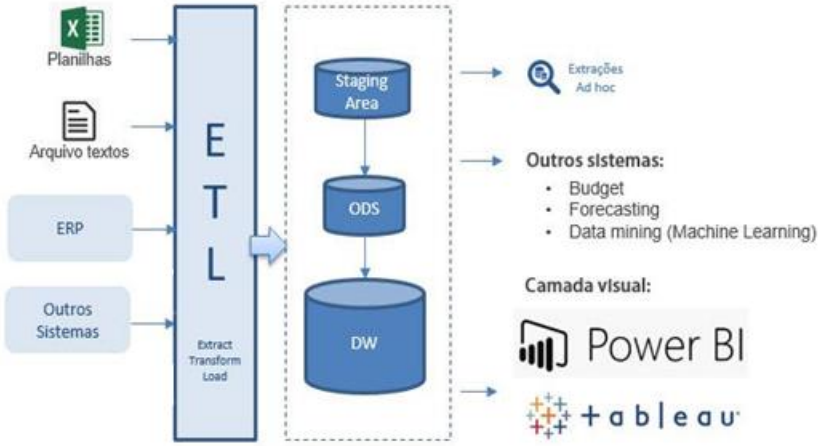


Fonte: ROMERO, WEMBEL E SONG, 2020.

Para automatizar o fluxo de processamento dos dados até alcançar seu objetivo final, são construídos os chamados Pipelines de Processamento de Dados (DPPs). Esses pipelines funcionam como uma orquestra, executando uma sequência de atividades simples e complexas que realizam diversas manipulações dos dados desde sua origem até sua finalidade desejada. Assim, tendo finalidade desde o armazenamento dos dados até sua aplicação em ferramentas de visualização de dados e modelos de Machine Learning (RAJ, BOSCH. et. al., 2020).

A sigla ETL (Extração, Transformação e Carregamento) representa um conjunto fundamental de etapas no processamento de dados. Como exemplificado na Figura 7, essas etapas envolvem a coleta de dados provenientes de diversas fontes, a transformação desses dados em um formato padronizado e o carregamento dos mesmos em um local específico para armazenamento e análise. Por meio desse processo, os dados passam por operações de processamento, limpeza, enriquecimento, integração e organização, de modo a torná-los adequados para análises posteriores mais eficientes e estratégicas. O objetivo principal é transformar os dados em informações de valor, demandando tarefas distintas e envolvendo diferentes áreas (JO & LEE, 2019; Garcia, 2020).

Figura 7 – Exemplo de Aplicação de ETL



Fonte: ITIGO, 2023.

Ao detalhar os componentes da Figura apresentada anteriormente, observamos a coleta de dados provenientes de diversas fontes, tais como planilhas, arquivos de texto, sistemas de Planejamento dos Recursos da Empresa (ERP) e outros sistemas. A etapa de ETL é responsável por processar esses dados de acordo com as necessidades da organização. Geralmente, os dados são inicialmente armazenados em uma área de armazenamento intermediário, conhecida como Staging Area, onde são preparados para uso. Em seguida, ocorre o Armazenamento dos Dados Operacionais (Operational Data Store - ODS), que possibilita a geração de relatórios operacionais. Por fim, o ODS é utilizado como uma das fonte de dados para o Data Warehouse (DW), um repositório centralizado que consolida informações sobre as atividades da organização (ITIGO, 2023; SNOWFLAKE, 2023)

Todos esses dados são extraídos para atender às necessidades específicas dos clientes (Ad hoc), bem como para alimentar outros sistemas, como orçamentos (Budget), modelos de previsões (Forecasting) e mineração de dados (Data mining). Além disso, esses dados são

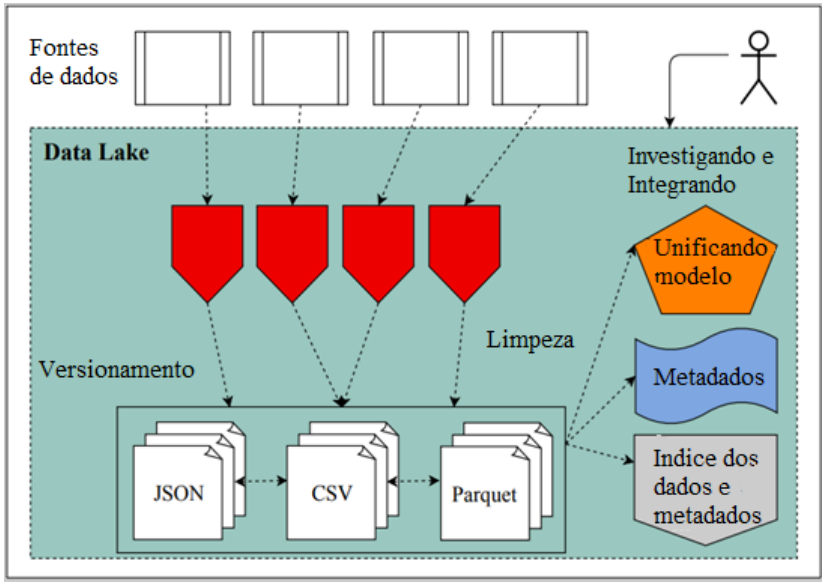
utilizados na criação de painéis de controle com ferramentas como Power BI e Tableau, permitindo a visualização dos dados e a geração de insights (ITIGO, 2023; POWERBI, 2023).

2.4 Data lake

O Data Lake (DL) é um repositório de dados que pode ser armazenado em diversos sistemas de armazenamento. Ele é capaz de armazenar dados de diferentes formatos, como estruturados, semi-estruturados e não estruturados, e pode conter metadados relevantes, como no exemplo da Figura 9. Além disso, o DL permite o uso de diferentes formatos para descrever esses metadados e possibilita mudanças autônomas ao longo do tempo. Essa abordagem oferece uma visão lógica dos dados em sua forma bruta, permitindo acesso, compreensão e extração de conhecimento de maneira conveniente para a ciência de dados (NARGESIAN, 2019; SAWADOGO & DARMONT, 2021).

Na Figura 8 apresentada, podemos observar que os dados têm origem em diversas fontes, como conexões de bancos de dados e sistemas de arquivos (Data Sources). Esses dados passam por um processo de ingestão, extração, versionamento e limpeza dentro do Data Lake. Nesse contexto, os arquivos podem ser armazenados no Data Lake em diferentes formatos, como JSON, CSV ou parquet. Finalmente, Com base nas necessidades de exploração e integração dos dados por parte dos usuários, ocorre a exploração dos metadados dos arquivos, bem como de seus índices e dados. Essas informações são utilizadas para a criação de modelos e diversos outros propósitos relacionados à análise e utilização dos dados (NARGESIAN, 2019). .

Figura 8 – Exemplo de Sistema de Gerenciamento de um Data Lake



Fonte: NARGESIAN, 2019.

De acordo com SHARMA (2018), a arquitetura de um DL pode ser dividida em zonas ou camadas, dependendo do nível de refinamento dos dados. Essa abordagem de arquitetura de referência pode ser adotada pelas organizações para seguir as melhores práticas. A visualização dessas zonas permite compreender os diferentes componentes e tecnologias envolvidos em cada processo e etapa, fornecendo uma estrutura que facilita a criação de modelos eficientes para atender às necessidades específicas de cada solução. Contribuindo para essa divisão das zonas, NARGESIAN (2019) propõe um modelo que estabelece:

- Ingestão de dados: processo de captura e aquisição de uma variedade de sistemas para serem armazenadas no data lake. A ingestão geralmente é realizada de forma paralela e com baixa latência, priorizando a velocidade sobre a análise profunda dos dados. No entanto, há verificações básicas nos dados e seus metadados para organizar os conjuntos de dados adquiridos;
- Extração de dados: transforma os conjuntos de dados brutos da ingestão de dados em um modelo de dados predefinido. Geralmente a extração ocorre em um arquivo por vez e podem estar relacionada com tarefas como a integração e limpeza do dado;
- Limpeza de dados: é uma área desafiadora que envolve o refinamento e a melhoria das informações armazenadas, com base em esquemas corretos e restrições de integridade dos dados. Nesse processo, são aplicadas técnicas e métodos para corrigir erros, eliminar duplicatas, tratar dados faltantes e padronizar formatos, garantindo a qualidade e confiabilidade dos dados para análises e tomadas de decisão;
- Gerenciamento de metadados: para evitar que um data lake se torne um "pântano de dados", é necessário categorizar e organizar os dados por meio da construção de metadados, como catálogos de dados. Esses catálogos desempenham um papel fundamental no entendimento e descoberta dos dados e sua integração nos data lakes;
- Integração de dados: é o processo de identificar conjuntos de dados relevantes e combiná-los de maneira significativa. No contexto dos data lakes, a descoberta e integração de dados estão intrinsecamente relacionadas, mas apresentam limitações. Para lidar com a variedade e qualidade dos dados, são desenvolvidas técnicas de integração que são adaptadas sob demanda.

No contexto das opções de ambiente discutidas, surge um modelo conhecido como on-premise, no qual ocorre a utilização de servidores internos para armazenamento de big data. Nesse caso, o repositório de dados volumosos é hospedado na infraestrutura da própria organização, conferindo-lhe a responsabilidade pela administração, controle e manutenção

desses recursos. O gerenciamento dessa infraestrutura requer um investimento significativo em termos de tempo e recursos humanos especializados para garantir a operação, segurança e confidencialidade dos dados nesse ambiente (DIAMOND, 2020; SHARMA, 2016).

2.5 Serviço de computação em nuvem

Considerando a relevância do uso de serviços de computação em nuvem neste projeto, foi selecionado um provedor entre os principais as três principais (*Google Cloud, Amazon Web Services* e *Microsoft Azure*) para o desenvolvimento das etapas subsequentes. Embora seja possível utilizar diferentes serviços de provedores distintos em uma mesma arquitetura, essa abordagem acarreta complexidade tanto na escolha e compatibilidade desses serviços, quanto nas configurações de segurança necessárias em um ambiente heterogêneo (HAEFFNER, 2022).

Dessa forma, o provedor mais compatível com o projeto foi o Microsoft Azure, fundamentado em alguns motivos essenciais. Destaca-se a maior familiaridade do autor com essa plataforma de nuvem, além do próprio projeto real que serviu como base para a concepção desta monografia ter sido desenvolvido em ambiente Azure. Ademais, a possibilidade de utilizar a conta estudantil fornecida pela Microsoft para alunos de universidades e escolas também contribuiu para essa escolha. É importante ressaltar que todos os serviços em nuvem empregados neste trabalho foram considerados dentro dos limites e condições impostos pela Microsoft Azure. Contudo, é relevante mencionar que existem ferramentas semelhantes ou equivalentes disponíveis nas três principais plataformas de nuvem em análise (HAEFFNER, 2022; MICROSOFT, 2023).

2.5.1 Azure data lake storage gen 2

O Microsoft Azure disponibiliza uma ampla gama de serviços destinados ao armazenamento de dados, abrangendo diversas estruturas. Esses serviços atuam como contêineres lógicos, permitindo a associação e agrupamento de vários recursos em um grupo existente. Dentre tais serviços, destaca-se o ADLS GEN2, que se destina à análise de big data, sendo construído sobre a plataforma Azure Blob Storage. Essa solução de armazenamento de dados apresenta notável escalabilidade e flexibilidade, especialmente projetada para acomodar grandes volumes de dados, independentemente de sua estrutura. Além disso, ao ser construído sobre o armazenamento de Blob, também se beneficia de camadas de armazenamento de baixo custo, juntamente com recursos robustos de alta disponibilidade e recuperação de desastres (MICROSOFT, 2023).

Este serviço é fundamentado no Azure Blob Storage, que é uma forma de armazenamento que não impõe restrições quanto aos tipos de dados aceitos, permitindo a inclusão de diversos formatos, tais como documentos PDF, imagens, arquivos JSON, CSV e vídeos, entre outros (OUSSOUS, 2018; MICROSOFT, 2023).

O ADLS GEN2 foi projetado principalmente para funcionar como o Hadoop e todas as estruturas que usam o Apache Hadoop Distributed File System (HDFS) como sua camada de acesso a dados. As distribuições do Hadoop incluem o driver ABFS (Sistema de Arquivos Blob do Azure), que permite que muitos aplicativos e estruturas acessem os dados do arquitetura de dados de etlArmazenamento de Blob do Azure diretamente. Assim o DL atribuí das principais características do Hadoop, como o sistema de arquivos distribuídos, o usuário adicionar módulos adicionas conforme a necessidade, desempenho, confiabilidade, escalabilidade e segurança (OUSSOUS, 2018; MICROSOFT, 2023).

2.6 Arquitetura de dados

A arquitetura de dados refere-se à estrutura que tem como objetivo gerenciar e manipular os dados em um ambiente específico (KALIPE & BEHERA, 2019). Ela abrange a definição das diferentes camadas e compontes envolvidos no processo de ingestão, processamento, visualização e análise de dados. A arquitetura de dados é projetada para garantir a integridade, eficiência e escalabilidade das operações relacionadas aos dados. Diversas empresas líderes em tecnologia, como IBM, Oracle e Microsoft, têm publicado arquiteturas de referência e eficientes para suprir os diversos requisitos dos seus negócios. Várias dessas abordagens foram utilizadas por pesquisadores na tentativa de criar uma arquitetura de referência que pudesse ser aplicada em diversos setores e em uma ampla gama de casos de uso (KALIPE & BEHERA, 2019; LEVINS, 2019).

Por exemplo, Pekka e Daniel (2015) propuseram uma arquitetura independente de tecnologia baseada no estudo de sete casos de uso de Big Data em empresas de tecnologia de ponta, como Facebook, LinkedIn e Netflix. Outros autores seguiram abordagens semelhantes ao propor uma arquitetura de referência de cinco camadas. Por exemplo, Mert (2017) examinaram mais de 19 milhões de projetos no banco de dados do GitHub e a lista de projetos da Apache Software Foundation em busca de projetos relacionados a Big Data.

A definição das arquiteturas de dados é uma tarefa que envolve a consideração de diversos fatores, como o tempo, o volume dos dados, a dependência de plataforma e os requisitos específicos do modelo de armazenamento. Além disso, aspectos cruciais, como a

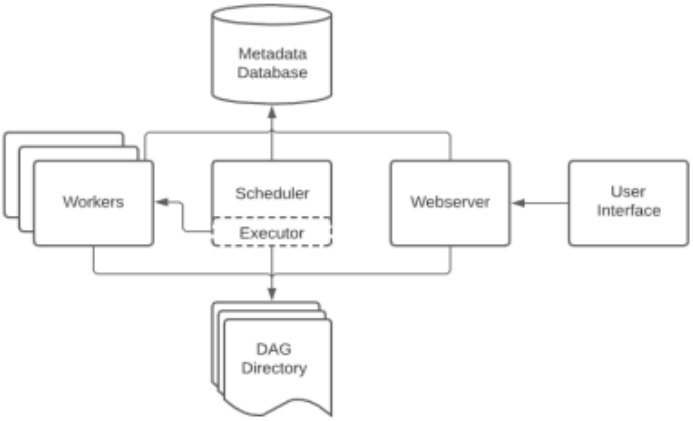
segurança em ecossistemas de Big Data, são abordados durante esse processo. A indústria oferece várias propostas de arquiteturas específicas, desenvolvidas para atender aos requisitos de casos de uso especiais em setores como telecomunicações, saúde, segurança de redes de comunicação, redes elétricas, ensino superior e universidades. Geralmente, essas propostas aproveitam, total ou parcialmente, as camadas definidas nas arquiteturas de referência comuns, adaptando os componentes e ferramentas utilizados de acordo com as necessidades específicas dos requisitos do caso de uso em questão (KALIPE & BEHERA, 2019; LEVINS, 2019; PETRILLO, 2020).

Entre as diversas arquiteturas de referência amplamente reconhecidas, a presente monografia compartilha semelhanças com a Arquitetura Zeta. Essa abordagem propõe uma solução tecnológica que se integra diretamente à arquitetura de negócios/sistemas da organização. Dessa forma, qualquer nova aplicação necessária para as operações da empresa pode ser facilmente incorporada a essa arquitetura. A Arquitetura Zeta oferece etapas de processos isolados, nos quais o software pode ser executado e integrar novos dados de maneira conveniente. Uma vantagem significativa é que o hardware não é especificamente dedicado a um conjunto particular de serviços, mas é compartilhado por todo o sistema, permitindo um uso mais eficiente e uma alocação mais fácil para atender às necessidades requeridas (Kalipe & Behera, 2019).

