LAB- 2

Building an end to end Data Analytics using Snowflake, Airflow, dbt, and a BI tool

Shaila Reddy Kankanala[1] Sreenidhi Hayagreevan[2]

[1,2]*Masters In Data Analytics*
[1,2]*San Jose State University*

[1]*shailareddy.kankanala@sjsu.edu* [2]*sreenidhi.hayagreevan@sjsu.edu*

Abstract: *The main objective of the lab is to be able to create an integrated, seamless, automated data pipeline, including ELT and visualization, to analyze stock prices. This pipeline will automatically ingest, process, and analyze stock data by using a data warehouse in Snowflake, Apache Airflow as the orchestrator, dbt for data transformation, and a BI tool such as Superset or Tableau for visualization. Alpha Vantage API ingests data on a daily basis. Dbt will perform key transformations: moving average, price momentum, and RSI. It visualizes output to a dynamic dashboard to improve insights from historical trends and potentially future movements in stocks. This lab will effectively demonstrate how to integrate ELT workflows with real-time visualization for data-driven decision-making.*

Keywords— *Alpha Vantage API, Stock Price Prediction, Airflow, Snowflake, dbt (Data Build Tool),BI Tool (Preset), ETL (Extract, Transform, Load), ELT (Extract, Load, Transform), Scheduling and Orchestration with Airflow, Visualization and Analysis, End-to-End Data Analytics Pipeline*

I. INTRODUCTION

Full-scale data analytics pipelines need to be built as part of any organization that wants to drive actionable insights with data. During this lab, it will be shown how an integrated ETL and ELT framework smoothes the journey of data from source systems into analytical applications for speed and efficiency in decision-making. Airflow is used to automate and schedule the ingestion of data, which fits well with Snowflake, a cloud-based data storage solution that can scale with ease and rapidly retrieve data. Transformations in dbt further prepare the data to calculate, for example, Moving Averages and RSI, thus enriching the analytic value of the data. Results are then visualized dynamically using Preset or Superset for the ability to explore historical patterns and develop forward-looking insights. This stack proves a powerful and scalable real-time analytics pipeline to underpin proactive strategy at an organizational level.

II. REQUIREMENTS & SPECIFICATIONS

1. PROBLEM STATEMENT

Organizations operate in a data-dominated environment. They are tasked with the responsibility of making volumes of data available to their users in such a manner that it is accessible, trustworthy, and informative. For such knowledge to be converted into actionable intelligence, there is a pressing need for automated end-to-end pipelines in data analytics right from ingestions to transformation and visualization. The lab gives reason for integration between ETL with ELT workflows, which is better at handling the real-time data streams. While deeper analytics can be achieved by focusing on ELT in the pipeline through the robustness of data-driven insight and scale with business and operations growth.

2. OBJECTIVES

- Airflow Implementation of ETL: Perform the population of raw data tables through an ETL on Airflow so that ingesting data gets automated and the data is always ready to be transformed.

- Transformation of Data using DBT: Create abstract tables from change data and computations like Moving Averages, RSI and Price momentum using dbt to make this underlying data useful for analysts.

- Visualization with a BI Tool: Use Superset, Preset, or Tableau to build a dashboard on transformed data that will enable its end-users to understand what the key metrics, trends, and patterns mean.

- Scheduling in Airflow: Wire up dbt with Airflow by building a scheduled DAG that automates the transformation processes so that the BI tool always sees the latest data possible.

3. RELEVANCE

This lab builds upon the foundations laid in Lab 1, where Lab 1 established initial steps of ETL by ingesting data, this lab continues building onto it with the scope increased to cover the whole pipeline, integrating data transformation and its visualization. In this lab, the use of a dataset like stock prices will demonstrate how dbt can enable data against complex analytics, while Airflow can orchestrate data pipelines to provide holistic stock trend analysis with key performance indicators. The purpose of this lab is to further develop an understanding of ELT transformations, how to apply Business Intelligence tools, and other key competencies critical for making the data pipelines scalable to elicit high-impact, data-informed insights.

4. SPECIFICATIONS

1. System Architecture

   Using Snowflake, Airflow, dbt, and a BI tool like Superset or Preset or Tableau create an integrated ETL/ELT pipeline that simplifies the flow of data from ingest to visualize. Data sources include either extension of Lab 1 dataset or new historical/time-series data relevant for the analysis - for instance, stock prices or active users.

2. Components and Functionality

   Airflow ETL: Use Airflow to generate DAGs that fill the Snowflake with raw data tables based on secure connections and variable management. Focus on idempotent SQL transactions; add try/catch for the most robust data ingestion possible.

   ELT with dbt: Transform raw data with dbt into analytical tables filled with business-critical metrics. Then, conduct data quality tests, leverage snapshots, monitor, and version these data transformations. Schedule dbt tasks with ease in an Airflow DAG to automate the steps above.

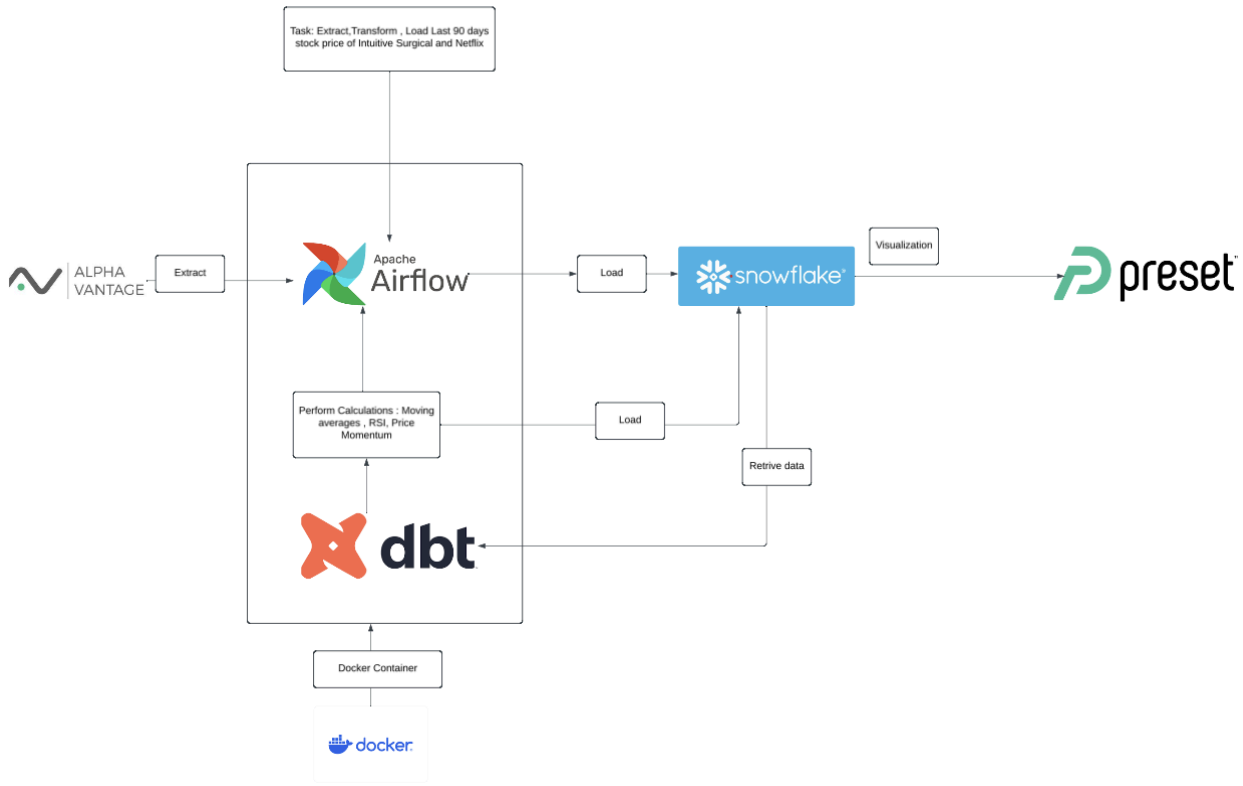
   Visualize it in a BI tool. Create an interactive dashboard that will visualize the trending and key metrics. You should include a minimum of two different charts with properties such as filters by date range that give flexibility to users for data exploration.

