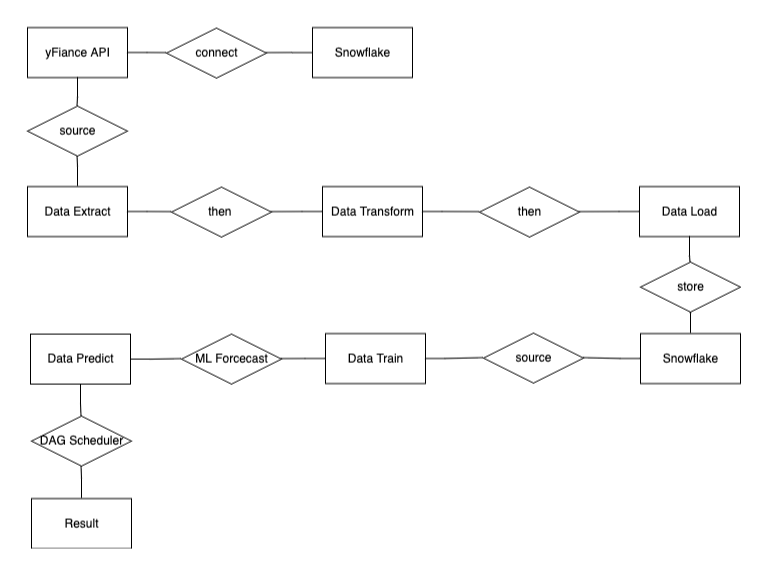
**About Our Project**

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: {...} |
| --- |

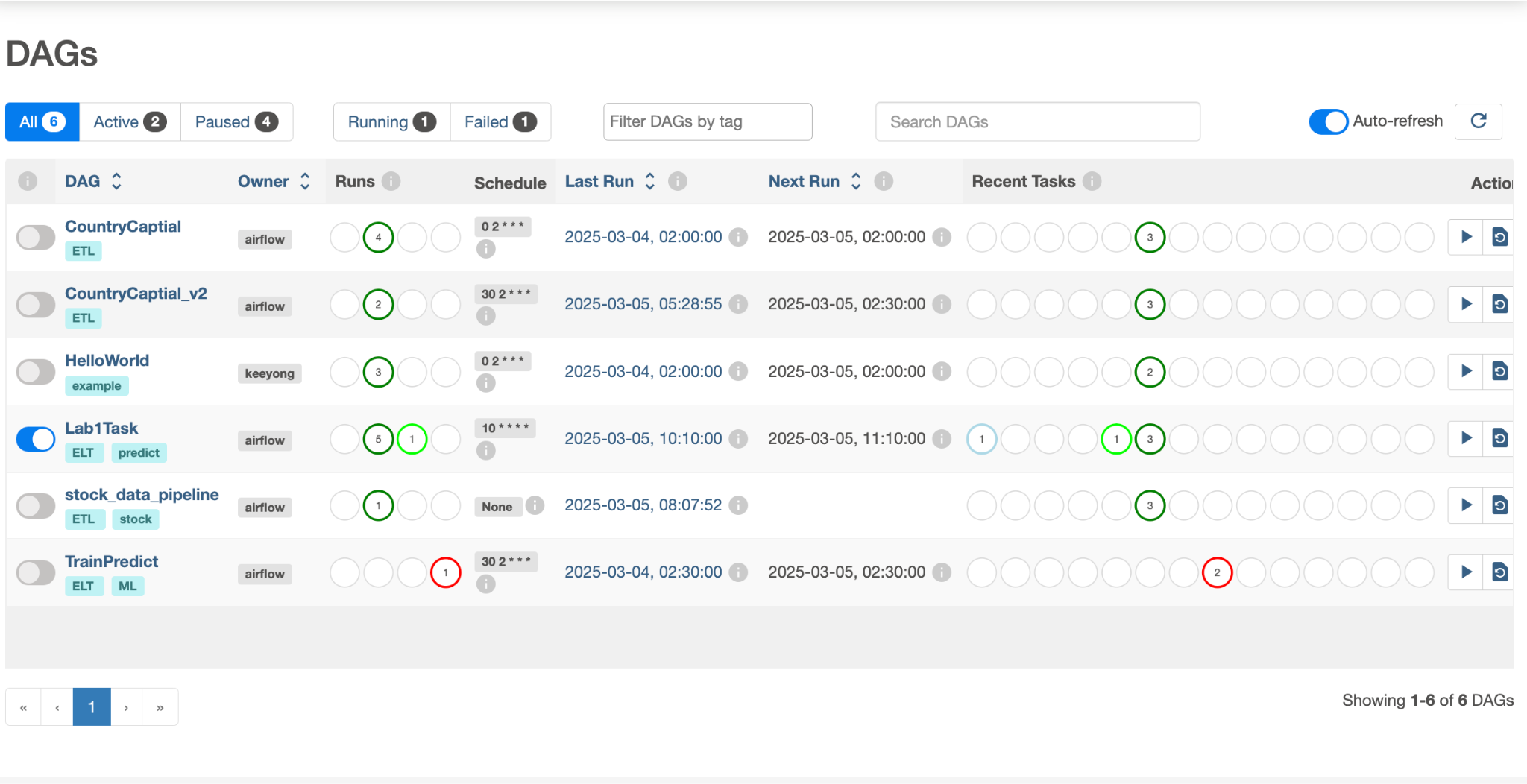
`Predict` is for generating future stock price predictions using a periously trained forecasting model. It then stores the predictions in a forecast table and combines them with the historical data to create a final table that includes both actual and forecasted values.

