Course: Learning Apache Airflow. Date: 09/05/23

Airflow: An open source platform for developing, scheduling, and monitoring batch-oriented workflow.

Key objective: It's a workflow manager.

Batch-oriented workflow: A batch-oriented work-flow is a type of processing workflow where task are executed in batches, with each bach typically processing a set of data records or job in predefiend and systematic order.

Key takeaways:

1. Data is divided into discreate groups or batches, and each batch processed as a unit. This is in contrast to real-time or stream processing, where data is processed as it arrives.

2. Batch job are typically sheduled to run at specific interval shuch as hourly, daily, or weekly.

3. Its often used in procesing large volume of data.

4. Its often involve data transformation task.

5. Batch job often require significant computing re-source, so resource allocation and management are essential consideration.

6. Batch workflow may have dependancies between jobs, where the output of one job depends on the input of other, so need to process these dependenies.

Airflow Architecture: Airflow has a modular architecture.

Components :-

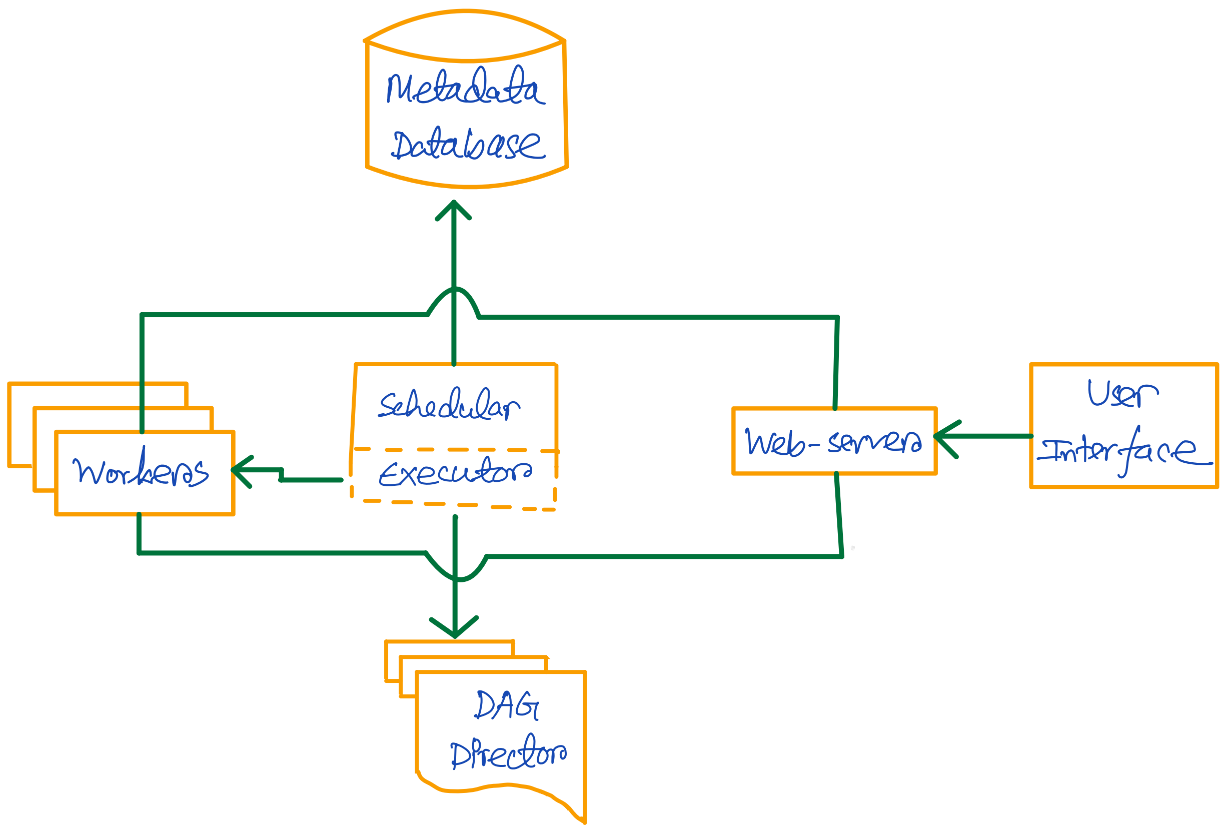
1. scheduler : Schedule the work-flow.

