Trabalho de Conclusão de Curso

Solução Integrada de Engenharia de Dados Para Gestão e Análise, Utilizando uma Abordagem Open Source e de Baixo Custo

Aldemir Melo Rocha Filho

Orientador:

Prof. Dr. Erick de Andrade Barboza

Maceió, Novembro de 2024

Aldemir Melo Rocha Filho

Solução Integrada de Engenharia de Dados Para Gestão e Análise, Utilizando uma Abordagem Open Source e de Baixo Custo

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Engenharia de Computação do Instituto de Computação da Universidade Federal de Alagoas.

Orientador:

Prof. Dr. Erick de Andrade Barboza

Maceió, Novembro de 2024

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecária: Helena Cristina Pimentel do Vale – CRB4 –661

R672s Rocha Filho, Aldemir Melo.
Solução integrada de engenharia de dados para gestão e análise, utilizando uma abordagem open source e de baixo custo / Aldemir Melo Rocha Filho. - 2024.
42 f : il.

Orientador: Erick de Andrade Barboza.
Monografia (Trabalho de Conclusão de Curso em Engenharia de Computação) – Universidade Federal de Alagoas, Instituto de Computação. Maceió, 2024.

Bibliografia: f. 41-42.

1. Gestão de dados. 2. Código aberto. 3. Democratização da informação. 4. Armazenamento de dados. 5. Processamento de dados. I. Título.

CDU: 004.93

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Engenharia de Computação do Instituto de Computação da Universidade Federal de Alagoas, aprovada pela comissão examinadora que abaixo assina.

Documento assinado digitalmente
gov.br
ERICK DE ANDRADE BARBOZA
Data: 27/11/2024 09:23:23-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Erick de Andrade Barboza - Orientador
Instituto de Computação
Universidade Federal de Alagoas

Documento assinado digitalmente
gov.br
JAIRO RAPHAEL MOREIRA CORREIA DE SOUZA
Data: 27/11/2024 10:47:53-0300
Verifique em https://validar.iti.gov.br

Prof. Me. Jairo Raphael Moreira Correia de Souza - Examinador
Instituto de Computação
Universidade Federal de Alagoas

Documento assinado digitalmente
gov.br
BALDOINO FONSECA DOS SANTOS NETO
Data: 27/11/2024 17:53:21-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Baldoino Fonseca dos Santos Neto - Examinador
Instituto de Computação
Universidade Federal de Alagoas

Maceió, Novembro de 2024

Agradecimentos

Gostaria de agradecer primeiramente a Deus e aos meus pais por ter chegado até aqui. Sem dúvida, nada disso seria possível sem o apoio deles. Cada sacrifício e cada escolha, sempre visando proporcionar a mim e às minhas irmãs o melhor ambiente para nos desenvolvermos, tanto no âmbito pessoal quanto profissional, foram essenciais e continuam sendo em vários aspectos da minha vida. Sou grato à minha companheira, Rebeca Regina, por sua paciência e apoio constante ao longo desses anos. Em todos os momentos, sejam eles de alegria ou de dificuldade, sua presença fez toda a diferença.

Agradeço ao meu orientador, Prof. Dr. Erick de Andrade Barboza, por todo o apoio dedicado à elaboração deste trabalho, bem como em outros momentos ao longo da graduação. Nesse mesmo sentido, gostaria de agradecer aos professores Thiago Cordeiro, Ícaro Bezerra, Márcio Ribeiro, Baldoino Fonseca e José Carlos, que foram fundamentais em momentos marcantes da minha trajetória acadêmica. Alguns, em especial, no contexto do Laboratório de Engenharia e Sistemas (EASY), onde tive a oportunidade de atuar em projetos que muito me engrandeceram profissionalmente, com experiências que ultrapassaram os limites da sala de aula. Também deixo meus agradecimentos a todos os outros profissionais do nosso instituto, que tornaram o dia a dia possível.

Da mesma forma, gostaria de agradecer aos meus companheiros de serviço, que me apoiaram com ideias e soluções ao longo deste trabalho, João Cabral, Yuri Dimitri e Alisson Sandes.

Por fim, e não menos importante, agradeço a todos os amigos que cultivei ao longo da graduação, pois foram essenciais e de grande valor para superar as barreiras encontradas nesse caminho: Bruna Damaris, Heitor Almeida, José Silvino, Tayco Murilo, Sandoval Almeida, Darlysson Olimpio e Derek Nielsen.

"A felicidade é o sentido e o propósito da vida, o único objetivo e a finalidade da existência humana."

– Aristóteles

Resumo

A crescente geração de dados, impulsionada pela transformação digital e pela disseminação de tecnologias como a Internet das Coisas, exige soluções inovadoras e acessíveis para a gestão eficiente das informações. A democratização do acesso à análise de dados, por meio da adoção de tecnologias open source e de baixo custo, permite que organizações de todos os portes e setores extraiam informaçoes valiosas para otimizar processos, tomar decisões mais assertivas e impulsionar o crescimento dos negócios. Ao combinar a escalabilidade e flexibilidade das soluções open source com a acessibilidade e o custo reduzido, essa abordagem possibilita que um número maior de negócios aproveite o potencial dos seus dados, impulsionando a inovação e a competitividade.

Este trabalho apresenta a implementação de um ambiente de gestão de dados baseado em ferramentas de código aberto, visando criar uma solução eficiente e de baixo custo para a construção de um lakehouse. O estudo aborda a crescente necessidade de gerenciar grandes volumes de dados, propondo um ecossistema acessível para organizações de diversos setores, integrando as funcionalidades de um Data Lake e um Data Warehouse para permitir a coleta, organização e análise de dados de forma estruturada.

Durante o desenvolvimento, um dos desafios foi integrar as diversas ferramentas open source, garantindo compatibilidade e segurança no processo de gerenciamento de dados. A abordagem modular adotada facilitou a configuração e automação das etapas, assegurando a integridade dos dados e otimizando o desempenho geral do ambiente, consolidando-se como uma solução eficaz para o armazenamento e processamento de grandes volumes de dados.

Os resultados demonstram a viabilidade técnica e econômica do ambiente, o qual foi aplicado ao Sistema Eletrônico de Informações (SEI) como estudo de caso em questão. O ambiente permitiu a criação de uma infraestrutura confiável para a organização, armazenamento e processamento eficiente de dados, oferecendo uma base otimizada para suportar análises detalhadas e melhorar a gestão de informações. A comparação com o Azure Synapse, uma solução amplamente utilizada no mercado para gerenciamento de dados, evidenciou não apenas a economia significativa, mas também a capacidade do ambiente open source de oferecer uma solução robusta sem exigir altos investimentos, tornando-se uma alternativa viável para instituições com recursos limitados.

Palavras-chave: democratização, escalabilidade, flexibilidade, competitividade, gestão de dados, código aberto, lakehouse, Data Lake, Data Warehouse, viabilidade, SEI, infraestrutura, armazenamento, processamento, Azure Synapse, economia, open source, integrar, automação.

Abstract

The increasing generation of data, driven by digital transformation and the spread of technologies like the Internet of Things, demands innovative and accessible solutions for efficient information management. The democratization of data analysis through the adoption of open source and low-cost technologies enables organizations of all sizes and sectors to extract valuable insights to optimize processes, make more accurate decisions, and drive business growth. By combining the scalability and flexibility of open-source solutions with accessibility and reduced costs, this approach allows a greater number of businesses to harness the potential of their data, fostering innovation and competitiveness.

This work presents the implementation of a data management environment based on open-source tools, aiming to create an efficient and low-cost solution for building a lakehouse. The study addresses the growing need to manage large volumes of data, proposing an accessible ecosystem for organizations in various sectors, integrating the functionalities of both a Data Lake and a Data Warehouse to enable the collection, organization, and analysis of data in a structured manner.

During the development, one of the challenges was integrating the various open-source tools, ensuring compatibility and security in the data management process. The modular approach adopted facilitated the configuration and automation of the stages, ensuring data integrity and optimizing the overall performance of the environment, consolidating itself as an effective solution for storing and processing large volumes of data.

The results demonstrate the technical and economic feasibility of the environment, which was applied to the Electronic Information System (SEI) as a case study. The environment enabled the creation of a reliable infrastructure for the organization, storage, and efficient processing of data, providing an optimized foundation to support detailed analyses and improve information management. The comparison with Azure Synapse, a widely used solution in the market for data management, highlighted not only significant cost savings but also the ability of the open-source environment to offer a robust solution without requiring large investments, making it a viable alternative for institutions with limited resources.

Keywords: democratization, scalability, flexibility, competitiveness, data management,

open-source tools, lakehouse, Data Lake, Data Warehouse, feasibility, SEI, infrastructure, storage, processing, Azure Synapse, cost savings, integrating, automation.

Sumário

Lista de Abreviaturas e Siglas

DW	*Data Warehouse*
DL	*Data Lake*
IoT	*Internet of Things* (Internet das Coisas)
SEI	*Sistema Eletrônico de Informações*
ETL	*Extract, Transform, Load* (Extrair, Transformar, Carregar)
ELT	*Extract, Load, Transform* (Extrair, Carregar, Transformar)
OLAP	*Online Analytical Processing* (Processamento Analítico Online)
OLTP	*Online Transaction Processing* (Processamento de Transações Online)
API	*Application Programming Interface* (Interface de Programação de Aplicações)
DWU	*Data Warehouse Unit* (Unidade de Data Warehouse)
LGPD	*Lei Geral de Proteção de Dados*

Lista de Figuras

1 Introdução

1.1 Contexto e Motivação

A quantidade de dados gerados por cada pessoa tem crescido exponencialmente ao longo do tempo, refletindo as mudanças sociais, tecnológicas e culturais que moldaram a humanidade.

Ao longo de séculos, foi visto o surgimento e a evolução de diversas formas de registro e comunicação, desde as pinturas rupestres, manuscritos até a invenção da imprensa, que possibilitou a produção em massa de livros e jornais. Cada avanço tecnológico expandiu as fronteiras da geração de dados, tornando a informação mais acessível e democrática.

No entanto, atualmente a geração de dados tem atingido novos patamares. A disseminação da Internet e das redes sociais transformou cada indivíduo em um potencial produtor de dados, compartilhando suas experiências, opiniões e preferências em plataformas online, em um volume sem precedentes. Além disso, a Internet das Coisas (IoT) trouxe uma nova dimensão à geração de dados, com dispositivos inteligentes interconectados coletando e transmitindo informações em tempo real.

Nesse contexto de abundância de dados, torna-se um desafio gerenciar e extrair valor de maneira responsável e eficiente, ações essenciais para transformar esse material em conhecimentos acionáveis, facilitando a tomada de decisões informadas. Um ambiente de gestão de dados eficaz, como um data warehouse, data lake, entre outros, desempenham um papel fundamental nesse processo, permitindo a coleta, organização e/ou análise sistemática de dados de diversas fontes.

Ainda nesse sentido, para tornar esse processo ainda mais viável e abrangente, é interessante que a gestão de dados seja acessível em termos de custo. Reduzir os custos associados não apenas torna o processo mais atrativo, mas também o torna mais inclusivo, permitindo que empresas e indivíduos de diversas classes socioeconômicas tenham acesso aos benefícios da análise de dados. Isso não só promove um amadurecimento na gestão de dados para uma varie-

dade de setores e públicos, mas também democratiza o acesso ao conhecimento e na tomada de decisões assertivas derivadas dos dados.

Porém, é importante destacar que, junto com o potencial para boas decisões, a má gestão de dados pode acarretar em consequências negativas. Desde preocupações com privacidade e segurança até questões éticas relacionadas ao uso inadequado de informações pessoais, sendo ações fundamentais para garantir a confiança e o respeito dos usuários.

Portanto, à medida que continuamos atuando e gerindo esses dados, é de suma importância investir em práticas de gestão de dados sólidas e éticas, a fim de desbloquear todo o potencial que essa riqueza de informações pode oferecer, ao mesmo tempo em que protegemos os direitos e interesses daqueles que contribuem para ela.

1.2 Objetivos

1.2.1 Objetivos Gerais

Avaliar a viabilidade e eficácia de um ambiente de gestão de dados de baixo custo capaz de realizar o processo de ELT e criar um lakehouse.

1.2.2 Objetivos Específicos

- Investigar ferramentas de código aberto aptas para serem usadas nos diversos passos do ELT.
- Definir as etapas de implementação, desde a seleção das ferramentas até a integração e operação do ambiente proposto.
- Validar a viabilidade técnica e a aplicabilidade prática do ambiente de gestão de dados proposto.

Dessa maneira, foram investigadas e analisadas diversas ferramentas com o objetivo de criar um ambiente que atendesse ao cenário descrito, visando o baixo custo e os resultados técnicos esperados, desprendendo-se de soluções premium e de renome no mercado. O intuito deste trabalho é fornecer um ecossistema capaz de elaborar um lakehouse, um ambiente eficiente no que diz repeito ao gerenciamento de dados e que, graças a facilidade de implantação nos moldes descritos aqui, democratiza o desenvolvimento de uma cultura de dados aos mais diversos públicos. Compor este ambiente foi em si um grande desafio, não só pelo objetivo de manter o custo baixo, mas também visando aplicações que atendessem a diversas exigências, como segurança, praticidade, disponibilidade de documentação, suporte e comunidade ativa. Serão também feitas análises comparativas de custos, entre o ambiente proposto e o Azure Synapse, uma opção de mercado bastante difundida.

Superada a etapa de seleção das ferramentas e construção da arquitetura do ambiente, as aplicações deste ecossistema deve atuar em sintonia na elaboração do lakehouse, o qual tem como característica a combinação de elementos de um Data Lake e de um Data Warehouse, sendo nesta etapa onde é realizado todo o processo de extração, carregamento e transformação (ELT), o qual terá os conceitos detalhados ao longo dos próximos capítulos, assim como também o porque de ser ELT e não ETL, o mais convencional.

Ainda neste trabalho, foi realizado um estudo de caso prático selecionando um sistema amplamente disseminado, preferencialmente público, para que as vantagens de um ambiente de gestão de dados acessível e de baixo custo possam ser disponibilizadas tanto ao serviço público quanto à sociedade. O escolhido foi o "SEI!", Sistema Eletrônico de Informações, criado pelo Tribunal Regional Federal da 4ª Região (TRF4), uma solução para a gestão de documentos e processos eletrônicos, que visa aumentar a eficiência administrativa. O produto do ambiente desenvolvido, mais especificamente as tabelas finais, possibilitam, entre outras funcionalidades, o controle de atrasos de processos e a verificação daqueles que se encontram parados, entre outras visualizações que podem ser elaboradas com a utilização das demais tabelas geradas.

