Extraction Process:

Kaggle Dataset Extraction Documentation:

Authentication with Kaggle API:

Kaggle username
API Token
Token Placement: Place the downloaded kaggle.json file in the ~/.kaggle/ directory.

Python Script for Extraction:

Parameters Used:

- dataset_name: The Kaggle dataset name.

Location of Extracted CSV File:

The extracted CSV file (crime.csv) is saved in a folder named after the dataset in the same directory as the script.

Meteostat HTML Dataset Extraction Documentation:

Meteostat Library:

- API Library

Python Function for Extraction:

Parameters Used:

- meteostation: ID of the meteostation.
- file_name: Name of the HTML file to save the data.
- start: Start date.
- end: End date.

Location of Extracted HTML File:

The extracted HTML file (boston_weather_data.html) is saved in a folder named after the file in the same directory as the script.

Boston Gov Website CSV Dataset Extraction Documentation:

URL for Dataset Access:

- URL: The URL used to access the dataset on the Boston Gov website: https://data.boston.gov/dataset/e63a37e1-be79-4722-89e6-9e7e2a3da6d1/resource/73c7e069-701f-4910-986d-b950f46c91a1/download/tmp8mntlmrz.csv

Python Functions for Extraction:

Parameters Used:

- url: URL of the dataset on the Boston Gov website.
- dataset_name: A descriptive name for the dataset (not used in the function).

Location of Extracted CSV File:

The extracted CSV file (ShootingsBostonGOV.csv) is saved in a folder named after the file in the same directory as the script.

Data Transformations Documentation:

Crime Dataset Transformations:

Loading Data:

- Load the crime dataset from a CSV file using the load_csv_to_dataframe function.

Column Projection:

- Project specific columns from the dataset using the project_columns function.

Boolean Conversion for Shooting Column:

- Convert the 'SHOOTING' column to boolean (True for 'Y', False for 'N') using the convert_shooting_to_boolean function.

Data Cleaning:

- Clean the crime data by:
    - Removing rows with missing or invalid location information and duplicate incident numbers using the clean_crime_data function.
    - Adding a surrogate key to the DataFrame using the add_surrogate_key function.
    - Removing columns that aren’t useful for further analysis

Column Renaming:

- Rename columns in the weather DataFrame using the rename_columns function.

Selection of Interesting Rows:

- Select rows based on specific conditions, such as Shooting, using the select_interesting_rows function.

CSV File Creation:

- Perform various CSV file creations based on the processed DataFrame using the new_csv function.

Weather Dataset Transformations:

Reading HTML File:

- Read an HTML file containing weather data using the read_weather_html_file function.

Column Renaming:

- Rename columns in the weather DataFrame using the rename_columns function.

Inner Join with Crime Data:

- Perform an inner join with the crime DataFrame based on the 'OCCURRED_ON_DATE' column using the join_dataframes function.

Selection of Interesting Rows:

- Select rows based on specific conditions, such as multiple victims, using the select_interesting_rows function.

Primary Key Addition:

- Add a primary key to the merged DataFrame using the add_primary_key function.

Saving Merged DataFrame to CSV:

- Save the resulting DataFrame to a CSV file using the save_dataframe_to_csv function.

Shootings Dataset Transformations:

Loading Data:

- Load the shootings dataset from a CSV file using the load_csv_to_dataframe function.

Column Renaming:

- Rename columns in the shootings DataFrame using the rename_columns function.

Boolean Conversion for Victims and Shootings:

- Convert columns related to victims and shootings to boolean (True for 'T'/'FATAL', False for 'F'/'NON-FATAL') using the convert_victims_to_boolean and convert_Shooting_to_boolean functions.

Selection of Interesting Rows:

- Select rows based on specific conditions, such as multiple victims and female gender, using the select_interesting_rows function.

Replacing NaN Values:

- Replace NaN values with 'unknown' in specified columns using the replace_nan_with_unknown function.

Primary Key Addition:

- Add a primary key to the DataFrame using the add_primary_key function.

Saving Processed DataFrame to CSV:

- Save the resulting DataFrame to a CSV file using the save_dataframe_to_csv function.
-

Data Flow Illustration:

Data Pipeline for Loading and Storing Processed Data into PostgreSQL Database:

Read Configuration File:

- Read database connection parameters from the provided JSON configuration file.

Read Data from Files:

    - Read data from CSV files into Pandas DataFrames.

  Execute DDL Queries:

    - Execute Data Definition Language (DDL) queries to create or drop tables as needed in the PostgreSQL database.

  Execute Insert Queries:

    - Execute insert queries to load data from Pandas DataFrames into respective tables in the database.

Key Components:

- JSON Configuration File: Contains database connection parameters.
- CSV Files: Source data files containing information about shootings, districts, offenses, locations, and crime weather.
- Pandas DataFrames: Data structures used to hold the data from CSV files.
- PostgreSQL Database: Destination database where the data is loaded after processing.

Processing Steps:

  Read Configuration File:

    - Reads the database connection parameters from the JSON configuration file.

  Read Data from Files:

    - Reads data from CSV files using Pandas.

  Execute DDL Queries:

    - Executes DDL queries to create or drop tables in the PostgreSQL database.
    - DDL queries include creating tables for shootings, districts, offenses, locations, and crime weather.

  Execute Insert Queries:

    - Executes insert queries to load data from Pandas DataFrames into the corresponding tables in the database.
    - Insert queries include inserting data into tables for shootings, districts, offenses, locations, and crime weather.

Outcome:

- Processed data from CSV files is successfully loaded into the PostgreSQL database, organized into respective tables based on the data content.

Data Loading:

Functions and Classes:

  read_config_file(config_file: str) -> Dict[str, Any]:

    - Description: Reads database connection parameters from a JSON configuration file.
    - Parameters:
        - config_file (str): Path to the JSON configuration file.
    - Returns:
        - Dict[str, Any]: Dictionary containing database connection parameters.

read_data_from_file(file_path: str) -> pd.DataFrame:
- Description: Reads data from a CSV file using Pandas.
- Parameters:
    - file_path (str): Path to the CSV file.
- Returns:
    - pd.DataFrame: DataFrame containing the data.

execute_ddl(conn_params: Dict[str, Any], ddl_statement: str) -> None:
- Description: Executes Data Definition Language (DDL) statements such as create, drop, or alter table.
- Parameters:
    - conn_params (dict): Connection parameters for the database.
    - ddl_statement (str): SQL query to execute.
- Returns:
    - None

execute_insert(conn_params: Dict[str, Any], insert_query: str, data: pd.DataFrame) -> None:
- Description: Executes insertion of data into the database.
- Parameters:
    - conn_params (dict): Connection parameters for the database.
    - insert_query (str): SQL query for insertion.
    - data (pd.DataFrame): DataFrame containing data to be inserted.
- Returns:
    - None

DimRegionsQueries (class):
- Description: Contains SQL queries related to the dim_regions table.
    - Table Drop Queries:
        - drop_table_crimes_weather_query
        - drop_table_Shootings_query
        - drop_table_district_query
        - drop_table_offense_query
        - drop_table_location_query
    - Table Creation Queries:
        - create_table_Shootings_query
        - create_table_district_query
        - create_table_offense_query
        - create_table_location_query
        - create_table_crimes_weather_query
    - Insertion Queries:
        - insert_crimes_weather_query
        - insert_shootings_query
        - insert_district_query
        - insert_offense_query
        - insert_location_query

main() (function):
- Description: Main function to execute database operations.