Disease Model Process Document

This document provides step-by-step instructions for creating the disease model database structure with SQL and implementing automated ETL/ELT processes to load the data using Python on Postgres.

File Structure/Hierarchy:

1. **Schema: raw_data_staging_layer -**
    a. 1. raw_data_table_creation_scripts.sql
    b. 2. raw_data_load.py
2. **Schema: curated_layer -**
    a. 3. curated_layer_transaction_table_creation_scripts.sql
    b. 4. curated_data_load.py
3. **Schema: reporting_layer_dw -**
    a. 5. Reporting_Layer_DW_creation_scripts.sql
    b. 6. ELT_curated_to_warehouse.py
    c. 7. views_creation_script.sql

Script Explanation:

1. raw_data_table_creation_scripts.sql

This script contains SQL queries that create raw data tables. These tables store the unprocessed data extracted from source files such as CSV, JSON, and other formats.

2. raw_data_load.py

A Python script that performs **Extract and Load (EL)** operations, transferring raw data from source files into the raw data tables.

3. curated_layer_transaction_table_creation_scripts.sql

This script contains SQL queries for creating transaction tables (final tables) that store transformed data derived from the raw data layer.

4. curated_data_load.py

A Python script that extracts data from raw data tables, transforms it into the required format, and loads it into the transaction tables within the curated data layer.

5. Reporting_Layer_DW_creation_scripts.sql

This script contains SQL queries to create a data warehouse, including dimension and fact tables, for reporting purposes.

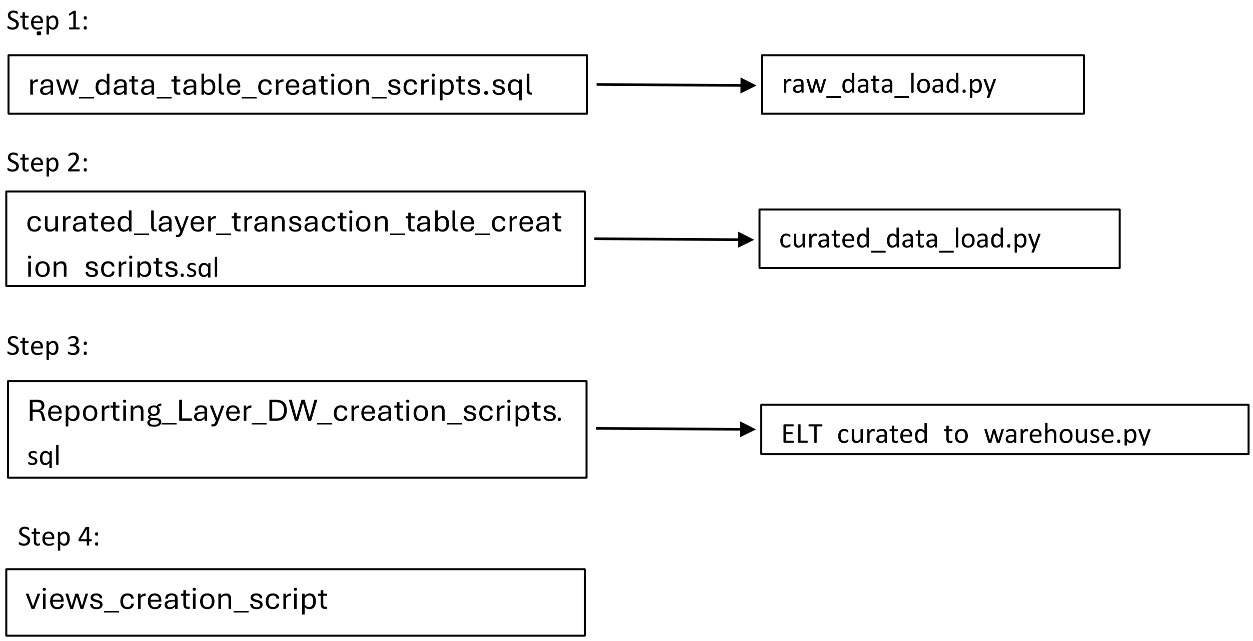
6. ELT_curated_to_warehouse.py

A Python script that extracts data from the curated data layer, transforms it as required using temporary tables, and loads it into the data warehouse.

7. views_creation_script.sql

This script creates views on top of the data warehouse, providing structured and query-ready data for building reports using BI tools.

Process Flow:



Step 1:

raw_data_table_creation_scripts.sql → raw_data_load.py

Step 2:

curated_layer_transaction_table_creation_scripts.sql → curated_data_load.py

Step 3:

Reporting_Layer_DW_creation_scripts.sql → ELT_curated_to_warehouse.py

Step 4:

views_creation_script

Requirements:

1. python3
2. pip3
3. psycopg2
4. pandas
5. Sqlalchemy

Instructions to install the project and perform the EL/ETL/ELT:

Step 1: Project Setup

1. Download the project.
2. Navigate to Code-Data-Instructions to Run folder under the Submissions folder.
3. Open the folder in an IDE of choice.
4. Create a postgres database by name **WHO_Health_Tracker**.

NOTE: All the csv files in the project folder serve as source or raw data for this project.

Step 2: Install the Requirements

1. Confirm that you have python3 with pip3 installed on your computer.
2. Install the packages mentioned in the requirements section.
3. If any of the package is missing, run pip3 install package_name to have the package installed.

Step 3: Create tables and run EL/ETL/ELT

1. Create 3 schemas on the WHO_Health_Tracker database, namely raw_data_staging_layer, cura**ted_layer** and **reporting_layer_dw.**
2. Run **raw_data_table_creation_scripts.sql** to create raw data staging tables.
3. Open **raw_data_load.py,** replace DB_Config dictionary with your database credentials and execute the script.
4. Execute the **curated_layer_transaction_table_creation_scripts.sql** followed by **curated_data_load.py.** Please remember to replace the DB_Config dictionary with your database credentials.
5. Similarly execute **Reporting_Layer_DW_creation_scripts.sql** followed by **ELT_curated_to_warehouse.py** after replacing the DB_Config dictionary values with your database credentials.
6. To create and test views, execute **views_creation_script.sql**

This should set up and create the database with all the tables and views. You can now go around, explore and play with the data!!!!