# **Relatório Técnico: Análise e Processamento de Dados de Vendas de Automóveis**

Kymie Karina Silva Saito

## **1. Introdução**

Este relatório técnico descreve o processo de análise e tratamento de dados de vendas de automóveis, utilizando técnicas de pré-processamento, consultas SQL e análises estratégicas. O objetivo é extrair insights valiosos para auxiliar na tomada de decisões estratégicas relacionadas ao desempenho de vendas, segmentação por faixa de preço e identificação de modelos mais vendidos.

Inicialmente, foi realizado duas tarefas principais: (1) recuperar e corrigir dados de dois arquivos JSON (broken\_database\_1.json e broken\_database\_2.json) contendo informações de vendas de veículos e marcas, e (2) gerar um relatório detalhado de desempenho de vendas utilizando consultas SQL no BigQuery. A solução foi implementada em um notebook Jupyter (SQLtest\_kymie.ipynb), utilizando Python para pré-processamento, SQL para consultas e SQLite como banco de dados em memória para prototipagem rápida.

## **2. Visão Geral do Projeto**

O case simula um cenário real onde um engenheiro de software gerencia o banco de dados de uma concessionária multimarcas. Os dados em broken\_database\_1.json (vendas) e broken\_database\_2.json (marcas) contêm erros como caracteres especiais, tipos de dados incorretos e nomes de colunas inconsistentes. As tarefas incluem:

* Corrigir formatos de dados e garantir integridade.
* Unir tabelas para criar um conjunto de dados coeso.
* Desenvolver consultas SQL para responder a perguntas de negócios sobre desempenho de vendas.
* Fornecer recomendações estratégicas baseadas na análise de dados.
* Documentar o processo e exportar os dados corrigidos.

## **3. Metodologia**

### **3.1. Pré-Processamento dos Dados**

Os dados foram extraídos de arquivos JSON (broken\_database\_1.json e broken\_database\_2.json) e tratados para garantir qualidade e consistência. As etapas incluíram:

* **Substituição de caracteres especiais** (ex.: 'æ' → 'a', 'ø' → 'o', 'å' → 'a').
* **Conversão de tipos de dados** (ex.: coluna "vendas" para numérico).
* **Padronização de nomes de colunas** (ex.: id\_marca\_ → id\_marca).

## **3. Metodologia**

### **3.1 Pré-processamento de Dados**

O pré-processamento envolveu a carga e correção dos arquivos JSON utilizando a biblioteca pandas:

* **Extração de Dados**:
  + broken\_database\_1.json foi carregado em um DataFrame pandas (vendas) com colunas como nome (nome do veículo), vendas (volume de vendas), valor\_do\_veiculo (preço do veículo), id\_marca (ID da marca) e data (data da venda).
  + broken\_database\_2.json foi carregado em um DataFrame (marcas) com colunas id\_marca e marca (nome da marca).
* **Correção de Caracteres**:
  + Uma função corrigir\_caracteres foi criada para substituir caracteres especiais (æ → a, ø → o, The vendasemarcasforam corrigidos diretamente no Python, garantindo que campos comonomeemarca` estivessem no formato correto.
  + Exemplo: Fiat com æ foi corrigido para Fiat.
* **Conversão de Tipos de Dados**:
  + A coluna vendas do DataFrame de vendas foi convertida para numérico usando pd.to\_numeric com errors='coerce' para lidar com entradas inválidas.
  + Valores ausentes foram preservados para manter a integridade dos dados.
* **Padronização de Colunas**:
  + A coluna id\_marca\_ no DataFrame de vendas foi renomeada para id\_marca para consistência com o DataFrame de marcas.
* **Configuração do Banco de Dados**:
  + Um banco de dados SQLite em memória foi criado usando SQLAlchemy (create\_engine('sqlite:///:memory:')).
  + Os DataFrames corrigidos foram carregados em tabelas SQL (vendas e marcas) para consultas.

### **3.2 Desenvolvimento de Consultas SQL**

As consultas SQL foram desenvolvidas para atender aos requisitos do case. Duas funções utilitárias foram criadas:

* execute\_sql(query): Executa consultas de modificação (INSERT, UPDATE, DELETE) com commit.
* query\_sql(query): Executa consultas SELECT e retorna resultados como um DataFrame pandas.

As consultas foram estruturadas para serem compatíveis com a sintaxe do BigQuery, utilizando recursos SQL padrão como CTEs, JOINs e funções de janela.

