

LEARNING PROGRESS REVIEW

WEEK 9

BY CHARLIE'S ANGELS



Apache
Airflow



Apache Airflow merupakan tools untuk manajemen workflow untuk data. airflow sering juga dikenal dengan ETL Framework

Apache Airflow diciptakan untuk solusi Batch Processing bukan Stream Processing atau Data Streaming, krna itu jika hendak mencari tool untuk Stream Processing maka pilihlah Apache Spark, Apache Storm dan lain lain

Keunggulan Apache Airflow

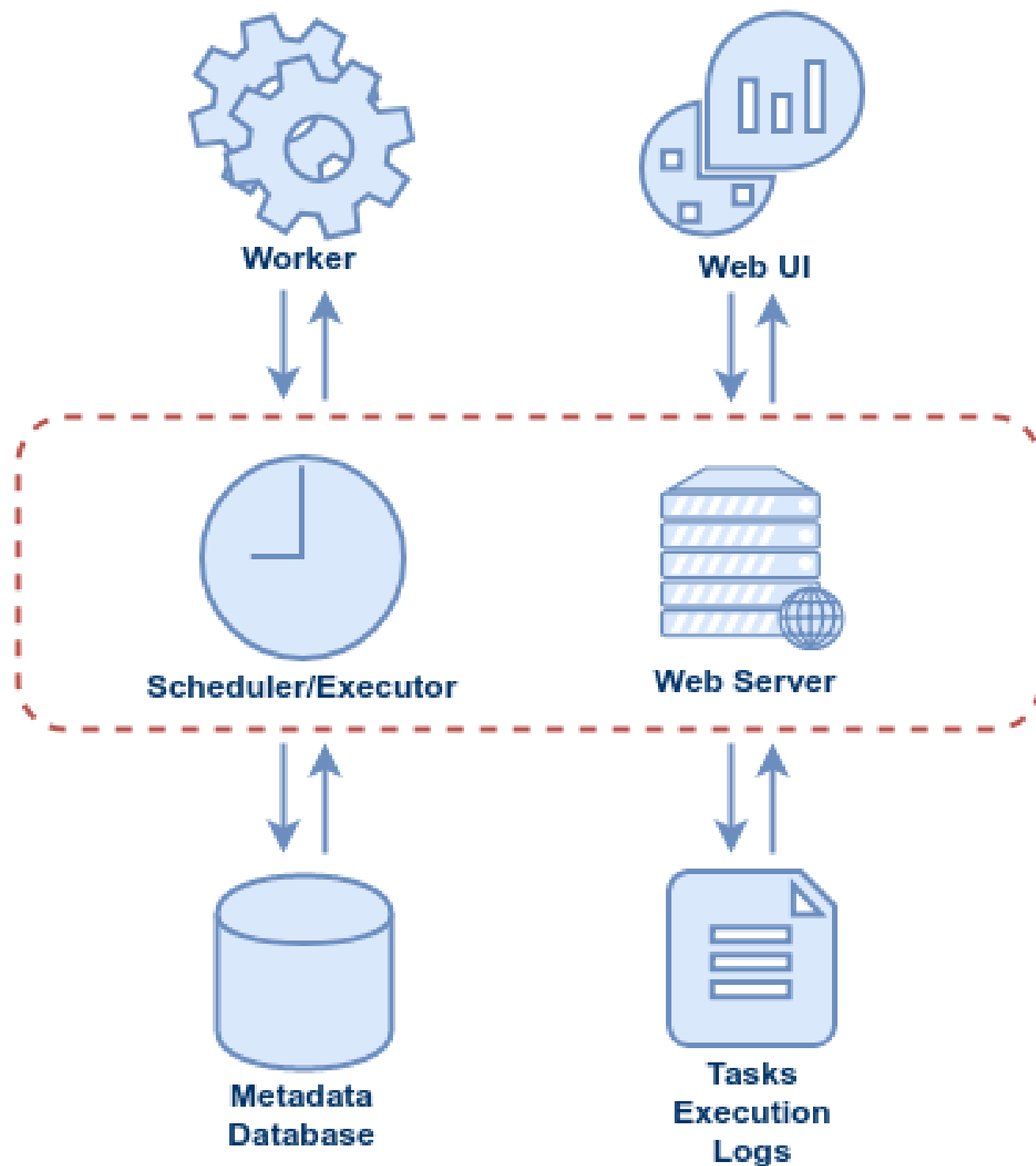


Apache Airflow memiliki banyak fitur, dan didukung dengan integrasi tool eksternal yang banyak seperti:

- Hive,
- Pig,
- Google BigQuery,
- Amazon Redshift,
- Amazon S3, dst

Apache Airflow memiliki keunggulan untuk urusan scaling. Wajar saja jika Apache Airflow menjadi pilihan yang tepat untuk membangun data pipeline saat ini.

