# Python on Airflow pipelines

As we want to run pipelines many times to collect historical data, we need an scheduling mechanism. This is precisly what Airflow will do for us.

# 1. Add Airflow code to the pipeline

We will refactor the code of the previous section to run exactly the same pipeline as an Airflow DAG that can be scheduled for a few days to collect historical information.

### 1.1 Review the changes in the code

The necessary changes for enabling the python pipeline to run as an Airflow DAG are shown in the next cell

 

As you can see the fundamental changes are locatedat the bottom of the file where we write a new header for Airflow

### 1.2. Edit the file pythondag\_airflow.py to include Postgres and Databand security parameters

All changes for Airflow are already done in [pythondag\_airflow.py](../dags/pythondag_airflow.py) but we need to enter the Postgres and Databand credentials for your particular environment. Please follow the same instructions as shown in the [previous chapter](./9_python_dag_dev.ipynb) under the paragraph 1.3. No more changes are necessary.

### 1.3. Transfer pythondag\_airflow.py to Airflow

As we modified the file, we need to transfer it to Airflow to be registered as a DAG. We begin with the usual login to the cluster

# Replace the command with your own one inside the single quotes and run the cell  
# Example OC\_LOGIN\_COMMAND='oc login --token=sha256~3bR5KXgwiUoaQiph2\_kIXCDQnVfm\_HQy3YwU2m-UOrs --server=https://c109-e.us-east.containers.cloud.ibm.com:31656'  
OC\_LOGIN\_COMMAND='oc login --token=sha256~6Xs6va20JZ2CFhS61HN6bpQC2z075XZbhIJt3tZ8L6w --server=https://c109-e.us-east.containers.cloud.ibm.com:31470'  
$OC\_LOGIN\_COMMAND  
oc project airflow

We need to verify that the file pythondag\_airflow.py is located in the DAGs directory.

# you may need to modify the cd command to place yourself in the DAGs directory  
pwd  
cd ../dags  
ls -l

Look for a pythondag\_airflow.py like this:

And then you can run this cell to transfer the file:

# Run this cell to copy the file to the openshift cluster  
oc cp pythondag\_airflow.py airflow-worker-0:dags/

### 1.4. Enable the run on Airflow

Once the file is transfered to Airflow you may need to wait about 5 minutes until the the DAG is visible. Then, you need to activate it:



Now, the DAG is activated and will run every 17 minutes. Leave it running for a few days if you whish to see historical data or go to the next section where we see how it will look like.

## 2. Display performance data with Databand

The new created pipeline can will be shown as Python\_Airflow\_DAG as it is hardcoded in the header section of the python code



The pipeline ran 240 times so far and the historical data can be shown like follows:



Unfortunately, when things run well, all graphics and trends look like very boring. In our case the number of rows written and read is the same during the whole history.

However, there are ways to see more exciting curves. Just proceed as instructed in the following picture:



Indeed, there are variations in the elapsed runtime caused by the concurrency of several jobs while the performance data was collected. You can now switch to the Datasets view and see the cumulated traffic of records in the last days:



Next Section: [DataStage pipelines](./11_datastage_dev.ipynb)  
Previous Section: [Python pipelines](./9_python_dag_dev.ipynb)

[Return to main](../README.md)