

Stock Price Prediction for Amazon (AMZN) and Meta (META) using Airflow, yfinance, and Snowflake

Danish Waseem (019101511), Srinidhi Jaya Revanth Srirangarajapally (019123143)

Department of Applied Data Science,
College of Information, Data, and Society,
San Jose State University, San Jose, California, USA 95192

Email: *danish.waseem@sjsu.edu*, *srinidhijayarevanth.srirangarajapally@sjsu.edu*

Abstract

This paper presents a daily automated system that downloads stock prices for Amazon (AMZN) and Meta (META), stores them in Snowflake, and predicts prices for the next seven days using Snowflake ML Forecast. The workflows are built and scheduled using Apache Airflow and monitored through its web interface. This report describes the motivation, requirements, workflow design, database schema, and implementation details of the system. Screenshots, code repositories, and outputs are referenced inline and in the Appendix.

I. TEAM INTRODUCTION

Danish Waseem led ETL pipeline development, Snowflake schema, and orchestration.

Srinidhi Jaya Revanth Srirangarajapally implemented forecasting DAG, SQL modeling in Snowflake ML, and documentation.

Both collaborated on system design, testing, and report writing.

II. PROBLEM STATEMENT

Stock price forecasting is a key application in data analytics that requires processing large volumes of time-series data. Manual data collection and modeling are error-prone and time-consuming. This project automates the entire workflow: data extraction using `yfinance`, transformation and loading through Airflow, and predictive modeling using Snowflake ML. The system collects 180 days of OHLCV (Open, High, Low, Close, Volume) data for AMZN and META, stores it in Snowflake, and automatically forecasts prices for the next seven days.

III. DATASET(S)

We use **Yahoo Finance** via the `yfinance` Python package:

- **Symbols:** AMZN, META
- **Fields:** Open, High, Low, Close, Volume
- **Horizon:** Most recent 180 trading days (daily interval)
- **Access:** Public API wrapper; no keys required

IV. SYSTEM DIAGRAM

Figure 1 depicts the end-to-end flow from extraction to prediction. The system ensures data integrity, modularity, and full automation from raw data ingestion to forecast output.