2.7 Airflow

Conforme a definição fornecida pelo próprio site oficial do Apache Airflow (2023), esta ferramenta foi desenvolvida em Python e utiliza scripts da linguagem para criar e gerenciar workflows. Inicialmente concebido no ambiente do Airbnb, por volta de 2014, para lidar com o fluxo complexo de dados da empresa, o Airflow foi posteriormente adotado pela Apache Foundation em 2016. Como um projeto de software livre, ele alcançou grande popularidade e se tornou um dos projetos mais proeminentes da organização, alcançando o status de *Top-Level Project* em 2019.

Figura 9 – Arquitetura do Airflow



Fonte: AIRFLOW, 2023.

Os componentes-chave do Airflow para a execução de tarefas em uma DAG são:

- DAG (Directed Acyclic Graph): responsável por coletar e organizar a sequência de tarefas, gerenciando suas dependências e relacionamentos;
- *Webserver*: fornece uma interface visual para os usuários, permitindo acompanhar o progresso das tarefas e visualizar logs de execução para facilitar a resolução de problemas;
- *Scheduler*: monitora e sincroniza as ações das tarefas e DAGs. Verifica periodicamente se houve alterações no código das DAGs e prepara a execução das tarefas ativas;
- *Metadata Database*: armazena dados relacionados a cada instância de tarefa em uma DAG, como horário de execução, estado da tarefa, tempo e agendamento;
- *Executors*: são responsáveis pela execução das instâncias de tarefas;
- *Workers*: executam efetivamente as tarefas em si.

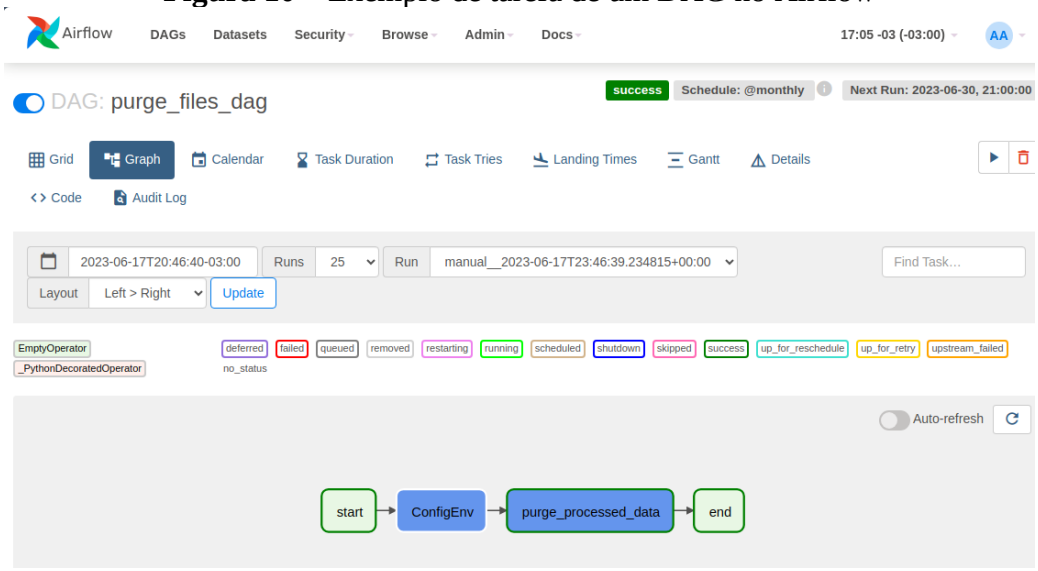
A integração de dados no Airflow é implementada pelo engenheiro de dados por meio de código, seguindo uma abordagem lógica. Todas as etapas do processo são cuidadosamente planejadas e modeladas de acordo com as regras estabelecidas, permitindo a configuração de fluxos contínuos ou paralelos, conforme necessário (AIRFLOW, 2023).

A Figura 10 apresenta a representação visual de um DAG, acessível através do servidor web, exibindo sua sequência de tarefas definidas. Nela, é possível observar diversas informações relevantes. Por exemplo, é apresentada a definição do agendador, configurado como mensal (*monthly*), juntamente com a data da próxima execução, o ID da DAG "purge_files_dag" e seus respectivos status. Além disso, são disponibilizadas opções de

visualização do processo em formato de grade (*Grid*), histórico de execuções com suas respectivas durações, registros de log e até mesmo o código-fonte que originou a DAG.

O processo de exemplo começa com a tarefa genérica "*start*" e finaliza com a tarefa "*end*". Ambas são *DummyOperators* e não executam nenhum algoritmo em si, servindo apenas como marcos visuais das etapas do processo. As tarefas "*ConfigEnv*" são responsáveis por adicionar e definir variáveis no código Python, conforme evidenciado pela ausência de alteração de cor durante a sua execução. Por sua vez, a tarefa "*purge_processed_date*" executa um algoritmo em Python, podendo estar encadeada a outros algoritmos ou sendo executada em paralelo. Exemplos concretos dessas situações serão apresentados nos resultados deste trabalho acadêmico.

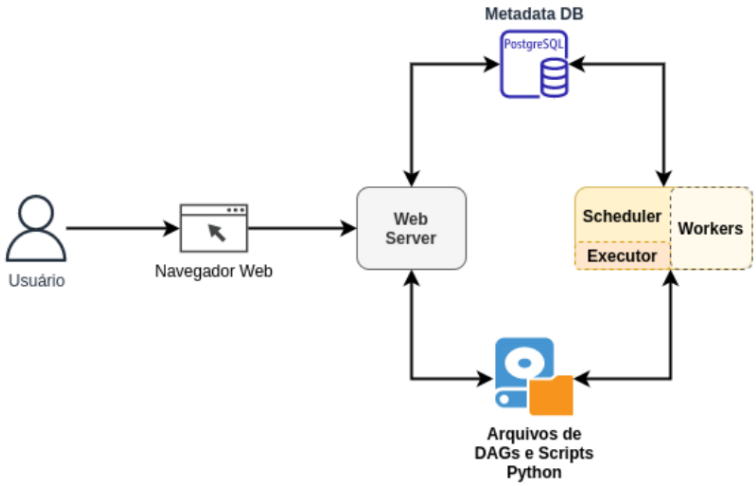
Figura 10 – Exemplo de tarefa de um DAG no Airflow



Fonte: Elaborado pelo autor, 2023.

A Figura 11 ilustra a relação entre os componentes fundamentais do Airflow, todos configurados em uma máquina local. É importante ressaltar que o Airflow não deve ser utilizado como um processador completo de dados ou como uma unidade de processamento que compartilha os dados processados em memória. Isso se deve ao fato de o banco de dados de metadados, responsável pela troca de informações entre as tarefas, não ser otimizado para lidar com uma alta demanda de dados. No entanto, ainda é possível transmitir informações relevantes dos metadados entre as tarefas, como resultados, erros e outras informações do fluxo (HAEFFNER, 2022).

Figura 11 – Relação entre os containers do Apache Airflow em configuração padrão



Fonte: HAEFFNER, 2022

Considerando que existem limitações de capacidade de processamento, é importante destacar que esses limites podem ser facilmente alcançados ao lidar com grandes volumes de transferência de informações e interações complexas entre tarefas. Mesmo ao utilizar o Airflow apenas como um orquestrador, é possível modularizar seus componentes, com exceção do banco de dados de metadados, a fim de permitir uma utilização escalável sob demanda por meio de contêineres, como foi implementado neste projeto. Essa abordagem de modularização contribui para lidar de forma mais eficiente para integrar sistemas complexos, garantindo a flexibilidade e a escalabilidade necessárias para atender às demandas do projeto (ZOU, 2021; HAEFFNER, 2022).

3 METODOLOGIA

Este estudo tem como propósito contribuir por meio de uma pesquisa qualitativa de carater exploratório, utilizando os conceitos estabelecidos por Raul Wazlawick (2009) para proporcionar uma maior compreensão da aplicação dos princípios de ingestão, extração, limpeza e integração de dados. Será realizada a implementação de um processamento de dados utilizando serviços de Nuvem, simulando a experiência de um projeto real, com dados fictícios provenientes de um e-commerce especializado em seguros de saúde. Assim, destaca-se a relevância da engenharia de dados no contexto das soluções tecnológicas corporativas, ressaltando seu valor como alicerce fundamental para o estabelecimento de uma solução robusta baseada em dados.

Neste projeto, será proposto e desenvolvido uma arquitetura de dados eficaz e escalável, com o objetivo de validar, processar, integrar dados provenientes de diferentes sistemas. Serão analisados diversos aspectos dessa arquitetura, como sua eficácia no escalonamento do projeto e a capacidade de adicionar novas bases de dados à pipeline de dados sem a necessidade de modificações. Para automatizar esses processos, foi utilziado a combinação do Apache Airflow, utilizando a linguagem Python, juntamento com os serviços da Microsoft Azure. Dessa forma, a arquitetura desenvolvida apresenta-se como uma solução tecnológica robusta e adaptável, proporcionando uma gestão de dados mais eficiente e contribuindo para o avanço das práticas de tomada de decisões fundamentada no dados.

3.1 Aquisição dos dados

Para viabilizar a exploração prática dos conceitos da engenharia de dados e dos processos de ETL, foi realizado um estudo de caso baseado em uma experiência real vivenciada pelo autor deste trabalho durante sua graduação. No entanto, devido à natureza sigilosa e interna dos dados da empresa parceira envolvida no referido projeto, foi necessário criar um protótipo com novas bases para ilustrar o desenvolvimento do pipeline.

Para tal propósito, foi empregado um conjunto de dados disponibilizado gratuitamente pela Kaggle, uma comunidade online de cientistas de dados afiliada à Google, que disponibiliza diversos conjuntos de dados para fins educacionais. Optou-se por utilizar o conjunto de dados Health Insurance Marketplace, que consiste em dados públicos sobre planos de saúde e odontológicos oferecidos a indivíduos e pequenas empresas por meio desta empresa (KAGGLE, 2017; KAGGLE, 2022; CMS, 2023).

3.2 Descrição dos dados

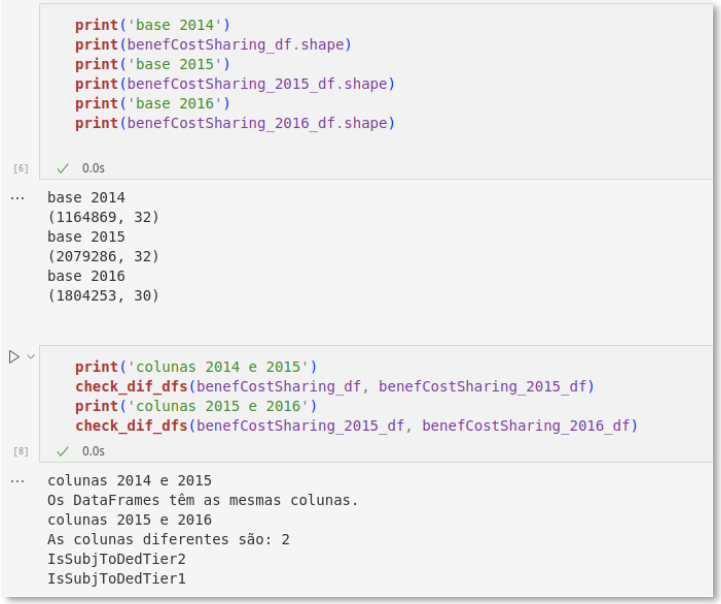
Foi necessário realizar um levantamento de informações sobre as bases de dados que seriam incorporadas ao fluxo, visando antecipar possíveis comportamentos e estabelecer validações apropriadas. A exploração dos dados é iniciada por meio de uma organização cuidadosa e a formulação de um plano para identificar seus padrões. Com o intuito de auxiliar na compreensão das bases de dados, foi feito uma análise dos dados com um script de notebook Python. Esse recurso foi criado para fornecer atualizações contínuas, descrevendo as colunas e as características comuns dos campos, tais como nome, definição, tipo e comprimento dos dados. Para um projeto contínuo, essa atualização é essencial a fim de acompanhar possíveis mudanças nas informações e garantir a relevância dos dados.

Conforme ilustrado na Figura 12 abaixo, após a compreensão das estruturas das bases de dados históricas do sistema BenCS-PUF, é realizada a execução da função *"check_dif_dfs"*, responsável por verificar as diferenças nas colunas dos dataframes fornecidos. A análise revelou que, a partir da base de 2016, houve a redução dos campos *"IsSubjToDedTier2"* e *"IsSubjToDedTier1"* no sistema. Essa informação é de extrema relevância para todo o fluxo de dados subsequente, pois auxilia na mitigação de possíveis erros.

Após realizar o mesmo procedimento em todos os outros sistemas utilizados, foram identificadas as seguintes oscilações históricas:

- BR-PUF: a coluna DentalOnly foi renomeada para *"DentalOnlyPlan"* em 2016;
- Plan-PUF: ocorreram 75 modificações nos campos entre os anos 2015 e 2016 nas bases de dados;
- Rate-PUF: estrutura manteve-se consistente ao longo de todos os anos;
- Ntwrk-PUF: a coluna *"DentalOnly"* foi renomeada para *"DentalOnlyPlan"* em 2016;
- SA-PUF: a coluna *"DentalOnly"* foi renomeada para *"DentalOnlyPlan"* em 2016.

Figura 12: Trecho do script de análise utilizado para conhecer as estruturas das bases de dados



```
print('base 2014')
print(benefCostSharing_df.shape)
print('base 2015')
print(benefCostSharing_2015_df.shape)
print('base 2016')
print(benefCostSharing_2016_df.shape)
```

```
[6]  ✓ 0.0s
```

```
base 2014
(1164869, 32)
base 2015
(2079286, 32)
base 2016
(1804253, 30)
```

```
print('colunas 2014 e 2015')
check_dif_dfs(benefCostSharing_df, benefCostSharing_2015_df)
print('colunas 2015 e 2016')
check_dif_dfs(benefCostSharing_2015_df, benefCostSharing_2016_df)
```

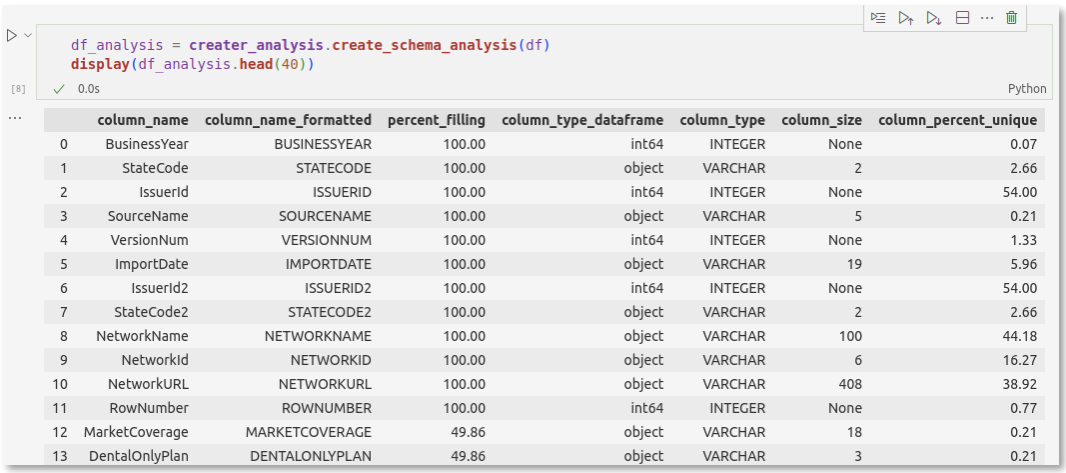
```
[8]  ✓ 0.0s
```

```
colunas 2014 e 2015
Os DataFrames têm as mesmas colunas.
colunas 2015 e 2016
As colunas diferentes são: 2
IsSubjToDedTier2
IsSubjToDedTier1
```

Fonte: Elaborado pelo autor, 2023.

Outra análise fundamental utilizado na fase de familiarização com as bases de dados foi feito em outro script de análise, conforme exemplificado no trecho da Figura 13. Esse notebook desempenha dois papéis essenciais. Para análise das bases fornecidas, com foco nas variações, especialmente nos nomes das colunas, percentual de preenchimento, percentual de preenchimento exclusivo, tipo de dados das colunas, tamanho das colunas de texto, valores a serem considerados para a anonimização e possíveis chaves primárias.

