Make Your First Conditional Task Using Branching
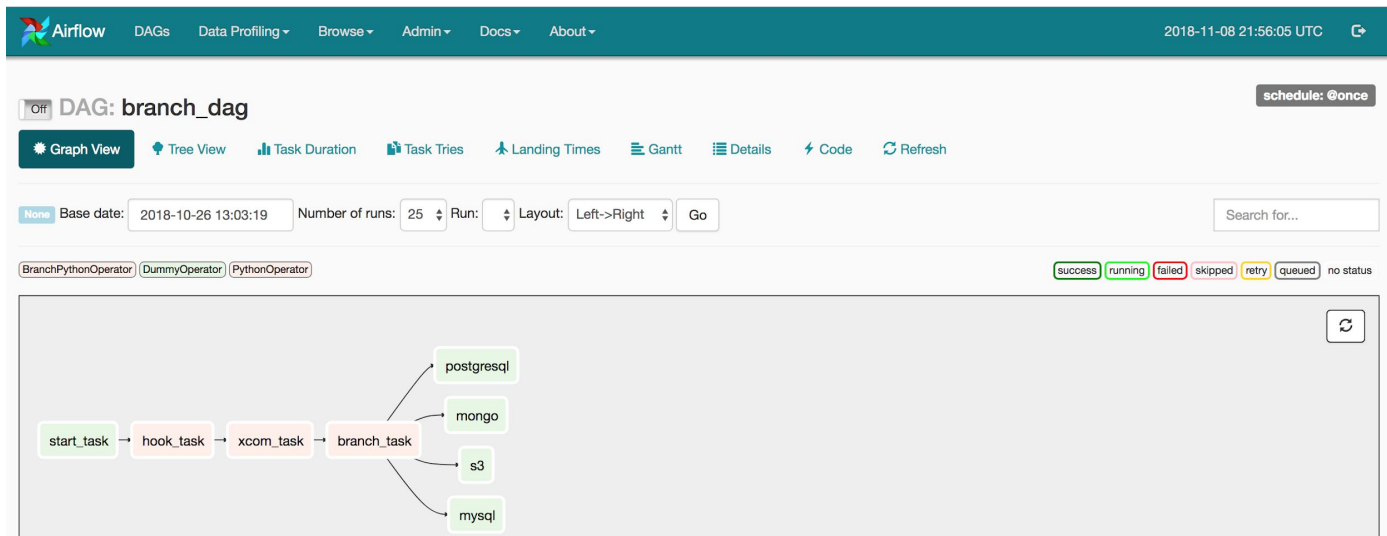
Branching in action!

Instructions

- `cd ~`
- `cp ~/airflow_files/branch_dag.py ~/airflow/dags/`
- Restart airflow webserver and scheduler if you don't want to wait for the DAG to be detected.
- `vim ~/airflow/dags/branch_dag.py` and take a look to the code.

Instructions

- Now, if you click on "branch_dag" and go to "Graph View" you should see the following DAG:

Airflow   DAGs   Data Profiling▾   Browse▾   Admin▾   Docs▾   About▾   2018-11-08 21:56:05 UTC

Off DAG: branch_dag   schedule: @once

☀ Graph View   🌱 Tree View   📊 Task Duration   📑 Task Tries   🏃 Landing Times   ☰ Gantt   ☰ Details   ⚡ Code   🔄 Refresh

None Base date: 2018-10-26 13:03:19   Number of runs: 25   Run: Layout: Left->Right   Go   Search for...

BranchPythonOperator   DummyOperator   PythonOperator   success   running   failed   skipped   retry   queued   no status

start_task → hook_task → xcom_task → branch_task → postgresql / mongo / s3 / mysql

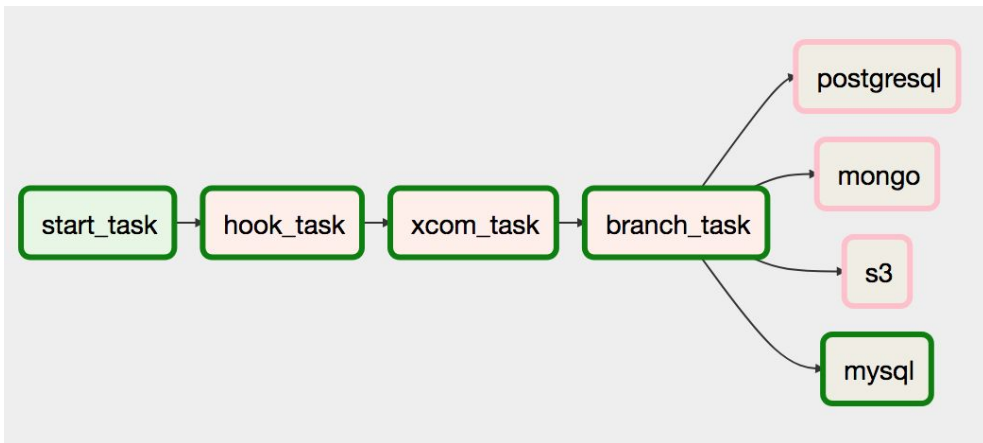# Instructions

- As you can see from the previous slide, we have a branch_task leading to 4 different tasks
  - postgresql, mysql, mongo, s3.
- Only one task is going to be executed according to the result returned by the task xcom_task to the branch_task.
- Turn On the toggle and trigger the DAG from the "DAGs View"

Instructions

- Once the DAG has finished executing you should have the following DAG from the "Graph View":

Instructions

- As you can see from the previous slide, only one task has been executed, the one having the same task_id as the value returned by xcom_task which we fetched using the function `check_for_activated_source` called by the BranchPythonOperator.
- All the other tasks have been set to the "skipped" status.
- If you click on the task "branch_task" and "View Log" you will see the lines below showing that PythonBranchOperator received the value mysql and followed the branch with the direct downstream task "mysql":

```
[2018-10-26 13:13:31,807] {base_task_runner.py:107} INFO - Job 46: Subtask branch_task [2018-10-26 13:13:31,806]
[2018-10-26 13:13:31,840] {python_operator.py:96} INFO - Done. Returned value was: mysql
[2018-10-26 13:13:31,841] {python_operator.py:124} INFO - Following branch mysql
[2018-10-26 13:13:31,841] {python_operator.py:125} INFO - Marking other directly downstream tasks as skipped
[2018-10-26 13:13:31,849] {python_operator.py:134} INFO - Done.
```