**Configure a DAG with Celery Executor,PostgreSQL and RabbitMQ**

Setup an Apache airflow to work with Distributed Mode.

Open three sessions.

1)Schedular

2)Webserver

3)Worker

Connect through the VM then airflow.

Now activate the python Virtual Environment for the each session mention above.

create another session to configure the Configuration file of Apache Airflow.

Make some modifications in the configuration file by open the airflow.cfg config file below.

vim airflow/airflow.cfg

In this file, change the

executor=LocalExecutor

change to

executor=CeleryExecutor

Now specify the broker Url

/broker\_url=Pyamqp://admin:rabbitmq@localhost/

This corresponds to rabbitmq.

Change the result back end parameter to use the Postgres SQL database.

There is backend corresponds to the database that would be used to track the task executions from the worker nodes.

result\_backend=db+postgresql://airflow@localhost:5432/airflow.mdb

The last parameter need to modify is the worker load server port

When we touch on airflow worker airflow starts a tiny web server step process to sell the worker to look files to the main web server.

The tiny web server we did the parameter log server port set to 8794

Worker\_log\_server\_port=8794

Save the airflow.cfg file.

Now start the web server at the scheduler and the work load.

But before starting these three components set the db.

airflow resetdb

now run the airflow web server

airflow webserver

now run the scheduler

airflow scheduler

lastly start airflow worker

airflow worker

Now worker load is running.

When we type airflow worker we actually run a Celery worker Node.

When Celery worker is running it creates the one parent process to manage the running tasks.

This process handles features like sending/receiving queue messages, tracking status registering and killing tasks etc.

The number of child worker processes can be determined by typing

airflow worker –c 2 (for 2 child worker processes)

or by changing the worker\_concurrency parameter in airflow.cfg

We can also an overview of whats going on through rabbitmq by accessing to its user interface

Here it will show how worker is well connected to broker.

Now restart the worker node by

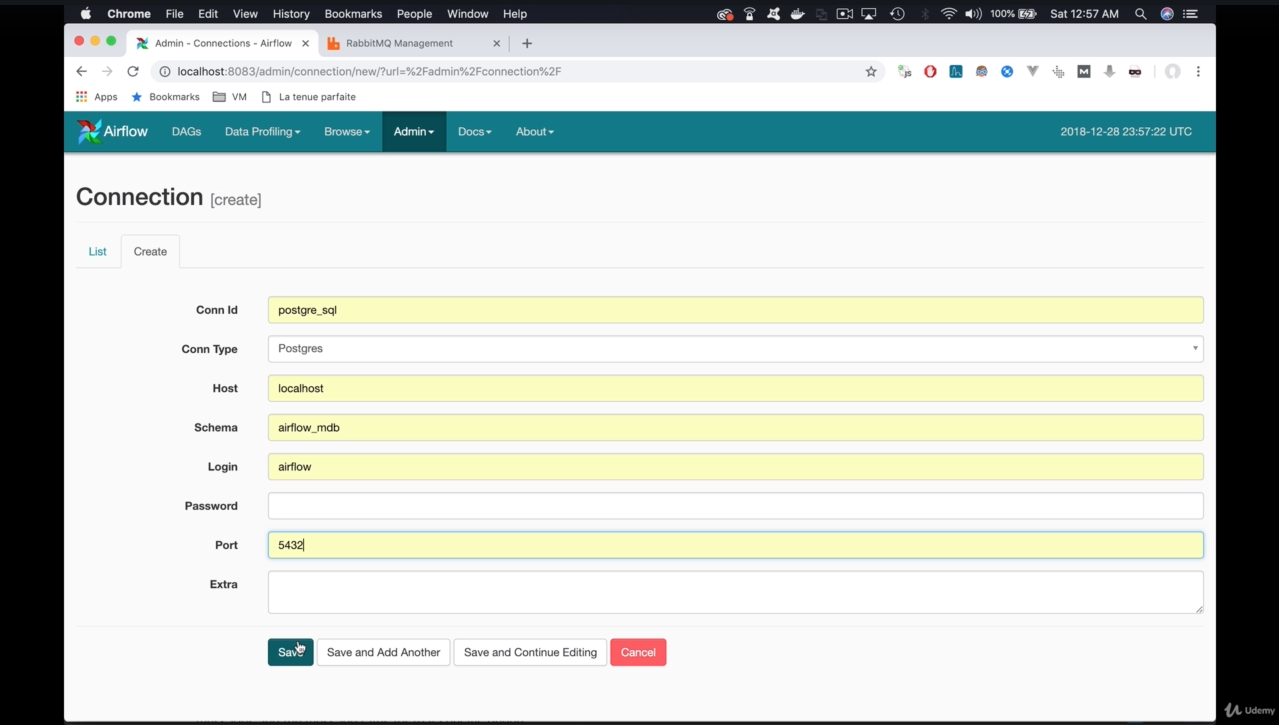
airflow worker

and go to the user interface of airflow towards the naming dag in distributed mode.

Because we have reset the database and we are going to run the dynamic dag which uses postgre SQL connection

Now need to create the connection from the connection view.

Admin---connections—create



Now go to DAGS in airflow distributed mode

And turn on toggle to schedule the dynamic dag and click the refresh

Now dag run is running and wait for the dag run is to finish.

Now go the worker node as we can see from the logs your tasks have been well distributed on this node before getting executed.

Don’t forget that need to copy the dag folders on each of our worker load.