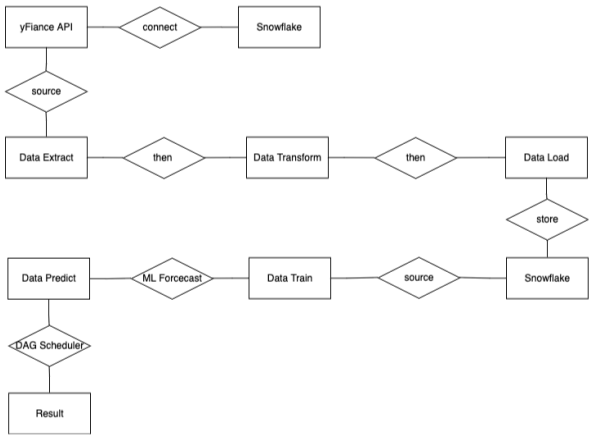
DATA 226 1 Report Yilin Sun, Yongxin

About Our Project

The goal of our project is to develop a system that extracts stock price data for a specified stock symbol (i.e., AAPL) using the yFinance API, processes loaded data using an ETL pipeline, and then forecasts future stock prices (future 7 days) using machine learning models. The system is automated using Apache Airflow to orchestrate the ETL process and the machine learning forecast pipeline, and the results are stored in Snowflake for further analysis.

Overall System Diagram



Requirements and Specifications

a. Data Extraction
- Extract stock price data using the yFinance API
- The API fetches stock data for a specified stock symbol over several days.

b. Data Transformation
- Transform the raw stock data into a structured format, including fields like stock symbol, date, open price, close price, high price, low price, and volume.

c. Data Loading
- Load the transformed data into a Snowflake table.

e. Machine Learning Forcasting
- Train a machine learning model to forecast future stock prices using historical data.
- The model should be deployed in Snowflake, and predictions should be generated and stored in a new table.

f. Integration
- Using Apache Airflow to automate the ETL and machine learning forecasting pipeline.
- Airflow DAGs should be defined for both the yFinance pipeline and the ML forecasting pipeline.

g. SQL and Transactions:
- Using SQL transactions with proper error handling (try/except) to ensure data integrity during the ETL and forecasting processes

Tables Structure

Stock Data Table (`Lab1_stock_data`)
- symbol (STRING): Stock symbol (Primary Key)
- date (DATE): Date of the symbol stock data (Primary Key)
- open (FLOAT): Opending price of the stock on the given date
- close (FLOAT): Closing price of the stock on the given date
- high (FLOAT): Highest price during the day
- low (FLOAT): Lowest price during the day
- volume (BIGINT): Trading volume for the day

```
CREATE OR REPLACE TABLE
DEV.RAW.LAB1_STOCK_DATA(
    SYMBOL STRING,
    DATE DATE,
    OPEN FLOAT,
    CLOSE FLOAT,
    HIGH FLOAT,
    LOW FLOAT,
    VOLUME BIGINT,
    PRIMARY KEY(SYMBOL, DATE)
);
```

Forecast Table (`lab1_forecast_data`)
- symbol (STRING): Stock symbol (Primary Key)
- date (DATE): Date of the symbol stock data (Primary Key)
- forecast (FLOAT): Forecasted closing price
- lower_bound (FLOAT): Lower bound of the forecast
- upper_bound (FLOAT): Upper bound of the forecast

```
CREATE OR REPLACE TABLE
DEV.ADHOC.LAB1_FORECAST_TABLE(
    SYMBOL STRING,
    DATE DATE,
    FORECAST FLOAT,
    LOWER_BOUND FLOAT,
```

```
    UPPER_BOUND FLOAT
);
```

Final Prediction Table (`lab1_prediction_results`)

- symbol (STRING): Stock symbol (Primary Key)
- date (DATE): Date of the symbol stock data (Primary Key)
- actual (FLOAT): actual closing price from stock data
- forecast (FLOAT): Forecasted closing price
- lower_bound (FLOAT): Lower bound of the forecast
- upper_bound (FLOAT): Upper bound of the forecast

```
CREATE OR REPLACE TABLE
DEV.ANALYTICS.LAB1_PREDICTION_RESULTS
(
    SYMBOL STRING,
    DATE DATE,
    ACTUAL FLOAT,
    FORECAST FLOAT,
    LOWER_BOUND FLOAT,
    UPPER_BOUND FLOAT
);
```

`Extract` retrieves stock data for a specified number of days for a given stock symbol, interacts with the API to pull daily time series data for the stock, then filters and structures those data into a simplified format for future processing.

```
Def extract(apikey, num_of_days,
stock_symbol):
{...}
if response.status_code == 200: {...}
else: {...}
```

`Transform` process raw stock data to extract relevant fields(symbol, date, open, close, high, low, volume) for a specified number of days. Returns a simplified dataset ready for loading into a database.

```
Def transform(input_data,
num_of_days, stock_symbol):
#initialize {...}
#populate {...}
For date, values in
time_seriess.items():
  if datetime.striptime({...})
return stock_data
```

`load` is responsible for taking the processed stock data and loading it into a target database table. It handles the creation of the target table and inserts the stock data into it.

```
Def laod(cursor, target_table,
stock_data_input):
try:
cursor.execute("BEGIN;")/("DELETE
FROM")/("COMMIT;"){...}
except:{...}
```

`Train` is responsible for creating a view in the database, which is used to prepare data for a forecasting model. It then creates and trains the ml model using the data from that view.

