HADOOP 101 - IBM

1. Introduction to Hadoop
2. Hadoop Architecture & HDFS
3. Hadoop Administration
4. Hadoop Components

DESCRIPTION

Module 1 - Introduction to Hadoop

- Understand what Hadoop is
- Understand what Big Data is
- Learn about other open source software related to Hadoop
- Understand how Big Data solutions can work on the Cloud

Module 2 - Hadoop Architecture

- Understand the main Hadoop components
- Learn how HDFS works
- List data access patterns for which HDFS is designed
- Describe how data is stored in an HDFS cluster

Module 3 - Hadoop Administration

- Add and remove nodes from a cluster
- Verify the health of a clusterStart and stop a clusters components
- Modify Hadoop configuration parameters
- Setup a rack topology

Module 4 - Hadoop Components

- Describe the MapReduce philosophy
- Explain how Pig and Hive can be used in a Hadoop environment
- Describe how Flume and Sqoop can be used to move data into Hadoop
- Describe how Oozie is used to schedule and control Hadoop job execution

HADOOP 101

- The definition of Big Data
- The Hadoop architecture including MapReduce and HDFS
- How to use the Hadoop file system shell and the Ambari Console to work with HDFS
- Starting and stopping Hadoop components
- How to add/remove a node to/from a Hadoop Cluster
- Determining how much space is available on each node in a Hadoop cluster
- How to modify Hadoop configuration parameters
- The concepts and purpose of other components such as MapReduce, Pig, Hive, Flume, Sqoop, and Oozie

I. INTRODUCTION TO HADOOP

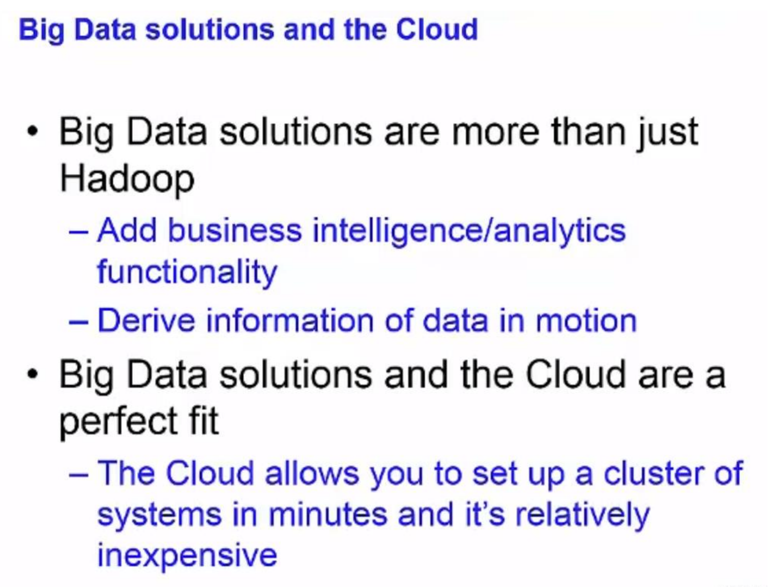
- Hadoop: Open Source from the Apache Software

Foundation developed in Java.

- Parallel Processing.
- Not used by OTLP, OLAP, DSS.

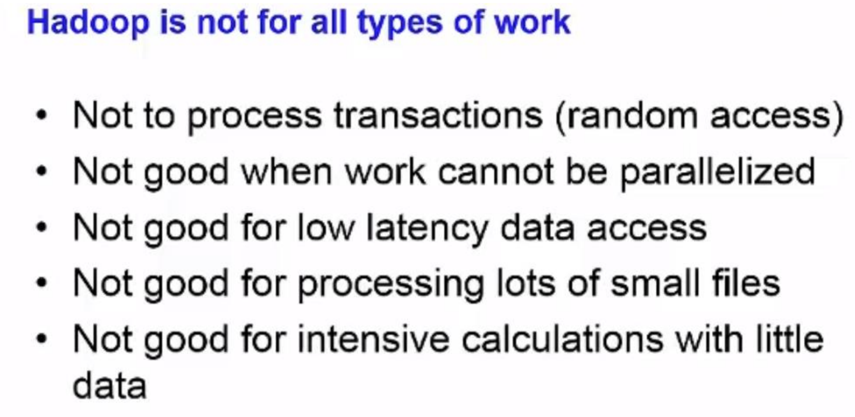
Hadoop-related open source projects

eclipse
Lucene™
HBASE
HIVE
PIG
Spark
ZooKeeper
AVRO™
UIMA
Unstructured Information Management Architecture
An Apache Project
Apache Ambari



Big Data solutions and the Cloud

- Big Data solutions are more than just Hadoop
  - Add business intelligence/analytics functionality
  - Derive information of data in motion
- Big Data solutions and the Cloud are a perfect fit
  - The Cloud allows you to set up a cluster of systems in minutes and it's relatively inexpensive



Hadoop is not for all types of work

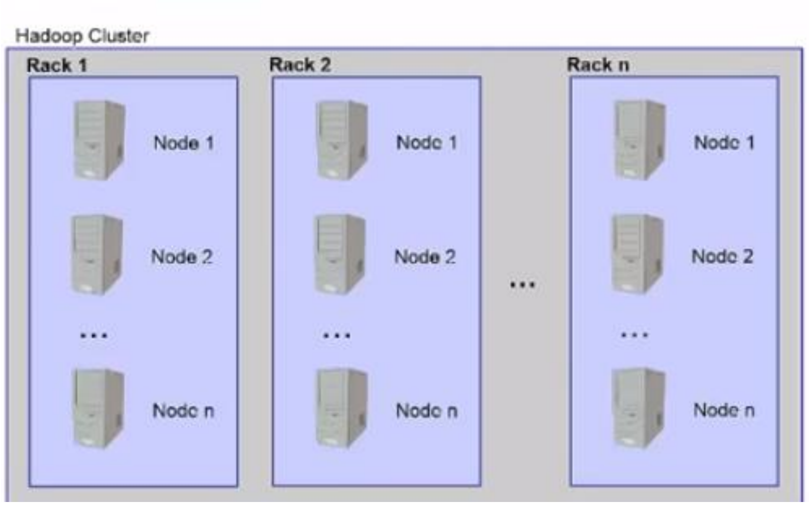
- Not to process transactions (random access)
- Not good when work cannot be parallelized
- Not good for low latency data access
- Not good for processing lots of small files
- Not good for intensive calculations with little data

2. HADOOP ARCHITECTURE

1. The main Hadoop components
2. How HDFS works
3. Data access patterns for which HDFS is designed
4. How data is stored in an HDFS cluster

Two main componentes:

