Relatório Técnico: Projeto de Engenharia de Dados: Pipeline ETL & Analytics (Olist)

Projeto: Construção de Data Warehouse e Análise de E-commerce

Dataset: Brazilian E-Commerce Public Dataset by Olist

Autores:

- **Alexandre Fonseca** - GitHub
- **Andre Oliveira** - GitHub
- **Carlos de Rezende** - GitHub
- **Matheus Mendonça** - GitHub

Data: Novembro/2025

---

1. Resumo Executivo

Este projeto consistiu na implementação de um pipeline de dados completo (End-to-End) para processar dados brutos de e-commerce, transformando-os em informações acionáveis para tomada de decisão. A solução arquitetada utilizou **Python** para orquestração e **PostgreSQL** para armazenamento e processamento, evoluindo os dados através de três camadas: *Staging* (Dados Brutos), *OLTP* (Normalizado) e *Data Warehouse* (Modelo Dimensional Star Schema).

2. Objetivos do Projeto

- **Engenharia de Dados:** Ingerir, limpar e normalizar dados desestruturados (CSVs).
- **Modelagem Dimensional:** Construir um Data Warehouse otimizado para consultas analíticas (OLAP).
- **Qualidade de Dados:** Garantir integridade referencial e consistência volumétrica.
- **Analytics:** Responder a perguntas de negócio (Ticket Médio, Churn, Sazonalidade) através de SQL e visualizações gráficas.

3. Arquitetura da Solução

A solução foi desenvolvida seguindo uma arquitetura de camadas dentro do banco de dados PostgreSQL, orquestrada via Jupyter Notebooks.

3.1. Stack Tecnológico

- **Linguagem:** Python 3.10+
- **Banco de Dados:** PostgreSQL 13+ (Hospedagem Cloud Aiven)
- **Bibliotecas Principais:**
    - pandas: Manipulação e validação de dados.
    - sqlalchemy / psycopg2: Conectividade e execução de queries.
    - plotly / matplotlib: Visualização de dados.

- kaleido: Exportação de imagens estáticas.

3.2. Fluxo de Dados (Pipeline)

O pipeline segue o fluxo lógico: **Raw CSVs** $\rightarrow$ **Staging** $\rightarrow$ **OLTP** $\rightarrow$ **Data Warehouse**.

Camada	Schema	Descrição
Staging	analytics.tmp_*	Tabelas UNLOGGED para ingestão rápida. Cópia fiel dos CSVs sem restrições de chave.
OLTP	analytics.tb_*	Tabelas relacionais normalizadas (3FN). Tratamento de nulos, remoção de duplicatas e limpeza de IDs.
DW	analytics.dim_*/fact_*	Modelo Dimensional (Star Schema). Histórico e métricas pré-calculadas.

---

4. Detalhamento da Implementação

4.1. Ingestão (Staging) - 00_staging.sql

A ingestão foi realizada utilizando o comando COPY do PostgreSQL para alta performance.

- **Decisão Técnica:** As tabelas foram criadas como UNLOGGED no schema analytics para evitar overhead de log de transação (WAL) e garantir persistência durante a sessão de conexão do Python.
- **Tipagem:** Utilizou-se TEXT para campos de ID para evitar erros de *overflow* e NUMERIC para valores monetários.

4.2. Normalização (OLTP) - 01_oltp.sql

Esta etapa focou na qualidade dos dados.

- **Sanitização de Chaves:** Implementou-se REPLACE(id, '-', '') para padronizar UUIDs que vinham com e sem hífens, garantindo o funcionamento dos JOINs.
- **Deduplicação:** Uso de DISTINCT ON para garantir unicidade nas chaves primárias.
- **Tratamento de Nulos:** Aplicação de COALESCE para preencher lacunas (ex: cidades nulas viraram 'Desconhecido').
- **Integridade:** As tabelas filhas (ex: tb_order_items) só receberam dados se as chaves estrangeiras (Clientes/Produtos) existissem nas tabelas pais.

4.3. Modelagem Dimensional (DW) - 02_dw_model.sql

Adotou-se o modelo **Star Schema** (Esquema Estrela) pela sua eficiência em consultas de leitura.

- **Fato:** fact_sales (Granularidade: Item do pedido).
- **Dimensões:**
    - dim_customer, dim_product, dim_seller: Contêm atributos descritivos.
    - dim_date: Dimensão de calendário gerada para análises temporais robustas.
- **Surrogate Keys (SK):** Criação de chaves artificiais (SERIAL) inteiras para otimizar a performance dos JOINs, desacoplando o DW das chaves de negócio (Business Keys).

4.4. Carga e Transformação (ETL Load) - 03_etl_load.sql

- **Geração de Datas:** Uso de generate_series do PostgreSQL para criar um calendário contínuo baseado no intervalo de datas dos pedidos.
- **Carga Incremental:** As dimensões foram carregadas com lógica de verificação (LEFT JOIN ... WHERE NULL) para garantir a idempotência (o script pode rodar várias vezes sem duplicar dados).
- **SCD (Slowly Changing Dimensions):** A estrutura da dim_customer foi preparada com campos effective_from e is_current para suportar rastreamento histórico (SCD Tipo 2).

5. Validação e Qualidade de Dados

Um script dedicado (04_validation.sql) foi desenvolvido para auditoria automática pós-carga:

1. **Volumetria:** Comparação de COUNT(*) entre OLTP e DW. O resultado mostrou consistência (112.650 itens processados).
2. **Integridade Referencial:** Verificação de chaves nulas na tabela fato. O teste retornou **0 erros**, confirmando que todas as vendas estão ligadas a dimensões válidas.
3. **Consistência Financeira:** O somatório do valor total de vendas (SUM(price)) foi idêntico na origem e no destino.

6. Resultados Analíticos

Através dos notebooks 02_analytics.ipynb e 03_visualizations.ipynb, foram gerados insights relevantes:

1. **Evolução Temporal:** Identificou-se um crescimento consistente nas vendas, com um pico significativo na **Black Friday de 2017 (Novembro)**.

2. **Performance de Produtos:** A análise "Melhor Mês vs. Pior Mês" (gráfico de barras comparativo) revelou que certos produtos têm alta volatilidade, dependendo fortemente de sazonalidade.
3. **Retenção (Cohort):** A análise de Coorte demonstrou que a taxa de recompra é baixa nos meses subsequentes à primeira compra, indicando uma oportunidade para estratégias de fidelização.
4. **Ticket Médio:** O monitoramento mensal mostrou estabilidade no valor médio gasto por pedido, sugerindo que o crescimento da receita é impulsionado pelo volume de novos clientes, e não pelo aumento do gasto individual.

7. Conclusão

O projeto atingiu com sucesso o objetivo de transformar dados brutos em um Data Warehouse confiável. A arquitetura modular (scripts SQL separados orquestrados por Python) provou-se robusta, facilitando a manutenção e a depuração. As validações implementadas garantem que os dados consumidos pelos dashboards são íntegros, fornecendo uma base sólida para a tomada de decisões estratégicas no contexto do e-commerce Olist.