

# BIG DATA AND HADOOP

**Venkatesh Vinayakarao**

[venkateshv@cmi.ac.in](mailto:venkateshv@cmi.ac.in)

<http://vvtesh.co.in>

---

Chennai Mathematical Institute

---

Data is the new oil. - Clive Humby, 2006.

# Know Your Instructor

**BE (Computer Science and Engineering)**

Java/J2EE  
Developer

**MS (Information Technology)**

SDE, Search  
Technologies  
Group, Bing,  
Microsoft

Principal  
Engineer, Cloud  
Platforms Group,  
Yahoo

**PhD (Computer Science)**

Intern, Porting ML  
Models to Azure,  
Microsoft  
Research

# Agenda

- Introduction to Big Data
- Course Dynamics
- Evolution of Systems and Technologies
  - Data Storage
  - Data Processing

# What Comes Next?

byte

kilobyte

megabyte

gigabyte

??

???

????

?????

# Sizes

Name	Size
Byte	8 bits
Kilobyte	1024 bytes
Megabyte	1024 kilobytes
Gigabyte	1024 megabytes
Terabyte	1024 gigabytes
Petabyte	1024 terabytes
Exabyte	1024 petabytes
Zettabyte	1024 exabytes
Yottabyte	1024 zettabytes

# The Impact of Big Data



Your train is on time thanks to **big data**

TNW - 31-Dec-2019

Thanks to thousands of sensors and **big data** analytics, train ... It's this data that keeps the Dutch rail network moving, and helps NS deliver a ...



The power of **data** in smart city developments

Independent Australia - 03-Jan-2020

Other fascinating **big data** developments that were presented included ... led to the production of the Australian **Cancer** Atlas — an interactive, ...



At HCA Healthcare, Real-Time **Data Saves Lives**

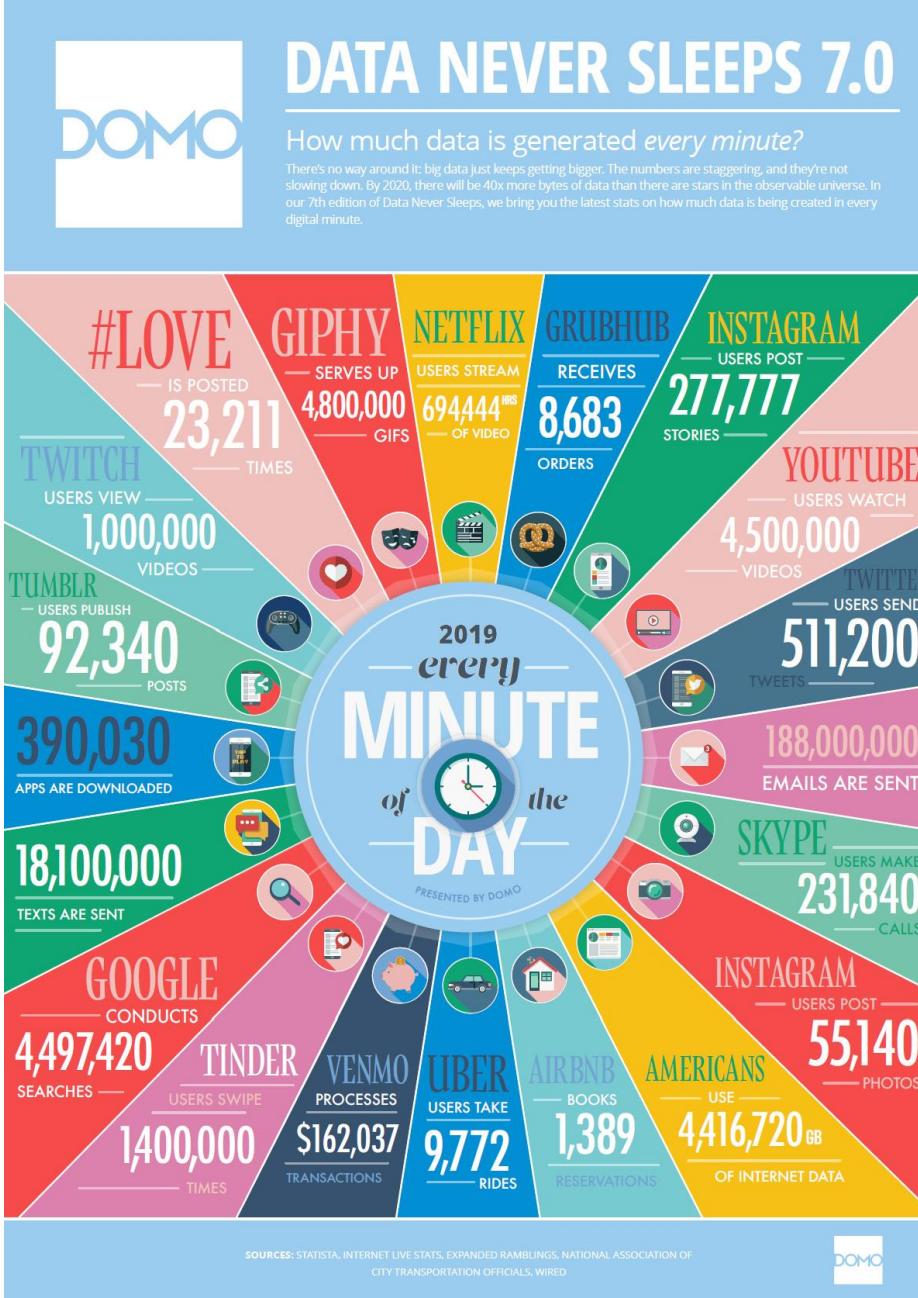
RTInsights (press release) (blog) - 01-Jun-2019

At HCA Healthcare, Real-Time **Data Saves Lives** ... "Our existing **data** infrastructure was designed for **large**-scale business intelligence and ...

# Big Data is Ubiquitous

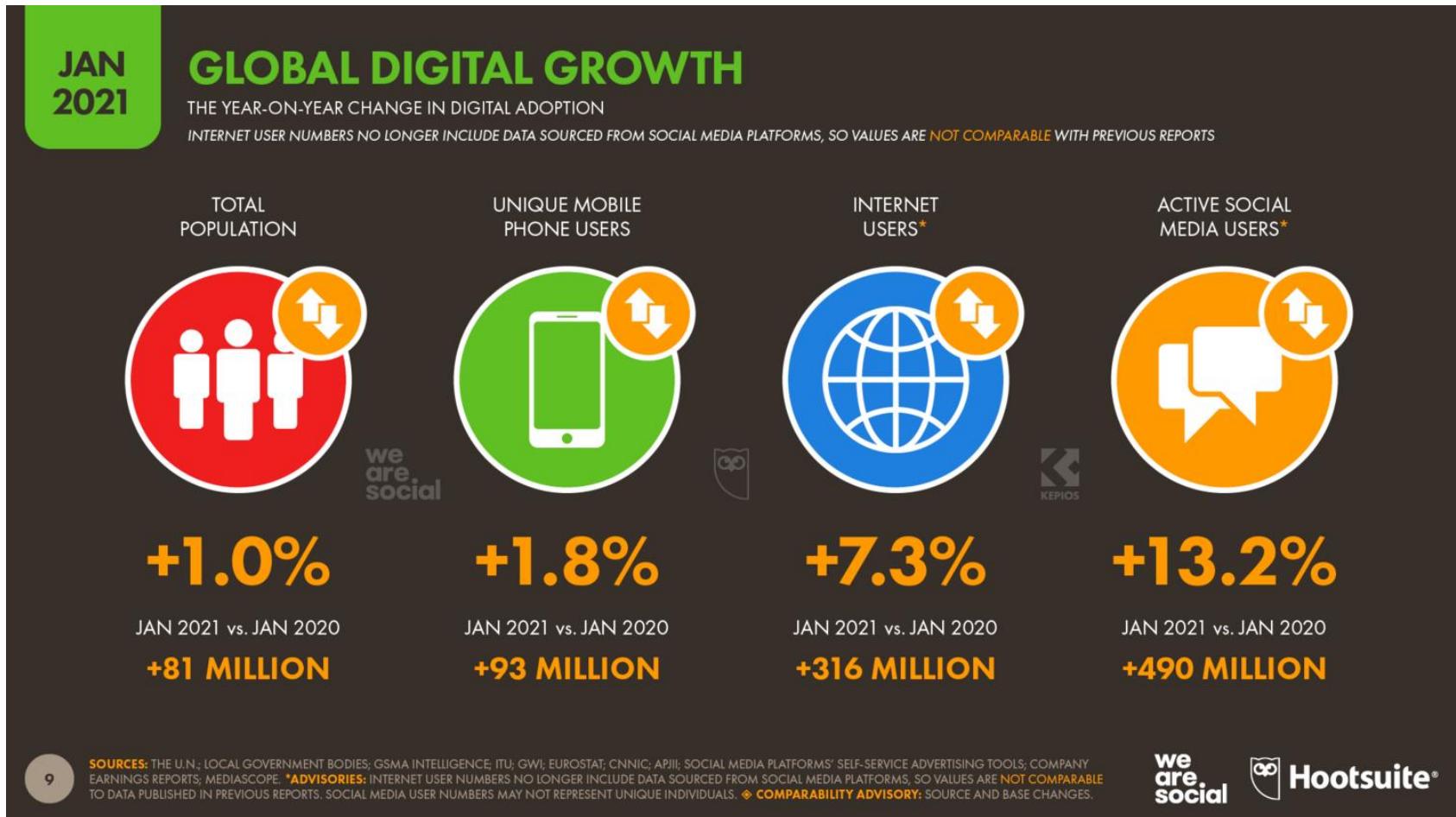
- Facebook (**per day** statistics)
  - 1.5 billion people are active on Facebook **daily!**
  - More than 300 million photos get uploaded **per day!**
  - Totally, more than 2.5 Trillion posts!
- Facebook (per minute statistics)
  - **Every minute** there are 510,000 comments posted and 293,000 statuses updated!
- Youtube (**per minute** statistics)
  - Users watch 4,146,600 YouTube videos!
- Guess... How many emails are sent per minute?

Source: [Forbes](#)

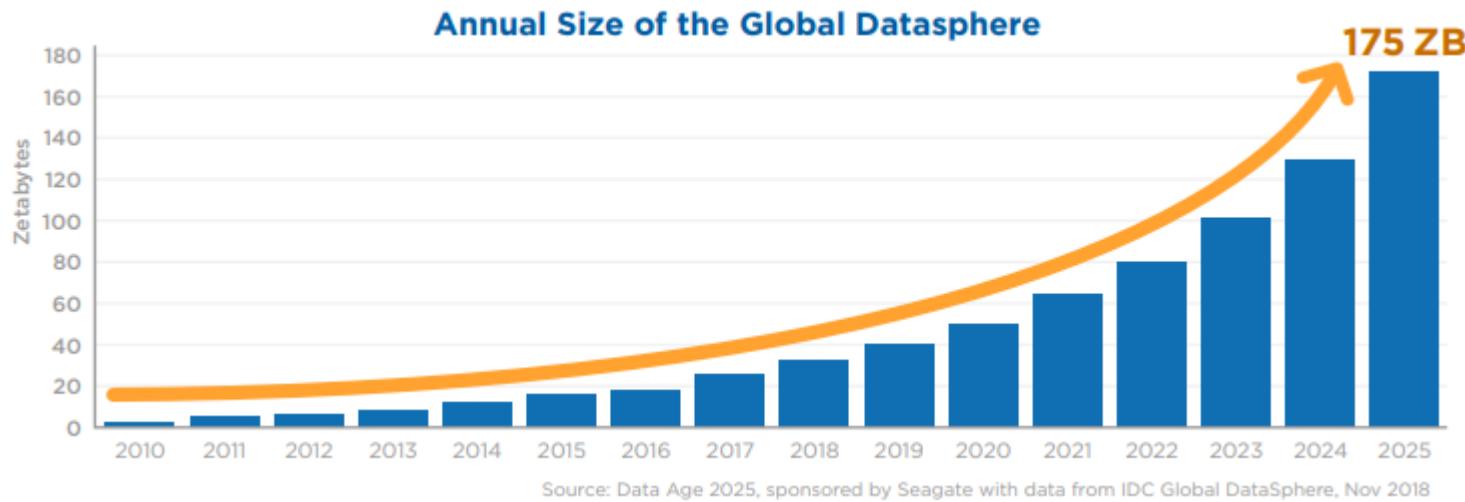


Source: <https://www.visualcapitalist.com/big-data-keeps-getting-bigger/>

# And, It is Growing!



# Data Growth



Mankind's quest to digitize the world!  
33 ZB (2018) → 175 ZB (2025)  
size of global datasphere\*

\*Source: <https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf>

**Global datasphere is growing!**

How have the computers evolved to capture,  
process and analyze these data?

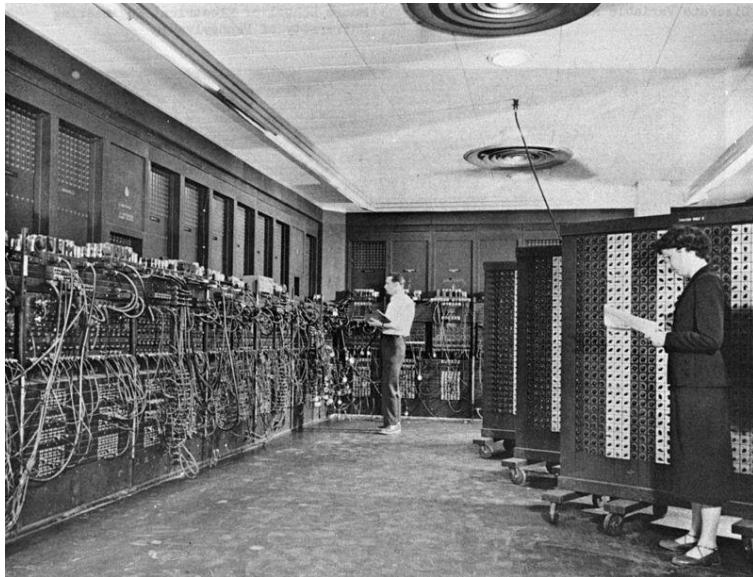
# Course Dynamics

<https://vvtesh.github.io/teaching/bdh-2022.html>

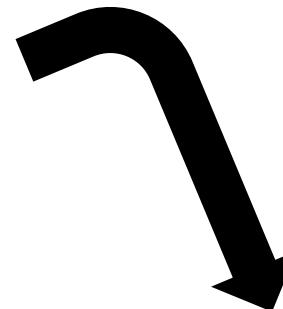
# Student Presentations

- Please register
  - your choice of presentation topic,
  - your team details (Team size: 3 or 4)
- Deadlines
  - (1 Mark) Register your topic before Feb 10<sup>th</sup>. Registration link will be available on moodle.
  - (2 Marks) Send a one-page abstract of the talk before mid-term. Clearly mention the team members.
  - (12 Marks) Complete the presentation one week before the final exam.

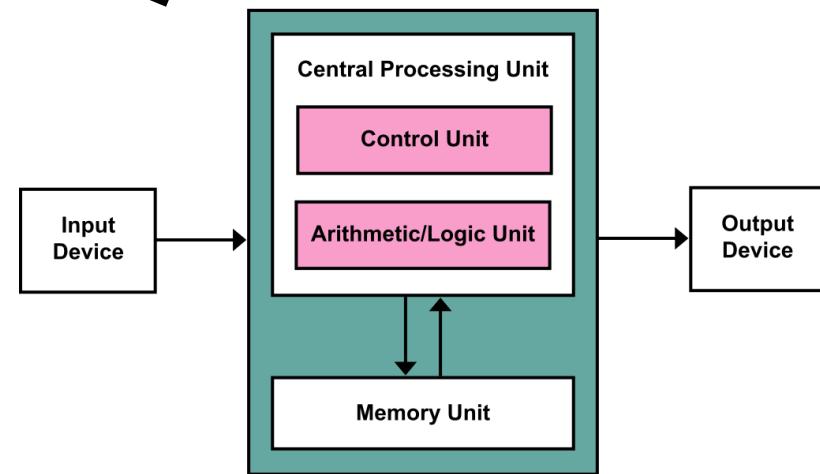
# Evolution of Computers



**ENIAC**  
**Early 1900s**



**Stored-program  
Von Neumann  
Architecture  
1940**



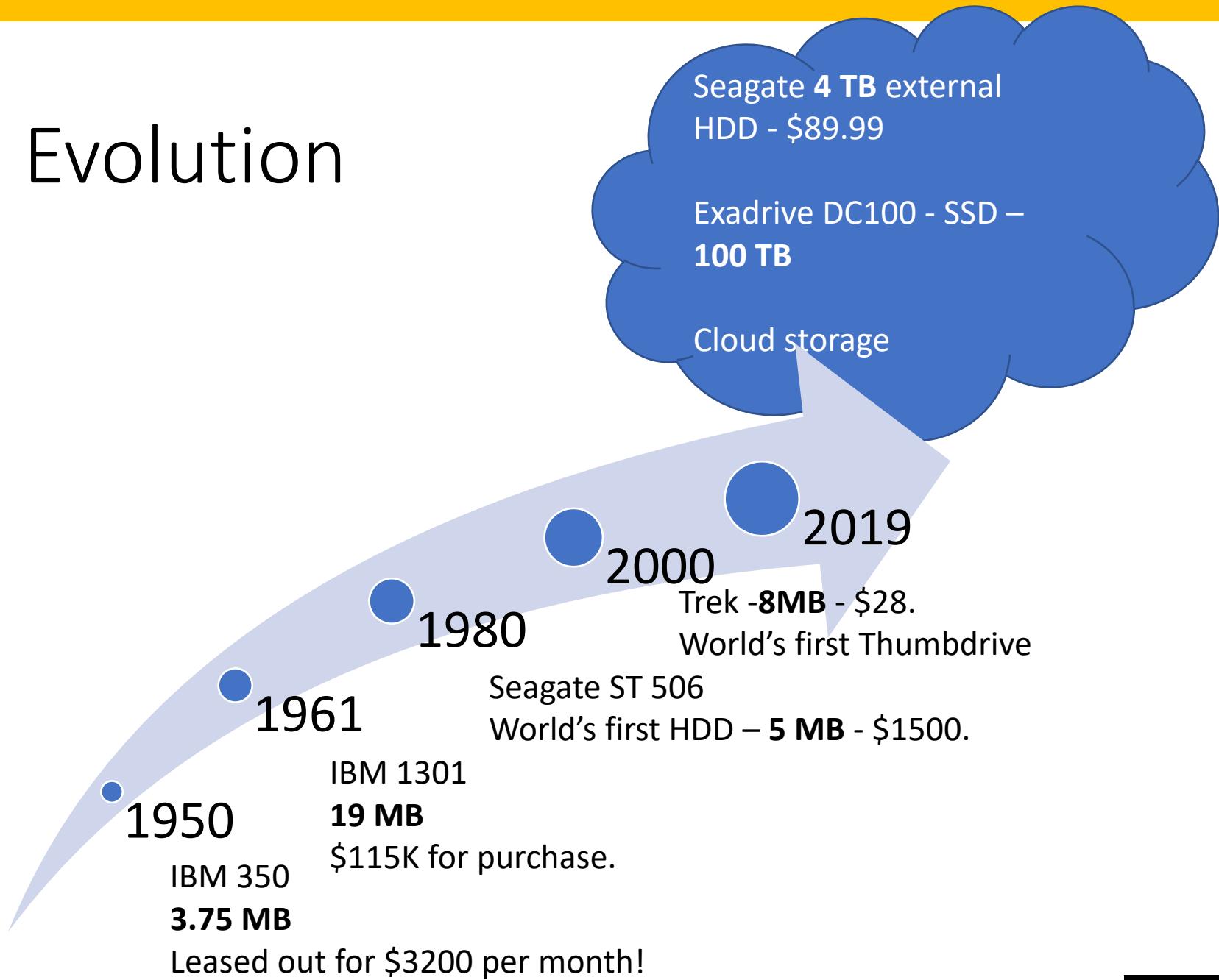
# Two Kinds of Problems

Storage

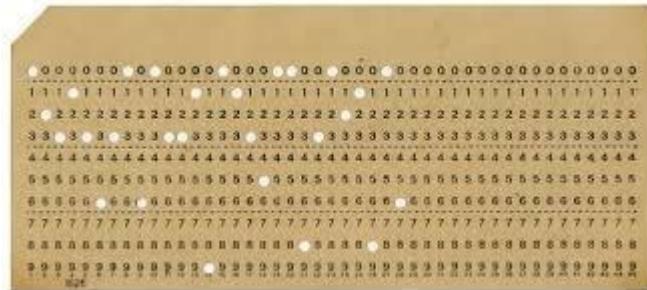
Processing

# Data Storage

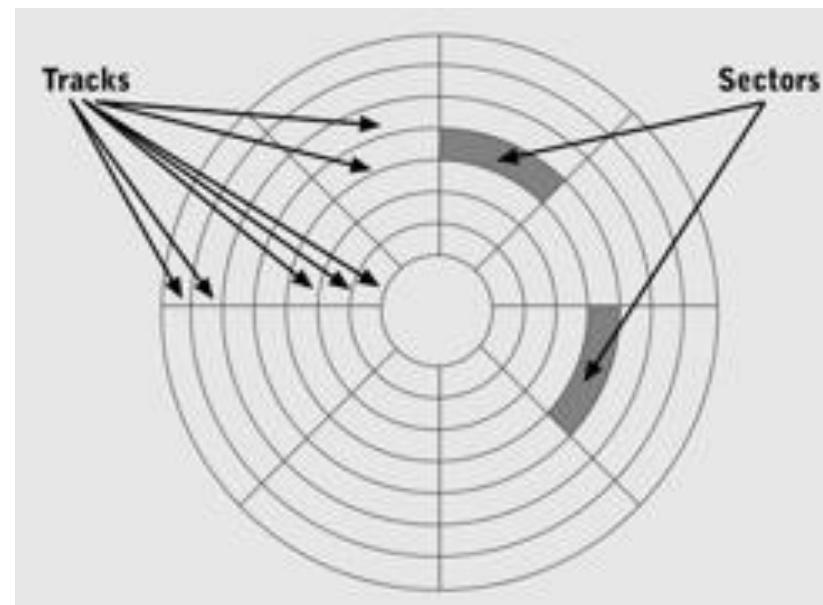
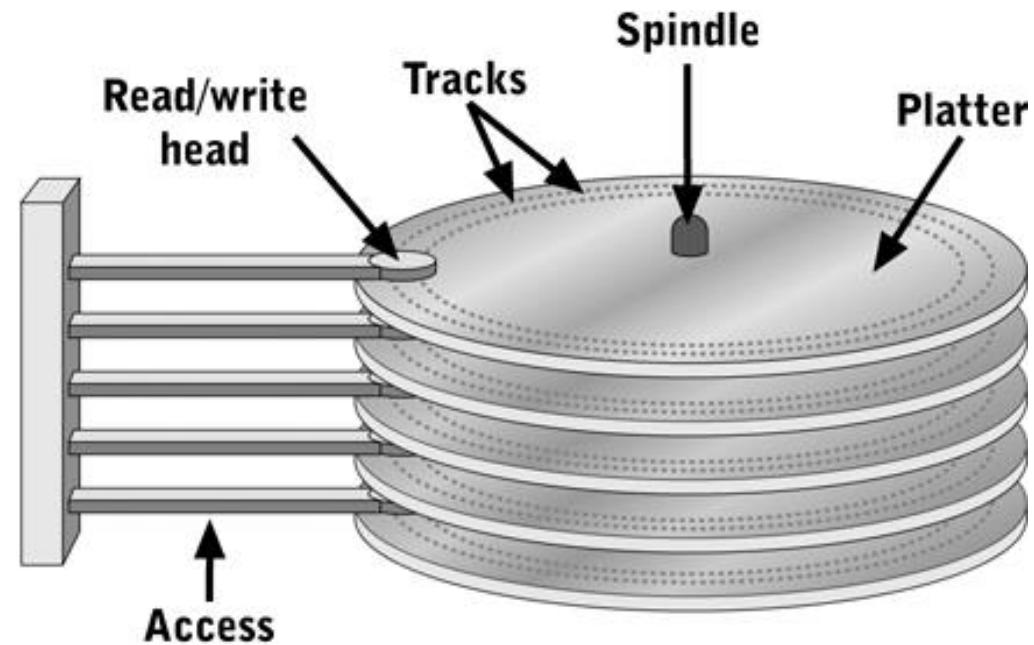
# Evolution



# (Secondary) Storage Technologies



# Disk Drive and Access Time



Source: Systems Architecture, Fifth Edition



Roll over image to zoom in

## Seagate 500GB SATA Laptop Hard Disk

by [Seagate](#)



279 ratings | 493 answered questions

M.R.P.: ₹ 2,999.00

Price: ₹ 1,433.00 + ₹ 77.00 Delivery charge [Details](#)

You Save: ₹ 1,566.00 (52%)

Inclusive of all taxes



Pay on  
Delivery



10 Days  
Replacement



Amazon  
Delivered



1 Year  
Warranty

In stock.

Delivery by: **Jan 8 - 10** [Details](#)

© Deliver to Venkatesh - Chennai 600014

Sold by **KCM\_STORE** (3.9 out of 5 stars | 29 ratings).

New (20) from ₹ 1,510.00 + FREE Shipping

- 500 GB capacity
- 5400 RPM spin speed, 16 MB cache buffer
- Designed for durability and low-power consumption
- SATA 3GB interface with native command queuing
- Perpendicular recording technology for increased storage capacity
- Fast performance and whisper quiet acoustics

# Average Access Time

- Head switching time is considered negligible (H)
- Head seek time (S)
- Rotational delay = Time taken for  $\frac{1}{2}$  a rotation (average) (R)
- Read time = time to spin an entire sector (T)
- Average Access Time =  $H + S + R + T$

\*Sector is a minimum storage unit

# Quiz

- If disk spins at 6000 RPM, compute the rotational delay.

# Quiz

- If disk spins at 6000 RPM, compute the rotational delay.
  - One turn takes  $1/6000$  min or  $1/100$  sec = 10ms
  - $\frac{1}{2}$  a turn takes 5ms.

# Read Time

- If the drive spins at 6000RPM and the disk has 20 sectors per track, what is the read time?

- Time for 1 full spin is  $\frac{1}{6000} \text{ min} = \frac{1}{100} \text{ sec} = 10\text{ms}$

- Time for 1/20 of a spin is  $10\text{ms} \times \frac{1}{20} = 0.5\text{ms}$

# Average Access Time

- Drive spins at 7200RPM and has average seek time of 8ms. The disk has 24 sectors per track. What is the average access time?

Head seek time	0.008 sec (Given)
Rotational delay	$1/120 * (1/2) = 0.0042$ sec
Read time	$0.0084 \text{ (full spin)} / 24 \text{ sectors} = 0.00035 \text{ sec}$
<b>Avg Seek Time</b>	$= 0.008 + 0.0042 + 0.00035$ <b><math>= 0.01255 \text{ sec or } 12.55 \text{ ms}</math></b>

# Characteristics

Attribute	Description
Speed	Time to read/write
Volatility	Data persistence even when powered off
Access Method	Serial, Parallel
Portability	Internal, External
Capacity	Volume of data storage

# Storing/Managing/Processing Data

- RDBMS
- ETL
- OLTP
- Data Warehouse
- Data Lake
- Cloud Storage
- STaaS

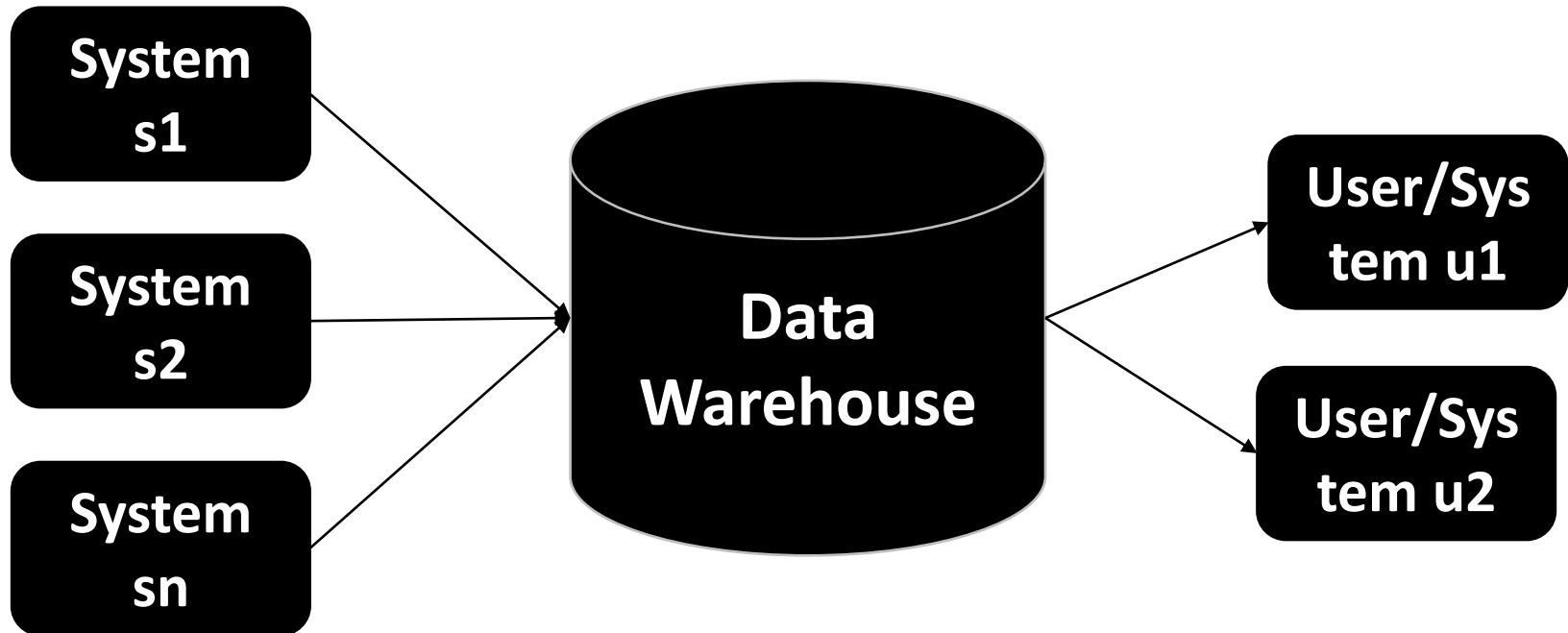
# ETL

- Extract, Transform and Load (ETL)
  - general procedure followed to address data variety
- Variety of data sources
  - Tabs, Sensors, Desktops, Bots, Multiple databases, Files,...
- Variety of data formats
  - Text, PDFs, XML, JSON, Images, Videos, ...

# Fast OLTP

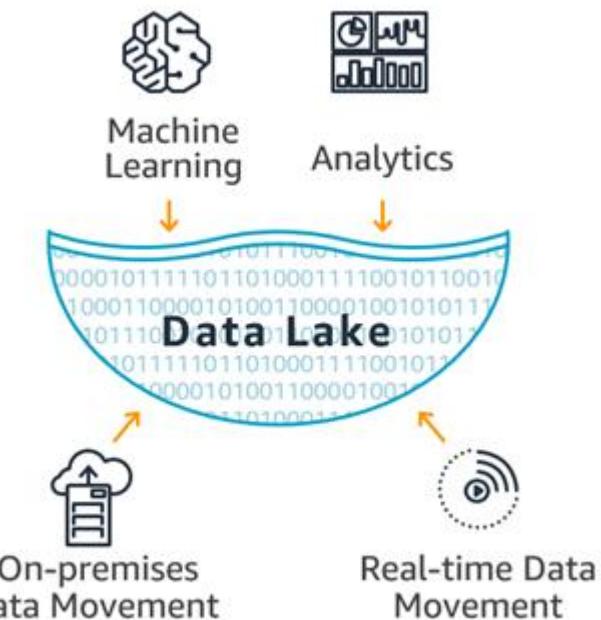
- Online Transaction Processing (OLTP)
- Real-time/Near Real-time Performance. Finds application in:
  - Banking
  - Railway Reservations
  - Stock Market Trading
    - Handle transactions in milliseconds.
      - VoltDB, MemSQL, ...

# Data Warehouse



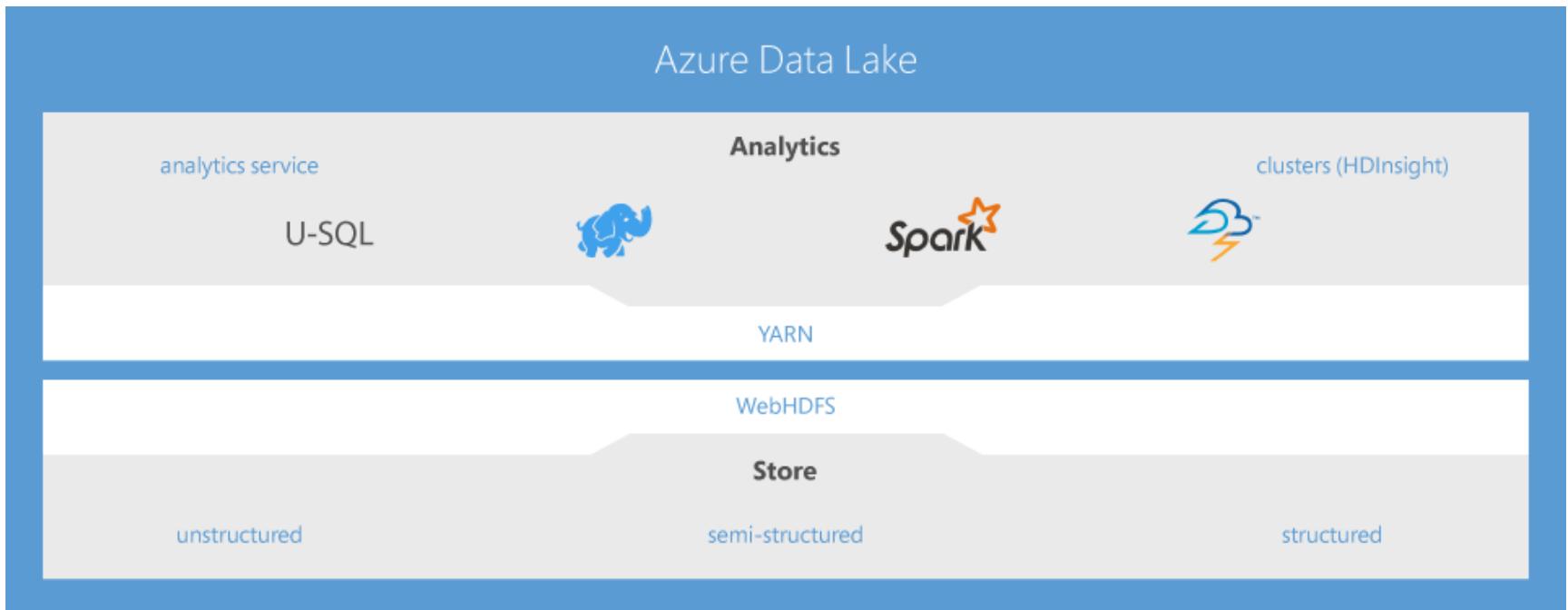
# Data Lakes

- No schema definition.
- Store everything
  - often without or with very little pre-processing, /cleaning.
- Use ML, analytics to query, or gather insights.



Source: <https://aws.amazon.com/big-data/datalakes-and-analytics/what-is-a-data-lake/>

# Microsoft's Azure Data Lake



More details at <https://azure.microsoft.com/en-us/resources/videos/azure-data-lake-making-big-data-easy/>

# Data Warehouse Vs. Data Lake

Characteristics	Data Warehouse	Data Lake
Data	<b>Relational</b> from transactional systems, operational databases, and line of business applications	<b>Non-relational and relational</b> from IoT devices, web sites, mobile apps, social media, and corporate applications
Schema	Designed <b>prior</b> to the DW implementation (schema-on-write)	Written <b>at the time of analysis</b> (schema-on-read)
Price/Performance	Fastest query results using <b>higher cost storage</b>	Query results getting faster using <b>low-cost storage</b>
Data Quality	Highly <b>curated data</b> that serves as the central version of the truth	Any data that may or may not be curated (ie. <b>raw data</b> )
Users	<b>Business analysts</b>	<b>Data scientists</b> , Data developers, and Business analysts (using curated data)
Analytics	<b>Batch</b> reporting, BI and visualizations	Machine <b>Learning</b> , Predictive analytics, data discovery and profiling

Source: <https://aws.amazon.com/big-data/datalakes-and-analytics/what-is-a-data-lake/>

# Storing on the Cloud

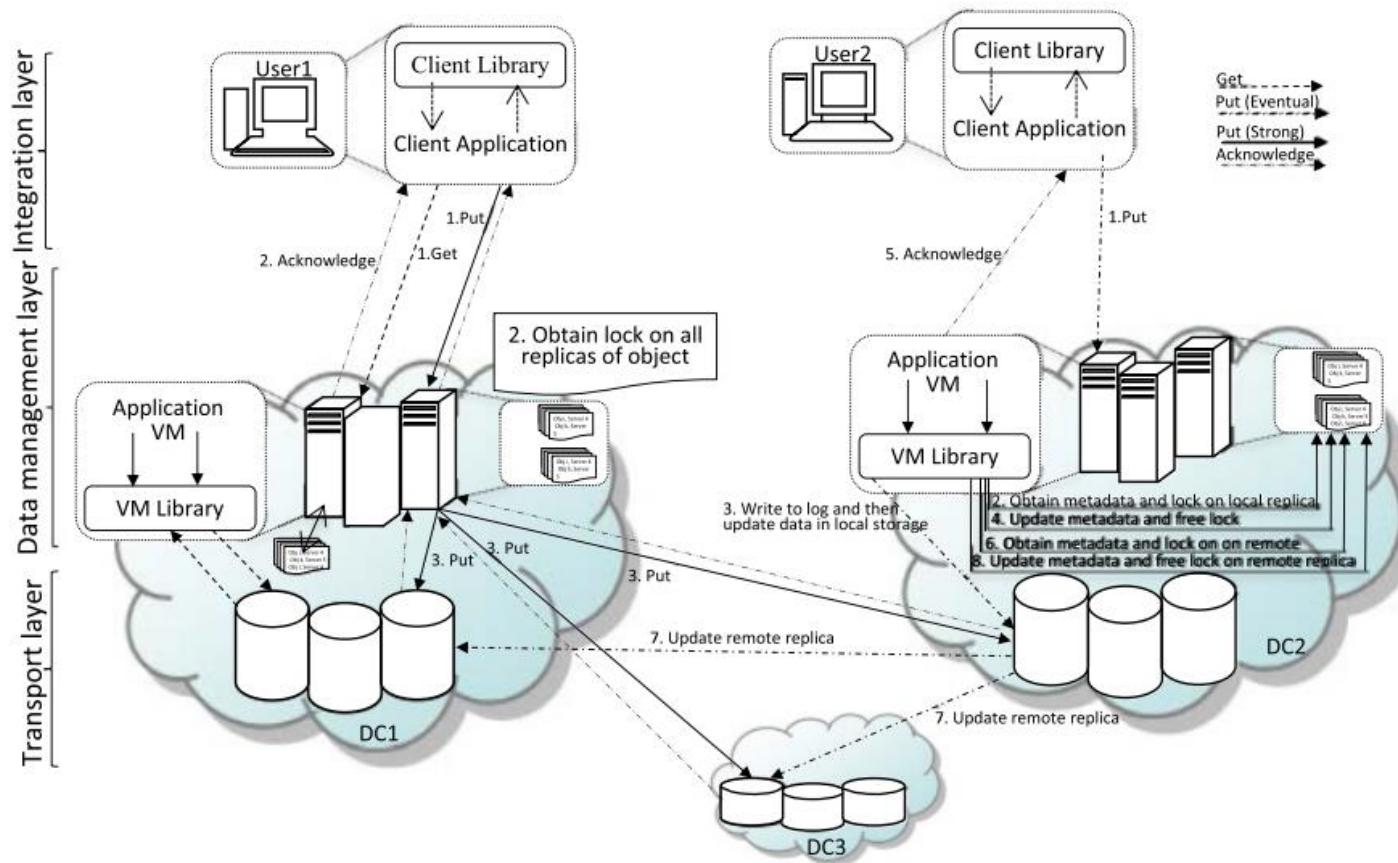
- Gmail: Gives 15 GB of free storage (as of 2020)
- Several online sites for storing images, apps, files, ...
  - Security
  - Ease of sharing
  - Backups
  - Availability



# Storage as a Service (STaaS)

- What is it?
  - A business model in which a company rents space in their storage infrastructure to another company or individual.
- How does it work?
  - STaaS provider rents space
  - cost-per-gigabyte-stored and cost-per-data-transfer basis.
- Benefits
  - Shifting from Capital Expenditure to Operational Expenditure
  - Scale up/down at will (temporarily)

# Cloud Storage



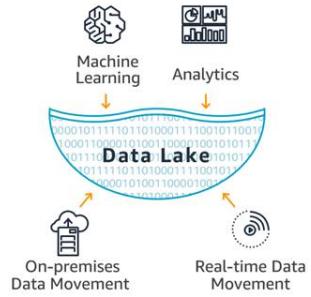
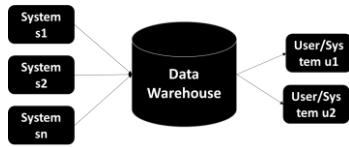
# Cloud Computing



A data center

Source: [https://resources.sei.cmu.edu/asset\\_files/WhitePaper/2010\\_019\\_001\\_28877.pdf](https://resources.sei.cmu.edu/asset_files/WhitePaper/2010_019_001_28877.pdf)

# Summary



Data Storage - Summary

# DATA PROCESSING

**Venkatesh Vinayakarao**

[venkateshv@cmi.ac.in](mailto:venkateshv@cmi.ac.in)

<http://vvtesh.co.in>

---

Chennai Mathematical Institute

---

Data is the new oil. - Clive Humby, 2006.

# What Comes Next?

byte

kilobyte

megabyte

gigabyte

??

???

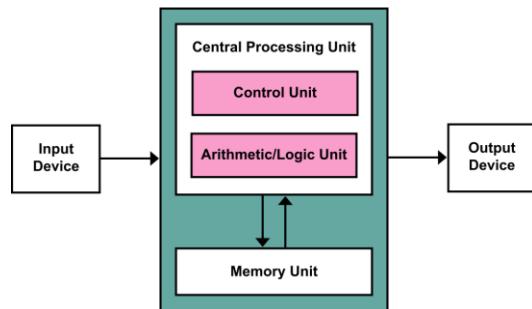
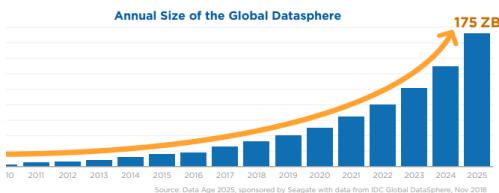
????

?????

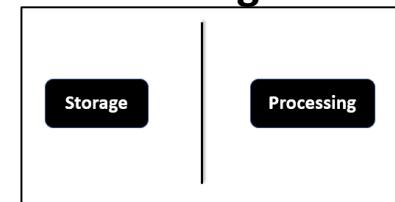
# Sizes

Name	Size
Byte	8 bits
Kilobyte	1024 bytes
Megabyte	1024 kilobytes
Gigabyte	1024 megabytes
Terabyte	1024 gigabytes
Petabyte	1024 terabytes
Exabyte	1024 petabytes
Zettabyte	1024 exabytes
Yottabyte	1024 zettabytes

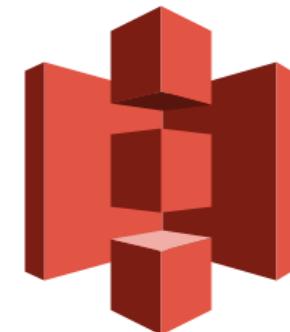
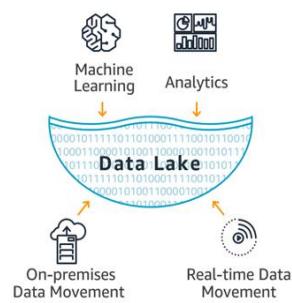
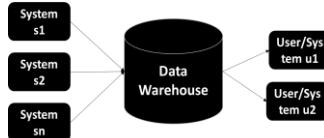
# Recap



## Challenges

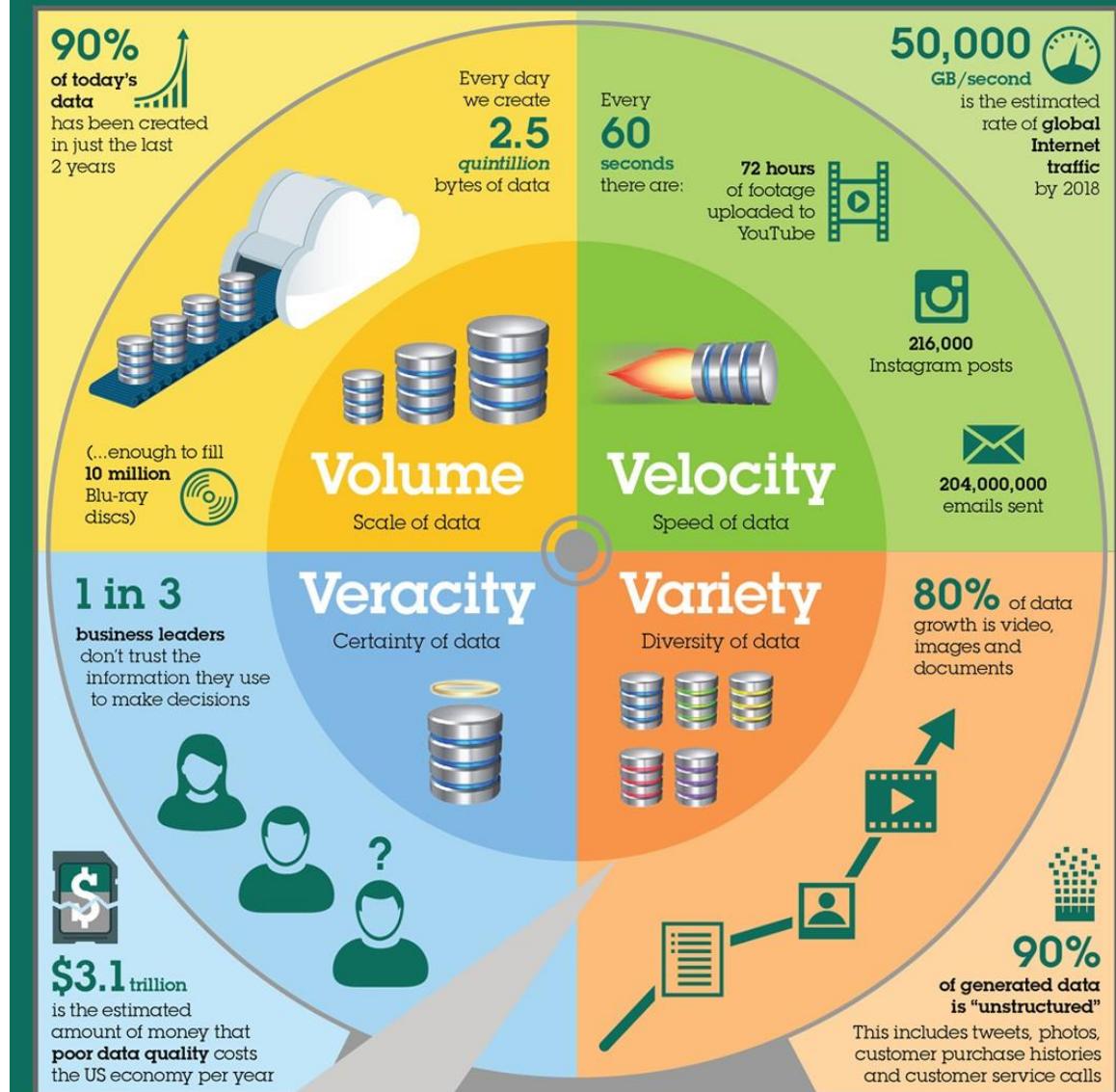


## Data Storage



Amazon S3  
STaaS

# Extracting business value from the 4 V's of big data



# Big Data Characteristics

- Volume
  - Petabytes, exabytes, ...
- Variety
  - pdf, json, text, images, ...
- Velocity
  - real-time, near real-time, batch
- Veracity
  - Trustworthiness, correctness and consistency

**“Where there is data smoke, there  
is business fire.”**

**— Thomas Redman, Author.**



Tuj mein rab diktha hai  
**Data** mein kya karoon

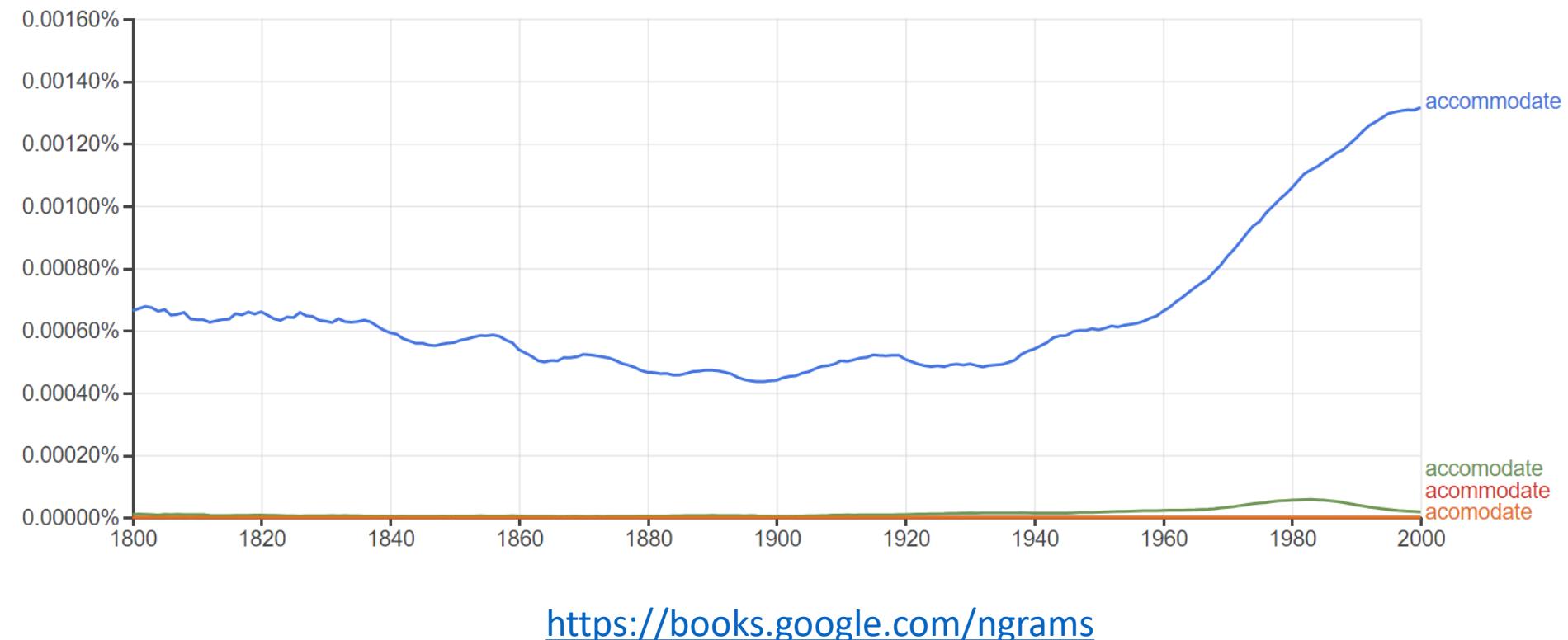


jab bhi koi **data** dekhun  
mera dil deewana bole  
ole ole ole...

# Quiz

- Which is right?
  - accommodate
  - acommodate
  - accomodate
  - acomodate

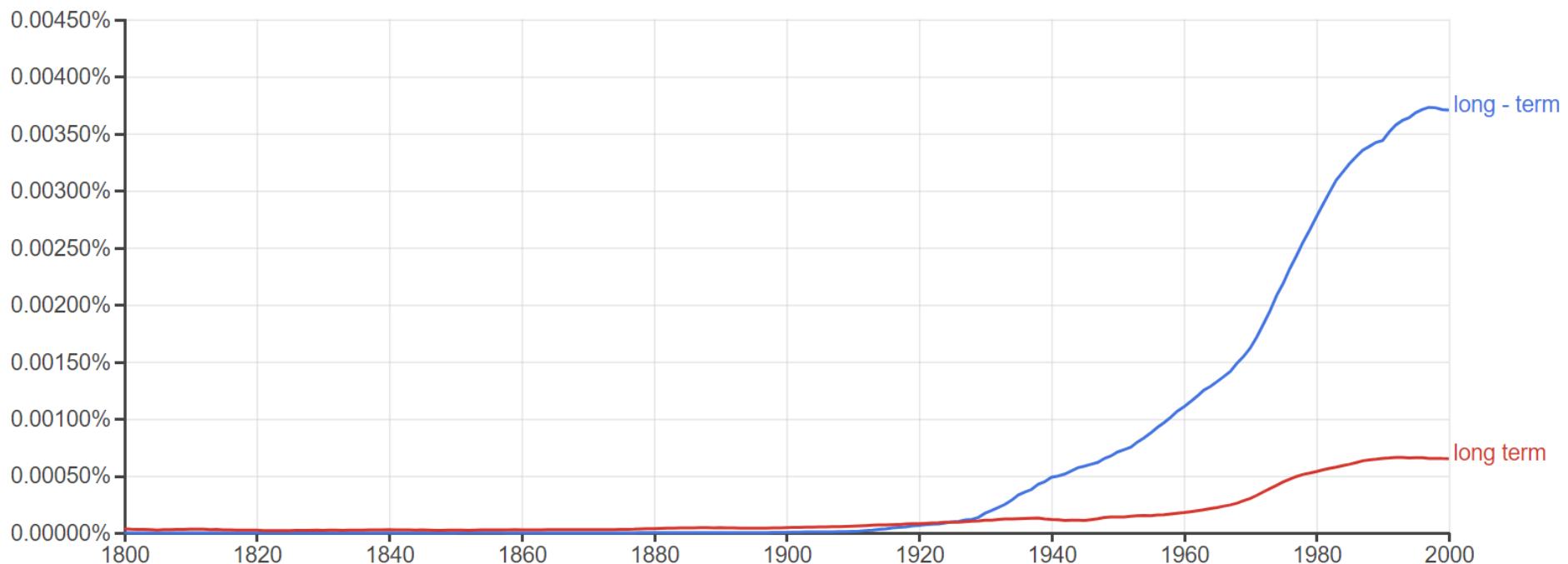
# Google n-gram Viewer



<https://books.google.com/ngrams>

# Quiz

- Long-term or long term



# Data Processing

- Microprocessors
- Multi-core Processors
- Supercomputers
- ... all roads lead to ~~Rome~~ Cloud!



## Microprocessors

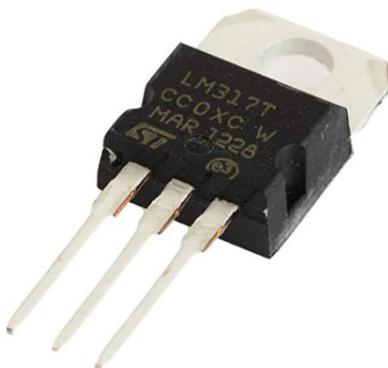
Processing unit on an integrated circuit

What are ICs made of?



# Transistors

- Basic electronic component that alters the flow of current.
- Form the basic building block of an integrated circuit.
- Think of it as an electronic switch



SunRobotics LM317 Voltage Regulator IC TO-220 Adjustable Three-Terminal Regulators Field Effect Transistor Original Integrated circuit electronic Components (5 Pcs)

by sunrobotics

1 rating

M.R.P.: ₹199.00

Price: ₹ 165.00

You Save: ₹ 34.00 (17%)

Inclusive of all taxes

prime FREE Delivery by Monday



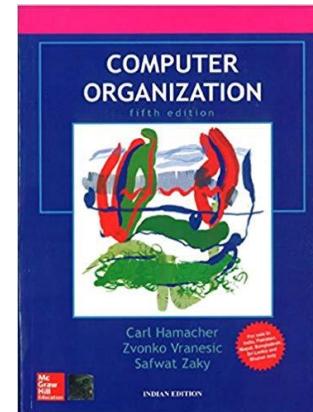
Pay on  
Delivery



10 Days  
Returnable



Amazon  
Delivered

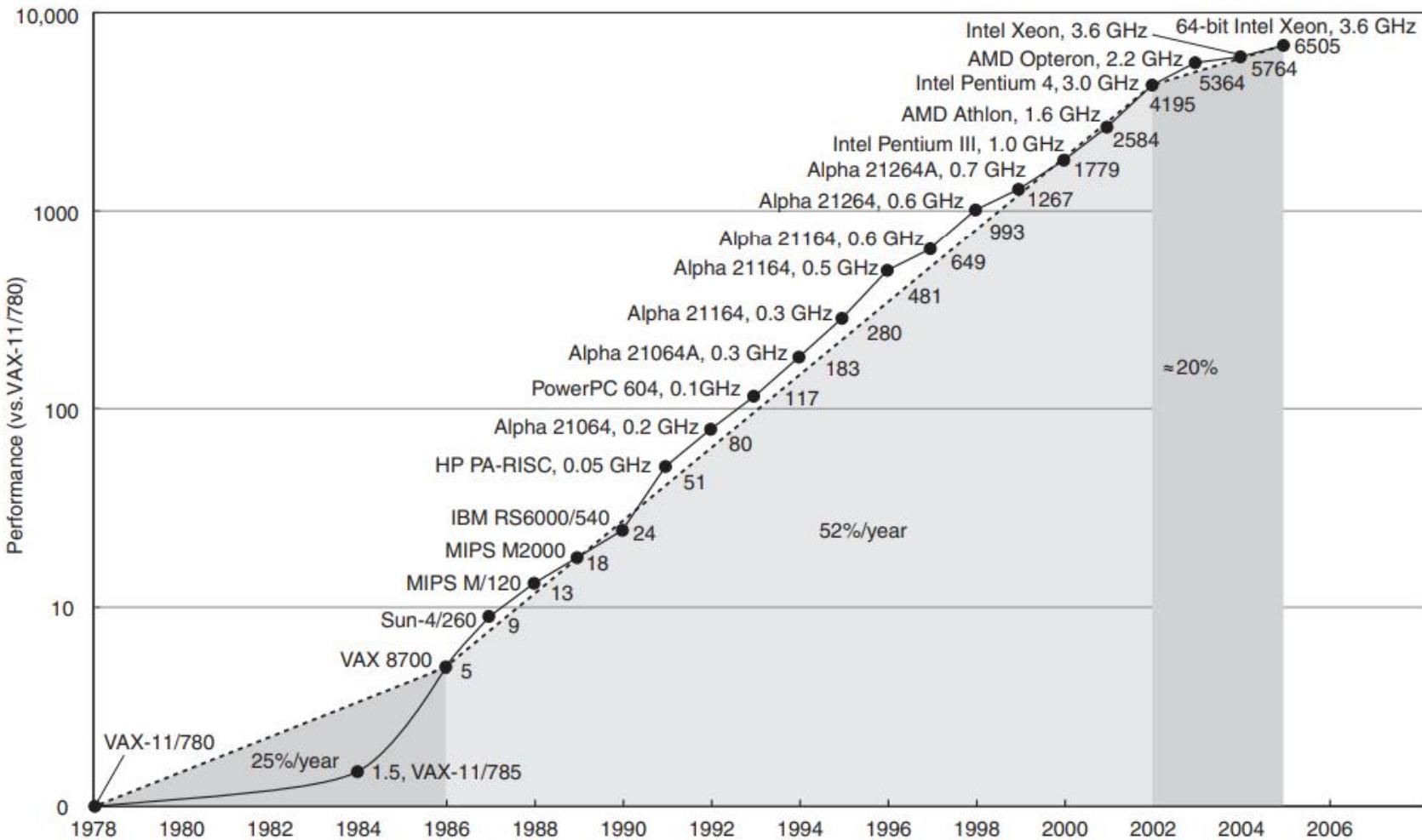


# Logic Gates

- Implements Boolean functions (thus performs logical operations)
- Implemented using Transistors

**Microprocessors contain millions of logic gates.**

# Processor Performance



# Moore's Law

**The number of transistors on a microchip doubles every two years, though the cost of computers is halved.**

[CES 2019: Moore's Law is dead, says Nvidia's CEO](#)

CNET - 10-Jan-2019

Intel, for its part, doesn't think **Moore's Law is dead**. Companies are just finding new ways to keep it going, like Intel's new 3D chip stacking.

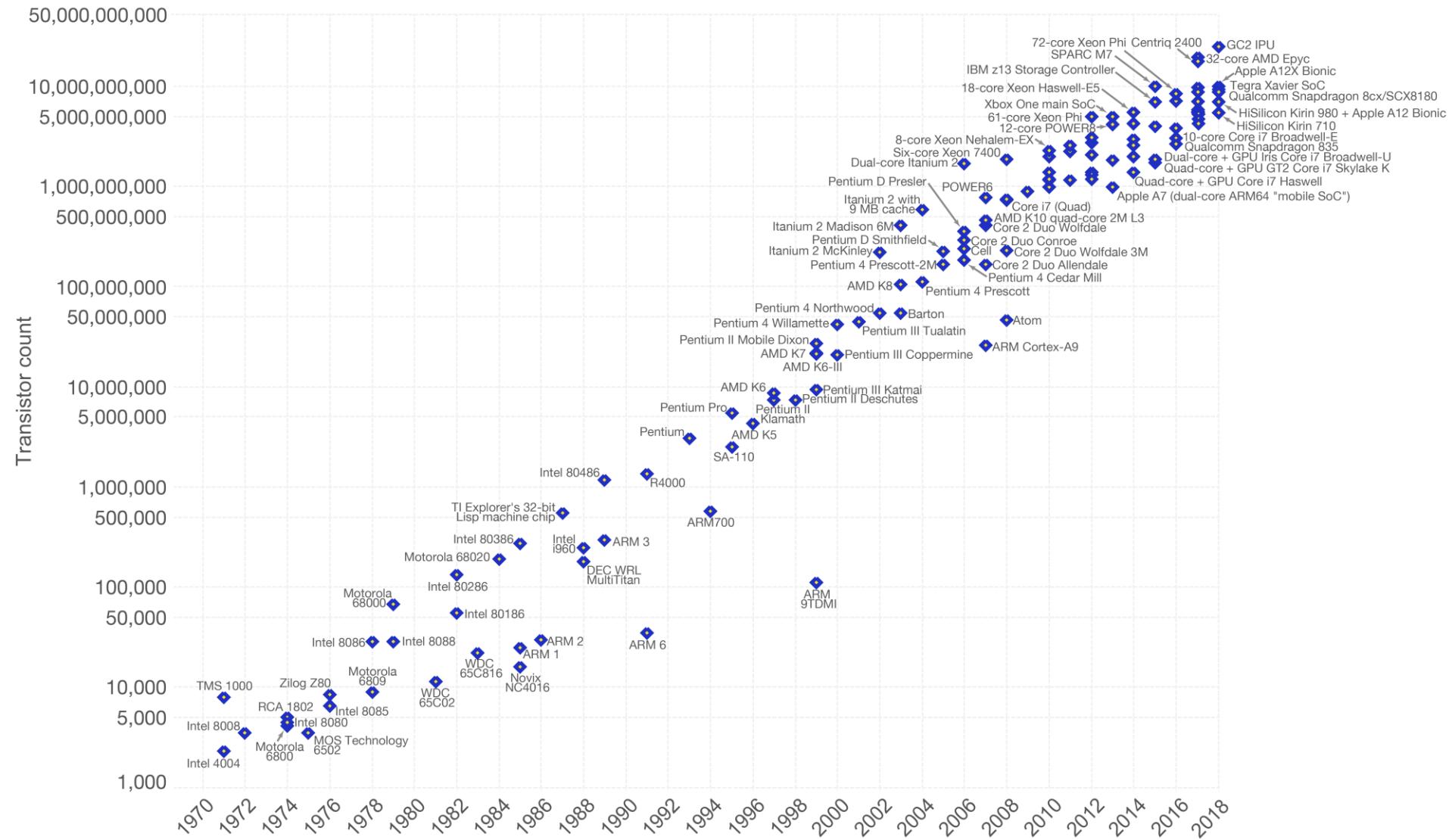
[Moores Law is far from death, according to Intel's Jim Keller](#)

TweakTown - 10-Dec-2019

"The death of **Moores Law** is coming and will limit how far we can go ... explanation of what **Moores law** is, from Intel's co-founder, engineer, ...

# Moore's Law – The number of transistors on integrated circuit chips (1971-2018)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are linked to Moore's law.



# Multi-Core Processors

- Two or more separate processing units (called cores)
- Enhances parallel processing



intel core duo  
(has 2 cores, 2.66 GHz)



intel core i7  
(has 4 cores, 4 GHz)

# Quality Up

**What do we achieve when we use p  
processors?**

$$\text{Quality Up} = \frac{\text{quality on } p \text{ processors}}{\text{quality on 1 processor}}$$

Read Section 1.1 of Jan Verschelde's book on "[Introduction to Supercomputing](#)".

# Can we use multiple processors?

- Amdahl's law
  - Let  $R$  be the fraction of the operations which cannot be parallelized. The speedup with  $p$  processors is bound by
$$\frac{1}{R + \frac{1-R}{p}}.$$
- Example
  - Say, 10% cannot be parallelized, and we have 8 processors. Best speedup =  $\frac{1}{1/10 + \frac{1-1/10}{8}} \approx 4.7x.$

# Multiple Processors for Speedup

- Amdahl's law
  - Let R be the fraction of the operations which cannot be parallelized. The speedup with p processors is bound by

$$\frac{1}{R + \frac{1-R}{p}}.$$

Speed up in terms of  
problem size.

- Example
  - Say, 10% cannot be parallelized, and we have 8 processors. Best speedup =  $\frac{1}{1/10 + \frac{1-1/10}{8}} \approx 4.7x$ .
  - What happens if we had infinite processors?



# Quiz

- Say, 10% of the operations cannot be parallelized.  
What happens if we had infinite processors?
- Answer: 10x.

# Speedup

Speed up in terms of time.

- Gustafson's Law
  - If  $s$  is the fraction of serial operations in a parallel program run on  $p$  processors, then the scaled speedup is bounded by  $p + (1 - p)s$ .
- Example
  - Say, all other seven processors are kept idle while one processor completes 5% work, scaled speedup =  $8 + (1 - 8) * 0.05 = 7.65$ .

Our ability to parallelize determines the successful use of multi-core processors.

# Supercomputer

- A computing system that provides close to the **best** currently **achievable sustained performance** on demanding computational problems.

How do supercomputers achieve such performance levels?

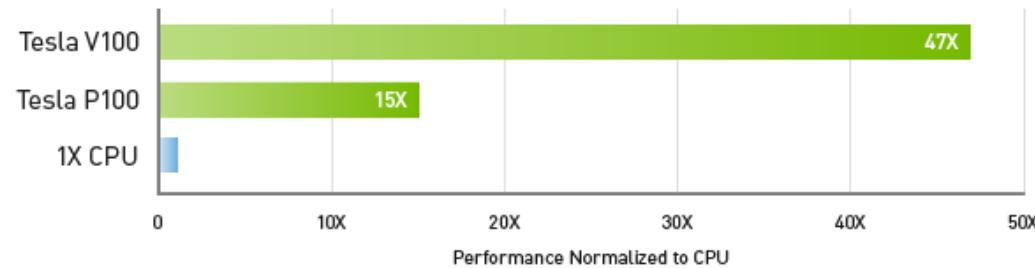
# GPUs and GPGPUs

- Graphics Processing Unit (GPU)
  - Massive parallelization
  - Thousands of cores
  - Originally created for the gaming industry
- General Purpose GPU (GPGPU)
  - Architecture allows for programming (Example: Compute Unified Device Architecture (CUDA) on NVIDIA GPGPUs).
- Performance is measured in FLOP (Floating Point Operation)
  - sometimes, FLOPS (floating point operations per second)

# CPU vs. GPU

- Say, two floating point operations could be performed in a clock cycle,
  - 3 GHz processor → 6 gigaflop per second.
- Top GPUs achieve petaflop per second.
  - Achieved through an array of cores (V100 has 5120 cores)

47X Higher Throughput Than CPU Server on Deep Learning Inference



Workload: ResNet-50 | CPU: 1X Xeon E5-2690v4 @ 2.6 GHz | GPU: Add 1X Tesla P100 or V100

# My System



My Lenovo X390 uses  
Intel® Core™ i7-8565U CPU @ 1.8 GHz

**4 cores only!** ☹

# Deep Blue

- Beat Chess World Champion Garry Kasparov in 1997



259th most powerful supercomputer.  
Achieved 11.38 GFLOPS.

# IBM Watson and Jeopardy Game, 2011

- Cluster of **90 servers** each having **3.5GHz eight-core processor** and **16 TB of RAM**.
- Equivalent to 80 Teraflops (a slow supercomputer by today's standards).



# Trivia

- Can you name the fastest supercomputer as of date?
  - How much data can it store?
  - How fast is it?

# Trivia

- Can you name the fastest supercomputer as of date? **IBM SUMMIT**
  - How much data can it store? **250 PB**
  - How fast is it? **200 petaflops**

# How fast is 200 petaflops?

Uses NVIDIA Tesla V100 GPU – How fast is its 200 petaflops?

"If every person on Earth completed one calculation per second, it would take the world population 305 days to do what Summit can do in 1 second" - Oak Ridge National Laboratory.

That is 200 quadrillion calculations in one second!

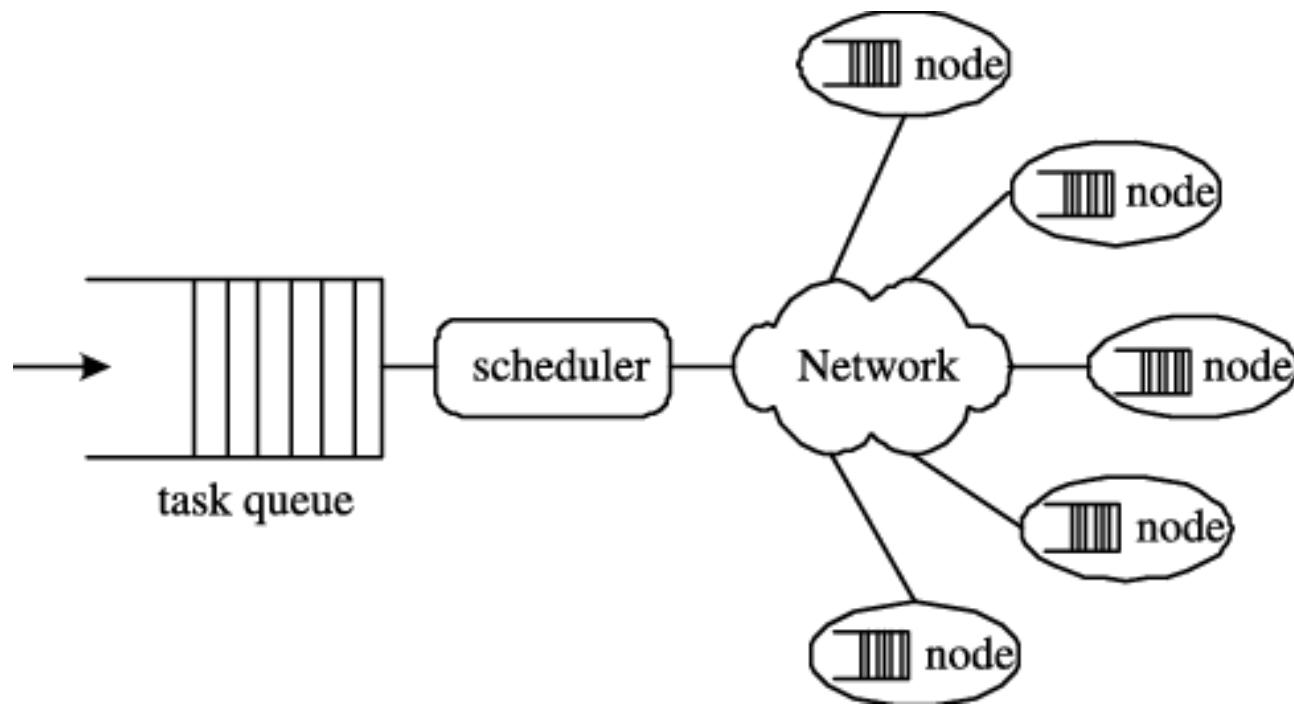
# Limitations and Opportunities

- Supercomputers
  - are too expensive
  - still far away from achieving desirable speedups
  - need skilled programming (distributed computing algorithms, parallelizable code)
- But,
  - GPUs are becoming commonplace
  - High Performance Clusters are increasingly available

# The Central Question!!!!

**Instead of using supercomputers,  
can we put commodity hardware  
into a cluster and achieve speedup?**

# Computing with Commodity Hardware – Distributed Computing



Sun et al., Dynamic Task Flow Scheduling for Heterogeneous Distributed Computing, 2007.

# Cluster Computing

- Multiple nodes acting as a single node
- High Performance Computing (HPC) clusters are becoming increasingly popular



Sun Microsystems, Solaris Cluster

Source: chrisdag, flickr.

# Grid Vs. Cluster Computing

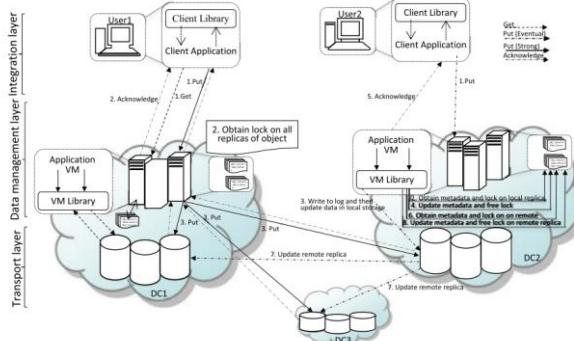
- Clusters have homogenous set of nodes.
- Grid refers to heterogenous systems.

# All Roads Lead To.... Cloud

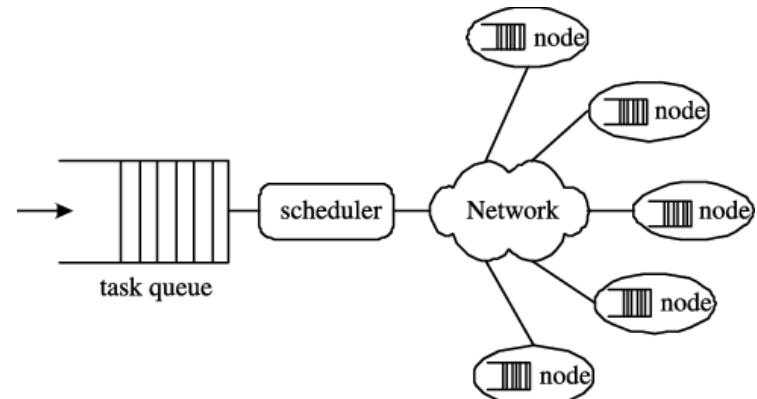
We are in the Big Data era!

Two kinds of Big Data Opportunities

Storage



Processing



# Imperatives for Big Data Platform

	Big Data Platform Imperatives	Technology Capability
1	Discover, explore and navigate big data sources	 Federated Discovery, Search and Navigation
2	Extreme Performance – run analytics closer to data	 Massively Parallel Processing Analytic appliances
3	Manage and analyze unstructured data	 Hadoop File System / MapReduce Text Analytics
4	Analyze data in real time	 Stream Computing
5	Rich library of analytical functions and tools	 In-Database Analytics Libraries Big Data visualization
6	Integrate and govern all data sources	 Integration, Data Quality, Security, Lifecycle Management, MDM

Source: <https://www.ibmbigdatahub.com/blog/part-ii-big-data-platform-manifesto>

# Key Questions

- How to setup and manage such clusters?
- How to achieve reliability, availability, scalability, ...?
- How to build services on cloud?

## Apache Hadoop

Open source platform - reliable, scalable, - distributed processing of large data sets - built on clusters of commodity computers.

# Remember...

- Presentation registration deadline is approaching.
- Register yourself on moodle.
- Do not ignore the readings.

