Apache Airflow

Tushar B. Kute,
http://tusharkute.com

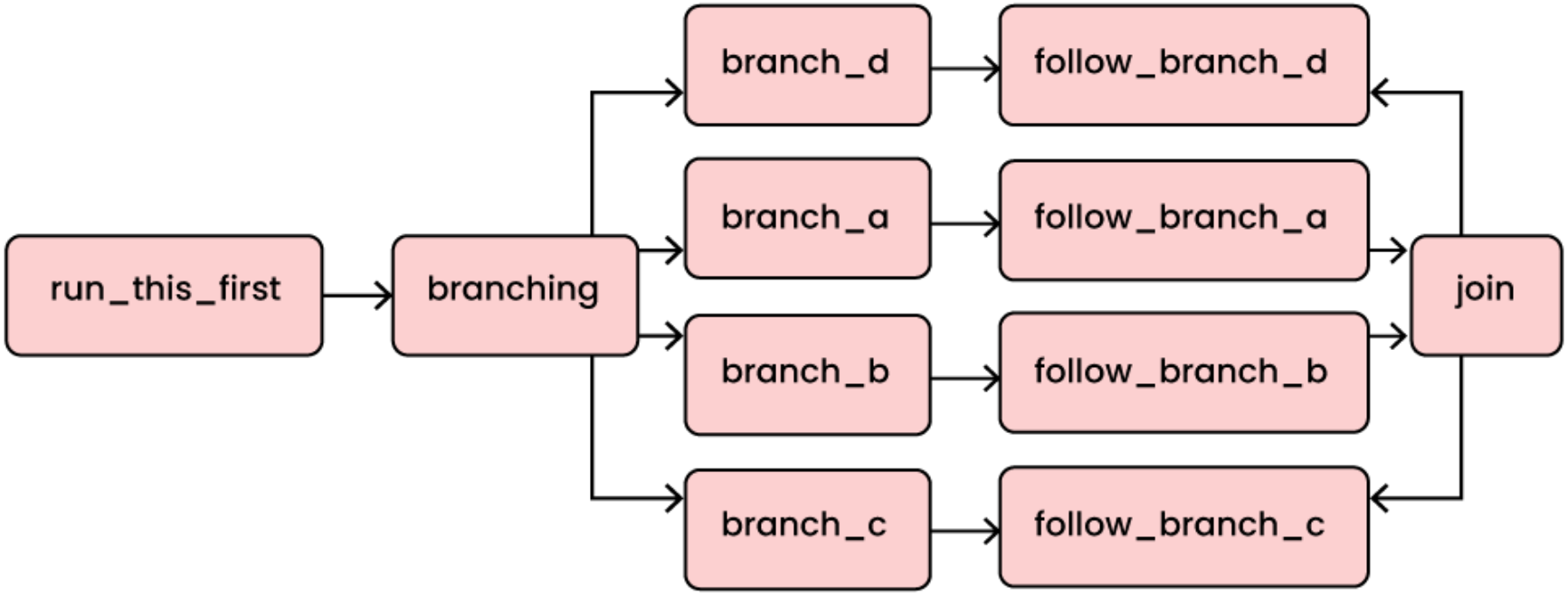
Introduction

- Apache Airflow is a must-have tool for Data Engineers.
- It makes it easier to create and monitor all your workflows.
- When you have multiple workflows, there are higher chances that you might be using the same databases and same file paths for multiple workflows.
- Using variables is one of the most efficient ways to define such shared information among different workflows.

Apache Airflow

- Apache Airflow is a workflow engine that will easily schedule and run your complex data pipelines.
- It will make sure that each task of your data pipeline will get executed in the correct order and each task gets the required resources.
- It will provide you an amazing user interface to monitor and fix any issues that may arise.

Apache Airflow



Apache Airflow

- Apache Airflow is one significant scheduler for programmatically scheduling, authoring, and monitoring the workflows in an organization.
- It is mainly designed to orchestrate and handle complex pipelines of data.
- Initially, it was designed to handle issues that correspond with long-term tasks and robust scripts.
- However, it has now grown to be a powerful data pipeline platform.