1. HDFS
2. MapReduce Engine



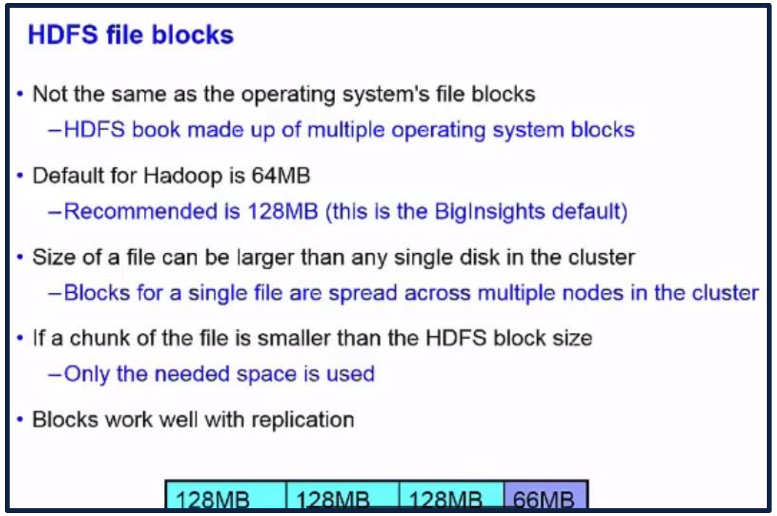
HDFS runs on the top of the existing file system:

- Not POSIX compilant
- Designed to tolerate high component failure rate

  (Reability is through replication)
- Designed to handle ver large files

  (Large streaming dataaccess patterns)

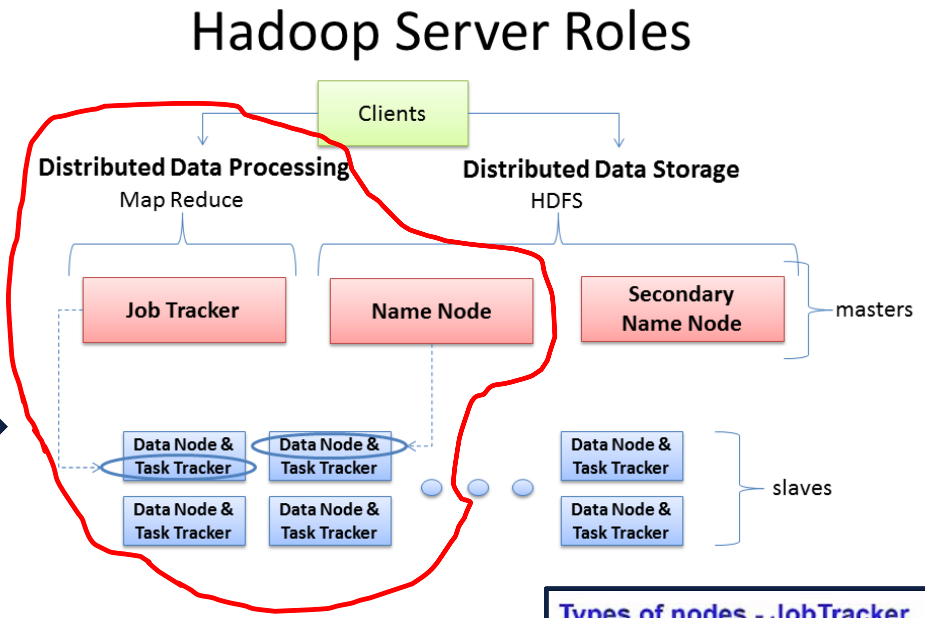
  (No random access)
- Uses blocks tos store a file or parts of a file



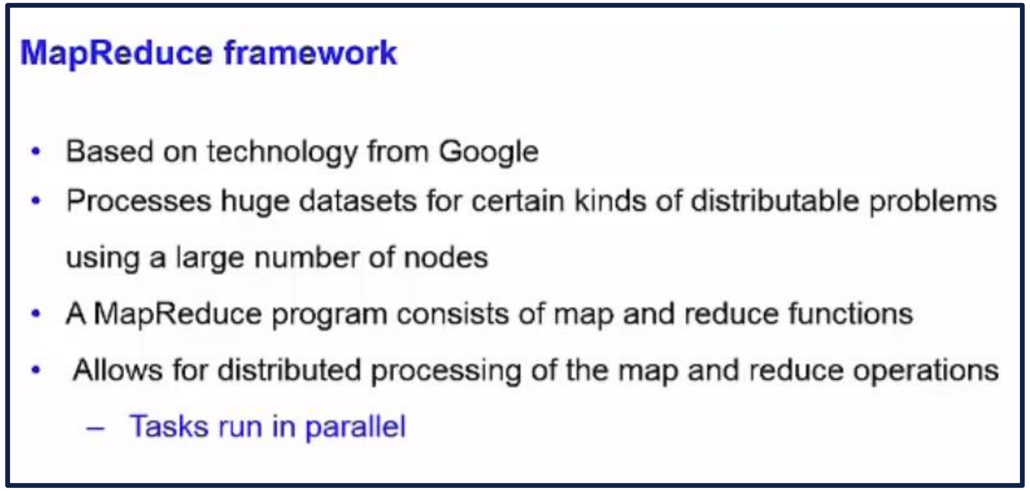
HDFS file blocks

- Not the same as the operating system's file blocks
  - –HDFS book made up of multiple operating system blocks
- Default for Hadoop is 64MB
  - –Recommended is 128MB (this is the BigInsights default)
- Size of a file can be larger than any single disk in the cluster
  - –Blocks for a single file are spread across multiple nodes in the cluster
- If a chunk of the file is smaller than the HDFS block size
  - –Only the needed space is used
- Blocks work well with replication