# CLOUDERA

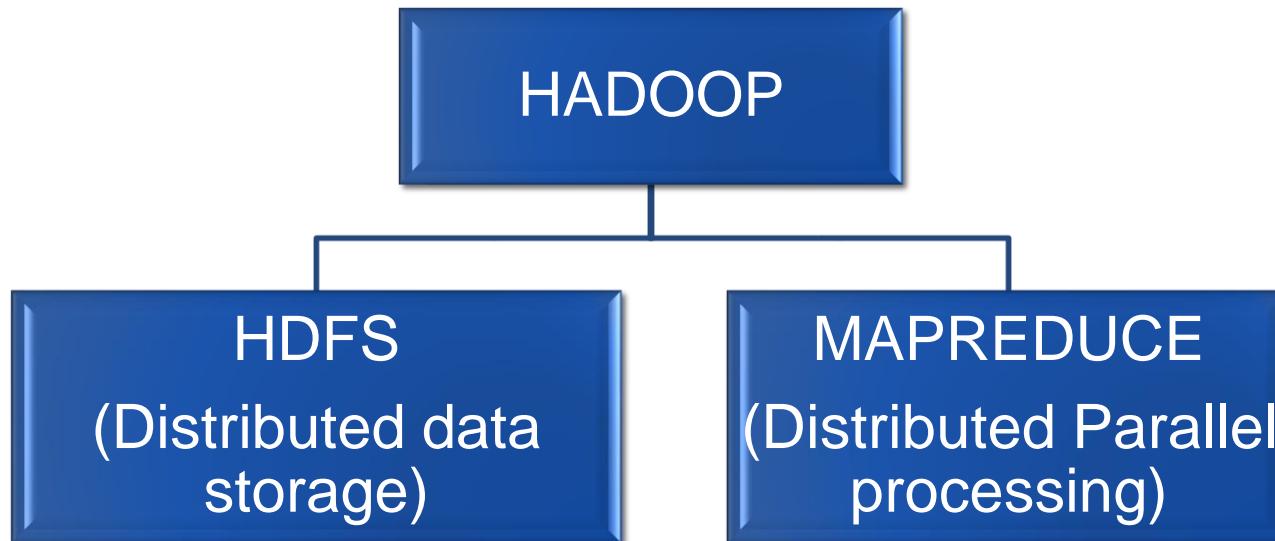
---

A Quick Overview

by Suchitra Jayaprakash  
[suchitra@cmi.ac.in](mailto:suchitra@cmi.ac.in)

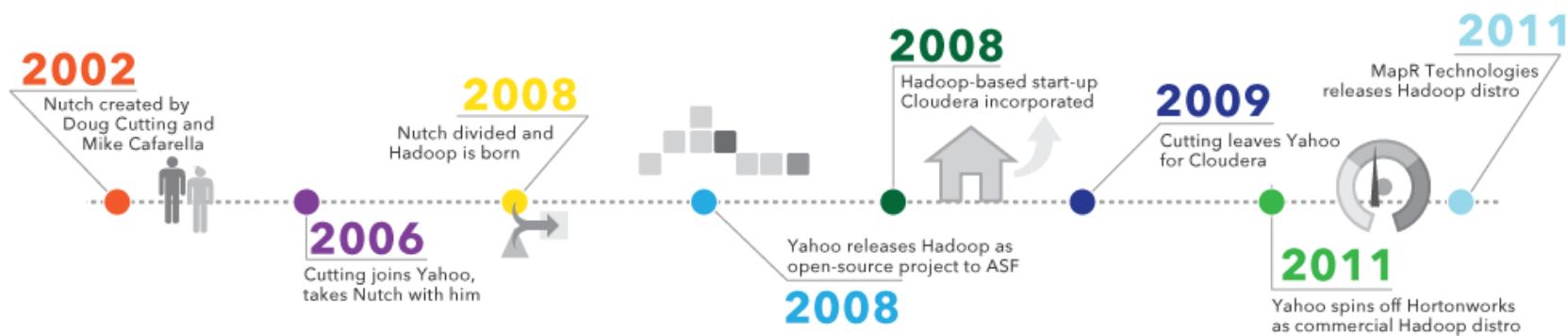
# Apache Hadoop

- Hadoop is open source software framework used for processing data on distributed commodity computing environment.



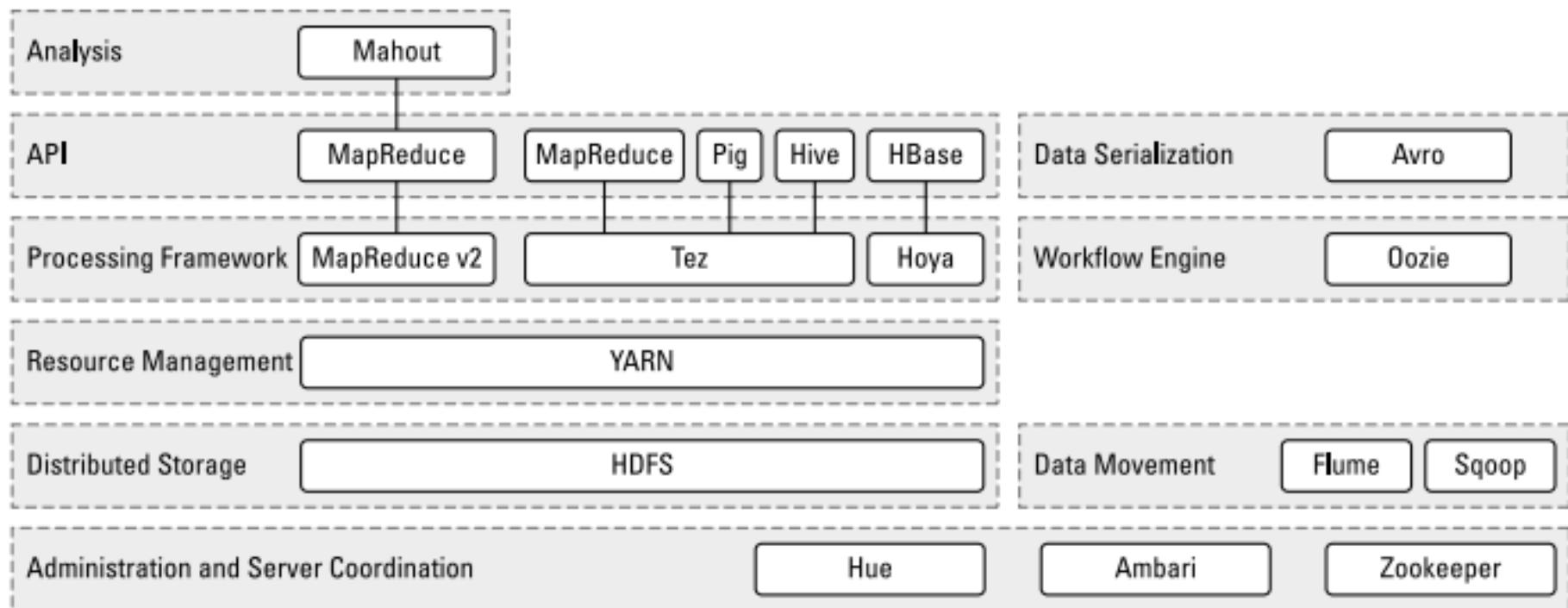
# Apache Hadoop

- It is a java based software managed by Apache Software Foundation.
- Hadoop is designed to scale up from single server to thousands of machines.
- Doug Cutting & Mike Cafarella are co-founders of Hadoop. It is based on google's white paper on Google File System & mapreduce.



(source: [https://www.sas.com/en\\_in/insights/big-data/hadoop.html](https://www.sas.com/en_in/insights/big-data/hadoop.html))

# Hadoop Ecosystem



(source: Hadoop for Dummies)

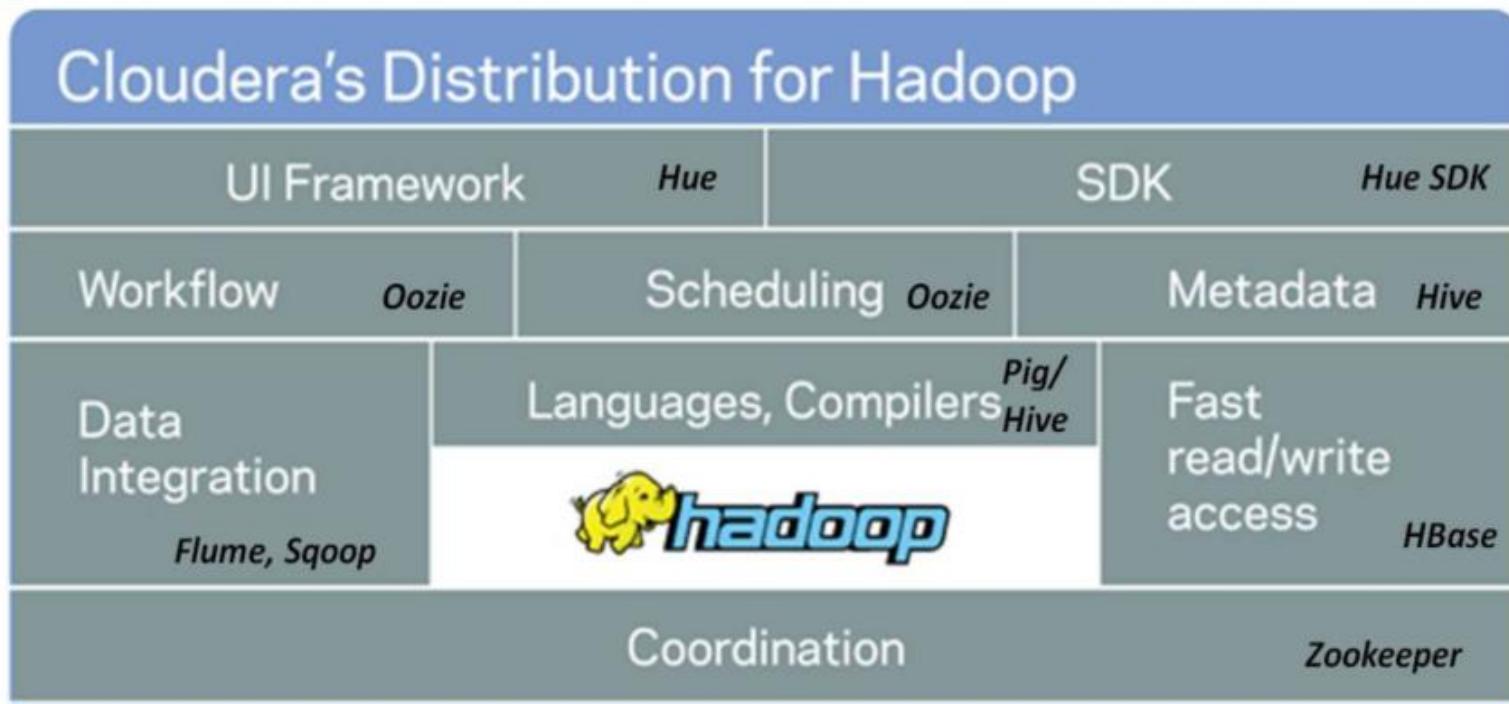
# HADOOP DISTRIBUTION

- Customisation for industry needs resulted in emergence of commercial distribution.
- Base version Apache Hadoop + features (UI , Security , Monitoring , logging, Support).
- Top Vendors offering Big Data Hadoop solution :
  - Cloudera **CLOUDERA**
  - Hortonworks  **Hortonworks®**
  - MapR  **MAPR.**
  - Amazon Web Services Elastic MapReduce Hadoop Distribution  **amazon web services™**
  - Microsoft Azure's HDInsight -Cloud based Hadoop Distribution  **Microsoft Azure**
  - IBM InfoSphere Insights 

# CLOUDERA

- Founded in 2008 by three engineers from Google, Yahoo! and Facebook (Christophe Bisciglia, Amr Awadallah and Jeff Hammerbacher).
- Major code contributor of Apache Hadoop ecosystem.
- First company to develop and distribute Apache Hadoop based software in March 2009.
- Additional feature includes user interface, security, interface for third party application integration.
- Offers customer support for installing , configuring , optimising Cloudera distribution through its enterprise subscription service.
- Provides a proprietary Cloudera Manager for easy installation , monitoring & trouble shooting.
- In 2016, Cloudera was ranked #5 on the Forbes Cloud 100 list  
(source: Cloudera wiki)

# CLOUDERA DISTRIBUTION



An illustration of Cloudera's open-source Hadoop distribution (source: cloudera website).

# CLOUDERA QUICKSTART

- Cloudera QuickStart VM is a sandbox environment of CDH.
- It gives a hands-on experience with CDH for demo and self-learning purposes.
- CDH deployed via Docker containers or VMs, are not intended for production use. Latest version is QuickStarts for CDH 5.13.
- System Requirement: Cloudera's 64-bit VMs require a 64-bit host OS and a virtualization product that can support a 64-bit guest.
- The amount of RAM required by the VM (separate from system RAM) varies by the run-time option you choose:

CDH and Cloudera Manager Version	RAM Required by VM
CDH 5 (default)	4+ GiB*
Cloudera Express	8+ GiB*
Cloudera Enterprise (trial)	12+ GiB*

\*Minimum recommended memory.

(source: Cloudera website)

# Quiz

Q) Which of the following is false?

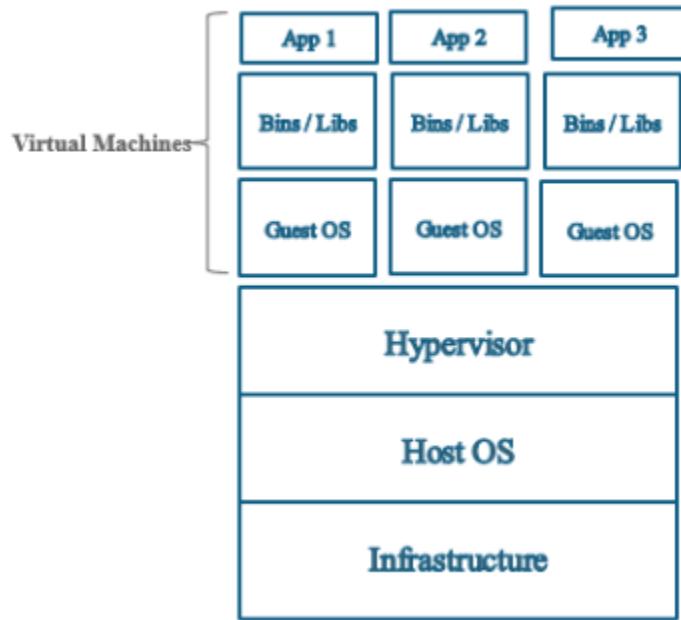
- A. Cloudera products and solutions enable you to deploy and manage Apache Hadoop and related projects.
- B. Cloudera QuickStart VM is a sandbox environment of CDH.
- C. CDH contains all the products and frameworks belonging to the hadoop ecosystem.
- D. Hadoop is open source software framework used for processing data on distributed commodity hardware.

# DEPLOYMENT MODES - DOCKER

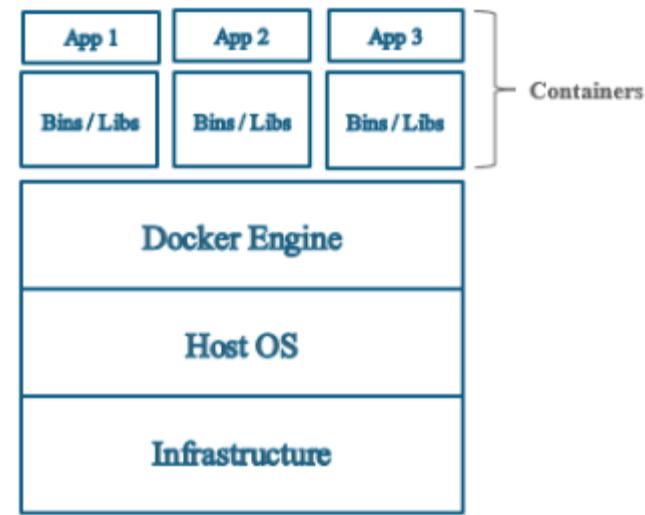


- Docker is an open source tool that uses containers to create, deploy, and manage distributed applications.
- Developers use containers to create packages for applications that include all libraries that are needed to run the application in isolation.

# DEPLOYMENT MODES : VM vs DOCKER



**Virtual Machine / Virtual Box**



**Docker Container**

- Virtual machine has its guest operating system above the host operating system.
- Docker containers share the host operating system.

# Virtual Machine vs Docker Container



```
MINGW64:/c/Program Files/Docker Toolbox
Setting OOZIE_HTTP_HOSTNAME: quickstart.cloudera
Setting OOZIE_HTTP_PORT: 11000
Setting OOZIE_ADMIN_PORT: 11001
Using OOZIE_HTTPS_PORT: 11443
Setting OOZIE_BASE_URL: http://quickstart.cloudera:11000/oozie
Using CATALINA_BASE: /var/lib/oozie/tomcat-deployment
Setting OOZIE_HTTPS_KEYSTORE_FILE: /var/lib/oozie/.keystore
Using OOZIE_HTTPS_KEYSTORE_PASS: password
Setting OOZIE_INSTANCE_ID: quickstart.cloudera
Setting CATALINA_OUT: /var/log/oozie/catalina.out
Using CATALINA_PID: /var/run/oozie/oozie.pid

Using CATALINA_OPTS: -Doozie.https.port=11443 -Doozie.https.keystore.pass=password -Xmx1024m -Dderby.stream.error.file=/var/log/oozie/derby.log
Adding to CATALINA_OPTS: -Doozie.home.dir=/usr/lib/oozie -Doozie.config.dir=/etc/oozie/conf -Doozie.log.dir=/var/log/oozie -Doozie.data.dir=/var/lib/oozie -Doozie.instance.id=quickstart.cloudera -Doozie.config.file=oozie-site.xml -Doozie.log4j.file=oozie-log4j.properties -Doozie.log4j.reload=10 -Doozie.http.hostname=quickstart.cloudera -Doozie.admin.port=11001 -Doozie.http.port=11000 -Doozie.https.port=11443 -Doozie.base.url=http://quickstart.cloudera:11000/oozie -Doozie.https.keystore.file=/var/lib/oozie/.keystore -Doozie.https.keystore.pass=password -Djava.library.path=/usr/lib/hadoop/lib/native

Using CATALINA_BASE: /var/lib/oozie/tomcat-deployment
Using CATALINA_HOME: /usr/lib/bigtop-tomcat
Using CATALINA_TMPDIR: /var/lib/oozie/tmp
Using JRE_HOME: /usr/java/jdk1.7.0_67-cloudera
Using CLASSPATH: /usr/lib/bigtop-tomcat/bin/bootstrap.jar
Using CATALINA_PID: /var/run/oozie/oozie.pid
Starting Solar server daemon: [ OK ]
Using CATALINA_BASE: /var/lib/solr/tomcat-deployment
Using CATALINA_HOME: /usr/lib/bigtop-tomcat
Using CATALINA_TMPDIR: /var/lib/solr/tmp
Using JRE_HOME: /usr/java/jdk1.7.0_67-cloudera
Using CLASSPATH: /usr/lib/solr/.../bigtop-tomcat/bin/bootstrap.jar
Using CATALINA_PID: /var/run/solr/solr.pid
Started Impala Catalog Server <catalogd>: [ OK ]
Started Impala Server <impalad>: [ OK ]
[root@quickstart ~]#
```

# QUICKSTART : DOCKER INSTALL

- The Cloudera Docker image is a single-host deployment of the Cloudera open-source distribution.
- Single Node Hadoop Cluster has only a single machine
  - DataNode, NameNode run on the same machine
- Multi-Node Hadoop Cluster will have more than one machine
  - DataNode, NameNode run on different machines.

# QUICKSTART : DOCKER INSTALL

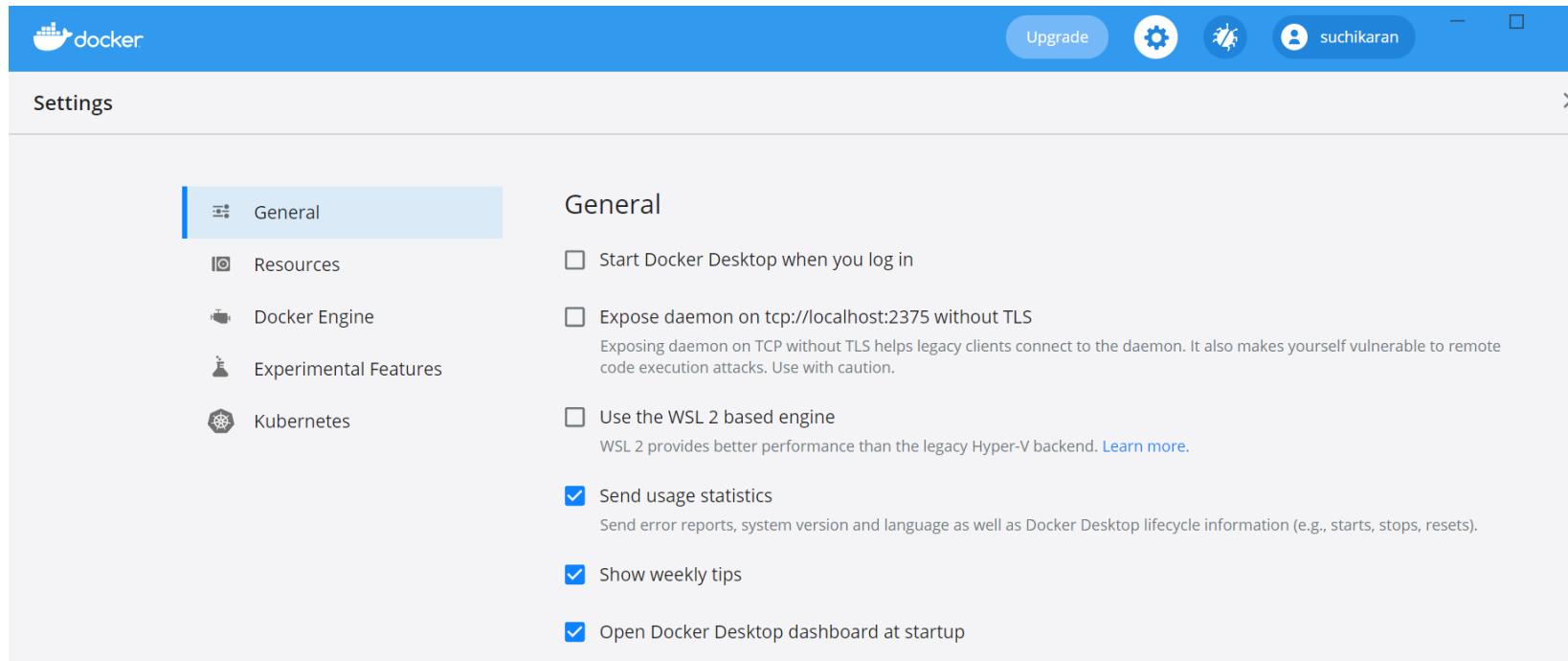
- **Installation Steps for Windows :**

1. **Install Docker :**

- Sign up to <https://docs.docker.com/>
- Follow instructions at <https://docs.docker.com/docker-for-windows/install/>
- For Windows 10 64-bit Home , Pro, Enterprise, or Education (Build 15063 or later) : Install Docker Desktop.
- For Other Windows OS :  
Install Docker Toolbox (refer below link for instructions.  
[https://docs.docker.com/toolbox/toolbox\\_install\\_windows/](https://docs.docker.com/toolbox/toolbox_install_windows/))

# QUICKSTART : DOCKER INSTALL

- Don't select WSL2 while installing docker. Cloudera Quick start VM is not compatible.



The screenshot shows the Docker Desktop settings interface. The top navigation bar includes the Docker logo, an 'Upgrade' button, a gear icon, a bug icon, and a user profile for 'suchikaran'. Below the header, the word 'Settings' is displayed. On the left, a sidebar lists categories: General (selected and highlighted in blue), Resources, Docker Engine, Experimental Features, and Kubernetes. The main content area is titled 'General' and contains the following configuration options:

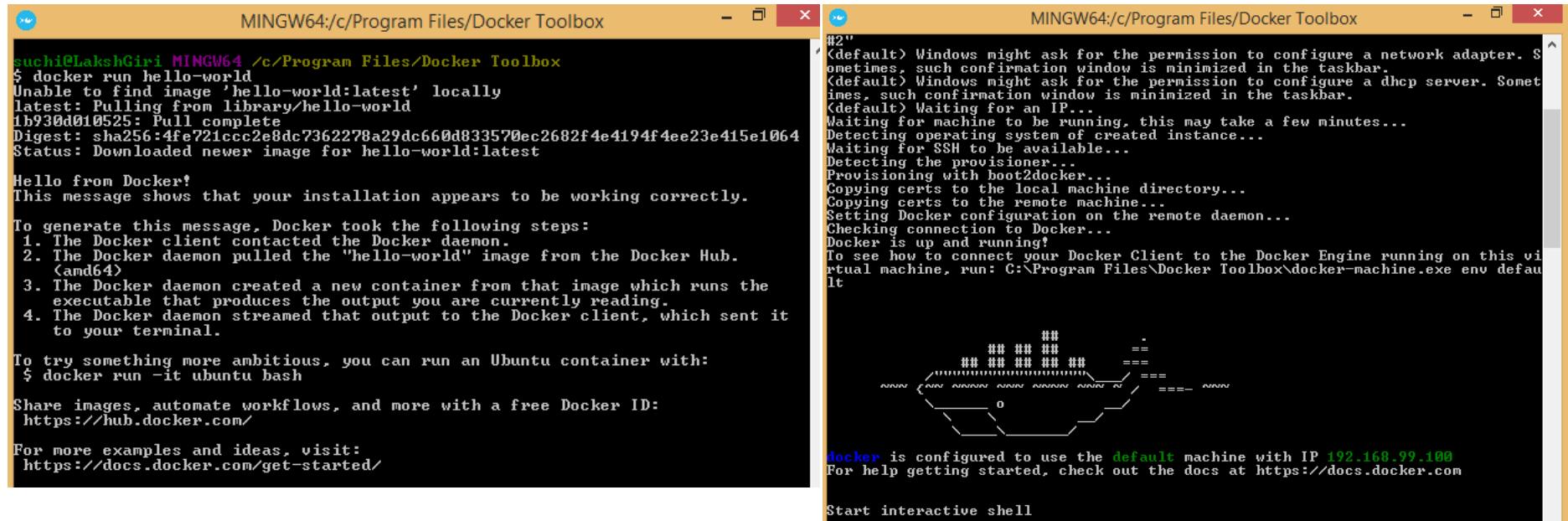
- Start Docker Desktop when you log in
- Expose daemon on tcp://localhost:2375 without TLS

Exposing daemon on TCP without TLS helps legacy clients connect to the daemon. It also makes yourself vulnerable to remote code execution attacks. Use with caution.
- Use the WSL 2 based engine

WSL 2 provides better performance than the legacy Hyper-V backend. [Learn more](#).
- Send usage statistics

Send error reports, system version and language as well as Docker Desktop lifecycle information (e.g., starts, stops, resets).
- Show weekly tips
- Open Docker Desktop dashboard at startup

# QUICKSTART : DOCKER INSTALL



The image shows two terminal windows side-by-side, both titled "MINGW64:/c/Program Files/Docker Toolbox".

**Terminal 1 (Left):**

```
suchi@LakshGiri MINGW64 ~/c/Program Files/Docker Toolbox
$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:4fe721ccc2e8dc7362278a29dc660d833570ec2682f4e4194f4ee23e415e1064
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
 executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
 to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

**Terminal 2 (Right):**

```
#2"
<default> Windows might ask for the permission to configure a network adapter. Sometimes, such confirmation window is minimized in the taskbar.
<default> Windows might ask for the permission to configure a dhcp server. Sometimes, such confirmation window is minimized in the taskbar.
<default> Waiting for an IP...
Waiting for machine to be running, this may take a few minutes...
Detecting operating system of created instance...
Waiting for SSH to be available...
Detecting the provisioner...
Provisioning with boot2docker...
Copying certs to the local machine directory...
Copying certs to the remote machine...
Setting Docker configuration on the remote daemon...
Checking connection to Docker...
Docker is up and running!
To see how to connect your Docker Client to the Docker Engine running on this virtual machine, run: C:\Program Files\Docke
lt



docker is configured to use the default machine with IP 192.168.99.100
For help getting started, check out the docs at https://docs.docker.com

Start interactive shell
```

- To check docker installation is proper , type below command in docker terminal.

**docker run hello-world**

- If you get above ouput in the terminal then docker installation is fine.

# QUICKSTART : DOCKER INSTALL

The screenshot shows the Docker Quickstart Guide interface. On the left, there's a sidebar with four numbered steps: 1. Clone (selected), 2. Build, 3. Run, 4. Share. The main content area has a title "First, clone a repository". Below it, text says: "The *Getting Started* project is a simple GitHub repository which contains everything you need to build an image and run it as a container." A note below says: "Clone the repository by running Git in a container." A code block shows the command: 

```
docker run --name repo alpine/git clone \
https://github.com/docker/getting-started.git  >>
docker cp repo:/git/getting-started/ .
```

 At the bottom of this section is a "Next Step" button. On the right, there's a Windows PowerShell window showing the command execution and its output.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\suchi> docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
b8dfde127a29: Pull complete
Digest: sha256:308866a43596e83578c7dfa15e27a73011bdd402185a84c
5cd7f32a88b501a24
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
     (amd64)
 3. The Docker daemon created a new container from that image
    which runs the
      executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client,
    which sent it
      to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

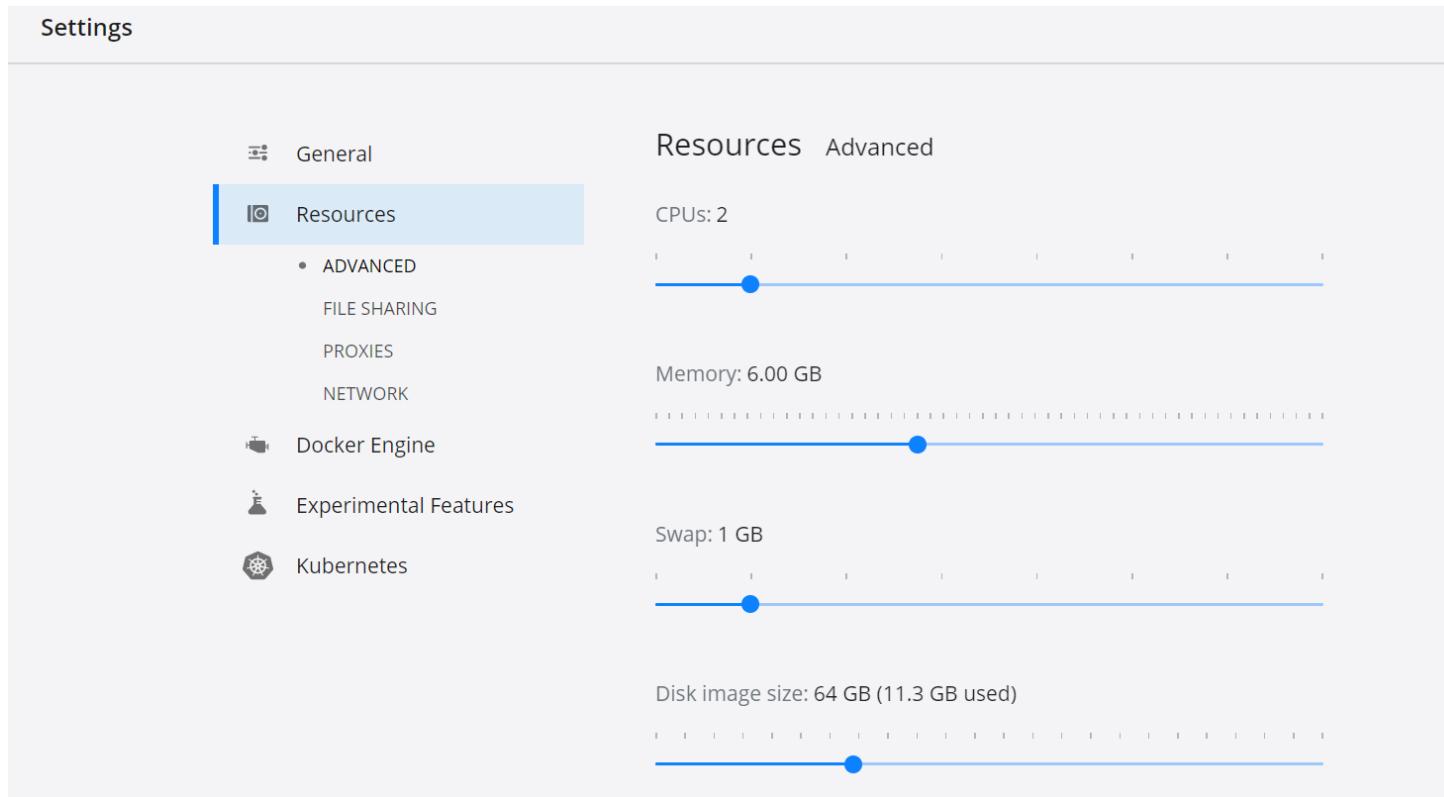
PS C:\Users\suchi>
```

- Docker for Desktop output
- For windows 10 : Run docker command in powershell or command prompt

# QUICKSTART : DOCKER INSTALL

## 2. Docker Desktop: Update Docker memory

Under setting select Resources and update CPU & Memory as mentioned below:



# QUICKSTART : DOCKER INSTALL

## 2. Docker Toolbox : Update Docker memory (optional)

2. Create a new VM with 1 CPUs and 4GB of memory (recommended).

3. Run the following command in docker terminal:

- Remove the default vm.

**docker-machine rm default**

- Re-create the default vm.

**docker-machine create -d virtualbox --virtualbox-cpu-count=1 --virtualbox-memory=4096 --virtualbox-disk-size=50000 default**

options	Description
--virtualbox-cpu-count	number of cpus
--virtualbox-memory	amount of RAM
-virtualbox-disk-size	amount of disk space

# QUICKSTART : DOCKER INSTALL

## 3. Install Cloudera Quickstart:

Type following command in the docker terminal to import Cloudera Quickstart image from Docker Hub:

***docker pull cloudera/quickstart:latest***

(refer link <https://hub.docker.com/r/cloudera/quickstart>)

```
$ docker pull cloudera/quickstart:latest
latest: Pulling from cloudera/quickstart
Image docker.io/cloudera/quickstart:latest uses outdated schema1 manifest format
Please upgrade to a schema2 image for better future compatibility. More information at https://docs.docker.com/registry/spec/deprecated-schema-v1/
ld00652ce734: Downloading 39.28MB/4.444GB
```

Cloudera quickstart download will take a while to complete. After download is complete , type following in terminal :

***docker images***

```
suchi@LakshGiri MINGW64 /c/Program Files/Docker Toolbox
$ docker images
REPOSITORY          TAG           IMAGE ID            CREATED             SIZE
cloudera/quickstart    latest        4239cd2958c6      3 years ago       6.34GB
```

# QUICKSTART : DOCKER INSTALL

## 4. Run Cloudera Quickstart container

- Click on “Docker Quickstart Terminal” Icon and Type below command in docker terminal to start Cloudera Quickstart

```
docker run --hostname=quickstart.cloudera --privileged=true -t -i -p 8888:8888 -p 8080:8080 -p 8088:8088 -p 7180:7180 -p 50070:50070 cloudera/quickstart /usr/bin/docker-quickstart
```

Options	Required	Description
--hostname=quickstart.cloudera	Yes	Pseudo-distributed configuration assumes this as hostname.
--privileged=true	Yes	For HBase, MySQL-backed Hive metastore, Hue, Oozie, Sentry, and Cloudera Manager.
-t	Yes	Allocate a pseudoterminal. Once services are started, a Bash shell takes over. This switch starts a terminal emulator to run the services.
-i	Yes	Enable interactive terminal i.e. If you want to use the terminal, either immediately or connect to the terminal later.
--publish-all=true	No	opens up all the host ports to the docker ports
-p 8888	Yes - Recommended	Map the Hue port in the guest to port on the host.
-p [PORT]	No	Map any other ports in the guest to port on the host.
cloudera/quickstart	Yes	Name of image which run as new container
/usr/bin/docker-quickstart	Yes	Start all CDH services, and then run a Bash shell.

# QUICKSTART : DOCKER INSTALL

List of common ports used in Cloudera :

Port	Purpose
8888	Hue web interface
50070	Name node web interface
8088	job tracker :- yarn
7180	Cloudera manager
80	Cloudera examples

## 5. Host – Guest port mapping

- Open new docker terminal & type below command.

**docker ps**

CONTAINER ID	IMAGE	COMMAND	CREATED	NAMES
STATUS				
b636a46d51d0	cloudera/quickstart	"/usr/bin/docker-qui"	4 minutes ago	crazy_proskur-iakova
Up 4 minutes		0.0.0.0:7180->7180/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8888->8888/tcp, 0.0.0.0:50070->50070/tcp, 0.0.0.0:8080->80/tcp		

- Copy the docker container ID.
- Type below to check memory allocation

**docker stats [CONTAINER ID]**

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT
MEM %	NET I/O	BLOCK I/O	PIDS
cde59eb01eeb	goofy_williamson	9.39%	3.362GiB / 3.856GiB
87.19%	2.39kB / 4.33kB	1.48GB / 59.4MB	1328

# QUICKSTART : DOCKER INSTALL

- Type below command and get see which Host port Hue and YARN are working.

**docker inspect [CONTAINER ID]**

- YARN is working on port  
**8088** inside the docker machine  
**8088** outside on host machine

Note : in case of docker tool box, host machine is mapped to ip address 192.168.99.100. Use url

<http://192.168.99.100:50070/>

For other docker install use localhost

<http://localhost:50070/>

```
"Ports": [
    "50070/tcp": [
        {
            "HostIp": "0.0.0.0",
            "HostPort": "50070"
        }
    ],
    "7180/tcp": [
        {
            "HostIp": "0.0.0.0",
            "HostPort": "7180"
        }
    ],
    "80/tcp": [
        {
            "HostIp": "0.0.0.0",
            "HostPort": "8080"
        }
    ],
    "8088/tcp": [
        {
            "HostIp": "0.0.0.0",
            "HostPort": "8088"
        }
    ],
    "8888/tcp": [
        {
            "HostIp": "0.0.0.0",
            "HostPort": "8888"
        }
    ]
],
```

- **Installation Steps for Ubuntu :** <https://medium.com/@dataakkadian/how-to-install-and-running-cloudera-docker-container-on-ubuntu-b7c77f147e03>

# QUICKSTART : DOCKER INSTALL

HUE- <http://localhost:8888/>

Default username / password : cloudera / cloudera

The screenshot shows the Hue UI with the title "Quick Start Wizard - Hue™ 3.9.0 - The Hadoop UI". The top navigation bar includes links for Apps, Password Express, and Entertainment. Below the navigation is a secondary menu with Query Editors, Data Browsers, Workflows, Search, and Security. The main content area displays the "About Hue" section, with "Quick Start" currently selected. A progress bar at the top of the content area shows four steps: Step 1: Check Configuration (highlighted in blue), Step 2: Examples, Step 3: Users, and Step 4: Go!. The main content below the progress bar says "Checking current configuration" and "Configuration files located in /etc/hue/conf.empty". It also states "All OK. Configuration check passed." At the bottom are "Back" and "Next" buttons, and a note: "Hue and the Hue logo are trademarks of Cloudera, Inc."

# QUICKSTART : DOCKER INSTALL

Name Node - <http://localhost:50070/>

The screenshot shows a web browser window with the URL `localhost:50070/dfshealth.html#tab-overview`. The browser interface includes standard navigation buttons (back, forward, search) and a toolbar with icons for Apps, Password Express, and Entertainment.

The main content area displays the HDFS Health Overview for a cluster named 'quickstart.cloudera:8020' (active). The page has a dark header bar with tabs for Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities.

### Overview 'quickstart.cloudera:8020' (active)

Started:	Fri Apr 16 20:56:39 +0530 2021
Version:	2.6.0-cdh5.7.0, rc00978c67b0d3fe9f3b896b5030741bd40bf541a
Compiled:	Thu Mar 24 00:06:00 +0530 2016 by jenkins from Unknown
Cluster ID:	CID-11ef0663-e698-48f8-bbee-7b664322ae19
Block Pool ID:	BP-1120155954-10.0.0.1-1459909528739

### Summary

Security is off.  
Safemode is off.  
992 files and directories, 913 blocks = 1,905 total filesystem object(s).  
Heap Memory used 152.06 MB of 263.5 MB Heap Memory. Max Heap Memory is 889 MB.

# QUICKSTART : DOCKER INSTALL

Yarn page - <http://192.168.99.100:8088/>

← → ⌂ ⌂ Not secure | 192.168.99.100:8088/cluster

Apps ⌂



## All Applications

Cluster Metrics														
	Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Vcores Used	Vcores Total	Vcores Reserved	Active Nodes	Decommission Nodes	
About Nodes	0	0	0	0	0	0 B	8 GB	0 B	0	8	0	1	0	

User Metrics for dr.who														
	Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Pending	Containers Reserved	Memory Used	Memory Pending	Memory Reserved	Memory Total	Nodes	Virtual Nodes	
	0	0	0	0	0	0	0	0 B	0 B	0 B	0 B	0	0	

Show 20 ▾ entries

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU Vcores	Allocated Memory MB

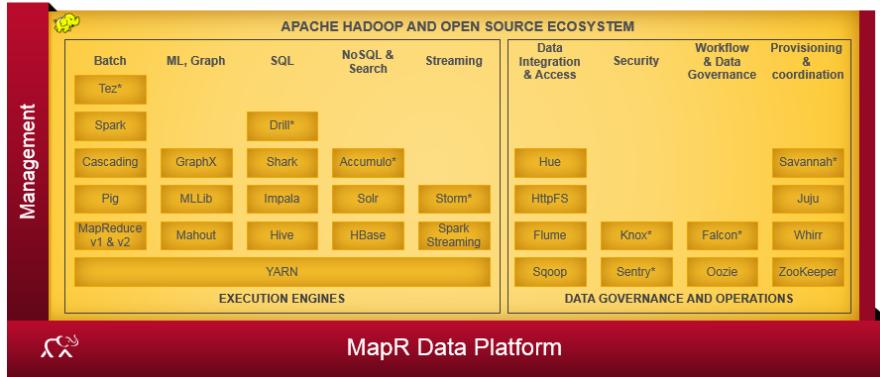
No data available in table

Showing 0 to 0 of 0 entries

Yarn is resource management layer of Apache Hadoop ecosystem.

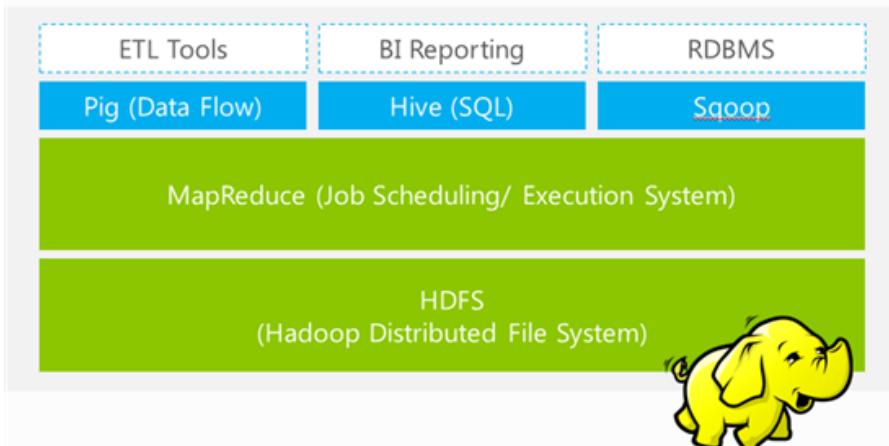
# Other Vendors

## MapR Distribution for Hadoop

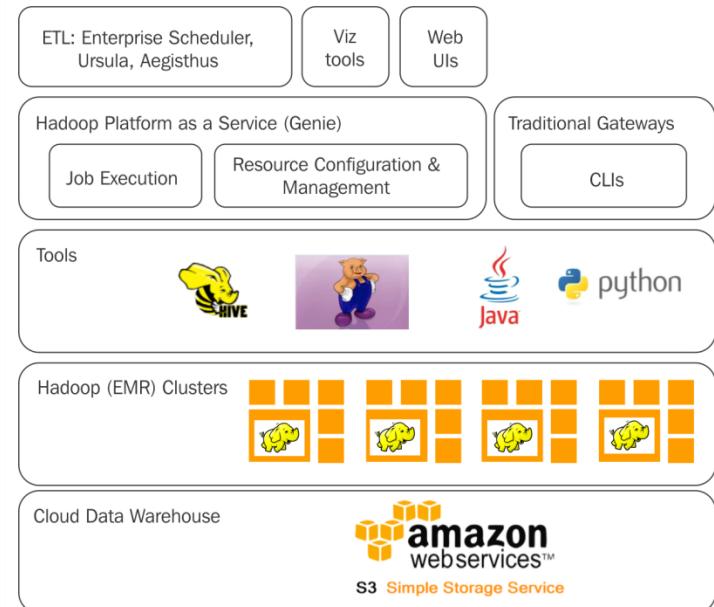


## Windows Azure HDInsight

### The Hadoop Ecosystem



## AWS EMR



**THANK YOU**

# DISTRIBUTED FILE SYSTEM

**Venkatesh Vinayakarao**

[venkateshv@cmi.ac.in](mailto:venkateshv@cmi.ac.in)

<http://vvtesh.co.in>

---

Chennai Mathematical Institute

---

**The ever-growing imbalance between computation and I/O is one of the fundamental challenges for current **petascale** and future **exascale** systems.** – Zhao and Raicu, Illinois Institute of Technology, 2013.

# What Comes Next?

byte

kilobyte

megabyte

gigabyte

??

???

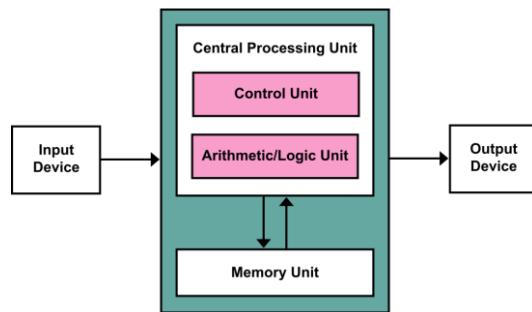
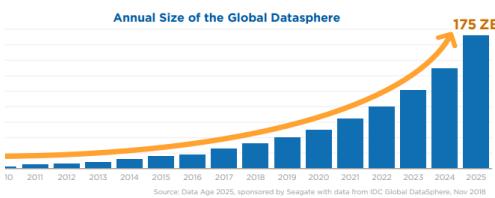
????

?????

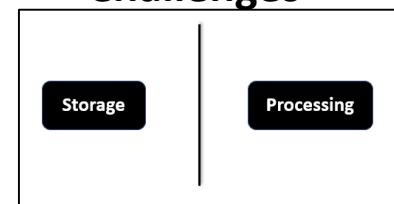
# Sizes

Name	Size
Byte	8 bits
Kilobyte	1024 bytes
Megabyte	1024 kilobytes
Gigabyte	1024 megabytes
Terabyte	1024 gigabytes
Petabyte	1024 terabytes
Exabyte	1024 petabytes
Zettabyte	1024 exabytes
Yottabyte	1024 zettabytes

# Recap

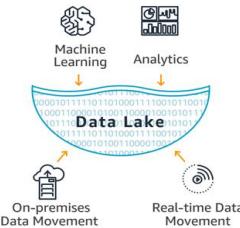


## Challenges

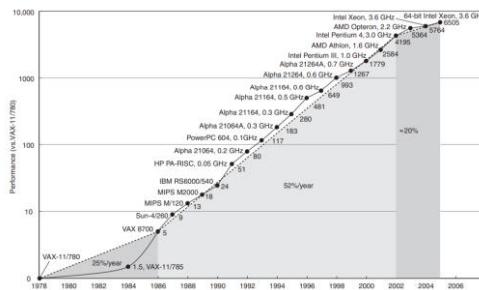


# Recap

## Data Storage

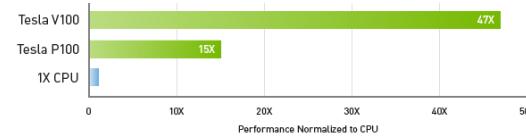


## Data Processing



CPU Performance

47X Higher Throughput Than CPU Server on Deep Learning Inference



GPU Performance

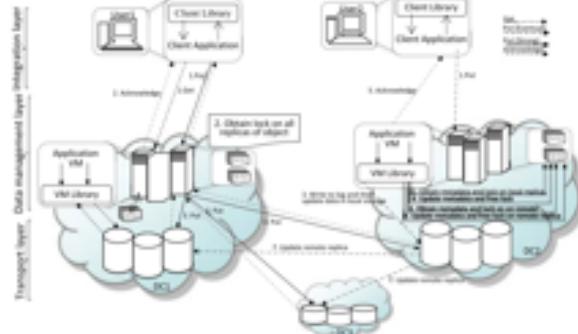


SuperComputers

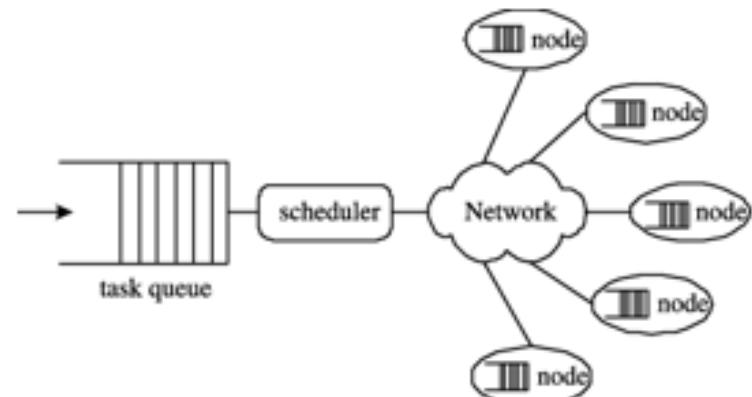
# Cloud Computing

Two kinds of Big Data Opportunities

## Storage



## Processing



So, we have the cloud. But, how to store and retrieve data? How to process jobs?

## **What is an operating system?**

Yarn is now the [Apache Hadoop Operating System](#)

### **Apache Hadoop**

Open source platform for reliable, scalable, distributed processing of large data sets, built on clusters of commodity computers.

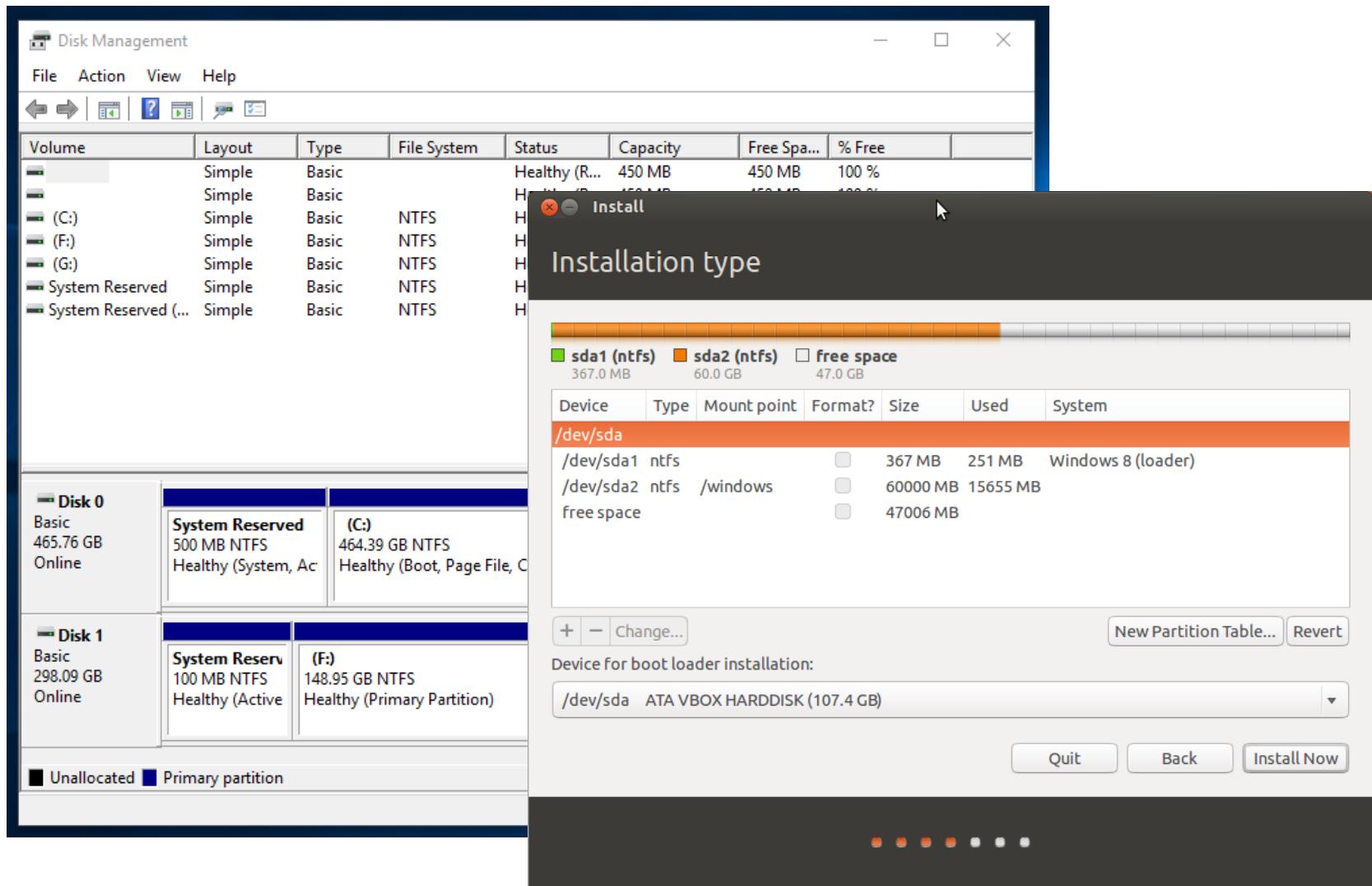
# Agenda

- File Systems
  - Introduction
  - File and Folders – How are they stored?
  - Windows/Unix/Miscellaneous File Systems
  - File Allocation Methods
  - Free Space Management
  - Compression
- Distributed File System
  - Hadoop Distributed File System (HDFS)

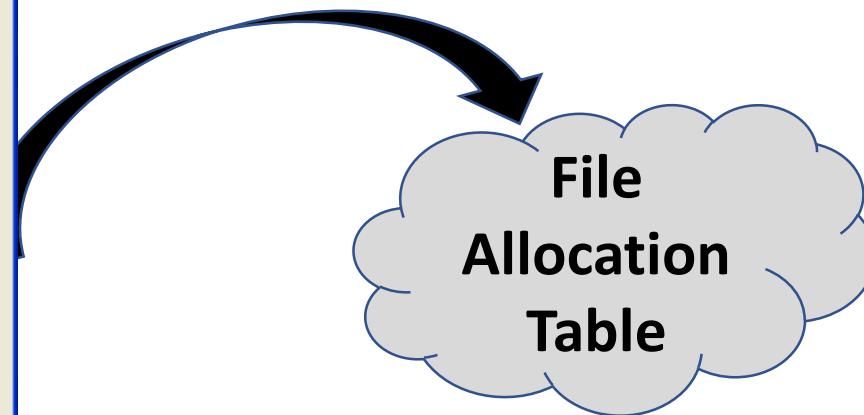
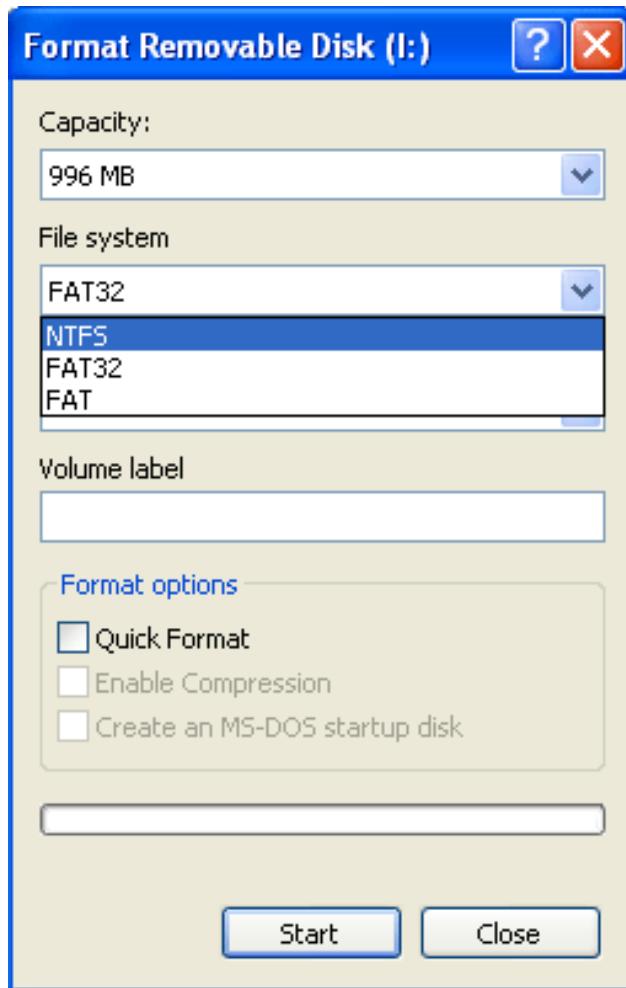
# File System

How to store and retrieve files?

# Disk Partitioning

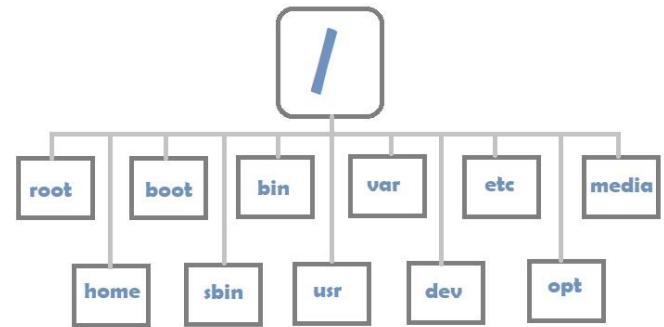
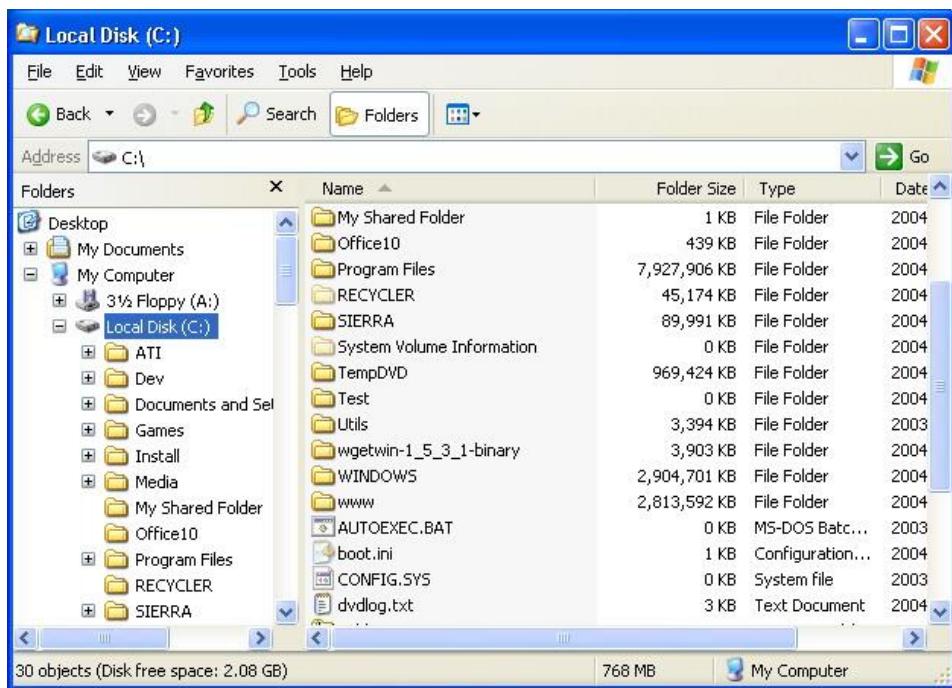


# Formatting



# Files and Folders

- An operating system interface to storage media.

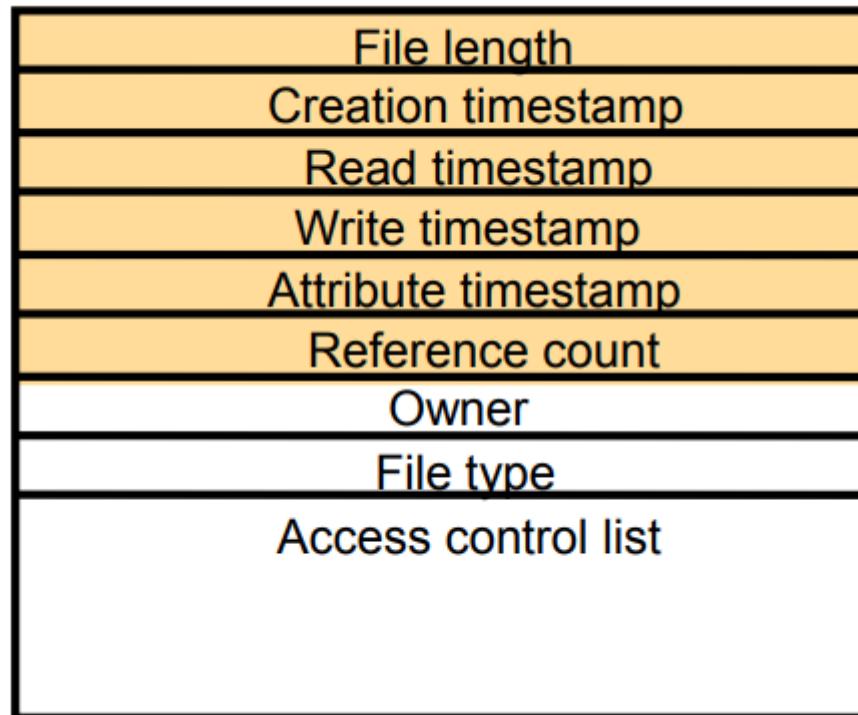


```
Terminal — dtrace — 80x24
0 16982 login      -1 /usr/etc/krb5.conf
0 16982 login      3 /dev/urandom
0 16982 login      -1 /Library/Preferences/edu.mit.Kerberos
0 16982 login      3 /var/run/utmpx
0 35 mds          13 /var/run/utmpx
0 14 configd     16 /var/run/utmpx
0 15 syslogd      17 /var/log/asl
0 15 syslogd      20 /var/log/asl/2010.05.11.asl
0 15 syslogd      20 /var/log/asl/2010.05.11.U0.G80.asl
0 15 syslogd      20 /var/log/asl/StoreData
0 15 syslogd      20 /var/log/asl/SweepStore
0 15 syslogd      17 /var/log/asl/StoreData
0 15 syslogd      17 /var/log/asl
0 15 syslogd      20 /var/log/asl/2010.05.11.asl
0 15 syslogd      20 /var/log/asl/2010.05.11.U0.G80.asl
0 15 syslogd      20 /var/log/asl/StoreData
0 15 syslogd      20 /var/log/asl/SweepStore
0 15 syslogd      17 /var/log/asl/StoreData
501 16983 bash      3 /dev/urandom
501 16983 bash      3 /dev/dtracehelper
501 16983 login     -1 /etc/motd
501 5004 TextWrangler 15 ./vol/234881826/23974096
501 5004 TextWrangler 15 ./vol/234881826/23974096
501 5004 TextWrangler 15 ./vol/234881826/23974096
```

A terminal window titled 'Terminal — dtrace — 80x24' displaying a list of dtrace probes. The output shows various system events being monitored, such as logins, file access, and system calls, along with their corresponding paths and process IDs.

# File

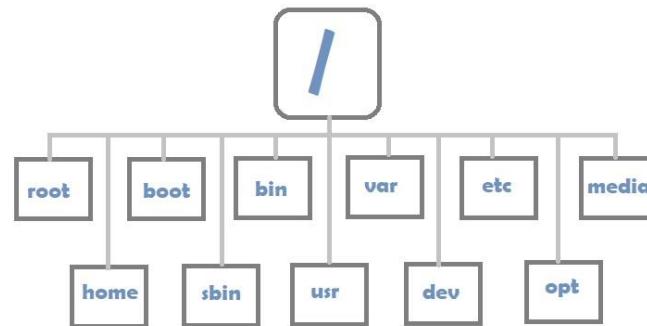
- A Central Object of a File System
- Made of Header and Content



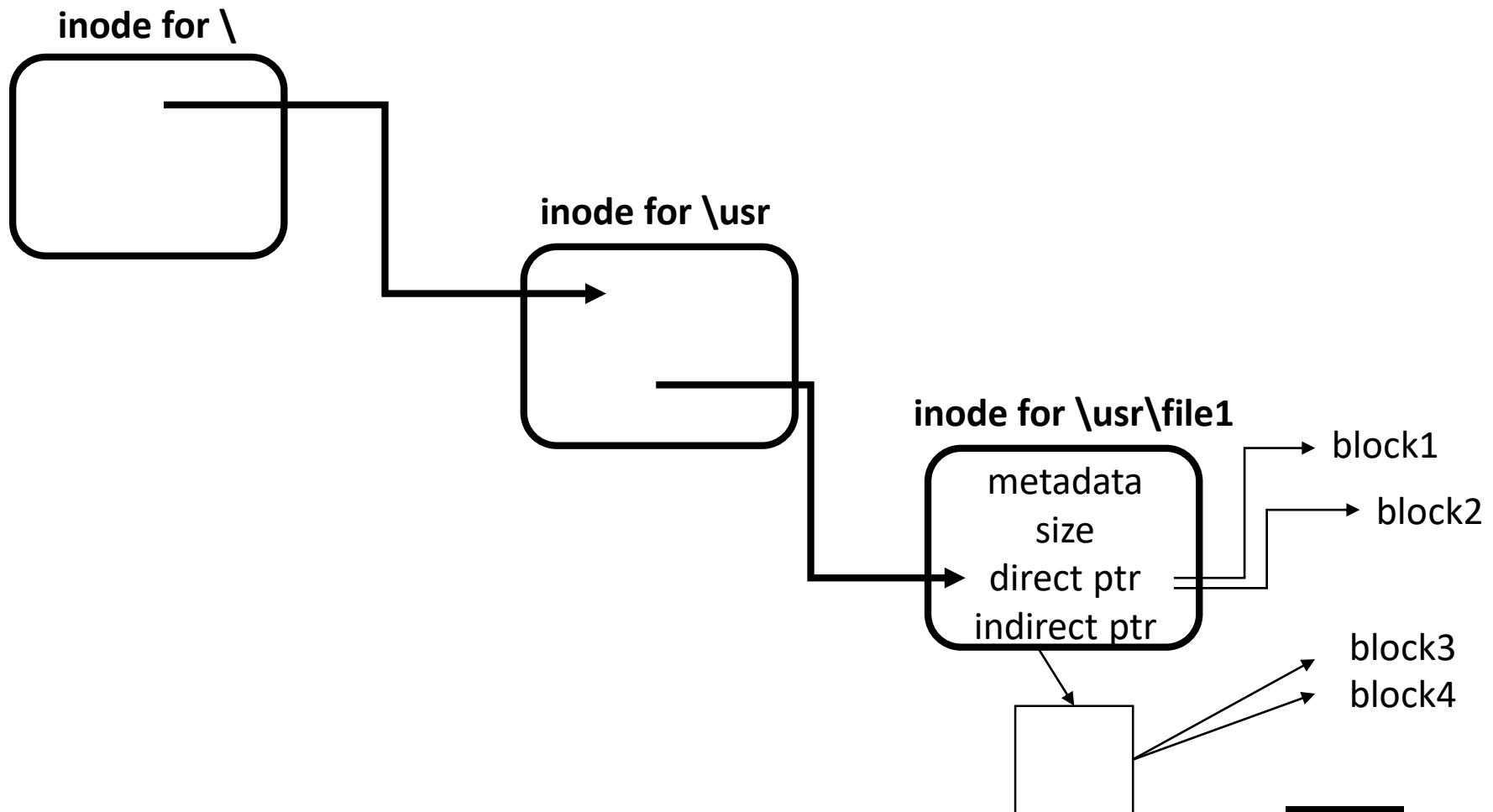
Source: Distributed Systems: Concepts and Design

# Unix/Linux File System

- Everything is a file!
  - CD/DVD, USB, ...
- Hierarchical
  - / (root) is the top level element
- Accessed through commands
  - cat, cd, cp, mkdir, ls, rmdir, ...



# inodes (in linux)



# Inodes

- Every file has an inode number

```
himanshu@ansh:~$ stat test.txt
  File: 'test.txt'
  Size: 22          Blocks: 8          IO Block: 4096   regular file
Device: 807h/2055d      Inode: 3673414      Links: 1
Access: (0664/-rw-rw-r--) Uid: ( 1000/himanshu)  Gid: ( 1000/himanshu)
Access: 2018-02-01 16:49:49.256422217 +0530
Modify: 2018-02-01 16:46:59.628037156 +0530
Change: 2018-02-01 16:46:59.708035450 +0530
 Birth: -
himanshu@ansh:~$
```

# Hardlinks

- Two filenames for the same file.
  - Both the names are mapped to same inode number.

```
root@tryit-right:~# touch f1
root@tryit-right:~# touch f2
root@tryit-right:~# ls
f1  f2
root@tryit-right:~# stat f1
  File: f1
  Size: 0          Blocks: 0          IO Block: 4096   regular empty file
Device: 68h/104d      Inode: 19497      Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/      root)  Gid: (    0/      root)
Access: 2019-12-21 05:58:56.820000000 +0000
Modify: 2019-12-21 05:58:56.820000000 +0000
Change: 2019-12-21 05:58:56.820000000 +0000
 Birth: -
root@tryit-right:~# ln f1 f3
root@tryit-right:~# ls
f1  f2  f3
root@tryit-right:~# stat f3
  File: f3
  Size: 0          Blocks: 0          IO Block: 4096   regular empty file
Device: 68h/104d      Inode: 19497      Links: 2
Access: (0644/-rw-r--r--)  Uid: (    0/      root)  Gid: (    0/      root)
Access: 2019-12-21 05:58:56.820000000 +0000
Modify: 2019-12-21 05:58:56.820000000 +0000
Change: 2019-12-21 05:58:56.820000000 +0000
 Birth: -
```

softlinks are just paths to file.

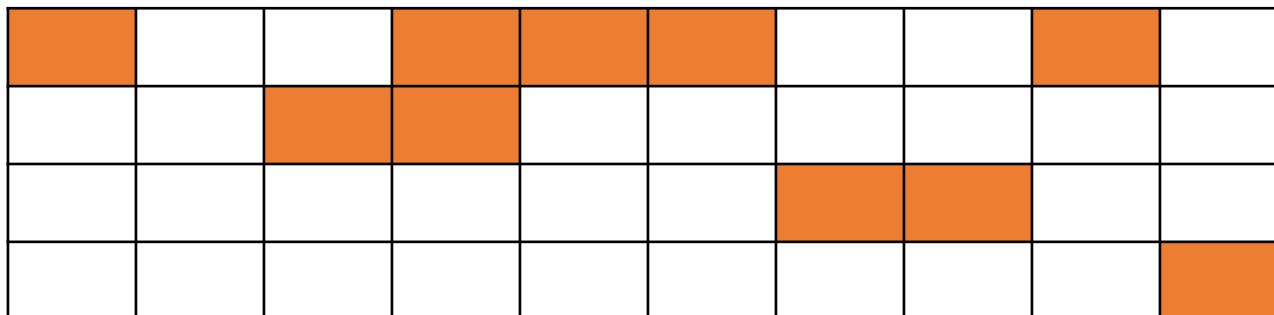
# File Permissions

```
dave@howtogeek:~/work$ ls -l
total 80
drwxr-xr-x 2 dave dave 4096 Aug 23 08:02 archive
-rw-rw-r-- 1 dave dave    780 Aug 20 11:11 command_cls.page
-rw-rw-r-- 1 dave dave   828 Aug 20 11:11 command_exit.page
-rw-rw-r-- 1 dave dave   819 Aug 20 11:11 command_gc.page
-rw-rw-r-- 1 dave dave   799 Aug 20 11:11 command_osm.page
-rw-rw-r-- 1 dave dave   829 Aug 20 11:11 command_quit.page
-rw-rw-r-- 1 dave dave   832 Aug 20 11:11 command_satellite.page
-rw-rw-r-- 1 dave dave   811 Aug 20 11:11 command_street.page
-rw-rw-r-- 1 dave dave 28127 Aug 20 11:11 GC Help.mm
-rwxrwxr-x 1 dave dave     46 Aug 20 11:11 mh.sh
-rw-rw-r-- 1 dave dave 16149 Aug 20 11:11 window_tool.page
dave@howtogeek:~/work$
```

# File Allocation Methods



**How would you like it if we  
contiguously write blocks to disk?**

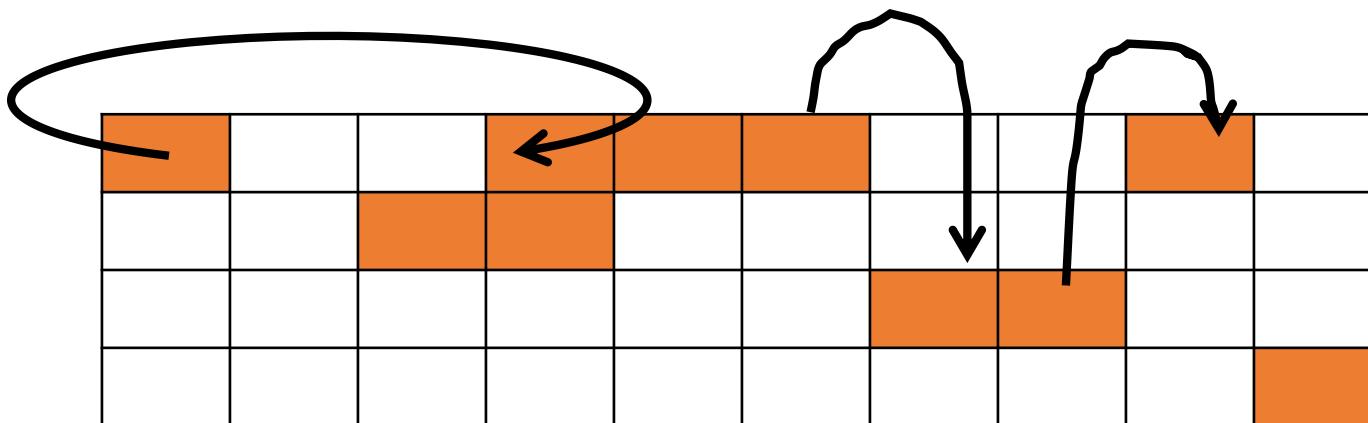


Data stored in blocks but need not be in contiguous blocks.

# File Allocation Methods



## Linked File Allocation



Each file is a linked list of disk blocks

# File Allocation Methods

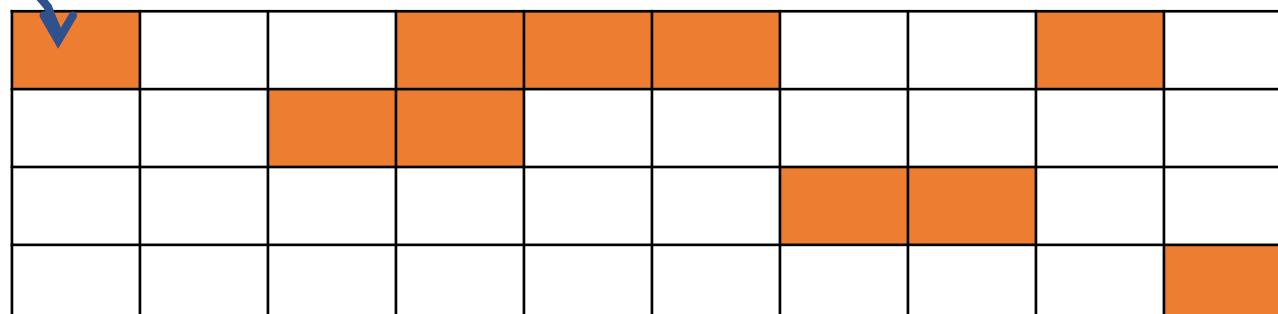


## Indexed Allocation

Each file has an index block that stores array of block addresses.