Apache Airflow

- Airflow can be described as a platform that helps define, monitoring and execute workflows.
- In simple words, workflow is a sequence of steps that you take to accomplish a certain objective.
- Also, Airflow is a code-first platform as well that is designed with the notion that data pipelines can be best expressed as codes.

Apache Airflow

- Apache Airflow was built to be expandable with plugins that enable interaction with a variety of common external systems along with other platforms to make one that is solely for you.
- With this platform, you can effortlessly run thousands of varying tasks each day; thereby, streamlining the entire workflow management.

Why to use Apache Airflow ?

- You can easily get a variety of reasons to use apache airflow as mentioned below:
  - This one is an open-source platform; hence, you can download Airflow and begin using it immediately, either individually or along with your team.
  - It is extremely scalable and can be deployed on either one server or can be scaled up to massive deployments with a variety of nodes.
  - Airflow apache runs extremely well with cloud environments; hence, you can easily gain a variety of options.

Why to use Apache Airflow ?

- It was developed to work with the standard architectures that are integrated into most software development environments. Also, you can have an array of customization options as well.
- Its active and large community lets you scale information and allows you to connect with peers easily.
- Airflow enables diverse methods of monitoring, making it easier for you to keep track of your tasks.
- Its dependability on code offers you the liberty to write whatever code you would want to execute at each step of the data pipeline.

Apache Airflow Fundamentals

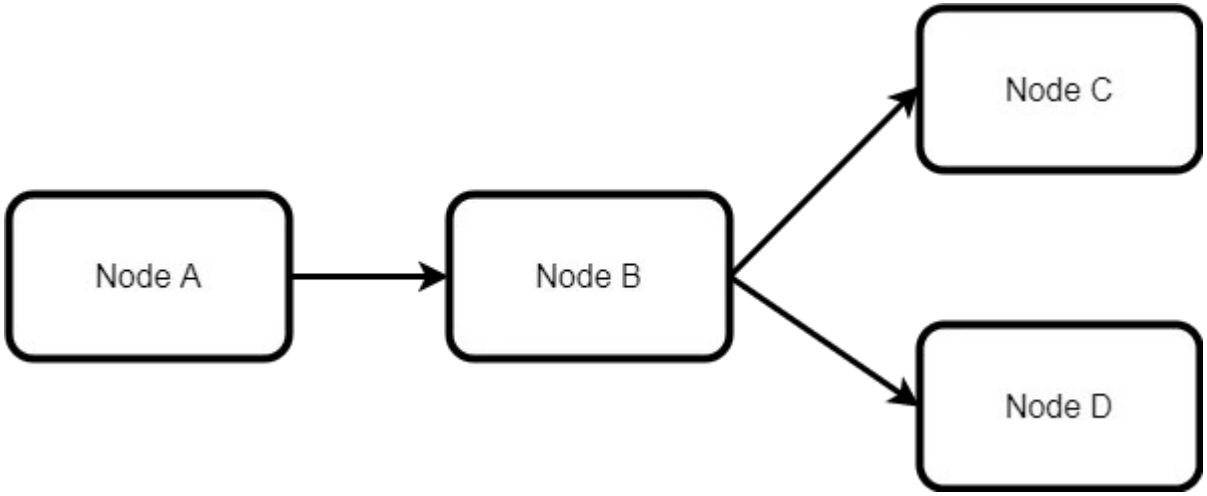
- #1. DAG in Airflow
  - Herein, workflows are generally defined with the help of Directed Acyclic Graphs (DAG).
  - These are created of those tasks that have to be executed along with their associated dependencies.
  - Every DAG is illustrating a group of tasks that you want to run.
  - And, they also showcase the relationship between tasks available in the user interface of the Apache Airflow.

