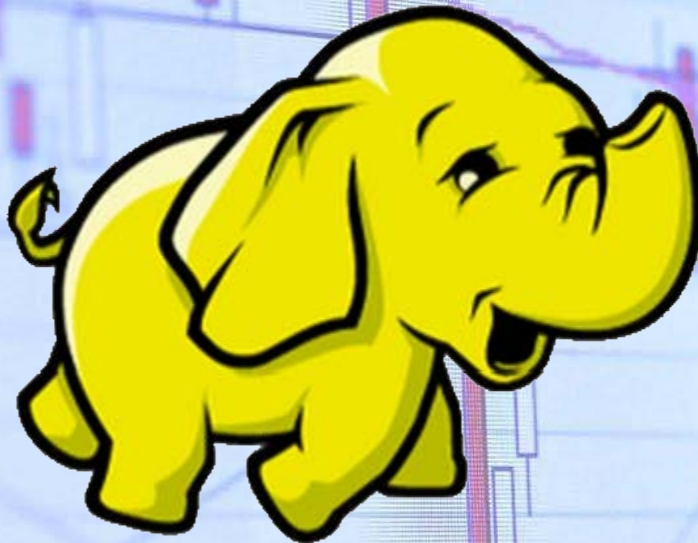


# Hadoop

Andy Catlin



“Hadoop is the standard tool for **distributed computing** across really large data sets... It has become an operating system for Big Data, providing a rich ecosystem of tools and techniques that allow you to use a large cluster of relatively cheap commodity hardware to do computing at supercomputer scale.”

source: “Getting Started with Spark (in Python),” Benjamin Bengfort,  
<https://districtdatalabs.silvrback.com/getting-started-with-spark-in-python>

“Any **distributed computing framework** needs to solve two problems: how to **distribute data** and how to **distribute computation**.”

source: “Getting Started with Spark (in Python),” Benjamin Bengfort,  
<https://districtdatalabs.silvrback.com/getting-started-with-spark-in-python>

# Distributed Computing

1. **computing frameworks**
2. distributed data
3. distributed processing

# Computing Frameworks

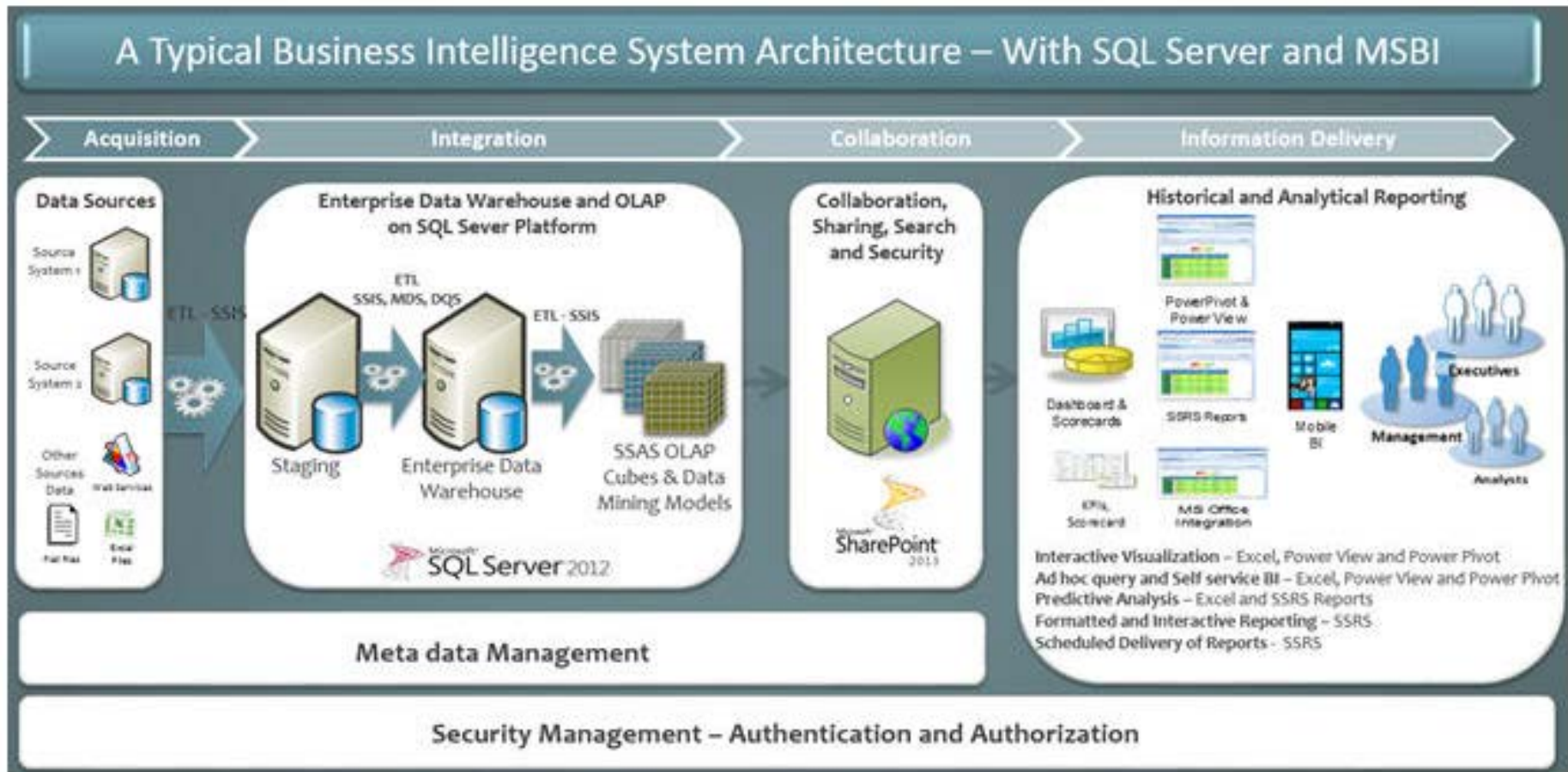
IF

“Hadoop is the standard tool for **distributed computing** across really large data sets... It has become an operating system for Big Data, providing a rich ecosystem of tools and techniques”

THEN

- What is the standard tool for computing across data sets that are *not* really large? –and–
- What is this tool’s ecosystem of tools and techniques?

