UNIVERSIDADE FEDERAL RURAL DO RIO DE JANEIRO
INSTITUTO MULTIDISCIPLINAR

RODRIGO VICENTE CALÁBRIA

Modelagem e Implementação de um *Data Warehouse* para Análise de Mensagens Multiplataforma

Prof. Fellipe Ribeiro Duarte, D.Sc.
Orientador

Profa. Natália Chaves Lessa, D.Sc.
Co-orientadora

Nova Iguaçu, Fevereiro de 2026

Modelagem e Implementação de um *Data Warehouse* para Análise de Mensagens Multiplataforma

Rodrigo Vicente Calábria

Projeto Final de Curso submetido ao Departamento de Ciência da Computação do Instituto Multidisciplinar da Universidade Federal Rural do Rio de Janeiro como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Apresentado por:

_____

Rodrigo Vicente Calábria

Aprovado por:

_____

Prof. Fellipe Ribeiro Duarte, D.Sc.

_____

Profa. Natália Chaves Lessa, D.Sc.

_____

Profa. Juliana Mendes Nascente e Silva Zamith, D.Sc.

_____

Prof. Marcel William Rocha da Silva, Ph.D.

NOVA IGUAÇU, RJ - BRASIL

Fevereiro de 2026

Agradecimentos

Gostaria de agradecer primeiramente a Deus, por me garantir a força e perseverança que tanto precisei para que conseguisse concluir esta e todas as outras etapas da minha vida.

Agradeço à Silvana, minha mãe, e à minha família por todo apoio, incentivo e compreensão ao longo de todo o período de desenvolvimento deste trabalho, que foram fundamentais para que eu conseguisse seguir em frente e chegar até aqui.

Agradeço à coordenadora do curso, Juliana Mendes, pela atenção, empatia e disponibilidade, além de todo o suporte oferecido para que eu pudesse concluir este ciclo.

Agradeço ao meu orientador, professor Fellipe Duarte, pela orientação, contribuições e paciência ao longo do projeto.

Agradeço também à professora Natália Lessa, por aceitar a coorientação e por me guiar na etapa final do trabalho, sendo essencial para sua conclusão.

Por fim, agradeço a todos aqueles que torceram para que este trabalho fosse realizado, e me auxiliaram, das mais diversas formas possíveis, para que eu conseguisse finalizar esta conquista.

RESUMO

Modelagem e Implementação de um *Data Warehouse* para Análise de Mensagens Multiplataforma

Rodrigo Vicente Calábria

Fevereiro/2026

Orientador: Fellipe Ribeiro Duarte, D.Sc.

O crescente uso de redes sociais e plataformas de comunicação teve como resultado um enorme aumento no volume de mensagens geradas diariamente, provenientes de diversas fontes diferentes, e com estruturas distintas. Esse cenário impõe desafios relacionados à integração, organização e armazenamento desses dados de forma consistente e adequada para análises futuras. Neste contexto, é proposto o projeto e a implementação de uma arquitetura para coleta, normalização e armazenamento de mensagens vindas de múltiplas plataformas de comunicação em um *Data Warehouse*.

A solução desenvolvida é composta por um *chatbot* multiplataforma, responsável pela interação com os usuários e pela ingestão das mensagens, e por um *pipeline* ETL estruturado seguindo a abordagem de Kimball. As mensagens coletadas são inicialmente armazenadas em sua forma bruta em uma camada de *staging* e, em seguida, transformadas em um modelo canônico independente da plataforma de origem, por meio de um processo de normalização semântica inspirado no estado da arte da literatura. Os dados transformados são então carregados em um *Data Warehouse* desenvolvido utilizando modelagem dimensional, definindo *data marts* voltados à análise de mensagens e de desinformação.

O *chatbot* foi modelado como uma máquina de estados finitos, permitindo interações estruturadas e extensíveis com os usuários. O projeto tem como prioridade a modularidade, extensibilidade e separação de responsabilidades em sua arquitetura, facilitando a inclusão de novas plataformas de mensagens, técnicas de extração de conteúdo e métodos de análise, minimizando o impacto nos componentes existentes.

ABSTRACT

Modelagem e Implementação de um *Data Warehouse* para Análise de Mensagens Multiplataforma

Rodrigo Vicente Calábria

Fevereiro/2026

Advisor: Fellipe Ribeiro Duarte, D.Sc.

The increasing use of social networks and communication platforms has resulted in a massive increase in the volume of messages generated daily, originating from various different sources and with distinct structures. This scenario imposes challenges related to the integration, organization, and storage of these data in a consistent and appropriate manner for future analysis. In this context, this work proposes the design and implementation of an architecture for the collection, normalization, and storage of messages from multiple communication platforms in a Data Warehouse.

The developed solution is composed of a multiplatform chatbot, responsible for interacting with users and ingesting messages, and an ETL pipeline structured according to the Kimball approach. The collected messages are initially stored in their raw form in a staging layer and subsequently transformed into a canonical model independent of the source platform, through a semantic normalization process inspired by the state of the art in the literature. The transformed data are then loaded into a Data Warehouse developed using dimensional modeling, defining data marts aimed at the analysis of messages and disinformation.

The chatbot was modeled as a finite state machine, allowing structured and extensible interactions with users. The project prioritizes modularity, extensibility, and separation of concerns in its architecture, facilitating the inclusion of new messaging platforms, content extraction techniques and analysis methods while minimizing the impact on existing components.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

DW	*Data Warehouse*
OLTP	*Online Transaction Processing*
OLAP	*Online Analytical Processing*
ETL	*Extract Transform Load*
API	*Application Programming Interface*
DBMS	*Database Management System*
URL	*Uniform Resource Locator*
OCR	*Optical character recognition*
FSM	*Finite State Machine*
HTTP	*Hypertext Transfer Protocol*
ASGI	*Asynchronous Server Gateway Interface*
JSON	*JavaScript Object Notation*
SQL	*Structured Query Language*

Sumário

Capítulo 1

Introdução

Nos últimos anos, o crescimento exponencial das redes sociais e plataformas de comunicação como Telegram[1], WhatsApp[2] e Messenger[3] transformou profundamente a forma como as pessoas se comunicam e compartilham informações. Junto com essas mudanças, surgiram também novos desafios, como o aumento da disseminação de notícias falsas, também conhecidas como *Fake News*, que podem ter impactos significativos na sociedade, como desinformação em massa e manipulação de opiniões públicas.

Nesse contexto, a análise de mensagens provenientes de múltiplas plataformas apresenta desafios significativos, especialmente pela heterogeneidade dos dados, com suas diferenças semânticas de cada *Application Programming Interface* (API) e pela falta de uma estrutura unificada para armazenamento e consulta analítica. A integração dessas informações se mostra ainda mais relevante quando o objetivo é estudar questões relacionadas à desinformação e *Fake News*, que dependem do acesso a dados históricos, estruturados e comparáveis.

Diante desse cenário, surge a necessidade de uma arquitetura capaz de coletar, integrar e armazenar mensagens de diferentes plataformas de forma consistente e extensível, oferecendo suporte a análises futuras. Este trabalho propõe o desen-

[1]https://web.telegram.org/
[2]https://web.whatsapp.com/
[3]https://www.messenger.com/

volvimento de uma solução baseada em *Data Warehouse* (DW) e em um *pipeline Extract Transform Load* (ETL), junto com um *chatbot* como forma de interação com o usuário, com foco na organização e integração dos dados, e não na detecção automática de *Fake News* propriamente dita.

Diferente das abordagens aplicadas a redes sociais, a coleta de dados realizada pelo sistema proposto tem um desafio a mais: o acesso aos dados a serem analisados. Em plataformas de comunicação instantânea, como WhatsApp e Telegram, o conteúdo trocado entre usuários geralmente não é público e, portanto, não pode ser coletado diretamente pelo sistema de forma automática. Dessa forma, o projeto depende da participação do usuário na coleta dos dados, onde apenas informações explicitamente fornecidas ao sistema, através de um *chatbot*, podem ser processadas.

1.1 Objetivo

O objetivo geral deste trabalho é projetar e implementar uma arquitetura para a coleta, integração e armazenamento de mensagens provenientes de múltiplas plataformas de comunicação, utilizando conceitos de DW e modelagem dimensional.

Os principais objetivos específicos deste trabalho são:

a) Desenvolver um *pipeline* ETL capaz de ingerir dados heterogêneos provenientes de diferentes plataformas de comunicação;

b) Definir e aplicar um modelo canônico de mensagens para a normalização semântica dos dados;

c) Projetar um DW relacional, voltado a consultas analíticas;

d) Implementar um *chatbot* como interface de interação com o usuário, estruturado a partir de uma máquina de estados finita; e

e) Viabilizar análises futuras relacionadas à *Fake News* por meio da estrutura de dados proposta.

1.2 Organização do Trabalho

Este trabalho está organizado da seguinte forma:

a) **Capítulo 2**: apresenta a fundamentação teórica, abordando conceitos relacionados a DW, modelagem dimensional, processos ETL e integração de dados provenientes de redes sociais, além de listar trabalhos relacionados.

b) **Capítulo 3**: descreve a metodologia adotada, detalhando as decisões de projeto, o processo de mapeamento e integração dos dados e a definição do modelo dimensional.

c) **Capítulo 4**: apresenta a arquitetura da aplicação, descrevendo os principais componentes do sistema, incluindo o *pipeline* ETL, o *chatbot* e o DW.

d) **Capítulo 5**: apresenta exemplos de consultas analíticas possíveis a partir do modelo proposto.

e) **Capítulo 6**: descreve as considerações finais do trabalho, bem como as limitações identificadas e sugestões para trabalhos futuros.

Capítulo 2

Fundamentação Teórica

Este capítulo apresenta os conceitos, abordagens e decisões técnicas que fundamentam o sistema proposto. São discutidos os principais paradigmas relacionados ao armazenamento e organização de dados analíticos, com foco no conceito de *Data Warehouse* e em diferentes métodos de implementação amplamente utilizados na literatura, como as abordagens de Inmon, Kimball, *Data Lake* (*Lakehouse*) e Data Vault.

Além disso, são abordados conceitos importantes de *Data Warehousing*, como modelagem dimensional, *Data Marts*, tabelas fato e dimensão e tipos de dimensões.

2.1 *Data Warehouse* e sua Utilidade

Uma das separações fundamentais na área de sistemas de dados é aquela existente entre ambientes *Online Transaction Processing* (OLTP) e *Online Analytical Processing* (OLAP) (CODD, 1993). Enquanto sistemas OLTP são otimizados para operações rápidas de inserção, atualização e exclusão de registros, sistemas OLAP dão prioridade a consultas complexas, agregações e análises multidimensionais, geralmente adotando técnicas específicas de modelagem e armazenamento, como a modelagem dimensional e o uso de esquemas Estrela (*Star*) ou Floco de neve (*Snowflake*).