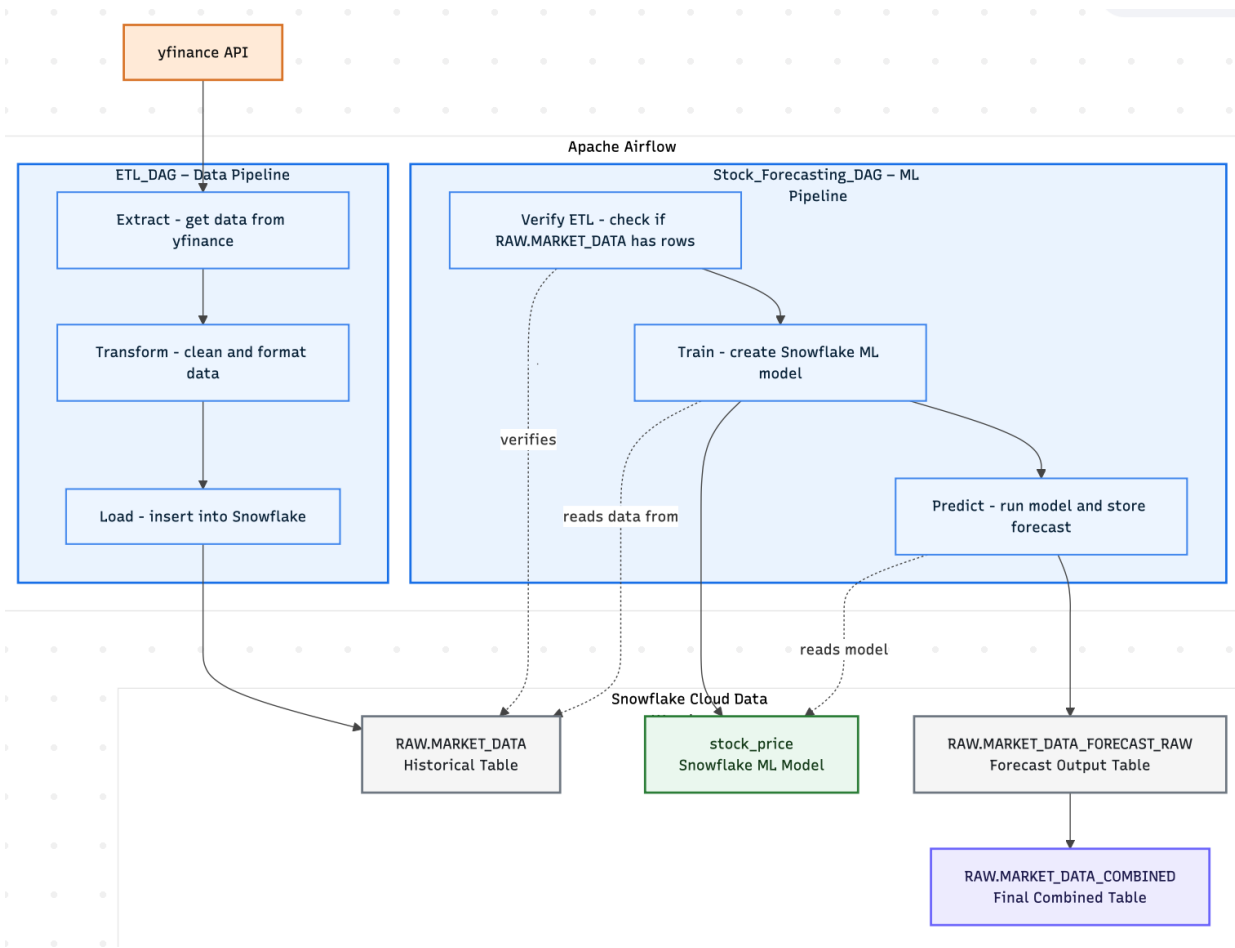


Fig. 1: System Architecture: yfinance → Airflow ETL DAG → Snowflake (RAW.MARKET_DATA) → Airflow Forecasting DAG → Final Output.

V. TABLES

A. RAW.MARKET_DATA (Historical Data)

TABLE I: Snowflake Table: RAW.MARKET_DATA

Column	Type	Null	Constraints / Notes
DATE	DATE	No	Trading day (UTC). Logical uniqueness with SYMBOL.
OPEN	FLOAT	No	Open price for the day.
HIGH	FLOAT	No	High price for the day.
LOW	FLOAT	No	Low price for the day.
CLOSE	FLOAT	No	Close price (fallback to Adj Close if needed).
VOLUME	BIGINT	No	Total traded volume for the day.
SYMBOL	VARCHAR(10)	No	Ticker symbol (AMZN, META).
CREATED_AT	TIMESTAMP_NTZ	Yes	Ingestion timestamp; default CURRENT_TIMESTAMP().

Logical Primary Key: (SYMBOL, DATE).

A populated view of RAW.MARKET_DATA is shown in Figure 2.

B. RAW.MARKET_DATA_COMBINED (History ∪ Forecast)

A sample of the final combined output is shown in Figure 3.

Fig. 2: Snowflake table `RAW.MARKET_DATA` populated by `ETL_DAG`.

TABLE II: Snowflake Table: `RAW.MARKET_DATA_COMBINED`

Column	Type	Null	Constraints / Notes
SYMBOL	VARCHAR(10)	No	Ticker symbol.
DATE	DATE	No	History date or forecasted date (cast from timestamp).
ACTUAL	FLOAT	Yes	Close price for history rows; NULL for forecasts.
FORECAST	FLOAT	Yes	Model forecast for forecast rows; NULL for history.
LOWER_BOUND	FLOAT	Yes	Lower bound of 95% interval.
UPPER_BOUND	FLOAT	Yes	Upper bound of 95% interval.
Row Semantics: Exactly one of {ACTUAL, FORECAST} is non-NULL.			

VI. IMPLEMENTATION DETAILS

A. Scheduling and Orchestration

Airflow schedules two DAGs:

- **ETL_DAG** at 02:30 daily (Extract → Transform → Load).
- **Stock_Forecasting_DAG** at 03:00 daily (Verify → Train → Predict).

