☕ PROJETO COMPLETO DE ANÁLISE DE DADOS: VENDAS DE CAFETERIA COM MYSQL E POWER BI

📋 Introdução e Contexto

Este projeto detalha a implementação de um pipeline completo de engenharia e análise de dados para uma rede de cafeterias fictícia, a Maven Roasters, com operações em três localizações em Nova York. O objetivo é demonstrar a aplicação prática de conceitos de ***engenharia de dados***, desde a ingestão de dados brutos até a criação de um dashboard interativo profissional no Power BI.

O dataset Coffee Shop Sales (obtido do Kaggle) compreende 172.881 registros transacionais. Ele inclui informações detalhadas sobre data, hora, localização, produtos vendidos, quantidades e valores. Um aspecto notável do dataset é a presença de características regionais brasileiras, como formatos de data (DD/MM/YYYY) e decimais com vírgula, o que introduziu desafios realistas de ETL e enriqueceu a experiência de desenvolvimento.

🎯 Objetivo Principal

Construir uma base analítica robusta, aderente aos princípios de arquitetura de dados corporativa, para possibilitar análises estratégicas sobre performance de vendas, comportamento do consumidor e eficiência operacional das cafeterias. O projeto foca em uma ***análise de dados end-to-end***.

🛠️ Tecnologias Utilizadas

- MySQL: Utilizado para processos de limpeza, transformação e modelagem de dados, incluindo a criação de uma arquitetura em camadas (Staging e Analytics).
- Power BI: Empregado para modelagem de dados, desenvolvimento de medidas DAX e construção de storytelling através de dashboards interativos.

📊 Campos do Dataset

O dataset original contém os seguintes campos:

- **transaction_id:** Identificador único da transação.
- **transaction_date:** Data da compra.
- **transaction_time:** Horário da compra.
- **store_location:** Localização da loja onde a transação ocorreu.
- **product_category:** Categoria do produto vendido.

- **product_type:** Tipo específico do produto.
- **product_detail:** Detalhe ou variação do item vendido.
- **transaction_qty:** Quantidade de itens na transação.
- **unit_price:** Preço unitário do produto.
- **total_price:** Valor total da transação (quantidade * preço unitário).

🗁 **Dataset:** Coffee Shop Sales (Kaggle)

🏗 FASE 1: Arquitetura e Engenharia de Dados

Objetivo:

Estabelecer uma base sólida e organizada para todo o ***pipeline de dados***, seguindo as melhores práticas de engenharia de dados. Esta fase é crucial, pois define a estrutura que sustentará todas as análises futuras, garantindo escalabilidade, manutenibilidade e confiabilidade dos dados.

Importância:

Uma arquitetura bem planejada previne problemas de performance, dificuldades de manutenção e falta de confiabilidade nos resultados analíticos.

1.1. VISÃO GERAL DA ARQUITETURA (FLUXO DE DADOS)

O fluxo de dados do projeto segue uma arquitetura em camadas:

CSV (Fonte Original) →

STAGING (raw_coffee_sales) →

CLEAN (clean_coffee_sales) →

MODELO ANALÍTICO (fact_coffee_sales + dims) →

VIEWS (v_vendas_completa, v_kpis_mensais, etc.) →

POWER BI (dashboards)

1.2. CRIAÇÃO DOS BANCOS DE DADOS

Foram criados dois bancos de dados distintos para segregar as camadas de ***Staging*** e ***Analytics***:

```sql
-- Camada 1: Staging (dados brutos)
CREATE DATABASE coffee_shop_staging;

-- Camada 2: Analytics (dados tratados e modelados)
CREATE DATABASE coffee_shop_analytics;
```

🔄 FASE 2: Ingestão e Transformação de Dados

Objetivo:

Garantir a qualidade e integridade dos dados através de processos robustos de **ETL** (Extract, Transform, Load). Esta fase converte dados brutos e potencialmente inconsistentes em informações confiáveis para análise.

Importância:

Dados de baixa qualidade podem levar a análises enganosas e decisões empresariais equivocadas. Um processo de ETL bem implementado é a base para qualquer projeto de analytics bem-sucedido.