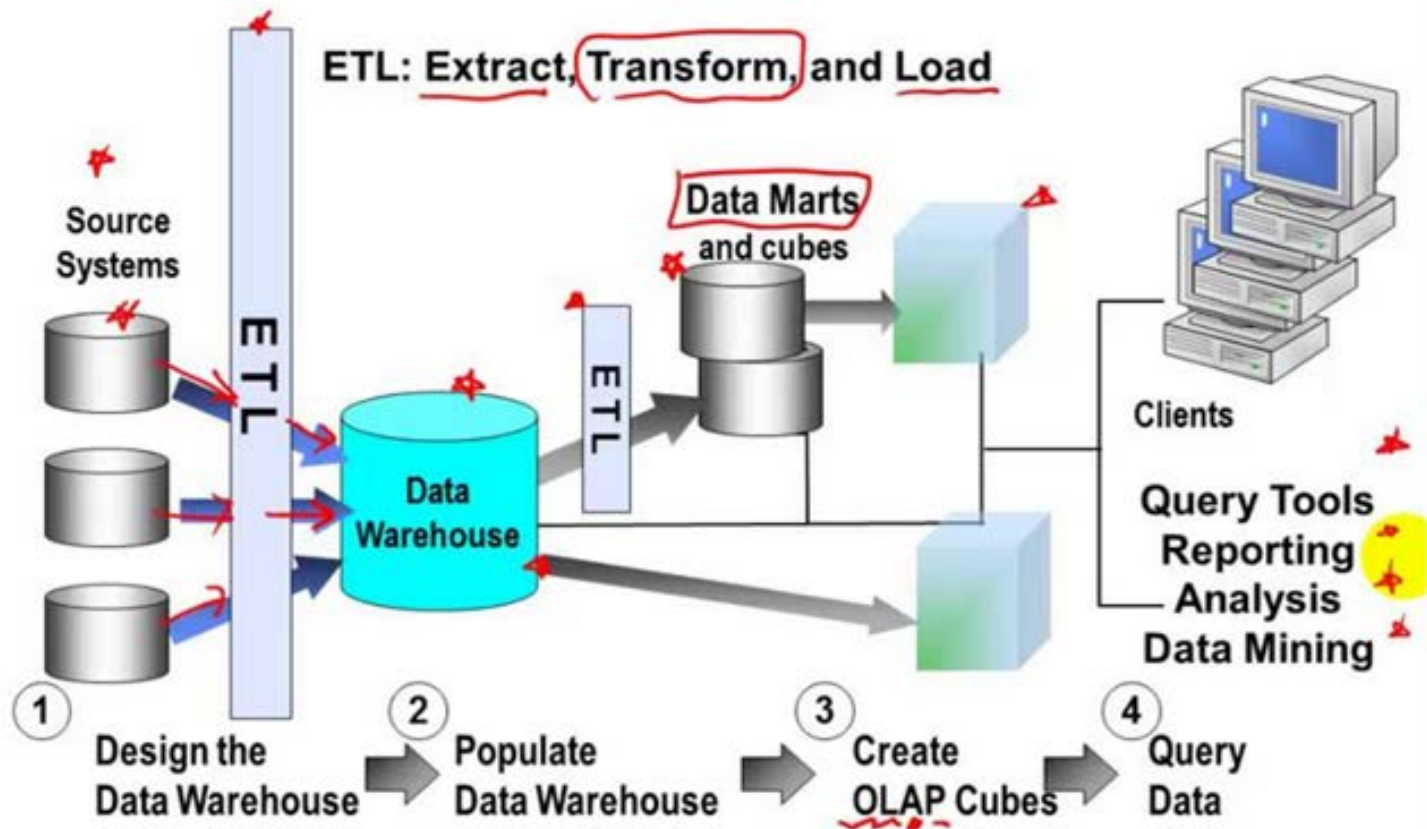
# SQL Data Framework: Ecosystem



Microsoft (shown here) and Oracle both have proprietary SQL framework ecosystems.  
Image Source: <http://www.mssqltips.com/sqlservertip/2970/microsoft-sql-server-business-intelligence-system-architecture--part-2/>



# SQL Data Framework: Workflow

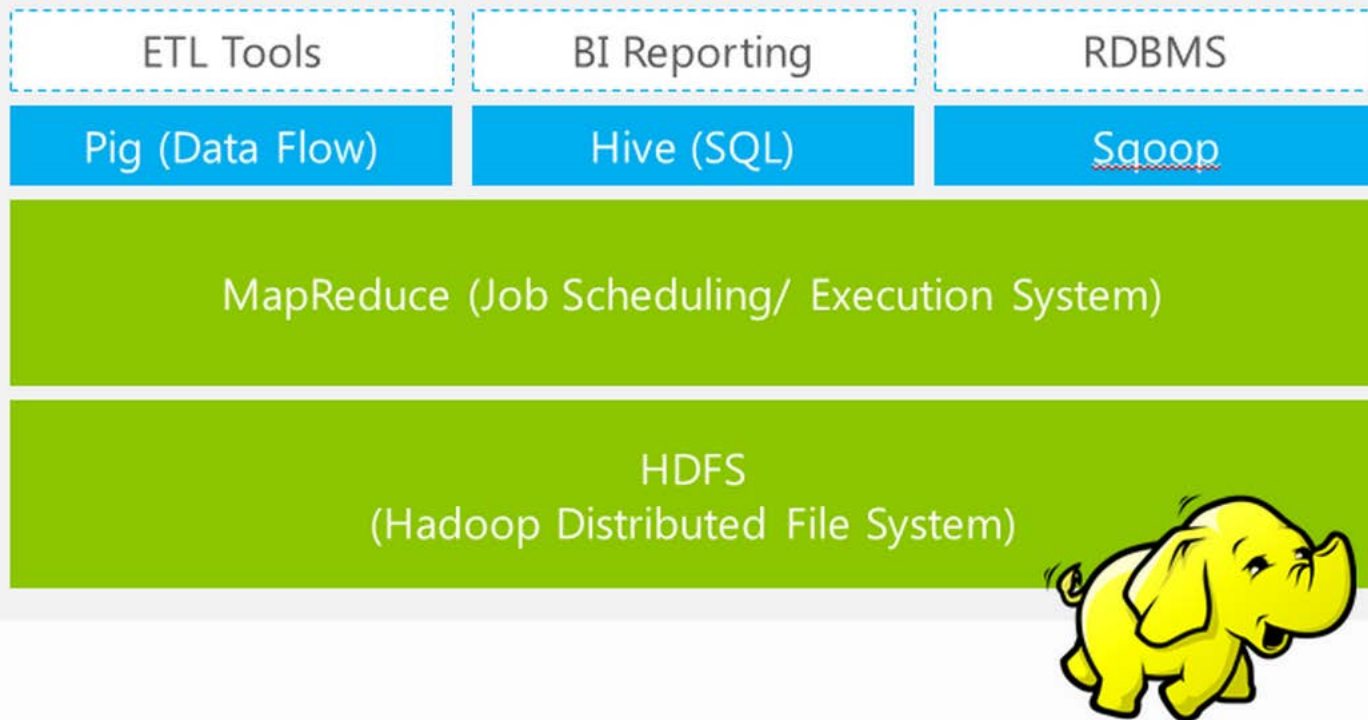


© Minder Chen. 2004-2014

DW & BI - 13

Source: "Data Warehouse and Business Intelligence: Systems Architecture and OLTP vs. OLAP," minderchen, <https://www.youtube.com/watch?v=Dff0Eb9fI>

# The Hadoop Ecosystem



*Think of MapReduce as like a SQL GROUP BY*

Very simplified and “Hadoop 1-centric” view. Image source:

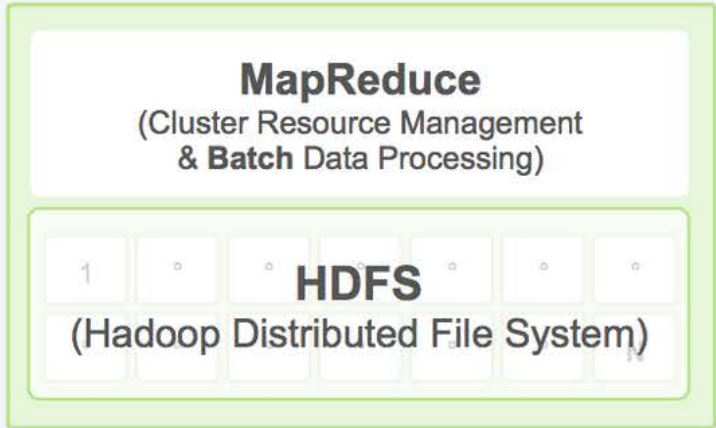
<http://social.technet.microsoft.com/wiki/contents/articles/13820.introduction-to-azure-hdinsight.aspx>