III . SYSTEM DESIGN

The diagram (Figure 1) illustrates the automated data analytics pipeline that is for processing and analyzing Intuitive Surgical and Netflix stock prices over the last 90 days. The pipeline is fed data from the Alpha Vantage API, which stocks daily stock price data. Apache Airflow orchestrates the ETL workflow-consolidation of task execution which includes data extraction, transformation, and loading into Snowflake, a cloud-based data warehouse. After loading it into Snowflake, it undergoes more transformations using dbt to compute moving averages, RSI, price momentum, and other calculations required to construct the meaningful metrics. Transformed data then flows into Preset, a BI tool for users to understand stock trends and key performance indicators through interactive visualizations. This pipeline thus allows efficient, scalable processing in enabling real-time insights to support data-driven decision-making.

Figure 1

End-to-End Data Analytics Pipeline for Stock Price Analysis



IV. TABLE STRUCTURE

ETL TABLE -LAB2

Fields	Attributes	Constraints
symbol	VARCHAR	NOT NULL
date	DATE	PRIMARY KEY, NOT NULL
open	NUMBER	NOT NULL
high	NUMBER	NOT NULL
low	NUMBER	NOT NULL
close	NUMBER	NOT NULL
volume	NUMBER	NOT NULL

ELT TABLE- STOCK_ANALYTICS

Fields	Attributes	Constraints
Symbol	VARCHAR	NOT NULL
Date	DATE	PRIMARY KEY, NOT NULL
Close	NUMBER	NOT NULL
Moving_average_7D	DECIMAL(10, 3)	NULL
Moving_average_30D	DECIMAL(10, 3)	NULL
RSI	DECIMAL(10, 3)	NULL
Price_Momentum_14D	DECIMAL(10, 4)	NULL

V. AIRFLOW IMPLEMENTATION

1) Airflow DAG Implementation

The pipeline was designed using Apache Airflow to automate data extraction, processing, and loading into a PostgreSQL database. The primary DAG consists of several tasks that are scheduled to run daily.

Figure 2

Airflow Web UI: Active and Running DAGs

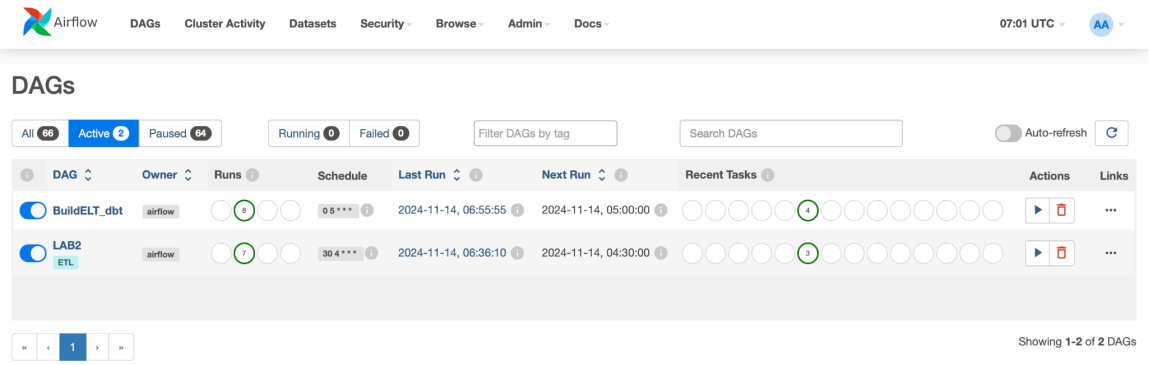


Figure 3

Airflow variables

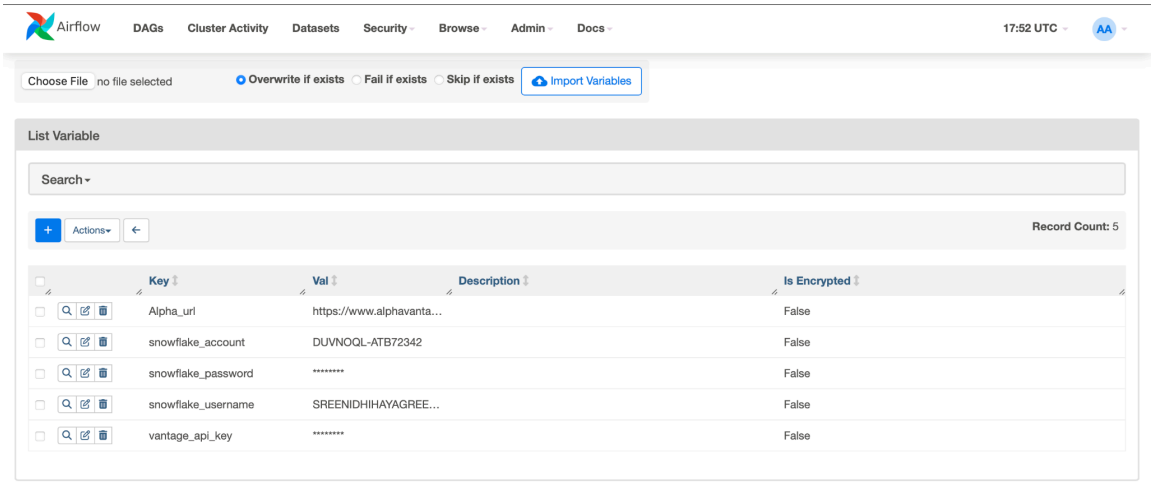
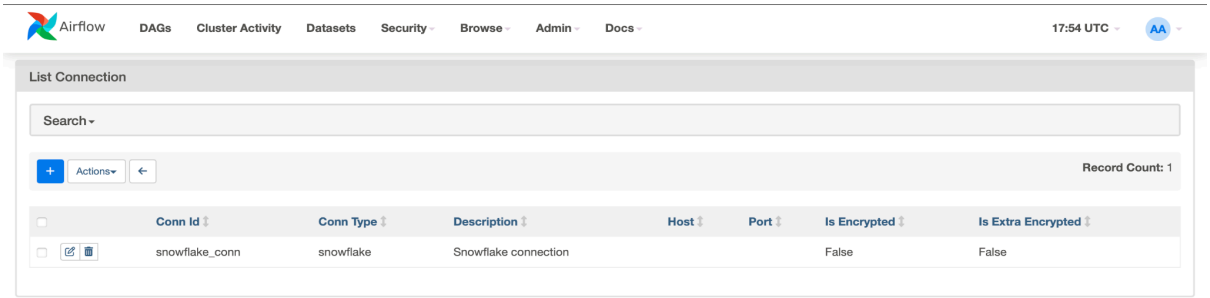


Figure 4

Airflow-snowflake connection



2) DAG Overview: The DAG consists of the following key tasks:

- Data Extraction: Fetching data from an external source (or generating synthetic data).
- Data Insertion: Inserting the data into the target DB, ensuring no duplicates using idempotency.
- Error Handling: Handling errors during SQL execution to maintain the integrity of the pipeline.

