Lab Group 17

Stock Price Prediction using Airflow

Omkar Rajale Yashashree Shinde

Abstract

This document serves as a comprehensive project proposal for the development of the **Stock Price Prediction System**. This application aims to provide daily forecasts of stock prices for selected companies (e.g., NVDA, APPL). The proposal details the requirement for a robust data analytics pipeline, outlining the extraction of 180 days of historical stock data via the **yfinance API**, orchestration of daily data ingestion and cleaning using **Apache Airflow**, and the execution of Machine Learning (ML) forecasting tasks within **Snowflake**. The goal is to deliver a reliable, automated, and scalable system capable of predicting stock prices for the next 7+ days.

1 Introduction

1.1 Project Overview

This section introduces the proposed application system, the **Stock Price Prediction Data Analytics System**, and briefly summarizes its primary goal: to deliver automated, daily, data-driven forecasts of future stock prices. The system is designed around modern data engineering practices, utilizing a cloud data warehouse (**Snowflake**) and an orchestration engine (**Airflow**).

2 Problem Statement

This section clearly defines the application system your team intends to build and articulates the necessity of the project.

2.1 The Problem

Stock price volatility and the difficulty of accurate short-term forecasting pose a significant challenge for individual investors and automated trading systems. The core problem is the lack of a standardized, automated, and repeatable process to ingest daily historical data, prepare it, and apply machine learning models to generate timely, reliable price predictions.

2.2 Proposed Application System

The **Stock Price Prediction Data Analytics System** is a serverless data pipeline designed to automate the end-to-end process of stock price forecasting. It focuses on repeatability and accuracy by ensuring a daily feed of fresh, 180-day historical data is available for ML tasks targeting the next 7+ days of price movement.

2.3 Rationale for Database and Data Pipelines

A persistent database (**Snowflake**) is essential for:

- **Data Integrity:** Storing raw and refined daily time-series data in a structured, queryable format.
- **ML Execution:** Providing a high-performance environment (**Snowflake ML**) where complex time-series forecasting models can be executed directly on the data with powerful SQL.

The data pipeline (**Airflow DAG**) is mandatory for:

- **Automation:** Ensuring the data ingestion from the yfinance API runs reliably daily at a scheduled time.
- **Repeatability:** Managing the complex multi-step ETL process (Extraction, Loading, ML Model Training, Prediction, and Result Aggregation) as a single, monitorable workflow.

3 Solution Requirements

This section details the necessary criteria, actions, and limitations of the final system, based on the requirements analysis.

3.1 Functional Requirements (FR)

The core capabilities and actions the application must perform are:

FR1: The system shall successfully connect to the **yfinance API** and retrieve the last 180 days of historical stock data (Open, Close, Min, Max, Volume) for at least two chosen companies (e.g., NVDA, APPL).

FR2: The system shall use an **Airflow DAG** to automate the data ingestion process and ensure it runs successfully every day.

FR3: The system shall load the ingested data into a designated table in **Snowflake** with the defined schema.

FR4: The system shall execute **SQL-based ML forecasting tasks** within Snowflake to predict stock prices for the next 7 days.

FR5: The system shall aggregate the historical data and the forecasted data into a final output table (**STOCK_PRICES_FINAL**) via a UNION operation.

3.2 Non-Functional Requirements (NFR)

3.2.1 Performance

The daily Airflow DAG execution (ingestion, processing, and forecasting) must complete within **30 minutes**.

3.2.2 Scalability

The pipeline must be designed to easily add new stock symbols with minimal configuration changes.

3.2.3 Data Integrity

Data loaded into Snowflake must pass a basic null check on the **Close** and **Date** columns.

3.2.4 Observability

The Airflow interface must provide clear logging and status monitoring for all daily pipeline runs.

3.3 System Users and Usage

The primary users of this system are **Data Analysts/Scientists** who rely on the final **STOCK_PRICES_FINAL** table for analysis and decision-making.

- **Use Case 1 (Daily Run):** The Airflow DAG automatically triggers at midnight, ingests fresh data, runs the ML model, and updates the STOCK_PRICES_FINAL table.
- **Use Case 2 (Analyst Query):** A user executes a simple SQL query against the STOCK_PRICES_FINAL table to see the predicted prices for NVDA and AAPL for the upcoming week.

