

TÓPICOS AVANÇADOS EM INTELIGÊNCIA ARTIFICIAL II

DW Northwind + Metabase

Consolidação de ERP (B2B) e SaaS (B2C) com dbt e PostgreSQL em um Data Warehouse

Visão do Projeto

O Cenário

Uma empresa opera com dois modelos de negócio distintos:

- ▶ **Northwind (B2B):** Venda de produtos alimentícios (ERP legado).
- ▶ **Metabase Sample (B2C):** SaaS de assinaturas e produtos tecnológicos.

O Objetivo

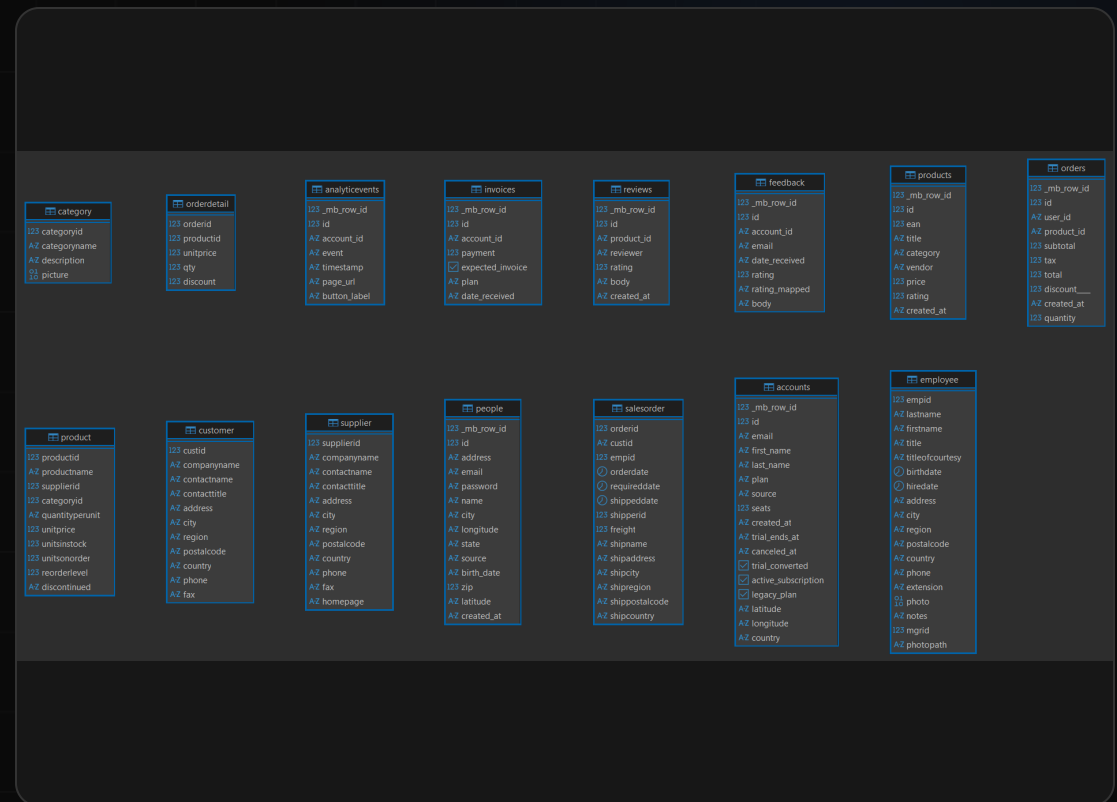
Unificar essas origens díspares em um único **Data Warehouse** no PostgreSQL, utilizando **dbt** para orquestração e transformação, habilitando análises cruzadas de produtos e performance de vendas.

Arquitetura Lógica (FDW)

Unificação via Foreign Data Wrapper

Para evitar pipelines complexos de ingestão inicial, utilizamos o recurso nativo **PostgreSQL FDW**.

- ▶ As bases northwind e sample_metabase são mapeadas como schemas estrangeiros dentro do DW.
- ▶ O dbt lê diretamente do schema public (Raw), onde essas tabelas estão expostas.



Pipeline de Transformação (DAG)



Raw (FDW) → Staging (Padronização) → Intermediate (Consolidação) → Snapshots (SCD2) → Fatos/Dimensões.

Gráfico interativo gerado pelo comando `dbt docs generate` seguido pelo comando `dbt docs serve`

Desafio: Historização (SCD2)

O Problema

Precisávamos historizar mudanças de clientes e produtos (ex: mudança de endereço ou preço), mas as bases de origem **não possuíam colunas confiáveis de auditoria** (updated_at).

A Solução: Strategy 'Check'

Adotamos a estratégia de snapshot check do dbt.