B. ETL_DAG Pipeline

Extract uses `yfinance.Ticker().history()` for a clean `DataFrame`. **Transform** drops invalid rows. **Load** executes DDL and idempotent upserts (implemented as delete+insert) inside a transaction. Figure 4 shows the ETL DAG in the Airflow UI.

C. Forecasting Pipeline

Verify checks that `RAW.MARKET_DATA` has rows. **Train** creates `MARKET_DATA_v1` and the model `stock_price`. **Predict** calls `!FORECAST`, stores the raw output, creates a clean view, and builds the combined table. Figure 5 shows the Forecasting DAG.

	SYMBOL	DATE	# ACTUAL	# FORECAST	# LOWER_BOUND	# UPPER_BOUND
358	AMZN	2025-10-02	222.410003662	null	null	null
359	AMZN	2025-10-03	219.509994507	null	null	null
360	AMZN	2025-10-06	220.899993896	null	null	null
361	META	2025-10-07	null	723.283589609	695.070817929	751.49639458
362	META	2025-10-08	null	726.302968271	687.253432031	765.352565546
363	META	2025-10-09	null	725.531270273	678.063458784	772.999120601
364	META	2025-10-10	null	724.714925058	670.113051722	779.316809491
365	META	2025-10-13	null	726.336201913	665.430788862	787.241642708
366	META	2025-10-14	null	727.304097421	660.689230834	793.919019495
367	META	2025-10-15	null	729.214528329	657.34245613	801.086650467
368	META	2025-10-16	null	730.414632089	653.644606632	807.184707484
369	META	2025-10-17	null	730.540517089	649.166853434	811.914202959
370	META	2025-10-20	null	730.864644296	645.134242885	816.595067902
371	META	2025-10-21	null	733.7509358	643.874780863	823.627146224
372	META	2025-10-22	null	734.909474618	641.070526674	828.748439208
373	META	2025-10-23	null	738.401021249	640.760034293	836.042074789
374	META	2025-10-24	null	733.49602247	632.195573183	834.796521695
375	AMZN	2025-10-07	null	220.798302739	211.361003454	230.235602024
376	AMZN	2025-10-08	null	220.798302739	208.029575648	233.56702983
377	AMZN	2025-10-09	null	220.798302739	205.402908452	236.193697026
378	AMZN	2025-10-10	null	220.798302739	203.16322579	
379	AMZN	2025-10-13	null	220.798302739	201.177555742	

Fig. 3: Forecast output in RAW.MARKET_DATA_COMBINED (history \cup forecast).

D. Key SQL (Snowflake)

Model creation:

```
CREATE OR REPLACE SNOWFLAKE.ML.FORECAST stock_price(
INPUT_DATA => SYSTEM$REFERENCE('VIEW', 'MARKET_DATA_v1'),
SERIES_COLNAME => 'SYMBOL',
TIMESTAMP_COLNAME => 'DATE_v1',
TARGET_COLNAME => 'CLOSE',
CONFIG_OBJECT => { 'ON_ERROR': 'SKIP' }
);
```

Forecast execution and capture:

```
BEGIN
CALL stock_price!FORECAST(
FORECASTING_PERIODS => 14,
CONFIG_OBJECT => {'prediction_interval': 0.95}
);
LET x := SQLID;
CREATE OR REPLACE TABLE RAW.MARKET_DATA_FORECAST_RAW AS
SELECT * FROM TABLE (RESULT_SCAN(:x));
END;
```

Final union:

```
CREATE OR REPLACE TABLE RAW.MARKET_DATA_COMBINED AS
SELECT SYMBOL, DATE, CLOSE AS ACTUAL, NULL AS FORECAST, NULL AS LOWER_BOUND, NULL AS
UPPER_BOUND
FROM RAW.MARKET_DATA
WHERE SYMBOL IN ('META', 'AMZN')
```

