Stock Data Load in Airflow

Shweta Ajay Shinde
Masters in Data Analytics, San Jose State University
Data 226: Data warehousing
Instructor: Keeyong Han
10th Oct 2024

Stock Data Load

Load Stocks Data in Snowflake Table using Airflow

- (+1) Import all the required python modules.

  We need to import all required python modules for smooth running of the code.

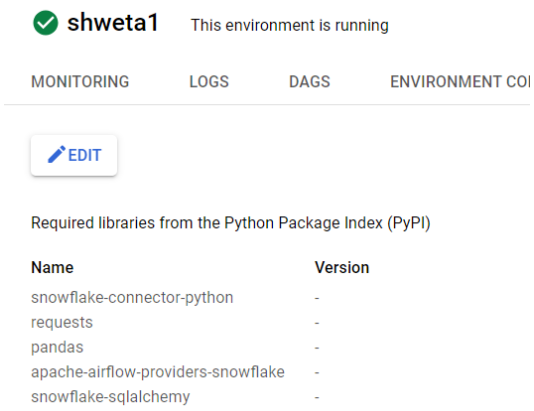
```
#Step 1 Import all required modules
from airflow import DAG
from airflow.models import Variable
from airflow.decorators import task
from airflow.providers.snowflake.hooks.snowflake import SnowflakeHook

from datetime import timedelta
from datetime import datetime
import snowflake.connector
import requests
import pandas as pd
```

- (+1) Ensure that any missing package(s) are added to the PYPI packages

  Step 1: After login into Cloud Composer click on PYPI packages.

  Step 2: Add all the require packages and save it



- (+3) Create tasks using @task decorator (refer to GitHub link to an external site.).

  The @task decorator tells the system that a function is a special task that can be run separately. It allows the function to work in the background or at the same time as other tasks. This is useful when you want to do multiple things at once, like processing data or sending emails. By using @task, you make it easier to organize and run different tasks in a program. For task we need to import: **from airflow.decorators import task**

```
@task
def return_last_90d_price(symbol):
    """
     - return the last 90 days of the stock prices of symbol as a list of json strings
    """
    vantage_api_key = Variable.get('apikey')
    url = f'https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol={symbol}&apikey={vantage_api_key}'
    r = requests.get(url)
    data = r.json()
    symbol_value = data["Meta Data"]["2. Symbol"]
```

Stock Data Load

```python
@task
def create_load_incremental(records):
    staging_table = "dev.stock.merck_stock_stage"
    target_table = "dev.stock.merck_stock_price_incremental"
    conn = return_snowflake_conn()
    try:
        conn.execute(f"""
            CREATE TABLE IF NOT EXISTS {target_table} (
                date DATE PRIMARY KEY NOT NULL,
                open DECIMAL(10, 2) NOT NULL,
                high DECIMAL(10, 2) NOT NULL,
                low DECIMAL(10, 2) NOT NULL,
```

```python
@task
def create_load_full(records):
    target_table = "dev.stock.merck_stock_price_full"
    conn = return_snowflake_conn()
    try:
        conn.execute(f"""
            CREATE OR REPLACE TABLE  {target_table} (
                date DATE PRIMARY KEY NOT NULL,
                open DECIMAL(10, 2) NOT NULL,
                high DECIMAL(10, 2) NOT NULL,
                low DECIMAL(10, 2) NOT NULL,
                close DECIMAL(10, 2) NOT NULL,
```

- (+1) Set up a variable for Alpha Vantage API key

  Under Admin we find option Variable where we can store all our confidential information in key value pair.

Key	Val	Description
apikey	********	Api key to access source Data
SNOWFLAKE_ACCOUNT	mihzyek-aeb64943	Snowflake account details
SNOWFLAKE_DATABASE	dev	
SNOWFLAKE_PASSWORD	********	Snowflake Password
SNOWFLAKE_SCHEMA	stock	
SNOWFLAKE_USER	shweta12shinde	Snowflake user
SNOWFLAKE_WAREHOUSE	compute_wh	