Arsitektur Apache Airflow

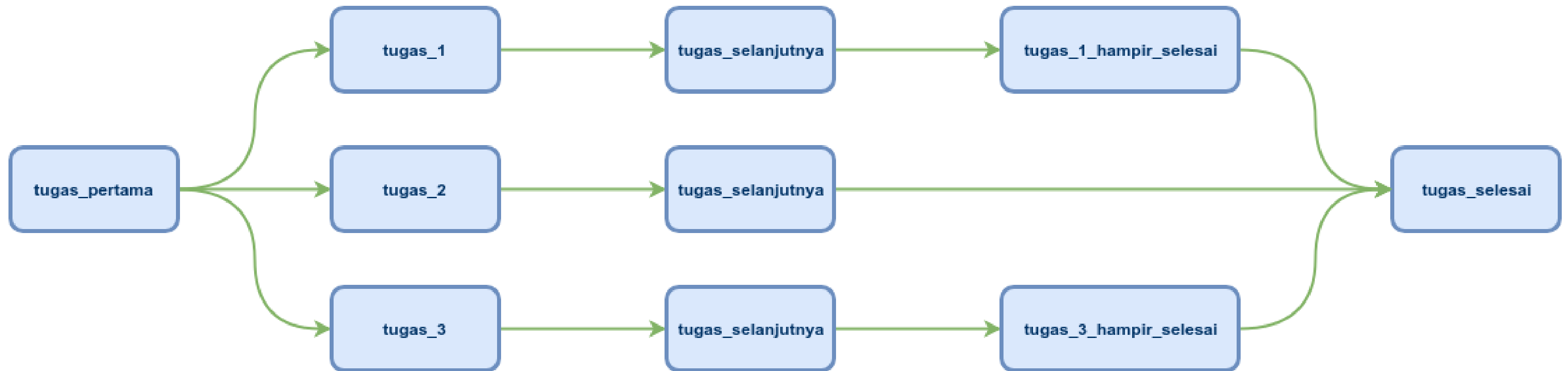


imamdigmi.github.io

Komponen utama dalam membangun workflow



Directed Acyclic Graphs



imamdigmi.github.io

Relasi
Tasks

DAG



DAG adalah kepanjangan dari Directed Acyclic Graphs yang kita gunakan untuk membuat suatu workflow atau kita juga dapat memahami DAG sebagai sekumpulan dari Tasks. DAG inilah yang mencerminkan tentang alur dari workflow beserta relasi antar proses dan ketergantungan antar prosesnya



1. DAG bersifat “acyclic” yang artinya tiap Task tidak akan berputar atau tidak kembali ke Task sebelumnya. Jika dianalogikan, ini seperti seorang ayah dari seorang anak, tentunya sang ayah kandung tidak akan menjadi seorang anak dari anak kandungnya sendiri
2. DAG tidak peduli apa yang sedang kita lakukan, karena DAG hanya bertugas dalam memastikan tugas-tugas didalamnya, dieksekusi pada waktu yang tepat, dengan urutan yang tepat, dan dengan penanganan yang benar atas masalah yang tidak terduga
3. Dan juga, di dalam membuat DAG terdapat dag_id yang digunakan Airflow untuk mengidentifikasi satu DAG saja, ini bersifat unik yang artinya kita tidak boleh menggunakan nama dag_id lebih dari satu kali



Task



Tasks adalah “aktivitas” yang kamu buat kemudian dijalankan oleh Operator. Task bisa berupa Python function atau eksternal yang bisa dipanggil. Tasks ini diharapkan bersifat idempotent yaitu jika nilai yang dimasukkan sama maka hasilnya akan tetap sama — tidak peduli berapa kali Tasks ini dijalankan

Yang perlu diingat bahwa: Dalam membuat Tasks, terdapat task_id sama halnya dengan dag_id ini bersifat unik tidak boleh digunakan berulang kali dalam satu konteks DAG itu sendiri tapi task_id boleh sama dengan DAG lainnya, misal: dag_1 memiliki task_a dan task_b maka kita boleh menggunakan task_id yang sama pada dag_2