See also: <https://hadooecosystemtable.github.io/>



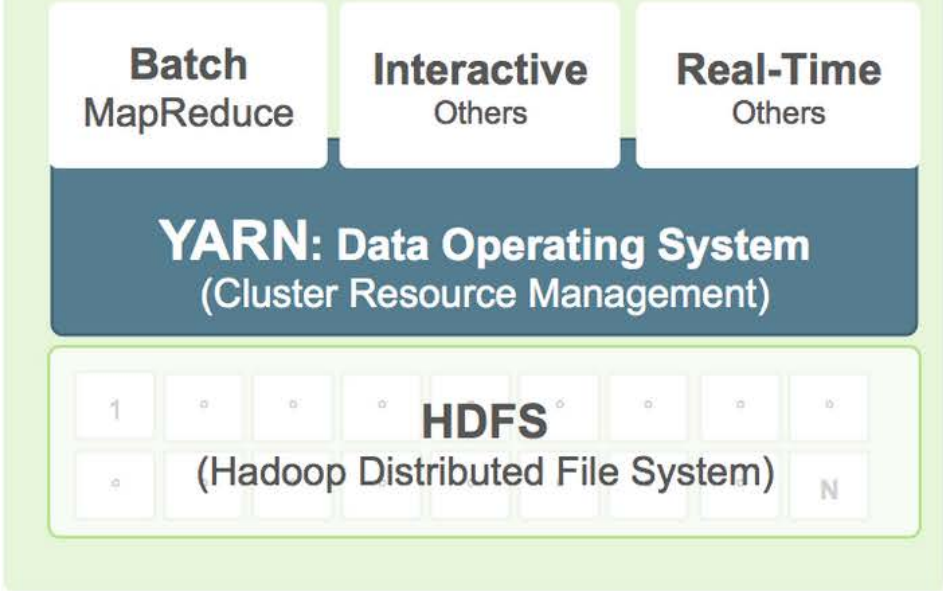
# Hadoop 1

- Silos & Largely batch
- Single Processing engine

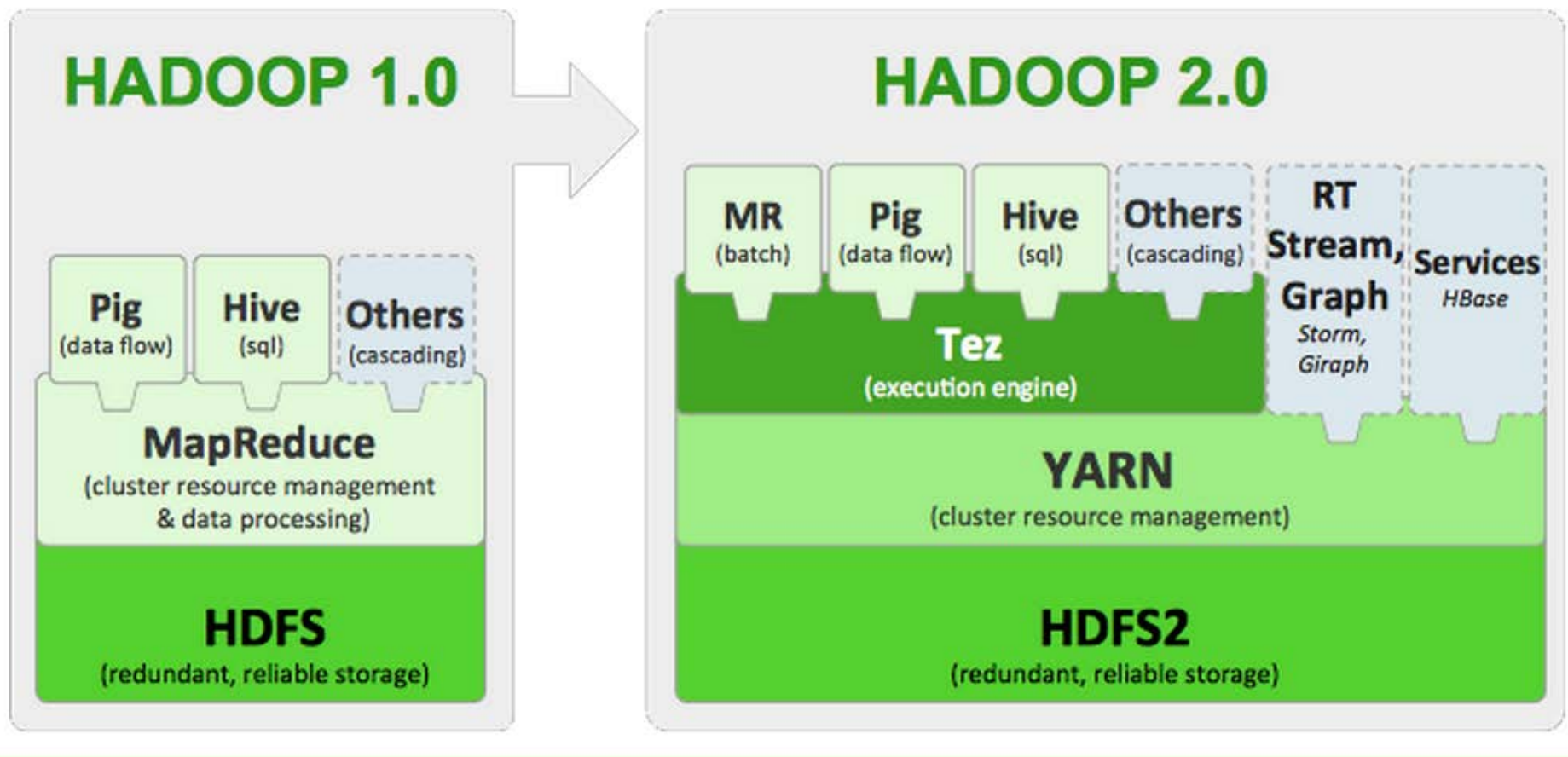


# Hadoop 2 w/YARN

- Multiple Engines, Single Data Set
- Batch, Interactive & Real-Time



source: <http://hortonworks.com/wp-content/uploads/2013/05/yarn.png>



Source: hortonworks.com

Great overview article: "Pig and Hive at Yahoo!," <https://developer.yahoo.com/blogs/hadoop/pig-hive-yahoo-464.html>