Os DW pertencem a este segundo grupo, sendo projetados para suportar consultas

analíticas de grande volume, comumente envolvendo grandes quantidades de dados históricos.

Um DW consiste em um repositório de dados projetado para apoiar atividades de análise e tomada de decisão, reunindo informações provenientes de múltiplas fontes de dados operacionais. Diferentemente dos bancos de dados transacionais, conhecidos como OLTP, focados no processamento de operações do dia a dia, o DW é estruturado levando em consideração o desempenho de consultas analíticas complexas, envolvendo grandes volumes de dados históricos (INMON, 2005).

De acordo com Inmon (2005), as principais características que um DW apresenta são a orientação a assuntos, a integração de dados heterogêneos, a não volatilidade dos dados armazenados e a variação temporal. Essas características tornam o DW uma excelente escolha para análises que envolvam comparação de períodos, identificação de padrões e consolidação de informações provenientes de diferentes sistemas.

A orientação a assunto refere-se ao fato de que os dados no DW são organizados em torno de temas centrais de interesse da organização, como clientes, produtos ou eventos, em vez de processos operacionais específicos. A integração de dados heterogêneos diz respeito à consolidação de dados provenientes de múltiplas fontes heterogêneas, garantindo a padronização de formatos, nomenclaturas e significados. A não volatilidade refere-se à não remoção ou alteração frequente de dados que, uma vez carregados no DW, tem como objetivo principal serem usados para leitura e análise. Por último, a característica de variabilidade no tempo diz respeito ao armazenamento de estados históricos, permitindo análises temporais e comparativas ao longo de diferentes períodos.

Há diferentes abordagens para a implementação de um DW. As principais abordagens utilizadas são apresentadas nas subseções a seguir.

2.1.1 Abordagem de Inmon

A abordagem proposta por Inmon (2005), considerada uma das primeiras formulações formais de DW e frequentemente chamada de "Paradigma Corporativo", adota

uma estratégia *top-down* para o desenvolvimento do ambiente analítico. Nesta abordagem, o DW é projetado como um repositório centralizado focado na escalabilidade corporativa, modelado de forma extensivamente normalizada, geralmente seguindo a terceira forma normal (3FN).

Neste modelo, os dados provenientes dos sistemas de origem são integrados e armazenados no DW corporativo e, somente após essa etapa, são construídos *data marts* específicos para atender a diferentes áreas de negócio. A abordagem de Inmon prioriza a consistência global dos dados, a governança e a visão integrada da organização.

Apesar de suas vantagens em ambientes corporativos, essa abordagem apresenta limitações quando aplicada a projetos menores. A complexidade da modelagem, o maior esforço inicial e o tempo necessário para atingir os resultados tornam sua adoção menos adequada para projetos acadêmicos, por demandar maiores restrições de tempo e recursos (KIMBALL; ROSS, 2013).

2.1.2 Abordagem de Kimball

A abordagem proposta por Kimball e Ross (2013) adota uma estratégia *bottom-up*, baseada na modelagem dimensional e na construção gradual do DW por meio de *data marts* que representam processos de negócio. Nessa abordagem, os dados são organizados em tabelas de fatos e dimensões, estruturadas em esquemas Estrela (*Star*) ou Floco de neve (*Snowflake*).

As tabelas de fatos armazenam medidas quantitativas relacionadas a eventos de interesse, além de referências a entradas de tabelas dimensão. Já as tabelas dimensão fornecem o contexto necessário para análise, como tempo, usuário, plataforma ou tipo de mensagem.

Este método faz uso de "dimensões conformadas", que são definidas como dimensões que possuem o mesmo significado, estrutura e conteúdo em diferentes *data marts* ou processos de negócio dentro de uma organização (KIMBALL; ROSS, 2013). Essas dimensões permitem que diferentes *data marts* compartilhem informações comuns,

garantindo consistência analítica. Em um contexto de banco de dados, elas seriam implementadas como suas próprias tabelas, as quais seriam utilizadas em mais de um escopo (*data mart*) do negócio, mantendo um tipo de uniformidade "global".

A abordagem de Kimball e Ross (2013) se destaca pela simplicidade de modelagem, pela facilidade de compreensão do esquema de dados e pela eficiência na execução de consultas analíticas. Essas características fizeram com que essa abordagem fosse amplamente adotada em sistemas OLAP e a torna especialmente adequada para projetos que buscam flexibilidade, expansão gradual e foco em análise.

2.1.3 Data Vault

O Data Vault é uma abordagem mais recente para modelagem de DW, que, de acordo com seus criadores Linstedt e Olschimke (2015), é voltada para ambientes que demandam alta flexibilidade, rastreabilidade e controle histórico. Essa abordagem organiza os dados em três principais tipos de entidades:

a) *Hubs*: representam conceitos de negócio estáveis, como usuários ou mensagens;

b) *Links*: representam os relacionamentos entre esses conceitos;

c) *Satellites*: armazenam atributos descritivos e históricos, permitindo rastrear alterações ao longo do tempo.

Entre as principais vantagens do Data Vault estão a facilidade de adaptação a mudanças nas fontes de dados e a capacidade de manter um histórico detalhado das informações. Por essas razões, essa abordagem é frequentemente utilizada em ambientes corporativos complexos e com alterações frequentes (LINSTEDT; OLSCHIMKE, 2015).

Apesar disso, Data Vaults não são projetados para suportarem diretamente consultas analíticas de ambientes OLAP e, por isso, são normalmente utilizados como uma camada intermediária antes da construção de modelos dimensionais.

2.2 Modelagem Dimensional em *Data Warehouses*

A modelagem dimensional é uma abordagem extensamente utilizada em DW, especialmente em projetos que adotam o método Kimball. Ela tem como principal finalidade organizar os dados para facilitar consultas analíticas, fazendo com que tais consultas sejam simples, eficientes e tenham clareza semântica para os usuários finais. Diferentemente da modelagem relacional tradicional, orientada à normalização e à integridade transacional, a modelagem dimensional é orientada à análise e à tomada de decisão (KIMBALL; ROSS, 2013).

Essa abordagem estrutura os dados a partir de dois elementos centrais: tabelas fato e tabelas dimensão, organizadas geralmente em esquemas do tipo Estrela (*Star schema*) ou Floco de neve (*Snowflake schema*). A seguir, são apresentados os principais conceitos relacionados à modelagem dimensional, de acordo com Kimball e Ross (2013), considerado o pai deste tipo de modelagem.

2.2.1 Tabelas Fato e Dimensões

As tabelas fato armazenam os dados quantitativos e mensuráveis de um processo de negócio, sendo o núcleo do modelo dimensional. Cada registro em uma tabela fato representa um evento ou ocorrência em um determinado nível de granularidade, definido de acordo com a finalidade de análise do DW. Além das medidas, as tabelas fato contêm também chaves que referenciam as tabelas dimensão associadas.

Na modelagem dimensional, existem diferentes padrões de tabelas fato, definidos de acordo com o tipo de processo de negócio analisado e com o que se deseja ser consultado. De acordo com Kimball e Ross (2013), há apenas três tipos fundamentais de tabelas fato: Transação, snapshot periódico e snapshot acumulativo. O grão para cada um dos três tipos é explicado a seguir.

Para tabelas fato de transação, o grão corresponde eventos pontuais, medidos em um único instante. Os fatos medidos são validos apenas para este evento específico neste determinado instante. Portanto, tabelas fato de transação podem ser imprevisivelmente densas ou esparsas, com a diferença entre entradas podendo

ser milisegundos, meses ou até mesmo nunca.

Tabelas fato de snapshot periódico correspondem a intervalos de tempo predefinidos. Os fatos medidos por esta resumem uma atividade durante ou no fianl de um período, e o seu grão garante que todas as entidades que reportam os dados irão aparece rem cada snapshot, mesmo se houver atividade. Tabelas fato de snapshot períodico são previsivelmente densas, e os aplicativos podem sempre contar com a presença de combinações de chaves.

Tabelas fato de snapshot acumulativo correspondem a processos previsíveis, com ínicio e fim bem definidos. Os registros deste tipo de tabela são revisados e sobrescritos à medida que o processo avança em suas etapas, fazendo com que geralmente estas sejam muito menores do que os outros dois tipos devido a essa estratégia de sobrescrita.

A escolha do padrão adequado depende do tipo de análise que se deseja fazer e do comportamento do processo de negócio representado, sendo comum o uso de diferentes padrões em um mesmo DW.

As tabelas dimensão, por sua vez, armazenam atributos descritivos que fornecem contexto ao que é representado pelas tabelas fato. Essas dimensões permitem que os dados sejam analisados sob diferentes perspectivas, como tempo, localização, entidade envolvida ou categoria. Em geral, as dimensões são mais estáveis ao longo do tempo e possuem volume de dados menor em comparação às tabelas fato.

2.2.2 *Data Marts*

Seguindo a definição de Kimball e Ross (2013), um *data mart* é um subconjunto de um DW, voltado para uma área de interesse específica do negócio ou para um conjunto específico de análises. Diferentemente de um DW corporativo abrangente, os *data marts* possuem um escopo menor e visam permitir análises de necessidades específicas.

Os *data marts* são considerados dependentes, isto é, fazem parte de uma arquitetura integrada, compartilhando dimensões conformadas (*conformed dimensions*)

entre si. Essa estratégia permite que diferentes *data marts* se mantenham consistentes ao realizar analises, mantendo a integração semântica e o compartilhamento de dados, e possibilitando análises cruzadas entre diferentes processos de negócio.

2.2.3 Dimensões Convencionais e Dimensões Degeneradas

As dimensões convencionais são aquelas que possuem uma tabela própria, contendo atributos que descrevem uma entidade específica. Exemplos comuns incluem dimensões de tempo, usuário, produto ou localização. Essas dimensões são referenciadas pelas tabelas fato por meio de chaves substitutas (*surrogate keys*).

Já as dimensões degeneradas consistem em atributos dimensionais que não possuem uma tabela dimensão própria, sendo armazenados diretamente na tabela fato. Esse tipo de dimensão é utilizado quando o atributo não possui outros elementos descritivos relevantes, mas ainda assim é importante para fins de identificação ou análise, como números de pedido, códigos de transação ou identificadores de mensagens. As dimensões degeneradas permitem preservar informações relevantes sem introduzir complexidade desnecessária no modelo.

