| **Project Title:** | **Enhanced ETL Workflow with Python, AWS S3, RDS, and Glue for Data Engineers** |
| --- | --- |
| **Technologies** | **Python, AWS , SQL** |
| **Data Format** | **CSV , JSON , XML** |

### 

### 

### **Project Orientation :** [**Project Orientation - Enhanced ETL**](https://drive.google.com/file/d/1qLHDXGkk-CWDbbpfwB7GI6N1FsLCu87b/view?usp=drive_link)

### **Introduction:**

This extended project will integrate AWS services into the ETL pipeline. Along with extracting, transforming, and loading data from CSV, JSON, and XML formats, we will:

* Use AWS S3 for storage and retrieval of raw and transformed data.
* Use AWS RDS (Relational Database Service) to load transformed data for further use.
* Optionally use AWS Glue to automate parts of the ETL process, such as schema inference and transformation.

### **Objectives:**

By the end of this project, you will:

1. Extract data from CSV, JSON, and XML files.
2. Transform the extracted data, including unit conversions.
3. Load the transformed data into AWS S3 and AWS RDS for persistence.
4. Optionally automate parts of the ETL process using AWS Glue.
5. Log the progress of ETL operations for monitoring purposes.

### **Dataset :**wget <https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0221EN-SkillsNetwork/labs/module%206/Lab%20-%20Extract%20Transform%20Load/data/source.zip>

### **Steps:**

#### **Step 1: Gather Data Files**

1. Open a terminal and download the dataset:
   * Use the wget command to download the dataset containing multiple file formats.

### wget <https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0221EN-SkillsNetwork/labs/module%206/Lab%20-%20Extract%20Transform%20Load/data/source.zip>

1. Unzip the downloaded file:

Expand-Archive -Path source.zip -DestinationPath ./unzipped\_folder

Note : You may use the unzip command to extract the contents of the downloaded zip file.

After completing this step, the project folder will have CSV, JSON, and XML files to work with.

#### **Step 2: AWS Setup**

1. **Create an S3 Bucket**
   * You will use this bucket to store raw data files (CSV, JSON, XML) and the transformed CSV file.
   * Example bucket name: my-etl-project-bucket.
2. **Set Up AWS RDS**
   * Create an RDS instance (MySQL/PostgreSQL) to store the transformed data.
   * Configure the security groups to allow access from your local machine or AWS Lambda functions.
3. **(Optional) Set Up AWS Glue**
   * Create a Glue crawler to automatically detect the schema of the raw files and transform data.
   * AWS Glue can be used to schedule and manage ETL jobs.

#### **Step 3: Import Libraries and Configure AWS Resources**

You'll need the following libraries for AWS integration:

* **boto3** for interacting with AWS S3 and RDS.
* **pandas** for data manipulation.
* **sqlalchemy** for interacting with RDS databases.

Set up the AWS credentials and connection details in your environment variables or AWS config files for secure access to S3 and RDS.

#### **Step 4: Define Functions for ETL with AWS Integration**

1. **Extract Data:**
   * Upload raw CSV, JSON, and XML files to the S3 bucket.
   * Download the files from S3 for processing in your ETL pipeline.
2. **Transform Data:**
   * Perform unit conversions (inches to meters, pounds to kilograms).
   * Clean and standardize the data.
3. **Load Data to AWS:**
   * **Load to RDS and S3:**After the data is transformed, store the resulting CSV file in the new S3 bucket with the name of transformed\_data.
   * **Load to RDS:**Connect to your RDS instance using sqlalchemy and pandas and load the data into a relational database table.
4. **(Optional) AWS Glue Transformation:**
   * You can use AWS Glue to automate the transformation step, where the raw files from S3 are processed, and the results are saved back to S3 or directly to RDS.

#### **Step 5: Logging**

* Use Python’s logging library to track the progress of the extraction, transformation, and loading phases.
* Save the logs in a text file and optionally upload them to S3 for centralized log storage.

#### **Step 6: Execution**

Follow the sequence of operations:

1. **Upload Raw Data to S3:**
   * Extract raw files from the ZIP and upload them to the S3 bucket.
2. **Extract and Transform Data:**
   * Download raw data files from S3, apply transformations, and save the results locally or directly to S3.
3. **Load Data into AWS Services:**
   * Upload the transformed CSV back to S3.
   * Load the final transformed data into AWS RDS using SQLAlchemy.
   * Also ensure RDS service will be accessible from your SQL workbench
4. **Monitor Logs:**
   * Ensure that the entire pipeline is logged, and store logs either locally or in S3.

### **Conclusion:**

By integrating AWS services like S3, RDS, and Glue, this project simulates a more realistic cloud-based ETL workflow. Data engineers can benefit from cloud storage and databases for better scalability and monitoring of ETL processes. Additionally, logging ensures full traceability, making it easier to debug and audit the pipeline.

* Provide a well-commented Python file containing the complete code for the project, organized into sections for data Pipeline and Analysis.
* Upload the same into github with a proper Readme file.
* Presentation on the entire project, including Problem Statement, Tools Used, Approaches and Insights Found.

**Evaluation Metrics:**

* Project evaluation will be done in the live session and have to showcase the approaches done to complete the project
* You are supposed to write a code in a modular fashion (in functional blocks)
* Secure code : follow the secure code guidelines
  + [Special session for Secure Code: Managing Credentials with Your Python Script(28/09/2024)](https://docs.google.com/document/d/1lJcPv5VGktqTl7SygS9Ot14zHCy7OjrXjoFSIDTFeMM/edit?usp=sharing)
* Maintainable: It can be maintained, even as your codebase grows.
* Portable: It works the same in every environment (operating system)
* You have to maintain your code on GitHub.(Mandatory)
* You have to keep your GitHub repo public so that anyone can check yourcode.(Mandatory)
* Proper readme file you have to maintain for any project development(Mandatory)
* Follow the coding standards:
  + https://www.python.org/dev/peps/pep-0008/
* You should include basic workflow and execution of the entire project in the readme file on GitHub

**GitHub Repo:**

The attached reference document will help you use GitHub effectively. - [Link](https://docs.google.com/presentation/d/1XHCbgUOqbcXNUyQ87vTlKdKRgAbBxtkA/edit?usp=sharing&ouid=106590842700357786537&rtpof=true&sd=true)

**Reference Material:**

Official Documentation:

* <https://www.python.org/doc/>

PROJECT DOUBT CLARIFICATION SESSION ( PROJECT AND CLASS DOUBTS)

About Session: The Project Doubt Clarification Session is a helpful resource for resolving questions and concerns about projects and class topics. It provides support in understanding project requirements, addressing code issues, and clarifying class concepts. The session aims to enhance comprehension and provide guidance to overcome challenges effectively.

Note: Book the slot at least before 12:00 Pm on the same day

Booking link :<https://forms.gle/NtkQ4UV9cBV7Ac3C8>

Session timing: 04:00 pm to 5:00 pm every saturday

LIVE EVALUATION SESSION (CAPSTONE AND FINAL PROJECT)

About Session: The Live Evaluation Session for Capstone and Final Projects allows participants to showcase their projects and receive real-time feedback for improvement. It assesses project quality and provides an opportunity for discussion and evaluation.

Note: This form will Open on Saturday (after 2 PM ) and Sunday Only on Every Week

Session Timing: Monday-Saturday (11:30AM to 1:00PM)

Booking link :<https://forms.gle/1m2Gsro41fLtZurRA>