3) Airflow DAGs:

ETL DAG

This DAG in Apache Airflow defines the following data pipeline in ETL: extraction of stock information from an API and then its transformation and loading to a Snowflake database table. In this example, extract is a task that gets an API key and URL template from Airflow variables, uses them to request stock data for some symbols (ISRG, NFLX), and for each symbol processes the data returning the last 180 records. This `transform` job will reshape each record to include fields for symbol, date, open, high, low, close, and volume with a view to uniform loading. Lastly, the `load` job will connect to Snowflake, create a target table, or replace it with an insert of transformed records, taking care of idempotency-checks, preventing duplication of rows according to date and symbol. SQL transactions will ensure consistency in data through commit success or rollbacks if there is an error. The ETL pipeline is scheduled daily at 4:30 AM and has been highly reliable so far, with no breaks in the flow from API to database.

Figure 5

Airflow Web UI: ETL Pipeline Execution Overview

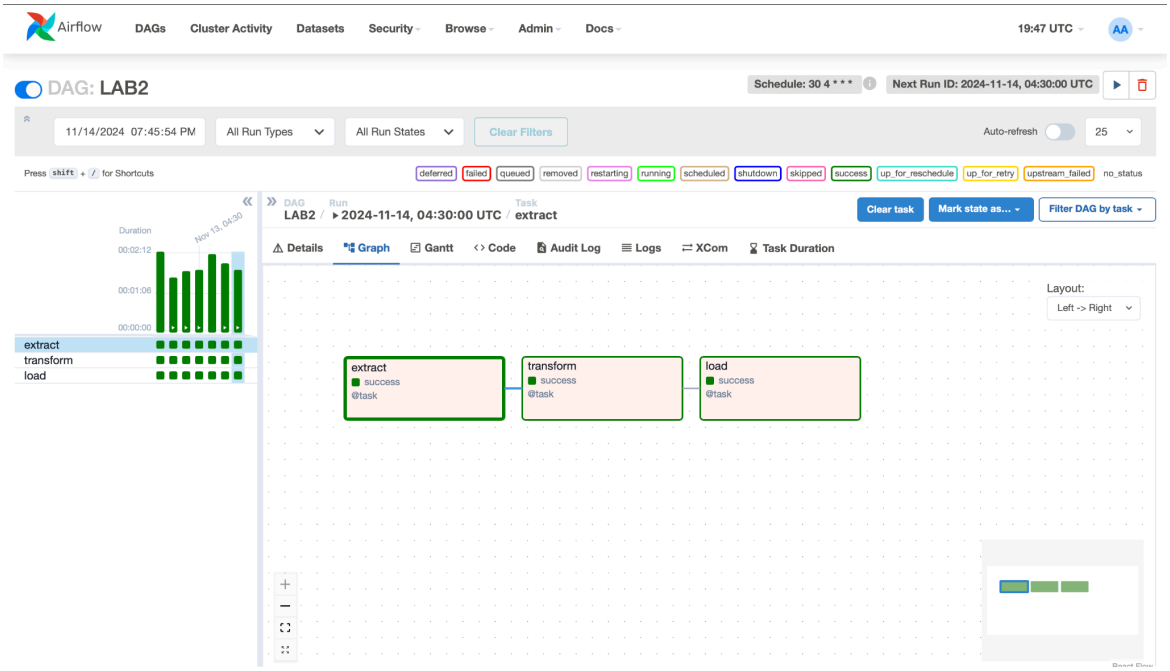
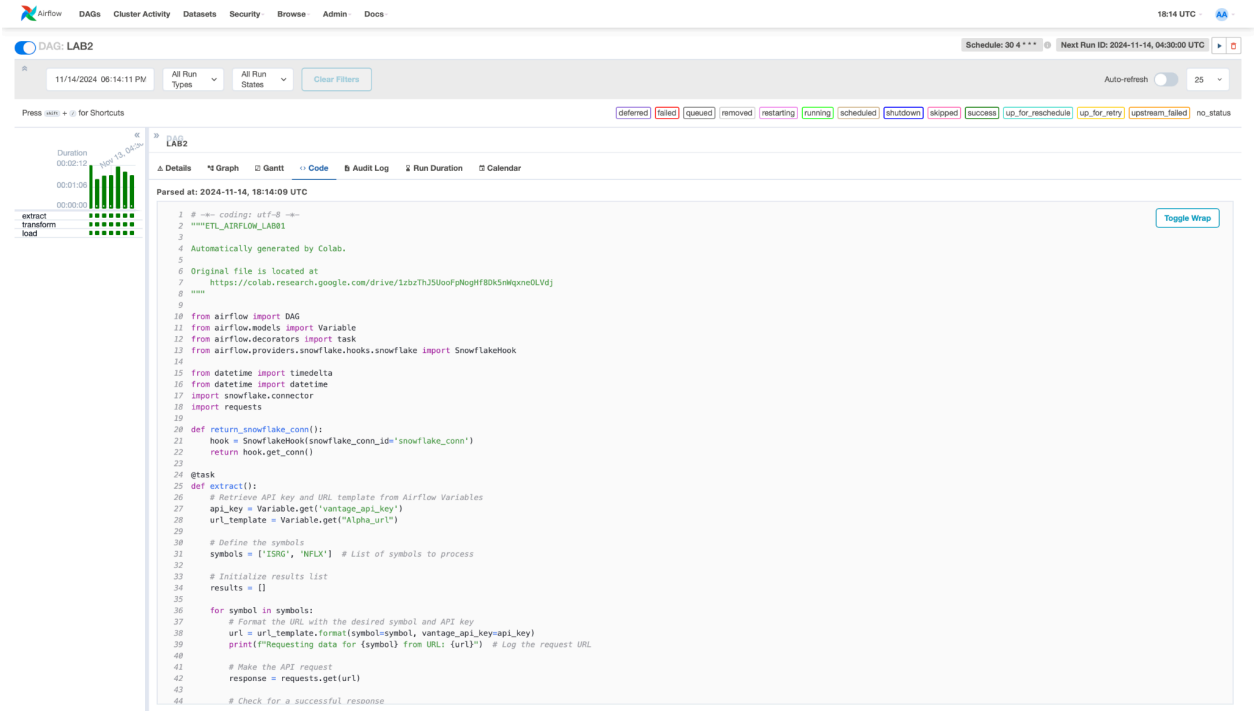