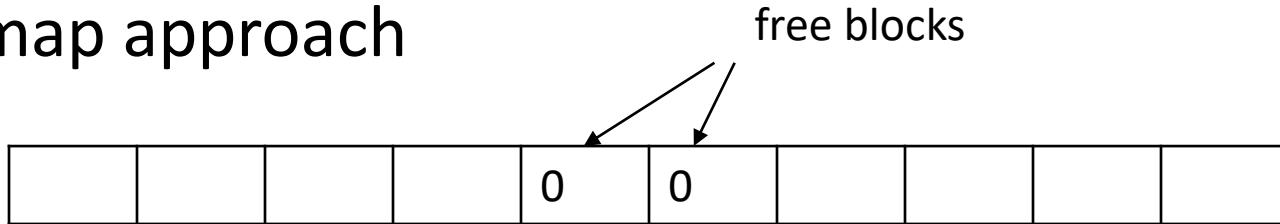
File	Index Block Address
cmi.txt	20

20: Index
1
4
5
6
9



# Free Space Management

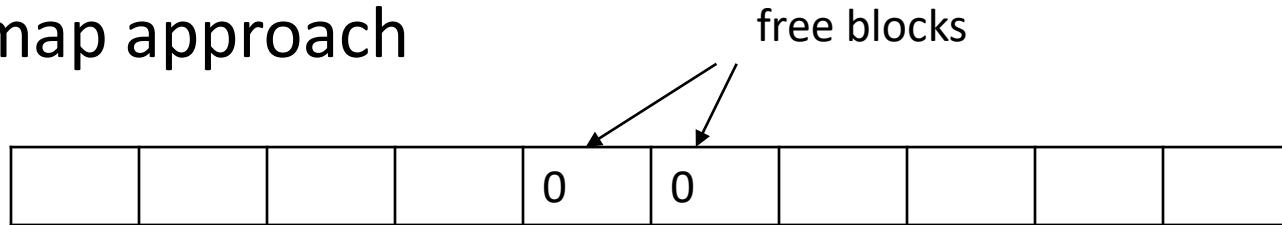
- Bitmap approach



- Assume disk size = 1 Terabyte, block size = 4 KB. How much space will we need to store the free space bitmap?

# Free Space Management

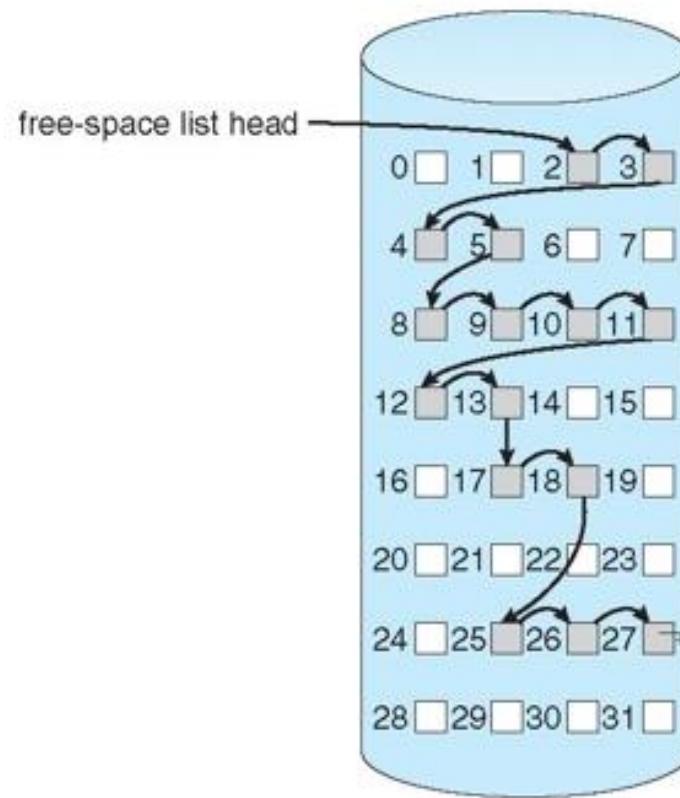
- Bitmap approach



- Assume disk size = 1 Terabyte, block size = 4 KB. How much space will we need to store the free space bitmap?
  - $1 \text{ TB} / 4 \text{ KB} = 2^{40}/2^{12} = 2^{28} = 32 \text{ MB.}$

# Free Space Management

- Free-list approach



# Windows File Systems

- CDFS
  - CD ROM File System: ISO 9660-compliant standard.
  - Directory/File names shorter than 32 characters, with max depth of 8 levels!
- UDF (Universal Data Format)
  - created primarily for DVD
  - ISO 13346-compliant
- FAT (File Allocation Table) File System
  - Used in DOS and Win 9x.
  - Serious restrictions on file size, filename length, etc.
- NTFS (Native FS for Windows)
  - Windows 10 uses NTFS!

Criteria	NTFS5	NTFS	exFAT	FAT32	FAT16	FAT12
Max Volume Size	$2^{64}$ clusters – 1 cluster	$2^{32}$ clusters – 1 cluster	128PB	32GB	2GB	16MB
Max Files on Volume	$2^{32} - 1$	$2^{32} - 1$	Nearly Unlimited	4194304	65536	
Max File Size	$2^{64}$ bytes	$2^{44}$ bytes	16EB	4GB minus 2 Bytes	2GB	16MB
Max Clusters Number	$2^{64}$ clusters – 1 cluster	$2^{32}$ clusters – 1 cluster	4294967295	4177918	65520	4080
Max File Name Length	Up to 255	Up to 255	Up to 255	Up to 255	8.3	Up to 254

# Compression

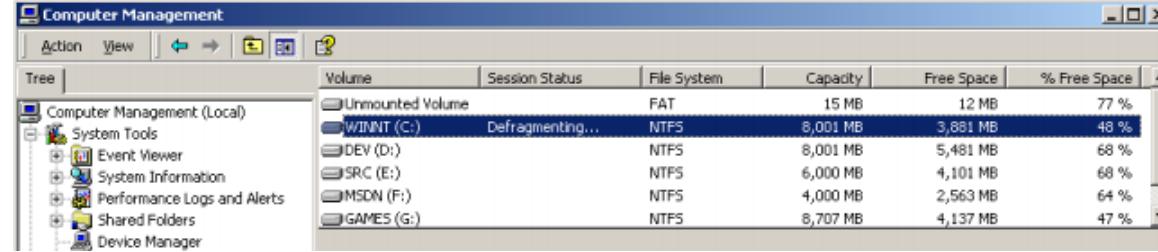
- Why compress while storage and retrieval?

# Compression

- Why compress while storage and retrieval?
  - To narrow the gap between computation and I/O
  - Usually computation power is much higher, I/O speed is too low.

# The Complex World of File Systems

- Defragmentation
- Partitioning
- Compression
- Sharing and Permissions
- Naming Convention
- File Allocation and Free Space Management
- Multiple users and multiple storage media
- ...



A screenshot of the Windows Computer Management console. The left pane shows a tree view with 'Computer Management (Local)' selected, and under it, 'System Tools' is expanded to show 'Event Viewer', 'System Information', 'Performance Logs and Alerts', 'Shared Folders', and 'Device Manager'. The right pane is a table titled 'Computer Management' with the following data:

Volume	Session Status	File System	Capacity	Free Space	% Free Space
Unmounted Volume		FAT	15 MB	12 MB	77 %
WIMINT (C:)	Defragmenting...	NTFS	8,001 MB	3,881 MB	48 %
DEV (D:)		NTFS	8,001 MB	5,481 MB	68 %
SRC (E:)		NTFS	6,000 MB	4,101 MB	68 %
MSDN (F:)		NTFS	4,000 MB	2,563 MB	64 %
GAMES (G:)		NTFS	8,707 MB	4,137 MB	47 %

# The Complex World of File Systems

Partitioning  
Multiple OS,  
Multiple File  
Systems

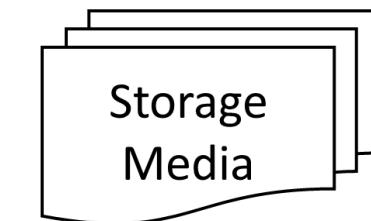


Multiple Users

Multi-Tenancy  
& data privacy  
Permissions and  
Sharing

Compression

High Data  
Transfer  
Time



Multiple Storage Devices

Defragmentation

High Seek  
Time

File Allocation,  
Free Space  
Management

Space  
Utilization

Data Variety  
Naming  
Convention -  
Standards



BUSINESS INSIDER  
INDIA

TECH INSIDER

ALL

BUSINESS

POLICY

STRATEGY

ADVERTISING

SCIENCE

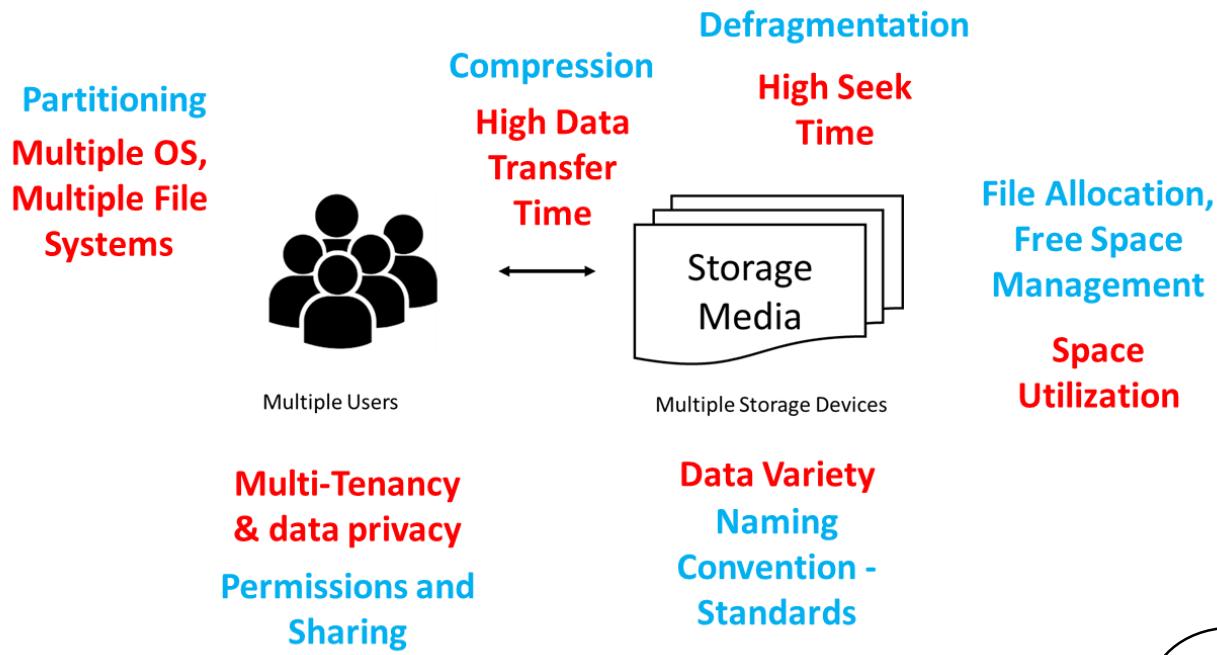
[Home](#) > [Tech](#) > [News](#) » Linus Torvalds, Creator Of The Linux Operating System, Warned Developers Not To Use An Oracle-Owned File System

# Linus Torvalds, creator of the Linux operating system, warned developers not to use an Oracle-owned file system because of the company's 'litigious nature'

ROSALIE CHAN | JAN 13, 2020, 23:36 IST



# Summary



File systems are key to handling data.

Variety of FS exist  
NTFS, FAT, DOS,  
CDFS, NFS, ...

# A MODEL FOR DISTRIBUTED COMPUTING

**Venkatesh Vinayakarao**

[venkateshv@cmi.ac.in](mailto:venkateshv@cmi.ac.in)

<http://vvtesh.co.in>

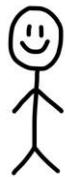
---

Chennai Mathematical Institute

---

Data is the new oil. - Clive Humby, 2006.

# A Distributed Computing Model



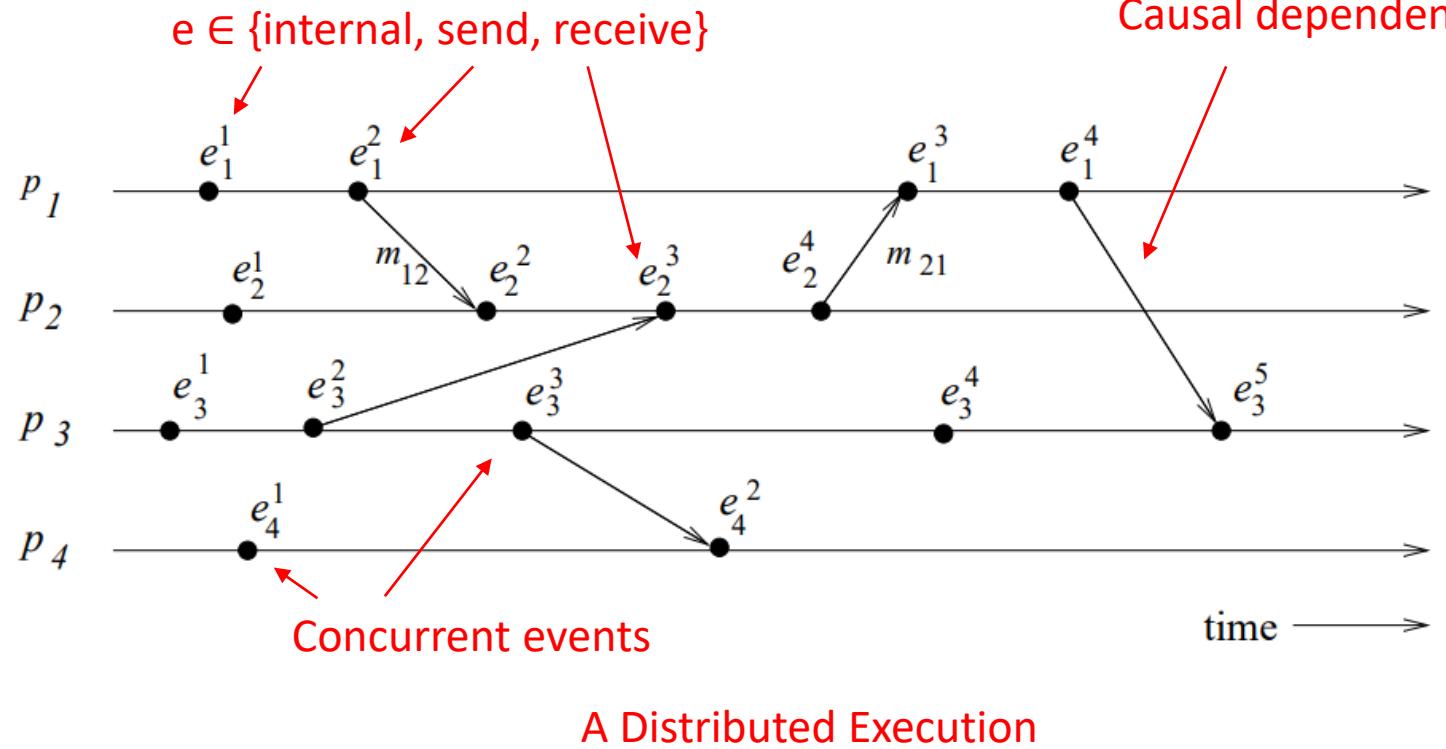
No shared memory.  
No notion of global clock.



How to co-ordinate to get a job done?

# Space-time diagram of a distributed execution

A Message through a channel from an event causes Causal dependency



# Causal dependencies between events

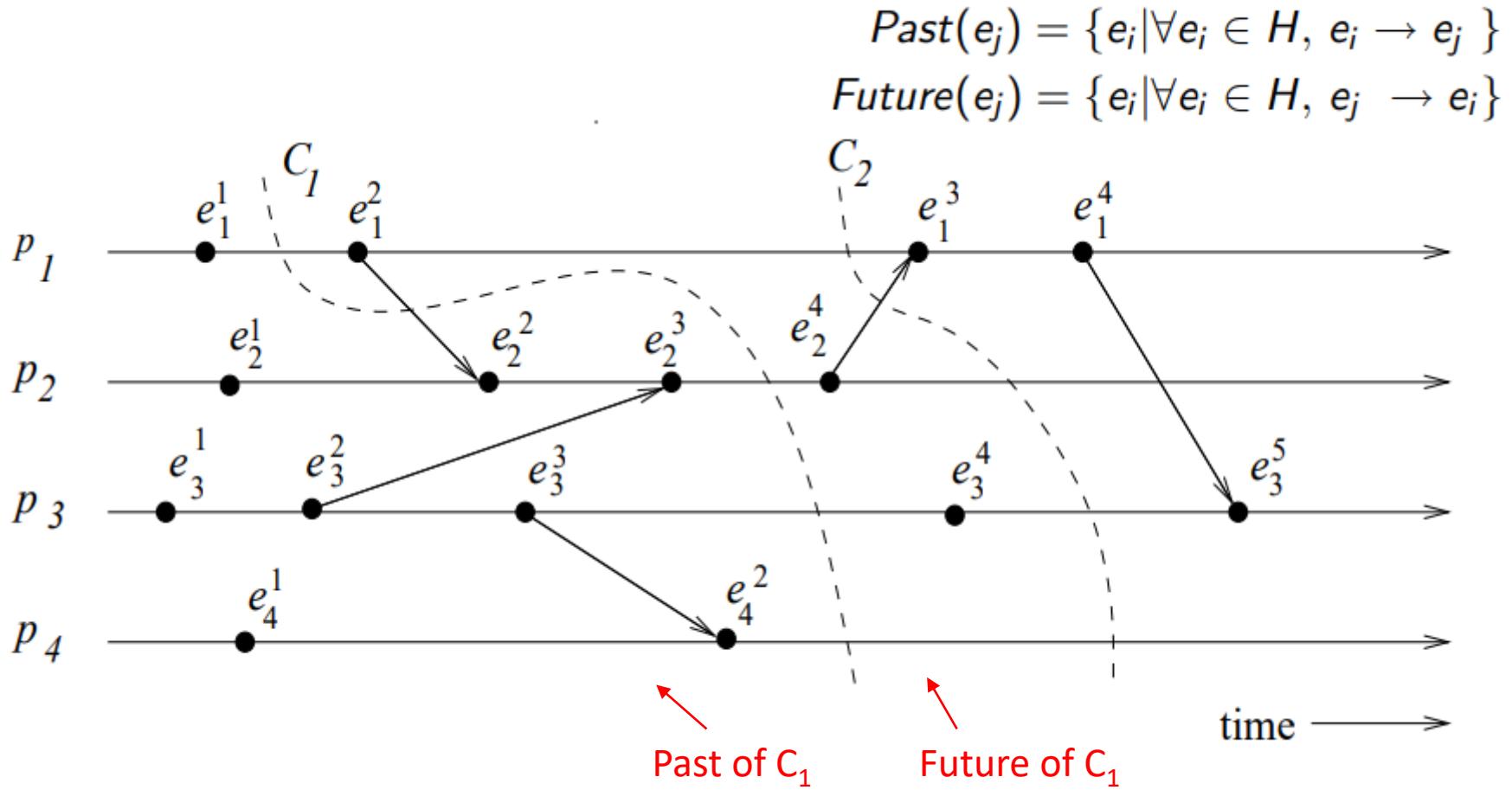
$$\forall e_i^x, \forall e_j^y \in H, e_i^x \rightarrow e_j^y \Leftrightarrow \left\{ \begin{array}{l} e_i^x \rightarrow_i e_j^y \text{ i.e., } (i = j) \wedge (x < y) \\ \text{or} \\ e_i^x \rightarrow_{msg} e_j^y \\ \text{or} \\ \exists e_k^z \in H : e_i^x \rightarrow e_k^z \wedge e_k^z \rightarrow e_j^y \end{array} \right.$$

$H = \bigcup_i h_i$  denotes set of all events of process  $p_i$

$e_1^1 \rightarrow e_3^3$  and  $e_3^3 \rightarrow e_2^6$  denotes a causal dependency from the figure.

$\mathcal{H} = (H, \rightarrow)$  denotes causal precedence relation

# Cuts of an execution



Note that  $C_1$  is inconsistent.  $C_2$  is consistent. Can you see why?

# Local and Global States

$$GS = \{\bigcup_i LS_i^{x_i}, \bigcup_{j,k} SC_{jk}^{y_j, z_k}\}$$

Local State

Channel State

$$SC_{ij}^{x,y} = \{m_{ij} \mid send(m_{ij}) \leq e_i^x \wedge rec(m_{ij}) \not\leq e_j^y\}$$

Thus, channel state  $SC_{ij}^{x,y}$  denotes all messages that  $p_i$  sent upto event  $e_i^x$  and which process  $p_j$  had not received until event  $e_j^y$ .

# Synchrony

- Synchronous communication model
  - On message *send*, the sender process blocks until the message has been *received*.
- Asynchronous
  - Non-blocking type.

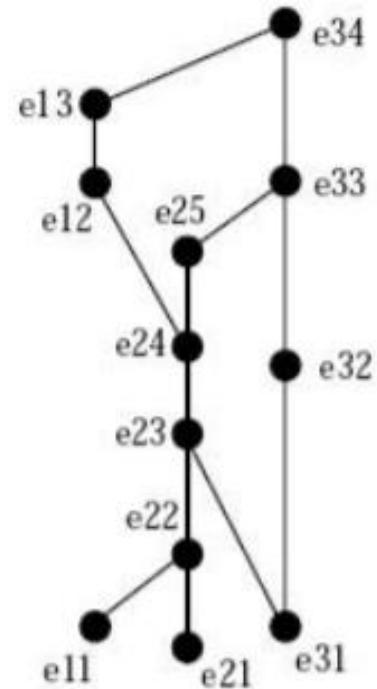
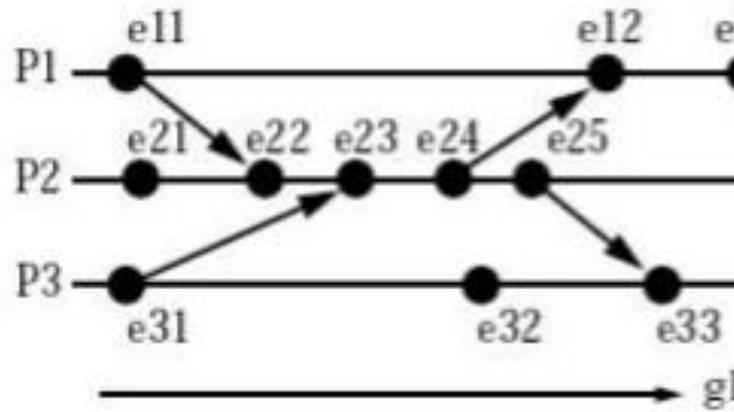
## How to track causality without the notion of global time?

Let us define a local logical clock

$$C : H \mapsto T \quad \text{---} \quad T \text{ is a timestamp}$$

$$e_i \rightarrow e_j \implies C(e_i) < C(e_j)$$

# Processes with Local Clocks



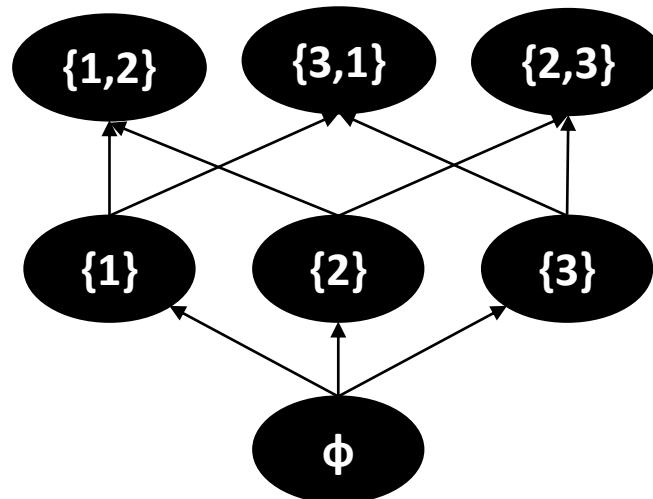
# Total Vs. Partial Order

The Pair  $(\{1,2,3\}, <)$



A strict total order.

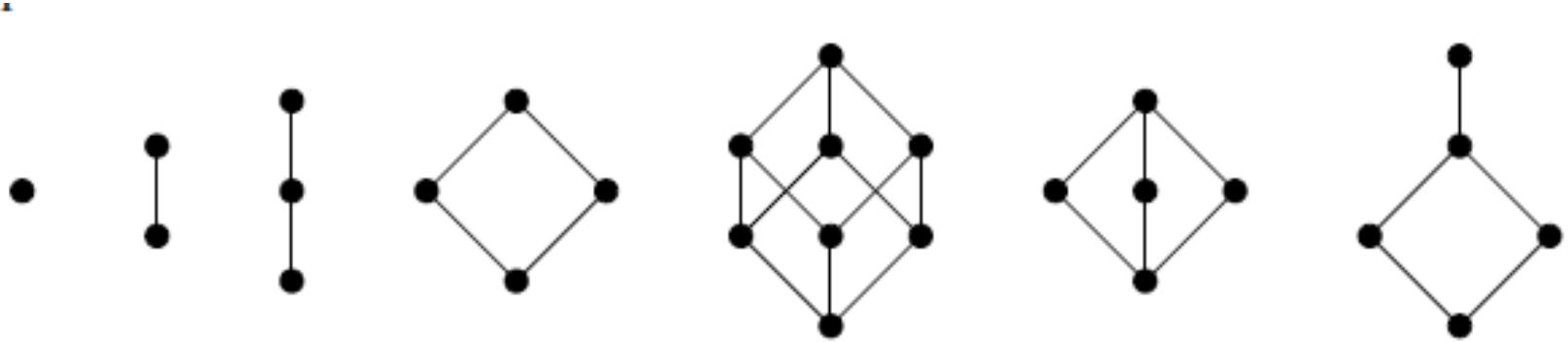
The Pair  $(\{\{\}, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}\}, \leq)$



Partially ordered under  
the  $\leq$  operation!

Reflexive, Transitive and Anti-symmetric  
 $a \leq a$     $a \leq b$  and  $b \leq c$     $a \leq b$  and  $b \leq a$   
implies  $a \leq c$       implies  $a = b$

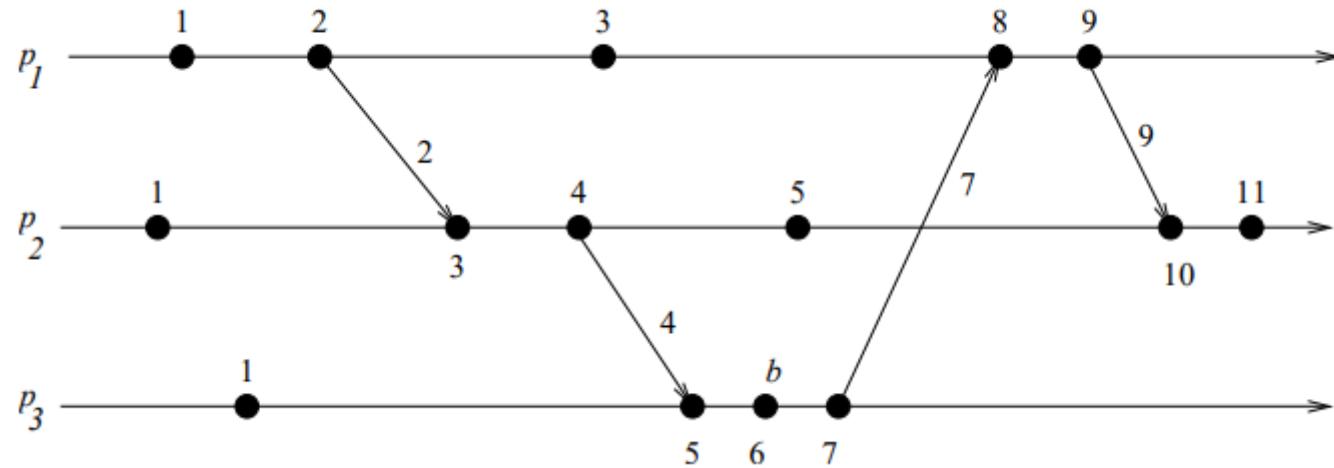
# Hasse Diagram



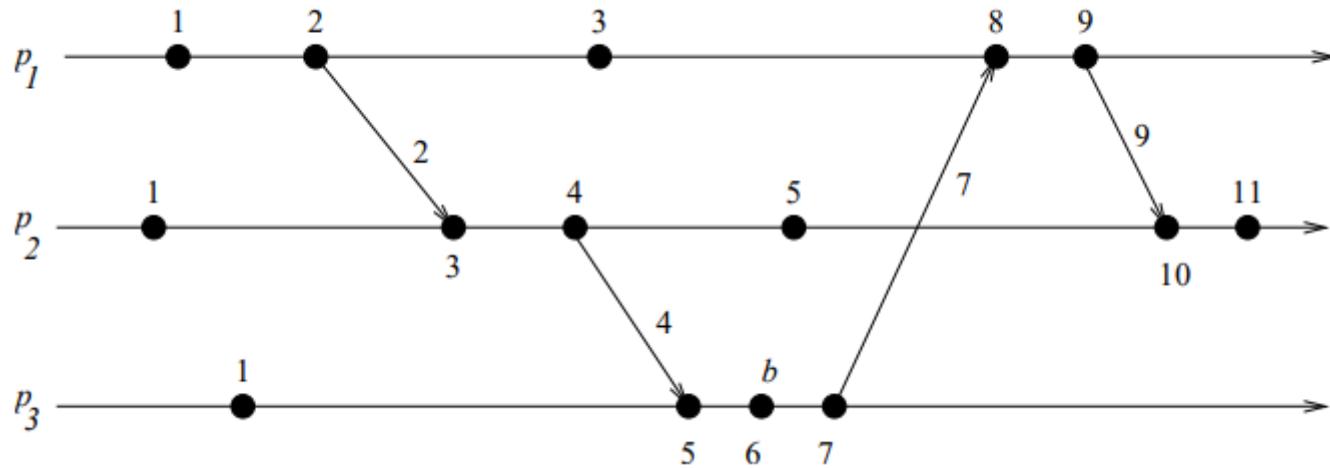
---

Image source: Static Program Analysis, Moller and Schwartzbach

# Scalar Time



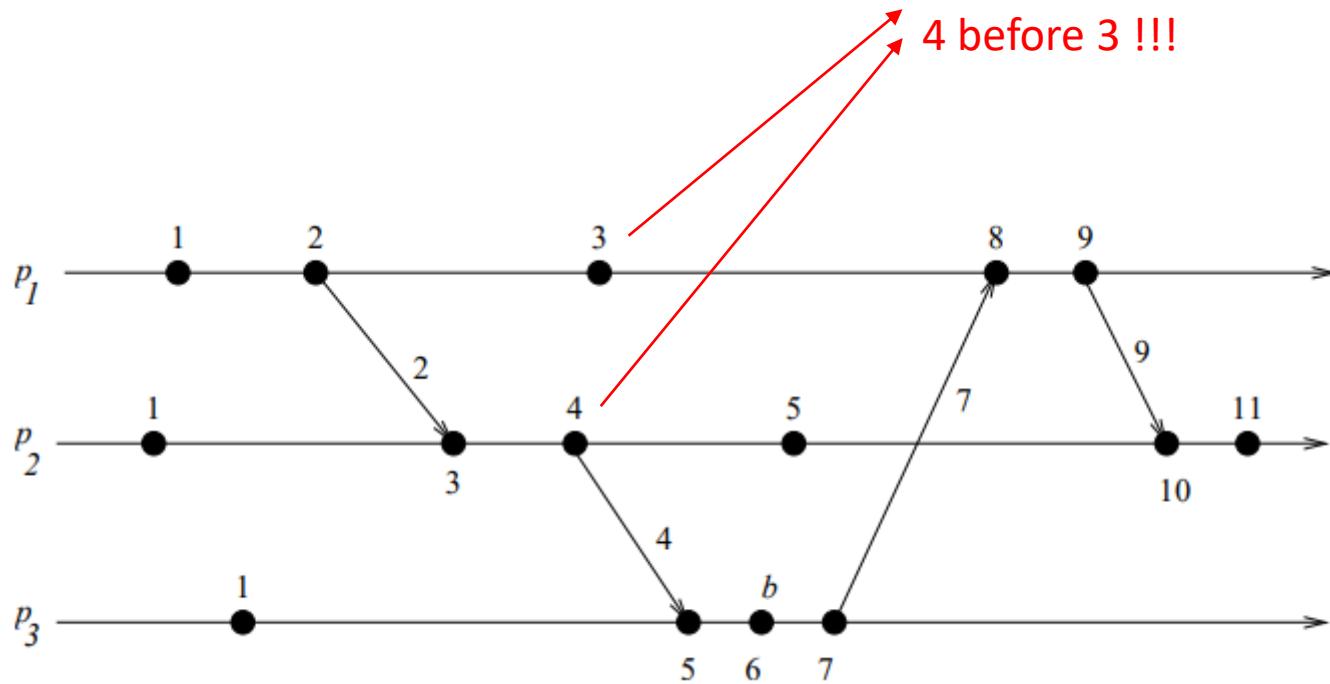
# Scalar Time



$$C(e_i) < C(e_j) \not\Rightarrow e_i \rightarrow e_j$$

Not strongly consistent

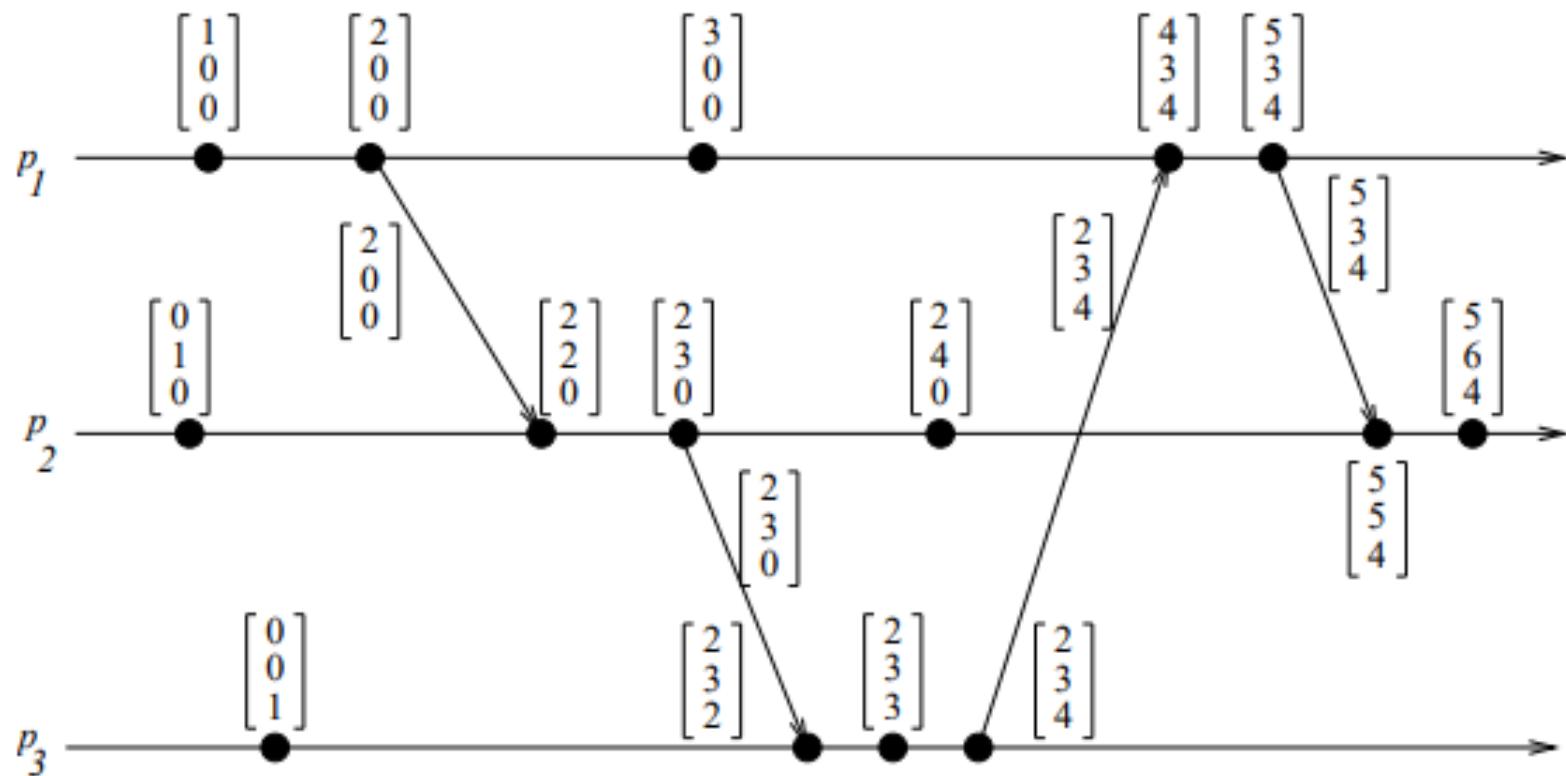
# Scalar Time



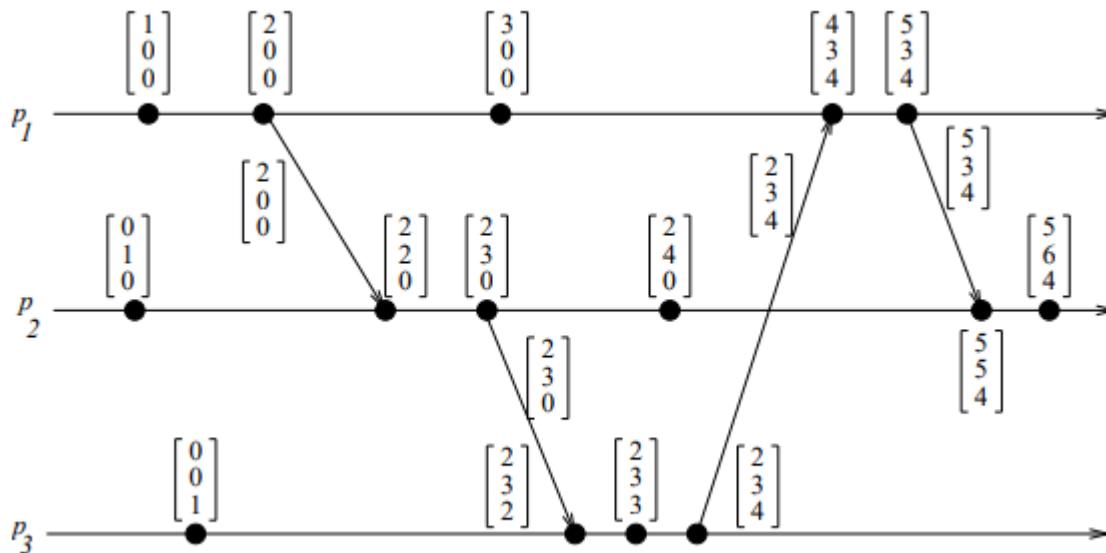
$$C(e_i) < C(e_j) \not\Rightarrow e_i \rightarrow e_j$$

Not strongly consistent

# Vector Time



# Vector Time

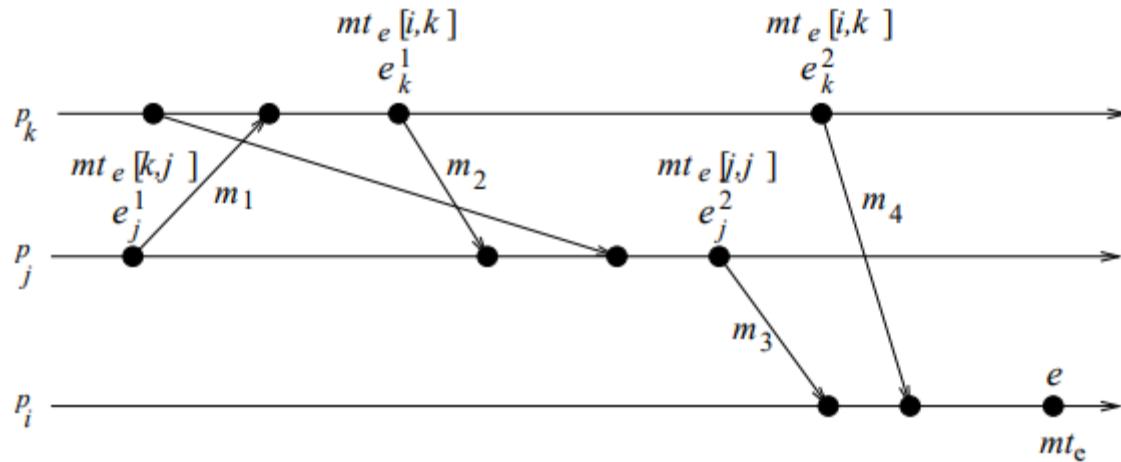


**Implementing Vector Time?** Notice that “*between successive message sends to the same process, only a few entries of the vector clock at the sender process are likely to change*”.

Singhal-Kshemkalyani’s differential technique uses this observation.

# Matrix Time

The entire matrix  $mt_i$  denotes  $p_i$ 's local view of the global logical time.



In addition, matrix clocks have the following property:

$\min_k(mt_i[k, l]) \geq t \Rightarrow$  process  $p_i$  knows that every other process  $p_k$  knows that  $p_i$ 's local time has progressed till  $t$ .

# A Distributed Computing Algorithm

- Consensus Problem
  - All processes have an initial value
  - All non-faulty processes must agree on the same (single) value.

# A Distributed Computing Algorithm

- Consensus Problem
  - All processes have an initial value.
  - All non-faulty processes must agree on the same (single) value.
  - Setting: Message-Passing, Synchronous.

```
(global constants)
integer:  $f$ ;                                // maximum number of crash failures tolerated
(local variables)
integer:  $x \leftarrow$  local value;

(1) Process  $P_i$  ( $1 \leq i \leq n$ ) executes the Consensus algorithm for up to  $f$  crash failures:
(1a) for  $round$  from 1 to  $f + 1$  do
(1b)   if the current value of  $x$  has not been broadcast then
(1c)     broadcast( $x$ );
(1d)    $y_j \leftarrow$  value (if any) received from process  $j$  in this round;
(1e)    $x \leftarrow \min(x, y_j)$ ;
(1f) output  $x$  as the consensus value.
```

# DISTRIBUTED FILE SYSTEM

**Venkatesh Vinayakarao**

[venkateshv@cmi.ac.in](mailto:venkateshv@cmi.ac.in)

<http://vvtesh.co.in>

---

Chennai Mathematical Institute

---

**“Hadoop” is a philosophy — a movement towards a modern architecture for managing and analyzing data.** – Arun Murthy, Hortonworks, Cloudera, 2019.

**The notion of time is an important concept in every day life of our decentralized “real world”** - Friedemann Mattern.

# What Comes Next?

byte

kilobyte

megabyte

gigabyte

??

???

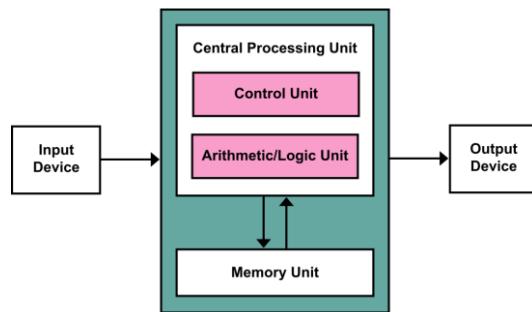
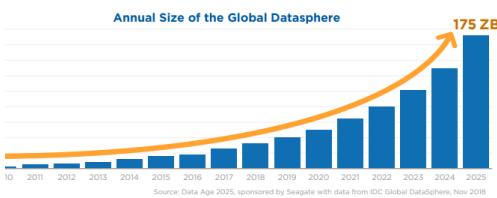
????

?????

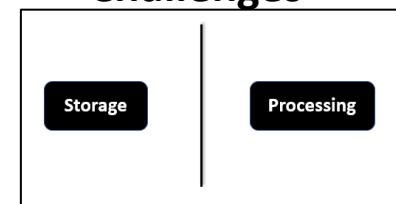
# Sizes

Name	Size
Byte	8 bits
Kilobyte	1024 bytes
Megabyte	1024 kilobytes
Gigabyte	1024 megabytes
Terabyte	1024 gigabytes
Petabyte	1024 terabytes
Exabyte	1024 petabytes
Zettabyte	1024 exabytes
Yottabyte	1024 zettabytes

# Recap

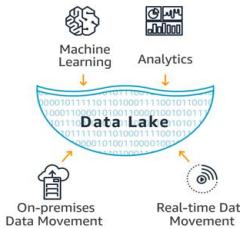


## Challenges

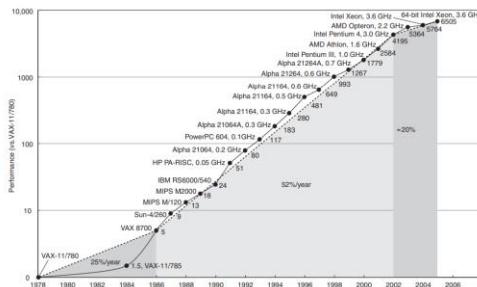


# Recap

## Data Storage

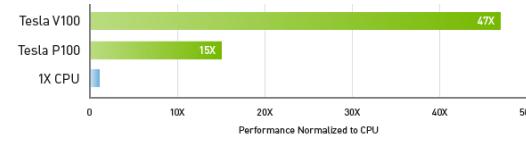


## Data Processing



CPU Performance

47X Higher Throughput Than CPU Server on Deep Learning Inference



GPU Performance

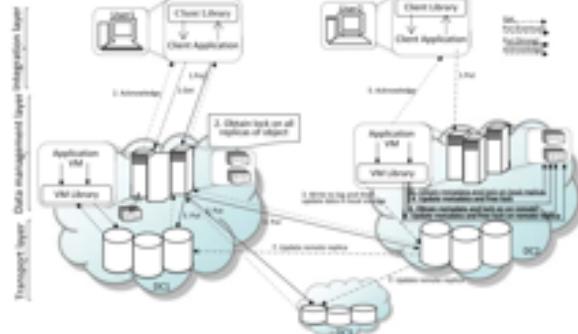


SuperComputers

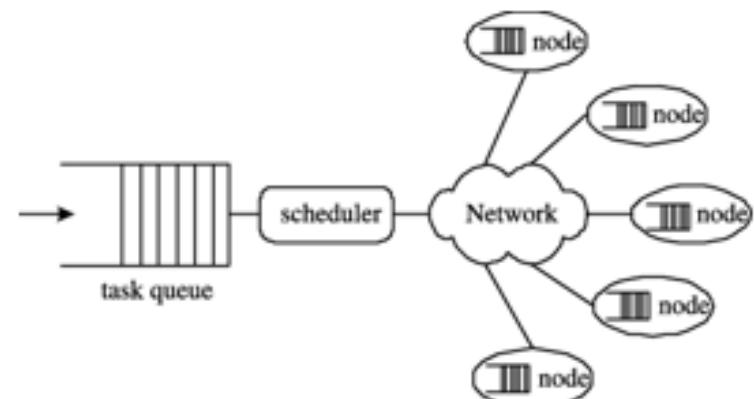
# Cloud Computing

Two kinds of Big Data Opportunities

## Storage

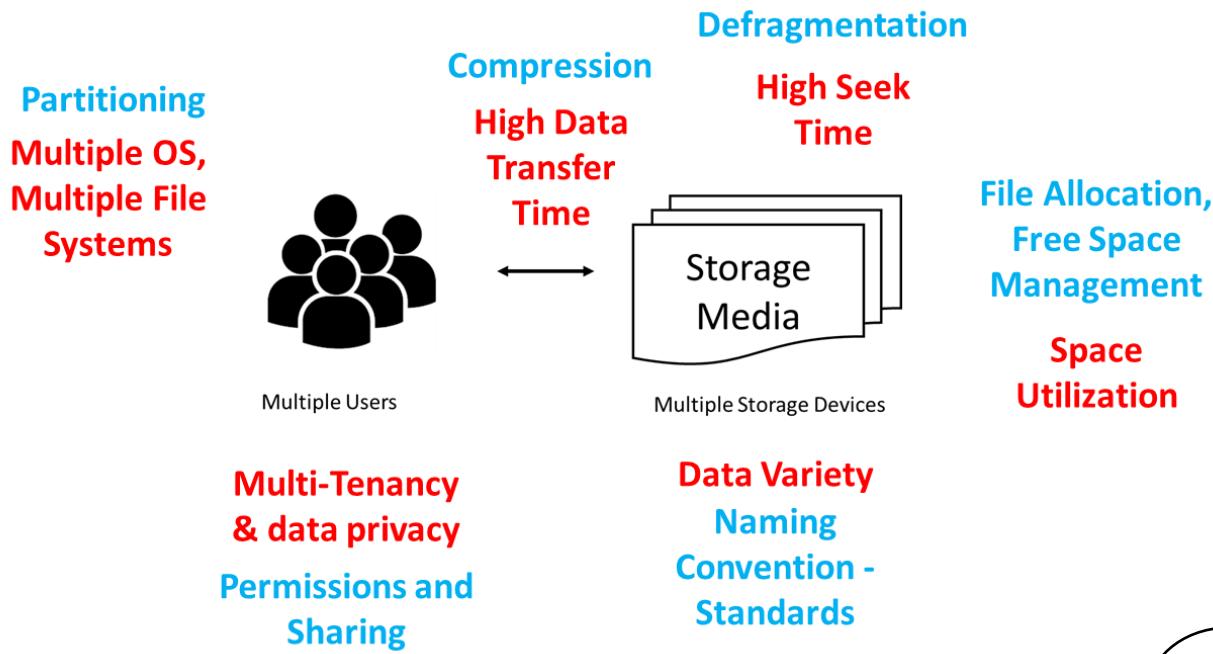


## Processing

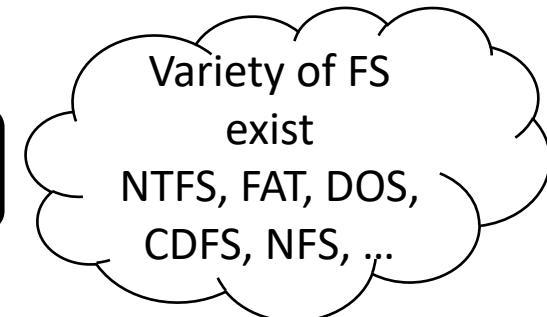


So, we have the cloud. But, how to store and retrieve data? How to process jobs?

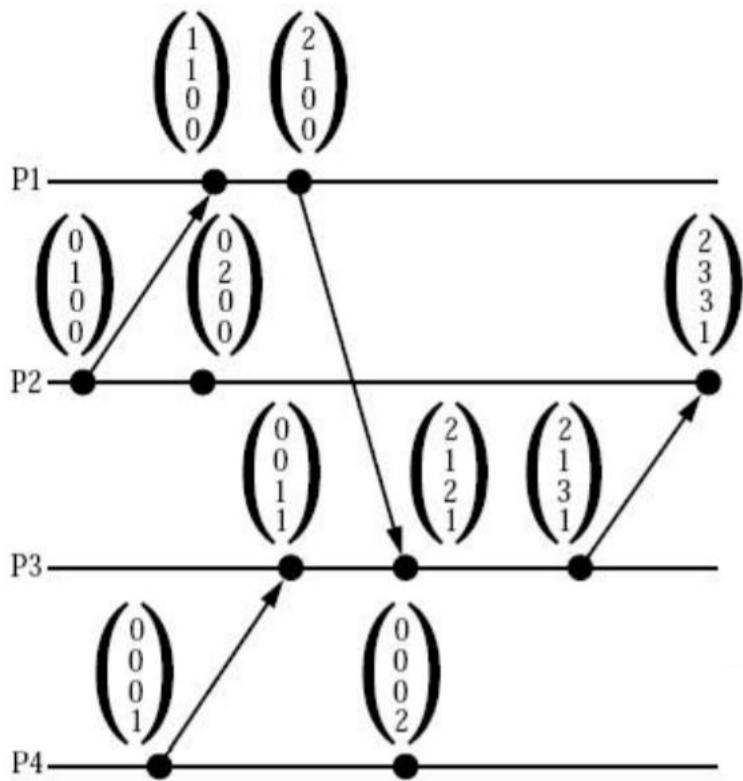
# Role of File Systems



File systems are key to handling data.

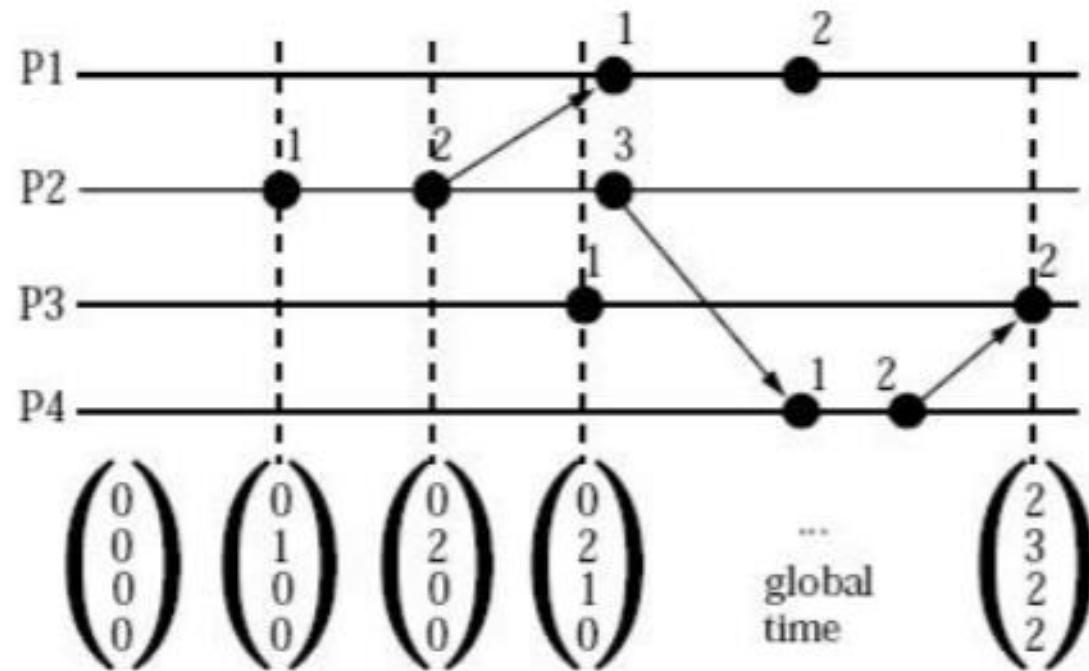


# Vector Time Stamps



- Local clock is incremented every time an event occurs.
- An external observer may know about all events.
- Global time knowledge can be saved as a vector, with one element per process.

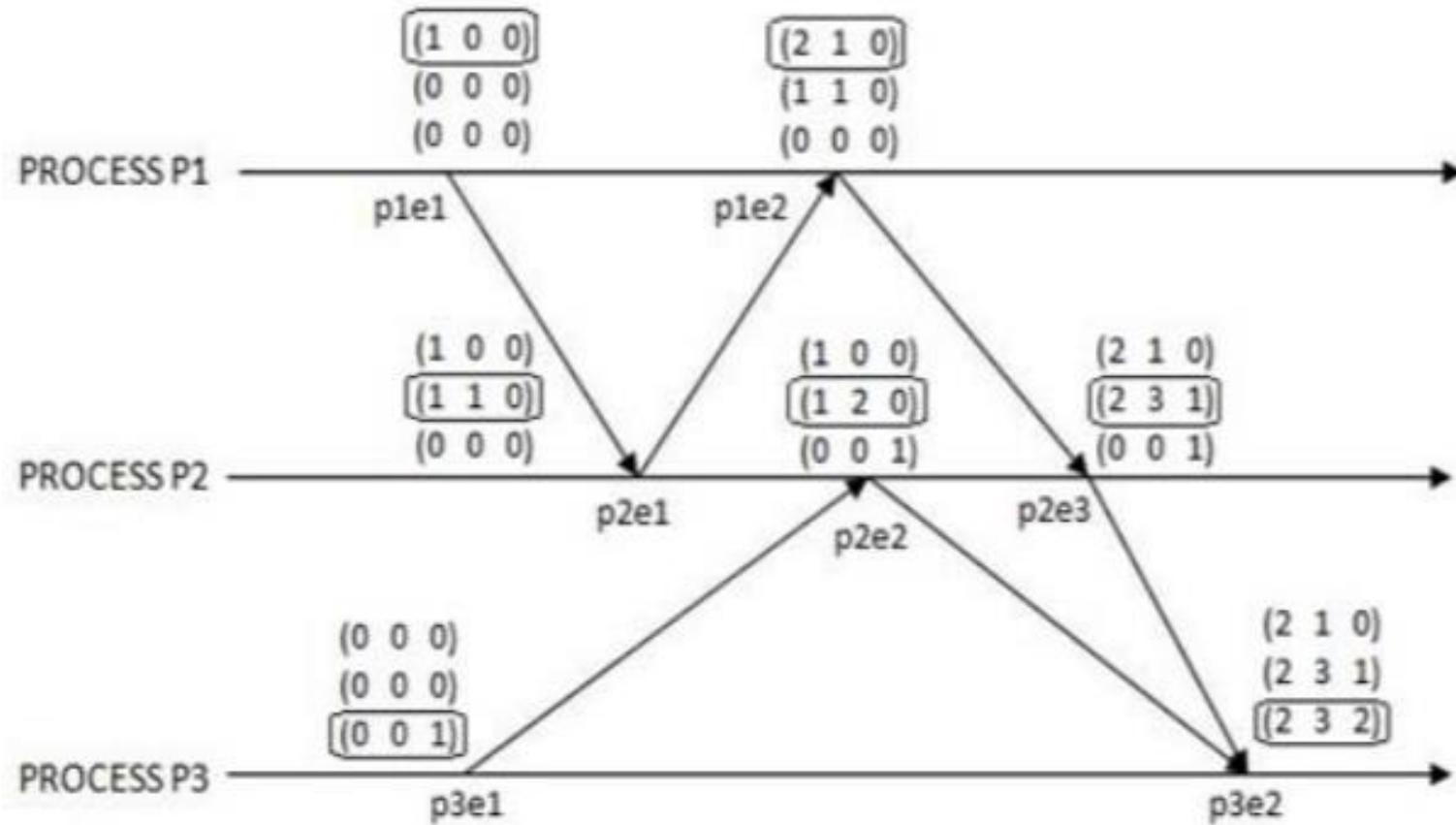
# Global Vector Time

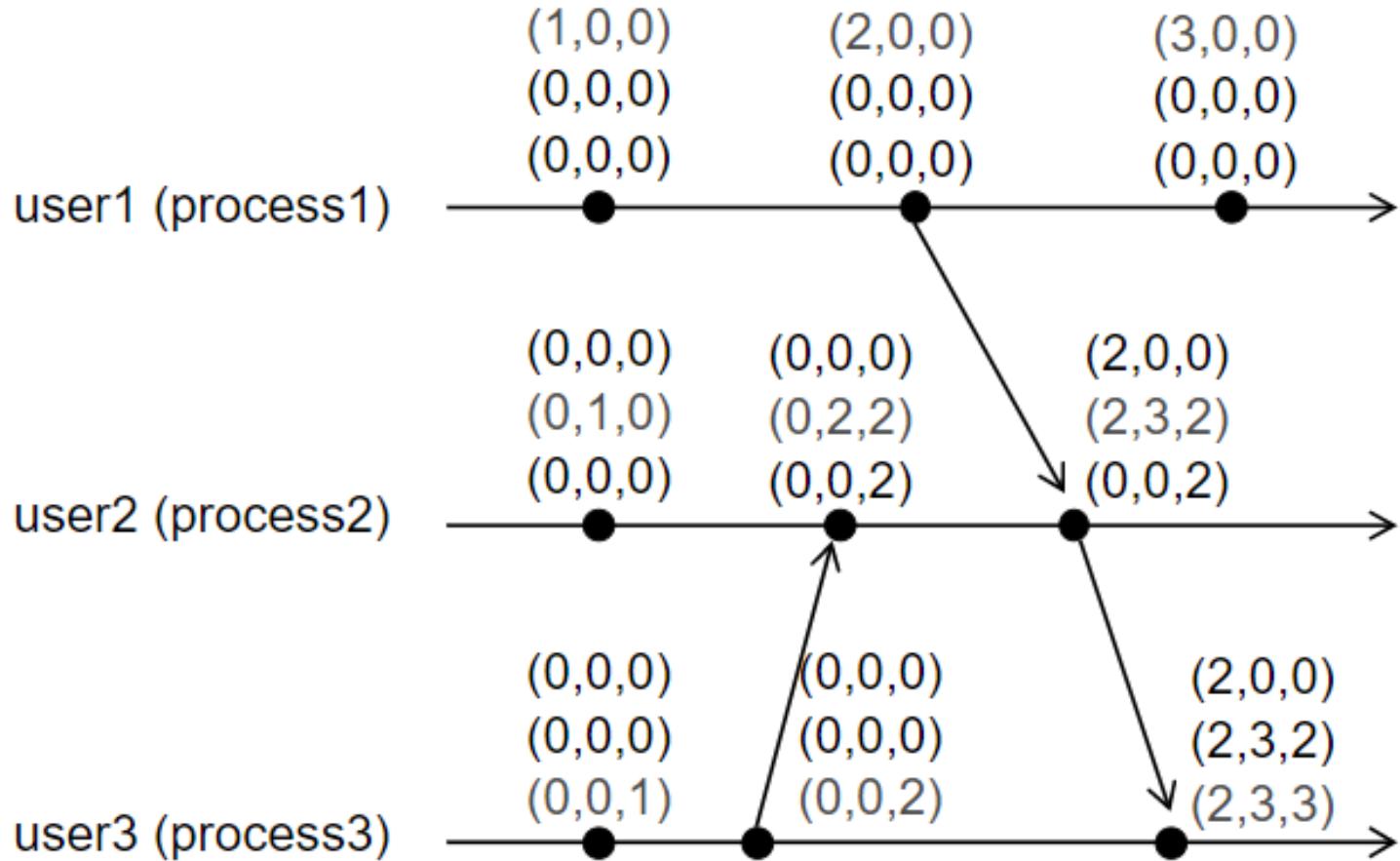


# Quiz

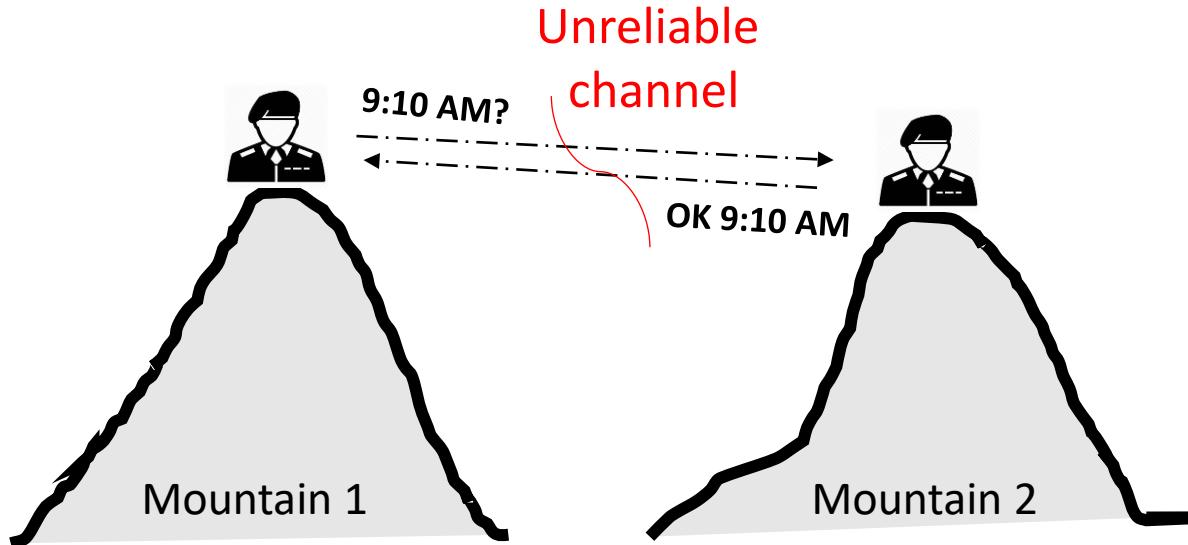
- Which of the below specify a strictly “**happens-before**” relationship between two event time stamps?
  - $(2,0,0) \rightarrow (3,0,0)$
  - $(2,2,1,3) \rightarrow (3,3,2,4)$
  - $(1,2,1,2) \rightarrow (1,1,2,2)$
  - $(3,3,2,4) \leftarrow (2,2,1,3)$

# Matrix Time





# Limitations



## General's Paradox

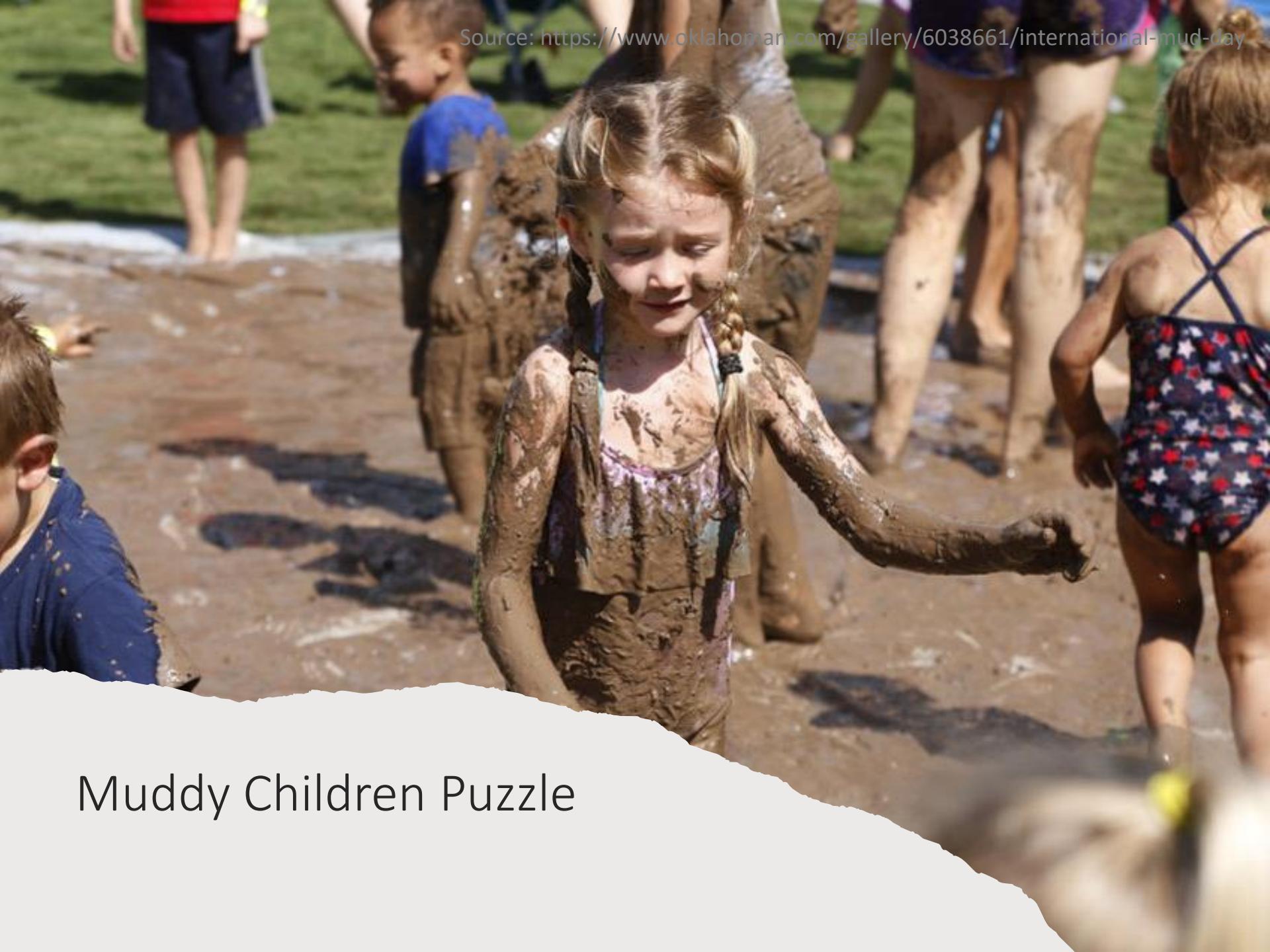
Two generals must attack at the same time, or they die.

Is there a way to coordinate?

What if, two machines need to coordinate,  
but not necessarily at the same time?

Is it possible?

Yes! Message Passing.



## Muddy Children Puzzle

# Muddy Children Puzzle

- At least one child has mud on forehead and every child knows that.
- There are  $k$  (where  $k \leq n$ ) out of  $n$  children with mud on head.
- Children are intelligent and truthful.
- A child can see all other children and say if others have mud on their forehead.
- A child does not know if his/her forehead is muddy.
- Children are smart.  
Assuming each child as a distributed process, how will they co-ordinate so that only the  $k$  children who have muddy forehead say 'yes'.

# Muddy Children Puzzle

- How would it work?
  - $n=1$ 
    - $k=1$
  - $n=2$ 
    - $k=1$
    - $K=2$
  - $n=3$ 
    - $k=1$
    - $k=2$
    - $k=3$



## What is an operating system?

Yarn is now the [Apache Hadoop Operating System](#)

### Apache Hadoop

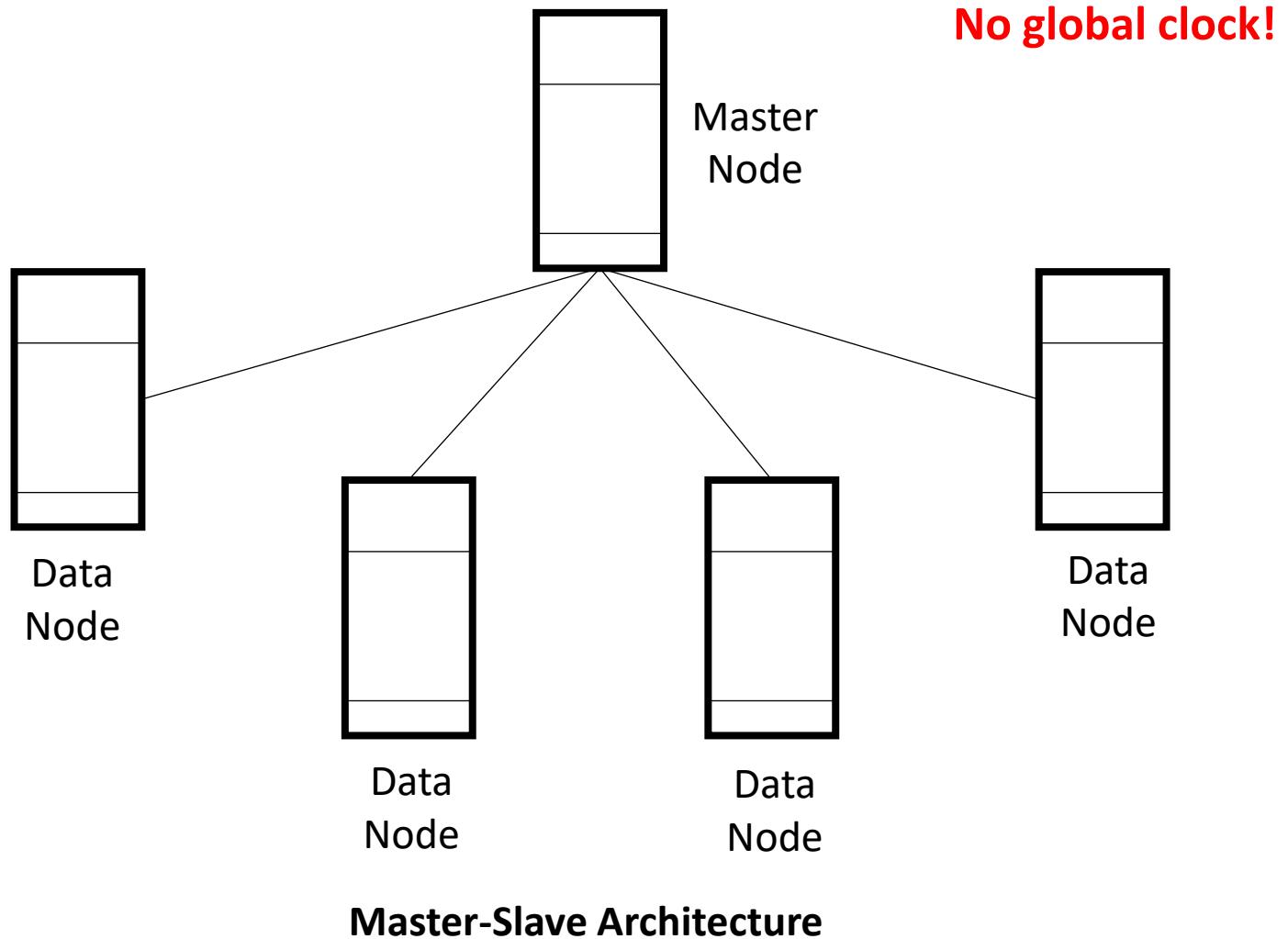
Open source platform for reliable, scalable, distributed processing of large data sets, built on clusters of commodity computers.