HADOOP ARCHITECTURE

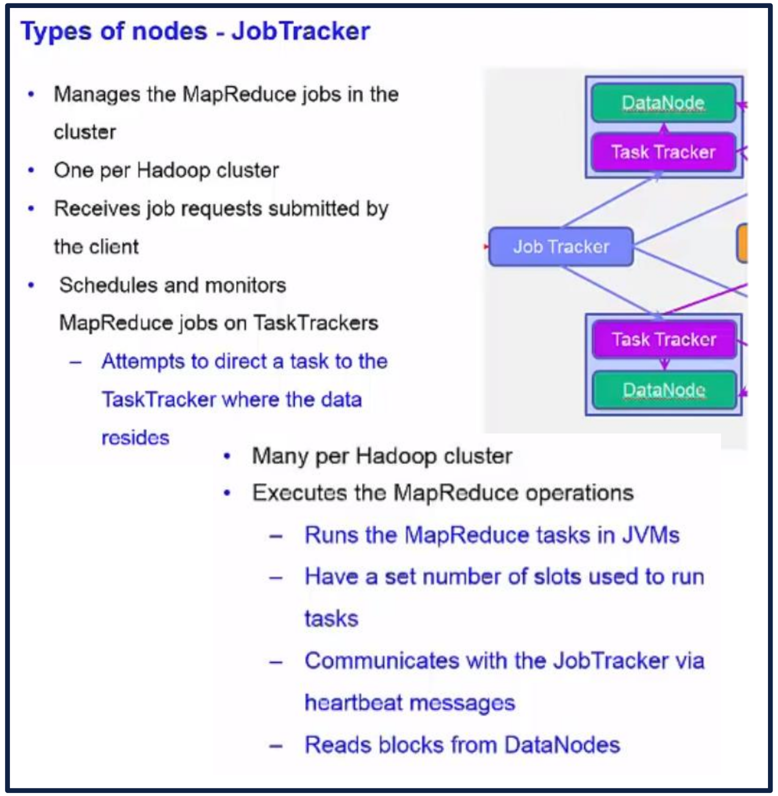


Hadoop Server Roles



MapReduce framework

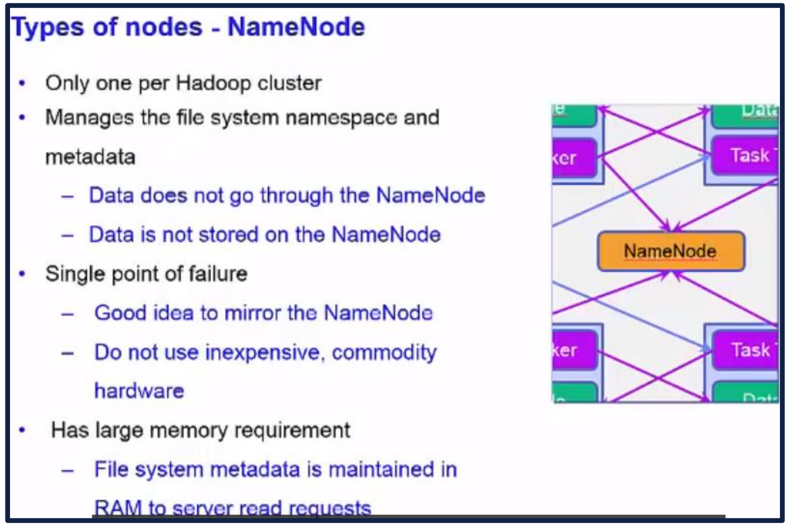
- Based on technology from Google
- Processes huge datasets for certain kinds of distributable problems using a large number of nodes
- A MapReduce program consists of map and reduce functions
- Allows for distributed processing of the map and reduce operations
  - Tasks run in parallel



Types of nodes - JobTracker

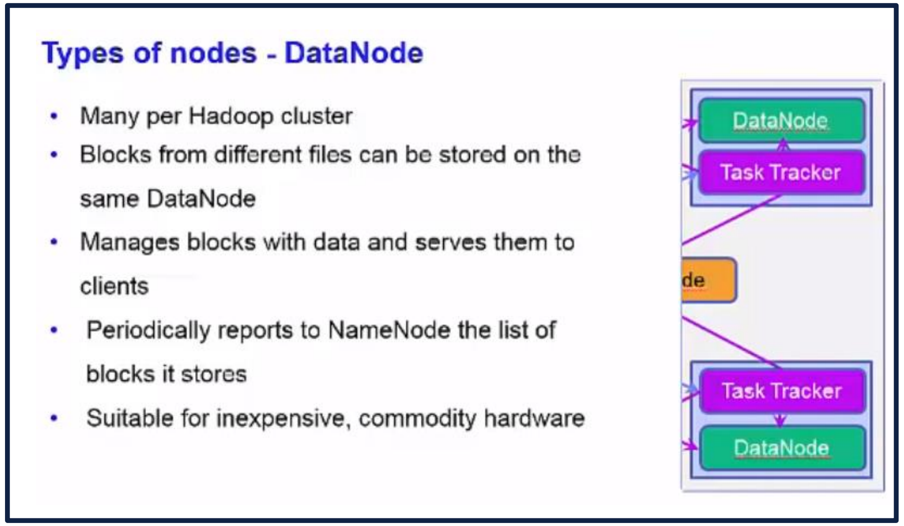
- Manages the MapReduce jobs in the cluster
- One per Hadoop cluster
- Receives job requests submitted by the client
- Schedules and monitors MapReduce jobs on TaskTrackers
  - Attempts to direct a task to the TaskTracker where the data resides

- Many per Hadoop cluster
- Executes the MapReduce operations
  - Runs the MapReduce tasks in JVMs
  - Have a set number of slots used to run tasks
  - Communicates with the JobTracker via heartbeat messages
  - Reads blocks from DataNodes



Types of nodes - NameNode

- Only one per Hadoop cluster
- Manages the file system namespace and metadata
  - Data does not go through the NameNode
  - Data is not stored on the NameNode
- Single point of failure
  - Good idea to mirror the NameNode
  - Do not use inexpensive, commodity hardware
- Has large memory requirement
  - File system metadata is maintained in RAM to server read requests