Figure 6

Airflow DAG Execution for ETL Pipeline code and log

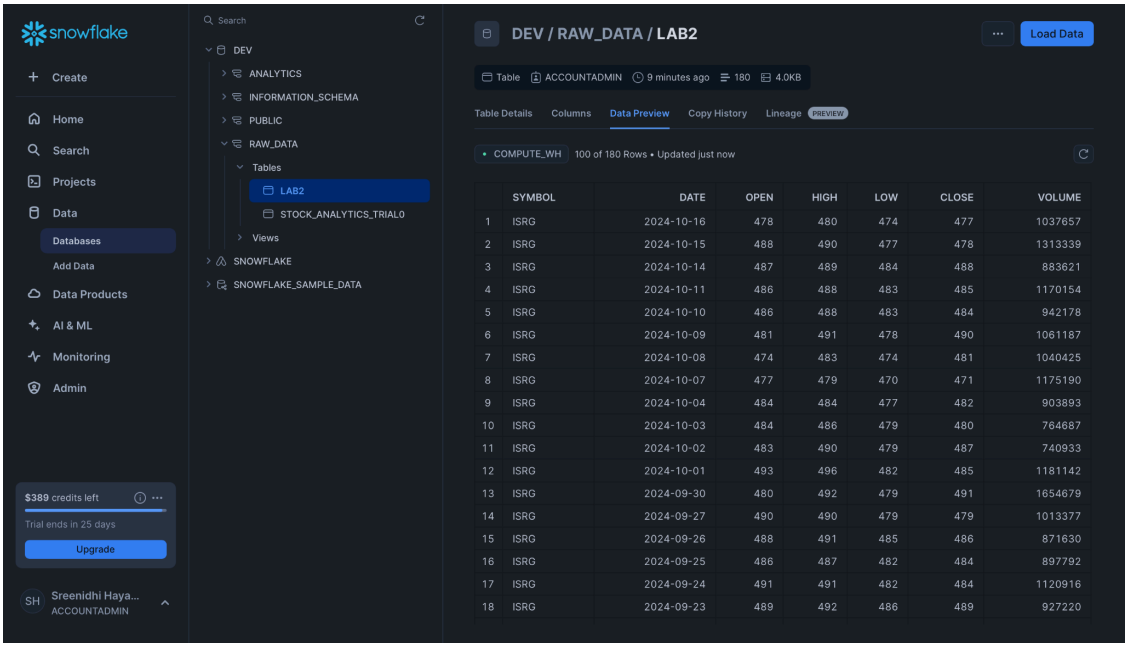


SQL Transaction for Data Verification in Snowflake

```
CREATE or REPLACE TABLE DEV.RAW_DATA.LAB2 (
      SYMBOL VARCHAR(16777216),
      DATE DATE,
      OPEN NUMBER(38,0),
      HIGH NUMBER(38,0),
      LOW NUMBER(38,0),
      CLOSE NUMBER(38,0),
      VOLUME NUMBER(38,0)
);
```

Figure 7

Final Loaded Dataset in Snowflake after ETL Pipeline Execution



	SYMBOL	DATE	OPEN	HIGH	LOW	CLOSE	VOLUME
1	ISRG	2024-10-16	478	480	474	477	1037657
2	ISRG	2024-10-15	488	490	477	478	1313339
3	ISRG	2024-10-14	487	489	484	488	883621
4	ISRG	2024-10-11	486	488	483	485	1170154
5	ISRG	2024-10-10	486	488	483	484	942178
6	ISRG	2024-10-09	481	491	478	490	1061187
7	ISRG	2024-10-08	474	483	474	481	1040425
8	ISRG	2024-10-07	477	479	470	471	1175190
9	ISRG	2024-10-04	484	484	477	482	903893
10	ISRG	2024-10-03	484	486	479	480	764687
11	ISRG	2024-10-02	483	490	479	487	740933
12	ISRG	2024-10-01	493	496	482	485	1181142
13	ISRG	2024-09-30	480	492	479	491	1654679
14	ISRG	2024-09-27	490	490	479	479	1013377
15	ISRG	2024-09-26	488	491	485	486	871630
16	ISRG	2024-09-25	486	487	482	484	897792
17	ISRG	2024-09-24	491	491	482	484	1120916
18	ISRG	2024-09-23	489	492	486	489	927220

ELT DAG

This DAG originally developed in Airflow will help automate the execution of DBT tasks in charge of the data transformation in Snowflake. It is set up to run once daily at 5:00 AM with three major tasks: dbt_run, dbt_test, and dbt_snapshot. These tasks will trigger the DBT commands for running models, testing the integrity of the data, and applying snapshots. The credentials of the Snowflake connection, like username, password, account, schema, database, role, and warehouse, are dynamically fetched from Airflow using BaseHook.get_connection. Moreover, the variables set the environment to execute the DBT with these credentials so that it gets integrated into Snowflake securely and smoothly. SQL transactions in DBT models are managed to keep the data integrity and manage consistency if something goes wrong while loading data. DAG shows the integration of Airflow, DBT, and Snowflake to have ELT workflows efficiently.

Figure 8

Airflow Web UI: ELT Pipeline Execution Overview

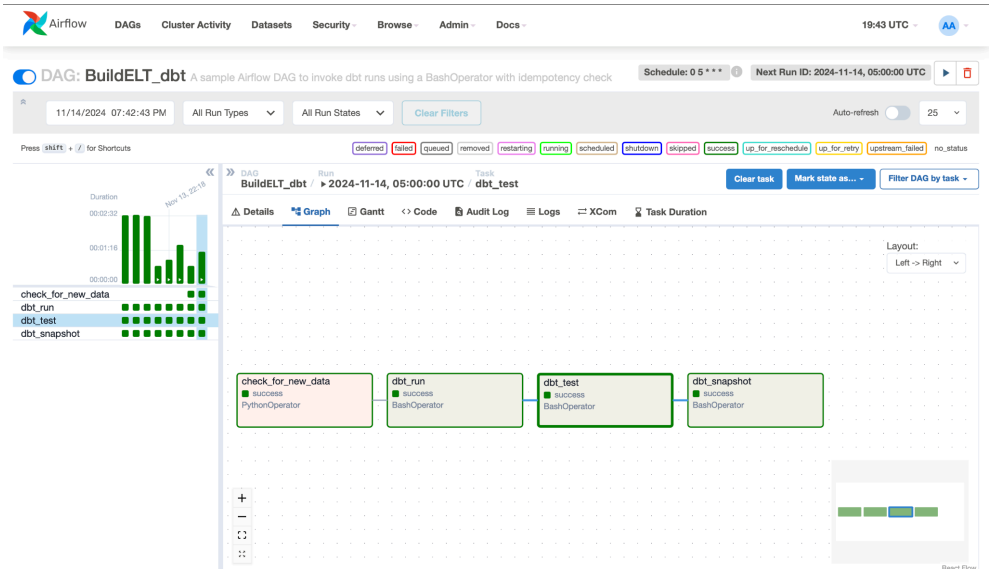
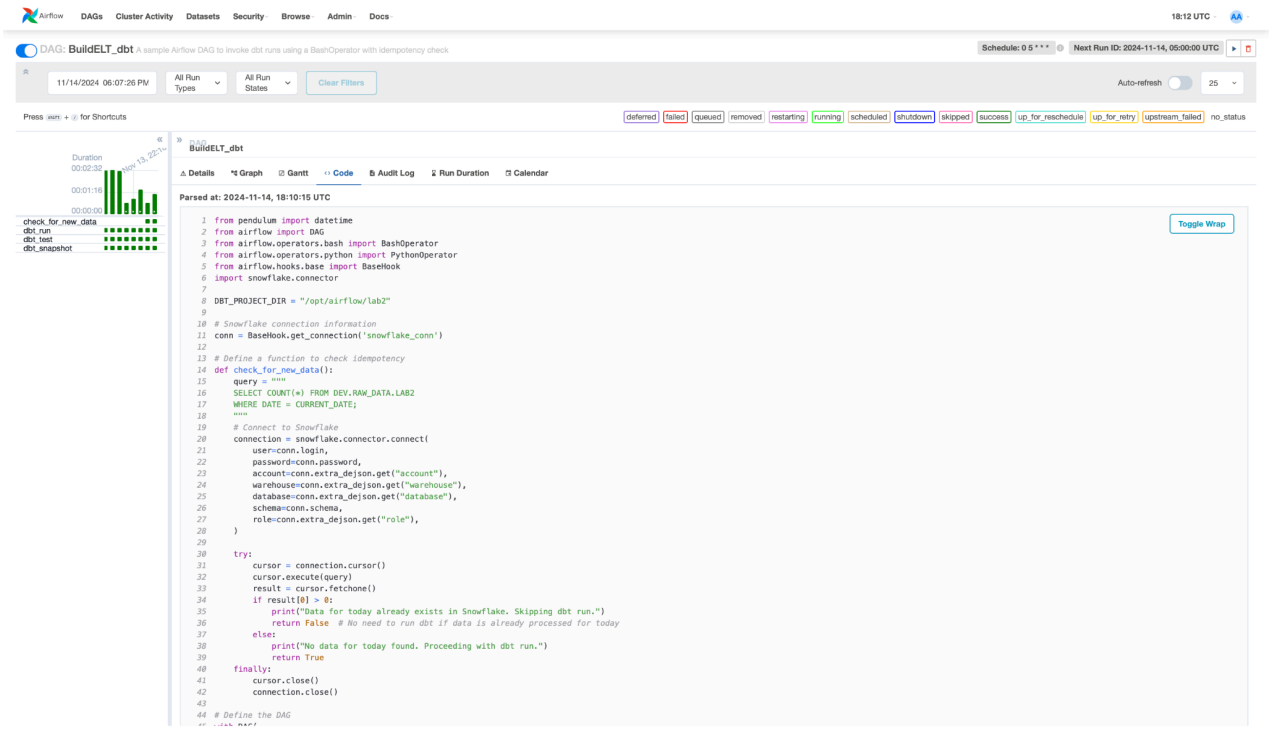