2.2.4 Dimensão Tempo

A dimensão tempo é considerada uma das dimensões mais importantes em um DW, pois praticamente todas as análises envolvem algum fator temporal. Em vez de utilizar diretamente campos do tipo *timestamp*, a literatura de modelagem dimensional recomenda a criação de uma tabela de tempo dedicada, contendo atributos como dia, mês, ano, trimestre, dia da semana ou outros períodos relevantes para o negócio.

Essa abordagem facilita agregações de medidas temporais, comparações entre períodos e análises históricas, além de padronizar a representação do tempo em todo o DW. A utilização de uma dimensão de tempo também contribui para a melhoria do desempenho das consultas e para a clareza semântica do modelo.

2.3 *Data Lake* (*Lakehouse*) e sua Utilidade

O *Data Lake* é uma abordagem de armazenamento de dados que prioriza a flexibilidade e a escalabilidade, permitindo o armazenamento de dados estruturados, semiestruturados e não estruturados em seu formato bruto. O autor do termo, Dixon (2010), utilizou a seguinte analogia para descrever *Data Lakes*: Enquanto um *Data Mart* pode ser comparado a uma garrafa de água purificada, o *Data Lake* assemelha-se a um corpo de água em seu estado bruto, onde os dados fluem de várias fontes para um reservatório comum.

Segundo Gorelik (2019), diferentemente do DW, que adota o princípio de *schema-on-write*, no qual os dados são transformados e organizados antes do armazenamento, o *Data Lake* utiliza o conceito de *schema-on-read*. Nesse modelo, os dados são armazenados sem uma estrutura fixa pré-definida, e a definição do esquema ocorre apenas no momento da leitura, de acordo com as necessidades da análise. Isso faz com que *Data Lakes* tenham maior flexibilidade, especialmente quando são ingeridos dados de muitas fontes diferentes ou em constante mudança.

Já o conceito mais moderno de *Lakehouse*, como descrito por Armbrust et al. (2021), surge como uma evolução do *Data Lake*, buscando combinar a flexibilidade do armazenamento de dados brutos com algumas características analíticas tradicionalmente associadas ao DW, como maior desempenho em consultas analíticas.

2.4 Trabalhos Relacionados

A utilização de DW para análise de dados provenientes de redes sociais tem sido explorada por diversos estudos, especialmente em contextos que envolvem múltiplas fontes e grande volume de dados. Esses trabalhos destacam a importância da integração, organização e modelagem adequada das informações para suporte a análises analíticas.

Valêncio et al. (2020) propõem um DW voltado à análise de redes sociais em um ambiente de *Big Data*, destacando a integração de dados e o suporte à tomada de

decisão. O trabalho demonstra a viabilidade do uso de arquiteturas analíticas para explorar métricas relacionadas ao comportamento e ao engajamento dos usuários em redes sociais.

Moalla et al. (2016) abordam especificamente o problema da heterogeneidade dos dados provenientes de diferentes plataformas sociais, propondo um método de mapeamento baseado em relações semânticas idênticas, equivalentes e complementares. Essa abordagem fundamenta o método de integração adotado neste projeto, aplicado às plataformas Telegram, WhatsApp e Messenger, que será explicado no Capítulo 3.

Kraiem e Feki (2024) apresentam uma revisão comparativa de abordagens para a construção de DW aplicados a mídias sociais, destacando desafios como a heterogeneidade dos dados, a qualidade da informação e a necessidade de modelos flexíveis e extensíveis.

Embora esses trabalhos se concentrem principalmente em postagens públicas de redes sociais, o projeto aqui proposto se diferencia ao focar na coleta e organização de mensagens provenientes de aplicativos de comunicação baseados em chat, estendendo conceitos consolidados da literatura para esse contexto específico e oferecendo uma base estruturada para análises.

Capítulo 3

Metodologia

Este capítulo apresenta a metodologia adotada para o desenvolvimento do projeto, descrevendo as decisões técnicas e conceituais que orientaram a construção do sistema de ingestão, integração e armazenamento das mensagens provenientes de plataformas de comunicação. São detalhadas as escolhas relacionadas à arquitetura de dados, ao processo de ETL e à modelagem do DW.

3.1 Escolha do *Data Warehouse* com Método Kimball

Diante das abordagens descritas no Capítulo 2, foi selecionado como mais adequado o uso de DW como solução para os objetivos deste projeto, por sua capacidade de integrar dados de múltiplas fontes, manter histórico e oferecer suporte eficiente para consultas analíticas. Essas características são importantes para a análise de mensagens provenientes de diferentes plataformas e a obtenção de métricas relevantes ao contexto de mensagens instantâneas.

Entre os métodos de projeto avaliados, optou-se pela abordagem Kimball devido à sua simplicidade de implementação, clareza conceitual e forte alinhamento com análises OLAP. A modelagem dimensional facilita a organização dos dados, melhora o desempenho das consultas e possibilita expansões futuras do sistema, como a criação de novos *data marts* voltados a análises específicas, como a de *fake news*.

A solução proposta adota uma arquitetura em camadas (BUSCHMANN et al., 1996), composta por: fontes de dados, camada de ingestão, camada de *staging*, processo de transformação e carga (ou ETL) e DW final. Essa separação facilita a visão sobre o fluxo de dados e a manutenção do sistema.

Redes sociais e plataformas de comunicação como Telegram, WhatsApp e Messenger constituem as fontes de dados do sistema. As mensagens são coletadas por meio de APIs (ingestão), e são armazenadas inicialmente na camada de *staging*, sem quaisquer alterações aos seus dados brutos. Essa estratégia garante a preservação das informações originais, o que permite reprocessamentos futuros e auditoria de dados.

Foi escolhido o esquema estrela como o principal formato de modelagem, pois oferece menor complexidade e melhor desempenho em consultas quando comparado a esquemas mais normalizados, como *snowflake*.

Dessa forma, a escolha de um DW relacional com modelagem dimensional segundo o método Kimball atende bem ao escopo do trabalho, oferecendo uma base sólida, extensível e alinhada às boas práticas descritas na literatura.

3.2 Ingestão de Dados Provenientes de Múltiplas Plataformas

A ingestão de dados constitui a primeira etapa operacional do processo de construção do DW, sendo responsável pela extração das mensagens e de seus respectivos metadados a partir das plataformas de comunicação abordadas neste projeto. A coleta é realizada por meio das interfaces de programação de aplicações (APIs) disponibilizadas pelas próprias plataformas, utilizando mecanismos como *webhooks*[1] ou *pooling*[2] direto às APIs.

A coleta de mensagens a partir de múltiplas plataformas de comunicação e redes sociais traz desafios devido à heterogeneidade dos dados. Embora plataformas como

[1]Webhook é um mecanismo de comunicação em tempo real entre sistemas, enviando dados automaticamente entre aplicações quando ocorre um evento específico.

[2]Pooling é uma técnica em que um cliente faz solicitações periódicas a um servidor para buscar dados atualizados.

Telegram, WhatsApp e Messenger compartilhem conceitos comuns de mensagens trocadas entre usuários, cada uma apresenta estruturas de dados, nomes de atributos e semânticas próprias. Dessa forma, torna-se necessária a criação de uma etapa de mapeamento de dados, com o objetivo de unir informações equivalentes e complementar dados ausentes.

Esse problema de integração é amplamente discutido na literatura. Segundo Moalla et al. (2016), a etapa de mapeamento é essencial quando os dados são extraídos de diferentes mídias sociais, uma vez que essas fontes podem conter informações redundantes, complementares ou semanticamente equivalentes, porém representadas de maneiras distintas.

As mensagens coletadas podem assumir diferentes formatos, incluindo texto, mídias (imagens, vídeos e áudios) e mensagens compostas que combinam texto e anexos. Independentemente do tipo de mensagem, todas as informações recebidas são encaminhadas para a camada de ingestão do sistema, sem a realização de transformações semânticas ou estruturais nesta camada.

Durante este processo de coleta, procura-se manter a integridade dos dados recebidos pelas APIs, mantendo seus identificadores originais, estruturas e atributos próprios, para que sejam armazenados de forma "pura" na camada de *staging*. Essa estratégia possibilita a rastreabilidade e o reprocessamento de dados em etapas posteriores, caso seja necessário implementar novos requisitos técnicos. O processo de ingestão é orientado a eventos, o que significa que a coleta é realizada de forma contínua, permitindo preservar a ordem de chegada das mensagens e possibilitando análises históricas sobre o conteúdo e a interação entre usuários.

Dessa forma, a etapa de coleta estabelece a base de dados inicial do sistema, fornecendo informações brutas e heterogêneas que serão posteriormente organizadas, integradas e transformadas nas camadas seguintes.

3.3 Camada de *Staging* e Dados Brutos

Os dados extraídos pelas APIs não são imediatamente adequados para análise, devido às diferenças de formatos entre eles. Por esse motivo, foi criada uma camada de *staging* para atuar como uma zona intermediária entre a coleta dos dados e o carregamento final no DW (também denominado *load* no processo ETL).

O objetivo desta camada é armazenar os dados brutos provenientes das plataformas de comunicação de forma temporária, preservando as características originais dos dados coletados. Essa estratégia evita a perda de informações relevantes que poderiam ser descartadas caso uma padronização prematura fosse aplicada.

A utilização da camada de *staging* possibilita a separação clara entre dados operacionais vindos das APIs e dados analíticos. Isto reduz o acoplamento entre as fontes de dados e o modelo dimensional do DW, tornando o sistema mais resiliente a mudanças nas APIs das plataformas. Além disso, essa abordagem facilita a identificação e a correção de inconsistências, duplicatas e dados incompletos antes da etapa de transformação final.

Outro ponto importante da camada de *staging* é a preservação do histórico completo dos dados coletados. A armazenagem dos dados originais das mensagens torna possível reprocessar os dados sempre que houver alterações no modelo ou nas regras de negócio, sem que seja preciso novas coletas de dados das plataformas.

Dessa forma, a camada de *staging* contribui para a consistência e flexibilidade do processo de ETL, funcionando como um elemento essencial para a construção de um DW robusto e extensível.

3.4 Heterogeneidade dos Dados e Método de Mapeamento Adotado

Conforme apresentado por Moalla et al. (2016), a heterogeneidade dos dados provenientes de redes sociais pode ser classificada em dois tipos principais: hete-

rogeneidade estrutural e heterogeneidade semântica. A heterogeneidade estrutural ocorre quando os dados são organizados em esquemas distintos, com diferentes estruturas e formatos. No contexto deste trabalho, Telegram, WhatsApp e Messenger fornecem dados por meio de APIs distintas, cada uma com seu próprio conjunto de campos, tipos de dados e hierarquias. Por exemplo, informações relacionadas a usuários, mensagens e mídias podem ser representadas de formas diferentes em cada plataforma.