| Def predict(cur, forecast\_function\_name, train\_input\_table, forecast\_table, final\_table) make\_prediction\_sql = {...} Create\_final\_table\_sql =  {...} UNION ALL {...} try: {...} Except: {...} |
| --- |

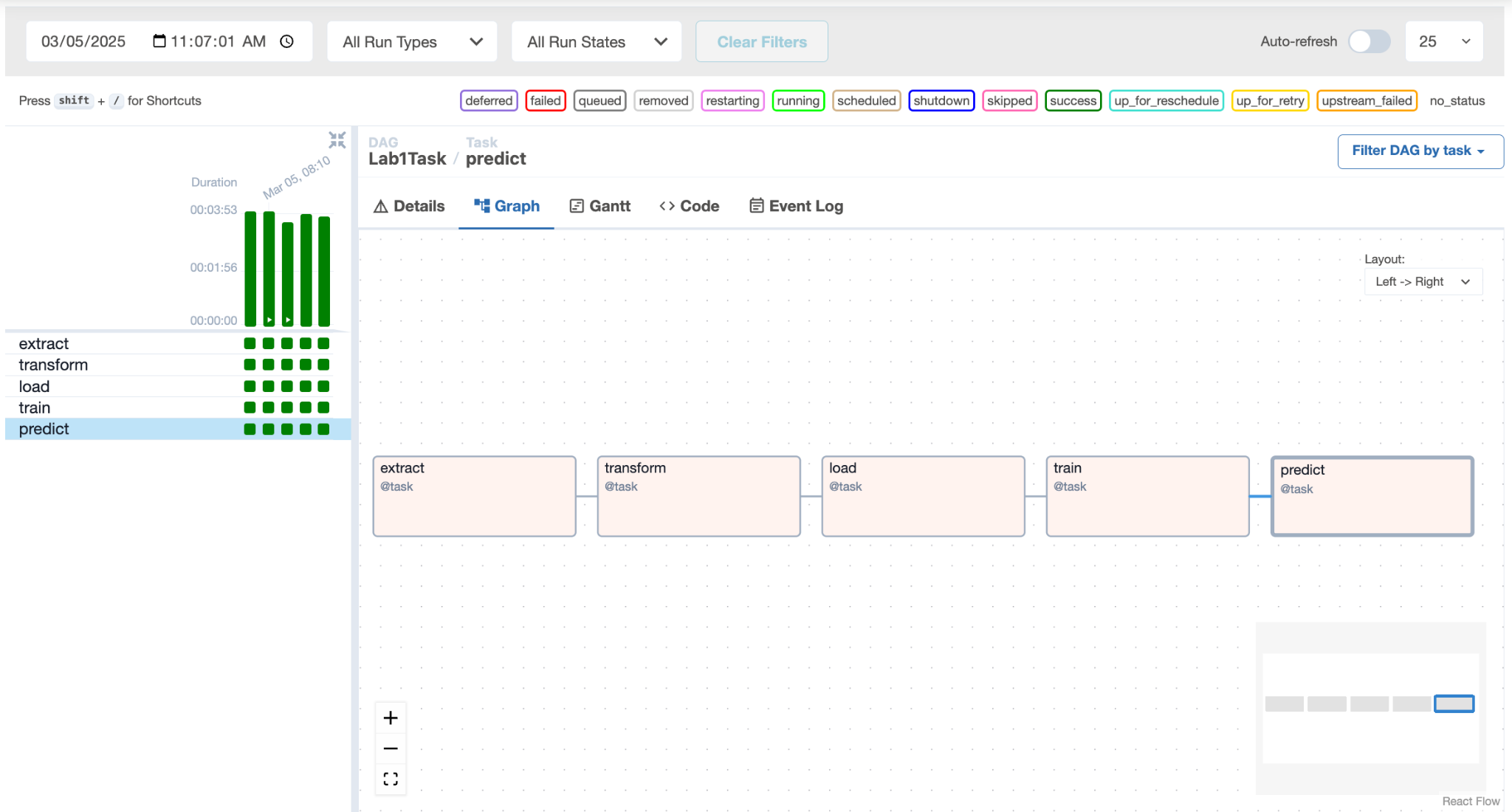
Please see the Github link for more details.

