Introduction to Big Data

Dr. Rajiv Misra
Dept. of Computer Science & Engg.
Indian Institute of Technology Patna
rajivm@iitp.ac.in

Preface

Content of this Lecture:

- In this lecture, we will discuss a brief introduction to Big Data: Why Big Data, Where did it come from?, Challenges and applications of Big Data, Characteristics of Big Data *i.e.* Volume, Velocity, Variety and more V's.

What's Big Data?

- **Big data** is the term for a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications.

- The challenges include **capture, curation, storage, search, sharing, transfer, analysis, and visualization**.
- The trend to larger data sets is due to the additional information derivable from analysis of a single large set of related data, as compared to separate smaller sets with the same total amount of data, allowing correlations to be found to "**spot business trends, determine quality of research, prevent diseases, link legal citations, combat crime, and determine real-time roadway traffic conditions**."

Facts and Figures

- **Walmart** handles 1 million customer transactions/hour.
- **Facebook** handles 40 billion photos from its user base!
- **Facebook** inserts 500 terabytes of new data every day.
- **Facebook** stores, accesses, and analyzes 30+ Petabytes of user generated data.
- **A flight generates** 240 terabytes of flight data in 6-8 hours of flight.
- **More than 5 billion people** are calling, texting, tweeting and browsing on mobile phones worldwide.
- **Decoding the human genome** originally took 10 years to process; now it can be achieved in one week.[8]
- **The largest AT&T database** boasts titles including the largest volume of data in one unique database (312 terabytes) and the second largest number of rows in a unique database (1.9 trillion), which comprises AT&T's extensive calling records.

An Insight

- **Byte:** One grain of rice
- **KB(3):** One cup of rice:
- **MB (6):** 8 bags of rice: **Desktop**
- **GB (9):** 3 Semi trucks of rice:
- **TB (12):** 2 container ships of rice **Internet**
- **PB (15):** Blankets ½ of Jaipur
- **Exabyte (18):** Blankets West coast **Big Data**

  Or 1/4th of India
- **Zettabyte (21):** Fills Pacific Ocean **Future**
- **Yottabyte(24):** An earth-sized rice bowl
- **Brontobyte (27):** Astronomical size

What's making so much data?

- Sources: People, machine, organization: Ubiquitous computing
- More people carrying data-generating devices (Mobile phones with facebook, GPS, Cameras, etc.)

- **Data on the Internet:**
- Internet live stats

Source of Data Generation

12+ TBs
of tweet data every day

? TBs of data every day

25+ TBs **of log data every day**

30 billion RFID tags today (1.3B in 2005)

4.6 billion camera phones world wide

100s of millions of GPS enabled devices sold annually

2+ billion people on the Web by end 2011

76 million smart meters in 2009… 200M by 2014

An Example of Big Data at Work

Crowdsourcing

Computing

Sensing

Real time traffic info

Where is the problem?

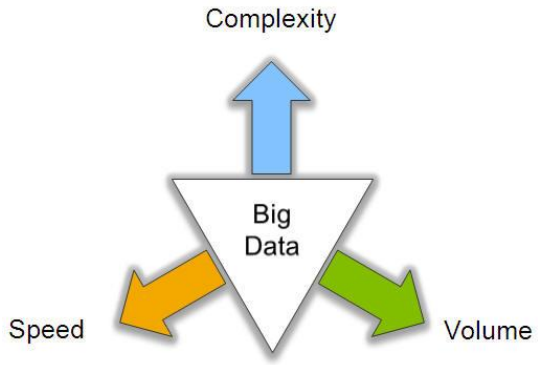
- Traditional RDBMS queries isn't sufficient to get useful information out of the huge volume of data
- To search it with traditional tools to find out if a particular topic was trending would take so long that the result would be meaningless by the time it was computed.
- Big Data come up with a solution to store this data in novel ways in order to make it more accessible, and also to come up with methods of performing analysis on it.

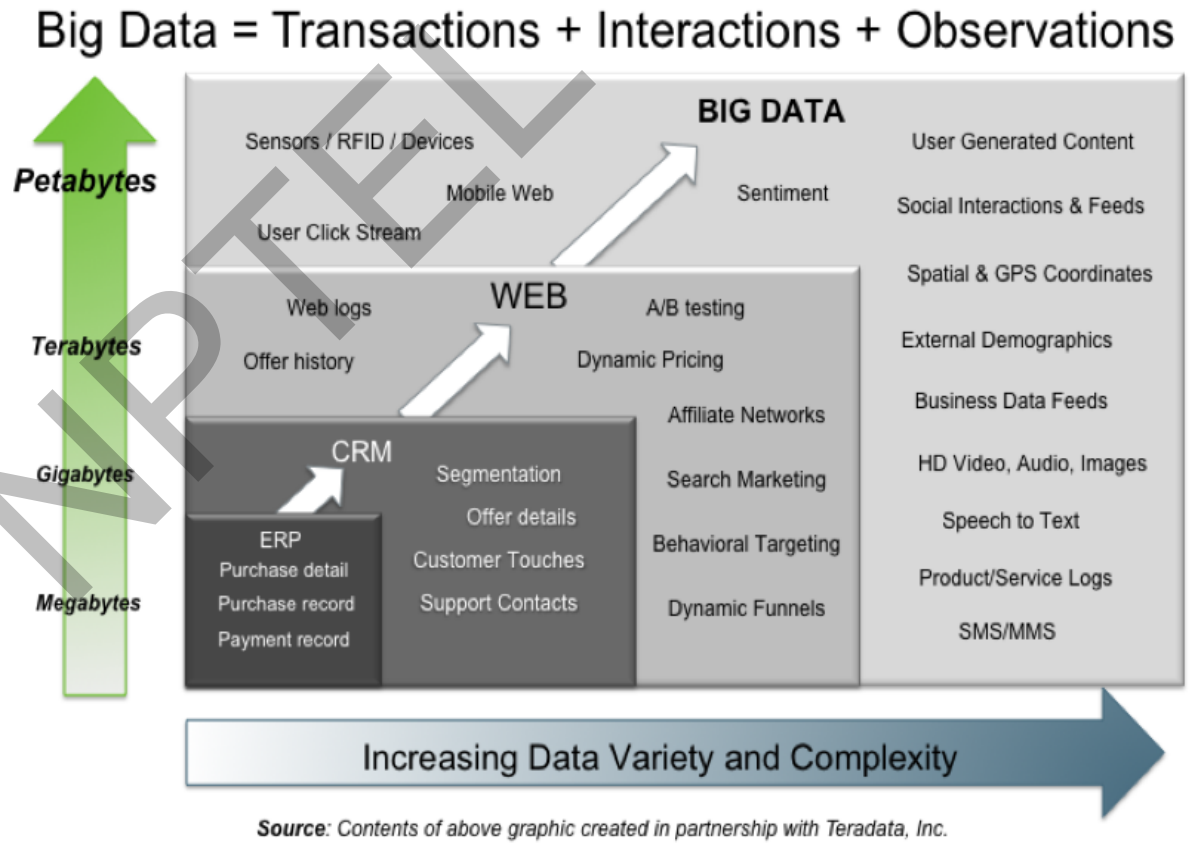
Challenges

- Capturing
- Storing
- Searching
- Sharing
- Analysing
- Visualization

IBM considers Big Data (3V's):

- The 3V's: Volume, Velocity and Variety.

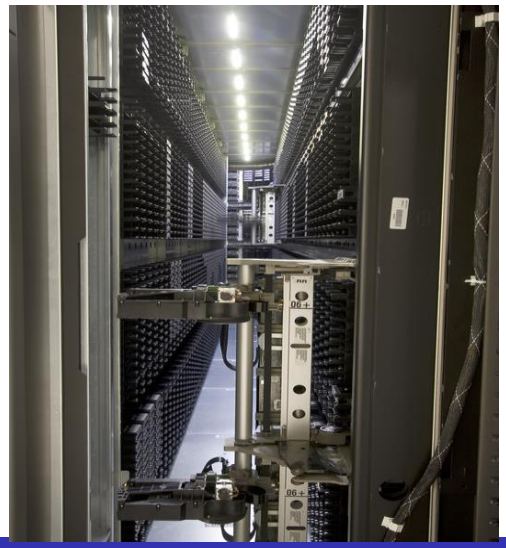




***Source**: Contents of above graphic created in partnership with Teradata, Inc.*

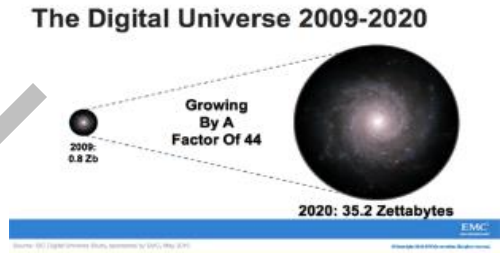
Volume (Scale)

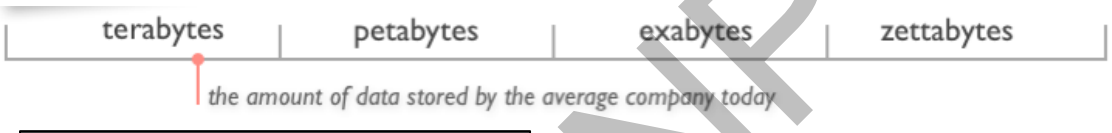
- **Volume:** Enterprises are awash with ever-growing data of all types, easily amassing terabytes even Petabytes of information.
  - Turn 12 terabytes of Tweets created each day into improved product sentiment analysis
  - Convert 350 billion annual meter readings to better predict power consumption