Figure 9

Airflow DAG Execution for ELT Pipeline

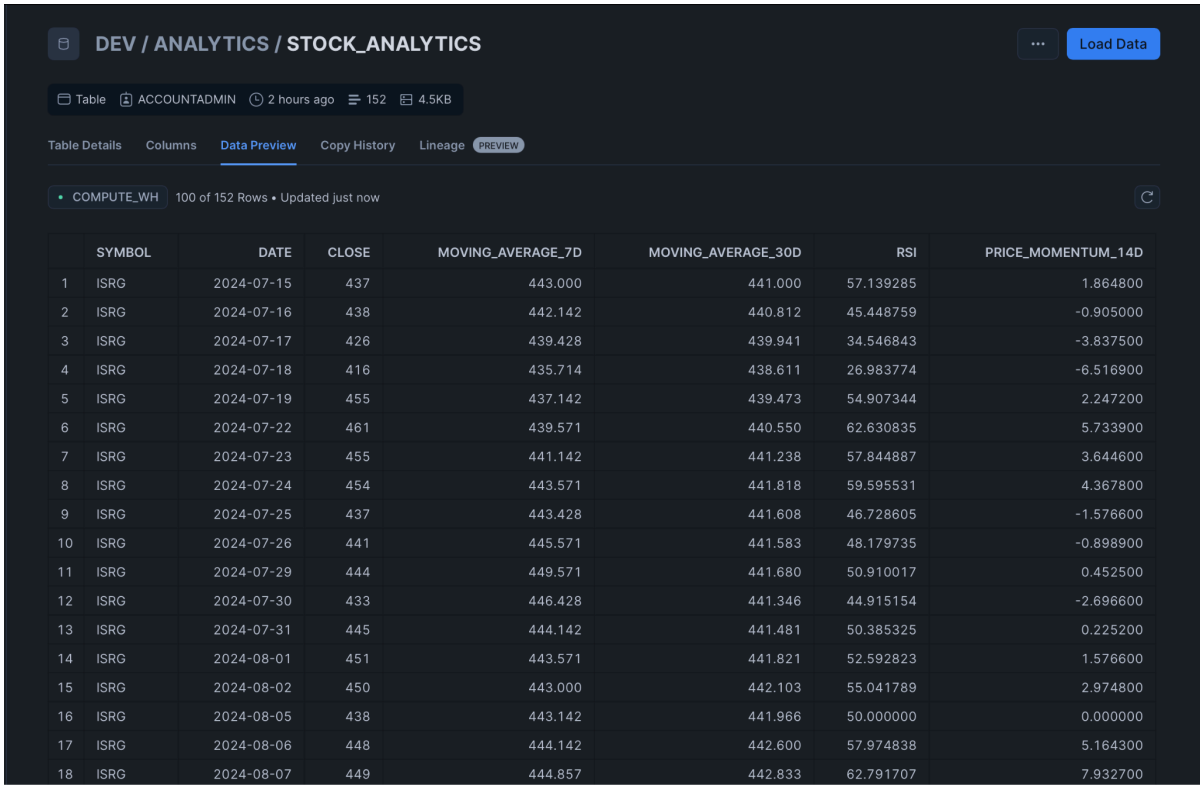


SQL Transaction for Data Verification in Snowflake

```
CREATE or REPLACE TRANSIENT TABLE DEV.ANALYTICS.STOCK_ANALYTICS (
      SYMBOL VARCHAR(16777216),
      DATE DATE,
      CLOSE NUMBER(38,0),
      MOVING_AVERAGE_7D NUMBER(38,3),
      MOVING_AVERAGE_30D NUMBER(38,3),
      RSI NUMBER(19,6),
      PRICE_MOMENTUM_14D NUMBER(38,6)
);
```

Figure 10

Final Loaded Dataset in Snowflake after ETL Pipeline Execution



	SYMBOL	DATE	CLOSE	MOVING_AVERAGE_7D	MOVING_AVERAGE_30D	RSI	PRICE_MOMENTUM_14D
1	ISRG	2024-07-15	437	443.000	441.000	57.139285	1.864800
2	ISRG	2024-07-16	438	442.142	440.812	45.448759	-0.905000
3	ISRG	2024-07-17	426	439.428	439.941	34.546843	-3.837500
4	ISRG	2024-07-18	416	435.714	438.611	26.983774	-6.516900
5	ISRG	2024-07-19	455	437.142	439.473	54.907344	2.247200
6	ISRG	2024-07-22	461	439.571	440.550	62.630835	5.733900
7	ISRG	2024-07-23	455	441.142	441.238	57.844887	3.644600
8	ISRG	2024-07-24	454	443.571	441.818	59.595531	4.367800
9	ISRG	2024-07-25	437	443.428	441.608	46.728605	-1.576600
10	ISRG	2024-07-26	441	445.571	441.583	48.179735	-0.898900
11	ISRG	2024-07-29	444	449.571	441.680	50.910017	0.452500
12	ISRG	2024-07-30	433	446.428	441.346	44.915154	-2.696600
13	ISRG	2024-07-31	445	444.142	441.481	50.385325	0.225200
14	ISRG	2024-08-01	451	443.571	441.821	52.592823	1.576600
15	ISRG	2024-08-02	450	443.000	442.103	55.041789	2.974800
16	ISRG	2024-08-05	438	443.142	441.966	50.000000	0.000000
17	ISRG	2024-08-06	448	444.142	442.600	57.974838	5.164300
18	ISRG	2024-08-07	449	444.857	442.833	62.791707	7.932700

4) Airflow Variables and Connections

Airflow variables and connections were used to handle configuration and secure credentials. The Postgres connection was set up using the Airflow UI under Admin > Connections to securely manage database credentials.

