

1 Introdução

O presente relatório tem como objetivo apresentar o processo completo de construção e análise dos dados energéticos, desde a coleta, tratamento e armazenamento, até a criação de indicadores e dashboards analíticos no Power BI.

O projeto foi desenvolvido com foco em fornecer insights estratégicos para as áreas de Operações e Comercial, permitindo uma visão integrada do comportamento de consumo energético, perdas técnicas e não técnicas, e eficiência operacional na resolução de ocorrências técnicas.

2 Etapas do Projeto

2.1 Tratamento e Padronização de Dados (Python)

Nesta etapa, foram utilizadas bibliotecas como pandas, pathlib, datetime e re para:

- Ler arquivos CSV contendo medições, ocorrências e dados de clientes;
- Padronizar formatos de data (para dd/mm/yyyy);
- Remover duplicatas e valores inconsistentes;
- Gerar identificadores únicos para cidades e estados (id_cidade, id_estado, id_cidade_estado);
- Salvar os arquivos tratados em formato CSV UTF-8, prontos para carga no banco PostgreSQL.

Principais funções criadas em Python:

- **converter_csv_para_xlsx()** → Leitura e limpeza dos dados brutos;
- **registrar_log()** → Registro de logs automáticos de execução;
- Funções auxiliares de padronização (**normalizar_texto**, **gerar_ids_localidades**);
- Criação de dicionário de mapeamento entre cidades e estados.

Essas funções garantiram a consistência dos dados e prepararam os arquivos para integração com o PostgreSQL.

2.2 Criação e Carga das Tabelas no PostgreSQL

Após o tratamento, os dados foram armazenados no **PostgreSQL**, utilizando scripts SQL para criação de tabelas dimensionais e fato, de acordo com o modelo estrela (**Star Schema**).

A etapa de criação e preparação dos dados no postgresql, é através do PgAdmin4, que primeiramente se realizou a criação do database, o qual o nome dele é: **database_light**:

```
1  -- Criação do banco de dados
2  CREATE DATABASE database_light;
```

Posteriormente, foram criados o Schema e as Tabelas necessárias.

Principais tabelas criadas:

- **dim_clientes** – informações cadastrais dos clientes;
- **dim_localidade** – dados geográficos de cidade e estado;
- **ft_medicoes_energia** – registros de consumo energético;
- **ft_ocorrencias_tecnicas** – incidentes e tempo de reparo;
- **ft_perdas_energia** – dados de perdas técnicas e não técnicas;

Exemplo abaixo mostra a criação do Schema do projeto dentro do PgAdmin4 cujo nome é **projeto_light**,

Em seguida a criação deste realiza-se as criações das tabelas.

```
1  CREATE SCHEMA IF NOT EXISTS projeto_light;
2
3  --Cria o schema
4  CREATE SCHEMA IF NOT EXISTS projeto_light AUTHORIZATION CURRENT_USER;
5
6  --Tabela dimensão para clientes
7  CREATE TABLE IF NOT EXISTS projeto_light.clientes (
8      id_cliente SERIAL PRIMARY KEY,
9      nome_cliente VARCHAR(255),
10     cidade VARCHAR(100),
11     estado VARCHAR(100),
12     tipo_cliente VARCHAR(50),
13     data_adesao DATE,
14     id_cidade INT,
15     id_estado INT,
16     id_cidade_estado INT
17 );
```

No script do sql, a tabela calendário foi comentada. Foi optado por utilizar a maneira dinâmica que o powerbi oferece para criar uma tabela calendário.

Após a criação das tabelas, fica-se apto a receber os dados das tabelas tratadas no python:

