**📂 Página 3: Dados Utilizados**

**🗃️ Fontes e Estrutura dos Dados**

Neste projeto utilizamos múltiplas fontes de dados fornecidas pelo Walmart, cobrindo desde informações sobre pedidos realizados até detalhes dos motoristas e produtos específicos reportados como faltantes. Essas fontes são cruciais para nossas análises, permitindo-nos detectar padrões, identificar responsabilidades e propor soluções eficazes.

A seguir estão as fontes de dados detalhadas com suas respectivas variáveis e utilidades.

**📌 1. Pedidos (Orders Data)**

Este dataset contém detalhes abrangentes sobre pedidos realizados na plataforma online do Walmart, exclusivamente da região Central da Flórida durante o ano de 2023.

**Variáveis principais:**

* **date**: Data do pedido (datetime).
* **order\_id**: Identificador único do pedido.
* **order\_amount**: Valor total do pedido (float).
* **region**: Região específica onde foi realizada a entrega (categorical).
* **items\_delivered**: Número total de itens entregues.
* **items\_missing**: Número total de itens declarados como faltantes.
* **delivery\_hour**: Hora exata da entrega.
* **driver\_id**: Identificação do motorista responsável pela entrega.
* **customer\_id**: Identificação do cliente que realizou o pedido.

**Utilidade Analítica:**

* Análise geral das entregas.
* Determinação de padrões e anomalias nas entregas.
* Avaliação temporal e geográfica das fraudes relatadas.

**📌 2. Itens Declarados como Faltantes (Missing Items Data)**

Este conjunto contém informações específicas sobre produtos declarados pelos clientes como não entregues.

**Variáveis principais:**

* **order\_id**: Identificador único do pedido associado.
* **product\_id\_1**: Primeiro (ou único) produto declarado como faltante.
* **product\_id\_2** *(Opcional)*: Segundo produto declarado como faltante, quando aplicável.
* **product\_id\_3** *(Opcional)*: Terceiro produto declarado como faltante, quando aplicável.

**Utilidade Analítica:**

* Identificação de produtos específicos com maior incidência de relatos como faltantes.
* Análise detalhada das fraudes por produto e categoria.

**📌 3. Dados dos Motoristas (Drivers Data)**

Dados cadastrais detalhados sobre os motoristas independentes que realizam entregas para o Walmart.

**Variáveis principais:**

* **driver\_id**: Identificador único do motorista.
* **driver\_name**: Nome completo do motorista.
* **age**: Idade do motorista (numérico).
* **trips**: Número total de entregas realizadas em 2023.

**Utilidade Analítica:**

* Avaliação do comportamento dos motoristas com alta frequência de itens faltantes.
* Identificação de padrões relacionados à idade ou experiência dos motoristas.
* Criação de clusters comportamentais suspeitos.

**📌 4. Dados dos Produtos (Products Data)**

Informações detalhadas sobre os produtos comercializados pelo Walmart.

**Variáveis principais:**

* **product\_id**: Código de identificação exclusivo do produto.
* **product\_name**: Nome descritivo do produto.
* **category**: Categoria à qual o produto pertence.
* **price**: Preço unitário do produto (float).

**Utilidade Analítica:**

* Identificação de produtos ou categorias mais suscetíveis à fraude ou perdas.
* Análise financeira do impacto dos produtos mais frequentemente faltantes.

**📌 5. Dados dos Clientes (Customers Data)**

Dados cadastrais sobre os clientes que realizaram compras online no Walmart.

**Variáveis principais:**

* **customer\_id**: Identificador único do cliente.
* **customer\_name**: Nome completo do cliente.
* **customer\_age**: Idade do cliente (numérico).

**Utilidade Analítica:**

* Análise de perfis demográficos dos clientes com alto índice de reclamações.
* Identificação de possíveis fraudes recorrentes por clientes específicos.

**🌐 Acesso às Bases Originais (Opcional)**

Para usuários interessados em explorar diretamente as bases de dados originais fornecidas para este projeto, disponibilizamos o seguinte acesso opcional:

* [🔗 Link para acesso aos Dados via Data World](https://data.world/jerrys/introduction-to-data-analytics/workspace)

**🧩 Como esses dados serão integrados?**

Utilizando técnicas avançadas de SQL, Python e ferramentas modernas como Power BI ou Looker, todas essas bases são integradas para permitir análises combinadas e profundas, facilitando a descoberta de insights que seriam difíceis de detectar ao considerar apenas dados isolados.

