Hadoop – An Overview

- Socrates CCDH

What is Big Data?

- **Volume** Not Gigabyte. Terabyte, Petabyte, Exabyte, Zettabyte
  - Due to handheld gadgets ,and HD format images and videos
  - In total data, 90% of them collected in last 2 years.
- **Velocity** Rate at which data is produced
  - Due to proliferation of technology, social media websites, texting, mms etc
- **Variety** Structured, Semi-structured, and un-structured data
  - Due to texting, mms, various audio and video formats, and file formats

Challenges with Big Data

- Storage
  - How can I store such large data?
  - What are the options available?
- Performance
  - How can I seek, retrieve, and work on the data?
  - What will be the performance?
- Network Bandwidth
  - How can I transport the data to my application?
  - How can I efficiently use the Corporate Network?
- Fault Tolerance
  - What if my database fails? Or What if I want to upgrade my DB?
  - Can I continue to serve my customers? In other words, is my system fault tolerant?

What is the Solution?

Hadoop

What is Hadoop?

- Not a single technology, but an Eco-system
- Not a RDBMS, but a File System
- Developed in Java – supports Ruby, Pearl, and Python scripts. Supports C, C++
- Can be integrated with NoSQL DB – Not Only SQL (Hbase, MongoDB, Cassandra, Neo4J, Riak)
- Not a silver-bullet solution – does not solve all the problems in an Enterprise

Two Core Projects

1. HDFS – Hadoop Distributed File System
   - A scalable, fault-tolerant file system
   - Meant for storing of large files (small number of large files)

2. MR (MapReduce) Framework
   - A computational framework that works on top of HDFS

Architecture & Processes

- Master - Slave Architecture

Components	Master	Slave
HDFS	Name Node, Secondary Name Node	Data Node
Map Reduce	Job Tracker	Task Tracker

- Daemon Processes

1. **Name Node** 2. **Secondary Name Node** 3. **Job Tracker**	**One process each**
4. **Data Node** 5. **Task Tracker**	**Many processes**