Apache Airflow Fundamentals

- Let’s break down DAG further to understand more about it:
  - Directed: If you have several tasks that further have dependencies, each one of them would require at least one specific upstream task or downstream task.
  - Acyclic: Here, tasks are not allowed to create data with self-references. This neglects the possibility of creating an infinite loop.
  - Graph: Tasks are generally in a logical structure with precisely defined relationships and processes in association with other tasks.

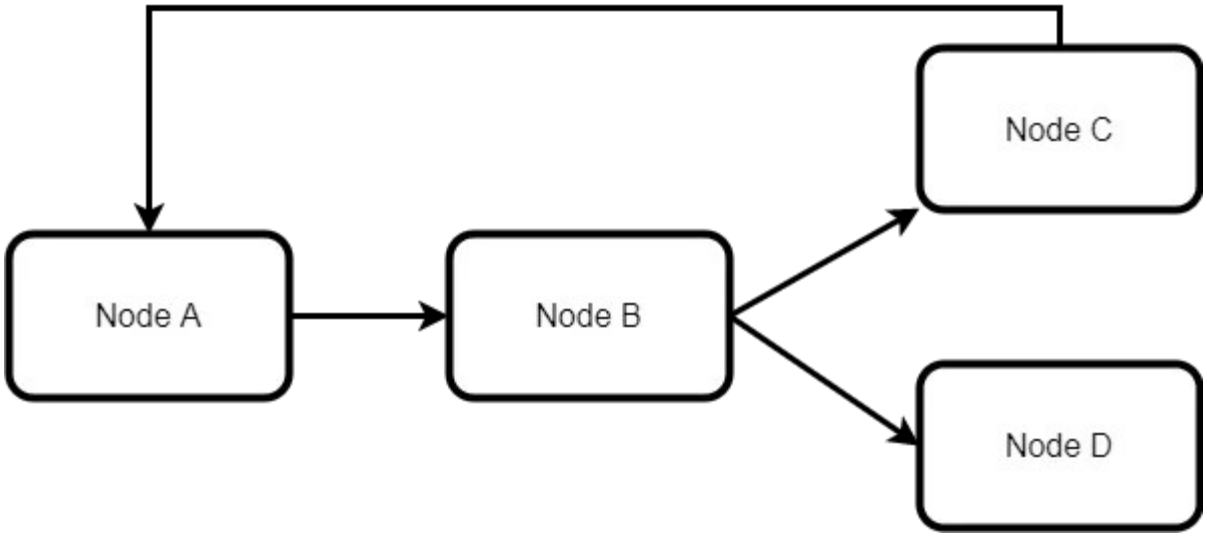
Apache Airflow Fundamentals

- In simple terms, a DAG is a graph with nodes connected via directed edges. Also, there should be no cycles within such a graph.
- For example, the below diagram represents a DAG.



Apache Airflow Fundamentals

- However, the below example is not a DAG:



Apache Airflow Fundamentals

- Why so?
  - It is because there is a cycle in the second diagram from Node C to Node A. Due to this cycle, this DAG will not execute. However, the first diagram is a valid DAG.
  - A valid DAG can execute in an Airflow installation. Whenever, a DAG is triggered, a DAGRun is created.
  - We can think of a DAGrun as an instance of the DAG with an execution timestamp.