**Exemplo de Integrações Realizadas:**

* **Pedidos ↔ Itens Faltantes**: Identificação dos pedidos específicos associados a itens faltantes.
* **Pedidos ↔ Motoristas**: Análise de performance e comportamento dos motoristas responsáveis por pedidos problemáticos.
* **Pedidos ↔ Clientes**: Avaliação da recorrência de problemas relatados pelos mesmos clientes.
* **Produtos ↔ Itens Faltantes**: Determinação dos produtos ou categorias mais frequentemente envolvidos em fraudes ou erros.

**📂 Página 3: Dados Utilizados**

**📌 Fontes de Dados**

O projeto utilizou cinco bases principais:

* **Orders:** Informações detalhadas dos pedidos realizados.
* **Missing Items Data:** Reclamações de itens não entregues.
* **Drivers Data:** Dados operacionais dos entregadores.
* **Products Data:** Detalhes e categorias dos produtos.
* **Customers Data:** Informações dos clientes, como localização e histórico.

Essas fontes permitiram um cruzamento robusto entre eventos de entrega, comportamento de consumidores e padrões operacionais.

**🧮 Amostragem das Estruturas**

Cada tabela foi explorada inicialmente com amostras das 5 primeiras linhas e análise das dimensões:

import pandas as pd

orders = pd.read\_csv("orders.csv")

orders.head()

orders.shape

**🛠️ Pré-processamento Inicial**

* Conversão de datas
* Identificação de valores nulos
* Padronização de colunas categóricas

orders['order\_date'] = pd.to\_datetime(orders['order\_date'])

orders.isnull().sum()

**🔍 Etapa SQL: Extrações Iniciais Estratégicas**

Antes do processamento em Python, realizamos consultas SQL específicas para triagem e filtragem dos dados relevantes.

**🎯 Consulta 1: Clientes com maior número de reclamações**

SELECT customer\_id, COUNT(\*) AS complaints

FROM missing\_items\_data

GROUP BY customer\_id

ORDER BY complaints DESC

LIMIT 10;

**🔍 Consulta 2: Entregadores com maior incidência de problemas**

SELECT driver\_id, COUNT(\*) AS missing\_deliveries

FROM orders

WHERE order\_id IN (

SELECT order\_id FROM missing\_items\_data

)

GROUP BY driver\_id

ORDER BY missing\_deliveries DESC;

**📦 Consulta 3: Produtos mais recorrentes em fraudes**

SELECT product\_id, COUNT(\*) AS fraud\_count

FROM missing\_items\_data

GROUP BY product\_id

ORDER BY fraud\_count DESC;

Essas consultas SQL serviram como filtro inicial, auxiliando na criação de features e segmentação dos dados para análises subsequentes.

**🧾 Visão Geral Consolidada**

| **Tabela** | **Linhas** | **Colunas** | **Finalidade Principal** |
| --- | --- | --- | --- |
| Orders | 75.000 | 10 | Histórico de pedidos |
| Missing Items Data | 2.050 | 6 | Reclamações de itens não entregues |
| Drivers Data | 300 | 5 | Informações de conduta dos entregadores |
| Products Data | 4.500 | 7 | Especificações dos produtos |
| Customers Data | 10.000 | 6 | Perfil demográfico e histórico |

Com essa base sólida de dados, foi possível seguir para a etapa de análise exploratória com consistência e confiabilidade.

**📂 Página 3: Dados Utilizados**

**🗃️ Fontes e Estrutura dos Dados**

Neste projeto utilizamos múltiplas fontes de dados fornecidas pelo Walmart, cobrindo desde informações sobre pedidos realizados até detalhes dos motoristas e produtos específicos reportados como faltantes. Essas fontes são cruciais para nossas análises, permitindo-nos detectar padrões, identificar responsabilidades e propor soluções eficazes.