Hadoop Infrastructure - PROD Cluster

Name Node

Sec Name Node

Job Tracker

Network Switch

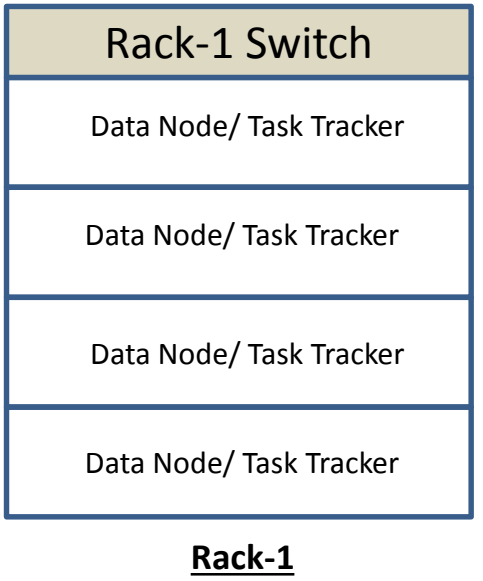
MASTER

SLAVE

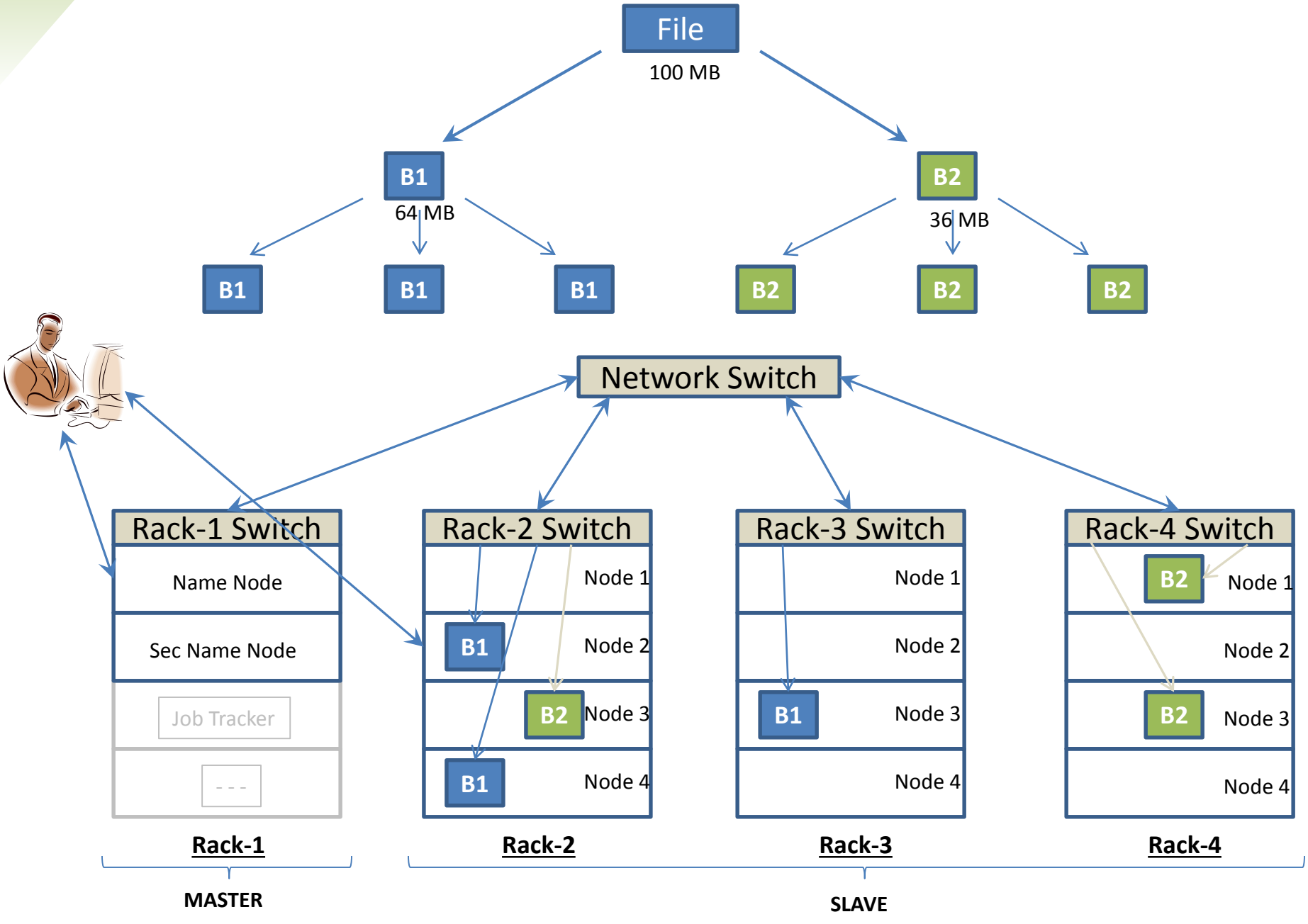
Rack-1 Switch	Rack-2 Switch	Rack-3 Switch
Data Node/ Task Tracker	Data Node/ Task Tracker	Data Node/ Task Tracker
Data Node/ Task Tracker	Data Node/ Task Tracker	Data Node/ Task Tracker
