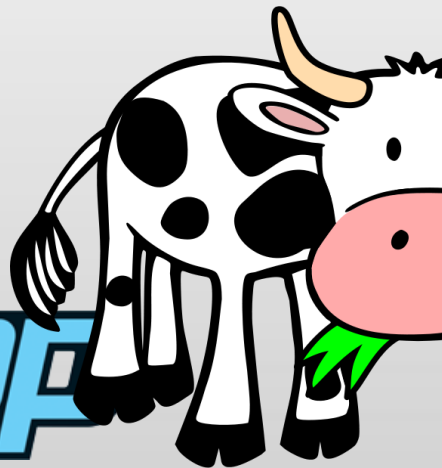
Hadoop, a distributed framework for Big Data

Move aside cows!
It's time for the BIG guys

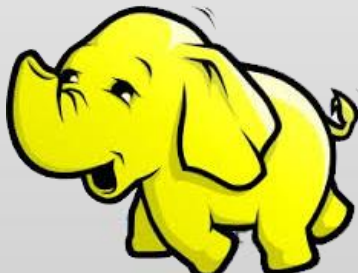
Slides and graphics borrow heavily from
Prof. Nalini Venkatasubramanian

http://www.ics.uci.edu/~cs237/



BIG Data, how big is BIG?

- Not about size, but how data is managed
- Relational databases was all about *organizing* data into tables
- Sometimes it is just too time consuming, or the data is just too big, to organize it in order to do simple queries
- Much data is unstructured or semi-structured and we'd like to process it in parallel
- Data warehouses

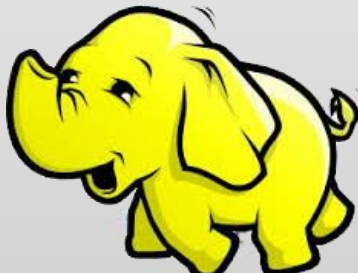


Introduction

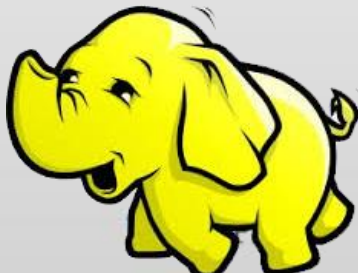
1. Introduction: Hadoop's history and advantages

2. Architecture in detail

3. Hadoop in industry

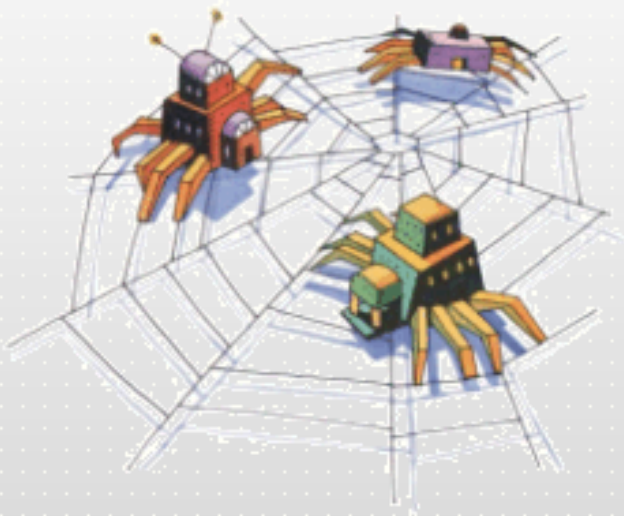


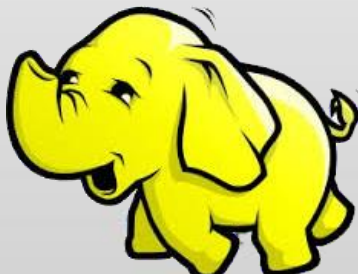
- Open-source implementation of a Map-Reduce framework for reliable, scalable, distributed computing and data storage.
- It is a flexible architecture for large scale computation and data processing on a network of commodity hardware.