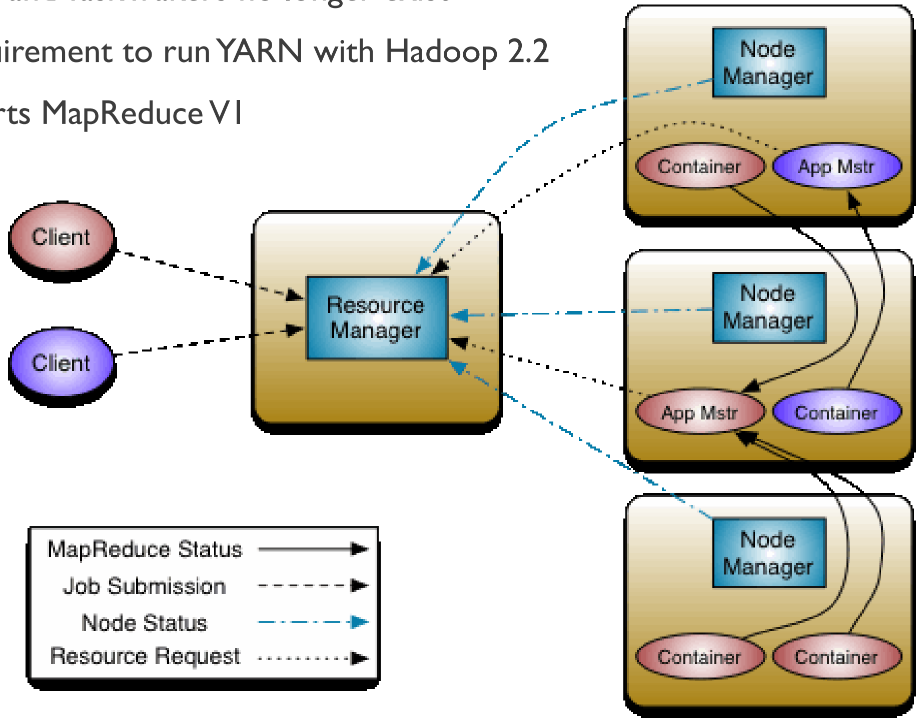


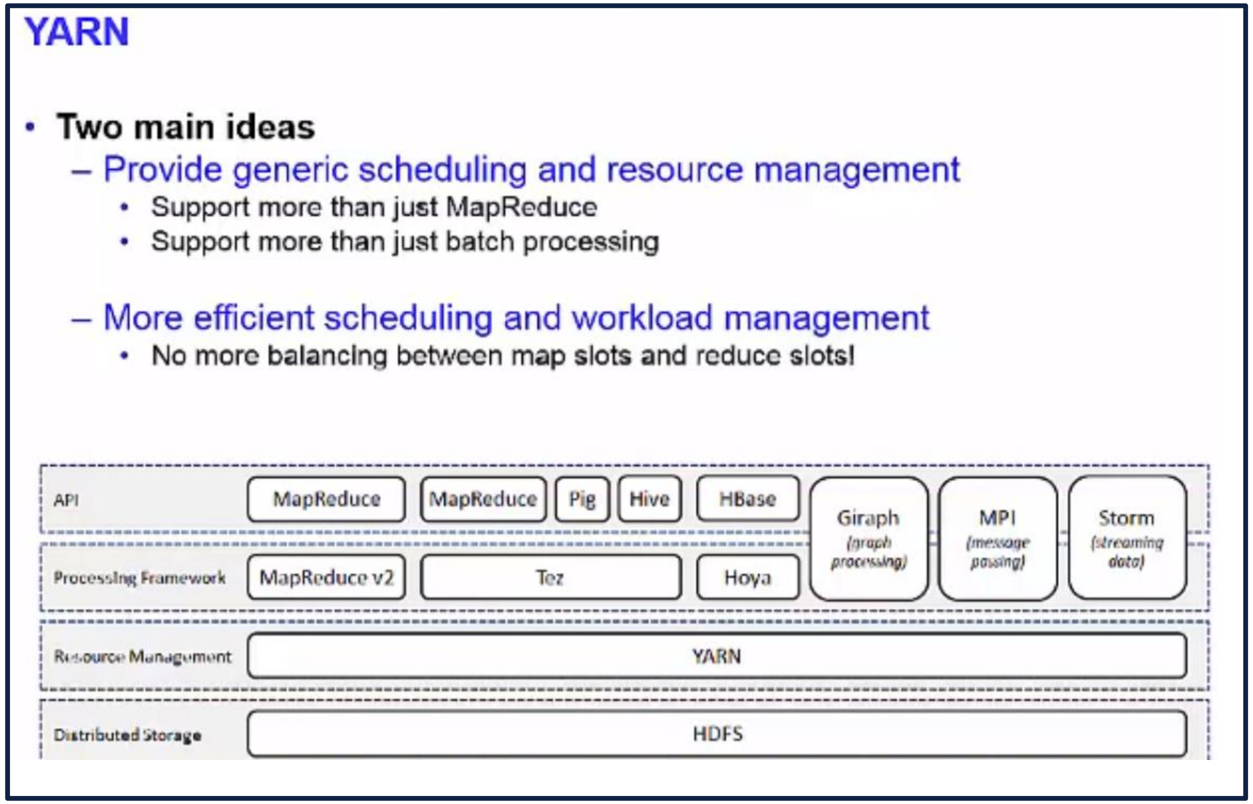
Types of nodes - DataNode

- Many per Hadoop cluster
- Blocks from different files can be stored on the same DataNode
- Manages blocks with data and serves them to clients
- Periodically reports to NameNode the list of blocks it stores
- Suitable for inexpensive, commodity hardware

HADOOP ARCHITECTURE

- Provides YARN
  - Referred to as MapReduce V2
  - Resource manager and scheduler external to any framework
  - DataNodes still exist
  - JobTracker and TaskTrakers no longer exist
- Not a requirement to run YARN with Hadoop 2.2
  - Still supports MapReduce V1



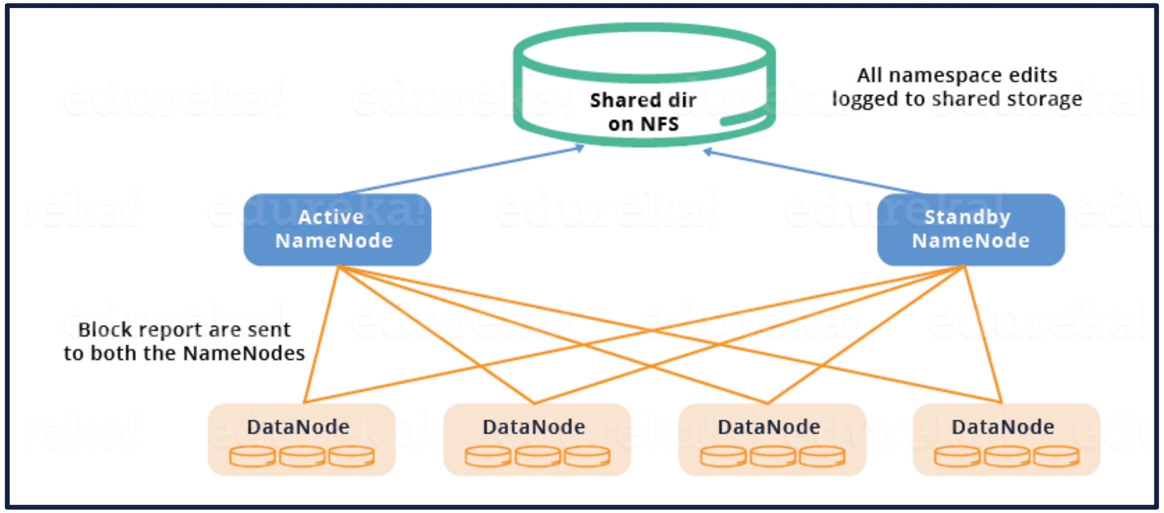


YARN

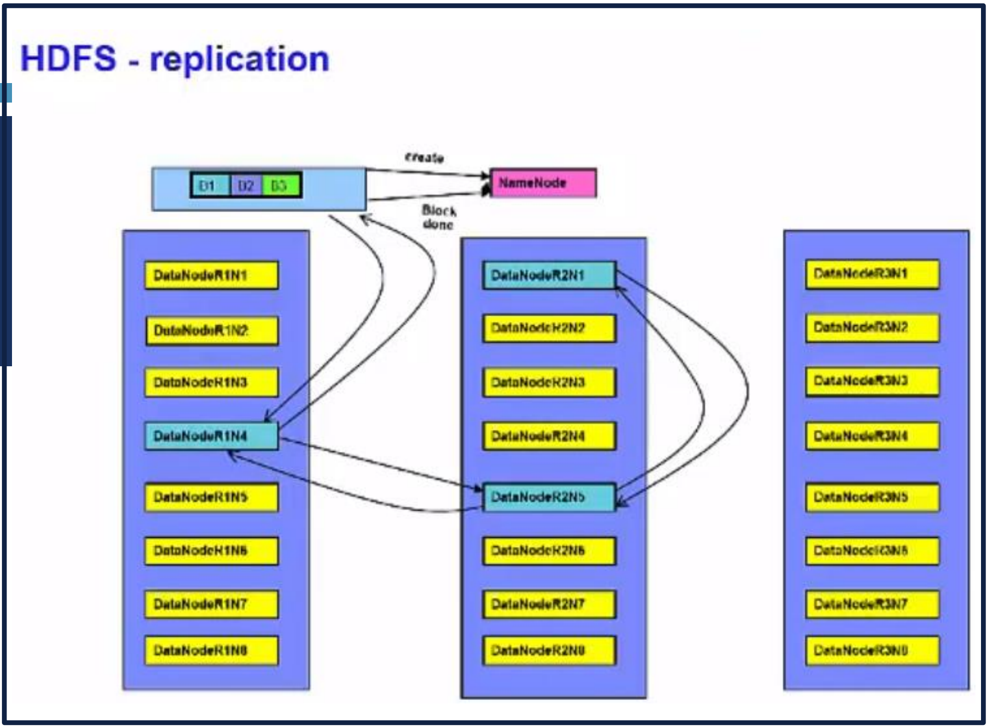
- **Two main ideas**
  - Provide generic scheduling and resource management
    - Support more than just MapReduce
    - Support more than just batch processing
  - More efficient scheduling and workload management
    - No more balancing between map slots and reduce slots!

