Unificando os Dados

```
import pandas as pd
import unicodedata
import os
import numpy as np

def normalizar_colunas(df: pd.DataFrame) -> pd.DataFrame:
    """
    Normaliza os nomes das colunas de um DataFrame.
    Remove acentos, espaços, caracteres especiais e converte para minúsculas.

    @param df: DataFrame do pandas.
    @return: DataFrame com colunas normalizadas.
    """
    # ... (seu código atual, que está eficiente para esta tarefa)
    df.columns = [
        unicodedata.normalize("NFKD", str(c))
        .encode("ascii", "ignore")
        .decode("ascii")
        .strip()
        .replace(" ", "_")
        .replace("/", "_")
        .replace("-", "_")
        .lower()
        for c in df.columns
    ]
    return df

def mesclar_csvs(*csv_paths, output_path="dados_unificados.csv",
preencher_ausentes=False) -> pd.DataFrame:
    """
    Mescla múltiplos arquivos CSV, normaliza colunas, converte dados de data/hora e
    agrega colunas de temperatura e consumo.

    @param csv_paths: Caminhos para os arquivos CSV a serem mesclados.
    @param output_path: Caminho para salvar o arquivo CSV unificado.
    @param preencher_ausentes: Se True, preenche valores ausentes com a média mensal.
    @return: DataFrame unificado.
    """
    if len(csv_paths) < 2:
        raise ValueError("Informe pelo menos dois arquivos CSV para mesclar.")

    dataframes = []

    for path in csv_paths:
        if not os.path.exists(path):
```

```
            print(f"[ERRO] Arquivo não encontrado: {path}")
            continue

        try:
            # Tenta UTF-8, se falhar, tenta latin-1
            df = pd.read_csv(path, encoding="utf-8")
        except UnicodeDecodeError:
            df = pd.read_csv(path, encoding="latin-1")

        # Tratamento de vírgula como separador decimal para colunas numéricas
        # Isso é crucial para dados brasileiros
        # Detecta automaticamente colunas com ',' e força a conversão
        for col in df.columns:
            if df[col].dtype == 'object' and df[col].str.contains(',').any():
                df[col] = df[col].str.replace('.', '').str.replace(',', '.')

        df = normalizar_colunas(df)

        if "data" not in df.columns:
            print(f"[AVISO] '{os.path.basename(path)}' não possui coluna 'data'. Ignorado.")
            continue

        # Garante que a coluna 'data' esteja no formato datetime
        df["data"] = pd.to_datetime(df["data"], errors="coerce")
        df = df.dropna(subset=["data"])

        colunas_temperatura = [c for c in df.columns if "temp" in c]
        colunas_consumo = [c for c in df.columns if "consumo" in c and "kw" in c]

        # Converte para numérico após o tratamento de vírgulas
        for c in colunas_temperatura + colunas_consumo:
            df[c] = pd.to_numeric(df[c], errors="coerce")

        # Agrega as colunas de temperatura e consumo
        df["temperatura_media"] = df[colunas_temperatura].mean(axis=1, skipna=True) if
colunas_temperatura else np.nan
        df["consumo_kw"] = df[colunas_consumo].mean(axis=1, skipna=True) if
colunas_consumo else np.nan

        df = df[["data", "consumo_kw", "temperatura_media"]].dropna(subset=["data"])

        # Agrupa por data e tira a média (para o caso de múltiplas linhas para a mesma data,
como no seu 'anuario')
        df = df.groupby("data", as_index=False).mean(numeric_only=True)
        dataframes.append(df)

    if not dataframes:
        raise ValueError("Nenhum arquivo válido encontrado.")
```

```
    # Concatena e faz um agrupamento final
    df_final = pd.concat(dataframes, ignore_index=True)
    df_final = df_final.groupby("data", as_index=False).mean(numeric_only=True)

    # Lógica de preenchimento (se solicitada)
    if preencher_ausentes:
        df_final["mes"] = df_final["data"].dt.month
        for col in ["consumo_kw", "temperatura_media"]:
            # Preenche ausentes com a média mensal do respectivo mês
            df_final[col] = df_final.groupby("mes")[col].transform(lambda x: x.fillna(x.mean()))
        df_final = df_final.drop(columns=["mes"])

    df_final = df_final.sort_values(by="data").reset_index(drop=True)
    df_final.to_csv(output_path, index=False, encoding="utf-8")

    print(f"[FINALIZADO] Arquivo unificado salvo em: {output_path}")
    print(f"[INFO] Linhas: {len(df_final)} | Colunas: {len(df_final.columns)}")
    return df_final

if __name__ == "__main__":
    # Mantém a execução principal, mas se atente à limpeza dos seus CSVs de entrada
    df_final = mesclar_csvs(
        "./dados filtrados/anuario estatistico de energia eletrica_filtrado.csv",
        "./dados filtrados/INMET_BRASILIA_01-01-2024_A_31-12-2024_filtrado.csv",
        "./dados filtrados/INMP 15102025-15102025_filtrado.csv",
        output_path="dados_unificados.csv",
        preencher_ausentes=True, # Alterei para True, pode ajudar na consistência
    )
```