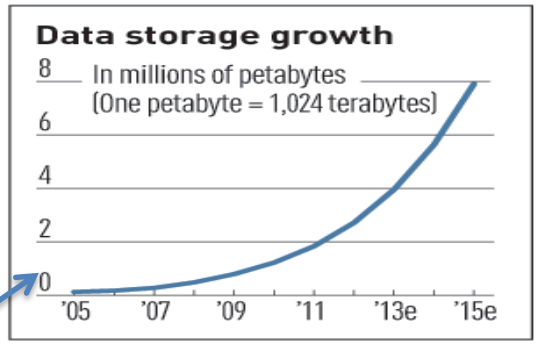


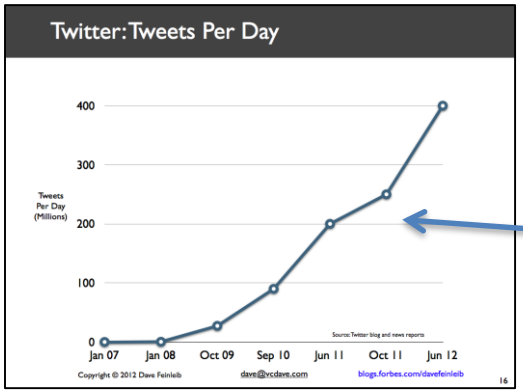
Volume (Scale)

- **Data Volume**
  - 44x increase from 2009 2020
  - From 0.8 zettabytes to 35zb
- Data volume is increasing exponentially









Exponential increase in collected/generated data

Example 1: CERN's Large Hydron Collider(LHC)

CERN's Large Hydron Collider (LHC) generates 15 PB a year

Example 2: The Earthscope

- **The Earthscope** is the world's largest science project. Designed to track North America's geological evolution, this observatory records data over 3.8 million square miles, amassing 67 terabytes of data. It analyzes seismic slips in the San Andreas fault, sure, but also the plume of magma underneath Yellowstone and much, much more.

(http://www.msnbc.msn.com/id/44363598/ns/technology_and_science-future_of_technology/#.TmetOdQ--uI)

Velocity (Speed)

- **Velocity:** Sometimes 2 minutes is too late. For time-sensitive processes such as catching fraud, big data must be used as it streams into your enterprise in order to maximize its value.
  - Scrutinize 5 million trade events created each day to identify potential fraud
  - Analyze 500 million daily call detail records in real-time to predict customer churn faster

Examples: Velocity (Speed)

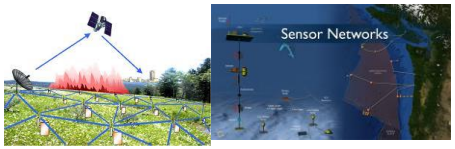
- Data is begin generated fast and need to be processed fast
- Online Data Analytics
- Late decisions ➔ missing opportunities
- **Examples**
  - **E-Promotions:** Based on your current location, your purchase history, what you like ➔ send promotions right now for store next to you
  - **Healthcare monitoring:** sensors monitoring your activities and body ➔ any abnormal measurements require immediate reaction

Real-time/Fast Data

Social media and networks
(all of us are generating data)

Scientific instruments
(collecting all sorts of data)

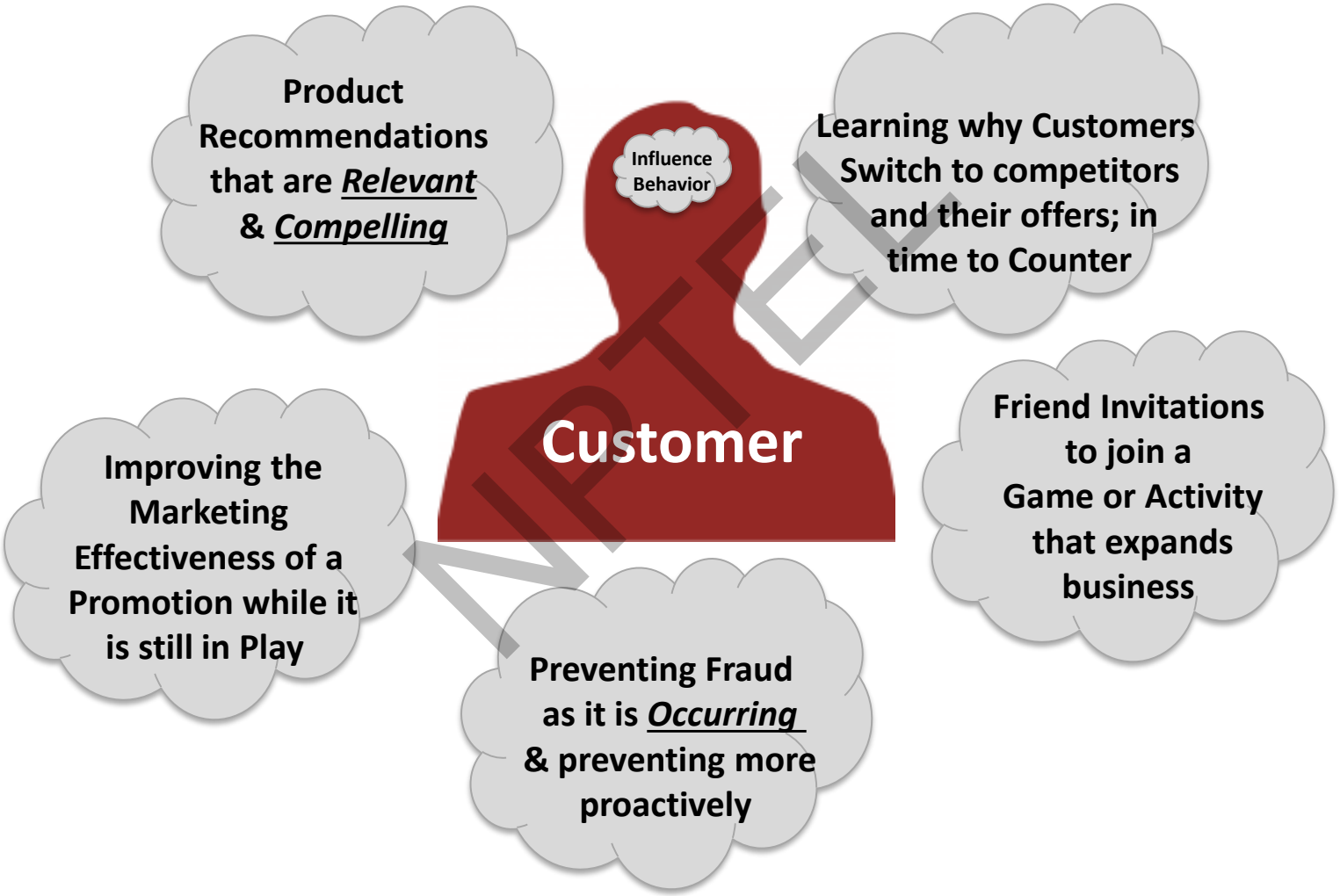
Mobile devices
(tracking all objects all the time)



Sensor technology and networks
(measuring all kinds of data)

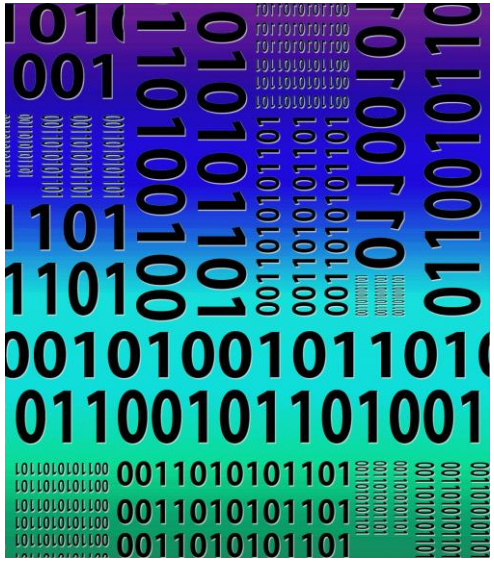
- The progress and innovation is no longer hindered by the ability to collect data
- But, by the ability to manage, analyze, summarize, visualize, and discover knowledge from the collected data in a timely manner and in a scalable fashion

Real-Time Analytics/Decision Requirement



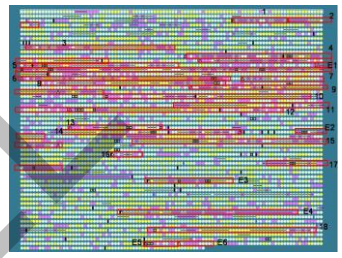
Variety (Complexity)

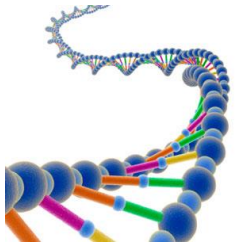
- **Variety:** Big data is any type of data –
  - Structured Data (example: tabular data)
  - Unstructured –text, sensor data, audio, video
  - Semi Structured : web data, log files

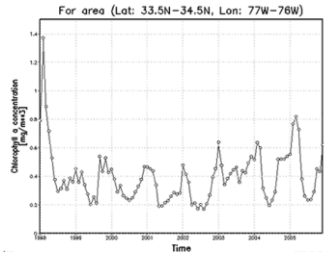


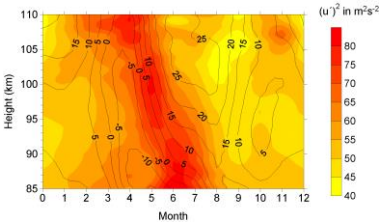
Examples: Variety (Complexity)

- Relational Data (Tables/Transaction/Legacy Data)
- Text Data (Web)
- Semi-structured Data (XML)
- Graph Data
  - Social Network, Semantic Web (RDF), …
- Streaming Data
  - You can only scan the data once
- A single application can be generating/collecting many types of data
- Big Public Data (online, weather, finance, etc)