3.4 Conceptual Architecture

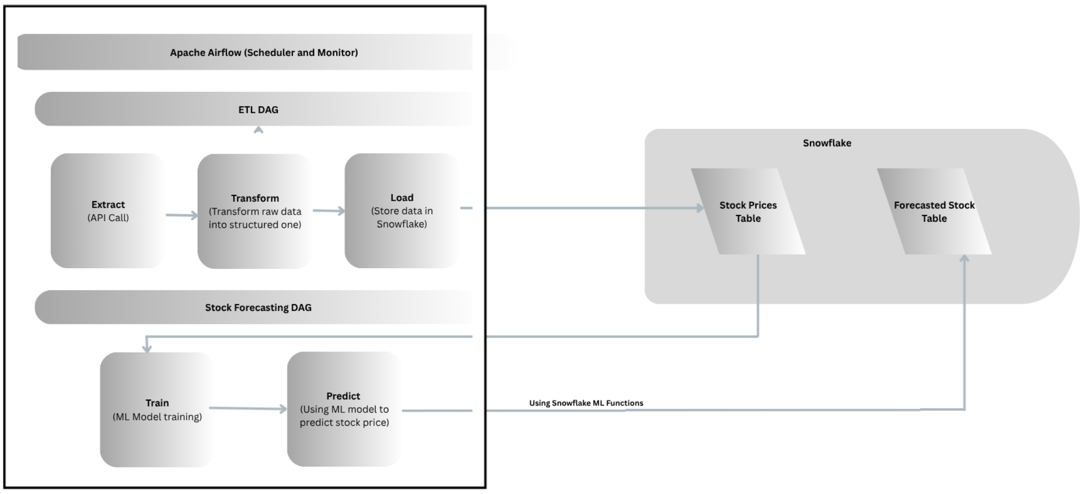

Figure 1: Conceptual Data Pipeline Architecture illustrating the sequential flow from the external yfinance API through the Airflow orchestration layer to the Snowflake Data Warehouse for storage and ML forecasting.

3.5 Functional Components Breakdown

3.5.1 Data Ingestion Module (Python/yfinance)

- **Role:** Extracts 180 days of stock data from the yfinance API.
- **Database Interactions:** Loads the raw/cleaned data into the Snowflake staging table (`STOCK_PRICES_180D`).

3.5.2 Orchestration Layer (Airflow DAG)

- **Role:** Schedules the daily execution of all pipeline tasks, manages dependencies, and handles failures.
- **Data Pipeline Interactions:** Defines the sequence: Ingest → Forecast → Union.

3.5.3 Data Warehouse & ML Engine (Snowflake)

- **Role:** Persistent storage for all historical data and the execution environment for ML models via SQL.
- **Database Interactions:** Executes `CREATE OR REPLACE MODEL ...` and `SELECT FORECAST(...)` SQL queries; writes predictions to `STOCK_PRICE_FORECAST_7D`.

4 Implementation Details and Appendices

4.1 Tables Structure

Table 1: Summary of Database Table Structures

Table Name	Key Fields	Other Columns	Description
STOCK_PRICES_180D	date (PK), symbol (PK)	open, close, min, max, volume	Stores the daily 180-day history pulled from yfinance
STOCK_PRICE_FORECAST_7D	date (PK), symbol (PK)	open, close, min, max, volume	Stores the 7+ day forecast generated by Snowflake ML.
STOCK_PRICES_FINAL	date (PK), symbol (PK)	open, close, min, max, volume	The final aggregated table (historical UNION prediction).

4.2 Screenshots

The following figures serve as visual evidence of the system's successful implementation, corresponding to the required artifacts (Airflow DAGs, logs, variables, connections, and final table).

