Assignment 2

Due: Feb 14th 2025, 03:59 pm EST

Required Attestation and Contribution Declaration

WE ATTEST THAT WE HAVEN'T USED ANY OTHER STUDENTS' WORK IN OUR ASSIGNMENT AND ABIDE BY THE POLICIES LISTED IN THE STUDENT HANDBOOK.

Contribution:
Example:

- Member 1: 33 1/3%
- Member 2: 33 1/3%
- Member 3: 33 1/3%

Links to GitHub tasks and tasks owned by each member.

Assignment Overview

Scenario:

You are working at a fintech company Findata Inc. that is building a master financial statement database to support analysts conducting fundamental analysis of US public companies. The database will be implemented using Snowflake and sourced from SEC Financial Statement Data Sets.

Tasks:

1. Data design
    - Scrape Data Links and retrieve all dataset links from the SEC Markets Data page.
    - Review the formatting of Financial Statements Data. (https://www.sec.gov/files/financial-statement-data-sets.pdf)
2. **Data Storage Design**
   Evaluate the following three approaches for storing financial statement data:
    - **Raw Staging:** Store the data as-is.
    - **JSON Transformation:** You could denormalize the data for faster data access. For example people have done transformations such as

SecFinancialStatementConverter to convert data into JSON . You plan to use this as a possible methodology to store json in Snowflake.
    - **Denormalized Fact Tables:** Transform SEC financial statement data into three denormalized fact tables (Balance Sheet, Income Statement, Cash Flow) with company identifiers (ticker, CIK), period identifiers (filing date, fiscal year, fiscal period), and key numeric fields.
    - Provide database schema designs for all three options.
    - Develop scripts and try in Snowflake on how to create tables and to upload datasets.
3. **Data Validation**
   Use Data Validation Tool (DVT) to validate data before loading it into four normalized tables (SUB, TAG, NUM, PRE) and before pivoting into the denormalized fact tables.
    - Validate file schema, headers, and data types.
    - Ensure uniqueness of key fields and integrity of foreign key relationships.
    - Confirm data adherence to domain constraints (e.g., enumerated values, booleans).
    - Check that numeric values fall within expected ranges and date fields are valid.
    - Ensure the ETL process correctly transforms and aggregates data (no duplicates or gaps).
    - Develop a Validation Design Document outlining your DVT validation strategy.
    - Implement Python functions for validation.

   Note:(I asked ChatGPT and Claude for some ideas for using DVT with the designed schemas and provided some details on tests I could do.This was just to illustrate the kinds of tests possible. You could use it for research but design your own strategy and tests with DVT and justify why this is comprehensive/sufficient)

4. **Operational Pipeline with Airflow**
   Design and implement three Airflow pipelines for data validation and staging. Use S3 for intermediate staging.
    - Create a job definition file including:
        - Year and quarter of data
        - Validation checks
        - Input/output staging areas
        - Configuration (S3, Snowflake credentials, etc.)
        - Processing methodology (JSON vs. RDBMS)
    - Run the pipeline on dataset Q4202{Your Team Number}.
    - Ensure pipelines can process datasets from any year/quarter.
5. **Post-Upload Testing**
   Develop a Testing Document outlining aspects such as:
    - Steps taken to ensure successful data upload across all three storage options.
    - Verification of data integrity in Snowflake.
    - Methods used to confirm pipeline execution.

    - Develop and run tests for the same.
    - Discuss why you think the data upload went fine and why your testing was sufficient!
6. **Testing and Recommendation**
    - Develop and deploy a Streamlit App that connects to your uploaded data.
    - Implement a FastAPI backend to connect with Snowflake.
    - Run SQL queries using SQLAlchemy and Snowflake SQL to illustrate data access.
    - Compare and discuss the advantages/disadvantages of Raw Storage, JSON, and RDBMS approaches.

Submission Requirements

1. GitHub Repository:
    - Project Summary, all code, tests, sql commands, logs and relevant information.
    - Use GitHub Issues to track tasks.
    - Include diagrams, fully documented Codelab, and a 5-minute video showcasing the solution.
    - Provide a link to the hosted application and backend services.
2. Documentation:
    - Comprehensive README.md detailing repository structure and usage.
    - Instructions for accessing and using deployed applications.
3. AI Use Disclosure:
    - Include AiUseDisclosure.md, specifying AI tools used and their purpose.

Resources:

1. **SEC Financial Statement Data Sets**
2. **SEC Financial Statement Converter (GitHub)**
3. **SEC Markets Data**