Airflow Mini-Project

DAG Scheduling

Estimated Time: 3-5 hours

In this project, you'll use Apache Airflow to create a data pipeline to extract online stock market data and deliver analytical results. You'll use Yahoo Finance as the data source. Yahoo Finance provides intra-day market price details down a one-minute interval.

You'll work with two stock symbols: AAPL and TSLA. The workflow should be scheduled to run at 6 PM on every weekday (Mon - Fri), which functions as below:

- Download the daily price data with one minute interval for the two symbols. Each symbol will have a separate task, Task1 (t1) and Task2 (t2), which run independently and in parallel.
- Save both datasets into CSV files and load them into HDFS. Each symbol will have a separate task, Task3 (t3) and Task4 (t4), which run independently and in parallel.
- Run your custom query on the downloaded dataset for both symbols, Task5 (t5). Before this step executes, all previous tasks must complete.
- Use Celery Executor in Airflow to run the job.

The source data has the following schema.

Column	**Type**
date time	STRING
open	DECIMAL
high	DECIMAL (highest price within the time interval)
low	DECIMAL (lowest price within the time interval)
close	DECIMAL (the last price of the time interval)

adj close	DECIMAL
volume	DECIMAL

Learning Objectives

With this mini-project, you will utilize Apache Airflow to orchestrate your pipeline, exercise the DAG creation, uses of various operators (BashOperator, PythonOperator, etc), setting up order of operation of each task.
In this mini-project, you will gain familiarity with how to use Apache Airflow to automate data pipelining tasks. Along the way, you will learn how to:

- Use Apache Airflow to orchestrate your pipeline
- Exercise DAG creation
- Use Various Airflow operators like BashOperator and PythonOperator
- Set up the order operation of each task
- Use Celery Executor to run your job

Prerequisites

- Install Airflow: http://airflow.apache.org/docs/stable/installation.html
- For this project, you'll need Yahoo Finance's Python library. You can install it using the command: **pip install yfinance**
- Install pandas using the command**: pip install pandas**

Instructions

1. Create the Airflow DAG

Create the DAG object with name "marketvol". Set the default arguments. Your DAG run should follows:

- Start time and date: 6 PM on the current date.
- Job interval: runs once daily.
- Only runs on weekdays (Mon-Fri).
- If failed: retry twice with a 5-minute interval.

```
dag = DAG(
    'marketvol',
    default_args=default_args,
    description='A simple DAG',
    schedule_interval=timedelta(days=1),
)
```

2. Create operators associated with the DAG

2.1. Create a BashOperator to initialize a temporary directory for data download (t0)

This temporary directory should be named after the execution date (for example “2020-09-24”) so that the data for each date is placed accordingly. You can use the shell command to create the directory:

```
mkdir -p /tmp/data/2020-09-24
```

2.2. Create a PythonOperator to download the market data (t1, t2)

This example downloads and saves the file. Make your own function the Airflow runs with the stock symbol type as a parameter.

```
import yfinance as yf
start_date = date.today()
end_date = start_date + timedelta(days=1)
tsla_df = yf.download('TSLA', start=start_date, end=end_date, interval='1m')
tsla_df.to_csv("data.csv" header=False)
```

The PythonOperator should call the above function. Name the operators t1 and t2 for the symbol AAPL and TSLA respectively.

2.3. Create BashOperator to move the downloaded file to a data location (t3, t4)

Once the file is downloaded for each stock symbol type, the next step is to move that file into the designated location where your query will target when it runs. This is easy to do with a BashOperator. You should create one task per symbol (t3, t4), for a total of two tasks. The query will operate on both symbols together, so both files should be moved to the same directory.

```
t3 = BashOperator(
    # [fill in options]
    dag=dag)
```

2.4. Create a PythonOperator to run a query on both data files in the specified location (t5)

For this step, run your query on the data you downloaded for both symbols. This step should run only when t3 and t4 are completed.

Step 3. Set job dependencies

After defining all the tasks, you need to set their job dependencies so:

- t1 and t2 must run only after t0
- t3 must run after t1
- t4 must run after t2
- t5 must run after both t3 and t4 are complete

Step 4. Schedule the job in Airflow

Now that you have prepared the job configuration, you can schedule the DAG in Airflow and let it run automatically as scheduled. The command is below:

```
airflow scheduler
```

Please run this Airflow for at least two days. In the next project, we will be using the scheduler log produced during this run.

Instruction for Submission:

- Push the Python code and shell script to GitHub.
- Add a readme file to include steps to run your code and verify the result. Your mentor should be able to run it by following your instructions.
    - Readings about readme file: Example 1, Example 2
- Attach the command line execution log for the successful job run. You can capture it in a text file.