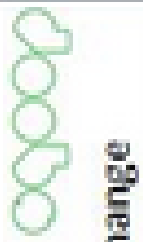


# Apache Hadoop Ecosystem



**Ambari**

Provisioning, Managing and Monitoring Hadoop Clusters



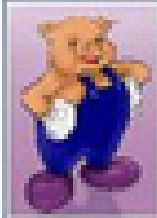
**Scoop**  
Data Exchange



**Zookeeper**  
Coordination



**Oozie**  
Workflow

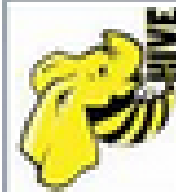


**Pig**  
Scripting



**Mahout**  
Machine Learning

**R Connectors**  
Statistics



**Hive**  
SQL Query



**Hbase**  
Columnar Store



**YARN Map Reduce v2**

Distributed Processing Framework

**HDFS**

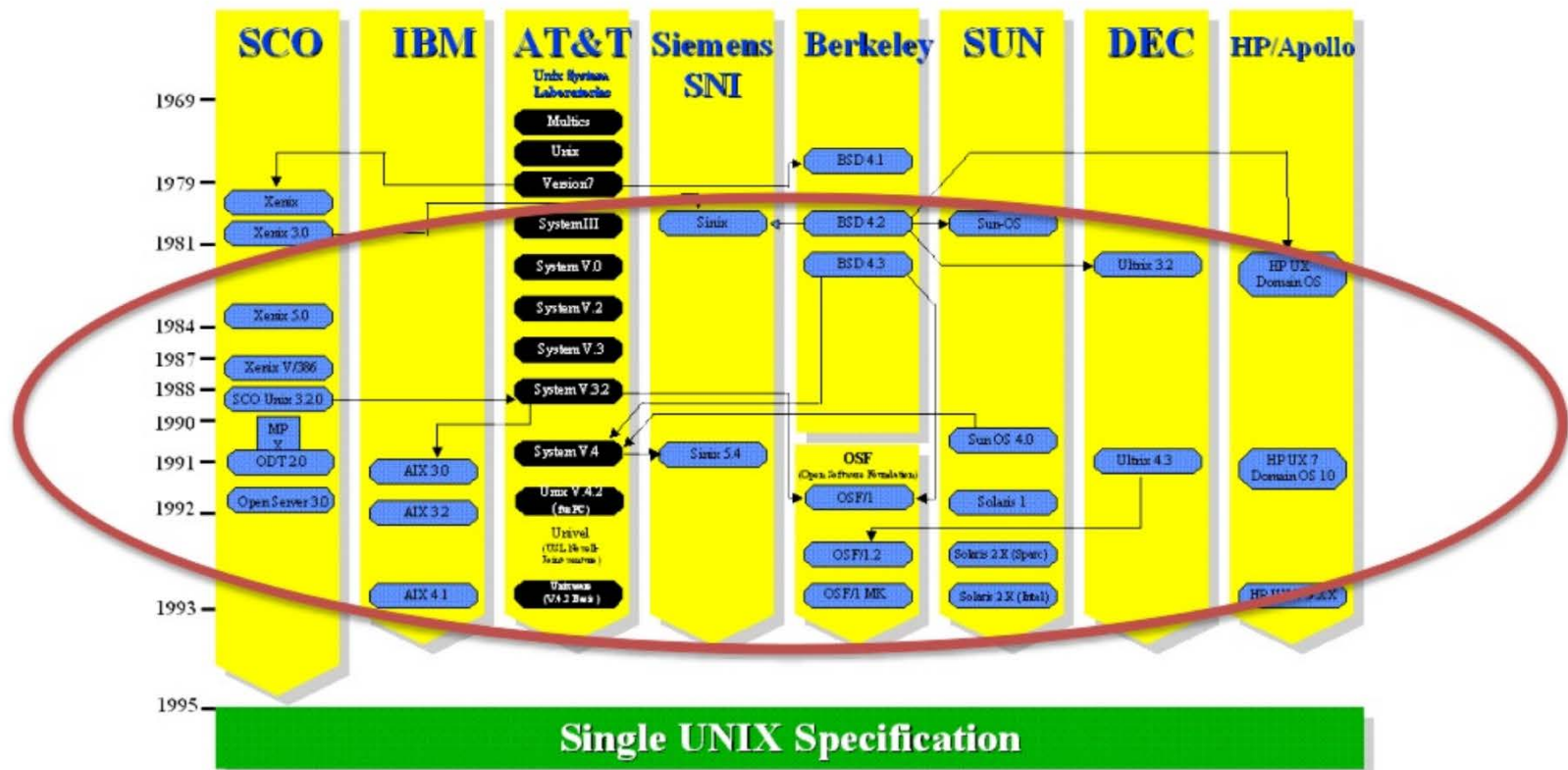
Hadoop Distributed File System



**Flume**  
Log Collector

# Remember the Lost Unix Decade?

## UNIX Chronology



Thanks: [http://www.unix.org/what\\_is\\_unix/flavors\\_of\\_unix.html](http://www.unix.org/what_is_unix/flavors_of_unix.html)

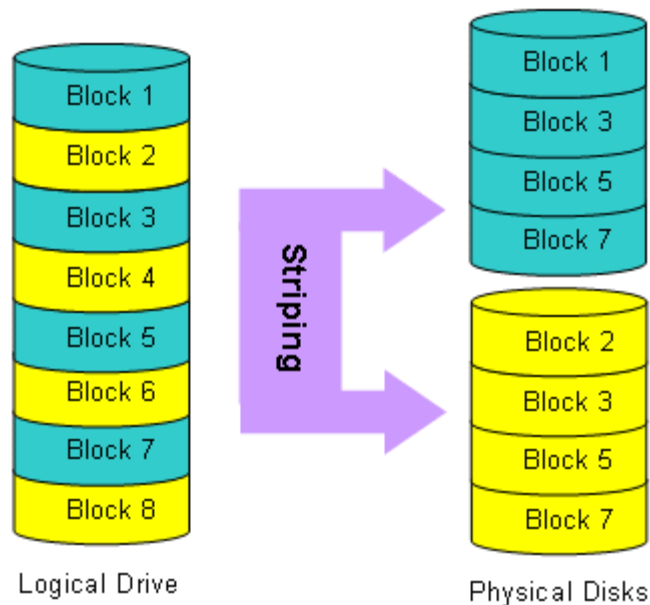
30

# Distributed Computing

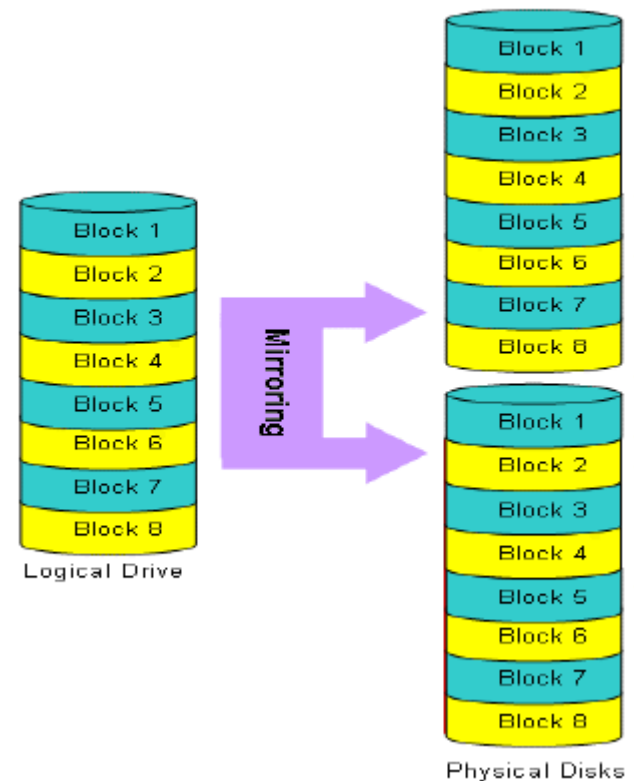
1. computing frameworks
2. **distributed data**
3. distributed processing

