Assignment 3

Due: Feb 28th 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:

Your team is hired by Snowflake as Data engineers to help increase the adoption of Snowflake and Snowpark for data pipelines and will be working on creating Quickstarts for illustrating various Data engineering concepts for clients. You were given two quickstarts* to illustrate various concepts which you worked on last week(See part 1 of the assignment). Your mentor calls you to a meeting and notifies your team that there is a demo coming up in a week and he want a quick turnaround to create a fully functional quickstart illustration.

You have reviewed earlier quickstarts and are ready to go and are provided with example datasets (See team allocations below).

Data source allocation:

Team	Dataset	Metrics	Time Period	Source
1	FRED Data: Daily Index Data	Daily, Monthly returns for NASDAQ, S&P, DOW	Jan 2020 - Current	FRED Website
2	Cryptocurrency Prices	Daily, Monthly returns for BTC, Ethereum, Dogecoin	Jan 2020 - Current	Yahoo Finance, CoinGecko
3	FRED Data: Daily Currency Prices	Daily, Monthly returns for USD-EURO, USD-Pound, USD-Rupee	Jan 2020 - Current	FRED Website
4	FRED Data: U.S. Treasury Yield Curve	10-Year, 2-Year Treasury Yields	Jan 2020 - Current	FRED Website
5	CO2 Emissions – from Mauna Loa Observatory	Daily Percentage Change	All	NOAA CO2 Data
6	Weather Data - NOAA (5 Zip Codes in Boston Area)	Daily Temperature Changes	Jan 2020 - Current	NOAA Web Services
7	Traffic Congestion Index – from TomTom	Daily Congestion Level % Change & Avg Time Per 6 Miles Change	Snapshot for today, updated daily	TomTom Traffic Index (Scraped)

You are excited to get started and know you can use Google Codelabs to create the Quickstart and you already have Snowflake, Github and Github codespaces access. But you want some more direction and you ask your mentor on how to get started?

Your mentor is working on a sample example for DOW30 datasets and provides a specification he is putting together for her client. He says, you may want to create a specification something like this to get started! He says, **make sure you cover all this specification but for your dataset!**! After that I do task allocation to my team and use Github to manage all my tasks. He advises you to get started early since the deadline is Friday 4.00pm for the upcoming demo! You are super excited to get started and convene a meeting with your team in the Snell Library first thing this Sunday morning! Here is the specification your mentor shared for your review:

Quickstart Specification: Building an Incremental Data Pipeline for Dow 30 Stock Data Using Snowflake and Snowpark

Overview

This Quickstart will guide users through building a **Snowflake-based data pipeline** that **ingests, transforms, and updates Dow 30 stock data daily** from a publicly available source. The pipeline will be implemented using **Snowpark Python** and **Snowflake Notebooks**, leveraging Snowflake Tasks, Streams, and Stored Procedures to enable **incremental data updates** starting from **January 1st, 2020**.

Expected Outcome

By following this Quickstart:

- **Architecture diagram illustrating the workflow**
- **Users will have a fully functional Snowflake-based data pipeline**.
- **Dow 30 stock market data will be updated daily in an incremental fashion starting from January 1st, 2020**.
- **Analytics tables will provide stock trends, daily & weekly performance metrics returns**.
- **The architecture uses at least two UDFs (one SQL and one Python) and a stored procedure for updates.**
- **A Snowflake Notebook will facilitate Snowpark-based transformations and analytics.**
- **Testing framework and Jinja-based scripts will ensure DEV and PROD support.**

Architecture Overview

Pipeline Components

1. **External Data Source**:
    - The pipeline will pull **Dow 30 stock market data** from a **public API (e.g., Yahoo Finance, Alpha Vantage, or Quandl)**.
    - Data will be **ingested in CSV/JSON format** via Snowflake **External Stages** (S3 or Google buckets).
2. **Raw Data Storage** (`RAW_DOW30` Schema):
    - The ingested data will be stored in a **staging table (`RAW_DOW30_STAGING`)**.
    - Snowflake **Streams** will track new records for incremental updates.

3. **Data Harmonization & Transformation** (`HARMONIZED_DOW30` Schema):
    - **Transformation logic** using Snowpark Python:
        - Convert raw API response into a structured table.
        - Standardize timestamps and financial indicators.
        - Ensure data consistency and remove duplicates.
    - A **harmonized table (`DOW30_HARMONIZED`)** will store **cleaned and structured stock market data**.
    - **User-Defined Functions (UDFs):**
        - **SQL UDF:** A function to normalize currency exchange rates.
        - **Python UDF:** A function to calculate stock price volatility.
4. **Analytics & Aggregation** (`ANALYTICS_DOW30` Schema):
    - Precomputed analytics tables for:
        - **Daily & weekly performance metrics returns**.
    - **Stored Procedure (`UPDATE_DOW30_SP`)** to handle incremental updates and apply transformations.
5. **Task Orchestration & Automation**:
    - **Snowflake Tasks** will automate:
        - Data ingestion (`LOAD_DOW30_TASK`).
        - Daily updates (`UPDATE_DOW30_METRICS_TASK`).
    - **A Snowflake Notebook will be used** for data engineering and the usage of Snowpark Python.
    - **GitHub Actions Integration** for CI/CD of **Snowpark Python code**.
6. **Testing & Validation**:
    - **Implement unit tests** for UDFs and stored procedures.
    - **Use sample datasets** for validating pipeline correctness.
    - **Monitor task execution logs** to ensure proper scheduling and data updates.
7. **Environment Management with Jinja Templates**:
    - **Create Jinja-based scripts** to support DEV and PROD environments.
    - **Use parameterized configurations** to dynamically adjust Snowflake roles, schemas, and warehouses per environment.

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:

Lab 1:

https://quickstarts.snowflake.com/guide/data_engineering_pipelines_with_snowpark_python/index.html?index=..%2F..index#0

https://github.com/Snowflake-Labs/sfguide-data-engineering-with-snowpark-python?_fsi=1GCzoPPC

Lab 2:

https://www.snowflake.com/webinars/virtual-hands-on-labs/build-data-engineering-pipelines-using-snowpark-in-snowflake-notebooks-2025-02-12/

https://quickstarts.snowflake.com/guide/data_engineering_with_notebooks/index.html?index=..%2F..index#0

*Note: Since we only have one week, I have removed the market place requirements and the native app requirements quickstart from this part of the assignment. We will incorporate that in a future assignment