**Airflow with docker**

[**https://airflow.apache.org/docs/apache-airflow/stable/tutorial/pipeline.html**](https://airflow.apache.org/docs/apache-airflow/stable/tutorial/pipeline.html)

**Detailed tricks & usage**

[**https://www.youtube.com/watch?v=K9AnJ9\_ZAXE&list=PLwFJcsJ61oujFW8pTo9S8\_b6wujg5NgGW**](https://www.youtube.com/watch?v=K9AnJ9_ZAXE&list=PLwFJcsJ61oujFW8pTo9S8_b6wujg5NgGW)

**to start airflow with docker**

*# Download the docker-compose.yaml file*

curl -LfO 'https://airflow.apache.org/docs/apache-airflow/stable/docker-compose.yaml'

*# Make expected directories and set an expected environment variable*

mkdir -p ./dags ./logs ./plugins

echo -e "AIRFLOW\_UID=**$(**id -u**)**" > .env

*# Initialize the database*

docker-compose up airflow-init

*# Start up all services*

docker-compose up

**turn on just airflow without init db**

docker-compose up

**turn down docker and release volume assigned to docker**

docker-compose down -v

**DAGS**

1. [task1, task2]>>task3

**# means task1 and task2 will run at the same time, once both complete, task 3 will run**

1. **Use bash command to execute python files/packages**
2. **Preferred way to use for python functions: taskflow api**
3. **Dag using python operator with parameter:**
4. **To backfill after setting catchup = False**

In terminal run:

docker ps

**this will give you instance of docker running**

Text

Description automatically generated

docker exec -it c4bc….. bash

**to backfill**

airflow dags backfill -s 2022-01-01 -e 2022-02-22

exit

**refresh UI to see refreshed in the past dates**

**Graphical user interface, text, application, chat or text message

Description automatically generated**

1. **a good DAG example:**

**https://docs.astronomer.io/learn/dags**

**Text

Description automatically generated**

**OLD**

[**https://www.youtube.com/watch?v=K9AnJ9\_ZAXE&list=PLwFJcsJ61oujFW8pTo9S8\_b6wujg5NgGW**](https://www.youtube.com/watch?v=K9AnJ9_ZAXE&list=PLwFJcsJ61oujFW8pTo9S8_b6wujg5NgGW)

**Set up airflow using pip and env**

**To run existing env**

python3 -m venv /path/to/new/virtual/environment

**to create env**

python3 -m venv [name of your env eg: env\_airflow]

source env\_ariflow/bin/activate

pip install apache-airflow

**if attribute column error pops up, use pip install -U marshmallow-sqlalchemy**

**Initialize the airflow home directory**

export AIRFLOW\_HOME=.

Or export AIRFLOW\_HOME=~/airflow

**create user**

airflow users create --role Admin --username admin --email admin --firstname admin --lastname admin --password admin

By default, if you are using the docker-compose example provided by airflow, the user is:

user: airflow

password: airflow

If you want to create another account with docker use this (you have to be in the same folder of the docker-compose.yaml file):

docker-compose run airflow-worker airflow users create --role Admin --username admin --email admin --firstname admin --lastname admin --password admin

**Airflow DB Init**

Airflow db init

**to start airflow server run:**

airflow webserver -p 8080

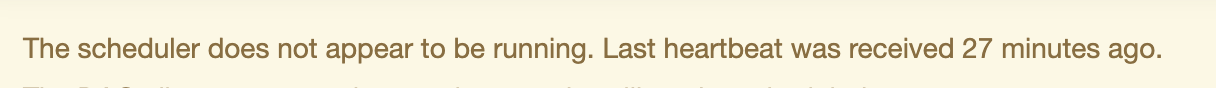
**to run airflow in new terminal**

source ~/Desktop/env\_airflow/bin/activate

**Error: The webserver is already running under PID 5338.**

sudo lsof -i tcp:8080

kill -9 5338(PID)­­­­­

if message

open a new terminal

**repeat: Initialize the airflow home directory**

**then run**

airflow scheduler

**if fail to run scheduler**

1. **Install postgres then switch airflow to use postgres**

In airflow.cfg file look for sql\_alchemy\_conn and update it to point to your PostgreSQL serv:

sql\_alchemy\_conn = postgresql+psycopg2://user:pass@hostadress:port/database

executor = LocalExecutor

For instance:

sql\_alchemy\_conn = postgresql+psycopg2://airflow:airflow@localhost:5432/airflow

As indicated in the above line you need both user and database called airflow, therefore you need to create that. To do so, open your psql command line and type the following commands to create a user and database called airflow and give all privileges over database airflow to user airflow:

CREATE USER airflow;

CREATE DATABASE airflow;

GRANT ALL PRIVILEGES ON DATABASE airflow TO airflow;

Now you are ready to init the airflow application using postgres:

airflow initdb

If everything was right, access the psql command line again, enter in airflow database with \c airflow command and type \dt command to list all tables of that database. You should see a list of airflow tables, currently it is 23.

**Full Airflow with Postgres installation**