# RAID configurations for SQL Database Servers

## RAID 0 Striping



## RAID 1 Mirroring

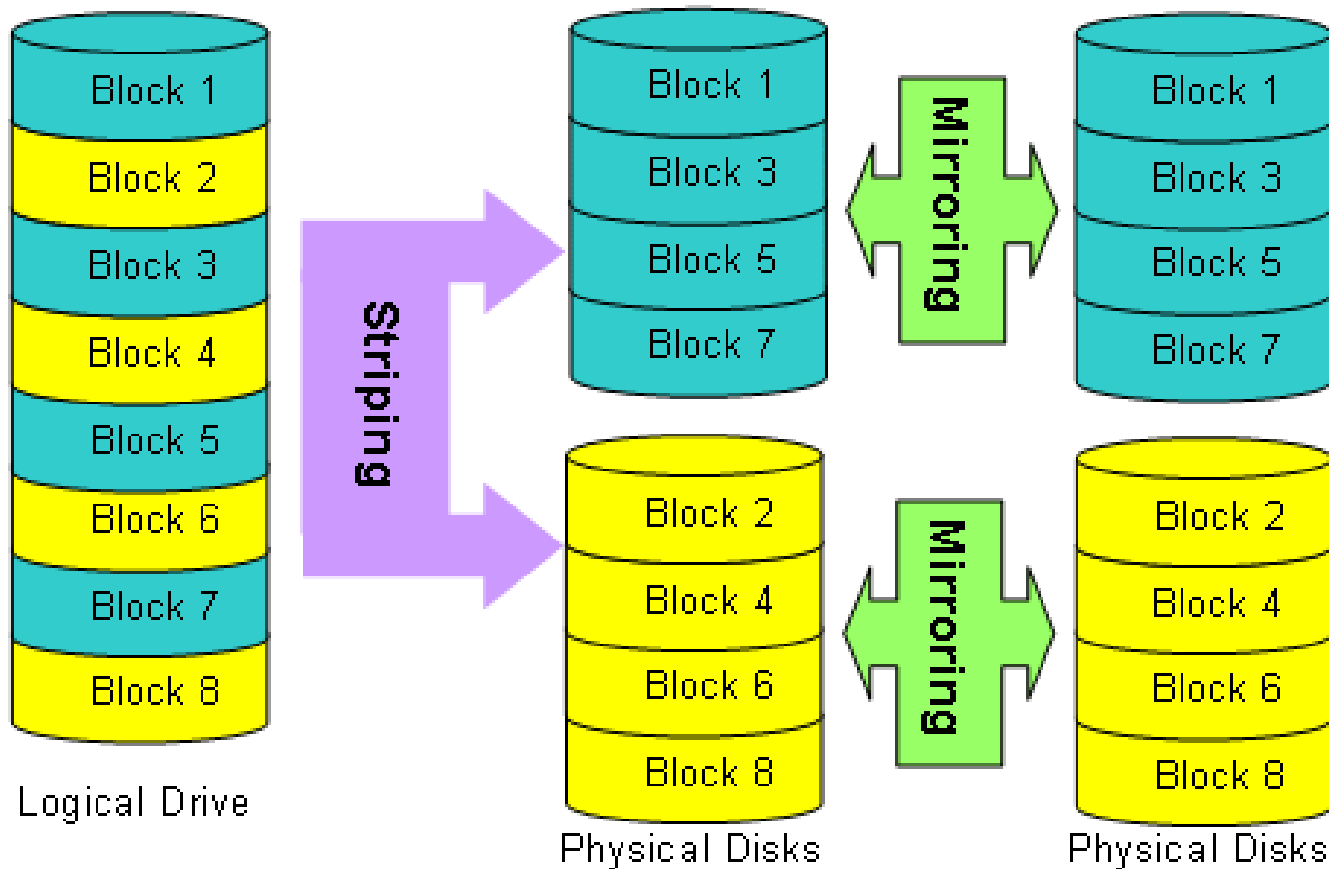


Striping improves performance; Mirroring provides greater protection against failure (redundancy).

Source: <http://www.pantherproducts.co.uk/Articles/Storage/RAID.shtml>



# Distributed Data: RAID 1 + RAID 0 (“10”)



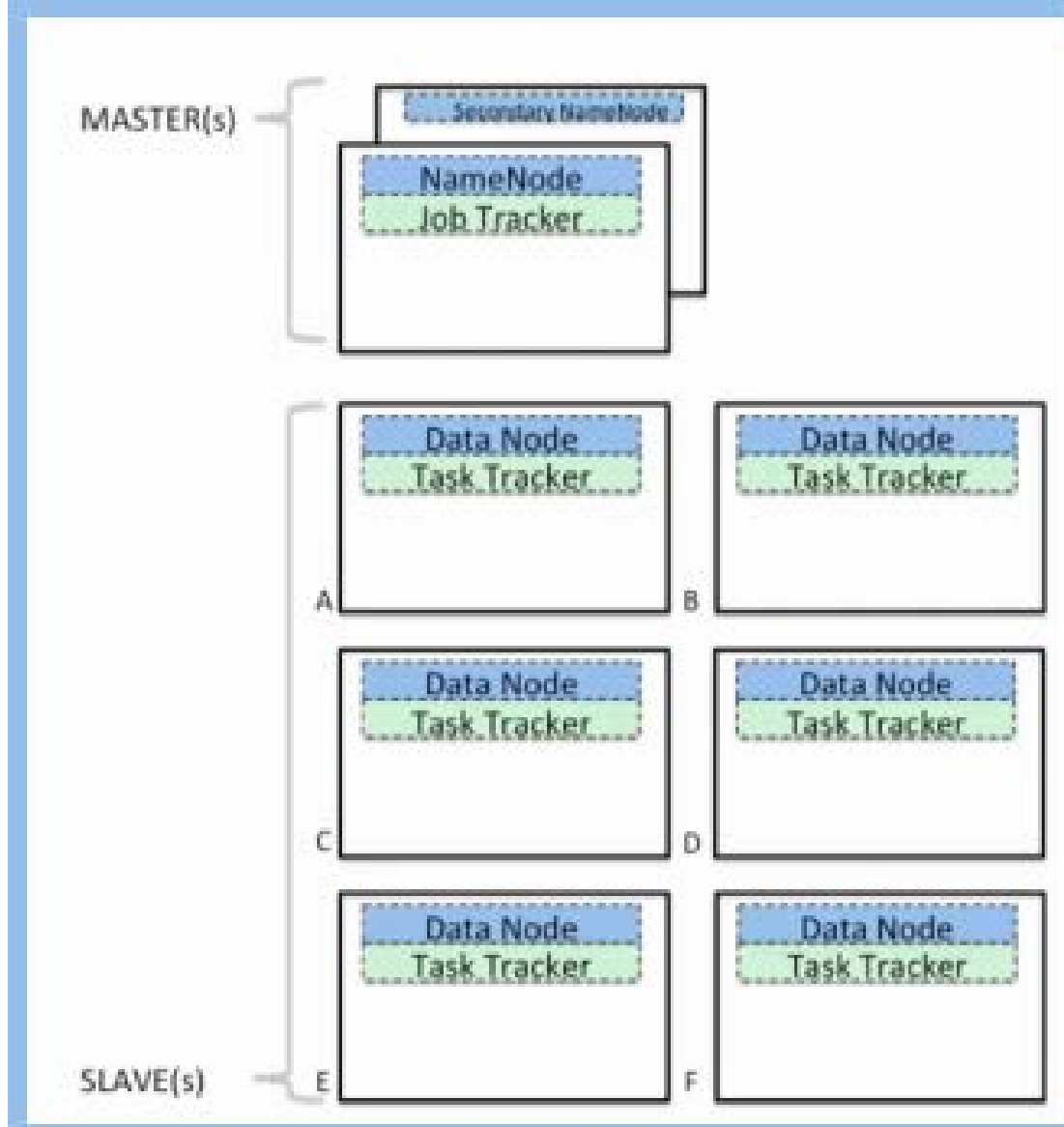
Source: <http://www.pantherproducts.co.uk/Articles/Storage/RAID.shtml>

# Core Hadoop:

## HDFS for storage, MapReduce (YARN) for processing

blue are HDFS daemons; green are  
MapReduce daemons

TaskTrackers in turn “stand up” map  
jobs and reduce jobs as needed.