# Distributed File Systems - Key Goals

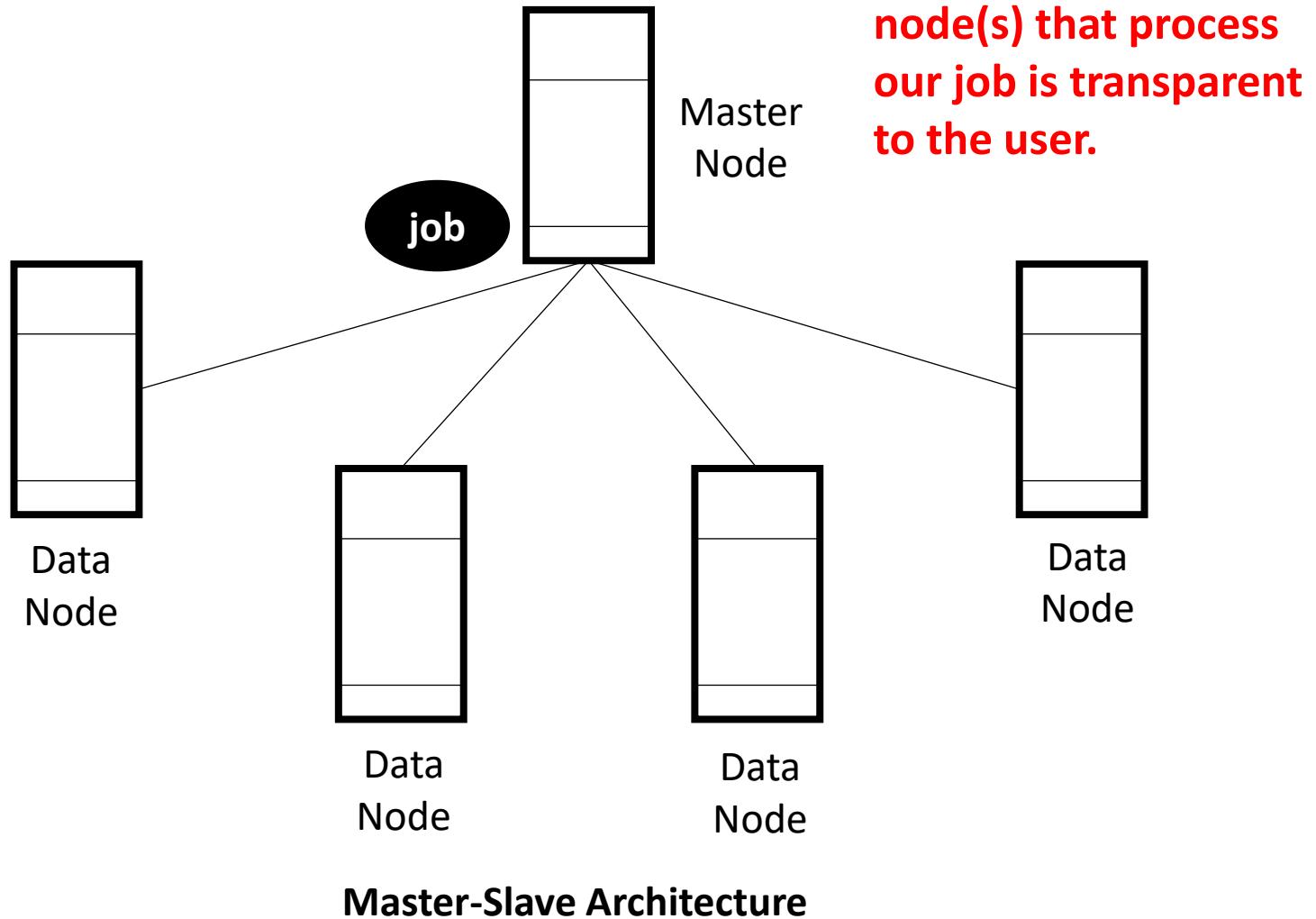
- Distribution Transparency
- Location Transparency
- Scalability
- Fault Tolerance
- Efficient Data Access
  - Specifically designed for batch jobs
- “Write Once Read Many” (WORM) model

Several examples: Andrew FS, Network FS, HDFS

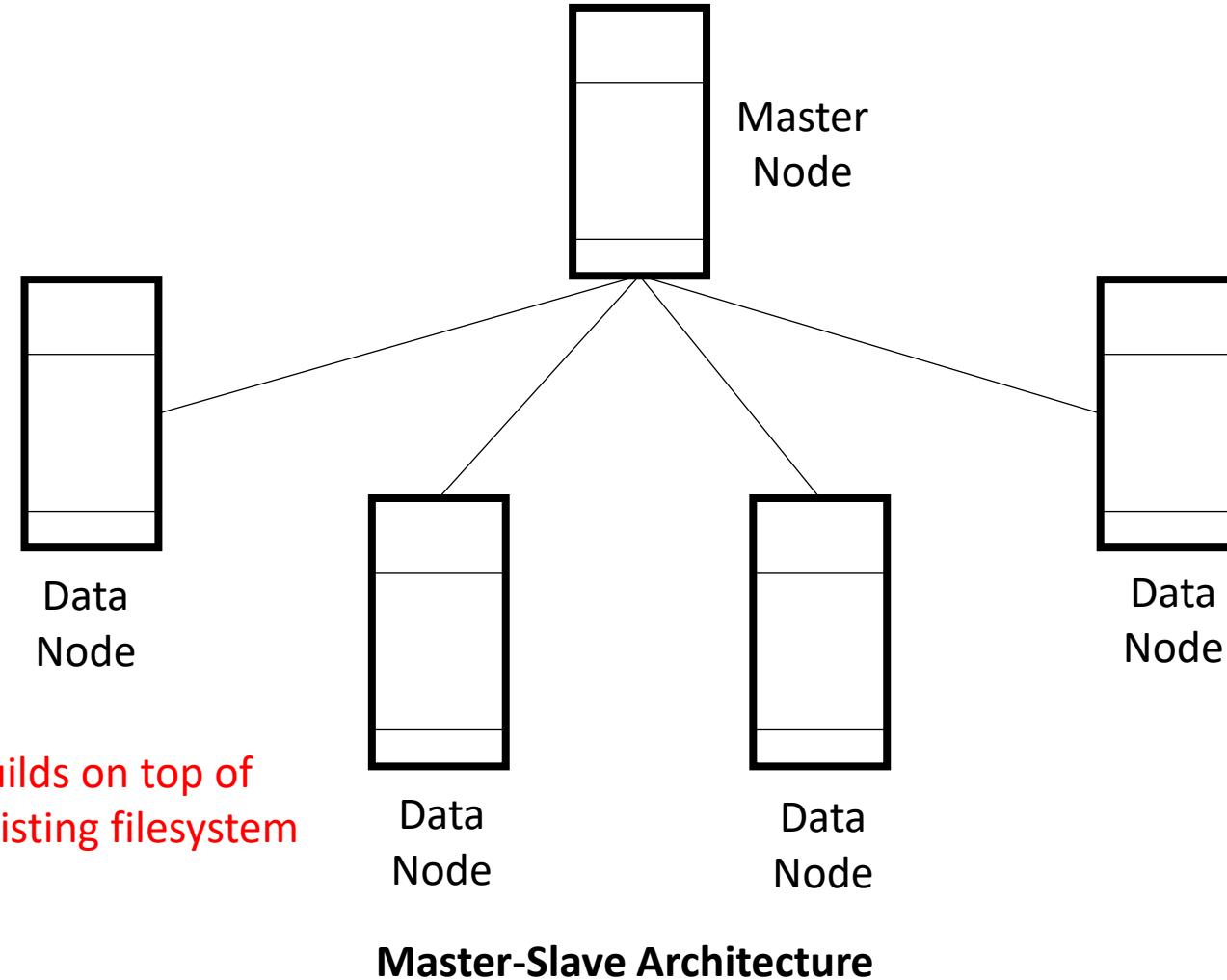
# Distributed System



# Distribution Transparency



# Hadoop Distributed File System Architecture



# Location Transparency



- Refers to uniform file namespace.

Example

```
hdfs dfs -cat hdfs://nn1.cmi.ac.in/file1 hdfs://nn1.cmi.ac.in/file2
```

<https://hadoop.apache.org/docs/r2.4.1/hadoop-project-dist/hadoop-common/FileSystemShell.html>

# HDFS

- HDFS commands are very similar to UNIX shell commands
  - ls
  - du
  - mkdir
- Some additional commands
  - copyToLocal
  - copyFromLocal



```
cd usr/data/
hdfs dfs -copyToLocal test/cmi.csv cmi.csv
```

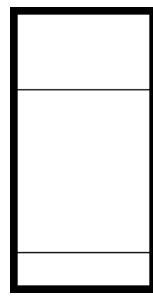
# HDFS Commands

```
$ bin/hadoop fs -ls /user/joe/wordcount/input/  
/user/joe/wordcount/input/file01  
/user/joe/wordcount/input/file02
```

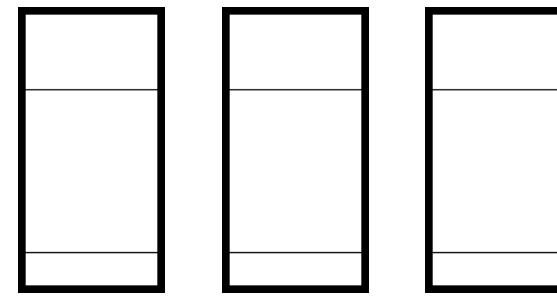
```
$ bin/hadoop fs -cat /user/joe/wordcount/input/file01  
Hello World Bye World
```

```
$ bin/hadoop fs -cat /user/joe/wordcount/input/file02  
Hello Hadoop Goodbye Hadoop
```

# Scalability



**Scale up**  
(add resources  
to a single node)  
Vertical Scaling



**Scale out**  
(add more nodes)  
Horizontal Scaling

With Hadoop, we can scale both vertically and horizontally.

# Scale-up or Scale-out?

- What would you prefer and why?

<https://www.microsoft.com/en-us/research/publication/scale-up-vs-scale-out-for-hadoop-time-to-rethink/>

# Scale-up or Scale-out?

- What would you prefer and why?
  - Depends on data size.
    - Majority of real-world analytic jobs process < 100 GB data.
    - Hadoop is designed for petascale processing.
    - An evaluation (done at Microsoft) across 11 representative Hadoop jobs shows that scale-up is competitive in all cases.

<https://www.microsoft.com/en-us/research/publication/scale-up-vs-scale-out-for-hadoop-time-to-rethink/>

# Adding a New Data Node is Easy

- Prepare the datanode
  - JDK, Hadoop, Environment Variables, Configuration (point to master)
- Start the datanode
  - `hadoop-daemon.sh start datanode`
- Run disk balancer to if you wish to redistribute existing data.



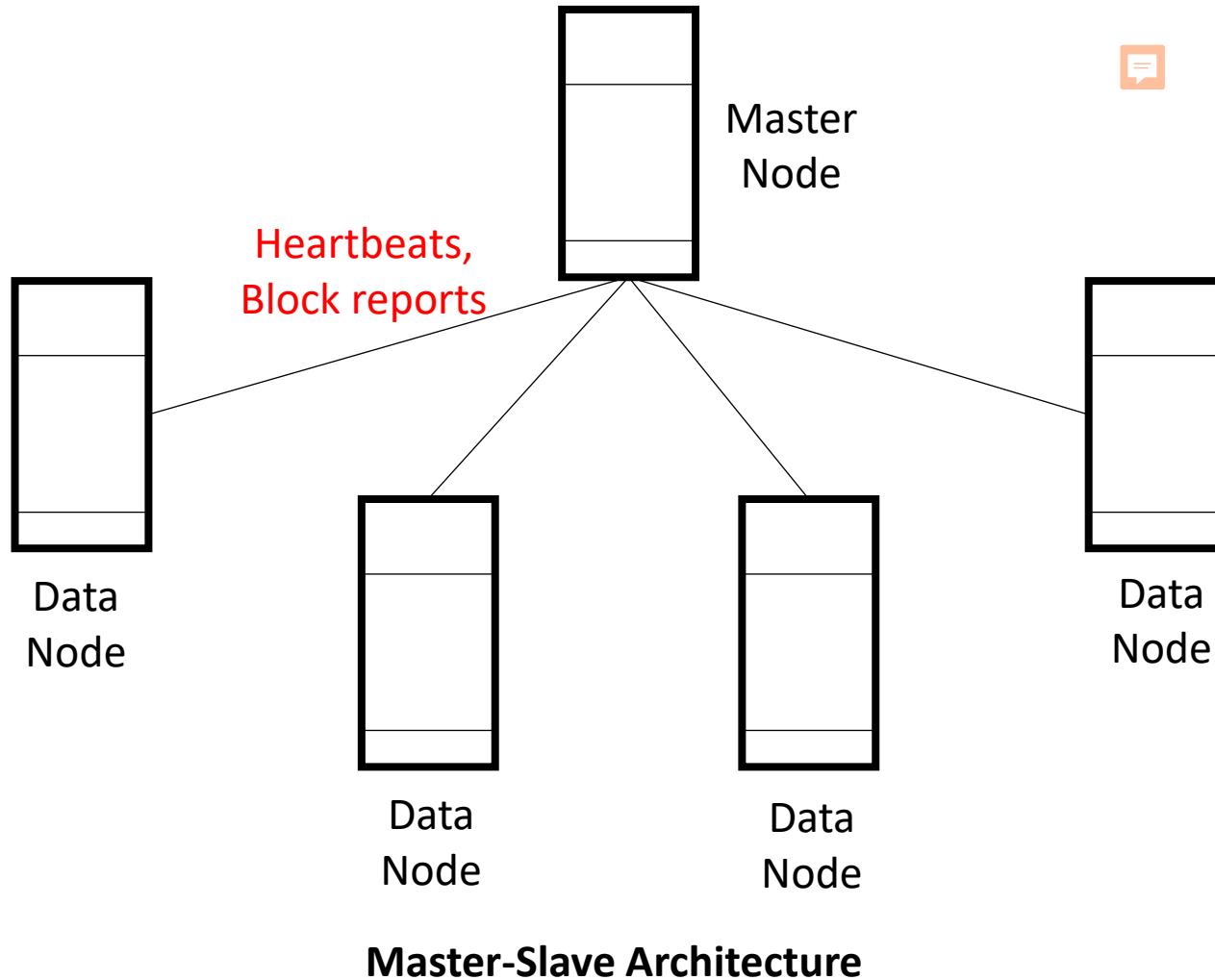
# Fault Tolerance

- Typical clusters have 1000+ datanodes.
- Unfavorable Situations
  - Blocks of data may get corrupted.
  - Datanodes may go down.
  - Network links may go down.

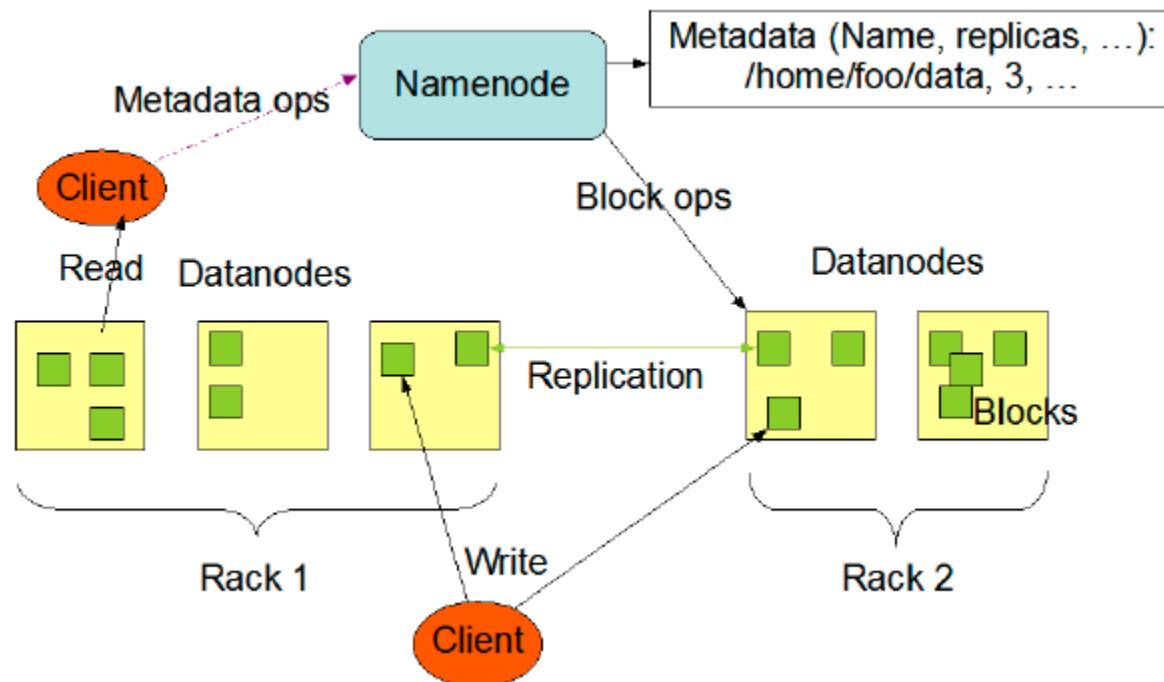
Try This!

Assume we have a **1000** node cluster with each node having a single disk. Also, assume that the disk life is such that every disk fails in **three years**. How many nodes will be down on an arbitrary day due to disk failure?

# Fault Tolerance

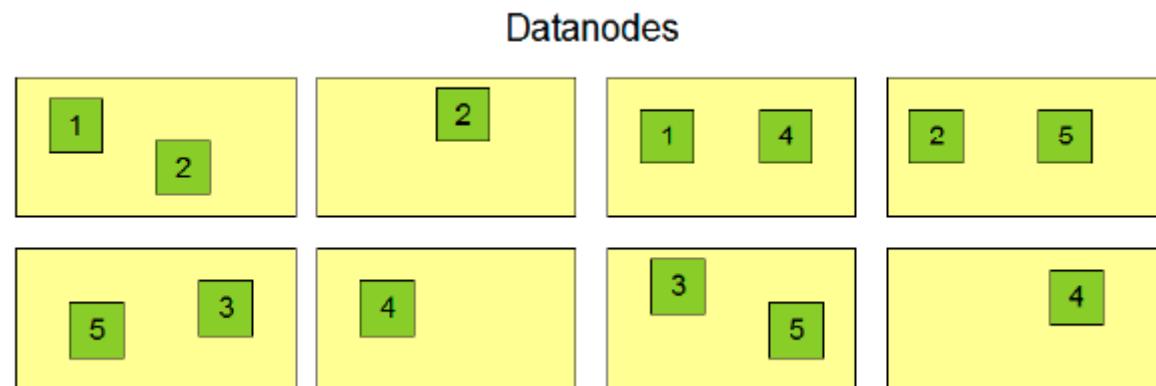


# HDFS Data Replication



Source: HDFS Architecture Guide, Dhruba Borthakur.

# Replication

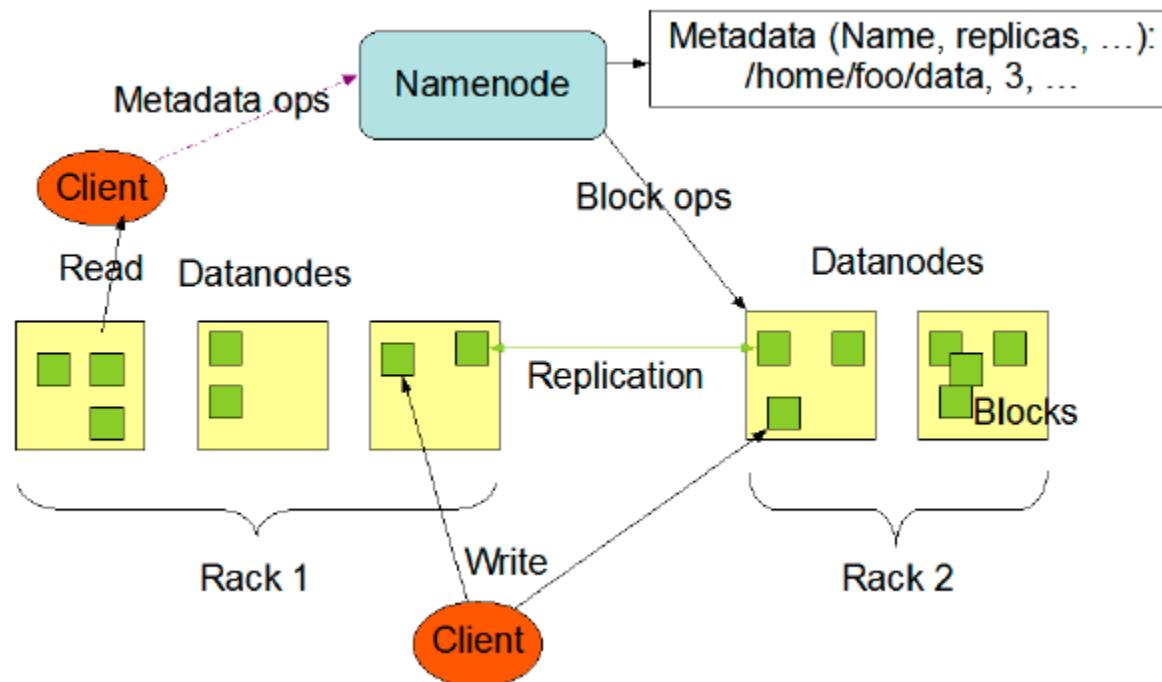


# Single Point of Failure



Is namenode a single point of failure?

Hadoop 2.0 supports primary and secondary namenodes



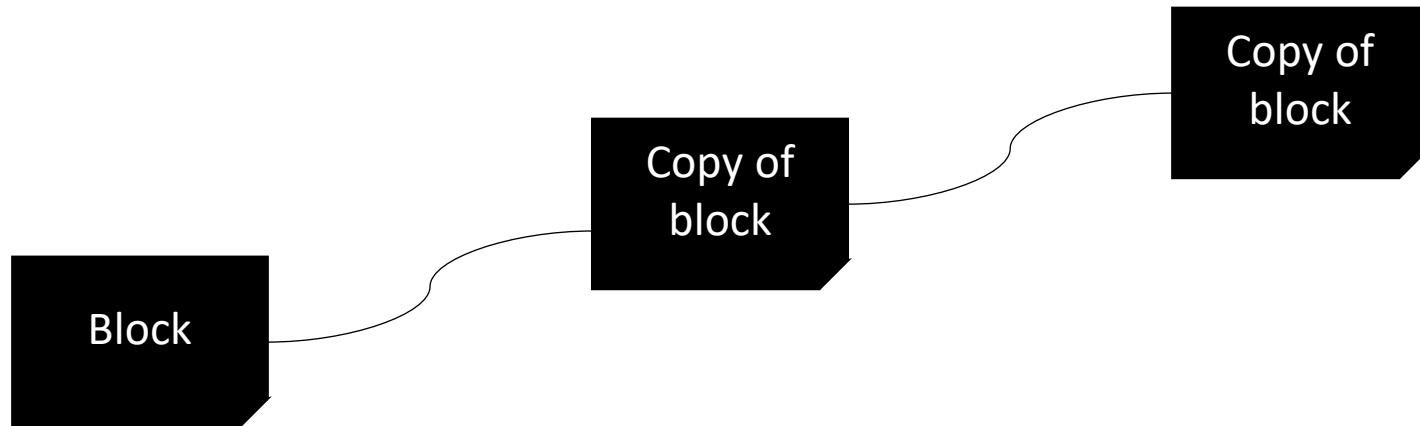
Source: HDFS Architecture Guide, Dhruba Borthakur.

# Efficient Data Access

- Write Once Read Many (WORM) model

# Write Once Read Many

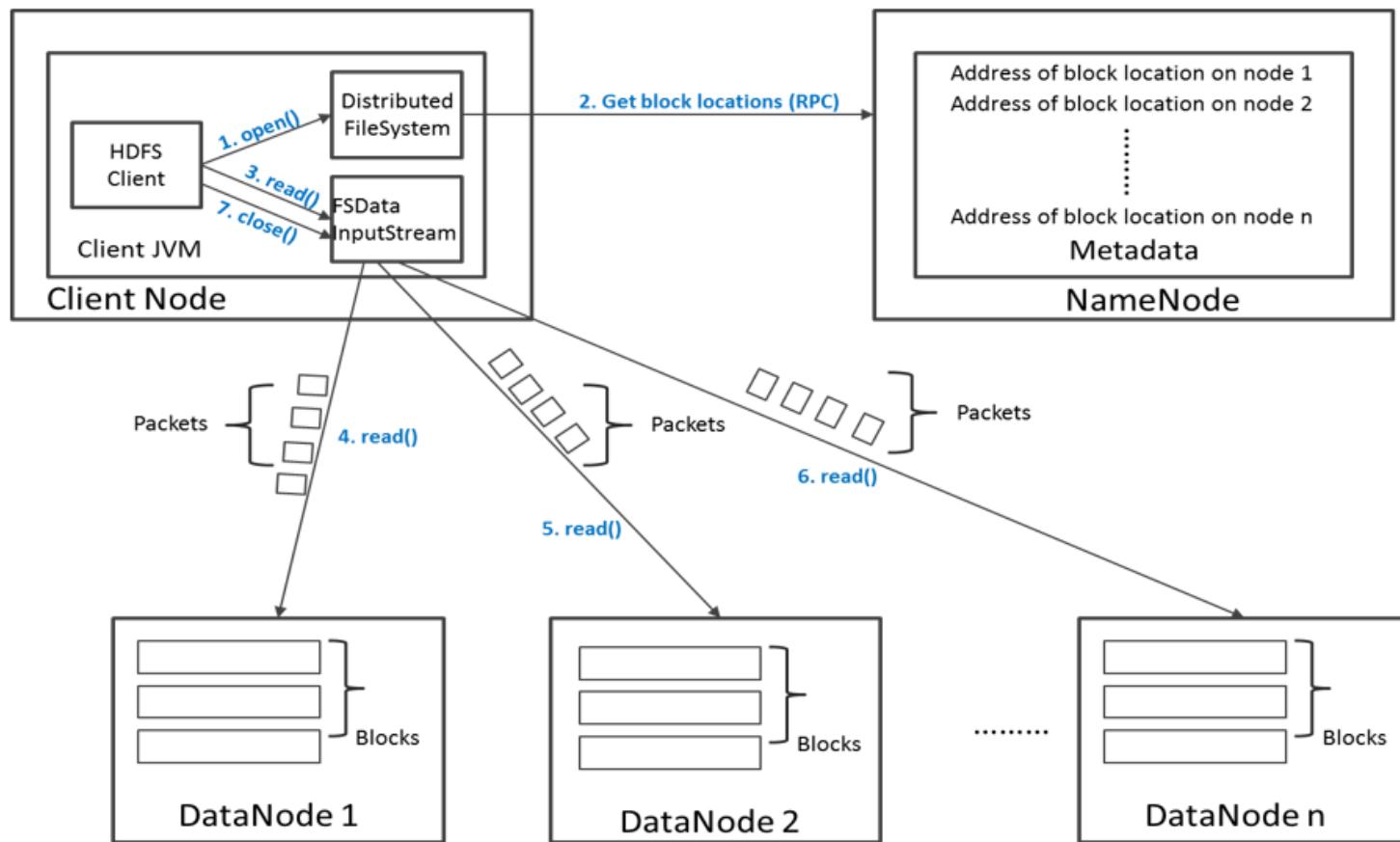
- Simplifies Data Coherency



Need to keep all the copies in sync.

- Designed for batch jobs

# HDFS – Data Read Operation



# Data Read Operation

- Client asks Namenode for block addresses
- Client accesses each block by accessing the datanodes **directly**.
- Since data is **accessed in parallel**, the reads are highly optimized.

# Data Write Operation

- Namenode **provides the address** of the datanodes
- Client **directly writes** data on the datanodes
- Datanode will **create data write pipeline**
  - First datanode copies the block to another datanode, which intern copy it to the third datanode
- Datanodes send **acknowledgment**

# Design Choices

- Default block size is 64 MB. Often used as is, or as 128 MB.
- How do you decide the right value for block size?

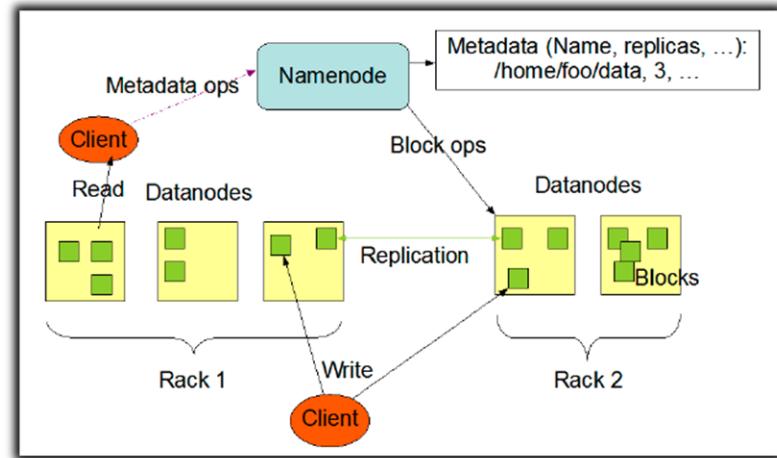
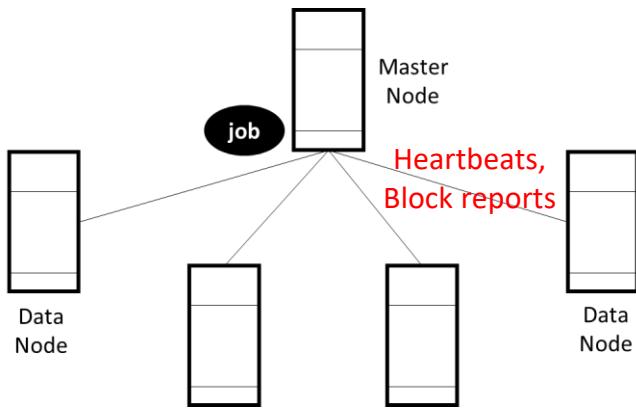
# Design Choices

- Default block size is 64 MB. Often used as is, or as 128 MB.
- Block Size – Concerns:
  - Designed to handle large files (not small files, not even large number of small files).
  - For large number of small files, namenode needs to store too much metadata.
    - Solution: Sequence files. 

# Sequence Files

- Hadoop specific file format.
- Files consisting of binary key/value pairs.
- Three types:
  - Uncompressed
  - Record Compressed
  - Block Compressed

# Summary



Distribution Transparency  
Location Transparency  
Scalability  
Fault Tolerance

Efficient Data Access  
“Write Once Read Many” (WORM) model

# DATA PROCESSING

**Venkatesh Vinayakarao**

[venkateshv@cmi.ac.in](mailto:venkateshv@cmi.ac.in)

<http://vvtesh.co.in>

---

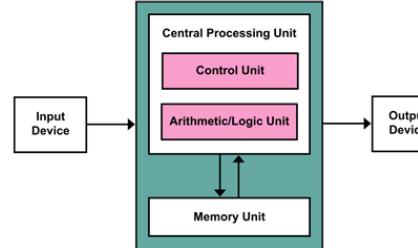
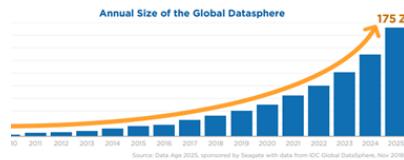
Chennai Mathematical Institute

---

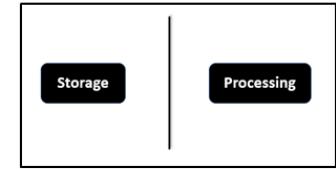
***“In God we trust; all others must bring data”.*** – William Edwards Deming.

# Recap

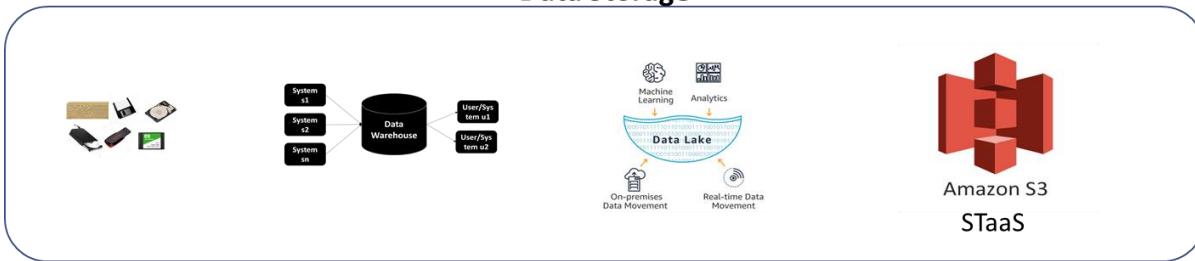
Name	Size
Byte	8 bits
Kilobyte	1024 bytes
Megabyte	1024 kilobytes
Gigabyte	1024 megabytes
Terabyte	1024 terabytes
Petabyte	1024 petabytes
Exabyte	1024 exabytes
Zettabyte	1024 zettabytes
Yottabyte	1024 yottabytes



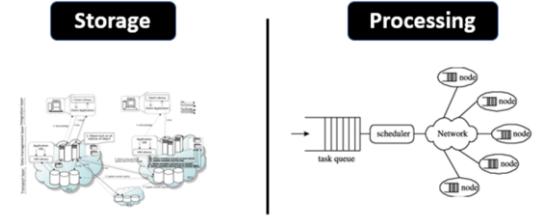
## Challenges



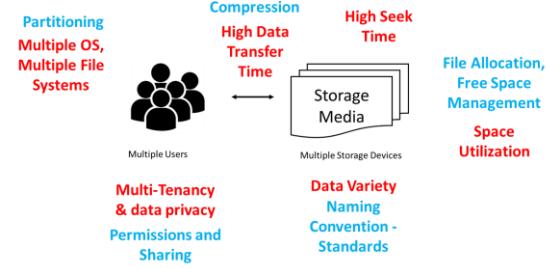
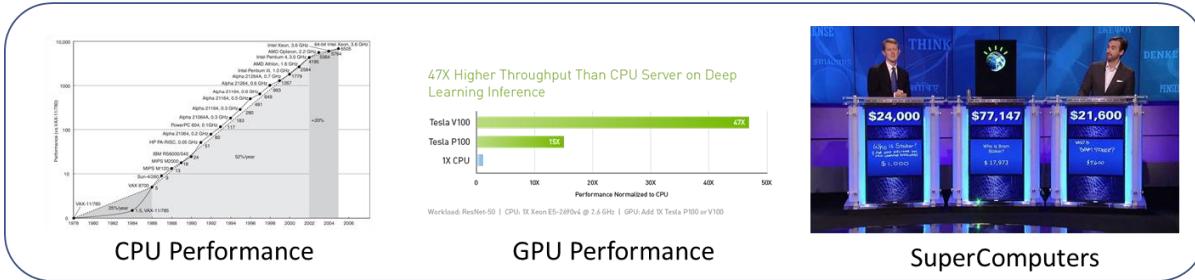
## Data Storage



Two kinds of Big Data Opportunities

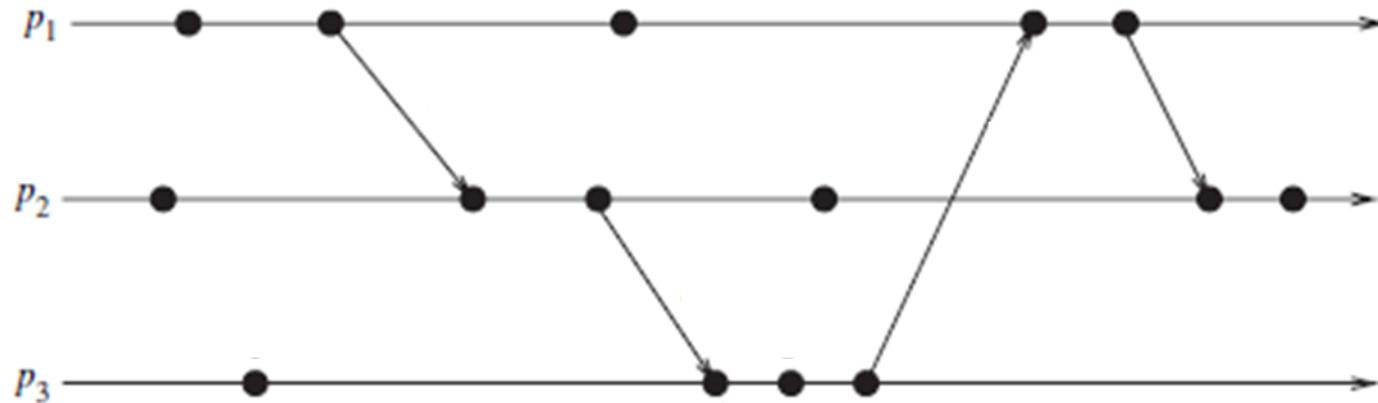


## Data Processing



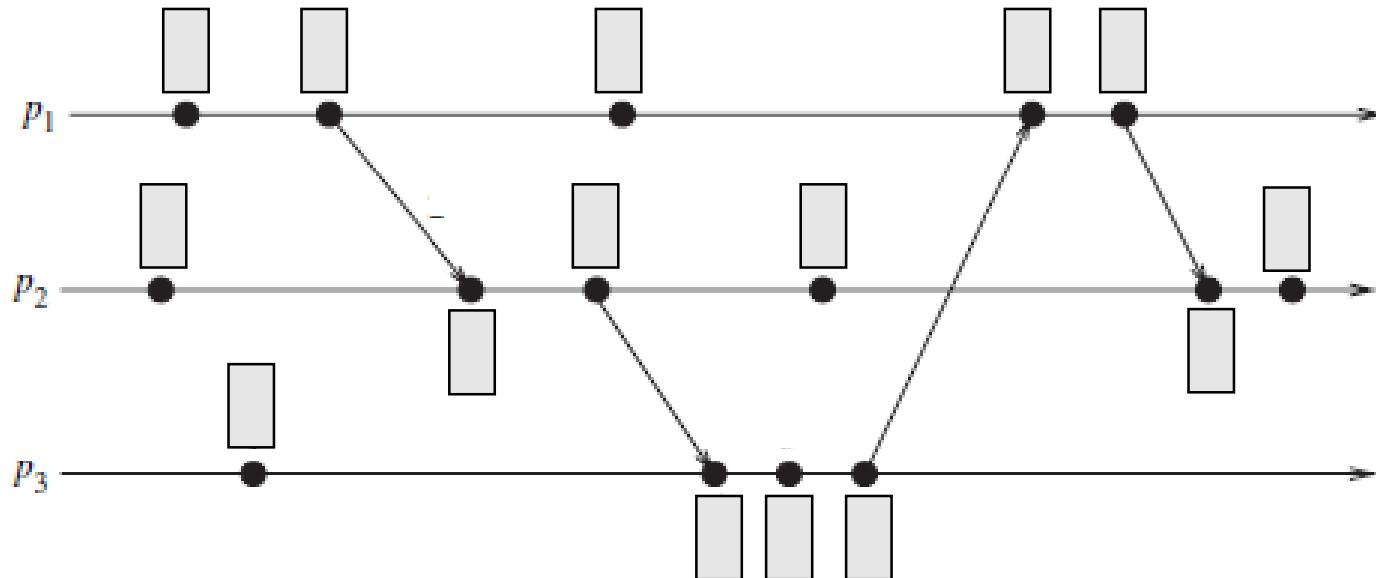
# Quiz

- Annotate the following space-time diagram with Lamport's scalar time.



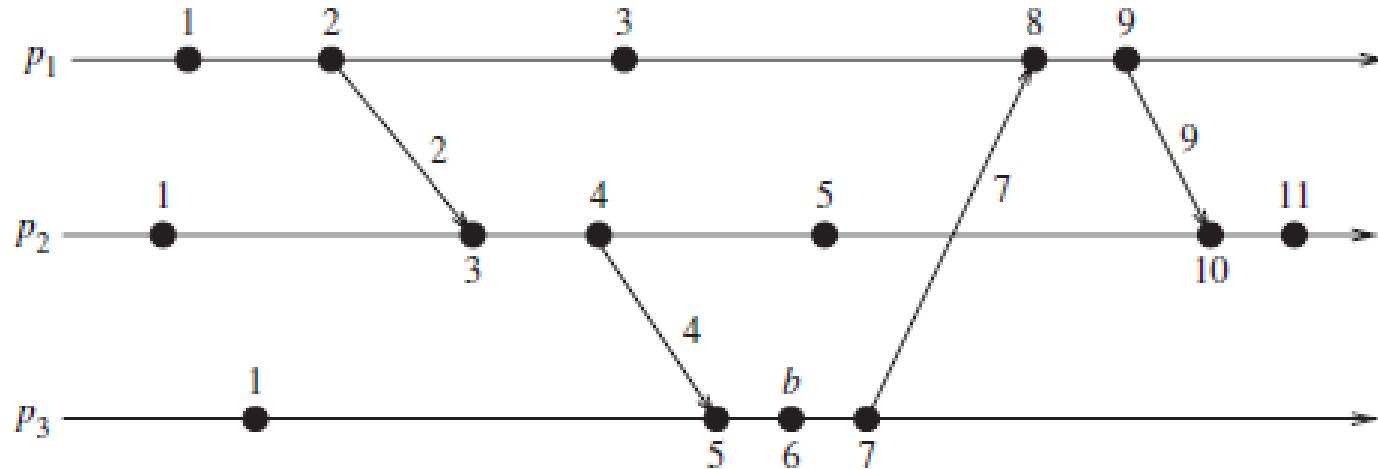
# Quiz

- Annotate the following space-time diagram with Lamport's scalar time.



# Quiz

- Annotate the following space-time diagram with Lamport's scalar time.

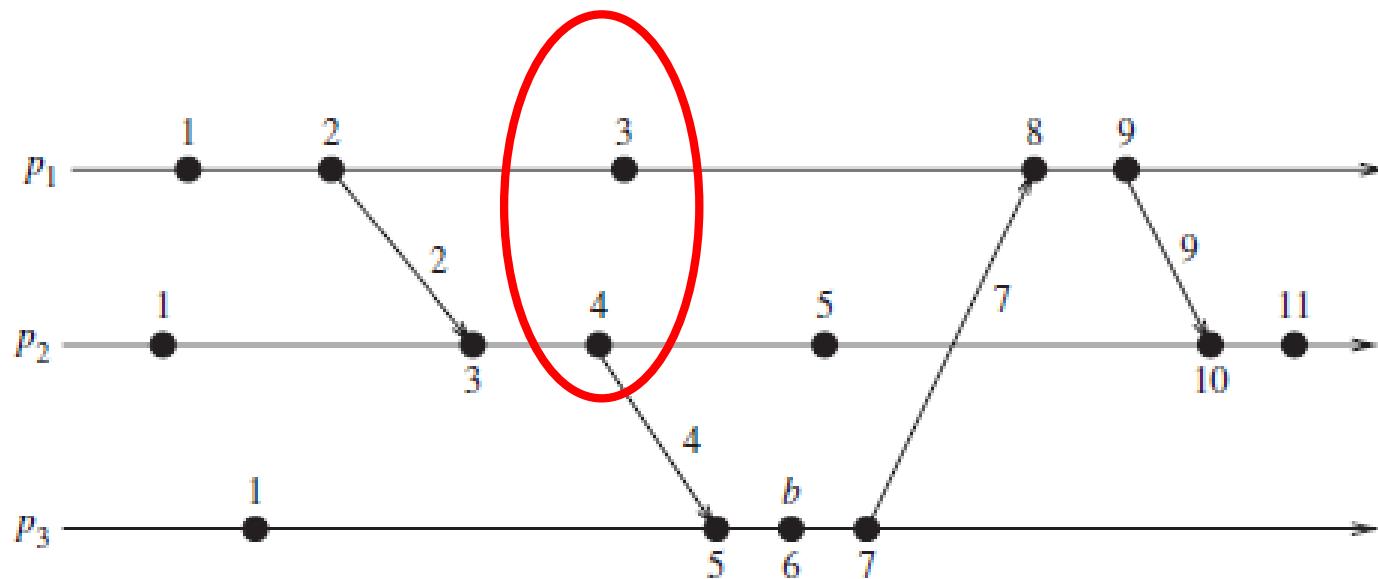


# The Problem

- Scalar clocks are consistent
  - $e_i$  happens-before  $e_j \rightarrow C(e_i) < C(e_j)$
- ...but not strongly consistent
  - $C(e_i) < C(e_j)$  does not imply  $e_i$  happens-before  $e_j$

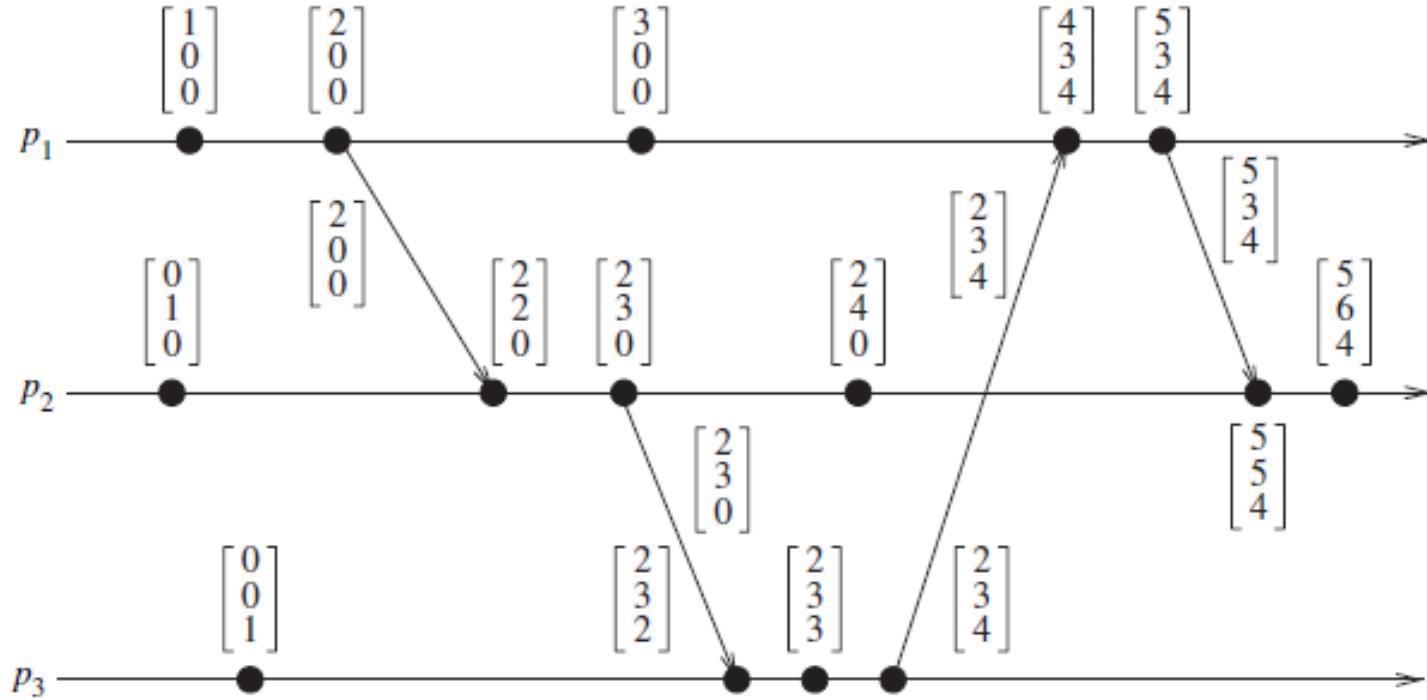
# Quiz

- Annotate the following space-time diagram with Lamport's scalar time.



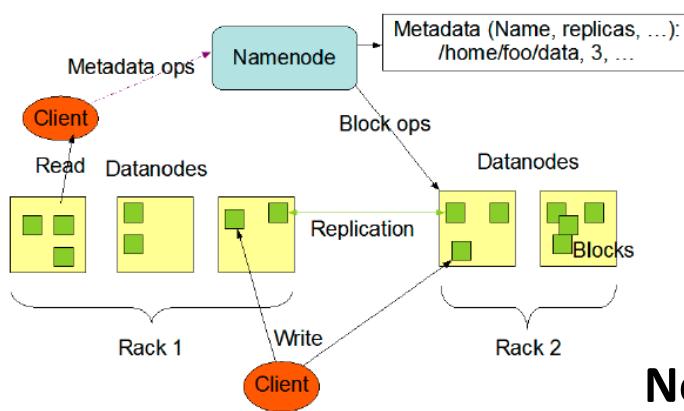
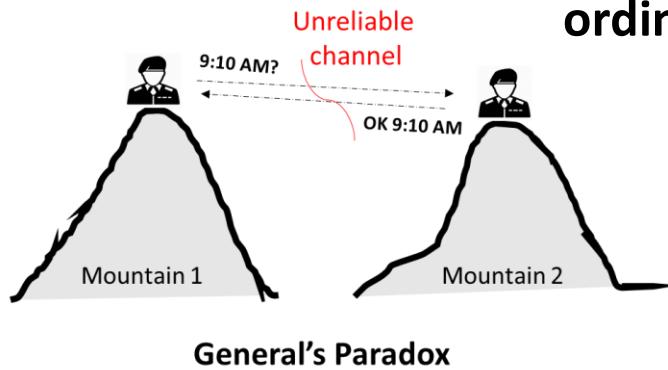
# Vector Time

- Annotated space-time diagram with vector time

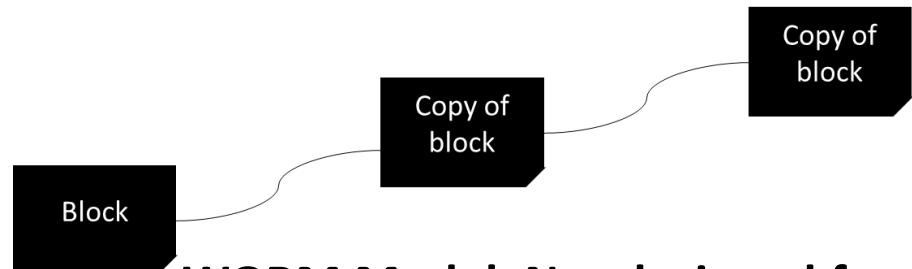


# Recap

**Not designed for co-ordination jobs.**



**Not designed for small files.**

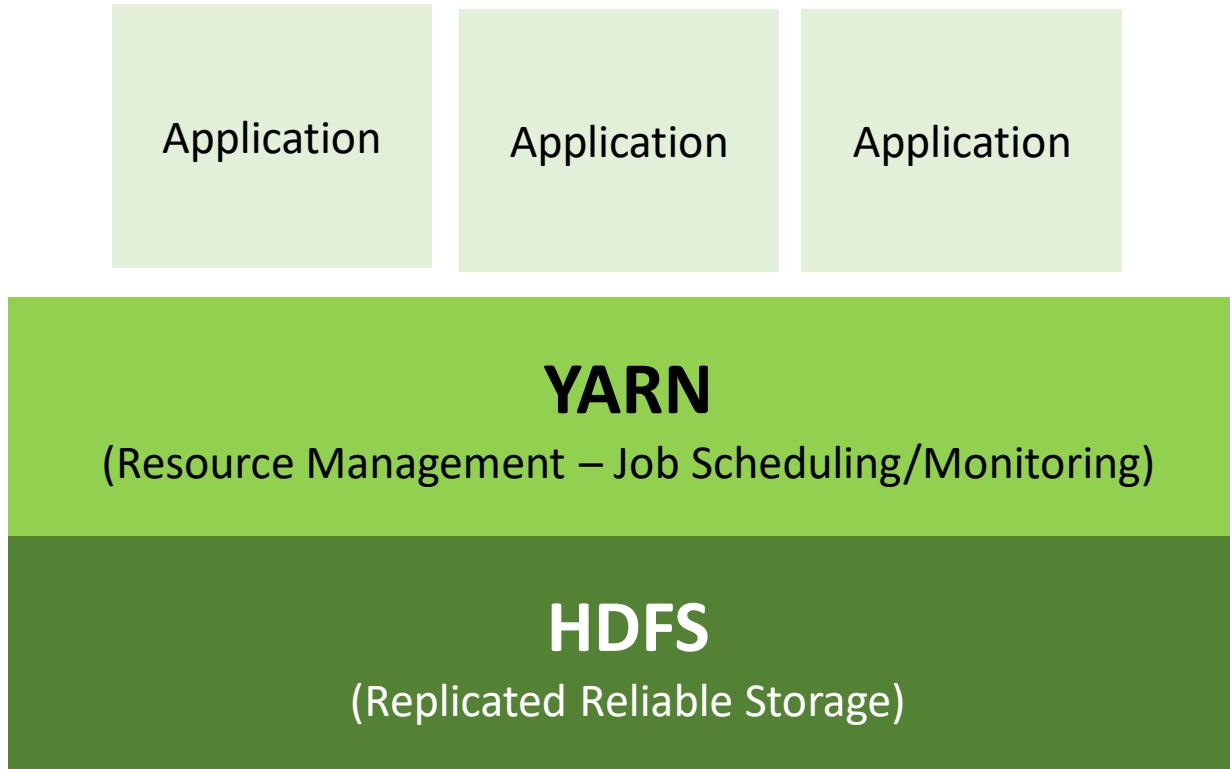


**WORM Model. Not designed for write-many (interactive) jobs.**

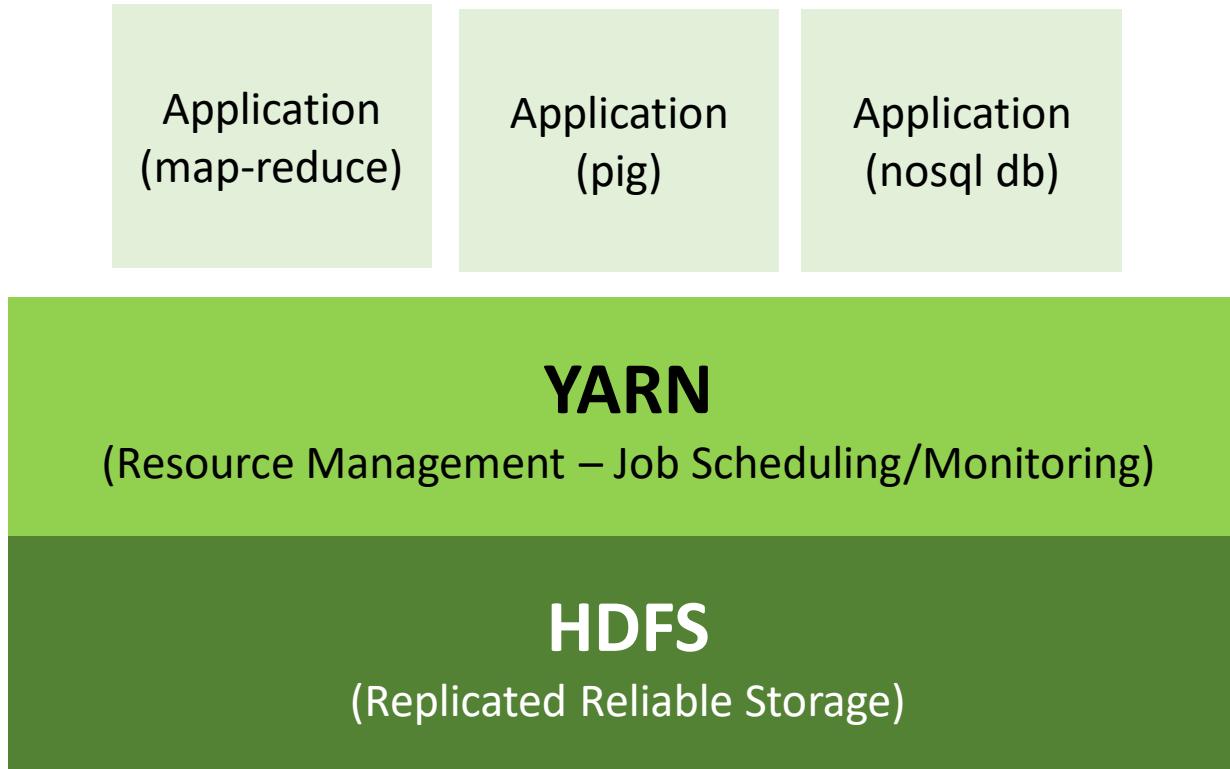
# Flex Your Brain!

## Distributed Discount Coupon Problem

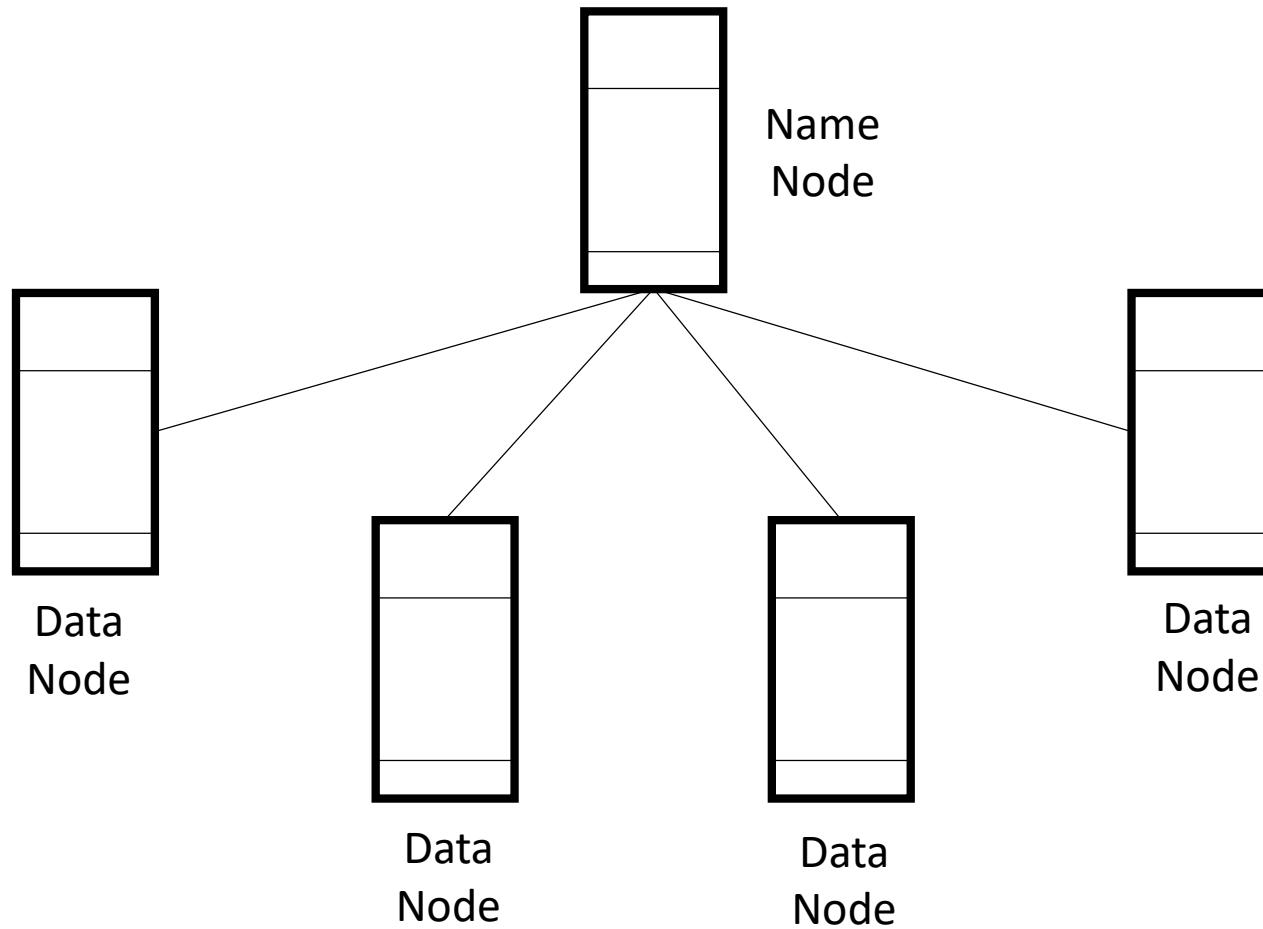
# Hadoop Eco System



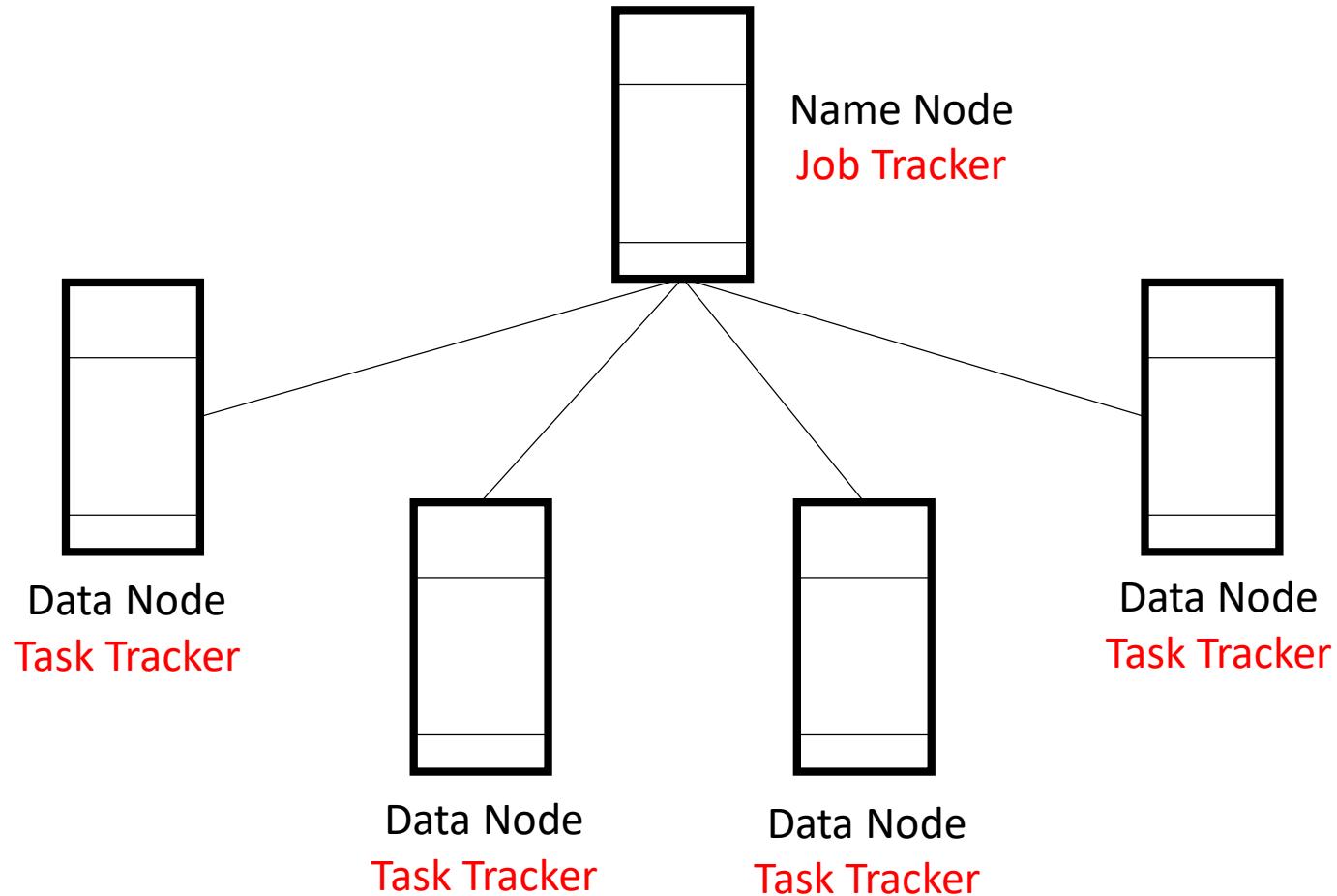
# Hadoop Eco System



# Hadoop Cluster



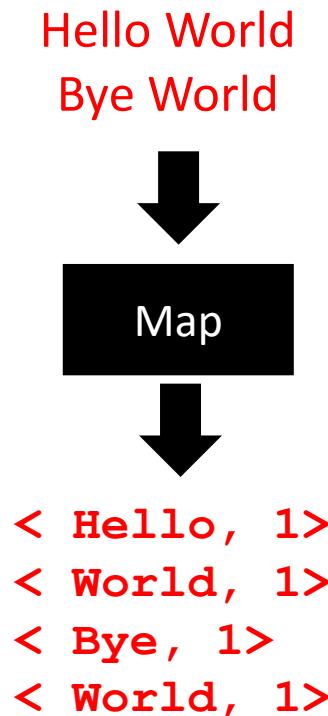
# Hadoop Cluster



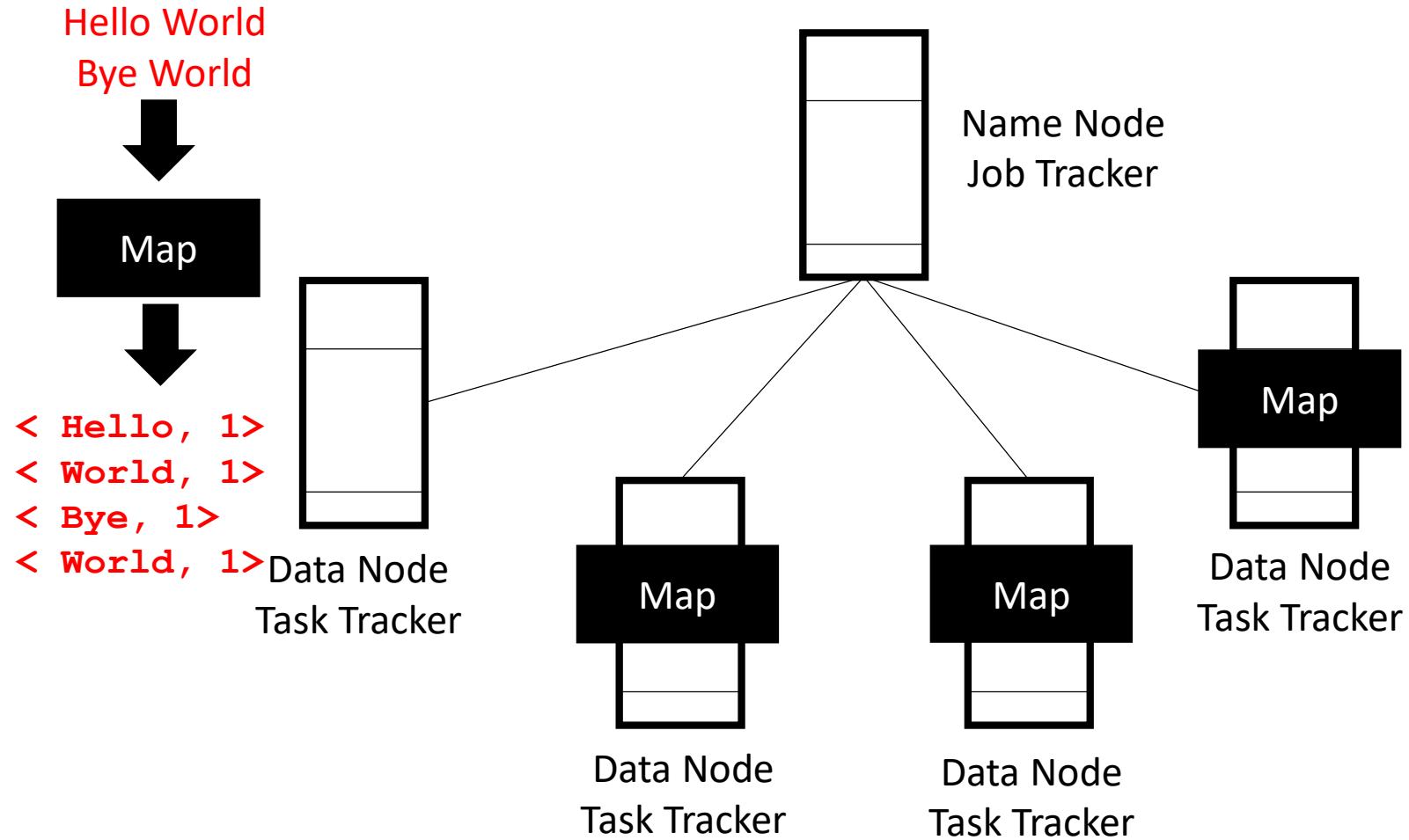
# Computing Over the Cluster

**“Moving Computation is Cheaper than Moving Data”**

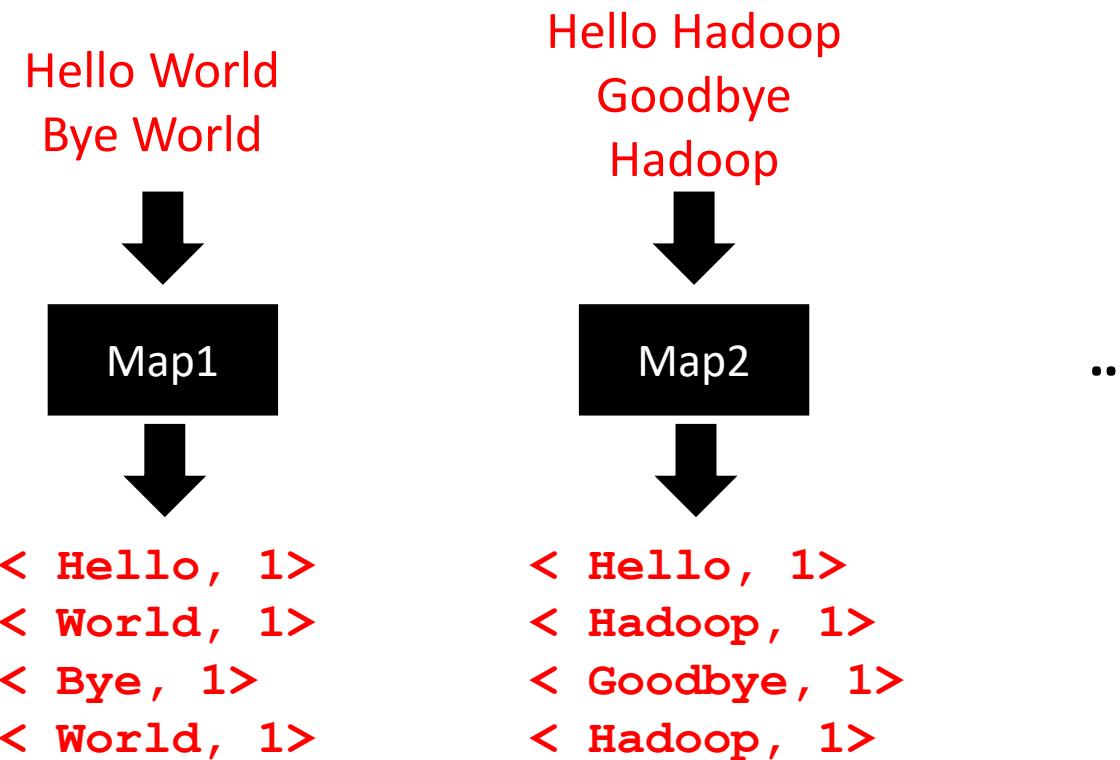
# Our Code in a Mapper



# Hadoop Cluster



# Our Code in a Mapper



Sorted output of the mappers are associated with reducers.

# Map-Reduce Processing

Mapper

```
< Hello, 1>
< World, 1>
< Bye, 1>
< World, 1>
```

```
< Hello, 1>
< Hadoop, 1>
< Goodbye, 1>
< Hadoop, 1>
```

1

Shuffle and Sort

```
< Bye, 1>
< Hadoop, 1>
< Hadoop, 1>
< Hello, 1>
< Hello, 1>
< GoodBye, 1>
```

```
< World, 1>
< World, 1>
```

2

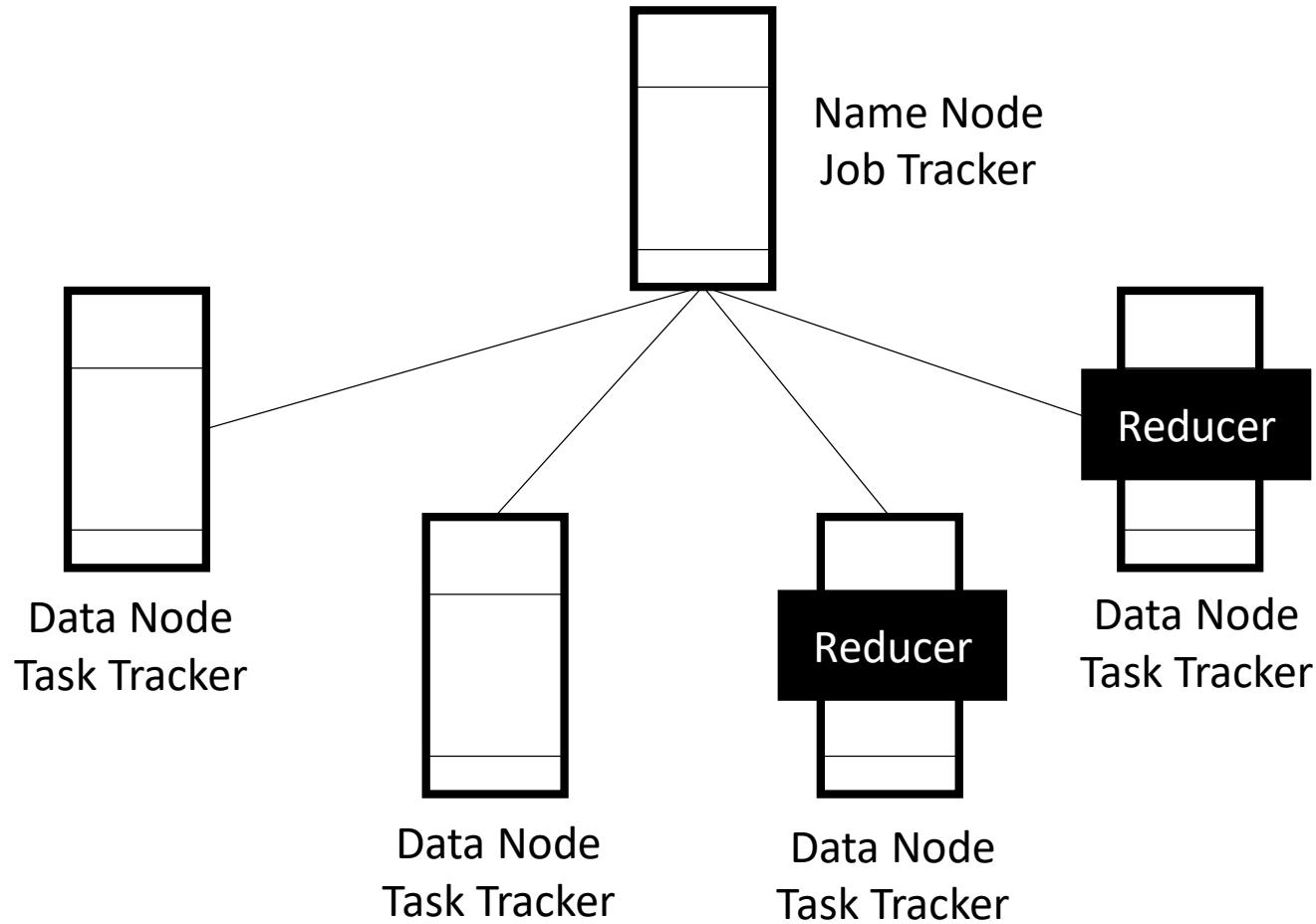
Reduce

```
< Bye, 1>
< Hadoop, 2>
< Hello, 3>
< GoodBye, 1>
```

```
< World, 2>
```

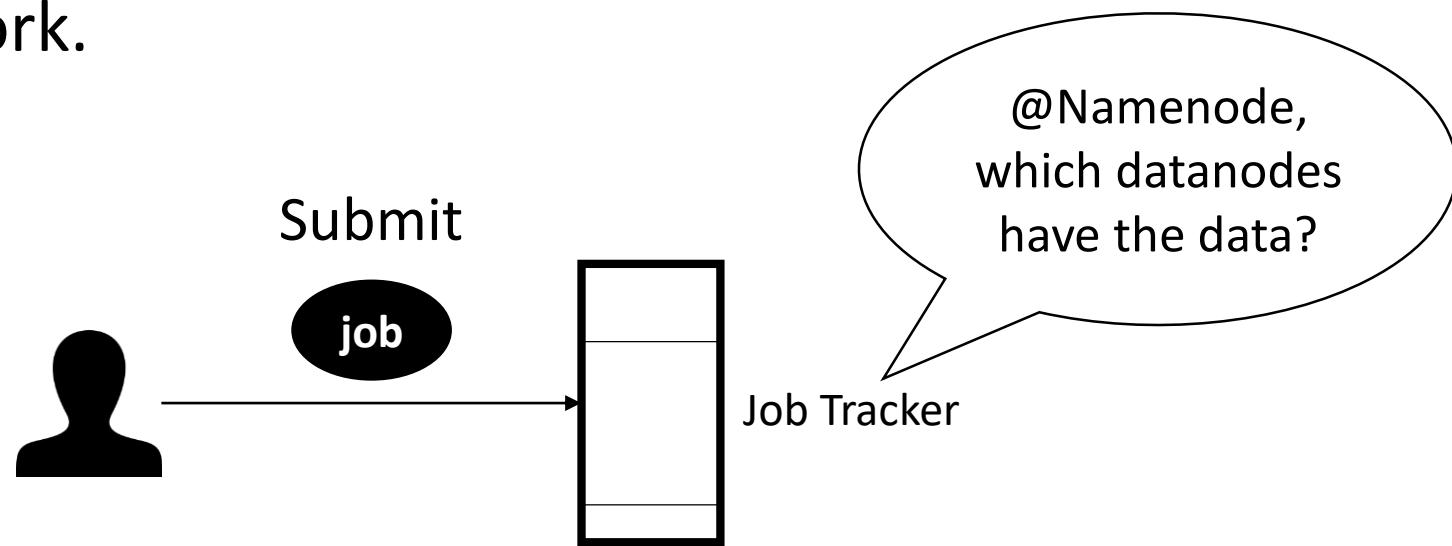
3

# Hadoop Cluster



# A Hadoop Map-Reduce Job

- Write the “map” code
- Write the “reduce” code
- Submit the map and reduce code to Hadoop framework.



# Submitting a Map-Reduce Job

hadoop jar

`/usr/joe/wordcount.jar`

`org.myorg.WordCount`

`/usr/joe/wordcount/input`

`/usr/joe/wordcount/output`

# Programming Language

- The Hadoop framework itself is mostly written in the Java programming language.
- Hadoop Streaming is a utility that comes with Hadoop.
  - Allows the use of Python for map-reduce coding.

```
hadoop jar
  /usr/lib/hadoop-2.2.0/share/hadoop/tools/lib/hadoop-streaming-2.2.0.jar
  -file /home/cmi/mapper.py
  -mapper mapper.py
  -file /home/cmi/reducer.py
  -reducer reducer.py
  -input /user/cmi/word
  -output /user/cmi/Wordcount
```

# Mapper

```
public void map(Object key, Text value, Context context
                ) throws IOException, InterruptedException {
    StringTokenizer itr = new StringTokenizer(value.toString());
    while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
    }
}
```

See <https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html> for details.