Data Node/ Task Tracker	Data Node/ Task Tracker	Data Node/ Task Tracker
Data Node/ Task Tracker	Data Node/ Task Tracker	Data Node/ Task Tracker
Rack-1	**Rack-2**	**Rack-3**

Hadoop Infrastructure

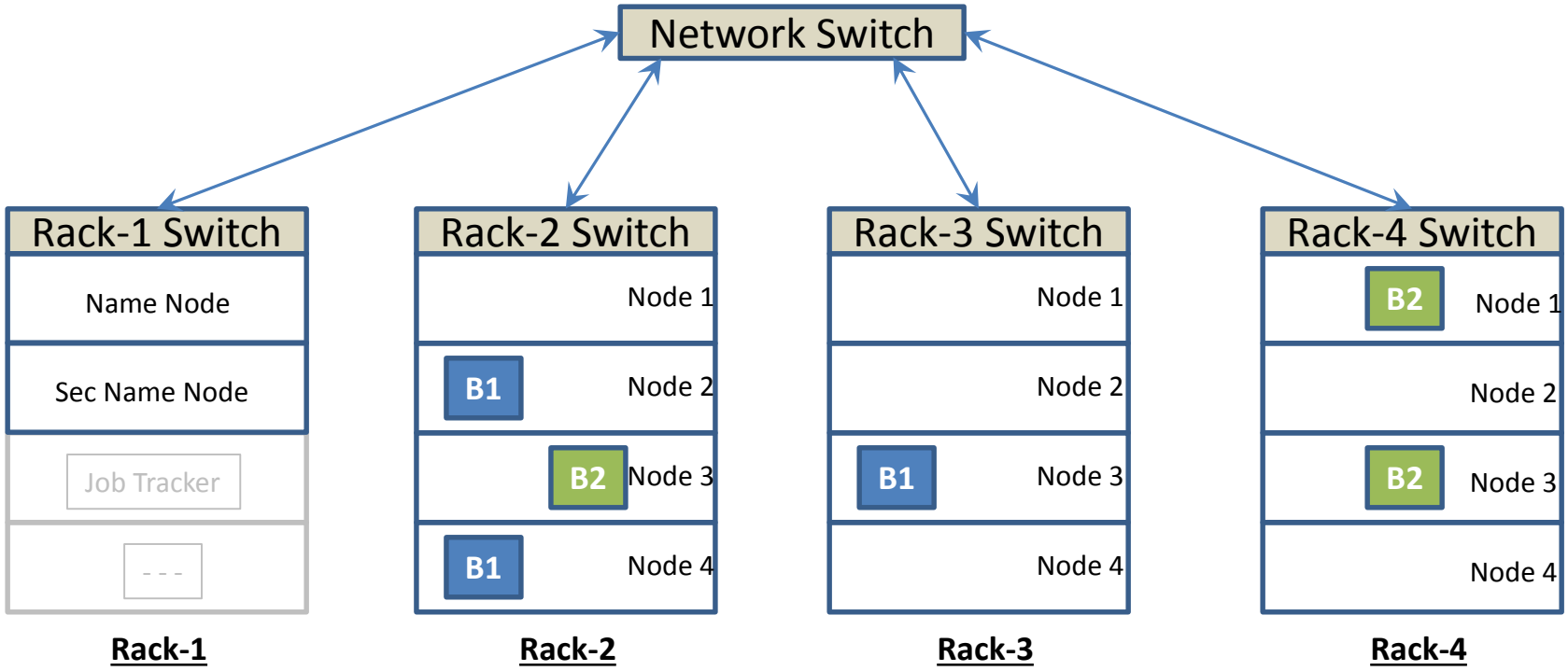
- Rack Servers as opposed to Blade Servers
- Commodity hardware
- Graceful addition or removal of nodes
- Fault Tolerant
- Can run on any OS
  (Unix, Mac, Windows)