1.3 Organização do Trabalho

Os demais capítulos deste estudo estão organizados com o objetivo de descrever cada etapa do desenvolvimento do trabalho, fornecendo detalhes teóricos e aplicados na implementação do ambiente proposto. O segundo capítulo discute a fundamentação teórica, abordando conceitos e tecnologias relevantes, os quais são fundamentais para o entendimento das escolhas e aplicações realizadas durante a formulação da arquitetura do contexto definido. O terceiro capítulo apresenta a metodologia utilizada, detalhando desde o levantamento dos requisitos até a implementação e o estudo de caso prático, evidenciando a efetividade do sistema. O quarto capítulo apresenta os resultados obtidos, como a arquitetura implementada, análise de desempenho e comparação entre ferramentas, em conformidade com os objetivos propostos. Por fim, o quinto capítulo conclui o trabalho, sintetizando os resultados e propondo futuras contribuições.

2

Fundamentação Teórica

Neste capítulo, serão definidos os conceitos fundamentais para a compreensão da construção de um ambiente de gestão de dados, com ênfase na criação de um lakehouse. Esses conceitos servirão como base tanto para entender a necessidade desse tipo de ambiente quanto para explicar tecnicamente os princípios aplicados.

2.1 Gerenciamento e Governança de Dados

Inicialmente, é importante definirmos como os termos "Gerenciamento de dados" e "Governança de dados" conversam entre si. À medida que a disponibilidade e a combinação de dados continuam a crescer, surge uma demanda recorrente por governança de dados, no que diz respeito ao exercício da autoridade, um conjunto de políticas, normas, práticas e estruturas organizacionais que definem como os dados devem ser gerenciados e utilizados. A governança de dados capacita organizações a assegurar uma gestão adequada de dados e informações, garantindo que informações pertinentes sejam entregues às pessoas corretas no momento adequado. Além disso, já o gerenciamento de dados, refere-se por exemplo às práticas e processos técnicos utilizados para coletar, armazenar, organizar, manter e utilizar os dados de uma organização, sempre priorizando a segurança dos mesmos(NAMBIAR; MUNDRA, 2022; BROUS; JANSSEN; VILMINKO-HEIKKINEN, 2016).

Dentre alguns princípios nesse contexto, podemos destacar os cuidados relacionados ao armazenamento de dados, como qualidade, integração entre as fontes, segurança, políticas, normas e responsabilidades dos envolvidos em todo o processo. Dessa forma, tratando de governança e do gerenciamento dos dados em si, será considerado aqui o primeiro como uma visão mais macro de todo o processo e, o segundo, uma visão mais micro. É evidente que são conceitos interligados e que caminham juntos no estabelecimento de um ambiente saudável na gestão de dados, os quais atuam em conformidade com as políticas e regulamentos de negócios

estabelecidos, como na manipulação, armazenamento e tratamentos dos dados propriamente ditos.

2.2 Ferramentas Open Source

Como um dos objetivos principais deste trabalho é a criação de um ambiente sem grandes impactos financeiros, é essencial definir do que se trata uma ferramenta "Open Source", visto que foi critério fundamental nas escolhas das aplicações que compõem este ecossistema.

Ferramentas Open Source, ou de código aberto, são softwares cujo código-fonte é disponibilizado publicamente. Isso significa que qualquer pessoa pode visualizar, modificar e distribuir o código-fonte do software conforme suas necessidades. A filosofia por trás do movimento open source promove a colaboração e a transparência, permitindo que desenvolvedores de todo o mundo contribuam para o aprimoramento das ferramentas, corrigindo bugs, adicionando funcionalidades e aumentando a segurança (FERREIRA, 2005; FUGGETTA, 2003).

Uma das características mais comuns das ferramentas open source é a licença sob a qual são distribuídas. Existem diversas licenças open source, como a GNU General Public License (GPL), a Apache License, a MIT License, entre outras. Essas licenças garantem que os direitos de uso, modificação e redistribuição sejam mantidos, desde que certas condições sejam atendidas. Por exemplo, a GPL exige que qualquer software derivado de um projeto GPL também seja distribuído sob a mesma licença, promovendo a continuidade da filosofia open source (FERREIRA, 2005; FUGGETTA, 2003).

A adoção de ferramentas open source apresenta várias vantagens, especialmente em ambientes de gestão de dados, como o que se pretende criar neste trabalho. Entre os principais benefícios estão:

- **Redução de Custos:** Como o software open source é geralmente gratuito, as organizações podem economizar significativamente em licenças de software, direcionando recursos para outras áreas críticas, como hardware e treinamento.
- **Flexibilidade e Customização:** Acesso ao código-fonte permite que as ferramentas sejam adaptadas às necessidades específicas do projeto. Isso é particularmente útil em gestão de dados, onde os requisitos podem variar amplamente entre diferentes organizações e projetos.
- **Segurança e Transparência:** Com o código-fonte aberto para inspeção, vulnerabilidades de segurança podem ser identificadas e corrigidas rapidamente pela comunidade global de desenvolvedores. Essa transparência também aumenta a confiança dos usuários nas ferramentas utilizadas.

- **Comunidade e Suporte Colaborativo:** Ferramentas open source geralmente possuem comunidades ativas que fornecem suporte, documentação, e soluções para problemas comuns. Esta rede de colaboração pode ser uma valiosa fonte de conhecimento e inovação.
- **Independência de Fornecedores:** Ao contrário de soluções proprietárias, as ferramentas open source evitam o lock-in (restrição) com fornecedores específicos, oferecendo maior liberdade para trocar de ferramentas ou fornecedores sem custos exorbitantes ou complicações técnicas.

Portanto, a escolha de ferramentas open source para a criação de um ambiente de gestão de dados não apenas atende ao objetivo de minimizar custos, mas também proporciona um ecossistema flexível, seguro e suportado por uma comunidade global de desenvolvedores e usuários. Essas características são essenciais para a construção de uma infraestrutura eficiente e sustentável, alinhada com os princípios e objetivos deste trabalho.

2.3 Data Warehouse, Data Lake e Lakehouse

Antes de iniciarmos a sessão sobre lakehouse, é importante definirmos aqui do que se trata um data warehouse e um data lake, dessa forma, teremos insumos suficientes para definição de uma arquitetura lakehouse , a qual é aplicada no ambiente proposto mais a seguir.

Data warehouse e data lake representam um dos tipos de plataformas de dados de mais destaque, como podemos ver no comparativo abaixo entre ambas as ferramentas na tabela 2.1, os conceitos referenciados na mesma serão descritos no decorrer do texto.

Tabela 2.1: From: The lakehouse: State of the Art on Concepts and Technologies

Property	**data warehouse**	**data lake**
Typical workloads	Reporting, OLAP	Advanced analytics
Users	Business users, data analysts	Data scientists
Data access	Query language, data export	Direct access on storage
Data independence	Physical, partly logical	Weak
Guarantees	ACID	Weak
Schema enforcement	Explicitly on all data operations (“on write”)	Implicitly when data is read (“on read”)
Type of data	Mainly structured	All types
Data addressing	Via relational structures	Via metadata
Data granularity	Aggregated	Raw and aggregated
Data storage	RDBMS	Polyglot
Flexibility	Low	High
Management features	Advanced	Rudimentary

fonte: Adaptado (SCHNEIDER et al., 2024)

2.3.1 Data Warehouse

A integração de dados de diversas fontes, como sistemas transacionais, bancos de dados relacionais, APIs e outras fontes, constitui a base de um data warehouse. Este repositório centralizado é projetado especificamente para facilitar a análise de dados, business intelligence (BI), mineração de dados, inteligência artificial (IA) e aprendizado de máquina (NAMBIAR; MUNDRA, 2022; CHANDRA; GUPTA, 2018), como pode ser analisado na figura 1.

No cenário atual, os data warehouses desempenham um papel muito importante, proporcionando às empresas a capacidade de analisar grandes quantidades de dados de forma eficiente. Isto, por sua vez, facilita a extração de informações valiosas e permite a tomada de decisões baseadas. Ao armazenar e estruturar dados de forma eficaz, esses sistemas permitem a criação de relatórios, paineis e outras ferramentas analíticas, garantindo ao mesmo tempo um desempenho ideal para vários usuários simultaneamente (NAMBIAR; MUNDRA, 2022; CHANDRA; GUPTA, 2018).

Além disso, considerando que o data warehouse é constituído por diversas fontes de dados, cada uma com seus contextos, regras e usuários específicos interessados, esta solução é dividida em data marts, que são subconjuntos do DW. Cada um desses subconjuntos é focado em uma determinada linha de negócios, departamento ou área, dessa forma o acesso fica ainda mais prático, rápido e objetivo. Exemplo de data marts de um data warehouse, podem ser: financeiro, pessoas, vendas, marketing e etc.

Surgindo como uma solução mais conveniente para análise de dados em larga escala, evoluindo a partir de bancos de dados relacionais, os data warehouses utilizam esquemas de dados bem definidos e multidimensionais (KIMBALL; ROSS, 2011), que são continuamente aplicados em todas as operações de dados, garantindo propriedades ACID (HAERDER; REUTER, 1983) e frequentemente oferecendo capacidades avançadas de gerenciamento. As propriedades ACID, referem-se às capacidades de atomicidade dos dados, consistência, isolabilidade e durabilidade. É comum atualmente que muitos data warehouses atuem na nuvem (DAGEVILLE et al., 2016), onde é proporcionado ao ambiente maior escalabilidade e menor esforço operacional. Os data warehouses são mais adequados para responder a perguntas de análises previamente conhecidas, como no contexto de relatórios e processamento analítico online (OLAP)(NAMBIAR; MUNDRA, 2022), o que será detalhado mais a seguir.

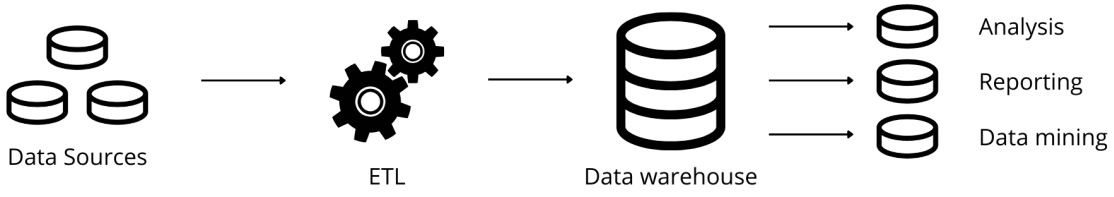


Figura 1: Fluxo data warehouse
Fonte: Autor.

Arquitetura de um Data Warehouse

O data warehouse têm uma arquitetura de três camadas, que consiste em:

- **Camada Inferior:** Esta camada se concentra na extração de dados de diversas fontes, limpando e transformando esses dados em um formato consistente e utilizável. O principal processo utilizado aqui é o ETL (Extract, Transform, Load), que garante que os dados brutos se tornem dados confiáveis e bem formatados. Além disso, também é nessa etapa que são inseridos os metadados, que nada mais são que dados sobre dados, como por exemplo autores, datas de carga, datas do ETL, versão e etc. O servidor do data warehouse geralmente utiliza um banco de dados relacional, que é onde são armazenados esses dados processados (NAMBIAR; MUNDRA, 2022).

- **Camada Intermediária:** Consiste em um servidor OLAP(Online Analytical Processing), fornecendo uma interface para cientistas de dados e analistas acessarem os dados preparados de forma rápida e prática da etapa anterior (NAMBIAR; MUNDRA, 2022).

- **Camada Superior:** Esta é a camada mais familiar da arquitetura do data warehouse, é representada por algum tipo de interface que fornece ao usuário condições para tomada de decisões. Por exemplo, relatórios, dashboards, gráficos entre outras opções de apresentação (NAMBIAR; MUNDRA, 2022).

Diferença OLAP e OLTP

A principal diferença entre OLAP(do inglês, Online Analytical Processing) e OLTP(do inglês, Online Transaction Processing) está no nome: OLAP é analítico por natureza, e OLTP é transacional.

OLAP é focado em realizar análises multidimensionais em alta velocidade em grandes volumes de um armazenamento de dados como um data warehouse, unificado e centralizado, que contém tanto dados históricos quanto transacionais. Este tipo de ferramenta consegue validar os dados em diferentes níveis de validação, por conseguir navegar em diferentes níveis de granularidade das informações (NAMBIAR; MUNDRA, 2022)..

Já o OLTP, é projetado para suportar aplicações orientadas a transações, processando transações recentes o mais rápido e precisamente possível. são usados pelos usuários em geral no dia a dia em seus processos e transações, gravação e leitura. Exemplos são: caixas eletrônicos, software de e-commerce, processamento de dados de pagamento com cartão de crédito, reservas online, sistemas de reserva e ferramentas de registro. Ou seja, o principal objetivo é eliminar ao máximo redundâncias (NAMBIAR; MUNDRA, 2022).

Esquemas e Tabelas Fato e Dimensões

Antes de detalhar os esquemas, é importante definir o que são as tabelas fatos e as tabelas dimensões:

- **Tabelas Fatos:** As tabelas fatos são componentes centrais em um esquema de banco de dados orientado a dados, como em um DW. Elas armazenam os dados transacionais e métricas de negócios que são analisadas para tomar decisões. É comum dados se repetirem nessas tabelas, por exemplo em uma tabela fato de vendas, assim como também é comum a presença de chaves estrangeiras nessa tabela. Cada linha em uma tabela fato representa uma transação ou evento de negócio específico que está sendo analisado (AUNG; LATT, 2006; LEVENE; LOIZOU, 2003).

- **Tabelas Dimensão:** As tabelas dimensão fornecem o contexto descritivo para as medidas contidas nas tabelas fatos. Elas contêm atributos detalhados que ajudam a categorizar, filtrar e agrupar os dados durante as análises. Seguindo o mesmo exemplo anterior, uma tabela dimensão de itens disponíveis à venda, os itens não se repetem e, para cada linha dessa tabela, existem as características de cada item. Nessa tabela existem os Identificadores únicos que são referenciados pelas chaves estrangeiras nas tabelas fatos, cada linha em uma tabela dimensão representa um membro dessa dimensão (AUNG; LATT, 2006; LEVENE; LOIZOU, 2003).

Já os esquemas se referem à maneira de organizar os dados dentro do data warehouse. As principais maneiras são: star schema(esquema estrela), snowflake(esquema floco de neve), Starflake e o esquema Galaxy.

