**PROJETO SALES TECH**

PARTICIPANTES:

Augusto Batista – RA: 10444612

Daniela Alexandra – RA: 10444894

Luan Ferrazzo – RA: 10397276

Objetivo:

Criar um fluxo que extraia dados transacionais de um banco de dados relacional, de um e-commerce brasileiro, e transformá-los para atender a camada analítica afim de promover melhorias nas vendas e promoções. O projeto será com processamento de rotina Batch devido atarefas que não demandam resposta rápida e que envolvem grandes volumes de dados estáticos, como relatórios financeiros, geralmente em intervalos programados (diário, semanal).

**Características:**

* Trabalha com **lotes grandes de dados**.
* Ideal para **tarefas não interativas** e que podem ser realizadas em horários fora de pico (ex.: durante a madrugada).
* Alta **latência**, já que o processamento só começa quando todos os dados estão disponíveis.
* Usado em sistemas como **ETL** (Extract, Transform, Load), relatórios gerenciais.

1. Etapas de criação do modelo relacional

Criação do banco de dados transacional – OLTP

DIAGRAMA MODELO RELACIONAL:

Diagrama

Descrição gerada automaticamente

CAMINHO:

\\DATA-PREP-TRANSFORMATION\ecommerce\SQL\ecommerce\_relacional.sql

Scripts das tabelas em SQL:

*-- Criação do banco de dados*

*CREATE DATABASE ecommerce;*

*-- Usar o banco de dados criado*

**USE** ecommerce;

*-- Criação da tabela* product\_category\_name\_translation

**CREATE** **TABLE** product\_category\_name\_translation (

        product\_category\_name **NVARCHAR**(50),

        product\_category\_name\_english **NVARCHAR**(255));

*-- Criação da tabela geolocation*

**CREATE** **TABLE** geolocation (

        geolocation\_zip\_code\_prefix **CHAR**(5),

        geolocation\_lat **FLOAT**,

        geolocation\_lng **FLOAT**,

        geolocation\_city **NVARCHAR**(255),

        geolocation\_state **CHAR**(2));

*-- Criação da tabela customers com chave estrangeira*

**CREATE** **TABLE** customers (

        customer\_id **NVARCHAR**(50) **PRIMARY KEY**,

        customer\_unique\_id **NVARCHAR**(50),

        customer\_zip\_code\_prefix **CHAR**(5),

        customer\_city **NVARCHAR**(255),

        customer\_state **CHAR**(2));

*-- Criação da tabela* sellers

**CREATE** **TABLE** sellers (

        seller\_id **NVARCHAR**(50) **PRIMARY KEY**,

        seller\_zip\_code\_prefix **CHAR**(5),

        seller\_city **NVARCHAR**(255),

        seller\_state **CHAR**(2));

*-- Criação da tabela* products

**CREATE** **TABLE** products (

        product\_id **NVARCHAR**(50) **PRIMARY KEY**,

        product\_category\_name **NVARCHAR**(50),

        product\_name\_length **NVARCHAR**(50),

        product\_description\_length **NVARCHAR**(50),

        product\_photos\_qty **INT**,

        product\_weight\_g **INT**,

        product\_length\_cm **FLOAT**,

        product\_height\_cm **FLOAT**,

        product\_width\_cm **FLOAT** );