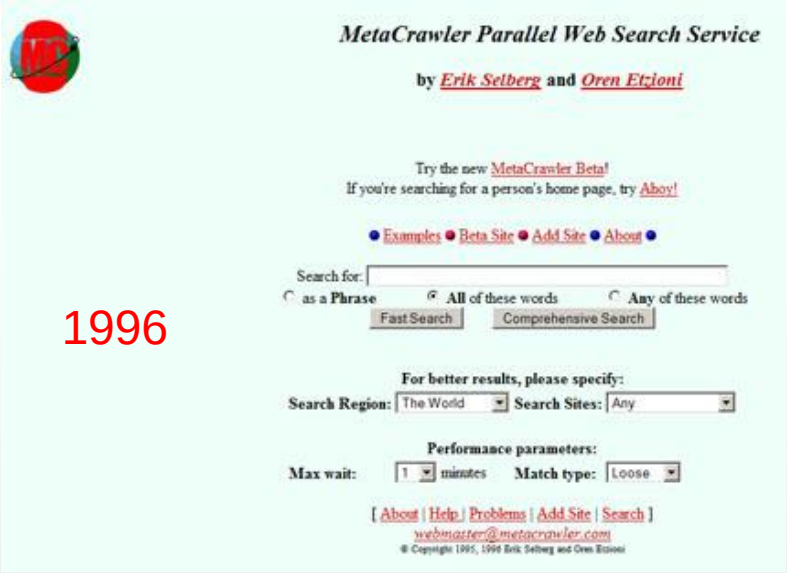
Brief History of Hadoop

- Designed to answer the question: **“How to process big data with reasonable cost and time?”**





Search engines in 1990s

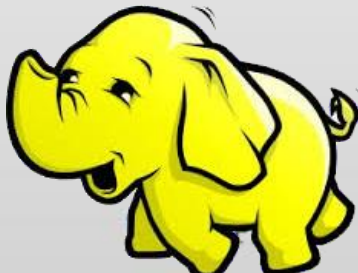


1996

1996

1996

1997



Google search engines

1998

2003

The Google File System

Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung

Google*

MapReduce: Simplified Data Processing on Large Clusters

2004

Jeffrey Dean and Sanjay Ghemawat

jeff@google.com, sanjay@google.com

Google, Inc.

Bigtable: A Distributed Storage System for Structured Data

Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach
Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber

{fay,jeff,sanjay,wilsonh,kerr,m3b,tushar,fikes,gruber}@google.com

Google, Inc.

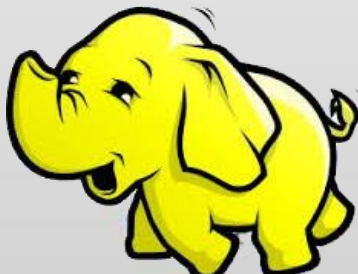
2006

Abstract

gtable is a distributed storage system for managing tured data that is designed to scale to a very large petabytes of data across thousands of commodity rs. Many projects at Google store data in Bigtable, ding web indexing, Google Earth, and Google Fi- e. These applications place very different demands igtable, both in terms of data size (from URLs to

achieved scalability and high performance, but Big provides a different interface than such systems. Big does not support a full relational data model; inste provides clients with a simple data model that sup dynamic control over data layout and format, an lows clients to reason about the locality properties o data represented in the underlying storage. Data i dexed using row and column names that can be arbi strings. Bigtable also treats data as uninterpreted str

2016



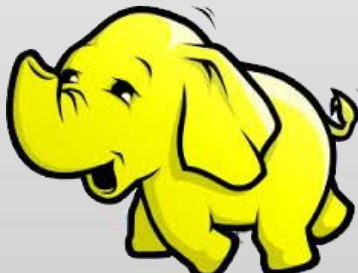
Hadoop's Developers

Doug Cutting

2005: Doug Cutting and Michael J. Cafarella developed Hadoop to support distribution for the Nutch search engine project.

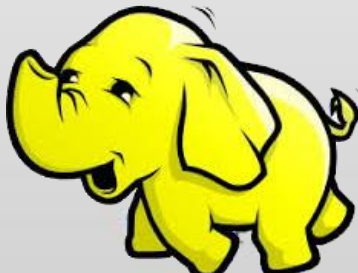
The project was funded by Yahoo.

2006: Yahoo gave the project to Apache Software Foundation.



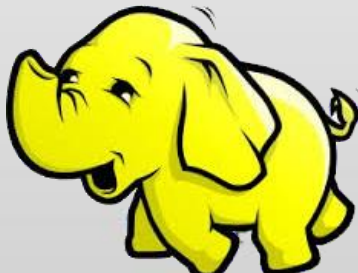
What is Hadoop?