Task Instances



Task Instance adalah Task di dalam DAG yang telah di instantiasi oleh Airflow. Semua Task terasosiasi dengan DAG Run dan Task tersebut direferensikan sebagai TaskInstance, dengan kata lain sebuah TaskInstance adalah Task yang terinstantiasi dan memiliki konteks ExecutionDate

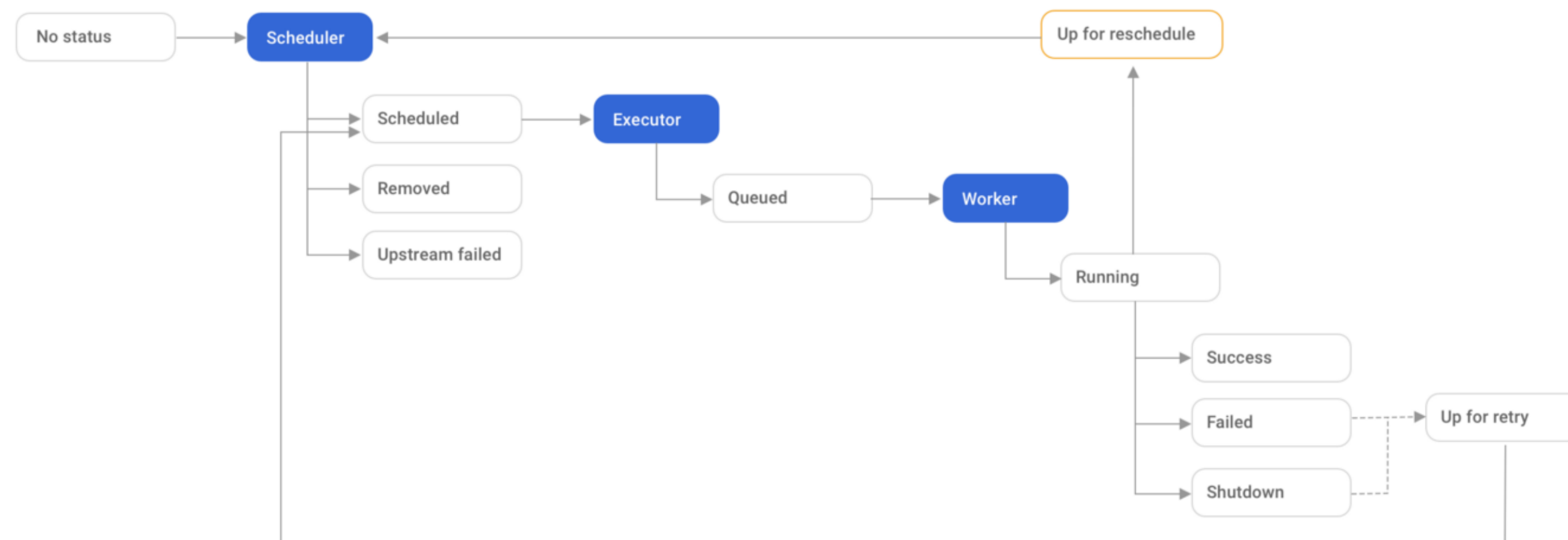
DAGRun dan TaskInstance juga konsep utama dalam Airflow, setiap DAGRun dan TaskInstance saling terhubung dengan entri dalam database metadata Airflow yang mencatat kondisi (status) mereka seperti: "queued", "running", "failed", "skipped", dan "up for retry". Membaca (read) dan memperbarui (update) status ini adalah kunci untuk penjadwalan dan proses eksekusi yang dilakukan oleh Airflow.

Task Lifecycle



Sebuah task berjalan dengan berbagai tahap dari awal hingga selesai. Pada Airflow UI, berbagai tahap tersebut direpresentasikan dengan warna.

■ success ■ running ■ failed ■ skipped ■ upstream_failed ■ up_for_reschedule ■ up_for_retry ■ queued ■ no_status



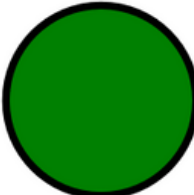



■ Component □ Task stage □ Task stage only for sensor — Stage transition --- Alternative stage transition

Contoh, flow task yang berjalan dengan baik :

1. No status (scheduler membuat task instance yang masih kosong)
2. Scheduled (task instance perlu dijalankan scheduler)
3. Queued (scheduler mengirim task ke executor untuk berjalan di-queue)
4. Running (worker mengambil task dan setelah itu dijalankan)
5. Success (task selesai dikerjakan dengan success)



pada web-ui Airflow juga dibedakan DAGs/task yang dijalankan oleh scheduler dengan yang dijalankan oleh trigger airflow trigger_dag ,

	sheduler	trigger
DAGs		
Task		

DAGs/Task yang dijalankan scheduler, memiliki border hitam, sedangkan yang dijalankan oleh trigger tanpa border.

