LAB 2

Project Report: Building an End-to-End Data Analytics Pipeline Using Snowflake, Airflow, dbt, and a BI Tool

San Jose State University

Members:

Elsa Rose: GitHub Link

Kruthika Virupakshappa: : GitHub Link

I. Team Introduction

Lab group: 9
Elsa Rose
Kruthika Virupakshappa

Our team consists of two passionate data science students specializing in applied analytics, data engineering, and financial modeling. With strong backgrounds in Python programming, cloud-based data pipelines, and statistical analysis, we collaborated to design, develop, and implement a robust stock price prediction system for this lab. Each team member contributed expertise in ETL pipeline development, machine learning modeling, and data visualization using industry-standard tools such as Apache Airflow, Docker, Python Colab, and Snowflake. Our complementary skills enabled us to tackle challenges in automation, data orchestration, and advanced time series forecasting, resulting in a comprehensive and scalable financial analytics solution for the Lab.

II. Introduction

Modern data engineering requires scalable tools that support reliable pipelines, modular transformations, and accessible analytics. Building on the foundation established in Lab 1, this lab extends the architecture into a complete ELT workflow by integrating Airflow for orchestration, Snowflake as the cloud data warehouse, dbt for transformations, data testing, and snapshots, and Preset for dashboarding and analytics. Through this lab, the key learning outcomes include gaining hands-on experience with Airflow ETL pipelines and workflow orchestration, constructing modular and reusable dbt models, executing dbt within Airflow as part of a scheduled pipeline, and designing BI dashboards that visualize insights derived from abstract analytical tables.

III. Problem Statement

The goal of this project is to design and implement a fully scheduled data analytics pipeline for historical stock price analysis. The pipeline must efficiently ingest raw stock price data into Snowflake through an ETL process, ensuring reliable and structured storage of market data. Once the raw data is available, dbt is used to build analytical ELT models that compute essential financial indicators such as moving averages, rolling sum and daily change. Apache Airflow serves as the central orchestrator, automating both the ETL ingestion and the ELT transformation workflows to ensure continuous, timely data updates. Finally, the transformed analytical datasets are delivered to a BI dashboard, enabling users to explore actionable insights and visualize key stock performance metrics through interactive charts and trend analyses.

IV. Solution Requirements and Specifications

Functional Requirements:

- Load raw stock price data from API to Snowflake using Airflow
- Transform raw data into analytical tables using dbt

- Schedule dbt tasks inside Airflow
- Visualize insights using a BI tool: Preset
- Ensure idempotency in ETL using SQL transactions + try/catch

Techincal Specifications

- Cloud Data Warehouse: Snowflake
- Orchestration Tool: Apache Airflow
- Transformation Tool: dbt
- BI / Visualization: Preset
- Version Control: GitHub
- Languages: SQL, Python, YAML

Data Specifications

- Dataset: Historical stock prices
- Fields: ticker, date, open, close, high, low, volume
- Granularity: Daily data
- History: Multi-year (depending on dataset)
- Pipeline Scheduling

Non-functional Requirements:

- Reproducible and version-controlled code (GitHub)
- Reliable scheduling and orchestration
- Query performance optimization
- Clear lineage between raw → staging → mart

V. Architectural Diagram

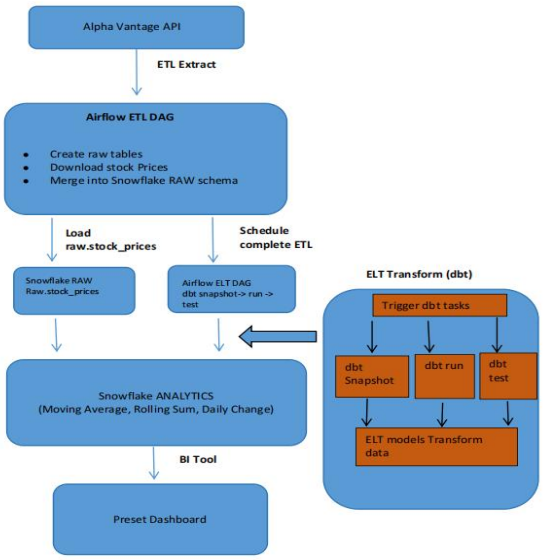


Fig: Architectural diagram of Airflow and ELT

Components Overview

1. Alpha Vantage API (Data Source)

✓ The pipeline begins with data extraction from the yFinance through Alpha Vantage API, which provides historical stock price data.