2.1 . PROBLEMAS IDENTIFICADOS NO CSV ORIGINAL

O arquivo CSV original apresentava as seguintes características que exigiram tratamento específico:

- Separador: Ponto-e-vírgula (;).
- Decimais: Vírgula (,) como separador decimal (ex: 3,1).
- Datas: Formato brasileiro (DD/MM/YYYY).
- Encoding: UTF-8 com potenciais caracteres especiais.

2.2 . PROCESSO DE INGESTÃO EM DUAS ETAPAS (PHPMYADMIN)

A ingestão dos dados brutos na tabela **raw_coffee_sales** foi realizada em duas etapas via phpMyAdmin para garantir a correta interpretação do arquivo CSV:

ETAPA 1A - Criar Estrutura (Importar no BANCO):

-- Configuração no phpMyAdmin para criar a estrutura da tabela:

-- Formato: CSV

-- Separador: ;

-- Delimitador: "

-- "Primeira linha contém nomes" - MARCAR

-- Nome da tabela: **raw_coffee_sales**

ETAPA 1B - Popular Dados (Importar na TABELA):

-- Configuração no phpMyAdmin para importar os dados na tabela existente:

-- Formato: CSV using LOAD DATA

-- Separador: ;

-- Delimitador: "

-- Campos divididos: [EM BRANCO]

-- "Primeira linha contém nomes" - DESMARCAR

-- "Ignorar primeiras linhas: 1" – MARCAR

-- Nomes das colunas: (especificar manualmente)

2.3 . SCRIPT DE TRANSFORMAÇÃO PROFISSIONAL (CLEAN LAYER)

Após a ingestão, os dados foram transformados e carregados em uma tabela **clean_coffee_sales** apartir dessa tabela foi criada a tabela fato ficando essa como uma intermediária.

Criação da Tabela clean_coffee-sales

```
-- Criar tabela clean_coffee_sales com estrutura otimizada na camada Analytics
USE coffee_shop_analytics;

CREATE TABLE clean_coffee_sales (
    transaction_id INT,
    transaction_date DATE,
    transaction_time TIME,
    transaction_qty INT,
    store_id INT,
    store_location VARCHAR(100),
    product_id INT,
    unit_price DECIMAL(10,2),
    product_category VARCHAR(50),
    product_type VARCHAR(100),
    product_detail VARCHAR(200)
);
```

Inserção & Tratamento de Dados para a nova tabela

```sql
-- ETL: Extrair, Transformar, Carregar dados da camada Staging para a Clean
INSERT INTO clean_coffee_sales
SELECT
    CAST(transaction_id AS UNSIGNED) AS transaction_id,
    STR_TO_DATE(transaction_date, '%d/%m/%Y') AS transaction_date,
    CAST(transaction_time AS TIME) AS transaction_time,
    transaction_qty,
    store_id,
    store_location,
    product_id,
    CAST(REPLACE(unit_price, ',', '.') AS DECIMAL(10,2)) AS unit_price,
    product_category,
    product_type,
    product_detail
FROM coffee_shop_staging.raw_coffee_sales
WHERE transaction_id REGEXP '^[0-9]+$'; -- Filtra linhas com IDs válidos
```

2.4 . LIMPEZA E QUALIDADE DE DADOS (CLEAN LAYER)

Foram aplicadas regras de limpeza para remover registros inconsistentes ou incompletos na tabela **clean_coffee_sales:**

```sql
-- Remover possíveis linhas problemáticas (valores nulos ou zeros inválidos)
DELETE FROM coffee_shop_analytics.clean_coffee_sales
WHERE transaction_date IS NULL
   OR store_id = 0
   OR product_id = 0
   OR transaction_qty = 0
   OR unit_price = 0.00;

-- Verificar contagem final de registros válidos após a limpeza
SELECT COUNT(*) as registros_validos
FROM coffee_shop_analytics.clean_coffee_sales;
```

📊 FASE 3: Modelagem Dimensional (Analytics Layer)

Objetivo:

Organizar os dados em uma estrutura otimizada para análise e consultas, seguindo o padrão de modelagem estrela. Esta fase transforma dados transacionais em um modelo dimensional que facilita agregações e análises complexas.

Importância:

Um bom modelo dimensional melhora drasticamente a performance das consultas, simplifica a criação de relatórios e torna o ***data warehouse*** mais intuitivo para usuários de negócio.

