



SPONSORS



UNIKUBE

ISOVALENT



portworx
by Pure Storage



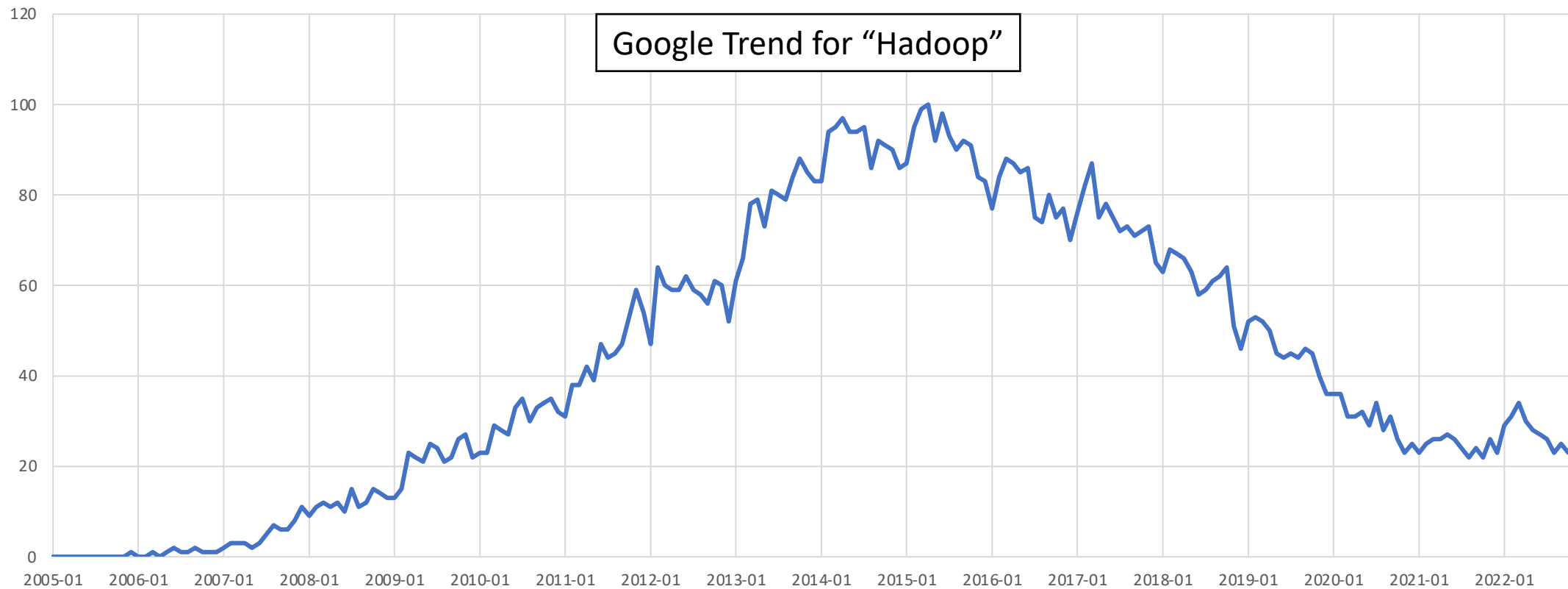
Speaker: Dr. Paul Staab

Company: Freelancer

Data Engineering, Machine Learning and DevOps

Running Big Data Pipelines on Kubernetes *without Crashing the Cluster*

On-premise Hadoop clusters are legacy and need to be migrated.



Public clouds have managed Hadoop Services.
There is no easy solution for Kubernetes.



