Plano de Cargas ETL com PostgreSQL

1. **Objetivo:**
   - O objetivo deste plano é realizar a extração, transformação e carregamento de dados do sistema fonte para o data warehouse, utilizando Python e Apache Hop para as etapas de extração e transformação, e PostgreSQL para o carregamento dos dados no destino final.

2. **Fonte de Dados:**
   - Os dados serão extraídos de um arquivo CSV.

3. **Destino dos Dados:**
   - Os dados transformados serão carregados no PostgreSQL, que servirá como o data warehouse para este projeto.

4. **Ferramentas Utilizadas:**
   - Python: Utilizado para scripts de extração e transformação dos dados.
   - PostgreSQL: Utilizado para a criação do data warehouse e o carregamento dos dados transformados.

5. **Processo ETL:**

   a) **Extração (Extract):**

   - Os dados serão extraídos da fonte de dados utilizando scripts Python e pipelines de dados do Apache Hop, que realizarão consultas e recuperarão as informações necessárias.

   b) **Transformação (Transform):**

   - As transformações serão aplicadas aos dados extraídos usando Python e Apache Hop, onde serão realizadas operações como limpeza, conversão de tipos, agregações, entre outras.

   c) **Carregamento (Load):**

   - Os dados transformados serão carregados no PostgreSQL utilizando scripts Python com o auxílio da biblioteca psycopg2 para interagir com o banco de dados.

6. **Monitoramento:**

- Será implementado um sistema de monitoramento para acompanhar o status e o desempenho do processo ETL, garantindo a integridade e a qualidade dos dados carregados no data warehouse.

7. **Manutenção:**
   - Atividades de manutenção serão realizadas periodicamente para garantir a eficiência do processo ETL, incluindo otimizações de desempenho, ajustes nas transformações, entre outras ações necessárias.

8. **Carga dos Dados:**

   - Aqui estão os scripts e pipelines utilizando para a carga dos dados no data warehouse no PostgreSQL.

Carga dim_candidato:

O script começa importando as bibliotecas pandas e psycopg2. O pandas é usado para manipulação de dados em formato de DataFrame, enquanto o psycopg2 é uma biblioteca Python para interação com bancos de dados PostgreSQL. Em seguida, o script estabelece uma conexão com o banco de dados PostgreSQL. São fornecidos os parâmetros de conexão, como nome do banco de dados, usuário, senha, host e porta.

O script faz a leitura de um arquivo CSV usando o pandas, criando um DataFrame chamado df_enem. O arquivo CSV está localizado no caminho especificado e utiliza ';' como separador e 'UTF-8' como codificação.

O script itera sobre as linhas do DataFrame df_enem utilizando o método iterrows(). Para cada linha, é executado um comando de inserção SQL na tabela dim_candidato do banco de dados. Os valores são inseridos usando placeholders %s e são passados como parâmetros para evitar problemas de segurança como SQL Injection.

Após a conclusão da iteração e inserção de dados, o script confirma a transação no banco de dados com conn.commit() e fecha o cursor e a conexão com o banco de dados utilizando cursor.close() e conn.close()

Script Python:

```
import pandas as pd
import psycopg2
```

```
    # Estabelece a conexão com o banco de dados
    conn = psycopg2.connect(
        dbname='Enem_2020',
        user='postgres',
        password=1234,
        host='localhost',
        port=5432
      )

    cursor = conn.cursor()

    # Leitura do CSV para DataFrame
    df_enem = pd.read_csv(r"C:/Users/peixe/Desktop/Desafio Técnico Mesha/Base Enem Tratada2.csv", sep=';', encoding='UTF-8')

     # Itera sobre as linhas do DataFrame e executa o comando de inserção
    for index, row in df_enem.iterrows():
        cursor.execute(
          """
          INSERT INTO dim_candidato (nu_inscricao, faixa_etaria, sexo, etnia, estado_civil, nacionalidade, st_conclusao, treineiro)
          VALUES (%s, %s, %s, %s, %s, %s, %s, %s)
          """,
          (
            row['NU_INSCRICAO'],
            row['TP_FAIXA_ETARIA'],
            row['SEXO'],
            row['ETNIA'],
            row['ESTADO_CIVIL'],
            row['NACIONALIDADE'],
            row['ST_CONCLUSAO'],
            row['TREINEIRO']
          )
        )

    # Confirma a transação
    conn.commit()
    # Fecha o cursor e a conexão
    cursor.close()
    conn.close()
```

Carga dim_escola:

O script que dá a carga na dim_escola segue o mesmo padrão do script de carga da dim_candidato, apenas com alterações nos campos a receberem a carga dos dados.