2. Airflow ETL DAG (Extract–Transform–Load)

✓ Apache Airflow orchestrates the ETL process through a dedicated DAG.

3. Snowflake RAW Layer

✓ The RAW layer stores unprocessed stock price data exactly as ingested.

4. Airflow Scheduling for ELT

✓ After the ETL task finishes, Airflow triggers a second DAG to perform ELT tasks using dbt.

5. BI Tool (Preset / Superset)

✓ The final layer of the pipeline is the BI visualization tool.
✓ Provides interactive filters for symbol and date range selection.

VI. Airflow Components and screenshots

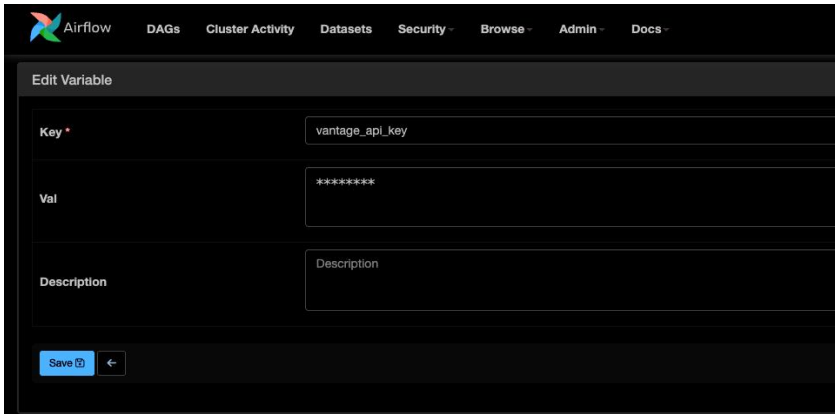


Fig: airflow variable initialization

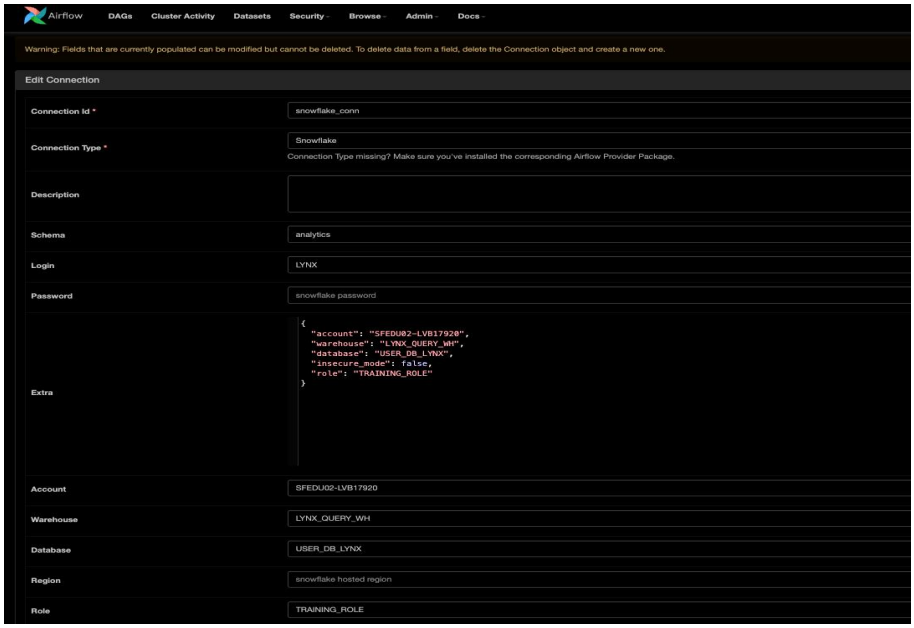


Fig: Airflow connection

Airflow: DAG stock_price_etl_vantage

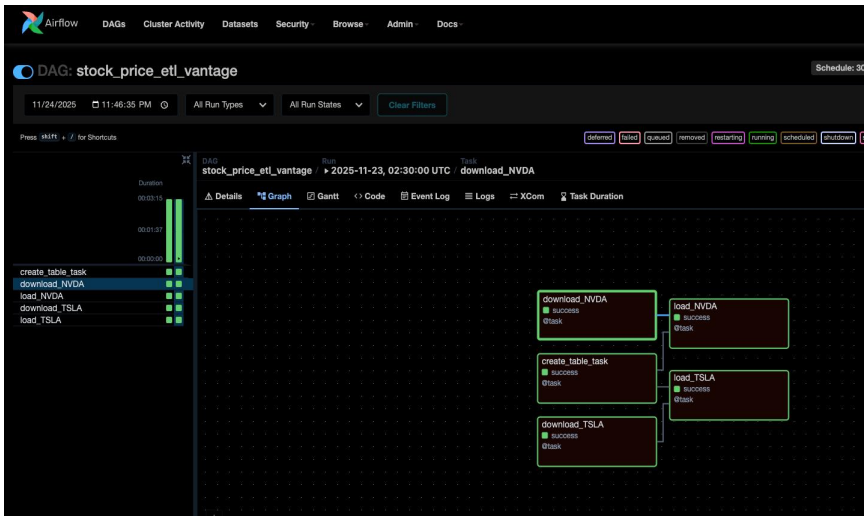


Fig: Representation of the stock_price_etl_vantage DAG

Components Overview

1. **DAG**

The DAG is named stock_price_etl_vantage and is scheduled to run daily at 2:30 AM. It starts on November 23, 2025, and does not catch up on missed runs. The tags=['ETL'] helps categorize it for easier monitoring and management.

2. Snowflake Connection and Table Creation

- return_snowflake_conn() uses the SnowflakeHook to connect to Snowflake using the connection ID snowflake_conn. This is a reusable function for all Snowflake operations in the DAG.

- create_table_task is a decorated task that executes a CREATE OR REPLACE TABLE

statement in Snowflake. The table schema includes columns for symbol, open, high, low, close, volume, and date, with a composite primary key on symbol and date.

3. Data Extraction from Alpha Vantage

download_stock_data is a task that fetches daily stock price data for a given symbol using the Alpha Vantage API. It retrieves JSON data, parses it, and returns a list of dictionaries containing the relevant fields. If the API returns an error or no data, it prints a warning and returns an empty list.

4. Data Loading into Snowflake

load_records is a task that takes the extracted data and loads it into the Snowflake table using a MERGE statement. For each record, it checks if a row with the same symbol and date exists. If it does, it updates the row; otherwise, it inserts a new row. This ensures data is up-to-date and avoids duplicates.

5. DAG Structure and Task Dependencies

The DAG first creates the table using setup = create_table_task(train_input_table).

For each symbol in the list ["NVDA", "TSLA"], it runs two tasks:

- download_stock_data to fetch the data for that symbol.
- load_records to insert or update the data in Snowflake.

The setup task is set as a dependency for all load_records tasks, ensuring the table is created before any data is loaded.