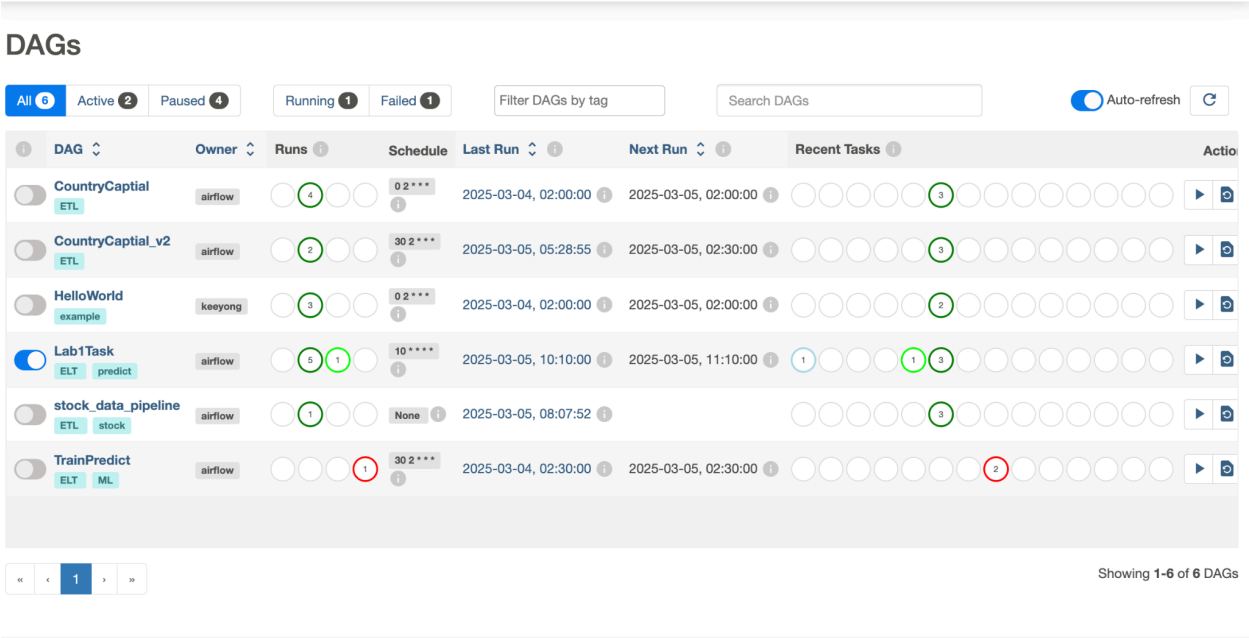
```
Def train(cur, train_input_table,
train_view, forecast_function_name):
create_view_sql = {...}
create_model_sql = {...}
try: {...}
except: {...}
```

`Predict` is for generating future stock price predictions using a periously trained forecasting model. It then stores the predictions in a forecast table and combines them with the historical data to create a final table that includes both actual and forecasted values.

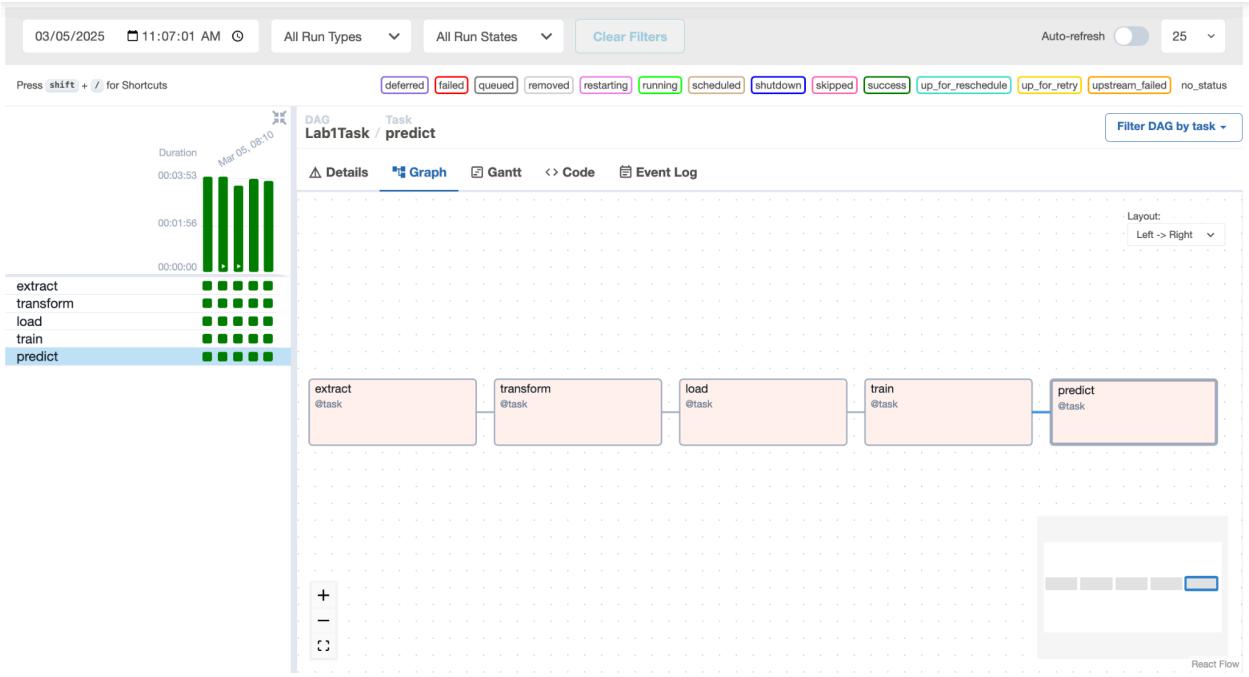
```
Def predict(cur,
forecast_function_name,
train_input_table, forecast_table,
final_table)
make_prediction_sql = {...}
Create_final_table_sql =
     {...} UNION ALL {...}
try: {...}
Except: {...}
```