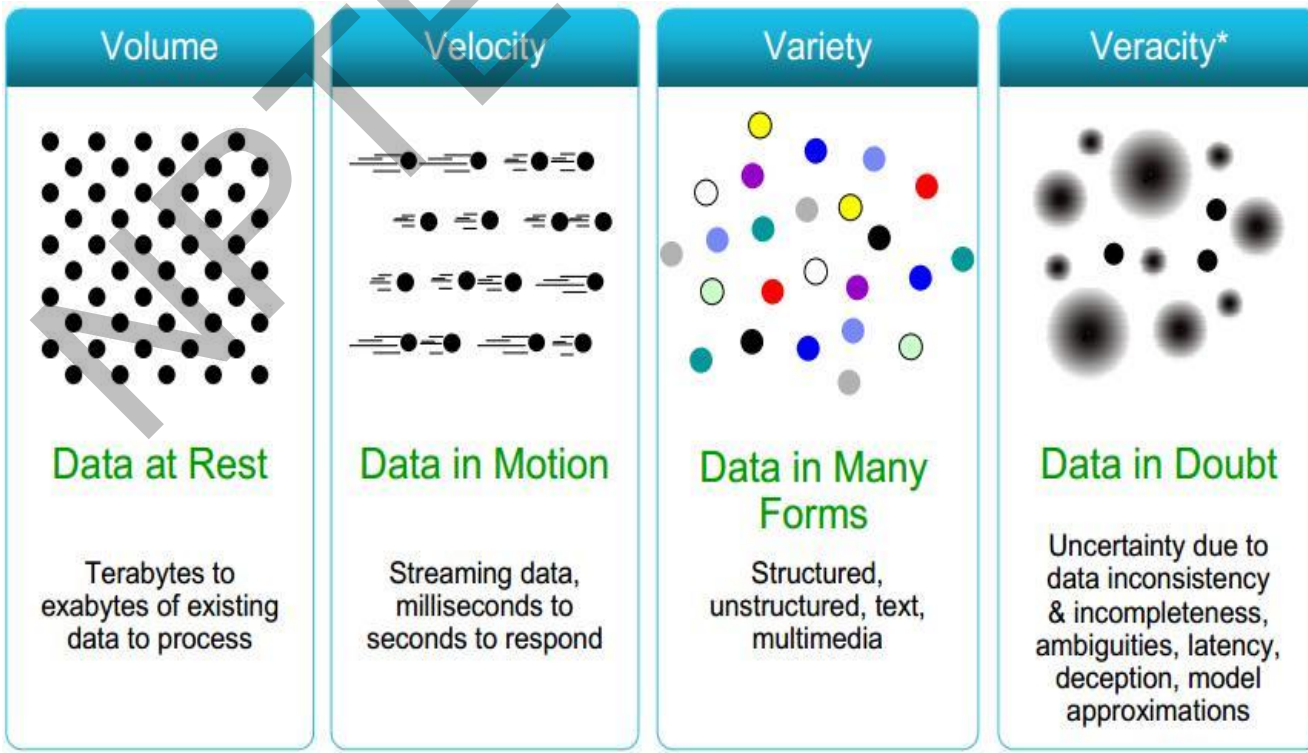




To extract knowledge➔ all these types of data need to linked together

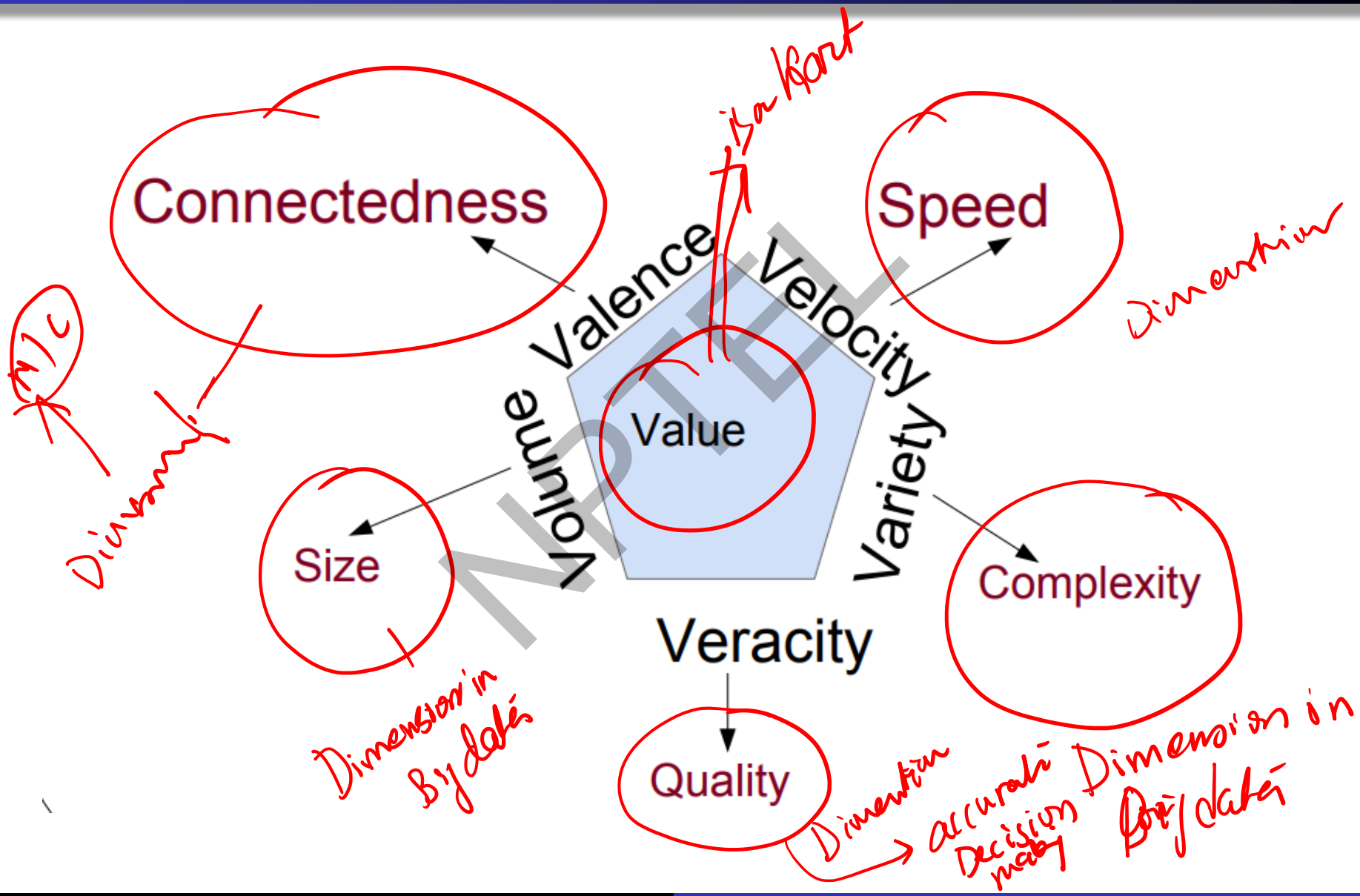
The 3 Big V's (+1)

- **Big 3V's**
  - Volume
  - Velocity
  - Variety
- **Plus 1**
  - Value



The 3 Big V's (+1) (+ N more)

- **Plus many more**
  - Veracity
  - Validity
  - Variability
  - Viscosity & Volatility
  - Viability,
    Venue,
    Vocabulary, Vagueness,
    …

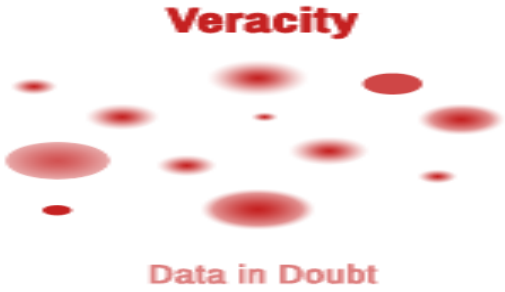

Connectedness
Speed
Valence
Velocity
Volume
Value
Variety
Size
Complexity
Veracity
Quality

Value

- Integrating Data
  - Reducing data complexity
  - Increase data availability
  - Unify your data systems
  - All 3 above will lead to increased data collaboration -> add value to your big data

Veracity

- **Veracity** refers to the biases ,noise and abnormality in data, trustworthiness of data.
- 1 in 3 business leaders don't trust the information they use to make decisions.
  - How can you act upon information if you don't trust it?
  - Establishing trust in big data presents a huge challenge as the variety and number of sources grows.