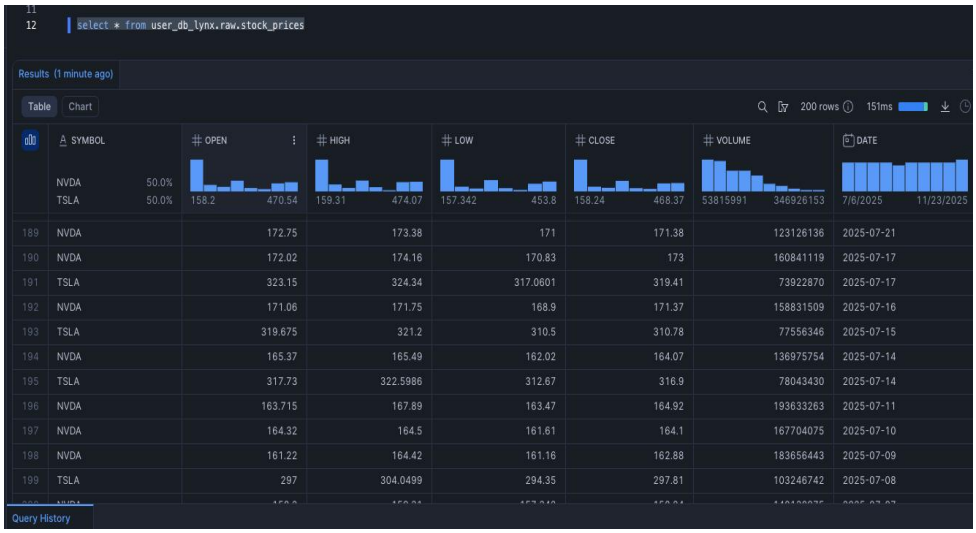


Fig: Screenshot of the stock prices from the snowflake

Airflow : DAG BuildELT_dbt

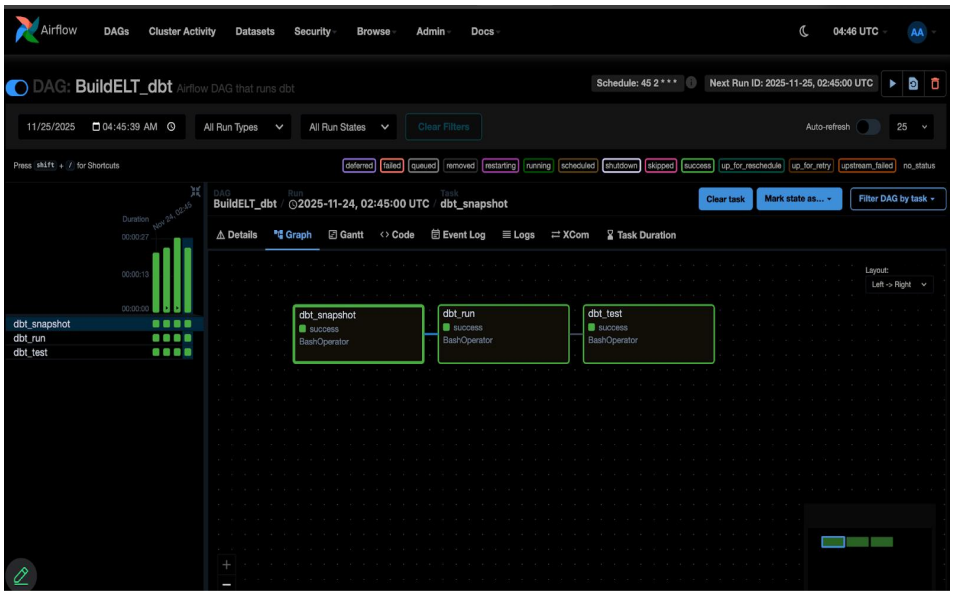


Fig: Representation of the BuildELT_DAG

Components Overview

1. DAG

The DAG is named "BuildELT_dbt" and set to start on November 23, 2025. It is scheduled to run daily at 2:45 AM using a cron expression. catchup is disabled so that backfill runs on startup are avoided. The DAG is not paused on creation, enabling it to run immediately once deployed.

2. Tasks in the DAG

There are three BashOperator tasks, each responsible for running dbt commands in sequence:

- ✓ **dbt_snapshot**: Runs dbt snapshot to capture state snapshots from the source data, useful for detecting data changes and building slowly changing dimension tables.