<https://github.com/lea2105/DATA226LAB1_SUN_LI>

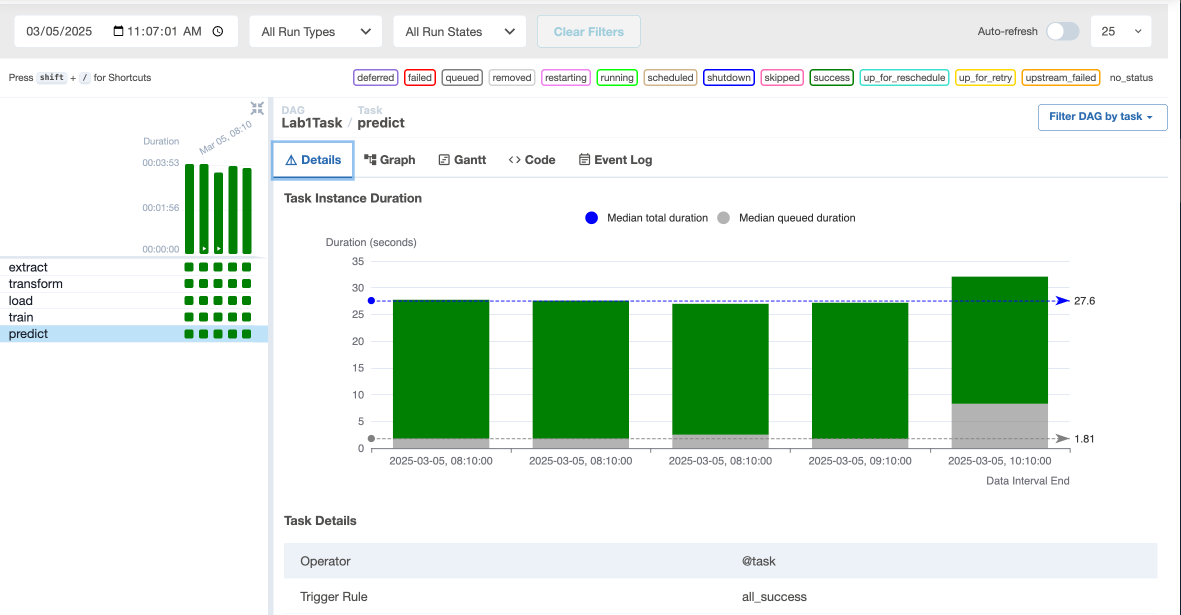
Airflow session:

dags

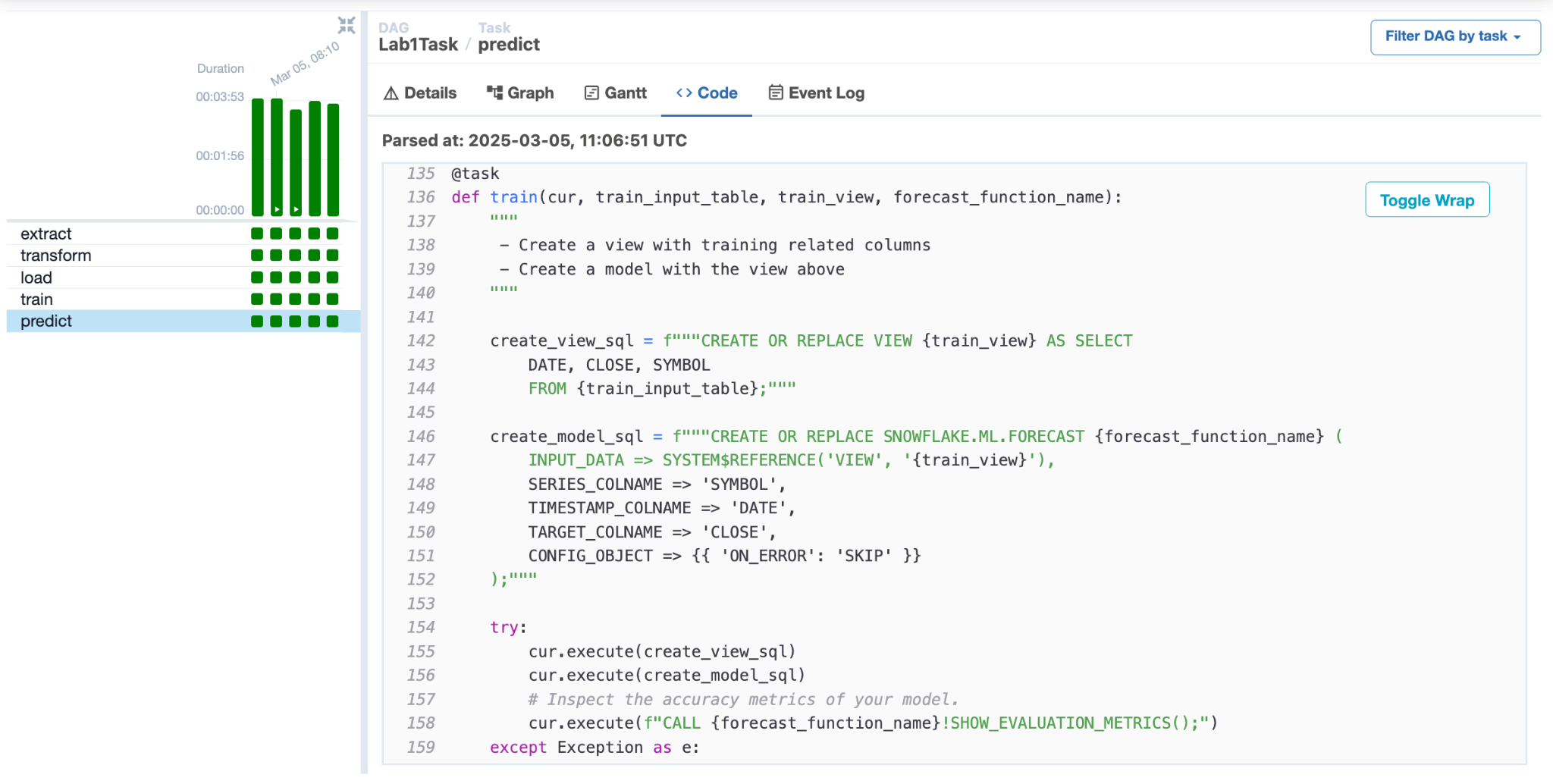
Dag\_graph

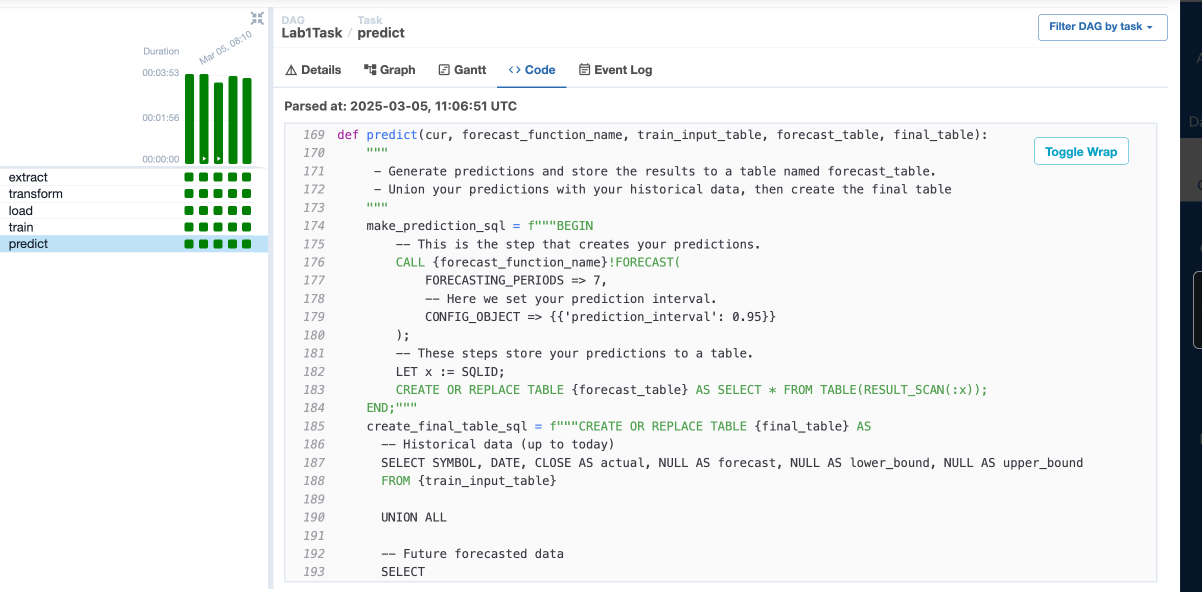


dag\_details

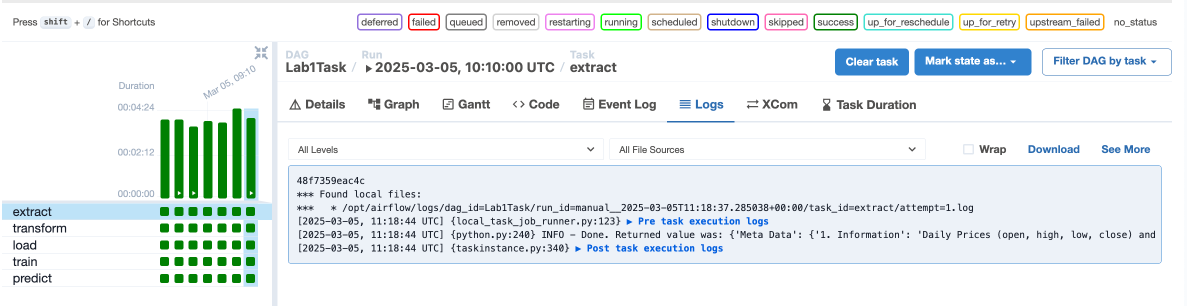


