Configure a DAG with Local Executor and PostgreSQL

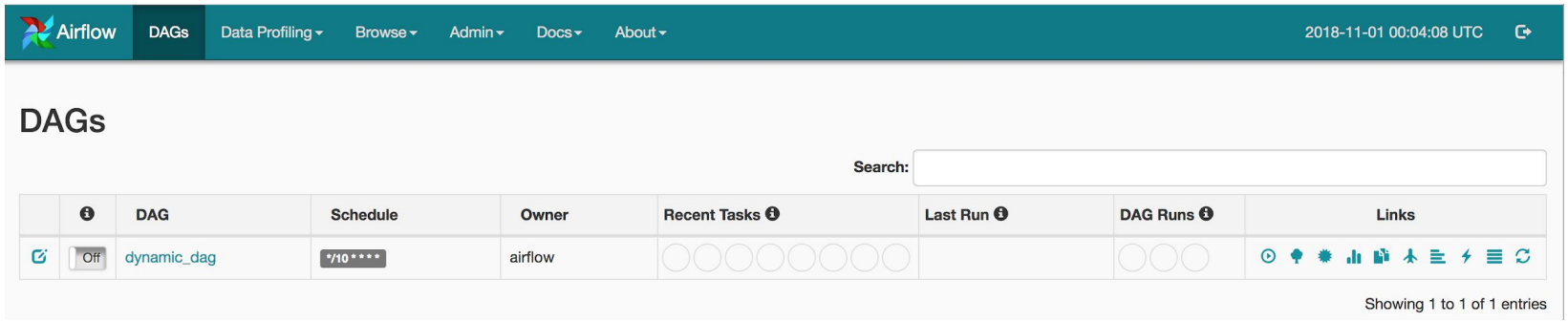
Time to practice!

Instructions

- If your airflow webserver and airflow scheduler are running, stop them by typing `ctrl-C` in their respective terminal.
- `vim ~/airflow/airflow.cfg`
- In the configuration change the following:
  - `executor = LocalExecutor`
  - `sql_alchemy_conn = postgresql+psycopg2://airflow@localhost:5432/airflow_mdb`
- Restart airflow
  - `airflow initdb`
  - `airflow webserver`
  - `airflow scheduler`

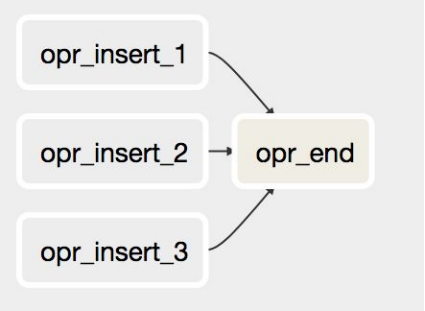
Instructions

- `cp ~/airflow_files/dynamic_dag.py ~/airflow/dags`
- You should see the following dag into Airflow UI:



Instructions

- Click on ‘dynamic_dag’ => ‘Graph View’ and you should have the following graph:



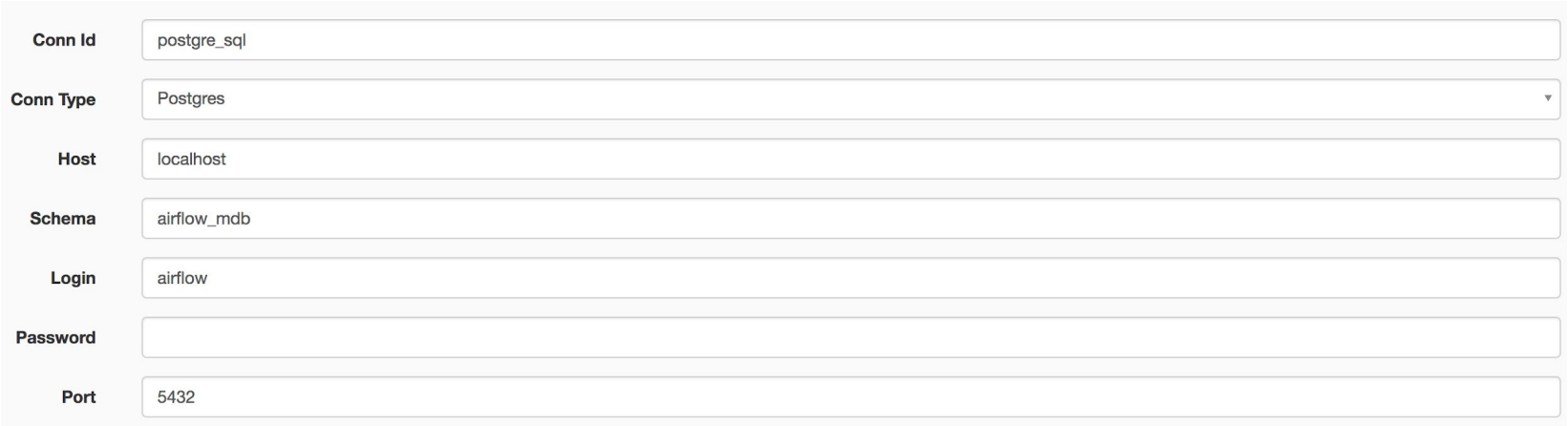
- As you can see we have created 3 operations dynamically linked to opr_end thanks to Python.

Instructions

- Let’s configure a connection from Airflow UI in order to use PostgreSQL with the DAG.
- Go to ‘Admin’ => ‘Connections’
- You should have a list of connections which you will be able to use in your DAGs.

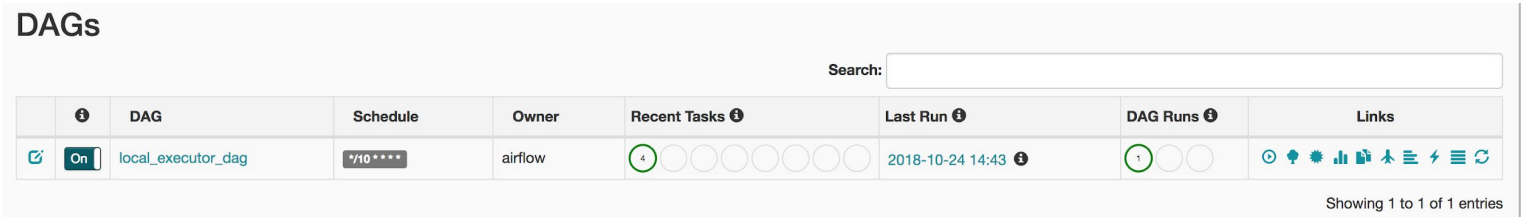
Instructions

- Click on ‘create’ at the top of the list and type the following information into the text fields:



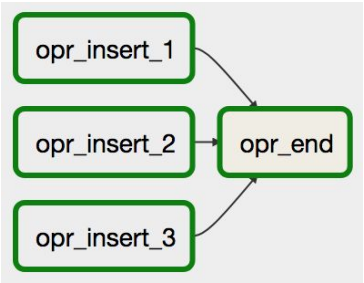
Instructions

- Click on ‘Save’ at the bottom of the form.
- Now go back to the DAG views.
- Turn ON the toggle of the DAG: dynamic_dag and click on ‘Trigger Dag’.
- Refresh a couple of times your navigator to see the following screen:



Instructions

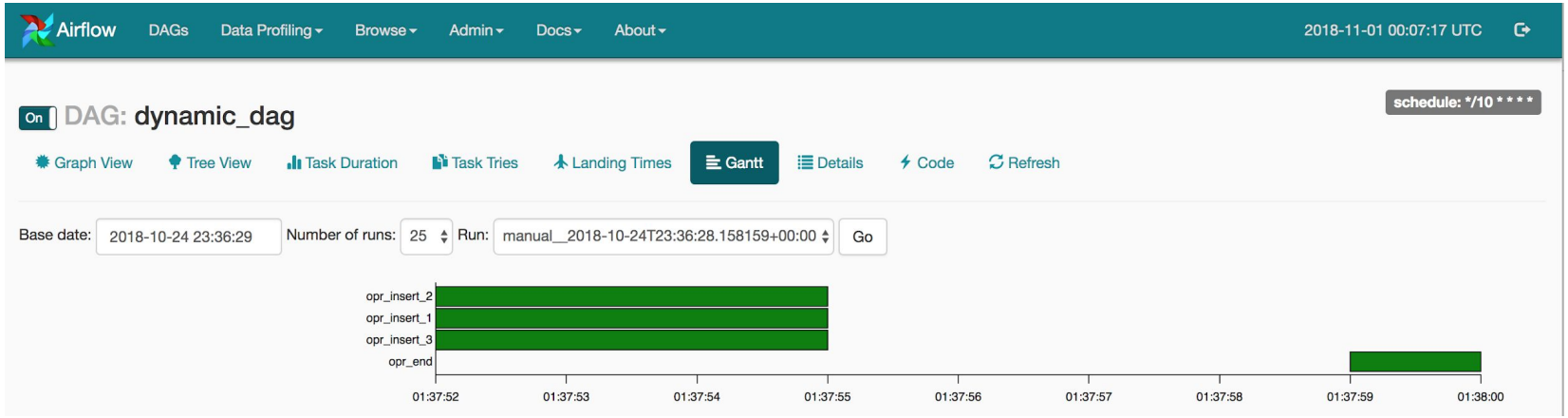
- Click on ‘dynamic_dag’ => ‘Graph View’ and you should have the following graph:



- Ok nice, the green borders are telling you that each task run successfully, but let’s see a view way more interesting...

Instructions

- Click on ‘Gantt’ and you should have the following screen:



Instructions

- The Gantt view allows you to see how long did your tasks took to execute and when do they started and finished.
- As you can see, the first 3 tasks were executed **in parallel** since Local Executor allows you to do so with PostgreSQL.
- Using LocalExecutor with PostgreSQL can dramatically improve your performances and allow you to execute multiple DAGs at the same time.

Instructions

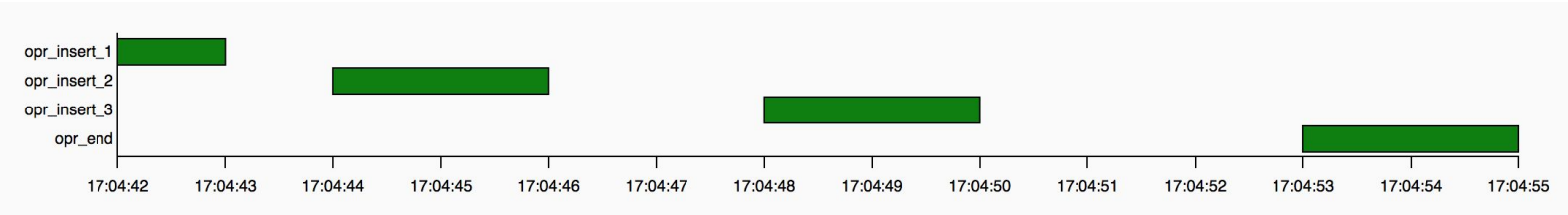
- Last thing but not least, click on ‘Data Profiling’ => ‘Ad Hoc Query’
- Select ‘postgre_sql’ from the list box which is basically the connection you have set earlier.
- From this view you can interact with whatever databases you are connected with directly with Airflow UI.
- Type:
  - `SELECT * FROM last_executor.task;`
- You should see the result of the DAG with task_id and timestamp.

Instructions

- Last thing to be sure you understand well what is going on. If you open the airflow.cfg configuration file and change the line:
  - `dag_concurrency=16`
- With
  - `dag_concurrency=1`
- And restart the web server and the scheduler.

Instructions

- If you run again the DAG as we did before in the previous slides, you should see the following Gantt view:



- Because we have limited the number of concurrent task instances to 1, each task is now executed sequentially like the Sequential Executor but we are still using Local Executor.
- *Also, with parallelism=1 and dag_concurrency=16, you will end up with the same result as parallelism=1 means you allow Apache Airflow to run 1 process (worker).*