3.1 CRIAÇÃO DAS TABELAS DIMENSÃO

As tabelas de dimensão foram criadas a partir da **clean_coffee_sales** para armazenar atributos descritivos de *produtos, lojas e tempo*:

DIM_PRODUCTS (Dimensão de Produtos)

```
CREATE TABLE dim_products AS
SELECT DISTINCT
    product_id,
    product_category,
    product_type,
    product_detail
FROM clean_coffee_sales;

ALTER TABLE dim_products ADD PRIMARY KEY (product_id);
```

DIM_STORES (Dimensão de Lojas)

```
CREATE TABLE dim_stores AS
SELECT DISTINCT
    store_id,
    store_location
FROM clean_coffee_sales;

ALTER TABLE dim_stores ADD PRIMARY KEY (store_id);
```

DIM_TIME (Dimensão de Tempo com Ordenação)

```
CREATE TABLE dim_time AS
WITH recursive date_series AS (
    SELECT MIN(transaction_date) as date_val FROM clean_coffee_sales
    UNION ALL
    SELECT date_val + INTERVAL 1 DAY
    FROM date_series
    WHERE date_val < (SELECT MAX(transaction_date) FROM clean_coffee_sales)
)
SELECT
    date_val as date_id,
    YEAR(date_val) as ano,
    MONTH(date_val) as mes_numero,
    DATE_FORMAT(date_val, '%M') as mes_nome,
    DAY(date_val) as dia,
    DAYNAME(date_val) as dia_semana,
```

Criação de Ordenação na Dimensão de Tempo

```
    -- Coluna para ordenação correta dos dias da semana no BI
    CASE
        WHEN DAYNAME(date_val) = 'Monday' THEN 1
        WHEN DAYNAME(date_val) = 'Tuesday' THEN 2
        WHEN DAYNAME(date_val) = 'Wednesday' THEN 3
        WHEN DAYNAME(date_val) = 'Thursday' THEN 4
        WHEN DAYNAME(date_val) = 'Friday' THEN 5
        WHEN DAYNAME(date_val) = 'Saturday' THEN 6
        WHEN DAYNAME(date_val) = 'Sunday' THEN 7
    END as ordem_semana,
    QUARTER(date_val) as trimestre,
    CASE WHEN DAYOFWEEK(date_val) IN (1,7) THEN 1 ELSE 0 END as eh_fim_semana
FROM date_series;

ALTER TABLE dim_time ADD PRIMARY KEY (date_id);
```

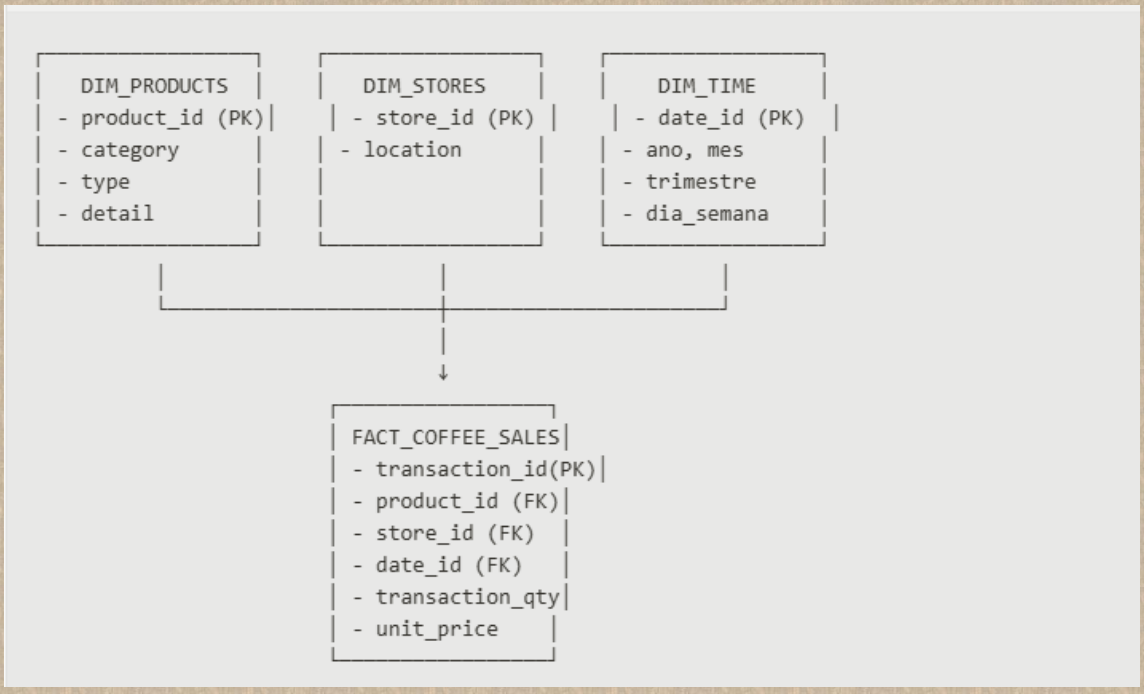
3.2 TABELA FATO OTIMIZADA (FACT_COFFEE_SALES)