In code we need **from airflow.models import Variable**

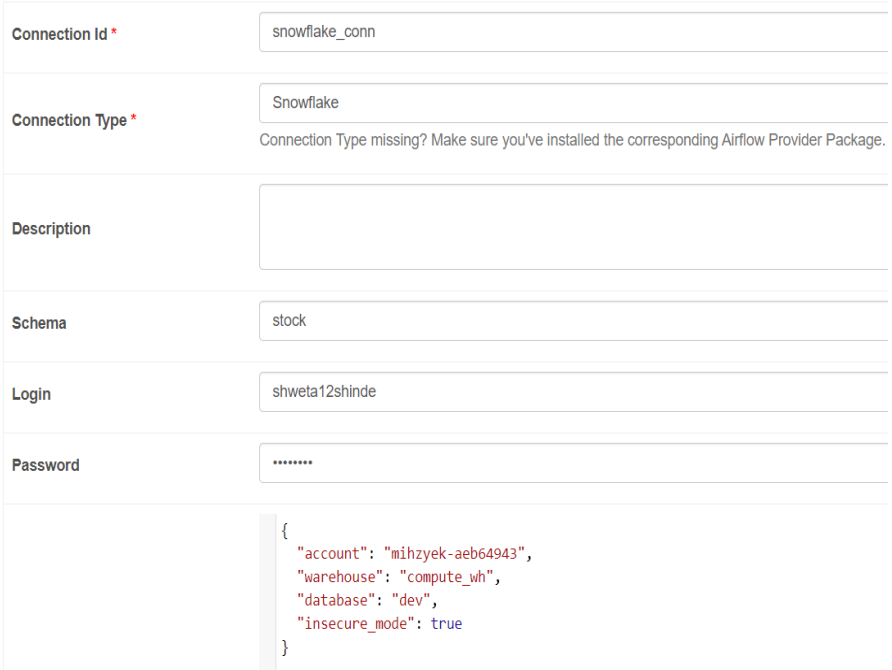
```python
def return_snowflake_conn():
    user_id = Variable.get('SNOWFLAKE_USER')
    password = Variable.get('SNOWFLAKE_PASSWORD')
    account = Variable.get('SNOWFLAKE_ACCOUNT')

    conn = snowflake.connector.connect(
        user=user_id,
        password=password,
        account=account,
        warehouse=Variable.get('SNOWFLAKE_WAREHOUSE'),
        database=Variable.get('SNOWFLAKE_DATABASE'),
        schema=Variable.get('SNOWFLAKE_SCHEMA')
    )
    return conn.cursor()
```

- (+2) Set up Snowflake Connection (refer to GitHub link to an external site.)

  We create this connection to connect with Snowflake

Stock Data Load



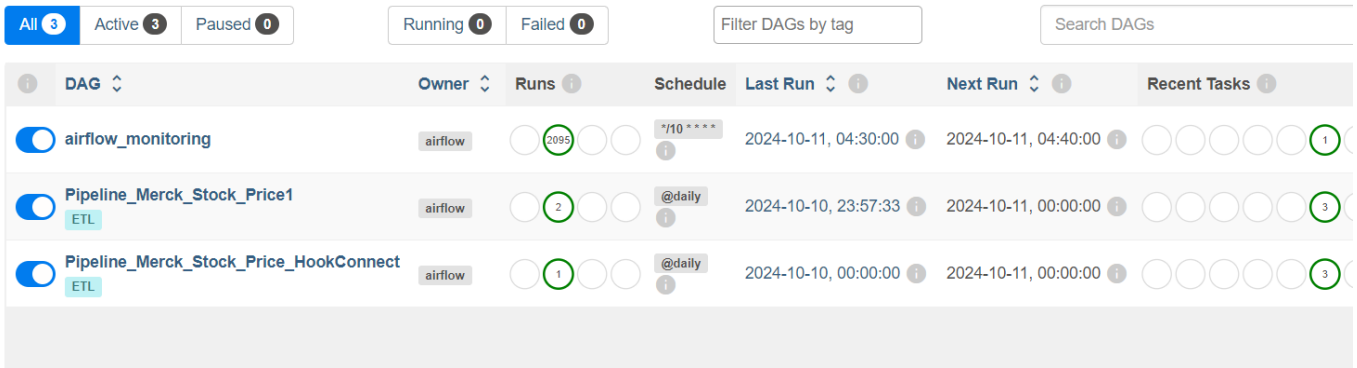
Code Screenshot:-

```
def return_snowflake_conn():
    # Initialize the SnowflakeHook
    hook = SnowflakeHook(snowflake_conn_id='snowflake_conn')
    # Execute the query and fetch results
    conn = hook.get_conn()
    return conn.cursor()
```