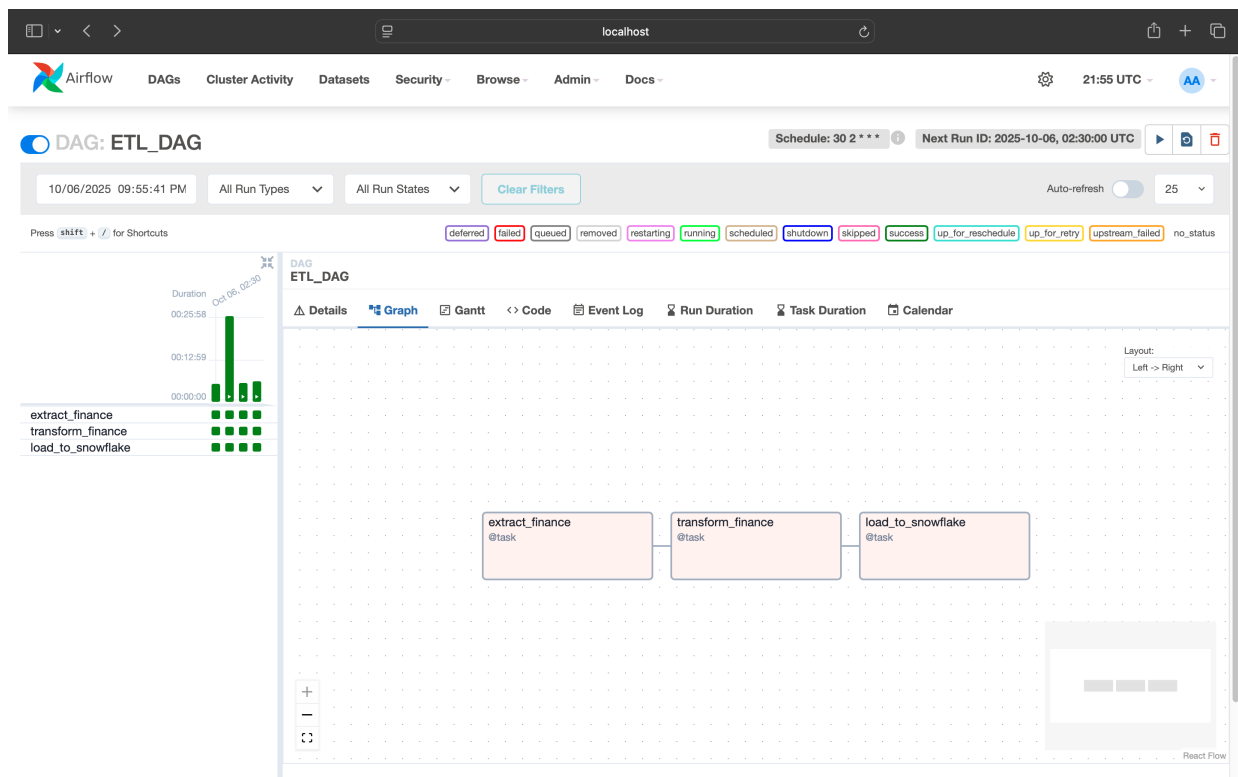


Fig. 4: Airflow ETL DAG: Extract \rightarrow Transform \rightarrow Load.

```
UNION ALL
SELECT SYMBOL, CAST(DATE_v1 AS DATE), NULL AS ACTUAL, FORECAST, LOWER_BOUND,
UPPER_BOUND
FROM MARKET_DATA_FORECAST_CLEAN;
```

VII. LESSONS

- Using `Ticker().history()` avoids `MultiIndex` issues from `yf.download()` for single symbols.
- Always wrap Snowflake writes in transactions for safe retries.
- Validate data before model training (our `Verify` task prevents empty-train runs).
- Keep DAGs modular; it simplifies debugging and re-runs.

VIII. FUTURE WORK

- Add more symbols and longer forecasting windows.
- Include technical indicators (moving averages, Bollinger Bands) for trend analysis.

IX. CONCLUSION

Two Airflow DAGs automate stock ingestion and forecasting for AMZN and META. Airflow provides scheduling/observability, while Snowflake ML enables fast model training and inference. The Verify–Train–Predict flow maintains data quality and reproducibility.

ACKNOWLEDGMENT

The authors thank Professor Keeyong Han and SA Jeff Chong for their valuable feedback and guidance throughout the project.