- **Hadoop:**
  - An open-source software framework that supports data-intensive distributed applications, licensed under the Apache v2 license.
- **Goals / Requirements:**
  - Data and Processing abstractions facilitate queries of large, dynamic, and rapidly growing data sets
    - Structured and non-structured data
    - Simple programming models
  - High scalability and availability
  - Use commodity (cheap!) hardware with little redundancy
  - Fault-tolerance
  - Move computation rather than data



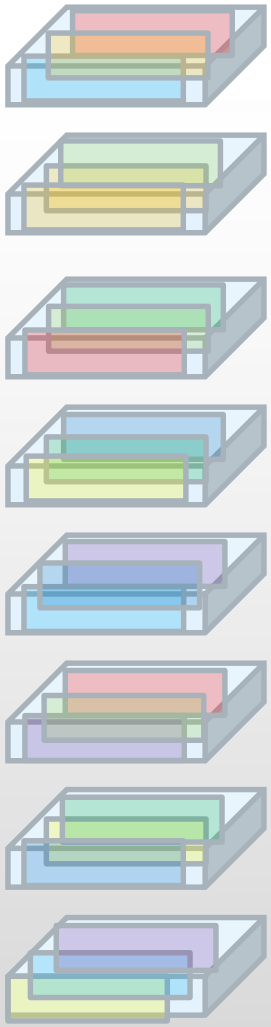
Hadoop's Architecture

- Distributed, with some *modest* centralization
- Main nodes of cluster are where most of the computational power and storage of the system lies
- Main nodes run TaskTracker to accept and reply to MapReduce tasks, and also DataNode to store needed blocks closely as possible
- Central control node runs NameNode to keep track of HDFS directories & files, and JobTracker to dispatch compute tasks to TaskTracker
- Written in Java, also supports Python and Ruby



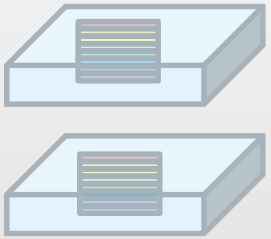
Hadoop's Data Model

1. Given giant files
2. Chops them up into good-sized chunks (64Mb)
3. Replicate and Distribute them

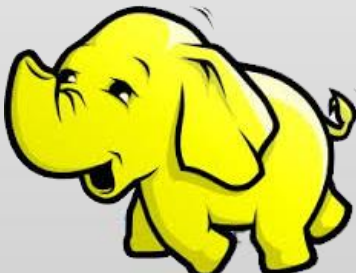


Each chunk is replicated 3 times, and placed on a different processing node

A name sever (actually 2) keeps track of where the chunks are



Hadoop's Distributed File System



Hadoop's Processing Model

Whenever we query the dataset,
Its done in the following stages:

Map:

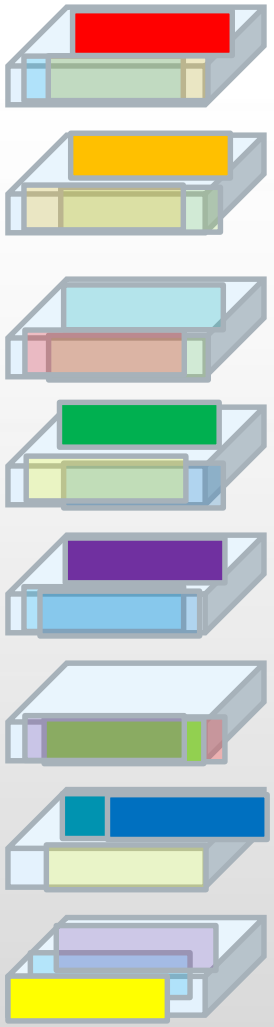
1. A processor is assigned to each chunk.
2. That processor scans, filters, and maps each data item into key-value pairs.
3. Keys are locally binned

Shuffle:

4. Bins with common keys are consolidated by broadcasting them to a common node

Reduce:

5. Final processing is done of within each bin, often agglomerative-like operations

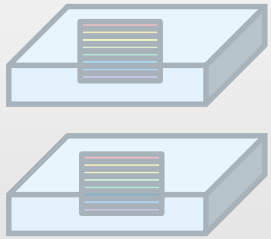


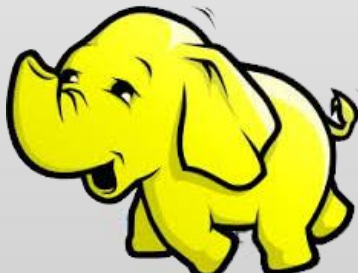
MapReduce

Distributed processing

Generally balanced, but no guarantees

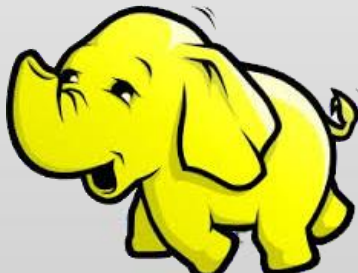
Processing occurs at the data source



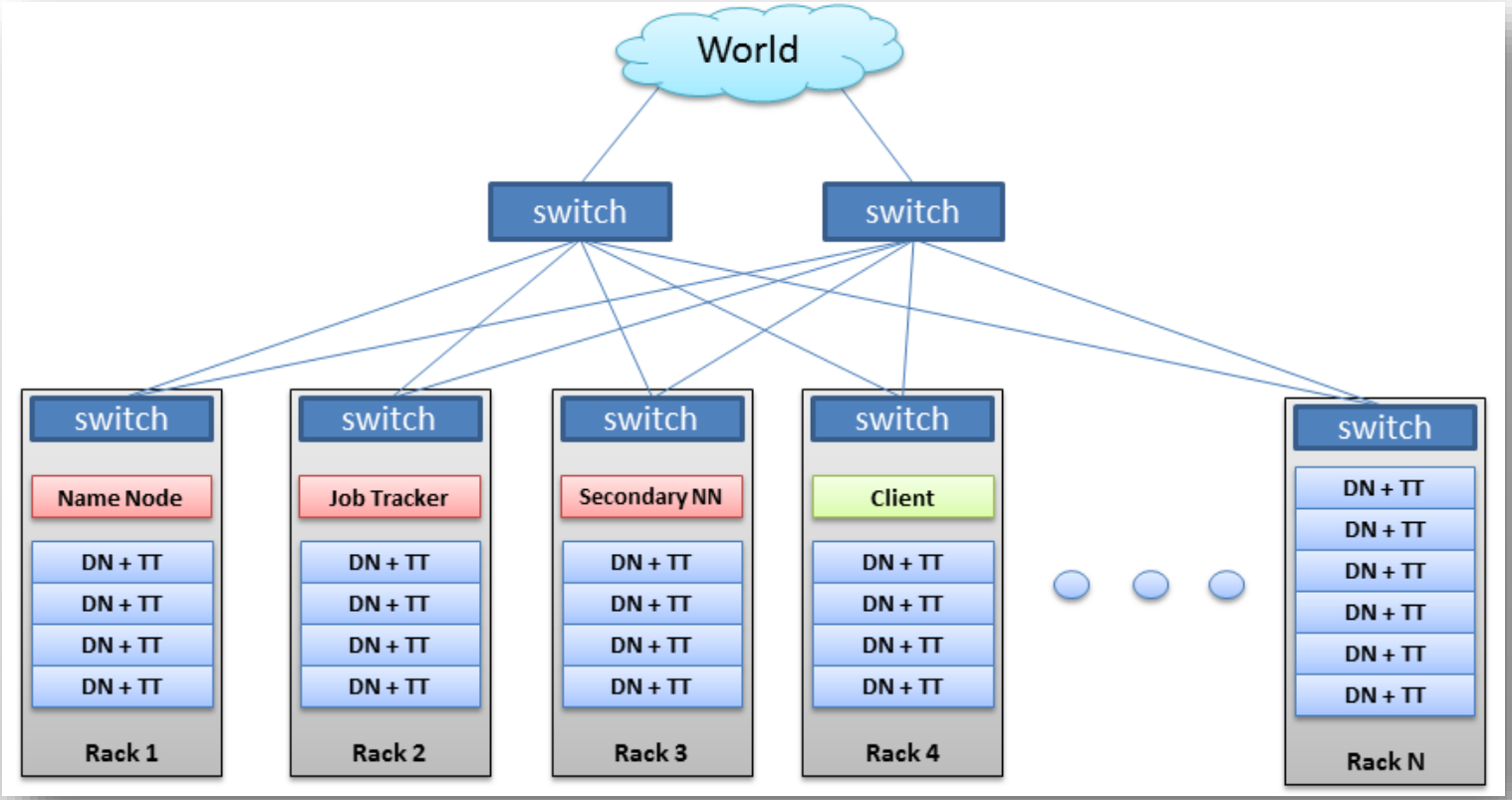


Hadoop's Architecture

- Hadoop Distributed FileSystem (Chops up and distributes data)
- Tailored to needs of MapReduce
- Targeted towards many reads of file streams
- ***Writes*** are more costly
- High degree of data replication (3x by default)
- No need for RAID on normal nodes
- Large blocksize (64MB, bigger than database pages)
- Location awareness of DataNodes in network