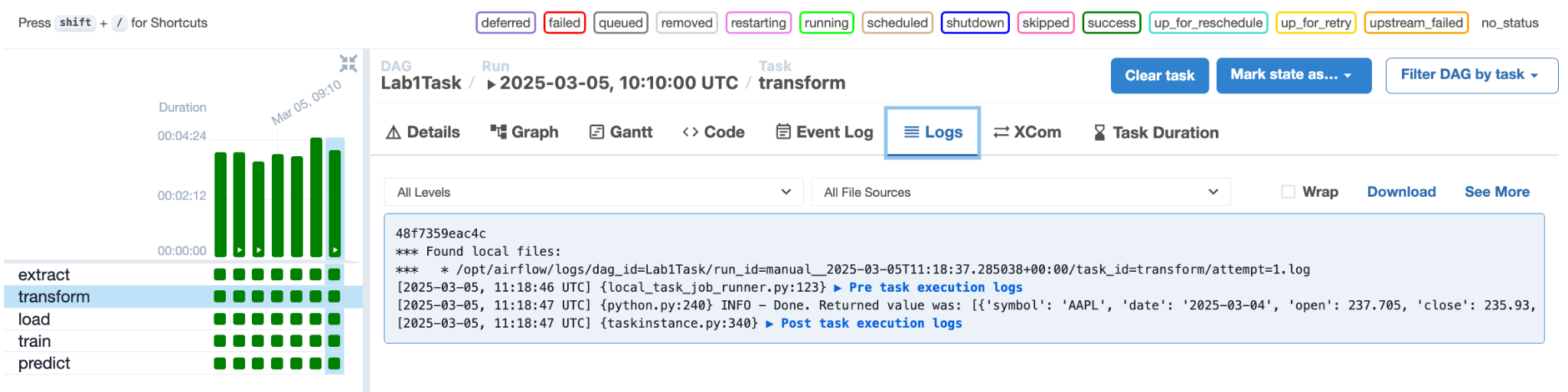
Dag\_train

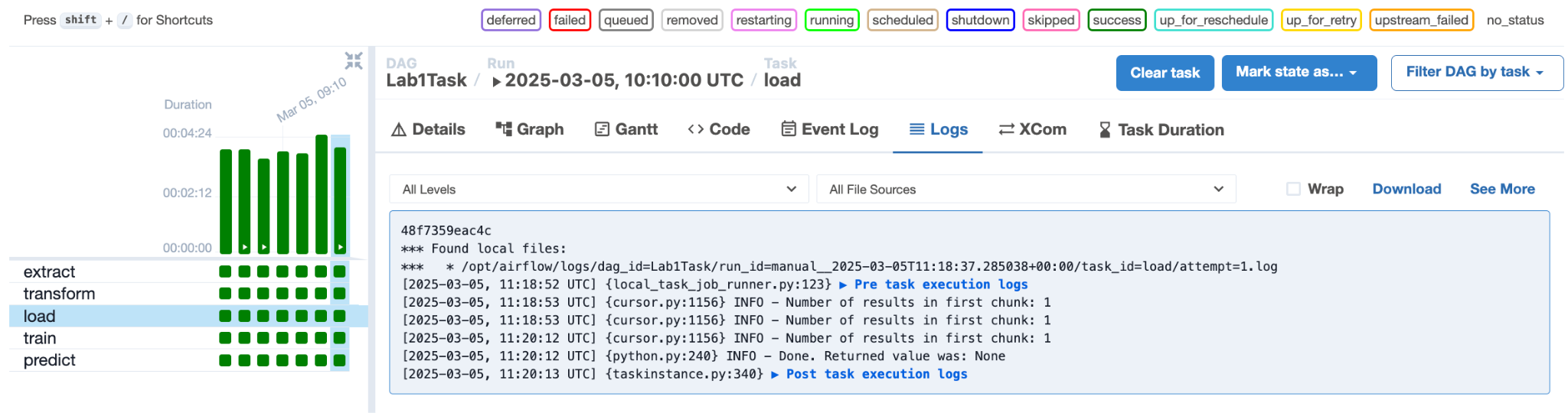


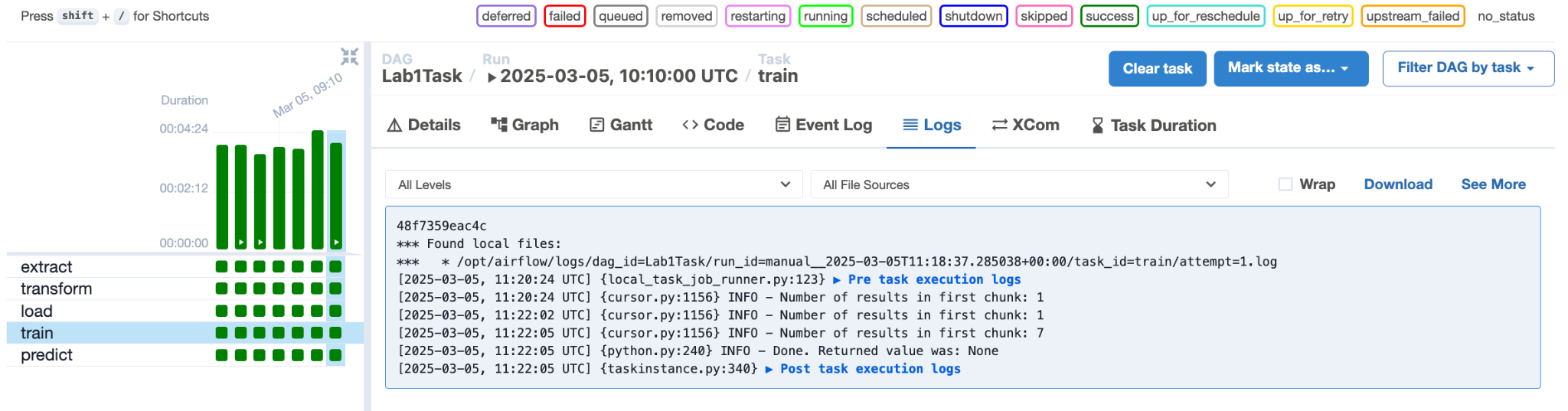