Google Cloud Platform



kubernetes



amazon
EMR



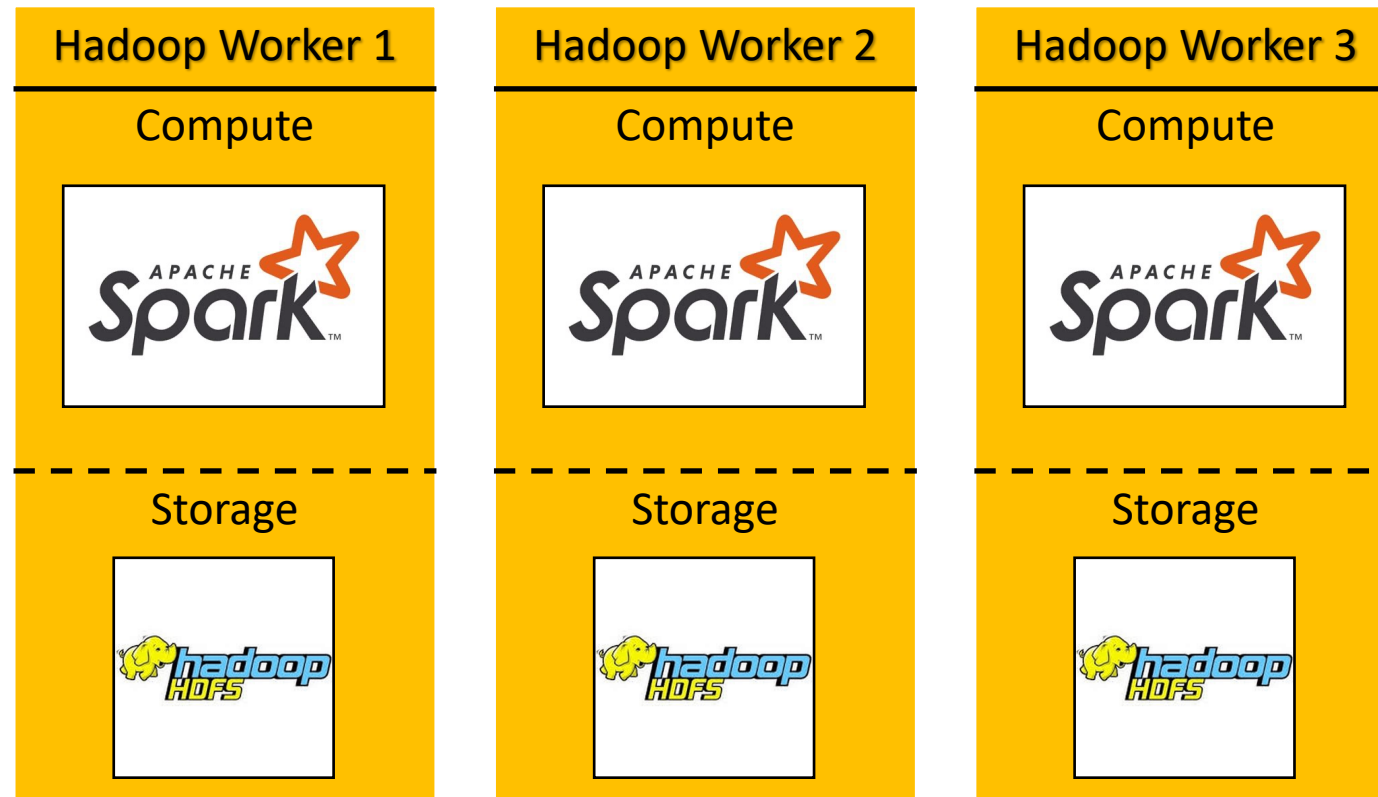
Azure Databricks



Cloud Dataproc



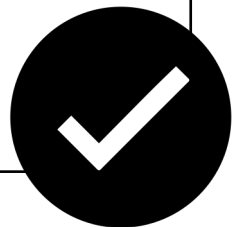
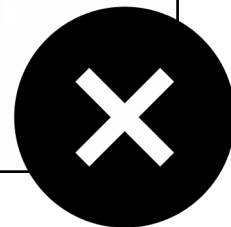
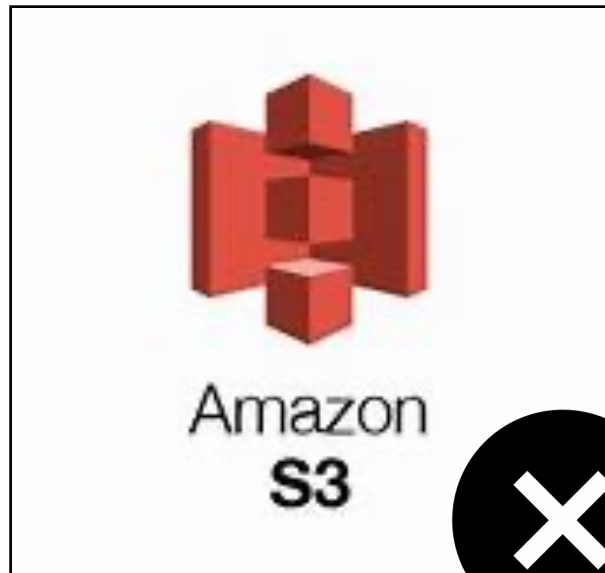