- **Esquema Estrela:** É uma abordagem simples e direta para organizar dados em um DW. Ele é chamado de "estrela"porque seu design se assemelha a uma estrela, com uma tabela fato no centro cercada por várias tabelas dimensões desnormalizadas. É considerado o tipo de esquema mais simples e comum, os usuários se beneficiam da simplicidade e desempenho de criação e execução da consulta, uma vez que é reduzida quantidade de junções entre as tabelas (AUNG; LATT, 2006; LEVENE; LOIZOU, 2003), como pode ser observado no modelo da figura 2.

- **Esquema Floco de Neve:** O Esquema Floco de Neve é uma extensão do Esquema Estrela, onde as tabelas dimensões são normalizadas em múltiplas tabelas relacionadas. Ele é chamado de "floco de neve"porque a estrutura resultante parece um floco de neve com muitas ramificações. É um tipo de esquema menos utilizado e possui menos redundância entre as tabelas dimensões por serem normalizadas, porém, devido a existência de mais dimensões, aumenta o número de junções, o que reduz o desempenho da consulta e consequentemente também aumenta a complexidade de construção da mesma (AUNG; LATT, 2006; LEVENE; LOIZOU, 2003). Um exemplo pode ser analisado na figura 3.

- **Esquema Starflake:** O Esquema Starflake combina características do Esquema Estrela e do Esquema Floco de Neve. Ele apresenta uma estrutura híbrida em que algumas dimensões são desnormalizadas, como no Esquema Estrela, enquanto outras são normalizadas, como no Esquema Floco de Neve. Esse tipo de esquema é usado quando é necessário um equilíbrio entre simplicidade e redução de redundância. Apesar de proporcionar flexibilidade, ele pode aumentar a complexidade das consultas e o esforço de manutenção, dependendo do nível de normalização e desnormalização aplicado (IBM, 2024b).

- **Esquema Galaxy:** O Esquema Galaxy, também conhecido como "constelação de fatos", é utilizado em data warehouses que possuem múltiplas tabelas fato relacionadas. Nesse esquema, diferentes tabelas fato compartilham dimensões comuns, formando uma estrutura complexa que se assemelha a uma constelação. É especialmente útil para suportar múltiplos processos de negócio dentro de um mesmo data warehouse. No entanto, devido à sua complexidade, o Esquema Galaxy exige maior esforço de design e pode reduzir o desempenho de consulta, dependendo da quantidade de junções necessárias entre as tabelas (TURCAN; PEKER, 2022).

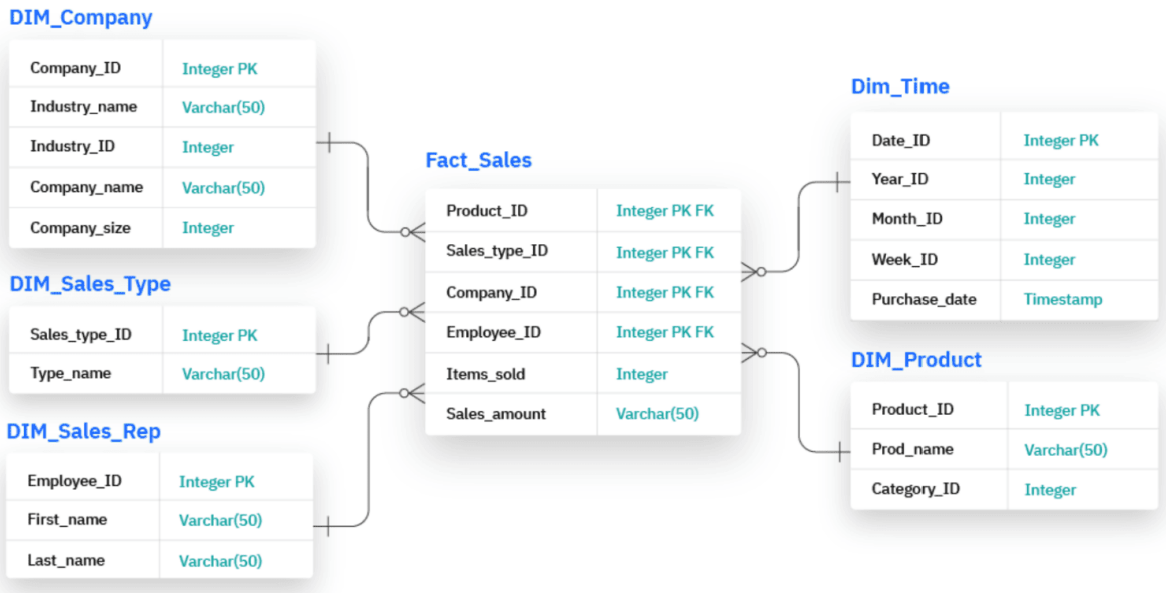


Figura 2: Esquema Estrela
Fonte: (IBM, 2024a).

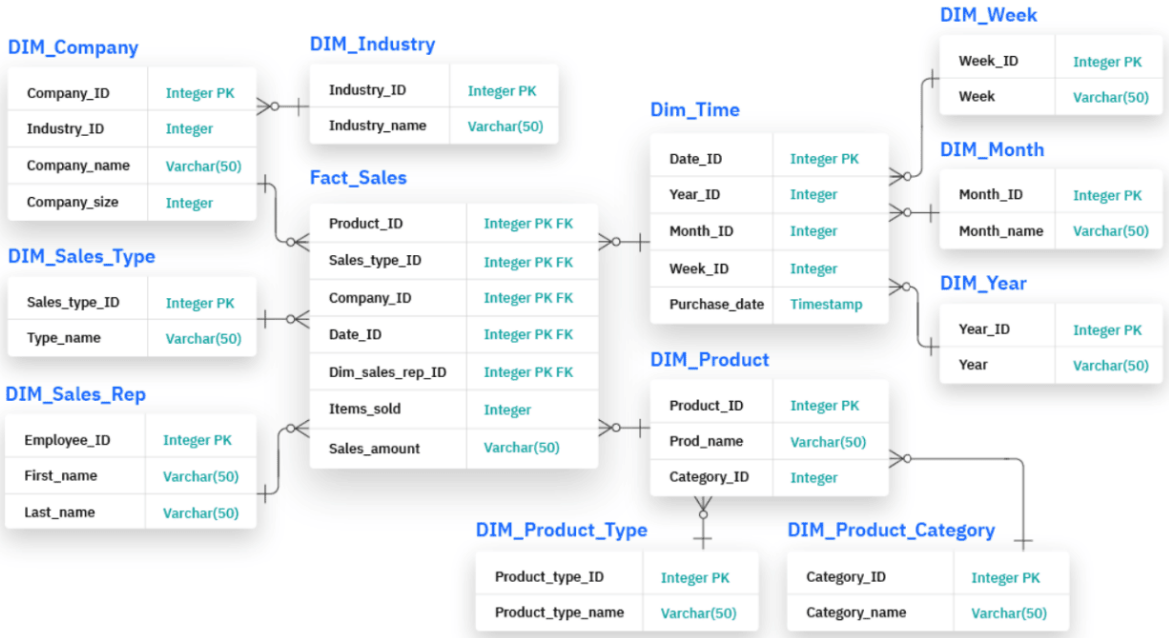


Figura 3: Esquema Floco de Neve
Fonte: (IBM, 2024a).

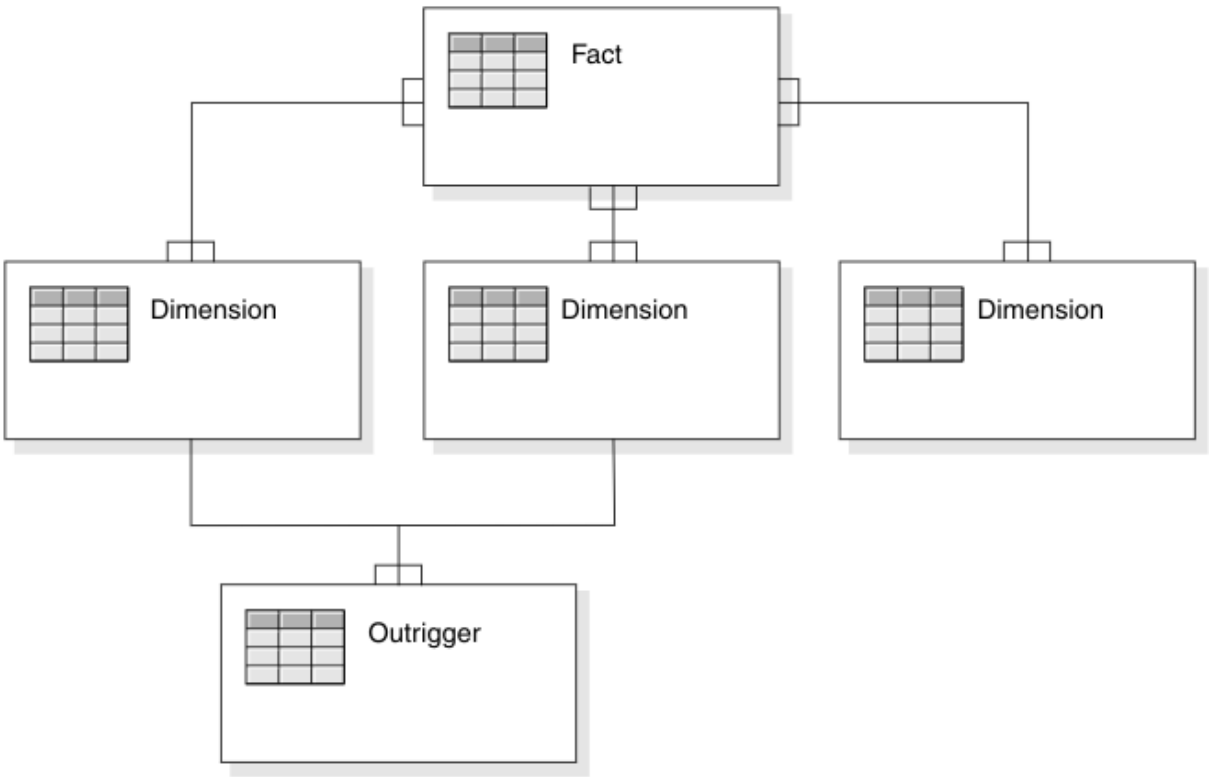


Figura 4: Esquema Starflake
Fonte: (IBM, 2024b).

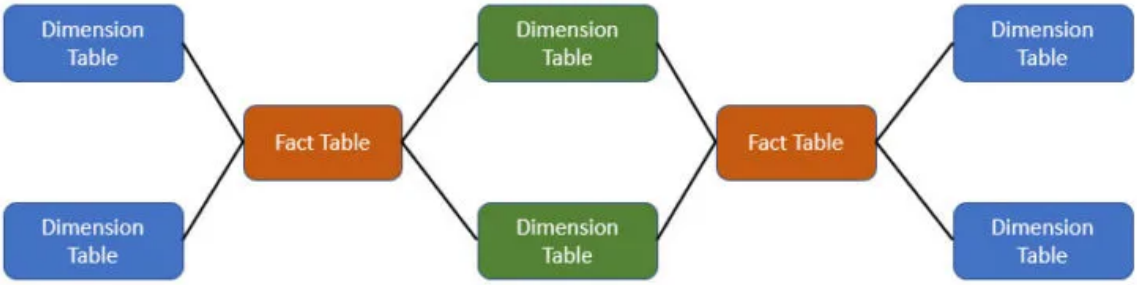


Figura 5: Esquema Galaxy
Fonte: (EDUCBA, 2024).

Benefícios de um Data Warehouse

Com todos os pontos anteriores definidos, existe de fato um data warehouse, dessa forma, a instituição que o possui passar a agregar os seguintes benefícios:

- **Melhor Qualidade de Dados:** centralização dos dados de várias fontes, limpando, eliminando duplicatas e padronizando-os para criar uma única fonte confiável.
- **Insights de Negócios Mais Rápidos:** Integração dos dados de diversas fontes, permitindo que os gestores utilizem todas as informações da empresa em cada decisão, gerando relatórios sobre temas, tendências e relações entre dados.
- **Tomada de Decisão Mais Inteligente:** Suporte á funções de BI em grande escala, como mineração de dados, inteligência artificial e aprendizado de máquina, oferecendo evidências sólidas para decisões mais inteligentes em várias áreas da organização.
- **Vantagem Competitiva:** A centralização de dados e análises em um data warehouse ajuda as organizações a identificar oportunidades mais rapidamente do que em sistemas de armazenamento de dados dispersos.

2.3.2 Data Lake

Surgimento e Definição de Data Lakes

Com o aparecimento de novas demandas que não eram atendidas pelos data warehouses, como armazenamento e consulta de dados não estruturados, extração em tempo real, armazenamento de grandes quantidades e entre outras, surgiram os data lakes, tornando possível essa análise do big data em diferentes formatos e trazendo alternativas a respeito das preocupações sobre custo e outras limitações dos data warehouses (NAMBIAR; MUNDRA, 2022).

Um data lake é um repositório centralizado que permite armazenar dados estruturados e não estruturados, é usado para armazenar grandes volumes em seu formato nativo, bruto. Neste

ambiente os dados não são processados pois eles ainda não tem um finalidade específica, podem vir a ser analisados em breve, no futuro ou até mesmo nunca. Reunir os dados em um único local ou a maior parte deles, torna a exploração muito mais simples e direta, porém, é um ambiente que exige governança e manutenção contínua, sem esse controle há o risco desses dados se tornarem inúteis, além de caros e pesados, dando surgimento aos chamados "data swamps"("pântanos de dados") (NAMBIAR; MUNDRA, 2022).

Todo esse contexto, proporciona aos data lakes condições para a gestão de dados pré-processados, agregados e armazenamento de dados brutos fornecidos pelas fontes originais, permitindo que esses dados sejam processados gradualmente ao longo do tempo (NAMBIAR; MUNDRA, 2022; GIEBLER et al., 2020). Esse método é conhecido como Extração-Carregamento-Transformação (ELT), em contraste com o processo de Extração-Transformação-Carregamento (ETL) utilizado pelos data warehouses.

A alta escalabilidade e flexibilidade dos data lakes, aliadas aos seus baixos custos operacionais, tornam frequente o uso de sistemas de arquivos distribuídos ou armazenamentos de objetos. No cenário deste trabalho é utilizado o Minio, o qual conta com buckets de armazenamento para armazenar tanto dados brutos quanto processados. Além disso, formatos de arquivos abertos e orientados a colunas, como Apache Parquet e Apache ORC, são amplamente adotados, pois organizam os dados em colunas, facilitando agregações mais eficientes.