Please see the Github link for more details.
https://github.com/lea2105/DATA226LAB1_SUN_LI

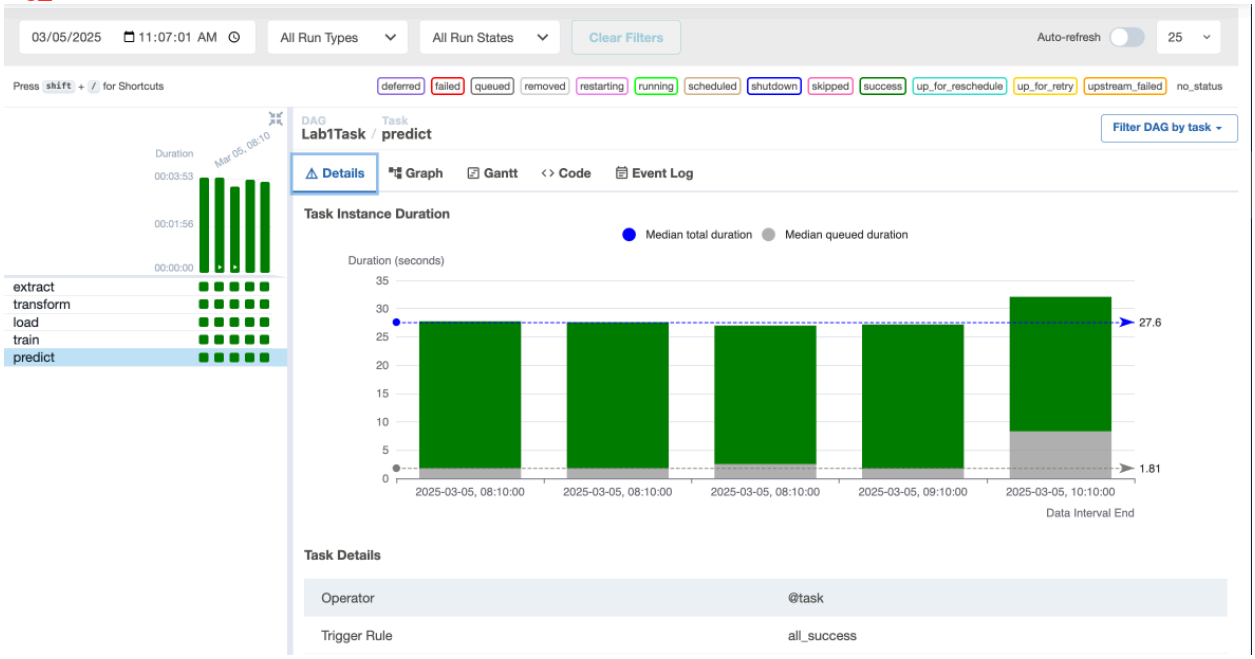
Airflow session:
dags



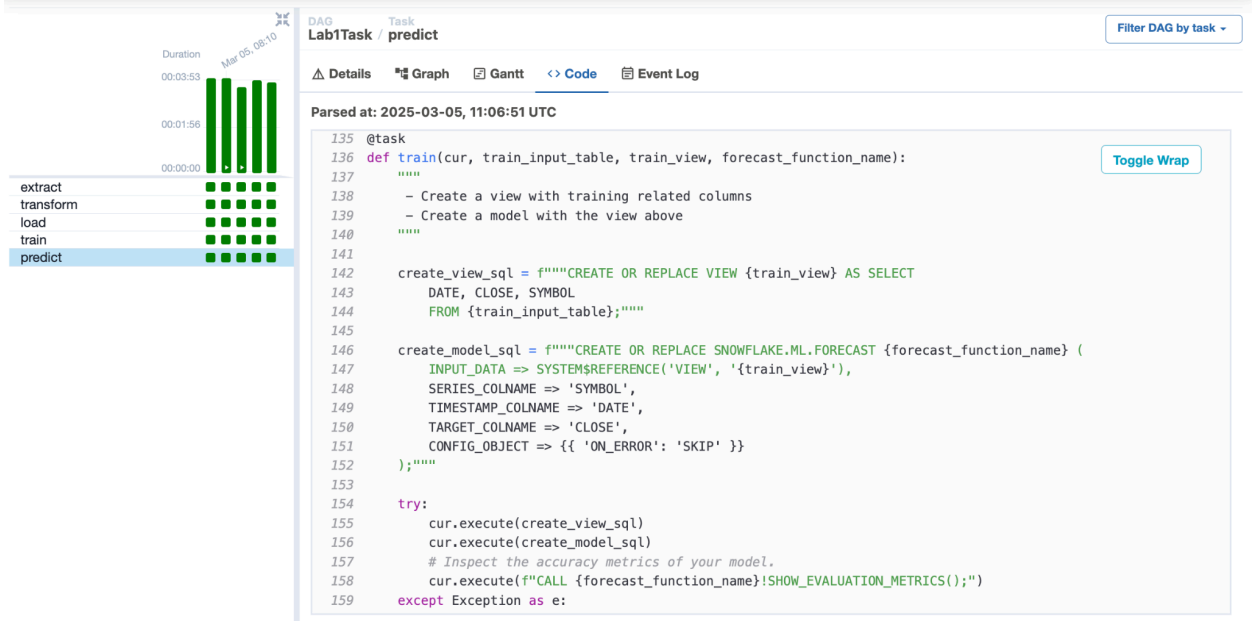