Já a heterogeneidade semântica refere-se às diferenças de significado entre atributos que, apesar de representarem conceitos semelhantes, possuem nomes distintos ou formas específicas de serem lidos. Além disso, certas informações podem estar disponíveis apenas em uma plataforma, enquanto não estão presentes em outras.

Inspirado no método proposto por Moalla et al. (2016), este trabalho adota uma abordagem de mapeamento manual de esquemas, aplicada durante a camada de extração do processo ETL. Nessa abordagem, os esquemas de dados extraídos de cada plataforma são analisados individualmente e comparados entre si, com o objetivo de identificar relações semânticas entre seus atributos.

De acordo com o método descrito por Moalla et al. (2016), três tipos de relações semânticas são considerados:

a) Idêntica (*Identical*): quando os atributos possuem o mesmo nome e o mesmo significado nas diferentes plataformas;

b) Equivalente (*Equivalent*): quando os atributos possuem nomes diferentes, mas representam o mesmo conceito;

c) Complementar (*Complementary*): quando o atributo está presente apenas em uma plataforma, fornecendo informação adicional que não existe nas demais.

Essas relações permitem resolver problemas de duplicação, inconsistência e incompletude dos dados, além de facilitar a integração das informações em um modelo unificado.

3.5 Processo de Extração, Transformação e Carga

O processo de extração, transformação e carga, conhecido como ETL, é responsável por converter os dados armazenados na camada de *staging* em um formato adequado para o DW. Nesta etapa, os dados passam por procedimentos de padronização e integração, tornando-os consistentes semântica e estruturalmente antes de serem carregados nas tabelas fato e dimensões.

A fase de transformação envolve a aplicação das regras de negócio definidas para o projeto, incluindo a normalização de formatos de data e hora, a padronização de textos, o tratamento de valores ausentes e a adequação de tipos de dados. A partir do mapeamento semântico previamente definido, atributos classificados como idênticos ou equivalentes são integrados em campos comuns, enquanto atributos complementares são preservados como informações adicionais sempre que sejam relevantes para análises futuras.

Durante o processo de ETL, é adotada a geração de chaves substitutas (*surrogate keys*) para identificar registros no DW. Essas chaves são geradas pelo próprio sistema de gerenciamento de banco de dados (*Database Management System* (DBMS)) e utilizadas como chaves primárias das tabelas fato e dimensões. A utilização de chaves substitutas, em vez dos identificadores fornecidos pelas APIs das plataformas, evita a dependência de sistemas externos, colisões entre plataformas distintas ou a reutilização de identificadores ao longo do tempo, facilitando a integração entre diferentes fontes de dados.

O tratamento de duplicatas é realizado por meio de um mecanismo de identificação de mensagens baseado em um *fingerprint* gerado a partir do conteúdo da mensagem e de metadados únicos, como conteúdo textual, identificador da plataforma, remetente e instante de envio. A partir desses atributos, é gerado um valor *hash* que permite identificar de forma única cada mensagem. Antes do carregamento no DW, esse valor é verificado para garantir que mensagens previamente armazenadas não sejam inseridas novamente.

Outro aspecto considerado no processo de transformação é o versionamento das

mensagens. Em casos em que uma mensagem é editada após o envio, cada versão é tratada como um novo registro no DW, preservando o histórico completo das alterações. Essa decisão permite análises temporais mais precisas e evita a perda de informações relacionadas à mudança de conteúdo das mensagens.

Após a conclusão das etapas de transformação, os dados são carregados no DW de acordo com o modelo dimensional definido. As tabelas fato recebem os registros correspondentes aos eventos analisados, enquanto as tabelas dimensão são atualizadas conforme necessário.

3.5.1 Aplicação do Mapeamento às Plataformas Telegram, WhatsApp e Messenger

No contexto deste projeto, o mapeamento é realizado entre os dados extraídos das plataformas Telegram, WhatsApp e Messenger, considerando entidades comuns como usuários, mensagens e metadados associados. Inicialmente, os atributos específicos de cada plataforma são mantidos em suas estruturas originais na camada de dados brutos. Em seguida, durante o estágio de transformação, é realizada a análise semântica dos atributos para identificar possíveis correspondências.

Assim como proposto por Moalla et al. (2016), uma tabela de mapeamento é utilizada para registrar as relações entre atributos equivalentes, idênticos ou complementares entre as plataformas. Este mapeamento é apresentado na Tabela 3.1. Os atributos específicos de cada plataforma permanecem disponíveis para análise futura, sendo categorizados como complementares. Já os atributos comuns são integrados em um modelo único consolidado que servirá de base para o DW.

A categoria "Complementar"é omitida na Tabela 3.1 quando há outro tipo de relação entre as plataformas presentes, ficando implícita a relação de complementaridade com a plataforma sem o atributo equivalente.

Os atributos exatos de cada plataforma não são fixados neste momento, permitindo flexibilidade para adaptações futuras e para a inclusão de novos campos conforme a evolução das APIs ou dos objetivos do sistema.

Tabela 3.1: Mapeamento semântico de atributos entre plataformas de mensagens

Telegram	**WhatsApp**	**Messenger**	**Relação Semântica**	**Descrição**
update_id	id (wamid)	mid	Todos equivalentes	Identificador global da mensagem
message_id	–	–	Complementar	Identificador da mensagem em um contexto específico
chat.id	–	–	Complementar	Identificador do contexto (chat, conversa ou grupo)
date	timestamp	timestamp	Equivalente (Telegram)/Idênticas	Momento em que a mensagem foi recebida
–	phone_number_id	recipient.id	Ambos equivalentes	Identificador do destinatário
from.id	from	sender.id	Todos equivalentes	Identificador do remetente
from.first_name	name	–	Ambos equivalentes	Nome do remetente
from.last_name	–	–	Complementar	Último nome do remetente
text / caption	body	text	Todos equivalentes	Conteúdo textual da mensagem
–	type	attachment.type	Ambos equivalentes	Tipo de mensagem (texto, imagem, vídeo, etc.)
url	url	attachment.url	Todos idênticas	URL associada à mídia
mime_type	mime_type	–	Ambos idênticas	Tipo MIME da mídia
file_unique_id	id	–	Ambos equivalentes	Identificador único da mídia
file_size	–	–	Complementar	Tamanho do arquivo de mídia
–	forwarded	–	Complementar	Indica se a mensagem foi encaminhada
forward_from_message_id	–	–	Complementar	Identificador da mensagem original encaminhada

Este processo de mapeamento possibilita a construção de um modelo de dados integrado, consistente e extensível, alinhado às boas práticas de projetos de DW. Ao resolver conflitos semânticos e estruturais na camada de extração, o DW pode ser modelado de forma dimensional, reduzindo redundâncias e facilitando consultas analíticas sobre mensagens provenientes de diferentes plataformas. Dessa forma, o método de mapeamento proposto por Moalla et al. (2016) contribui diretamente para a qualidade dos dados armazenados e para a viabilidade de análises futuras.

3.6 Modelagem do Data Warehouse

A modelagem do DW foi realizada seguindo os princípios da modelagem dimensional proposta pelo método Kimball (KIMBALL; ROSS, 2013), com o objetivo de facilitar consultas analíticas, garantir desempenho e preservar o histórico dos dados coletados. Essa etapa define a estrutura lógica das tabelas fato e dimensão, bem como o nível de granularidade adotado para cada conjunto de dados.

3.6.1 *Data marts* presentes no projeto

Neste projeto, a organização dos dados foi projetada em dois *data marts* principais, definidos a partir dos dois principais objetivos analíticos do sistema: O armazenamento e análise de mensagens, e o armazenamento de análises de *Fake News*.

O *data mart* de mensagens é responsável pelo armazenamento e análise das mensagens vindas das diferentes plataformas utilizadas no sistema. Esse *data mart* é composto principalmente pela tabela fato `FACT_Message`, que registra cada mensagem enviada como um evento transacional, e pela tabela `FACT_Media`, que armazena informações relacionadas às mídias associadas às mensagens, como imagens e vídeos. As tabelas fatos são associadas a tabelas dimensões conformadas, como plataforma, usuário e tempo, possibilitando análises sobre volume de mensagens, origem e questões temporais.

Já o *data mart* de análise de *Fake News* é destinado ao armazenamento dos

resultados das análises de desinformação aplicadas às mensagens. Esse *data mart* tem como principal tabela fato `FACT_FakeNews_Analysis`, que registra cada análise única de uma mensagem como um evento independente, permitindo a observação dos resultados ao longo do tempo. As análises são associadas à tabela fato mensagem e às dimensões tempo e modelo de análise. Isto possibilita comparações entre diferentes modelos e versões, além de diferentes períodos.

Ambos os *data marts* compartilham dimensões conformadas, garantindo que haja consistência de dados entre as diferentes áreas de análise. Essa organização permite a expansão futura do DW através da criação de novos *data marts*.

3.6.2 Definição do Grão das Tabelas Fato

A definição do grão representa o nível mais detalhado de informação armazenada em uma tabela fato e é uma decisão fundamental na modelagem dimensional. Neste projeto, o grão da tabela fato de mensagens é definido como uma única mensagem individual em um determinado instante de tempo, considerando sua plataforma de origem e seu remetente.

Com essa definição, é possível realizar análises sobre o volume de mensagens, os períodos temporais em que foram recebidas, quais são os usuários envolvidos e de quais plataformas, bem como a evolução do conteúdo ao longo do tempo. Em caso de edições de mensagens já recebidas, cada versão deve ser tratada como entradas diferentes no DW, permitindo o armazenamento de seu histórico.

Para a tabela fato de mídias, o grão é definido como uma única mídia relacionada a uma mensagem específica. Nela, é possível haver mais de uma mídia relacionada a uma única mensagem, possibilitando a integração com APIs que tratam mídias como anexos e não como o conteúdo principal da mensagem.

Para a tabela fato relacionada à análise de *fake news*, o grão é definido como uma única análise de uma mensagem específica. A definição desta tabela possibilita a associação de métricas analíticas futuras ao conteúdo armazenado.

Todas as três tabelas fato do projeto utilizam o padrão de Tabela Fato de

Transação, como definido na Seção 2.2.1. Apesar de a tabela fato de análise de *fake news*, à primeira vista, parecer seguir o padrão de *snapshot* periódico, esta, na verdade, ainda segue o padrão de Transição. Cada entrada na tabela representa o resultado de uma única análise de uma mensagem, sem intervalos predefinidos e sem sobrescrita.

3.6.3 Organização das Dimensões

O objetivo das tabelas dimensão é fornecer contexto às tabelas fato, permitindo que as mensagens sejam analisadas sob diferentes perspectivas. Entre as principais dimensões consideradas estão as dimensões data e tempo, a dimensão plataforma, a dimensão usuário e outras dimensões descritivas relacionadas às mensagens, mídias e análises de *fake news*.