2. Executor: Runs the workflows.

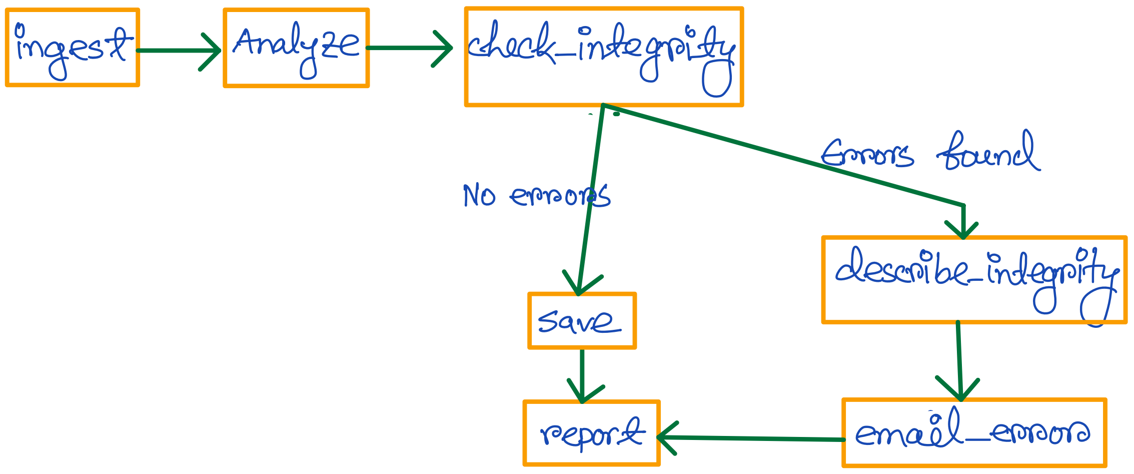
3. Meta-data database: holds additional information.

4. Webserver : Interective UI for managing the work-flow.

Airflow Architecture :



→ when using Apache Airflow, we can define all of the workflows as Directed Acyclic graph (DAG). The node of the graphs are the task that need to executed. The edges connecting different nodes are the dependencies in the workflow.



Airflow Use cases :

1. ETL Pipelines.
2. Machine learning workflows.
3. Data transfer and processing pipeline.
4. Automating business intelligence and reporting.
5. DevOps & infrastructure automation.

Operators : Predefined tasks that can be used to build the nodes in workflow DAG. Examples: bash, python, SQLite.

XComs : "Cross-communication" — a system to pass data between tasks where tasks can push and pull small bits of metadata.

Airflow commands:

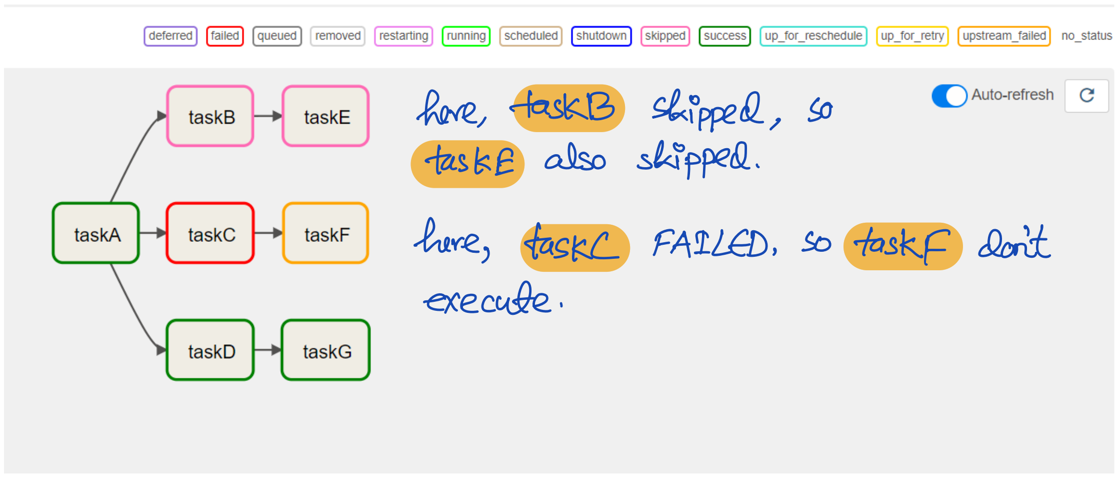
- To get the list of command: airflow -h.
- To get the information of current instalation:
  airflow info
- To get the users/admin info: airflow users -h
- To get the lists of all users: airflow users list
- To create an airflow users:

```
airflow users create \
-e cloud.user@adrahat.com \
-f CloudUser \
-l adrahat \
-p password \
-r Admin \
-u cloud.user
```

* In Airflow the exit has several status code:

exit 130 → failed.

exit 99 → skipped.



here, taskB skipped, so taskE also skipped.

here, taskC FAILED, so taskF don't execute.

* Python Operator used to constructs task using python operation.

* Catchup: Scheduling and executing all DAG runs that would have been run if the DAG had been created earlier.

* Catchup using corn-expression:

example: schedule_interval = '0 0 * * *'

● 5 different components:

1. M – represent the minutes. Values: 0, 30
2. H – represent the hours. Values: 0–23
3. D – represent the day of weeks. Values: 1–31
4. M – represent the month of the schedule. Values: 1-12
5. Y – represent the year of the schedule. Values: Any year.

The * sign repersent each day/month/year.