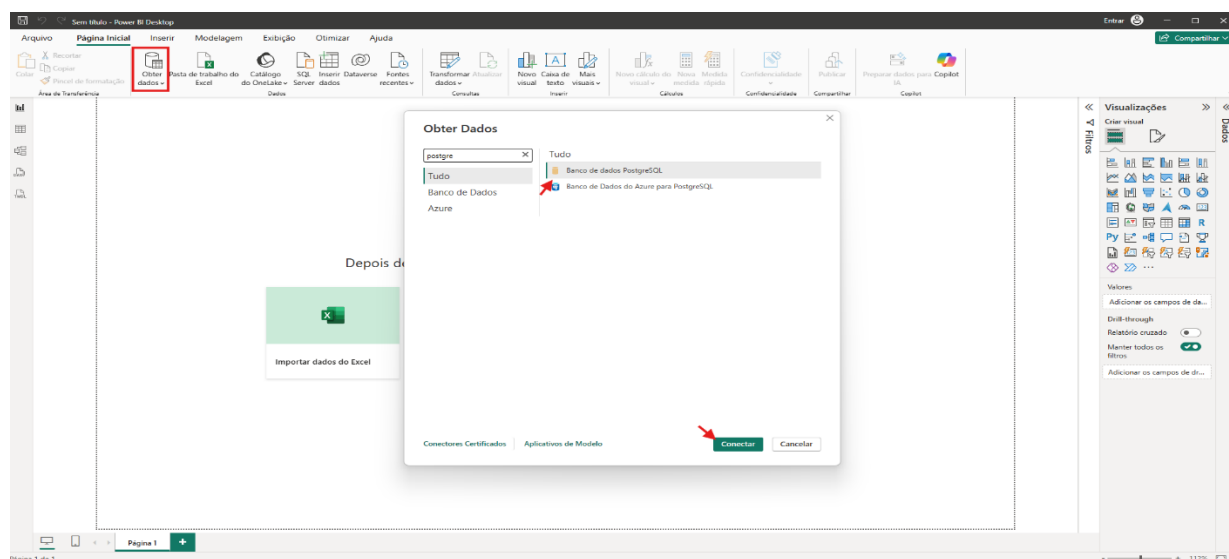
```
1 COPY projeto_light.clientes (  
2     id_cliente,  
3     nome_cliente,  
4     cidade,  
5     estado,  
6     tipo_cliente,  
7     data_adesao,  
8     id_cidade,  
9     id_estado,  
10    id_cidade_estado  
11 )  
12 FROM 'C:\\database_final\\clientes_tratado.csv'  
13 DELIMITER ',' CSV HEADER ENCODING 'UTF8';
```

2.3 Conexão e Modelagem no Power BI

O arquivo do Power BI, chamado **projeto_light.pbix**, foi conectado diretamente ao **banco PostgreSQL**, utilizando a conexão que o próprio Power Bi oferece.

Etapas realizadas:

1. Configuração da conexão PostgreSQL (host, porta, banco e credenciais);



Banco de dados PostgreSQL

Servidor

localhost

Banco de Dados

database_light

Modo de Conectividade de Dados ⓘ

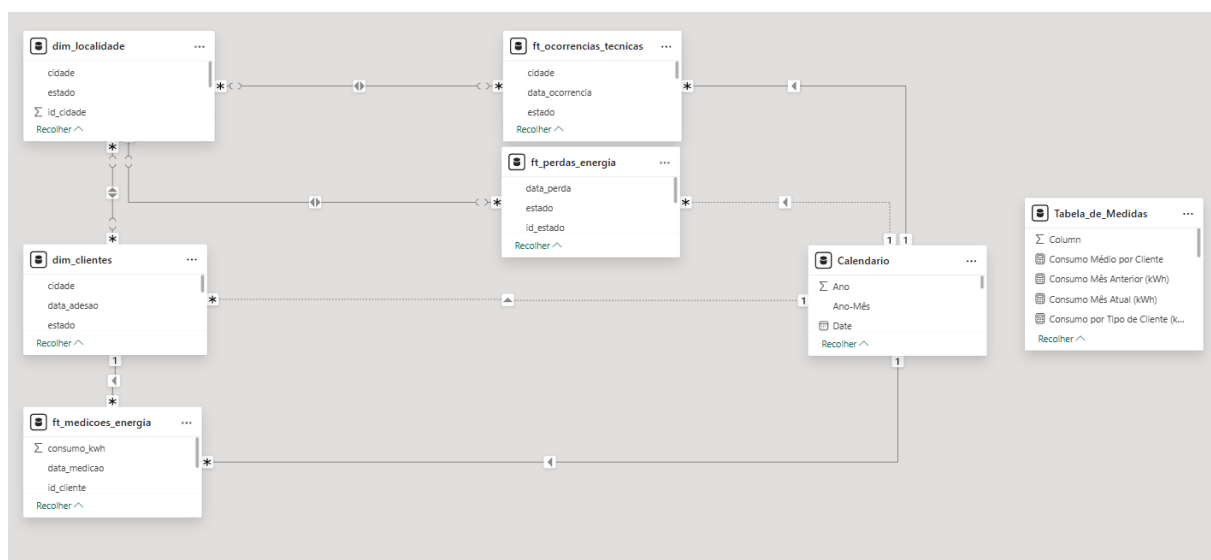
☒ Importar
 ☐ DirectQuery

Opções avançadas

OK

Cancelar

2. Importação das tabelas **dimensões** e **fatos**;
3. Criação dos relacionamentos entre tabelas;



4. Implementação da tabela **Calendário** para análises temporais;

```

1 Calendario =
2 VAR DataInicial = DATE(2020,1,1)
3 VAR DataFinal   = DATE(2030,12,31)
4 RETURN
5 ADDCOLUMNS (
6     CALENDAR (DataInicial, DataFinal),
7     "Ano", YEAR ( [Date] ),
8     "Mês", MONTH ( [Date] ),
9     "Dia", DAY ( [Date] ),
10    "Nome Mês", FORMAT ( [Date], "mmmm" ),
11    "Nome Mês Abrev", FORMAT ( [Date], "mmm" ),
12    "Ano-Mês", FORMAT ( [Date], "YYYY-MM" ),
13    "Trimestre", "T" & FORMAT ( [Date], "Q" ),
14    "Semestre", IF ( MONTH ( [Date] ) <= 6, "1º Semestre", "2º Semestre" ),
15    "Nome Dia Semana", FORMAT ( [Date], "dddd" ),
16    "Nome Dia Semana Abrev", FORMAT ( [Date], "ddd" ),
17    "Número Dia Semana", WEEKDAY ( [Date], 2 ),
18    "É Final de Semana", IF ( WEEKDAY ( [Date], 2 ) >= 6, TRUE, FALSE )
19 )
20

```

5. Criação de medidas DAX para cálculo de indicadores.

3 Análises e Indicadores Principais

Filtros:

- Número do Cliente;
- Número do Estado;
- Nome do Cliente;
- Número da Cidade;
- Nome do Estado;
- Data.

Indicadores criados do lado comercial:

- Cards: Consumo Total em kWh;
- Mapa de filtro por Estado;
- Consumo total por data;
- Consumo total por tipo de cliente;
- Ranking de Clientes.

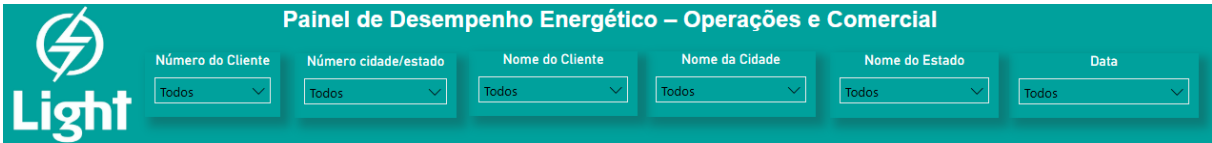
Indicadores criados do lado operacional:

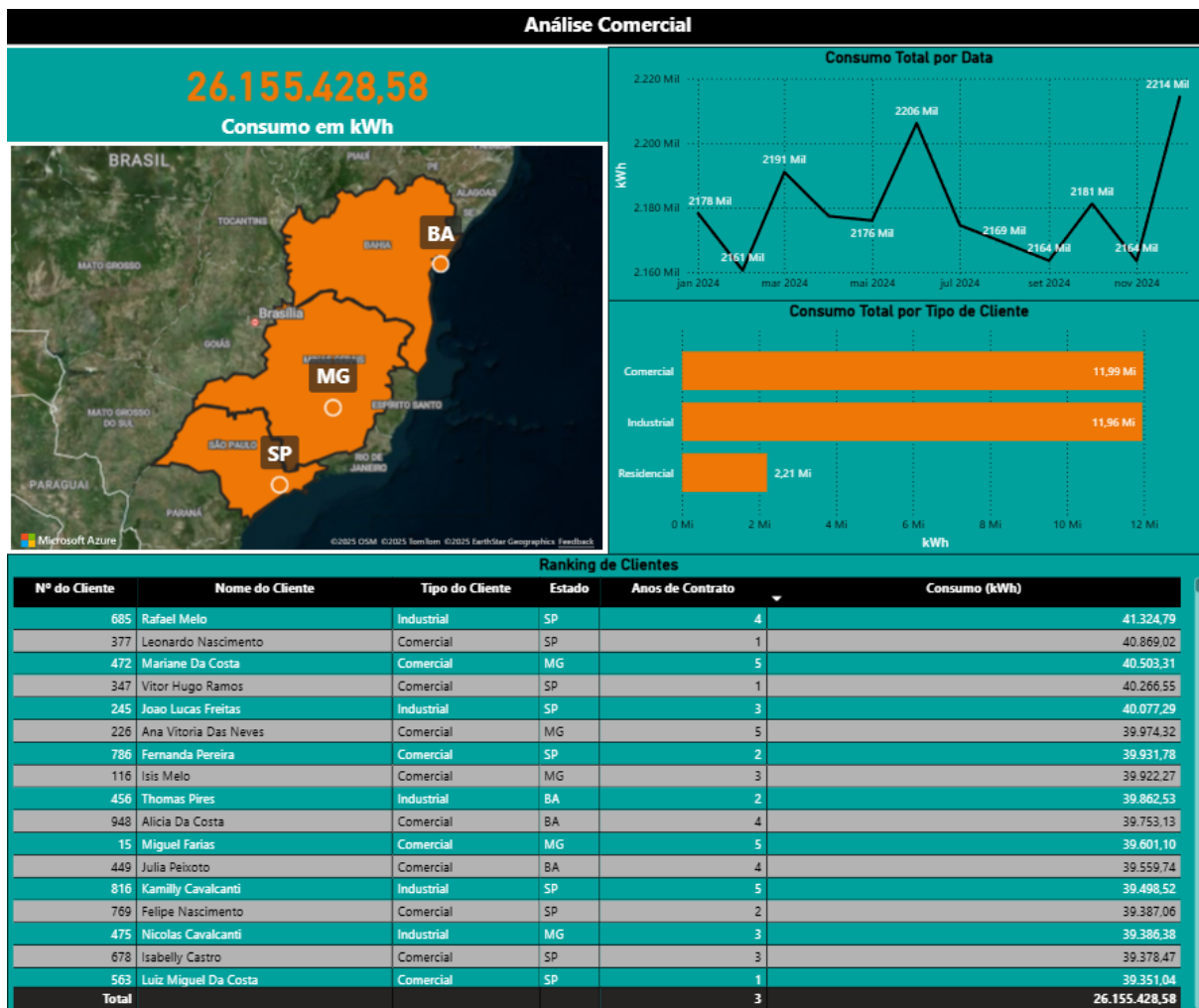
- Cards: Quantidade de Ocorrências, Média de Horas de Reparo, Percentual de Perdas e Média de Ocorrências por Cidade;
- Consumo x Perdas;
- Quantidade de Ocorrências por Estado;
- Tempo Médio de Reparo em Horas;
- Percentual de Perdas Mensal;
- Perda Técnica x Não Técnica por Data.