Cada dimensão é identificada por uma chave substituta (*surrogate key*), gerada internamente pelo banco de dados, o que garante independência em relação aos identificadores externos fornecidos pelas APIs. Essa abordagem facilita a integração de dados vindos de múltiplas fontes, evitando colisões e garantindo a consistência dos dados.

3.6.4 Uso das Dimensões Data e Tempo

A dimensão tempo é utilizada como uma tabela dedicada, em substituição ao uso direto de campos do tipo *timestamp*. Essa decisão permite evitar a repetição de informações temporais nas tabelas fato e possibilita a normalização dos atributos relacionados ao tempo, como dia, mês, ano, trimestre e dia da semana.

Além de reduzir redundâncias, a utilização de uma dimensão tempo facilita agregações temporais e comparações entre períodos distintos, sendo especialmente relevante para análises históricas e para o estudo da evolução da disseminação de mensagens ao longo do tempo.

Dimensões temporais são um componente essencial em modelos de DW, uma vez que praticamente todas as análises envolvem a avaliação de eventos ao longo do

tempo. Conforme indicado por Kimball e Ross (2013), recomenda-se que as tabelas fato estejam associadas a dimensões temporais específicas, em vez do uso direto de campos do tipo *timestamp*.

Neste projeto, as informações sobre o tempo são representadas por duas dimensões diferentes: Dimensão Data e Dimensão Tempo. Essa abordagem é adotada em cenários em que a hora exata de um evento é relevante, como no caso do armazenamento de mensagens.

A Dimensão Data contém atributos relacionados ao calendário, como dia, mês, ano e dia da semana. Enquanto a Dimensão Tempo representa o horário do evento, contendo atributos como hora, minuto e segundo, viabilizando análises detalhadas dentro de um mesmo dia.

A utilização dessas dimensões evita a repetição de valores temporais nas tabelas fato, melhora o desempenho das consultas analíticas e possibilita a inclusão de atributos descritivos adicionais, como o dia da semana. Ambas as dimensões utilizam chaves substitutas geradas pelo DW, seguindo as boas práticas da modelagem dimensional.

3.7 Tratamento de Mídias

As mensagens que contêm mídias, como imagens, vídeos e áudios, apresentam características distintas das mensagens exclusivamente textuais, exigindo um tratamento específico no processo de modelagem e armazenamento.

Neste trabalho, as mídias associadas às mensagens são modeladas em uma tabela fato própria, separada da tabela fato de mensagens textuais. Essa decisão se deve às diferenças de granularidade, volume e atributos associados às mídias, como tipo de arquivo, tamanho, *Uniform Resource Locator* (URL) e formato.

A separação das mídias em uma tabela fato específica permite análises independentes sobre o uso e a distribuição de conteúdos multimídia, além de evitar a sobrecarga da tabela fato principal de mensagens com atributos não aplicáveis a

todos os registros.

Com o objetivo de aumentar a capacidade analítica do DW, foi implementada uma forma de armazenar textos obtidos a partir da extração de conteúdo textual de mídias, por meio de técnicas como reconhecimento óptico de caracteres (*Optical character recognition* (OCR)) para imagens e transcrição automática para conteúdos de áudio e vídeo.

Apesar de não haver uma implementação direta, a biblioteca desenvolvida permite a injeção de uma função capaz de extrair o conteúdo textual de mídias associadas às mensagens. Essa função deve ter como entrada a URL da mídia (imagem, vídeo ou áudio) e retornar o texto extraído, quando aplicável. Com essa abordagem, o sistema se mantém desacoplado de técnicas específicas de extração, possibilitando a utilização de diferentes métodos.

O texto extraído deve ser tratado como um dado derivado, sendo armazenado de forma associada à mídia original. Essa abordagem permite que conteúdos multimídia sejam incluídos em análises textuais futuras sem comprometer a integridade dos dados originais.

3.8 Biblioteca Responsável pelo Armazenamento de Dados

Além da modelagem do DW, este projeto contempla o desenvolvimento de uma biblioteca responsável pela coleta, padronização e encaminhamento dos dados ao processo de ETL.

A biblioteca foi projetada com o intuito de abstrair as particularidades das plataformas de origem, utilizando uma interface comum para a recepção de mensagens. Essa abordagem permite que novas plataformas sejam integradas ao sistema com impacto mínimo sobre as outras camadas, facilitando a extensão das aplicações.

Foi implementado um modelo canônico de mensagens, no qual uma estrutura única que padroniza o formato dos principais dados das mensagens é utilizada, independentemente da plataforma de origem, garantindo consistência no tratamento dos

dados. Esse modelo é criado na camada de transformação, facilitando o mapeamento de campos durante o carregamento dos dados no DW.

A utilização de um modelo canônico reduz a complexidade da integração, centraliza a definição de padrões e contribui para a manutenção do sistema a longo prazo.

Capítulo 4

Implementação da Aplicação

Este capítulo descreve a arquitetura da aplicação desenvolvida neste trabalho, apresentando seus principais componentes e o fluxo de dados entre eles. A aplicação foi projetada com o objetivo de coletar mensagens provenientes de múltiplas plataformas de comunicação, normalizá-las semanticamente e armazená-las em um DW, de forma que possibilite análises futuras, com ênfase em análises relacionadas a *Fake News*.

A arquitetura adotada prioriza modularidade, extensibilidade e separação de responsabilidades, permitindo que novas plataformas de mensagens, técnicas de extração de conteúdo, formas de armazenamento e métodos de análise sejam alterados no sistema com impacto mínimo sobre os componentes existentes. Para isso, o projeto foi organizado em camadas bem definidas, que incluem desde a integração com APIs externas até o processo de ETL e a persistência dos dados no DW.

Ao longo deste capítulo, serão detalhadas as camadas e as decisões de projeto da aplicação, incluindo a integração com plataformas de mensagens, o modelo canônico adotado para a representação das mensagens, o *pipeline* ETL implementado e seus componentes responsáveis pela transformação e pelo carregamento dos dados.

4.1 Visão Geral da Arquitetura

A aplicação desenvolvida neste trabalho adota uma arquitetura modular e orientada a camadas, projetada para integrar a coleta de mensagens, a interação com os usuários e o processamento analítico dos dados de forma desacoplada e extensível. Essa arquitetura foi definida com o objetivo de permitir a ingestão contínua de mensagens provenientes de múltiplas plataformas de comunicação, bem como a condução da interação com o usuário por meio de um *chatbot*, mantendo a separação clara entre os componentes de interação, processamento e armazenamento.

De forma geral, a arquitetura é composta por três blocos principais: o módulo de *chatbot* e integração com plataformas de mensagens, o *pipeline* de ETL e a camada de persistência no DW. Uma visão geral da arquitetura do projeto é ilustrada no diagrama da Figura 4.1.

O funcionamento do sistema envolve dois atores principais. O primeiro é o usuário solicitante, responsável por interagir com o *chatbot* e fornecer a mensagem que deseja verificar. O segundo é o analista, que acessa o DW para realizar consultas e explorar os dados armazenados, obtendo métricas, padrões e informações sobre os dados das mensagens processadas.

O módulo de *chatbot* é responsável pela interação direta com os usuários e foi modelado com base no conceito de máquina de estados finitos (*Finite State Machine* (FSM)). Nesse modelo, a conversa é representada como um conjunto de estados e transições, em que cada estado define o comportamento do sistema diante das possíveis entradas predefinidas do usuário. Embora o conjunto de estados seja finito, a execução do sistema pode ocorrer de forma contínua ao longo do tempo, retornando a estados anteriores. O uso de máquinas de estados finitos é amplamente adotado na modelagem de sistemas interativos, por sua facilidade de compreensão e previsibilidade de comportamento (HOPCROFT; MOTWANI; ULLMAN, 2006).

O *pipeline* ETL constitui o ponto central da aplicação, onde as mensagens recebidas e, inicialmente, encapsuladas como mensagens brutas armazenadas na camada de *staging*, são buscadas durante a fase de extração, garantindo a preservação dos dados

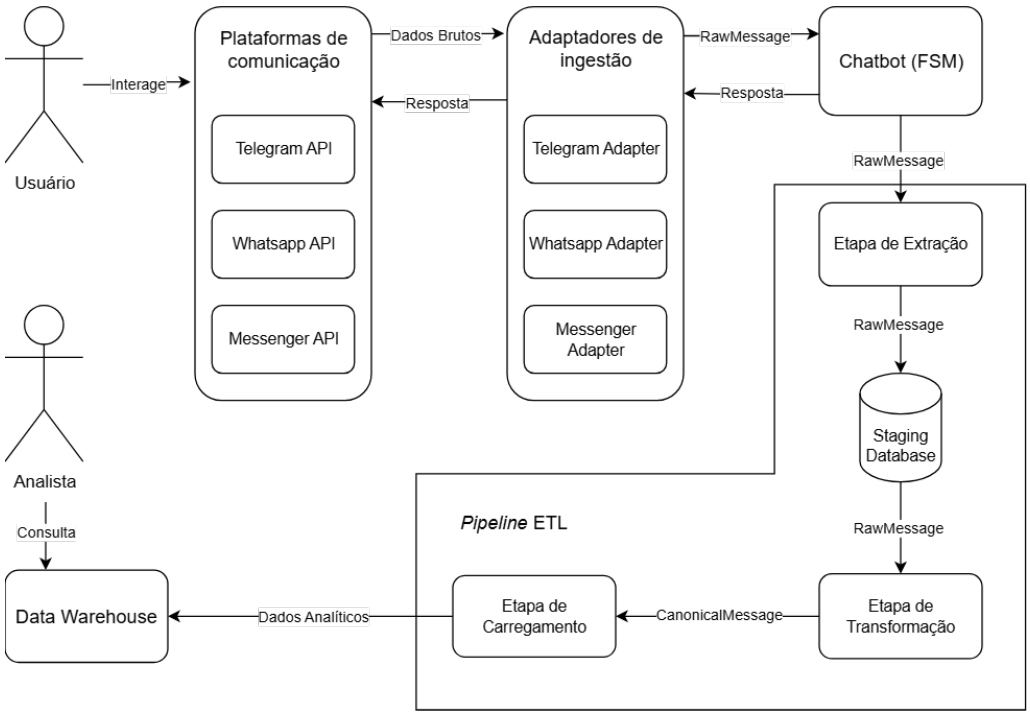

Figura 4.1: Diagrama da Arquitetura Geral do Projeto

originais. Em seguida, essas mensagens são transformadas e normalizadas semanticamente em um modelo canônico independente da plataforma, sendo posteriormente carregadas no DW segundo a modelagem dimensional adotada no projeto.