*-- Criação da tabela* orders

**CREATE** **TABLE** orders (

        order\_id **NVARCHAR**(50) **PRIMARY KEY**,

        customer\_id **NVARCHAR**(50),

        order\_status **NVARCHAR**(50),

        order\_purchase\_timestamp **DATETIME**,

        order\_approved\_at **DATETIME**,

        order\_delivered\_carrier\_date **DATETIME**,

        order\_delivered\_customer\_date **DATETIME**,

        order\_estimated\_delivery\_date **DATETIME**,

**FOREIGN KEY** (customer\_id) **REFERENCES** customers(customer\_id));

*-- Criação da tabela*  order\_reviews

**CREATE** **TABLE** order\_reviews (

        review\_id **NVARCHAR**(50),

        order\_id **NVARCHAR**(50),

        review\_score **INT**,

        review\_comment\_title **NVARCHAR**(MAX),

        review\_comment\_message **NVARCHAR**(MAX),

        review\_creation\_date **DATETIME**,

        review\_answer\_timestamp **DATETIME**,

**CONSTRAINT** PK\_order\_reviews **PRIMARY KEY** (review\_id, order\_id),

*-- Chave composta*

**FOREIGN KEY** (order\_id) **REFERENCES** orders(order\_id));

*-- Criação da tabela*  order\_payments

**CREATE** **TABLE** order\_payments (

        order\_id **NVARCHAR**(50),

        payment\_sequential **INT**,

        payment\_type **NVARCHAR**(50),

        payment\_installments **INT**,

        payment\_value **DECIMAL**(10, 2),

**PRIMARY KEY** (order\_id, payment\_sequential),

**FOREIGN KEY** (order\_id) **REFERENCES** orders(order\_id));

*-- Criação da tabela*  order\_items

**CREATE** **TABLE** order\_items (

        order\_id **NVARCHAR**(50),

        order\_item\_id **INT**,

        product\_id **NVARCHAR**(50),

        seller\_id **NVARCHAR**(50),

        shipping\_limit\_date **DATETIME**,

        price **DECIMAL**(10, 2),

        freight\_value **DECIMAL**(10, 2),

**PRIMARY KEY** (order\_id, order\_item\_id),

**FOREIGN KEY** (order\_id) **REFERENCES** orders(order\_id),

**FOREIGN KEY** (product\_id) **REFERENCES** products(product\_id),

**FOREIGN KEY** (seller\_id) **REFERENCES** sellers(seller\_id));

1. Etapa de Carregamento dos dados transacionais:

Foi gerado um script em python para fazermos ETL e após a inserção dos dados dos respectivos csv’s para o banco relacional:

**ARQUIVOS: TABLE\_NAME**:

  "product\_category\_name\_translation.csv": "product\_category\_name\_translation",

    "olist\_geolocation\_dataset.csv": “geolocation",

    "olist\_customers\_dataset.csv": "customers",

    "olist\_sellers\_dataset.csv": "sellers",

    "olist\_products\_dataset.csv": "products",

    "olist\_orders\_dataset.csv": "orders",

    "olist\_order\_reviews\_dataset.csv": "order\_reviews",

    "olist\_order\_payments\_dataset.csv": "order\_payments",

    "olist\_order\_items\_dataset.csv": "order\_items"

CAMINHO:

[\\DATA-PREP-TRANSFORMATION\ecommerce\ETL\notebook\_ ecommerce\_csv\_to\_relacional.py](file:///\\DATA-PREP-TRANSFORMATION\ecommerce\ETL\notebook_%20ecommerce_csv_to_relacional.py)

1. Etapa de Criação dos modelos Estrela:

O **modelo Star Schema** é um modelo de dados analítico amplamente utilizado em **data warehouses** para facilitar consultas e análises de dados. Ele organiza os dados em uma estrutura simples e intuitiva, com uma tabela central chamada **fato**(contém os dados quantitativos (medidas ou métricas)) e tabelas auxiliares chamadas **dimensões(**Armazenam os atributos descritivos usados para filtrar, categorizar ou detalhar os dados da tabela fato).

**Estrutura:**

* **Tabela Fato**: faro\_sales
* **Tabelas Dimensão**: dim\_date, dim\_seller, dim\_customer e dim\_product

CAMINHO: [**\\DATA-PREP-TRANSFORMATION\ecommerce\SQL\ecommerce\_fato\_dimensao.sql**](file:///\\DATA-PREP-TRANSFORMATION\ecommerce\SQL\ecommerce_fato_dimensao.sql)

DIAGRAMA MODELO STAR SCHEMA: Diagrama

Descrição gerada automaticamente

Script modelo:

*-- Criação do banco de dados*

*--CREATE DATABASE ecommerce\_fatodim;*

*-- Usar o banco de dados criado*

**USE** ecommerce\_fatodim;

*-- Tabela de data*

**CREATE** **TABLE** dim\_date (

    date\_id **INT** **PRIMARY KEY**,

**year** **INT**,

**month** **INT**,

**day** **INT**);

*-- Tabela de produto*

**CREATE** **TABLE** dim\_product (

    product\_id **NVARCHAR**(50) **PRIMARY KEY**,

    product\_category\_name **NVARCHAR**(50),

    product\_name\_length **NVARCHAR**(50),

    product\_description\_length **NVARCHAR**(50),

    product\_photos\_qty **INT**,

    product\_weight\_g **INT**,

    product\_length\_cm **FLOAT**,

    product\_height\_cm **FLOAT**,

    product\_width\_cm **FLOAT**);

*-- Tabela de cliente*

**CREATE** **TABLE** dim\_customer (

    customer\_id **NVARCHAR**(50) **PRIMARY KEY**,

    customer\_city **NVARCHAR**(255),

    customer\_state **CHAR**(2));

*-- Tabela de vendedor*

**CREATE** **TABLE** dim\_seller (

    seller\_id **NVARCHAR**(50) **PRIMARY KEY**,

    seller\_city **NVARCHAR**(255),

    seller\_state **CHAR**(2));