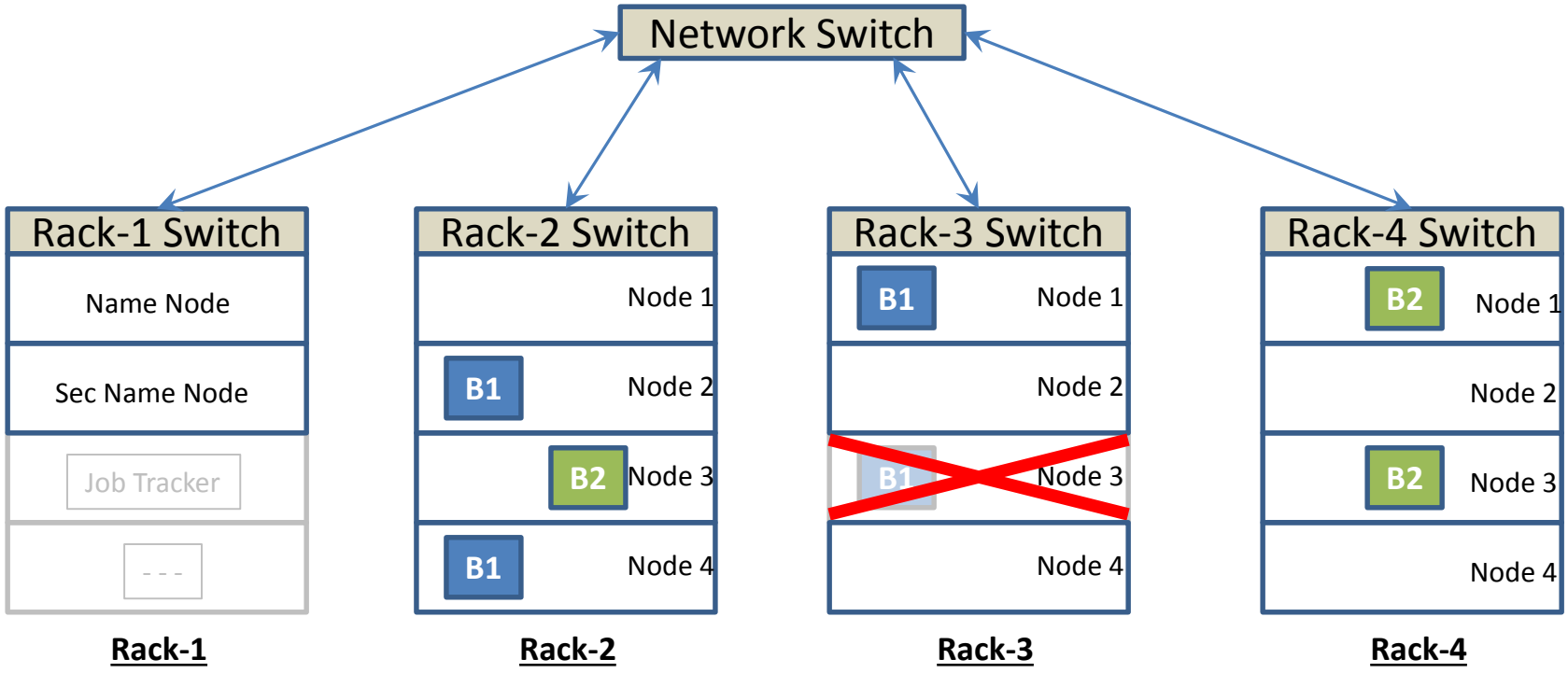
HDFS – File Copy



HDFS – Fault Tolerance



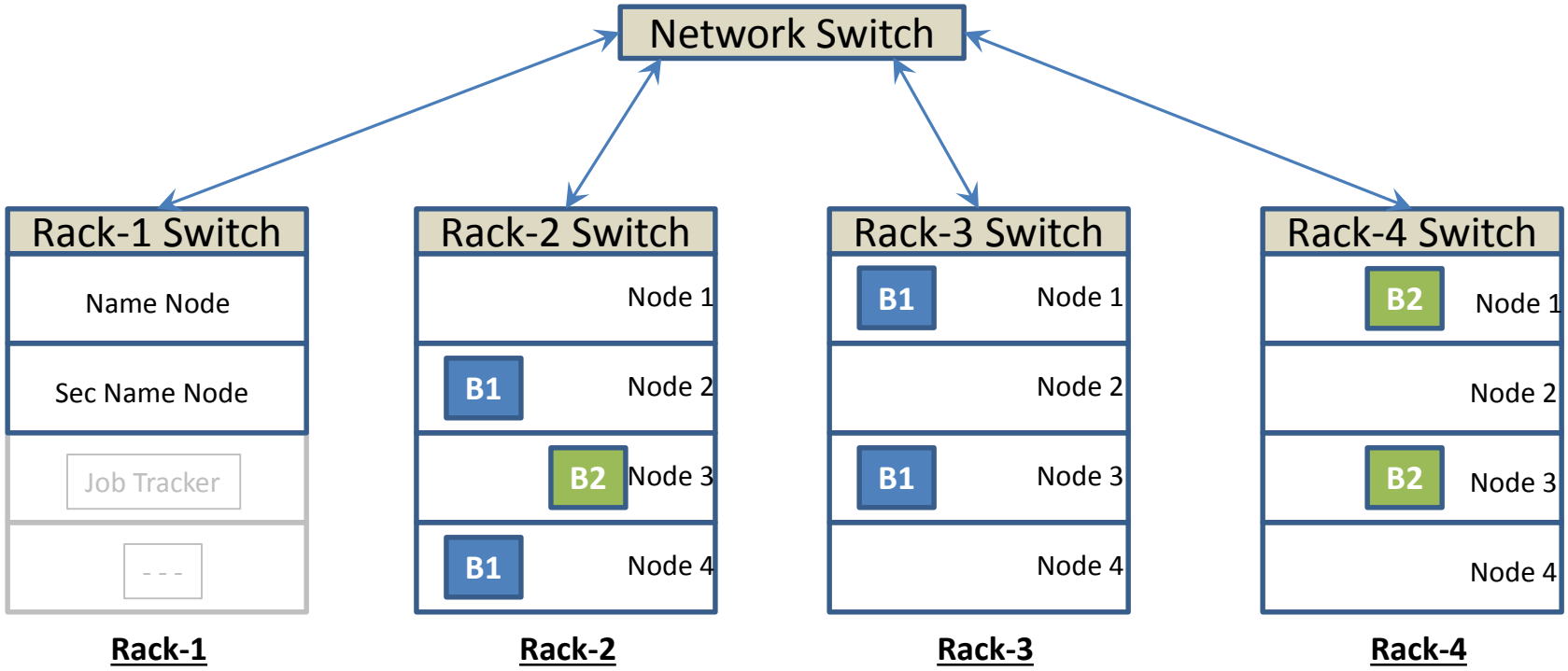
HDFS – Fault Tolerance



1. Node 3 in Rack-3 goes down

2. Block B1 gets copied onto Node 1 in Rack-3 to maintain replication factor

HDFS – Fault Tolerance



1. Node 3 in Rack-3 goes down
2. Block B1 gets copied onto Node 1 in Rack-3 to maintain replication factor
3. **Node 3 in Rack-3 comes-up after some time**
4. **Block B1 from Node 3 in Rack-3 gets deleted to maintain replication factor**