Source: *Learning Apache Hadoop*, [video available on Safari], Rich Morrow, Infinite Skills. Apr 30, 2014.

# Motivation for Hadoop

Moore's Law for computational power and cost per gigabyte doubling every 18 months has not kept pace for data transfer rates



Typical time to copy 10 TB data: ~ 22 hours

Estimated time for Google to crawl the web

... on one machine: ~46 days

... on 1,000 machines: < 1 hour

A typical commodity server's mean time to failure is 3 years... Google has ~1M commodity servers... 1,000 server fails/day

# Distributed Computing

1. computing frameworks
2. distributed data
3. **distributed processing**

This map shows SETI@home transmissions during a period of five minutes collected within the past 24 hours, and includes IP2Location LITE data available from [ip2location.com](http://ip2location.com). Current data is entirely collected at Arecibo. We are working to include data from GBT and LOFAR.

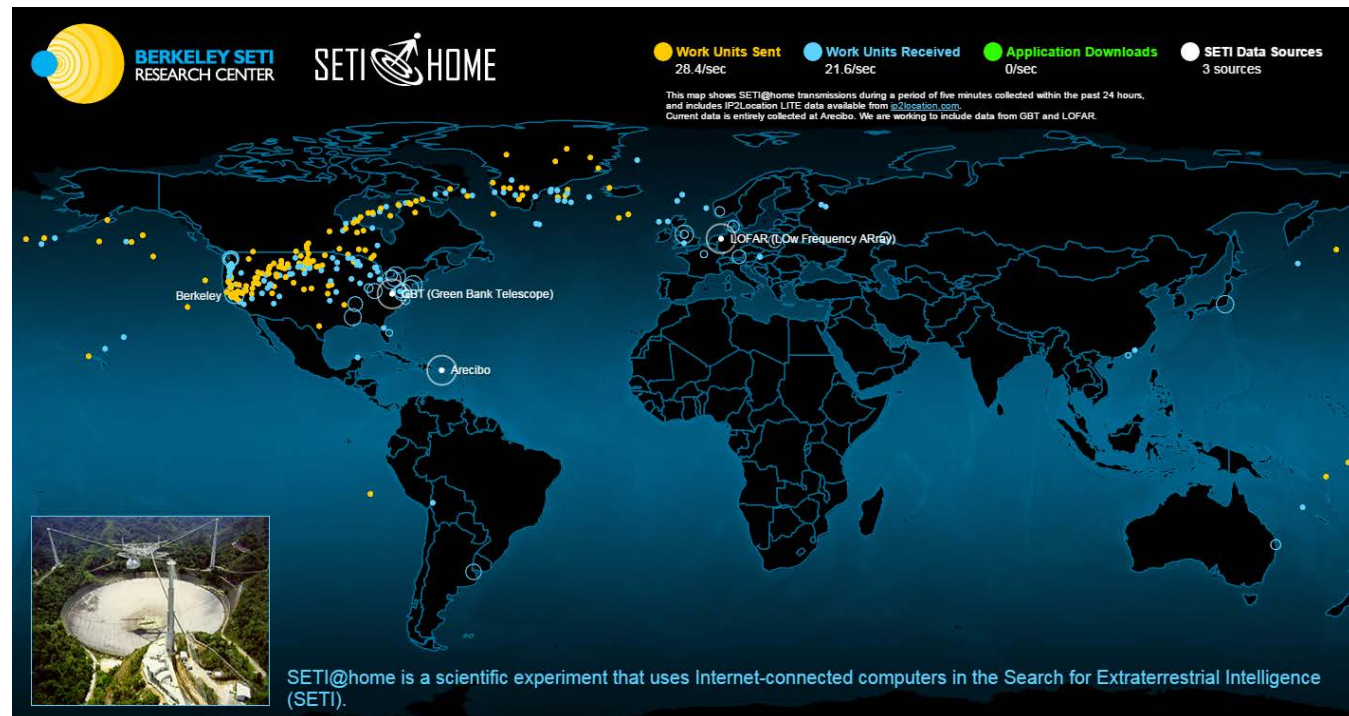


SETI@home is a scientific experiment that uses Internet-connected computers in the Search for Extraterrestrial Intelligence (SETI).

Source: <http://setiathome.berkeley.edu/kiosk/>

[After SETI], many organizations worked to implement distributed processing models. 85 to 90% of the effort was subsumed in getting the infrastructure working.

**Ruthless refactoring.** As Google, Yahoo, Facebook, and others struggled with storing and processing large datasets, they open sourced and built on top of each other's work, which led to the Hadoop ecosystem.





## Motivation for Hadoop

Moore's Law for computational power and cost per gigabyte doubling every 18 months has not kept pace for data transfer rates



Typical time to copy 10 TB data: ~ 22 hours

Estimated time for Google to crawl the web

... on one machine: ~46 days

... on 1,000 machines: < 1 hour

A typical commodity server's mean time to failure is 3 years... Google has ~1M commodity servers... 1,000 server fails/day

*What would be your design criteria for an environment to address these bottleneck issues?*

# Design Goals

We need to minimize moving data around: “code to data” instead of “data to code”

We need to gracefully handle hardware failures

We need to “factor down” a lot of the details to the “operating system” level so that our programming model is not too complex.

Low Operating Costs

# Low Operating Costs

Open Source Software

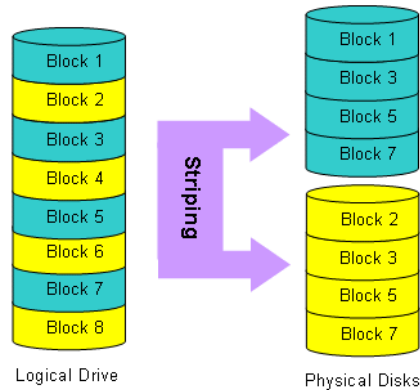
Runs on commodity hardware

Yahoo: 1 system administrator manages 8,000 machines...

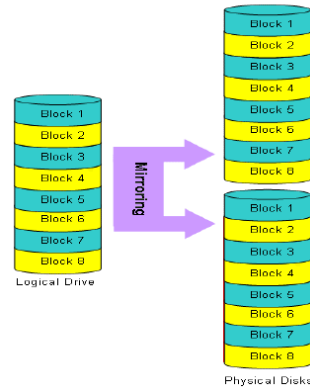
*Why would an organization choose to manage its servers in-house instead of in the cloud?*