Valence

- **Valence** refers to the connectedness of big data.
- Such as in the form of graph networks

Validity

Accuracy and correctness of the data relative to a particular use

- Example: **Gauging storm intensity**

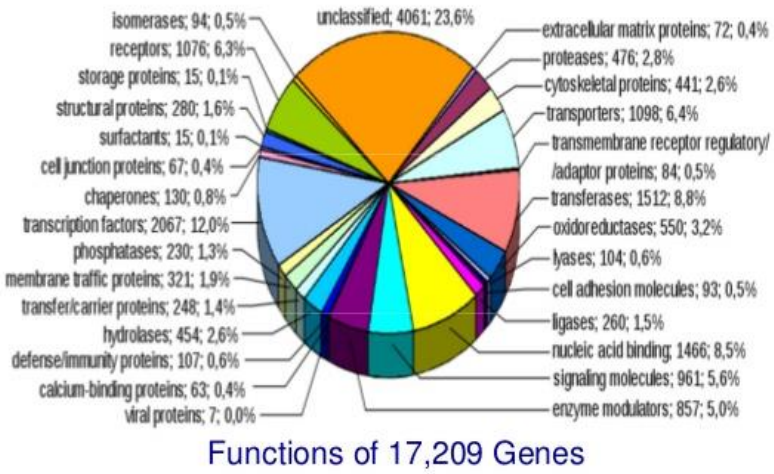
  satellite imagery vs social media posts

- prediction quality vs human impact

Variability

How the meaning of the data changes over time

- Language evolution
- Data availability
- Sampling processes
- Changes in characteristics of the data source



Functions of 17,209 Genes

Viscosity & Volatility

- Both related to velocity
- **Viscosity:** *data velocity relative to timescale of event being studied*
- **Volatility:** *rate of data loss and stable lifetime of data*
  - Scientific data often has practically unlimited lifespan, but social / business data may evaporate in finite time

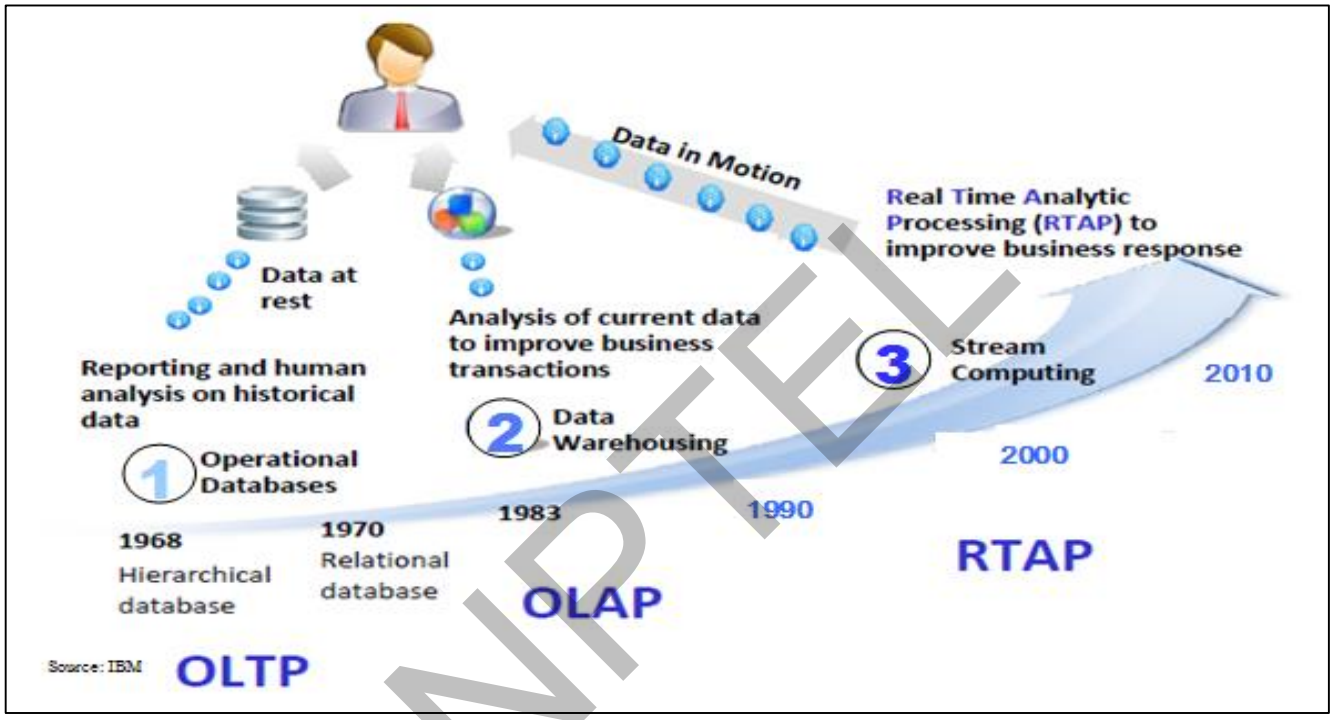
More V's

- **Viability**
  - Which data has meaningful relations to questions of interest?
- **Venue**
  - Where does the data live and how do you get it?
- **Vocabulary**
  - Metadata describing structure, content, & provenance
  - Schemas, semantics, ontologies, taxonomies, vocabularies
- **Vagueness**
  - Confusion about what "Big Data" means

Dealing with Volume

- Distill big data down to small information
- Parallel and automated analysis
- Automation requires standardization
- Standardize by reducing Variety:
  - Format
  - Standards
  - Structure

Harnessing Big Data



- **OLTP:** Online Transaction Processing (DBMSs)
- **OLAP:** Online Analytical Processing (Data Warehousing)
- **RTAP:** Real-Time Analytics Processing (Big Data Architecture & technology)

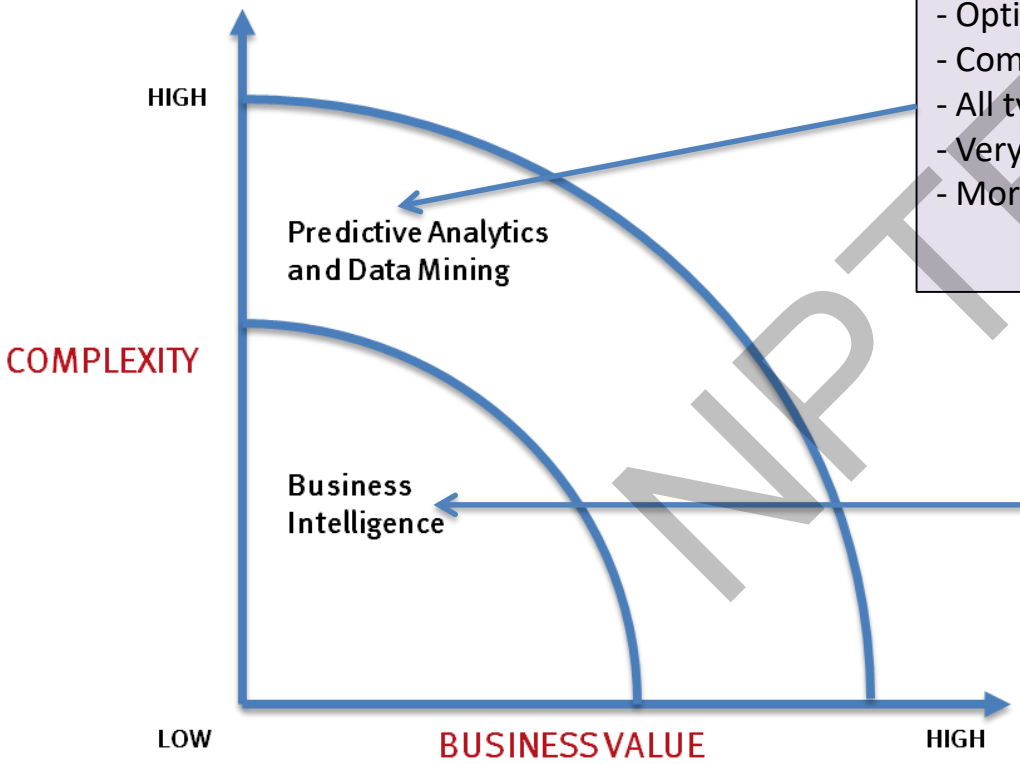
The Model Has Changed…

- **The Model of Generating/Consuming Data has Changed**

Old Model: Few companies are generating data, all others are consuming data

New Model: all of us are generating data, and all of us are consuming data

What's driving Big Data

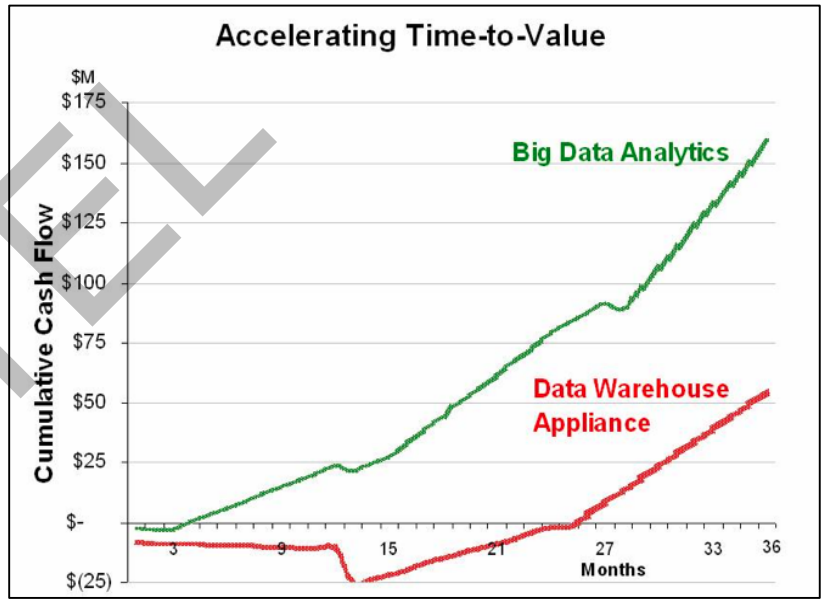


- Optimizations and predictive analytics
- Complex statistical analysis
- All types of data, and many sources
- Very large datasets
- More of a real-time