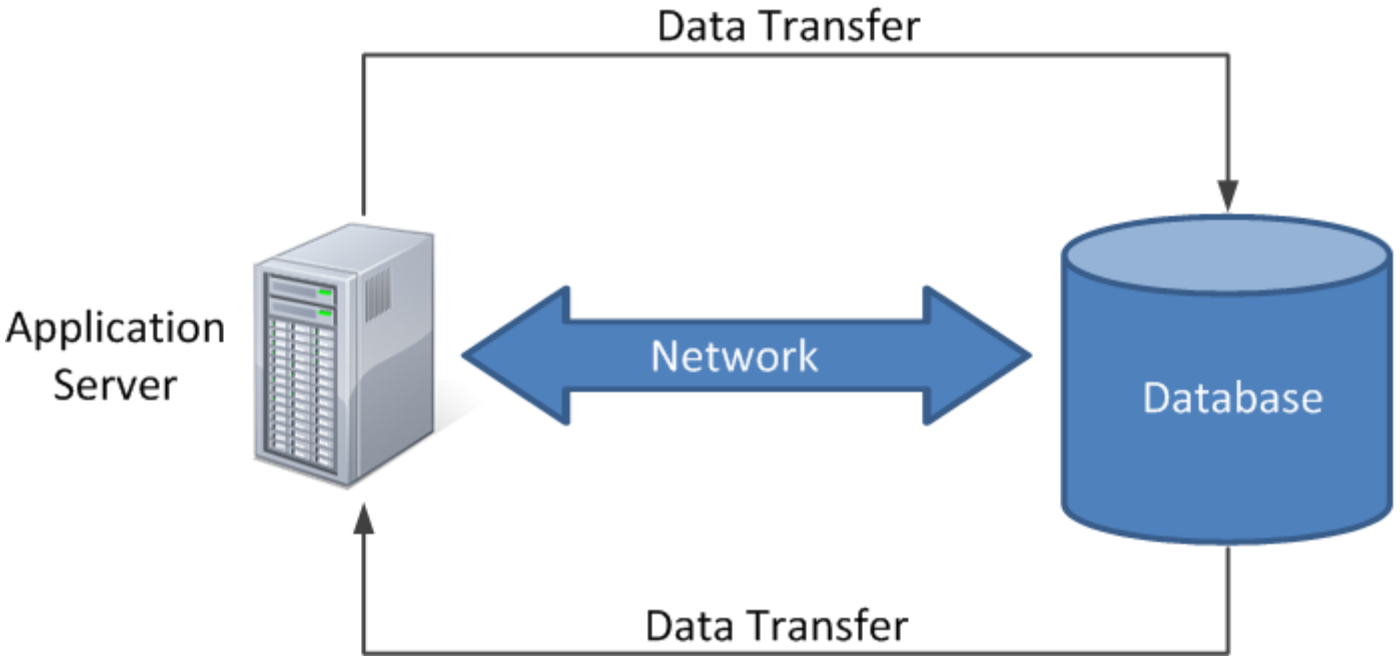
HDFS Summary

- Default block size: 64 MB
- Default replication factor: 3
- HDFS storage capacity: (Cluster Capacity/ Repl factor)
- Name Node is a Single Point of Failure (SPOF) in HDFS
- Name Node maintains metadata. They are, list of
  1. blocks in each data node – Helps replicate the blocks in the event of Data Node failure
  2. files and directories in HDFS. `hadoop fs –ls`
  3. # of blocks and its location for a given file (`foo.txt-B1,B2; faa.txt-B3,B4,B5`)
  4. operations carried out in HDFS
- Secondary Name Node:
  1. NOT a backup name node
  2. Copies namespace image and log data from Name Node to permanent storage
  3. Helps bring-up the Name Node in case of failure

HDFS Summary – Contd..

- Data Node stores actual data
- Data Node sends periodical “Heartbeat signal” (block report, storage capacity) - say every few secs to Name Node
- Name Node contacts Data node as response to Heartbeat signal only
- Under-replication and Over-replication carried-out through Heartbeat signals.