Implementation of Airflow connections and variables in **ETL**

```
def return_snowflake_conn():
    hook = SnowflakeHook(snowflake_conn_id='snowflake_conn')
    return hook.get_conn()
@task
def extract():
    # Retrieve API key and URL template from Airflow Variables
    api_key = Variable.get('vantage_api_key')
    url_template = Variable.get("Alpha_url")
    # Define the symbols
    symbols = ['ISRG', 'NFLX']  # List of symbols to process
    # Initialize results list
    results = []
    for symbol in symbols:
        # Format the URL with the desired symbol and API key
        url = url_template.format(symbol=symbol,
vantage_api_key=api_key)
        print(f"Requesting data for {symbol} from URL: {url}")
# Log the request URL
```

Implementation of Airflow connections and variables in **ELT**

```
conn = BaseHook.get_connection('snowflake_conn')
with DAG(
    "BuildELT_dbt",
    start_date=datetime(2024, 11, 10),  # Updated start date to Nov
10, 2024
    description="A sample Airflow DAG to invoke dbt runs using a
BashOperator",
    schedule_interval="0 5 * * *",  # Scheduled daily at 5:00 AM
    catchup=True,  # Enables catchup
    default_args={
        "env": {
            "DBT_USER": conn.login,
            "DBT_PASSWORD": conn.password,
            "DBT_ACCOUNT": conn.extra_dejson.get("account"),
            "DBT_SCHEMA": conn.schema,
            "DBT_DATABASE": conn.extra_dejson.get("database"),
            "DBT_ROLE": conn.extra_dejson.get("role"),
            "DBT_WAREHOUSE": conn.extra_dejson.get("warehouse"),
```

```
            "DBT_TYPE": "snowflake"
        }
    },
```

VI. Dbt IMPLEMENTATION

In this section, we provide the essential DBT project components that implement data models, perform tests, and manage snapshots. The DBT models will let us define transformations to take raw data in forms that are usable to an end user. This logic of transformation is written in SQL, hence it is pretty straightforward to implement and maintain.

Models:

Input model:

```
SELECT
    symbol,
    date,
    close
FROM DEV.RAW_DATA.LAB2
```

Output model:

```
SELECT
    ma.symbol,
    ma.date,
    ma.close,
    ma.moving_average_7d,
    ma.moving_average_30d,
    rc.rsi,
    (pm.close - pm.prev_close_14) / pm.prev_close_14 * 100 AS
price_momentum_14d
FROM
    (SELECT
        symbol,
        date,
        close,
        AVG(close) OVER (PARTITION BY symbol ORDER BY date ROWS BETWEEN 6
PRECEDING AND CURRENT ROW) AS moving_average_7d,
        AVG(close) OVER (PARTITION BY symbol ORDER BY date ROWS BETWEEN 29
PRECEDING AND CURRENT ROW) AS moving_average_30d
    FROM DEV.ANALYTICS.STOCK_ABSTRACT_VIEW) AS ma
```

```
JOIN
    (SELECT
        symbol,
        date,
        CASE
            WHEN avg_loss = 0 THEN 100
            WHEN avg_gain = 0 THEN 0
            ELSE 100 - (100 / (1 + (avg_gain / avg_loss)))
        END AS rsi
    FROM (
        SELECT
            symbol,
            date,
            AVG(CASE WHEN delta > 0 THEN delta ELSE 0 END) OVER (PARTITION
BY symbol ORDER BY date ROWS BETWEEN 13 PRECEDING AND CURRENT ROW) AS
avg_gain,
            AVG(CASE WHEN delta < 0 THEN ABS(delta) ELSE 0 END) OVER
(PARTITION BY symbol ORDER BY date ROWS BETWEEN 13 PRECEDING AND CURRENT
ROW) AS avg_loss
        FROM (
            SELECT
                symbol,
                date,
                close - LAG(close) OVER (PARTITION BY symbol ORDER BY
date) AS delta
            FROM DEV.ANALYTICS.STOCK_ABSTRACT_VIEW
        )
    )) AS rc ON ma.symbol = rc.symbol AND ma.date = rc.date
JOIN
    (SELECT
        symbol,
        date,
        close,
        LAG(close, 14) OVER (PARTITION BY symbol ORDER BY date) AS
prev_close_14
    FROM DEV.ANALYTICS.STOCK_ABSTRACT_VIEW) AS pm ON ma.symbol = pm.symbol
AND ma.date = pm.date
WHERE pm.prev_close_14 IS NOT NULL
```