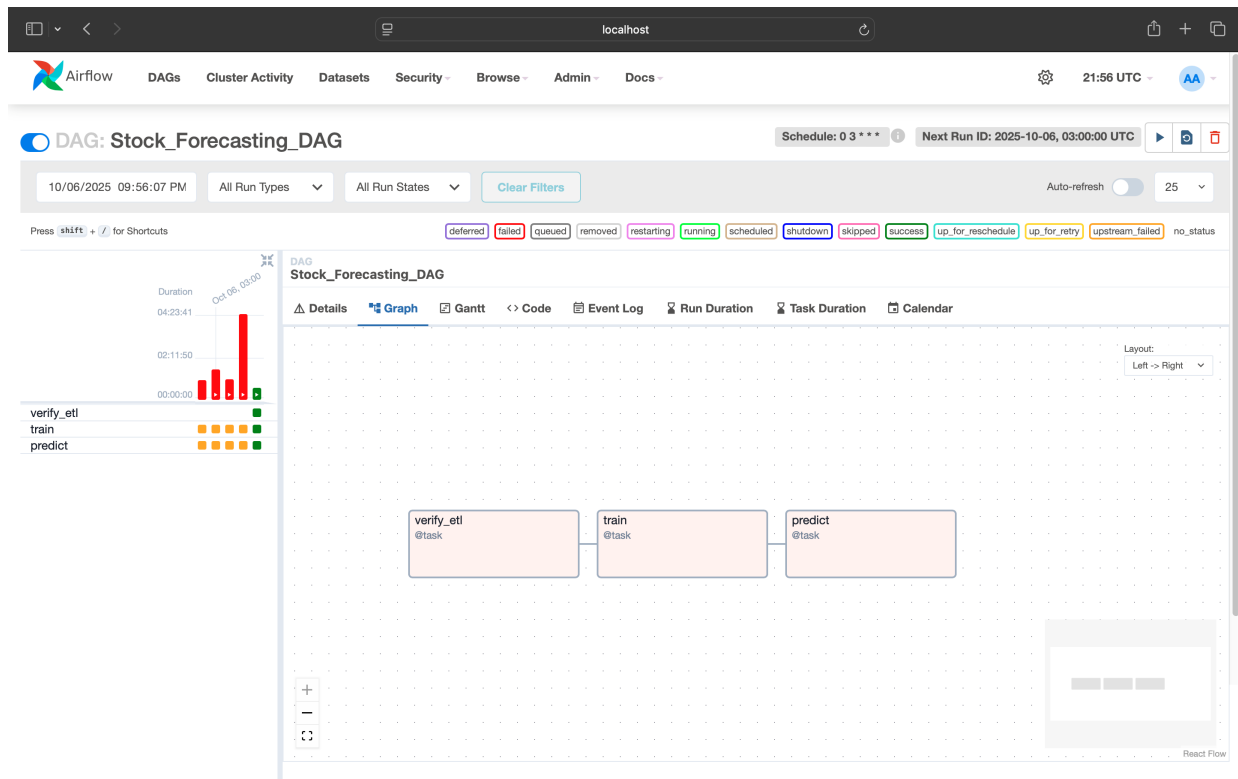


Fig. 5: Airflow Stock Forecasting DAG: Verify → Train → Predict.

REFERENCES

- 1) yfinance: <https://pypi.org/project/yfinance/>
- 2) Snowflake ML Forecast: <https://docs.snowflake.com/en/sql-reference/ml-forecast>
- 3) Apache Airflow: <https://airflow.apache.org/docs/>