# Reducer

```
public void reduce(Text key, Iterable<IntWritable> values,
                  Context context
                  ) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
        sum += val.get();
    }
    result.set(sum);
    context.write(key, result);
}
```

# Create a Job

```
public static void main(String[] args) throws Exception {  
    Configuration conf = new Configuration();  
    Job job = Job.getInstance(conf, "word count");  
    job.setJarByClass(WordCount.class);  
    job.setMapperClass(TokenizerMapper.class);  
    job.setCombinerClass(IntSumReducer.class);  
    job.setReducerClass(IntSumReducer.class);  
    job.setOutputKeyClass(Text.class);  
    job.setOutputValueClass(IntWritable.class);  
    FileInputFormat.addInputPath(job, new Path(args[0]));  
    FileOutputFormat.setOutputPath(job, new Path(args[1]));  
    System.exit(job.waitForCompletion(true) ? 0 : 1);  
}
```

# Submit Job to Hadoop

```
$ bin/hadoop jar wc.jar WordCount /user/joe/wordcount/input  
/user/joe/wordcount/output
```

```
$ bin/hadoop fs -ls /user/joe/wordcount/input/  
/user/joe/wordcount/input/file01  
/user/joe/wordcount/input/file02

$ bin/hadoop fs -cat /user/joe/wordcount/input/file01  
Hello World Bye World

$ bin/hadoop fs -cat /user/joe/wordcount/input/file02  
Hello Hadoop Goodbye Hadoop
```

# Output

```
$ bin/hadoop fs -cat /user/joe/wordcount/output/part-r-00000
Bye 1
Goodbye 1
Hadoop 2
Hello 2
World 2
```

## **Key and Value**

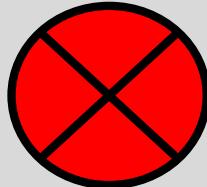
Together, they make a lovely pair!

# Role of Key-Value Pairs

- Map-Reduce coding
- Key-Value datastores (Amazon DynamoDB)
- Windows registry entries
- JSON files
- Hashing
- ...

# Summary

## When not to use Hadoop?



No Interactive Jobs  
No Jobs Requiring Co-ordination  
No Small Files

## Hadoop Architecture

Application (map-reduce) Application (pig) Application (nosql db)

**YARN**  
(Resource Management – Job Scheduling/Monitoring)

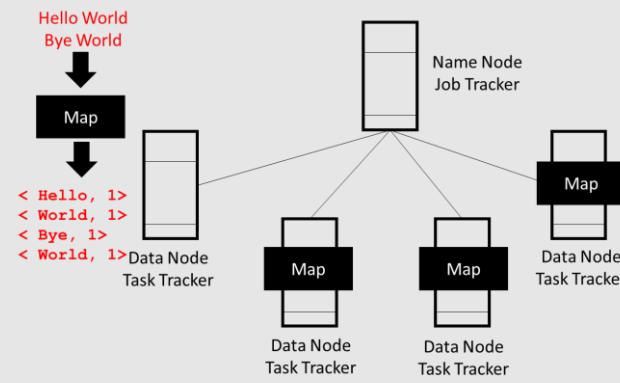
**HDFS**  
(Replicated Reliable Storage)

## Map-reduce Model

Map

Shuffle and Sort

Reduce



# TUTORIAL 3

---

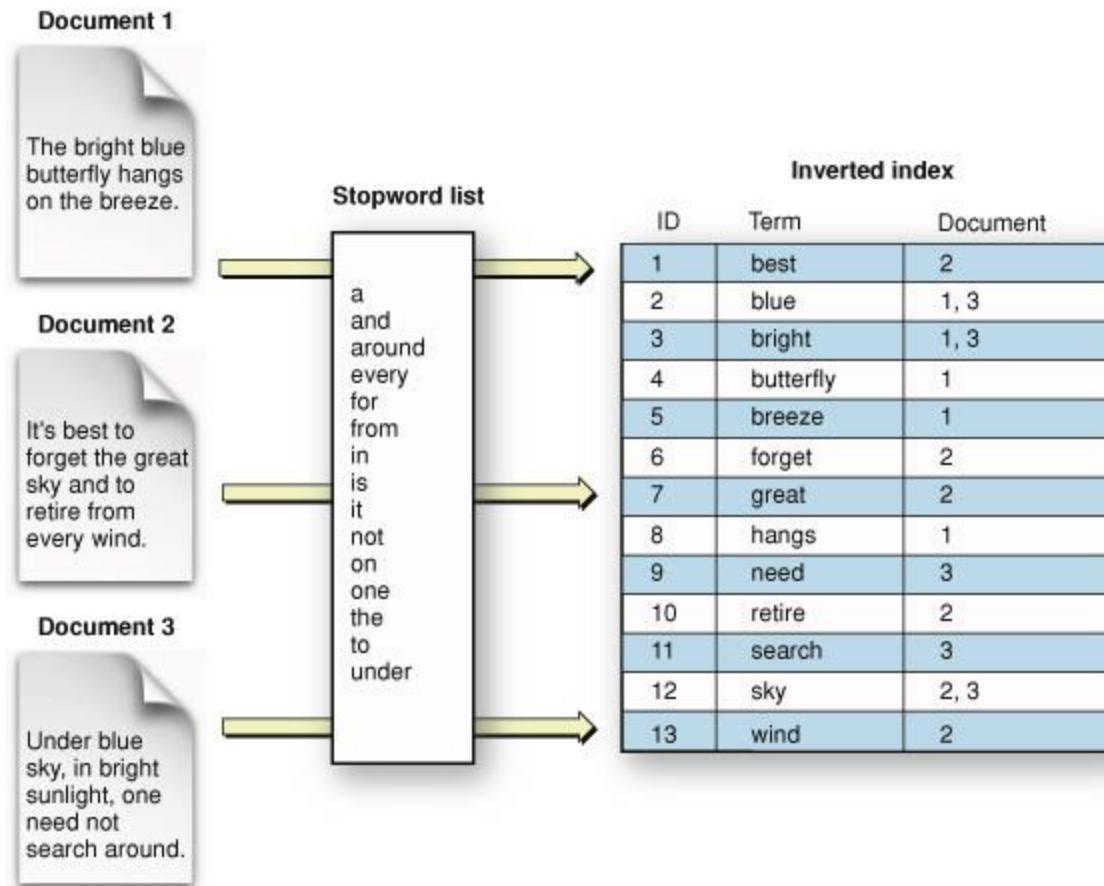
- MapReduce Examples
- Apache Solr

by Suchitra Jayaprakash  
[suchitra@cmi.ac.in](mailto:suchitra@cmi.ac.in)

# MAPREDUCE

- MapReduce is a programming paradigm with two phases:
  - Map
  - Shuffle & Sort
  - Reduce

# Inverted Index with MapReduce



# Implementation

- WORD\_RE = re.compile(r"[a-zA-Z]{2,}\b")
- class MRInvertedIndex(MRJob):
  - def mapper(self, \_, line):
    - ## getting input file name
    - filepath = os.environ['map\_input\_file']
    - filename=filepath.split('/')[-1]
    - for word in WORD\_RE.findall(line):
      - yield (word.lower(), filename)
  - def reducer(self, word, filenames):
    - yield (word, ",".join(list(set(filenames))))

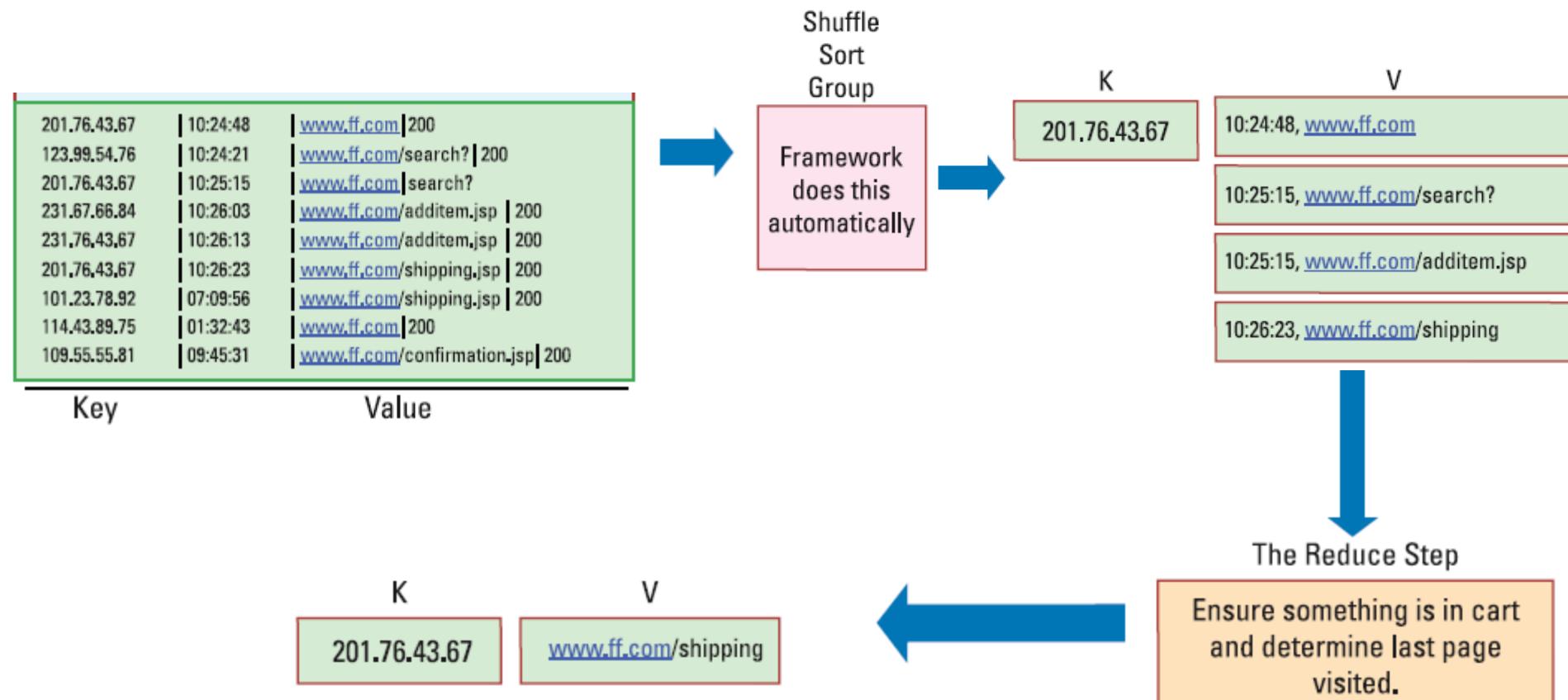
# Capture useful insights from Log data

- An ecommerce web site collects click stream data as log file.

201.76.43.67	10:24:48	<a href="http://www.ff.com">www.ff.com</a>   200
123.99.54.76	10:24:21	<a href="http://www.ff.com/search?">www.ff.com/search?</a>   200
201.76.43.67	10:25:15	<a href="http://www.ff.com/search?">www.ff.com/search?</a>
231.67.66.84	10:26:03	<a href="http://www.ff.com/additem.jsp">www.ff.com/additem.jsp</a>   200
231.76.43.67	10:26:13	<a href="http://www.ff.com/additem.jsp">www.ff.com/additem.jsp</a>   200
201.76.43.67	10:26:23	<a href="http://www.ff.com/shipping.jsp">www.ff.com/shipping.jsp</a>   200
101.23.78.92	07:09:56	<a href="http://www.ff.com/shipping.jsp">www.ff.com/shipping.jsp</a>   200
114.43.89.75	01:32:43	<a href="http://www.ff.com">www.ff.com</a>   200
109.55.55.81	09:45:31	<a href="http://www.ff.com/confirmation.jsp">www.ff.com/confirmation.jsp</a>   200

- Identify key factors behind abandoned shopping carts

# MapReduce Implementation



# Build a search engine

- Over 10 billion products are sold on an ecommerce web site.
- Company wants to build search tool on their site.

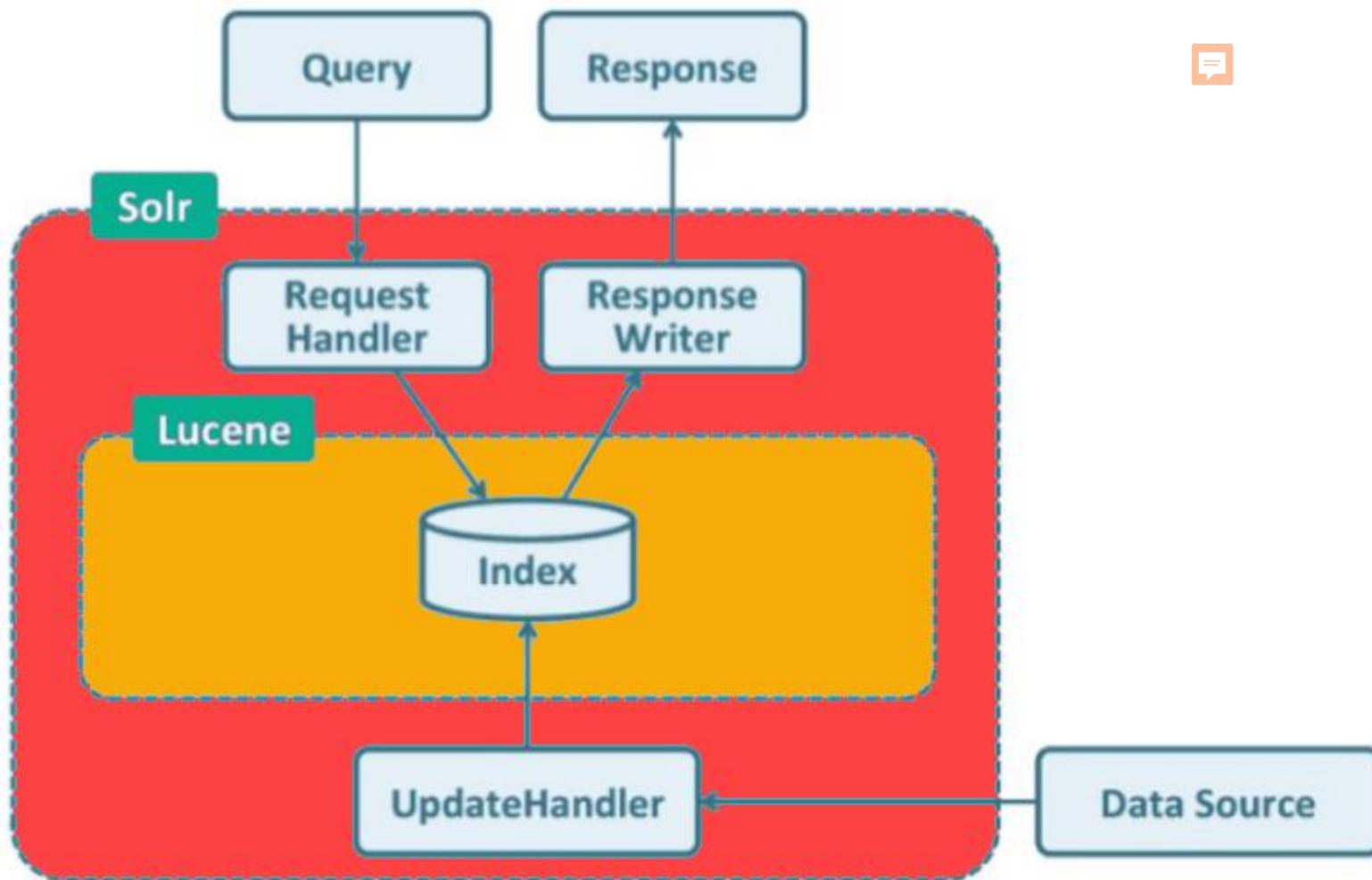
# WHAT IS SOLR ?

- Open-source REST-API based Enterprise Real-time Search and Analytics Engine Server.
- Highly scalable search applications
- Ready to deploy
- Built using Apache Lucene Framework , Java
- Optimized to search large volumes of text-centric data
- Document-based NoSQL Data Store
- Main Functionality – Indexing & Searching

# SOLR FEATURES

- Full text search
- Faceted Navigation
- Spell check
- Hit highlighting
- Relevant result
- Recommendation
- Geo-Spatial Search
- supports Distributed and Cloud Technology
- Built in Authentication & authorisation
- Learning to rank

# SOLR ARCHITECTURE



# INSTALL SOLR LOCALLY

- Java Runtime Environment version 8 or greater
- Download Solr from lucene site and unzip for installation.
- Solr Instance is an instance a Solr running in the JVM
- Start/Stop Apache solr
  - bin/solr start
  - bin/solr stop
- Access the Solr Admin User Interface
  - <http://localhost:8983/solr/>
-

# CREATE CORE

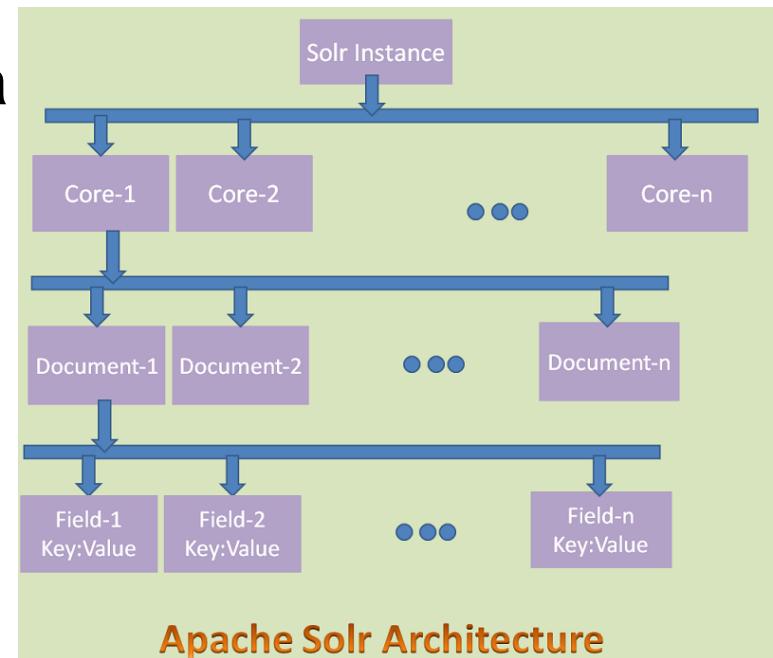
- A Solr Core is a running instance of a Lucene index that contains all the Solr configuration files required to use it.

Core = an instance of Lucene Index + Solr configuration

- We need to create a Solr Core to perform operations like indexing and analyzing
- `solr create_core -c XXXX -p 8983`

# INDEXING DATA

- Solr can ingest many types of files like .csv .json, .xml, .html,.pdf
  - We can add data to Solr index via
    - Solr Web Interface.
    - Client APIs like Java, Python, etc.
    - post tool.
  - Use Admin console or POST
  - command to upload document
- ```
java -Dc=XXXX -Durl=http://localhost:8983/solr/XXXX/update/ -Dtype=application/xml -jar ..../example/exemplledocs/post.jar "../example/films/films.xml"
```



# CONFIGURATION FILES

- solr.xml - server instance configurations
- core.properties - core configurations such as names, locations and files in the core
- conf/solrconfig.xml - core configurations for field guessing, directories, query settings, spell checking, keyword highlighting and query response formats
- conf/managed-schema - core configurations for field processing

# SEARCH IN ADMIN UI

- Open Solr Admin Console: <http://localhost:8983/solr/>
- Select the core module
- Click on “Query” from the left navigation
- Enter search query in q text box & df (default field)
- Click on “Execute Query” button
- It retrieves matching document from selected core

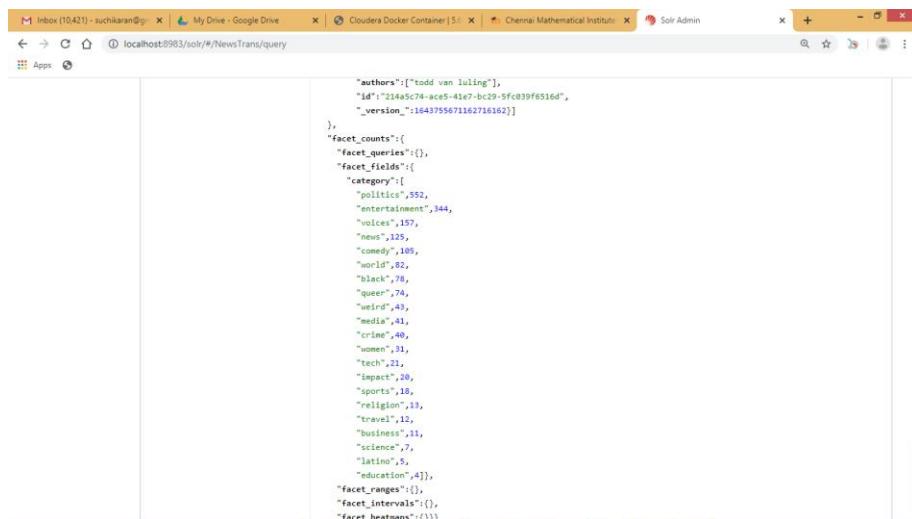
# SEARCH ....

- You can also do search using rest api call
  - [http://localhost:8983/solr/<corename>/select?indent=on&q=\\*&wt=json](http://localhost:8983/solr/<corename>/select?indent=on&q=*&wt=json)
  - http://localhost:8983/solr/<corename>/select?q=black&mlt=true&mlt.fl=genre&mlt.mindf=1&mlt.mintf=1&fl=id,score&df=genre
  - http://localhost:8983/solr/<corename>/spell?spellcheck=true&qt=spellchecker&spellcheck.accuracy=0.5&spellcheck.collate=true&q=Cime

# Sample Output

## Faceting :

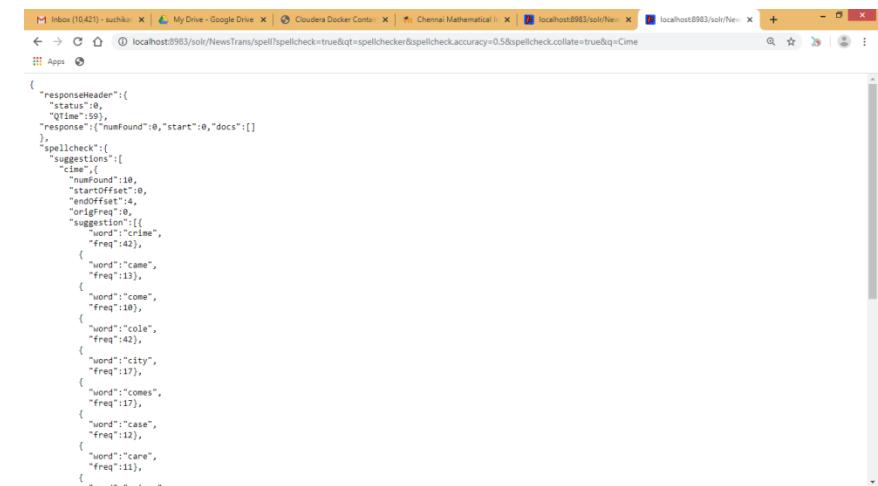
<http://localhost:8983/solr/NewsTrans/select?facet.field=category&facet.on&q=%22crime%22>



```
{
  "authors": ["todd van luling"],
  "id": "21a4a5c7-ac5-41e7-bc29-5fc039f6516d",
  "version": 1643755671162716162
},
"facet_counts": {
  "facet_queries": {},
  "facet_fields": {
    "category": [
      "politics", 552,
      "entertainment", 344,
      "voices", 157,
      "news", 125,
      "comedy", 105,
      "world", 82,
      "black", 76,
      "queer", 74,
      "seirid", 43,
      "media", 41,
      "crime", 40,
      "women", 31,
      "tech", 21,
      "impact", 20,
      "sports", 18,
      "religion", 13,
      "travel", 12,
      "books", 11,
      "science", 7,
      "latino", 5,
      "education", 41
    ],
    "facet_ranges": {},
    "facet_intervals": {},
    "facet_heatmaps": {}
  }
}
```

## Spellcheck :

<http://localhost:8983/solr/NewsTrans/spell?spellcheck=true&qt=spellchecker&spellcheck.accuracy=0.5&spellcheck.collate=true&q=Cime>



```
{
  "responseHeader": {
    "status": 0,
    "QTime": 199,
    "response": {
      "numFound": 0,
      "start": 0,
      "docs": []
    }
  },
  "spellcheck": {
    "suggestions": [
      {
        "word": "cime",
        "numFound": 19,
        "startOffset": 0,
        "endOffset": 4,
        "origFreq": 0,
        "suggestions": [
          {
            "word": "crime",
            "freq": 42,
            "freq": 42,
            "freq": 42,
            "freq": 13,
            "freq": 10,
            "freq": 42,
            "freq": 17,
            "freq": 17,
            "freq": 12,
            "freq": 11
          }
        ]
      }
    ]
  }
}
```

# Quiz 3

- Identity mapper is used to do "nothing". It converts the input to output "as is". Assume we have a long list of words. Can you suggest where identity mapper on this word list could be used?
  - A) Sorting
  - B) Searching
  - C) Aggregating
  - D) All of the above
  - E) None of the above

**THANK YOU**

# Object Oriented Analysis and Design (OOAD)

## An Introduction

**Venkatesh Vinayakarao**

[venkatesh.v@iiits.in](mailto:venkatesh.v@iiits.in)

<http://vvtesh.co.in>

---

Assistant Professor  
Indian Institute of Information Technology, Sri City, Chittoor

---

The art of simplicity is a puzzle of complexity. –**Douglas Horton.**



# Agenda

- Introduction
  - Why is large-scale software engineering so complex?
  - How to address this complexity?
- Object Oriented Analysis and Design (OOAD)
  - Hierarchy and Abstraction
  - Classes and Objects
- Unified Modeling Language (UML)

# House Rule

- Raise your hand if you know the answer.
- Do not shout out unless offered a chance by your instructor.



**Software Engineering (SE) is very  
challenging! Why?**

# Challenges in SE

- Over-specification and under-specification
- Relying on star performers
- Weak personnel and problem employees
- Requirements creep
- Code not even understood by the author – done by trial and error
- Designing, Coding, ...
- ...
- .... and >100 more!

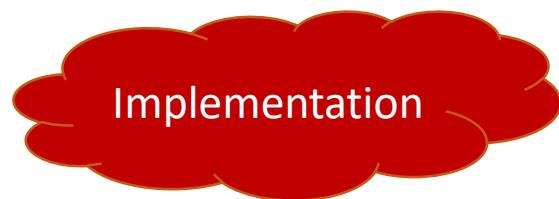
# Challenges in RE

**How to deal with these challenges?**

Challenges in the  
Problem Space  
(Analysis)



Challenges in the  
Solution Space  
(Design)



## How to tame complexity?

Can we learn from the world around us?

# How to get All India Rank 1 in JEE?

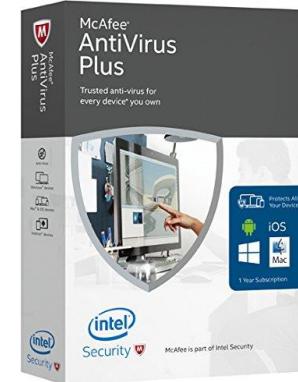


\*Am not AIR 1 in JEE. Just for fun.

# How did we humans build this?



# We could build these too!



## How to tame complexity?

Can we learn from the world around us?

# Taming Complexity

## Key Principles

1. Hierarchy
2. Abstraction
3. Keeping Related Things Together
4. Polymorphism

# Do you recognize this?

|                                                 |            |
|-------------------------------------------------|------------|
| <b>3. The Loop Control Structure</b>            | <b>97</b>  |
| Loops                                           | 98         |
| The <i>while</i> Loop                           | 99         |
| Tips and Traps                                  | 101        |
| More Operators                                  | 105        |
| The <i>for</i> Loop                             | 107        |
| Nesting of Loops                                | 114        |
| Multiple Initialisations in the <i>for</i> Loop | 115        |
| The Odd Loop                                    | 116        |
| The <i>break</i> Statement                      | 118        |
| The <i>continue</i> Statement                   | 120        |
| The <i>do-while</i> Loop                        | 121        |
| Summary                                         | 124        |
| Exercise                                        | 124        |
| <b>4. The Case Control Structure</b>            | <b>135</b> |
| Decisions Using <i>switch</i>                   | 136        |
| The Tips and Traps                              | 140        |
| <i>switch</i> Versus <i>if-else</i> Ladder      | 144        |
| The <i>goto</i> Keyword                         | 145        |
| Summary                                         | 148        |
| Exercise                                        | 149        |

**Table of Contents of C Programming Book**

# Taking These Ideas to Software Engineering!

Object Oriented Analysis and Design

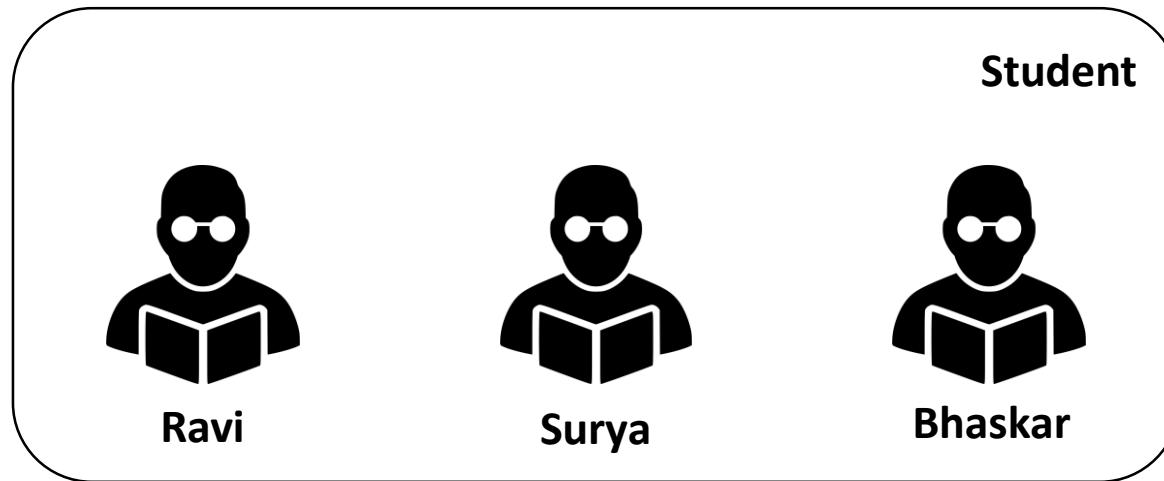
# Objects are Everywhere!

- We are inherently **Object Oriented!**
- An Example Object:
  - Identity (Venkatesh)
  - Data (Height = 5' 6'', Weight = 80 KG\*, ...)
  - Behavior (Lecture, Put Students to Sleep, ...)

Objects have **Identity**, **Data** and  
**Behavior** associated with them.

\*May be slightly more. ☺

# Abstraction



“*Student*” is a **class**.

The **objects** namely “*Ravi*”, “*Surya*” and “*Bhaskar*”  
belong to the “*Student*” **class**.

# Abstraction

- Classroom
    - Classroom 101
  - Student
    - Student1
    - Student2
    - ...
  - Faculty
    - Faculty1
  - Electrical Fitting
    - Light
      - Power Saver Light
        - PSL 1
        - PSL 2 ... and so on.
- 
- A class called “Student”.
- Objects are instances of a class.

# Class Vs. Object

- Object has:
  - Identity (Jimmy).
  - Data (four legged).
  - Behavior (barks).
- Class has:
  - Data (Dogs are four legged [quadruped]).
  - Behavior (Dogs bark).

# Quiz

- Which of the following is not a “class”?
  - Employee
  - Train
  - Vehicle
  - India



# Quiz

- Which of the following is not a “class”?
  - Employee
  - Train
  - Vehicle
  - India

# Quiz

- Which of the following is not an “Object”?
  - LG Q6 Mobile Phone
  - Venkatesh V
  - Pinakini Express
  - Employee with ID 231



# Quiz

- Which of the following is not an “Object”?
  - **LG Q6 Mobile Phone**
  - Venkatesh V
  - Pinakini Express
  - Employee with ID 231

**Two Volunteers Please...**



# Describe this!



**Part of the problem in industry  
today is related to**

1. Specification
2. Visualization
3. Construction

# Unified Modeling Language

A way to address these problems!

# What is UML?

*Unified Modeling language (UML)*  
is a  
**standardized** modeling language  
enabling developers to  
**specify, visualize, construct and document**  
*artifacts of a software system.*

# Object Oriented Analysis and Design (OOAD)

## An Introduction

**Venkatesh Vinayakarao**

venkateshv@cmi.ac.in

<http://vvtesh.co.in>

---

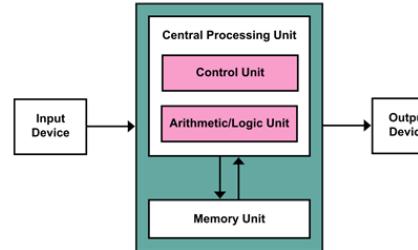
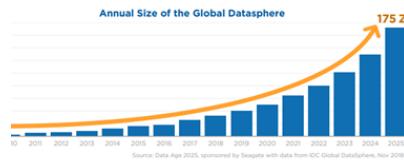
Chennai Mathematical Institute

---

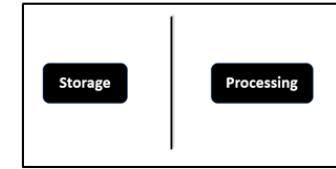
The art of simplicity is a puzzle of complexity. –**Douglas Horton.**

# Recap

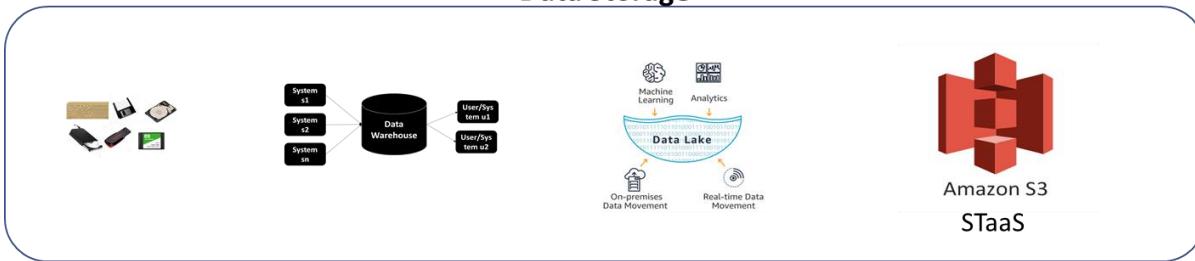
| Name      | Size            |
|-----------|-----------------|
| Byte      | 8 bits          |
| Kilobyte  | 1024 bytes      |
| Megabyte  | 1024 kilobytes  |
| Gigabyte  | 1024 megabytes  |
| Terabyte  | 1024 terabytes  |
| Petabyte  | 1024 petabytes  |
| Exabyte   | 1024 exabytes   |
| Zettabyte | 1024 zettabytes |
| Yottabyte | 1024 yottabytes |



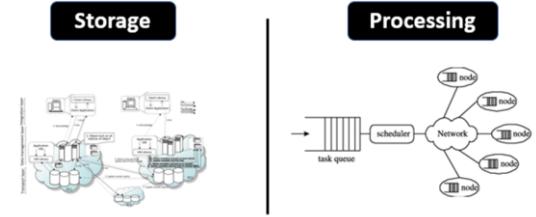
## Challenges



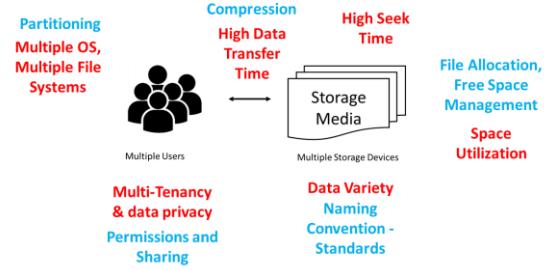
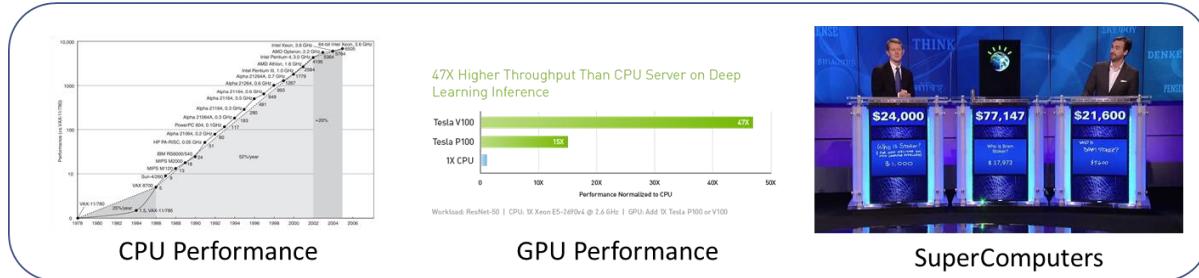
## Data Storage



Two kinds of Big Data Opportunities



## Data Processing



# Recap

## When not to use Hadoop?



No Interactive Jobs  
No Jobs Requiring Co-ordination  
No Small Files

## Hadoop Architecture

Application (map-reduce) Application (pig) Application (nosql db)

**YARN**  
(Resource Management – Job Scheduling/Monitoring)

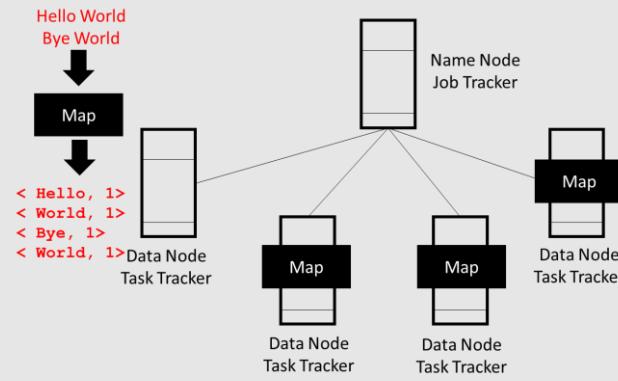
**HDFS**  
(Replicated Reliable Storage)

## Map-reduce Model

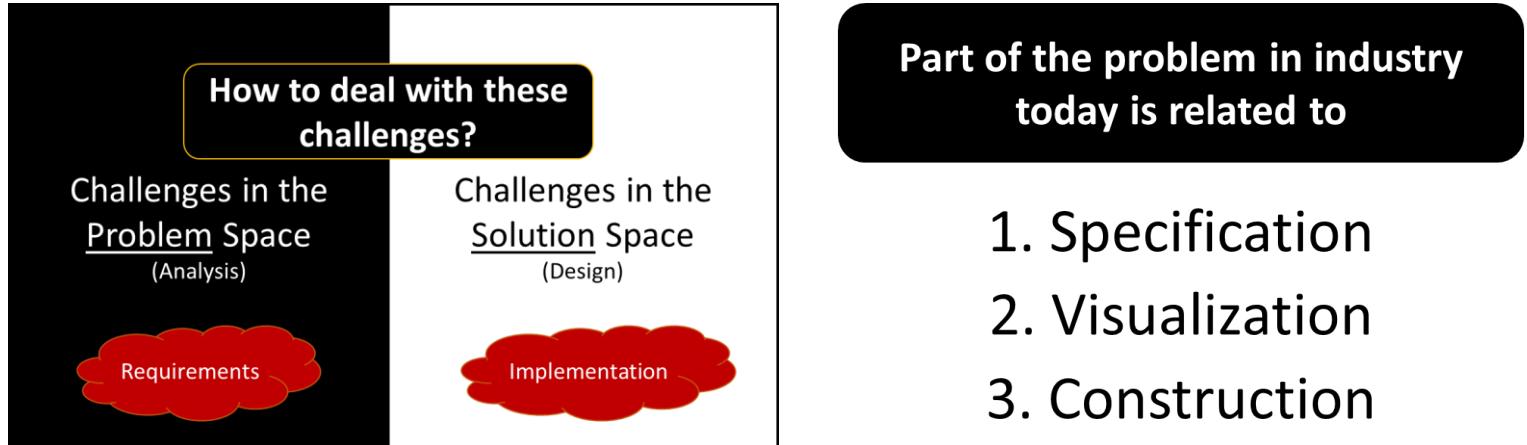
**Map**

**Shuffle and Sort**

**Reduce**



# Recap



*Unified Modeling language (UML) is a standardized modeling language enabling developers to specify, visualize, construct and document artifacts of a software system.*

# History

- UML combines best of three principal methods:
  - **The Booch method**, devised by Grady Booch,
  - **Object-oriented Modeling Technique (OMT)**, devised by Jim Rumbaugh,
  - **Object-oriented Software Engineering** (also known as Objectory), devised by Ivar Jacobson.

Hence called “**Unified**”

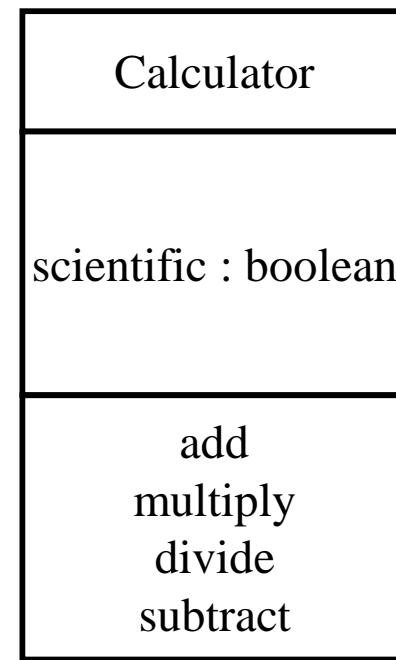
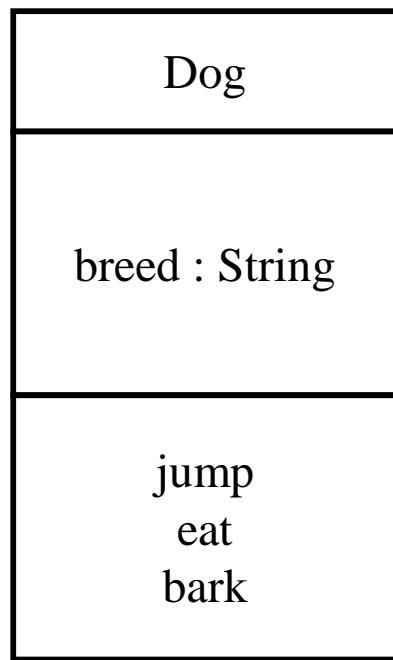
# History

- **1994**
  - Jim Rumbaugh joins Grady Booch at Rational Software to merge their methods.
- **1995**
  - Booch and Rumbaugh published version 8 of the Unified method. Rational Software buys Objectory and Ivar Jacobson joins the company.
- **1997**
  - Booch, Rumbaugh and Jacobson release (through Rational) a proposal of version 1 of UML.
- **1997**
  - UML version 1.1 was adopted by The Object Management Group (OMG), a non-profit organization.

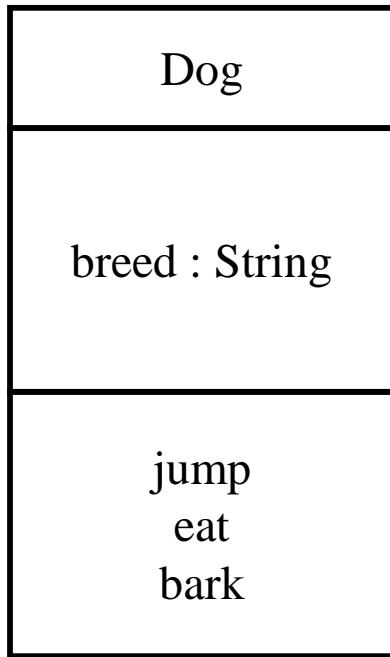
# Modeling Software Systems

- How is the software structured? (**Structural** Description)
  - **Class Diagram**
  - **Object Diagram**
  - Component Diagram
  - Deployment Diagram
  - Composite Structure Diagram
  - Package Diagram
- What does the software do? (**Behavioral** Description)
  - Use Case Diagram
  - Activity Diagram
  - Interaction Overview
  - How do multiple components interact? (Interaction Description)
    - *Sequence Diagram*
    - *Communication Diagram*
    - *Timing Diagram*
    - *Interaction Overview Diagram*

# Class Notation



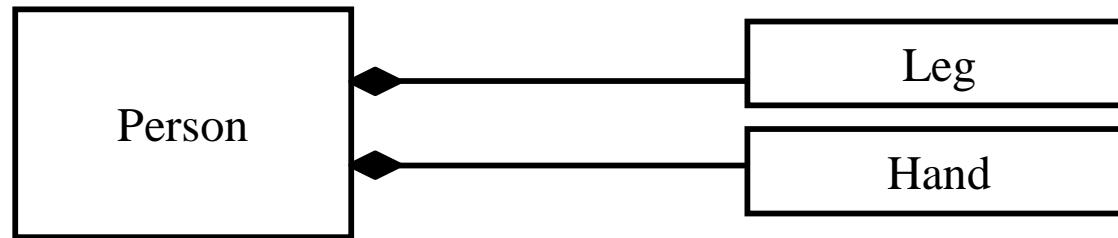
# Class → Code Transformation



```
class Dog\n{\n    String breed;\n\n    int bark()\n    {\n        ...\n    }\n}
```

# Relationships

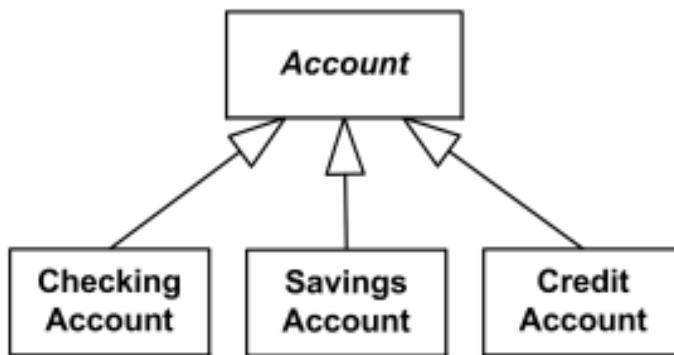
- Composition: Part-Whole Relationship where part cannot exist independently without the whole.



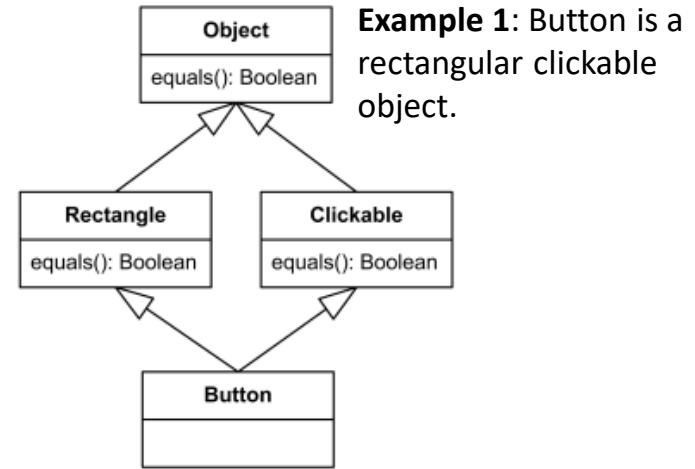
- Aggregation: Part-Whole Relationship where part may exist without the whole. Can you think of one?
  - Course – Student Relationship.

# Relationships - Generalization

- **Supertype – subtype** relationship.
- Also known as “**is a**” relationship.
- Any instance of the subtype is also an instance of the supertype.

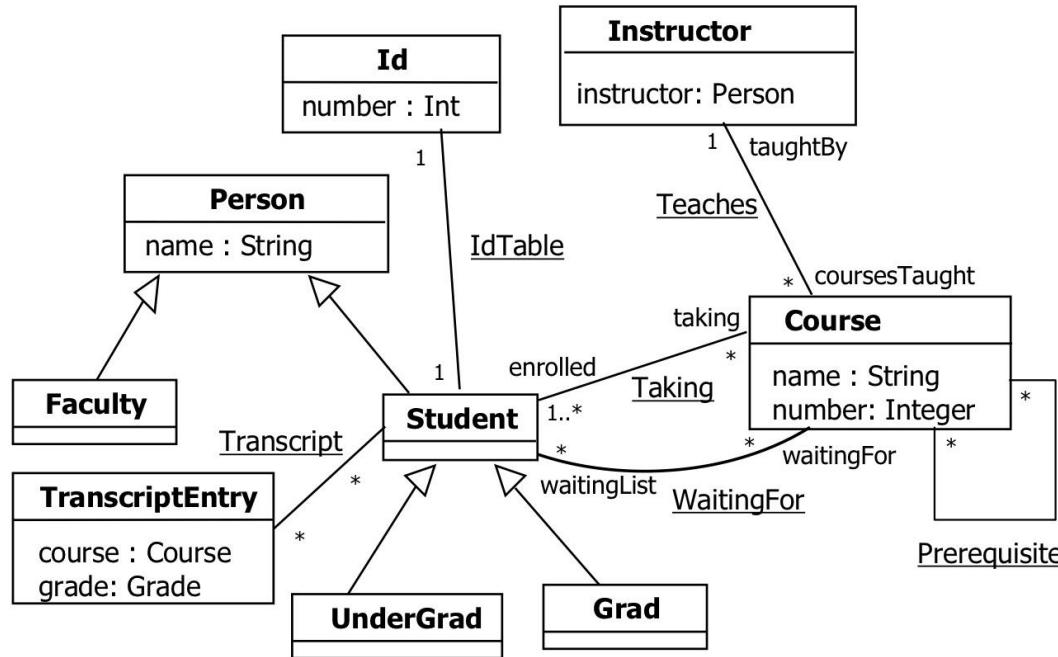


**Example 2:** There are three account types: Checking, Savings and Credit.



# Class Diagram

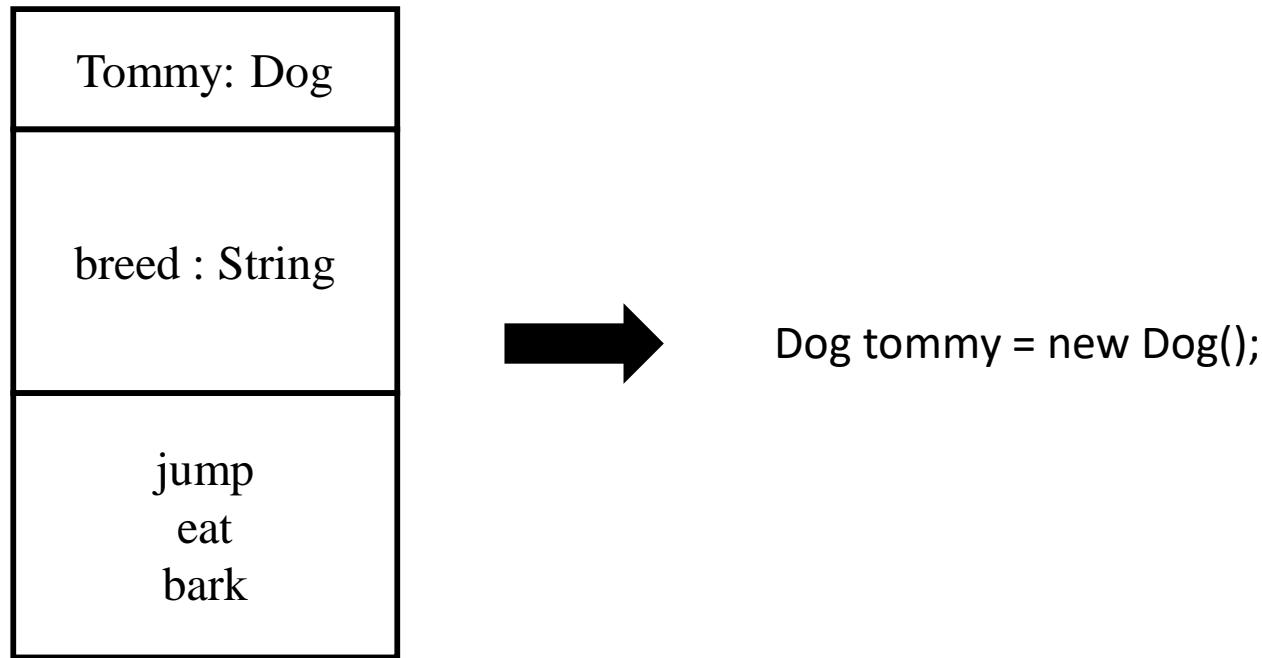
## University Scenario



Source: [uiowa.edu](http://uiowa.edu).

# Object Diagram

- At a specific time, shows the object instances and relationships.



# Quiz

- Draw an object diagram for the following scenario.

The course BDH is offered in 2020. Raj is a student enrolled in this course. The course instructor is Venkatesh.

# Identify the Classes

The **course** BDH is offered in 2020. Raj is a **student** enrolled in this course. The **course instructor** is Venkatesh.

# Identify the Objects

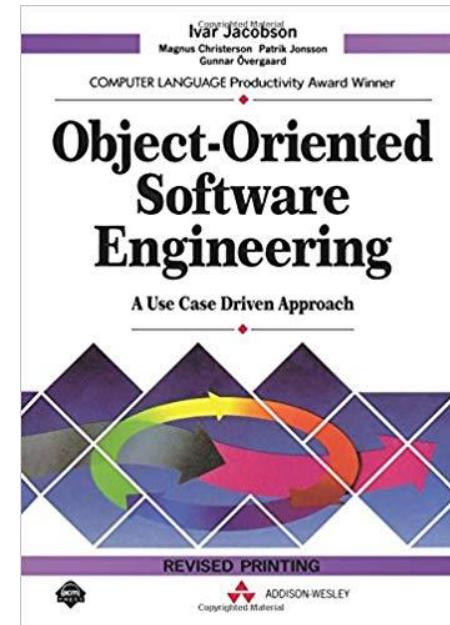
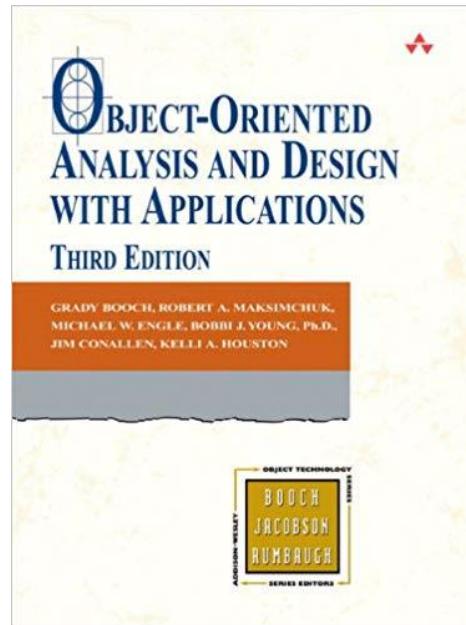
The course BDH is offered in 2020. Raj is a student enrolled in this course. The course instructor is Venkatesh.

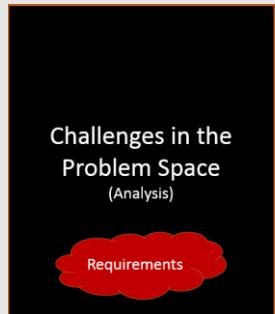
# Limitations

- Complex to hand-write UML diagrams. We **need tools.**
- Even with UML, **auto-generation of code** requires the model to be at very low level. This is considered impractical.
- There are **too many diagrams** and yet **descriptions** are not well captured.

# Resources

- Tools: ArgoUML (<http://argouml.tigris.org/>)
- Books:

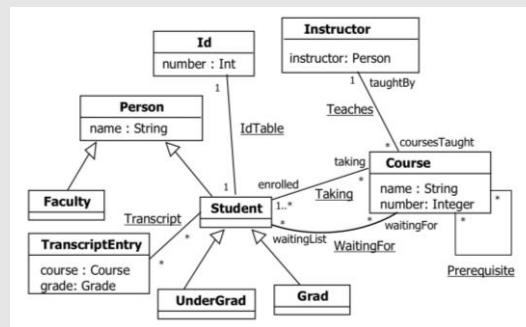




- ### Key Principles
1. Hierarchy
  2. Abstraction
  3. Keeping Related Things Together
  4. Polymorphism

#### Class Vs. Object

- Class has:
  - Data (Dogs are four legged [quadruped]).
  - Behavior (Dogs bark).
- Object has:
  - Identity (Jimmy).
  - Data (four legged).
  - Behavior (barks).



*Unified Modeling language (UML)*  
is a  
standardized modeling language  
enabling developers to  
specify, visualize, construct and document  
artifacts of a software system.

# THANK YOU

The art of simplicity is a puzzle of complexity. –Douglas Horton.



# Map-Reduce Design Patterns

**Venkatesh Vinayakarao**

venkateshv@cmi.ac.in

<http://vvtesh.co.in>

---

Chennai Mathematical Institute

---

**Finding patterns is the essence of wisdom. – Dennis Prager**

# I have a story for you!

Patterns here, Patterns there,  
Patterns, Patterns everywhere...

# Are beauty and quality objective?

- Can we agree some things are beautiful and some are not?



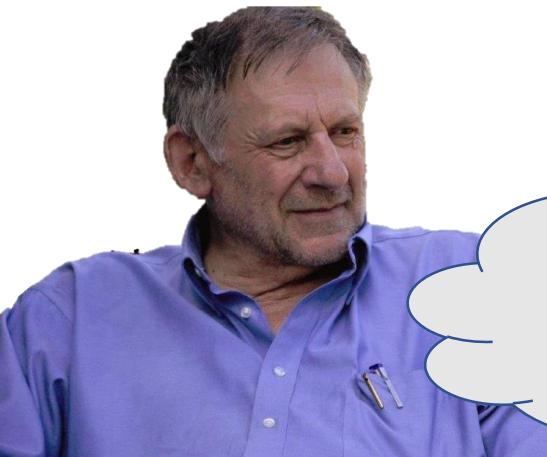
# Christopher Alexander Asked...

## What is a good design?



# Good Design

**Cultural Anthropology:** Within a culture, individuals agree what is good design, what is beautiful.



**Example:** Symmetry is good

**Beauty can be  
objectively  
measured.**

# Patterns

- Good design structures had similarities between them.
- Alexander called these similarities *patterns*.

*A pattern is a solution to a problem in a context.*

- "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over..."

Christopher Alexander, A Pattern Language:  
Towns/Buildings/Construction, 1977

# A question

- Can you tell me one design that is absolutely symmetrical?

**Equivalent ideas exist in software design**

# Good Design

- What according to you are the two biggest factors that determine a good/bad design?

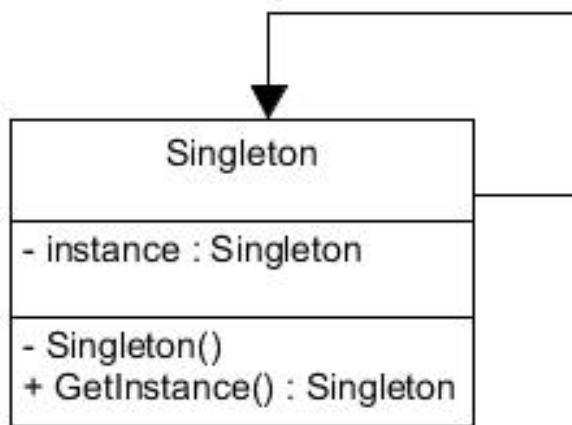
# Good and Bad Design

- What are the commonalities in what is viewed as good (and what is viewed as bad)?
  - A software system that is easy to maintain is considered good
    - A fragile software system is considered bad
  - A software system that is easy to understand is considered good
    - Obfuscated “spaghetti code” is considered bad

# Quiz

- Have you come across software patterns? Can you give one example?

# Singleton Pattern

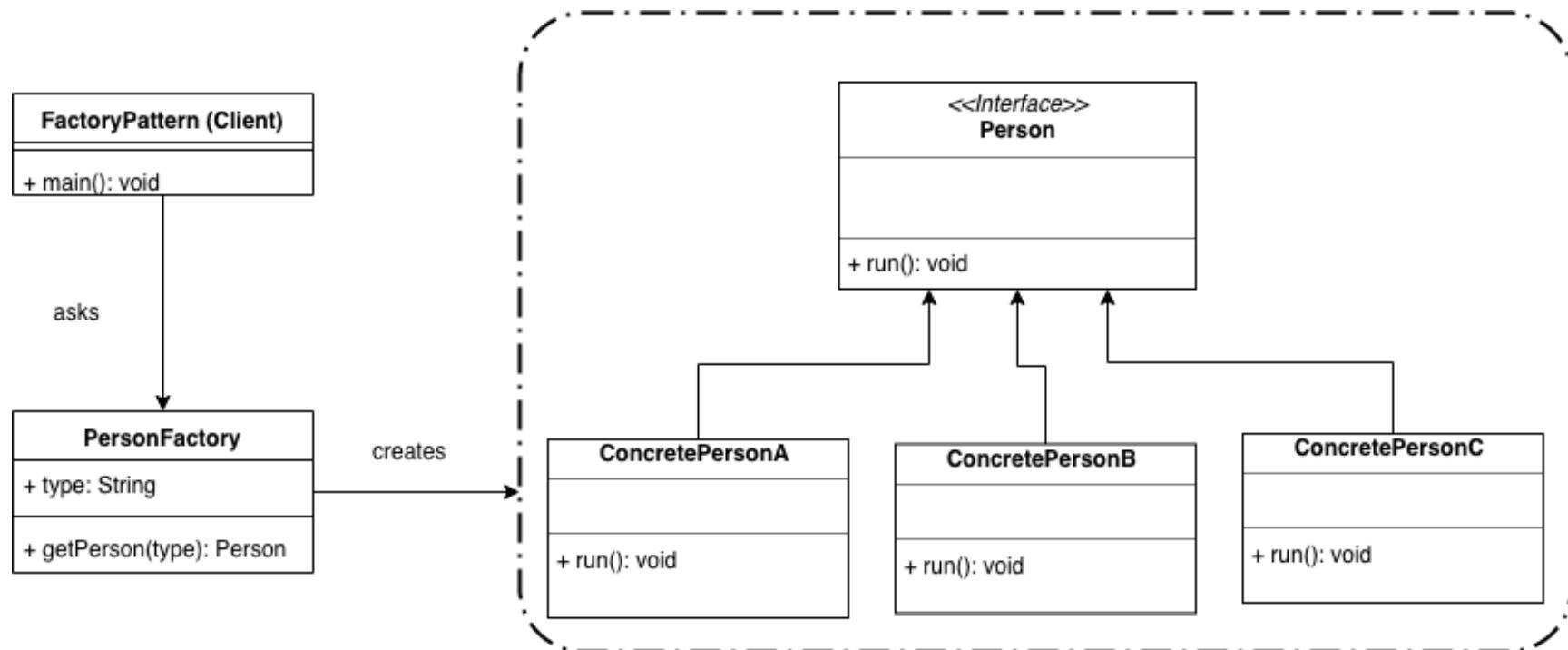


```
public class CMI
{
    private static final CMI instance = new CMI();

    private CMI()
    {
        // private constructor
    }

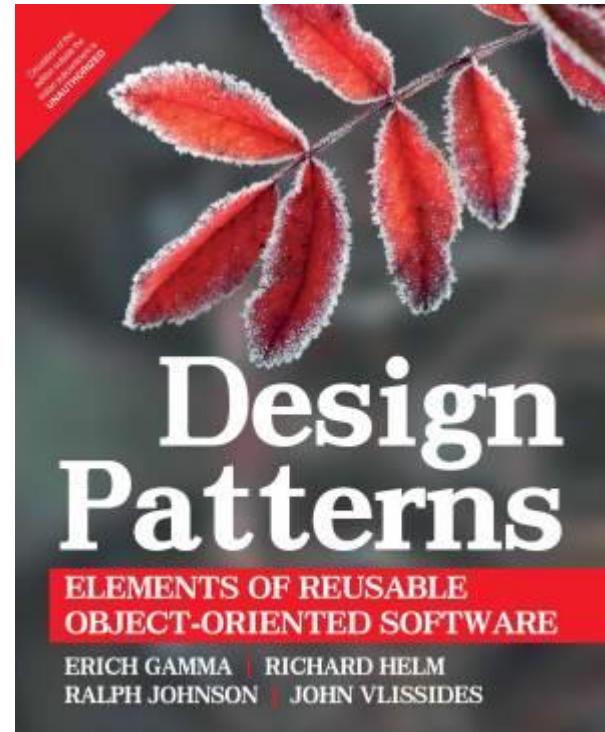
    public static CMI getInstance(){
        return instance;
    }
}
```

# Factory Pattern



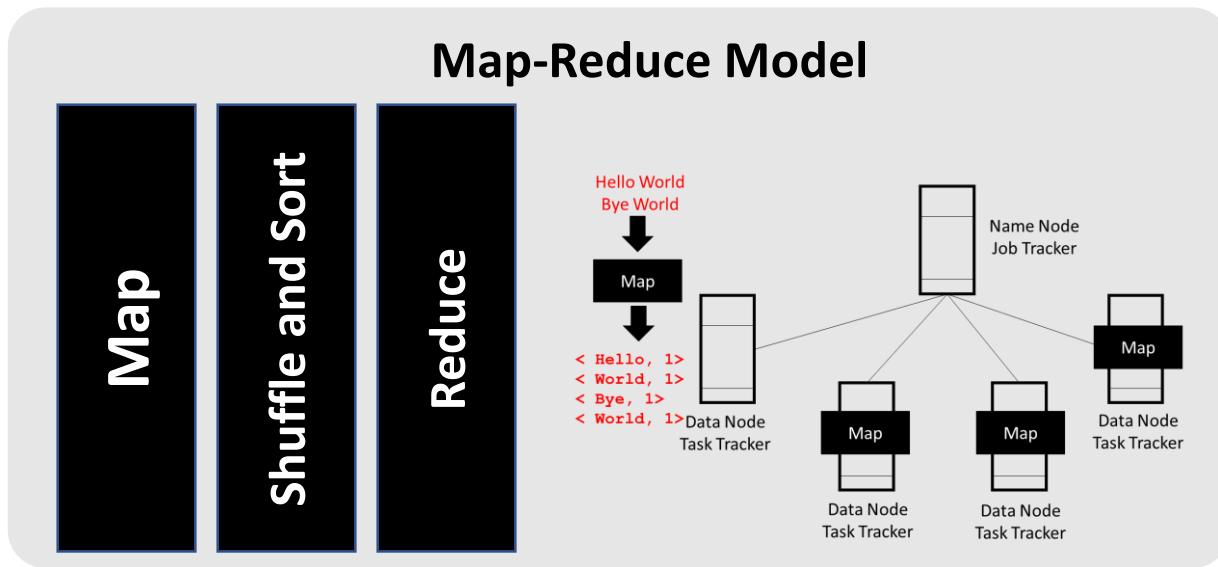
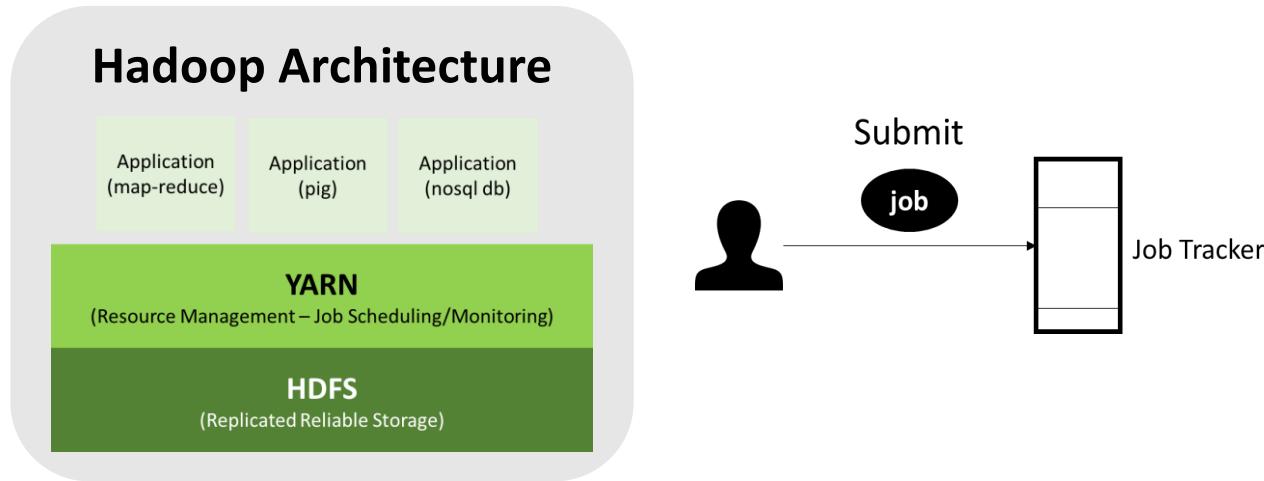
Source: <https://dzone.com/articles/creational-design-pattern-series-factory-method-pa>

# For More on Design Patterns

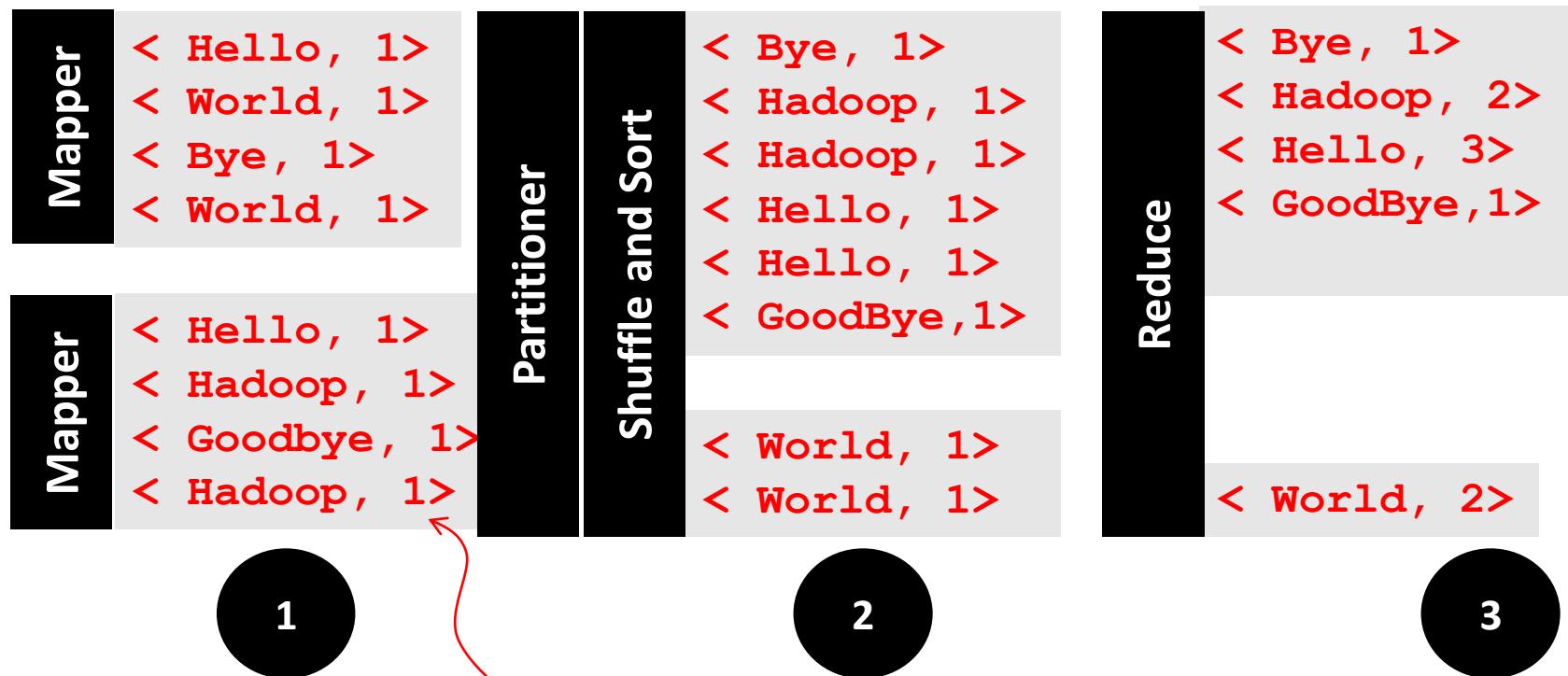


We shall now look at some Map-Reduce design patterns.

# Recap

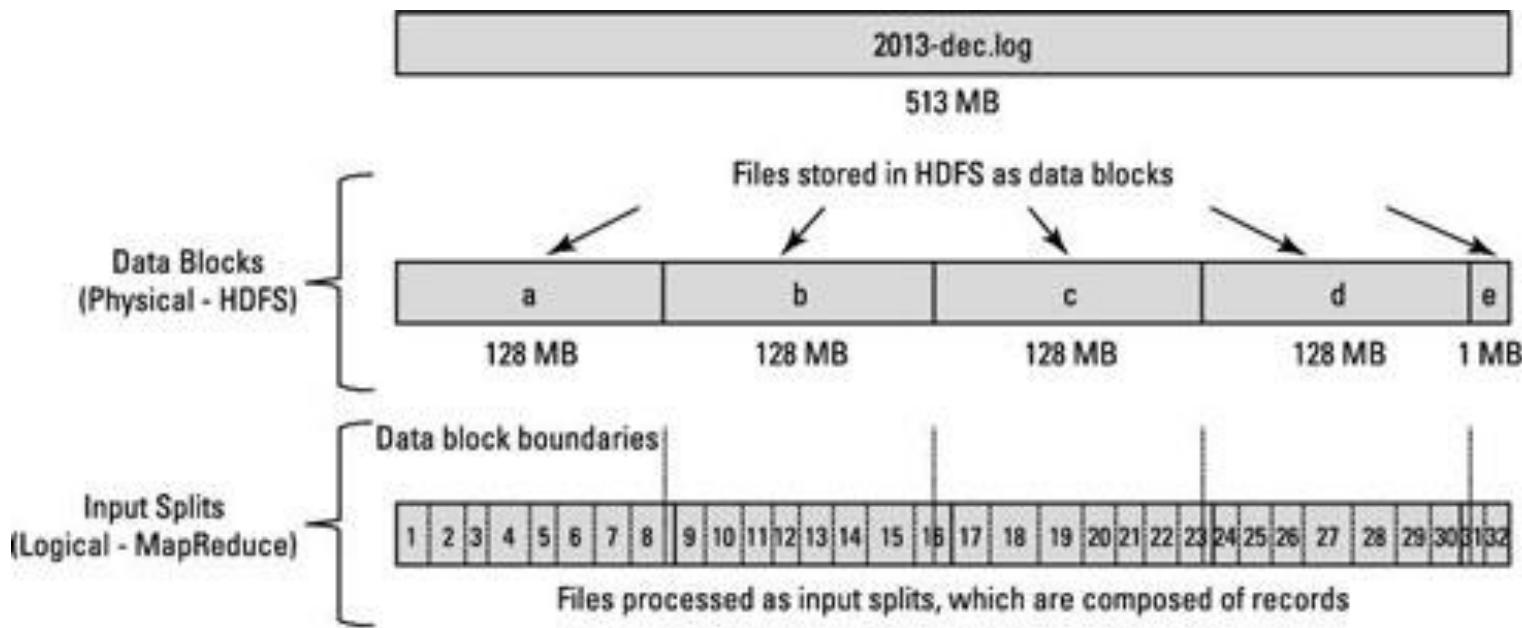


# Map-Reduce Processing



Combiner can be used to summarize locally per mapper.

# Input Splits



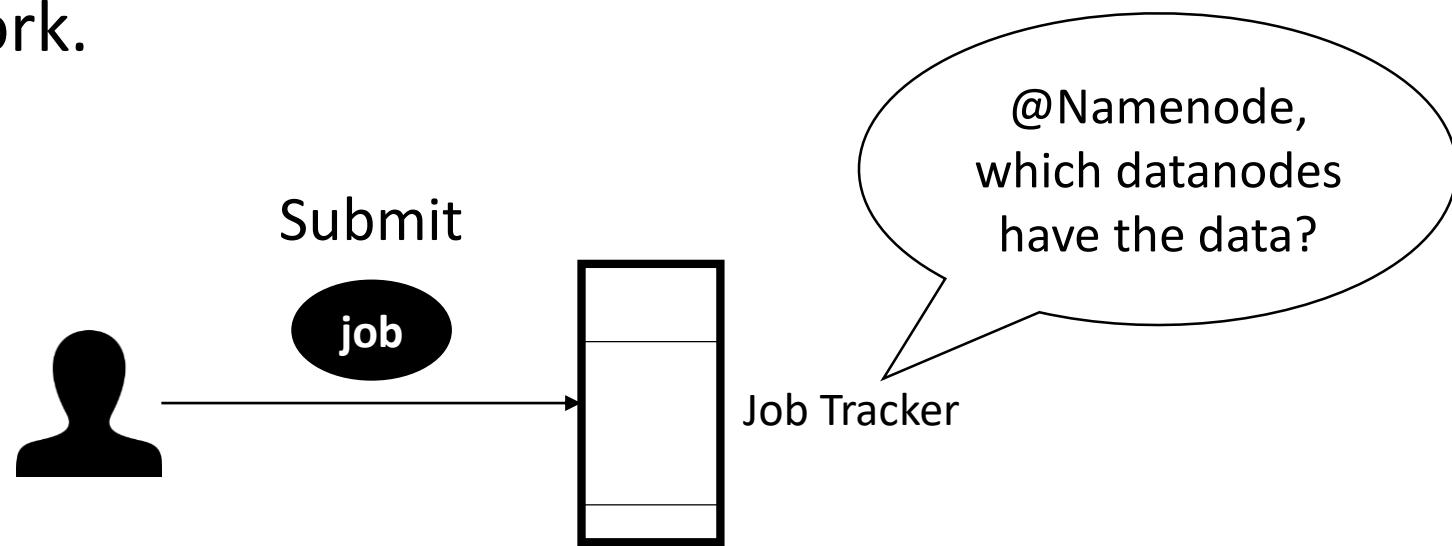
Note that a remote read may be required at block boundaries.

map: (K1, V1) → list(K2, V2)

reduce: (K2, list(V2)) → list(K3, V3)

# A Hadoop Map-Reduce Developer

- Writes the “map” code
- Writes the “reduce” code
- Submits the map and reduce code to Hadoop framework.