Dag_graph



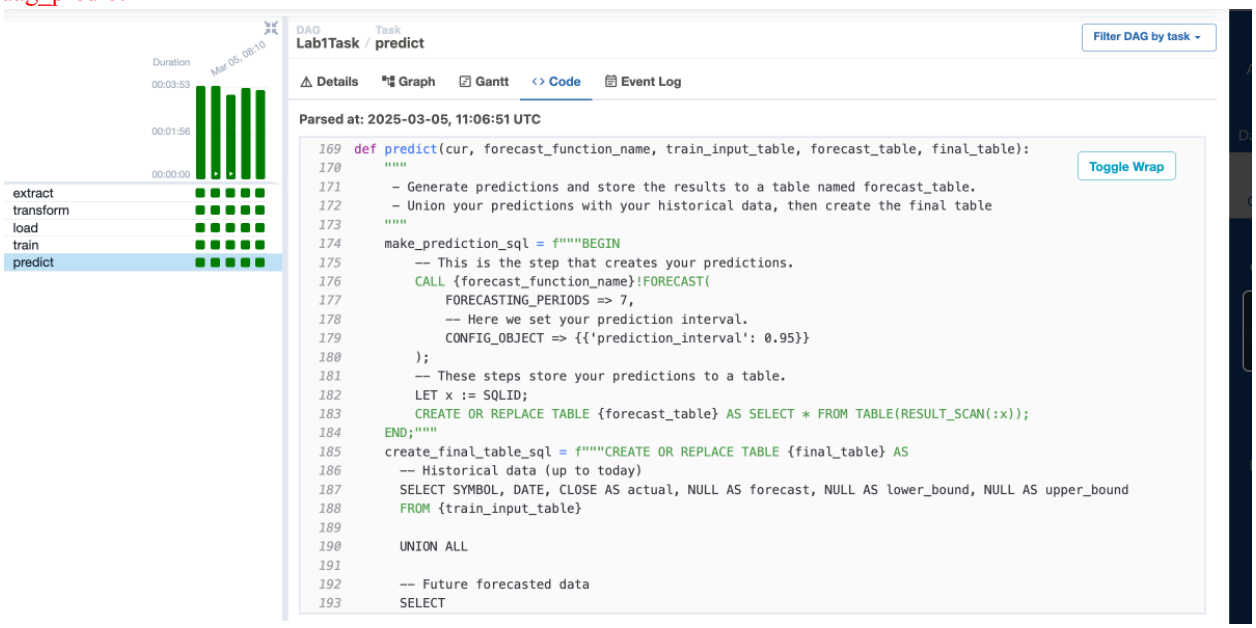
dag_details



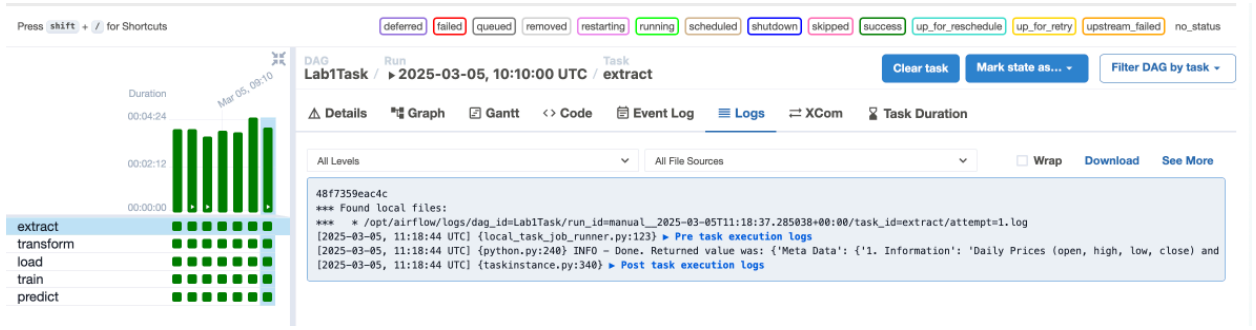
Dag_train



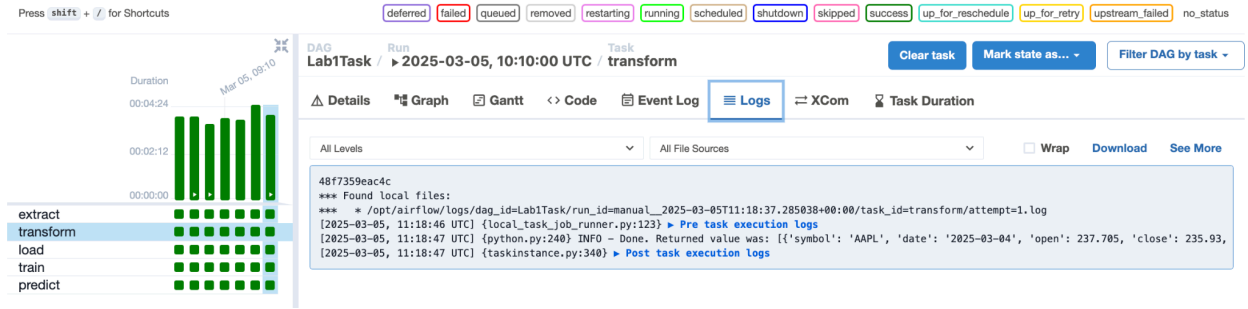