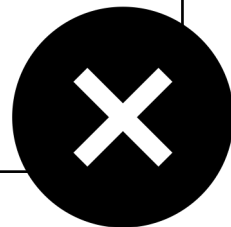
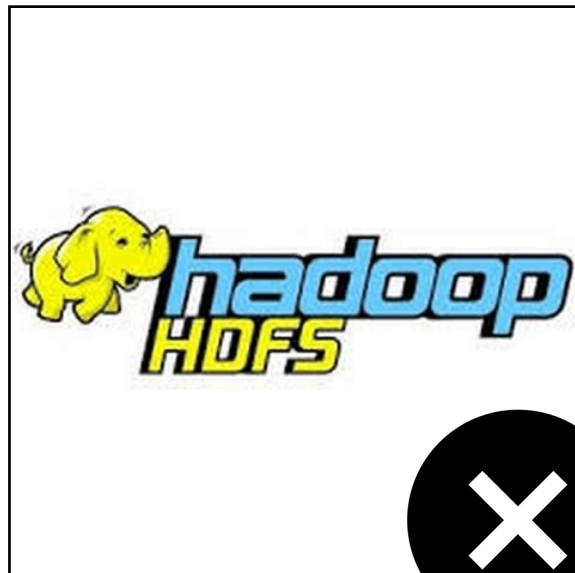
Hadoop clusters provide co-located compute and storage resources.



Batch Jobs

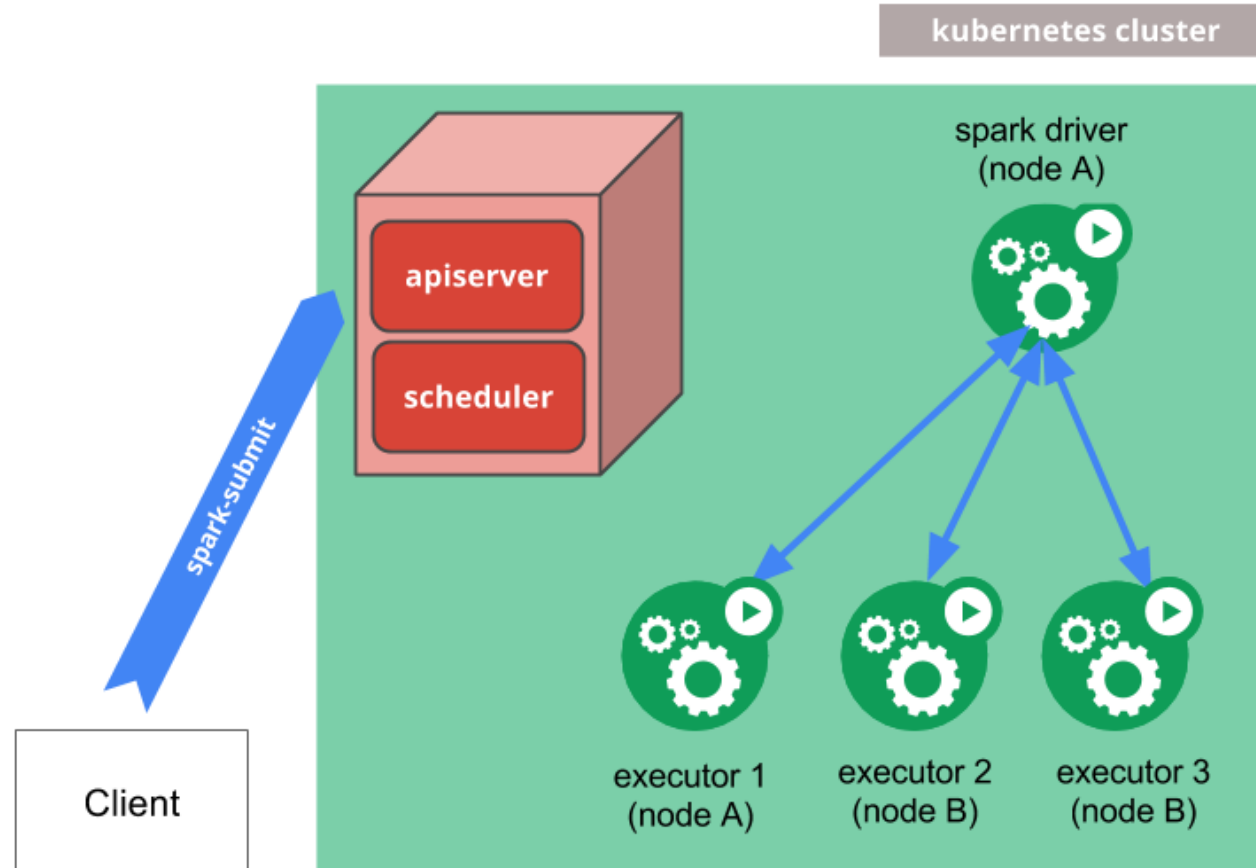
Using non-local object storage instead of HDFS works *fine*.