4.2 Integração com Plataformas de Mensagens

A integração com plataformas de mensagens constitui o ponto de entrada dos dados na aplicação e foi realizada por meio do padrão de projeto *Adapter* (GAMMA et al., 1995), onde são criados adaptadores específicos para cada plataforma suportada. A biblioteca foi projetada com o suporte de diferentes estratégias de ingestão em mente, como o uso de *webhooks*, mecanismos de *pooling* ou outras abordagens de coleta oferecidas pelas APIs das plataformas.

A arquitetura da aplicação oferece suporte à exposição de *endpoints Hypertext Transfer Protocol* (HTTP) para o recebimento de eventos enviados pelas plataformas por meio de *webhooks*, quando essa abordagem é adotada. Para isso, são utilizadas bibliotecas como `FastAPI`, responsável pela definição das rotas e pelo tratamento

das requisições, e `Uvicorn`, que atua como servidor *Asynchronous Server Gateway Interface* (ASGI) para a execução da aplicação. Essa infraestrutura possibilita o processamento assíncrono e concorrente de eventos, sendo adequada para cenários de alto volume de mensagens.

Independentemente da forma de coleta utilizada, esses adaptadores são responsáveis por interagir com as APIs externas, receber eventos de mensagens e encapsular os dados recebidos em uma representação padronizada de mensagem bruta, denominada `RawMessage`. Esta representação contém informações sobre a plataforma de origem da mensagem, o momento em que foi extraída e o momento em que foi processada, o *payload* bruto recebido pela API em formato *JavaScript Object Notation* (JSON), um identificador único da mensagem (utilizado para evitar redundâncias na tabela de *staging*) e um campo para armazenar a descrição de erros, caso tenham ocorrido no processamento da mensagem.

Cada adaptador implementa um contrato comum, permitindo o isolamento de particularidades de cada plataforma, como o formato do *payload* e os processos de autenticação. Essa abordagem reduz o acoplamento entre o sistema e as APIs externas, facilitando a manutenção e a adição de novas plataformas no futuro. Neste trabalho, foram implementados adaptadores para as plataformas Telegram, WhatsApp e Messenger, demonstrando sua capacidade de integração de múltiplas fontes de dados diferentes.

Os adaptadores não realizam interpretação semântica ou transformação dos dados. Sua função se limita à recepção da mensagem, à extração das informações mínimas necessárias para a identificação da plataforma, ao encapsulamento do *payload* original e ao armazenamento desses dados em um banco de dados, permitindo reprocessamentos e auditoria do processo de ingestão. Nesta parte do sistema também é utilizado o padrão de projeto *Repository* (FOWLER, 2002), utilizado na comunicação com o banco de dados através de uma interface predefinida, facilitando a mudança do DBMS utilizado (PostgreSQL[1] foi o escolhido para a implementação deste projeto).

[1] https://www.postgresql.org/

Além de atuar como interface com as plataformas externas, os adaptadores também fazem parte da ligação entre o módulo de *chatbot* e o *pipeline* ETL. As mensagens recebidas, convertidas em objetos do tipo `RawMessage`, são encaminhadas para a etapa de extração, adicionando um grau de uniformidade às mensagens enviadas pelos usuários, independentemente de sua origem.

4.3 Módulo de *Chatbot*

O módulo de *chatbot* é responsável pela interação com os usuários e atua como uma interface entre as plataformas de mensagens e o pipeline ETL. Seu objetivo principal é oferecer as opções possíveis, interpretar as respostas, receber mensagens com textos ou mídias enviadas pelos usuários e encaminhar os dados e metadados dessas mensagens para a parte do sistema responsável pela ingestão e armazenamento desses, além de fornecer respostas de acordo com o estado atual da conversa.

A lógica do *chatbot* foi modelada seguindo o conceito de máquina de estados finitos (FSM), no qual a conversa é representada por um conjunto finito de estados, transições e regras de validação de entrada. Cada estado corresponde a uma etapa do fluxo de diálogo e contém a mensagem apresentada ao usuário, o tipo de entrada esperada e quais são os próximos estados, dependendo das respostas do usuário. Essa abordagem facilita a definição da estrutura da conversa, além de facilitar a manutenção e a expansão dessa estrutura.

O gerenciamento de conversas é feito pela classe `ChatSession`, responsável por manter o estado atual da conversa e gerenciar as transições entre os estados. A cada mensagem recebida, a sessão valida a entrada do usuário de acordo com as regras definidas pelo estado atual, executa ações associadas à entrada, caso estejam definidas, e direciona a conversa para o próximo estado.

Cada estado da conversa é representado pela classe `ChatState`, que encapsula as informações necessárias para a definição do comportamento do *chatbot* em cada etapa. Cada estado define o texto da mensagem a ser exibido, o tipo de entrada esperada (opções predefinidas, texto livre ou envio de mídia) e as possíveis transições.

Opcionalmente, o sistema permite a injeção de uma função de processamento à entrada do usuário nas instâncias de estados, permitindo a execução dessa função, recebendo o conteúdo da resposta do usuário como argumento. É neste ponto que o *pipeline* ETL se conecta com o *chatbot*, injetando o método responsável pela persistência das mensagens na camada de *staging* e providenciando a entrada de dados na etapa de extração. Esta abordagem segue o padrão de projeto *Dependency Injection* (FOWLER, 2004).

Essa abordagem para o *chatbot* aqui descrita pode ser interpretada como uma aplicação do padrão de projeto *State* (GAMMA et al., 1995), no qual o comportamento do sistema varia dinamicamente conforme o estado atual. A Figura 4.2 mostra os estados e transições implementados no *chatbot* e, embora seja uma máquina simples, a combinação entre FSM e *State Pattern* reduz o acoplamento entre as diferentes etapas da conversa e facilita a inclusão de novos estados ou transições sem impacto significativo sobre a arquitetura existente.

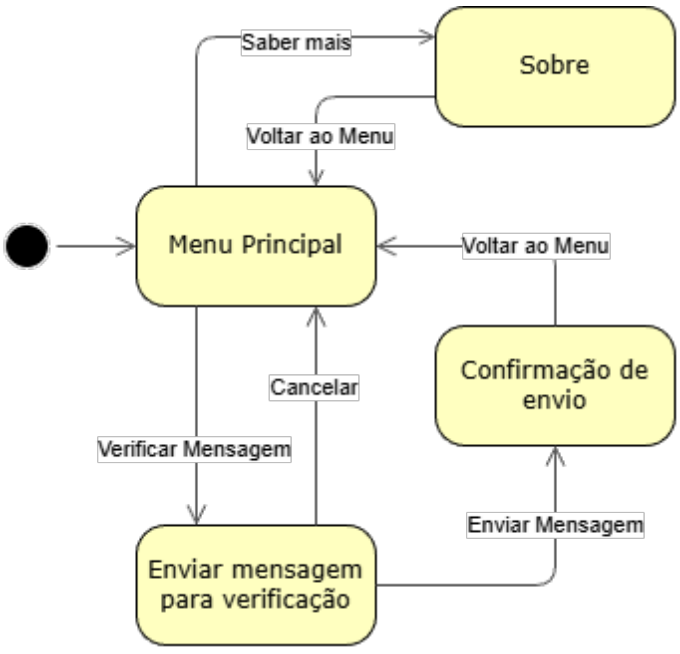

Figura 4.2: Diagrama de Máquina de Estados Finitos do *Chatbot*

As mensagens recebidas durante as interações com o *chatbot* são encapsuladas em objetos do tipo `RawMessage`, permitindo a integração do *chatbot* de maneira transparente ao *pipeline* ETL, sem realizar transformações semânticas ou normalizações, preservando a separação de responsabilidades entre os módulos de interação e

processamento de dados.

A adoção dessa arquitetura permite que o *chatbot* atue não apenas como um componente de interface com o usuário, mas também como um mecanismo de decisão para a coleta de dados, evitando o armazenamento descomedido de todas as mensagens recebidas pelo sistema, permitindo o armazenamento apenas de mensagens aprovadas pelo usuário.

4.4 *Pipeline* ETL e Processamento das Mensagens

O processamento das mensagens coletadas pela aplicação é organizado por meio de um pipeline do tipo ETL. Essa organização permite o tratamento de dados desde sua origem nas plataformas de comunicação até seu armazenamento no DW, garantindo modularidade, separação clara de responsabilidades, isolamento de falhas e facilidade de manutenção.

A etapa de extração é responsável pela coleta das mensagens a partir das diferentes plataformas, seja por meio de *webhooks*, mecanismos de *pooling* ou outras estratégias suportadas pelas APIs externas. Como descrito na Seção 4.2, independentemente da forma de obtenção, as mensagens são encapsuladas em objetos do tipo `RawMessage`, que preservam o *payload* original fornecido pela plataforma de origem, juntamente com metadados como a identificação da plataforma e o momento da ingestão. Nessa etapa, não é realizada qualquer interpretação semântica ou normalização dos dados.

As mensagens brutas extraídas são armazenadas em uma camada intermediária de persistência, aqui denominada *staging*. Essa camada tem como objetivo garantir a conservação dos dados, possibilitando reprocessamentos futuros, além de desacoplar a etapa de extração das etapas seguintes do *pipeline*.

A etapa de transformação é responsável pela conversão das mensagens brutas em um modelo canônico unificado, denominado `CanonicalMessage`. Essa conversão é realizada por transformadores específicos para cada plataforma, que conhecem a estrutura e a semântica dos dados de origem. Durante essa etapa, são aplicadas regras de normalização semântica, padronização de atributos e extração de informações

relevantes, como dados de usuários, contexto da mensagem e conteúdo textual.

Durante a criação do modelo canônico, atributos equivalentes provenientes de diferentes plataformas são mapeados para um conjunto comum de campos, como identificadores da mensagem, informações do remetente, contexto da conversa, conteúdo textual e dados temporais. A aplicação deste mapeamento está ilustrada na Tabela 3.1 e corresponde à normalização semântica dos dados descrita por Moalla et al. (2016), que garante consistência entre mensagens originadas de fontes distintas.

Quando a mensagem contém mídias associadas, a etapa de transformação também pode executar métodos adicionais de processamento, como a extração de texto a partir de imagens, vídeos ou áudios. Essa funcionalidade é desacoplada da lógica principal por meio da injeção de funções específicas, permitindo a substituição ou extensão dos algoritmos utilizados sem impactar a arquitetura do *pipeline*.