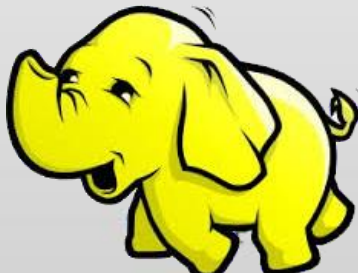


Hadoop's Reality



Also need to keep track of:

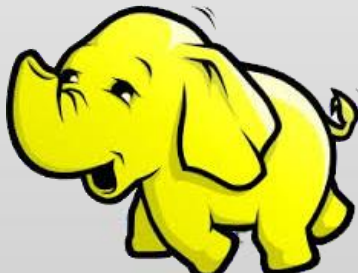
1. Where the data chunks are
2. What the state of multiple MapReduce jobs are in
3. Redundancy in case there are either H/W or network issues



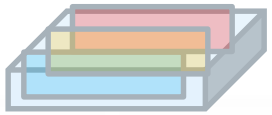
Hadoop's Architecture

NameNode:

- Stores metadata for the files, like the directory structure of a typical FS.
- The server holding the NameNode instance is quite crucial, so we keep a replicate.
- Transaction log for file deletes/adds, etc. Does not use transactions for whole blocks or file-streams, only metadata.
- Handles creation of more replica blocks when necessary after a DataNode failure