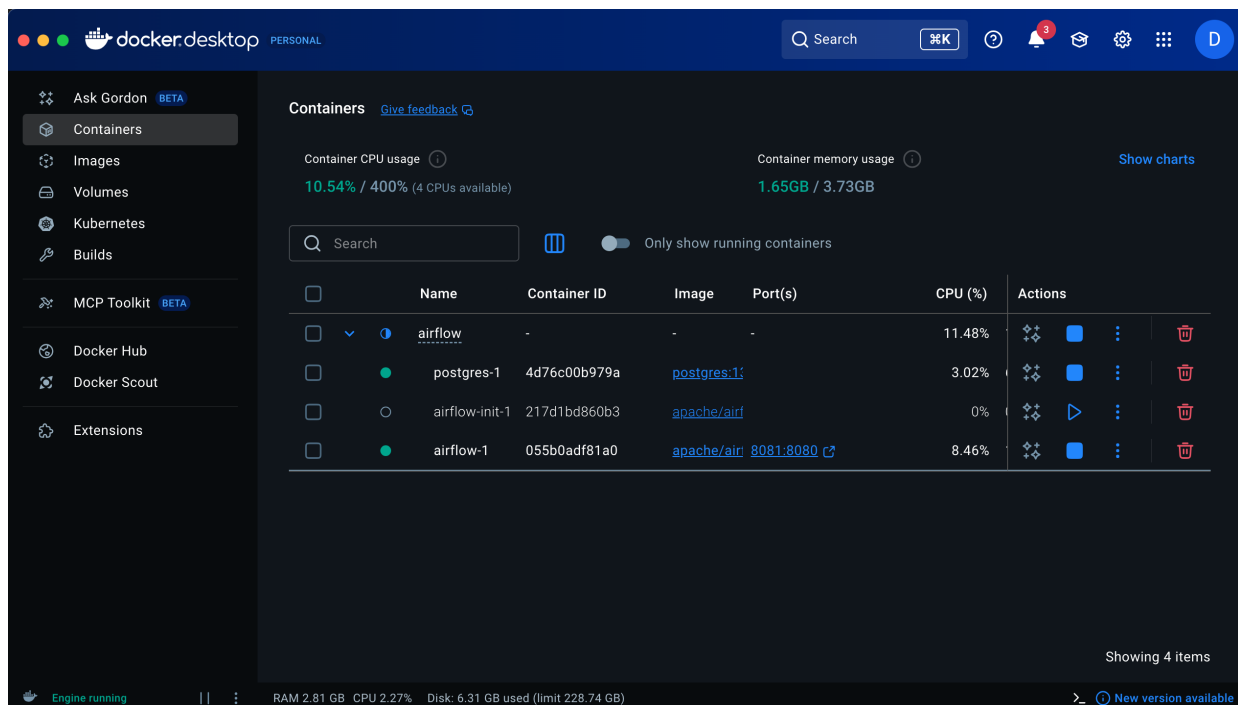


Fig. 6: Docker Desktop

APPENDIX: SOURCE CODE AND REPOSITORIES

All Python DAGs, SQL scripts, and supplementary files are hosted on GitHub:

- Danish Waseem (Main Repo): <https://github.com/danwaseem/SJSU-DATA226/tree/main/LAB1>
- Srinidhi Jaya Revanth Srirangarajapally (Main Repo): <https://github.com/Revanth0211/DATA-226-Lab-1->
- Danish Waseem (Airflow DAGs): <https://github.com/danwaseem/SJSU-DATA226/tree/main/Airflow>
- Srinidhi Jaya Revanth Srirangarajapally (Airflow DAGs): <https://github.com/Revanth0211/Stock-Prices-ETL-Airflow>