Script Python:

```
import pandas as pd
import psycopg2

# Estabelece a conexão com o banco de dados
conn = psycopg2.connect(
        dbname='Enem_2020',
        user='postgres',
        password=1234,
        host='localhost',
        port=5432
    )

cursor = conn.cursor()

# Leitura do CSV para DataFrame
df_enem = pd.read_csv(r"C:/Users/peixe/Desktop/Desafio Técnico Mesha/Base Enem Tratada2.csv", sep=';', encoding='UTF-8')

 # Itera sobre as linhas do DataFrame e executa o comando de inserção
for index, row in df_enem.iterrows():
        cursor.execute(
            """
            INSERT INTO dim_escola (tipo_escola, localizacao_esc)
            VALUES (%s, %s)
            """,
            (
                row['TIPO_ESCOLA'],
                row['LOCALIZACAO_ESC']
            )
                )

# Confirma a transação
conn.commit()
# Fecha o cursor e a conexão
cursor.close()
conn.close()
```

Carga dim_presenca:

Também segue o mesmo padrão das demais cargas, só alterando as colunas de inserção de dados.

Script Python:

```
import pandas as pd
import psycopg2

# Estabelece a conexão com o banco de dados
conn = psycopg2.connect(
        dbname='Enem_2020',
        user='postgres',
        password=1234,
        host='localhost',
        port=5432
    )

cursor = conn.cursor()

# Leitura do CSV para DataFrame
df_enem = pd.read_csv(r"C:/Users/peixe/Desktop/Desafio Técnico Mesha/Base Enem Tratada2.csv", sep=';', encoding='UTF-8')

 # Itera sobre as linhas do DataFrame e executa o comando de inserção
for index, row in df_enem.iterrows():
        cursor.execute(

            """
            INSERT INTO dim_presenca (presenca_cn, presenca_ch, presenca_lc, presenca_mt)
            VALUES (%s, %s, %s, %s)
            """,
            (
               row['PRESENCA_CN'],
```

```
            row['PRESENCA_CH'],
            row['PRESENCA_LC'],
            row['PRESENCA_MT']
        )
    )

    # Confirma a transação
    conn.commit()
    # Fecha o cursor e a conexão
    cursor.close()
    conn.close()
```

Carga dim_localidade:

Também segue o mesmo padrão das outras cargas, alterando as colunas de inserção.

```
    Script Python:
    import pandas as pd
    import psycopg2

    # Estabelece a conexão com o banco de dados
    conn = psycopg2.connect(
        dbname='Enem_2020',
        user='postgres',
        password=1234,
        host='localhost',
        port=5432
      )

    cursor = conn.cursor()
```

```
# Leitura do CSV para DataFrame
df_enem = pd.read_csv(r"C:/Users/peixe/Desktop/Desafio Técnico Mesha/Base Enem Tratada2.csv", sep=';', encoding='UTF-8')


 # Itera sobre as linhas do DataFrame e executa o comando de inserção
for index, row in df_enem.iterrows():
      cursor.execute(


  INSERT INTO dim_localidade (uf, nome_municipio)
        VALUES (%s, %s)
        """,
        (
           row['UF'],
           row['NOME_MUNICIPIO']
        ))


# Confirma a transação
conn.commit()
# Fecha o cursor e a conexão
cursor.close()
conn.close()
```

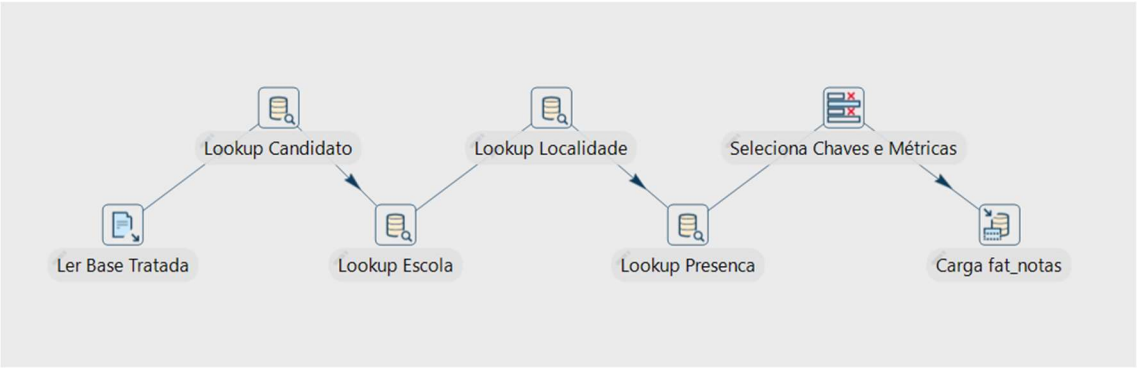
Carga fat_notas:

Seguindo uma abordagem diferente, utilizando o apache hop como ferramenta de ETL, a carga da tabela fat_notas, tabela que leva as chaves primárias para relacionamento entre as tabelas e as suas devidas métricas, a tabela fat_notas foi a carga considerada mais complexa devida a sua alta escalabilidade de dados, gerando inúmeras consultas para busca de chaves primárias, como por exemplo o uso de joins (Lookups) entre as tabelas para fazer as localizações da métricas de forma correta e consistente.