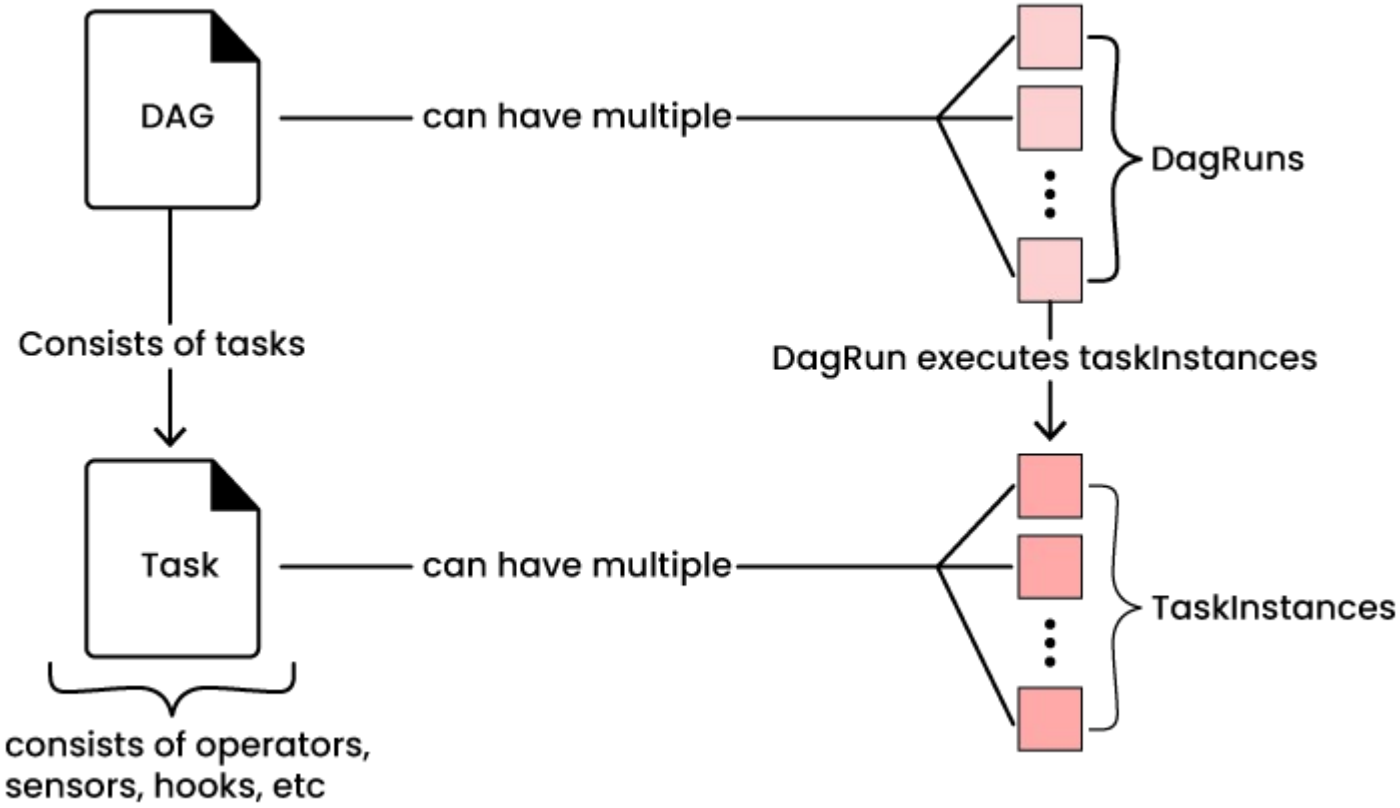
Apache Airflow Fundamentals

- The next aspect to understand is the meaning of a Node in a DAG. A Node is nothing but an operator.
- To elaborate, an operator is a class that contains the logic of what we want to achieve in the DAG.
- For example, if we want to execute a Python script, we will have a Python operator. If we wish to execute a Bash command, we have Bash operator.
- There are several in-built operators available to us as part of Airflow.

Apache Airflow Fundamentals

- #2. DAG Run
  - Basically, when a DAG gets executed, it is known as a DAG run. Let's assume that you have a DAG scheduled and it should run every hour.
  - This way, every instantiation of the DAG will establish a DAG run.
  - There could be several DAG runs connected to one DAG running simultaneously.

Apache Airflow Fundamentals



Apache Airflow Fundamentals

- #3. Task
  - Tasks vary in terms of complexity and they are operators' instantiations.
  - You can take them up as work units that are showcased by nodes in the DAG.
  - They illustrate the work that is completed at every step of the workflow with real work that will be portrayed by being defined by the operators.