Diferente de um data warehouse, os data lakes permitem o armazenamento de dados brutos, preservando todas as opções para análises futuras. No entanto, essa flexibilidade acarreta uma menor robustez, visto que esses dados não poderão ser validados em um esquema predefinido na ingestão. A natureza diversificada da arquitetura dos data lakes, assim como a aceitação variada de formatos de arquivos, também complica a obtenção de uma visão unificada dos dados e um acesso padronizado a eles (NAMBIAR; MUNDRA, 2022).

Vantagens e Desvantagens

- **Vantagens:**
  - Centralização da coleta e armazenamento de dados;
  - Batch (Lotes) ou streaming (Fluxo Continuo) para coleta de dados;
  - Flexibilidade;
  - Transformação de dados;
  - Análises com Machine Learning;
  - Cultura data-driven;

- **Desvantagens:**
  - Alto custo com processamento;
  - Maior governaça de dados;
  - Risco de "data swamps"("pântanos de dados");
  - Exigência de sistemas de armazenamento e processamento distribuído;
  - Maior cuidado com a segurança;

Arquitetura

A arquitetura de um data lake pode ser dividida em quatro grandes atividades, a seguir mais detalhes sobre cada uma delas:

- **Aquisição:** A fase de aquisição envolve a coleta e a ingestão de dados brutos de várias fontes para o data lake (NAMBIAR; MUNDRA, 2022). Isso inclui:
  - Fontes de Dados: Dados podem ser coletados de diversas fontes, como bancos de dados relacionais, sistemas de arquivos, APIs, logs de servidores, redes sociais, sensores IoT, entre outros.
  - Ingestão de Dados: O processo de ingestão pode ser realizado em tempo real (streaming) ou em lotes (batch).
  - Qualidade dos Dados: É importante garantir a qualidade dos dados durante a aquisição, aplicando validações e limpezas básicas antes do armazenamento.
- **Processamento**: Após a aquisição, os dados precisam ser processados para torná-los utilizáveis, a fim de evitar alguma inconsistência (NAMBIAR; MUNDRA, 2022). Isso pode envolver diversas atividades:
  - Transformação de Dados: Inclui a limpeza, normalização, enriquecimento, agregação e formatação dos dados.
  - ETL/ELT: Processos de Extração, Transformação e Carga (ETL) ou Extração, Carga e Transformação (ELT) são fundamentais. No ETL, os dados são transformados antes de serem carregados no data lake. No ELT, os dados são carregados em sua forma bruta e transformados posteriormente.
  - Escalabilidade: O processamento de dados deve ser escalável para lidar com grandes volumes de dados, garantindo eficiência e rapidez nas operações.

- **Análise:** Uma vez que os dados foram processados, eles podem ser analisados para gerar valor acerca daqueles dados, seja utilizando ferramenta de BI ou machine learning (NAMBIAR; MUNDRA, 2022), como por exemplo:
  - Data Mining e Machine Learning: Técnicas de mineração de dados e aprendizado de máquina são aplicadas para descobrir padrões ocultos e previsões.
  - BI e Relatórios: Ferramentas de Business Intelligence (BI) como Tableau, Power BI e QlikSense podem ser utilizadas para criar relatórios e dashboards interativos.
  - Exploração de Dados: Cientistas de dados e analistas exploram os dados usando linguagens de programação como Python e R, e ferramentas como Jupyter Notebooks para realizar análises ad-hoc e experimentações.
- **Armazenamento:** A fase de armazenamento é importante para garantir que os dados sejam mantidos de forma eficiente e segura (NAMBIAR; MUNDRA, 2022) , alguns cuidados são:
  - Armazenamento Escalável: data lakes geralmente utilizam sistemas de armazenamento escaláveis, como Amazon S3, Azure data lake Storage ou Hadoop Distributed File System (HDFS).
  - Formato dos Dados: Os dados podem ser armazenados em diversos formatos, como JSON, CSV, Parquet, Avro, entre outros. Escolher o formato correto é muito importante a fim de otimizar a eficiência de armazenamento e processamento.
  - Gerenciamento de Metadados: É essencial manter um catálogo de metadados para organizar e facilitar a busca e acesso aos dados epecifico que se tem interesse.
  - Segurança e Governança: Implementar políticas de segurança e governança é fundamental para proteger os dados contra acessos não autorizados e garantir a conformidade com regulamentos. Isso inclui criptografia, controle de acesso, auditorias e monitoramento.

Método de Implantação

A implantação de um data lake pode ser realizada de várias maneiras, dependendo das necessidades, infraestrutura e estratégia de TI da organização. Dentre os métodos de implantação de um data lake, os principais são: cloud, on-premises, multi-cloud e híbrido.

- **Cloud:** Implantar um data lake na nuvem oferece escalabilidade rápida, flexibilidade e um modelo de pagamento conforme o uso, reduzindo a necessidade de grandes investimentos iniciais e manutenção. No entanto, pode levar à dependência de um fornecedor específico (lock-in) e apresentar maior latência no acesso aos dados.

- **On-Premises:** Um data lake on-premises proporciona controle total sobre infraestrutura e dados, garantindo segurança e conformidade, além de baixa latência. Contudo, requer investimentos iniciais altos em hardware, manutenção contínua e pode ser caro e complexo para escalar.

- **Multi-Cloud:** A abordagem multi-cloud melhora a resiliência e flexibilidade, aproveitando os pontos fortes de diferentes provedores e facilitando a negociação de preços. Em contrapartida, a integração e gestão de múltiplas nuvens é complexa e pode aumentar os custos devido à necessidade de soluções avançadas.

- **Híbrido:** Um data lake híbrido combina on-premises e nuvem, oferecendo flexibilidade, custo-eficiência e melhor continuidade de negócios ao distribuir dados conforme a criticidade. No entanto, a integração e gestão de dados entre os dois ambientes pode ser desafiadora e requer políticas robustas de segurança e sincronização.

2.3.3 Arquitetura Lakehouse

Um data lakehouse é uma plataforma de dados moderna que combina as características de um data lake e um data warehouse. Essencialmente, um data lakehouse integra o armazenamento flexível de dados não estruturados de um data lake com os recursos de gerenciamento e ferramentas avançadas dos data warehouses, formando um sistema unificado. Esta fusão oferece uma solução que aproveita o melhor dos dois mundos. Utilizaremos neste trabalho a definição do trabalho "The lakehouse: State of the Art on Concepts and Technologies", que busca superar os problemas envolvidos entre outras definições existentes, os quais são mais descritos no parágrafo e no capítulo a seguir.

Como apresentado anteriormente na Tabela 2.1, a maioria das características típicas de data warehouses e data lakes são bastante opostas, especialmente no que diz respeito ao acesso aos dados, independência e tipo de armazenamento. Portanto, a combinação direta desses dois conceitos, que tenha como objetivo manter todas as vantagens de ambas as abordagens, não é viável. Em vez disso, data warehouses e data lakes geralmente precisam abrir mão ou flexibilizar algumas de suas características para incorporar funcionalidades da outra plataforma. Por exemplo, um data warehouse que pretende fornecer acesso direto aos seus dados para suportar mineração de dados e aprendizado de máquina deve sacrificar parte de sua independência de dados. Da mesma forma, um data lake só pode integrar recursos de gerenciamento dos data warehouses, como características ACID, se limitar a liberdade dos usuários no armazenamento de dados, introduzindo novas regras para operações de leitura e escrita (SCHNEIDER et al., 2024).

É importante destacar que, no contexto deste trabalho, os dados armazenados na última camada do data lakehouse, ou seja, nos buckets, já são acessíveis e podem ser consumidos diretamente por ferramentas que suportem a interação com esse formato de armazenamento, como

engines específicas para consulta e análise. No entanto, o envio desses dados para o banco de dados relacional reflete a adição de uma camada complementar de disponibilização, visando atender a um público que, muitas vezes, não possui familiaridade com o consumo de dados diretamente nos buckets. Esse banco de dados serve como um meio mais acessível e estruturado para a consulta das informações por usuários finais, sem comprometer a flexibilidade e a riqueza do armazenamento inicial no data lake, sendo um espelho do conteudo do bucket final de armazenamento.

A seguir, na metodologia, serão detalhadas as camadas desse lakehouse no contexto das ferramentas utilizadas na criação deste ambiente.

3

Metodologia

A metodologia adotada neste estudo será conduzida visando explorar a viabilidade e eficácia do ambiente de gestão de dados de baixo custo proposto. Inicialmente, será realizado um levantamento detalhado dos requisitos e necessidades específicas do ambiente de gestão de dados no contexto em questão, levando em consideração as limitações financeiras impostas. Esse processo envolverá uma análise cuidadosa dos aspectos funcionais e operacionais relacionados, garantindo que todas as exigências sejam devidamente satisfeitas.

A seleção das ferramentas open source foi realizada por meio de uma pesquisa criteriosa, visando identificar as opções mais adequadas para cada etapa do processo de gestão de dados, incluindo ELT, armazenamento, visualização e monitoramento, etapas estas contidas nas 4 grandes fases da aplicação: extract, standardized, refined e consume. Essa seleção será baseada em critérios como funcionalidade, compatibilidade e escalabilidade, garantindo a escolha das ferramentas mais apropriadas para o ambiente proposto.

Após a seleção das ferramentas, foi realizada a implementação do ambiente de gestão de dados, seguindo uma sequência de processos ELT: Extract, Load e Transform. Isso incluirá a configuração e integração das ferramentas selecionadas, visando criar um ambiente funcional e coeso que atenda às necessidades identificadas durante o levantamento de requisitos.

Durante a implementação, será conduzida uma análise comparativa entre as funcionalidades, desempenho e custos das ferramentas open source selecionadas e soluções comerciais estabelecidas, como o Azure Synapse. Essa análise permitirá avaliar a eficiência operacional e a economia obtida ao utilizar ferramentas de código aberto em comparação com soluções comerciais, considerando aspectos como desempenho, escalabilidade e custo total de operação.

Após a implementação do ambiente de gestão de dados, será realizado um estudo de caso prático para demonstrar sua aplicabilidade e eficácia na prática. Esse estudo de caso permitirá validar os resultados obtidos durante a implementação e destacar as principais contribuições do trabalho. Por fim, os resultados serão cuidadosamente analisados e documentados, comparando-os com as expectativas iniciais e destacando as principais conclusões derivadas do estudo.

A seguir, é apresentada a Figura 6 com o fluxo do processo descrito anteriormente.

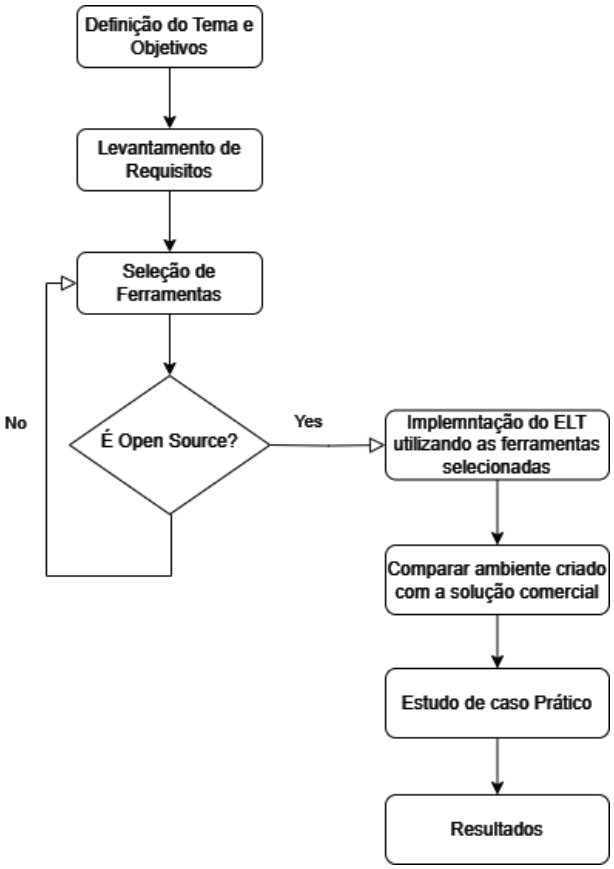


Figura 6: Fluxo de Atividades
Fonte: Autor.

3.1 Levantamento de Requisitos e Seleção de Ferramentas

O processo de levantamento de requisitos está intimamente ligado ao objetivo estabelecido para a criação do ambiente de gestão de dados e às barreiras existentes que influenciam sua realização. Como ressaltado anteriormente, o foco na criação de um data lakehouse enfatiza a importância de garantir a qualidade dos dados em todo o processo, abrangendo aspectos como completude, consistência, conformidade, integridade, precisão e atualidade das informações.

Durante a etapa de levantamento de requisitos, diversos pontos foram discutidos, levando à definição das ferramentas e sistemas escolhidos. Esses pontos incluem segurança, controle de rede, gerenciamento e escalabilidade de recursos, praticidade de operação e maturidade das ferramentas e, além disso, levaram-se em consideração aspectos como uma comunidade ativa, disponibilidade de documentação e suporte técnico.

Ao considerar o sistema operacional, a escolha pelo Linux se destacou como a solução mais viável devido à sua robustez em termos de segurança, controle acerca do sistema operacional e compatibilidade com a execução de containers Docker. Estes, por sua vez, foram adotados

para a execução das ferramentas, permitindo o encapsulamento das aplicações e simplificando o gerenciamento tanto na rede interna quanto externa. Isso resulta em benefícios tangíveis, como maior segurança, velocidade de comunicação entre os serviços, rapidez na implantação e padronização do ambiente.

Em relação às ferramentas selecionadas, cada uma foi escolhida com base em critérios específicos que se alinham com os requisitos e as considerações levantadas durante o processo de requisitos. O Portainer, por exemplo, foi escolhido pela sua capacidade de simplificar a gestão de containers Docker, facilitando o gerenciamento por meio de uma interface gráfica intuitiva. Já o DBeaver foi selecionado devido à sua versatilidade como cliente de banco de dados, eliminando algumas barreiras técnicas para manutenção e suporte da ferramenta.