*-- Tabela de fato de vendas*

**CREATE** **TABLE** fato\_sales (

    sale\_id **INT**,

    date\_id **INT**,

    order\_id **NVARCHAR**(50),

    product\_id **NVARCHAR**(50),

    seller\_id **NVARCHAR**(50),

    customer\_id **NVARCHAR**(50),

    order\_purchase\_timestamp **DATETIME**,

    price **DECIMAL**(10,2),

    freight\_value **DECIMAL**(10,2),

    total\_value **DECIMAL**(10,2),

    quantity\_sold **INT**,

**FOREIGN KEY** (product\_id) **REFERENCES** dim\_product(product\_id),

**FOREIGN KEY** (seller\_id) **REFERENCES** dim\_seller(seller\_id),

**FOREIGN KEY** (customer\_id) **REFERENCES** dim\_customer(customer\_id),

**FOREIGN KEY** (date\_id) **REFERENCES** dim\_date(date\_id));

1. Etapa de Criação do modelo Wide Table:

O **modelo Wide Table** é uma abordagem analítica em que todos os dados relevantes para análise são consolidados em uma única tabela ampla, contendo um grande número de colunas. Cada linha representa uma unidade de análise (como um cliente ou transação), enquanto as colunas incluem todas as métricas, atributos e variáveis necessárias.

**Características:**

* Uma única tabela com muitas colunas (potencialmente milhares).
* Dados preparados para facilitar consultas diretas sem a necessidade de *joins* entre tabelas.
* Geralmente usados em tarefas de **machine learning** ou análises exploratórias, onde é necessário acessar muitos atributos relacionados ao mesmo objeto.

CAMINHO: [\\DATA-PREP-TRANSFORMATION\ecommerce\SQL\ecommerce\_ride\_table.sql](file:///\\DATA-PREP-TRANSFORMATION\ecommerce\SQL\ecommerce_ride_table.sql)

DIAGRAMA MODELO WIDE TABLE:

Interface gráfica do usuário

Descrição gerada automaticamente com confiança média

Script SQL:

*-- Criação do banco de dados*

**CREATE** **DATABASE** ecommerce\_wide;

*-- Usar o banco de dados criado*

**USE** ecommerce\_wide;

**CREATE** **TABLE** wide\_sales (

**year** **INT**,

**month** **INT**,

    total\_sales\_value **DECIMAL**(10,2),

    total\_units\_sold **INT**,

    total\_freight\_value **DECIMAL**(10,2),

    total\_orders **INT**,

    product\_category\_name **NVARCHAR**(50),

    product\_name **NVARCHAR**(255),

    customer\_city **NVARCHAR**(255),

    seller\_city **NVARCHAR**(255));

1. Etapa de Carregamento dos dados analíticos:

Após efetuado a criação e execução dos scripts das tabelas nas camadas analítica efetuamos o processamento batch dos dados – ETL onde foi disponibilizado as informações do transacional para os modelos analíticos Star Schema e Wide Table garantindo dados e informação para análises:

CAMINHO StarSchema:

\ \DATA-PREP-TRANSFORMATION\ecommerce\ETL\notebook\_ ecommerce\_relacional\_to\_fatodim.py

CAMINHO WideTable:

\ \DATA-PREP-TRANSFORMATION\ecommerce\ETL\notebook\_ ecommerce\_relacional\_to\_widetable.py

---------------------------------------------------------------------------------------------------------

POSSIVEIS ANÁLISES:

Desempenho de Vendedores por Estado

SELECT

ds.seller\_state AS Estado,

COUNT(DISTINCT fs.seller\_id) AS TotalVendedores,

SUM(fs.total\_value) AS TotalVendas

FROM

fato\_sales fs

JOIN

dim\_seller ds ON fs.seller\_id = ds.seller\_id

GROUP BY

ds.seller\_state

ORDER BY

TotalVendas DESC;

Total de Vendas por Categoria de Produto em um Ano

SELECT

dp.product\_category\_name AS Categoria,

SUM(fs.total\_value) AS TotalVendas

FROM

fato\_sales fs

JOIN

dim\_product dp ON fs.product\_id = dp.product\_id

JOIN

dim\_date dd ON fs.date\_id = dd.date\_id

WHERE

dd.year = 2023

GROUP BY

dp.product\_category\_name

ORDER BY

TotalVendas DESC;

## Conclusão

O projeto de e-commerce demonstrou como a organização e transformação de dados pode ser usada para atender a demandas analíticas. A implementação de rotinas Batch, o uso do modelo Star Schema e Wide Table possibilitaram um fluxo de dados otimizado, fornecendo informações valiosas para estratégias de vendas e promoções.