Interface gráfica do usuário, Texto, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

### 1. Fontes de Dados (Extração)

A fonte primária utilizada para este PoC é o padrão \*\*GTFS\*\* (\*General Transit Feed Specification\*), que fornece dados de rotas e horários. Para fins de demonstração, os dados foram simulados.

\* \*\*Link de Origem (Exemplo):\*\* [Acesse o Portal de Dados Abertos da SPTrans Aqui]

### 2. Definição das Tecnologias

| Etapa | Tecnologia | Função Detalhada |

| :--- | :--- | :--- |

| \*\*ETL (Extração/Transformação)\*\* | \*\*Python com Pandas\*\* | Utilizado para limpeza, cruzamento de dados e cálculo da métrica de pontualidade. |

| \*\*Carga (L) (DDL/DML)\*\* | \*\*Arquivos CSV no GitHub\*\* | O `dados\_tratados.csv` serve como Data Warehouse simplificado (DDL: estrutura, DML: dados). |

| \*\*OLAP / Visualização\*\* | \*\*Power BI Desktop\*\* | Ferramenta visual para análises multidimensionais (\*\*OLAP\*\*) e criação de \*dashboards\*. |

### 3. Detalhamento Técnico (Divisão de 6 Membros)

| Membro | Etapa Principal | Contribuição e Lógica |

| :--- | :--- | :--- |

| \*\*Membro 1\*\* (Gerente) | Gestão e Estrutura | Criação do repositório e inclusão dos dados simulados. |

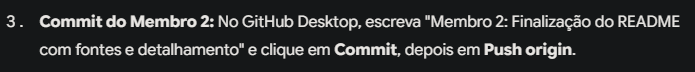
| \*\*Membro 2\*\* (Extração) | Documentação e Fonte | Finaliza o `README.md` e aponta a origem da extração. |

| \*\*Membro 3\*\* (T - Limpeza) | Transformação (T) - Limpeza | Código Python para ler dados e garantir a qualidade (remover nulos/duplicatas). |

| \*\*Membro 4\*\* (T - Métricas) | Transformação (T) - Enriquecimento | Código Python para cruzar dados e calcular o \*\*Índice de Pontualidade\*\*. |

| \*\*Membro 5\*\* (Carga) | Carga (L) | Executa o script final, gera o `dados\_tratados.csv` (evidência da Carga). |

| \*\*Membro 6\*\* (Visualização) | OLAP, Dashboards e Vídeo | Cria o painel analítico no Power BI e grava o vídeo demonstrativo. |



Texto

O conteúdo gerado por IA pode estar incorreto.

import pandas as pd

import os

# Define o caminho base do projeto para encontrar as pastas de dados

BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

RAW\_PATH = os.path.join(BASE\_DIR, 'data\_raw')

TREATED\_PATH = os.path.join(BASE\_DIR, 'data\_treated')

OUTPUT\_FILE = os.path.join(TREATED\_PATH, 'dados\_tratados.csv')

# ==============================================================================

# TRECHO DO MEMBRO 3: LEITURA, LIMPEZA E PREPARAÇÃO DE DADOS (T)

# ==============================================================================

print("Membro 3: Iniciando leitura, limpeza e preparação de dados...")

try:

df\_reais = pd.read\_csv(os.path.join(RAW\_PATH, 'dados\_reais\_simulados.csv'))

df\_viagens = pd.read\_csv(os.path.join(RAW\_PATH, 'gtfs\_viagens\_simuladas.csv'))

except FileNotFoundError as e:

print(f"ERRO CRÍTICO: Arquivo de dados não encontrado. Verifique se estão na pasta data\_raw/. Detalhe: {e}")

exit()

# 1. Limpeza Básica: Remoção de Duplicatas e Nulos

df\_reais\_limpo = df\_reais.drop\_duplicates()

df\_reais\_limpo = df\_reais\_limpo.dropna(subset=['horario\_real', 'lotacao', 'trip\_id'])