O Jenkins foi adotado para a automação de processos, agendando a execução dos serviços diariamente de forma eficiente e com uma interface de fácil utilização, assim como o Portainer. Ainda sobre o Jenkins, o fator determinante para sua escolha foi sua sutil curva de aprendizado. De fato, existem outras alternativas de ferramentas open source no mercado com essa finalidade, porém com uma curva de aprendizado um pouco mais desgastante. E, embora essas alternativas sejam mais completas, para a finalidade aqui definida, o Jenkins cumpre perfeitamente o seu papel. O Jupyter foi o ambiente escolhido para desenvolver os notebooks Python, responsáveis por toda a etapa de ELT dos dados, utilizando bibliotecas como PySpark para facilitar a interação entre as ferramentas. Dada sua consolidação no mercado e sua ampla difusão atualmente, sendo aprovado nos mais diversos cenários em que é utilizado, mostrou-se uma ferramenta apta para este ecossistema, com uma vasta quantidade de material disponível para consulta. O Metabase foi selecionado para criação de dashboards e relatórios, enquanto o MinIO foi escolhido para armazenamento dos dados nos buckets, responsáveis pela conservação dos dados em formatos como JSON e Parquet durante as diferentes etapas do ELT. O Grafana foi empregado para monitoramento e visualização de métricas, abrangendo informações de infraestrutura, como consumo de recursos, em nível global ou por aplicação. Por fim, o PostgreSQL foi escolhido como sistema gerenciador de banco de dados devido à sua eficácia, flexibilidade e complementaridade dentro do ambiente de gestão de dados proposto.

O detalhamento das ferramentas será realizado posteriormente na análise comparativa entre opções do mercado e o ambiente open-source proposto. Porém, abaixo na Tabela 3.1, é possível observar um resumo das ferramentas utilizadas e, além disso, também foram acrescentados o Python e o PySpark, pois são elementos essenciais no desenvolvimento do ambiente, sendo o Pyspark uma interface de programação em Python para o Apache Spark, uma plataforma de computação distribuída voltada para o processamento de grandes volumes de dados.

Tabela 3.1: Principais Ferramentas

Nome	Tipo da ferramenta	Link para a versão utilizada	Descrição do uso
Linux	Sistema Operacional	V22.04	Escolhido pela robustez em segurança, custo e compatibilidade com containers Docker.
Cloud DBeaver	Cliente de Banco de Dados	V24.1.3	Versátil para manutenção e suporte de bancos de dados.
Jupyter	Ambiente de Desenvolvimento	V3.11.6	Utilizado para desenvolver notebooks Python para ELT.
Metabase	Ferramenta de BI	V0.48.3	Selecionado para criação de dashboards e relatórios.
Jenkins	Automação de Processos	V2.426.2	Automação de serviços com interface intuitiva.
Grafana	Monitoramento e Visualização	V9.3.6	Monitoramento e visualização de métricas de infraestrutura.
Prometheus	Monitoramento e Visualização	Vv2.42.0	Monitoramento e visualização de métricas de infraestrutura.
Portainer	Gerenciamento de Containers	V2.16.2	Simplifica a gestão de containers Docker com interface gráfica.
Minio	Armazenamento de Dados	V2023-10-16T04-13-43Z	Armazenamento de dados nos formatos JSON e Parquet.
Docker	Virtualização de Containers	V25.0.3	Ferramenta para execução e gerenciamento de containers.
Python	Linguagem de Programação	V3.11.6	Usada para scripts e ELT com bibliotecas como PySpark.
PySpark	Ferramenta de Processamento de Dados	V3.5.0	Facilita a interação entre ferramentas no processo ELT e uso com big data.
PostgreSQL	Sistema Gerenciador de Banco de Dados	V12	Eficaz, flexível, barato e complementar no ambiente de dados.

3.2 Implementação do Ambiente de Gestão de Dados

A implementação dos notebooks utilizados em todo o processo de ELT dos dados divide-se em quatro grandes etapas: Extract, Standardize, Refine e Consume. Cada uma dessas etapas possui responsabilidades diferentes e utiliza mecanismos distintos para alcançar seus objetivos, os quais serão descritos a seguir com detalhes sobre seus métodos de abordagem. Abaixo, na Figura 7, é possível analisar a dependência entre as etapas e seus resultados.

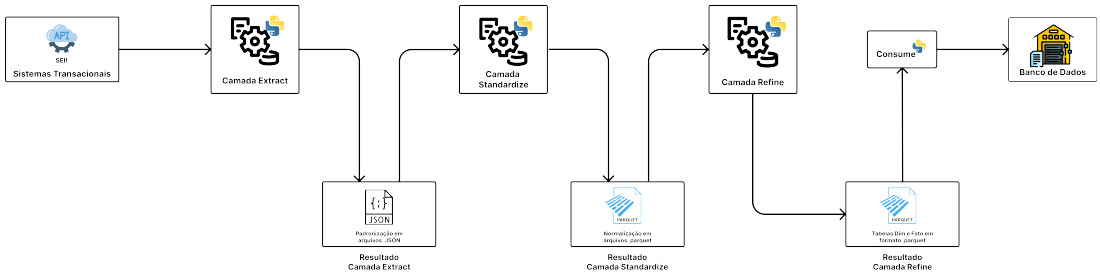


Figura 7: Dependência entre as camadas e seus resultados

Fonte: Autor.

Antes de explicar cada etapa em específico, é importante mencionar que durante todo o processo, em diferentes etapas, são utilizadas funções disponíveis na biblioteca "utils", desenvolvida com o intuito de conter procedimentos recorrentes durante o desenvolvimento, os quais serão mencionados ao esclarecer cada etapa. Estes métodos, em sua grande maioria, foram desenvolvidos utilizando o PySpark, e funções que ajudaram na integração entre as diversas ferramentas e classes criadas no módulo Utils. Todo o conteúdo encontra-se a disposição no repositório [1] deste trabalho.

Na etapa de Extração, é realizado o primeiro contato com os dados, os quais podem ser acessíveis de diferentes fontes, como bancos de dados, APIs, arquivos CSV, Excel, entre outras. A implementação varia de acordo com a fonte definida, porém padrões devem ser mantidos, como a transformação desses dados ao serem consumidos.

Como resultado da etapa de Extração, é construído um arquivo JSON, o qual será armazenado no bucket raw do Minio. Além disso, esse arquivo deve ser armazenado nesse bucket seguindo padrões de caminhos definidos, como por exemplo: <área de negócio>/<método de extração>/<subárea>/<data da extração>/subárea_(data da extração).json. Este procedimento é realizado utilizando o módulo utils desenvolvido, mais especificamente a classe SendMinio desse módulo, a qual está no repositório [1] deste trabalho. Concluída com sucesso, para cada área e subárea do negócio, temos como resultado os dados brutos armazenados como JSON no bucket raw do Minio.

Dando sequência para a próxima etapa, é iniciada a etapa Standardize. Esta etapa tem como objetivo normalizar os arquivos JSON para o formato Parquet. Além disso, são realizadas verificações de integridade dos dados, a fim de detectar duplicatas, por exemplo. Também são adicionados metadados ao conjunto de informações, como data da carga, versão do ELT, e timestamp do início e fim do ELT. Todos esses procedimentos são realizados utilizando o módulo utils desenvolvido, mais especificamente a classe SparkStandardize desse módulo, a qual está no repositório [1] deste trabalho. Feito isso, os arquivos Parquet são armazenados. No início desse processo, são definidos os tipos de arquivos que serão analisados, no caso JSON, além de ser definida a área de negócio a ser analisada, escrita da mesma forma como foi definida no caminho de armazenamento na etapa de Extração. Desta forma, ao finalizar todo o procedimento, os arquivos Parquet resultantes do processo de Padronização são armazenados em caminhos semelhantes aos da etapa de Extração, porém no bucket standardized do Minio.

Na etapa seguinte, Refine, são criadas as tabelas dimensões e fatos, sendo geralmente um pouco mais complexas que as anteriores. No início deste passo, são definidos alguns parâmetros, como o bucket fonte, que neste caso vem da etapa anterior(Standardize), a área de negócio e o método de extração. Inicialmente, para os casos das tabelas dimensões, utilizando a classe SparkRefine do módulo utils, é utilizada a função “create_spark_temp_view_with_transform” que, utilizando o PySpark, cria views temporárias com base no conteúdo Parquet do caminho

[1] Disponível em: <https://github.com/aldemirfilho/TCC-Ambiente-Gestao-de-Dados>

passado como parâmetro. Além disso, também é possível executar transformações que a própria função realiza, como por exemplo renomear colunas e realizar mudanças dos tipos de dados destas. Como mais uma opção, também é possível criar chaves sintéticas que sirvam como um novo ID para a tabela. Criada essa view, é possível acessá-la utilizando o formato de query SQL, utilizando a função "generate_parquet_dataframe_from_sql", a qual permite, utilizando um "select", transformar esses dados em um dataframe e, posteriormente, enviá-lo para o bucket refined do Minio, utilizando a função "send_dataframe_to_minio". Todas estas funções, definidas na classe SparkRefine do módulo utils, estão no repositório [1] deste trabalho.

Na criação das tabelas fatos, ainda na etapa Refine, são desenvolvidos notebooks que também utilizam do módulo utils e da classe SparkRefine. Nesse momento, para cada tabela, são criadas views temporárias de todas as tabelas dimensões dependentes para criação da fato em questão. Aqui é utilizada a mesma função usada no processo de criação das dimensões anteriores, "create_spark_temp_view", e dessa forma são nomeadas para serem usadas na construção da consulta que dará origem as tabelas fato. Fica claro que, para a construção da fato, ela pode possuir tabelas de diferentes origens e áreas de negócios, definindo assim antes da criação de cada view, o bucket, área de negócio e método de extração de cada dimensão, o que também explicita que as dimensões devem ser construídas antes das fatos.

Com todas as dependências de criação da fato definidas, utilizando a função "generate_parquet_dataframe_from_sql", é possível elaborar a consulta SQL que dará origem a tabela fato, sendo processada pelo PySpark, a qual, após ser executada, irá gerar um dataframe que será enviado para o bucket refined e sua respectiva área de negócio, utilizando a função "send_dataframe_to_minio" da classe SparkRefine.

Por fim, criadas todas as tabelas dimensões e fatos e armazenadas no bucket refined, ficamos pendentes do último passo: enviar essas tabelas para o banco de dados. Nessa etapa, a Consume, existem notebooks para cada tabela definida, e basicamente é importada a classe "MinioToPostgresLoader", presente no módulo utils, a qual, definindo a área de negócio, método de extração, nome da view definida na etapa anterior e o método de escrita, realiza o envio desses dados para o banco utilizando o método "MinioToPostgresLoader".

3.3 Análise Comparativa Entre Ferramentas Open Source e Soluções Comerciais

No cenário atual, entre as diferentes opções de soluções disponíveis no mercado, capazes de cumprir o objetivo estabelecido neste trabalho, abordaremos o Azure Synapse Analytics, comparando cada subproduto desta solução com as diferentes ferramentas que utilizamos na construção do ambiente de gestão de dados, open-source, proposto.

Entre as funcionalidades essenciais, podemos destacar a necessidade de criação de scripts SQL, os quais são bastante úteis para verificação das cargas, análise de dados, geração de rela-

tórios, manutenção e gerenciamento do banco de dados, monitoramento de desempenho e entre outros objetivos. No ambiente comercial apresentado, estas atividades podem ser desempenhadas no Synapse Studio no Azure Synapse Analytics. O Synapse Studio fornece uma interface web de script SQL para você criar consultas SQL. Nesse ambiente, é possível criar diversas consultas, renomeando-as, exportando seus resultados, visualizando-os, analisar o create de tabelas e etc. No ambiente proposto, todas essas atividades também podem ser realizadas utilizando o Cloud Dbeaver, um aplicativo web leve projetado para gerenciamento abrangente de dados. Ele permite que você trabalhe com várias fontes de dados, incluindo bancos de dados SQL, NoSQL e em nuvem, tudo por meio de uma única solução em nuvem segura acessível por meio de um navegador. Por ser um ferramenta acoplada com todas as outras e fazer parte da mesma rede interna do docker, toda carga feita ou consulta aos bancos fontes de dados podem ser acessadas por meio desta ferramenta e, assim, desenvolver consultas para monitoramento, geração de relatórios, monitoramento do banco de dados, dentre as outras funcionalidades já citadas.

Agora, tendo como objetivo criar, desenvolver e manter notebooks python, o Azure Synapse Analytics oferece um ambiente de desenvolvimento, uma interface web, para você criar arquivos que contêm código ativo, visualizações e texto narrativo. Tem como objetivo ser uma área de acesso rápido, a fim de validar ideias, realizar experimentos e obter insights de seus dados. Os notebooks também são amplamente utilizados na preparação, visualização de dados, aprendizado de máquina e outros cenários de Big Data. Com tudo isso em mente, no ambiente em questão proposto, é possível realizar todas essas atividades utilizando Jupyter, uma interface de notebook gratuita de última geração, é o mais recente ambiente de desenvolvimento iterativo baseado na web para notebooks, código e dados. Sua interface flexível permite aos usuários configurar e organizar fluxos de trabalho em ciência de dados, computação científica, jornalismo computacional e aprendizado de máquina. Além de tudo isso, assim como como no ambiente do Azure Synapse, é possível modificar a linguagem do ambiente, analisar saídas de cada célula executada, integrar com ferramentas de big data, como Apache Spark, e enviar e consumir os dados de diferentes fontes, dando suporte para realização de todo o ELT.

Em relação ao desenvolvimento em Data Flow, que são transformações de dados projetadas visualmente, permitem que os engenheiros de dados desenvolvam lógica de transformação de dados sem escrever código. Os fluxos de dados resultantes são executados como atividades dentro de pipelines do Azure Synapse Analytics que usam clusters Apache Spark ampliados. As atividades de fluxo de dados podem ser operacionalizadas utilizando as capacidades existentes de agendamento, controle, fluxo e monitorização do Azure Synapse Analytics. Os fluxos de dados fornecem uma experiência totalmente visual, sem necessidade de codificação. O Azure Synapse Analytics trata de toda a tradução de código, otimização de caminho e execução dos seus trabalhos de fluxo de dados.