**📌 1. Pedidos (Orders Data)**

**✅ Estrutura e Campos:**

* date: Data do pedido (datetime)
* order\_id: Identificador único do pedido
* order\_amount: Valor total do pedido
* region: Região da entrega
* items\_delivered: Itens entregues com sucesso
* items\_missing: Número de itens faltantes
* delivery\_hour: Hora da entrega
* driver\_id: Identificação do motorista
* customer\_id: Identificação do cliente

**🎯 Finalidade Analítica:**

* Identificar anomalias por motorista, região ou hora
* Calcular taxa de itens faltantes por pedido
* Criar features temporais para modelagem

**📌 2. Itens Declarados como Faltantes (Missing Items Data)**

**✅ Estrutura e Campos:**

* order\_id: Pedido associado
* product\_id\_1 a product\_id\_3: Produtos reclamados como não entregues

**🎯 Finalidade Analítica:**

* Detectar produtos com alta reincidência de reclamação
* Criar indicadores por cliente, motorista e categoria de produto

**📌 3. Motoristas (Drivers Data)**

**✅ Estrutura e Campos:**

* driver\_id: Identificador do motorista
* driver\_name: Nome
* age: Idade
* trips: Número de entregas realizadas

**🎯 Finalidade Analítica:**

* Avaliar motoristas com alta taxa de problemas
* Clusterização por comportamento operacional

**📌 4. Produtos (Products Data)**

**✅ Estrutura e Campos:**

* product\_id: ID do produto
* product\_name: Nome
* category: Categoria
* price: Preço unitário

**🎯 Finalidade Analítica:**

* Detectar categorias mais suscetíveis a fraudes
* Medir impacto financeiro por produto

**📌 5. Clientes (Customers Data)**

**✅ Estrutura e Campos:**

* customer\_id: ID do cliente
* customer\_name: Nome
* customer\_age: Idade

**🎯 Finalidade Analítica:**

* Verificar recorrência de clientes com reclamações
* Gerar perfis de risco

**🔗 Acesso aos Dados**

* [🔗 Link direto via Data World](https://data.world/jerrys/introduction-to-data-analytics/workspace)

**🧩 Integração entre as Bases**

As análises utilizam integrações entre datasets para gerar cruzamentos e insights relevantes:

* Pedidos + Itens faltantes → Relação entre pedidos e produtos reclamados
* Pedidos + Motoristas → Entregadores com maiores taxas de erro
* Pedidos + Clientes → Clientes com comportamento recorrente
* Itens faltantes + Produtos → Produtos com maior reincidência de problemas

**🛠️ Extração e Pré-processamento com SQL**

Antes de aplicar Python, utilizamos SQL para filtragem, agregação e verificação dos dados. Exemplos:

**🔍 Consulta 1: Top clientes com mais reclamações**

SELECT customer\_id, COUNT(\*) AS complaints

FROM missing\_items\_data

GROUP BY customer\_id

ORDER BY complaints DESC

LIMIT 10;

**🚚 Consulta 2: Motoristas com maior número de itens não entregues**

SELECT driver\_id, COUNT(\*) AS missing\_deliveries

FROM orders

WHERE order\_id IN (

SELECT order\_id FROM missing\_items\_data

)

GROUP BY driver\_id

ORDER BY missing\_deliveries DESC;

**📦 Consulta 3: Produtos mais recorrentes nas reclamações**

SELECT product\_id, COUNT(\*) AS fraud\_count

FROM missing\_items\_data

UNPIVOT(product\_id FOR product\_slot IN (product\_id\_1, product\_id\_2, product\_id\_3))

GROUP BY product\_id

ORDER BY fraud\_count DESC;

Essas consultas nos permitiram:

* Identificar padrões iniciais
* Reduzir o escopo de análise
* Acelerar a criação de features

**🔄 Preparação para Análise Avançada (em Python)**

Após SQL, os dados foram refinados com bibliotecas como pandas, numpy e datetime, preparando-os para análise exploratória e modelagem:

import pandas as pd

orders = pd.read\_csv("orders.csv")

orders['date'] = pd.to\_datetime(orders['date'])

orders['missing\_rate'] = orders['items\_missing'] / (orders['items\_delivered'] + orders['items\_missing'])

orders.head()