#### **3.2.1 Recuperação de Dados (Tarefa 1)**

As seguintes consultas SQL foram desenvolvidas para recuperar os dados originais:

**1.1 Correção de Campos de Veículos e Marcas**:

A correção de caracteres foi realizada no pré-processamento em Python devido ao formato de entrada JSON. Os dados corrigidos foram carregados no banco SQLite, garantindo que campos como nome e marca estivessem no formato correto.

Não foi necessário usar SQL para esta etapa, pois as correções foram aplicadas antes do carregamento no banco.

**1.2 Conversão de Valores de Vendas**:

A coluna vendas foi pré-processada para garantir o formato numérico.

**1.3 Unindo Tabelas**:

Um conjunto de dados coeso foi criado unindo as tabelas vendas e marcas.

Um conjunto de dados coeso foi criado unindo as tabelas vendas e marcas com um INNER JOIN, garantindo que apenas registros com id\_marca correspondente fossem incluídos, preservando a integridade dos dados.

**1.4 Identificação de Marcas Sem Vendas**:

Um LEFT JOIN foi utilizado para identificar marcas sem registros de vendas. O resultado foi um conjunto vazio, indicando que todas as marcas possuíam vendas associadas.

**1.5 Uso de CTE e Funções de Janela**:

Uma CTE foi empregada para calcular a média de vendas por marca e identificar aquelas com desempenho acima da média. Marcas como Fiat, Volkswagen e Kia foram destacadas como de alto desempenho.

#### **3.2.2 Relatório de Vendas (Tarefa 2)**

As seguintes consultas abordaram os requisitos do relatório de vendas:

* **2.1 Marca com Maior Volume de Vendas**:
  + A consulta identificou a marca com maior volume de vendas, como Fiat ou Volkswagen, dependendo dos dados analisados.
* **2.2 Maior e Menor Receita Gerada por Veículo**:
  + Utilizando a função de janela RANK(), a consulta destacou veículos com maior e menor receita, como Subaru Forester (maior) e Peugeot 307 (menor).
* **2.3 Faixas de Preço de Venda**:
  + A consulta gerou uma tabela mostrando a distribuição de vendas por faixas de preço, organizadas em intervalos de R$10.000, facilitando a análise de padrões de compra.
* **2.4 Receita das 3 Marcas com Menores Tickets Médios**:
  + A consulta identificou JaC Motors, Nissan e Chevrolet como as marcas com os menores tickets médios, fornecendo também a receita total gerada por cada uma.
* **2.5 Relação entre os 5 Veículos Mais Vendidos**:
  + A análise dos cinco veículos mais vendidos (Fiat Mobi, Volkswagen Up, Kia Picanto, Peugeot 208, Toyota Corolla) revelou:
    - **Tendências de Mercado**: Predominância de veículos compactos e econômicos, refletindo a preferência por modelos acessíveis no Brasil.
    - **Percepção do Cliente**: Marcas como Fiat e Volkswagen são valorizadas por confiabilidade e assistência técnica, enquanto Kia e Peugeot atraem por design moderno.
    - **Características dos Veículos**: Modelos como Fiat Mobi e Volkswagen Up são ideais para uso urbano, enquanto o Toyota Corolla destaca-se por durabilidade e valor de revenda.

### **3.3 Recomendações Estratégicas**

Uma consulta foi desenvolvida para fornecer recomendações estratégicas baseadas em volume de vendas, receita e meses ativos. Modelos como Fiat Mobi, Volkswagen Up e Kia Picanto foram classificados como "Modelo estrela - manter/maximizar" devido ao alto desempenho em vendas e receita. Modelos com alta receita, mas baixo volume, como Subaru XV, foram identificados como tendo potencial para aumento de vendas com estratégias de marketing direcionadas.

### **3.4 Exportação de Resultados**

Os dados corrigidos e análises adicionais foram exportados como arquivos JSON:

* dados\_corrigidos.json: Conjunto completo com vendas e marcas unidas.
* marcas\_corrigidas.json: Dados de marcas corrigidos.
* Análises adicionais: top\_modelos.json, mix\_produtos.json, elasticidade.json, recomendacoes.json.

## 

## 

## **4. Documentação**

A documentação foi estruturada para incluir:

1. **Códigos SQL**:
   * Consultas para correção de tabelas (realizadas via pré-processamento em Python).
   * Consultas para cada questão com explicações detalhadas no notebook.
2. **Arquivos Corrigidos**:
   * Links para os arquivos JSON corrigidos, exportados como dados\_corrigidos.json e marcas\_corrigidas.json.
3. **Observações Adicionais**:
   * O notebook apresenta seções claras com relatórios, análises estratégicas e recomendações acionáveis, enfatizando organização e clareza.

O documento foi configurado para compartilhamento no Google Drive com permissão de edição para qualquer pessoa com o link, conforme instruções.