- ▶ O dbt cria um hash das colunas monitoradas (ex: city, unit_price).
- ▶ Quando o hash muda, uma nova versão é criada no DW.
- ▶ **Trade-off:** Aceita-se uma latência de D-1 em troca de não precisar alterar o banco de produção com triggers.

Snapshot Check

Entidade	Colunas Monitoradas
Customers	customer_name, contact_name, first_name, last_name, address, city, region, country, postal_code, plan, segment, seats, active_subscription, canceled_at
Products	product_name, supplier_name, unit_price, rating_score
Suppliers	supplier_name, contact_name, city, region, country
Employees	first_name, last_name, job_title, city, region, country, postal_code, manager_empid

Lógica de Backdating

Evitando "Fatos Órfãos"

Como o snapshot começou a rodar hoje, vendas de 1998 ficariam sem dimensão correspondente.

Implementação:

- ▶ Na dimensão, se for a primeira versão do registro, forçamos o `dbt_valid_from` para 1900-01-01.
- ▶ Isso garante que joins históricos (ex: `fact_sales` join `dim_customer`) funcionem para dados legados.

suppliers_snapshot	employees_snapshot	products_snapshot	customers_snapshot
AZ supplier_nk	AZ employee_nk	AZ product_nk	AZ customer_nk
AZ source_system	AZ source_system	AZ source_system	AZ source_system
AZ source_supplier_id	123 source_employee_id	123 source_product_id	AZ source_customer_id
AZ supplier_name	AZ first_name	AZ supplier_nk	AZ customer_name
AZ contact_name	AZ last_name	AZ category_nk	AZ contact_name
AZ contact_title	AZ job_title	AZ product_name	AZ contact_title
AZ city	AZ courtesy_title	AZ category_name	AZ first_name
AZ region	🕒 birth_date	AZ supplier_name	AZ last_name
AZ country	🕒 hire_date	AZ package_details	AZ address
AZ phone	AZ city	123 unit_price	AZ city
🕒 dw_load_ts	AZ region	123 units_in_stock	AZ region
AZ dbt_scd_id	AZ country	123 units_on_order	AZ country
🕒 dbt_updated_at	AZ postal_code	123 reorder_level	AZ postal_code
🕒 dbt_valid_from	AZ phone	123 rating_score	AZ plan
🕒 dbt_valid_to	AZ phone_extension	🕒 product_created_at	AZ segment
	123 manager_empid	<input checked="" type="checkbox"/> is_discontinued	123 seats
	🕒 dw_load_ts	🕒 dw_load_ts	<input checked="" type="checkbox"/> trial_converted
	AZ dbt_scd_id	AZ dbt_scd_id	<input checked="" type="checkbox"/> active_subscription
	🕒 dbt_updated_at	🕒 dbt_updated_at	<input checked="" type="checkbox"/> legacy_plan
	🕒 dbt_valid_from	🕒 dbt_valid_from	🕒 created_at
	🕒 dbt_valid_to	🕒 dbt_valid_to	🕒 trial_ends_at
			🕒 canceled_at
			🕒 dw_load_ts
			AZ dbt_scd_id
			🕒 dbt_updated_at
			🕒 dbt_valid_from
			🕒 dbt_valid_to

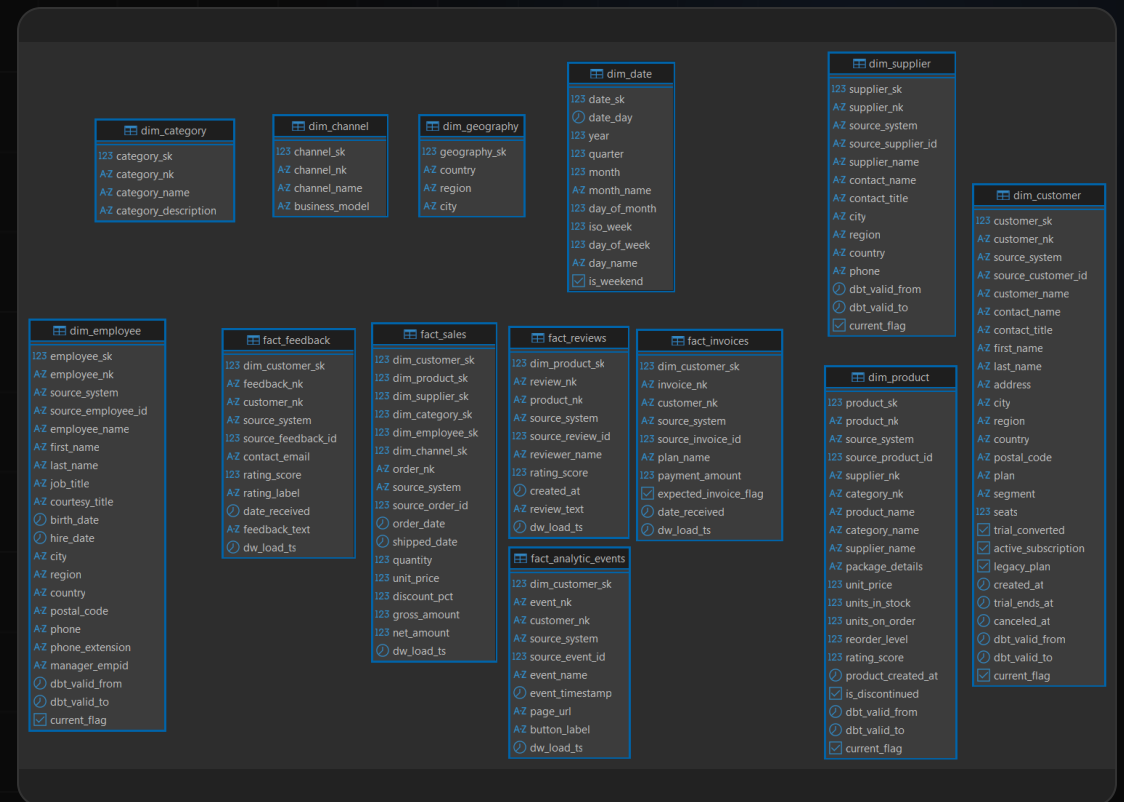
Modelagem Star Schema (DW Core)

