Apache Airflow

Workflow manager

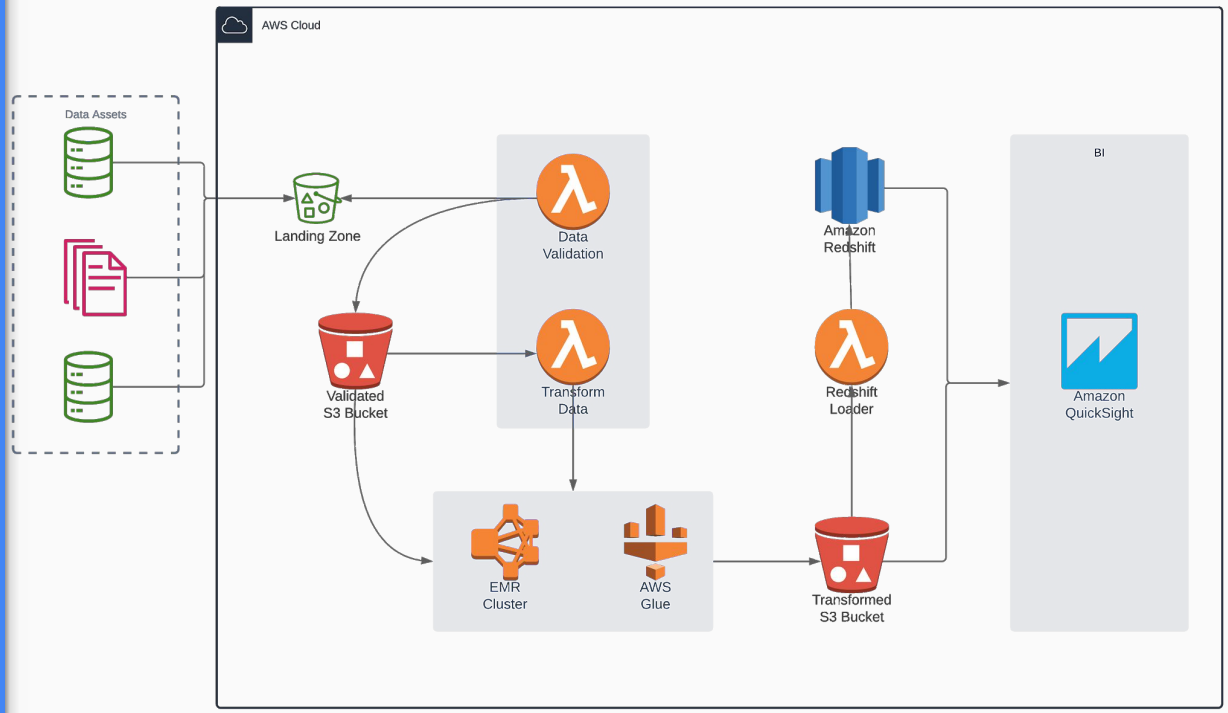
Agenda

- Introduction to use-case
- Theoretical Example
- Airflow description
- Demo - GUI
- Demo - DAG

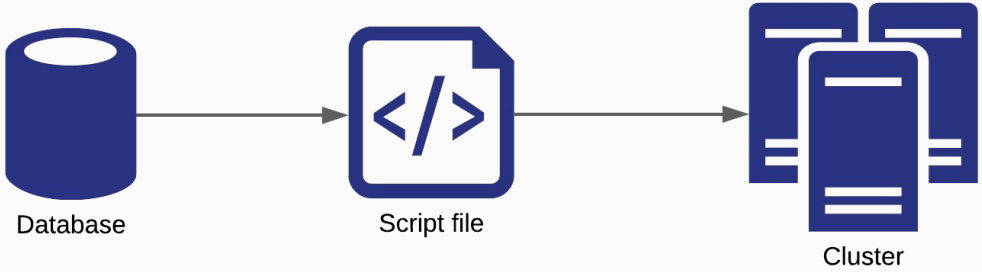
Imagine following use-case

You are working on the project responsible for collecting data from various sources, their transformation, visualization, and archiving.

Your system consists of a vast amount of steps dependent on each other and preferably being executed based on some schedule.



ETL - Naive approach



- A script that pulls data from a database and sends it to the HDFS to post-processing
- Script scheduled for instance as a Cronjob

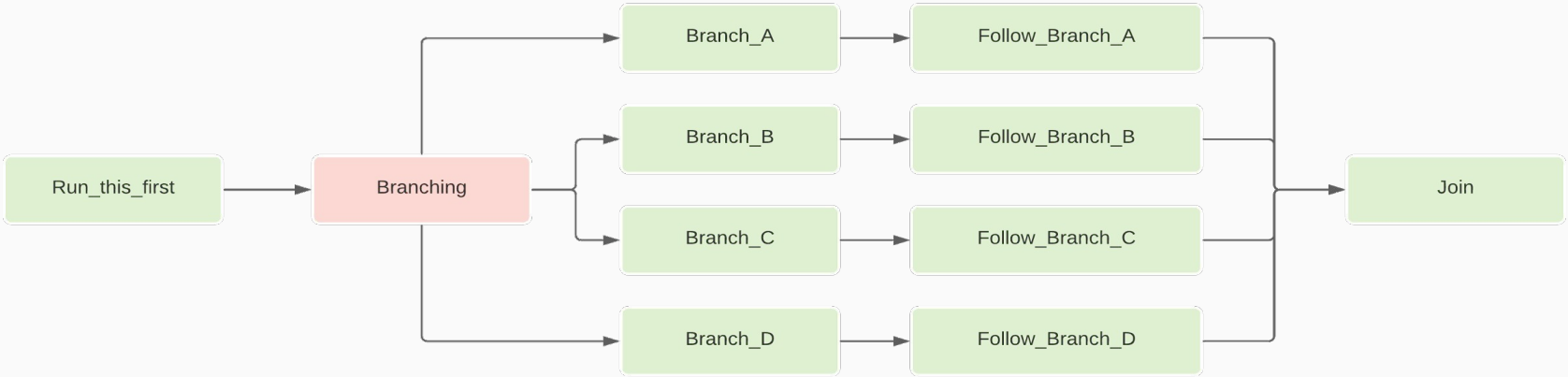
Here comes the Airflow

A platform to programmatically schedule, author and monitor workflows or data pipelines.

- Developed by the Airbnb & Maintained by Apache Software Foundation
- Distributed executing and scheduling tasks across worker nodes
- Framework to define tasks and dependencies in Python
- REST-API - metadata accessing
- Wide-ranged database support
- Plugins architecture
- Excellent logging service. View of present and past runs

How it works

A workflow is represented by a DAG where nodes represent multiple tasks which can be executed in parallel and edges represent order and dependencies among these tasks.



Demo

The following demo aims to introduce a fundamental overview of the Airflow ecosystem and its utilization.

Prerequisites

- Python 3+
- Docker
- Docker-Compose
- Git

Next steps:

1. Navigate to the following address:
   a. https://github.com/MichalKyjovsky/NSWI126
2. Clone the repository into your local filesystem
3. Start the Docker
4. Navigate to the *airflow* subdirectory
5. Run following commands:
   a. *docker-compose up -d --build*
   b. *docker-compose logs -f*
6. Navigate to the following address
   a. http://localhost:8080/admin

Thank you for your attention