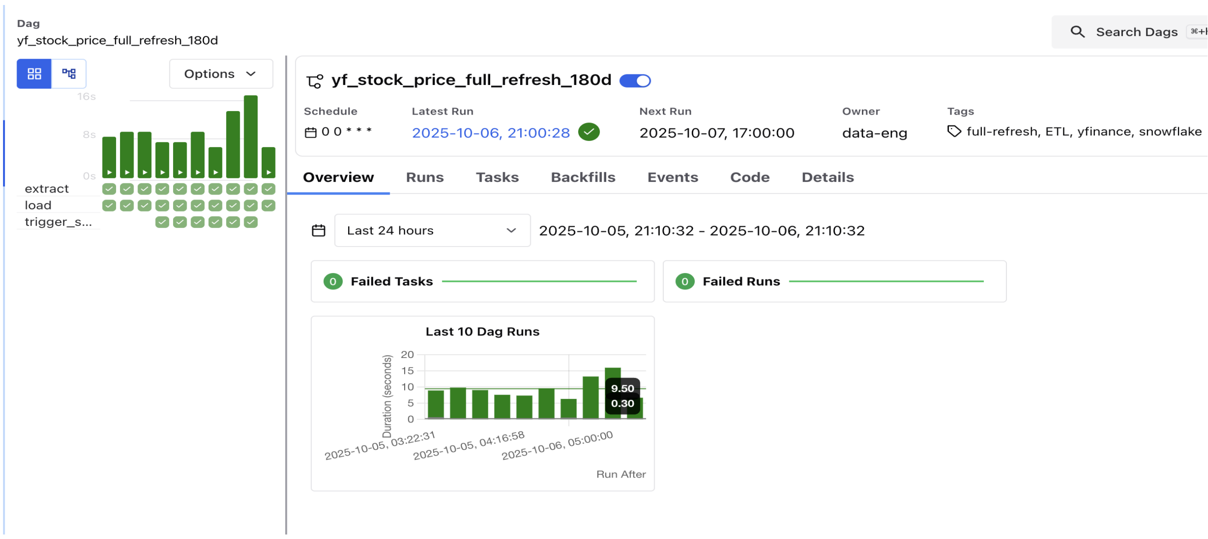

Figure 2: Airflow DAG Graph View, showing the sequential ETL task flow: Extract → Transform → Load.

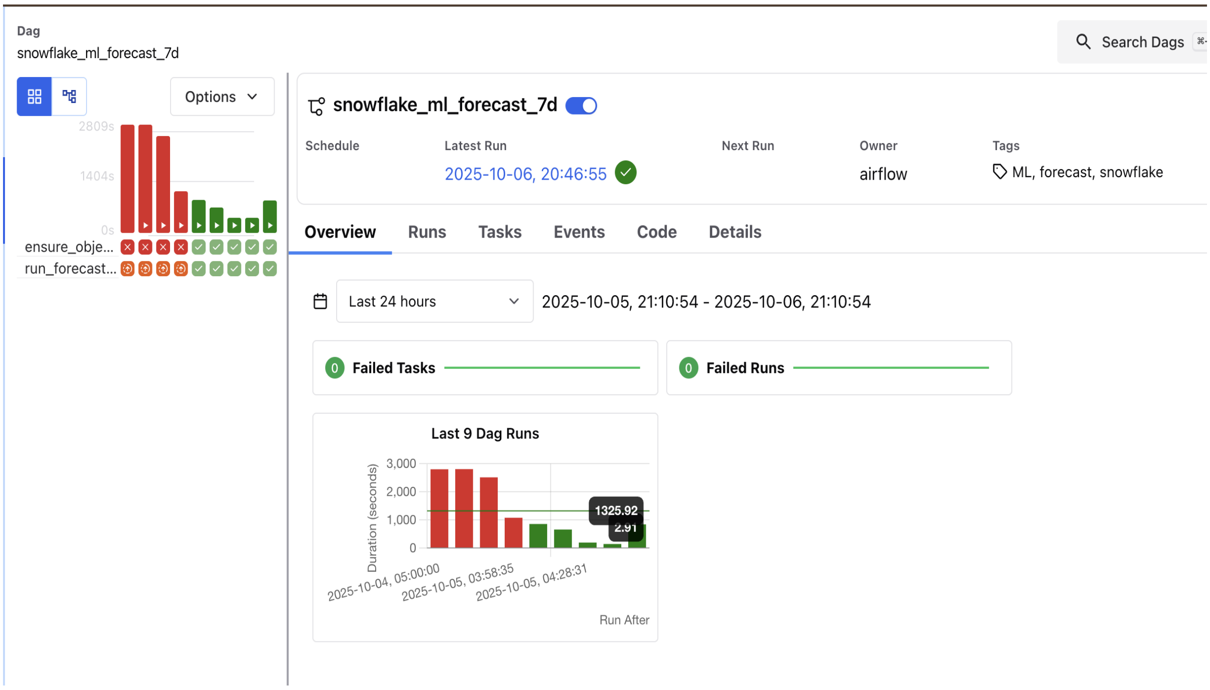

Figure 3: Airflow DAG Graph View, showing the train and ML predict: Train → Predict.