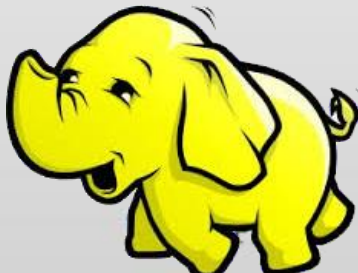


Hadoop's Architecture

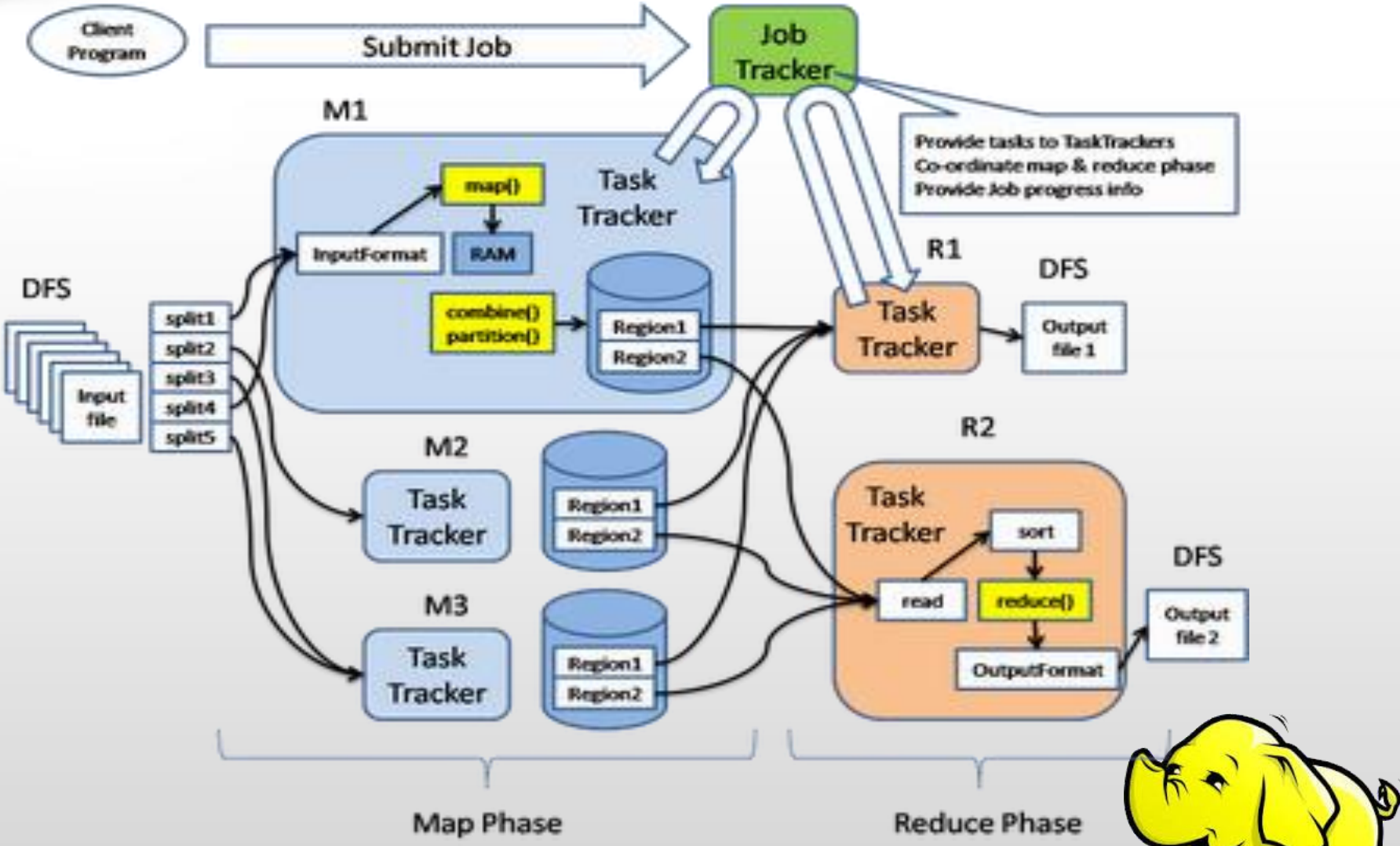


DataNode:

- Stores the actual data in HDFS
- Can run on any underlying filesystem (ext3/4, NTFS, etc)
- NameNode decides and tracks which blocks it has
- NameNode replicates blocks 3x
- Don't need to Homogenous
  - Different levels of performance
  - Different operating systems

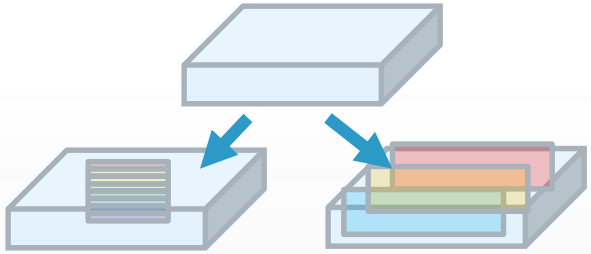


Job-Tracker has a key role in the MapReduce Engine

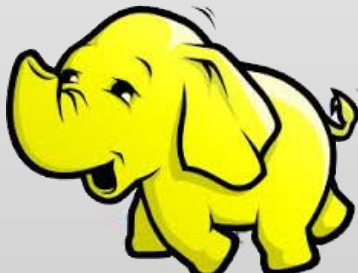


Hadoop's Architecture

MapReduce Engine:



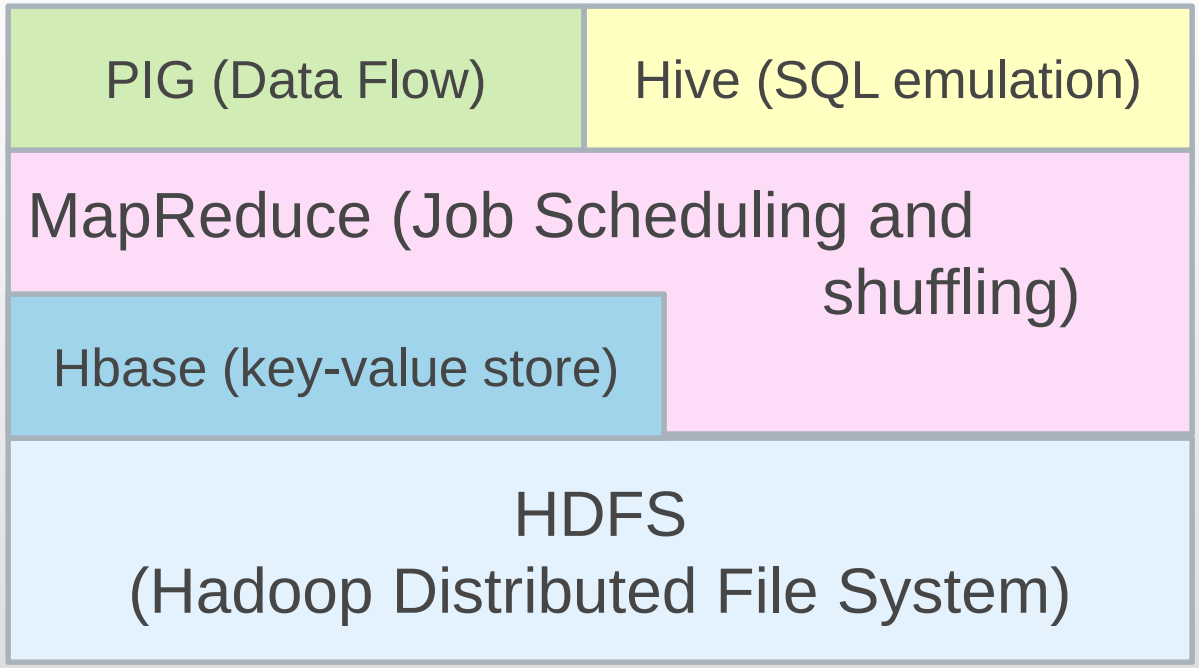
- JobTracker & TaskTracker
- JobTracker splits up data into smaller tasks("Map") and sends it to the TaskTracker process in each node
- TaskTracker reports back to the JobTracker node and reports on job progress, sends data ("Reduce") or requests new jobs
- You can have multiple of these, but only one is responsible for a given query

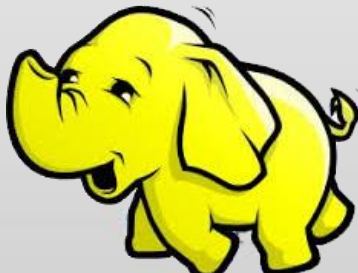


Hadoop Layer Cake

Most interaction with Hadoop is mediated by job managers using high-level APIs

1. PIG, a scripting language, with FOREACH, GROUP, FILTER, and ORDER constructs
2. Hive, SQL syntax, declarative specification



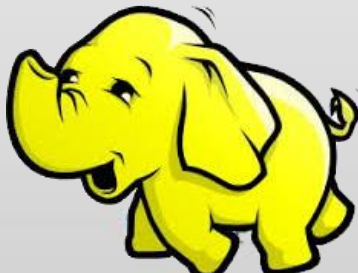


Hadoop in the Wild

- Hadoop is in use at most organizations that handle big data:
  - Yahoo!
  - Facebook
  - Amazon
  - Netflix
  - Etc…

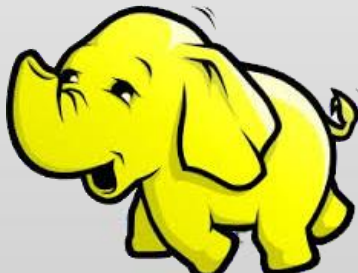
- Some examples of scale:
  - Yahoo!'s Search Webmap runs on 10,000 core Linux cluster and powers Yahoo! Web search

  - FB's Hadoop cluster hosts 100+ PB of data (July, 2012) & growing at ½ PB/day (Nov, 2012)



Hadoop in the Wild

- System requirements
  - High write throughput
  - Cheap, elastic storage
  - Low latency
  - High consistency (within a single data center good enough)
  - Disk-efficient sequential and random read performance



Hadoop in the Wild

- Facebook's solution
  - Hadoop + HBase as foundations
  - Improve & adapt HDFS and HBase to scale to FB's workload and operational considerations
    - Major concern was availability: NameNode is SPOF & failover times are at least 20 minutes
    - Proprietary "AvatarNode": eliminates SPOF, makes HDFS safe to deploy even with 24/7 uptime requirement
    - Performance improvements for realtime workload: RPC timeout. Rather fail fast and try a different DataNode