A tabela fato foi criada para conter as chaves estrangeiras para as dimensões e as medidas numéricas das transações, otimizada para cálculos no Power BI.

```
CREATE TABLE fact_coffee_sales AS
SELECT
    transaction_id,
    transaction_date AS date_id, -- Renomeado para facilitar o join com dim_time
    store_id,
    product_id,
    transaction_qty,
    unit_price
FROM clean_coffee_sales;

-- Adicionar chaves primárias e estrangeiras (exemplo, dependendo da necessidade)
ALTER TABLE fact_coffee_sales ADD PRIMARY KEY (transaction_id);
ALTER TABLE fact_coffee_sales ADD FOREIGN KEY (date_id) REFERENCES dim_time(date_id);
ALTER TABLE fact_coffee_sales ADD FOREIGN KEY (store_id) REFERENCES
dim_stores(store_id);
ALTER TABLE fact_coffee_sales ADD FOREIGN KEY (product_id) REFERENCES
dim_products(product_id);
```

3.3 DIAGRAMA DO MODELO ESTRELA



🔍 FASE 4: Views Analíticas

Objetivo:

Criar camadas de abstração que simplifiquem o consumo dos dados pelo Power BI e outros clientes analíticos. As *views* encapsulam a complexidade dos *joins* e *agregações*, fornecendo dados prontos para análise.

Importância:

Views bem desenhadas aumentam a produtividade dos analistas, reduzem erros em consultas complexas e centralizam a lógica de negócio no banco de dados.

4.1 VIEW MESTRA - VENDAS COMPLETA (V_VENDAS_COMPLETA)

Esta view une todas as tabelas de fato e dimensão, fornecendo uma base completa e desnormalizada para análises detalhadas no Power BI.

```sql
CREATE OR REPLACE VIEW v_vendas_completa AS
SELECT
    f.transaction_id,
    f.date_id AS transaction_date, -- Usando date_id para consistência
    f.transaction_qty,
    f.unit_price,
    (f.transaction_qty * f.unit_price) as total_venda,

    -- Dimensão Produtos
    p.product_id,
    p.product_category,
    p.product_type,
    p.product_detail,

    -- Dimensão Lojas
    s.store_id,
    s.store_location,

    -- Dimensão Tempo
    t.ano,
    t.mes_numero,
    t.mes_nome,
    t.trimestre,
    t.dia,
    t.dia_semana,
    t.ordem_semana,
    t.eh_fim_semana
FROM fact_coffee_sales f
JOIN dim_products p ON f.product_id = p.product_id
JOIN dim_stores s ON f.store_id = s.store_id
JOIN dim_time t ON f.date_id = t.date_id;
```

4.2 VIEW DE KPIS MENSAIS (V_KPIS_MENSAIS)

Esta view pré-agrega dados em nível mensal, ideal para ***análises de tendências*** e ***variação mês a mês***.

```sql
CREATE VIEW v_kpis_mensais AS
SELECT
    DATE_FORMAT(t.date_id, '%Y-%m') as mes_ano,
    t.ano,
    t.mes_nome,
    t.mes_numero,
    t.trimestre,
    SUM(f.transaction_qty * f.unit_price) as vendas_totais,
    COUNT(DISTINCT f.transaction_id) as total_pedidos,
    SUM(f.transaction_qty) as quantidade_vendida,
    LAG(SUM(f.transaction_qty * f.unit_price)) OVER (ORDER BY t.ano, t.mes_numero) as
vendas_mes_anterior
FROM fact_coffee_sales f
JOIN dim_time t ON f.date_id = t.date_id
GROUP BY DATE_FORMAT(t.date_id, '%Y-%m'), t.ano, t.mes_nome, t.mes_numero, t.trimestre;
```

