Project Assignment: Building an Incremental Data Processing Pipeline with AWS

Objective:

Develop a data engineering solution that simulates the real-world scenario of processing sales transaction data. This project focuses on implementing an incremental data ingestion pipeline into Amazon Redshift, using AWS Step Functions for orchestration. The pipeline will handle CSV files arriving in an S3 bucket, process the data with AWS Lambda, and perform upsert operations in Redshift to maintain the most current dataset.

Project Scope:

- **Data Description:** You will work with a sales transactions dataset that includes details such as transaction ID, date, product ID, quantity, price, and store location.
- **Data Ingestion**: CSV files will be simulated to arrive in an S3 bucket. Each file represents new transactions that may include new records or updates to existing records based on the transaction ID.

Tasks:

- **Data Modeling:**
    - Design a table schema for the sales transactions in Amazon Redshift. Consider how you will handle the upsert operation for new and existing records.
- **Mock Data Generation:**
    - Write a script to generate mock CSV files for the sales transactions dataset. Each file should contain multiple records of sales transactions. Consider using Python for script development.
    - Simulate the arrival of these CSV files into an S3 bucket. You may manually upload them or automate this process as an extension of your script.
    - Sample CSV Data

```
TransactionID,Date,ProductID,Quantity,Price,StoreLocation
1,2024-04-01,1001,2,20.00,New York
2,2024-04-01,1002,1,35.00,Los Angeles
3,2024-04-02,1003,1,15.00,Chicago
4,2024-04-02,1001,1,20.00,New York
5,2024-04-03,1004,2,40.00,Houston
```

- **AWS Step Functions Workflow:**
    - Design a state machine in AWS Step Functions that orchestrates the following tasks:

        - Triggering a Lambda function when a new CSV file is uploaded to S3.
        - Processing the CSV file with the Lambda function to read its contents.
        - Inserting new records or updating existing records in the Amazon Redshift table (upsert operation).

- **Lambda Function for Data Processing:**

    - Develop a Lambda function in Python that is triggered by S3 events. This function should:
        - Read the incoming CSV file from S3.
        - Parse the CSV file and transform the data if necessary.
        - Connect to Amazon Redshift to perform the upsert operations. Consider using the psycopg2 library or a similar tool for database interactions.

- **Data Upsert in Amazon Redshift:**

    - Implement the logic for data upsert (insert or update) in Redshift. Ensure that:
        - New records are inserted.
        - Existing records (based on the transaction ID) are updated with new information.

- **Documentation:**

    - Document your workflow design, including the Step Functions state machine, Lambda function logic, and Redshift upsert strategy.
    - Provide a brief explanation of your mock data generation process and the structure of your Redshift table schema.

Deliverables:

- Python script for mock data generation.
- AWS Step Functions state machine definition (JSON format).
- Source code for the AWS Lambda function.
- SQL script for the Redshift table schema and upsert logic.
- A comprehensive documentation file explaining your solution and design choices.