Traditional Application

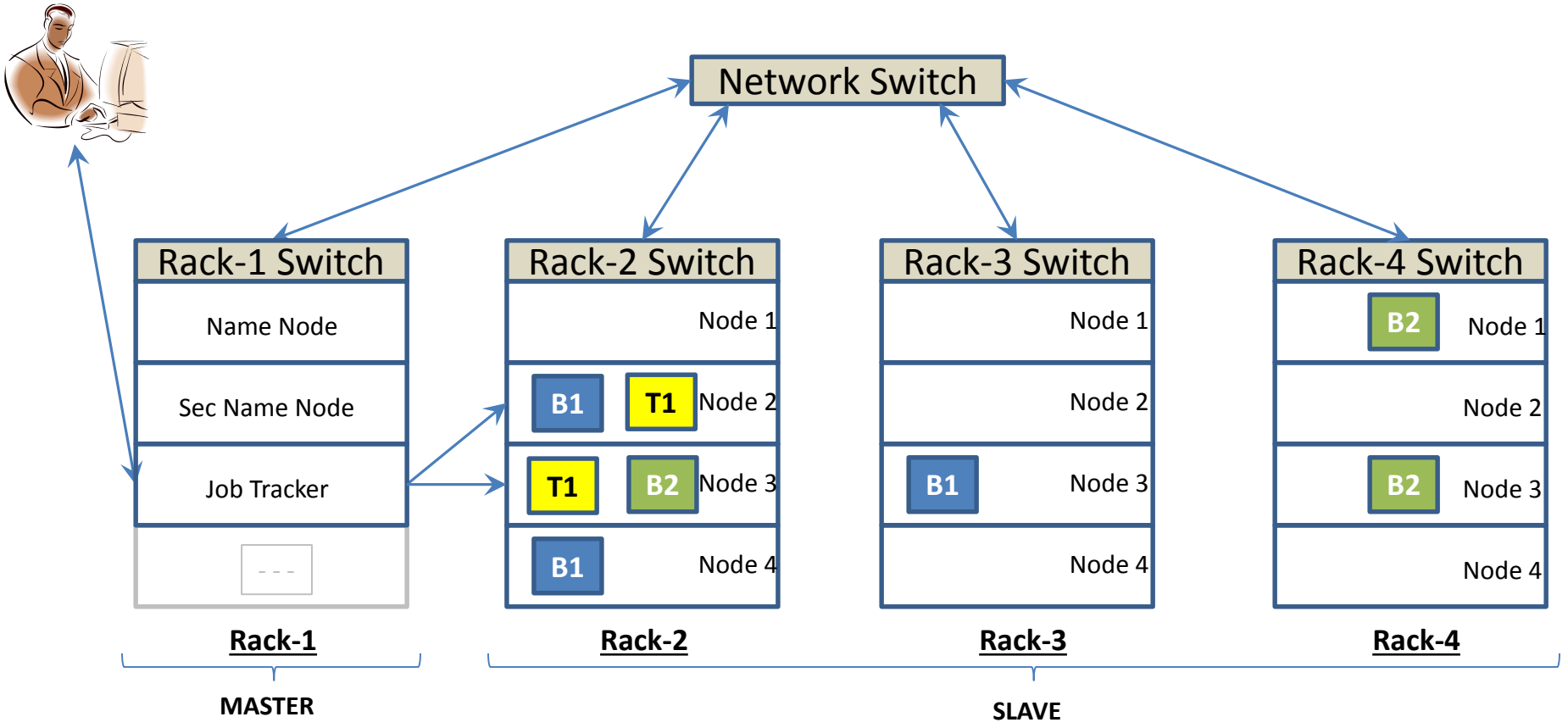


- Application size is constant, but data size varies
- Data gets transmitted through network where process/program resides
- Network capacity remains same, but “Volume” keeps increasing (remember ‘3-V’s ??)

Map Reduce

Move the code to where data resides – Data Localization

Components	Master	Slave
Map Reduce	Job Tracker	Task Tracker



Map Reduce

- Map Reduce consists of two phases
  - Map Phase
  - Reduce Phase
- Mapper accepts input as key/value pair and emits result as key/value pair
- Map function is called for one record per time.
- Reducer task starts after all Map tasks are done
- Reducer gets executed on the Mapper output
- Mapper & Reducer may output zero or more key/value pair
- Speculative Execution

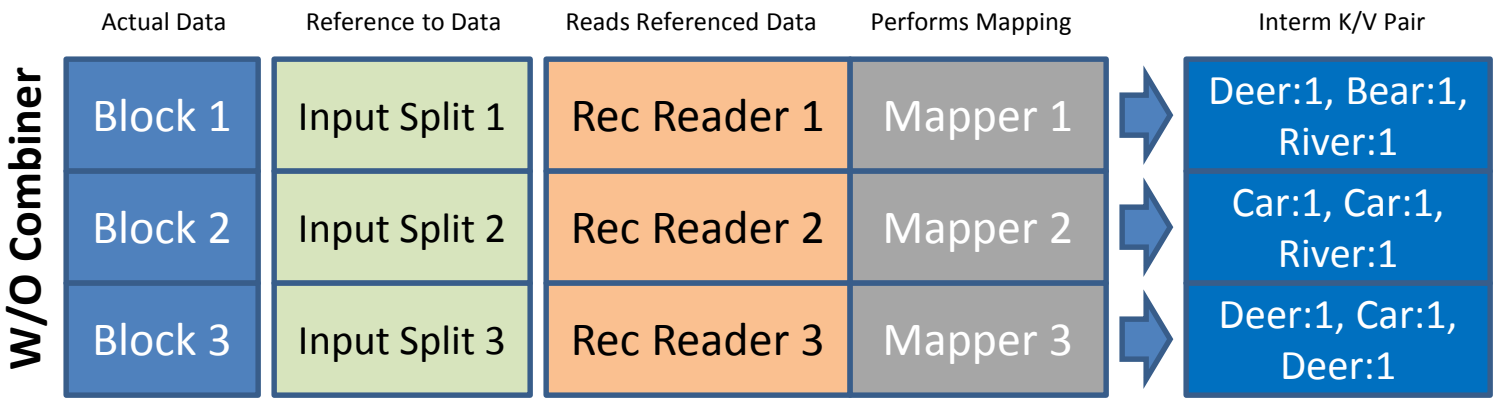
How Mapper Works?

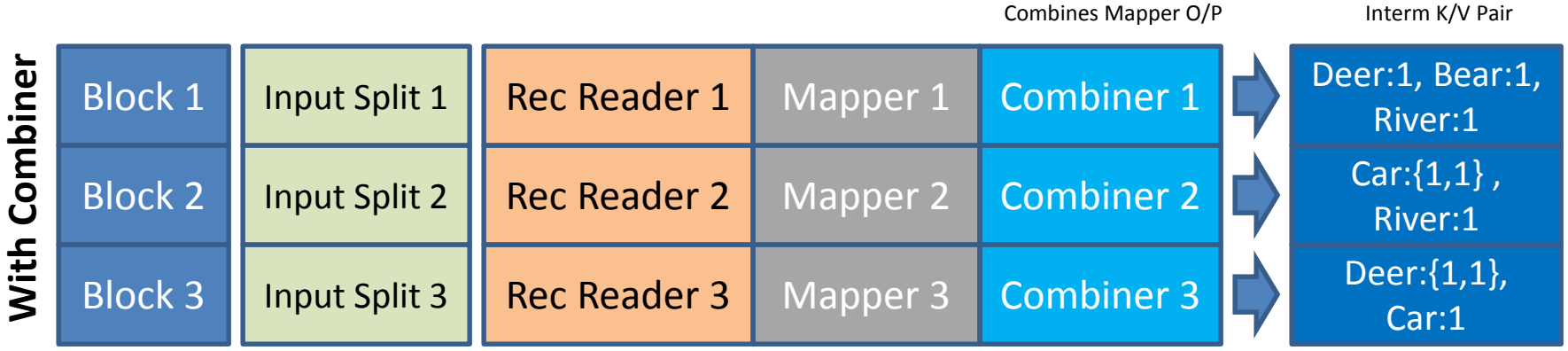
(Word count problem)