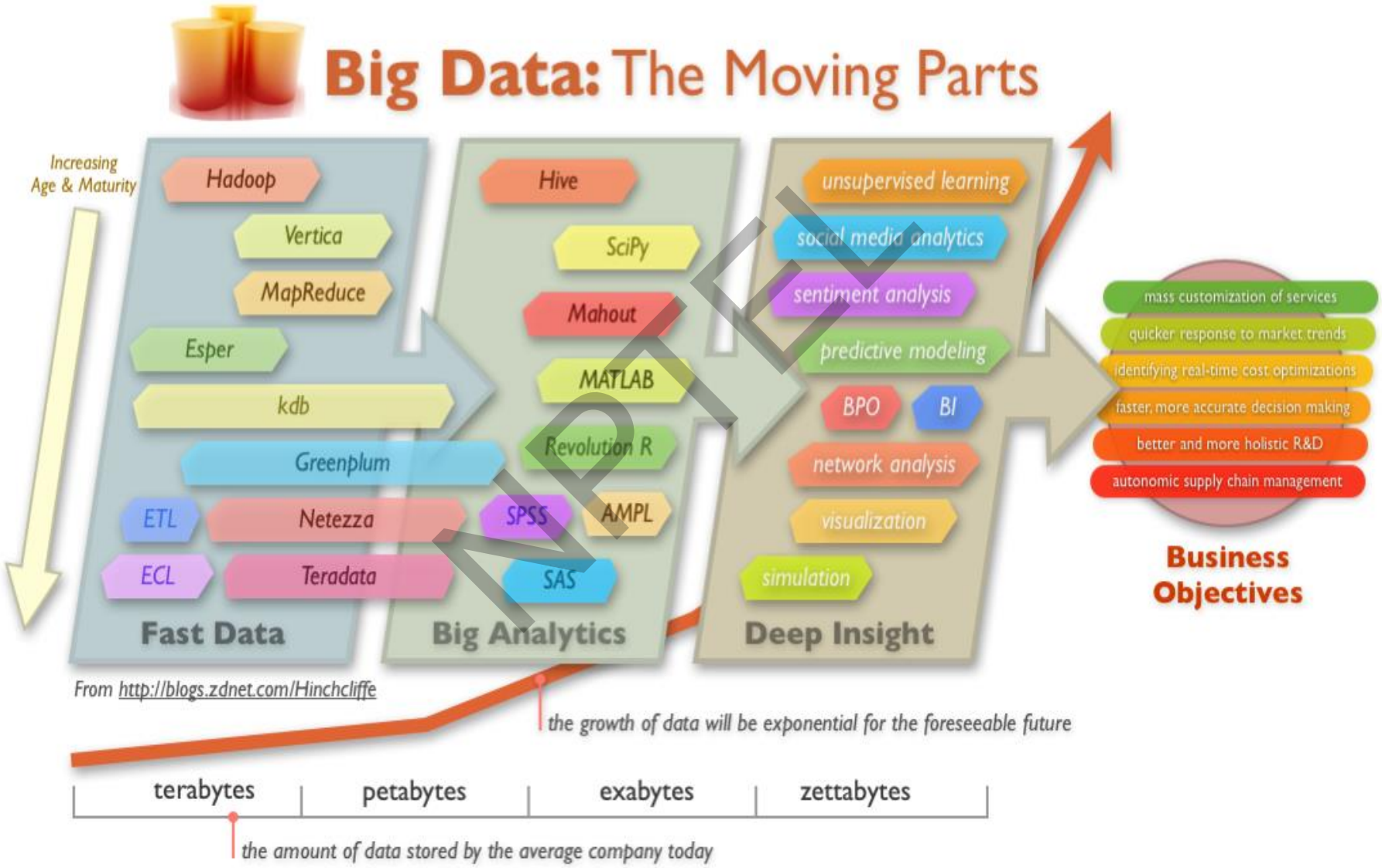
- Ad-hoc querying and reporting
- Data mining techniques
- Structured data, typical sources
- Small to mid-size datasets

Big Data Analytics

- Big data is more real-time in nature than traditional Dataware house (DW) applications
- Traditional DW architectures (e.g. Exadata, Teradata) are not well-suited for big data apps
- Shared nothing, massively parallel processing, scale out architectures are well-suited for big data apps



Big Data Technology



Conclusion

- In this lecture, we have **defined Big Data** and discussed the challenges and applications of Big Data.

- We have also described **characteristics of Big Data** *i.e.* Volume, Velocity, Variety and more V's, Big Data Analytics, Big Data Landscape and Big Data Technology.

Big Data Enabling Technologies

Dr. Rajiv Misra
Dept. of Computer Science & Engg.
Indian Institute of Technology Patna
rajivm@iitp.ac.in

Preface

Content of this Lecture:

- In this lecture, we will discuss a brief introduction to Big Data Enabling Technologies.

Introduction

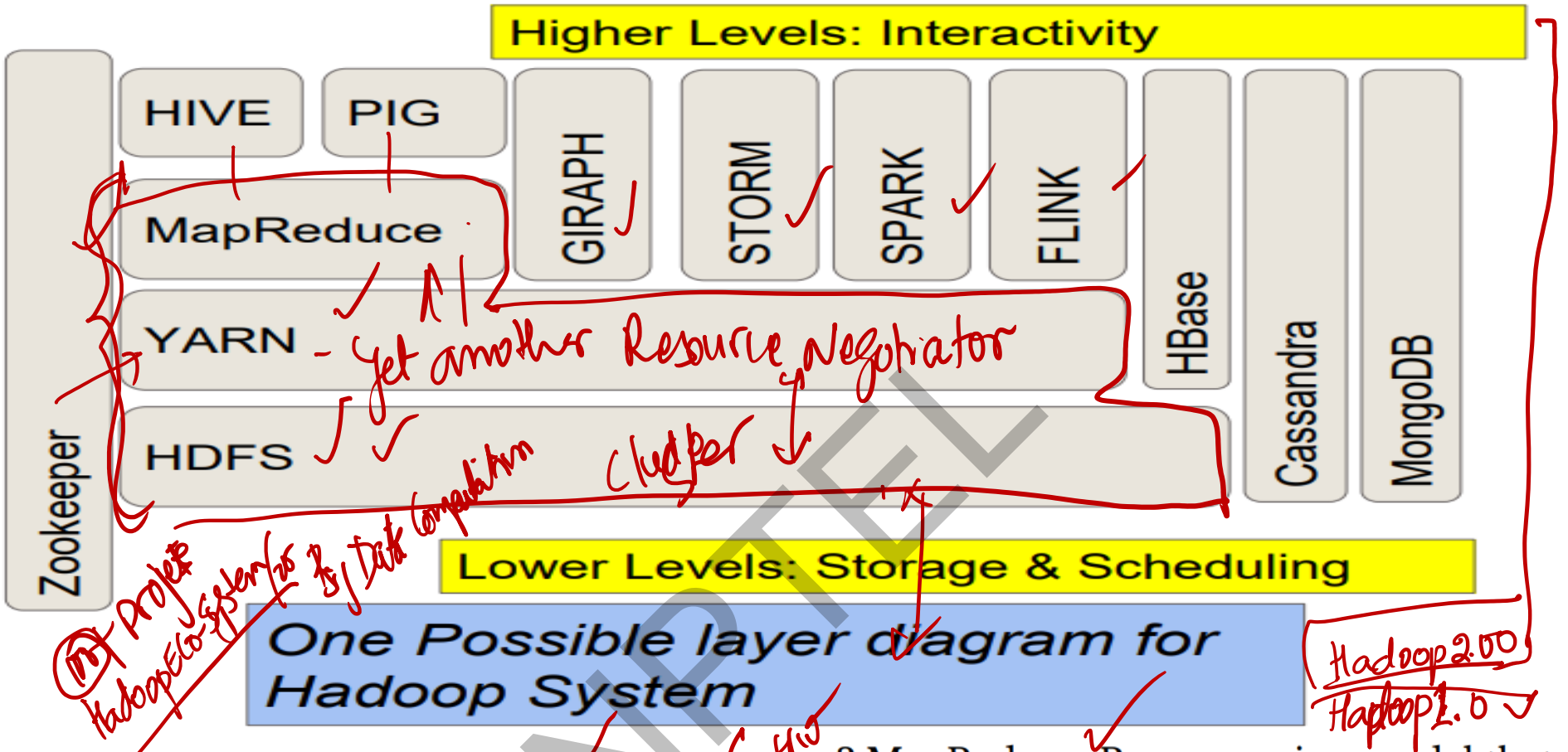
- Big Data is used for a collection of data sets so large and complex that it is difficult to process using traditional tools.
- **A recent survey says that 80% of the data created in the world are unstructured.**
- One challenge is how we can store and process this big amount of data. In this lecture, we will discuss the top technologies used to store and analyse Big Data.

Apache Hadoop

- **Apache Hadoop** is an open source software framework for big data.

  It has two basic parts:

- **Hadoop Distributed File System (HDFS)** is the storage system of Hadoop which splits big data and distribute across many nodes in a cluster.
  a. Scaling out of H/W resources
  b. Fault Tolerant
- **MapReduce:** Programming model that simplifies parallel programming.
  a. Map-> apply ()
  b. Reduce-> summarize ()
  c. Google used MapReduce for Indexing websites.