4.3 VIEW DE TOP PRODUTOS (V_TOP_PRODUTOS)

Esta view classifica os produtos com base nas vendas, útil para identificar os itens de maior e menor desempenho.

```
CREATE VIEW v_top_produtos AS
SELECT
    p.product_category,
    p.product_type,
    p.product_detail,
    SUM(f.transaction_qty * f.unit_price) as vendas_totais,
    SUM(f.transaction_qty) as quantidade_vendida,
    COUNT(f.transaction_id) as total_pedidos,
    RANK() OVER (ORDER BY SUM(f.transaction_qty * f.unit_price) DESC) as ranking_geral
FROM fact_coffee_sales f
JOIN dim_products p ON f.product_id = p.product_id
GROUP BY p.product_category, p.product_type, p.product_detail;
```

🖌️ FASE 5: Conexão do Banco de Dados com Power BI

Objetivo:

Estabelecer uma conexão segura e eficiente entre o MySQL e o Power BI, permitindo a atualização dinâmica dos dashboards com os dados processados no banco de dados.

Importância:

Esta fase é crucial para transformar os dados estruturados em insights visuais interativos, conectando a infraestrutura de dados backend com a interface de análise frontend.

5.1 CONFIGURAÇÃO DA CONEXÃO MYSQL

ETAPA 1: Preparação do Ambiente

- **Driver MySQL:** Instalação do MySQL Connector/NET (versão 8.0 ou superior)
- **Power BI Desktop:** Versão mais recente para compatibilidade
- **Credenciais:** Usuário e senha com permissões de leitura no ***banco coffee_shop_analytics***

ETAPA 2: Conexão no Power BI

No Power BI: Menu “Obter Dados”> “Mais” > “Banco de Dados MySQL”

Servidor : 127.0.0.1

Banco de Dados: coffee_shop_analytics

Usuário: root

Senha: Em Branco

☑ FASE 6: Power BI e Medidas DAX

Objetivo:

Transformar dados em insights acionáveis através de visualizações interativas e medidas de negócio calculadas dinamicamente. Esta fase conecta a base técnica à tomada de decisão empresarial.

Importância:

Um dashboard bem construído comunica eficientemente informações complexas, permite exploração interativa dos dados e suporta decisões baseadas em evidências.

6.1 ESTRUTURA DE MEDIDAS (TABELA 00_MEDIDAS)

Todas as medidas DAX são centralizadas em ***uma tabela dedicada*** **(00_Medidas)** para organização e fácil acesso.

Construir uma tabela de medidas no Power BI, em vez de manter ***colunas calculadas*** no SQL, é uma boa prática porque centraliza a lógica de negócio na **camada de visualização**, onde ela pode ser dinamicamente recalculada com base no contexto do relatório.

Enquanto colunas calculadas no SQL são estáticas e pré-agregadas, as medidas DAX no Power BI se adaptam automaticamente aos filtros, segmentações e hierarquias aplicadas pelo usuário, permitindo análises interativas e multidimensionais.

Além disso, essa abordagem reduz a carga no banco de dados, já que os cálculos são processados na memória do Power BI, e facilita a manutenção, pois as regras de negócio ficam concentradas no arquivo .pbix, separadas da camada de dados.

Criar a tabela dedicada de Medidas no Power BI

Página Inicial >Inserir Dados> Nomeie a Tabela: “00_Medidas”.

Medidas Base

```
Total Vendas =
SUMX(
    'v_vendas_completa',
    'v_vendas_completa'[transaction_qty] *
    'v_vendas_completa'[unit_price]
)

Total Pedidos =
DISTINCTCOUNT('v_vendas_completa'[transaction_id])

Ticket Médio =
DIVIDE([Total Vendas], [Total Pedidos], 0)

Total Itens = SUM('v_vendas_completa'[transaction_qty])
```