Apache Airflow Fundamentals

- #4. Airflow Operators
  - In Apache Airflow, operators are meant to define the work. An operator is much like a class or a template that helps execute a specific task.
  - All of the operators are originated from BaseOperator. You can find operators for a variety of basic tasks, like:
    - PythonOperator
    - BashOperator
    - MySqlOperator
    - EmailOperator

Apache Airflow Fundamentals

- These operators are generally used to specify actions that must be executed in Python, Bash, MySQL, and Email.
- In Apache Airflow, you can find three primary types of operators:
  - Operators that can run until specific conditions are fulfilled
  - Operators that execute an action or request a different system to execute an action
  - Operators that can move data from one system to the other

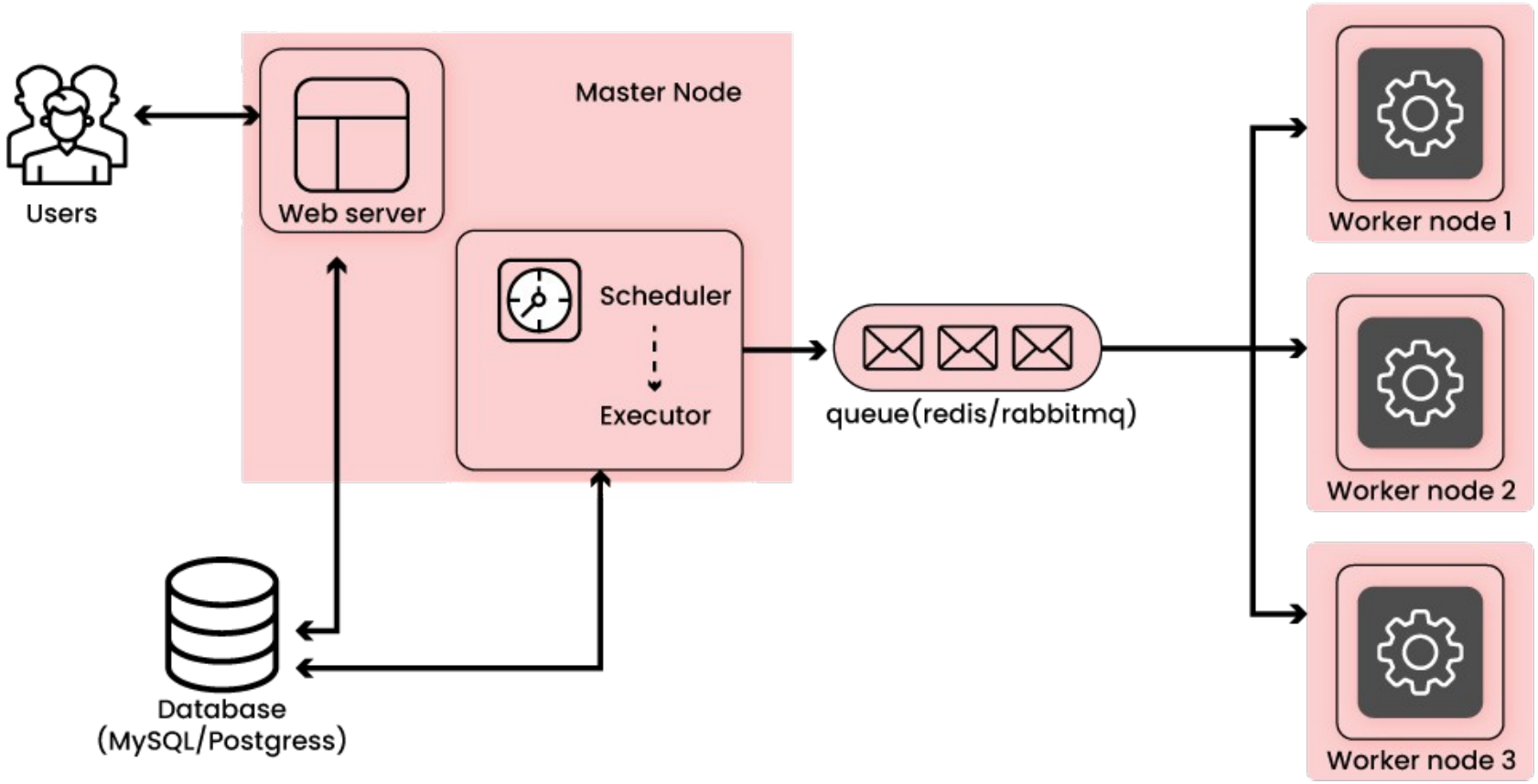
Apache Airflow Fundamentals

- #5. Hooks
  - Hooks enable Airflow to interface with third-party systems.
  - With them, you can effortlessly connect with the outside APIs and databases, such as Hive, MySQL, GCS, and many more.
  - Basically, hooks are much like building blocks for operators.
  - There will be no secured information in them. Rather, it is stored in the encrypted metadata database of Airflow.

Apache Airflow Fundamentals

- #6. Relationships
  - Between tasks, airflow exceeds at defining complicated relationships.
  - Let’s say that you wish to designate a task and that T1 should get executed before T2. Thus, there will be varying statements that you can use to define this precise relationship, like:
    - T2 << T1
    - T1 >> T2
    - T1.set_downstream (T2)
    - T2.set_upstream (T1)

How it works?


Users
Master Node
Web server
Scheduler
Executor
queue(redis/rabbitmq)
Worker node 1
Worker node 2
Worker node 3
Database
(MySQL/Postgress)

How it works?

- To understand how does Apache Airflow works, you must understand there are four major components that create this scalable and robust workflow scheduling platform:
- Scheduler: The scheduler monitors all of the DAGs and their linked tasks. For a task, when dependencies are met, the scheduler initiates the task. It checks active tasks for initiation periodically.
- Web Server: This is the user interface of Airflow. It displays the status of responsibilities and lets the user interact with databases and read log files from remote file stores, such as Google Cloud Storage, S3, Microsoft Azure blobs, and more.