# Hadoop:

RAID 0 Striping



RAID 1 Mirroring

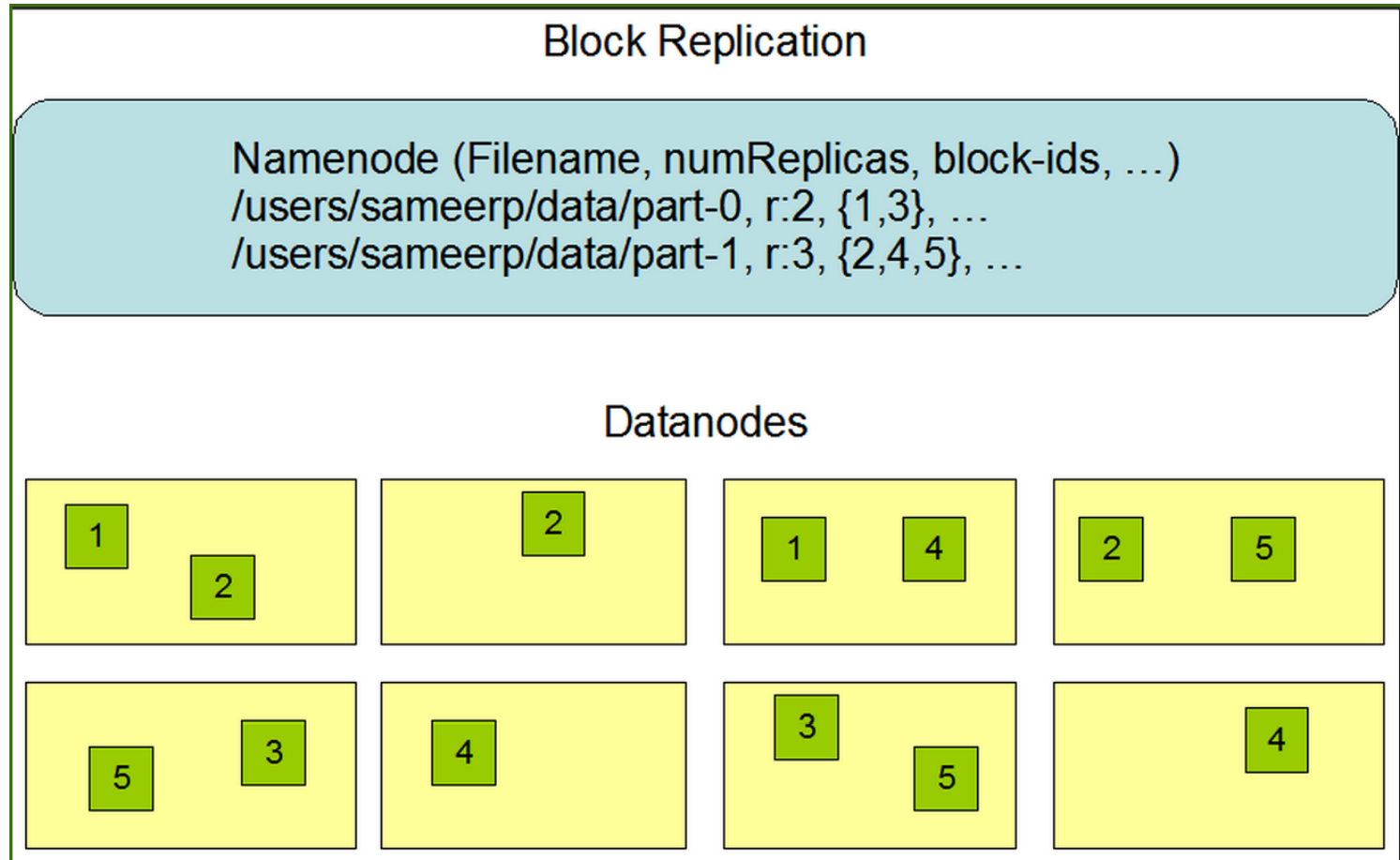


**Horizontal Scaling.** Striping improves performance; Mirroring provides greater protection against failure (redundancy). Typically, these drives are all on a single machine.

**Vertical Scaling.** Hadoop uses a similar strategy to RAID's striping and mirroring with its two core components spread across multiple *machines*:

- **HDFS** – distributed data
- **MapReduce / YARN** – distributed processing

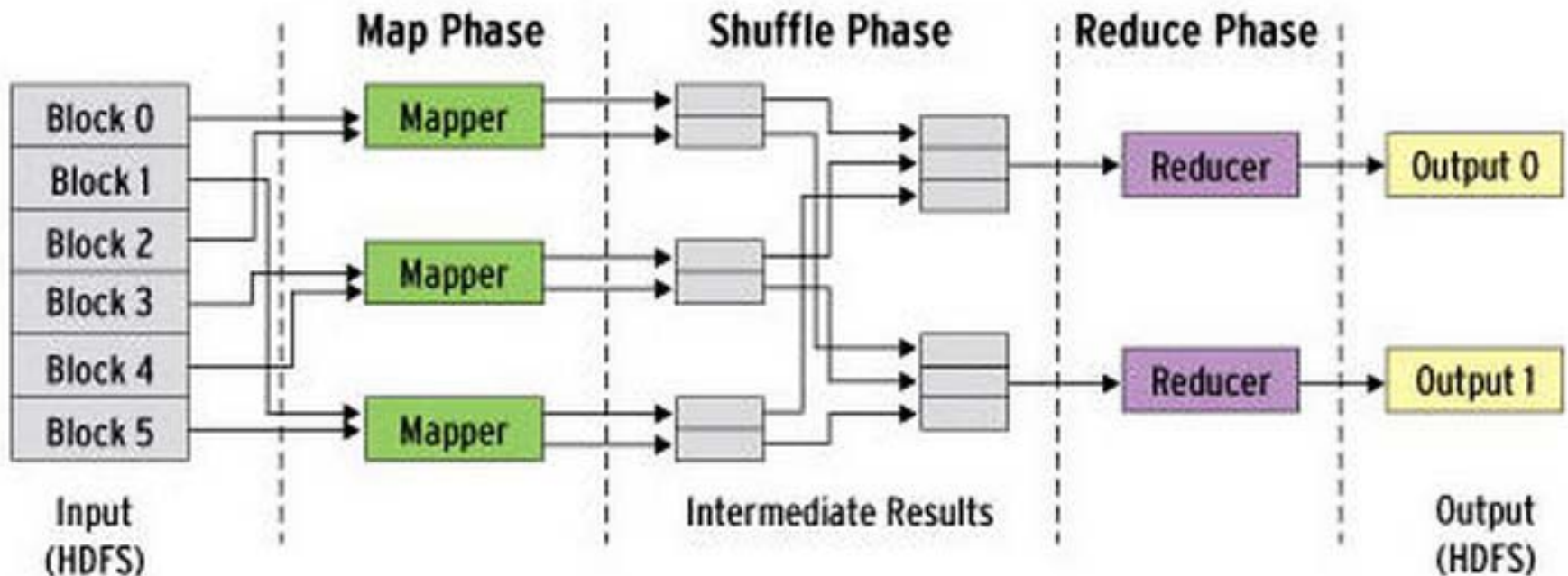
# Graceful Hardware Fails



Source: "HDFS Architecture Guide," [http://hadoop.apache.org/docs/r1.2.1/hdfs\\_design.html](http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html)