dag\_predict

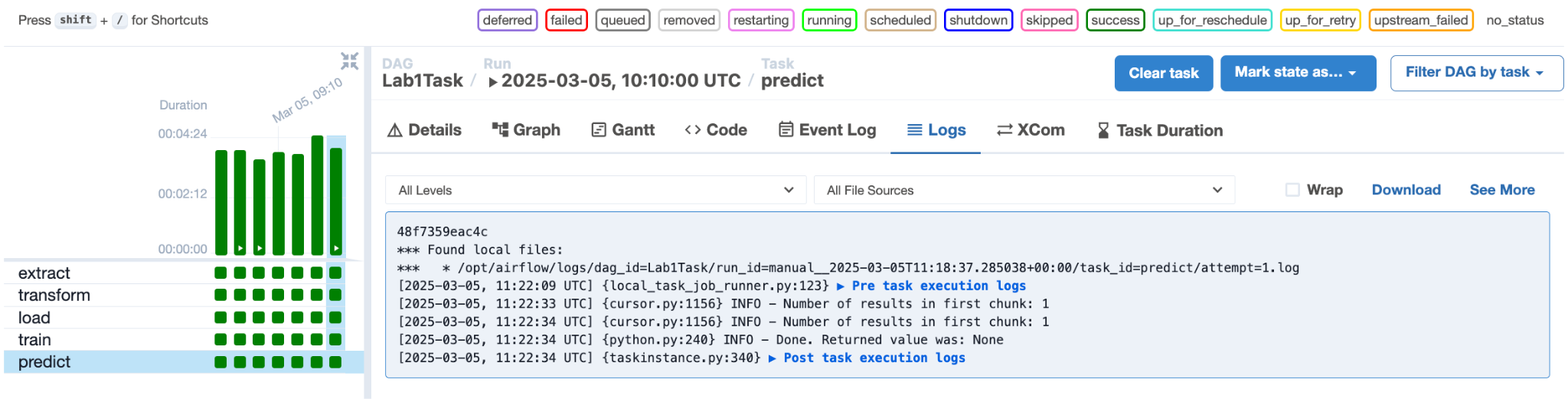
Log for extract



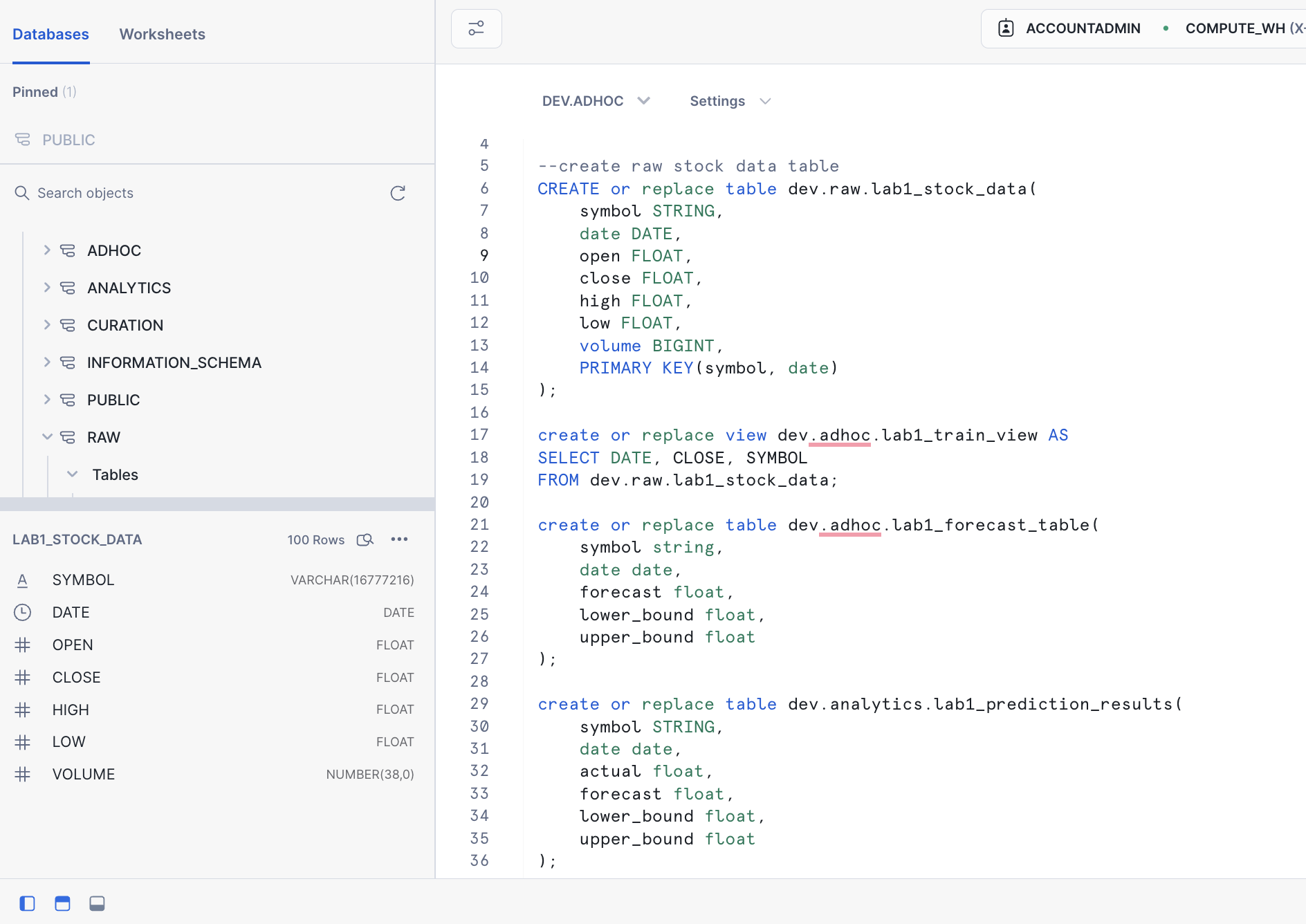
Log for transform