How it works?

- Database: In the database, the state of the DAGs and their linked tasks are saved to make sure the schedule remembers the information of metadata. Airflow uses Object Relational Mapping (ORM) and SQLAlchemy to connect to the metadata database. The scheduler evaluates all of the DAGs and stores important information, such as task instances, statistics from every run, and schedule intervals.
- Executor: The executor zeroes down upon how the work will get done. There is a variety of executors that you can use for diverse use cases, such as SequentialExecutor, LocalExecutor, CeleryExecutor, and KubernetsExecutor.

How it works?

- Airflow evaluates all of the DAGs in the background at a specific period.
- This period is set with the help of processor_poll_interval config and equals one second.
- Once a DAG file is evaluated, DAG runs are made as per the parameters of scheduling.
- Then, task instances are instantiated for such tasks that must be performed and their status is changed to SCHEDULED in the metadata database.

How it works?

- The next step is when the schedule questions the database, retrieves tasks when they are in the scheduled state, and distributes them to all of the executors.
- And then, the task's state changes to QUEUED. The queued tasks are drawn from the queue by executors. When this happens, the status of the task is changed to RUNNING.
- Once a task is finished, it will be marked as either finished or failed. And then, the scheduler will update the final status in the database.

Features of Airflow

- Easy to Use: If you have a bit of python knowledge, you are good to go and deploy on Airflow.
- Open Source: It is free and open-source with a lot of active users.
- Robust Integrations: It will give you ready to use operators so that you can work with Google Cloud Platform, Amazon AWS, Microsoft Azure, etc.
- Use Standard Python to code: You can use python to create simple to complex workflows with complete flexibility.
- Amazing User Interface: You can monitor and manage your workflows. It will allow you to check the status of completed and ongoing tasks.

Thank you

This presentation is created using LibreOffice Impress 5.1.6.2, can be used freely as per GNU General Public License

@mitu_skillologies

/mITuSkillologies

@mitu_group

/company/mitu-skillologies

MITUSkillologies

Web Resources
https://mitu.co.in
http://tusharkute.com

contact@mitu.co.in

tushar@tusharkute.com