# Submitting a Map-Reduce Job

hadoop jar

`/usr/joe/wordcount.jar`

`org.myorg.WordCount`

`/usr/joe/wordcount/input`

`/usr/joe/wordcount/output`

# Mapper

```
public void map(Object key, Text value, Context context  
                ) throws IOException, InterruptedException {  
    StringTokenizer itr = new StringTokenizer(value.toString());  
    while (itr.hasMoreTokens()) {  
        word.set(itr.nextToken());  
        context.write(word, one);  
    }  
}
```

See <https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html> for details.

# Reducer

```
public void reduce(Text key, Iterable<IntWritable> values,
                  Context context
                  ) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
        sum += val.get();
    }
    result.set(sum);
    context.write(key, result);
}
```

# Create a Job

```
public static void main(String[] args) throws Exception {  
    Configuration conf = new Configuration();  
    Job job = Job.getInstance(conf, "word count");  
    job.setJarByClass(WordCount.class);  
    job.setMapperClass(TokenizerMapper.class);  
    job.setCombinerClass(IntSumReducer.class);  
    job.setReducerClass(IntSumReducer.class);  
    job.setOutputKeyClass(Text.class);  
    job.setOutputValueClass(IntWritable.class);  
    FileInputFormat.addInputPath(job, new Path(args[0]));  
    FileOutputFormat.setOutputPath(job, new Path(args[1]));  
    System.exit(job.waitForCompletion(true) ? 0 : 1);  
}
```

# Submit Job to Hadoop

```
$ bin/hadoop jar wc.jar WordCount /user/joe/wordcount/input  
/user/joe/wordcount/output
```

```
$ bin/hadoop fs -ls /user/joe/wordcount/input/  
/user/joe/wordcount/input/file01  
/user/joe/wordcount/input/file02

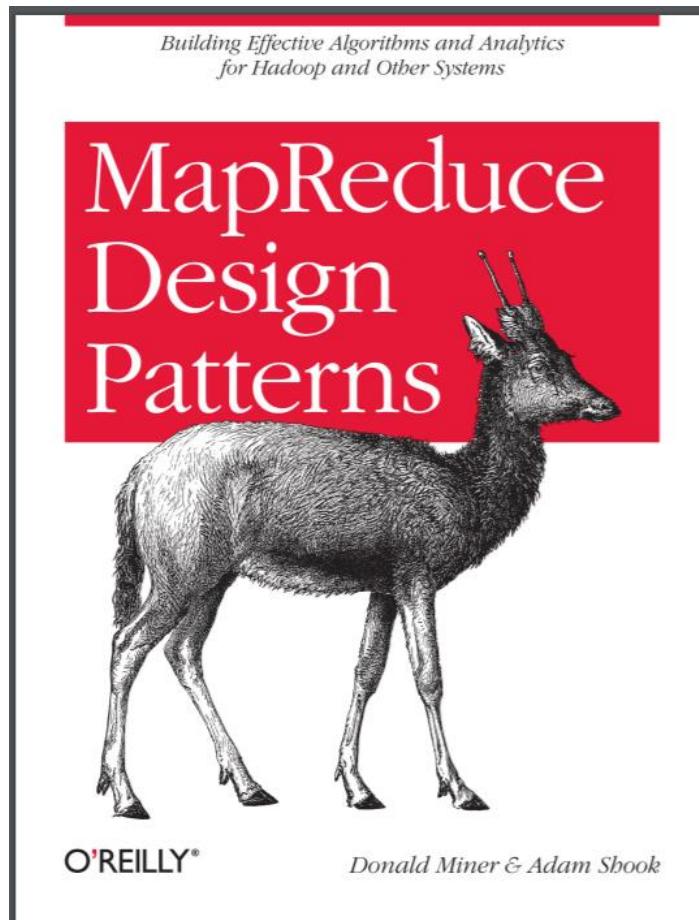
$ bin/hadoop fs -cat /user/joe/wordcount/input/file01  
Hello World Bye World

$ bin/hadoop fs -cat /user/joe/wordcount/input/file02  
Hello Hadoop Goodbye Hadoop
```

# Output

```
$ bin/hadoop fs -cat /user/joe/wordcount/output/part-r-00000
Bye 1
Goodbye 1
Hadoop 2
Hello 2
World 2
```

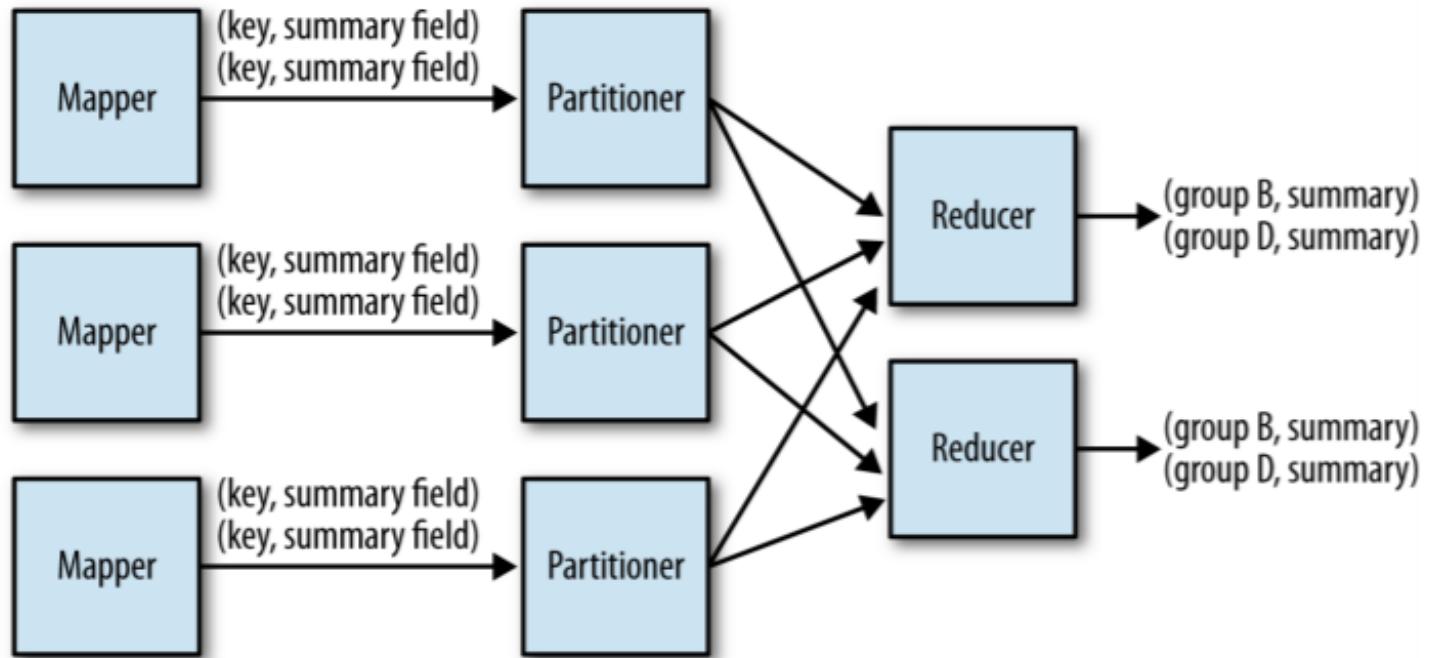
# Readings



# How Will You Implement These With Map Reduce?

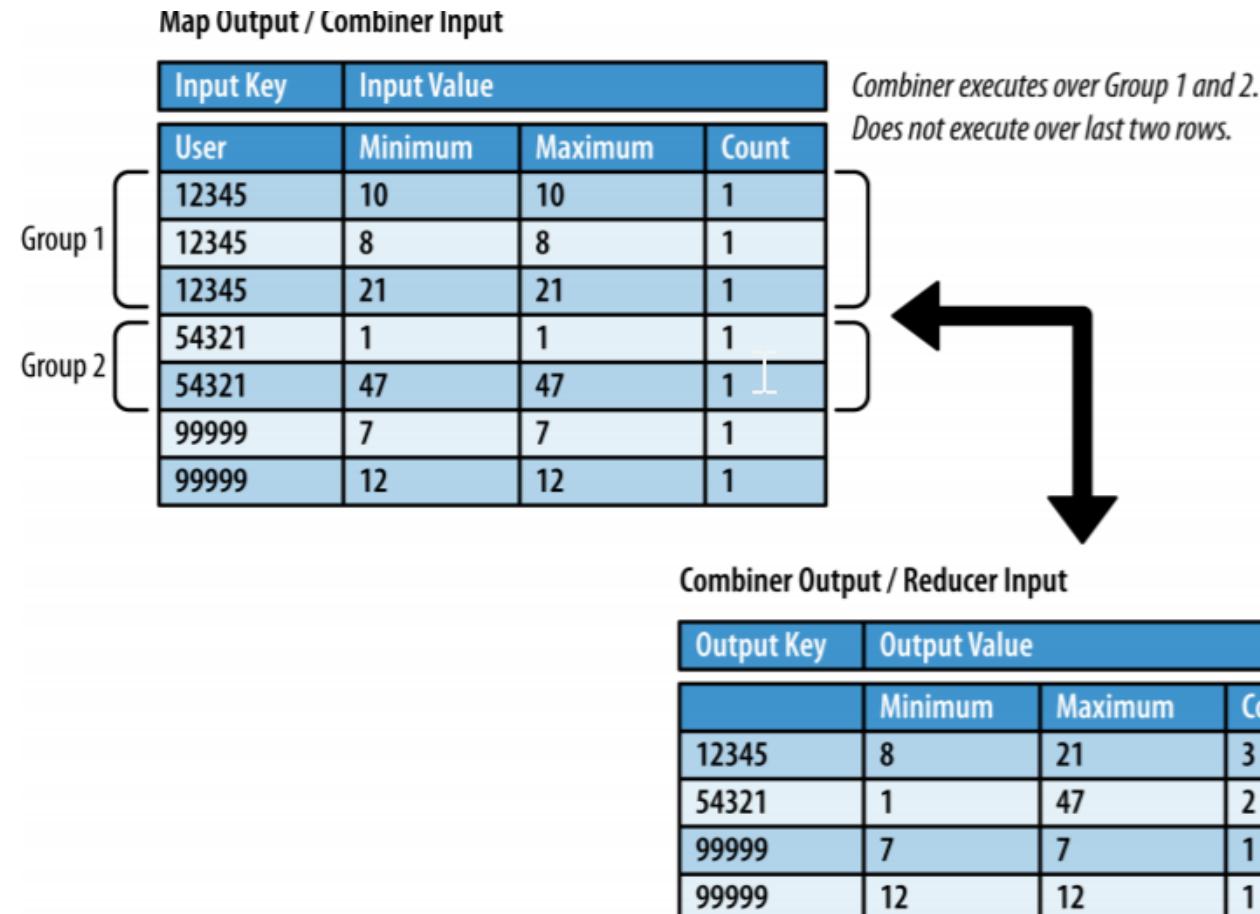
- Min/Max
- Average
- Count
- Median
- Filtering
- Top 10
- Convert key-values to hierarchy
- Partitioning
- Sorting

# Summarization Pattern

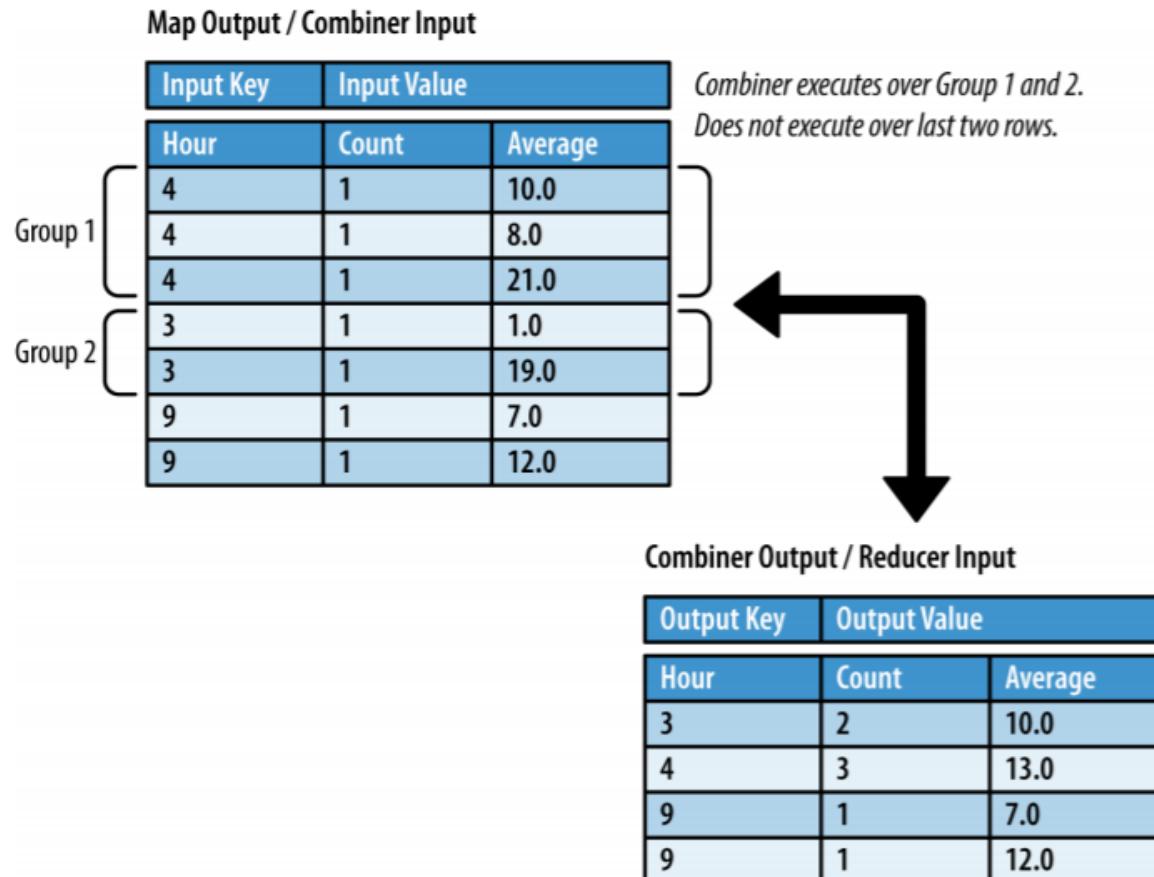


A **partitioner** controls the logical grouping keys of the intermediate map output.

# Min/Max/Count



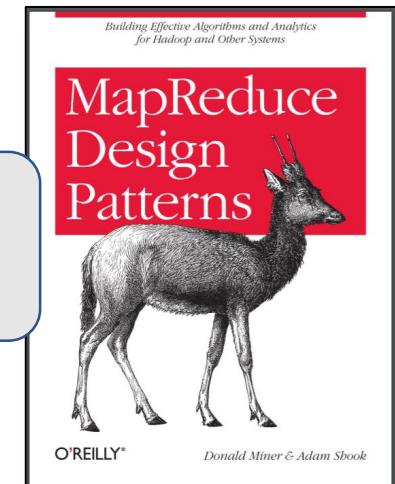
# Average



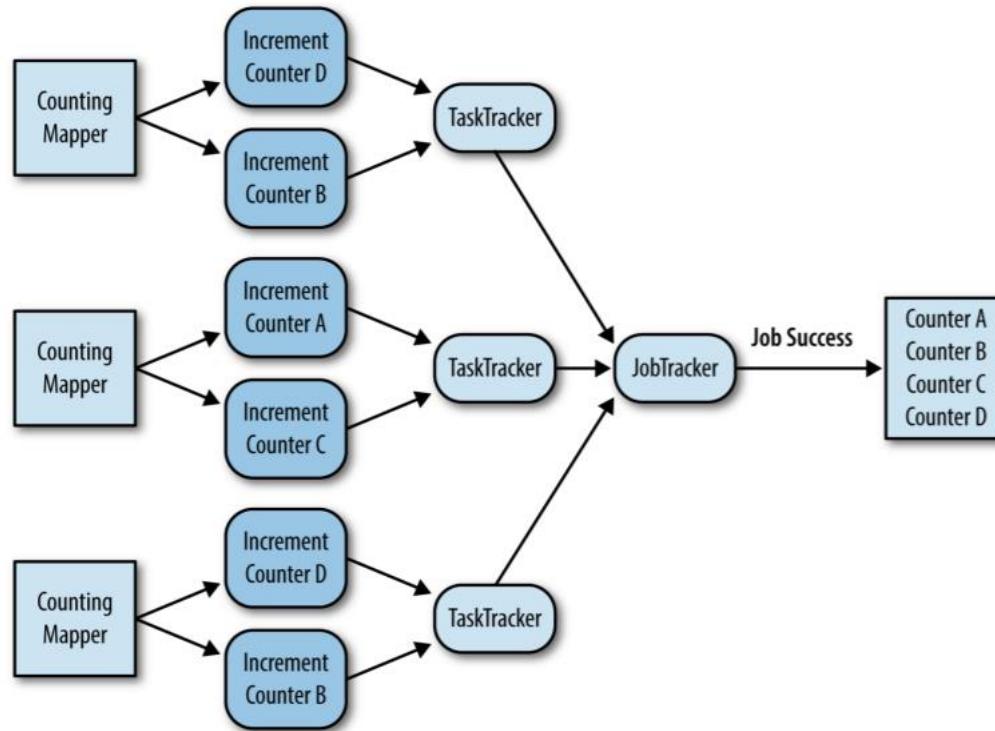
# Flex Your Brain!

- How will you compute the median?

Refer to Chapter 2 of MR Design Patterns Book.

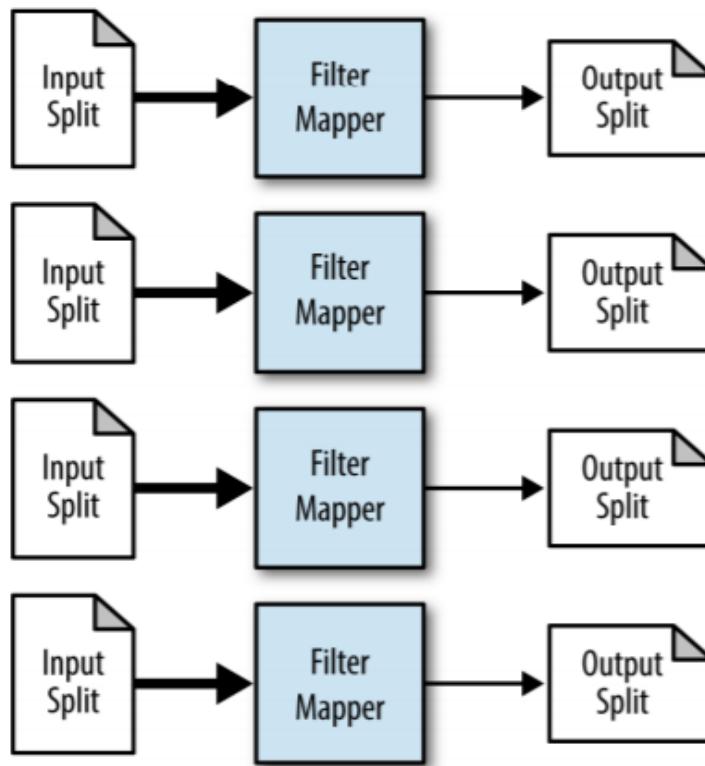


# Counting Mappers



Global counters belong to job-tracker. Use responsibly.

# Filtering



No  
Reducer  
Required.

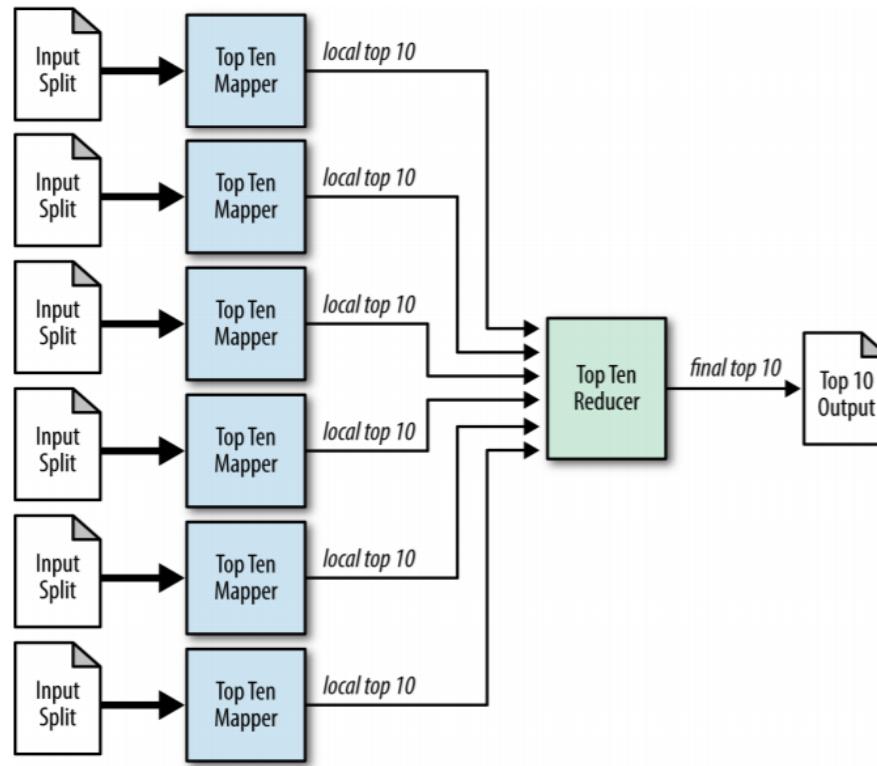
# Filter Example

```
public void map(Object key, Text value, Context context)
    throws IOException, InterruptedException {

    if (rands.nextDouble() < percentage) {
        context.write(NullWritable.get(), value);
    }
}
```

# Top 10 Pattern

- How will you determine the top 10 numbers in petabytes of numbers?



# Structure to Hierarchy

How to store this  
in RDBMS?

```
Posts
  Post
    Comment
    Comment
  Post
    Comment
    Comment
    Comment
```



## Apache Pig

<https://pig.apache.org/>

**Venkatesh Vinayakarao**

venkateshv@cmi.ac.in

<http://vvtesh.co.in>

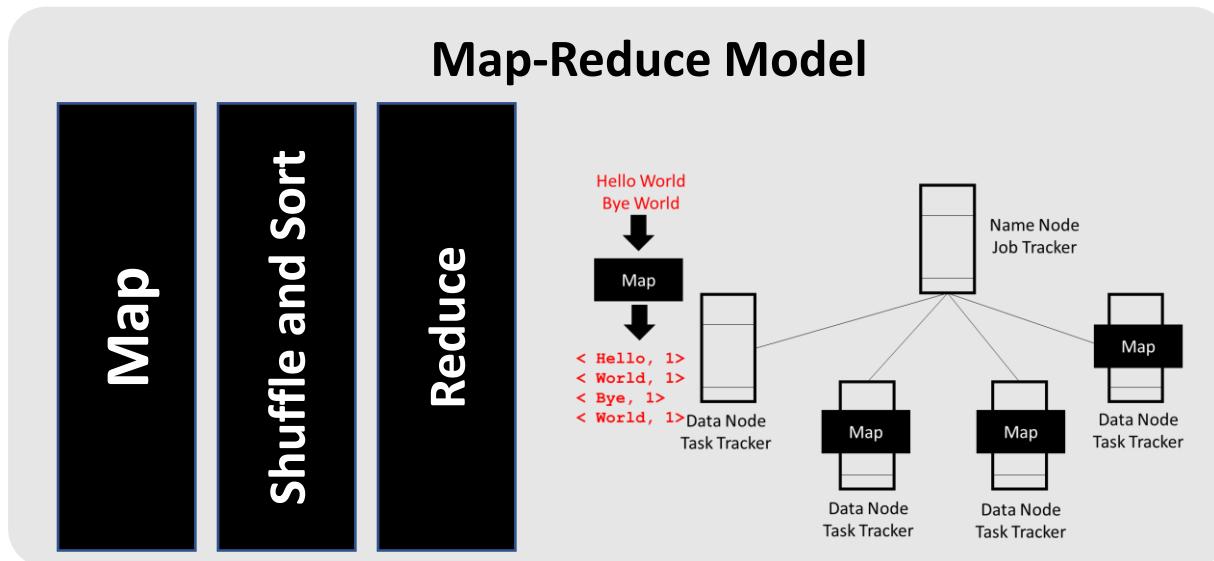
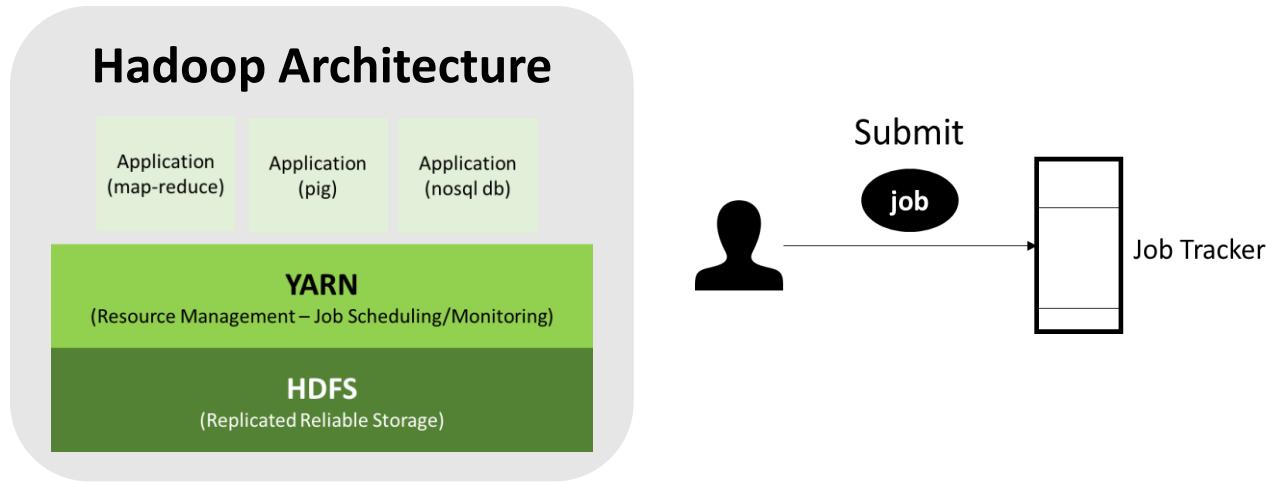
---

Chennai Mathematical Institute

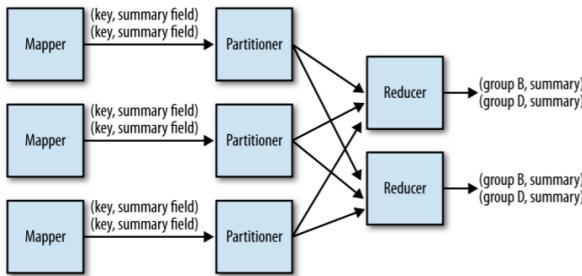
---

Making Pig Fly – Thejas Nair.

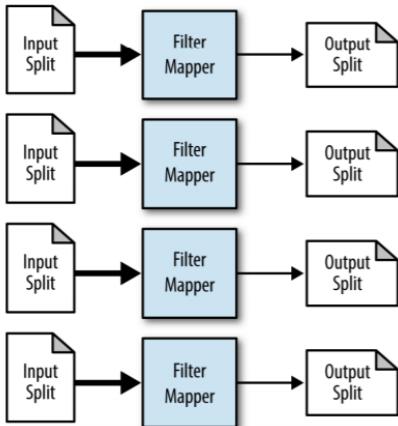
# Recap



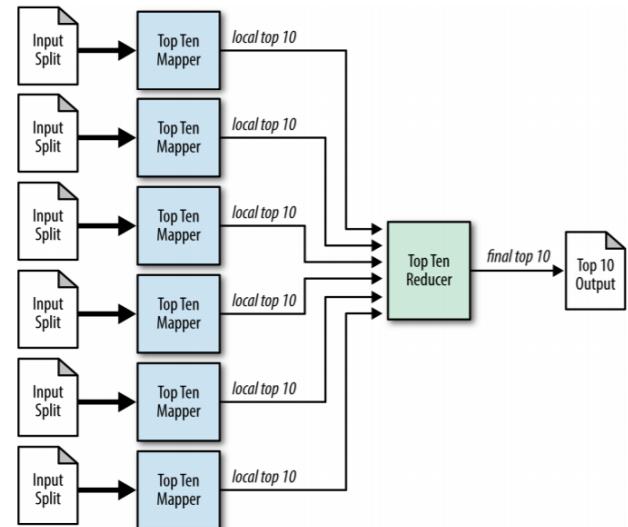
# Map-Reduce Patterns



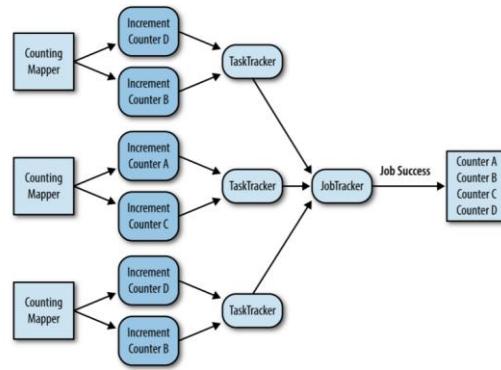
**Summarization**



**Filtering**



**Top 10**



**Counting**

# Code

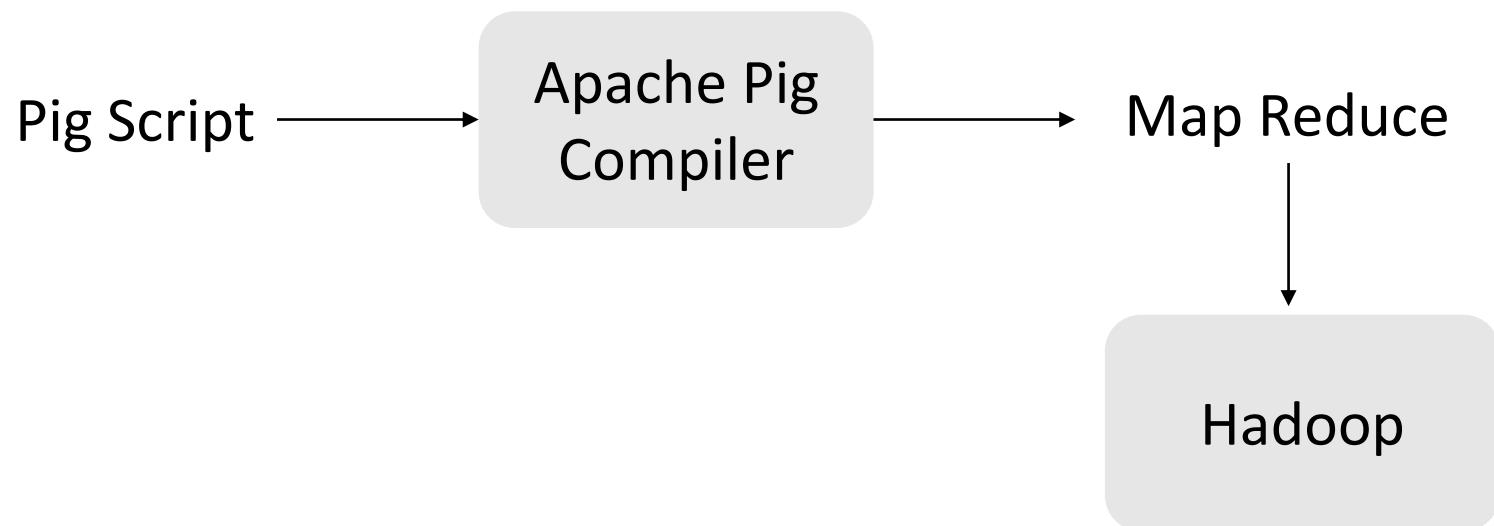
```
public void map(Object key, Text value, Context context
                ) throws IOException, InterruptedException {
    StringTokenizer itr = new StringTokenizer(value.toString());
    while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
    }
}

public void reduce(Text key, Iterable<IntWritable> values,
                  Context context
                  ) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
        sum += val.get();
    }
    result.set(sum);
    context.write(key, result);
}
```

# But...

What if...  
We are not good at coding?

# Scripting instead of Coding



# A Sample Pig Script

## LOAD Command Syntax

```
LOAD 'data' [USING function] [AS schema];
```

```
A = LOAD 'student' USING PigStorage()  
      AS (name:chararray, age:int, gpa:float);
```

```
B = FOREACH A GENERATE name;
```

```
DUMP B;
```

[Read: https://pig.apache.org/docs/r0.16.0/basic.html#load](https://pig.apache.org/docs/r0.16.0/basic.html#load)

# Benefits & Limitations

- Benefits
  - 10 lines of Pig Latin (approx.) = 200 lines in Java
  - 15 minutes in Pig Latin (approx.) = 3 hours in Java
    - Simple
    - Easy
    - Quick to Code
  - Provides in-built functions to load, process and print data.
  - Similar to SQL
    - Can perform join and order by
- Limitations
  - Slower than Map-Reduce

# Pig in Real-World

- Yahoo uses it extensively (>70% of jobs)
- Facebook – Process Logs
- Twitter – Process Logs
- eBay – Data processing for intelligence
- ...

# Grunt Shell

```
$ pig -x local  
... - Connecting to ...  
grunt>
```

Or

```
pig -x local id.pig
```

# Tutorial

- Download
  - wget <http://www-us.apache.org/dist/pig/pig-0.17.0/pig-0.17.0.tar.gz>
  - tar -xzf pig-0.17.0.tar.gz
  - cd bin
- Check
  - ./pig –version
- Execute
  - ./pig -x local
  - data = LOAD 'file1' using PigStorage(',') AS (name:chararray,age:int);
  - data1 = filter data by \$1 > 2;
  - dump data1;
  - quit
- Don't forget the semicolon

# Pig Philosophy

- Pigs eat anything
  - Input can be of a variety of formats
- Pigs live anywhere
  - Not only for hadoop
- Pigs are domestic animals
  - Easy to master
- Pigs fly
  - Ultimately map-reduce code. Improving performance is a priority to the pig team.

# Welcome to the World of Pig

- Pig Latin
  - For the language
- Grunt
  - For the shell
- Piggy-bank
  - For the shared reusable modules

# More Examples

```
A = LOAD 'data' AS (f1,f2,f3);  
B = FOREACH A GENERATE f1 + 5;  
C = FOREACH A generate f1 + f2;
```

# Referencing Fields

```
A = LOAD 'student' USING PigStorage() AS  
    (name:chararray, age:int, gpa:float);
```

```
X = FOREACH A GENERATE name,$2;
```

```
DUMP X;
```

(John,4.0F)

(Mary,3.8F)

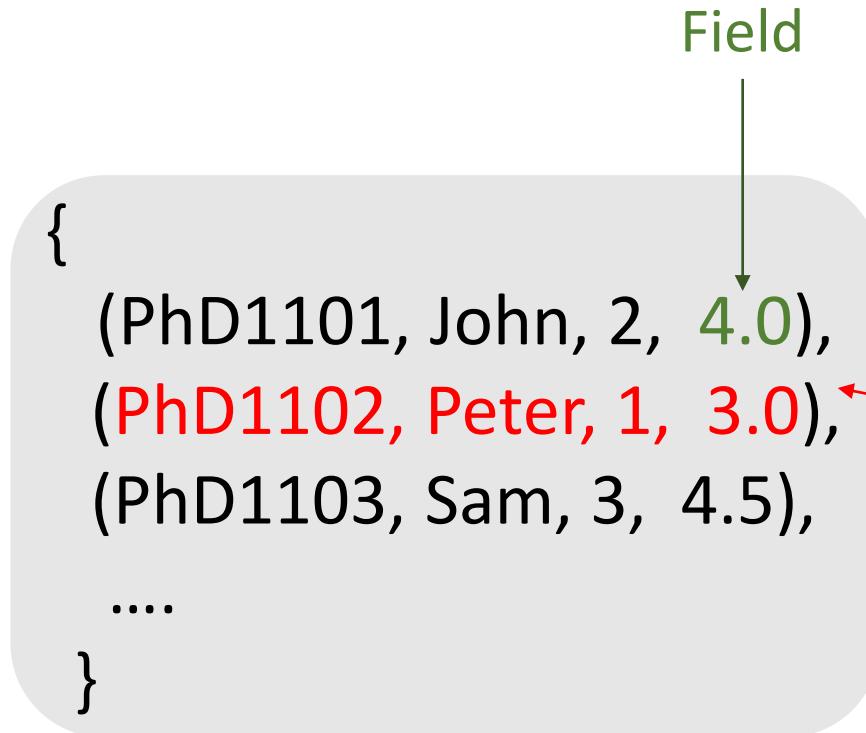
(Bill,3.9F)

(Joe,3.8F)

# Data Types

- Scalar Types:
  - Int, long, float, double, boolean, null, chararray, bytearray;
- Complex Types:
  - Field, Tuple and Relation/Bag
  - Map [key#value]

# Data Types in Pig Latin



Tuple  
An ordered  
set of fields.

Relation/Bag  
An ordered set of tuples.

# Load and Dump

```
A = LOAD 'data' AS (f1:int,f2:int,f3:int);  
DUMP A;
```

(1,2,3)

(4,2,1)

(8,3,4)

(4,3,3)

(7,2,5)

(8,4,3)

## Input

(3,8,9) (4,5,6)

(1,4,7) (3,7,5)

(2,5,8) (9,5,8)

```
A = LOAD 'data' AS (
    t1:tuple(t1a:int, t1b:int,t1c:int),
    t2:tuple(t2a:int,t2b:int,t2c:int)
);
```

DUMP A;

## Output

((3,8,9),(4,5,6))

((1,4,7),(3,7,5))

((2,5,8),(9,5,8))

**Guess the output**

```
X = FOREACH A GENERATE
    t1.t1a,t2.$0;
DUMP X;
```

# The Answer

```
X = FOREACH A GENERATE t1.t1a,t2.$0;  
DUMP X;
```

(3,4)

(1,3)

(2,9)

# Tuples

```
A = LOAD 'data' as (f1:int,  
                  f2:tuple(t1:int,t2:int,t3:int));  
  
DUMP A;
```

(1,(1,2,3))  
(2,(4,5,6))  
(3,(7,8,9))  
(4,(1,4,7))  
(5,(2,5,8))

# Map

Data

```
328;ADMIN HEARNG;[street#939 W El Camino,city#Chicago,state#IL]
43;ANIMAL CONTRL;[street#415 N Mary Ave,city#Chicago,state#IL]
```

Usage

```
grunt> departments = LOAD 'somefile'
      AS (dept_id:int, dept_name:chararray, address:map[]);
```

```
grunt> dept_addr = FOREACH departments
      GENERATE dept_name,
              address#'street' as street,
              address#'city' as city,
              address#'state' as state;
```

<https://www.hadoopinrealworld.com/beginners-apache-pig-tutorial-map/>

# Operations

- Loading data
  - **LOAD** loads input data
  - Lines=**LOAD** 'input/access.log' AS (line: chararray);
- Projection
  - **FOREACH ... GENERTE ...** (similar to SELECT)
  - takes a set of expressions and applies them to every record.
- Grouping
  - **GROUP** collects together records with the same key
- Dump/Store
  - **DUMP** displays results to screen, **STORE** save results to file system
- Aggregation
  - **AVG, COUNT, MAX, MIN, SUM**

# Example

- students = **LOAD** 'student.txt' **USING**  
**PigStorage('\t')** **AS** (studentid: int, name:chararray,  
age:int, gpa:double);
- studentid = **FOREACH** students **GENERATE**  
studentid, name;

# Filter

**Data:**

year,product,quantity

---

2000, iphone, 1000

2001, iphone, 1500

2002, iphone, 2000

```
grunt> A = LOAD '/user/hadoop/sales' USING PigStorage(',')  
AS (year:int,product:chararray,quantity:int);  
grunt> B = FILTER A BY quantity >= 1500;  
grunt> DUMP B;
```

# How to run Pig Scripts?

- Local mode
  - Local host and local file system is used
  - Neither Hadoop nor HDFS is required
  - Useful for prototyping and debugging
- MapReduce mode
  - Run on a Hadoop cluster and HDFS
- Batch mode - run a script directly
  - Pig –x local my\_pig\_script.pig
  - Pig –x mapreduce my\_pig\_script.pig
- Interactive mode use the Pig shell to run script
  - Grunt> Lines = LOAD '/input/input.txt' AS (line:chararray);
  - Grunt> Unique = DISTINCT Lines;
  - Grunt> DUMP Unique;

# Flatten

Let the Input -> (a,(b,c)) be in A.

B = foreach A generate \$0 , flatten (\$1)

Output -> (a,b,c)

# Tokenize

- Input
  - 001,Raj Reddy,21,Hyderabad
  - 002,Raj Chatterjee,22,Kolkata
  - 003,Raj Khanna,22,Delhi

```
grunt> student_details = LOAD  
'hdfs://localhost:9000/pig_data/student_details.txt' USING  
PigStorage(',') as (id:int, name:chararray, age:int, city:chararray);  
  
grunt> student_name_tokenize = foreach student_details Generate  
TOKENIZE(name);  
  
grunt> Dump student_name_tokenize;
```

# Output

({{Raj),(Reddy}})

({{Raj),(Chatterjee}})

({{Raj),(Khanna}})

Splits a string. Creates tuples of names. Outputs the bag.

# Store

```
STORE student INTO '  
hdfs://localhost:9000/pig_Output/' USING  
PigStorage (',');
```

You can write your own functions! In this class, we will use the built-in PigStorage.

# Word Count

Lines=LOAD 'input/hadoop.log' AS (line: chararray);

Words = FOREACH Lines GENERATE  
    FLATTEN(TOKENIZE(line)) AS word;

Groups = GROUP Words BY word;

Counts = FOREACH Groups GENERATE group,  
    COUNT(Words);

Results = ORDER Words BY Counts DESC;

Top5 = LIMIT Results 5;

STORE Top5 INTO /output/top5words;

# User Defined Functions

- What is UDF
  - Way to do an operation on a field or fields
  - Called from within a pig script
  - Currently all done in Java
- Why use UDF
  - You need to do more than grouping or filtering
  - Maybe more comfortable in Java land than in SQL/Pig Latin

# UDF in Pig

-- myscript.pig

```
REGISTER myudfs.jar;
A = LOAD 'student_data' AS (name: chararray, age: int, gpa: float);
B = FOREACH A GENERATE myudfs.UPPER(name);
DUMP B;
```

UDFs can be written using a variety of languages including Python. See  
<https://pig.apache.org/docs/r0.17.0/udf.html>

# Simple UDF

```
public class UPPER extends EvalFunc<String> {  
    public String exec(Tuple input) throws IOException {  
        if (input == null || input.size() == 0)  
            return null;  
        try{  
            String str = (String)input.get(0);  
            return str.toUpperCase();  
        } catch(Exception e) {  
            throw new IOException("Caught exception", e);  
        }  
    }  
}
```

Source: <https://pig.apache.org/docs/r0.10.0/udf.html>

# Creating the Jar

```
jar -cf exampleudf.jar exampleudf
```

Know where have you placed this jar.

In Pig Script:

- REGISTER ‘...path to jar’;
- DEFINE SIMPLEUPPER exampleudf.UPPER();
- ... now you can use this method.

<https://pig.apache.org/docs/latest/basic.html#define-udfs>

# Thank You!

Appendix: Presentations

I can't pass an extremely competitive test to become a surgeon. But you give me any operation on a heart. I can perhaps do much better than most people. I am like an artist. Don't expect me to compete in an exam. Give me the job and I will show you how good I am.

# NoSQL DB

**Venkatesh Vinayakarao**

venkateshv@cmi.ac.in  
<http://vvtesh.co.in>

---

Chennai Mathematical Institute

---

The cost of managing traditional databases is high. Mistakes made during routine maintenance are responsible for 80 percent of application downtime. – **Dev Ittycheria, MongoDB.**

# A Relation as a Data Model

- Let the set,  $\text{id} = \{1,2,3\}$
- Let the set,  $\text{names} = \{\text{vv}, \text{sd}\}$
- What is  $\text{id} \times \text{names}$ ? \_\_\_\_\_
- We have a **relation** if we assign a sequential id to each name.

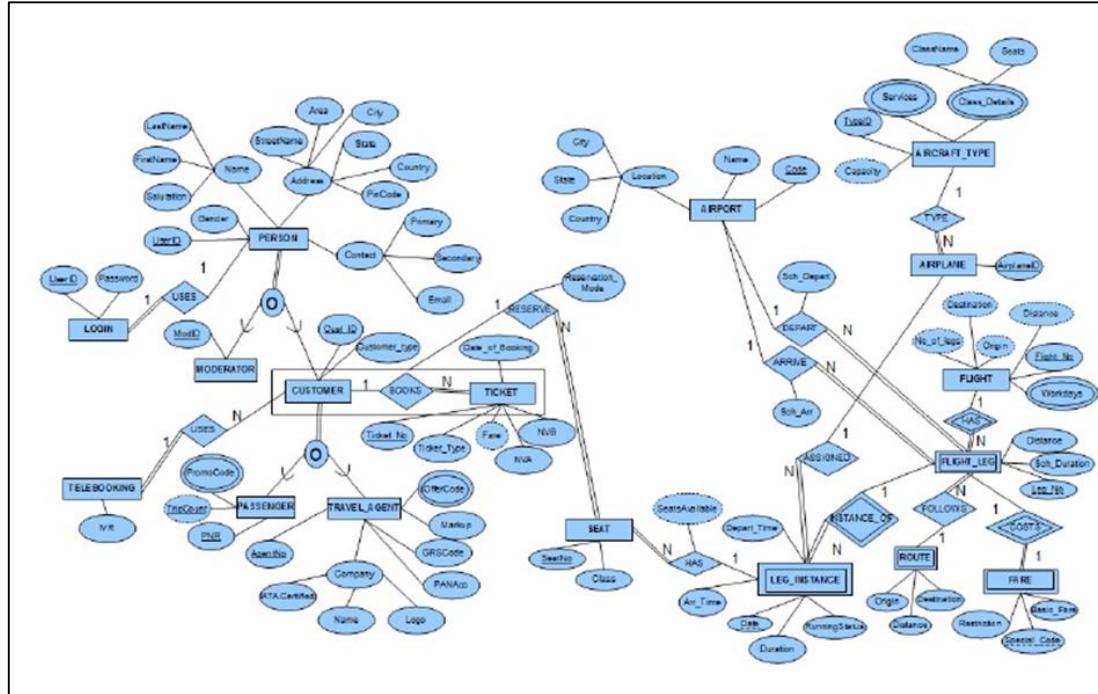
| <b>id</b> | <b>name</b> |
|-----------|-------------|
| 1         | sd          |
| 2         | vv          |



| <b>id</b> | <b>name</b> |
|-----------|-------------|
| 1         | sd          |
| 1         | vv          |
| 2         | sd          |
| 2         | vv          |
| 3         | sd          |
| 3         | vv          |

... and thus we had the relational database.

# An Entity-Relationship Design



## DB Designs:

- Can get too **complex!**
- May become too **hard to maintain!!**

# Key Challenges of Relational DB

- Schema needs to be defined.
- Maintenance becomes harder over time.
- Impedance mismatch problem. 
- Does not scale out by design.
- ACID Transactions – Consistency Vs. Availability Trade-off.

# Impedance Mismatch

# DB Design

| APPLICATION FOR EMPLOYMENT                                                                                      |                                  |                                          |                      |                       |
|-----------------------------------------------------------------------------------------------------------------|----------------------------------|------------------------------------------|----------------------|-----------------------|
| <b>PERSONAL INFORMATION</b>                                                                                     |                                  | <b>DATE</b>                              |                      |                       |
| NAME (LAST NAME FIRST)                                                                                          |                                  | PHONE NO.                                |                      |                       |
| PRESENT ADDRESS                                                                                                 |                                  |                                          |                      |                       |
| PERMANENT ADDRESS                                                                                               |                                  |                                          |                      |                       |
| SOCIAL SECURITY NO.                                                                                             |                                  | REFERRED BY                              |                      |                       |
| <b>DESIRED POSITION</b>                                                                                         |                                  |                                          |                      |                       |
| TITLE OF POSITION                                                                                               |                                  | DESIRED SALARY/WAGE                      | DATE YOU CAN START   |                       |
| ARE YOU CURRENTLY EMPLOYED?                                                                                     |                                  | MAY WE CONTACT YOUR PRESENT EMPLOYER, IF |                      |                       |
| HAVE YOU EVER APPLIED TO THIS COMPANY AND IF SO, WHEN?                                                          |                                  |                                          |                      |                       |
| <b>EDUCATIONAL BACKGROUND</b>                                                                                   |                                  |                                          |                      |                       |
| HIGH SCHOOL                                                                                                     | SCHOOL NAME & LOCATION           | DATES                                    | GRADUATED? (IF APP.) | SUBJECTS? (IF APP.)   |
|                                                                                                                 |                                  |                                          |                      |                       |
| COLLEGE                                                                                                         |                                  |                                          |                      |                       |
|                                                                                                                 |                                  |                                          |                      |                       |
| BUSINESS, TRADE OR CORRESPONDENCE SCHOOL(S)                                                                     |                                  |                                          |                      |                       |
|                                                                                                                 |                                  |                                          |                      |                       |
| <b>EMPLOYMENT HISTORY</b>                                                                                       |                                  |                                          |                      |                       |
| DATE<br>MONTH & YEAR                                                                                            | NAME & ADDRESS<br>OF EMPLOYER(S) | ENDING<br>SALARY                         | POSITION<br>HELD     | REASON FOR<br>LEAVING |
| FROM                                                                                                            |                                  |                                          |                      |                       |
| TO                                                                                                              |                                  |                                          |                      |                       |
| FROM                                                                                                            |                                  |                                          |                      |                       |
| TO                                                                                                              |                                  |                                          |                      |                       |
| FROM                                                                                                            |                                  |                                          |                      |                       |
| TO                                                                                                              |                                  |                                          |                      |                       |
| <b>REFERENCES</b> GIVE BELOW THE NAMES OF THREE PERSONS NOT RELATED TO YOU, WHOM YOU HAVE KNOWN AT LEAST 1 YEAR |                                  |                                          |                      |                       |
| NAME                                                                                                            | ADDRESS & PHONE NO.              | TYPE OF BUSINESS                         | YEARS KNOWN          |                       |
|                                                                                                                 |                                  |                                          |                      |                       |
|                                                                                                                 |                                  |                                          |                      |                       |
|                                                                                                                 |                                  |                                          |                      |                       |
|                                                                                                                 |                                  |                                          |                      |                       |

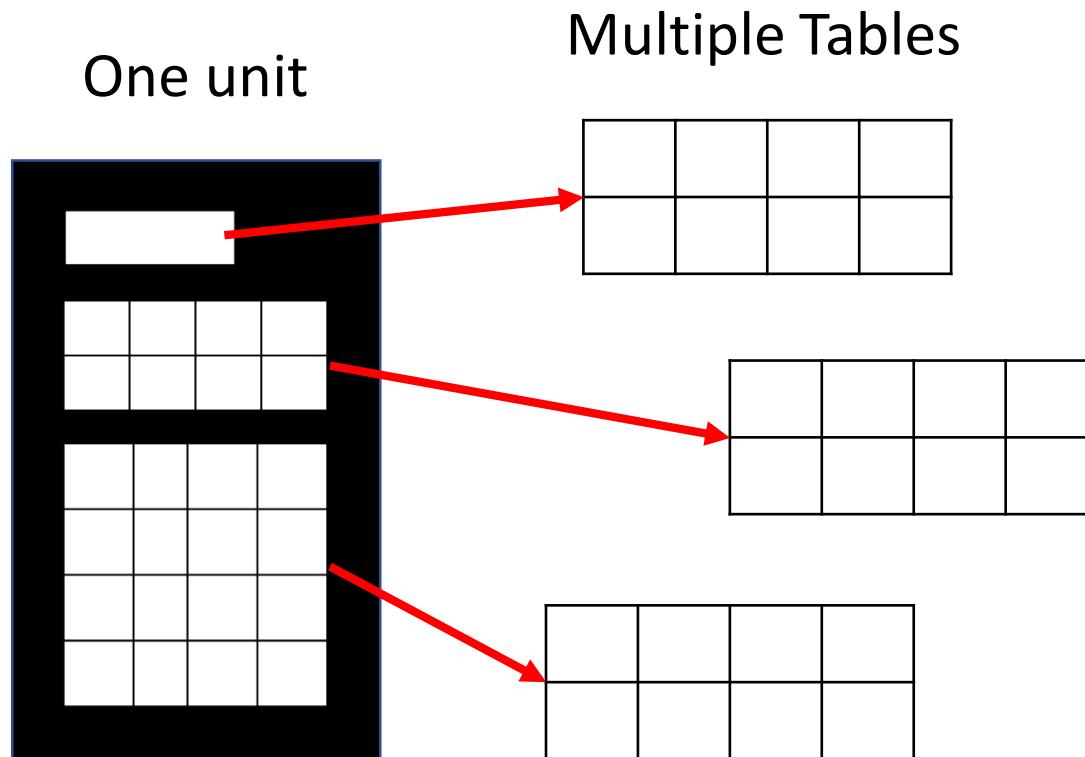
**Personal Info**

**Academic Profile**

**Employment**

**How will you design the DB for this content?**

# Impedance Mismatch Problem

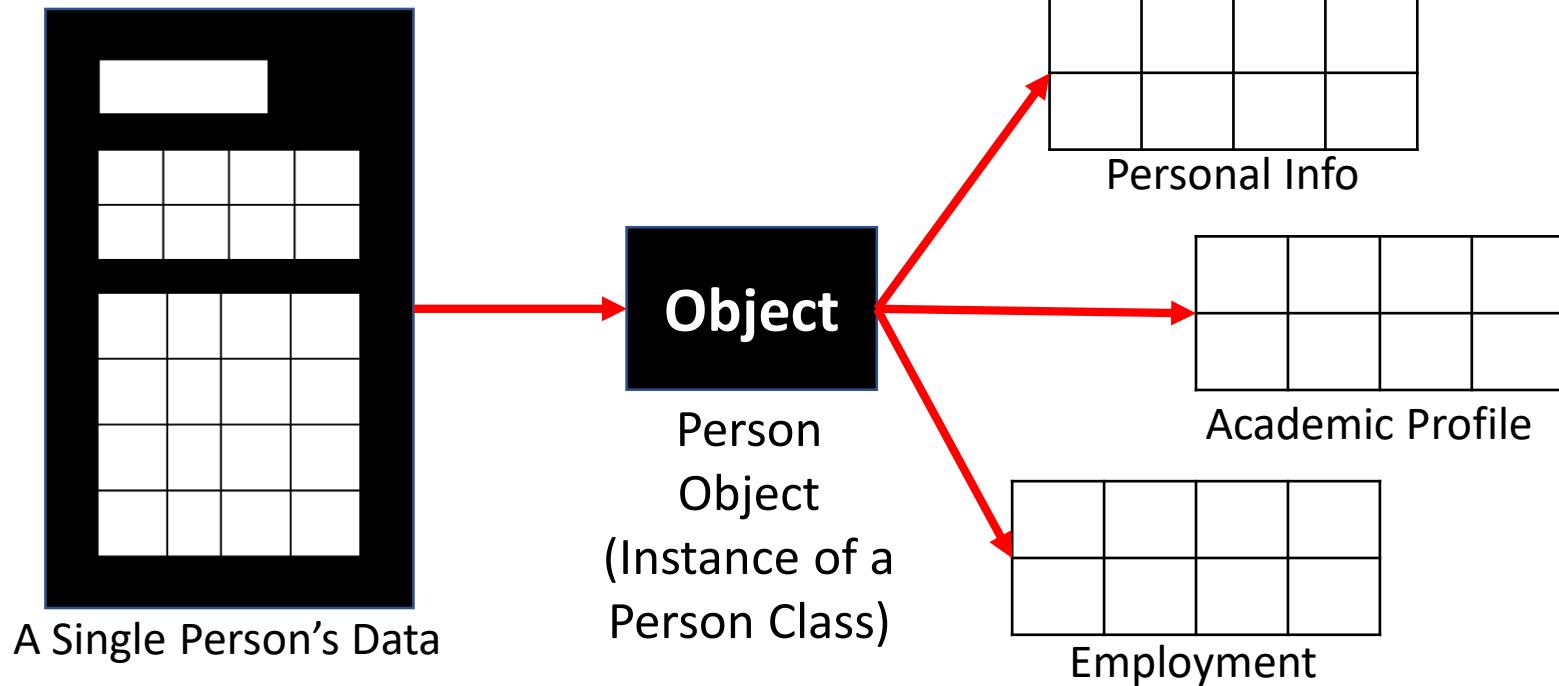


Intermediate Solution: Object Relational Mapping (ORM)

# Object Relational Mapping



Multiple Tables



Hibernate Framework, Java Data Objects, ... and  
many other ORM frameworks emerged.

# Scaling Out

# Table Joins Using MapReduce

- How would you do it?

Map-side  
Join

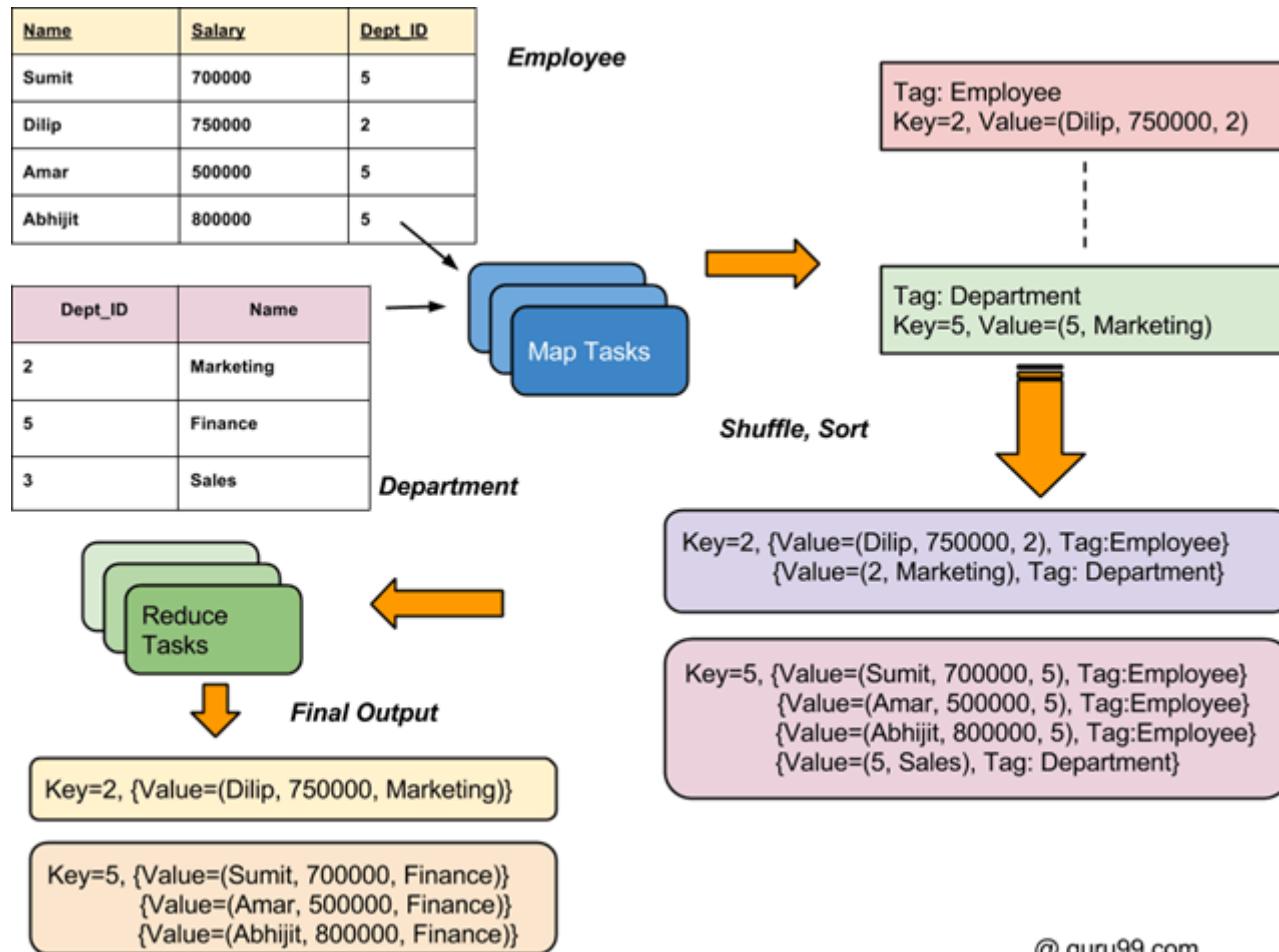
Join is performed by  
the mapper.

Reduce-side  
Join

Join is performed by  
the reducer.

Table joins are expensive. So, new solutions emerged. Google BigTable, Amazon Dynamo...

# Join Pattern



@ guru99.com

# A New Movement was Born

- We needed a
  - Not only relational
  - Cluster friendly
  - Schemaless
- way to store and retrieve data.
- Johan Oskarsson proposed a meetup. He needed a twitter hashtag. He used, “**nosql**”.

# Transactions, Consistency and CAP Theorem

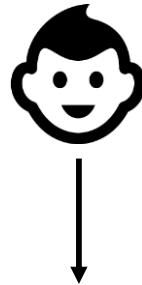
# Transaction

1. **read( $A$ )**
2.  $A := A - 50$
3. **write( $A$ )**
4. **read( $B$ )**
5.  $B := B + 50$
6. **write( $B$ )**

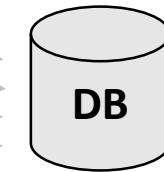
transfer \$50 from  
account A to account B

A **transaction** is a *unit* of program execution  
that accesses and possibly updates various  
data items.

# Do You See Any Issues Here?



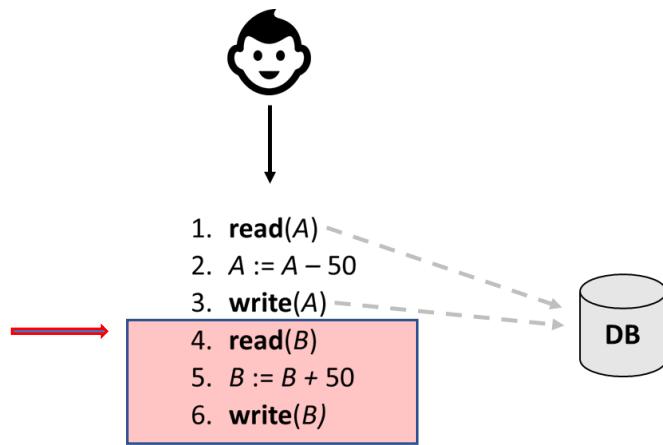
1. **read(A)**
2.  $A := A - 50$
3. **write(A)**
4. **read(B)**
5.  $B := B + 50$
6. **write(B)**



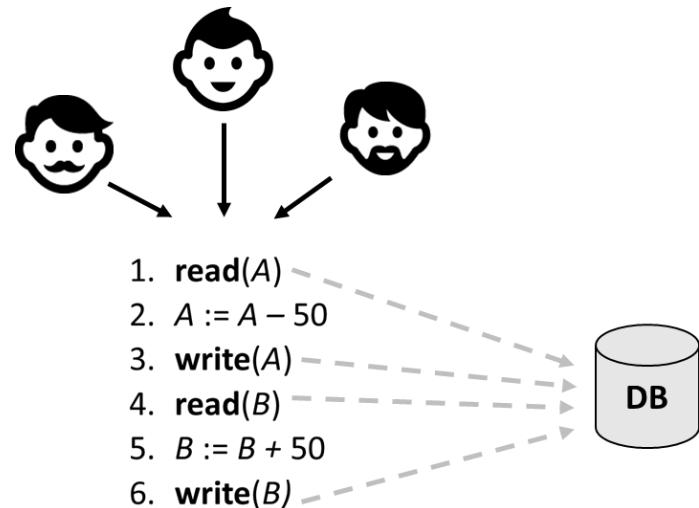
A transaction that reads  
and writes to disk.

# Issues

- Two main issues to deal with:



**Failure (hardware failure,  
system crash, software  
defect...)**



**concurrent execution**

# Atomicity

- **What happens if step 3 is executed but not step 6?**
    - Failure could be due to software or hardware
    - The system should ensure that updates of a partially executed transaction are not reflected in the database.
1. **read(A)**
  2.  $A := A - 50$
  3. **write(A)**
  4. **read(B)**
  5.  $B := B + 50$
  6. **write(B)**

# Consistency

- Respect

- Explicitly specified integrity constraints
- Implicit integrity constraints
  - e.g., sum of balances of all accounts stays constant

Temporarily  
Inconsistent  
State

Consistent State

- 
1. **read(A)**  
2.  $A := A - 50$   
3. **write(A)**  
4. **read(B)**  
5.  $B := B + 50$   
6. **write(B)**
- The diagram illustrates a sequence of database operations. It starts with a 'Consistent State' at the top, indicated by a red arrow pointing down to a bracket. Inside the bracket, six operations are listed: 1. **read(A)**, 2.  $A := A - 50$ , 3. **write(A)**, 4. **read(B)**, 5.  $B := B + 50$ , and 6. **write(B)**. After these operations, another red arrow points down to a second 'Consistent State' at the bottom.

Consistent State

# Isolation

- T2 sees an inconsistent database if T1 and T2 are concurrent.

| T1                 | T2                           |
|--------------------|------------------------------|
| 1. <b>read(A)</b>  |                              |
| 2. $A := A - 50$   |                              |
| 3. <b>write(A)</b> | read(A), read(B), print(A+B) |
| 4. <b>read(B)</b>  |                              |
| 5. $B := B + 50$   |                              |
| 6. <b>write(B)</b> |                              |

- Isolation can be ensured trivially by running transactions **serially**
  - That is, one after the other.

# Durability

- After step 6, the updates to the database by the transaction must
    - persist even if there are software or hardware failures.
1. **read(A)**
  2.  $A := A - 50$
  3. **write(A)**
  4. **read(B)**
  5.  $B := B + 50$
  6. **write(B)**

# ACID Properties

- **Atomicity.** Either all operations of the transaction are properly reflected in the database or none are.
- **Consistency.** Execution of a transaction in isolation preserves the consistency of the database.
- **Isolation.** Although multiple transactions may execute concurrently, each transaction must be unaware of other concurrently executing transactions. Intermediate transaction results must be hidden from other concurrently executed transactions.
  - That is, for every pair of transactions  $T_i$  and  $T_j$ , it appears to  $T_i$  that either  $T_j$  finished execution before  $T_i$  started, or  $T_j$  started execution after  $T_i$  finished.
- **Durability.** After a transaction completes successfully, the changes it has made to the database persist, even if there are system failures.

But, as a facebook user, I had a different observation...

# Eventual Consistency

- I updated my facebook status and asked my friend to check it out.
- But she found nothing there!!!
- Asked her to wait a bit and check again.
- Now, she finds it!



# Eventual Consistency

- Facebook is **eventually consistent**.
- Why not use a strongly consistent model?
  - Stores Petabytes of data.
  - We have **Availability** vs. **Consistency** tradeoff.

# CAP Theorem

- Concerns while designing distributed systems:
  - **Consistency** – all clients of a data store get responses to requests that ‘make sense’. For example, if Client A writes 1 and later 2 to location X, Client B cannot read 2 followed by 1.
  - **Availability** – all operations on a data store eventually return successfully. We say that a data store is ‘available’ for, e.g. write operations.
  - **Partition tolerance** – if the network stops delivering messages between two sets of servers, will the system continue to work correctly?

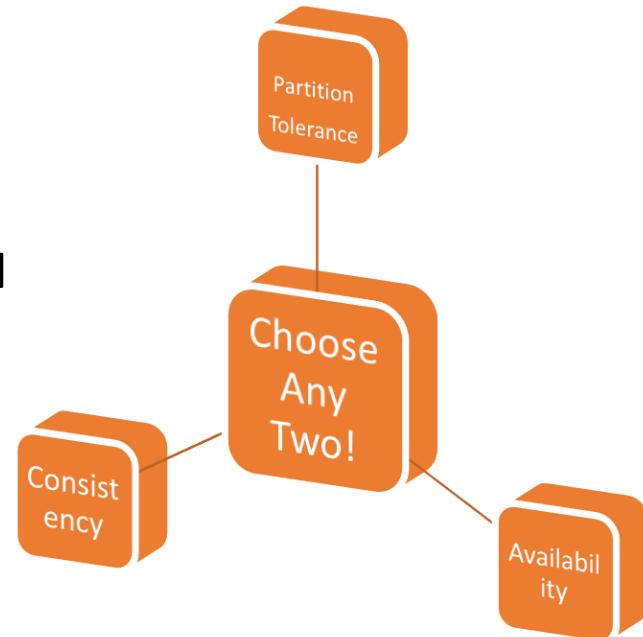
# The CAP Message

If you:

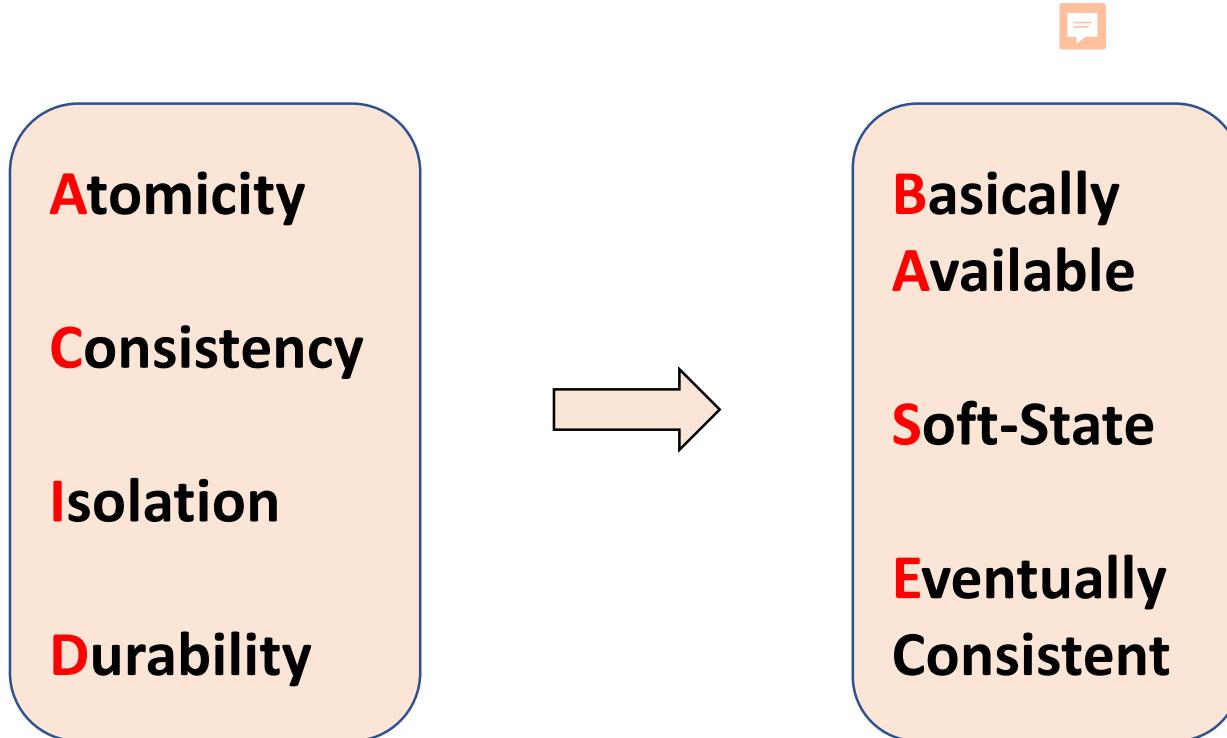
- cannot limit the number of faults,
- requests can be directed to any server, and
- insist on serving every request you receive,

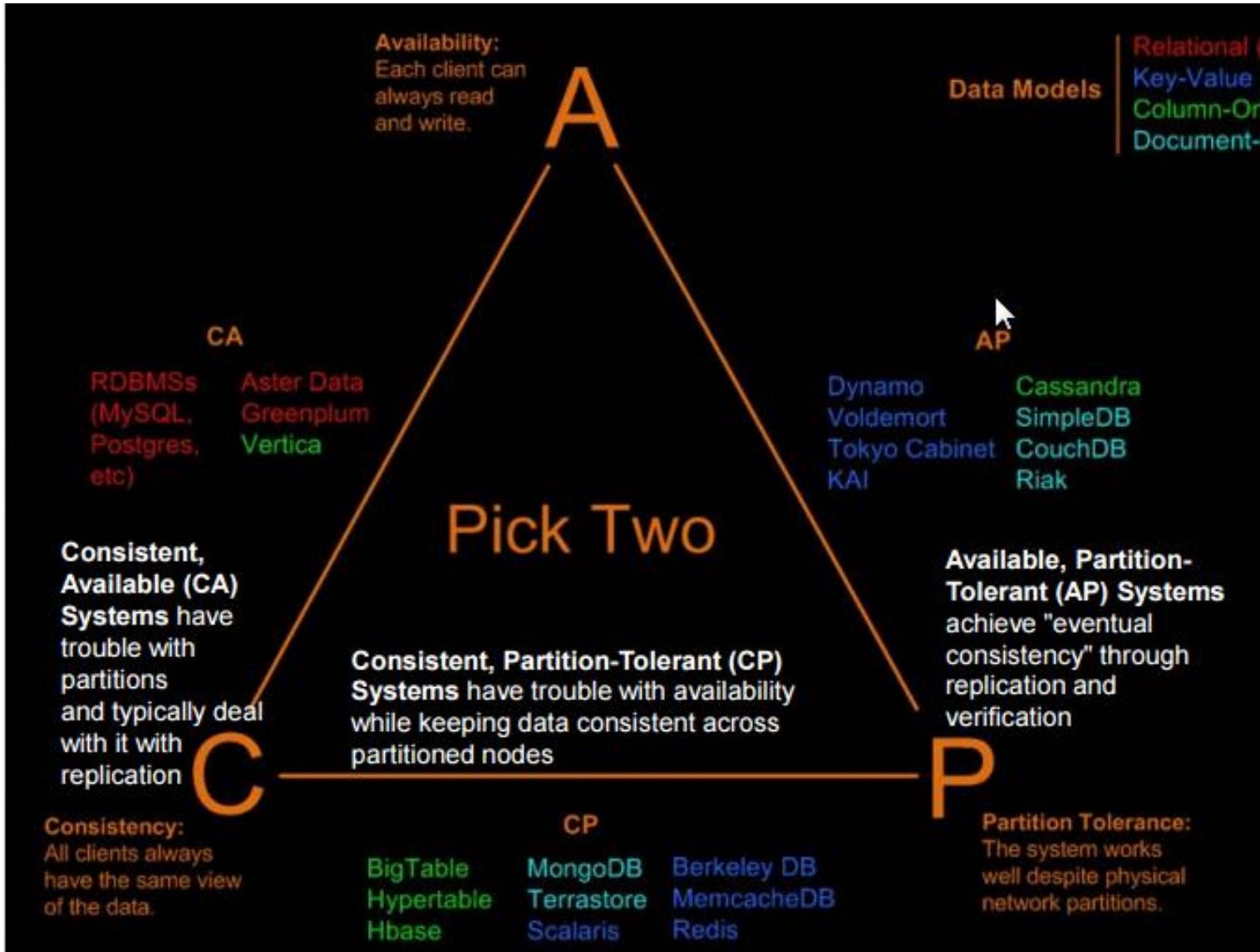
Then:

- you cannot possibly be consistent.