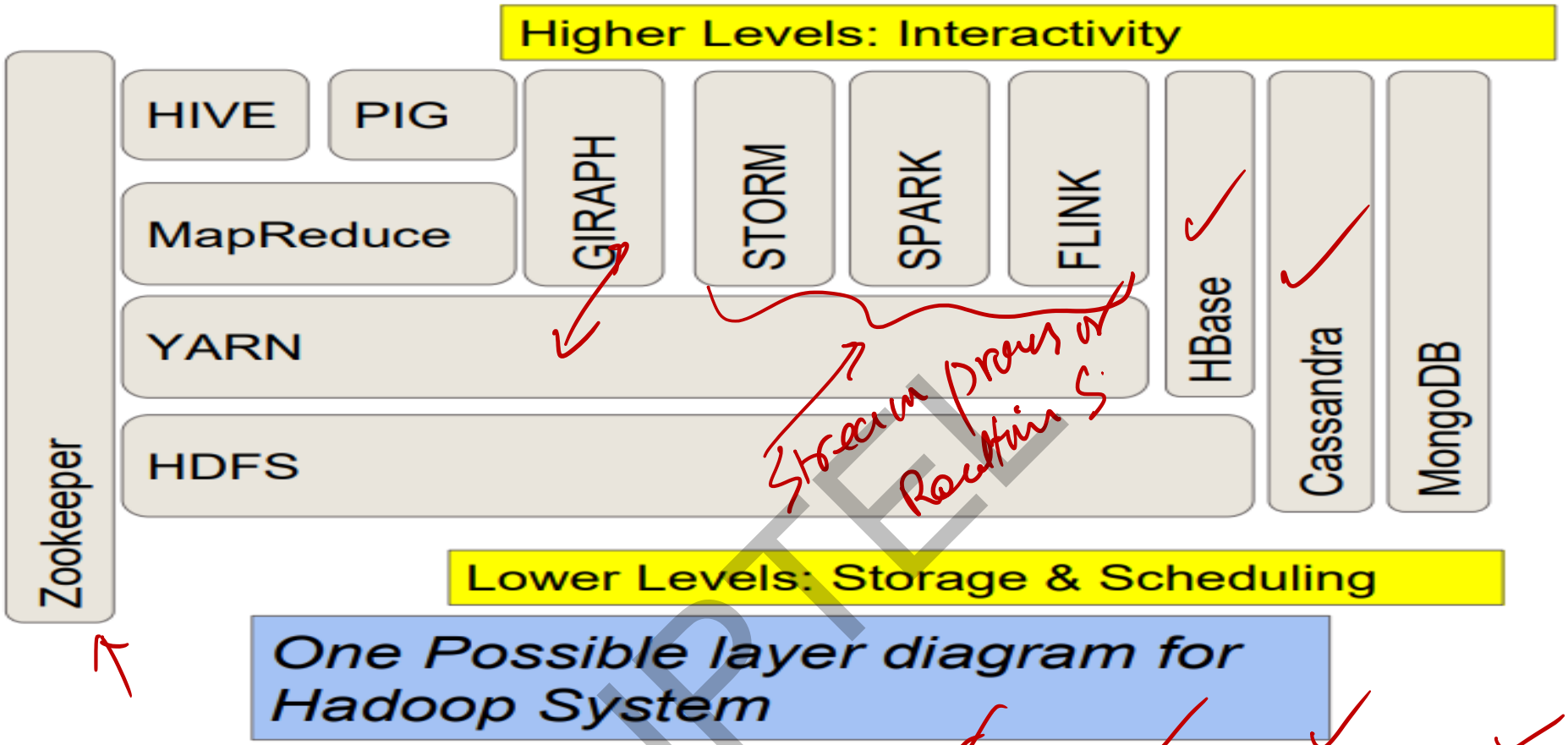


1.Hadoop Distributed File System: HDFS
   a.Scaling out of H/W resources
   b.Fault Tolerant

2.YARN: Flexible scheduling & Resource management over HDFS
   a.Yahoo uses YARN to schedule jobs over 40,000 servers

3.MapReduce: Programming model that simplifies parallel programming
   a.Map-> apply()
   b.Reduce-> summarize()
   c.Google used MR for Indexing websites

4.Pig & Hive: Augment MapReduce
   a.Pig: Dataflow based script programming
   b.Hive: SQL like queries (Created at Facebook)



One Possible layer diagram for Hadoop System

5.Giraph: Specialized models for graph processing
   a.Used by Facebook to analyze social graphs
6.Storm/Spark/Flink: Real time, in-memory processing of data on top of YARN/HDFS, 100 times faster than regular processing

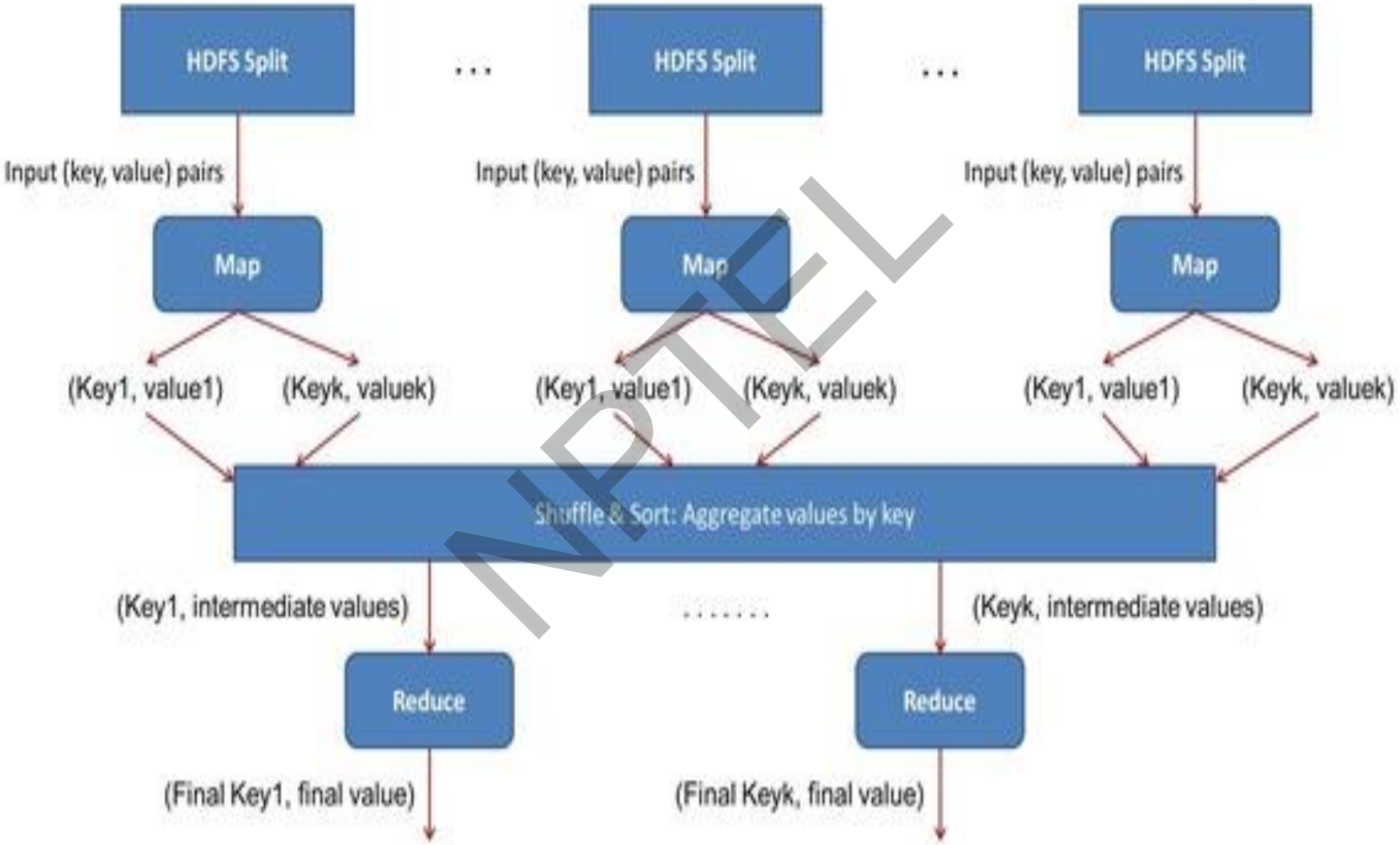
5.Cassandra/MongoDB/HBase: NOSQL database
   a.HBase is used for Facebook's Messaging Platform
   b.Sparse tables
6.Zookeeper: Created by YAHOO to perform the duties of centralized management system for synchronization, configuration and to ensure high availability

Map Reduce

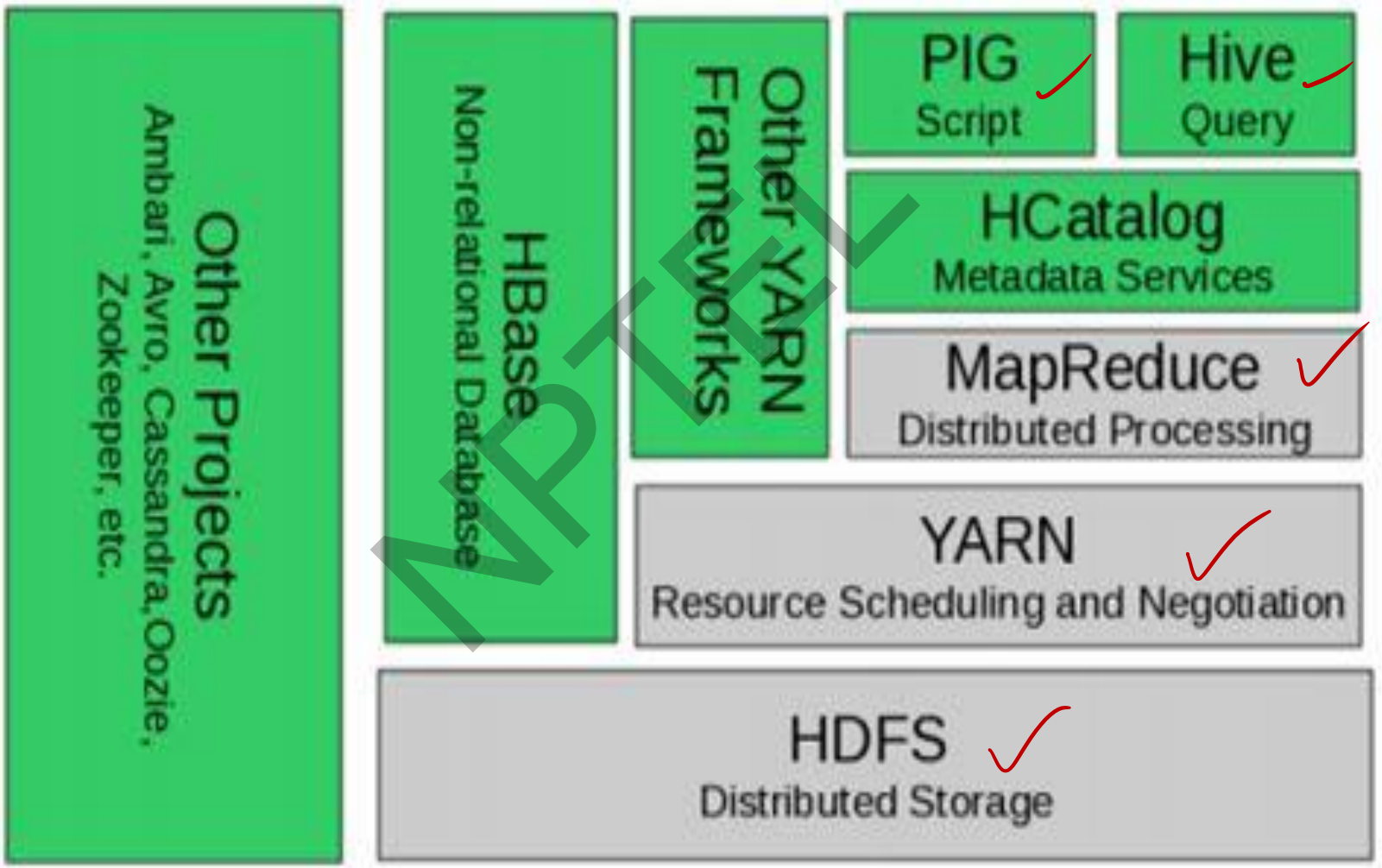
- **MapReduce** is a programming model and an associated implementation for **processing and generating large data sets.**