- ✓ **dbt_run**: Executes dbt run to perform data transformations defined in dbt models, applying SQL transformations on the Snowflake warehouse.

- ✓ **dbt_test**: Runs dbt test to validate data quality and consistency using tests defined in the dbt project.

3. Task Dependencies

- ✓ The tasks are chained in order: snapshot → run → test.

- ✓ This ensures the snapshot runs first to capture data state, followed by transformations, and finally tests to validate everything completed successfully.

Dbt_snapshot

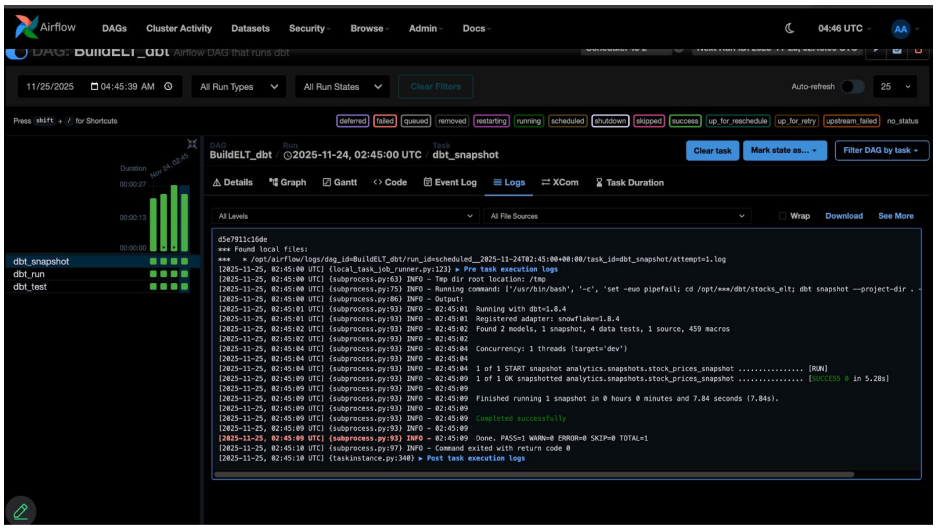


Fig: Screenshot of the dbt_snapshot logs

Dbt_run

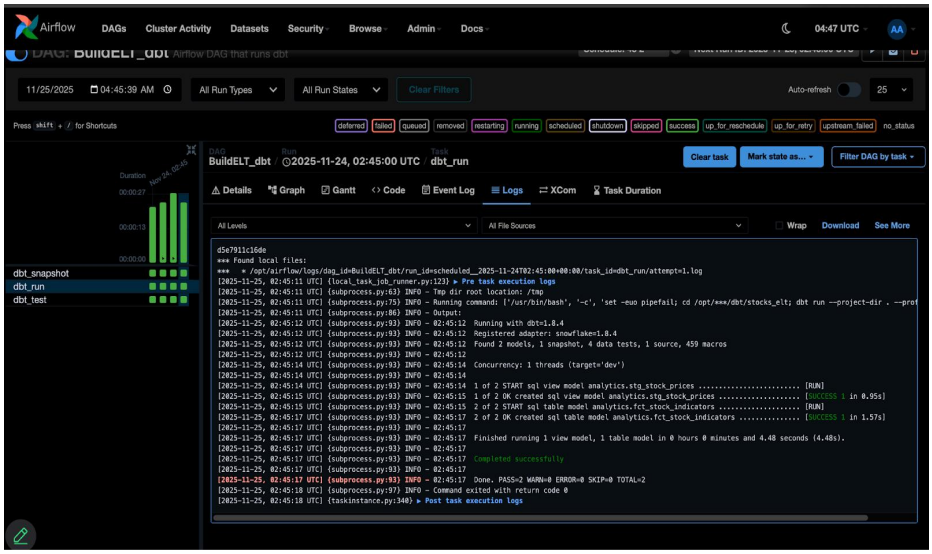


Fig: Screenshot of the dbt_run logs

Steps in Dbt_run process:

1. It creates a new column called prev_close, which captures the previous day's closing price to enable comparison of today's data versus yesterday's. This allows calculation of daily changes in stock price.