API	MapReduce	MapReduce	Pig	Hive	HBase	Giraph *(graph processing)*	MPI *(message passing)*	Storm *(streaming data)*
Processing Framework	MapReduce v2	Tez			Hoya			
Resource Management	YARN							
Distributed Storage	HDFS							

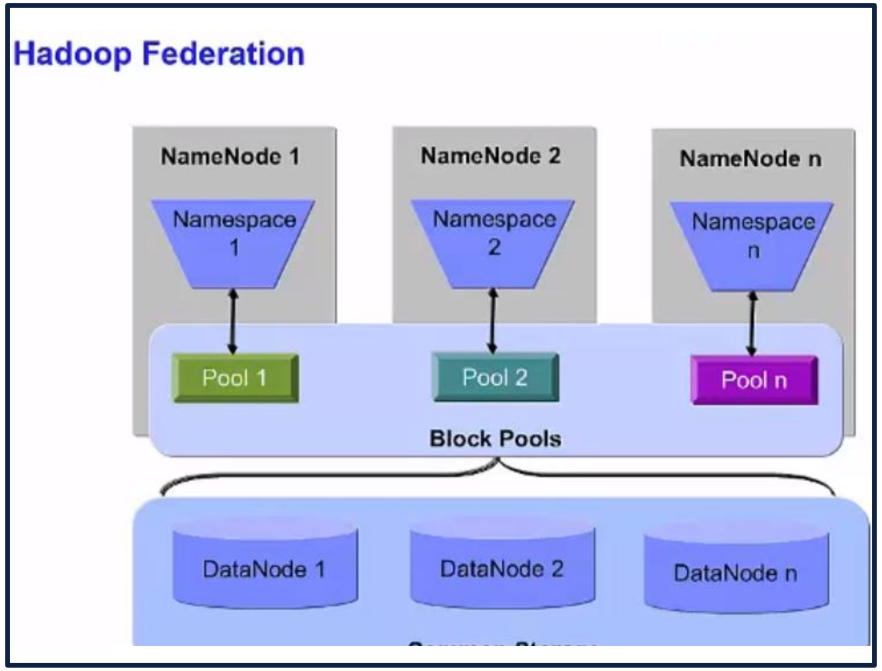
HADOOP ARCHITECTURE



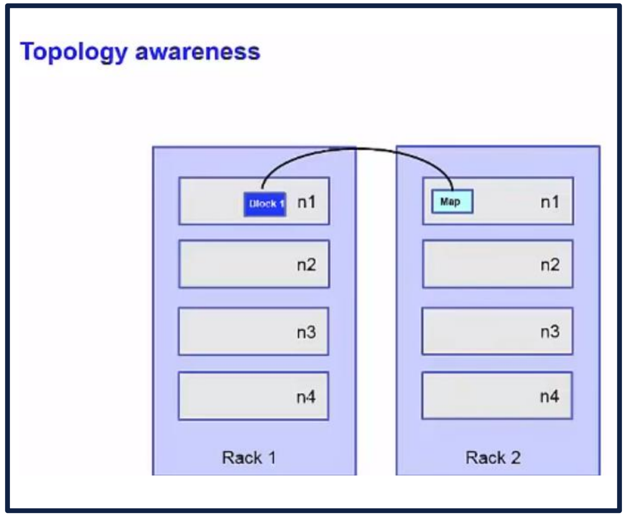
HDFS - replication



Hadoop Federation



Topology awareness



HDFS COMMAND LINE

- How to invoke the HDFS shell:

hdfs dfs <args>

hdfs dfs -ls

Hdfs dfs –cp (ex: copy from local to HDFS)

Defaults: *core-site.xml*

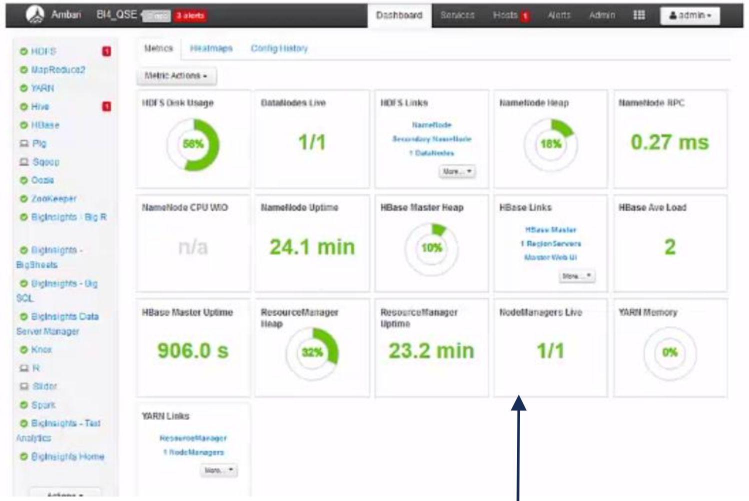
- List the HDFS commands:

A number of POSIX-like commands: cat, chgrp, chmod, chown, cp, du, ls, mkdir, mv, rm, stat, tail

Some HDFS-specific commands: copyFromLocal / put, copyToLocal / get, getMerge, setRep

- Compare executing HDFS shell commands with using the Ambari Console:

Ambari Console



HDFS Disk Usage 56%

DataNodes Live 1/1

HDFS Links

NameNode Heap 18%

NameNode RPC 0.27 ms

NameNode CPU WIO n/a

NameNode Uptime 24.1 min

HBase Master Heap 10%

HBase Links

HBase Ave Load 2

HBase Master Uptime 906.0 s

ResourceManager Heap 32%

ResourceManager Uptime 23.2 min

NodeManagers Live 1/1

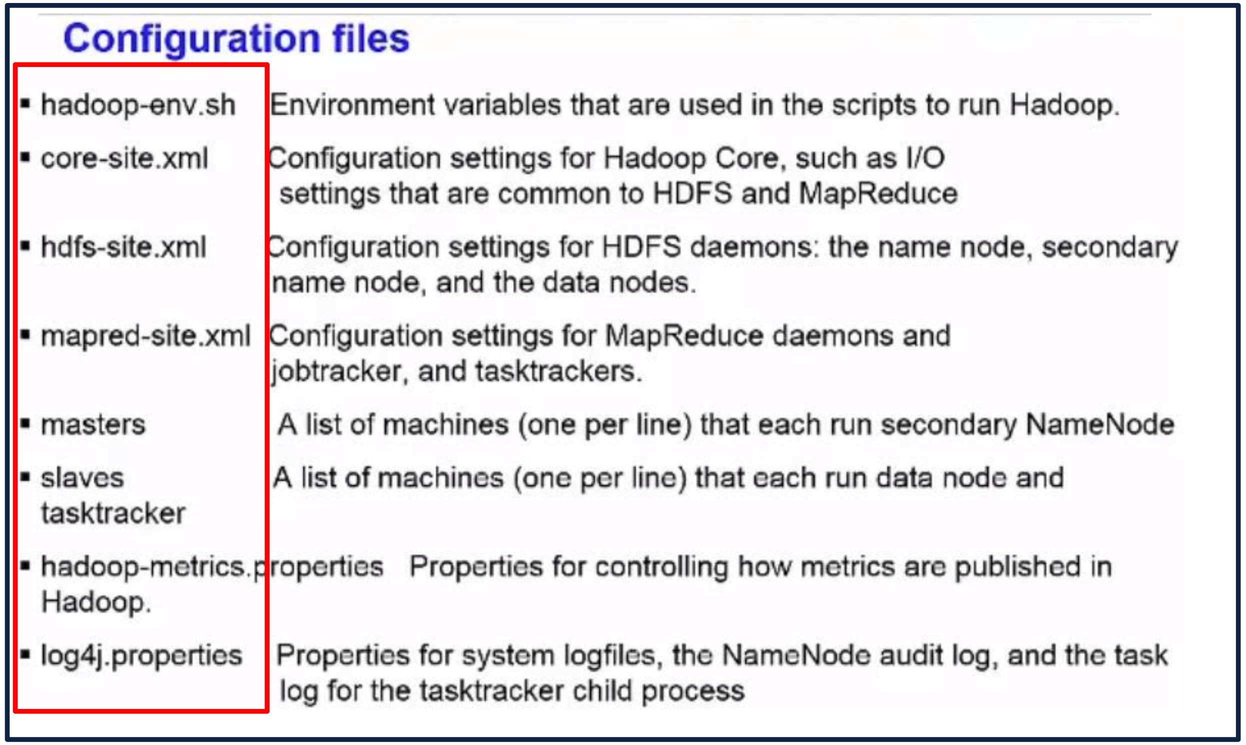
YARN Memory 0%

YARN Links

The Ambari Console is a graphical way to work with HDFS, the services tab provides a simple way to view the status os the Hadoop componentes.

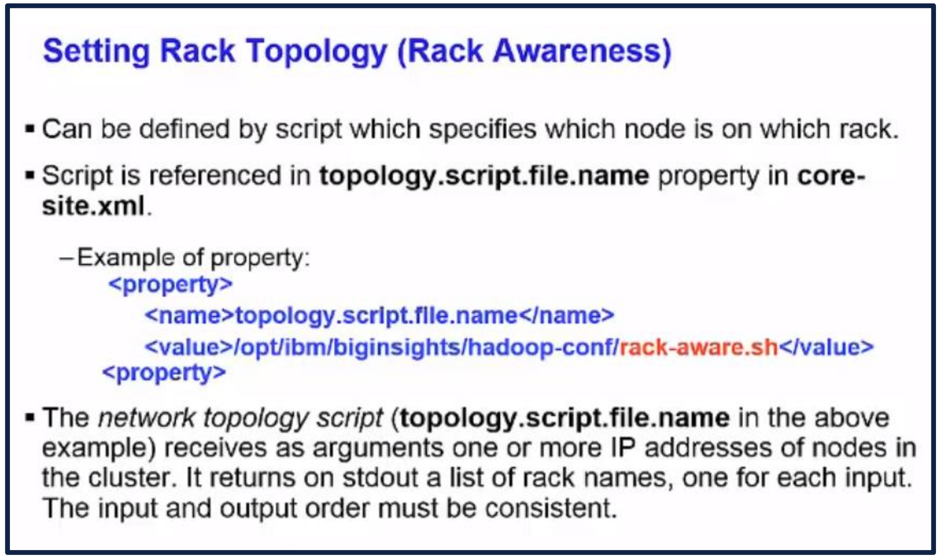
3. HADOOP ADMINISTRATION

- Adding and removing nodes from a cluster: using web console(Ambari) to remove nodes
- How to verify the health of a cluster: hdfs dfsadmin –report
- How to start and stop a cluster's components: Ambari console
- Modifying Hadoop configuration parameters →
- Setting up a rack topology



Configuration files

- hadoop-env.sh Environment variables that are used in the scripts to run Hadoop.
- core-site.xml Configuration settings for Hadoop Core, such as I/O settings that are common to HDFS and MapReduce
- hdfs-site.xml Configuration settings for HDFS daemons: the name node, secondary name node, and the data nodes.
- mapred-site.xml Configuration settings for MapReduce daemons and jobtracker, and tasktrackers.
- masters A list of machines (one per line) that each run secondary NameNode
- slaves A list of machines (one per line) that each run data node and tasktracker
- hadoop-metrics.properties Properties for controlling how metrics are published in Hadoop.
- log4j.properties Properties for system logfiles, the NameNode audit log, and the task log for the tasktracker child process



Setting Rack Topology (Rack Awareness)

- Can be defined by script which specifies which node is on which rack.
- Script is referenced in **topology.script.file.name** property in **core-site.xml**.
  - Example of property:

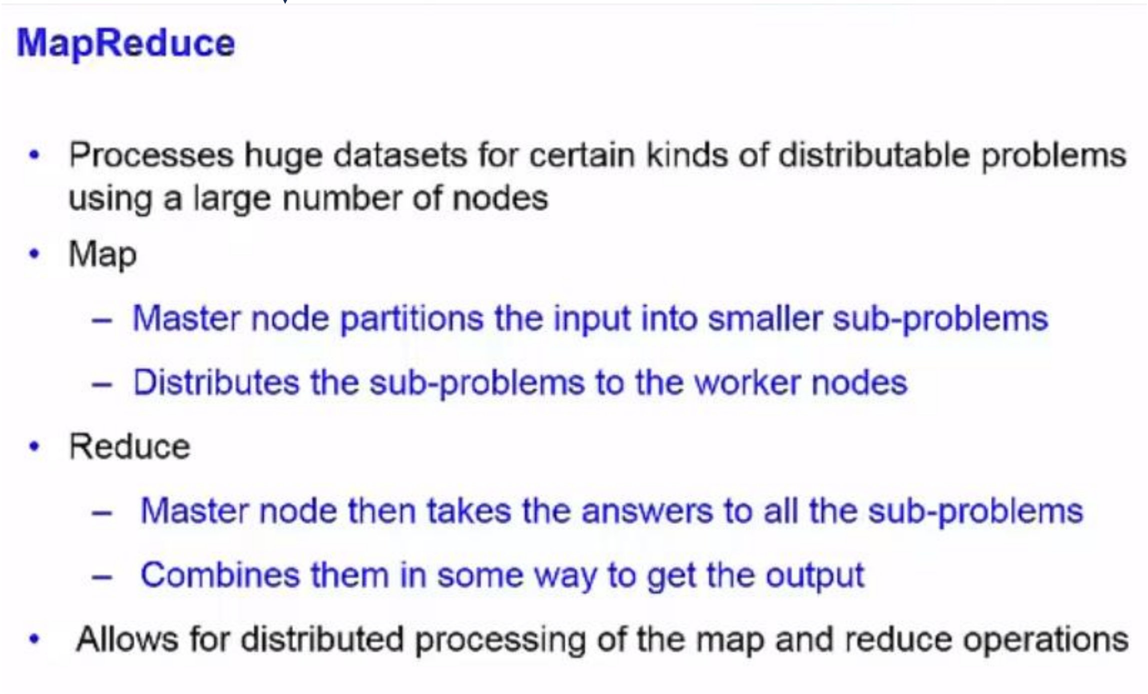
```
<property>
    <name>topology.script.flle.name</name>
    <value>/opt/ibm/biginsights/hadoop-conf/rack-aware.sh</value>
<property>
```