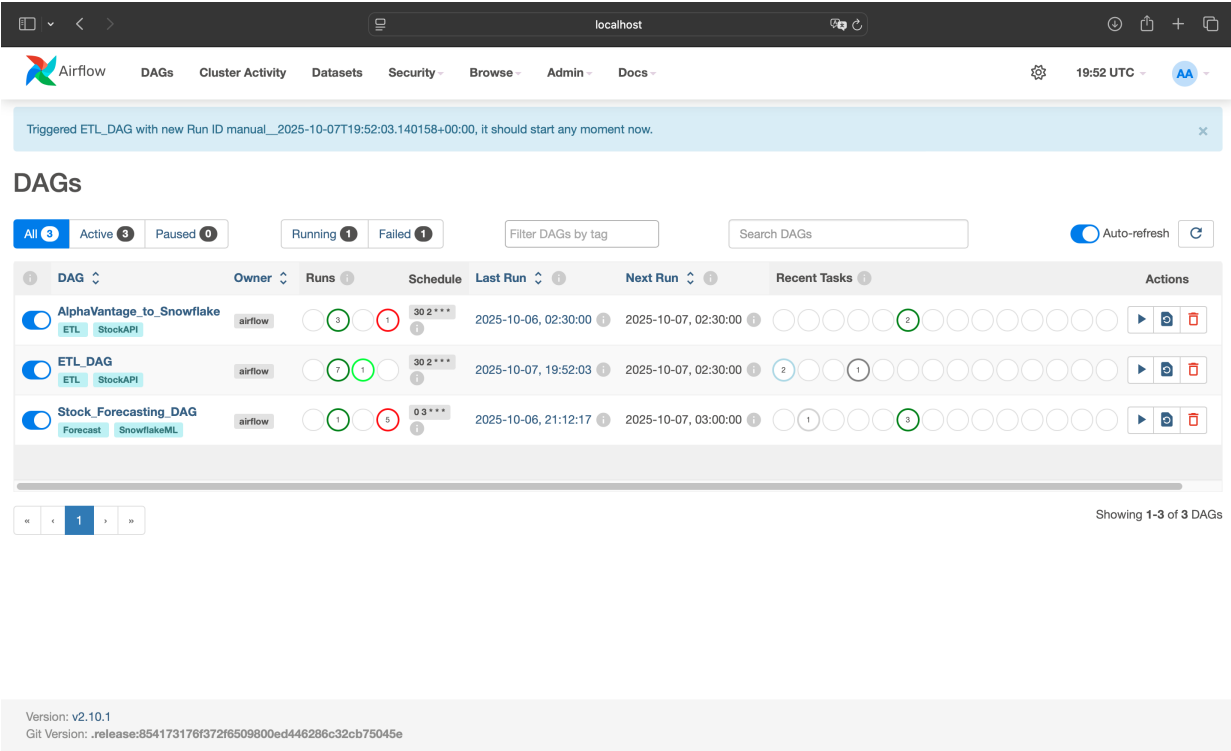


Fig. 7: DAGs

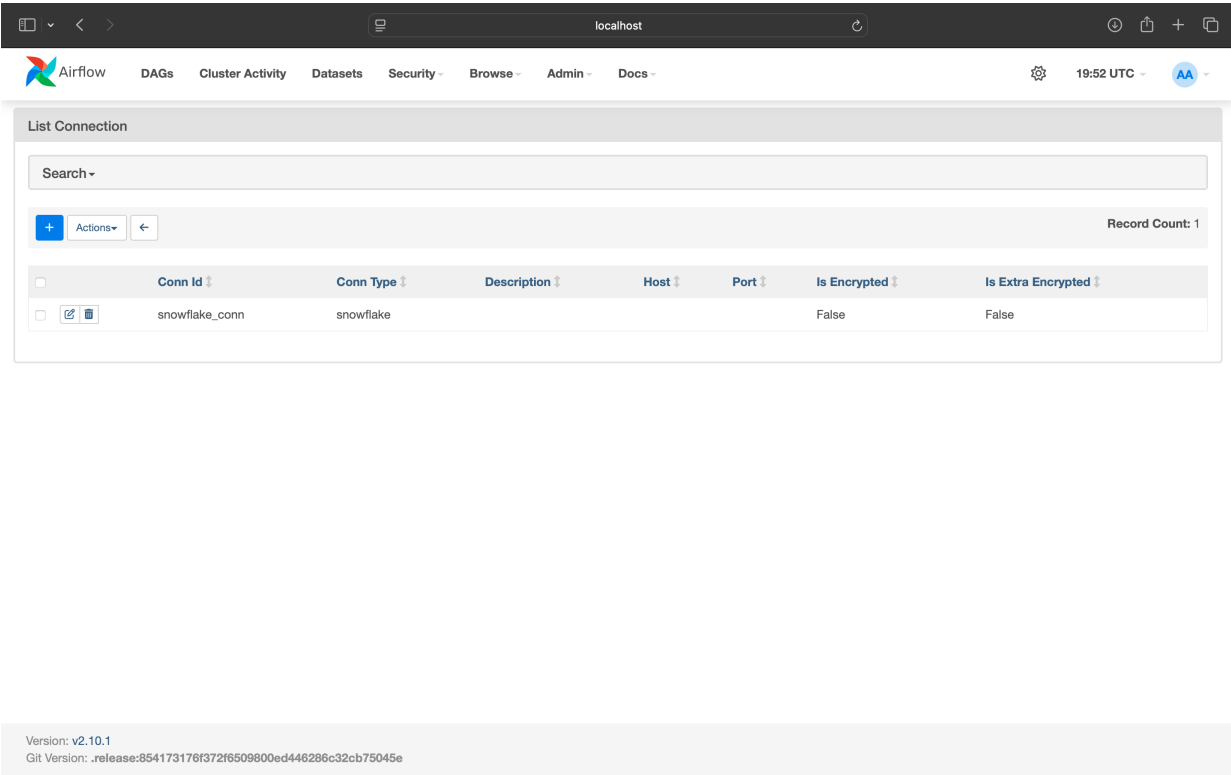


Fig. 8: Snowflake Connection in Airflow

USER_DB_HEDGEHOG / RAW Schema TRAINING_ROLE 1 month ago Create

Schema Details **Tables** Views Stages File Formats

7 Tables Search All Tables Refresh

NAME ↑	TYPE	CLASSIFICATION	OWNER	ROWS	BYTES	CREATED	
MARKET_DATA	Table	—	TRAINING_R...	360	12.0KB	2 weeks ago	...
MARKET_DATA_COMBINED	Table	—	TRAINING_R...	388	5.0KB	22 hours ago	...
MARKET_DATA_FORECAST_RAW	Table	—	TRAINING_R...	28	3.0KB	22 hours ago	...

Fig. 9: Snowflake Tables

DAG: ETL_DAG Schedule: 30 2 * * * Next Run ID: 2025-10-07, 02:30:00 UTC Auto-refresh 25

10/07/2025 08:16:25 PM All Run Types All Run States Clear Filters

Press **shift** + **/** for Shortcuts deferred failed queued removed restarting running scheduled shutdown skipped success up_for_reschedule up_for_retry upstream_failed no_status

ETL_DAG / **2025-10-06, 02:30:00 UTC** Clear Mark state as...

Details **Graph** **Gantt** **Code** **Event Log** View full cluster Audit Log

Show Logs After: 10/07/2025 08:16:53 PM Show Logs Before: 10/07/2025 08:16:53 PM Events to ☒ Include ☐ Exclude Select...

WHEN *	TASK ID *	MAP INDEX	TRY NUMBER	EVENT *	USER *	DETAILS *
2025-10-07, 19:57:46 UTC	load_to_snowflake		1	success	airflow	
2025-10-07, 19:52:41 UTC	load_to_snowflake		1	running	airflow	