Input: Key-Value pair

Output: Zero, one, or more interm Key-Value pair

1	Deer Bear River
2	Car Car River
3	Deer Car Deer

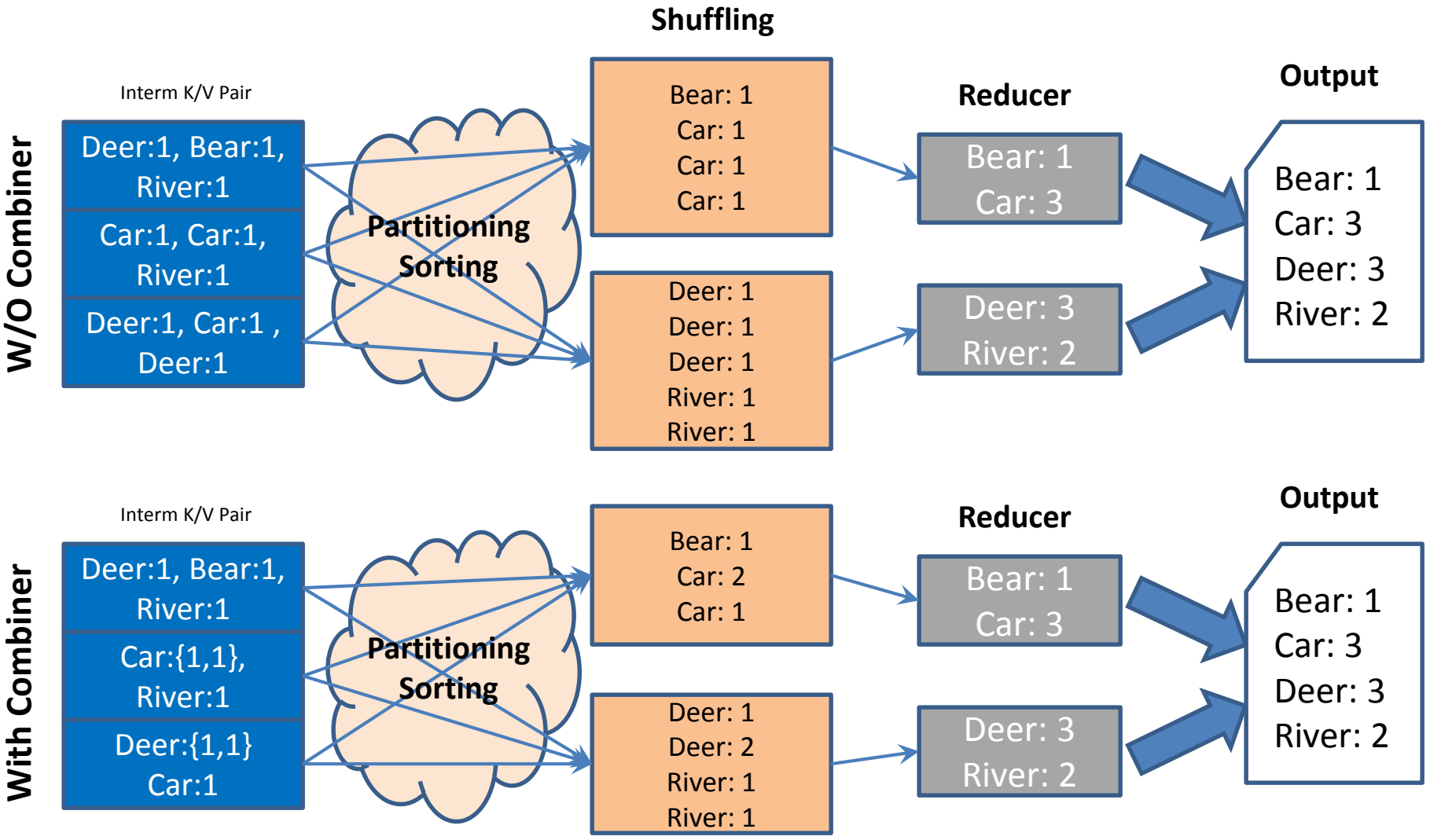




How Reducer Works?

Input: Interm Key-Value pair from Mapper

Output: One value per key



Map Reduce Summary

- Data Localization
- Speculative Execution
- Parallel Processing
- Batch Processing
- Sorted Output based on key
- Combiners and Reducers are optional

Challenges with Big Data

- Storage
  - How can I store such large data?
  - What are the options available?

1. Distributed storage
2. Add more machine w/o affecting the Data/ App
3. Increase hard disk capacity

- Performance
  - How can I seek, retrieve, and work
  - What will be the performance?