- The *network topology script* (**topology.script.file.name** in the above example) receives as arguments one or more IP addresses of nodes in the cluster. It returns on stdout a list of rack names, one for each input. The input and output order must be consistent.

4. HADOOP COMPONENTS

4

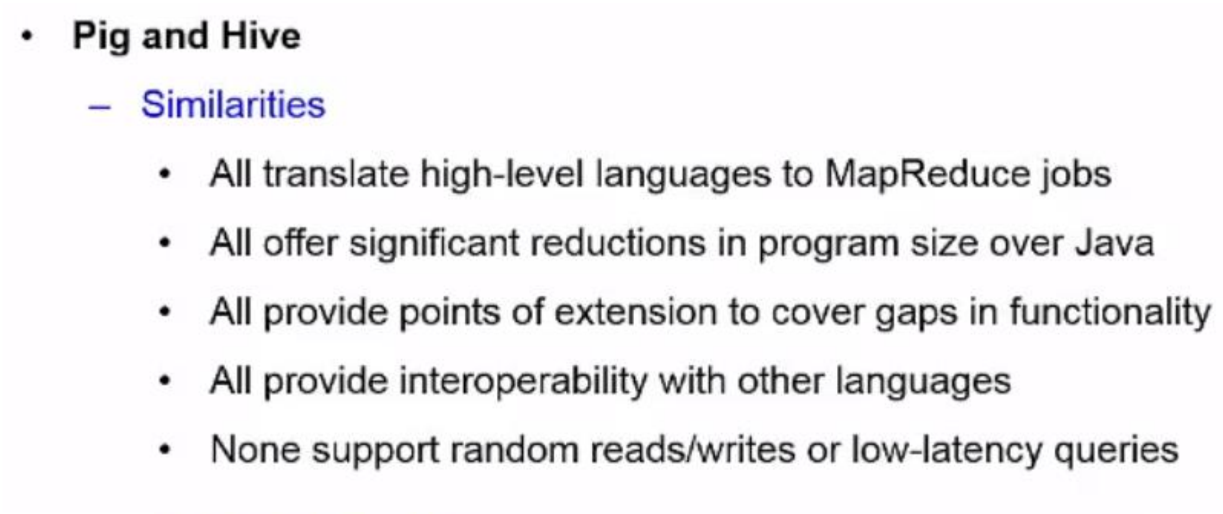
- Describe the MapReduce philosophy
- Describe the usage of Pig and Hive in a Hadoop environment
- Moving data into Hadoop using Flume and Sqoop
- Scheduling and controlling Hadoop job execution using Oozie



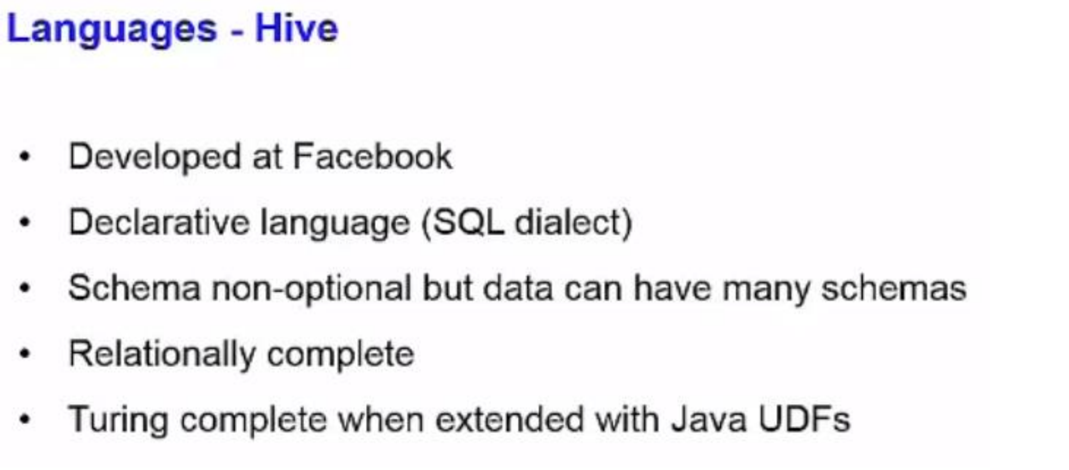
MapReduce

- Processes huge datasets for certain kinds of distributable problems using a large number of nodes
- Map
  - Master node partitions the input into smaller sub-problems
  - Distributes the sub-problems to the worker nodes
- Reduce
  - Master node then takes the answers to all the sub-problems
  - Combines them in some way to get the output
- Allows for distributed processing of the map and reduce operations

HADOOP COMPONENTS

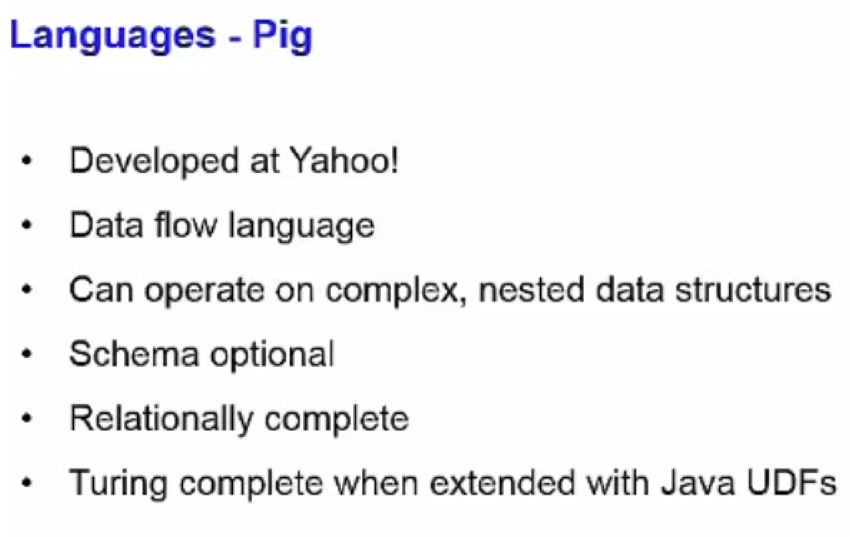


- **Pig and Hive**
  - Similarities
    - All translate high-level languages to MapReduce jobs
    - All offer significant reductions in program size over Java
    - All provide points of extension to cover gaps in functionality
    - All provide interoperability with other languages
    - None support random reads/writes or low-latency queries