Operator



Operator adalah yang “menjalankan” Tasks yang telah dibuat. Ketika kita membuat suatu workflow (data-pipeline) di Airflow, maka workflow tersebut didefinisikan menggunakan Operator di dalam DAG, karena setiap operator menjalankan Tasks tertentu yang ditulis sebagai Python function atau perintah shell

Scheduler



Scheduler adalah otak di balik pengaturan workflow di Airflow atau jika dianalogikan, Scheduler adalah “petugas” yang bertanggung jawab dalam memantau semua DAG beserta Tasks yang ada, dan memicu (men-trigger) semua Task Instances yang dependensinya telah terpenuhi, scheduler ini juga yang memastikan DAGs yang ada di dalam DagBag tetap tersinkronisasi dengan Airflow, makanya setiap kali kita menambahkan satu berkas Python berisi DAG kedalam DagBag Airflow akan segera mengetahuinya dan menampilkannya di Web Dashboard (selama scheduler dijalankan).



- memeriksa kondisi (status) dari TaskInstances yang terhubung dengan DagRun yang sedang aktif
- menyelesaikan semua ketergantungan tiap TaskInstance
- mengidentifikasi TaskInstance yang harus dieksekusi
- menambahkannya kedalam Worker queue
- memperbarui status dari TaskInstance dari “newly-queued” ke “queued” pada database

Schedule Interval



schedule_interval adalah rentang waktu DAG dijalankan. schedule_interval ini diletakkan pada argument DAG yang nilainya dapat berupa timedelta(n) atau berupa string yang memuat Cron Expression atau preset yang telah disediakan oleh Airflow.

Kode	Rentang waktu	Jenis
DAG(..., schedule_interval=timedelta(days=1))	Perhari	Python Timedelta
DAG(..., schedule_interval='0 0 * * 1')	Tiap hari Senin jam 00:00	Cron Expression
DAG(..., schedule_interval='@hourly')	Per satu jam	Preset

Schedule Interval



berikut ini adalah daftar preset yang diambil dari Official Documentation Ariflow

Preset	Penjadwalan	Cron
None	Tidak terjadwal	
@once	Hanya sekali	
@hourly	Tiap jam	0 * * * *
@daily	Tiap hari pada tengah malam	0 0 * * *
@weekly	Tiap minggu pagi pada tengah malam	0 0 * * 0
@monthly	Tiap awal bulan pada tengah malam	0 0 1 * *
@yearly	Tiap awal tahun pada tengah malam (1 Januari)	0 0 1 1 *

Executor



Executor adalah message queuing yang terikat erat dengan Scheduler dan menentukan proses worker yang benar-benar mengeksekusi setiap Task yang dijadwalkan. Ada berbagai macam jenis Executor, yang masing-masing menggunakan class Executor khusus. Sebagai contoh, LocalExecutor mengeksekusi Task secara paralel yang berjalan pada mesin yang sama dengan Scheduler. Executor lainnya seperti CeleryExecutor mengeksekusi Task menggunakan worker yang ada pada “sekelompok mesin” worker yang terpisah ini biasa disebut sebagai distributed worker atau dengan kata lain worker yang terdistribusi.

DAG Run



DAG Run adalah DAG yang dijalankan oleh Airflow, sebenarnya dibalik ini Airflow membuat objek yang mewakili instantiasi DAG dalam waktu tertentu.

contoh pada sub bagian Start Date, jika sekarang tanggal 2018-11-04 dan kita menetapkan `schedule_interval` nya `@daily` maka DAG Run yang kita miliki adalah 4 karena `start_date` kita adalah 2018-11-01, yang artinya Airflow sudah menjalankan DAG tersebut empat kali.

Lalu kapan DAG kita dijalankan oleh Airflow? Jawabannya adalah, ketika `start_date` + `schedule_interval` telah terlewati.

Execution Date



Adalah tanggal atau waktu yang menunjukkan kapan DAG dijalankan. Karena tentunya kita ingin mengetahui kapan saja DAG tersebut dijalankan dalam waktu tertentu maka Execution Date muncul ketika DAG Run diinstantiasi.

Terima Kasih!

Sampai jumpa di
Learning Progress Review
kami berikutnya!



Hilda Meiranita Prastika Dewi



Nur Indrasari



Rezha Sulvian



Thasha Dinya Ainsha