Figura 13 – Trecho do script de análise utilizado para conhecer detalhadamente as bases



```
df_analysis = creater_analysis.create_schema_analysis(df)
display(df_analysis.head(40))
```

```
[8]  ✓ 0.0s  Python
```

	column_name	column_name_formatted	percent_filling	column_type_dataframe	column_type	column_size	column_percent_unique
0	BusinessYear	BUSINESSYEAR	100.00	int64	INTEGER	None	0.07
1	StateCode	STATECODE	100.00	object	VARCHAR	2	2.66
2	IssuerId	ISSUERID	100.00	int64	INTEGER	None	54.00
3	SourceName	SOURCENAME	100.00	object	VARCHAR	5	0.21
4	VersionNum	VERSIONNUM	100.00	int64	INTEGER	None	1.33
5	ImportDate	IMPORTDATE	100.00	object	VARCHAR	19	5.96
6	IssuerId2	ISSUERID2	100.00	int64	INTEGER	None	54.00
7	StateCode2	STATECODE2	100.00	object	VARCHAR	2	2.66
8	NetworkName	NETWORKNAME	100.00	object	VARCHAR	100	44.18
9	NetworkId	NETWORKID	100.00	object	VARCHAR	6	16.27
10	NetworkURL	NETWORKURL	100.00	object	VARCHAR	408	38.92
11	RowNumber	ROWNUMBER	100.00	int64	INTEGER	None	0.77
12	MarketCoverage	MARKETCOVERAGE	49.86	object	VARCHAR	18	0.21
13	DentalOnlyPlan	DENTALONLYPLAN	49.86	object	VARCHAR	3	0.21

Fonte: Elaborado pelo autor, 2023.

O segundo objetivo fundamental deste notebook consistia em desempenhar o papel de auxiliar na criação de arquivos JSON que seriam utilizados nas etapas subsequentes da pipeline.

Esses arquivos continham parâmetros pré-definidos que especificam as características esperadas dos campos para cada área de negócio, incluindo a formatação dos nomes tratados, os tipos de dados resultantes do processamento e a identificação dos campos de chave primária.

3.3 Definição da arquitetura dos dados

A fim de viabilizar a construção do software, promover a documentação adequada e facilitar a comunicação entre as equipes, optou-se por adotar o modelo arquitetural C4 para cada componente do projeto, bem como suas respectivas interações. Essa arquitetura foi continuamente revisada ao longo do desenvolvimento, levando em consideração a evolução dos modelos de dados, avanços nos sistemas, implementação da pipeline e distribuição do data lake. Essas decisões foram baseadas nas necessidades do cliente e nos feedbacks das demais equipes envolvidas, culminando em uma arquitetura final ao término do projeto (C4 MODEL, 2023).

Optou-se por utilizar o serviço Apache Airflow para criação e orquestração das pipelines, bem como manipulação dos dados, contemplando desde a extração até a integração. Além disso, definiu-se o Blob Storage da Azure como o serviço de armazenamento a ser empregado, permitindo a aplicação dos conceitos de armazenamento de dados voltados para o ADLS GEN 2.

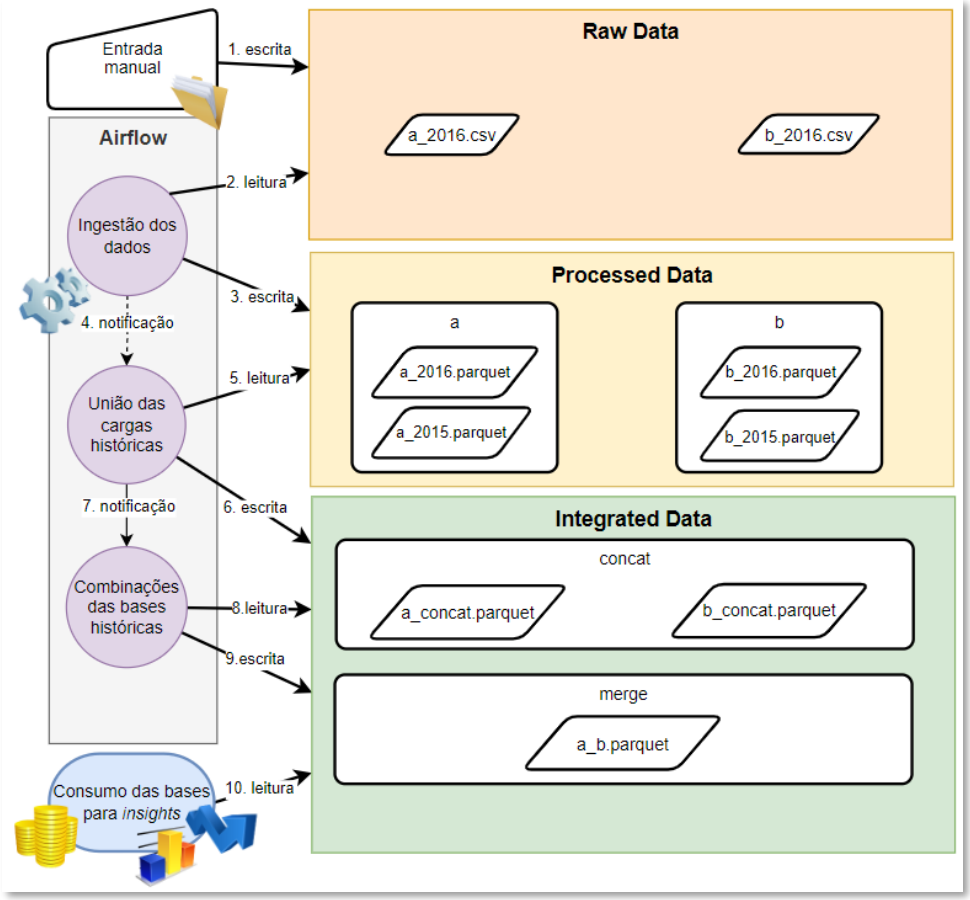
Para a organização dos dados, é adotada a divisão em zonas no Data Lake (DL), a fim de separar os dados com base em seu status de processamento. Com as zonas definidas, é estabelecido um fluxograma da pipeline de atividades , conforme exemplificado na Figura 14. Inicialmente, os arquivos são armazenados na área de entrada do DL, denominada "Raw Data", onde cada fonte de dados é responsável por designar seu esquema, seja em formato tabular ou semiestruturado. Após as etapas de validação de esquema, validação de dados e transformação de dados, os dados são armazenados na zona "Processed Data", com subpastas correspondentes às respectivas áreas de negócio.

Os dados nessa zona podem ser acessados individualmente ou combinados com outras áreas de negócio para análises Ad Hoc. Além disso, foi criada a zona "Integrated Data" para armazenar os dados concatenados de acordo com seus sistemas de origem, permitindo o consumo por outros sistemas e a camada visual.

Etapas do fluxograma da pipeline de atividades:

- Extração dos dados da Health Insurance Marketplace, disponibilizados de forma aberta na internet, e carga sem aplicação de transformação na zona bruta do Data Lake (1).
- Ingestão dos dados da zona bruta, realizando validação, anonimização e carregamento para a zona dos dados processados no Data Lake (2 e 3).
- União das bases históricas de acordo com seu sistema de origem e seleção dos dados-chave para integração na atividade posterior (5, 6 e 8).
- Leitura dos dados concatenados e combinados na zona de dados integrados. Os dados consolidados estão prontos para realização de análise e geração de insights (10).
- Chamada de uma DAG para outra para iniciar a sequência de processos (4 e 7).

Figura 14 – Fluxograma da pipeline e status dos dados em cada etapa do processo



Fonte: Elaborado pelo autor, 2023.

Dessa forma, a pipeline estabelece um fluxo estruturado e ordenado de atividades, garantindo a integração e processamento adequado dos dados ao longo do processo. Ao concluir a execução da pipeline de dados desenvolvida, que engloba todas as etapas de extração, armazenamento, transformação, integração e disponibilização dos dados, é possível avaliar o

sucesso da execução de todo o fluxo de atividades estabelecido. Esse resultado representa a conquista do objetivo final almejado.

4 RESULTADOS E DISCUSSÃO

Nesta seção, apresentamos uma análise detalhada do trabalho prático realizado utilizando o Apache Airflow e Microsoft Azure. O foco foi a criação de pipelines de dados abrangendo etapas como leitura, validação, anonimização, carregamento, concatenação e combinação dos dados provenientes do Health Insurance Marketplace. Além disso, são discutidos os resultados obtidos ao longo do processo, bem como os desafios enfrentados durante a execução do projeto.

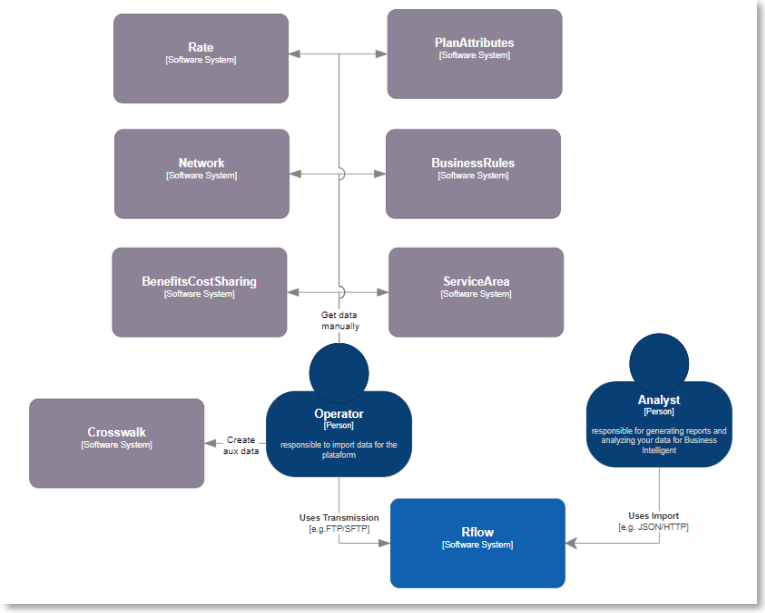
4.1 Construção de arquitetura e pipeline

Através do fluxograma da pipeline e o status das bases em cada etapa do processo, conforme ilustrado na Figura 15, foi possível visualizar a estrutura do sistema desde o contexto mais abrangente até os detalhes de suas partes constituintes. Dessa maneira, a arquitetura desenvolvida com o C4 Model resultou em uma representação visual precisa e organizada do sistema.

4.1.1 Contexto do sistema

Foi estabelecido o Contexto do Sistema, que descreveu as interações, relacionamentos e dependências entre o ambiente e o sistema em questão. Nesse estágio, foi feito com especial atenção à identificação dos potenciais sistemas externos com os quais ocorreram interações, bem como às pessoas e outras entidades externas que estarão presentes no contexto do sistema. Esses sistemas externos desempenham um papel essencial ao fornecer os dados necessários para o pipeline de dados. Os usuários do sistema interagem com ele por meio de dois cenários principais: (a) fornecendo dados ao ambiente e (b) gerando relatórios e analisando os painéis disponibilizados pela plataforma. Os elementos identificados podem ser observados na representação gráfica abaixo:

Figura 15 – Diagrama C4 em nível de contexto do sistema



Fonte: Elaborado pelo autor, 2023.

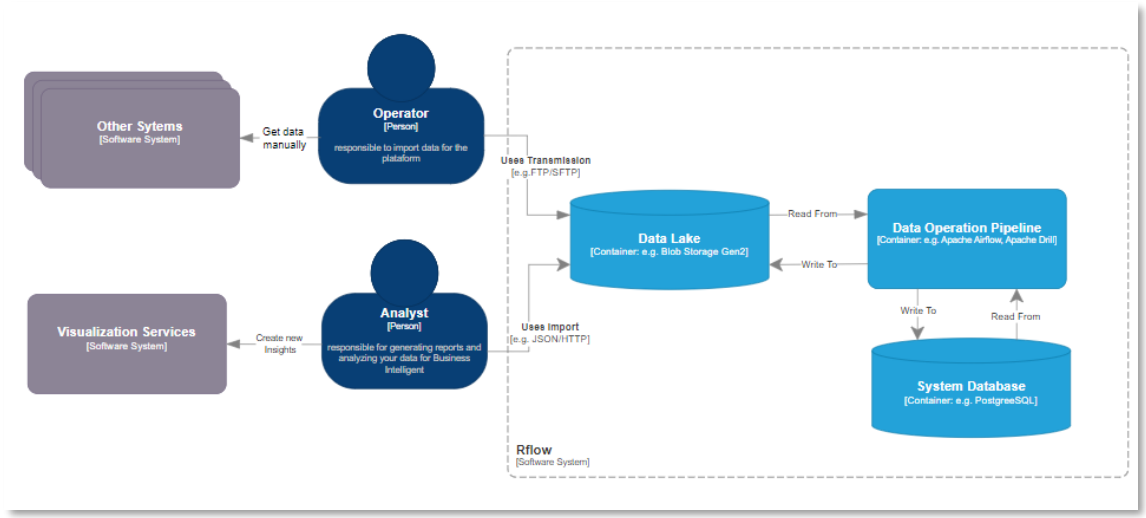
Os usuários identificados foram classificados em duas personas distintas: Operador e Analista. O Operador é responsável por carregar todos os dados que serão processados pelo sistema, tendo acesso aos sistemas externos e a capacidade de enviar os dados no formato CSV ou XLSX. Após o carregamento dessas informações, o Airflow executará uma rotina interna para processar esses dados na plataforma. Por outro lado, o Analista tem permissão para visualizar e criar relatórios com base nos dados tratados pela plataforma. Todos os dados analisados correspondem aos dados carregados na plataforma pelo Operador.

Os sistemas externos identificados são *Benefits and Cost Sharing*, *Rate*, *Plan Attributes*, *Network*, *Business Rules*, *Service Area* e *Crosswalk*, já descritos anteriormente. Esses dados serão integrados diretamente ao sistema, no entanto, considera-se que o Operador faça a manutenção e atualização desses dados além da disponibilização de *dumps* sobre os *datasets*.

4.1.2 Contêiner do sistema

Na segunda camada do modelo C4, o sistema Rflow representado na primeira camada foi refinado em contêineres que trabalham juntos para executar todas as funções necessárias. Na Figura 16 podemos observar um diagrama que inclui os sistemas externos, usuários do sistema e contêineres suportados pela plataforma Rflow.

Figura 16 – Diagrama C4 em nível de contêiner do sistema



Fonte: Elaborado pelo autor, 2023.

Antes de descrever o Data Lake construído é importante explicar sua entrada. A entrada do Data Lake foi definido como um arquivo exportado do sistema existente (por exemplo, *Business Rules*). No contexto de dados, foram usadas fontes de dados para designarem cada sistema e seu respectivo esquema de dados. Por exemplo, as fonte de dados *Business Rules* forneceram as regras de negócios de classificação, enquanto outro sistema como o *Service Area* forneceram as áreas geográficas do serviço. Os dados disponibilizados pelo Operador foram de formato tabular (arquivos CSV ou XLS) e foram utilizados dados semi estruturados para auxílio do sistema.

O Data Lake, elemento arquitetônico para o armazenamento dos dados brutos e pré-processados provenientes de diversas fontes. Inicialmente, o Operador carregava os dados brutos por meio de um serviço SFTP ou diretamente na plataforma Azure. Em seguida, o *Data Operation Pipeline* processava esses dados brutos, gerando uma versão pré-processada específica para cada base. Por fim, os dados pré-processados eram consolidados com base em seus sistemas de origem e foram realizados merges de serviços-chave para disponibilização dos dados às áreas de *Business Intelligence* e Ciência de Dados. Assim, permitindo uma melhor organização e preparação dos dados para análises e tomadas de decisão.

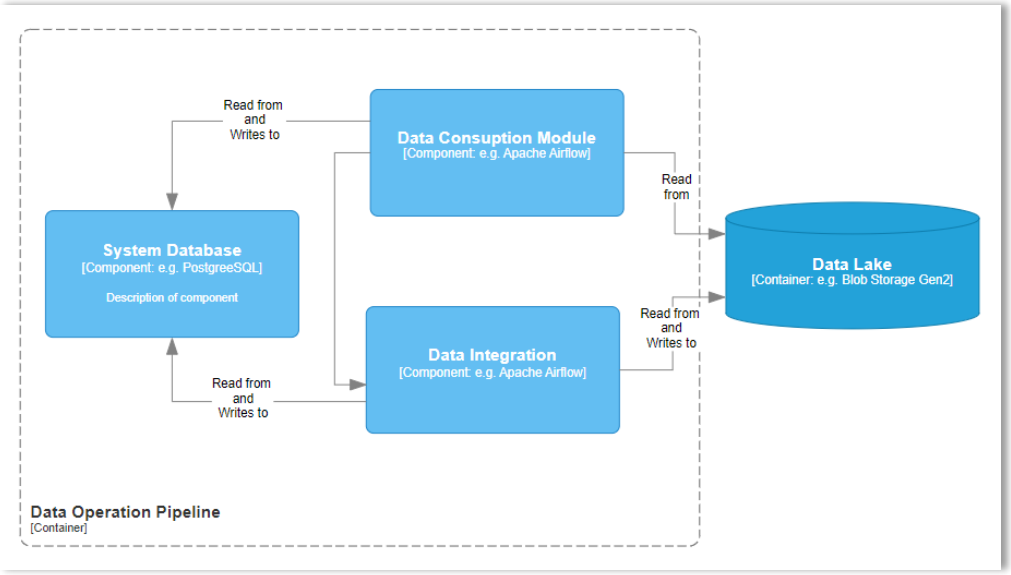
O *Data Operation Pipeline* desempenha na conversão dos dados brutos em um formato padronizado para cada fonte de dados. Além disso, esse componente oferece uma camada de privacidade para dados confidenciais dos usuários. Através do pipeline de operação de dados, também é fornecida uma camada de consolidação histórica das bases e integração para alinhar os dados provenientes de diferentes fontes. Esses dados são disponibilizados no *Data Lake* para serem consumidos pelo Analista.