Languages - Hive

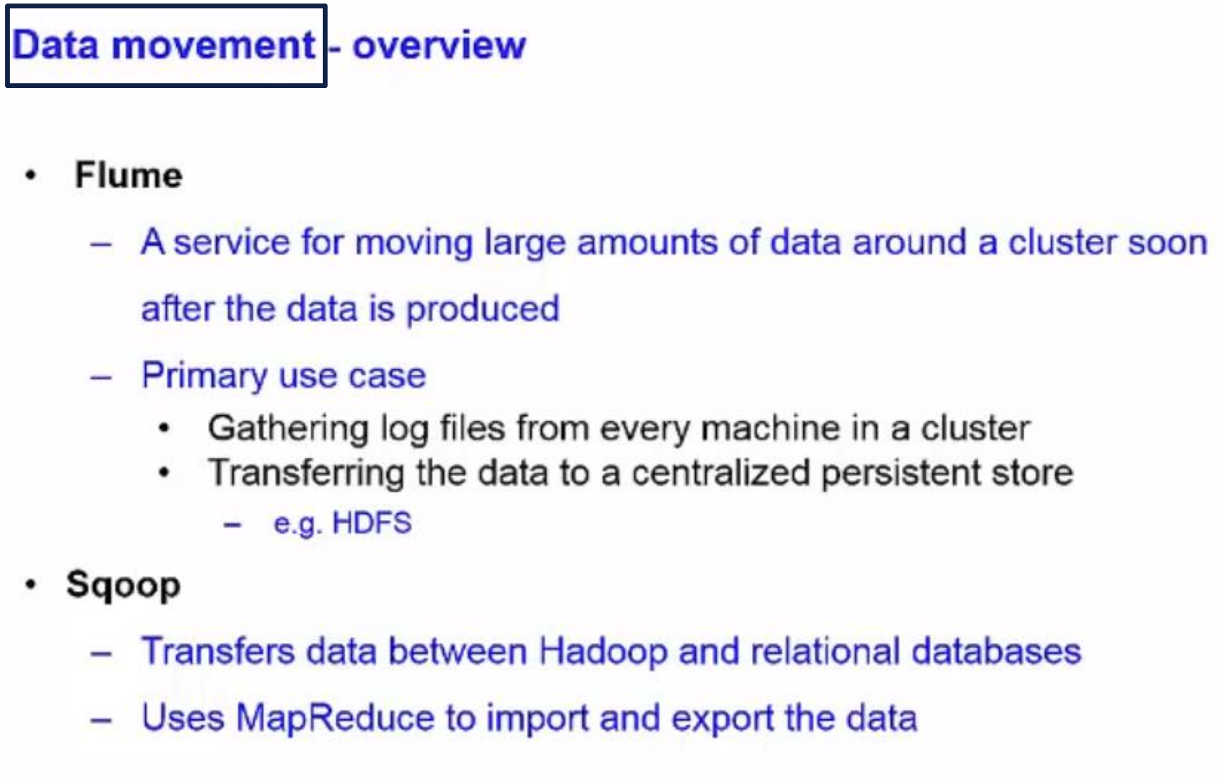
- Developed at Facebook
- Declarative language (SQL dialect)
- Schema non-optional but data can have many schemas
- Relationally complete
- Turing complete when extended with Java UDFs



Languages - Pig

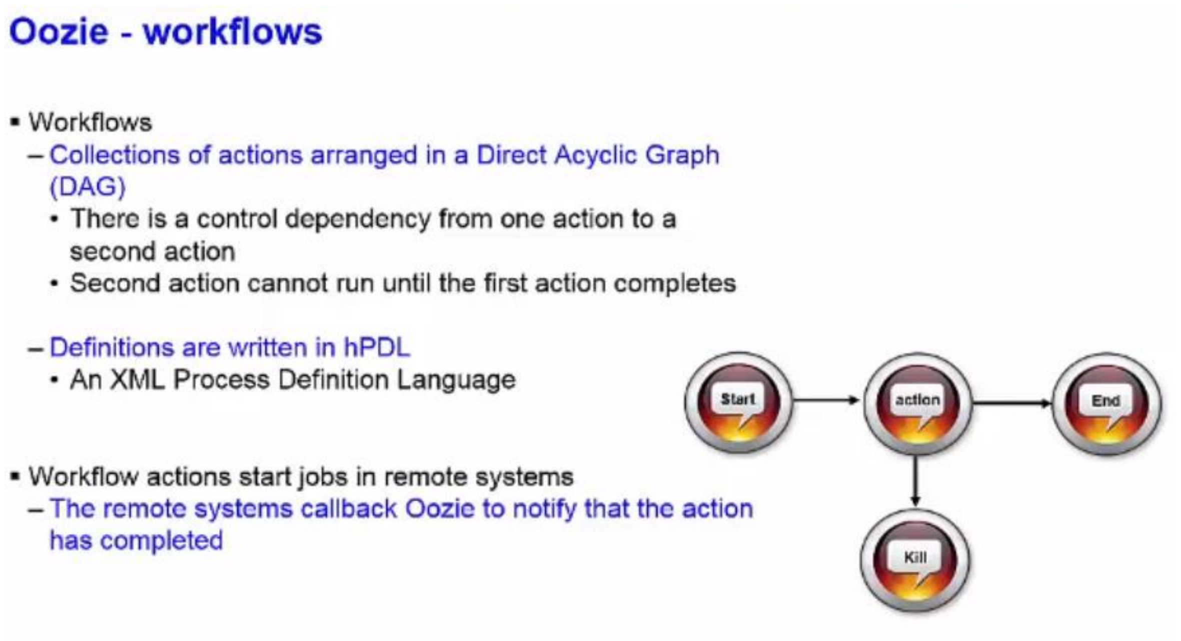
- Developed at Yahoo!
- Data flow language
- Can operate on complex, nested data structures
- Schema optional
- Relationally complete
- Turing complete when extended with Java UDFs

HADOOP COMPONENTS



Data movement - overview

- **Flume**
  - A service for moving large amounts of data around a cluster soon after the data is produced
  - Primary use case
    - Gathering log files from every machine in a cluster
    - Transferring the data to a centralized persistent store
      - e.g. HDFS
- **Sqoop**
  - Transfers data between Hadoop and relational databases
  - Uses MapReduce to import and export the data



Oozie - workflows

- Workflows
  - Collections of actions arranged in a Direct Acyclic Graph (DAG)
    - There is a control dependency from one action to a second action
    - Second action cannot run until the first action completes
  - Definitions are written in hPDL
    - An XML Process Definition Language
- Workflow actions start jobs in remote systems
  - The remote systems callback Oozie to notify that the action has completed

THANKS