Medidas Avançadas

a) Produtos Mais Vendidos

```
Produto Mais Vendido =
VAR TopProduto =
    TOPN(
        1,
        SUMMARIZE(
            ALL('dim_products'),
            'dim_products'[product_type],
            "VendasProduto", [Total Vendas]
        ),
        [VendasProduto],
        DESC
    )
VAR NomeProduto = MAXX(TopProduto, 'dim_products'[product_type])
VAR ValorVendas = MAXX(TopProduto, [VendasProduto])
RETURN
    NomeProduto & " - " & FORMAT(ValorVendas, "$ #,##0")
```

b) Produtos Menos Vendidos

```
Produto Menos Vendido =
VAR ProdutosComVenda =
    FILTER(
        SUMMARIZE(
            ALL('dim_products'),
            'dim_products'[product_type],
            "VendasProduto", [Total Vendas]
        ),
        [VendasProduto] > 0
    )
VAR BottomProduto =
    TOPN(
        1,
        ProdutosComVenda,
        [VendasProduto],
        ASC
    )
VAR NomeProduto = MAXX(BottomProduto, 'dim_products'[product_type])
VAR ValorVendas = MAXX(BottomProduto, [VendasProduto])
RETURN
    NomeProduto & " - " & FORMAT(ValorVendas, "$ #,##0")
```

c) Vendas Dias Úteis e Finais de Semana

```
Vendas Dias Úteis =
CALCULATE([Total Vendas], 'dim_time'[eh_fim_semana] = 0)

Vendas Fim Semana =
CALCULATE([Total Vendas], 'dim_time'[eh_fim_semana] = 1)
```

5.2 ARQUITETURA DO DASHBOARD

O dashboard final no Power BI é estruturado em abas, cada uma focada em um aspecto específico da análise de vendas:

5.3 . DIAGRAMA DE ARQUITETURA COMPLETA

O diagrama abaixo representa um pipeline completo de engenharia de dados que implementa o padrão em camadas, evoluindo dos dados brutos em CSV até um dashboard analítico final.

```
[1] Dados Brutos (CSV)
    ↓
[2] Staging Layer (raw_coffee_sales) - Dados brutos, cópia fiel da fonte
    ↓
[3] Clean Layer (clean_coffee_sales) - Dados limpos, padronizados e transformados
    ↓
[4] Analytics Layer (Modelo Estrela)
    ├─ fact_coffee_sales (tabela fato com medidas)
    ├─ dim_products (dimensão de produtos)
    ├─ dim_stores (dimensão de lojas)
    └─ dim_time (dimensão de tempo)
    ↓
[5] Views Analíticas (6 views otimizadas para consumo)
    ↓
[6] Power BI (Modelo de dados + Medidas DAX)
    ↓
[7] Dashboard Final (Distribuído em 4 abas estratégicas)
```

A arquitetura segue fielmente a metodologia Kimball* através de um **esquema estrela** na camada Analytics, composto por uma tabela fato (fact_coffee_sales) que armazena as métricas de negócio atômicas e três tabelas dimensão (produtos, lojas e tempo) que fornecem o contexto descritivo.

Esta estrutura transforma dados transacionais complexos em informação estratégica consumível, permitindo análises multidimensionais intuitivas, performance otimizada para consultas e uma base sólida para tomada de decisão baseada em dados.

*KIMBALL, R.; ROSS, M. The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling. 3. ed. Indianapolis: John Wiley & Sons, 2013.

💡 FASE 6: Lições Aprendidas e Insights

Objetivo:

Consolidar o conhecimento adquirido durante o projeto e extrair valor de negócio tangível dos dados processados. Esta fase transforma experiência técnica em inteligência estratégica.

Importância:

A reflexão sobre o processo e resultados garante que o aprendizado seja internalizado e aplicado em projetos futuros, enquanto os insights gerados justificam o investimento em analytics.

6.1 INSIGHTS DE NEGÓCIO IDENTIFICADOS

Visão Geral

Performance Sólida com Oportunidades Claras:

Receita total de $809K com um ticket médio saudável de $5,43. Foram processados 149K pedidos, com uma densidade média de 1,66 itens por pedido. Junho se destacou como o melhor mês, e as segundas-feiras apresentaram as vendas mais fortes.

Visão Temporal

Crescimento Consistente com Desequilíbrio Temporal:

O segundo trimestre apresentou performance máxima. Um gap crítico identificado é a disparidade entre vendas em dias úteis (72%) e fins de semana (28%), indicando uma diferença de 9x. A meta é reduzir a dependência de dias úteis e explorar o potencial dos fins de semana.