Já no ambiente proposto deste trabalho, não foram aplicadas ferramentas visuais de desenvolvimento, ficando restrito aos métodos citados anteriormente no Jupyter. Essa escolha foi motivada pela necessidade de evitar o aumento da robustez do ambiente em quesitos de infra-

estrutura, o que elevaria os custos de manutenção, considerando que os notebooks Python já seriam suficientes para suprir as demandas. Além disso, embora ferramentas de data flow facilitem o desenvolvimento, elas podem dificultar a observabilidade da infraestrutura como código e a revisão do código das pipelines. De modo algum essa abordagem compromete a realização e conclusão do objetivo em questão: a criação de um Data Lakehouse. Desse modo, fica aqui sugestões e possíveis aplicações, também open-source, de ferramentas que atendem a esse quesito, e que podem ser usadas com o mesmo objetivo, são elas:

- **Pentaho Data Integration:** O Pentaho Data Integration (PDI) é uma ferramenta de integração de dados que facilita a extração, transformação e carga de dados de diversas fontes para diferentes destinos. Com uma interface gráfica intuitiva, oferece uma ampla gama de transformações e componentes para processamento de dados. É flexível, suportando diversas fontes de dados e execução distribuída para lidar com grandes volumes de dados. Pode ser agendado e monitorado para execução automática e se integra facilmente com outras ferramentas Pentaho, proporcionando soluções completas de Business Intelligence.

- **Apache Nifi:** O Apache NiFi é uma plataforma de código aberto para automação de fluxo de dados em tempo real. Ele permite a automação de processos de coleta, ingestão, processamento e distribuição de dados, oferecendo uma interface gráfica intuitiva para projetar e gerenciar fluxos de dados de forma visual. NiFi é altamente escalável, confiável e extensível, sendo usado em uma variedade de casos de uso, como IoT, Big Data, processamento de logs e integração de sistemas.

Sobre o tópico relacionado a criação de paineis, o ambiente do Azure Synapse é associado ao Power Bi, um serviço de análise de negócios e análise de dados desenvolvido pela própria Microsoft, bastante consolidado, e que permite a elaboração de painéis complexos, consumindo de diferentes fontes de dados, capazes de oferecer diversos insights e informações relevantes, quando bem apresentados, para os interessados. Com esse mesmo objetivo, no ambiente proposto foi utilizado o Metabase, um ambiente de análise rápida com UX amigável e ferramentas integradas para permitir que se explore os dados por conta própria. É possível conectar-se a mais de 20 fontes de dados diferentes, sendo possível exibir os resultados desses dados em painéis com mais de 15 opções de visuais bastante diversificados. Além disso, os painéis são de fácil compartilhamento e o ambiente conta com uma rica documentação para desenvolvimento e manutenção.

No agendamento dos processos, no Azure Synapses existe uma opção chamada Integrate, nela é possível selecionar um dos notebooks existentes e configurar as propriedades de atividades do notebook, configurando seu gatilho de execução e o intervalo entre cada uma. Além disso, também é possível monitorar o progresso de execução da pipeline, clicando nas execuções da mesma e visualizando a de interesse. Semelhante a isso, a utilização do Jenkins fornece

todos esses atributos, sendo possível agendar os notebooks python, desenvolvidos no Jupyter, e configurar a execução destes. É possível definir horários específicos para a execução, ou configurar dependências como gatilhos de disparo, como por exemplo após a execução, com sucesso ou não, de um outro notebook. O Jenkins conta com uma documentação extensa e detalhada, a qual contribui bastante para execução, aprendizado e manutenção no gerenciamento das pipelines.

Referente ao monitoramento de toda a infraestrutura, no ambiente do Azure Synapse, o Monitor recolhe e agrega métricas e registros de cada componente do sistema. O Azure Monitor fornece uma visão de disponibilidade, desempenho, resiliência e notifica o usuário em caso de problema. No ambiente proposto, como alternativa, temos a aplicação do Grafana, uma solução de visualização e monitoramento de dados de código aberto que orienta decisões informadas, melhora o desempenho do sistema e agiliza a solução de problemas. Nesta aplicação, foram construídos quatro gráficos, são eles: Alerts, Docker Container, Node Monitoring e PostgreSQL Metrics.

- **Alerts:** Neste painel, o intuito é que sejam informados alertas sobre questões bastante pontuais, como por exemplo queda de containers de algumas das aplicações, quedas ou reinício do servidor, picos de utilização da CPU e/ou da memória, tanto de um container em específico ou do ambiente como um todo, além disso, alertar em casos de pouco espaço no disco.

- **Docker Container:** Nessa aba, o intuito é monitorar por contêineres o uso da CPU, da memória e do tráfego de rede em gráficos de linhas, sendo possível identificar qual aplicação está envolvida em possíveis casos de pico de uso de memória e/ou CPU, além do tráfego na rede. Também é exibido o uso de CPU do ambiente como um todo e o fluxo de dados.

- **Node Monitoring:** Nesta aba, o foco é analisar o ambiente como um todo, como está o uso da CPU, uso de memória RAM, tráfego de rede, uso de espaço do disco, quantidade de cores disponíveis e entre outros aspectos, trazendo essa visão mais ampla do cenário utilizado.

- **PostgreSQL Metrics:** Este painel refere-se às métricas específicas do banco de dados. Nele são informados o uso de CPU do banco, memória RAM, memória em disco, memória em cache, além de outras informações referente às transações de dados.

Por fim, com intuito de gerenciar todo o ambiente e cada aplicação em seus respectivos contêineres, é utilizado o Portainer, uma plataforma de gerenciamento de contêineres que simplifica a administração de ambientes Docker. Ele fornece uma interface web intuitiva para facilitar a visualização, gerenciamento e monitoramento de contêineres, imagens, volumes e redes em um

ambiente. Um dos grandes motivadores para escolha do Portainer é sua facilidade de uso, especialmente para usuários que estão começando com Docker ou que preferem uma interface gráfica para gerenciamento. Ele elimina a necessidade de usar comandos de linha complexos, tornando a administração dos containers mais acessível. Além disso, ele pode ser implantado em vários ambientes, incluindo instalações locais, máquinas virtuais e ambientes de nuvem, suportando autenticação e autorização, permitindo que os administradores controlem quem pode acessar e realizar ações no Portainer.

3.4 Estudo de Caso Prático

Como caso de estudo, foi levado em consideração na escolha um sistema bem disseminado, de preferência público, para que assim pudesse ser fornecido ao serviço público e à sociedade as vantagens de um ambiente de gestão de dados acessível de baixo custo.

Nesse sentido, foi escolhido o "SEI!", o Sistema Eletrônico de Informações, desenvolvido pelo Tribunal Regional Federal da 4ª Região (TRF4). É uma ferramenta de gestão de documentos e processos eletrônicos, com o objetivo de promover a eficiência administrativa. O SEI integra o Processo Eletrônico Nacional (PEN), uma iniciativa conjunta de órgãos e entidades de diversas esferas da administração pública, com o intuito de construir uma infraestrutura pública de processos e documentos administrativos eletrônicos.

Através de uma API, foi possível obter os dados do SEI, os quais foram utilizados em todas as etapas mencionadas anteriormente, desde a extração até a inserção desses dados em suas devidas tabelas do lakehouse, considerando também as áreas de negócios, os datamarts. Além disso, este cenário, foi estabelecido com o intuito de enfatizar a relevância e a utilidade desse ambiente, a fim de que os dados fossem verdadeiramente úteis e relevantes para os interessados. Dessa forma, foram definidas como um dos resultados tabelas capazes de agregar valor aos usuários e gestores que utilizam o SEI, possibilitando, entre outras funcionalidades, o controle de atrasos de processos e a verificação daqueles que se encontram parados.

Dessa forma, as tabelas ficam disponibilizadas para análises ou até criação de dashboards, nesse cenário, na abordagem de processos parados por setor, por tipo, quantidade de dias sem movimentação ou mesmo evidenciando a quantidade de processos em cada um desses contextos.

Mais detalhes serão discutidos na seção de resultados.

4

Resultados

Neste capítulo, serão apresentados os resultados obtidos, considerando a eficácia do ambiente proposto em conjunto com as ferramentas selecionadas, assim como a avaliação da criação do lakehouse utilizando tal infraestrutura.

O ambiente proposto é concebido como um ecossistema integrado, onde diversas ferramentas foram cuidadosamente selecionadas para garantir a eficácia no processo de ELT dos dados. Isso é fundamental para assegurar um cenário robusto e organizado, capaz de armazenar e processar grandes volumes de dados de forma estruturada e acessível.

Com o objetivo de demonstrar esse grande potencial do ambiente, a criação do lakehouse foi uma estratégia utilizada para demonstrar a versatilidade do mesmo, que pode ser empregado em diversas finalidades, incluindo, nesse caso, a criação bem-sucedida de um ambiente com utilidade real. O uso de uma aplicação amplamente utilizada como fonte dos dados, foi fundamental nesse processo, a fim de demonstrar um ambiente capaz de oferecer tal solução de forma robusta e barata.

A seguir, também serão abordadas informações a respeito do desempenho do sistema, além de testes e validações do funcionamento ao decorrer de diferentes etapas da execução do ambiente.

4.1 Definição da Arquitetura e Ferramentas

A partir dos objetivos propostos, é possível elencar como um dos resultados a definição da arquitetura e das ferramentas escolhidas para atender tal necessidade.

Como uma visão macro, podemos analisar a Figura 8, nela é evidenciado a arquitetura escolhida e como ficou distribuída cada aplicação e as tecnologias aplicadas nas diferentes etapas do processo. O lakehouse foi escolhido por incorporar a flexibilidade, a economia e a escalabilidade de um data lake com o gerenciamento de dados e os recursos de transações ACID de um data warehouse.

Além disso, é importante relembrar que, foi possível construir todo esse ambiente utilizando apenas ferramentas open source, o que também foi um desafio concluído, visto que era necessário encontrar aplicações que atendessem a vários pré-requisitos para serem escolhidas, como comunidade ativa, disponibilidade de documentação, suporte técnico, tempo de atuação no mercado, segurança e citações usando tais ferramentas como referências.

O ambiente como um todo, é possível ser analisado na Figura 8 abaixo:

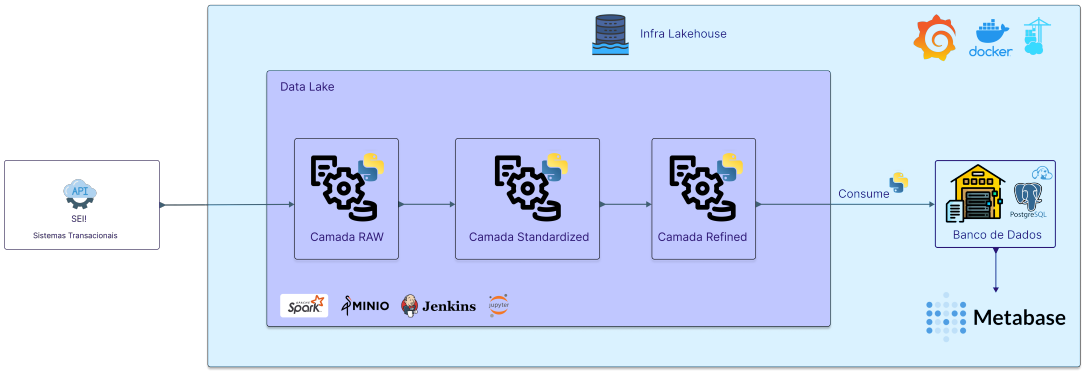


Figura 8: Ambiente de Gestão de Dados - Lakehouse
Fonte: Autor.

4.2 ELT dos Dados e Criação do Lakehouse

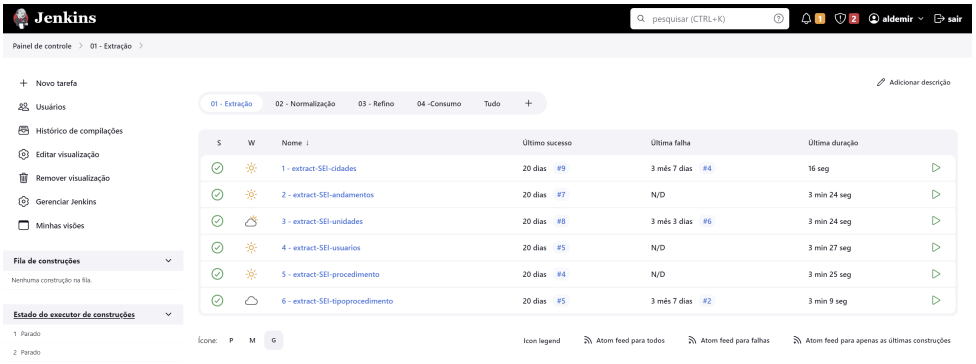
Como descrito no capitulo anterior e com base na Figura 8, é possível observar a disposição das três grandes camadas, Raw, Standardized e Refined. Abaixo é sintetizado o resultado obtido em cada uma dessas etapas:

- **Raw:** O resultado principal é a sua consolidação em formato JSON, armazenado no bucket Raw do Minio. A uniformização desses dados em um único tipo de arquivo facilita as etapas subsequentes de processamento, pois permite uma generalização máxima para lidar com essas informações.

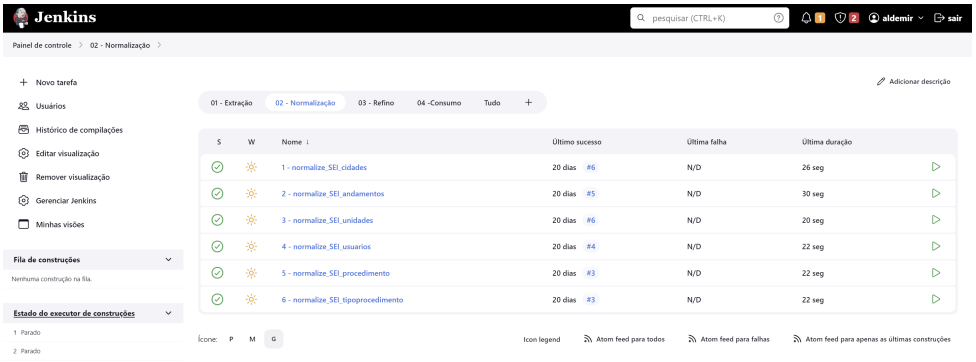
- **Standardized:** Na camada Standardized, o principal resultado fica na garantia da integridade dos dados, pois é onde são feitas verificações de duplicatas, por exemplo, e onde é realizado o enriquecimento do conteúdo com acréscimo de metadados, como data da carga, versão do ELT, timestamp do início e fim do ELT. Além disso, depois de todo esse processo, é gerado um parquet como resultado, o que garante um processamento e armazenamento mais eficiente para as próximas etapas.

- **Refined:** Nesta camada, é onde são gerados os primeiros arquivos parquet que de fato constituirão o ambiente, referentes às tabelas fatos e dimensões.