Figure 11

Screenshot of dbt debug-successful connection of dbt and snowflake

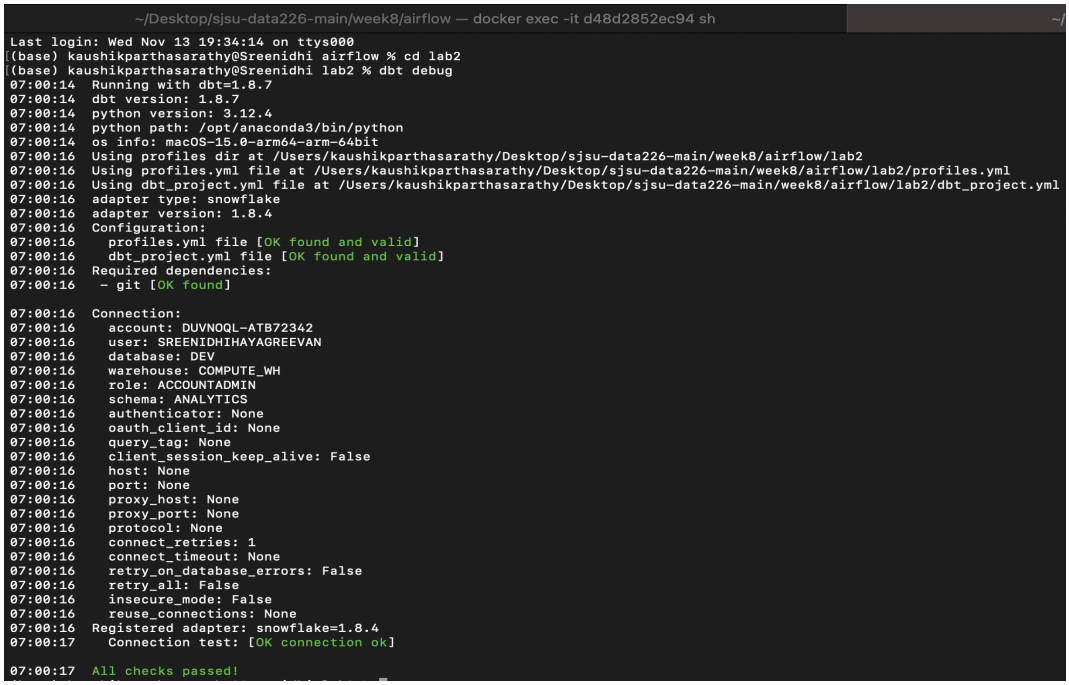


Figure 12

Screenshot of dbt run-executing all model successfully via terminal

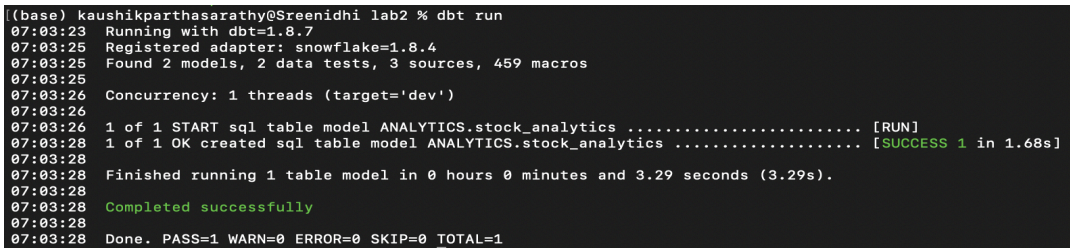


Figure 13

Screenshot of dbt test successfully completed via terminal

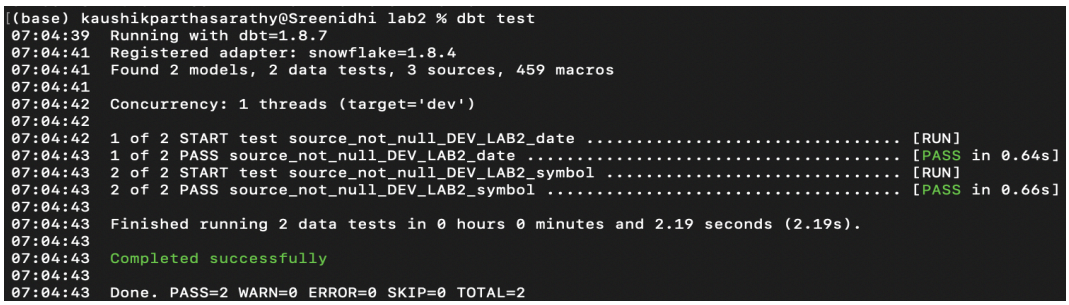


Figure 14

Screenshot of dbt snapshot successfully completed via terminal

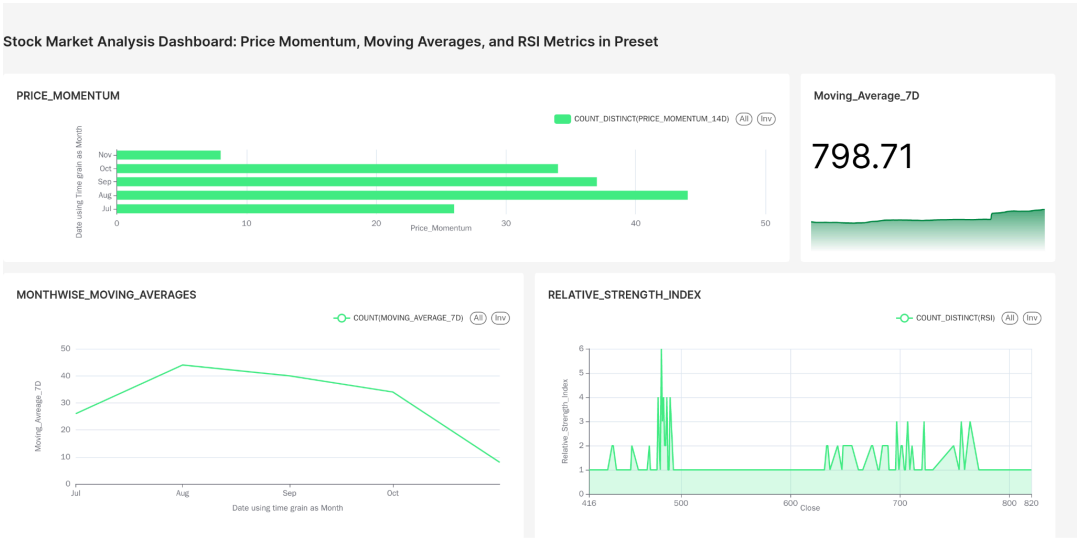
```
(base) kaushikparthasarathy@Sreenidhi lab2 % dbt snapshot
18:42:29  Running with dbt=1.8.7
18:42:31  Registered adapter: snowflake=1.8.4
18:42:32  Found 2 models, 2 data tests, 1 snapshot, 3 sources, 459 macros
18:42:32
18:42:34  Concurrency: 1 threads (target='dev')
18:42:34
18:42:34  1 of 1 START snapshot dev.raw_data.lab2_snapshot ..............................  [RUN]
18:42:36  1 of 1 OK snapshotted dev.raw_data.lab2_snapshot ..............................  [SUCCESS 1 in 2.21s]
18:42:36
18:42:36  Finished running 1 snapshot in 0 hours 0 minutes and 4.64 seconds (4.64s).
18:42:36
18:42:36  Completed successfully
18:42:36
18:42:36  Done. PASS=1 WARN=0 ERROR=0 SKIP=0 TOTAL=1
(base) kaushikparthasarathy@Sreenidhi lab2 %
```

VII. BI TOOL (PRESET)

This dashboard represents an overview of the stock market through the key metrics, which include the price momentum, moving averages, and relative strength index. It consists of many panels, where each panel presents different aspects of the movement in the market.

Figure 15

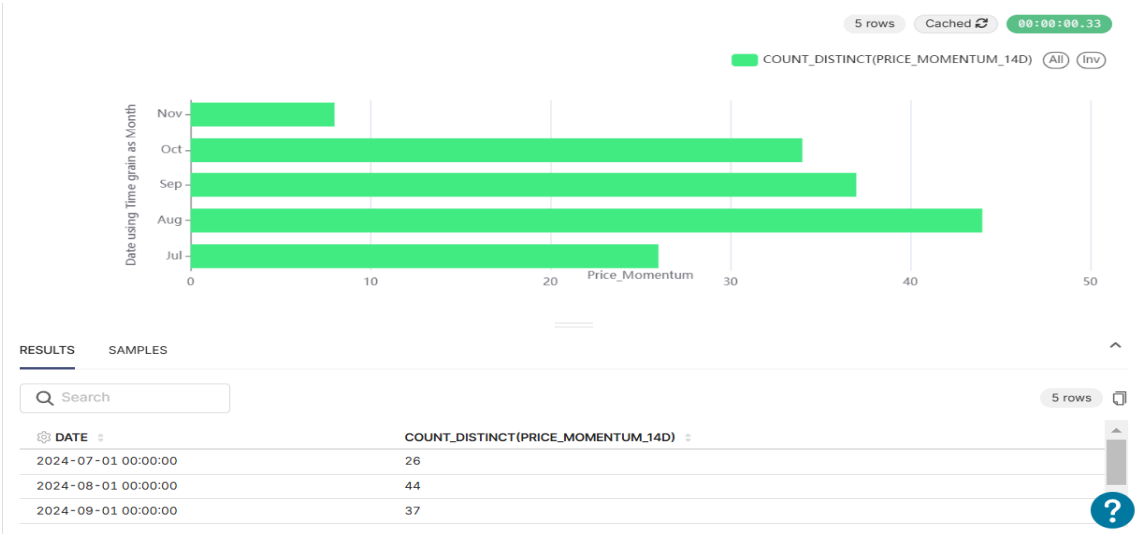
Stock Market Analysis Dashboard



1. Price Momentum: This is the price momentum bar chart for various months. It shows the count of distinct values for 14-day price momentum in such a time-series manner that users can view changes over time. The chart below will give a rough idea of how much momentum is happening in which month-to show the high and low months of trading activities.