dag_predict



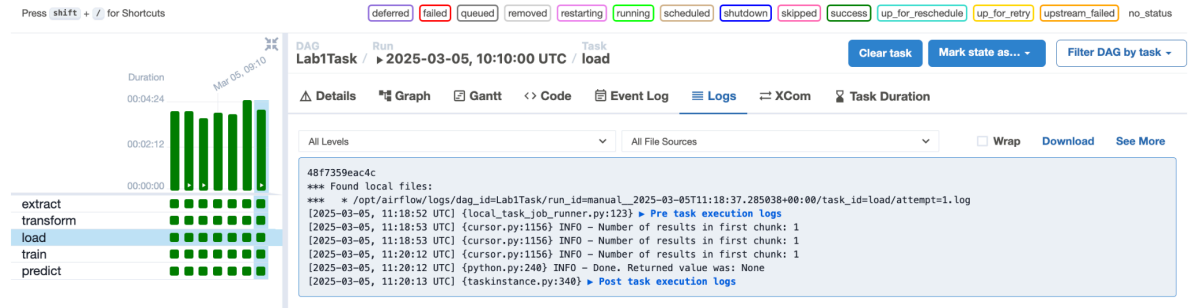
Log for extract



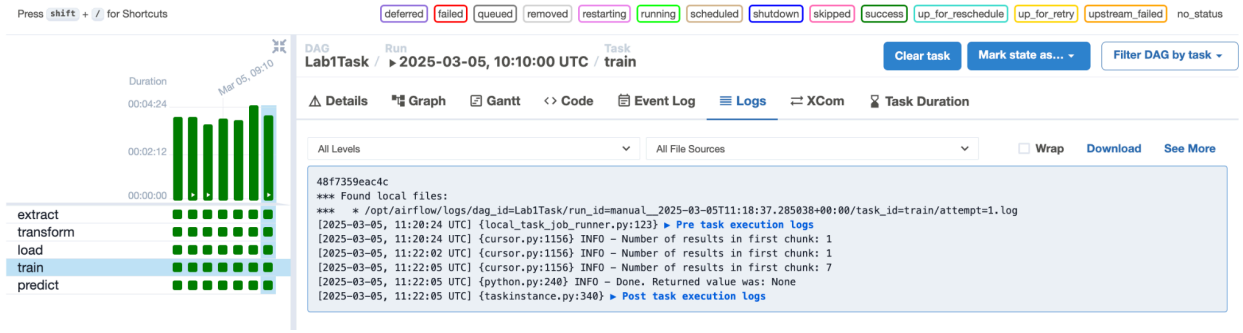
Log for transform



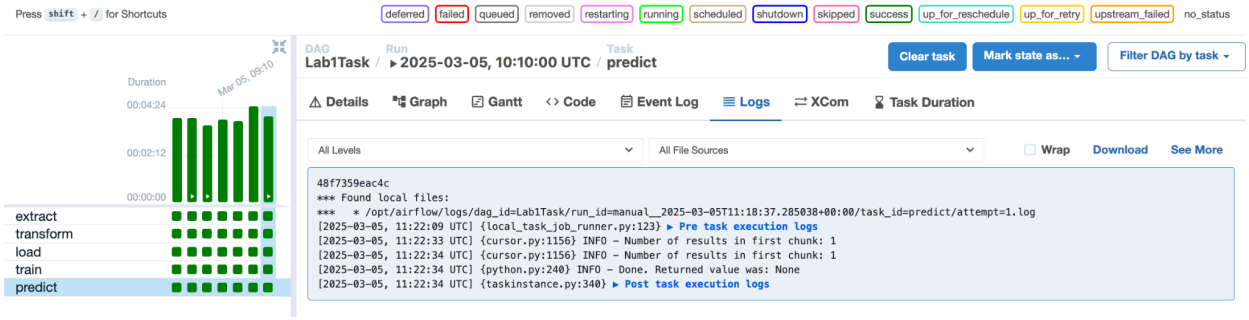
Log for load