- Users specify a **map** function that processes a key/value pair to generate a set of intermediate key/value pairs, and a **reduce** function that merges all intermediate values associated with the same intermediate key

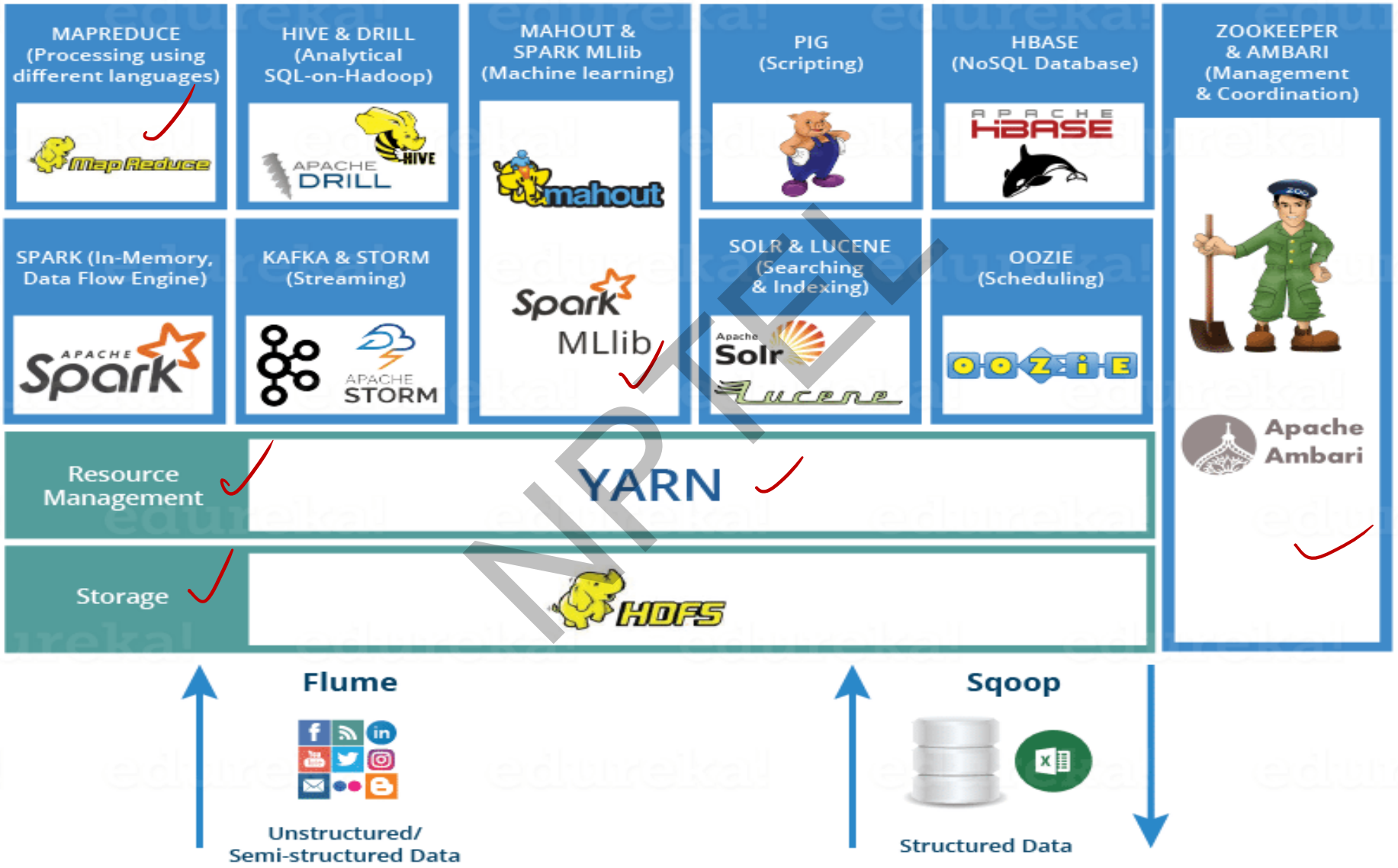
Map Reduce



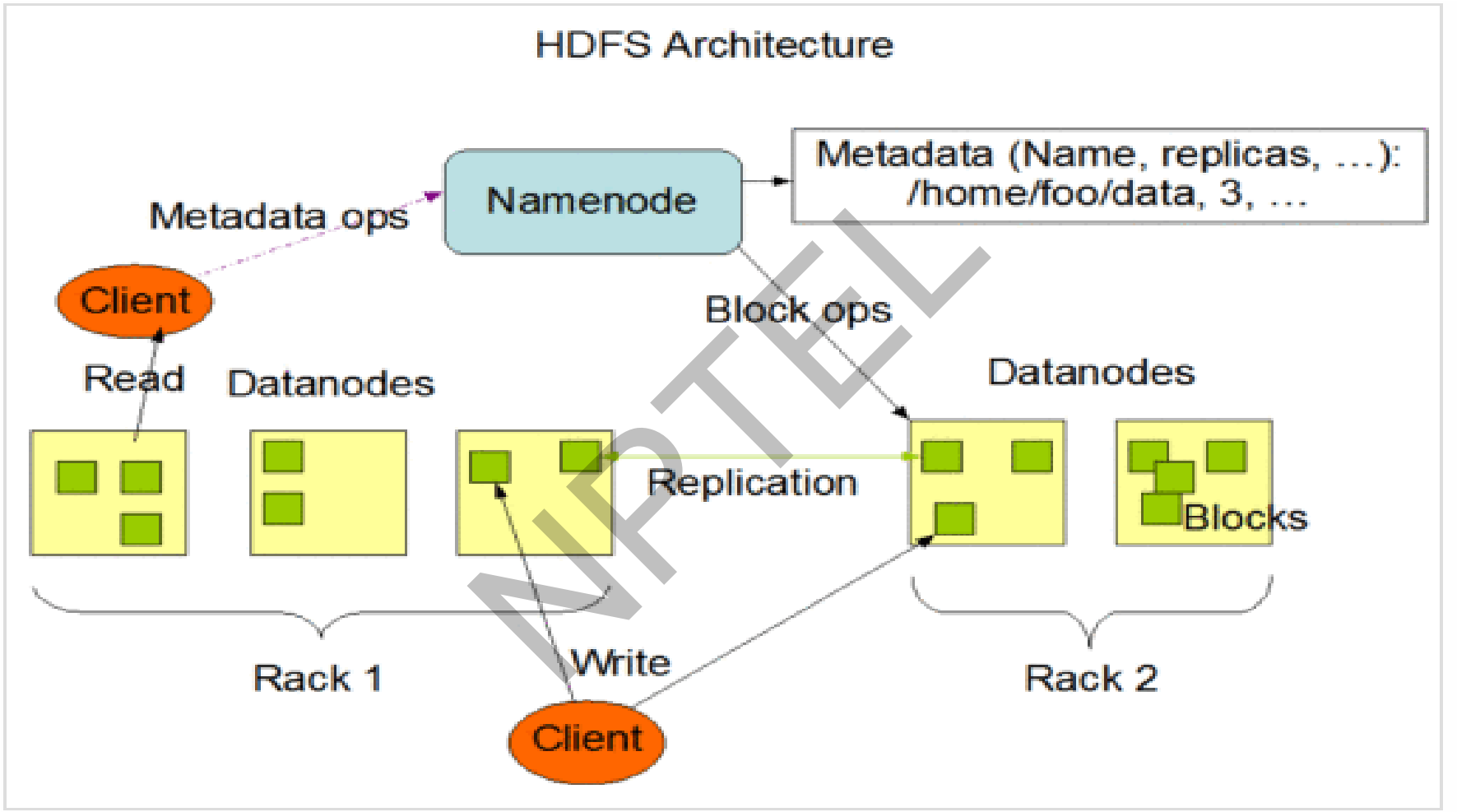
Hadoop Ecosystem



Hadoop Ecosystem



HDFS Architecture



Basic architecture of Hadoop distributed file system (HDFS)