4.1.3 Componentes do sistema

Na camada de componentes do sistema possibilitou a construção de uma visão detalhada dos componentes internos de cada contêiner, bem como das conexões com os bancos de dados correspondentes. O nível de detalhamento dos componentes possibilitou a definição das tecnologias utilizadas em sua implementação, incluindo a escolha da linguagem de programação, banco de dados, bibliotecas, entre outros aspectos relevantes.

Em resumo, destacam-se os serviços de *Data Lake Storage*, responsáveis pelo armazenamento dos arquivos, e o *Data Operation Pipeline*, responsável pela validação e disponibilização dos dados processados para os serviços de *Business Intelligence* e análise do *Health Insurance Marketplace*. Esses elementos desempenham um papel crucial no fluxo de dados, garantindo o armazenamento eficiente, a integridade dos dados e o fornecimento de informações para as análises e tomadas de decisão.

Figura 17 – Diagrama C4 em nível de componentes do sistema

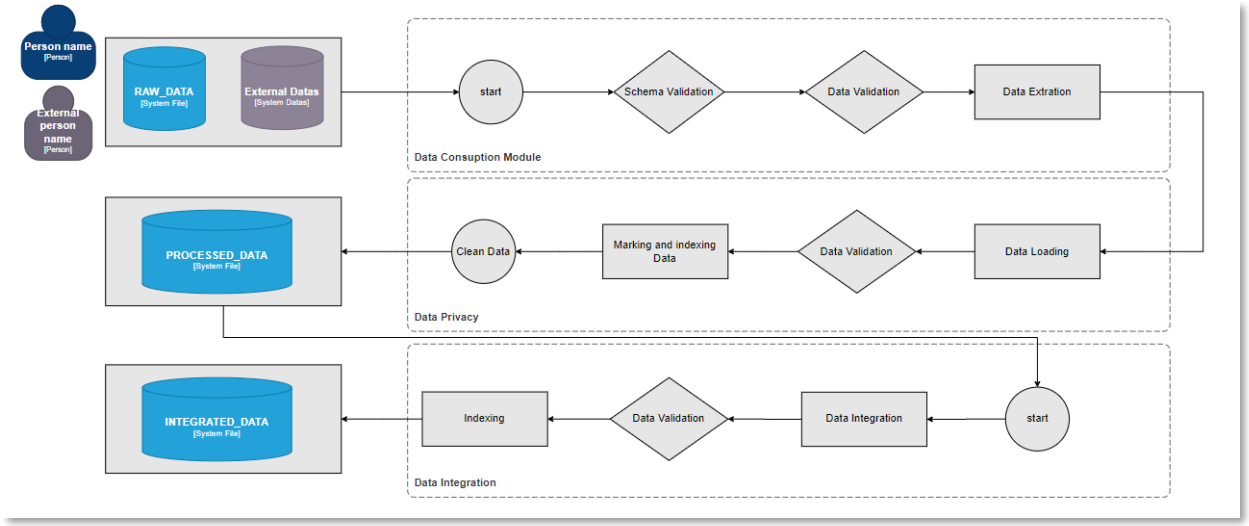


Fonte: Elaborado pelo autor, 2023.

4.1.4 Data operation pipeline components

Foi criado um diagrama adicional para orientar o comportamento da pipeline de dados, o qual não se enquadra nos níveis do C4 Model, mas é de extrema importância para acompanhar e compreender o fluxo de dados. O diagrama adicional da Figura 18 forneceu uma visão mais detalhada e abrangente do processo, contribuindo para uma compreensão completa do sistema e de sua operação.

Figura 18 – Data Operation Pipeline Components Diagrama



Fonte: Elaborado pelo autor, 2023.

Com o auxílio do diagrama adicional criado, podemos acompanhar o fluxo dos dados de maneira mais detalhada. Primeiramente, os dados brutos são carregados pelo Operador. Em seguida, o Data Operation Pipeline consome esses dados brutos e os processa, gerando uma versão pré-processada e anonimizada da fonte de dados, conforme definida pelos dados brutos. Os dados pré-processados são então combinados com base em suas áreas de negócio e são mesclados em pontos-chave, resultando em um conjunto de dados facilmente consumível para análise e criação de modelos.

Dessa maneira, o data lake foi dividido em três áreas distintas:

- *Raw Data Storage* (RDS): Nessa área, os dados brutos dos sistemas existentes são armazenados pelo operador em formato tabular, juntamente com metadados padronizados para o processamento subsequente na pipeline.
- *Processed Data Storage* (PDS): Após passarem pelas etapas de validação de esquema, validação de dados, transformação e aplicação de medidas de privacidade, os dados são armazenados em subpastas, organizadas de acordo com a área de negócio e o ano correspondente. Esses dados estão prontos para serem acessados individualmente ou combinados com outras áreas de negócio para análises e validações específicas.
- *Integrated Data Storage* (IDS): Após a conclusão das etapas de alinhamento, concatenação, merge, validação pós merge e indexação, os dados são disponibilizados para consumo e uso estratégico. Essa área representa o resultado

final da pipeline, onde os dados são entregues de forma integrada e pronta para serem utilizados em análises e tomadas de decisão estratégicas.

Assim como o data lake, o fluxo de dados também foi dividido em três partes principais, cada uma desempenhando um papel específico na transformação e integração dos dados:

- *Data Consumption* (DS): essa etapa tem como objetivo realizar o processo de ETL em cada fonte de dados. Ela inicia com o escalonamento e identificação do esquema da fonte de dados, seguida pela validação do esquema para garantir sua consistência e integridade. Em seguida, a validação dos dados é realizada para assegurar a integridade dos dados. Durante esse processo, é utilizado um mecanismo de hash para permitir a leitura única dos arquivos, e os metadados são monitorados e alertas são gerados pelo Airflow.
- Data Privacy (DP): essa etapa tem como objetivo fornecer um mecanismo para preservar a privacidade dos dados. Ela envolve a aplicação de restrições e medidas de segurança em uma lista de informações e parâmetros que devem ser protegidos, garantindo a conformidade com regulamentações e políticas de privacidade.
- Data Integration (DI): essa etapa tem como objetivo fornecer uma visão unificada dos dados históricos e combinações-chave após o processo de privacidade dos dados. Após a conclusão das etapas anteriores, o alinhamento e a indexação das novas bases criadas são realizados para garantir a integração dos dados em uma visão consolidada e única. Além disso, os dados passam por transformações para padronizar as bases.

Essas três partes do fluxo de dados trabalham em conjunto para converter os dados brutos em um formato padronizado, aplicar restrições de privacidade, realizar a concatenação e integração de diversas fontes, resultando em um conjunto de dados preparado e integrado para análises e tomadas de decisão.

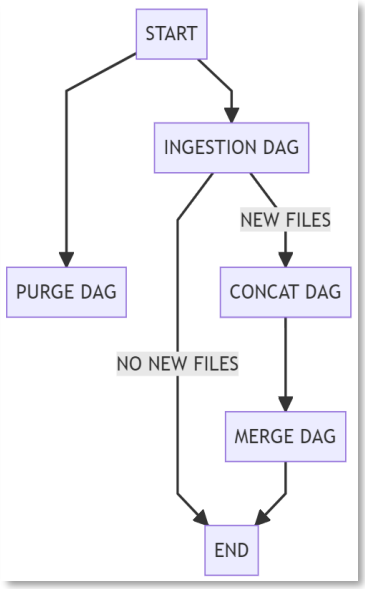
4.1.5 Fluxo de trabalho do processamento de dados

Conforme ilustrado na Figura 13, o primeiro passo consiste na execução da DAG de ingestão, na qual os arquivos brutos são lidos, processados, renomeados de acordo com o padrão adotado e, em seguida, armazenados como arquivos processados (no formato parquet) na área de dados processados. Após essa etapa, o Airflow verifica se há arquivos não processados e, em caso afirmativo, prossegue com as etapas de concatenação dos arquivos históricos por área de negócio na DAG de concatenação, bem como a combinação das áreas-chave na DAG de

mesclagem, resultando no armazenamento da versão mais recente e do histórico de todas as combinações realizadas.

Foi adicionado ao fluxo o processo de *Purge* para realizar a limpeza do data lake, removendo os dados históricos e evitando sobrecarga no ambiente. Esse processo possui um agendamento diferente, adaptado às necessidades percebidas do data lake e aos requisitos específicos do cliente. Inicialmente, foi estabelecido um critério de limpeza, no qual todos os dados com data histórica superior a 6 meses são excluídos, e essa purgação é ativada mensalmente.

Figura 19 – Diagrama do fluxo de trabalho de processamento de dados no Airflow



Fonte: Elaborado pelo autor, 2023.

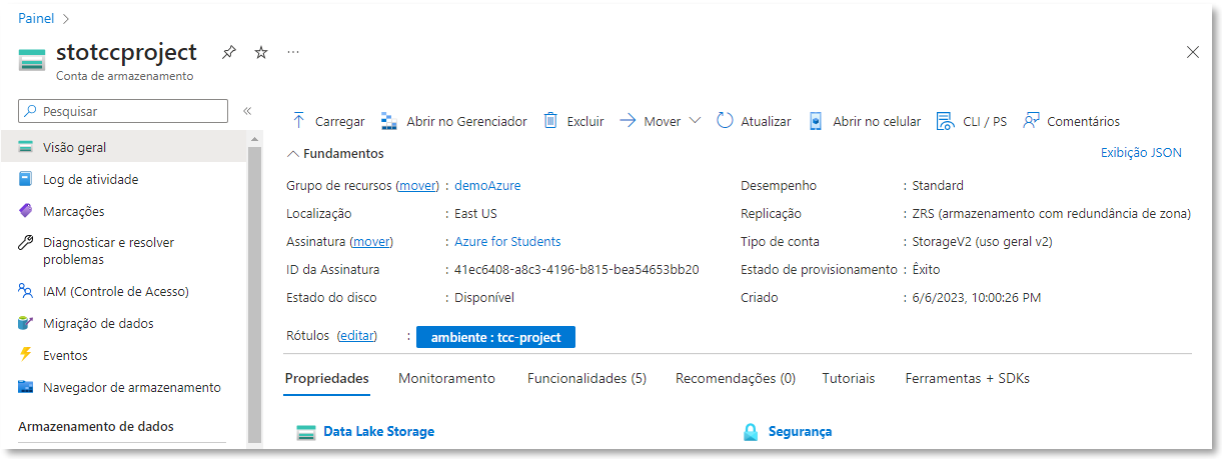
4.2 Descrição do ambiente

Para dar início à elaboração deste trabalho, cujo objetivo é criar um pipeline para a ingestão e tratamento de dados, foram realizadas algumas configurações iniciais no portal da Azure, para uso do Blob Storage 2 como serviço de armazenamento em nuvem e a configuração de um Docker para disponibilização dos serviços de Apache Airflow para a execução e agendamento do fluxo dos dados e o PostgreeSQL como banco de dados auxiliar ao processo de pipeline. Essas configurações são essenciais para viabilizar a ingestão dos dados, manutenção e acompanhamento do processo.

Foi utilizado o serviço pago da *Azure for Students* por meio da conta estudantil universitária do autor, que concede acesso a recursos e até 100 reais para utilização dos serviços

da plataforma. Para iniciar o processo, foi criado um *resource group* na *Azure*, o qual funciona como um contêiner para manter recursos relacionados a uma solução específica. Em seguida, foram criados um storage account e o DL como um recurso de armazenamento em nuvem, oferecendo alta disponibilidade, escalabilidade e segurança. A Figura 19 abaixo ilustra essa configuração:

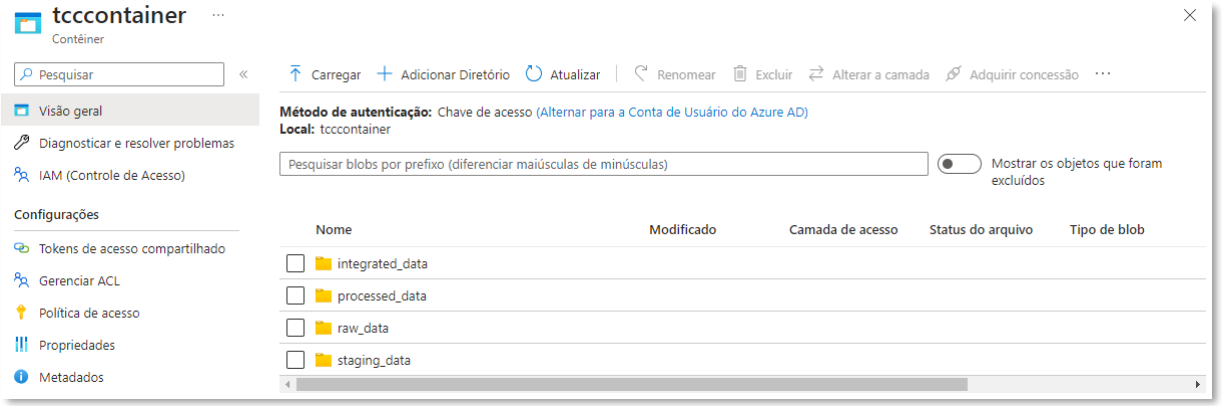
Figura 20 – Interface Azure com a visão do Data Lake Storage



Fonte: Elaborado pelo autor, 2023.

Após a criação do Data Lake, foram criados os containers, que atuam como diretórios no sistema de arquivos, permitindo a organização de diversos tipos de dados. No contexto deste projeto, foram criadas várias partições nesses diretórios, onde os dados serão armazenados em diferentes estágios de manipulação e/ou tratamento, conforme definido previamente na arquitetura. A Figura 21 ilustra essa estrutura:

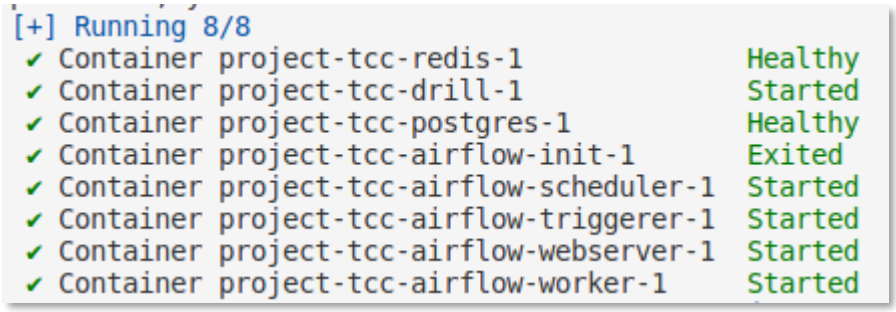
Figura 21 – Interface Azure do Container com suas zonas criadas



Fonte: Elaborado pelo autor, 2023.

Além disso, foi realizado um processo de configuração do Docker Compose, mostrado na Figura 22, para executar as imagens do Apache Airflow e PostgreSQL com todas as dependências corretamente configuradas. Essa abordagem simplifica a gestão das dependências, bem como a definição das variáveis de ambiente, evitando possíveis conflitos entre as diferentes versões de software utilizadas. Além disso, a utilização do Docker permitiu a criação de um ambiente isolado, assegurando a replicabilidade e a consistência dos resultados obtidos. Essas medidas foram essenciais para alcançar um ambiente de desenvolvimento controlado e estável.