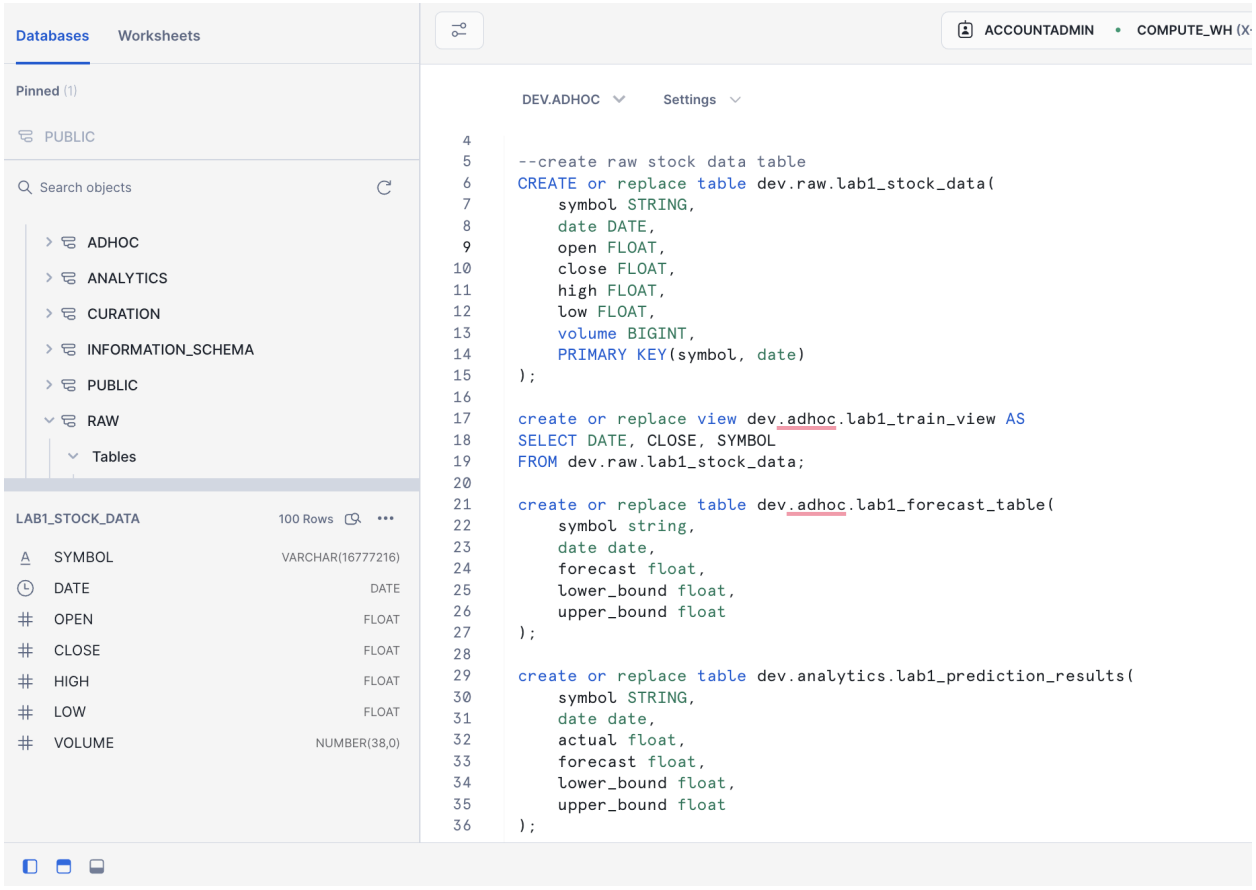
Log for train



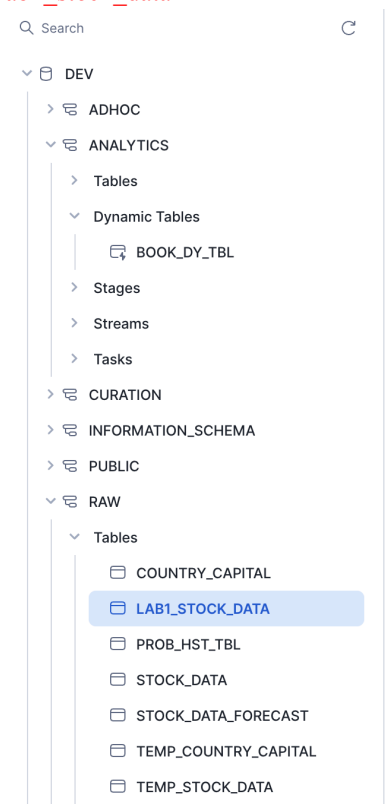
Log for predict

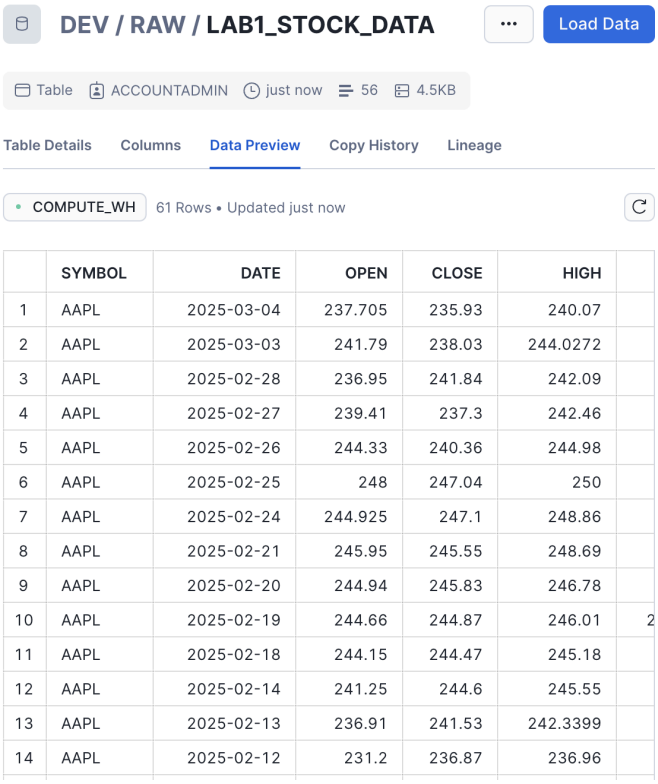


Snowflake session:



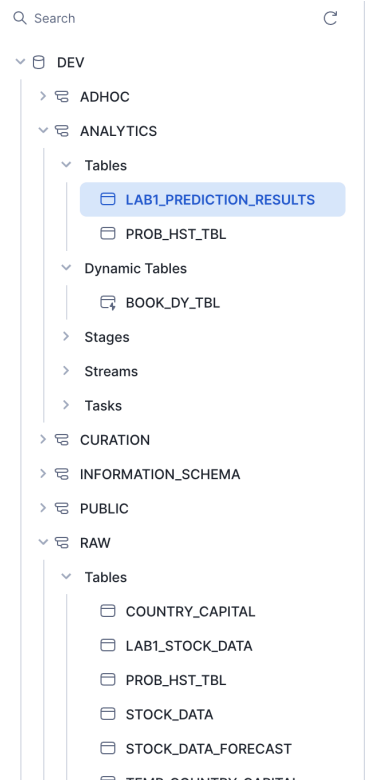
lab1_stock_data

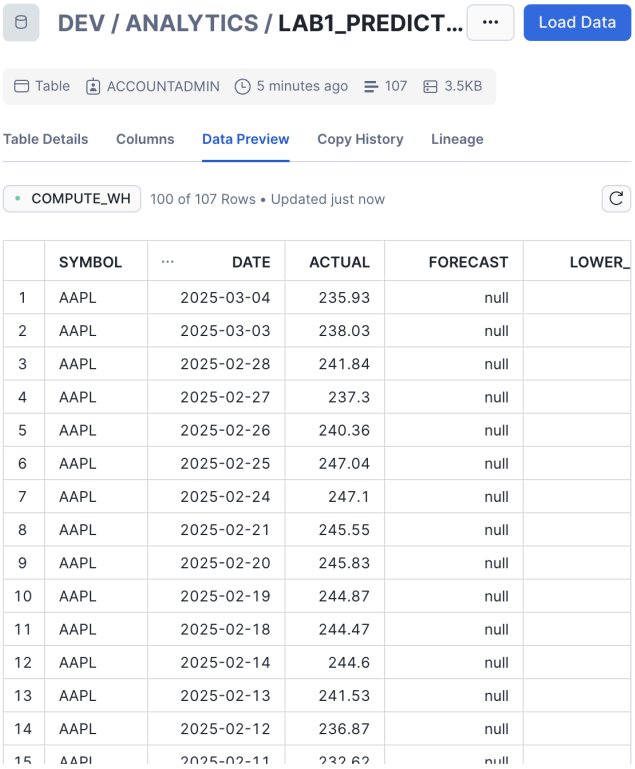




	SYMBOL	DATE	OPEN	CLOSE	HIGH
1	AAPL	2025-03-04	237.705	235.93	240.07
2	AAPL	2025-03-03	241.79	238.03	244.0272
3	AAPL	2025-02-28	236.95	241.84	242.09
4	AAPL	2025-02-27	239.41	237.3	242.46
5	AAPL	2025-02-26	244.33	240.36	244.98
6	AAPL	2025-02-25	248	247.04	250
7	AAPL	2025-02-24	244.925	247.1	248.86
8	AAPL	2025-02-21	245.95	245.55	248.69
9	AAPL	2025-02-20	244.94	245.83	246.78
10	AAPL	2025-02-19	244.66	244.87	246.01
11	AAPL	2025-02-18	244.15	244.47	245.18
12	AAPL	2025-02-14	241.25	244.6	245.55
13	AAPL	2025-02-13	236.91	241.53	242.3399
14	AAPL	2025-02-12	231.2	236.87	236.96