Figure 16

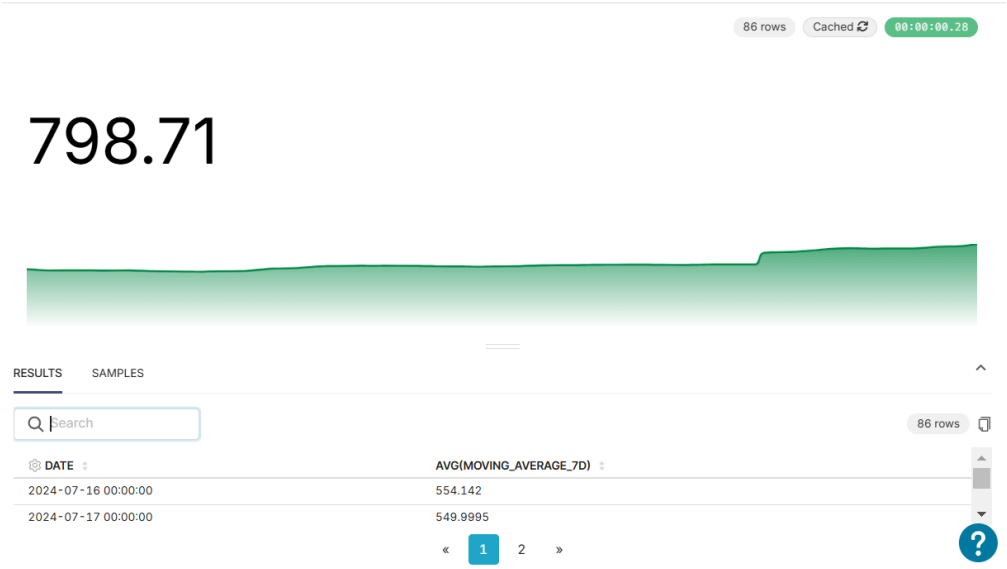
Monthly 14-Day Price Momentum Count



2. 7-Day Moving Average: This is a single-value metric that illustrates the last observed value of the 7-day moving average, which is 798.71, and a snapshot of the recent average price. The small area chart beside it reflects how this average has changed over time and may provide insight into the general trend of stock prices in the short term.

Figure 17

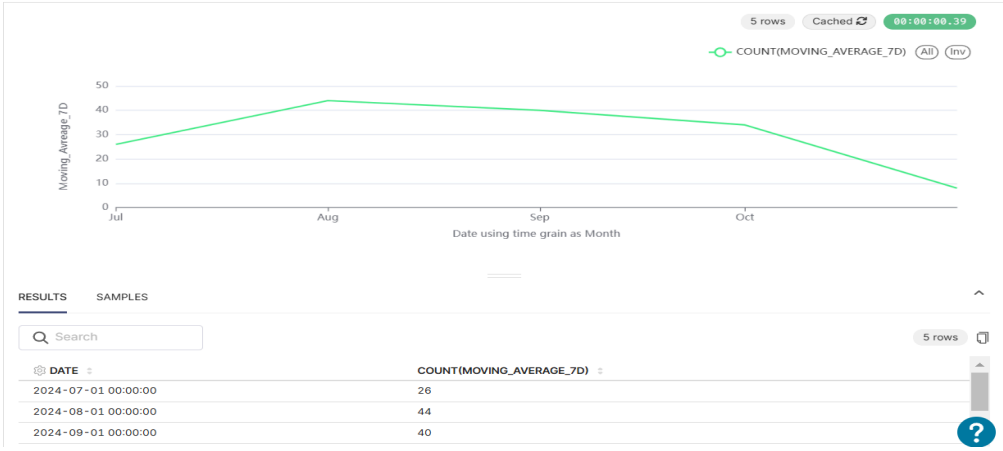
7-Day Moving Average Trend Over Time



3. Monthwise Moving Averages: The chart plots a line chart against the count of the 7-day moving averages by month. It serves to observe month-over-month changes in order to determine whether stock prices are, on average, increasing or decreasing on a short-term basis.

Figure 18

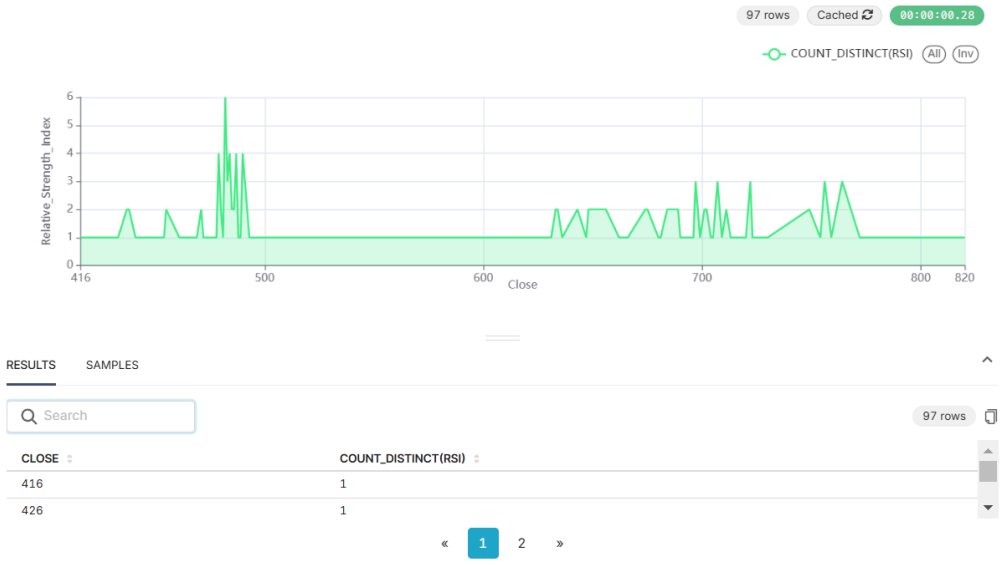
Monthly Moving Averages over time



4. Relative Strength Index: This line chart graphs RSI values against stock closing prices. The user can decide whether the stocks are overbought or oversold. In the RSI chart, it will be easier to detect a probable reversal or continuation trend because of its peaks and troughs, which might give a clue for trading decisions.

Figure 19

Relative Strength Index (RSI) Count Across Closing Prices



This will yield an overall easy-to-view, aesthetically pleasing dashboard of important stock market metrics for monitoring short-term trends in Preset and making investment decisions based on a set of user-friendly and interconnected interactive visualizations.

VIII. GITHUB REPOSITORY

Team member 1-Shaila Reddy Kankanala: https://github.com/Shailareddy2716/DATA226-LAB-02

Team member 2-Sreenidhi Hayagreevan: https://github.com/SreenidhiHayagreevan/data226-lab2

IX. CONCLUSION

This lab successfully demonstrates the creation of an end-to-end data analytics pipeline that automates the ingestion, transformation, and visualization of stock data, enabling real-time insights into stock price trends and key performance indicators. By integrating Apache Airflow for scheduling ETL and ELT workflows, Snowflake for scalable data warehousing, dbt for SQL-based transformations, and Preset for interactive visualization, this project provides a robust framework that can be adapted to various data analytics applications. The automated pipeline exemplifies how organizations can leverage modern data engineering tools to streamline data flow from ingestion to actionable insights, ultimately enhancing data-driven decision-making processes. Additionally, the project highlights the effectiveness of modular components in adapting to real-time data needs, ensuring that the system can scale with data growth and complexity, making it a powerful asset in today's data-centric business environment.

X. ACKNOWLEDGMENT

The authors would like to express their most profound appreciation to Professor Keeyong Han in the Department of Applied Science at San Jose State University. One learns much from outstanding teaching, patience, and devotion to his students. Professor Han continued to provide important inputs, taking hours of detailed guidance and encouragement to ensure that the students succeeded. His support has been invaluable, as has his encouragement and motivation. Above all, we are grateful for the mentorship he so willingly provided in the completion of this project.

XI. REFERENCES

How Airflow + dbt Work Together | dbt Labs. (n.d.). Dbt Labs. https://www.getdbt.com/blog/dbt-airflow

Snowflake Connection — apache-airflow-providers-snowflake Documentation. (n.d.). https://airflow.apache.org/docs/apache-airflow-providers-snowflake/stable/connections/snowflake.html

GeeksforGeeks. (2024, September 16). *Stock Price Prediction using Machine Learning in Python*. GeeksforGeeks. https://www.geeksforgeeks.org/stock-price-prediction-using-machine-learning-in-python/