O modelo canônico resultante é independente do modelo dimensional adotado no DW e não reflete diretamente a estrutura das tabelas analíticas. Sua função é servir como uma representação intermediária consistente e validada dos dados, que pode ser reutilizada por diferentes processos de carga, análises futuras ou integrações adicionais. Essa separação entre modelo canônico e modelo dimensional segue boas práticas de arquitetura de dados e favorece a manutenção e a escalabilidade da solução.

Por fim, a etapa de carga é responsável por persistir os dados transformados no DW. Nesta etapa, os objetos canônicos são convertidos para o modelo dimensional adotado no projeto, populando as tabelas fato e dimensão de acordo com o esquema definido. A separação clara entre transformação e carga permite que ajustes no modelo analítico ou na tecnologia de armazenamento sejam realizados com impacto mínimo nas etapas anteriores.

O fluxo de processamento é gerenciado por um serviço central que coordena a recuperação das mensagens não processadas, a aplicação da transformação adequada conforme a plataforma de origem e o encaminhamento dos dados para a camada de carga. Essa organização segue o padrão de projeto *Pipeline*, e a Figura 4.4 ilustra o

diagrama de classes com as principais entidades do sistema. Segundo Buschmann et al. (1996), este padrão favorece a modularidade e a evolução independente das fases de extração, transformação e carga dos dados.

4.5 Modelo Dimensional e *Data Warehouse*

O DW desenvolvido neste projeto foi modelado seguindo a abordagem dimensional proposta por Kimball e Ross (2013), com o objetivo de facilitar análises analíticas sobre mensagens provenientes de múltiplas plataformas de comunicação. O modelo adota o esquema estrela, no qual tabelas fato representam os principais eventos de interesse e se relacionam com tabelas de dimensão que contém contexto adicional para análises.

A Figura 4.3 apresenta o diagrama entidade-relacionamento (ER) do DW, ilustrando as tabelas fato e dimensão que compõem o modelo. Neste modelo, há dois principais *Data Marts*, um focado nos dados de mensagens e outro na análise de *Fake News*.

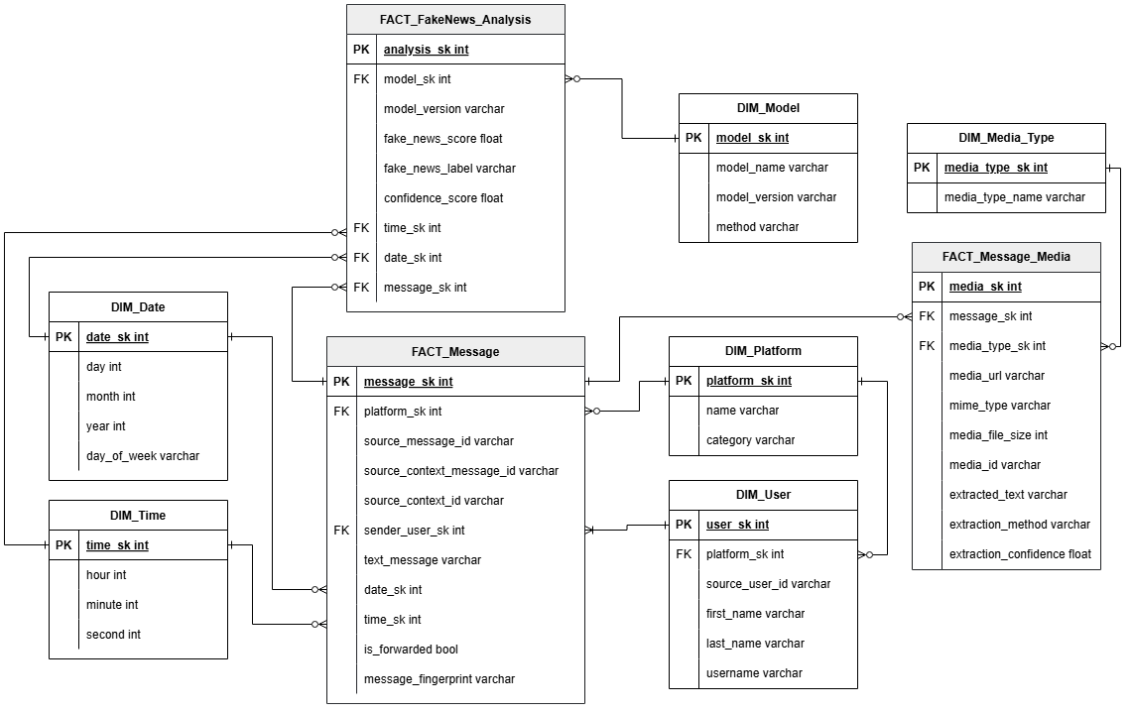

Figura 4.3: Diagrama do Esquema Estrela do Data Warehouse

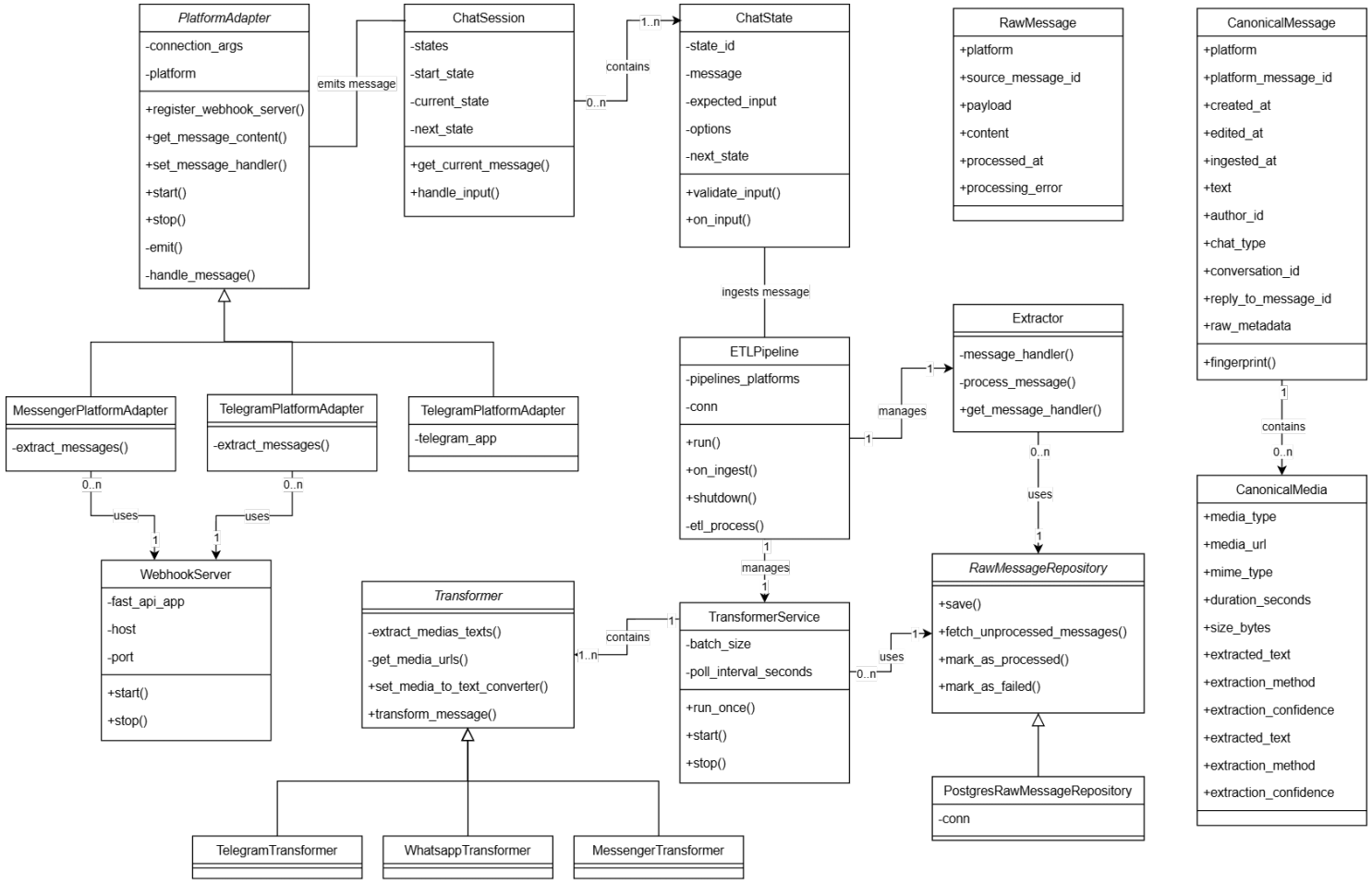

Figura 4.4: Diagrama de Classes do Sistema

O *Message Mart* pode ser considerado o núcleo do DW por ser responsável por armazenar informações relacionadas às mensagens coletadas, representadas principalmente pela tabela fato `FACT_Message`. Essa tabela registra cada mensagem individual como um evento transacional, ligando a mensagem a dimensões como plataforma, tempo, remetente e mídias. Identificadores fornecidos pelas plataformas de origem, como identificadores globais e de contexto, são tratados como dimensões degeneradas, preservando informações relevantes sem a criação de tabelas adicionais.

As mídias associadas às mensagens são armazenadas separadamente na tabela fato `FACT_Message_Media`, uma vez que uma mesma mensagem pode conter múltiplos elementos de mídia. Essa separação permite a análise independente de conteúdos multimídia, bem como o armazenamento de informações relacionadas à extração de texto, como o método utilizado e o nível de confiança, caso um método de extração tenha sido fornecido.

Além do *Message Mart*, o modelo contempla o *Fake News Analysis Mart*, representado pela tabela fato `FACT_FakeNews_Analysis`. Nesta tabela, cada registro corresponde a uma execução individual de um método ou modelo de análise sobre uma mensagem específica. Dessa forma, múltiplas análises podem ser registradas para a mesma mensagem ao longo do tempo, preservando o histórico completo de avaliações sem sobrescrita de dados, permitindo acompanhar a evolução de classificações e pontuações produzidas por diferentes modelos ou versões de algoritmos de análise.

O uso de dimensões de tempo normalizadas, em especial a separação entre data e horário, segue as boas práticas recomendadas na literatura de DW, reduzindo redundâncias e facilitando agregações temporais. Chaves substitutas são utilizadas como chaves primárias nas tabelas dimensionais, garantindo independência em relação aos identificadores fornecidos por sistemas externos e aumentando a robustez do modelo em casos de mudanças nas APIs das plataformas.

Capítulo 5

Exemplo de Uso

Este capítulo apresenta um exemplo de uso com o objetivo de demonstrar o funcionamento prático do sistema proposto, desde a recepção de mensagens vindas de plataformas de comunicação até o armazenamento dos dados no DW. O exemplo de uso ilustra o fluxo de execução do sistema com o intuito de validar a arquitetura e as decisões metodológicas adotadas, mostrando a integração entre o *chatbot*, o *pipeline* ETL e o modelo dimensional.