Figura 22 – Start dos container Docker utilizados no sistema



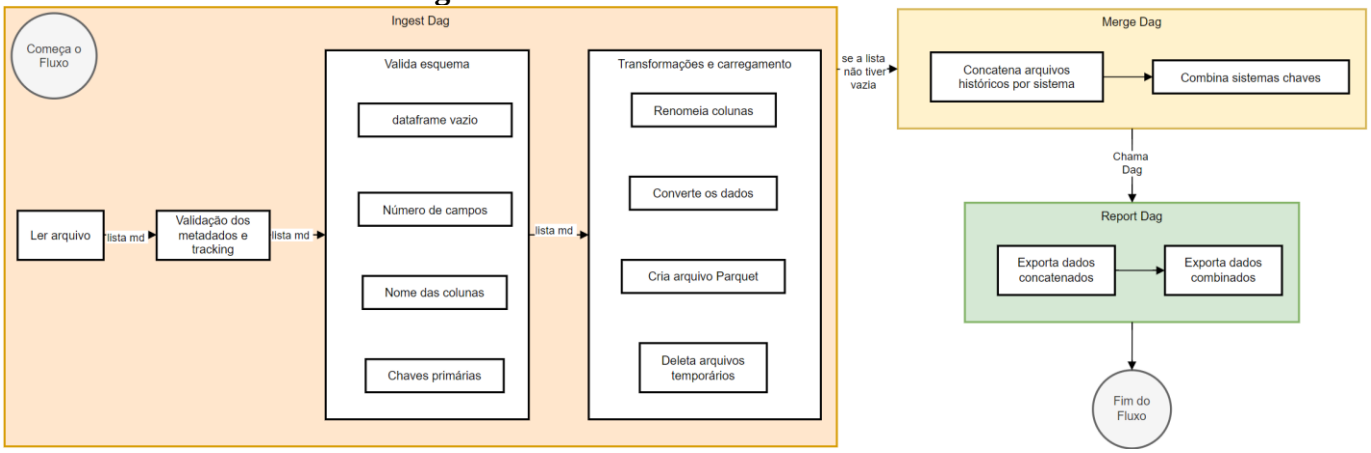
Fonte: Elaborado pelo autor, 2023.

4.3 Descrição do fluxo de dados

Após a completa execução do fluxo da arquitetura de dados desenvolvida, foi realizada a sumarização de suas etapas, conforme apresentado na Figura 23. Para isso, foram desenvolvidas as Dags de ingestão, responsáveis por ler cada um dos arquivos, validar os metadados, rastrear o processo (tracking), validar o esquema do arquivo, realizar as transformações necessárias e carregar os dados na zona denominada "processed data".

Na Dag de merge, os dados provenientes de sistemas de origem respectivos são concatenados de acordo com regras pré-estabelecidas para lidar com as diferenças históricas. Além disso, os arquivos históricos de sistemas-chave são combinados e salvos na zona “integrated data”. Por fim, esses novos arquivos concatenados e combinados são exportados para um banco de dados PostgreSQL, onde poderão ser consumidos para a geração de modelos e insights, de acordo com as necessidades das áreas de negócio.

Figura 23 – Visão sumarizada do fluxo de dados



Fonte: Elaborado pelo autor, 2023.

A arquitetura e orquestração utilizando o Apache Airflow neste estudo contribui para a proposta de sustentabilidade e manutenção da ETL. Essa abordagem segue as recomendações encontradas nos estudos de Qaiser (2023) e Matskin (2021), que exploram o uso de diversas ferramentas para processos de ETL. A utilização de reutilização na implementação, integração fácil com outras ferramentas e a personalização adequada para o uso da ETL são considerações importantes, conforme indicado pela literatura. Além disso, a recomendação de utilizar tecnologias de código aberto para lidar com grandes volumes de dados é respaldada por estudos como o de Martins (2014).

No que diz respeito ao armazenamento de dados diversificados, a abordagem adotada neste trabalho está alinhada com a proposta de Finnigan (2021), que enfatiza o uso flexível e escalável de metadados. Adicionalmente, a transferência de dados brutos entre sistemas é evitada, seguindo a sugestão de Hilgendorf (2022) e DeCarlo (2022) por meio da utilização de arquivos .parquet e tratamento dos dados nas etapas iniciais da pipeline. No entanto, a recomendação de Hilgendorf (2022) de seletividade de dados, considerando apenas aqueles pertinentes a condições e eventos relevantes, não foi totalmente adotada. Da mesma forma, o paralelismo na etapa de ingestão dos dados, como sugerido por Nargesian (2019), não foi amplamente aplicado.

Em relação às arquiteturas de Big Data, a arquitetura desenvolvida neste estudo apresenta maior semelhança com a Arquitetura Zeta, como identificado por Kalipe e Behera (2019). Ambas propõem etapas de processos isolados, nos quais o software pode ser executado e novos dados podem ser integrados de forma conveniente.

A arquitetura desenvolvida também resolve desafios semelhantes ao trabalho de Haeffner (2022), desenvolvido para a plataforma AWS, ao permitir a adição ou remoção de novos dados sem a necessidade de alterar os componentes da arquitetura. Além disso, em termos de complexidade das tarefas realizadas e foco no comportamento do Apache Airflow, o trabalho se assemelha ao estudo comparativo de ferramentas de pipeline de dados realizado por Leal (2021) e à pipeline de dados desenvolvida por Arrais (2022) utilizando o Data Factory da Azure.

Dessa forma, conclui-se a experiência completa de uma pipeline de engenharia de dados, evidenciando sua importância e os desafios enfrentados ao longo do processo. Essa abordagem pode ser aplicada em diversos serviços, destacando a relevância da engenharia de dados no contexto atual. Com isso, reforça-se a importância de investir em estratégias eficientes de ingestão, transformação e disponibilização, transformando dados, em informações e em conhecimento.

4.4 Ingestão dos dados

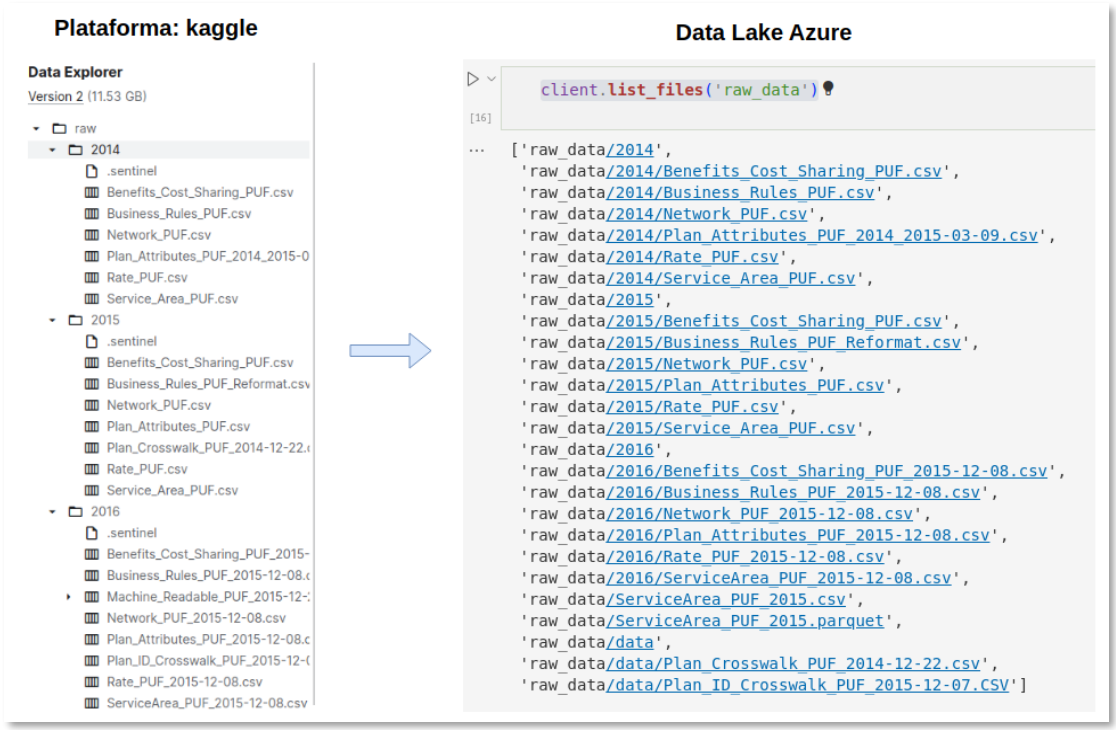
O referido conjunto de dados, originalmente preparado e divulgado pelo Centro de Serviços Medicare e Medicaid (CMS), é composto por sete fontes de bases com os arquivos distribuídos da seguinte maneira:

- BenefíciosCostSharing (BenCS-PUF): 3 arquivos CSV representando os anos de 2014, 2015 e 2016 de dados em nível de variante sobre benefícios de saúde essenciais, limites de cobertura e compartilhamento de custos.
- BusinessRules (BR-PUF): 3 arquivos CSV representando os anos de 2014, 2015 e 2016 de dados em nível de regras de negócios de classificação, como idade máxima para um dependente e relacionamentos dependentes permitidos.
- Network (Ntwrk-PUF): 3 arquivos CSV representando os anos de 2014, 2015 e 2016 de dados que identificam as URLs da rede do provedor.
- PlanAttributes (Plan-PUF): 3 arquivos CSV representando os anos de 2014, 2015 e 2016 de dados em nível sobre pagamentos diretos máximos, franquias, elegibilidade e outros atributos do plano.
- Rate (Rate-PUF): 3 arquivos CSV representando os anos de 2014, 2015 e 2016 de dados sobre tarifas com base na idade de um assinante qualificado, uso de tabaco, localização geográfica e taxas de nível familiar.

- ServiceArea (SA-PUF): 3 arquivos CSV representando os anos de 2014, 2015 e 2016 de dados do emissor sobre áreas geográficas de serviço, incluindo estado, município e código postal.
- Crosswalk(CW-PUF): 2 arquivos auxiliares que conectam as bases históricas com 2015 e 2016 de mapeamento de dados dos planos oferecidos no ano do plano anterior para planos oferecidos no ano do plano atual.

A primeira etapa na disponibilização dos dados na pipeline de dados é representada na Figura 25 consistiu na extração dos dados de suas fontes originais, seguida pela sua inclusão na área de dados brutos (RDS) do data lake. Além disso, foram adicionados metadados básicos aos arquivos para o gerenciamento na pipeline, e subpastas foram criadas para a separação de arquivos com nomes iguais, mas conteúdos diferentes. Essa abordagem garantiu a integridade e a viabilidade da base original de todas as fontes e seria a responsabilidade principal do Operador.

Figura 24 – Inserção das bases disponibilizadas pela kaggle no data lake do projeto



Fonte: Elaborado pelo autor, 2023.

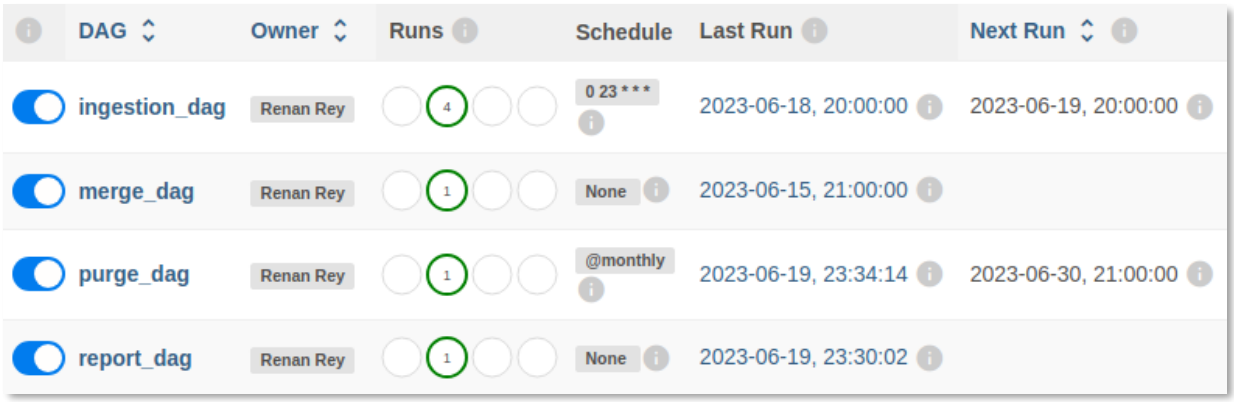
Com o intuito de tratar com o volume de dados utilizados da Health Insurance Marketplace, foi desenvolvido um script em Python para automatizar a inserção das bases de dados disponibilizadas pela plataforma Kaggle no data lake do projeto. Esse script também foi

responsável por adicionar metadados pertinentes a cada arquivo, como nome, hash, ano, ID e área de negócio. É importante ressaltar que cada um desses arquivos poderia ser adicionado tanto pela própria plataforma Azure quanto por meio de inserção direta no data lake. O script desenvolvido desempenhou apenas um auxílio na agilidade e padronização na inserção das bases.

4.5 Ingestão dos dados

Foi criada a DAG de ingestão, que é acionada diariamente às 23h e é responsável pela leitura e carregamento de cada base de entrada. Essa DAG segue as práticas propostas por NARGESIAN (2019), priorizando a velocidade em detrimento de uma análise aprofundada dos dados, realizando verificações básicas nos dados e seus metadados. Caso novos arquivos sejam carregados com sucesso, essa DAG aciona a DAG de merge, que, por sua vez, aciona a DAG de expotação – "report_dag", conforme Figura 25. Em momento diferente a essas DAGs, temos a DAG de "purge", que é executada mensalmente e tem a responsabilidade de limpar os arquivos antigos das subpastas do DL.

Figura 25 – Interface geral do Airflow com as DAGs desenvolvidas e seus Schedules

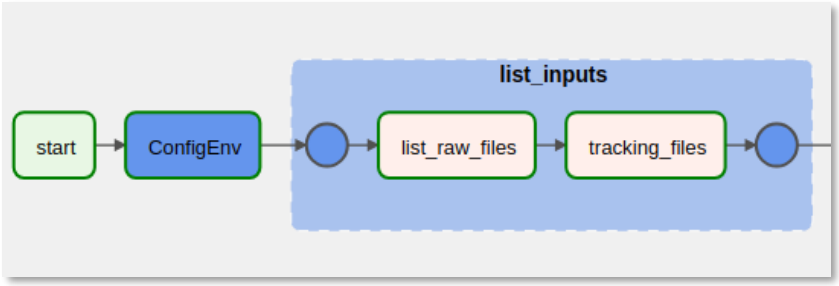


Fonte: Elaborado pelo autor, 2023.

Às 23h, o processo é iniciado e começa com a leitura de todos os arquivos disponíveis no raw data. Nesse momento, são realizadas algumas filtragens, como a verificação das extensões dos arquivos aceitos (por exemplo, csv), a presença de metadados relevantes para o processamento, entre outros critérios. Esses metadados são adicionados a uma lista que será utilizada ao longo de todo o processo. Os arquivos considerados "validos" para prosseguir no fluxo são encaminhados para a etapa de "tracking files", onde o hash dos arquivos é verificado para determinar se o arquivo já foi lido anteriormente. Em caso afirmativo, o arquivo é removido da lista de arquivos consumidos, caso contrário, seu hash é armazenado para futuras

verificações e o arquivo é disponibilizado na lista para as etapas subsequentes do processo, conforme o grupo "*list inputs*" representado na Figura 26 abaixo.

Figura 26 – Fluxo inicial da DAG de ingestão com leitura e rastreamento das bases



Fonte: Elaborado pelo autor, 2023.

4.6 Validação, transformação e carregamento

Após a leitura e rastreamento dos dados, a etapa de validação das tabelas é iniciada. A validação desempenha um papel fundamental na garantia da qualidade dos dados processados e atua como um processo de limpeza na pipeline. O trecho de código apresentado na Figura 27 demonstra a realização das validações, utilizando o auxílio do arquivo JSON criado anteriormente (conforme ilustrado na Figura 14). As validações incluem verificar se o dataframe está vazio, validar o número esperado de campos conforme o arquivo de entrada, assegurar que os nomes das colunas de entrada correspondem às colunas validadas, garantir que a chave primária contenha apenas registros únicos, verificar se os tipos das colunas de entrada estão em conformidade com os tipos esperados, e identificar a presença de campos importantes, porém vazios, na base de entrada.

Figura 27 – Trecho do código da Dag Ingestion utilizado para validar os campos de entrada.

```
df_analysis = creater_analysis.create_schema_analysis(df)
df_validator = read_validator(metadata_file)

validatordata = schema_validator.SchemaValidator(
    df_analysis, df_validator)

validate_results.append(
    (validatordata.check_if_dataframe_is_empty()))

validate_results.append(
    (validatordata.validate_number_fields()))

validate_results.append(
    (validatordata.validate_matched_columns("column_name")))

validate_results.append(
    (validatordata.check_unique_primary_key()))

validate_results.append(
    (validatordata.validate_data_types()))

validate_results.append(
    (validatordata.check_for_missing_values()))

errors_filtered = [
    error_filtered for error_filtered in validate_results if error_filtered != None]
```

Fonte: Elaborado pelo autor, 2023.