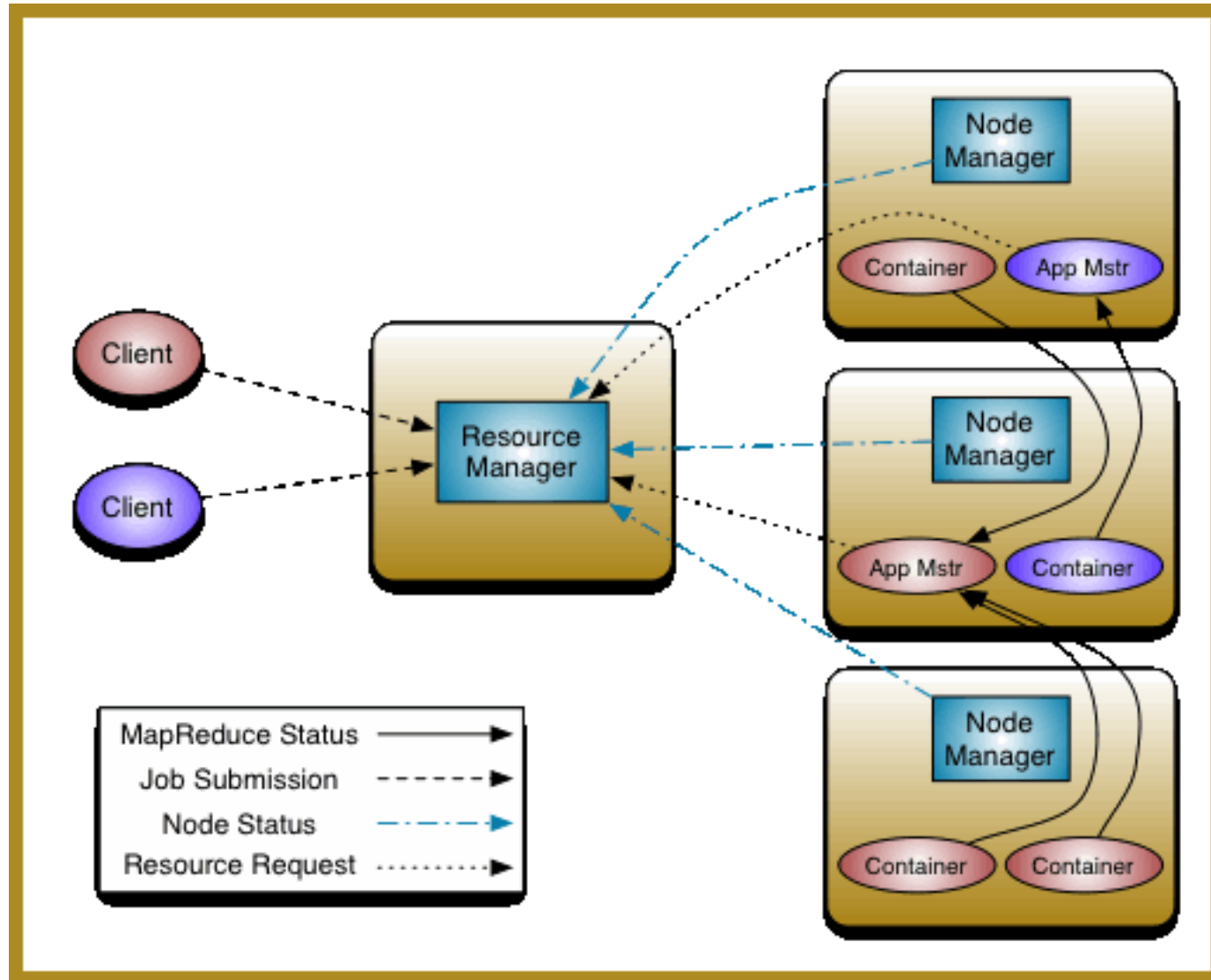
# (Relatively) Simple Programming Model

We write the Mapper and the Reducer; the Hadoop environment handles the other details (graceful recovery from fails, moving data around; allocating mappers and reducers; assigning code to run on nodes; sort/shuffle “magic”...)



Source: <http://www.admin-magazine.com/HPC/Articles/MapReduce-and-Hadoop>





source: "Apache Hadoop NextGen MapReduce (YARN),"

<http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html>

# Where to Get Hadoop

## Hadoop Distributions

- HortonWorks
- Cloudera
- MapR

# Where to Run Hadoop

## In the Cloud

- Amazon, Microsoft Azure
- Other providers (IBM, Google, Rackspace,...)

## In Your Enterprise

- Starting point: 4-6 repurposed machines

## On Your desktop in a sandbox

- Can be a better dev environment (e.g. much easier to set up first class debugging)

# Three Run Modes

- **Standalone Mode** – single machine (or VM). Good for testing code. Much faster for small datasets.
- **Pseudodistributed Mode** – single machine (or VM). Good for testing code and data. Faster and cheaper for small datasets.
- **Fully Distributed Mode** – cluster of machines. Run here when necessary.

“Standing up” new Java Virtual machines can takes 10s of seconds, so fully distributed mode is much slower for smaller datasets!

Candidate best practice: Run in standalone mode, then in pseudodistributed mode, then in fully distributed mode with ~10% of data), then with all data.

# MapReduce Development

- Java (first class environment)
- Streaming. R, Hadoop, others
- Analytics: Spark, Pig, Hive, others
- Recommendation Systems: Mahout!?

# Hadoop and R

“RHadoop: Ecosystem of R packages”:  
<http://www.r-bloggers.com/search/hadoop>

Hadoop	RHadoop
hdfs	rhdfs
hbase	rhbase
mapreduce	rmr2, plyrmr



## Hive / Pig Syntax comparison

### # HIVE QUERY

```
SELECT loc, AVG(sal)
FROM emp JOIN dept USING(deptno)
WHERE sal > 3000 GROUP BY loc;
```

### # PIG QUERY

```
emp = LOAD '/hdfs/path/emp_file.txt' AS (loc);
dept = LOAD '/hdfs/path/dept_file.txt' AS (uid,amt);
filtered = FILTER emp BY sal > 3000;
joined = JOIN filtered_emp BY deptno, dept BY deptno;
grouped = GROUP emp_join_dept BY loc;
result = FOREACH grouped_by_loc GENERATE group,
AVG(emp_join_dept.sal);
dump result;
```

In Hive, we had to:

- Create tables
- Populate tables

Pig looks “heavier”

- And it often is slightly more verbose
- But, remember...

Source: *Learning Apache Hadoop*, [video available on Safari], Rich Morrow, Infinite Skills. Apr 30, 2014.

# Metastore

```
CREATE EXTERNAL TABLE page_view(  
  viewTime INT, userid BIGINT,  
  page_url STRING, referrer_url STRING,  
  ip STRING)  
COMMENT 'This is the page view table'  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY '\t'  
LOCATION '/some/path/in/HDFS/';
```

# data in HDFS:

```
1391102617^t1234^thttp://quicloud.com/^thttp://google.com/^t198.211.110.9
```

...

Now, in Hive I can run a query like:

```
SELECT * FROM page_view WHERE page_url LIKE "%quicloud%"
```



Source: *Learning Apache Hadoop*, [video available on Safari], Rich Morrow, Infinite Skills. Apr 30, 2014.

# Use Cases

- Yelp. 400GB/log added per day.
- New York Times generated 11M pdfs for \$240.

Self-Service, Prorated Supercomputing Fun!, Derek Gottfried,  
<http://open.blogs.nytimes.com/2007/11/01/self-service-prorated-super-computing-fun>, Nov 1, 2007; <http://www.roughtype.com/?p=1189>

← NIH announces all funding decisions to be made by someone who hates you

No-one quite sure what the point is anymore, survey finds


→


## BREAKING: NoSQL just “huge text file and grep”, study finds

Posted on [October 28, 2014](#) by [jovialscientist](#)

BREAKING NEWS: a study by the institute for distributed investigation of technologies (IDIOT) has found that all NoSQL technologies are essentially just a massive text file combined with the UNIX tool ‘grep’

NoSQL has risen in popularity in recent years as a hipster alternative to relational databases. Technology companies tend to move their entire data management system to a NoSQL backend, before moving it back to a relational database management system less than 12 months later.

“This is a huge shock!” said Dr Rick Slowman of the University of Birmingham. “We’ve all written  scripts that grep huge files to link them together, but noone thought an entire tech sector would be based on this!” he continued.

In a related report, IDIOT have confirmed that XML is totally 

Source:

<https://thescienceweb.wordpress.com/2014/10/28/breaking-nosql-just-huge-text-file-and-grep-study-finds/>

Command-line tools can be 235x faster than your Hadoop cluster,” Adam Drake, <http://aadrake.com/command-line-tools-can-be-235x-faster-than-your-hadoop-cluster.html>, Jan 25, 2014.

# Hadoop Use Cases / Hype Cycle

# Discussion