Log for load

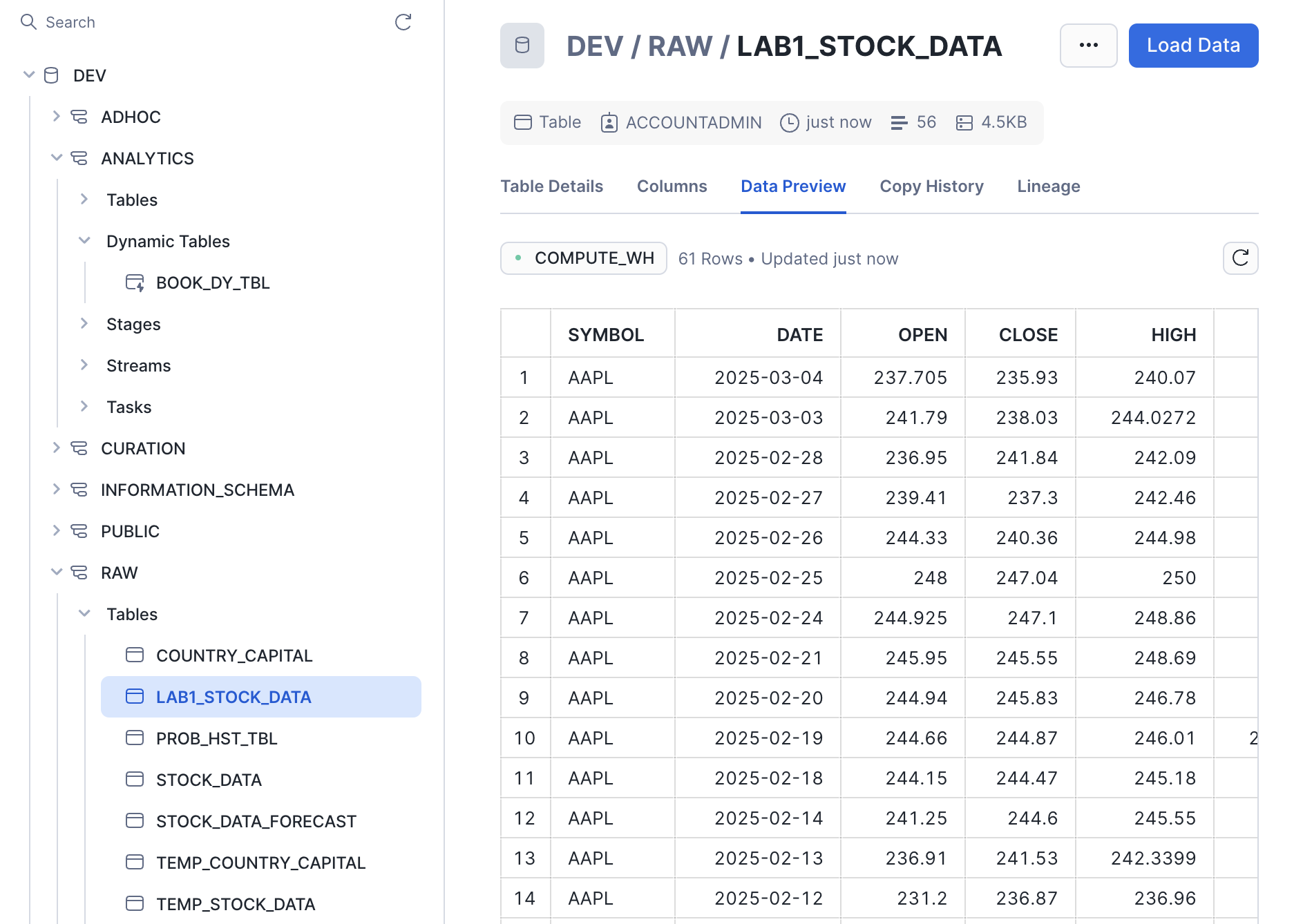
Log for train

Log for predict

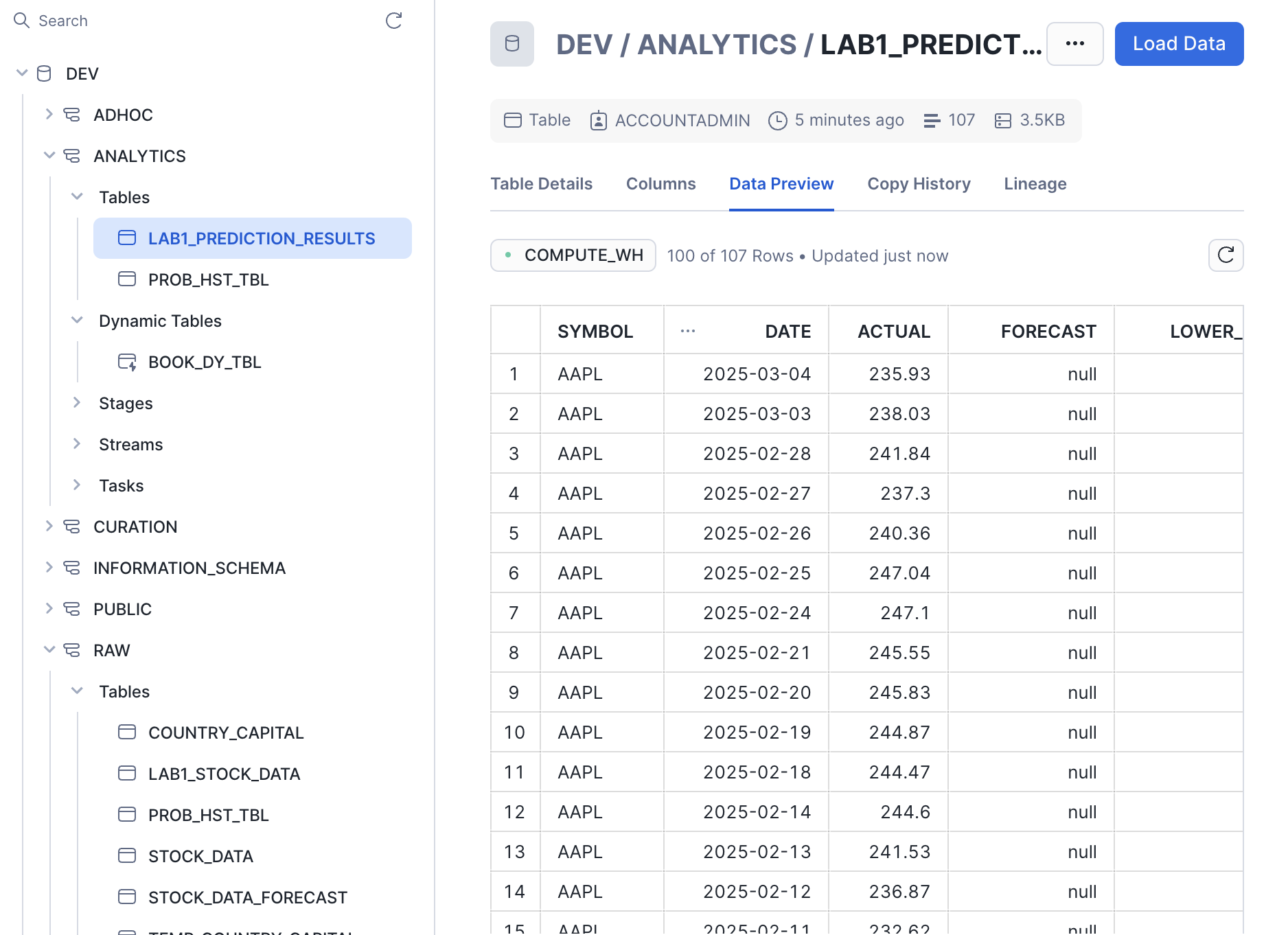
Snowflake session:



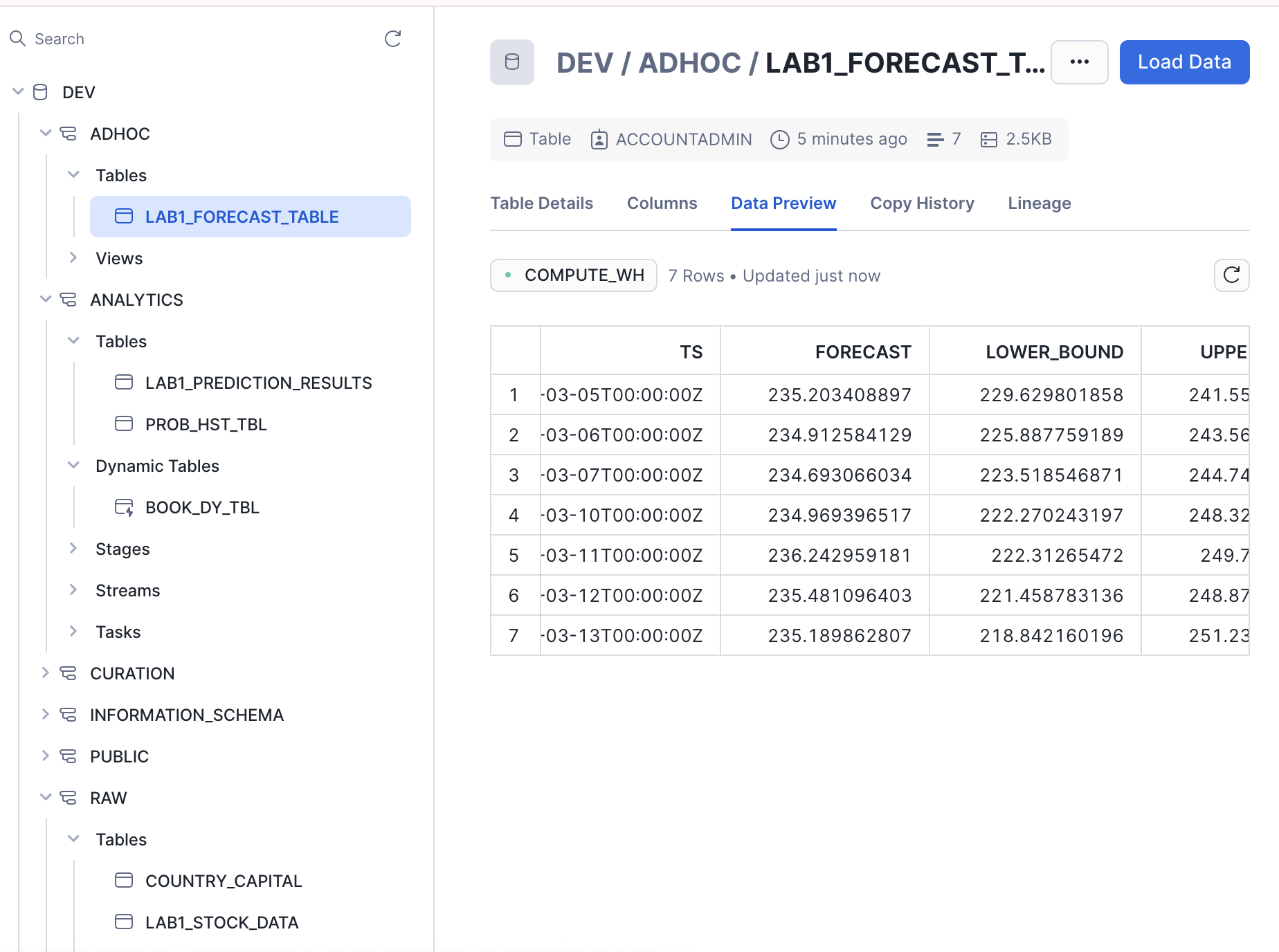
lab1\_stock\_data



Lab1\_prediction\_results



lab1\_forcecast\_table



Lab1\_train\_view