A Figura 28 ilustra um exemplo de carga parcial das fontes de dados do Health Insurance Marketplace, incluindo um arquivo de exemplo com erro inserido para demonstrar o processo de captura das informações durante a etapa de validação. É importante ressaltar que, caso haja mais de um erro, esses erros serão listados junto com o primeiro erro no metadado. Isso permite acompanhar o status de cada arquivo específico e utilizar essas informações em etapas e sistemas subsequentes para visualização e tratamento adequado dos arquivos.

Figura 28 – Saída dos metadados validados com sucesso e com erros na interface do Airflow

Key	Value
return_value	[{'content_hash': '2014_Benefits_Cost_Sharing_PUF.csv', 'filename': 'Benefits_Cost_Sharing_PUF.csv', 'id': '3', 'schema': 'BenefitsCostSharing', 'year': '2014'}, {'content_hash': '2016_ServiceArea_PUF_2015-12-08.csv', 'filename': 'ServiceArea_PUF_2015-12-08.csv', 'id': '18', 'schema': 'ServiceArea', 'year': '2016'}, {'content_hash': 'data_Plan_Crosswalk_PUF_2014-12-22.csv', 'filename': 'Plan_Crosswalk_PUF_2014-12-22.csv', 'id': '7', 'schema': 'Crosswalk2015', 'year': 'data'}]
schema_error_files	[['Número de campos é menor do que o mínimo esperado.', '100', 'Benefits_Cost_Sharing_PUF_with_errors.csv', 'BenefitsCostSharing']]

Fonte: Elaborado pelo autor, 2023.

Na Figura 29, apresenta-se outra etapa fundamental do código responsável por realizar transformações, processamento e carregamento dos dados para o diretório "processed data". Destaca-se a adoção das considerações de HAEFFNER (2022) e ZOU (2021) sobre as limitações de capacidade de processamento do Airflow e a não recomendação de transferência

de grandes volumes de informações e interações complexas entre as tarefas para garantir a escalabilidade do projeto. Portanto, foram adotadas duas estratégias: a passagem de metadados entre as tarefas, reduzindo consideravelmente o tamanho dos dados, conforme ilustrado na Figura 27, ou a realização de um conjunto de ações, como exemplificado na Figura abaixo.

Figura 29 – Saída dos metadados validados com sucesso e com erros na interface do Airflow

```
try:
    df = read_input(client, RAW_PATH + md_file[FILE_YEAR_LABEL], md_file)
    df_validator = read_validator(md_file)
    df = rename_columns(
        df, df_validator, md_file[SCHEMA_LABEL])

    #anonimization secrety columns
    filtered_columns = df_validator.loc[df_validator['is_anonymizated_field']
    df = anonymization.encode_columns(df, filtered_columns)

    df = handle_columns.handle_df(df, md_file[SCHEMA_LABEL])
except ValueError as e:…

try:
    date = datetime.fromtimestamp(int(dtime))
    formated_date = date.strftime("%Y_%m_%d")
    md_file[VALIDATION_DATE_LABEL] = str(dtime)
    md_file[PDSA_UPDATE_DATE_LABEL] = str(dtime)
    md_file[PROCESSED_DATA_FORMAT_LABEL] = _output_format
    md_file[NEW_FILE_NAME_LABEL] = md_file[SCHEMA_LABEL] + '_' + \
    md_file[FILE_YEAR_LABEL] + '.' + _output_format

except ValueError as e:…

try:
    client.upload_data_frame(df, PDS_TMP_PATH, md_file[NEW_FILE_NAME_LABEL],…
    processed_md.append(md_file)
except ValueError as e:…
```

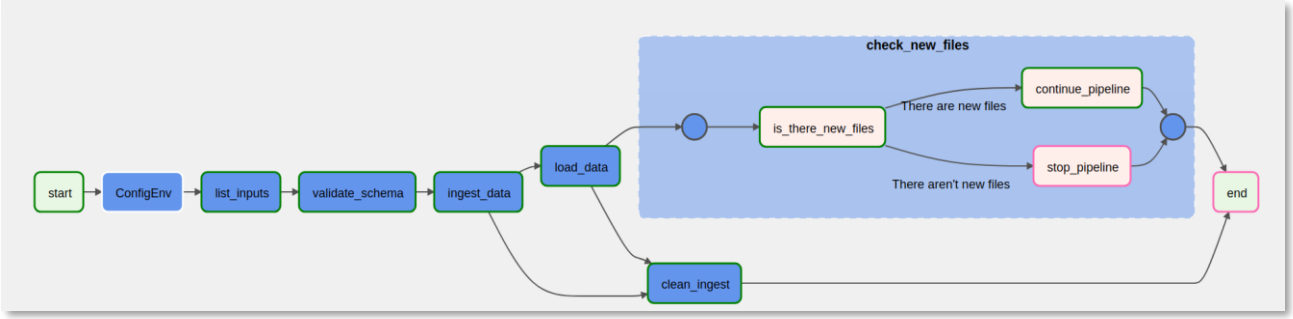
Fonte: Elaborado pelo autor, 2023.

Nessa etapa, são realizados quatro pontos-chave. Primeiramente, na função "rename_columns", ocorre a atualização das colunas da base de dados com os campos formatados provenientes do arquivo JSON respectivo ao sistema da base criado anteriormente. Em seguida, é realizada a anonimização dos campos marcados também pelo referido arquivo JSON. Posteriormente, a função "handle_df" é responsável por assegurar os tipos de dados corretos na saída da respectiva base de entrada. São adicionados novos metadados ao arquivo, fornecendo informações extras sobre o processamento. Por fim, o arquivo é carregado na pasta "processed_data/tmp", dentro dos dados processados.

A Figura 30 apresenta o fluxo completo e encadeado das tarefas da ingestão de dados, com a transferência de metadados e informações cruciais entre elas, conforme mencionado anteriormente. Além disso, existem etapas adicionais não mencionadas, como o "load_data", que formaliza as bases dentro dos arquivos processados, o "clean_ingest", que realiza a limpeza

dos arquivos processados com sucesso na pasta temporária, e o "check_new_files", que determina a continuidade do fluxo com base na presença ou ausência de novos arquivos processados, acionando ou não a execução da "merge dag". Na Figura 30, é apresentado o resultado dessa etapa do processo, onde cada arquivo processado é dividido em subpastas de acordo com seus sistemas de origem e armazenado no formato Parquet, conforme definido anteriormente.

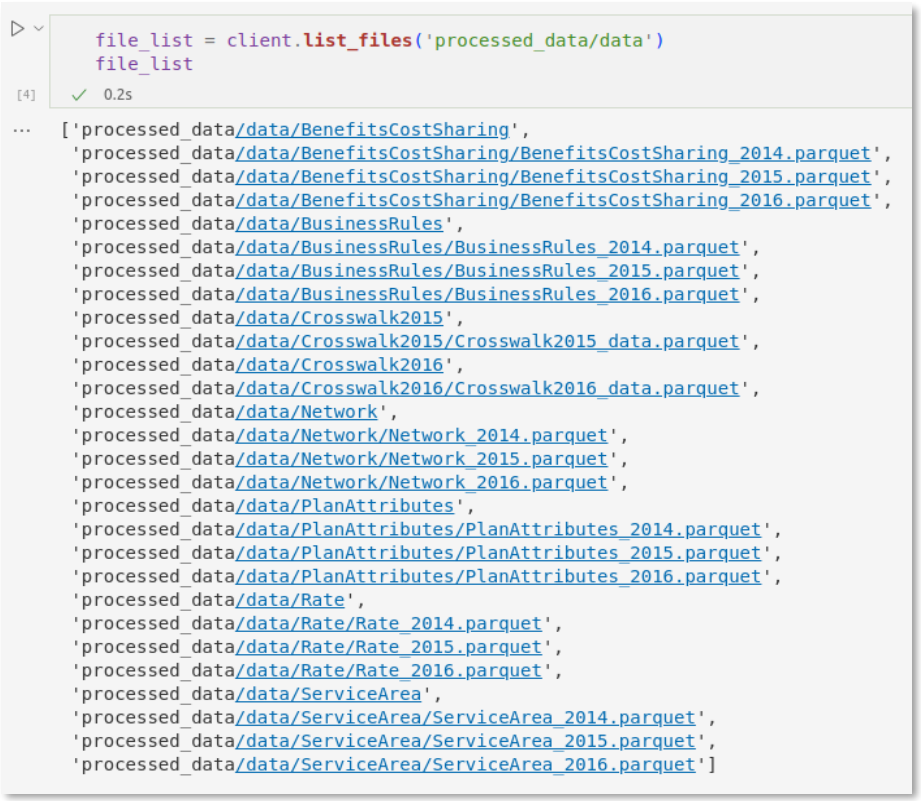
Figura 30 – Fluxo completo da DAG de ingestão no Airflow



Fonte: Elaborado pelo autor, 2023.

É relevante ressaltar que, apesar de a Figura 31 apresentar apenas uma entrega parcial da volumetria de cada conjunto de dados carregados devido às limitações de processamento do Azure Students, mencionadas anteriormente, é possível observar o desempenho do tempo de processamento. Em média, cada arquivo foi processado em 9 minutos e 30 segundos, o que corresponde a um tempo médio de 28,5 segundos por arquivo para percorrer todas as etapas da DAG. Não foi possível mensurar o tempo de carregamento com as bases completas no sistema.

Figura 31 – Visualização dos dados processados divididos por sistemas de origem



```
file_list = client.list_files('processed_data/data')
file_list
```

[4] ✓ 0.2s

```
['processed_data/data/BenefitsCostSharing',
 'processed_data/data/BenefitsCostSharing/BenefitsCostSharing_2014.parquet',
 'processed_data/data/BenefitsCostSharing/BenefitsCostSharing_2015.parquet',
 'processed_data/data/BenefitsCostSharing/BenefitsCostSharing_2016.parquet',
 'processed_data/data/BusinessRules',
 'processed_data/data/BusinessRules/BusinessRules_2014.parquet',
 'processed_data/data/BusinessRules/BusinessRules_2015.parquet',
 'processed_data/data/BusinessRules/BusinessRules_2016.parquet',
 'processed_data/data/Crosswalk2015',
 'processed_data/data/Crosswalk2015/Crosswalk2015_data.parquet',
 'processed_data/data/Crosswalk2016',
 'processed_data/data/Crosswalk2016/Crosswalk2016_data.parquet',
 'processed_data/data/Network',
 'processed_data/data/Network/Network_2014.parquet',
 'processed_data/data/Network/Network_2015.parquet',
 'processed_data/data/Network/Network_2016.parquet',
 'processed_data/data/PlanAttributes',
 'processed_data/data/PlanAttributes/PlanAttributes_2014.parquet',
 'processed_data/data/PlanAttributes/PlanAttributes_2015.parquet',
 'processed_data/data/PlanAttributes/PlanAttributes_2016.parquet',
 'processed_data/data/Rate',
 'processed_data/data/Rate/Rate_2014.parquet',
 'processed_data/data/Rate/Rate_2015.parquet',
 'processed_data/data/Rate/Rate_2016.parquet',
 'processed_data/data/ServiceArea',
 'processed_data/data/ServiceArea/ServiceArea_2014.parquet',
 'processed_data/data/ServiceArea/ServiceArea_2015.parquet',
 'processed_data/data/ServiceArea/ServiceArea_2016.parquet']
```

Fonte: Elaborado pelo autor, 2023.

4.7 União e integração dos dados

Após a conclusão da DAG de ingestão dos dados, é iniciada a DAG de combinação caso haja arquivos novos. Essa DAG possui duas etapas principais. A primeira etapa consiste na concatenação histórica das bases de acordo com seu sistema de origem, utilizando as oscilações históricas mapeadas no tópico de "Conhecendo os Dados". O script apresentado na Figura 32 realiza essa concatenação, considerando as variações ao longo dos anos para cada sistema de origem. Vale ressaltar dois pontos importantes: primeiro, devido à complexidade das variações de mais de 75 campos nos *Plan Attributes*, nenhuma regra de tratamento específica foi adotada entre os anos; segundo, esse trecho do código é a única parte que requer revisão a cada nova variação de campos nos anos subsequentes. Outras alterações podem ser feitas por meio das variáveis de ambiente do Airflow.

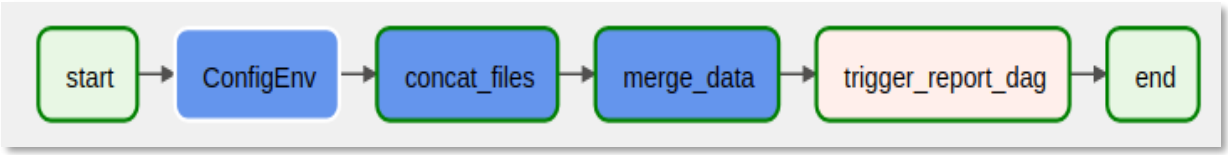
Figura 32 – Trecho de Script para Concatenação das Bases Históricas por Sistema

```
if table[:9] != "Crosswalk":
    buffer, _, md = client.get_file(PDS_CLEAN_PATH + table, tables[table]["2014"])
    d2014 = pd.read_parquet(BytesIO(buffer))
    buffer, _, md = client.get_file(PDS_CLEAN_PATH + table, tables[table]["2015"])
    d2015 = pd.read_parquet(BytesIO(buffer))
    buffer, _, md = client.get_file(PDS_CLEAN_PATH + table, tables[table]["2016"])
    d2016 = pd.read_parquet(BytesIO(buffer))
    md_file = md['metadata']


    if table in ["ServiceArea", "BusinessRules", "Network"]:
        d2015 = d2015.rename(columns={"DentalOnly":"DentalOnlyPlan"})
        d2014 = d2014.rename(columns={"DentalOnly":"DentalOnlyPlan"})

    if table=="BenefitsCostSharing":
        d2015 = d2015.rename(columns={"EHBPercentPremiumS4":"EHBPercentTotalPremium"})
        d2014 = d2014.rename(columns={"EHBPercentPremiumS4":"EHBPercentTotalPremium"})

    if table != "Crosswalk":
        df = pd.concat([d2014, d2015, d2016])
    else:
        df = pd.concat([d2015, d2016])
else:
    buffer, _, md = client.get_file(PDS_CLEAN_PATH + table, tables[table]["data"])
    df = pd.read_parquet(BytesIO(buffer))
    md_file = md['metadata']
```

Fonte: Elaborado pelo autor, 2023.

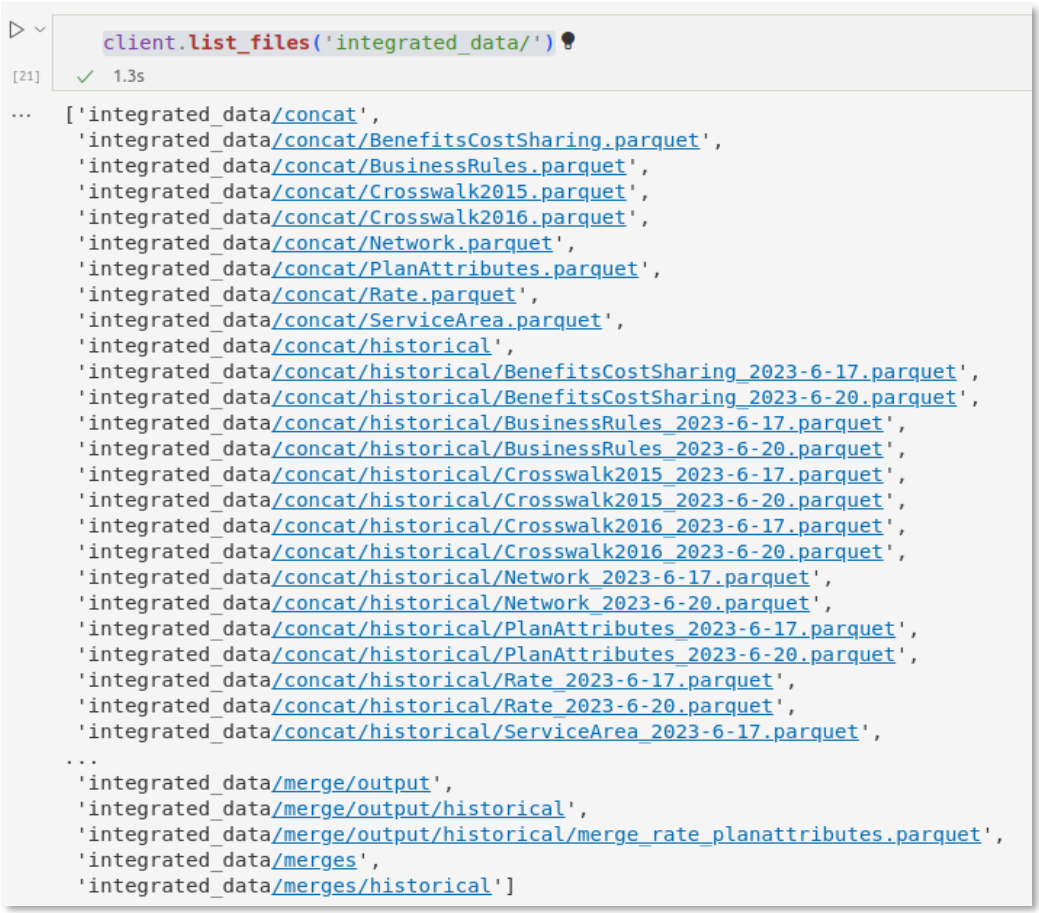
Após a finalização da tarefa de concatenação das bases históricas, a DAG avança para a etapa de combinação, conforme ilustrado no fluxo da Figura 33. A tarefa "merge_data" desempenha um papel fundamental nessa etapa, unindo os sistemas de origem relevantes e fornecendo informações diretas para atender às solicitações dos analistas responsáveis pela criação de dashboards, modelagem e análise dos dados. Na Figura 34, são apresentados os resultados das bases geradas nessas duas etapas: os arquivos unidos pela área de negócio e seu histórico de concatenações, bem como os arquivos combinados e seu histórico. Após a conclusão das concatenações e combinações das bases, a próxima DAG, denominada "Report Dag", é acionada para dar continuidade ao processo.

