#### **1. ETL Project**

**File**: data\_extraction.py

**Modules**:

* extract\_from\_csv(file\_to\_process): Extracts data from a CSV file.
* extract\_from\_json(file\_to\_process): Extracts data from a JSON file.
* extract\_from\_xml(file\_to\_process): Extracts data from an XML file.
* extract(): Consolidates data extraction from multiple file formats.
* transform(data): Transforms the extracted data (e.g., converts units).
* load\_data(target\_file, transformed\_data): Loads the transformed data into a CSV file.
* log\_progress(message): Logs the progress of the ETL process.

**Workflow**:

1. Log the start of the ETL job.
2. Extract data from CSV, JSON, and XML files.
3. Log the completion of the extraction phase.
4. Transform the extracted data.
5. Log the completion of the transformation phase.
6. Load the transformed data into a CSV file.
7. Log the completion of the loading phase.
8. Log the end of the ETL job.

#### **2. Web Scraping and Data Extraction Project**

**File**: webscraping.py

**Modules**:

* requests.get(url): Fetches the HTML content of the web page.
* BeautifulSoup(html\_page, 'html.parser'): Parses the HTML content.
* data.find\_all('tbody'): Finds all table bodies in the HTML.
* row.find\_all('td'): Finds all table data cells in each row.
* pd.DataFrame(data\_dict, index=[0]): Creates a DataFrame from the extracted data.
* pd.concat([df, df1], ignore\_index=True): Concatenates the extracted data into a single DataFrame.
* df.to\_csv(csv\_path): Saves the DataFrame to a CSV file.
* df.to\_sql(table\_name, conn, if\_exists='replace', index=False): Saves the DataFrame to an SQLite table.

**Workflow**:W

1. Fetch the HTML content of the specified URL.
2. Parse the HTML content to extract film data.
3. Create a DataFrame from the extracted data.
4. Save the DataFrame to a CSV file.
5. Save the DataFrame to an SQLite database.

#### **3. SQLite Database Operations Project**

**File**: sqlite\_operations.py

**Modules**:

* sqlite3.connect('STAFF.db'): Connects to the SQLite database.
* pd.read\_csv(file\_path, names=attribute\_list): Reads data from a CSV file.
* df.to\_sql(table\_name, conn, if\_exists='replace', index=False): Loads the DataFrame into an SQLite table.
* pd.read\_sql(query\_statement, conn): Executes an SQL query and returns the result as a DataFrame.

**Workflow**:

1. Connect to the SQLite database.
2. Read data from CSV files.
3. Create and populate SQLite tables from the DataFrame.
4. Perform SQL queries and operations on the database.
5. Append new data to the tables.
6. Execute and print SQL queries for validation.

#### **4. Requirements Installation**

**File**: requirements.txt

**Description**:

* The requirements.txt file lists all the necessary Python libraries to run the scripts.
* To install the required libraries, navigate to the directory containing requirements.txt and run: ***pip install -r requirements.txt***