# The Transaction Properties





# NoSQL DB Types

# Types of NoSQL DB

- Key-Value Stores
  - Simplest. Every item is a key-value pair.
  - Examples: Riak, Voldemort, and Redis
- Document DB
  - Complex data structures are represented as documents.
  - Examples: MongoDB
- Wide-Column Stores
  - Data stored as columns.
  - Examples: Cassandra and Hbase
- Graph DB
  - Examples: Neo4J and HyperGraphDB

# Redis DB – Key Value Store

```
redis> GET nonexisting
```

```
(nil)
```

```
redis> SET mykey "Hello"
```

```
"OK"
```

```
redis> GET mykey
```

```
"Hello"
```

```
redis>
```

Read <https://redis.io/commands/get>

# Voldemort DB

```
> bin/voldemort-shell.sh test tcp://localhost:6666
Established connection to test via tcp://localhost:6666
> put "hello" "world"
> get "hello"
version(0:1): "world"
> delete "hello"
> get "hello"
null
> help
...
> exit
k k thx bye.
```

# mongoDB – Document Database

- mongoDB = “**Humongous DB**”
  - Open-source
  - Document-based data model
  - “High performance, high availability”
  - Automatic scaling
  - C-P on CAP

# MongoDB vs. RDBMS

- Collection vs. table
- Document vs. row
- Field vs. column
- Schema-less

## Document Data Model

- Documents are a natural way to represent data.
- Here is a “Person” object represented as a JSON document.
- MongoDB stores this as a BSON document (Binary representation of JSON).

```
{  
  name: "sue",  
  age: 26,  
  status: "A",  
  groups: [ "news", "sports" ]  
}
```

← field: value  
← field: value  
← field: value  
← field: value

A record in MongoDB  
is a document



```
db.myNewCollection2.insertOne( { x: 1 } )  
db.orders.deleteOne({ "name" : "al" });  
Commands
```

Read <https://docs.mongodb.com/manual/core/databases-and-collections/>

# Operations on MongoDB Data

Collection

```
db.orders.distinct( "cust_id" )
```

|                                                           |
|-----------------------------------------------------------|
| {<br>cust_id: "A123",<br>amount: 500,<br>status: "A"<br>} |
| {<br>cust_id: "A123",<br>amount: 250,<br>status: "A"<br>} |
| {<br>cust_id: "B212",<br>amount: 200,<br>status: "A"<br>} |
| {<br>cust_id: "A123",<br>amount: 300,<br>status: "D"<br>} |

distinct

```
[ "A123", "B212" ]
```

orders

# Columnar Storage

| SSN       | Name  | Age | Addr          | City    | St |
|-----------|-------|-----|---------------|---------|----|
| 101259797 | SMITH | 88  | 899 FIRST ST  | JUNO    | AL |
| 892375862 | CHIN  | 37  | 16137 MAIN ST | POMONA  | CA |
| 318370701 | HANDU | 12  | 42 JUNE ST    | CHICAGO | IL |

101259797|SMITH|88|899 FIRST ST|JUNO|AL    892375862|CHIN|37|16137 MAIN ST|POMONA|CA    318370701|HANDU|12|42 JUNE ST|CHICAGO|IL

Block 1

Block 2

Block 3

| SSN       | Name  | Age | Addr          | City    | St |
|-----------|-------|-----|---------------|---------|----|
| 101259797 | SMITH | 88  | 899 FIRST ST  | JUNO    | AL |
| 892375862 | CHIN  | 37  | 16137 MAIN ST | POMONA  | CA |
| 318370701 | HANDU | 12  | 42 JUNE ST    | CHICAGO | IL |

101259797 | 892375862 | 318370701 | 468248180 | 378568310 | 231346875 | 317346551 | 770336528 | 277332171 | 455124598 | 735885647 | 387586301

Block 1

Read [https://docs.aws.amazon.com/redshift/latest/dg/c\\_columnar\\_storage\\_disk\\_mem\\_mgmnt.html](https://docs.aws.amazon.com/redshift/latest/dg/c_columnar_storage_disk_mem_mgmnt.html)

# Columnar Storage

| SSN       | Name  | Age | Addr          | City    | St |
|-----------|-------|-----|---------------|---------|----|
| 101259797 | SMITH | 88  | 899 FIRST ST  | JUNO    | AL |
| 892375862 | CHIN  | 37  | 16137 MAIN ST | POMONA  | CA |
| 318370701 | HANDU | 12  | 42 JUNE ST    | CHICAGO | IL |

101259797 | 892375862 | 318370701 | 468248180 | 378568310 | 231346875 | 317346551 | 770336528 | 277332171 | 455124598 | 735885647 | 387586301

Block 1



**Same datatype in a block helps in devising efficient compression schemes. Therefore, improve storage efficiency.**

**Assumption:** “OLTP transactions typically involve most or all of the columns in a row for a small number of records, data warehouse queries commonly read only a few columns for a very large number of rows”

# Cassandra - Wide-Column Store

- A **column** is the basic data structure of Cassandra.
- A Column has three values, namely key or column name, value, and a time stamp.

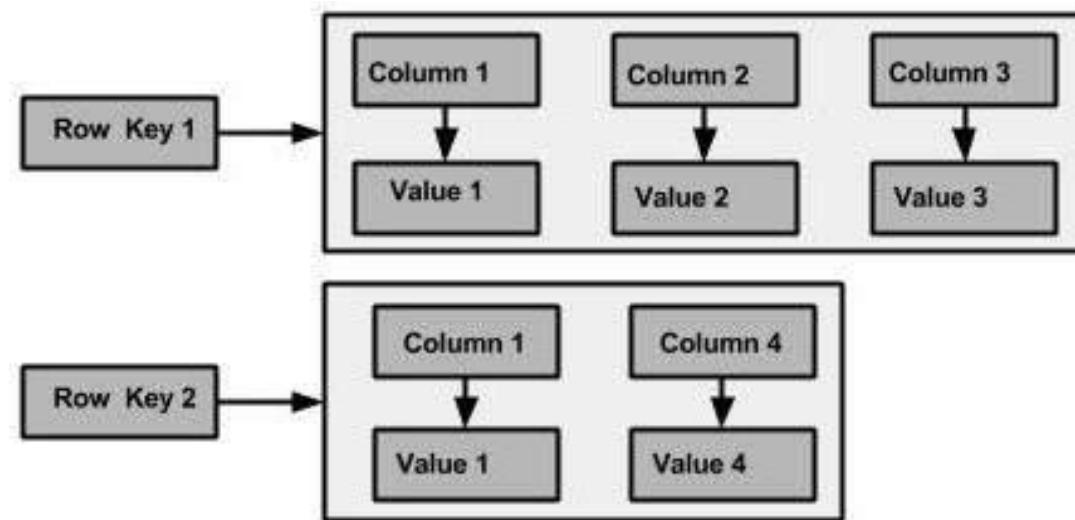
| Column        |                |                 |
|---------------|----------------|-----------------|
| name : byte[] | value : byte[] | clock : clock[] |

- A **super column** is a special column. stores a map of sub-columns.

| Super Column  |                            |
|---------------|----------------------------|
| name : byte[] | cols : map<byte[], column> |

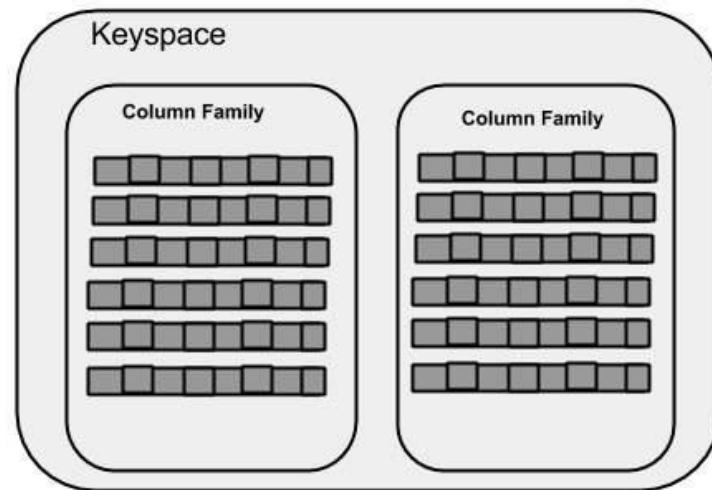
# Column-Family DB

- Cassandra does not force individual rows to have all the columns.
- An example of a Cassandra column family:



# Cassandra Keyspace

- Keyspace is a container for a list of one or more column families.
- A column family, in turn, is a container of a collection of rows.
- Each row contains ordered columns.



# cqlsh

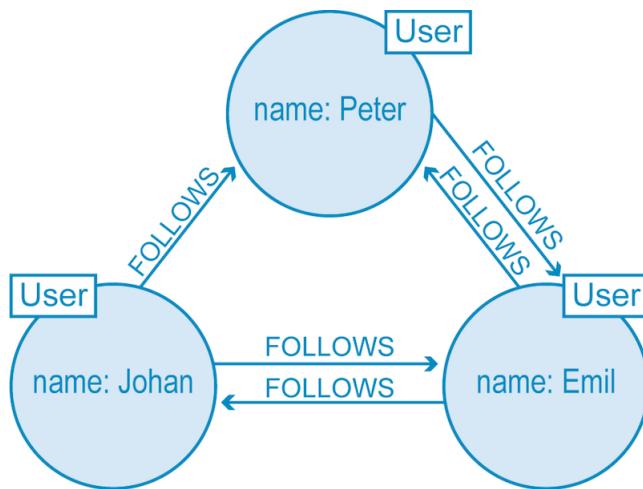
- Cassandra Query Language Shell

```
[hadoop@linux bin]$ cqlsh  
Connected to ... Cluster at ....  
cqlsh> select * from emp;
```

- Note: Cassandra does not join!
- If you need to lookup several tables, create another column-family.

# Graph DB

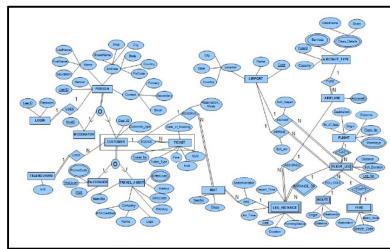
- Facebook, LinkedIn, Google ...have connected data.
- It is natural to store and retrieve data as graphs.



Twitter users represented in a graph database model.

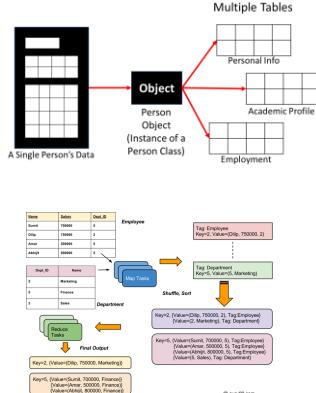
Read <https://neo4j.com/blog/why-graph-databases-are-the-future/>

# Summary

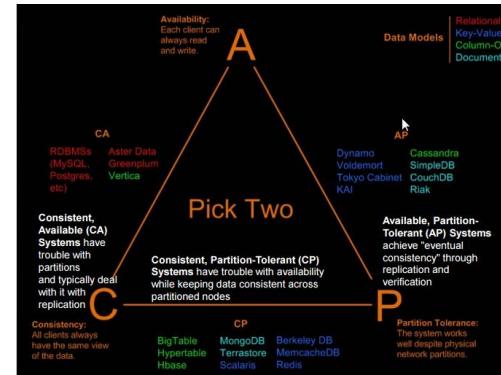


Schema-based  
Relational Model -  
maintenance  
problems

## Impedance Mismatch



## Scale-up Challenges



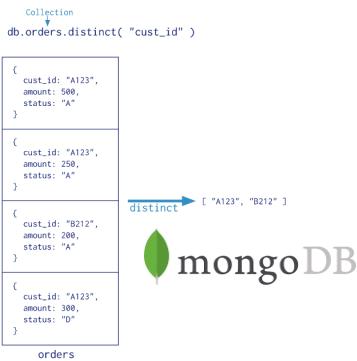
## CAP Theorem

### Key-Value

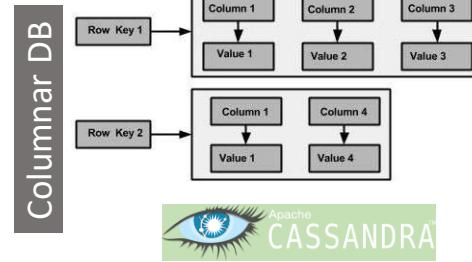
```
redis> GET nonexistent
(nil)
redis> SET mykey "Hello"
"OK"
redis> GET mykey
"Hello"
redis>
```



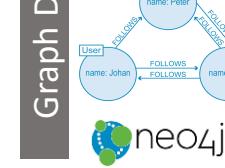
### Doc-based



### Columnar DB



### Graph DB



Thank You

# MONGO DB TUTORIAL

---

Hands-on Session

by Suchitra Jayaprakash  
[suchitra@cmi.ac.in](mailto:suchitra@cmi.ac.in)

# MONGO DB

- Document Oriented Database.
- Data is stored as documents - JSON like syntax - javascript object notation.
- Database : Physical container for collection.
- Collection: Group of MongoDB documents.
- Document: Set of key-value pairs.
- Dynamic schema.

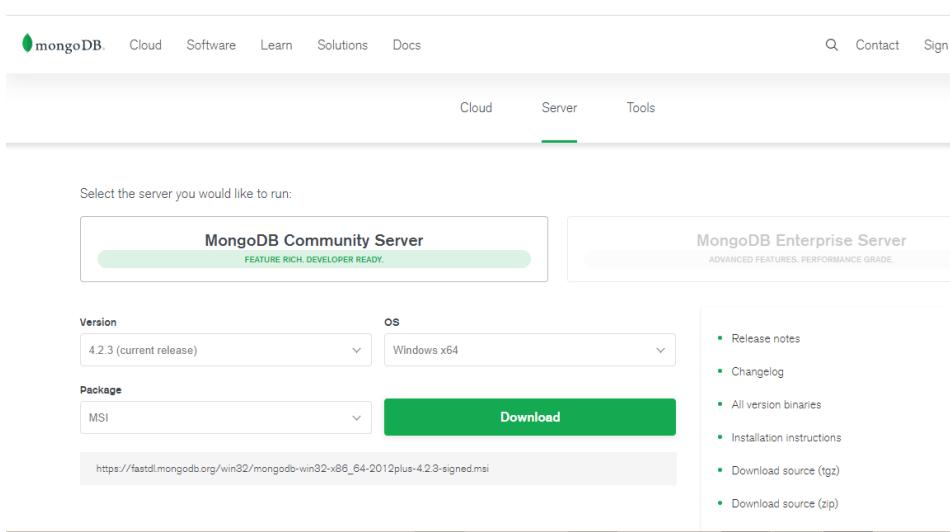
# MONGO DB

| RDBMS       | MONGODB         |
|-------------|-----------------|
| Database    | Database        |
| Table       | Collection      |
| Row         | Document        |
| Column      | Field           |
| Primary Key | Default key _id |

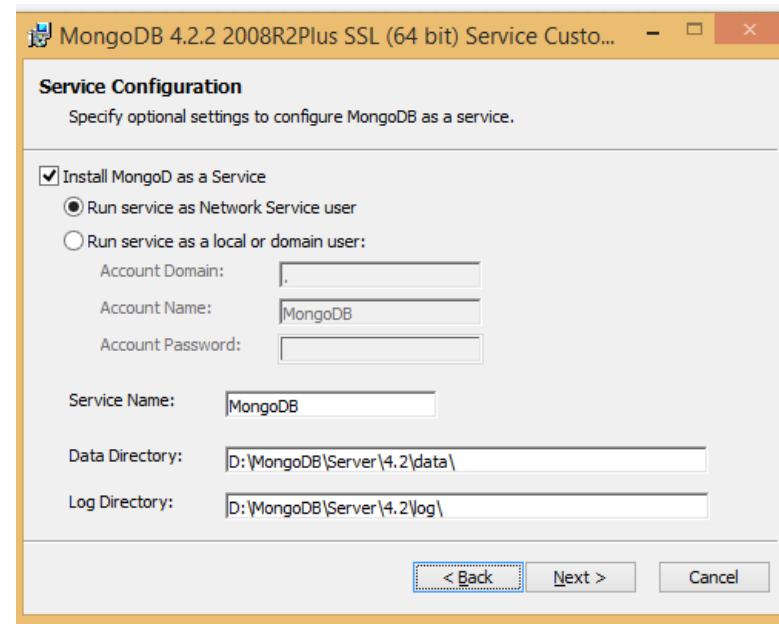
- **Some advantages :**
  - Schema less.
  - Conversion of application objects to database objects is not required.
  - No complex joins

# MONGO DB Installation

- Download the latest release of MongoDB from  
<https://www.mongodb.com/download-center>
  - Refer below site for detailed instructions:  
<https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows/>
- Run installation file
  - Select complete install



The screenshot shows the MongoDB download center. At the top, there are navigation links: Cloud, Software, Learn, Solutions, and Docs. Below that is a navigation bar with Cloud, Server (which is selected), and Tools. A search bar, Contact link, and Sign In button are also present. The main content area is titled "Select the server you would like to run:" and features two options: "MongoDB Community Server" (selected) and "MongoDB Enterprise Server". Under "MongoDB Community Server", there are dropdown menus for "Version" (4.2.3 (current release)) and "OS" (Windows x64). A "Package" dropdown is set to "MSI". A large green "Download" button is prominently displayed. To the right of the download button is a list of download links: Release notes, Changelog, All version binaries, Installation instructions, Download source (.tgz), and Download source (.zip). At the bottom of the page, the URL [https://fastdl.mongodb.org/win32/mongodb-win32-x86\\_64-2012plus-4.2.3-signed.msi](https://fastdl.mongodb.org/win32/mongodb-win32-x86_64-2012plus-4.2.3-signed.msi) is provided.



# MONGO DB Installation

- MongoDB will be installed in the folder C:\Program Files\
- MongoDB requires a data folder to store its files.
- Specify the dbpath while executing mongod.exe.
- In the command prompt, navigate to the bin directory current in the MongoDB installation folder.
- Go to " C:\Program Files\MongoDB\Server\4.2\bin"
- First run the Mongo Daemon :
  - mongod.exe --dbpath "C:\data"
- Double click on mongo.exe

# MONGO DB COMMANDS

- **Create Database**

use <dbname>

to know current database , type db

db.stats()

shows database metadata

show dbs

- **Create user**

db.createUser({

  user:"suchi",

  pwd:"123",

  roles:["readWrite","dbAdmin"]

})

```
> use myfirst
switched to db myfirst
> db
myfirst
> db.stats()
{
  "db" : "myfirst",
  "collections" : 0,
  "views" : 0,
  "objects" : 0,
  "avgObjSize" : 0,
  "dataSize" : 0,
  "storageSize" : 0,
  "numExtents" : 0,
  "indexes" : 0,
  "indexSize" : 0,
  "scaleFactor" : 1,
  "fileSize" : 0,
  "fsUsedSize" : 0,
  "fsTotalSize" : 0,
  "ok" : 1
}
> db.createUser({
...   user:"suchi",
...   pwd:"123",
...   roles:["readWrite","dbAdmin"]
... });
Successfully added user: { "user" : "suchi", "roles" : [ "readWrite", "dbAdmin" ] }
```

# MONGO DB COMMANDS

- Create Collections

```
db.createCollection('students');
```

- Displays all collections

```
show collections
```

- Insert Data

```
db.students.insert({  
    First_Name:"John",  
    Last_Name:"Dicken"  
})
```

```
db.students.insert({  
    First_Name:"Mary",  
    Last_Name:"Kate",  
    Gender:"Female",  
    age:23})
```

- Bulk Insert Data

```
db.students.insert([  
    {  
        First_Name:"Sam",  
        Last_Name:"Jackson",  
        email:["sam@cmi.ac.in","sam@gmail.com"]},  
    { First_Name:"Chris",  
    Last_Name:"Jack",  
    Gender:"Male"}  
])
```

```
> db.students.insert({  
...   First_Name:"John",  
...   Last_Name:"Dicken"  
... },  
... );  
WriteResult{  
  "nInserted": 1  
}  
> db.students.insert({  
...   First_Name:"Mary",  
...   Last_Name:"Kate",  
...   Gender:"Female",  
...   age:23  
... },  
... );  
WriteResult{  
  "nInserted": 1  
}  
> db.students.insert([  
...   {  
...     First_Name:"Sam",  
...     Last_Name:"Jackson",  
...     email:["sam@cmi.ac.in","sam@gmail.com"]},  
...   { First_Name:"Chris",  
...     Last_Name:"Jack",  
...     Gender:"Male"}  
... ],  
... );  
BulkWriteResult{  
  "writeErrors": [ ],  
  "writeConcernErrors": [ ],  
  "nInserted": 2,  
  "nUpserted": 0,  
  "nMatched": 0,  
  "nModified": 0,  
  "nRemoved": 0,  
  "upserted": [ ]  
}  
> ■
```

# MONGO DB COMMANDS

## List of records:

```
db.students.find()
```

```
db.students.find().pretty()
```

```
db.students.insert([
```

```
    {First_Name:"Jim",
     Last_Name:"Carry",
     email:["Jim@cmi.ac.in","Jim@gmail.com"],
     Address:{  

         houseno:"123/1",
         street:"1st cross st",
         locality:"chennai",
         pincode:"600073"}  

    }  

])
```

```
db.students.find( { 'Address.pincode': "600073" })
```

```
> db.students.find( { 'Address.pincode': "600073" });
{ "_id" : ObjectId("5e5a80fd94d2a62f45369249"), "First_Name" : "Jim", "Last_Name" : "Carry", "email" : [ "Jim@cmi.ac.in", "Jim@gmail.com" ], "Address" : { "houseno" : "123/1", "street" : "1st cross st", "locality" : "chennai", "pincode" : "600073" } }
```

```
> db.students.find()
{ "_id" : ObjectId("5e5a7b6494d2a62f45369245"), "First_Name" : "John", "Last_Name" : "Dicken" }
{ "_id" : ObjectId("5e5a7b7594d2a62f45369246"), "First_Name" : "Mary", "Last_Name" : "Kate", "Gender" : "Female", "age" : 23 }
{ "_id" : ObjectId("5e5a7c1a94d2a62f45369247"), "First_Name" : "Sam", "Last_Name" : "Jackson", "email" : [ "sam@cmi.ac.in", "sam@gmail.com" ] }
{ "_id" : ObjectId("5e5a7c1a94d2a62f45369248"), "First_Name" : "Chris", "Last_Name" : "Jack", "Gender" : "Male" }
> db.students.find().pretty()
{
  "_id" : ObjectId("5e5a7b6494d2a62f45369245"),
  "First_Name" : "John",
  "Last_Name" : "Dicken"
}
{
  "_id" : ObjectId("5e5a7b7594d2a62f45369246"),
  "First_Name" : "Mary",
  "Last_Name" : "Kate",
  "Gender" : "Female",
  "age" : 23
}
{
  "_id" : ObjectId("5e5a7c1a94d2a62f45369247"),
  "First_Name" : "Sam",
  "Last_Name" : "Jackson",
  "email" : [
    "sam@cmi.ac.in",
    "sam@gmail.com"
  ]
}
{
  "_id" : ObjectId("5e5a7c1a94d2a62f45369248"),
  "First_Name" : "Chris",
  "Last_Name" : "Jack",
  "Gender" : "Male"
}
```

```
@shell>9:5
> db.students.insert([
...   {First_Name:"Jim",
...    Last_Name:"Carry",
...    email:["Jim@cmi.ac.in","Jim@gmail.com"],
...    Address:{  

...      houseno:"123/1",
...      street:"1st cross st",
...      locality:"chennai",
...      pincode:"600073"}  

...    }  

...  ]);
BulkWriteResult<{
  "writeErrors" : [ ],
  "writeConcernErrors" : [ ],
  "nInserted" : 1,
  "nUpserted" : 0,
  "nMatched" : 0,
  "nModified" : 0,
  "nRemoved" : 0,
  "upserted" : [ ]
}>
```

# MONGO DB COMMANDS

- **update record**

```
db.students.update({First_Name:"Sam"},{First_Name:"Samuel",Last_Name:"Jackson"})
```

```
db.students.update({First_Name:"Samuel"},{$set:{Gender:"Male"}})
```

```
> db.students.update({First_Name:"Sam"},{First_Name:"Samuel",Last_Name:"Jackson"})
> WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.students.update({First_Name:"Samuel"},{$set:{Gender:"Male"}});
> WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

```
db.students.update({First_Name:"Juli"},{$inc:{age:2}})
```

```
> db.students.update({First_Name:"Juli"},{$inc:{age:2}});
> WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
```

```
db.students.update({First_Name:"Mary"},{$unset:{Last_Name:""}})
```

```
> db.students.update({First_Name:"Mary"},{$unset:{Last_Name:""}});
> WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

# MONGO DB COMMANDS

- Remove record

```
db.students.remove({First_Name:"Chris"});
```

```
> db.students.remove({First_Name:"Chris"});
WriteResult({ "nRemoved" : 1 })
> █
```

- Find record

```
db.students.find({First_Name:"Mary"})
```

```
db.students.find({$or:[{First_Name:"Samuel"}, {First_Name:"Mary"}]})
```

```
db.students.find({age:{$gt:12}})
```

```
> db.students.find({First_Name:"Mary"});
{
  "_id" : ObjectId("5e5a7b7594d2a62f45369246"),
  "First_Name" : "Mary",
  "Gender" : "Female",
  "age" : 25
}
> db.students.find({$or:[{First_Name:"Samuel"}, {First_Name:"Mary"}]});
{
  "_id" : ObjectId("5e5a7b7594d2a62f45369246"),
  "First_Name" : "Mary",
  "Gender" : "Female",
  "age" : 25
}
{
  "_id" : ObjectId("5e5a7c1a94d2a62f45369247"),
  "First_Name" : "Samuel",
  "Last_Name" : "Jackson",
  "Gender" : "Male"
}
> db.students.find({age:{$gt:12}});
{
  "_id" : ObjectId("5e5a7b7594d2a62f45369246"),
  "First_Name" : "Mary",
  "Gender" : "Female",
  "age" : 25
}
```

```
db.students.find().count()
```

```
>
> db.students.find().count();
4
> █
```

# MONGO DB COMMANDS

- Sort

```
db.students.find().sort({Last_name:-1}).pretty()
```

```
db.students.find().sort({Last_name:1}).pretty()
```

1 is used for ascending order while -1 is used for descending order.

```
> db.students.find().sort({Last_Name:-1}).pretty();
{
  "_id" : ObjectId("5e5a7c1a94d2a62f45369247"),
  "First_Name" : "Samuel",
  "Last_Name" : "Jackson",
  "Gender" : "Male"

  "_id" : ObjectId("5e5a7b6494d2a62f45369245"),
  "First_Name" : "John",
  "Last_Name" : "Dicken"

  "_id" : ObjectId("5e5a80fd94d2a62f45369249"),
  "First_Name" : "Jim",
  "Last_Name" : "Carry",
  "email" : [
    "Jim@cmi.ac.in",
    "Jim@gmail.com"
  ],
  "Address" : {
    "houseno" : "123/1",
    "street" : "1st cross st",
    "locality" : "chennai",
    "pincode" : "600073"
  }

  "_id" : ObjectId("5e5a7b7594d2a62f45369246"),
  "First_Name" : "Mary",
  "Gender" : "Female",
  "age" : 25
}
```

```
> db.students.find().sort({Last_Name:1}).pretty();
{
  "_id" : ObjectId("5e5a7b7594d2a62f45369246"),
  "First_Name" : "Mary",
  "Gender" : "Female",
  "age" : 25

  "_id" : ObjectId("5e5a80fd94d2a62f45369249"),
  "First_Name" : "Jim",
  "Last_Name" : "Carry",
  "email" : [
    "Jim@cmi.ac.in",
    "Jim@gmail.com"
  ],
  "Address" : {
    "houseno" : "123/1",
    "street" : "1st cross st",
    "locality" : "chennai",
    "pincode" : "600073"
  }

  "_id" : ObjectId("5e5a7b6494d2a62f45369245"),
  "First_Name" : "John",
  "Last_Name" : "Dicken"

  "_id" : ObjectId("5e5a7c1a94d2a62f45369247"),
  "First_Name" : "Samuel",
  "Last_Name" : "Jackson",
  "Gender" : "Male"
}
```

# MONGO DB COMMANDS

- Aggregation

```
db.students.aggregate([{$group : {_id : "$Gender", count : {$sum : 1}}}] )
```

```
> db.students.aggregate([{$group : {_id : "$Gender", count : {$sum : 1}}}] )  
{ "_id" : null, "count" : 4 }  
{ "_id" : "Male", "count" : 2 }  
{ "_id" : "Female", "count" : 2 }  
>
```

- User Defined function

```
db.students.find().forEach(function(doc){print( "First Name is "+ doc.First_Name )})
```

```
> db.students.find().forEach(function(doc){print( "First Name is "+ doc.First_Name )});  
First Name is John  
First Name is Mary  
First Name is Samuel  
First Name is Jim  
> █
```

Reference: <https://docs.mongodb.com/manual/crud/>

**THANK YOU**

# HBASE TUTORIAL

---

Hands-on Session

by Suchitra Jayaprakash  
[suchitra@cmi.ac.in](mailto:suchitra@cmi.ac.in)

# HBASE

- Distributed column-oriented database built on top of the Hadoop file system
- Designed for quick random access.
- Columns are grouped into column families, which must be defined during table creation.
- Basic structure
  - Column – single field in table
  - Column-family – is group of columns
  - Row-key – it is a mandatory field which serves as the unique identifier for every record.

# Run HBASE

- Start Cloudera server

```
docker run --hostname=quickstart.cloudera --privileged=true -t -i --  
publish-all=true -p 8888:8888 -p 8080:80 -p 50070:50070 -p 8088:8088 -p  
50075:50075 -p 8032:8032 -p 8042:8042 -p 19888:19888  
cloudera/quickstart /usr/bin/docker-quickstart
```

- To get HBASE command prompt
  - type "hbase shell" and press enter

# HBASE COMMANDS

- Create table

create <'tablename'>, <'columnfamilyname'>

create 'Student' , 'personal\_data','academic\_data','other\_data'

```
[root@quickstart ~]# hbase shell
2020-03-03 14:35:40,686 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.7.0, rUnknown, Wed Mar 23 11:39:14 PDT 2016

hbase(main):001:0> create 'Student' , 'personal_data','academic_data','other_data'
0 row(s) in 4.3130 seconds

=> Hbase::Table - Student
hbase(main):002:0> █
```

- Display All tables

list

displays all the tables that are present or created in HBase

```
[~] hbase>Table - Student
hbase(main):002:0> list
TABLE
Student
1 row(s) in 0.1070 seconds

=> ["Student"]
hbase(main):003:0> █
```

# HBASE COMMANDS

- Add cell value

```
put <'tablename'>,<'rowname'>,<'colfamily:colname'>,<'value'>
```

```
put 'Student','S101','personal_data:name','John'  
put 'Student','S101','personal_data:address','#145, New Road,Chennai'  
put 'Student','S101','academic_data:class','Course A'  
put 'Student','S101','academic_data:year','second'
```

Inserting data into HBase

- Get a value

```
get '<table-name>','<row-key>','<column-family>'
```

```
get 'Student','S101'
```

retrieve single record from HBase table

```
=> ["Student"]  
hbase(main):003:0> put 'Student','S101','personal_data:name','John'  
0 row(s) in 0.8380 seconds  
  
hbase(main):004:0> put 'Student','S101','personal_data:address','#145, New Road,  
Chennai'  
0 row(s) in 0.0580 seconds  
  
hbase(main):005:0> put 'Student','S101','academic_data:class','Course A'  
0 row(s) in 0.0350 seconds  
  
hbase(main):006:0> put 'Student','S101','academic_data:year','second'  
0 row(s) in 0.0650 seconds
```

```
hbase(main):012:0> get 'Student','S101'  
COLUMN CELL  
academic_data:class timestamp=1583246362745, value=Course A  
academic_data:year timestamp=1583246363123, value=second  
personal_data:address timestamp=1583246362285, value=#145, New Road,Chennai  
personal_data:name timestamp=1583246430789, value=John  
4 row(s) in 0.1480 seconds  
  
hbase(main):013:0>
```

# HBASE COMMANDS

- Display row count of table

count <'tablename'>

count 'Student'

```
hbase(main):013:0> count 'Student'  
1 row(s) in 0.3770 seconds  
=> 1  
hbase(main):014:0> █
```

- Display table data

scan '<table -name>'

scan 'Student'

get the all the records

```
hbase(main):020:0> scan 'Student'  
ROW                                COLUMN+CELL  
S101                               column=academic_data:class, timestamp=1583246362745, value=  
   =Course A  
S101                               column=academic_data:year, timestamp=1583246363123, value=  
   second  
S101                               column=personal_data:address, timestamp=1583246362285, val  
ue=#145, New Road,Chennai  
S101                               column=personal_data:name, timestamp=1583246430789, value=  
John  
1 row(s) in 0.1530 seconds
```

# HBASE COMMANDS

- **Deleting records**

- delete '<table-name>','<rowkey>','<column-family>'

- delete 'Student','S101','academic\_data:class'

```
hbase(main):021:0> delete 'Student','S101','academic_data:class'
0 row(s) in 0.1480 seconds
```

- **Delete table**

disable 'Employee'

```
hbase(main):022:0> scan 'Student'
ROW                               COLUMN+CELL
  $101                            column=academic_data:year, timestamp=1583246363123, value=
   second
  $101                            column=personal_data:address, timestamp=1583246362285, val
   ue=#145, New Road,Chennai
  $101                            column=personal_data:name, timestamp=1583246430789, value=
   John
1 row(s) in 0.2120 seconds
hbase(main):022:0>
```

drop 'Employee'

# HBASE COMMANDS

- **loading HDFS file as table**
- 1)Using ImportTsv to load txt to Hbase

```
docker cp E:/MSc_Datascience/BigDataHadoop/Slides/SampleName.txt  
c1ce02333912:/tmp/SampleName.txt
```

```
hadoop fs -copyFromLocal /tmp/SampleName.txt SampleName.txt
```

- 2) create table
- create 'Person' , 'Name'
- 3) import data (execute it outside hbase shell)

```
hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -  
Dimporttsv.separator=',' -Dimporttsv.columns='HBASE_ROW_KEY,Name'  
Person hdfs://quickstart.cloudera/user/root/SampleName.txt
```

# OUTPUT

```
2020-03-03 15:11:51,768 INFO [main] mapreduce.Job:  map 100% Reduce 0%
2020-03-03 15:11:54,141 INFO [main] mapreduce.Job: Job job_1583245414405_0001 completed successfully
2020-03-03 15:11:54,936 INFO [main] mapreduce.Job: Counters: 31
    File System Counters
        FILE: Number of bytes read=0
        FILE: Number of bytes written=146345
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=155
        HDFS: Number of bytes written=0
        HDFS: Number of read operations=2
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=0
    Job Counters
        Launched map tasks=1
        Data-local map tasks=1
        Total time spent by all maps in occupied slots (ms)=34621
        Total time spent by all reduces in occupied slots (ms)=0
        Total time spent by all map tasks (ms)=34621
        Total vcore-seconds taken by all map tasks=34621
        Total megabyte-seconds taken by all map tasks=35451904
    Map-Reduce Framework
        Map input records=5
        Map output records=5
        Input split bytes=116
        Spilled Records=0
        Failed Shuffles=0
        Merged Map outputs=0
        GC time elapsed (ms)=578
        CPU time spent (ms)=6810
        Physical memory (bytes) snapshot=142856192
        Virtual memory (bytes) snapshot=1307754496
        Total committed heap usage (bytes)=62652416
    ImportTsv
        Bad Lines=0
    File Input Format Counters
        Bytes Read=39
    File Output Format Counters
        Bytes Written=0
[root@quickstart ~]#
```

```
hbase(main):001:0> scan 'Person'
ROW                               COLUMN+CELL
1       column=Name:, timestamp=1583248180951, value=John
2       column=Name:, timestamp=1583248180951, value=Mary
3       column=Name:, timestamp=1583248180951, value=Sam
4       column=Name:, timestamp=1583248180951, value=Jerry
5       column=Name:, timestamp=1583248180951, value=Tom
5 row(s) in 1.5610 seconds

hbase(main):002:0> █
```

**THANK YOU**

# Web Application Development and Web Services

**Venkatesh Vinayakarao**

venkateshv@cmi.ac.in

<http://vvtesh.co.in>

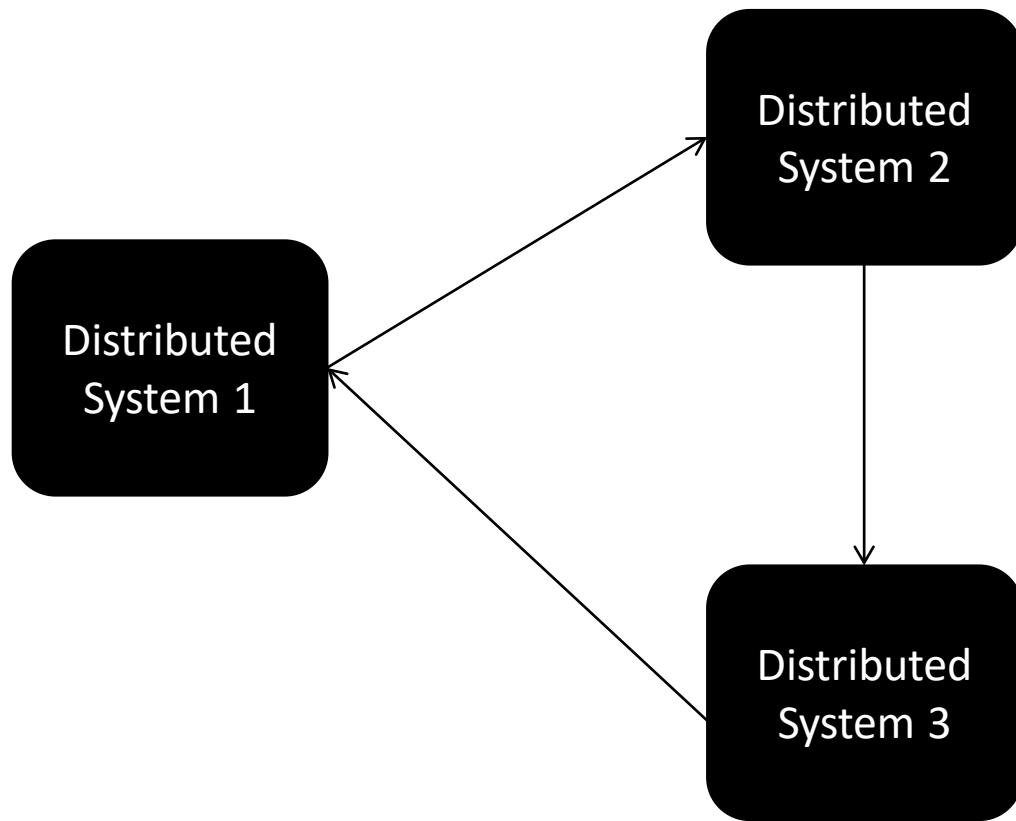
---

Chennai Mathematical Institute

---

If You Think Math is Hard Try Web Design. – **PixxelzNet.**

# How to Achieve Interoperability?

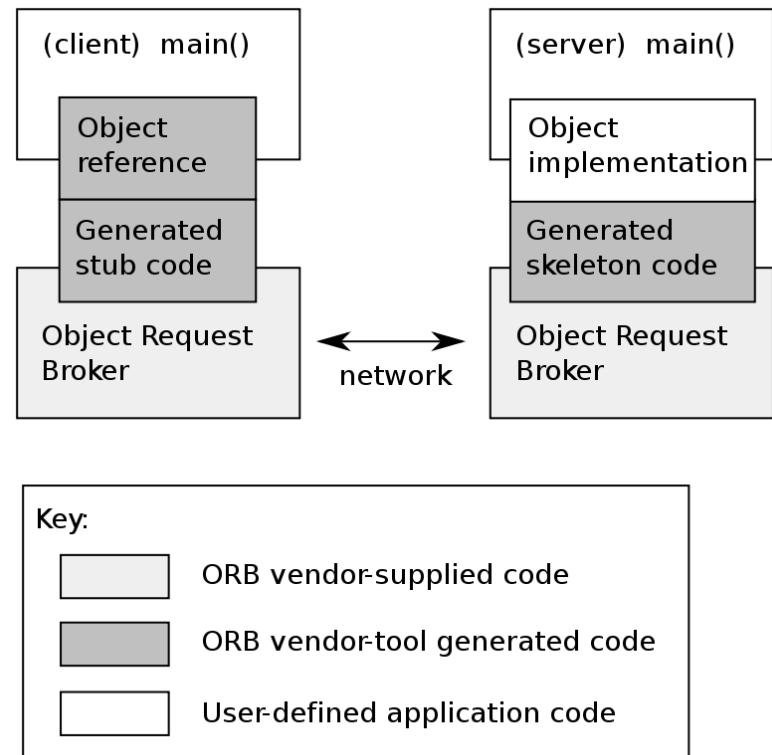


# Interoperability Solutions

- Many Solutions
  - File Transfer
  - Shared DB
  - Remote Procedure Calls
  - Message Passing
- Middleware platforms aimed at making it more structured and easier
  - CORBA, DCOM, RMI, ...
  - Web Services

# Interoperability Solutions

- CORBA (1991)
  - Standards-based, vendor-neutral, and language-agnostic.
  - Communicate by message passing over network
  - Read [Corba: Gone But \(Hopefully\) Not Forgotten](#), Queue Vol 5, No. 4.



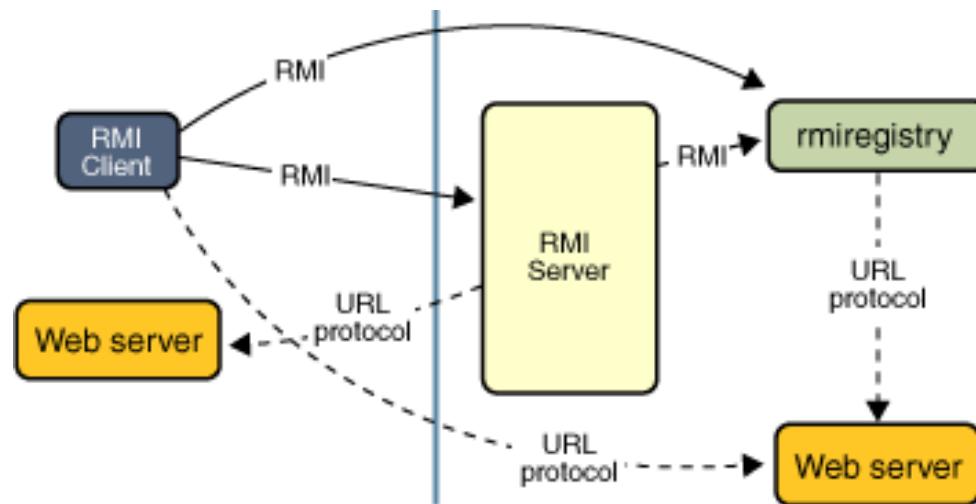
<https://www.omg.org/spec/CORBA/>

[https://en.wikipedia.org/wiki/Common\\_Object\\_Request\\_Broker\\_Architecture](https://en.wikipedia.org/wiki/Common_Object_Request_Broker_Architecture)

<https://docs.oracle.com/javase/8/docs/technotes/guides/idl/jidlExample.html>

# More Interoperability Solutions

- Distributed Component Object Model (DCOM) (Microsoft)
- RMI (Sun Microsystems)
  - Invoke method on a remote object.



<https://docs.oracle.com/javase/tutorial/rmi/overview.html>

# Web Services

- A “**service**” is a software component provided through an (often, network-accessible) endpoint.
- Service consumer and provider use messages to exchange invocation request and response information in the form of self-containing documents.

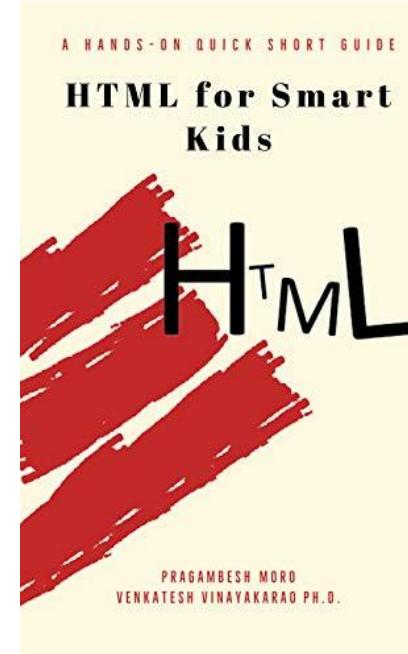
What do you understand by  
“**Web**”?

# Early Static Web

- Developed in 1990 at CERN
- NCSA Mosaic 1.0 was the first browser, released by the National Center for Supercomputer Applications (NCSA).

# Creating Web Pages

- Write HTML code.
- Move it to a *Web Server*.
- Access it over the web.



# The Dynamic Web

- Httpd 1.0 web server allowed Common Gateway Interface (CGI).
- CGI allows a browser client to request data from a program running on a Web server.

# CGI Script



```
#!/usr/local/bin/perl
# Display the form data set sent in a GET or POST request.

print "Content-type: text/html\n\n";
print "<html><head><title>Form Data</title></head>\n";
print "<body bgcolor=\"#FFFFFF\"\n>

if ($ENV{'REQUEST_METHOD'} eq 'POST') {
    read (STDIN, $buffer, $ENV{'CONTENT_LENGTH'});
    @pairs = split(/&/, $buffer);
} elsif ($ENV{'REQUEST_METHOD'} eq 'GET') {
    @pairs = split(/&/, $ENV{'QUERY_STRING'});
} else {
    print "<p>$ENV{'REQUEST_METHOD'} message received</p>";
}
foreach $pair (@pairs) {
    ($name, $value) = split(/=/, $pair);
    $value =~ tr/+/ /;
    $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
    $name =~ tr/+/ /;
    $name =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
    print "<p>Field $name has the value $value</p>\n";
    $FORM{$name} = $value;
}
print "</body></html>\n";
```

# Server-Side (javascript) Scripting

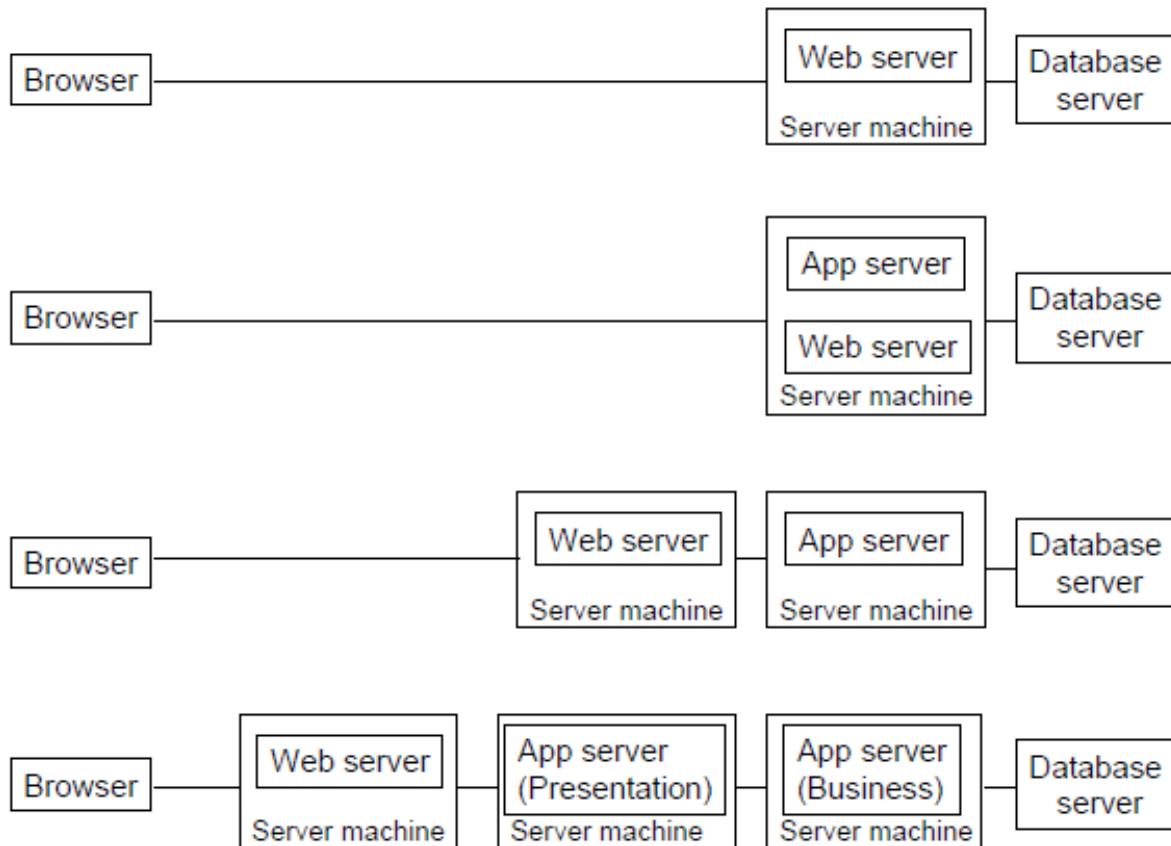
```
<html>
  <head>  <title>Server-Side JavaScript Example Author Listing</title>  </head>
  <body>
    <h1>Author List</h1>
    <server>
      if (!database.connected()){
        database.connect("ODBC","bookdb","admin","","");
      }
      if (database.connected()) {
        qs = "SELECT au_id, au_fname, au_lname FROM authors";
        results = database.cursor(qs);
        write("<table border=2 cellpadding=2 cellspacing=2>" +
              "<tr><th>ID</th><th>First Name</th><th>Last Name </th></tr>\n");
        while(results.next()) {
          write("<tr><td>" + results.au_id + "</td> + "<td>" +
                results.au_fname + "</td>" +
                "<td>" + results.au_lname + "</td></tr>\n");
        }
        results.close(); write("</table>\n");
      }
      else {
        write("<p>Database connection failed");
      }
    </server>
  </body>
</html>
```

# ASP Page



```
<%
    Dim conn, rs
    Set conn = Server.CreateObject("ADODB.Connection")
    Set rs = Server.CreateObject("ADODB.Recordset")
    conn.Open "bookdb", "sa", "password"
    Set rs = conn.Execute("select au_id, au_fname, au_lname from authors")
%>
<html>
    <head>  <title>ASP Example Author Listing</title></head>
    <body>
        <h1>Author List</h1>
        <table>
            <tr><th>ID</th><th>First Name</th><th>Last Name</th></tr>
            <% Do Until rs.EOF %>
                <tr><td><%=rs("au_id") %></td>
                <td><%=rs("au_fname") %></td>
                <td><%=rs("au_lname") %></td></tr>
                <% rs.movenext
                    Loop
                %>
            </table>
        </body>
    </html>
```

# Evolution of Web and App Servers



# Software as a Service (SaaS)

<https://od-api.oxforddictionaries.com/api/v2/entries/en-us/ubiquitous>

```
5  
6  
7 # TODO: replace with your own app_id and app_key  
8 app_id = '<my app_id>'  
9 app_key = '<my app_key>'  
10 language = 'en'  
11 word_id = 'Ace'  
12  
13 url = 'https://od-api.oxforddictionaries.com:443/api/v2/entries/' + language + '/' + word_id.lower()  
14 #url Normalized frequency  
15 urlFR = 'https://od-api.oxforddictionaries.com:443/api/v2/stats/frequency/word/' + language + '/?corpus=nmc&lemma=' + word_id.lower()  
16  
17 r = requests.get(url, headers = {'app_id': app_id, 'app_key': app_key})  
18  
19 print("code {}\n".format(r.status_code))  
20 print("text\n" + r.text)  
21 print("json\n" + json.dumps(r.json()))
```

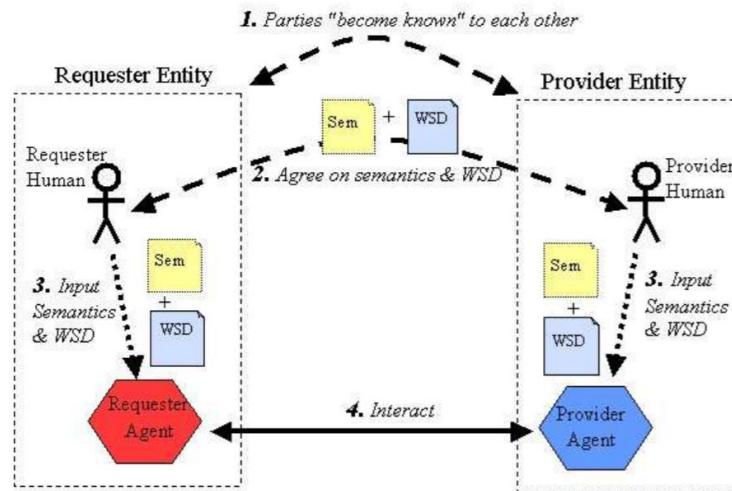
```
{  
  "definitions": [  
    "present, appearing,  
    or found everywhere"]  
}
```

**Response in  
JSON format**

API Service from Oxford Dictionary  
<https://developer.oxforddictionaries.com/>

# Web Services

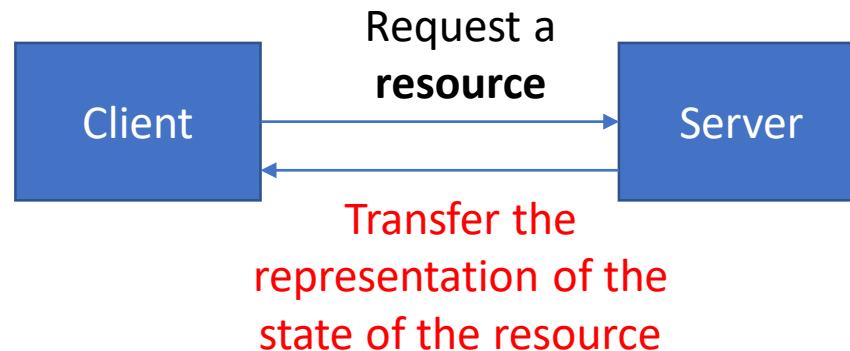
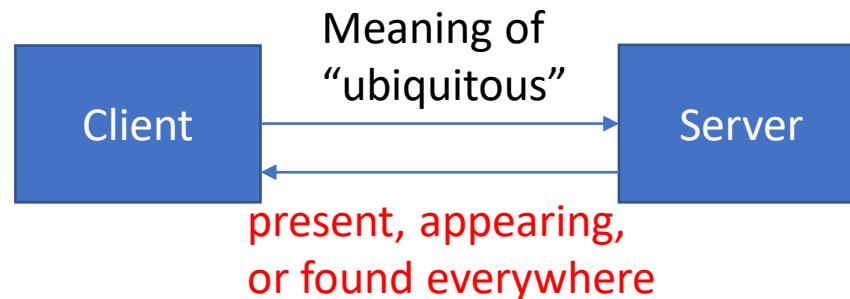
- A Web service is a software system designed to support interoperable machine-to-machine interaction over a network.



<https://www.w3.org/TR/ws-arch/wsa.pdf>

# REST API

- REST = Representational State Transfer
  - Proposed by Roy Fielding in 2000.



# Resource

- Any information that can be named is a **resource**
  - Document, image, or any other object.
- Description of the state of the resource at any timestamp is known as resource **representation**
  - Representation consists of data describing the resource.
- Resource methods are used to **transfer** the resource state representations.
  - Need not be always HTTP (GET/POST/...).

# RESTful Web Services API

- Let us retrieve an existing configuration:
  - <http://example.com/network-app/configurations/678678>
  - HTTP GET /configurations/{id}
- Similarly, we can POST, PUT, and DELETE.
  - HTTP POST /devices
  - HTTP POST /configurations
  - HTTP PUT /devices/{id}/configurations
  - HTTP DELETE /devices/{id}/configurations/{id}

---

<https://restfulapi.net/rest-api-design-tutorial-with-example/>

# HTTP

- HTTP Methods

HTTP Method	Purpose
POST	Create
GET	Retrieve
PUT	Update
DELETE	Delete

- “An *idempotent* HTTP method is an HTTP method that can be called many times without different outcomes.”
  - POST is NOT idempotent.
  - GET, PUT, DELETE are idempotent.

# HTTP Response Codes

- 2xx
  - Success
  - Example: 200 = OK, 201 = Created, 202 = Accepted (if it is a long-running task)
- 4xx
  - Client Error
  - Example: 400 = Bad Request, 404 = Not Found.
- 5xx
  - Server Error
  - Example: 500 = Internal Server Error

---

<https://restfulapi.net/http-status-codes/>

# REST in Real World

news.google.com/?hl=en-IN&gl=IN&ceid=IN:en

Google News

### Headlines

Yes Bank Rescue Plan "Bizarre", Huge Loan Spike Allowed: P Chidambaram

NDTV News · 24 minutes ago

- Yes Bank crisis: Sitharaman blames stressed loans u Chidambaram hits back
- P Chidambaram says RBI's draft for Yes Bank is bizarre; asks how nobody noticed big jump in loan book
- Yes bank, no bank

Times of India · Yesterday

The Financial Express · 28 minutes ago

More Headlines

Request URL: https://play.google.com/log?format=json&hasfast=true&authuser=0

Request Method: POST

Status Code: 200

Remote Address: 172.217.166.110:443

Referrer Policy: origin

access-control-allow-credentials: true

64 requests | 429 KB transferred

Google News

### Headlines

Yes Bank Rescue Plan "Bizarre", Huge Loan Spike Allowed: P Chidambaram

NDTV News · 24 minutes ago

- Yes Bank crisis: Sitharaman blames stressed loans u Chidambaram hits back
- P Chidambaram says RBI's draft for Yes Bank is bizarre; asks how nobody noticed big jump in loan book
- Yes bank, no bank

Times of India · Yesterday

The Financial Express · 28 minutes ago

More Headlines

Line 1, Column 1

# Designing REST API

- Identify the object model
- Create Model URIs
- Determine Representations
- Assign HTTP Methods

```
<device id="12345">
  <link rel="self" href="/devices/12345"/>
  <deviceFamily>apple-es</deviceFamily>
  <OSVersion>10.3R2.11</OSVersion>
  <platform>SRX100B</platform>
  <serialNumber>32423457</serialNumber>
  <connectionStatus>up</connectionStatus>
  <ipAddr>192.168.21.9</ipAddr>
  <name>apple-srx_200</name>
  <status>active</status>
</device>
```

# Web Services for a Banking Application



- Designing the REST API
  - Object Model
    - Customer, Account
  - Create Model URIs
    - /customers/{customerId}
    - /customers/{customerId}/accounts
    - /customers/{customerId}/accounts/{accountId}
  - Determine Representations
    - Represent all Account information as an XML/JSON
    - Represent all Customer information as XML/JSON
  - Assign HTTP Methods
    - Open Account = Create an Account Resource → HTTP POST
    - Close Account = Delete the Account → HTTP DELETE

# Implementing RESTful web services

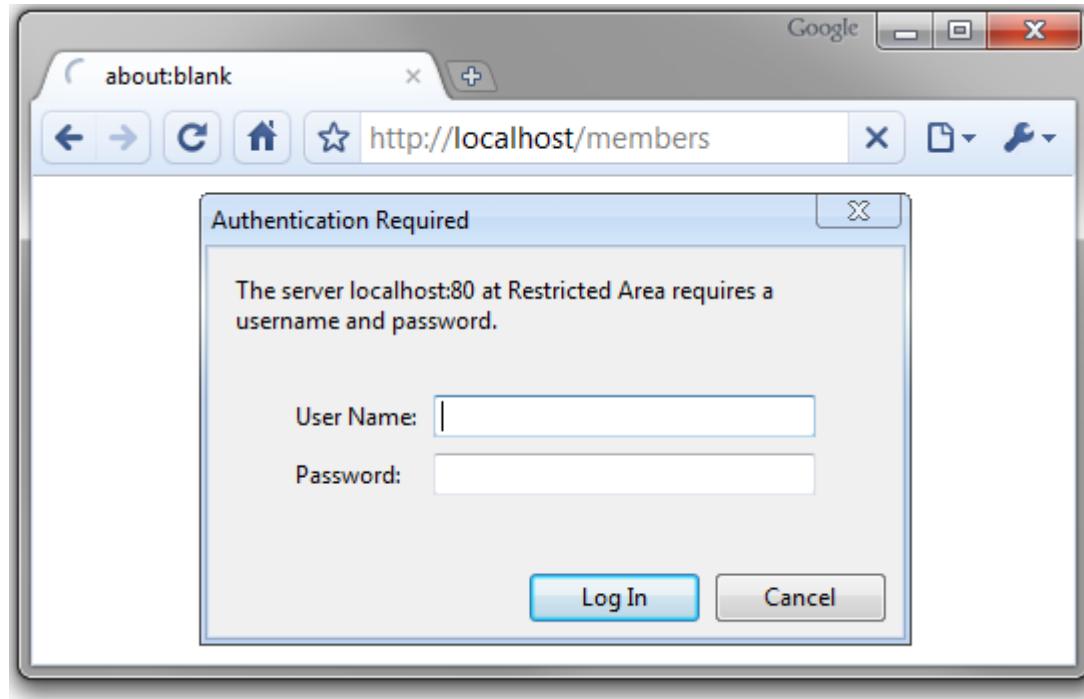
- Java API for RESTful web services (JAX-RS) [[JSR 311](#)] is specification.
- Jersey is a popular JAX-RS implementation.
- JAX-RS Annotations helps in building web services

```
..  
@Path("/configurations")  
public class ConfigurationResource  
{  
    @Path("/{id}")  
    @GET  
    public Response getConfigurationById(@PathParam("id") Integer id){  
        ...  
    }  
}
```

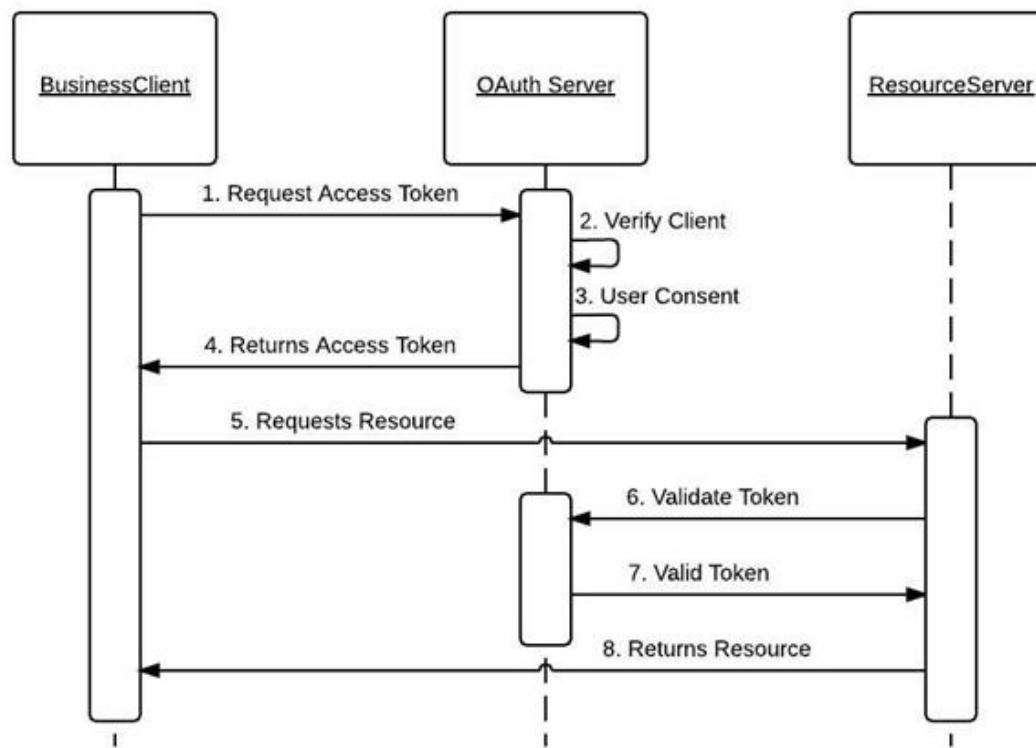
# Authentication

- Basic HTTP Authentication
  - User enters the credentials
- Query String Authentication
  - URL has the credentials
- API Keys
  - Server generated keys are used to identify the user.
- Token-based Authentication
  - oAuth method
  - Most secure form of authentication out of these four.

# Basic HTTP Authentication

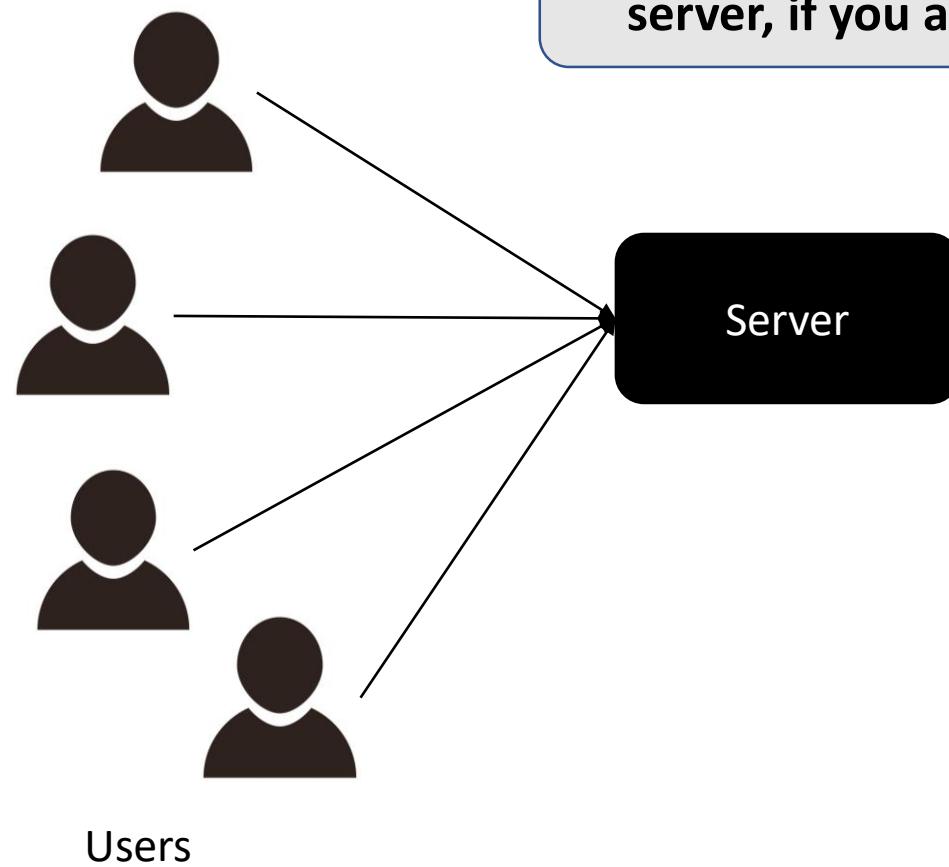


# oAuth 2.0 Architecture



[https://docs.oracle.com/cd/E82085\\_01/160027/JOS%20Implementation%20Guide/Output/oauth.htm](https://docs.oracle.com/cd/E82085_01/160027/JOS%20Implementation%20Guide/Output/oauth.htm)

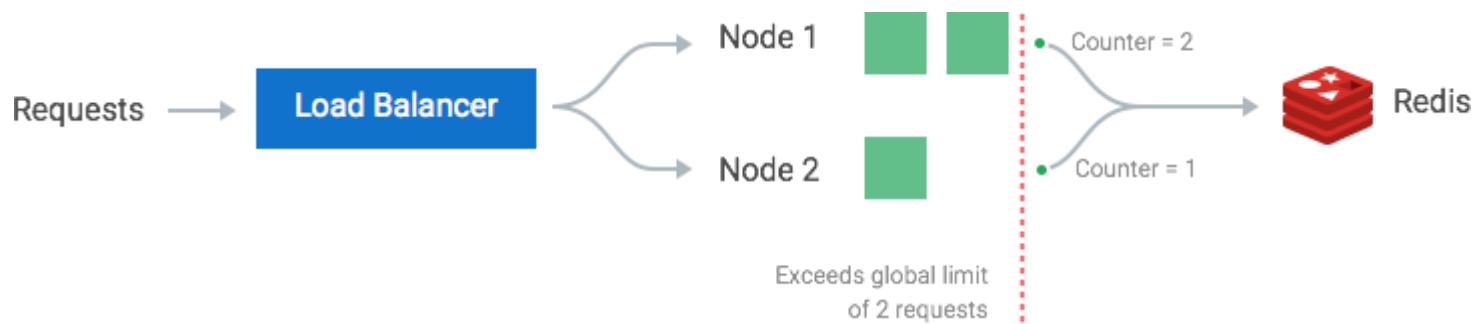
# Web Services – Rate Limiting



Can you think of a way to bring down a server, if you are one of the users?

# Rate Limiting

- A Leaky Bucket Solution
  - Queue up and service at a specific rate.
- Fixed Window Approach
  - Every request is served in a fixed time slot.
  - If the counter exceeds a threshold, the request is discarded.



<https://konghq.com/blog/how-to-design-a-scalable-rate-limiting-algorithm/>

# Putting it all Together!

The screenshot shows a browser window with the Google News homepage. The main content area displays headlines about the Yes Bank rescue plan, featuring a photo of P Chidambaram. Below the headlines, there are several news items with titles and brief descriptions from NDTV News, Times of India, and The Financial Express. To the right of the browser window, the developer tools Network tab is open, showing network requests. One request is selected, revealing details such as the Request URL (https://play.google.com/log?format=json&hasfast=true&authuser=0), Request Method (POST), Status Code (200), and Headers (including access-control-allow-credentials: true). The Network tab also shows a timeline of requests and a list of names for other requests.

# Private Cloud

- Many companies build and use their own private cloud.
  - Each private cloud is a single-tenant server or cluster of servers
  - Total control over the resources of the physical hardware layer.
  - No risk of resource or capacity contention.
  - Best suited for privacy and compliance.
  - Expensive!
- Smaller companies that cannot afford a private cloud buy infrastructure (from IaaS) on a public cloud.
- There are also corporates that believe in hybrid cloud.
  - For economies of scale.

# Public Cloud

- Storage and Computing services offered by third-party providers over the public Internet, making them available to anyone who wants to use or purchase them.
- Often pay-as-you-go service.
- Sold on-demand.
- No management and maintenance overhead.
- May have restrictions due to security concerns (say, can't open certain ports).

# Hybrid Cloud

- Combines a public cloud and a private cloud by allowing data and applications to be shared between them.
- As demand fluctuates, hybrid cloud computing gives businesses the ability to seamlessly scale their on-premises infrastructure up to the public cloud.
  - No need to make massive capital expenditures to handle short-term spikes.
  - Companies will pay only for resources they temporarily use.

# Thank You

# BDH Class Notes - Web Services

venkatesh V  
venakteshv@cmi.ac.in

March 2020

## 1 Introduction

Having discussed the advances in data storage and computation, we now turn our attention to building useful big data applications. Towards this purpose, it is insufficient to understand how to manage big data as we have learned so far using distributed computing platforms such as Hadoop HDFS, programming models such as map-reduce, and data stores such as NoSQL. The missing piece is about handling interoperability of applications and distributed systems. We leverage the advances in the area of web technologies to build services that expose application features built over heterogeneous distributed systems. In this article, we discuss the design and implementation of web services.

## 2 Interoperability of Distributed Systems

Different distributed systems present interesting challenges of interoperability. The systems may be heterogeneous by nature. The heterogeneity might be driven not only by the nature of computing systems and programming languages/scripts/models used but also the data representations used within each system. For example, let us say, we would like to use both the google maps functionality and Facebook's face recognition technology. In an ideal world, both these companies expose their features as "services" that any developer may use in building his own application. In the present era, our applications depend on large and complex heterogeneous distributed systems.

## 3 A Brief Survey of Interoperability Solutions

We may consider file passing as a simple means of communication between two distributed systems. However, this has several limitations. For instance, we need to design common file formats and semantics of serializing the data into a file. Moreover, the encryption, compression, encoding and transmission of the file are concerns that every system developer needs to solve. Hence, a middleware platform will be very useful from separations of concerns and a

reuse perspective. By separation of concerns, we mean the application developer can worry about the application details without worrying about the "big data" management details. The "big data" related features can be transferred to the middleware community.

Instead of transferring files, transferring objects would solve the impedance mismatch problem. Thus, passing objects will only be more natural for client-server communication. Some technologies use remote procedure calls. In other words, they invoke methods on a remote object. Another architectural decision passes "messages".

Object Management Group's (OMG) Common Object Broker Architecture (CORBA) is an open standards based solution proposed to meet this objective. CORBA is a standards-based, vendor neutral and language agnostic solution for interoperability. A server makes objects available for clients to use through a middleware layer named Object Request Broker (ORB). Client requests for an object (served by the server) through the ORB API. The ORB API supplies the object instance after communicating and obtaining that instance from the server often over a network. Notice that ORBs make the sharing of objects seamless and easy. Given the standard architectural specification, vendor-specific code could be provided by the specific programming language community.

Soon, vendor specific implementations were out in the market. Microsoft implemented DCOM. Sun Microsystems came up with RMI. In the early 2000's, DCOM and RMI were popular. The central idea remains the same across both DCOM and RMI. The Remote Method Invocation (RMI) model used RMI registry to advertise the services. First, the client "discovers" the service it needs. Then, it directly talks to the RMI server to "invoke" that service which is essentially a method call on an object. DCOM supports remote objects by running on a protocol called Object Remote Procedure Call (ORPC). A DCOM client calls the exposed methods of a DCOM server by acquiring a pointer to one of the server object's interfaces. In this course, we will not go any deeper into OMG, DCOM or RMI. Instead, we focus on building web services which has revolutionized the industry, of-late.

## 4 Web and its Disruptions

With the broad penetration of internet, it made sense to make method invocations over the web. This led to the idea of "web services".

Early static web was developed in 1990 at European Organization for Nuclear Research (CERN). NCSA Mosaic 1.0 was the first browser, released by the National Center for Supercomputer Applications (NCSA). So, how does it work? First, we need to write code in the Hyper Text Mark Up (HTML) language. Then, we move it to a machine which has a software module named web server running on it. Web servers wait for client requests, usually sent as Hyper Text Transfer Protocol (HTTP) requests. In return, the server sends the HTML back to the client. The client then renders the HTML on the screen.

Static web pages were extremely limiting our abilities to implement online

transactions, maintain sessions, remember data over sessions and so on. This called for a more powerful technology which could render HTML output dynamically (i.e., programmatically). Apache Httpd 1.0 web server allowed Common Gateway Interface (CGI) which enabled the dynamic web. Microsoft proposed Active Server Pages (ASP). The details of coding with CGI or ASP is beyond the scope of this course. However, I encourage you to spend few minutes looking at some sample ASP and CGI scripts.

A web server's responsibility is to receive requests for web pages and serve them. Soon, technologies evolved to do more complex server-side processing. Full-stack applications could be coded on the server end. Such servers came to be known as Application Servers (also called App Servers). The technologies in this space went through a churn from being together in a single module to an architecture where a web server still receives web requests and forwards them to another app server which in-turn may work with several other app servers to complete the task. Needless to mention that availability of cloud data stores greatly improved our ability to solve bigger and more interesting business problems.

## 5 Web Services

A “service” is a software component provided through an (often, network-accessible) endpoint. Service consumer and provider use messages to exchange invocation request and response information in the form of self-containing documents. A *Web service* is an application that exposes certain application features (services) over the web. They are used for application-to-application integration.

A Web Resource is a named object that is accessible over the web. Uniform Resource Identifiers (URI), are used to identify them. A web page is an example of a web resource. An Uniform Resource Locator (URL), say <http://www.google.com> locates the webpage. In the context of web pages, URLs and URIs are used interchangeably. This is because, in this context, a resource is identified by its location. However, unique ids such as ISBN strings of books may serve as URIs.

### 5.1 Application Interoperability with Web Services

Let us assume that we need to create an application where we need to show meanings of words. We do not need to re-implement a dictionary. Instead, we may just use an existing dictionary. Oxford Dictionary offers a web service through which we can easily access word senses. The URL <https://od-api.oxforddictionaries.com/api/v2/entries/en-us/ubiquitous> returns a Javascript Object Notation (JSON) document containing the definitions. Part of the response containing the definition is shown below.

{

```

    "definitions": [
        "present, appearing, or found everywhere"
    ]
}

```

You may use these service to build your own applications.