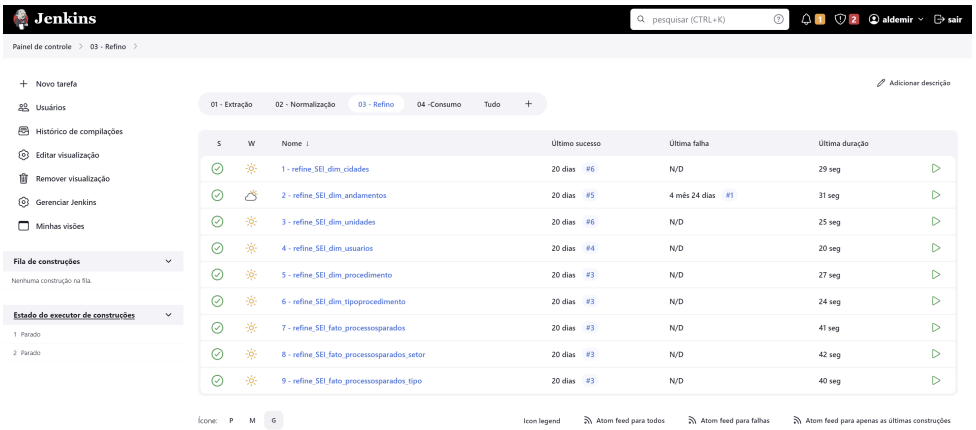
A execução das pipelines, como foi descrito no capítulo anterior, é feita com o Jenkins, na Figura 9 abaixo é possível verificar o agendamento e a dependência entre a as tarefas, onde todo o disparo é feito de maneira automática pré-configurada, seguindo a sequencia a seguir:



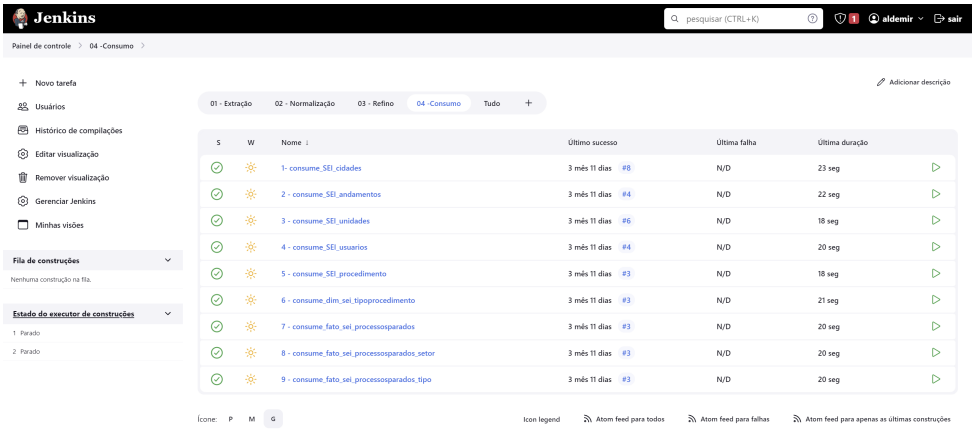
(a) Pipelines de Extração dos Dados



(b) Pipelines de Normalização dos Dados



(c) Pipelines de Refino dos Dados



(d) Pipelines de Armazenamento dos Dados

Figura 9: Disposição no Jenkins das Pipelines.
Fonte: Autor.

Como é possível analisar na Figura 10, essas foram as tabelas produtos de todo o processo, lembrando que trata-se de um recorte dos dados com o intuito de evidenciar o funcionamento do ambiente. Com essas tabelas é possível analisar a lista dos processos existentes, cidades envolvidas, tipos de processos, usuários, andamentos e, como produto resultado das manipulações do ambiente, a criação das tabelas fato de processos parados, evidenciando a quantidade em dias que os processos estão sem movimentação.

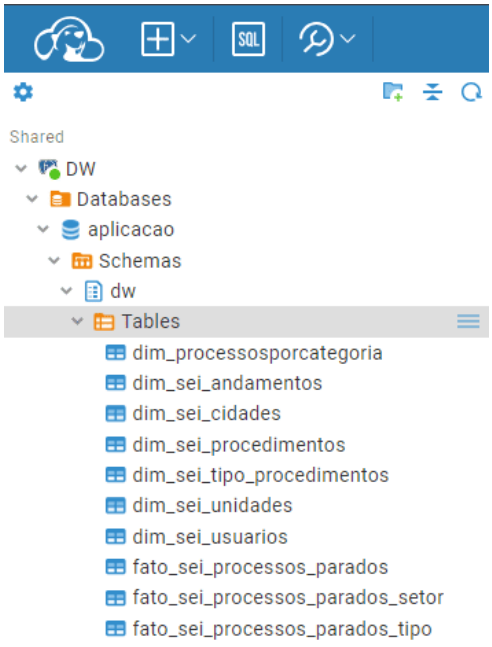


Figura 10: Tabelas Lakehouse
Fonte: Autor.

Nesse contexto, com base no estudo de caso, as tabelas resultantes agregam valor aos usuários do SEI, possibilitando análises detalhadas e a criação de dashboards. Entre as funcionalidades disponíveis estão o controle de atrasos de processos, a verificação de processos parados por setor, tipo, e quantidade de dias sem movimentação. Essas tabelas permitem uma visão abrangente e detalhada, facilitando a tomada de decisões e melhorando a eficiência administrativa, o que também classifica-se como um resultado.

4.3 Comparação Financeira de Ambiente

Um dos grandes desafios está relacionado em conseguir atender aos objetivos e manter o custo baixo, desta forma, um dos maiores resultados refere-se a economia que o ambiente proporciona em comparação com a utilização de alguns sistemas disponíveis no mercado.

A seguir, na Tabela 4.1 é possível analisar uma tabela comparando os custos entre o ambiente que utiliza ferramentas de código aberto e o ambiente fornecido pela Microsoft com o Azure Synapse.

É importante mencionar que os valores foram cotados no dia 20/08/2024. Além disso, as máquinas em questão possuem objetivos distintos, uma vez que no ambiente proposto foi cotada uma máquina Linux com 8 núcleos, 32GB de RAM e 2TB de armazenamento, sendo SSD premium, como intenção de executar todas as aplicações. Já no Azure Synapse, todo o ambiente é executado como um serviço da microsoft, sem necessidade de uma máquina dedicada. Dessa forma, a única máquina em questão cotada trata-se de uma máquina windows, com 4 núcleos, 8GB de RAM e 256GB de armazenamento, seu propósito é executar o Power Bi e suas dependências, visto que tal aplicação funciona apenas em ambientes microsoft. Além disso, aos demais serviços do Azure Synapse, foram selecionados valores de poder de execução semelhantes aos oferecidos pela máquina do ambiente de custo baixo. É importante também destacar que os valores estão em 730 horas, o que equivale a aproximadamente um mês.

Destacando também que os valores de algumas ferramentas do Azure Synapse podem variar, especificamente para algumas ferramentas, dessa forma será assumido o valor médio recomendado, equivalente ao poder de processamento do ambiente de menor custo.

É importante lembrar também que, comparar DWUs do Azure Synapse (medida utilizada no poder de processamento do Pool de SQL Dedicado e exploração de dados) com uma máquina física (ou VM) pode ser desafiador, pois os DWUs são uma métrica específica da nuvem da Microsoft que abrange várias dimensões de desempenho (CPU, memória, I/O), enquanto que, em uma máquina física ou VM, essas dimensões são mais diretamente controláveis.

Tabela 4.1: Comparação entre Ambiente Proposto e Azure Synapse.

	Ambiente Proposto		**Azure Synapse**	
Funcionalidades	**Ferramenta**	**Custo**	**Ferramenta**	**Custo**
Rede virtual	Rede	US$ 4,00	Rede	US$ 0,00
Ambiente de hospedagem	Máquina	US$ 647,69	Máquina	US$ 237,58
Desenvolvimento e análises utilizando scripts SQL	Cloud DBeaver	US$ 0,00	Scripts SQL do Synapse Studio	US$ 5.592,05
Desenvolvimento, análise e ELT através de notebooks	Jupyter	US$ 0,00	Análise e ELT através de notebooks	US$ 402,96
Desenvolvimento de DataFlow	-	-	Desenvolvimento de DataFlow	Não aplicado
Criação de Painéis	Metabase	US$ 0,00	Power BI	US$ 10,90
Gerenciamento e controle de automação	Jenkins	US$ 0,00	Integrate	US$ 5,65
Monitoramento da infraestrutura e atividades	Grafana	US$ 0,00	Monitor	0,00
Área de configurações gerais	Portainer	US$ 0,00	Manage	0,00
TOTAL		US$ 651,69		US$ 6.249,14

fonte: Autor

Como é possível perceber, a economia é bastante relevante, mostrando-se como um ambiente amplamente viável para entidades com menor poder aquisitivo e de manutenção da ferramenta. Lembrando que, o valor da implantação completa do Azure Synapse ainda levaria em consideração mais valores, que não entraram na tabela comparativa de ferramentas por não possuir equivalência direta, ficando assim destacado as ferramentas que possuem elementos comparativos.

4.4 Desempenho do Sistema

Analisando o desempenho do sistema para realizar todas as operações mencionadas anteriormente, de ponta a ponta, desde a extração até a criação das tabelas, é possível verificarmos a performance do ambiente em relação ao consumo dos recursos disponíveis, como por exemplo: memória RAM, sistema de armazenamento, CPU e tráfego de rede. Dessa forma, podemos analisar os resultados em relação ao desempenho do ambiente durante a execução das operações

para criação do lakehouse.

A análise a seguir baseia-se nos gráficos gerados pelo Grafana, a ferramenta que proporciona uma visualização dos recursos disponíveis no ambiente, os dados foram exportado dessa ferramenta e gerados a parte utilizando Python, a fim de garantir uma maior qualidade às imagens. Inicialmente, serão apresentados os gráficos principais referentes a cada contêiner de aplicação, em seguida, será oferecida uma visão abrangente da utilização dos recursos disponíveis na máquina como um todo. A primeira etapa, extração, foi iniciada às 19 horas e 50 minutos, sendo a ultima execução da ultima etapa, o envio dos dados para o banco, às 20 horas e 10 minutos.

A máquina analisada nos gráficos abaixo é uma VM que possui 8GB de memória RAM com 4 núcleos de processamento e sistema operacional Linux.

A carga utilizada para realização destas operações teve o intuito de demonstrar a operabilidade de todo o ambiente, sendo um recorte de uma operação completa, para execução de maiores volumes de dados é necessário um ambiente com mais recursos à disposição, como o mencionado na cotação de um máquina em tópicos anteriores deste trabalho.

Abaixo seguem gráficos evidenciando o desempenho dos contêineres e da máquina como um todo.

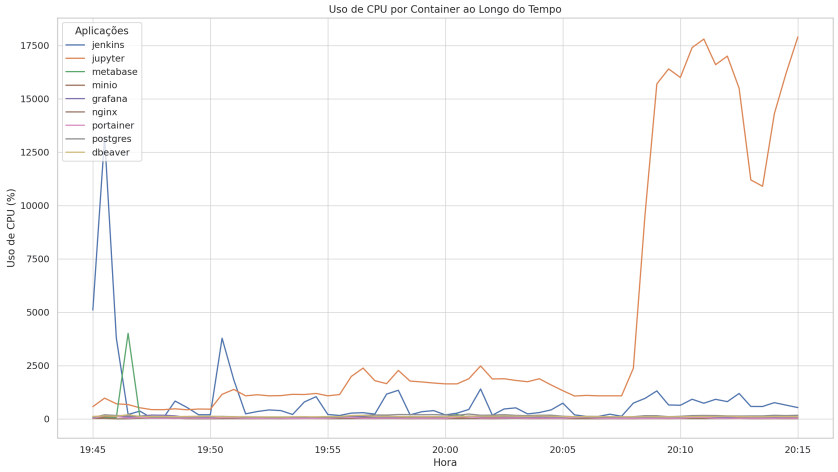


Figura 11: Uso de CPU por Container
Descrição: Distribuição do uso de CPU por container ao longo do tempo, com picos de uso por serviços específicos, destacando a alocação de recursos computacionais.
Fonte: Autor.

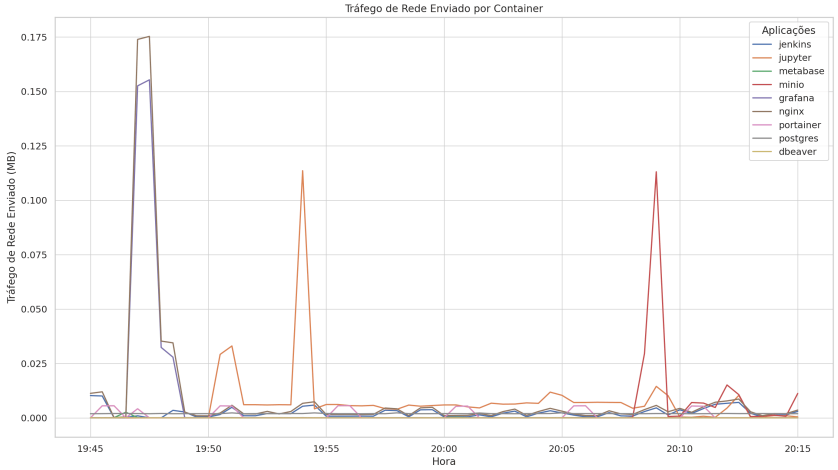


Figura 12: Tráfego de Rede Enviado por Container
Descrição: Gráfico que demonstra o tráfego de rede enviado por container ao longo do tempo, destacando picos de atividade para diferentes serviços como Minio, Jupyter e Grafana.
Fonte: Autor.