Storage on Kubernetes



Spark natively supports running on k8s.
But it wants to submit itself from somewhere.

Compute
on
Kubernetes



GCP created an operator so that we can submit YAMLS instead.

Compute
on
Kubernetes

The screenshot shows the GitHub repository page for `GoogleCloudPlatform/spark-on-k8s-operator`. The repository is public and has 84 watches, 1.1k forks, and 2.1k stars. The main branch is `master`. The repository description is "Kubernetes operator for managing the lifecycle of Apache Spark applications on Kubernetes." The repository contains several folders: `.github/workflows`, `charts/spark-operat...`, `docs`, `examples`, `hack`, `manifest`, `pkg`, `spark-docker`, and `sparkctl`. The repository is licensed under Apache-2.0 and has 2.1k stars, 84 watchers, and 1.1k forks.

```
! example-spark-application.yaml > ...
1  ---
2  apiVersion: sparkoperator.k8s.io/v1beta2
3  kind: SparkApplication
4  metadata:
5    # ...
6  spec:
7    driver:
8      cores: 1
9      memory: 1g
10     serviceAccount: spark
11   executor:
12     cores: 1
13     instances: 20
14     memory: 5g
15     image: gcr.io/ynli-k8s/spark:v3.1.1
16     type: Python
17     pythonVersion: 3
18     mainApplicationFile: local:///opt/my-big-big-data-application.py
19     # ...
20  ---
21
```

Lessons Learned

- We were too optimistic. Expect failures, do load testing, limit the blast radius and do not optimize too early.
- Try to keep pods small and run more of them. This makes scheduling easier and more reliable and avoids many problems (e.g., PIDs).
- Cluster auto scaling and priority classes are a good idea when running batch jobs.

Once we get more batch jobs, we need more sophisticated scheduling.

	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00
Job A	█							
Job B		█	█					
Job C		█	█	█				
Job D					█	█		
Job E							█	█
Job F							█	█