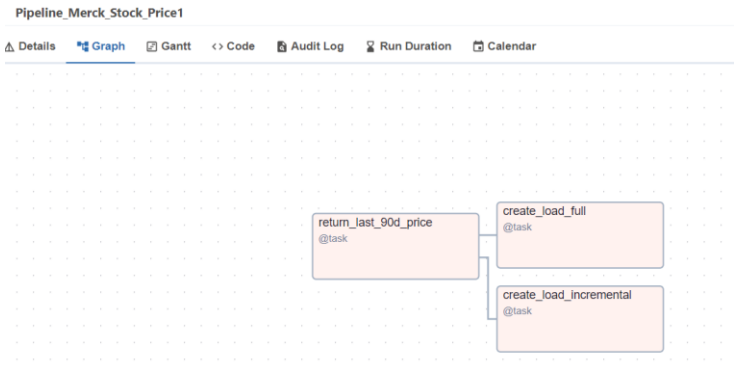
- (+4) Ensure the overall DAG runs successfully

GitHub link to the code:-
https://github.com/ShindeShwetaK/DW_Project/blob/main/Merck_stock_pipeline_new.py

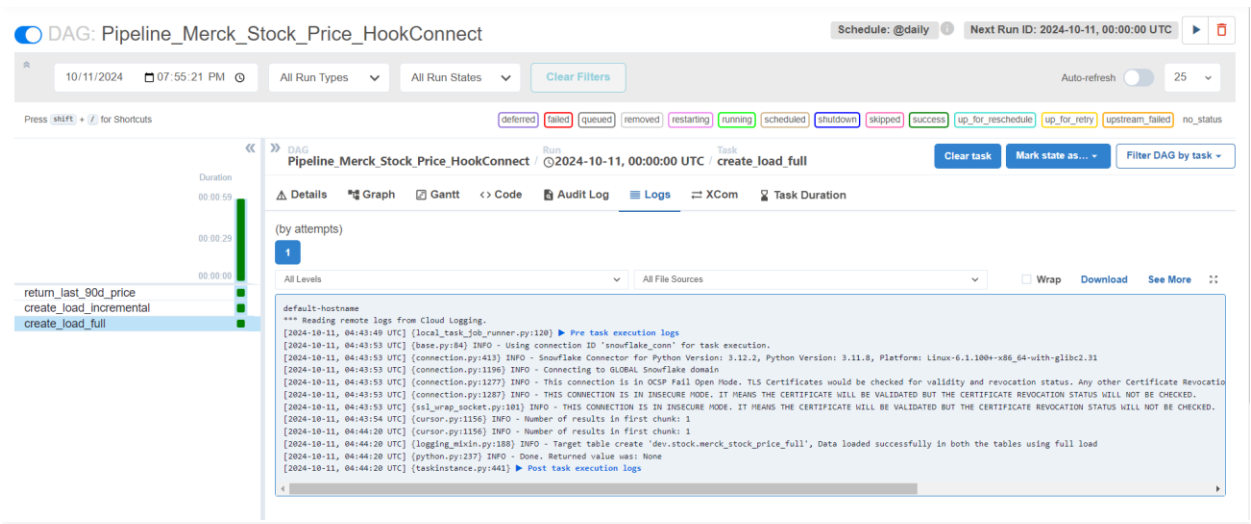
- (+2) Capture two screenshot of your Airflow Web UI (examples to follow)
  - One with the Airflow homepage showing the DAG (④)