2025-10-07, 19:52:32 UTC	transform_finance		1	success	airflow	
2025-10-07, 19:52:30 UTC	transform_finance		1	running	airflow	
2025-10-07, 19:52:22 UTC	extract_finance		1	success	airflow	
2025-10-07, 19:52:19 UTC	extract_finance		1	running	airflow	

Fig. 10: ETL DAG Event Log

DAG: Stock_Forecasting_DAG Schedule: 0 3 * * * Next Run ID: 2025-10-07, 03:00:00 UTC Auto-refresh 25

10/07/2025 08:17:09 PM All Run Types All Run States Clear Filters

Press **shift** + **/** for Shortcuts deferred failed queued removed restarting running scheduled shutdown skipped success up_for_reschedule up_for_retry upstream_failed no_status

Stock_Forecasting_DAG / **2025-10-06, 03:00:00 UTC** Clear Mark state as...

Details **Graph** **Gantt** **Code** **Event Log** View full cluster Audit Log

Show Logs After: 10/07/2025 08:17:20 PM Show Logs Before: 10/07/2025 08:17:20 PM Events to ☒ Include ☐ Exclude Select...

WHEN *	TASK ID *	MAP INDEX	TRY NUMBER	EVENT *	USER *	DETAILS *
2025-10-07, 20:04:33 UTC	predict		1	success	airflow	
2025-10-07, 20:04:07 UTC	predict		1	running	airflow	
2025-10-07, 20:04:02 UTC	train		1	success	airflow	
2025-10-07, 20:02:18 UTC	train		1	running	airflow	
2025-10-07, 20:02:10 UTC	verify_etl		1	success	airflow	
2025-10-07, 20:02:09 UTC	verify_etl		1	running	airflow	

Fig. 11: Stock Forecasting DAG Event Log