Devido a essa alta complexidade de carga da tabela fat_notas, o tempo de processamento da carga passou de 20 horas, e a carga não foi completa,

carregando apenas 840 mil registros, e com isso inviabilizou a continuidade do processo. A seguir o pipeline de carga no Apache Hop.

Pipeline de Dados:



1º Step = Leitura da base de dados tratada do Enem 2020 e reconhecimento dos campos de dados.

2º Step = Lookup da dimensão dim_candidato, buscando seus campos de dados e chave primária.

3º Step = Lookup da dimensão dim_escola, buscando seus campos de dados e chave primária.

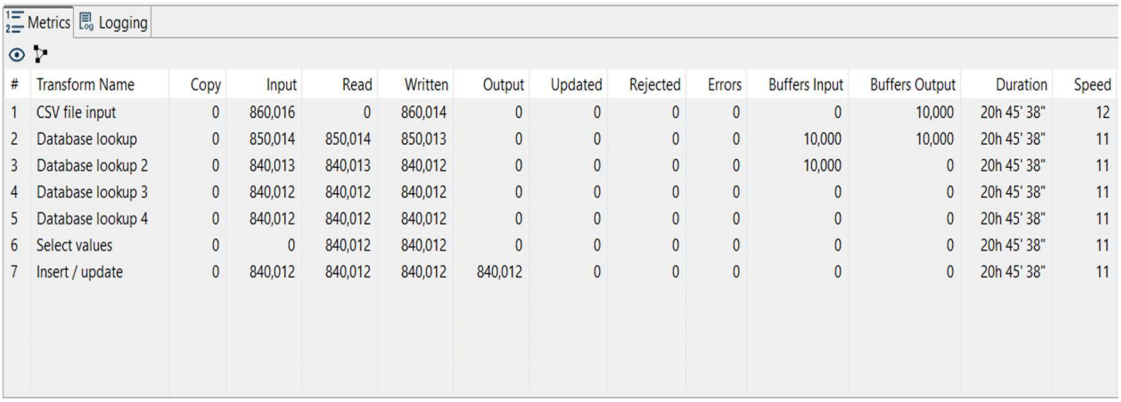
4º Step = Lookup da dimensão dim_localidade, buscando seus campos de dados e chave primária.

5º Step = Lookup da dimensão dim_presenca, buscando seus campos de dados e chave primária.

6º Step = Select Values (Seleciona Campos) para selecionar chaves primárias e métricas para carga de dados.

7º Step = Insert/Update para selecionar todos os campos de métricas e chaves primárias e selecionar os campos atualizáveis ou não.

Todos os steps estão devidamente conectados com o banco de dados PostgreSQL no database "**Enem_2020**".



Metrics | Logging

#	Transform Name	Copy	Input	Read	Written	Output	Updated	Rejected	Errors	Buffers Input	Buffers Output	Duration	Speed
1	CSV file input	0	860,016	0	860,014	0	0	0	0	0	10,000	20h 45' 38"	12
2	Database lookup	0	850,014	850,014	850,013	0	0	0	0	10,000	10,000	20h 45' 38"	11
3	Database lookup 2	0	840,013	840,013	840,012	0	0	0	0	10,000	0	20h 45' 38"	11
4	Database lookup 3	0	840,012	840,012	840,012	0	0	0	0	0	0	20h 45' 38"	11
5	Database lookup 4	0	840,012	840,012	840,012	0	0	0	0	0	0	20h 45' 38"	11
6	Select values	0	0	840,012	840,012	0	0	0	0	0	0	20h 45' 38"	11
7	Insert / update	0	840,012	840,012	840,012	840,012	0	0	0	0	0	20h 45' 38"	11

Imagem que mostra a duração da carga fat_notas (quase 21 horas de carga) e a quantidade de registros inseridos (Output).

As cargas realizadas nas dimensões foram todas completadas de acordo com a base de dados, porém devido ao tempo de processamento da carga da tabela fat_notas e o prazo de entrega do projeto, a carga da fat_notas não foi realizada por completo, e por isso pode apresentar alguns valores diferentes em comparação a valores já conhecidos (caso existam).