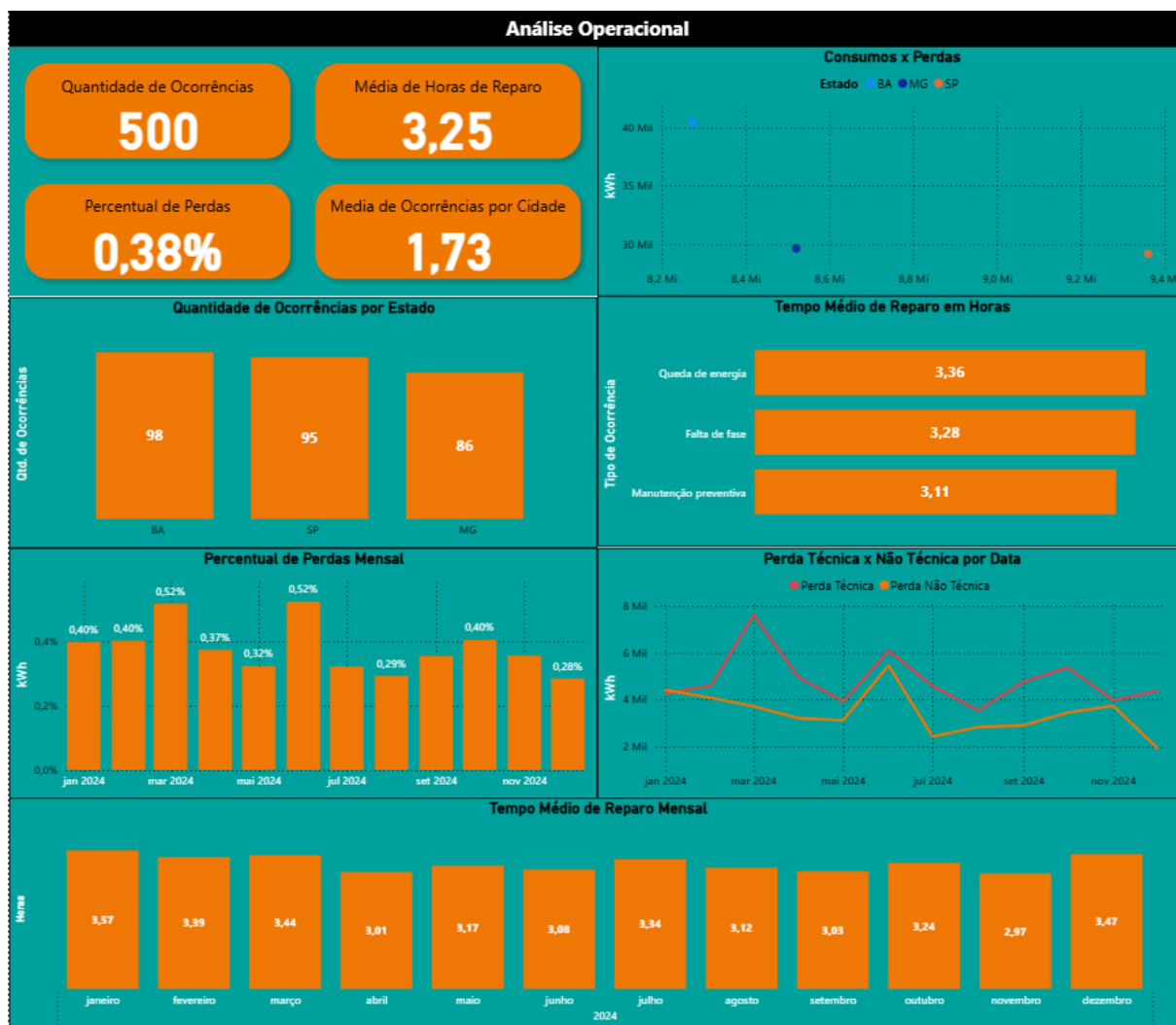
Principais conclusões:

- Regiões com alto consumo apresentaram também maiores quantidades ocorrências;
- O consumo em kWh é alto no verão;
- O tempo médio de reparo é alto no verão;
- Clientes comerciais concentram a maior parte do consumo energético;
- Região de SP é a que mais consumiu, porém BA foi que teve maior quantidade de ocorrências;
- Correlação positiva entre consumo e aumento das perdas técnicas.

4 Layout final segmentado







5 Hipóteses e Recomendações

Hipóteses:

- Perdas não técnicas podem estar associadas a ligações irregulares;
- Residencial tem consumo menor por não usufruir de muitos equipamentos que consomem energia comparado com comercial e industrial;
- Tempo médio de Reparo maior no verão deve-se a quantidade de ocorrências serem maiores de queda de energia, o que gera complexidade na solução por ter diversas formas desta queda ocorrer;
- Regiões de alto consumo exigem reforço de rede e inspeções mais frequentes.

Recomendações:

Desenvolvido por:

Vinicius Nunes Ribeiro Rigoni

- Reforçar inspeções em áreas críticas;
- Criar programas de eficiência energética;
- Desenvolver ações comerciais voltadas à fidelização de clientes antigos.

6 Conclusão

A integração entre **Python**, **PostgreSQL** e **Power BI** permitiu construir um pipeline completo de dados desde o processo da inserção dos dados até a visualização dos mesmos, garantindo assim, qualidade, confiabilidade e insights estratégicos para apoio à decisão nas áreas de Operações e Comercial.