Dimensões Unificadas

- ▶ **dim_customer:** Unifica clientes B2B (Northwind) e usuários B2C (Metabase).
- ▶ **dim_product:** Produtos físicos e planos de assinatura.

Fatos de Negócio

- ▶ **fact_sales:** Vendas consolidadas.
- ▶ **fact_analytic_events:** Eventos de clique/navegação do SaaS.



Data Marts

A camada final (Gold) entrega tabelas prontas para consumo (OBT - One Big Table):

- ▶ **dm_sales:** Visão geral de receita.
- ▶ **dm_b2b / dm_b2c:** Visões segmentadas por modelo de negócio.
- ▶ **dm_saas:** Métricas de recorrência e churn.
- ▶ **dm_product_analytics:** Performance de produtos vs avaliações.

dm_product_analytics
AZ record_type
AZ record_nk
AZ customer_nk
AZ product_nk
AZ metric_name
123 metric_value
🕒 metric_timestamp
AZ context_info
AZ customer_name
AZ segment
AZ product_name

dm_saas
AZ invoice_nk
AZ customer_nk
AZ plan_name
123 payment_amount
<input checked="" type="checkbox"/> expected_invoice_flag
🕒 invoice_date
AZ customer_name
AZ segment
AZ plan
123 year
123 month
AZ month_name

dm_b2b
AZ order_nk
AZ source_system
🕒 order_date
123 quantity
123 gross_amount
123 net_amount
AZ customer_name
AZ customer_city
AZ segment
AZ customer_country
AZ product_name
AZ category_name
AZ supplier_name
AZ employee_name
AZ job_title
AZ channel_name

dm_b2c
AZ order_nk
AZ source_system
🕒 order_date
123 quantity
123 gross_amount
123 net_amount
AZ customer_name
AZ customer_city
AZ segment
AZ customer_country
AZ product_name
AZ category_name
AZ supplier_name
AZ employee_name
AZ job_title
AZ channel_name

dm_sales
AZ order_nk
AZ source_system
🕒 order_date
123 quantity
123 gross_amount
123 net_amount
AZ customer_name
AZ customer_city
AZ segment
AZ customer_country
AZ product_name
AZ category_name
AZ supplier_name
AZ employee_name
AZ job_title
AZ channel_name

Stack Tecnológico

Python 3.13

Ambiente virtual e gestão de dependências.

dbt-postgres 1.9

Modelagem, testes (Data Quality) e Snapshots.

PostgreSQL

Data Warehouse local com extensão FDW ativa.

** Utilizamos o PostgreSQL 12 pela facilidade de já termos instalado na máquina, mas o código é aderente a versões mais recentes.*

Dúvidas?

Documentação completa e código fonte:

github.com/thalessalvador/DW

Comandos principais:

```
dbt run --select tag:staging
dbt run --select intermediate
dbt snapshot --select products_snapshot suppliers_snapshot employees_snapshot customers_snapshot
dbt run --select dim_customer dim_product dim_supplier dim_employee dim_category dim_date dim_geography dim_channel fact_sales fact_invoices fact_feedback
fact_reviews fact_analytic_events dbt run --select dm_sales dm_b2b dm_b2c dm_saas dm_product_analytics
```

Comandos acessórios:

```
dbt run tests
dbt docs generate
dbt docs serve
```

Alunos

Thales Augusto Salvador

Carlos Henrique Barbosa da Silva