Figura 33 – Fluxo completo da DAG de Merge no Airflow



Fonte: Elaborado pelo autor, 2023.

Figura 34 – Resultados das bases por área de negócio, combinações e históricos de geração

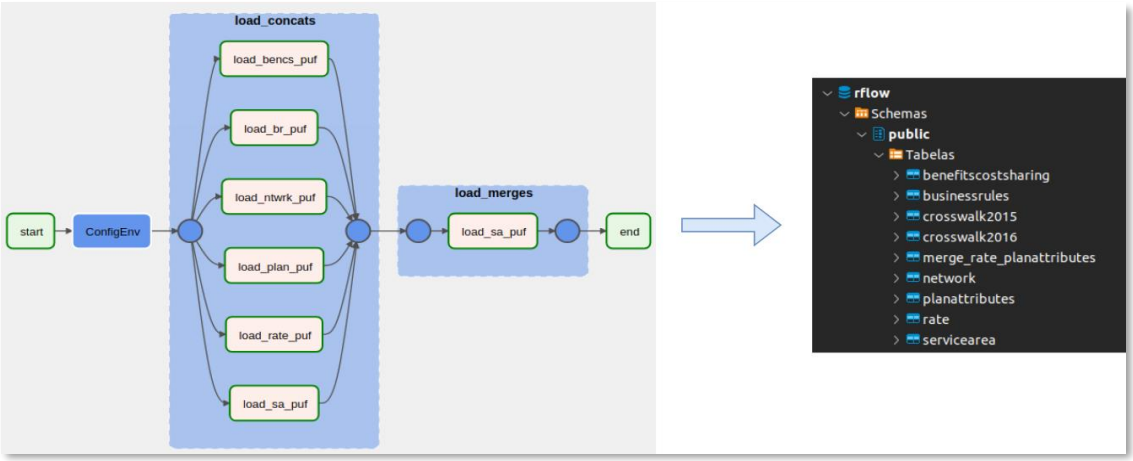


Fonte: Elaborado pelo autor, 2023.

4.8 Transmissão e visualização dos resultados

Com a conclusão dos processos anteriores, a pipeline das DAGs é finalizada, disponibilizando os arquivos concatenados e suas combinações-chave em um banco de dados PostgreSQL para facilitar o consumo. Conforme ilustrado na Figura 35, o processo de criação das tabelas e o carregamento dos dados foram realizados de forma paralela, seguindo os conceitos de NARGESIAN (2019). Para cada área de negócio dos dados da Health Insurance Marketplace, foi designada uma tarefa específica, resultando em um carregamento praticamente em tempo real devido à velocidade aprimorada na leitura de arquivos parquet e o uso de paralelismo. À direita da Figura abaixo, podemos observar as tabelas criadas dentro do banco de dados "rflow".

Figura 35 – Fluxo completo da DAG de Report no Airflow e evidência das tabelas criadas



Fonte: Elaborado pelo autor, 2023.

Após a disponibilização das tabelas, torna-se possível conectar-se a elas e realizar análises, modelagens e extrair insights relevantes. No Apêndice A, encontra-se as consultas das tabelas criadas mais em detalhe dos campos e amostra dos seus valores. Para demonstrar o uso prático dessas tabelas, foi desenvolvido um dashboard no Power BI no Apêndice B.

O dashboard gerado no âmbito deste estudo apresenta uma abordagem eficaz para visualização dos dados após o processamento. Por meio de uma visão em forma de funil, é possível realizar análises que transitam entre uma perspectiva macro e detalhada dos dados. Por exemplo, é possível observar a variação dos valores de descontos ao longo dos anos, chegando até a visão diária ou por dia da semana. Essa abordagem permite a geração de indicadores-chave de desempenho (KPIs) com base nos requisitos do cliente, abrangendo desde uma visão ampla até detalhes específicos de campos de interesse.

O dashboard oferece recursos para filtrar campos, períodos e valores, permitindo a combinação e comparação desses elementos. Isso possibilita a criação de um mapeamento completo de visualização de dados personalizado para a organização. Por fim, o dashboard fornece suporte para tomada de decisões, acompanhamento de resultados e análises visuais, transformando dados que antes eram de difícil interpretação em informações apresentadas de forma clara e acessível por meio de gráficos e outras representações visuais.

5 CONCLUSÕES

A arquitetura proposta neste estudo foi implementada, integrando sistemas adaptados de diversas origens. Utilizando o Apache Airflow e os serviços em nuvem fornecidos pela plataforma Microsoft Azure. Foi abordado o processamento escalável das tarefas, ajustando-se dinamicamente às demandas do ambiente, cliente e interações com outros times.

Após a análise de todo o fluxo comparando com as recomendações encontrada na litetatura academica, verificou-se que a arquitetura desenvolvida atendeu às necessidades do problema apresentado. Destaca-se que a inclusão ou remoção de novas áreas de serviços com um conjunto de dados original não requer modificações nos componentes da arquitetura, desde que sejam atualizados seus validadores de ambiente.

Este estudo se insere no contexto atual, no qual há um crescente interesse por parte de empresas, governos e comunidades em melhorar a gestão da qualidade dos dados. Uma abordagem eficiente no processamento de dados pode proporcionar valor agregado e *insights* relevantes para as organizações, servindo como base para diversas áreas de aplicação.

Como possibilidade de futuros trabalhos, destaca-se a adoção de variações de arquitetura a fim de comparar e identificar aquela que melhor atende às demandas de desempenho e velocidade do sistema. O uso de ferramentas e recursos auxiliares pode ser explorado, permitindo substituições, como a implementação do mesmo serviço em um ambiente Apache Hadoop ou soluções totalmente gerenciadas pela plataforma Azure, envolvendo ferramentas de seu ecossistema, embora com a consideração de custos financeiros adicionais. Embora essas escolhas não afetem o funcionamento e comportamento da solução proposta, a seleção das ferramentas é condicionada à realidade do problema em questão, exigindo uma reavaliação das estruturas desejadas em cenários mais ou menos complexos e com maior ou menor diversidade de fontes de dados.

Também é possível avaliar e comparar a utilização de recursos distintos entre outros provedores de serviços em nuvem, oferecendo uma maior versatilidade tanto para as ferramentas selecionadas quanto para os custos envolvidos na solução. Além disso, no contexto da adoção de variações de arquitetura, é recomendado realizar uma avaliação minuciosa a fim de identificar a abordagem que melhor atenda às necessidades do sistema.

Em suma, ao longo deste estudo, foram apresentados os fundamentos e a implementação de uma arquitetura de engenharia de dados eficaz e escalável. A partir dessa abordagem, foi

possível desenvolver uma solução prática e funcional, demonstrando sua aplicabilidade e efetividade no processamento, armazenamento e análise de dados provenientes do Health Insurance Marketplace. Por meio dessa arquitetura, a disponibilização de dados refinados para diversas áreas, como análise de dados, modelagem matemática e inteligência empresarial, torna-se possível, contribuindo para a geração de informações valiosas e conhecimentos relevantes.

REFERÊNCIAS

APACHE AIRFLOW. **Apache Airflow**. Disponível em: https://airflow.apache.org. Site oficial. Acesso em: 23 mai. 2023.

APACHE HIVE. **Apache Hive TM**. Disponível em: https://hive.apache.org. Site oficial. Acesso em: 09 mai. 2023.

ARRAIS. KAROLINA. **Construção de uma Pipeline de Dados Utilizando Serviços da Nuvem**. Universidade Federal de São Carlos. São Carlos – SP. 2022.

C4MODEL. **The C4 model for visualizing software architecture – Context, containers, components, and code**. 2023. Disponível em: https://c4model.com/. Acesso em: 31 de maio de 2023.

CMS, 2023. **Health Insurance Exchange Public Use Files (Exchange PUFs)**. Disponível em: https://www.cms.gov/cciio/resources/data-resources/marketplace-puf. Acesso em 30 mai 2023.

DECARLO, G., OLIVEIRA, D., PORTO, F. **Otimização de Dataflows em Frameworks de Big Data por meio do Reúso de Dados**. Universidade Federal Fluminense. Niterói, RJ – Brasil. 2022.

DHAR, Vasant. **Data science and prediction**. Communications of the ACM. Volume 56, Issue12, December 2013, pp. 64–73.

DIAMOND, P. **Armazenamento em nuvem vs. servidores locais: nove considerações importantes**. 2020. Disponível em: < https://www.microsoft.com/pt-br/microsoft-365/businessinsights-ideas/resources/cloud-storage-vs-on-premises-servers>

DIJCKS, J.ÿP. **Oracle: Big Data para a Empresa**: **Um White Paper da Oracle**. Oracle Corporation. 2013.

Elisa; BUDRIONIS, Andrius; GODTLIEBSEN, Fred. **Challenges and opportunities beyond structured data in analysis of electronic health records**. Wiley Interdisciplinary Reviews: Computational Statistics, p. e1549, 2021.

ESCOVEDO. **Introdução a Data Science: algoritmos de Machine Learning e métodos de análise**. ALURA: Casa do Código. 2020. ISBN 9788572540544
.
FATIMA, H.; WASNIK, K. **Comparison of SQL, NoSQL and NewSQL databases for internet of things**. In: 2016 IEEE Bombay Section Symposium (IBSS). IEEE, 2016.

FÁVERO, L.P.; BELFIORE, P. **Manual de Análise de Dados - Estatística e Modelagem Multivariada com Excel, SPSSe Stata**. Cerqueira César: Elsevier Editora Ltda., 2017.

FINNIGAN, L. & TONER, E. **Building and Maintaining Metadata Aggregation Workflows Using Apache Airflow**. Code4lib Journal. 2021.

GARCIA, MARCO. **Data Enginner ou Engenheiro de Dados - Conheça mais sobre**. 2020. Disponível em: https://www.cetax.com.br/blog/data-engineer-ou-engenheiro-de-dados/. Acesso em: 24 de maio de 2023.

GOLDMAN, Alfredo; KON, Fábio; PEREIRA JUNIOR, Francisco; POLATO, Ivanilto; PEREIRA, Rosangela de Fátima. **Apache hadoop: conceitos teóricos e práticos, evolução e novas possibilidades**. Porto Alegre: SBC, 2012.Disponível em: https://repositorio.usp.br/item/002338037

HAEFFNER, YAN. **Arquitetura em Nuvem Escalonável Para Coleta e Processamento de Dados**. Universidade Federal do Rio Grande do Sul. 2022.

HUMBLE, J.; FARLEY, D.; KIM, G. **Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation**. Nova York, Estados Unidos: AddisonWesley Professional, 2010. ISBN 9781282852860.

ITIGO. **Implementação de ETL**. Disponível em: https://www.itigoconsulting.com/implementacao-de-etl/. Acesso em: 25 de maio de 2023.

JO, J.; LEE, K. **MapReduce-Based D_ELT Framework to Address the Challenges of Geospatial Big Data**. ISPRS Int. J. Geo-Inf. 2019, 8, 475. Disponível em: <https://www.mdpi.com/2220-9964/8/11/475>. Acesso em: 24 de maio de 2023.

KALLIPE, G. & BEHERA R. **Big Data Architectures: A Detailed and Application Oriented Review**. Kalinga Institute of Industrial Technology. Odisha, India. 2019.

KAGGLE, 2017. **Health Insurance Marketplace dataset**. Disponível em: https://www.kaggle.com/datasets/hhs/health-insurance-marketplace. Site oficial. Acesso em 30 mai 2023.

KAGGLE, 2022. **What is a kaggle**. Disponível em: https://www.kaggle.com/general/328265. Site oficial. Acesso em 30 mai 2023.

LEAL, D. **Pentaho, Airflow, e Python: Avaliação de Ferramentas para a Criação de Pipeline de Dados**. Instituto Federal de Educação, Ciência e Tecnologia De São Paulo. Campinas, São Paulo. 2021.

LI, X.; ZOU, B. **An Automated Data Engineering Pipeline for Anomaly Detection of IoT Sensor Data**. [S.l.]: arXiv, 2021. DOI: 10.48550/ARXIV.2109.13828.
MICROSOFT. Introduction to Azure Data Lake Storage Gen2. Disponível em: https://learn.microsoft.com/en-us/azure/storage/blobs/data-lake-storage-introduction. Site oficial. Acesso em 15 jun 2023.

MARTINS, C., SIMOES, P., SÁ, J. **Uma Arquitetura Moderna de Dados: Um Caso de Teste**. Universidade de Minho. Portugal. 2014.

MATSKIN. M. & al. **A survey of Big Data Pipeline Orchestration Tools from the Perspective of the Data Cloud Project**. KTH Royal Institute of Technology. Stockholm, Sweden. 2021.

MERT, O. G., & al. **Big-Data Analytics Architecture for Businesses: a comprehensive review on new open-source big-data tools**. Cambridge Service Alliance. Retrieved from https://cambridgeservicealliance.eng.cam.ac.uk/news/2017OctPaper.

MICROSOFT. **Overview of data analysis**. Disponível em: < https://learn.microsoft.com/en-us/training/modules/data-analytics-microsoft/2-data-analysis?ns-enrollment-type=learningpath&ns-enrollment-id=learn-bizapps.get-started-data-analytics>. Acesso em: 22 mar. 2023.

NARGESIAN, F.; ZHU, E.; MILLER, R.; etc. **Data Lake Management: Challenges and Opportunities**. Proceedings of the VLDB Endowment 12.12 (2019): 1986-989. Web.
ORACLE. O que é Gerenciamento de Dados?. 2023. Disponível no link: https://www.oracle.com/br/database/what-is-data-management/.

OLIVEIRA, M. **Os Dados Não São Mais o Novo Petróleo**. Exame. 2022. Disponível no link: https://exame.com/colunistas/relacionamento-antes-do-marketing/os-dados-nao-sao-mais-o-novo-petroleo/. Acesso em 12 jun. 2023.

OUSSOUS, A.; BENJELLOUN F.; LAHCEN, A.; BELFKIH, S. **Big Data technologies: A survey**. Journal of King Saud University – Computer and Information Sciences. 2018. pp. 13-20.

PEKKA, P., & DANIEL, P. **Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems**. Big Data Research 2(4). 166-186. doi: https://doi.org/10.1016/j.bdr.2015.01.001

PETRILLO, A. & al. **Model-based vehicular prognostics framework using Big Data architecture**. 2020. doi: https://doi.org/10.1016/j.compind.2019.103177.

POWERBI. **Transforme os Dados em Impacto Imediato**. Disponível no link: https://powerbi.microsoft.com/pt-br/. Acesso em: 22 jun. 2023.

PRESSMAN, R. S. **Testing Computer Software**. New York, United States: John Wiley Sons, 1998. ISBN 9780470311998.

PROVOST, Foster; FAWCETT, Tom. **Data Science and its Relationship to Big Data and Data-Driven Decision Making**. Big Data, 2013.