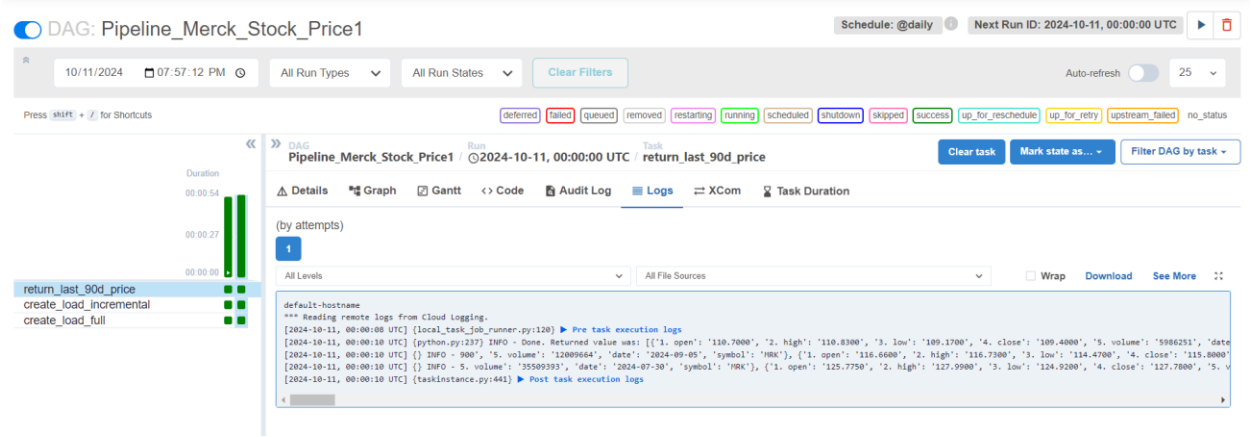


Stock Data Load



The other with the log screen of the DAG (⑤)