Job B depends on job A

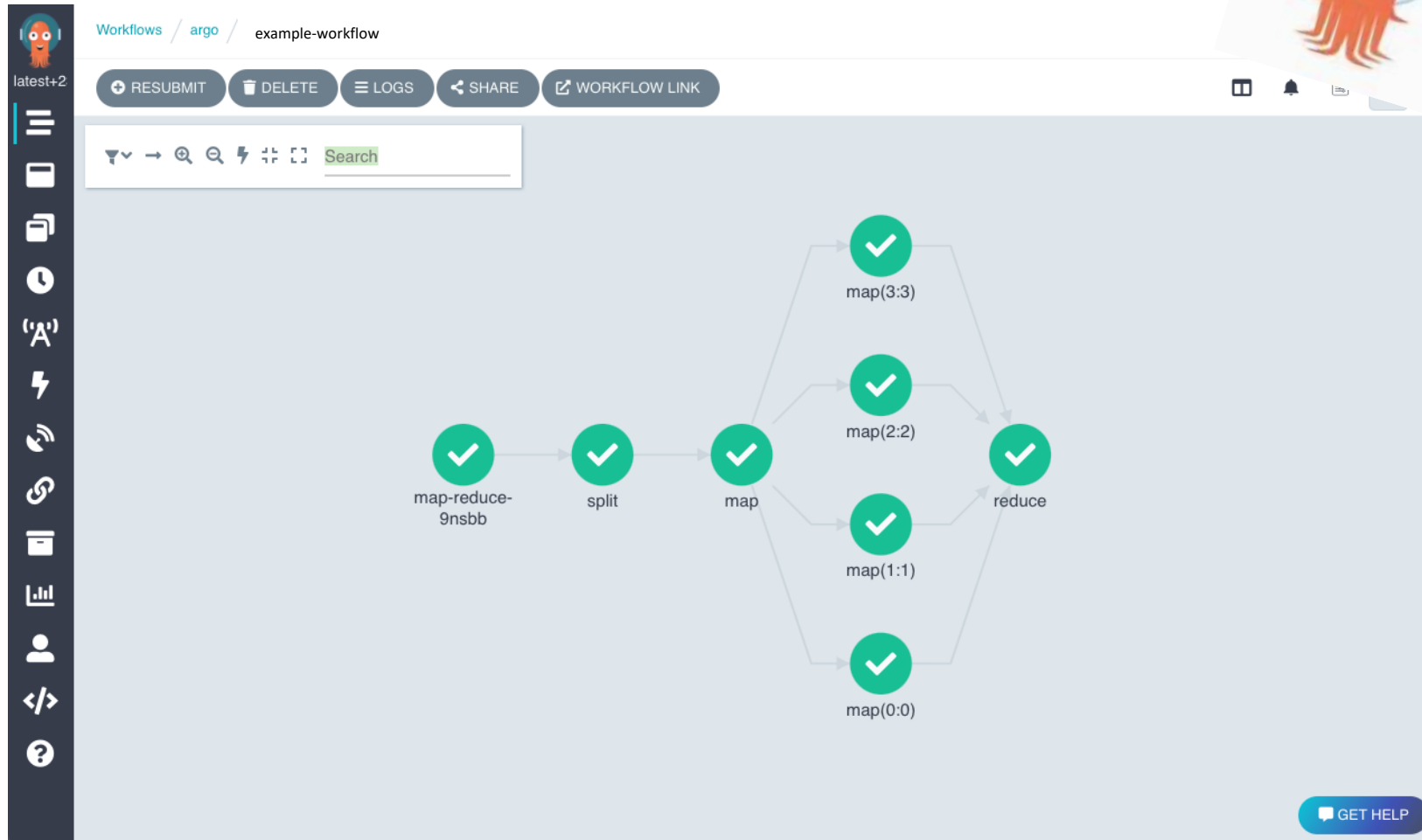
Job C depends on job A

Job D depends on jobs A, B & C

Job E depends on job D

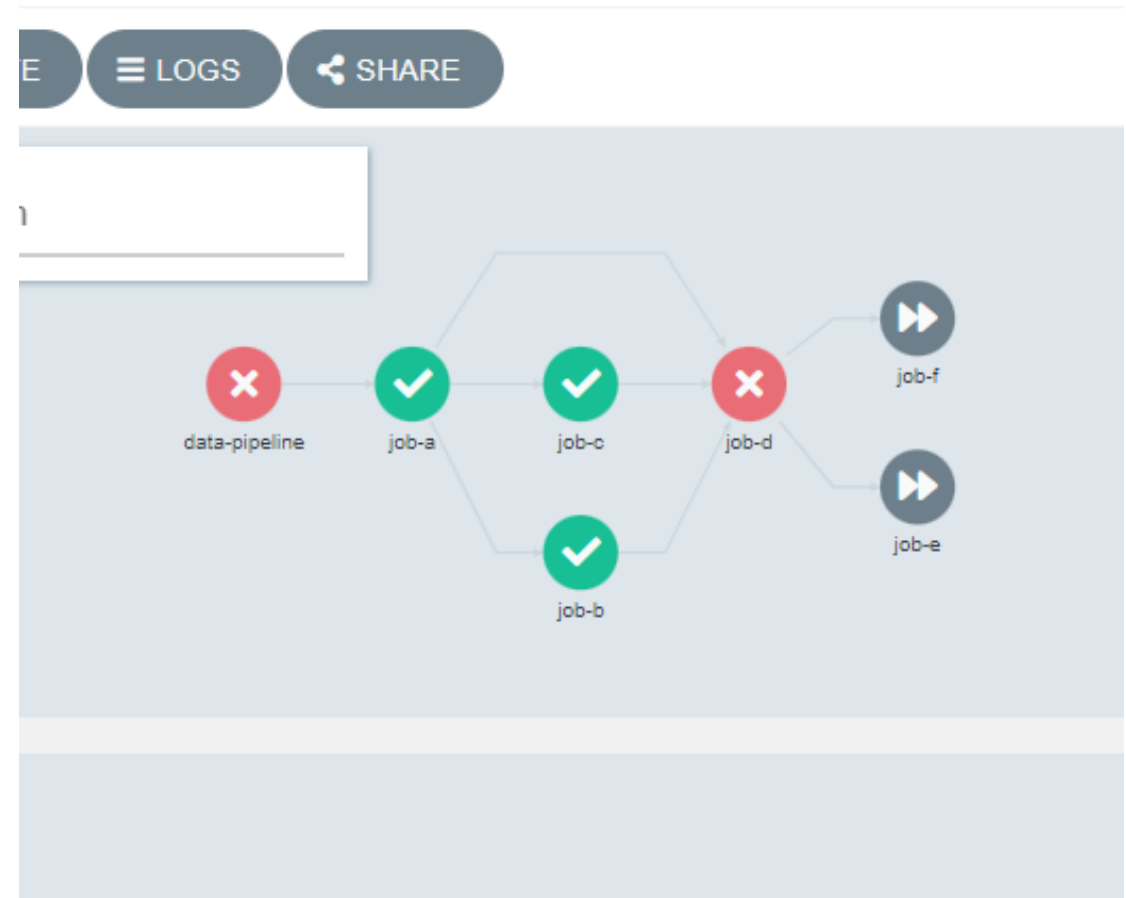
Job F depends on job D

Argo Workflows is a cloud-native orchestrator for batch pipelines.



With Argo Workflows we can define complex batch pipelines.

```
! example-argo-workflow.yaml > ...
1  ---
2  apiVersion: argoproj.io/v1alpha1
3  kind: Workflow
4  metadata:
5    name: data-pipeline
6  spec:
7    entrypoint: run-data-pipeline
8    templates:
9      - name: run-data-pipeline
10     dag:
11       tasks:
12         - name: job-a
13           template: python-job-a
14         - name: job-b
15           depends: "job-a"
16           template: spark-job-b
17         - name: job-c
18           depends: "job-a"
19           template: sql-query-c
20         - name: job-d
21           depends: "job-a & job-b & job-c"
22           template: python-job-d
23         - name: job-e
24           depends: "job-d"
25           template: sql-query-e
26         - name: job-f
27           depends: "job-d"
28           template: sql-query-f
29       - name: python-job-a
30         container:
31           name: main
32           image: 'python:3.9'
33           command:
34             - python
35           args:
36             - etl_a.py
37         # [...]
38  ---
39
```



Take Away Messages

- Migrating a Hadoop cluster to Kubernetes is possible, but not easy. Spark on Kubernetes is still in an early stage.
- Argo Workflows is fantastic for orchestrating complex batch pipelines and integrates well with existing Kubernetes tooling.
- Expect failure initially, limit your blast radius and only optimize cost when everything is stable.