# 2. Conversão de Tempo para Datetime (CRÍTICO para o cálculo)

# Converte a coluna de hora (string) para um objeto datetime, necessário para fazer a subtração

df\_reais\_limpo['horario\_real\_dt'] = pd.to\_datetime(df\_reais\_limpo['horario\_real'], format='%H:%M:%S', errors='coerce').dt.time

print(f"Membro 3: Limpeza concluída. Total de registros para trabalho: {len(df\_reais\_limpo)}")

# df\_trabalho é a base de dados limpa que será usada pelo Membro 4

df\_trabalho = df\_reais\_limpo

# Commit do Membro 3: Salva o arquivo e envia

# ==============================================================================

# TRECHO DO MEMBRO 4: CRUZAMENTO DE DADOS E CÁLCULO DE MÉTRICAS (T)

# ==============================================================================

print("Membro 4: Iniciando cruzamento de dados e cálculo de métricas...")

# 1. Preparação da Tabela de Viagens (Horário Previsto)

df\_viagens['arrival\_time\_dt'] = pd.to\_datetime(df\_viagens['arrival\_time'], format='%H:%M:%S', errors='coerce').dt.time

# 2. Cruzamento (Merge): Junta a tabela REAL com a tabela PREVISTA (schedule)

df\_analise = pd.merge(

df\_trabalho,

df\_viagens[['trip\_id', 'arrival\_time\_dt']],

on='trip\_id',

how='left'

)

# 3. Função Auxiliar para Calcular Tempo Total em Segundos (Necessário para subtração)

def time\_to\_seconds(t):

if pd.isna(t):

return 0

return t.hour \* 3600 + t.minute \* 60 + t.second

# Conversão para segundos e cálculo da métrica

df\_analise['real\_segundos'] = df\_analise['horario\_real\_dt'].apply(time\_to\_seconds)

df\_analise['previsto\_segundos'] = df\_analise['arrival\_time\_dt'].apply(time\_to\_seconds)

# FÓRMULA DO ÍNDICE DE PONTUALIDADE (atraso em minutos)

# Atraso = (Tempo Real - Tempo Previsto) / 60

df\_analise['atraso\_segundos'] = df\_analise['real\_segundos'] - df\_analise['previsto\_segundos']

df\_analise['atraso\_minutos'] = df\_analise['atraso\_segundos'] / 60

# 4. Enriquecimento: Criação da Dimensão "Período do Dia"

# Usaremos 'lotacao' > 40 como um índice de lotação alta

df\_analise['indice\_lotacao'] = df\_analise['lotacao'].apply(lambda x: 'Alta' if x >= 40 else 'Baixa')

# df\_final é a base de dados pronta para análise

df\_final = df\_analise.dropna(subset=['atraso\_minutos'])

print("Membro 4: Cálculo de Pontualidade e enriquecimento concluídos.")

# Commit do Membro 4: Salva o arquivo e envia

# ==============================================================================

# ==============================================================================

# TRECHO DO MEMBRO 5: CARGA DOS DADOS TRATADOS (L)

# ==============================================================================

print("Membro 5: Iniciando a Carga dos Dados Tratados...")

# Seleciona APENAS as colunas que são relevantes para o BI (o Data Warehouse)

colunas\_bi = [

'trip\_id',

'lotacao',

'atraso\_minutos',

'indice\_lotacao',

'horario\_real'

]

# 1. Filtra as colunas finais

df\_final\_bi = df\_final[colunas\_bi]

# 2. Salva o resultado no caminho definido

# Este arquivo CSV é a Carga (L) e o DDL/DML do nosso Data Warehouse simplificado.

df\_final\_bi.to\_csv(OUTPUT\_FILE, index=False)

print(f"Membro 5: Carga concluída. Arquivo salvo em: {OUTPUT\_FILE}. Total de {len(df\_final\_bi)} registros.")

# Commit do Membro 5: Salva o arquivo e envia

# ==============================================================================

Texto

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.