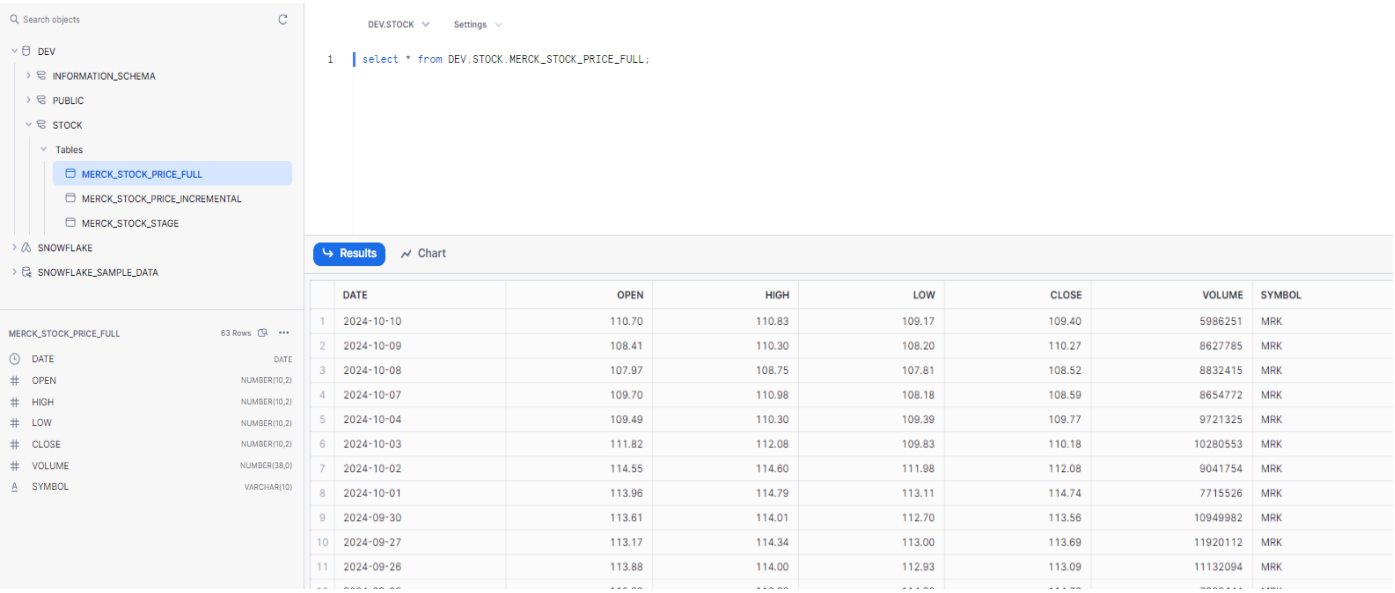




Stock Data Load

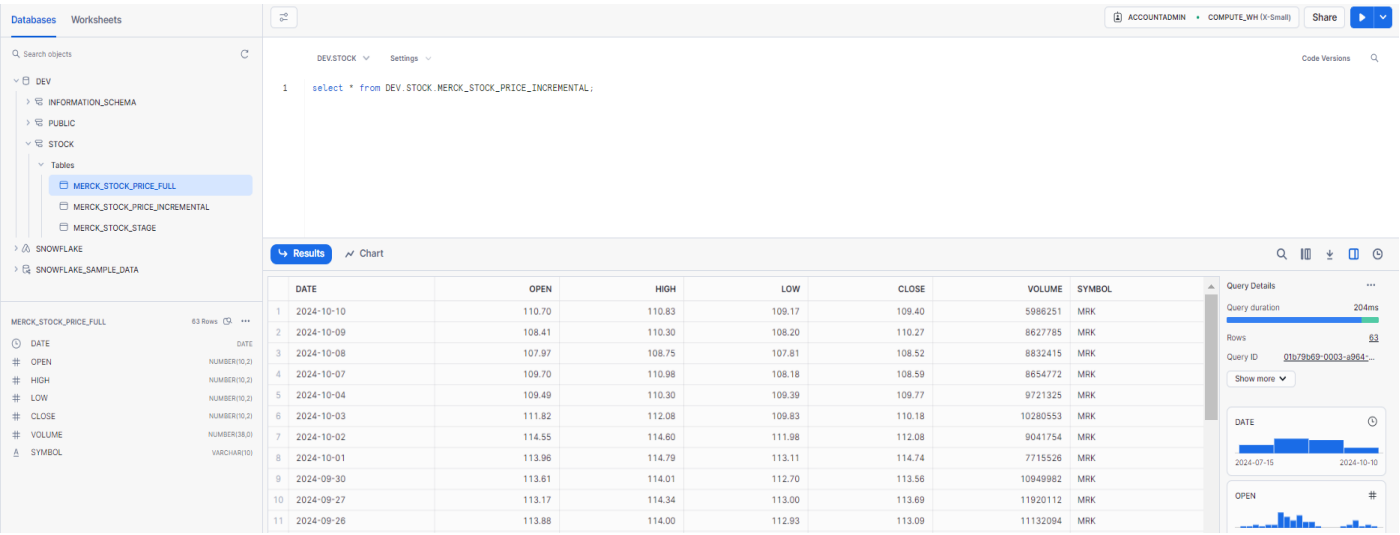
Snowflake Screenshot:-

Full Load



	DATE	OPEN	HIGH	LOW	CLOSE	VOLUME	SYMBOL
1	2024-10-10	110.70	110.83	109.17	109.40	5986251	MRK
2	2024-10-09	108.41	110.30	108.20	110.27	8627785	MRK
3	2024-10-08	107.97	108.75	107.81	108.52	8832415	MRK
4	2024-10-07	109.70	110.98	108.18	108.59	8654772	MRK
5	2024-10-04	109.49	110.30	109.39	109.77	9721325	MRK
6	2024-10-03	111.82	112.08	109.83	110.18	10280553	MRK
7	2024-10-02	114.55	114.60	111.98	112.08	9041754	MRK
8	2024-10-01	113.96	114.79	113.11	114.74	7715526	MRK
9	2024-09-30	113.61	114.01	112.70	113.56	10949982	MRK
10	2024-09-27	113.17	114.34	113.00	113.69	11920112	MRK
11	2024-09-26	113.88	114.00	112.93	113.09	11132094	MRK

Incremental Load



	DATE	OPEN	HIGH	LOW	CLOSE	VOLUME	SYMBOL
1	2024-10-10	110.70	110.83	109.17	109.40	5986251	MRK
2	2024-10-09	108.41	110.30	108.20	110.27	8627785	MRK
3	2024-10-08	107.97	108.75	107.81	108.52	8832415	MRK
4	2024-10-07	109.70	110.98	108.18	108.59	8654772	MRK
5	2024-10-04	109.49	110.30	109.39	109.77	9721325	MRK
6	2024-10-03	111.82	112.08	109.83	110.18	10280553	MRK
7	2024-10-02	114.55	114.60	111.98	112.08	9041754	MRK
8	2024-10-01	113.96	114.79	113.11	114.74	7715526	MRK
9	2024-09-30	113.61	114.01	112.70	113.56	10949982	MRK
10	2024-09-27	113.17	114.34	113.00	113.69	11920112	MRK
11	2024-09-26	113.88	114.00	112.93	113.09	11132094	MRK