## 5.2 REST API

The example of Oxford Dictionary discussed above is an example of REST API. Representational State Transfer (REST) was proposed by Roy Fielding in the year 2000. The central idea is to identify resources, and then use a representation of the resource that can be transferred over the web.

Each word is seen as a web resource that is represented using JSON format where we capture all data elements describing the term such as the definitions, usage and synonyms.

We use Hypertext Transfer Protocol (HTTP) methods to transfer the response. By default, when nothing is specified, we assume the HTTP Get method is invoked when we access the URL. In this document, we use the notation HTTP GET /ubiquitous to represent the firing of URL <http://od-api.oxforddictionaries.com/api/v2/entries/en-us/ubiquitous> over the web. The server returns the JSON response as a result.

REST allows the Create/Retrieve/Update/Delete (CRUD) operations through the HTTP methods. The HTTP Get method is mapped to the retrieval action. Instead, if you intend to create a new resource, you perform HTTP Post and send the representation of the resource along with the post request. For instance, HTTP POST /ubiquitous would create a new instance of the resource on the server. HTTP Put is used to update a resource. If you wish to add a new usage of the term ubiquitous, you would create a new representation which contains the update and submit as a HTTP Put request. Similarly, you may use HTTP Delete to remove the resource from the server. Thus you could manage the states of the resource through transferring its representation.

An idempotent HTTP method is an HTTP method that can be called many times without different outcomes. Notice that Get, Put and Delete are idempotent. Irrespective of how many ever times you invoke these methods, you get the same response from the server. Invoking Post multiple times might (not necessarily, depends on application design) create multiple instances of the resource.

To understand this better, refer to the example discussed at the [restfulapi.net](https://restfulapi.net/)<sup>1</sup>. In our word sense case, ubiquitous is a singleton. However, in reality, we could have different requirements. The example at this website discusses creating several devices and configurations through REST service. HTTP POST /devices creates a new device. HTTP GET /devices/id will retrieve it.

A HTTP response code is also sent along with each response. For instance, a 500 error code means that there was some internal server error while serving the request. In fact, all 5xx errors indicate server failures. Similarly, 4xx errors

---

<sup>1</sup><https://restfulapi.net/rest-api-design-tutorial-with-example/>

indicate that the request was not well formed. For more details on HTTP response codes, refer restfulapi.net<sup>2</sup>.

## 6 Designing REST API

There are four major steps in the designing of REST APIs:

- Identify the object model
- Create Model URIs
- Determine Representations
- Assign HTTP Methods

### 6.1 Identifying Object Model

Assuming that we are designing the REST API for an ecommerce application. Say, we have several products to sell. These "Products" become a candidate for resource. Similarly, we need to identify the other objects, say users, reviews, etc. Coming with an object model is the subject of class diagram design in UML.

### 6.2 Create Model URIs

Next, we are interested in creating the model URIs. HTTP GET /product/id retrieves a product. To create a product, we do a HTTP Post /product. Remember, we submit an XML or a JSON representation of the resource along with the post request. In our case, we need to design the representation and that is our third step. Often, we include version numbers in our URL for maintenance reasons. So, you will see URLs such as <https://od-api.oxforddictionaries.com/api/v2/entries/english/ubiquitous>. Since we are concerned about resources, the URIs always contain nouns and never verbs.

### 6.3 Determine Representations

To create or update a resource, client sends the resource representation, usually (not necessarily) in XML or JSON format. For a product, the representation may contain details such as name, price, description and so on. Here is an example:

```
<product xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <name type="xs:string">N95 Mask</name>
  <price type="xs:int">100</price>
  <description type="xs:string">
    98% 3-layer filtration of pollution
```

---

<sup>2</sup><https://restfulapi.net/http-status-codes/>

```

    particles PM2.5 99% filtration of
    bacteria, tested by Nelson Laboratories
  </description>
</product>
```

For a JSON example, refer the restfulapi website<sup>3</sup>.

## 6.4 Assign HTTP Methods

The last step is to assign the http methods for the resource actions. To add products in the server, we use Post method. To update, we use Put and to retrieve, we use Get method. Since our products are uniquely identified by name, we could have a Get method such as HTTP GET /products/n95nelsonmask. This URI could map to an URL <http://mywebsite/products/n95nelsonmask>. On the other hand, if we model users as non-singleton, we will use HTTP GET /users/id to retrieve a specific user resource. An interesting fact to notice here is that the interaction between client and server is stateless. By stateless, we mean that the server need not store any information at its end to use across successive client requests. It is the client's responsibility to send all the necessary information along with each request. This design helps in keeping the server design lean (demanding less memory) and simple.

## 6.5 Implementing REST API

Java<sup>TM</sup> API for RESTful Web Services (JAX-RS) delivers API for RESTful Web Services development in Java SE and Java EE. JAX-RS is implemented in Jersey. It provides all the required libraries to implement REST API. The annotations based implementation simplifies the web services development. You are encouraged to read the Jersey documentation<sup>4</sup> to learn web services implementation with Jersey.

# 7 REST API in the Real World

It is easy to spot REST api in the real world. They have in fact become a de-facto standard for web services implementation. The developer tools in chrome browser lets you inspect the URIs and responses. Figure 1 shows an example from the google news website.

# 8 Summary

Interoperability of applications is central to the success of big data systems. Web Services play a key role in language-neutral, vendor-neutral application

---

<sup>3</sup><https://restful-api-design.readthedocs.io/en/latest/resources.html>

<sup>4</sup><https://eclipse-ee4j.github.io/jersey.github.io/documentation/latest/getting-started.html>

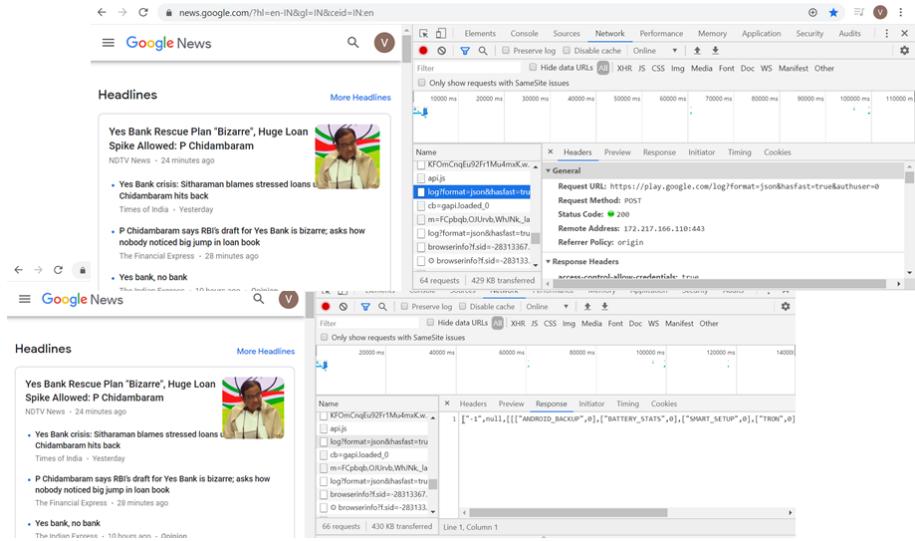


Figure 1: A snapshot of REST API usage in Google News.

interoperability. In this lecture, we discussed REST API for implementing web services. They can be easily implemented using Jersey-like libraries.

# Graph DB

**Venkatesh Vinayakarao**

venkateshv@cmi.ac.in

<http://vvtesh.co.in>

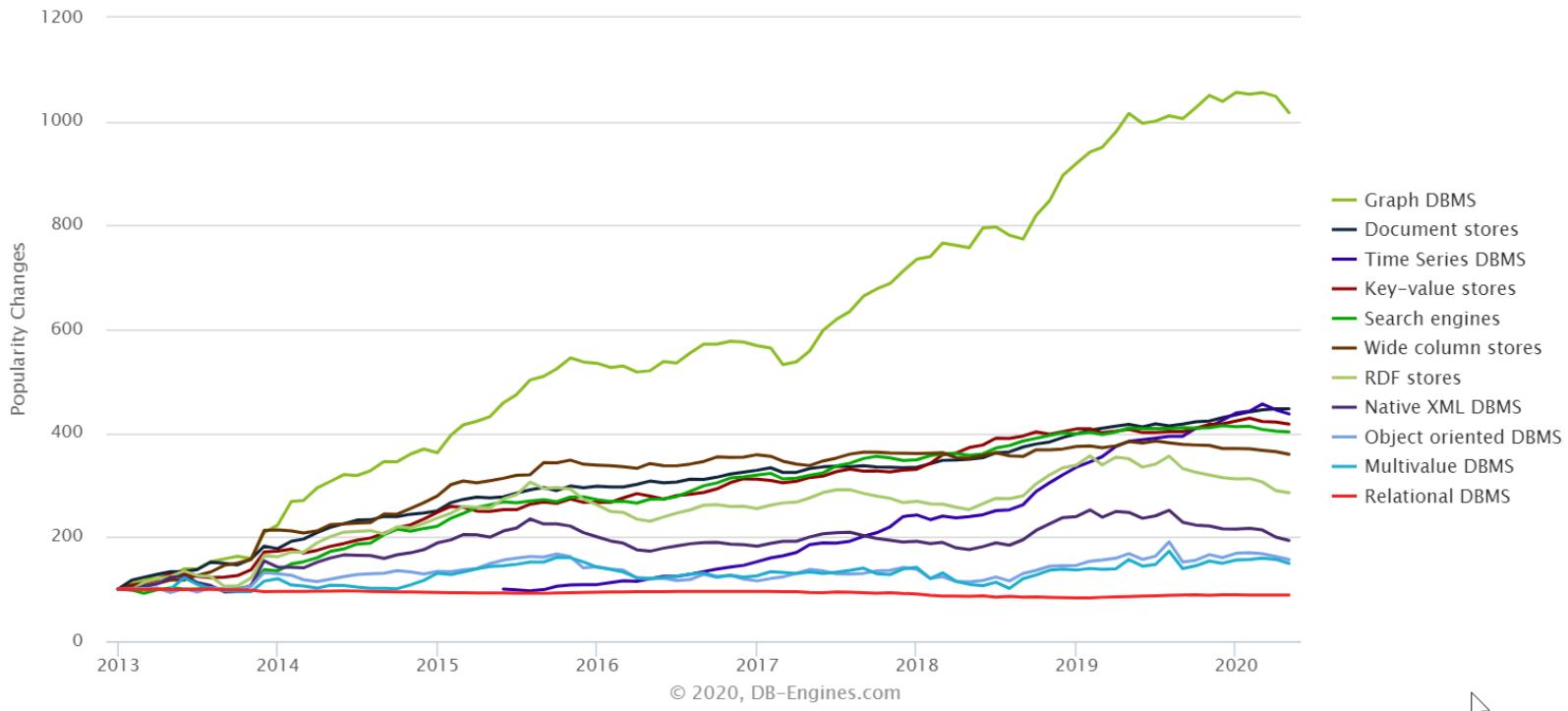
---

Chennai Mathematical Institute

---

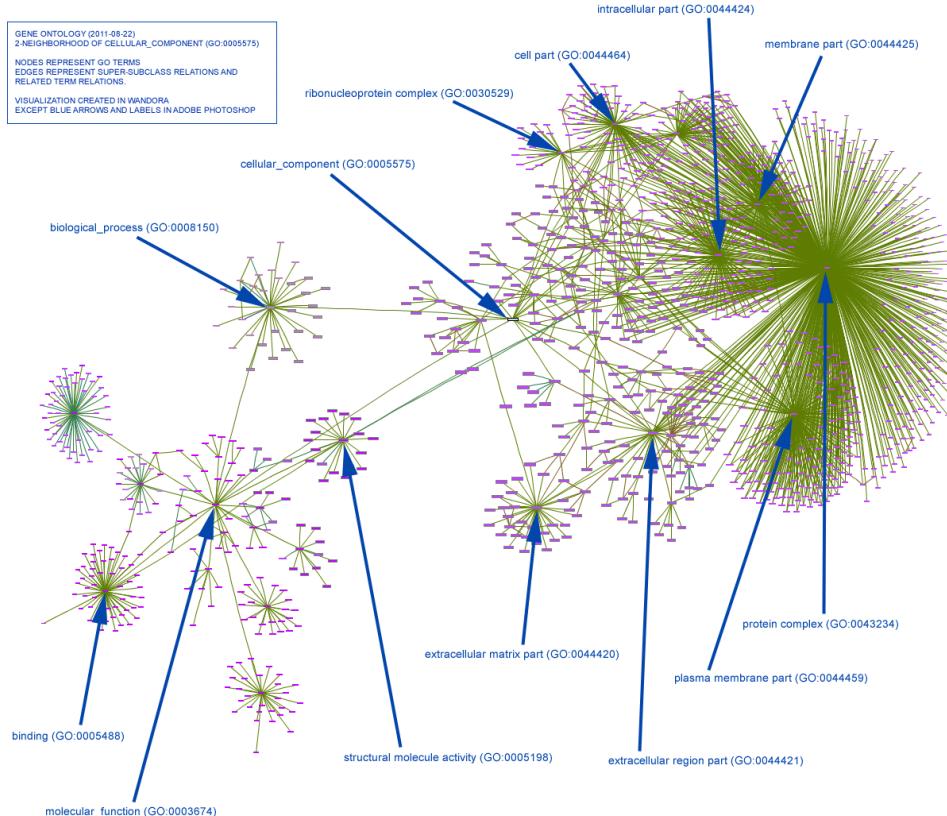
We live in a connected world! . – **Neo4j**.  
(Neo4j)-[:LOVES]-(Developers)

# Change in Popularity



Source: [https://db-engines.com/en/ranking\\_categories](https://db-engines.com/en/ranking_categories)

# Gene Ontology Model

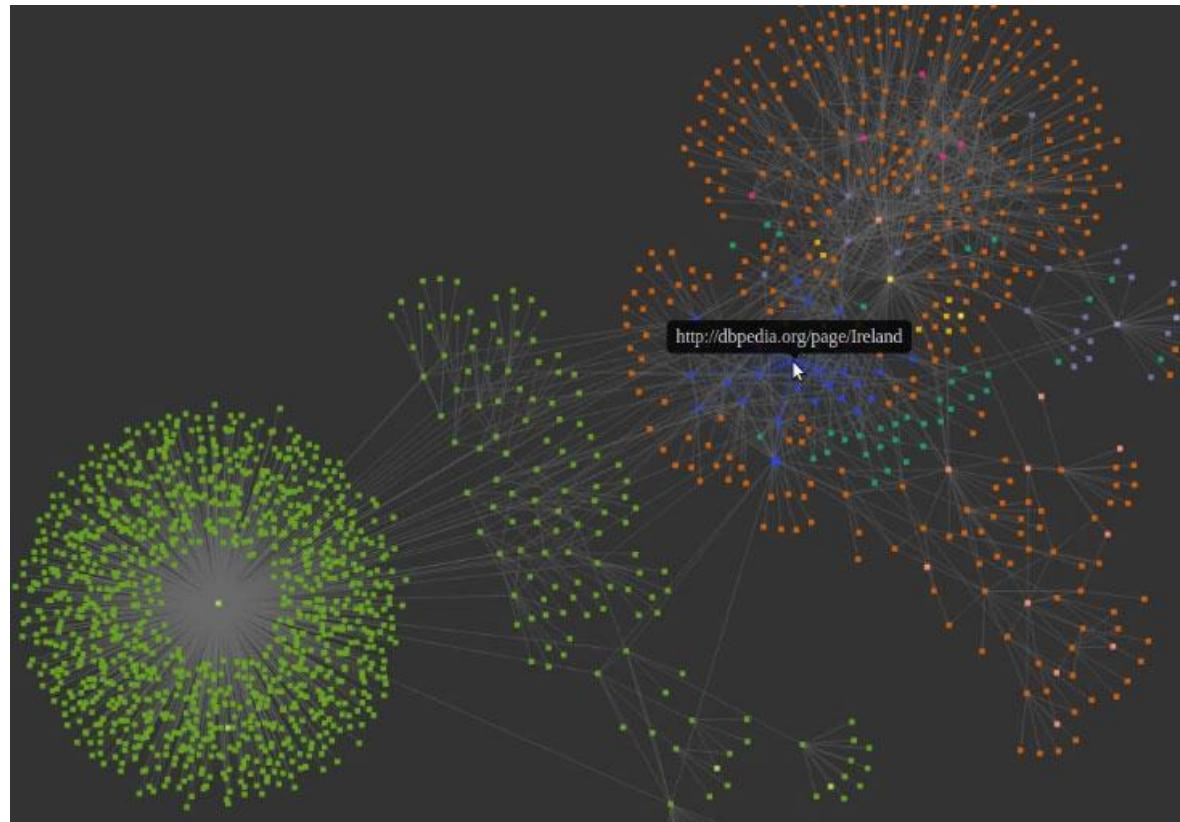


Number of topics: 177301

Number of associations:  
280198

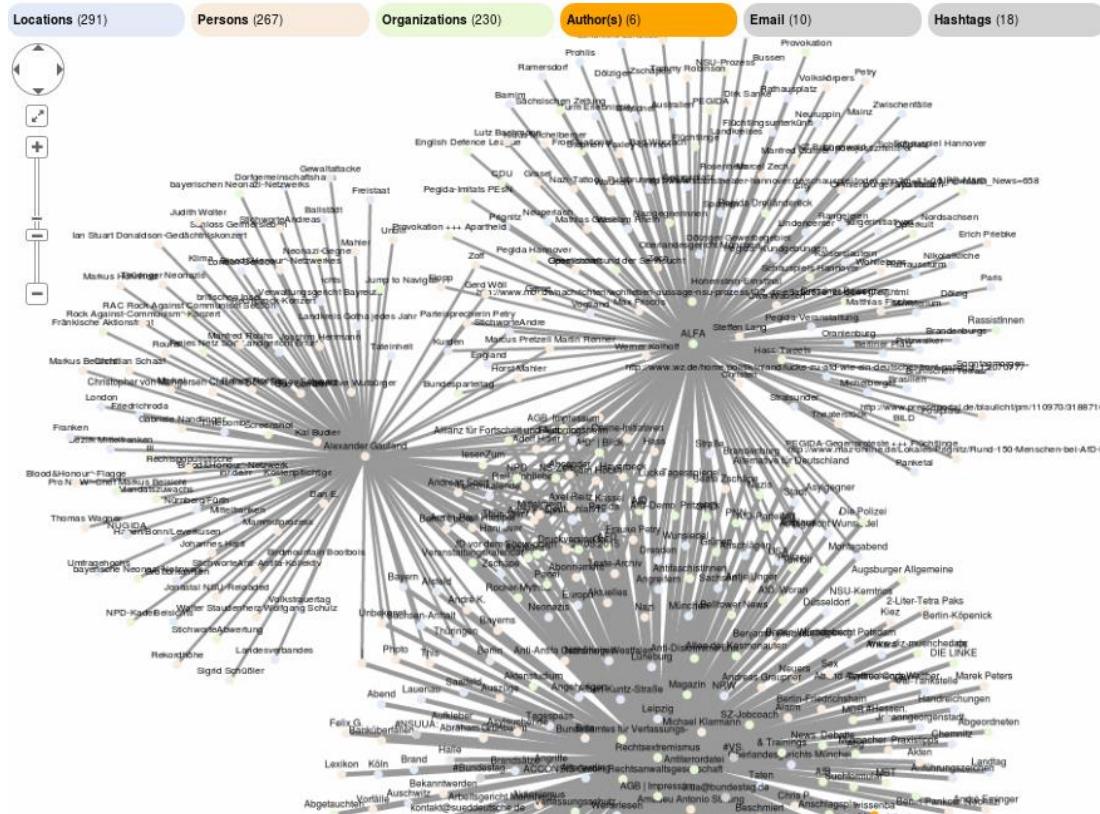
Source: [http://wandora.org/wiki/Topic\\_map\\_conversion\\_of\\_Gene\\_Ontology](http://wandora.org/wiki/Topic_map_conversion_of_Gene_Ontology)

# Knowledge Graphs



---

[Source: https://www.ibm.com/blogs/research/2016/01/from-knowledge-graphs-to-cognitive-computing/](https://www.ibm.com/blogs/research/2016/01/from-knowledge-graphs-to-cognitive-computing/)



Source: <https://www.opensemanticsearch.org/doc/search/graph>

# Graph Database

Relationships between data is equally as important  
as the data itself.

# Neo4j

- A leading graph database, with native graph storage and processing.
- Open Source
- NoSQL
- ACID compliant

Neo4j Sandbox

<https://sandbox.neo4j.com/>

Neo4j Desktop

<https://neo4j.com/download>

# Data Model

- create (p:Person {name:'Venkatesh'})-[:Teaches]->(c:Course {name:'BigData'})

# Query Language

- Cypher Query Language
  - Similar to SQL
  - Optimized for graphs
  - Used by Neo4j, SAP HANA Graph, Redis Graph, etc.

# CQL

- `create (p:Person {name:'Venkatesh'})-[:Teaches]->(c:Course {name:'BigData'})`
- Don't forget the single quotes.

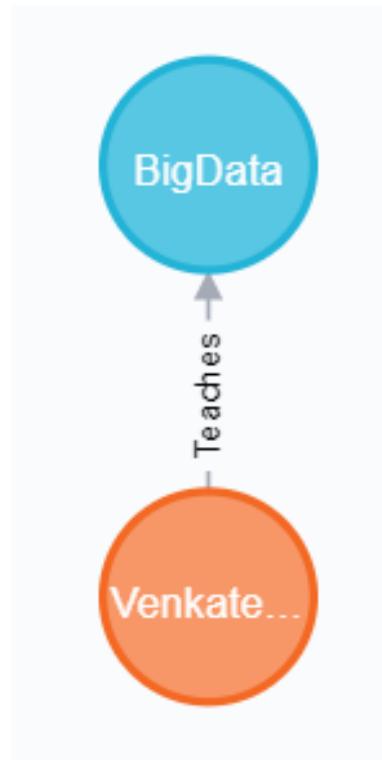
```
neo4j$ create (p:Person {name:'Venkatesh'})-[:Teaches]→(c:Course {name:'BigData'})
```

	Table
	Code

Added 2 labels, created 2 nodes, set 2 properties, created 1 relationship, completed after 30 ms.

# CQL

- Match (n) return n



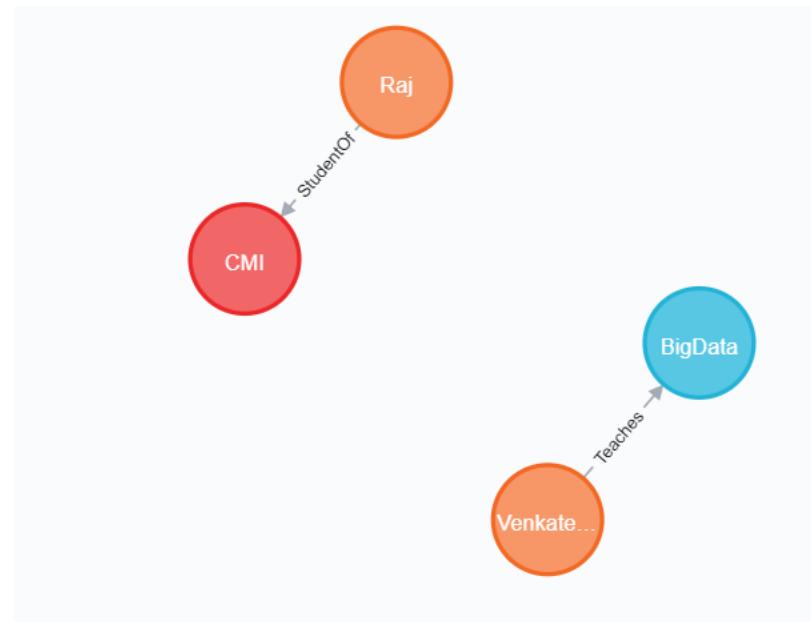
- `match(p:Person {name:'Venkatesh'}) set p.surname='Vinayakarao' return p`

```
neo4j$ match(p:Person {name:'Venkatesh'}) set p.surname='Vinayakarao' return p
```

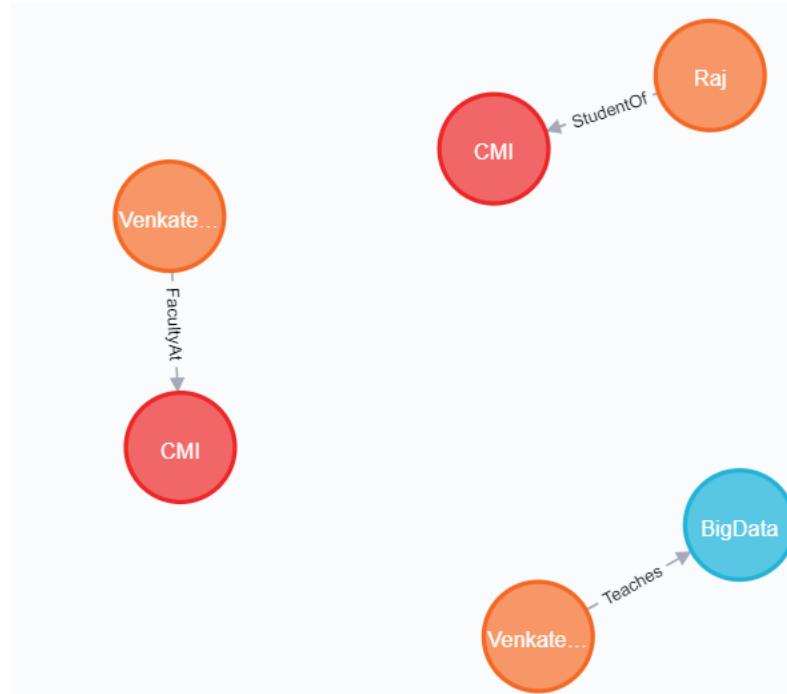
The screenshot shows the Neo4j browser interface. On the left, there is a vertical toolbar with four items: 'Graph' (selected), 'Table' (highlighted in dark grey), 'Text', and 'Code'. The main area displays the results of the executed Cypher query. The results are shown in a table with one row and two columns. The first column contains the variable 'p', and the second column contains the JSON object representing the node. The JSON object is:

```
{  
  "name": "Venkatesh",  
  "surname": "Vinayakarao"  
}
```

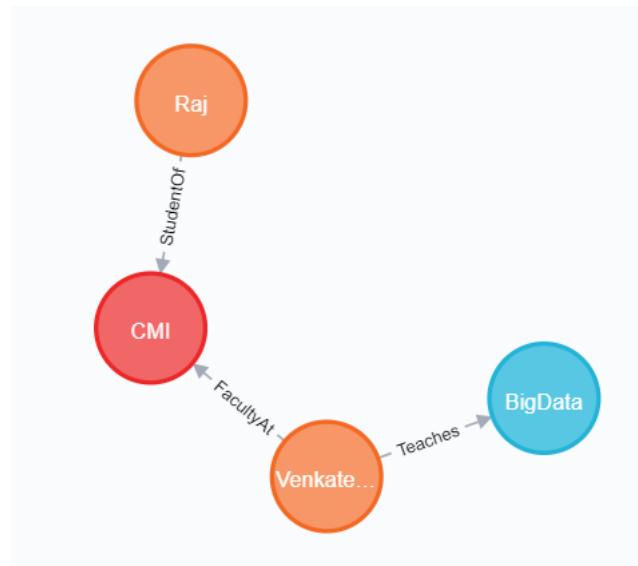
- Create (p:Person {name:'Raj'})-[:StudentOf]->(o:Org {name:'CMI'})
- Match (n) return n



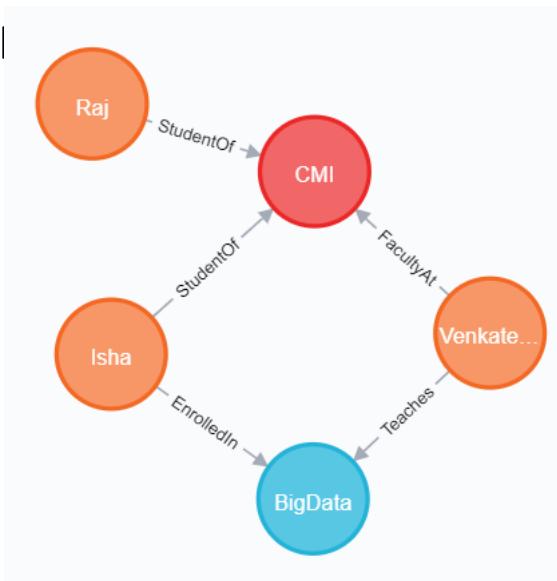
- create (p:Person {name:'Venkatesh'})-[:FacultyAt]->(o:Org {name:'CMI'})
- Match (n) return n



- MATCH (p:Person) where ID(p)=4
  - DELETE p
  - MATCH (o:Org) where ID(o)=5
  - DELETE o
- 
- MATCH (a:Person),(b:Org)
  - WHERE a.name = 'Venkatesh' AND b.name = 'CMI'
  - CREATE (a)-[:FacultyAt]->(b)



- MATCH (a:Person),(b:Course)
- WHERE a.name = 'Isha' and b.name = 'BigData'
- CREATE (a)-[:StudentOf]->(b)
  
- MATCH (a:Person)-[o:StudentOf] ID(o)=4
- DELETE o
  
- MATCH (a:Person),(b:Course)
- WHERE a.name = 'Isha' and b.name = 'BigData'
- CREATE (a)-[:EnrolledIn]->(b)



Thank You

# HIVE & WEBSERVICE TUTORIAL

---

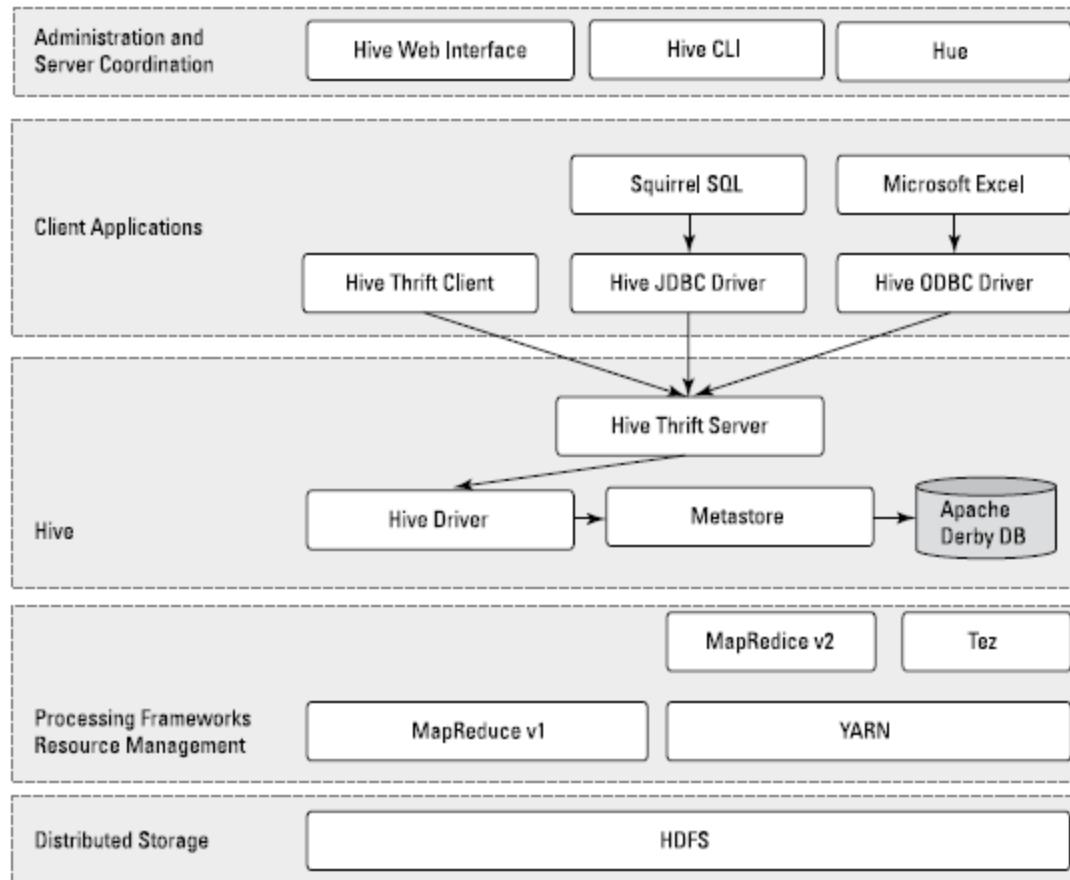
Hands-on Session

by Suchitra Jayaprakash  
[suchitra@cmi.ac.in](mailto:suchitra@cmi.ac.in)

# Apache HIVE

- HIVE hides the complexity of MapReduce. It provides SQL type script to perform MapReduce task.
- HIVE uses SQL dialect known as HIVE QUERY LANGUAGE (HiveQL).
- HIVE is data warehouse for managing and processing structured data.
- Hive supports "**READ Many WRITE Once**" pattern. Hive is "**Schema on READ only**".
- Apache Hive was created at Facebook by a team of engineers led by Jeff Hammerbacher.

# Apache Hive Architecture



(source: Hadoop for Dummies)

# Run HIVE

- **Start Cloudera server**

```
docker run --hostname=quickstart.cloudera --privileged=true -t -i --  
publish-all=true -p 8888:8888 -p 8080:80 -p 50070:50070 -p 8088:8088 -p  
50075:50075 -p 8032:8032 -p 8042:8042 -p 19888:19888 -p 10000:10000  
cloudera/quickstart /usr/bin/docker-quickstart
```

- **To get HIVE command prompt**
- Type `hive` and press enter to get hive command line interface.

# HIVE CLI

- Create Database Statement:

**CREATE SCHEMA <database name>;**

It creates database in hive. Database is collection of table.

**SHOW DATABASES;**

It displays the list of databases in hive instance.

- Create Table Statement:

CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db\_name.] table\_name  
[(col\_name data\_type , ...)]  
[COMMENT table\_comment]  
[ROW FORMAT row\_format]  
[STORED AS file\_format]

```
Logging initialized using configuration in fi
roperties
WARNING: Hive CLI is deprecated and migration
hive>
> CREATE SCHEMA user_db;
OK
Time taken: 3.927 seconds
hive> show databases;
OK
default
user_db
Time taken: 1.808 seconds, Fetched: 2 row(s)
hive>
```

# HiveQL

- Load OnlineStore dataset into HDFS using HIVE and perform few aggregation operations:

## Step 1 – Create Table

```
CREATE TABLE IF NOT EXISTS Online_Retail ( InvoiceNo STRING, StockCode STRING, Description  
STRING,Quantity INT, InvoiceDate TIMESTAMP, UnitPrice double ,CustomerID INT,Country STRING )  
COMMENT 'Online Retail Data Set'  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\n'  
STORED AS TEXTFILE;
```

```
CREATE TABLE IF NOT EXISTS tmp(InvoiceNo STRING, StockCode STRING, Description STRING,  
Quantity INT, InvoiceDate STRING,UnitPrice double ,CustomerID STRING,Country STRING)
```

```
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\n'  
STORED AS TEXTFILE;
```

```
hive> CREATE TABLE IF NOT EXISTS Online_Retail < InvoiceNo STRING, StockCode STR  
ING, Description STRING,  
> Quantity INT, InvoiceDate TIMESTAMP,UnitPrice double ,CustomerID INT,Count  
ry STRING >  
> COMMENT 'Online Retail Data Set'  
> ROW FORMAT DELIMITED  
> FIELDS TERMINATED BY ','  
> LINES TERMINATED BY '\n'  
> STORED AS TEXTFILE;  
OK  
Time taken: 2.193 seconds  
hive> CREATE TABLE IF NOT EXISTS tmp<InvoiceNo STRING, StockCode STRING, Descrip  
tion STRING.  
> Quantity INT, InvoiceDate STRING,UnitPrice double ,CustomerID STRING,Count  
ry STRING>  
> ROW FORMAT DELIMITED  
> FIELDS TERMINATED BY ','  
> LINES TERMINATED BY '\n'  
> STORED AS TEXTFILE;  
OK  
Time taken: 0.417 seconds  
hive>
```

# HiveQL

## Step 2 – Load Data

- Copy Text file to docker container

```
docker cp E:/MSc_Datascience/BigDataHadoop/Slides/hive/Online_Retail.csv  
<containerid>:/tmp/Online_Retail.csv
```

- Load Hive tables

```
LOAD DATA LOCAL INPATH '/tmp/Online_Retail.csv'  
OVERWRITE INTO TABLE tmp;
```

```
INSERT INTO TABLE Online_Retail  
SELECT InvoiceNo, StockCode, Description, Quantity,  
from_unixtime(unix_timestamp(InvoiceDate, 'dd-MM-yyyy HH:mm')),  
UnitPrice,CustomerID,Country  
FROM tmp;
```

# HiveQL

```
Time taken: 8.551 seconds
hive> LOAD DATA LOCAL INPATH '/tmp/Online_Retail.csv'
    > OVERWRITE INTO TABLE tmp;
Loading data to table default.tmp
Table default.tmp stats: [numFiles=1, numRows=0, totalSize=46123538, rawDataSize
=0]
OK
Time taken: 8.551 seconds
hive>

hive> INSERT INTO TABLE Online_Retail
    > SELECT InvoiceNo, StockCode, Description, Quantity,
    > from_unixtime(unix_timestamp(InvoiceDate, 'dd-MM-yyyy HH:mm')), 
    > UnitPrice,CustomerID,Country
    > FROM tmp;
Query ID = root_20200314175050_d13e5e65-f5bd-4c27-a85d-9fc45fe129a9
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1584207403190_0001, Tracking URL = http://quickstart.cloudera
:8088/proxy/application_1584207403190_0001/
Kill Command = /usr/lib/hadoop/bin/hadoop job -kill job_1584207403190_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2020-03-14 18:00:16,234 Stage-1 map = 0%, reduce = 0%
2020-03-14 18:01:17,174 Stage-1 map = 0%, reduce = 0%, Cumulative CPU 15.73 sec
2020-03-14 18:02:17,926 Stage-1 map = 0%, reduce = 0%, Cumulative CPU 53.73 sec
2020-03-14 18:02:24,806 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 58.01 s
ec
MapReduce Total cumulative CPU time: 58 seconds 10 msec
Ended Job = job_1584207403190_0001
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://quickstart.cloudera:8020/user/hive/warehouse/online_retail
/.hive-staging_hive_2020-03-14_17-58-35_519_7469576875740326583-1/-ext-10000
Loading data to table default.online_retail
Table default.online_retail stats: [numFiles=1, numRows=541909, totalSize=474865
22, rawDataSize=46944613]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 58.01 sec   HDFS Read: 46128029 HDFS Wri
te: 47486609 SUCCESS
Total MapReduce CPU Time Spent: 58 seconds 10 msec
OK
Time taken: 240.053 seconds
hive>
```

# HiveQL

## Step 3 – Select operation

**SELECT \* from Online\_Retail LIMIT 5;**

```
hive> SELECT * from Online_Retail LIMIT 5;
OK
536365 85123A WHITE HANGING HEART T-LIGHT HOLDER      6      2010-12-01 08:26
:00     2.55    17850 United Kingdom
536365 71053  WHITE METAL LANTERN       6      2010-12-01 08:26:00   3.39  1
7850     United Kingdom
536365 84406B CREAM CUPID HEARTS COAT HANGER     8      2010-12-01 08:26:00   2
.75     17850 United Kingdom
536365 84029G KNITTED UNION FLAG HOT WATER BOTTLE  6      2010-12-01 08:26
:00     3.39    17850 United Kingdom
536365 84029E RED WOOLLY HOTTIE WHITE HEART.  6      2010-12-01 08:26:00   3
.39     17850 United Kingdom
Time taken: 0.355 seconds. Fetched: 5 row(s)
hive> █
```

**SELECT InvoiceDate from Online\_Retail LIMIT 5;**

**SELECT Country,count(\*) FROM Online\_Retail GROUP BY Country;**

**SELECT \* FROM Online\_Retail WHERE UnitPrice>1000 AND Country = 'United Kingdom';**

## Drop table

**DROP TABLE IF EXISTS tmp;**

# Java Client - Hive JDBC Example

- Create Java project using IDE like netbeans.
- Create a java client to execute hive queries.
- Connect to “jdbc:hive2://host:port/dbname” using Hive jdbc driver.
- Submit SQL query by creating a Statement object and using its executeQuery() method.
- Process the result set.

# JAVA Client : Code

```
package hiveclient;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class HiveClient {

    public static void main(String[] args) {

        System.out.println("Getting Data from Hadoop using Hive driver");

        try {
            String driverName = "org.apache.hive.jdbc.HiveDriver";
            Class.forName(driverName);
        } catch (ClassNotFoundException e) {

            e.printStackTrace();
            System.exit(1);
        }
    }

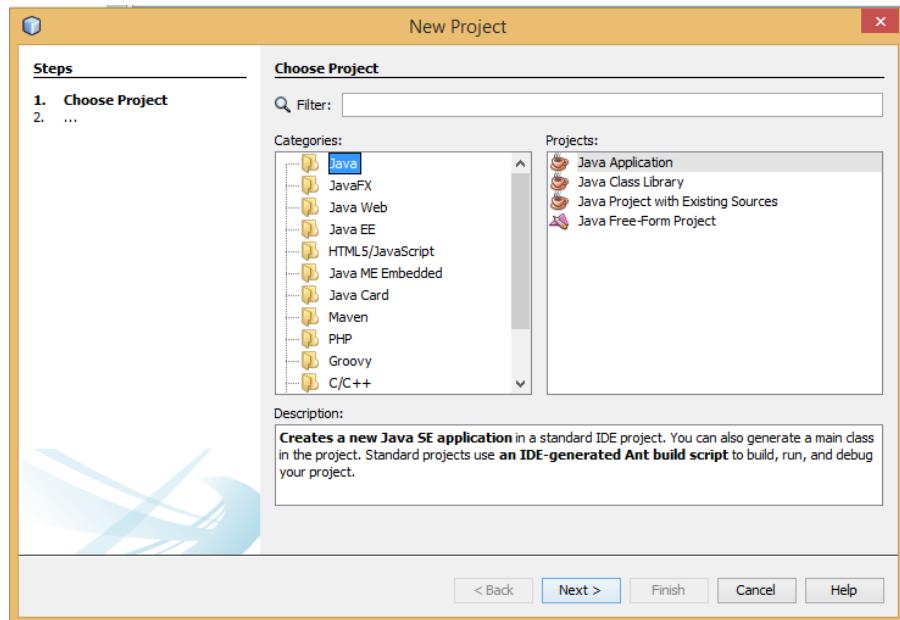
try {
    Connection con = DriverManager.getConnection("jdbc:hive2://192.168.99.100:10000/", "cloudera", "cloudera");

    Statement stmt = con.createStatement();
    String sql = "SELECT * FROM Online_Retail WHERE UnitPrice>1000 AND Country = 'United Kingdom'";
    System.out.println("Running: " + sql);
    ResultSet res = stmt.executeQuery(sql);
    while (res.next()) {
        System.out.println(res.getString(1) + "," + res.getString(2) + "," + res.getString(3) + "," + res.getString(4) + res.getString(5));
    }
    res.close();

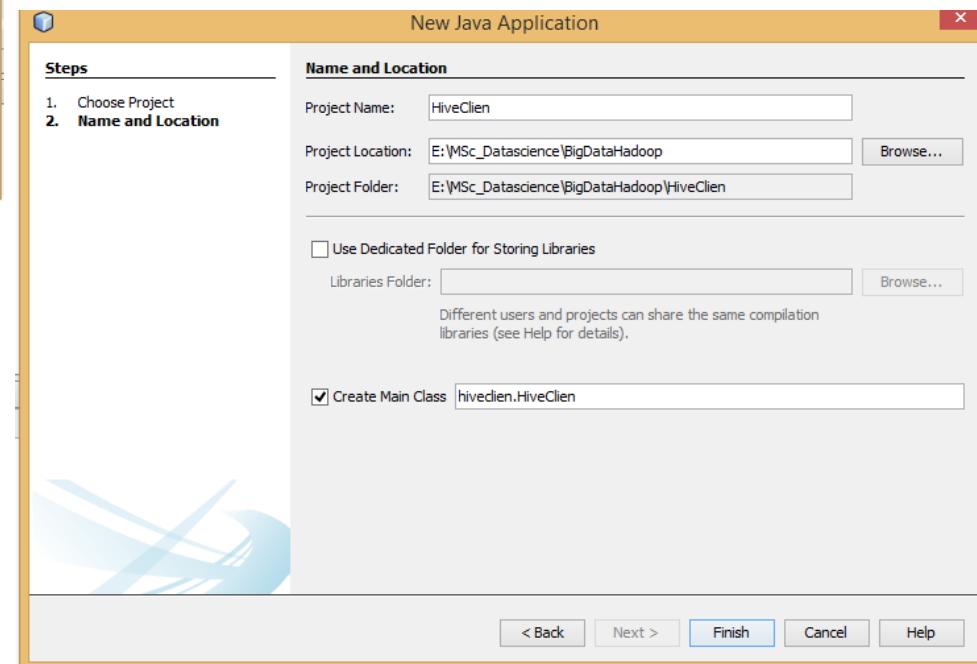
    System.out.println("Query execution complete");
} catch (SQLException e) {

    e.printStackTrace();
}
```

# JAVA Client : Set up



Create Java project HiveClient  
using IDE like eclipse



# JAVA Client : library files

- Copy HIVE Jar from Cloudera instance

1. Create folder in docker

```
mkdir /tmp/hivejar/
```

2. Create folder in docker

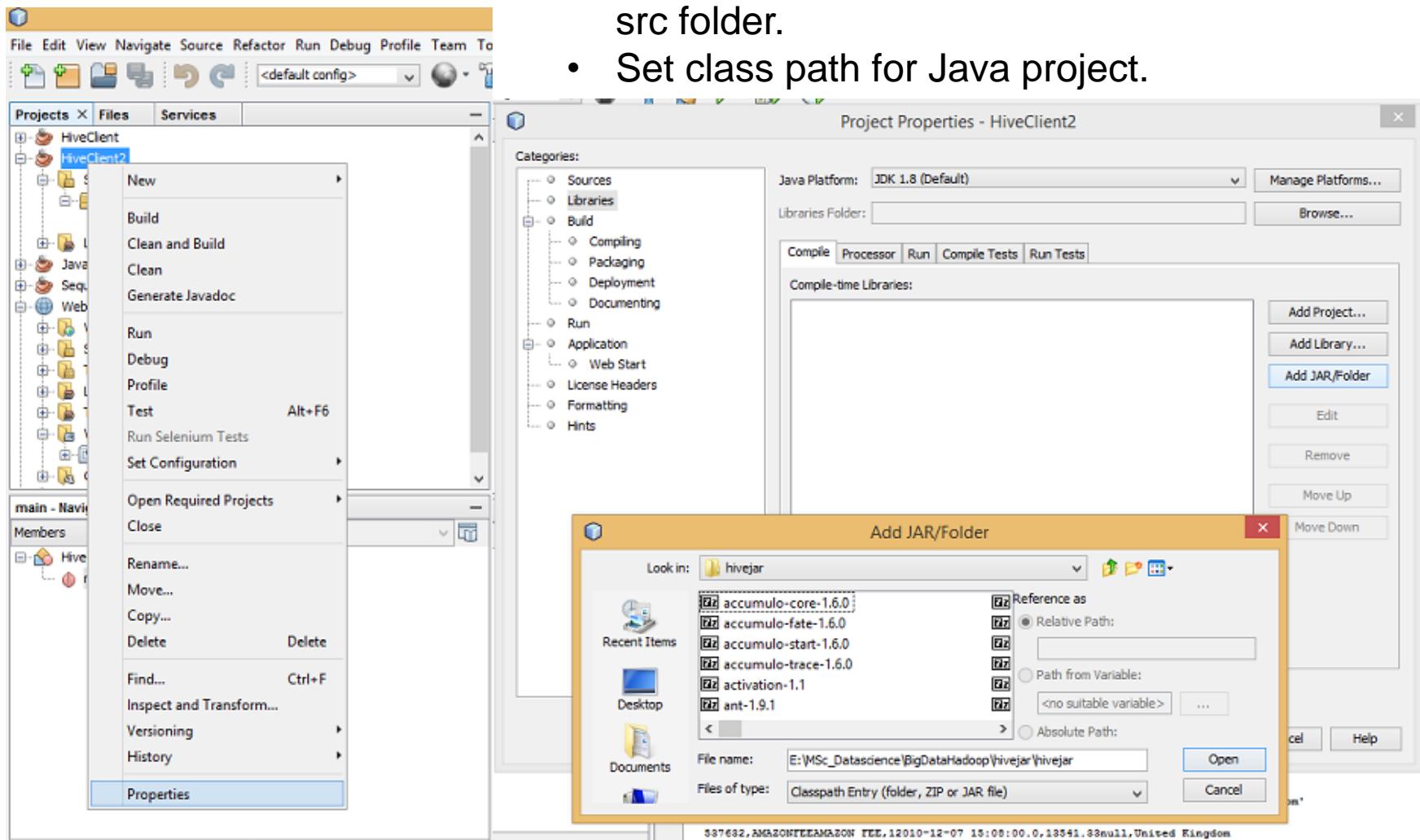
```
cp *.jar /tmp/hivejar/
```

3. Copy folder from docker to local drive

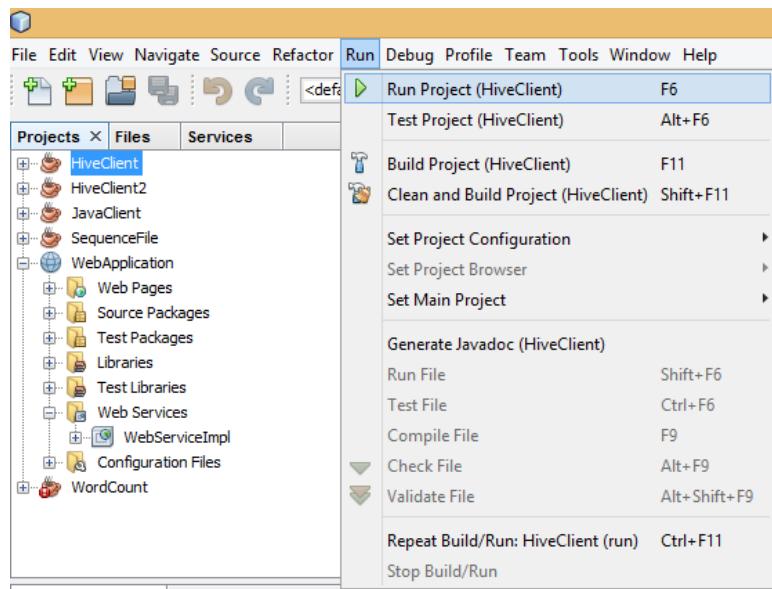
```
docker cp 26ffefdcf2f3:/tmp/hivejar  
E:/MSc_Datascience/BigDataHadoop/hivejar
```

# JAVA Client : Set up

- Copy the HiveClient.java class file into the src folder.
- Set class path for Java project.



# JAVA Client : Execute



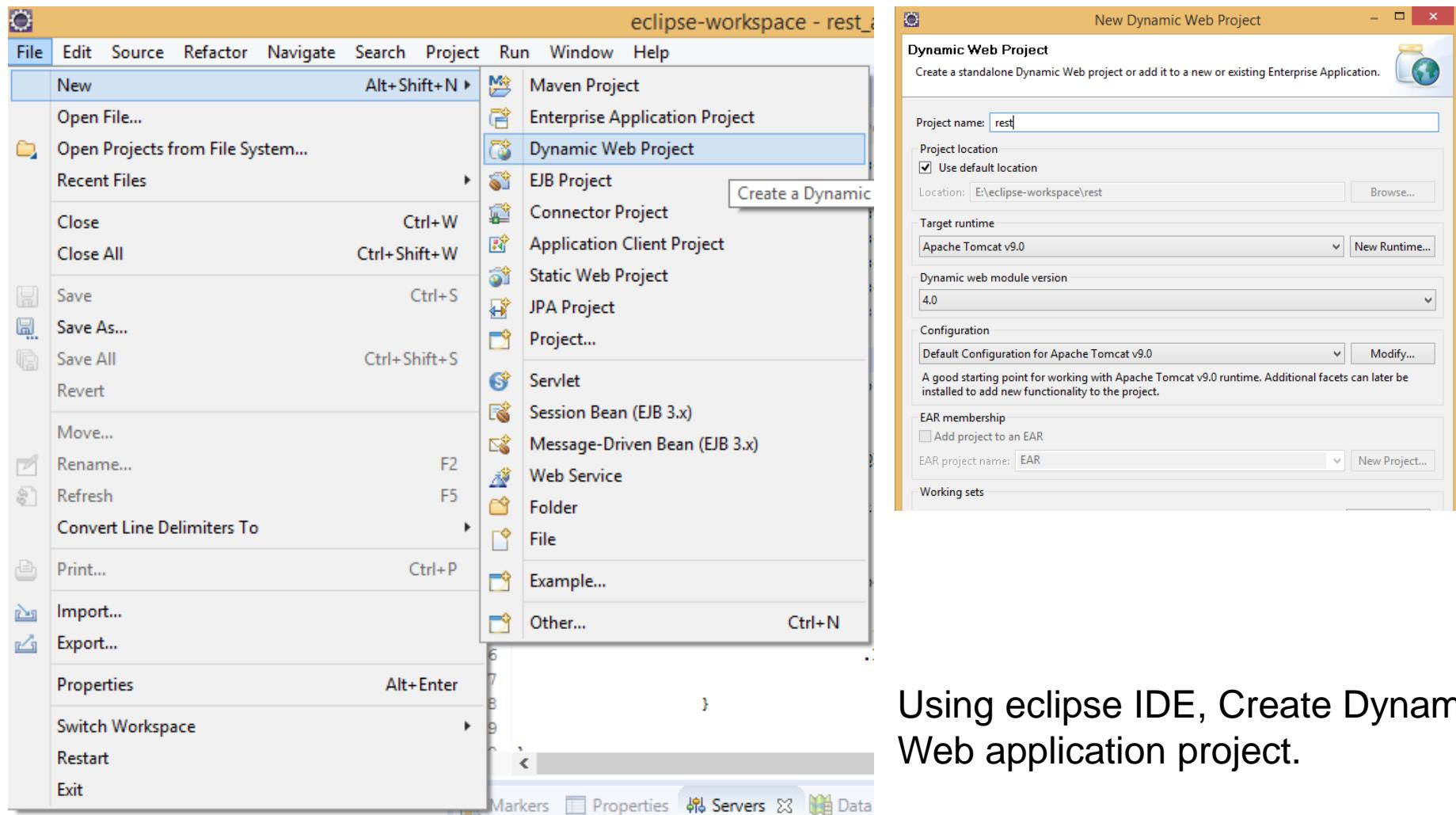
- Click Run Project
- Output would be displayed in the output window.

A screenshot of the Eclipse IDE showing the 'Output' view. The tab bar at the top of the view has three tabs: 'Output - HiveClient (run)' (which is active and highlighted in blue), 'HTTP Server Monitor', and 'Java Call Hierarchy'. The main area of the 'Output' view displays the results of the project execution. The output starts with two red error messages from SLF4J: 'SLF4J: Defaulting to no-operation (NOP) logger implementation' and 'SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.'. Following these errors, there is a series of log entries in black text, all starting with 'Running:'. These log entries list various database queries and their results, such as 'SELECT \* FROM Online\_Retail WHERE UnitPrice>1000 AND Country = 'United Kingdom'', followed by a list of product IDs and their details like 'C537630,AMAZONFEEAMAZON FEE,-12010-12-07 15:04:00.0,13541.33null,United Kingdom', 'C537632,AMAZONFEEAMAZON FEE,12010-12-07 15:08:00.0,13541.33null,United Kingdom', etc. The log continues with more entries, including some with 'MManual' values.

# REST API using JAX RS

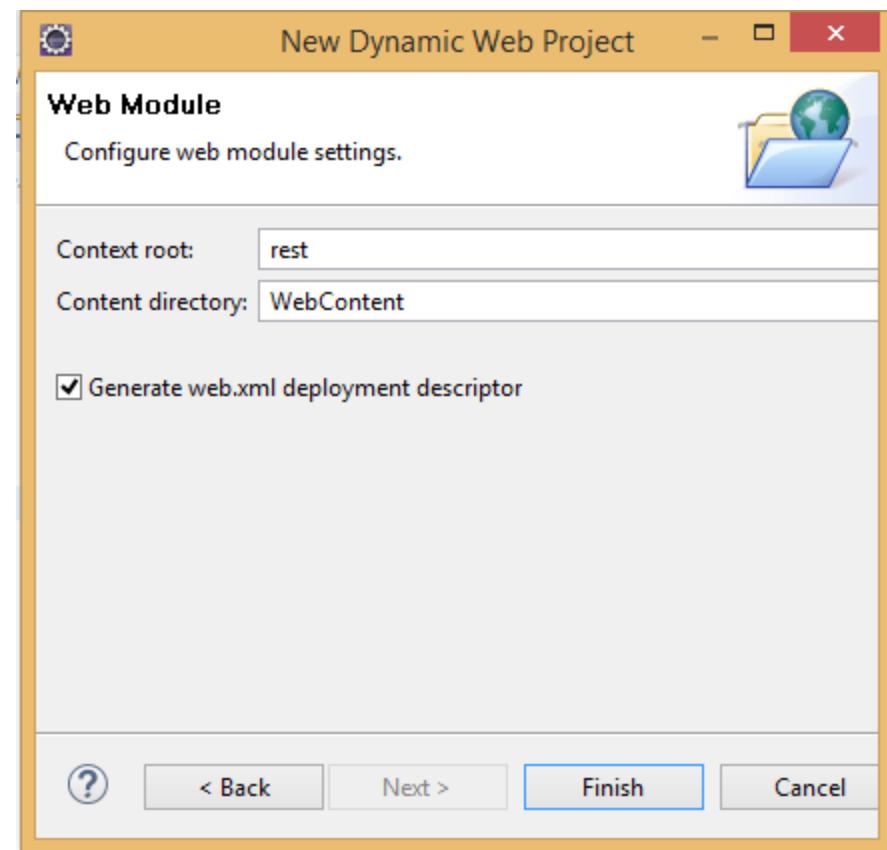
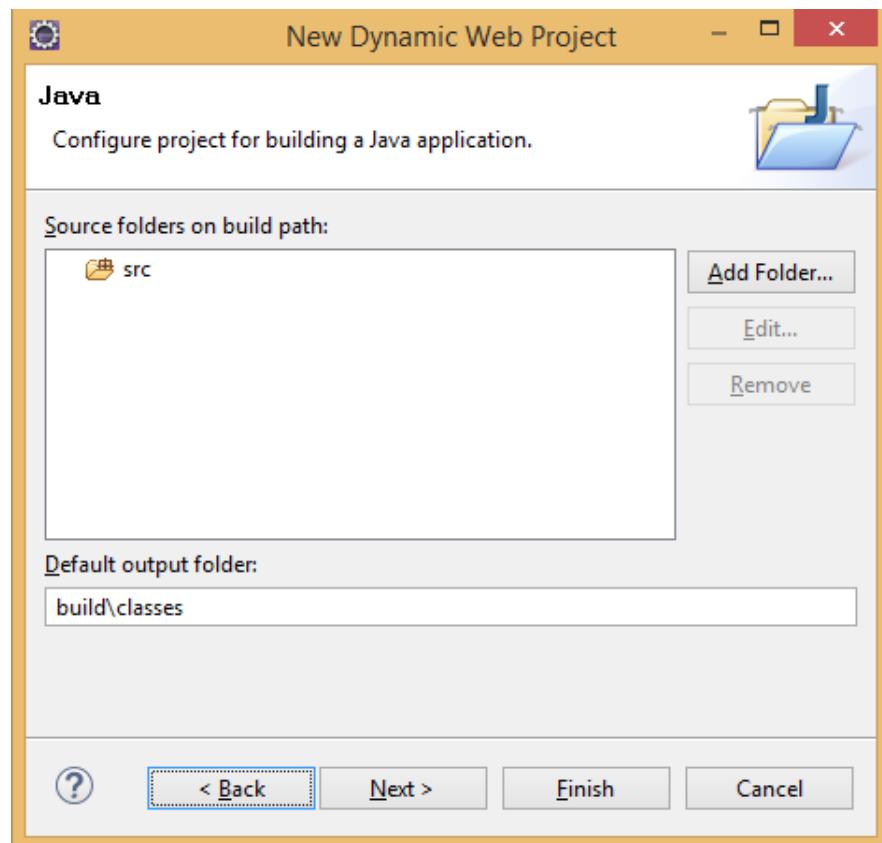
- Download eclipse IDE.
- Download jaxrs-ri JAR 2.27 with all dependencies.
- Create a new dynamic web project in Eclipse.
- Copy jaxrs jars and hive jar to WEB-INF\lib directory of the dynamic web application.
- Create three java class for Rest API

# Create dynamic web project



Using eclipse IDE, Create Dynamic Web application project.

# Create dynamic web project



Select Generate web.xml & click finish

# REST API – Java Classes

- Message.java

- Data class to hold each column value.
- Contains getter setter methods for all column.
- @XmlRootElement annotation is required for Json response conversion.

```
1 package rest_api;
2 import javax.xml.bind.annotation.XmlRootElement
3 @XmlRootElement
4 public class Message {
5
6     private String invoiceNo;
7
8     private String stockCode;
9
10    private String description;
11
12    private String quantity;
13
14    private String invoiceDate;
15
16    private String unitPrice;
17
18    private String customerId;
19    private String country;
20
21
22    public String getInvoiceNo() {
23        return invoiceNo;
24    }
25
26    public void setInvoiceNo(String invoiceNo) {
27        this.invoiceNo = invoiceNo;
28    }
29
30    public String getStockCode() {
31        return stockCode;
32    }
33
34    public void setStockCode(String stockCode) {
35        this.stockCode = stockCode;
36    }
37
38    public String getDescription() {
39        return description;
40    }
41
42    public void setDescription(String description) {
43        this.description = description;
44    }
45
46    public String getUnitPrice() {
47        return unitPrice;
48    }
49
50    public void setUnitPrice(String unitPrice) {
51        this.unitPrice = unitPrice;
52    }
53
54    public String getCustomerId() {
55        return customerId;
56    }
57
58    public void setCustomerId(String customerId) {
59        this.customerId = customerId;
60    }
61
62    public String getCountry() {
63        return country;
64    }
65
66    public void setCountry(String country) {
67        this.country = country;
68    }
69}
```

# REST API – Java Classes

- HiveDataService.java
  - It contains the Implementation of service ( In our case hive jdbc call)
  - @Path annotation to provide the context path.
  - Following annotation is used:

URI	HTTP Method	Response Type
@Path("/HiveData/fetchData")	@GET	@Produces(MediaType.APPLICATION_JSON)

```

try {
    Connection con = DriverManager.getConnection("jdbc:hive2://192.168.1.10:10000");
    Statement stmt = con.createStatement();
    String sql = "SELECT * FROM Online_Retail WHERE UnitPrice>1000 AND StockCode='223270' ";
    System.out.println("Running: " + sql);
    ResultSet res = stmt.executeQuery(sql);
    while (res.next()) {
        Message m = new Message();
        m.setInvoiceNo(res.getString(1));
        m.setStockCode(res.getString(2));
        m.setDescription(res.getString(3));
        m.setQuantity(res.getString(4));
        m.setInvoiceDate(res.getString(5));
        m.setUnitPrice(res.getString(6));
        m.setCustomerId(res.getString(7));
        m.setCountry(res.getString(8));
    }
    res.close();
}
System.out.println("Query execution complete");
} catch (SQLException e) {
    e.printStackTrace();
}

return Response
    .status(Response.Status.OK)
    .entity(messages)
    .build();

```

- Setting value to data object
- Converting data into Json response

# REST API – Java Classes

- HiveDataApplication.java

- This class extends Jersey JAX-RS ResourceConfig with an @ApplicationPath annotation.
- It acts as a hook between RESTful application and the web container.
- When server is started, it will examine this class and look for JAX-RS annotated classes inside the package listed in the this class constructor.

```
package rest_api;

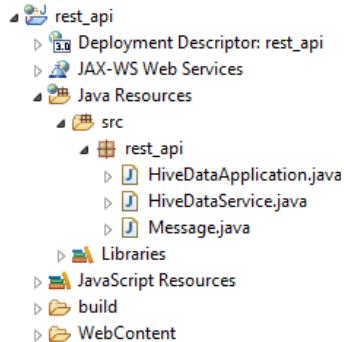
import javax.ws.rs.ApplicationPath;
import org.glassfish.jersey.server.ResourceConfig;

@Path("/")
public class HiveDataApplication extends ResourceConfig {

    public HiveDataApplication() {
        packages("rest_api");
    }
}
```

# REST API – Project Setup

- 1) Copy 3 java file into src folder.

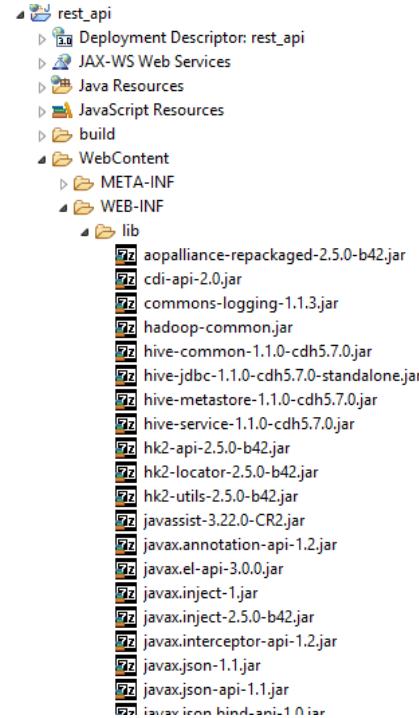


- 2) Download jaxrs-ri JAR 2.27 with all dependency. <https://jar-download.com/artifacts/org.glassfish.jersey.bundles/jaxrs-ri/2.27/source-code>

- 3) Copy following jars from downloaded hive jar.

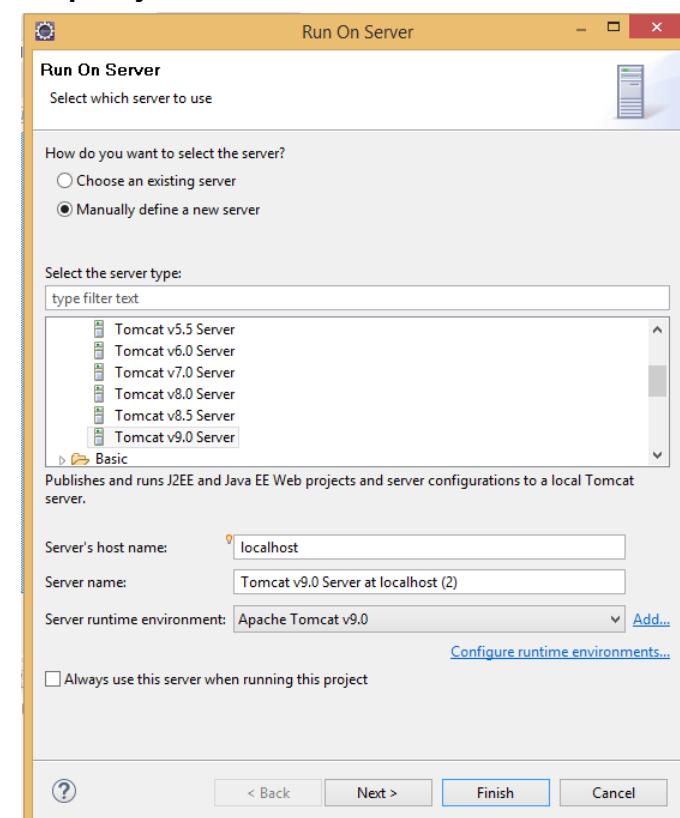
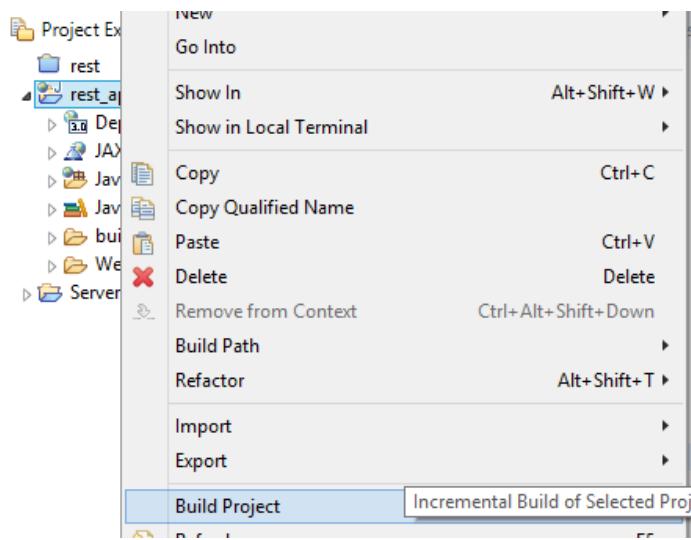
- commons-logging-1.1.3
- hadoop-common
- hive-common-1.1.0-cdh5.7.0
- hive-jdbc-1.1.0-cdh5.7.0-standalone
- hive-metastore-1.1.0-cdh5.7.0
- hive-service-1.1.0-cdh5.7.0
- libthrift-0.9.2

- 4) Copy all Jar jaxrs jar and hive jar tp WEB-INF lib folder.

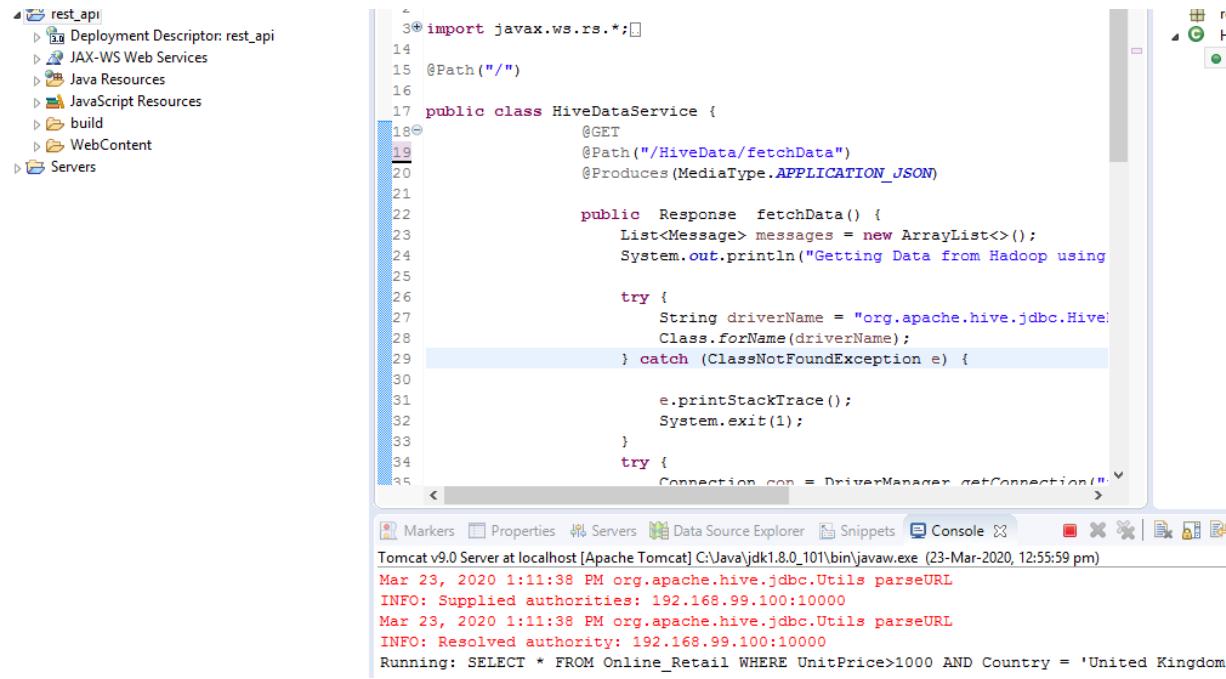


# REST API – Deployment

- 1) Right Click on project
- 2) Select Build project to create deployment war file
- 3) Right Click on project.
- 4) Click Run on server.
- 5) Configure Tomcat 9.0 server and run the project



# REST API - OUTPUT



```
rest_api
Deployment Descriptor: rest_api
JAX-WS Web Services
Java Resources
JavaScript Resources
build
WebContent
Servers

import javax.ws.rs.*;
@Path("/")
public class HiveDataService {
    @GET
    @Path("/HiveData/fetchData")
    @Produces(MediaType.APPLICATION_JSON)

    public Response fetchData() {
        List<Message> messages = new ArrayList<>();
        System.out.println("Getting Data from Hadoop using");

        try {
            String driverName = "org.apache.hive.jdbc.Hive";
            Class.forName(driverName);
        } catch (ClassNotFoundException e) {

            e.printStackTrace();
            System.exit(1);
        }
        try {
            Connection con = DriverManager.getConnection("jdbc:hive2://localhost:10000");
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("SELECT * FROM Online_Retail WHERE UnitPrice>1000 AND Country = 'United Kingdom'");

            while(rs.next()) {
                messages.add(new Message(rs.getString("Country"), rs.getString("Description"), rs.getString("InvoiceDate"), rs.getString("InvoiceNo"), rs.getInt("Quantity"), rs.getDouble("UnitPrice")));
            }
            con.close();
        } catch (SQLException e) {
            e.printStackTrace();
            System.exit(1);
        }
        return Response.ok(messages).build();
    }
}
```

- 1) Web application would be deployed on server.
- 2) Open below url in browser  
[http://localhost:8080/rest\\_api/HiveData/PurchaseDetail](http://localhost:8080/rest_api/HiveData/PurchaseDetail)

```
[{"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2010-12-07 15:04:00.0", "invoiceNo": "C537630", "quantity": "-1", "stockCode": "AMAZONFEE", "unitPrice": "13541.33"}, {"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2010-12-07 15:08:00.0", "invoiceNo": "C537632", "quantity": "1", "stockCode": "AMAZONFEE", "unitPrice": "13541.33"}, {"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2010-12-07 15:34:00.0", "invoiceNo": "C537644", "quantity": "-1", "stockCode": "AMAZONFEE", "unitPrice": "15:41:00.0"}, {"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2010-12-07 15:49:00.0", "invoiceNo": "C537651", "quantity": "-1", "stockCode": "AMAZONFEE", "unitPrice": "15:41:00.0"}, {"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2010-12-07 15:51:00.0", "invoiceNo": "C537652", "quantity": "1", "stockCode": "AMAZONFEE", "unitPrice": "15:41:00.0"}, {"country": "United Kingdom", "description": "Manual", "invoiceDate": "2010-12-13 17:14:00.0", "invoiceNo": "C538682", "quantity": "-1", "stockCode": "M", "unitPrice": "1283.8"}, {"country": "United Kingdom", "description": "Manual", "invoiceDate": "2010-12-22 14:41:00.0", "invoiceNo": "C539856", "quantity": "1", "stockCode": "M", "unitPrice": "1283.8"}, {"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2011-01-05 09:57:00.0", "invoiceNo": "C540118", "quantity": "-1", "stockCode": "AMAZONFEE", "unitPrice": "16888.02"}, {"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2011-01-20 11:48:00.0", "invoiceNo": "C541651", "quantity": "-1", "stockCode": "AMAZONFEE", "unitPrice": "16453.71"}, {"country": "United Kingdom", "description": "Manual", "invoiceDate": "2011-01-20 11:50:00.0", "invoiceNo": "C541653", "quantity": "1", "stockCode": "BANK CHARGES", "unitPrice": "1283.8"}, {"country": "United Kingdom", "description": "Manual", "invoiceDate": "2011-02-15 12:36:00.0", "invoiceNo": "C544647", "quantity": "-1", "stockCode": "M", "unitPrice": "1283.8"}, {"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2011-02-21 11:00:00.0", "invoiceNo": "C544589", "quantity": "-1", "stockCode": "AMAZONFEE", "unitPrice": "5575.28"}, {"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2011-03-18 12:56:00.0", "invoiceNo": "C546987", "quantity": "-1", "stockCode": "AMAZONFEE", "unitPrice": "5258.77"}, {"country": "United Kingdom", "description": "AMAZON FEE", "invoiceDate": "2011-03-18 12:59:00.0", "invoiceNo": "C546989", "quantity": "-1", "stockCode": "AMAZONFEE", "unitPrice": "5225.03"}, {"country": "United Kingdom", "description": "Manual", "invoiceDate": "2011-03-28 11:51:00.0", "invoiceNo": "C547899", "quantity": "-1", "stockCode": "M", "unitPrice": "1486.12"}]
```

**THANK YOU**