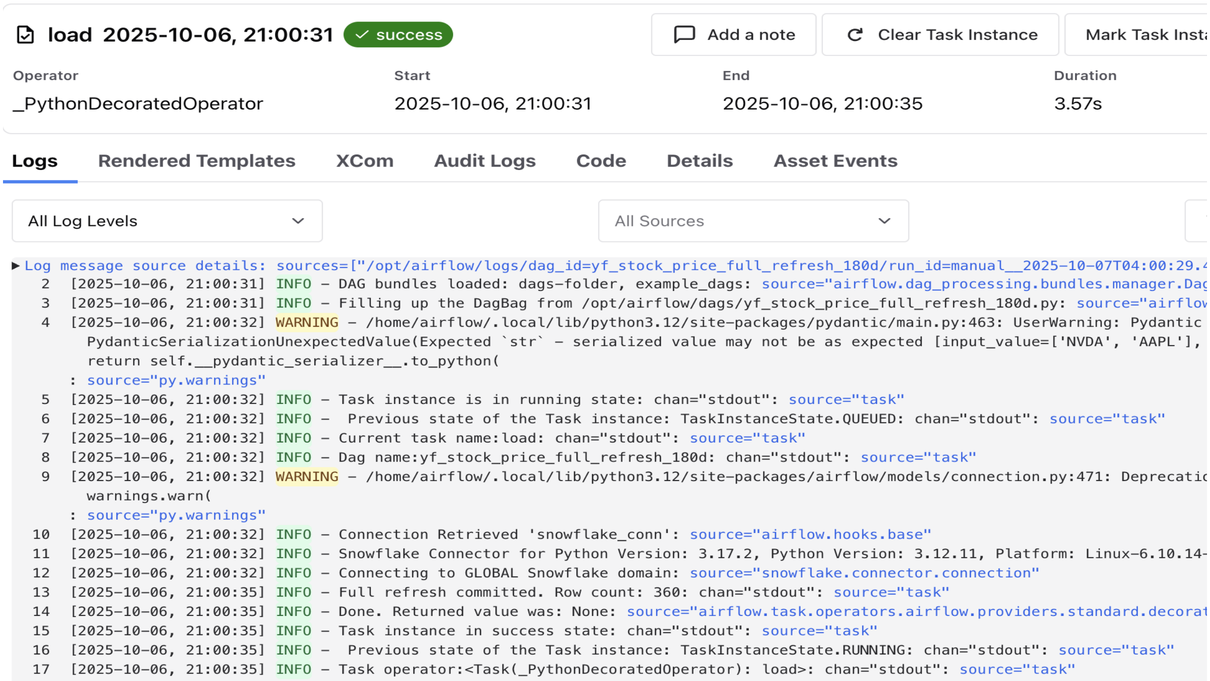

Figure 4: Airflow Dag Logs

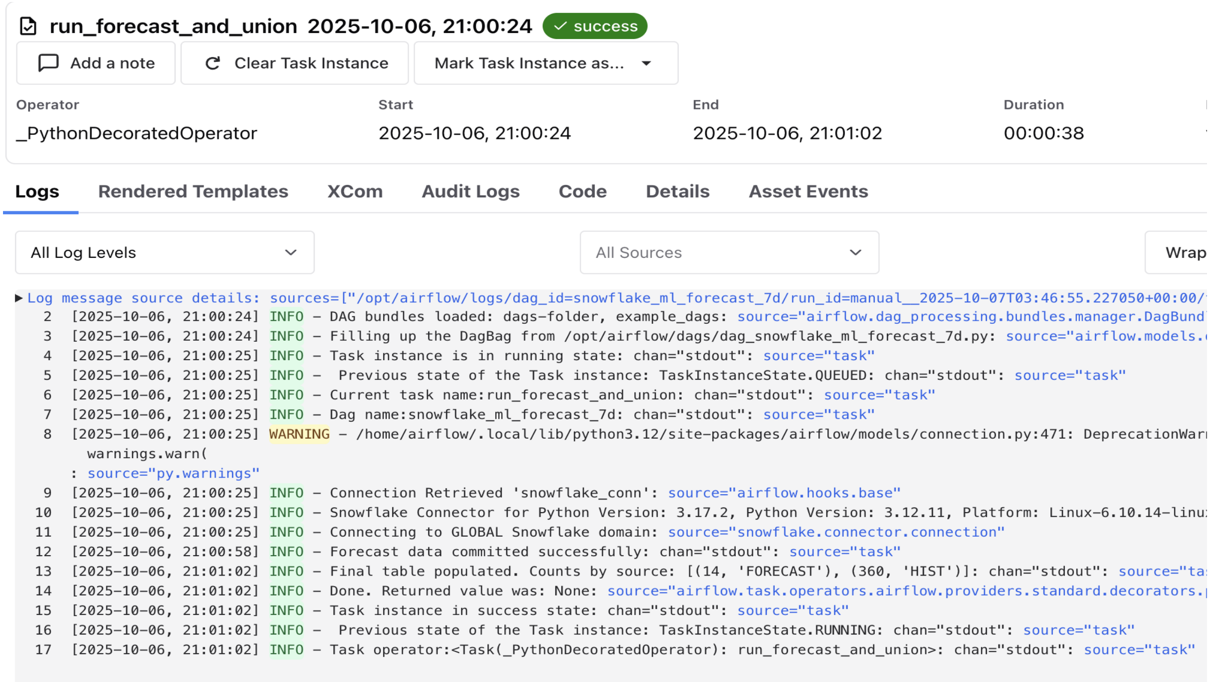

Figure 5: Airflow Dag Logs

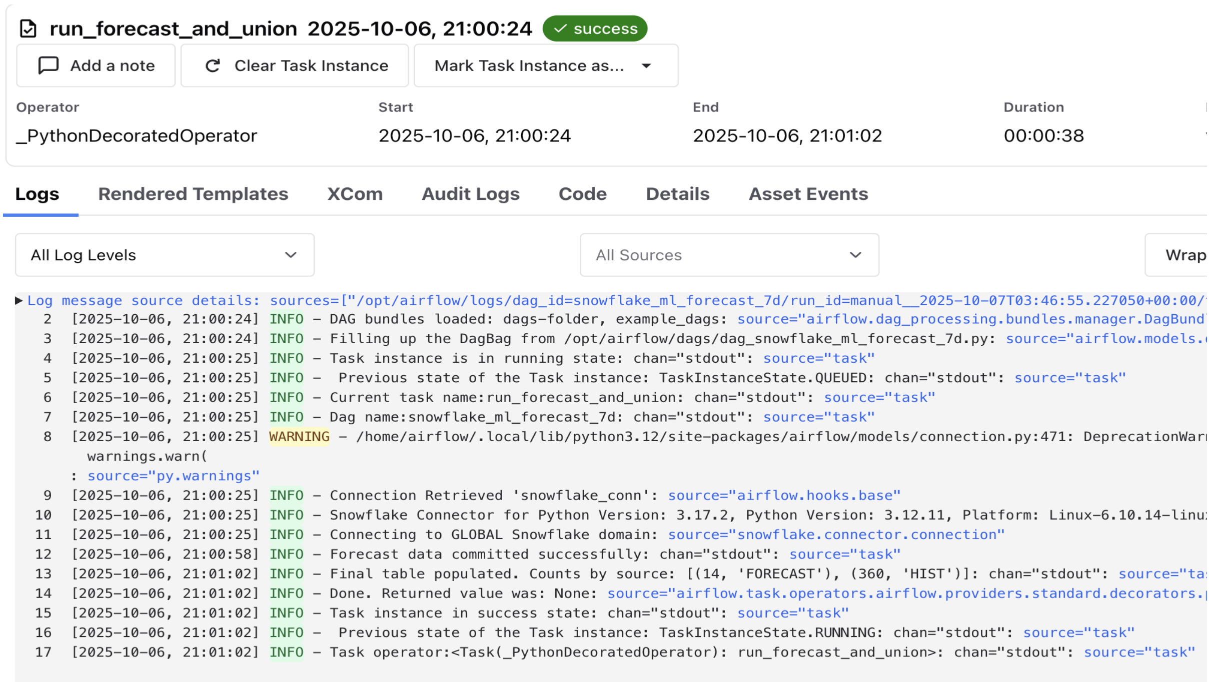

Figure 6: Airflow Dag Logs

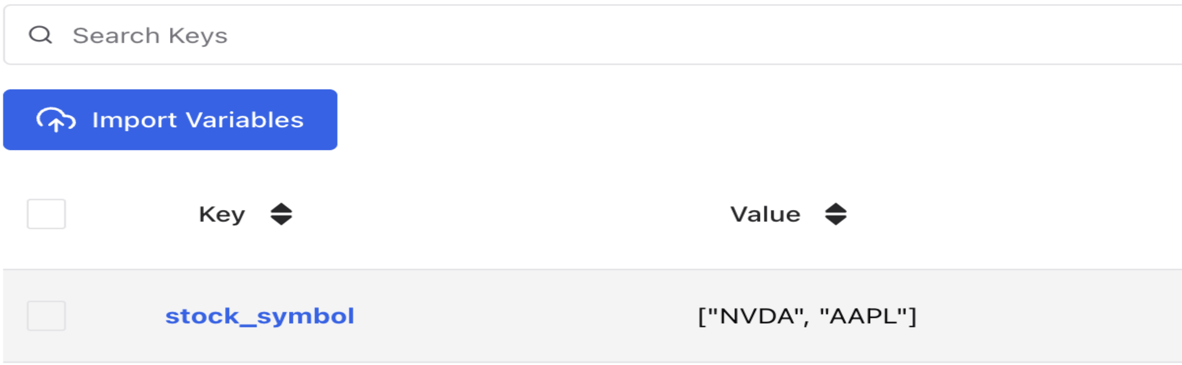

Figure 7: Airflow Variables