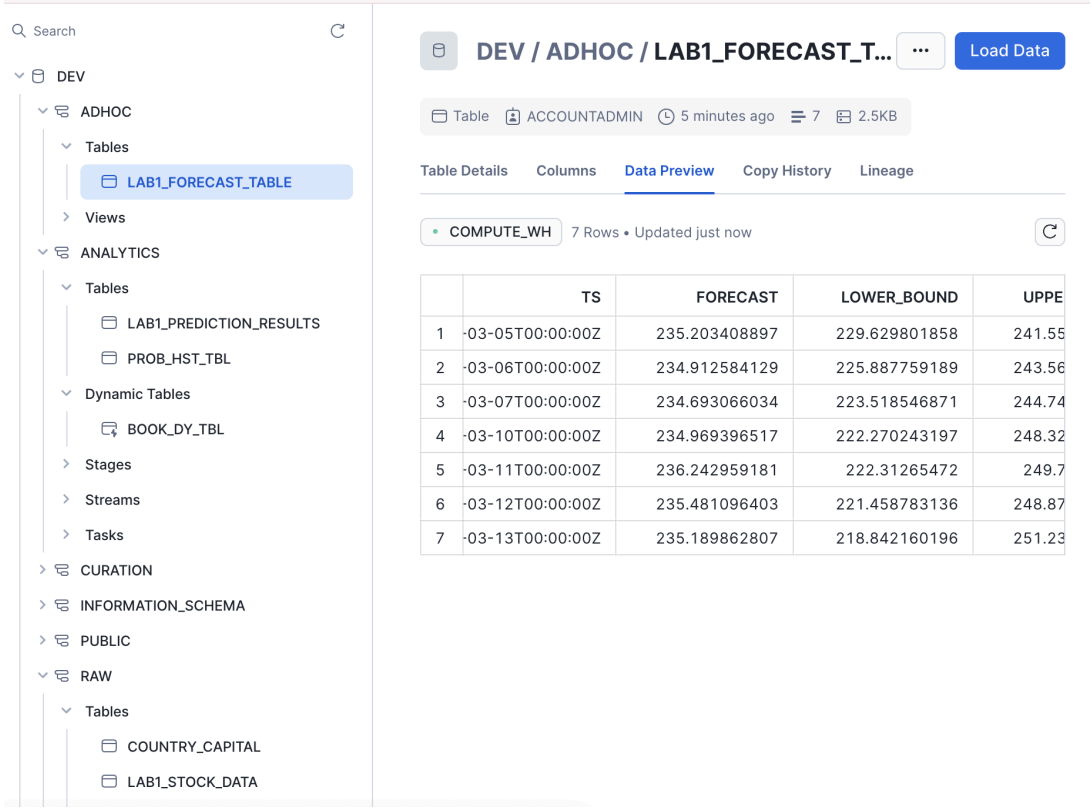
Lab1_prediction_results





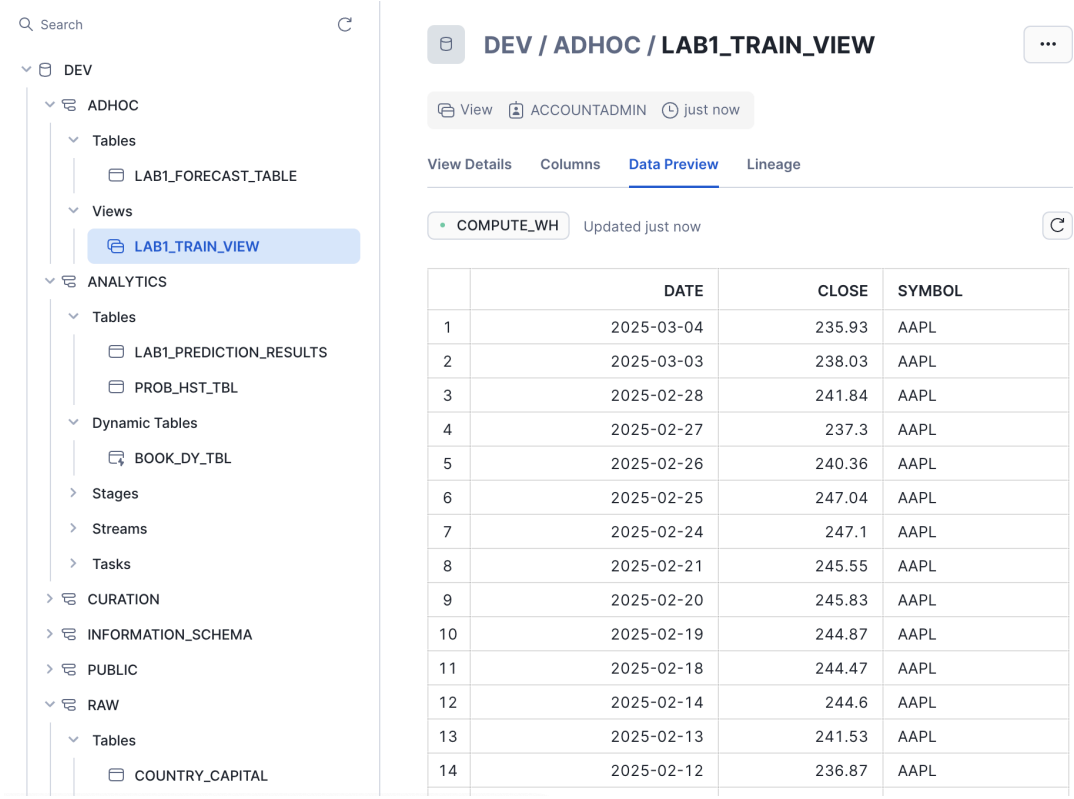
	SYMBOL	DATE	ACTUAL	FORECAST	LOWER_
1	AAPL	2025-03-04	235.93	null	
2	AAPL	2025-03-03	238.03	null	
3	AAPL	2025-02-28	241.84	null	
4	AAPL	2025-02-27	237.3	null	
5	AAPL	2025-02-26	240.36	null	
6	AAPL	2025-02-25	247.04	null	
7	AAPL	2025-02-24	247.1	null	
8	AAPL	2025-02-21	245.55	null	
9	AAPL	2025-02-20	245.83	null	
10	AAPL	2025-02-19	244.87	null	
11	AAPL	2025-02-18	244.47	null	
12	AAPL	2025-02-14	244.6	null	
13	AAPL	2025-02-13	241.53	null	
14	AAPL	2025-02-12	236.87	null	
15	AAPL	2025-02-11	232.62	null	

lab1_forcecast_table



	TS	FORECAST	LOWER_BOUND	UPPE
1	-03-05T00:00:00Z	235.203408897	229.629801858	241.55
2	-03-06T00:00:00Z	234.912584129	225.887759189	243.56
3	-03-07T00:00:00Z	234.693066034	223.518546871	244.74
4	-03-10T00:00:00Z	234.969396517	222.270243197	248.32
5	-03-11T00:00:00Z	236.242959181	222.31265472	249.7
6	-03-12T00:00:00Z	235.481096403	221.458783136	248.87
7	-03-13T00:00:00Z	235.189862807	218.842160196	251.23

Lab1_train_view



	DATE	CLOSE	SYMBOL
1	2025-03-04	235.93	AAPL
2	2025-03-03	238.03	AAPL
3	2025-02-28	241.84	AAPL
4	2025-02-27	237.3	AAPL
5	2025-02-26	240.36	AAPL
6	2025-02-25	247.04	AAPL
7	2025-02-24	247.1	AAPL
8	2025-02-21	245.55	AAPL
9	2025-02-20	245.83	AAPL
10	2025-02-19	244.87	AAPL
11	2025-02-18	244.47	AAPL
12	2025-02-14	244.6	AAPL
13	2025-02-13	241.53	AAPL
14	2025-02-12	236.87	AAPL