2. Additionally, it calculates a 20-day Simple Moving Average (SMA 20) by averaging the close price for the current row plus the prior 19 rows. This rolling average smooths out short-term volatility and highlights longer-term price trends.

3. It also computes a 50-day rolling sum (SUM 50) of the closing prices, summing the current close price plus the previous 49 days. This aggregation can indicate cumulative volume or price performance over a longer horizon.

4. Finally, the day_change column is calculated by subtracting prev_close from the current close, providing the exact dollar amount the stock went up or down on that day.
5.

Github link for the stock indicators: Link
Github link for the output:Link

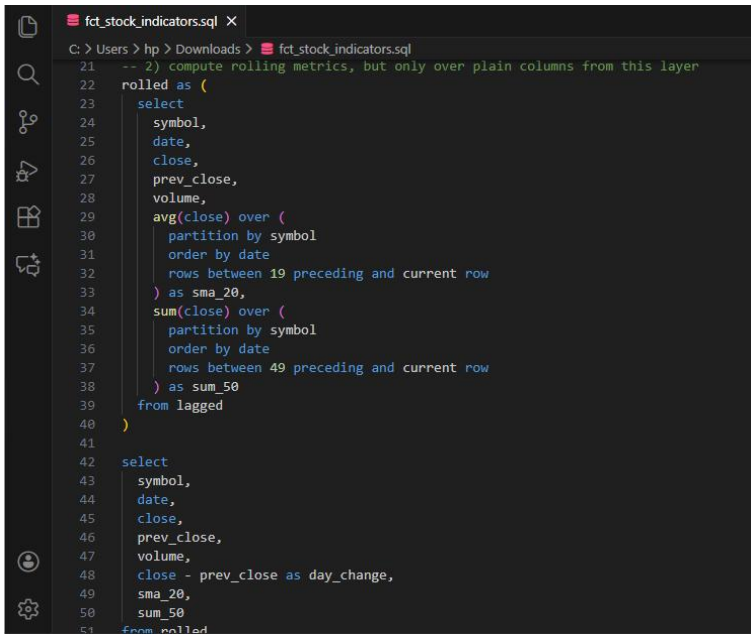
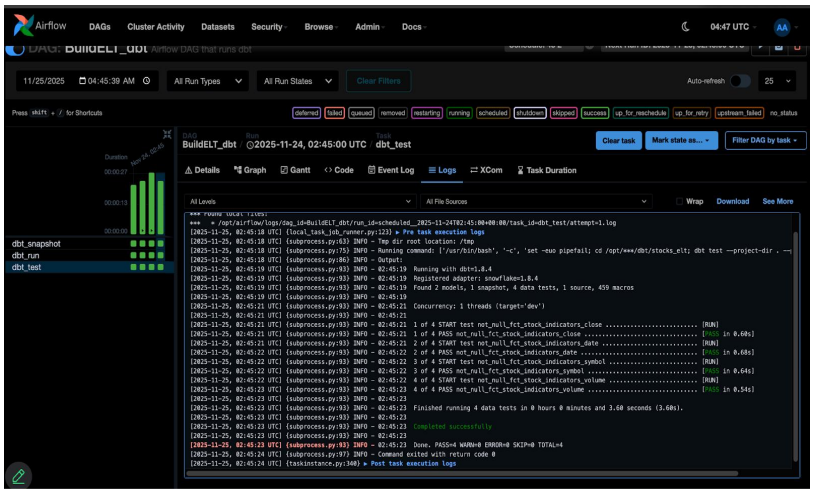