A. QAISER. & al. **Comparative Analysis of ETL Tools in Big Data Analytics**. College of University of Engineering & Technology. Karachi, Pakistan. 2023.

RAJ, A., BOSCH, J, OLSSON, HH e WANG, TJ. **Modelling Data Pipelines**. 2020, 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2020.

RAUTENBER, S.; CARMO, P. **Big Data E Ciência De Dados: Complementariedade Conceitual No Processo De Tomada De Decisão**. Brazilian Journal of Information Studios: Research Trends. 2019. P.56-p.67

REGO, Bergson. **Gestão e Governança de Dados: Promovendo Dados como Ativo de Valor nas Empresas**. Rio de Janeiro: Brasport, 2013. ISBN: 978-85-7452-589-1.

SANTOS, CAIO. **Transformação de dados em informação estruturada para o apoio à decisão: estudo de caso do Portal de Compras do Governo Federal**. Brasília, 2022. Disponível no link: https://bdm.unb.br/handle/10483/31553. Acesso em: 22 jun. 2023.

SAWADOGO, P.; DARMONT, J. **On data lake architectures and metadata management**. Journal of Intelligent Information Systems, Springer Verlag, 2021, 56 (1), pp.97-120.

SHARMA, B.: **Architecting Data Lakes Data Management Architectures for Advanced Business Use Cases**. O'Reilly Media, Inc., 2018

SNOWFLAKE. **A Modern Approach to the Operational Data Store**. Data Cloud Guides. Disponível no link: https://www.snowflake.com/guides/modern-approach-operational-data-store. Acesso em: 22 jun. 2023.

TANENBAUM. A.; STEEN VAN, M. **Distributed System: Principles and Paradigms**. 2. Ed. Nova Jersey: Prentice Hall, 2006.

TAYEFI, Maryam; NGO, Phuong; CHOMUTARE, Taridzo; DALIANIS, Hercules; SALVI, VIANNA, W. B.; DUTRA, M. L. **Big Data e gestão da informação: Modelagem do Contexto Decisional Apoiado pela Sistemografia**. Revista Informação e Informação, Londrina, v. 21, n. 1, p. 185 - 212, jan./abr. 2016.

ZIKOPOULOS, Paul; EATON, Chris. **Understanding big data: Analytics for enterprise class hadoop and streaming data**. McGraw-Hill Osborne Media, 2011.

APÊNDICE A – CONSULTA DETALHADA DAS TABELAS GERADAS

```sql
select * from benefitscostsharing b S
```

efitscostsharing 1

ct * from benefitscostsharing b

benefitscostsharing_businessyear	benefitscostsharing_statecode	benefitscostsharing_issuerid	benefitscostsharing_sourcename	benefitscostsharing_versionnum	benefitscostsharing_importdate
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49

```sql
select * from businessrules
```

nessrules 1

ct * from businessrules

businessrules_businessyear	businessrules_statecode	businessrules_issuerid	businessrules_sourcename	businessrules_versionnum	businessrules_importdate	businessrules_issuerid2
2.014	AL	82.285	HIOS	7	2014-01-21 08:29:49	82.285
2.014	AL	82.285	HIOS	7	2014-01-21 08:29:49	82.285
2.014	AL	82.285	HIOS	7	2014-01-21 08:29:49	82.285
2.014	AL	82.285	HIOS	7	2014-01-21 08:29:49	82.285
2.014	AL	82.285	HIOS	7	2014-01-21 08:29:49	82.285
2.014	AZ	17.100	HIOS	7	2013-10-15 07:27:56	17.100
2.014	AZ	17.100	HIOS	7	2013-10-15 07:27:56	17.100
2.014	AZ	18.156	HIOS	1	2013-06-06 10:50:48	18.156
2.014	AZ	23.307	HIOS	8	2014-01-16 07:24:04	23.307
2.014	AZ	30.045	HIOS	5	2014-03-19 07:06:49	30.045

```sql
select * from crosswalk2015
```

swalk2015 1

ct * from crosswalk2015

crosswalk2015_state	crosswalk2015_dentalplan	crosswalk2015_planid_2014	crosswalk2015_issuerid_2014	crosswalk2015_multistateplan_2014	crosswalk2015_metallevel_2014
AK	Y	21989AK0010001	21.989	N	Low
AK	Y	21989AK0010001	21.989	N	Low
AK	Y	21989AK0010001	21.989	N	Low
AK	Y	21989AK0010001	21.989	N	Low
AK	Y	21989AK0010001	21.989	N	Low
AK	Y	21989AK0010001	21.989	N	Low
AK	Y	21989AK0010001	21.989	N	Low
AK	Y	21989AK0010001	21.989	N	Low
AK	Y	21989AK0010001	21.989	N	Low
AK	Y	21989AK0010001	21.989	N	Low

```sql
select * from crosswalk2016
```

alk2016 1

* from crosswalk2016

crosswalk2016_state	crosswalk2016_dentalplan	crosswalk2016_planid_2015	crosswalk2016_issuerid_2015	crosswalk2016_multistateplan_2015	crosswalk2016_metallevel_2015
OR	N	10091OR0360004	10.091	N	Bronze
OR	N	10091OR0360004	10.091	N	Bronze
OR	N	10091OR0360004	10.091	N	Bronze
OR	N	10091OR0360004	10.091	N	Bronze
OR	N	10091OR0360004	10.091	N	Bronze
OR	N	10091OR0360004	10.091	N	Bronze
OR	N	10091OR0360004	10.091	N	Bronze
OR	N	10091OR0360004	10.091	N	Bronze
OR	N	10091OR0360004	10.091	N	Bronze
OR	N	10091OR0360004	10.091	N	Bronze

```
select * from network n
```

network_businessyear	network_statecode	network_issuerid	network_sourcename	network_versionnum	network_importdate	network_issuerid2	network_statecode2
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	AK
2.014	AK	38.344	HIOS	6	2013-08-28 08:15:53	38.344	AK
2.014	AK	38.536	HIOS	2	2013-08-01 12:48:00	38.536	AK
2.014	AK	42.507	HIOS	3	2013-09-02 11:39:25	42.507	AK
2.014	AK	73.836	HIOS	6	2014-04-18 11:49:29	73.836	AK
2.014	AK	73.836	HIOS	6	2014-04-18 11:49:29	73.836	AK
2.014	AK	74.819	HIOS	7	2014-01-21 08:29:49	74.819	AK
2.014	AK	84.859	HIOS	1	2013-06-06 10:50:48	84.859	AK
2.014	AL	12.538	HIOS	8	2014-01-21 08:29:49	12.538	AL
2.014	AL	12.538	HIOS	8	2014-01-21 08:29:49	12.538	AL

```
select * from planattributes p
```

planattributes_businessyear	planattributes_statecode	planattributes_issuerid	planattributes_sourcename	planattributes_versionnum	planattributes_importdate
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49
2.014	AK	73.836	HIOS	6	2014-04-18 11:49:29
2.014	AK	73.836	HIOS	6	2014-04-18 11:49:29
2.014	AK	73.836	HIOS	6	2014-04-18 11:49:29
2.014	AK	73.836	HIOS	6	2014-04-18 11:49:29

```
select * from rate
```

rate_businessyear	rate_statecode	rate_issuerid	rate_sourcename	rate_versionnum	rate_importdate	rate_issuerid2	rate_federaltin	rate_rateeffectivedate
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01
2.014	AK	21.989	HIOS	6	2014-03-19 07:06:49	21.989	93-0438772	2014-01-01

```
select * from servicearea s
```

servicearea_businessyear	servicearea_statecode	servicearea_issuerid	servicearea_sourcename	servicearea_versionnum	servicearea_importdate	servicearea_issuerid2
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444
2.014	PA	22.444	HIOS	9	2014-01-21 08:29:49	22.444

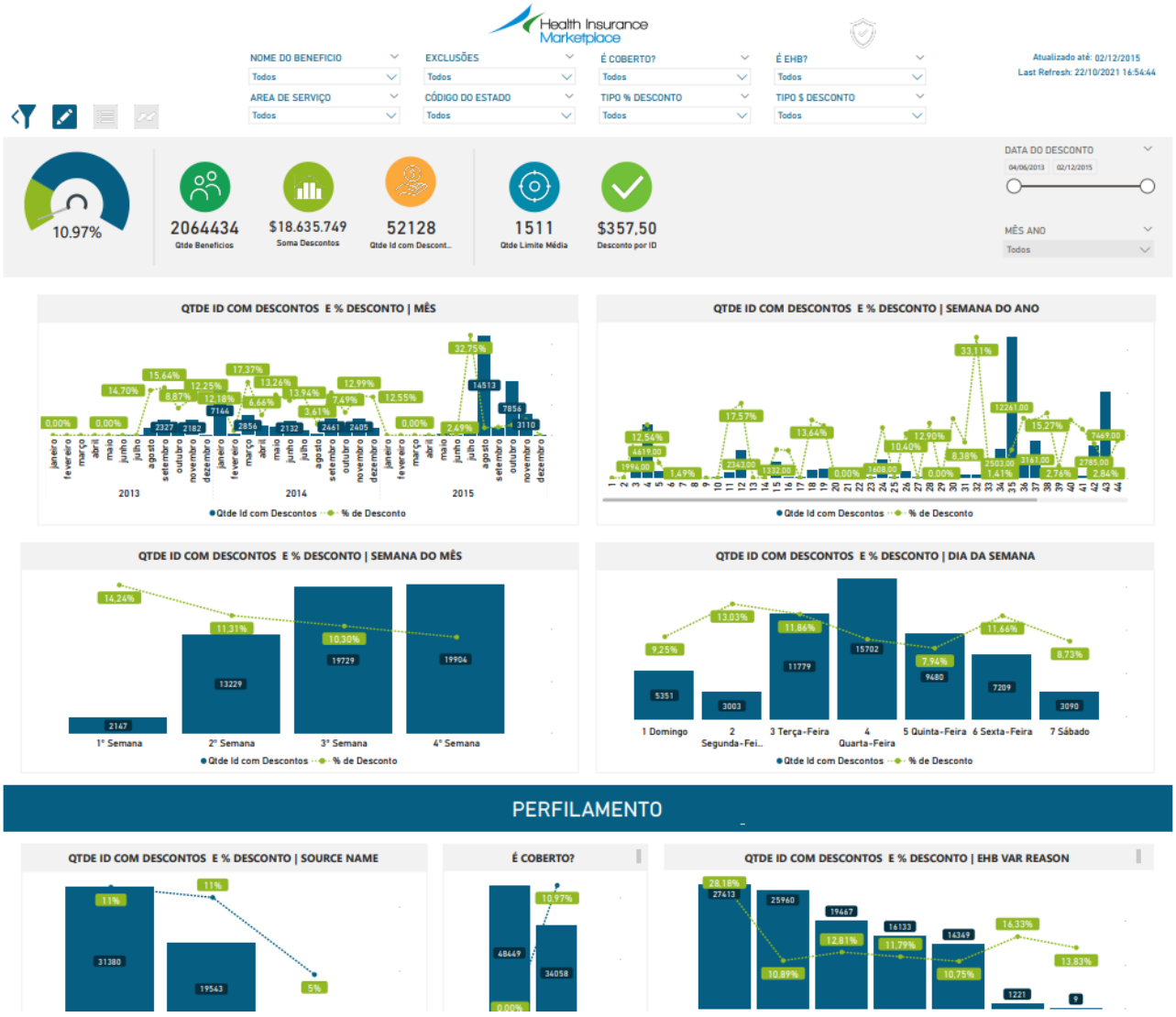
```
select * from merge_rate_planattributes mrp
```

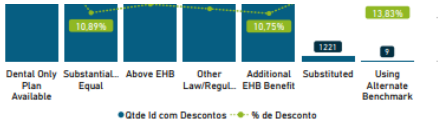
rate_planattributes 1

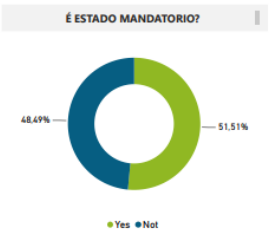
* from merge_rate_planattributes mrp

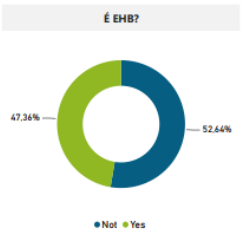
rate_statecode	rate_businessyear	rate_individualrate
AK	2.014	29,2583364486
AK	2.015	19,6268901569
AK	2.016	36,7223066755
AK	2.014	29,2583364486
AK	2.015	19,6268901569
AK	2.016	36,7223066755

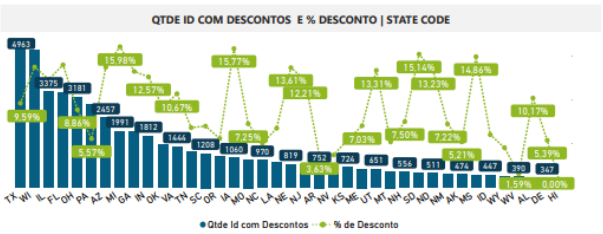
APÊNDICE B – DASHBOARD COMPLETO NO POWER BI











QTDE DE ID, QTDE DE BENEFICIOS, % DESCONTO, SOMA DESCONTO, DESCONTO POR ID, QTDE LIMITE MÉDIA | ESTADO E CIDADE

StateCode	Qtde Id com Descontos	Qtde Beneficios	% de Desconto	Soma Descontos	Desconto por ID	Qtde Limite Média
⊟ WI	4836	218530	13,69%	$157.615,00	$3,02	1489
Exclusive	1420	48640	10,12%	$0,00	$0,00	835
Health Tradition North West	401	30880	13,14%	$8.700,00	$0,17	364
Unity UW Health	765	27137	13,21%	$0,00	$0,00	655
Dean Health Plan HMO Network	317	18327	7,37%	$49.840,00	$0,96	309
SHP Broad Network	238	17994	18,01%	$7.200,00	$0,14	229
GundersenOne	133	15807	18,89%	$0,00	$0,00	359
GHC-SCW Small Group Service Area	307	15051	23,43%	$0,00	$0,00	5900
Managed Health Services	191	10068	17,12%	$0,00	$0,00	8227
CGHC Service Area	146	9078	2,82%	$0,00	$0,00	172
South Central WI	185	5767	8,85%	$26.000,00	$0,50	1109
Compcare Health Serv Ins Co (Anthem BCBS)	136	5730	3,58%	$44.500,00	$0,85	2333
Dane	141	3298	3,27%	$12.550,00	$0,24	1277
Medica Applause	113	3155	20,78%	$0,00	$0,00	300
Wisconsin - IEX and SHOP	67	2666	2,48%	$8.825,00	$0,17	222
WI	15	1676	17,18%	$0,00	$0,00	224
NH Northeast Wisconsin	44	1380	0,00%	$0,00	$0,00	3789
Total	**52128**	**2064434**	**10,97%**	**$18.635.749,00**	**$357,50**	**1511**

QTDE DE ID, QTDE DE BENEFICIOS, % DESCONTO, SOMA DESCONTO, DESCONTO POR ID, QTDE LIMITE MÉDIA | NOME DO BENEFICIO

BenefitName	Qtde Id com Descontos	Qtde Beneficios	% de Desconto	Soma Descontos	Desconto por ID	Qtde Limite Média
Orthodontia - Adult	51712	76150	21,69%	$0,00	$0,00	10
Major Dental Care - Adult	48191	70205	39,83%	$1.782,00	$0,03	8682
Basic Dental Care - Adult	47473	69052	35,85%	$1.782,00	$0,03	8346
Routine Dental Services (Adult)	47179	68581	0,18%	$855,00	$0,02	7096
Long-Term/Custodial Nursing Home Care	45249	65704	0,00%			
Abortion for Which Public Funding is Prohibited	43166	62506	16,33%	$0,00	$0,00	20
Weight Loss Programs	43043	61738	8,06%	$0,00	$0,00	169
Acupuncture	42784	62409	7,88%	$120,00	$0,00	1823
Cosmetic Surgery	40265	58542	9,46%	$12.250,00	$0,24	10
Non-Emergency Care When Traveling Outside the U.S.	36044	51022	15,20%	$1.805,00	$0,03	
Bariatric Surgery	35894	51243	15,53%	$105.632,00	$2,03	10
Orthodontia - Child						
Major Dental Care - Child	33825	48422	22,75%	$3.040,00	$0,06	11
Private-Duty Nursing	33643	47538	11,15%	$1.275,00	$0,02	905
Basic Dental Care - Child	33593	48166	17,19%	$3.040,00	$0,06	21
Total	**52128**	**2064434**	**10,97%**	**$18.635.749,00**	**$357,50**	**1511**

NUVEM DE PALAVRAS DE EXPLANAÇÕES