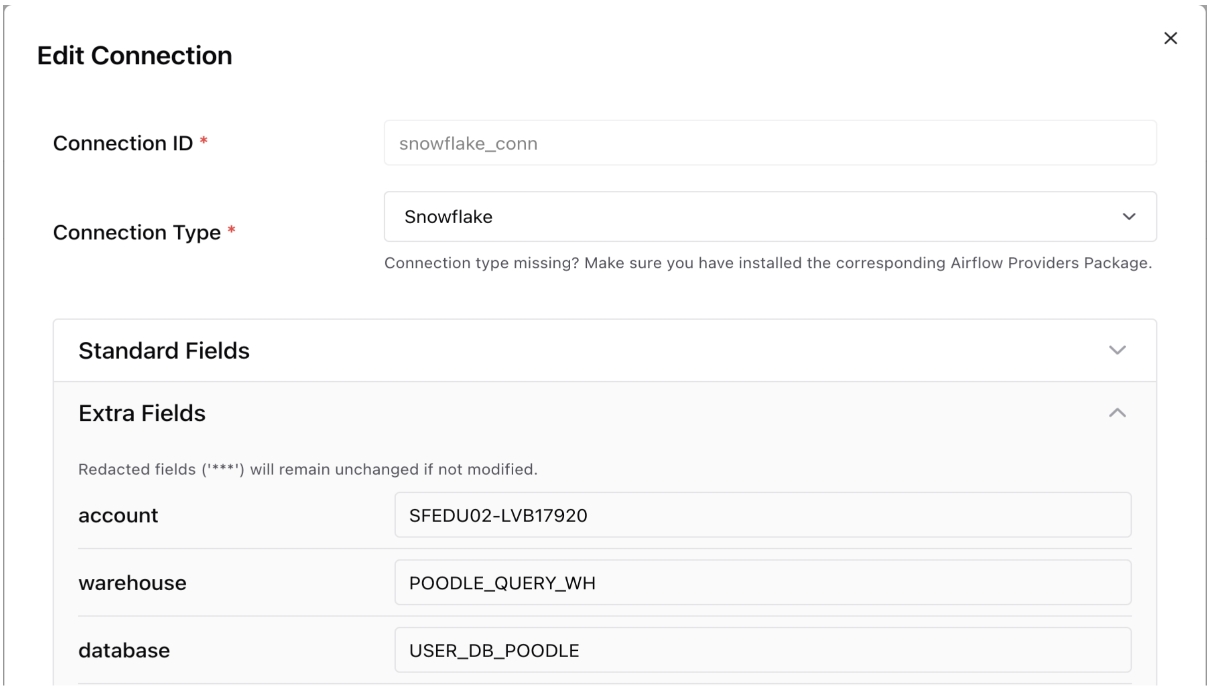

Figure 8: Airflow Connection

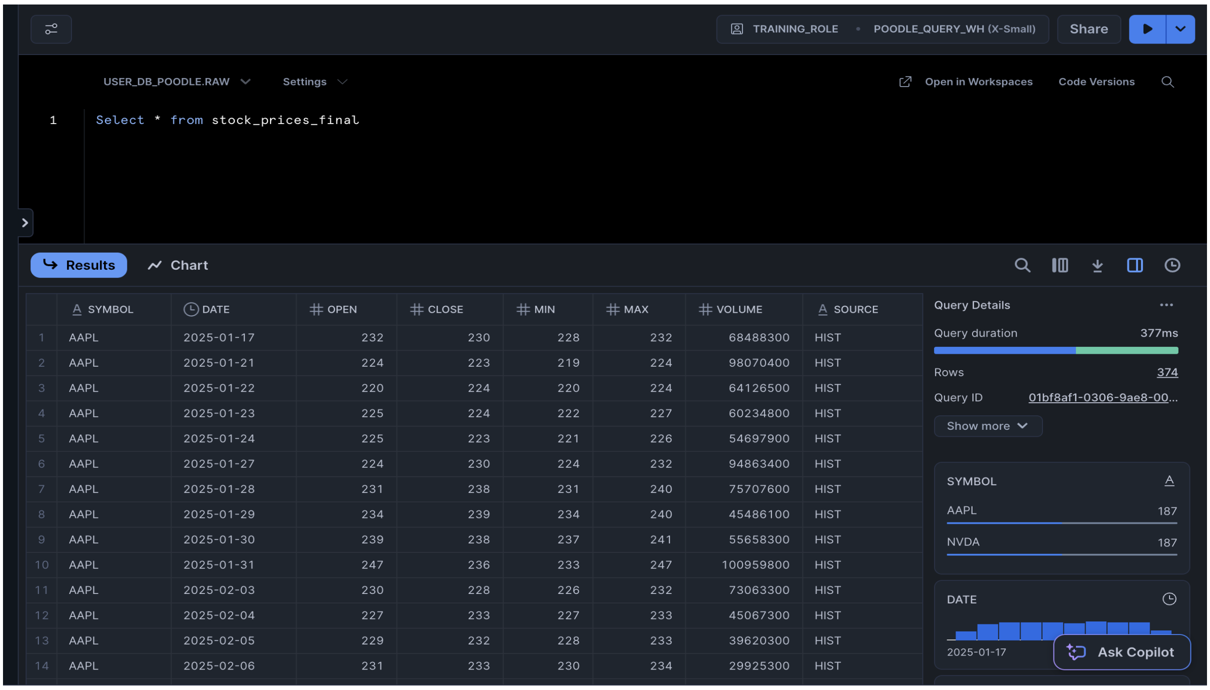

Figure 9: Final `STOCK_PRICES_FINAL` Table in Snowflake, showcasing the union of historical data (HIST) and the forecasted predictions (e.g., FORECAST).

4.3 Code Repository Link

GITHUB REPO LINK: `https://github.com/omkarrajale1499/Stock_Price_Prediction_Airflow.git`

5 Conclusion

This proposal outlines a clear plan for the development of the **Stock Price Prediction System** using Airflow. The requirements analysis confirms the critical need for an automated daily forecasting solution, and the conceptual design provides a robust framework utilizing industry-standard tools (Airflow, Snowflake, Python). We are confident that this approach will yield a successful and impactful application that meets all specified requirements for daily, 7+ day stock price prediction.