1. MapReduce: Perform parallel processing
2. Minimize seek rate by reading full block of data

- Network Bandwidth
  - How can I transport the data to my
  - How can I efficiently use the Corpora

1. Data localization: Ship code to the data (as opposed to traditional model)

- Fault Tolerance
  - What if my database fails? Or What if I w
  - Can I continue to serve my customers? Ir

1. Make redundant data copy
2. Speculative Execution

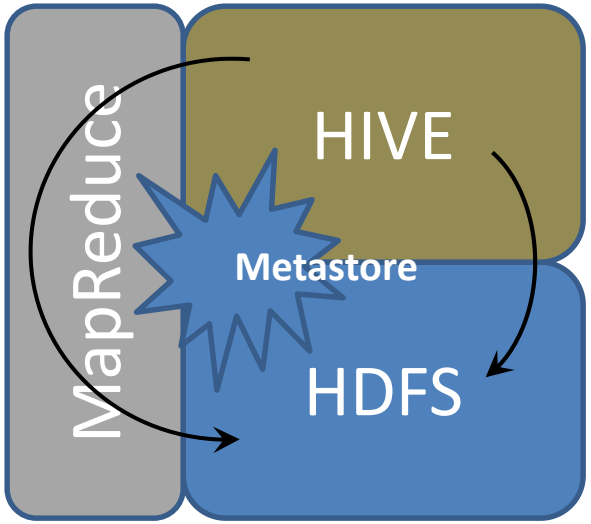
Hadoop Ecosystem

Hive – Developed by Facebook

- Provides SQL like interface
- Commands are converted into *MapReduce* jobs
- Can be used by users familiar with SQL
- Does not support all SQL capabilities (say *sub-query*)

```
SELECT * FROM employee;

SELECT * FROM employee
WHERE id > 100
ORDER BY ASC
LIMIT 20;
```



Hadoop Ecosystem

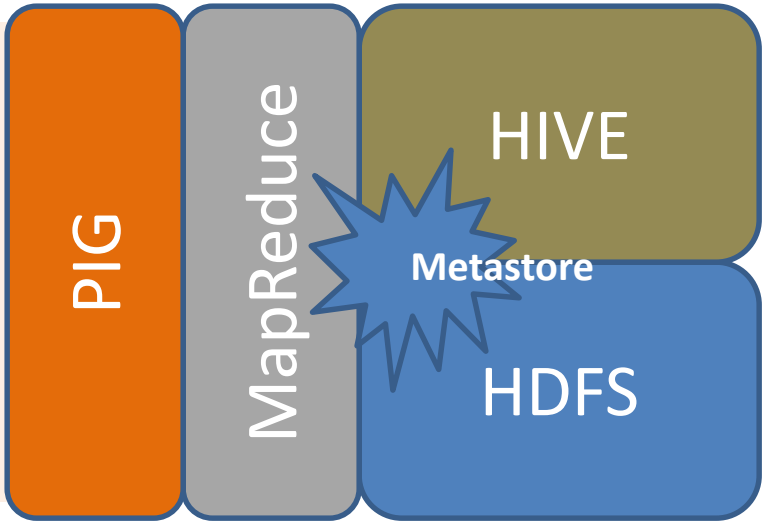
Pig – Developed by Yahoo*!*

- High level platform for MapReduce
- Uses *PigLatin* language
- Commands are converted into *MapReduce* jobs
- Does not require *Metastore* like *HIVE*

```
results = FILTER records BY
   project == 'en';

sorted_results = ORDER results
   BY $1 desc;

STORE sorted_results INTO
   'myResults';
```



Hadoop Ecosystem

Projects	Description
HBase	Scalable distributed DB for random read/write
Ambari	Web-based tool for managing and monitoring Hadoop clusters. Also provides Dashboard
Sqoop	*SQL-to-Hadoop*. Helps transferring data from RDBMS to HDFS
Flume	Distributed service for moving large data to HDFS. (say, server logs to HDFS)
Avro	Data serialization mechanism
Mahout	Scalable Machine learning, data mining library
YARN	Yet Another Resource Negotiator MapReduce (V2)
Zookeeper	High-performance co-ordination service

Hadoop Ecosystem

Projects	Description
Whirr	Java API that allows Hadoop to run on EC2 and other cloud services
G(i)raph	Iterative graph processing system currently used by FB for analyzing social graph formed by users
HUE	UI Framework and SDK for visual Hadoop apps
Impala	Interface that provides SQL Query execution. Implemented in C++, does not use MR, 10-100 times faster than Hive
OOZIE	Workflow engine that executes Pig, Hive, Sqoop, or MapReduce jobs. *Workflow* Engine & *Coordinator* Engine

Hadoop Ecosystem

Ingest / Propagate
Flume, Sqoop

Describe, Develop
Hive, Pig

Compute, Search
MapReduce, Giraph

Persist
File System: HDFS, MapR DFS
Serialization: Avro
DBMS: Cassandra, Riak, MongoDB, Hbase, Neo4J, CouchBase

Monitor, Admin
Ambari, Oozie, Zookeeper

Analytics, Machine Learning
Mahout

Who is using Hadoop?

- Netflix
- Yahoo*!*
- Facebook
- BFIS

Thank you!

Questions?