Fig: Computation of the rolling metrics

Fig: output for the calculations

Dbt_test



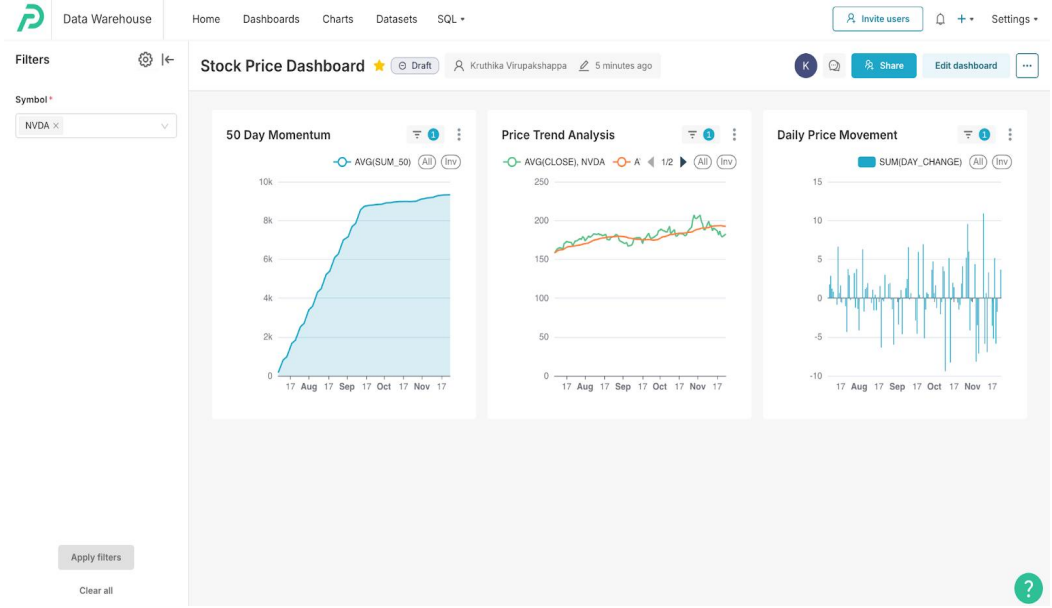
Steps in Dbt_test:

It is verifying that the following critical columns never contain NULL values

1. symbol: Every row must have a stock ticker
2. date: Every row must have a valid date.
3. close: Every row must have a closing price.
4. volume: Every row must have a volume count.

VII.BI tool: Preset

Stockprice dashboard for the NVIDIA:



Dashboard explantion for the NVIDIA

1. 50-Day Momentum (First Chart)

✓ This chart displays the momentum indicator for NVDA over time. Momentum measures

the rate at which the stock's price is increasing or decreasing.

- ✓ A sharp rise in momentum often indicates growing investor confidence and bullish sentiment.
- ✓ NVDA has maintained a strong pace of growth during this time, suggesting strong market demand and upward movement.