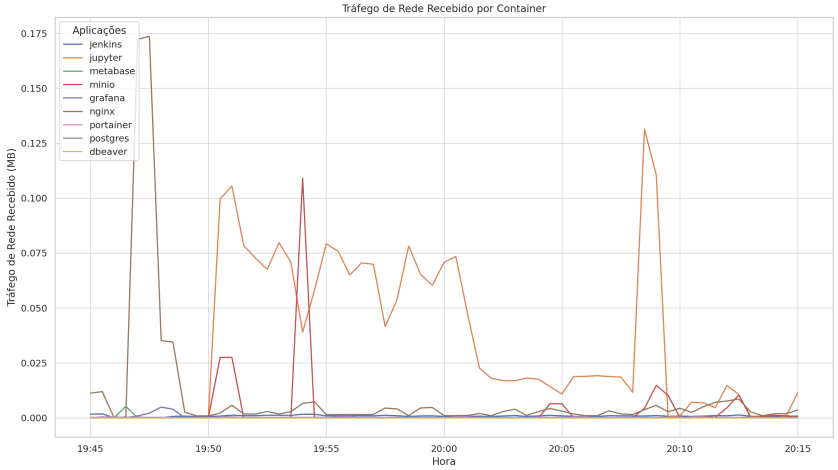


Figura 13: Tráfego de Rede Recebido por Container
Descrição: Gráfico detalhando o tráfego de rede recebido por container, ressaltando variações significativas e a interação de rede entre serviços como Jupyter e Minio.
Fonte: Autor.

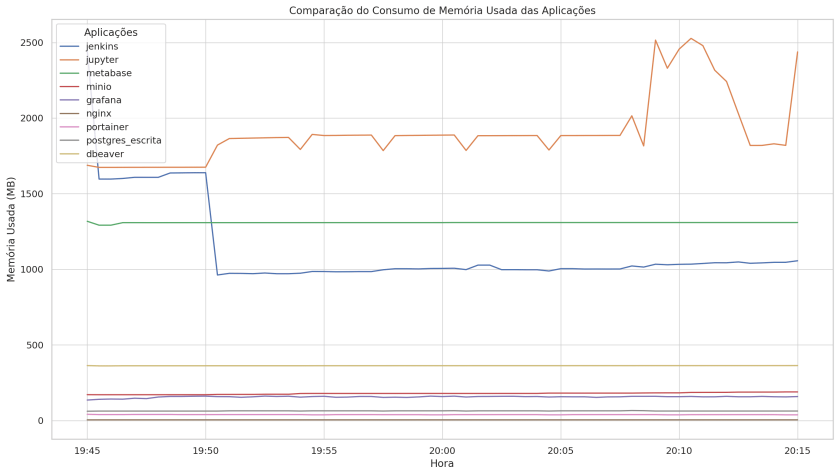


Figura 14: Uso de Memória por Container
Descrição: Visão do uso de memória por container, ilustrando como diferentes serviços contribuem para a utilização total da memória em um intervalo de tempo.
Fonte: Autor.

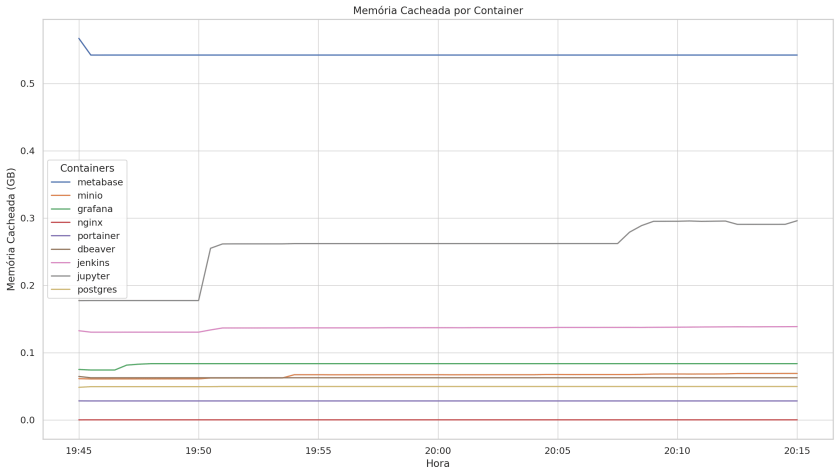


Figura 15: Memória em Cache por Container
Descrição: Gráfico mostrando a memória em cache de cada container.
Fonte: Autor.

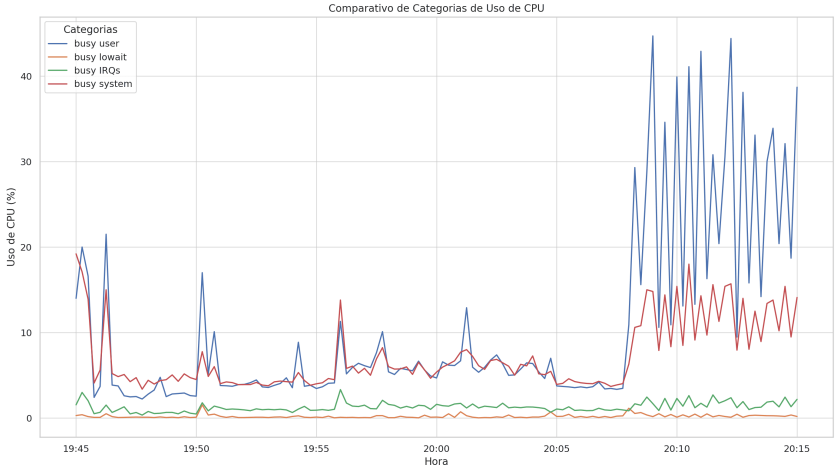


Figura 16: CPU Basic
Descrição: Gráfico de utilização básica de CPU ao longo do tempo, mostrando a porcentagem de CPU ocupada por diferentes tipos de processos como sistema, usuário, e interrupções, além do tempo ocioso.

Fonte: Autor.

5

Conclusão

A implementação do ambiente de gestão de dados proposto neste trabalho, que visa uma solução de baixo custo para a criação de um lakehouse, trouxe à tona importantes descobertas e contribuições. Os resultados demonstraram que é possível utilizar ferramentas de código aberto para elaborar um sistema robusto, eficiente e financeiramente viável para o processamento e análise de grandes volumes de dados. Além disso, o estudo de caso com o Sistema Eletrônico de Informações (SEI) mostrou a aplicabilidade prática e os benefícios diretos desse ambiente no contexto de gestão pública, aumentando a eficiência administrativa e proporcionando maior controle sobre os processos.

5.1 Síntese dos Resultados

Os resultados obtidos confirmam a viabilidade técnica e econômica do ambiente proposto. A criação bem-sucedida de um lakehouse a partir das camadas Raw, Standardized e Refined comprovou a eficácia do processo de ELT no ecossistema desenvolvido. O uso de ferramentas como Minio, Jenkins, e Metabase para o gerenciamento e visualização de dados mostrou-se eficiente, culminando em uma estrutura escalável e adaptável a diferentes tipos de organizações, especialmente aquelas com restrições orçamentárias.

A comparação financeira com o Azure Synapse evidenciou uma economia significativa, reforçando o potencial de adoção do ambiente por instituições públicas e privadas que necessitam de soluções de baixo custo. Os testes de desempenho confirmaram a capacidade de processamento do ambiente dentro dos limites de hardware disponibilizados, demonstrando que, com ajustes, o sistema pode ser ampliado para lidar com volumes de dados ainda maiores.

5.2 Desafios e Soluções

Durante o desenvolvimento do projeto, alguns desafios notáveis foram identificados, especialmente relacionados à escolha de ferramentas que fossem simultaneamente econômicas e capazes de garantir alta performance. A busca por alternativas que oferecessem suporte adequado e documentações acessíveis foi uma barreira significativa. No entanto, esse desafio foi superado com a integração de ferramentas de código aberto com comunidades ativas, o que facilitou a resolução de problemas técnicos e a customização do ambiente de acordo com as necessidades do projeto.

A integração de diferentes ferramentas de código aberto apresentou desafios, especialmente na coordenação de componentes que nem sempre foram projetados para funcionar em conjunto. Ajustar as dependências, configurar pipelines de dados adequados e garantir a comunicação eficaz entre sistemas como Minio, Jenkins e Metabase exigiu esforço adicional em termos de configuração e testes. A solução foi investir em uma abordagem modular, que permitiu a integração gradual e a automatização de processos, minimizando problemas de compatibilidade e aumentando a interoperabilidade entre os sistemas.

A comparação entre o ambiente proposto, baseado em ferramentas de código aberto, e o Azure Synapse, uma solução consolidada no mercado, também encontra-se entre um dos desafios. O Azure Synapse utiliza diversas unidades de medida próprias, como as Data Warehouse Units (DWUs), que agregam múltiplos fatores de desempenho, como processamento, memória e I/O. Essa abordagem torna difícil estabelecer uma correspondência direta entre o ambiente customizado e o ambiente oferecido pela plataforma da Microsoft, especialmente quando se busca recriar configurações de poder de processamento e desempenho semelhantes para uma comparação justa.

Enquanto no ambiente proposto as especificações de hardware, como núcleos de CPU, memória RAM e armazenamento, são definidas de forma explícita, no Azure Synapse essas métricas são abstraídas, o que complica a replicação de um ambiente idêntico.

Outro desafio foi a necessidade de garantir a segurança e integridade dos dados ao longo das etapas de processamento. As camadas de padronização e refinamento implementadas ajudaram a solucionar esses problemas, assegurando a qualidade dos dados ao final do processo, com destaque para a geração dos arquivos parquet, que otimizam o armazenamento e a performance.

5.3 Contribuições do Projeto

Este trabalho contribui de forma significativa para a democratização da gestão de dados, ao demonstrar a viabilidade de soluções acessíveis e de baixo custo para o processamento e análise de grandes volumes de dados. A criação do ambiente lakehouse utilizando apenas ferramentas de código aberto é um marco importante, pois amplia o acesso a tecnologias avançadas para instituições com menores orçamentos.

Além disso, o estudo de caso com o SEI mostrou que esse modelo pode ser replicado em outros contextos, tanto no setor público quanto privado, oferecendo maior controle e visibilidade sobre dados e processos. Essa abordagem pode influenciar diretamente a tomada de decisões informadas e assertivas, elevando o nível de eficiência administrativa e operacional de organizações que adotarem esse modelo.

5.4 Trabalhos Futuros

Para trabalhos futuros, algumas melhorias e expansões são sugeridas. Primeiramente, recomenda-se explorar a escalabilidade do ambiente para lidar com volumes de dados ainda maiores, pois trata-se de um ponto totalmente flexível quanto ao aumento de recursos, bem como a possibilidade de integrar novas ferramentas de visualização e análise, como o uso de machine learning para prever tendências e identificar padrões em dados históricos.

Além disso, sugere-se a integração de ferramentas de data flow, como Pentaho e Apache NiFi, ambas gratuitas e amplamente utilizadas. Essas ferramentas têm o potencial de aprimorar significativamente o processo de ETL, oferecendo maior flexibilidade e eficiência no fluxo e transformação dos dados dentro do ambiente proposto.

Outro ponto de investigação futura é a integração de soluções de governança de dados que garantam a conformidade com regulamentos de privacidade e segurança, como a LGPD, promovendo a adoção segura e ética dessa tecnologia em ambientes corporativos e governamentais.

Por fim, o uso de tecnologias de cloud computing pode ser uma abordagem interessante para ampliar ainda mais a acessibilidade do ambiente, possibilitando que ele seja utilizado por organizações de diferentes portes, sem a necessidade de grandes investimentos em infraestrutura física.

Referências Bibliográficas

AUNG, H. S.; LATT, Y. K. *DATA WAREHOUSE FOR STUDENT INFORMATION SYSTEM USING STAR AND SNOWFLAKE SCHEMA*. Tese (Doutorado) — MERAL Portal, 2006.

BROUS, P.; JANSSEN, M.; VILMINKO-HEIKKINEN, R. Coordinating decision-making in data management activities: A systematic review of data governance principles. In: SCHOLL, H. J. et al. (Ed.). *Electronic Government*. Cham: Springer International Publishing, 2016. p. 115–125. ISBN 978-3-319-44421-5.

CHANDRA, P.; GUPTA, M. K. Comprehensive survey on data warehousing research. *International Journal of Information Technology*, v. 10, n. 2, p. 217–224, Jun 2018. ISSN 2511-2112. Disponível em: <https://doi.org/10.1007/s41870-017-0067-y>.

DAGEVILLE, B. et al. The snowflake elastic data warehouse. In: *Proceedings of the 2016 International Conference on Management of Data*. New York, NY, USA: Association for Computing Machinery, 2016. (SIGMOD '16), p. 215–226. ISBN 9781450335317. Disponível em: <https://doi.org/10.1145/2882903.2903741>.

EDUCBA. *Fact Constellation Schema*. 2024. Accessed: 2024-11-25. Disponível em: <https://www.educba.com/fact-constellation-schema/>.

FERREIRA, A. Open source software. *Departamento de Engenharia Informática. Universidade de Coimbra*, 2005.

FUGGETTA, A. Open source software—an evaluation. *Journal of Systems and Software*, v. 66, n. 1, p. 77–90, 2003. ISSN 0164-1212. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0164121202000651>.

GIEBLER, C. et al. A zone reference model for enterprise-grade data lake management. In: *2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)*. [S.l.: s.n.], 2020. p. 57–66.

HAERDER, T.; REUTER, A. Principles of transaction-oriented database recovery. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 15, n. 4, p. 287–317, dec 1983. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/289.291>.

IBM. *O que é Data Warehouse?* 2024. Acessado em: 10 ago. 2024. Disponível em: <https://www.ibm.com/br-pt/topics/data-warehouse>.

IBM. *Star and Flake Schemas*. 2024. Accessed: 2024-11-25. Disponível em: <https://www.ibm.com/docs/en/ida/9.1.2?topic=schemas-starflake>.

KIMBALL, R.; ROSS, M. *The data warehouse toolkit: the complete guide to dimensional modeling*. [S.l.]: John Wiley & Sons, 2011.

LEVENE, M.; LOIZOU, G. Why is the snowflake schema a good data warehouse design? *Information Systems*, v. 28, n. 3, p. 225–240, 2003. ISSN 0306-4379. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0306437902000212>.

NAMBIAR, A.; MUNDRA, D. An overview of data warehouse and data lake in modern enterprise data management. *Big Data and Cognitive Computing*, v. 6, n. 4, 2022. ISSN 2504-2289. Disponível em: <https://www.mdpi.com/2504-2289/6/4/132>.

SCHNEIDER, J. et al. The lakehouse: State of the art on concepts and technologies. *SN Computer Science*, v. 5, n. 5, p. 449, Apr 2024. ISSN 2661-8907. Disponível em: <https://doi.org/10.1007/s42979-024-02737-0>.

TURCAN, G.; PEKER, S. A multidimensional data warehouse design to combat the health pandemics. *Journal of Data, Information and Management*, v. 4, n. 3, p. 371–386, 2022. ISSN 2524-6364. Disponível em: <https://doi.org/10.1007/s42488-022-00082-6>.