5.1 Fluxo de Execução do *Pipeline* ETL

Considerando o cenário em que o usuário inicia a interação com o *chatbot* por meio de diferentes plataformas de mensagens, como Telegram, WhatsApp e Messenger, com as mensagens podendo conter texto, mídias ou ambos, este exemplo de uso simula uma situação real em que conteúdos potencialmente relacionados à desinformação são compartilhados.

O *chatbot* atua como ponto de entrada do sistema, recebendo as mensagens e encaminhando-as para o *pipeline* de ingestão. A partir desse momento, os dados passam pelas etapas de extração, transformação e carga, e em seguida são armazenados no DW em formato adequado para análise.

O fluxo de execução do *pipeline* ETL se inicia com a captura da mensagem bruta

(`RawMessage`), que contém os dados no formato fornecido pela API da plataforma de origem. Esses dados são armazenados na camada de *staging*, preservando sua estrutura original.

Em seguida, a etapa de transformação é responsável por converter a mensagem bruta em um modelo canônico (`CanonicalMessage`), realizando o mapeamento semântico dos atributos e a normalização dos dados. Essa etapa também pode acionar mecanismos opcionais de extração de texto a partir de mídias associadas à mensagem.

Após a transformação, os dados são encaminhados para a etapa de carregamento, onde são persistidos no DW, respeitando o modelo dimensional definido. Cada mensagem, mídia e análise realizada é registrada como um evento transacional, garantindo o histórico completo das informações.

5.2 Consultas Analíticas Possíveis

Com os dados armazenados de forma dimensional, torna-se possível realizar diversas consultas analíticas ao DW. A seguir, são mostrados alguns exemplos de *queries* em *Structured Query Language* (SQL) para análises suportadas pelo sistema.

5.2.1 Distribuição de Mensagens por Plataforma e Período de Tempo

Esta consulta calcula a distribuição do volume de mensagens por plataforma ao longo do tempo, agrupando os registros por plataforma, ano e mês.

```
SELECT
    p.platform_name,
    d.year,
    d.month,
    COUNT(f.message_sk) AS total_messages
FROM FACT_Message f
JOIN DIM_Platform p ON f.platform_sk = p.platform_sk
JOIN DIM_Date d ON f.date_sk = d.date_sk
```

```
GROUP BY
    p.platform_name,
    d.year,
    d.month
ORDER BY
    d.year,
    d.month,
    total_messages DESC;
```

Tendo uma tabela resultante similar a esta:

platform_name	year	month	total_messages
WhatsApp	2025	12	610
Messenger	2025	12	390
Telegram	2025	12	150
WhatsApp	2026	01	700
Messenger	2026	01	640
Telegram	2026	01	210

5.2.2 Frequência de Mensagens Contendo Mídias

Esta consulta mostra a proporção de mensagens que possuem ao menos uma mídia associada.

```
SELECT
    COUNT(DISTINCT fmm.message_sk) AS messages_with_media,
    COUNT(DISTINCT fm.message_sk) AS total_messages,
    (COUNT(
        DISTINCT fmm.message_sk
    ) * 100.0 / COUNT(
        DISTINCT fm.message_sk
    )) AS percentage
FROM FACT_Message_Media fmm
```

```
RIGHT JOIN FACT_Message fm ON fmm.message_sk = fm.message_sk;
```

Tendo uma tabela resultante similar a esta:

messages_with_media	total_messages	percentage
400	1600	25.00

5.2.3 Volume de Análises de *Fake Fews* Realizadas por Modelo

Esta consulta permite comparar o uso de diferentes modelos e suas versões.

```
SELECT
    m.model_name,
    m.model_version,
    COUNT(fa.analysis_sk) AS total_analyses
FROM FACT_FakeNews_Analysis fa
JOIN DIM_Model m ON fa.model_sk = m.model_sk
GROUP BY
    m.model_name,
    m.model_version
ORDER BY total_analyses DESC;
```

Tendo uma tabela resultante similar a esta:

model_name	model_version	$\text{total}_a nalyses$
ClaimBuster	v1.0	580
FakerFact	v2.1	400
CrossCheck	beta	210

5.2.4 Correlação Entre Tipos de Mídia e Resultados de Análises

Esta consulta permite observar se determinados tipos de mídia estão associados a certos resultados de análise (considera pontuação ou classificação).

```
SELECT
```

```
    mt.media_type_name,
    fa.analysis_result,
    COUNT(*) AS total_occurrences
FROM FACT_FakeNews_Analysis fa
JOIN FACT_Message_Media fmm
ON fa.message_sk = fmm.message_sk
JOIN DIM_Media_Type mt
ON fmm.media_type_sk = mt.media_type_sk
GROUP BY
    mt.media_type_name,
    fa.analysis_result
ORDER BY
    mt.media_type_name,
    total_occurrences DESC;
```

Tendo uma tabela resultante similar a esta:

media_type_name	analysis_result	total_occurrences
Audio	Fake	120
Audio	True	80
Audio	Satire	25
Image	Fake	220
Image	True	150
Image	Satire	45
Video	Fake	90
Video	True	70
Video	Satire	20

Essas consultas evidenciam a utilidade do DW como base para análises exploratórias e futuras aplicações analíticas mais avançadas.

Capítulo 6

Conclusão

Este capítulo apresenta as considerações finais do trabalho, recapitulando os principais resultados obtidos a partir do desenvolvimento da arquitetura proposta, discutindo suas limitações e possibilidades de evolução, a fim de retomar os objetivos definidos inicialmente e avaliar as formas como estes foram atendidos ao longo do projeto.

6.1 Considerações finais

O objetivo deste trabalho foi desenvolver uma arquitetura capaz de coletar, integrar e armazenar mensagens provenientes de diferentes plataformas de comunicação, utilizando conceitos de *Data Warehousing* e modelagem dimensional, de modo a viabilizar análises futuras relacionadas à *Fake News*. Para isso, foi proposta uma solução baseada em um *pipeline* ETL modular, responsável pela ingestão de dados heterogêneos, sua normalização semântica por meio de um modelo canônico e o armazenamento dos dados em um DW relacional projetado segundo o método Kimball.

A separação clara entre as etapas de extração, transformação e carregamento permitiu maior organização do processo de ingestão e facilitou a extensibilidade da solução para novas plataformas. A modelagem dimensional adotada mostrou-

se adequada para consultas analíticas, possibilitando análises sobre o volume das mensagens, suas plataformas de origem, tipos de mídia e resultados de análises de *Fake News*.

Para possibilitar a interação com o sistema, foi desenvolvido um *chatbot* definido a partir de uma máquina de estados finita, permitindo um fluxo de conversa controlado e previsível. Essa abordagem contribuiu para a validação das entradas do usuário e para a inclusão das mensagens no *pipeline* ETL, mostrando que sua aplicação é adequada ao sistema proposto.

Os resultados obtidos indicam que a arquitetura proposta atende aos objetivos definidos, fornecendo um sistema flexível e extensível para o armazenamento e análise de dados provenientes de redes sociais e plataformas de comunicação.

6.2 Limitações e trabalhos futuros

Apesar dos resultados alcançados, o trabalho apresenta algumas limitações, sendo a principal delas a ausência de uma implementação concreta de métodos de detecção de *Fake News*, uma vez que tal análise foi considerada fora do escopo principal do projeto. Dessa forma, a tabela de análises foi projetada para suportar esses resultados, mas os modelos em si não foram integrados ao projeto.

Outra limitação está relacionada à cobertura das plataformas, uma vez que apenas um pequeno grupo das plataformas possíveis foi implementado, ainda que a arquitetura permita a inclusão de novas fontes de forma facilitada. Além disso, os mecanismos de extração de texto a partir de mídias foram tratados de forma modular, mas não foram implementados nem explorados no projeto.

Como trabalhos futuros, destaca-se a integração de diferentes modelos de detecção de desinformação, permitindo a comparação de métodos e versões diretamente no DW. Algumas outras possíveis expansões do sistema são novos adaptadores para novas plataformas, a aplicação de processos de extração de conteúdo de mídias e a realização de testes de carga e estresse para avaliar a escalabilidade do sistema.

Referências

ARMBRUST, M. et al. Lakehouse: a new generation of open platforms that unify data warehousing and advanced analytics. In: *Proceedings of CIDR*. [S.l.: s.n.], 2021. v. 8, p. 28.

BUSCHMANN, F. et al. *Pattern-Oriented Software Architecture: A System of Patterns*. Chichester, UK: John Wiley & Sons, 1996. v. 1.

CODD, E. F. Providing olap (on-line analytical processing) to user-analysts: An it mandate. *http://www. arborsoft. com/papers/coddTOC. html*, 1993.

DIXON, J. *Pentaho, Open Source Business Intelligence: Pentaho, Hadoop, and Data Lakes*. 2010. Acessado em: 26 jan. 2026. Disponível em: <https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/>.

FOWLER, M. *Patterns of Enterprise Application Architecture*. Boston, MA: Addison-Wesley Professional, 2002.

FOWLER, M. *Inversion of Control Containers and the Dependency Injection pattern*. 2004. Disponível em: <https://martinfowler.com/articles/injection.html>.

GAMMA, E. et al. *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, Mass.: Addison-Wesley Professional, 1995. ISBN 0-201-63361-2.

GORELIK, A. *The enterprise big data lake: Delivering the promise of big data and data science*. [S.l.]: O'Reilly Media, 2019.

HOPCROFT, J. E.; MOTWANI, R.; ULLMAN, J. D. *Introduction to Automata Theory, Languages, and Computation*. 3. ed. Boston, MA: Pearson/Addison-Wesley, 2006.

INMON, W. H. *Building the Data Warehouse*. [S.l.]: John wiley & sons, 2005.

KIMBALL, R.; ROSS, M. *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. [S.l.]: John Wiley & Sons, 2013.

KRAIEM, M. B.; FEKI, J. Building a data warehouse for social media: Review and comparison. *Computación y Sistemas*, Instituto Politécnico Nacional, Centro de Investigación en Computación, v. 28, n. 1, p. 19–39, 2024.

LINSTEDT, D.; OLSCHIMKE, M. *Building a scalable data warehouse with Data Vault 2.0*. [S.l.]: Morgan Kaufmann, 2015.

MOALLA, I. et al. Data warehouse design from social media for opinion analysis: The case of facebook and twitter. In: IEEE. *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*. [S.l.], 2016. p. 1–8.

VALÊNCIO, C. R. et al. Data warehouse design to support social media analysis in a big data environment. *Journal of Computer Science*, v. 16, n. 2, p. 126–136, 2020.