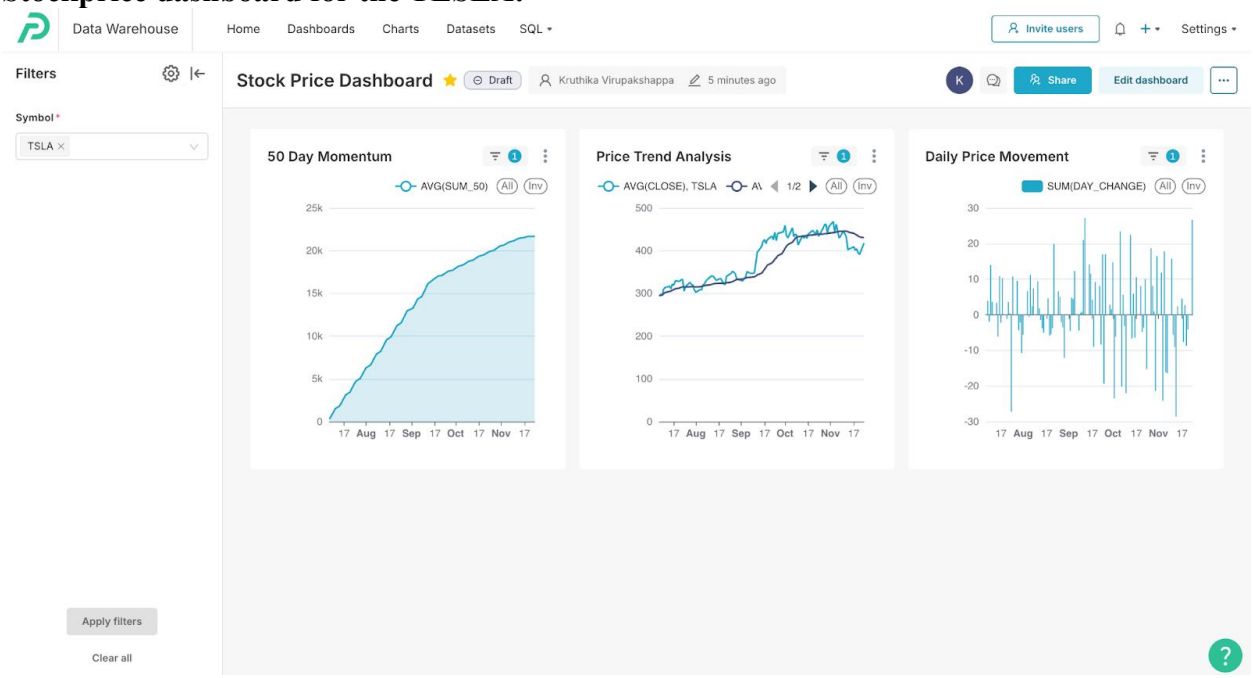
3. Price Trend Analysis (Second Chart)

- ✓ The chart illustrates an overall upward trend, indicating that NVDA's stock price increased steadily.
- ✓ NVDA experienced a period of stable price appreciation, consistent with positive financial performance or favorable market factors.

4. **Daily Price Movement (Third Chart)**

- ✓ This chart tracks day-to-day price changes for NVDA, showing how much the price moved up or down compared to the previous day.
- ✓ NVDA shows moderate daily fluctuations, typical for high-growth tech stocks. Occasional larger bars indicate days with significant news or market activity.

Stockprice dashboard for the TESLA:



Dashboard explanation for the TESLA

1. 50-Day Momentum (First Chart)

- ✓ Displays TSLA's 50-day momentum, a measure of how quickly the stock price is rising or falling.

- ✓ The chart shows a sharp upward movement, indicating strong positive momentum.

2. Price Trend Analysis (Second Chart)

- ✓ Shows the closing price trend of TSLA over time.
- ✓ The price steadily increases across the period.
- ✓ A small downward correction appears toward the end.
- ✓ Reflects long-term price behavior rather than day-to-day fluctuations.

3. Daily Price Movement (Third Chart)

- ✓ Displays day-to-day price changes compared to the previous trading day.
- ✓ Chart exhibits frequent and sharp fluctuations, indicating high volatility.

VIII. Conclusion:

This lab successfully implemented an end-to-end data analytics pipeline using Snowflake, Airflow, dbt, and Superset. Raw data was ingested through Airflow, transformed into analytical models using dbt, and visualized through interactive dashboards. The integration of ETL, ELT, workflow orchestration, and BI visualization demonstrates a structured and efficient modern data engineering workflow. The completed system provides a reliable foundation for scalable analytics and supports clear, data-driven insights.

References:

1. Snowflake Inc., *Snowflake Documentation*. Accessed: Jan. 2025. [Online]. Available: https://docs.snowflake.com/

3. Apache Software Foundation, *Apache Airflow Documentation*. Accessed: Jan. 2025. [Online]. Available: https://airflow.apache.org/docs/

3. dbt Labs, *dbt Documentation (Core & Cloud)*. Accessed: Jan. 2025. [Online]. Available: https://docs.getdbt.com/