Visão de Produtos

Portfólio Diversificado com Alta Concentração:

O portfólio inclui 81 produtos em 10 categorias. O ***Barista Espresso*** lidera com $106K, representando 13% da receita. Os três produtos mais vendidos (Top 3) respondem por 34,5% da receita total. A estratégia sugerida é diversificar com base nos sucessos e revisar o desempenho de produtos com baixa performance.

Visão de Lojas

Equilíbrio Excepcional com Perfis Distintos:

Há uma diferença mínima de apenas 1,06% na performance entre as lojas. Hell's Kitchen lidera com $274,5K, caracterizando-se por um perfil corporativo. Lower Manhattan se destaca com um ticket médio 21% mais alto. Oportunidades incluem a customização de estratégias por perfil local e a transferência de melhores práticas entre as unidades.

Síntese Estratégica

O negócio, embora saudável, possui três focos principais para otimização:

1. Diversificar a receita, reduzindo a dependência de produtos âncora.
2. Explorar o potencial não aproveitado nos fins de semana.
3. Customizar estratégias por loja, mantendo a excelência operacional.

6.2 Lições Técnicas Aprendidas

1. **Arquitetura em Camadas:** A implementação das camadas ***Staging, Clean*** e ***Analytics*** é fundamental para a organização e escalabilidade do pipeline de dados.
2. **Modelagem Dimensional:** A adoção do esquema estrela otimiza a performance das consultas e a clareza para análises de BI.
3. **Qualidade de Dados:** A necessidade de validação e tratamento rigoroso dos dados brutos para garantir a confiabilidade das análises.

4. **Separação de Responsabilidades:** A distinção clara entre o uso de SQL para ETL e DAX para cálculos analíticos no Power BI é crucial para flexibilidade e manutenção.
5. **Documentação:** A importância de documentar processos e decisões para garantir reprodutibilidade e versionamento do projeto.

🗀 Estrutura para Portfólio

1 FONTE ORIGINAL

- **CSV:** coffee_shop_sales.csv
- **Descrição:** Dados brutos extraídos do Kaggle contendo as vendas da cafeteria.

2 CAMADA DE STAGING (BANCO: COFFEE_SHOP_STAGING)

- **Tabela:** raw_coffee_sales
- **Descrição:** Cópia fiel do CSV, sem transformações. Serve para manter o registro histórico original (Golden Source).
- **Função:** Armazenar os dados brutos importados do CSV e garantir reprodutibilidade e integridade dos dados.

3 Camada de Analytics (Banco: coffee_shop_analytics)

- Esta camada é onde ocorre o processamento, limpeza e modelagem dos dados, estruturada em tabelas dimensionais, fato e views analíticas.

📦 Tabelas Principais

Tabela	Tipo	Função
clean_coffee_sales	Intermediária	Dados tratados e padronizados (limpeza, trim, upper, cast de datas, etc.)
fact_coffee_sales	Fato	Contém as medidas principais (vendas, quantidade, preço, etc.)
dim_products	Dimensão	Catálogo de produtos, categorias e tipos
dim_stores	Dimensão	Informações das lojas e localizações
dim_time	Dimensão	Datas, dias da semana, meses, trimestres e anos

🔍 Views Analíticas

Estas views servem como o elo entre o banco de dados e o Power BI, fornecendo dados pré-processados e otimizados para visualização.

View	Finalidade
v_vendas_completa	Tabela base com todas as vendas (para gráficos gerais)
v_kpis_mensais	Métricas mensais e variação Mês a Mês (MoM)
v_performance_agregada	Consolidação de performance geral (por loja, categoria, etc.)
v_top_produtos	Ranking e análise de produtos mais vendidos

4 Camada de Visualização (Power BI)

- Fonte de Dados: coffee_shop_analytics → Views analíticas (v_*)
- Ações Realizadas:
- Criação da tabela 00_Medidas com DAX (KPIs e métricas derivadas).
- Modelagem relacional entre tabelas de fato e dimensão.
- Construção das abas do dashboard:
    - Visão Geral → Performance macro do negócio
    - Visão Temporal → Comportamento ao longo do tempo
    - Visão Produtos → O que está vendendo e porquê
    - Visão Lojas → Onde estão os melhores resultados