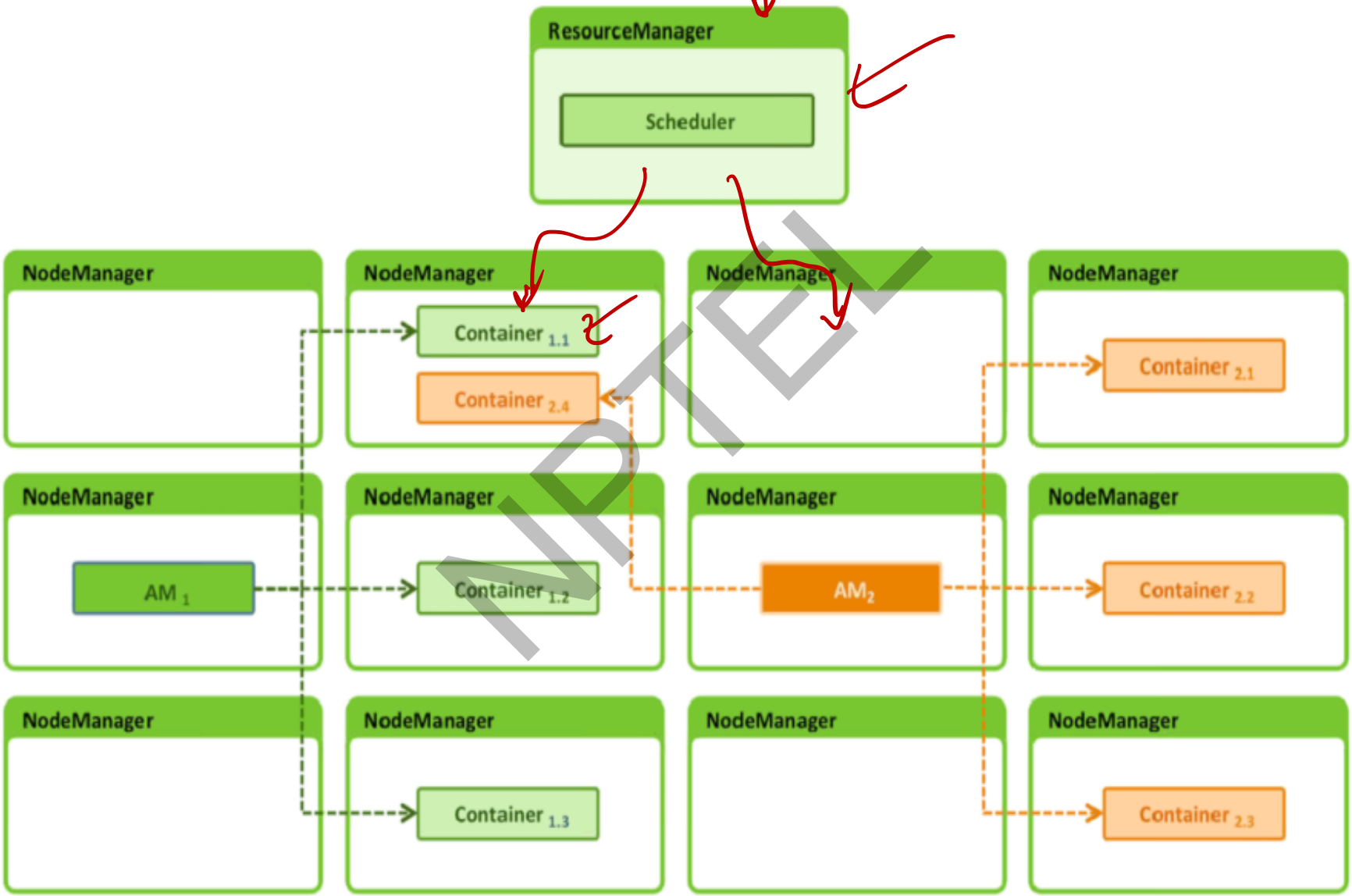
YARN

- **YARN – Yet Another Resource Manager.**

- Apache Hadoop YARN is the resource management and job scheduling technology in the open source Hadoop distributed processing framework.

- YARN is responsible for allocating system resources to the various applications running in a Hadoop cluster and scheduling tasks to be executed on different cluster nodes.

YARN Architecture



Hive

- **Hive** is a distributed data management for Hadoop.
- It supports SQL-like query option HiveSQL (HSQL) to access big data.
- It can be primarily used for Data mining purpose.
- It runs on top of Hadoop.

Apache Spark

- **Apache Spark** is a big data analytics framework that was originally developed at the University of California, Berkeley's AMPLab, in 2012. Since then, it has gained a lot of attraction both in academia and in industry.

- Apache Spark is a lightning-fast cluster computing technology, designed for fast computation.

- Apache Spark is a lightning-fast cluster computing technology, designed for fast computation

ZooKeeper

- **ZooKeeper is a highly reliable distributed coordination kernel**, which can be used for distributed locking, configuration management, leadership election, work queues,….
- Zookeeper is a replicated service that holds the metadata of distributed applications.
- **Key attributed of such data**
  - **Small size**
  - **Performance sensitive**
  - **Dynamic**
  - **Critical**

https://zookeeper.apache.org/

- **In very simple words,** it is a central store of key-value using which distributed systems can coordinate. Since it needs to be able to handle the load, Zookeeper itself runs on many machines.

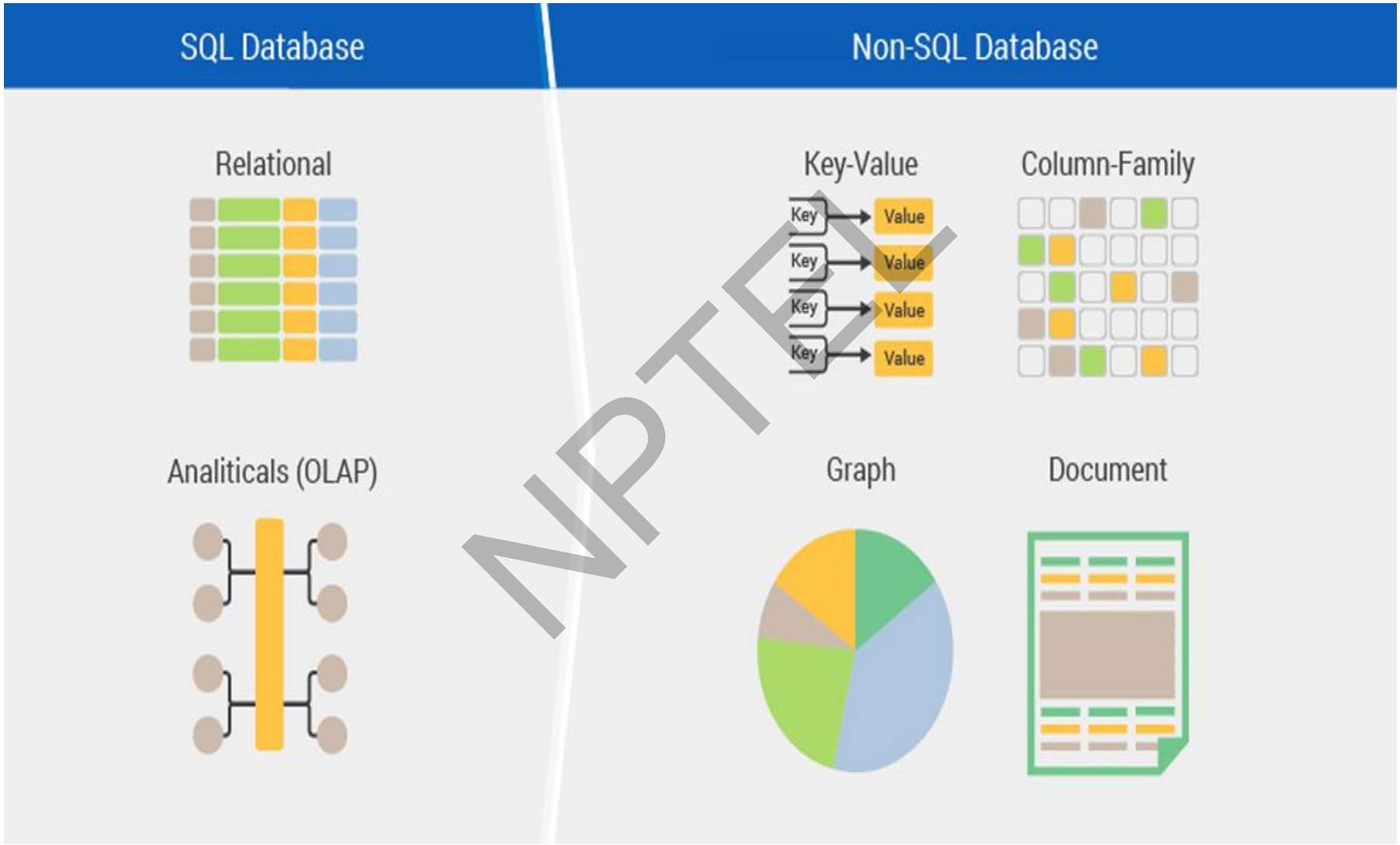
NoSQL

- While the traditional SQL can be effectively used to handle large amount of structured data, we need **NoSQL (Not Only SQL) to handle unstructured data.**

- NoSQL databases store unstructured data with no particular schema

- Each row can have its own set of column values. NoSQL gives better performance in storing massive amount of data.

NoSQL


SQL Database
Non-SQL Database
Relational
Key-Value
Key
Value
Key
Value
Key
Value
Key
Value
Column-Family
Analiticals (OLAP)
Graph
Document

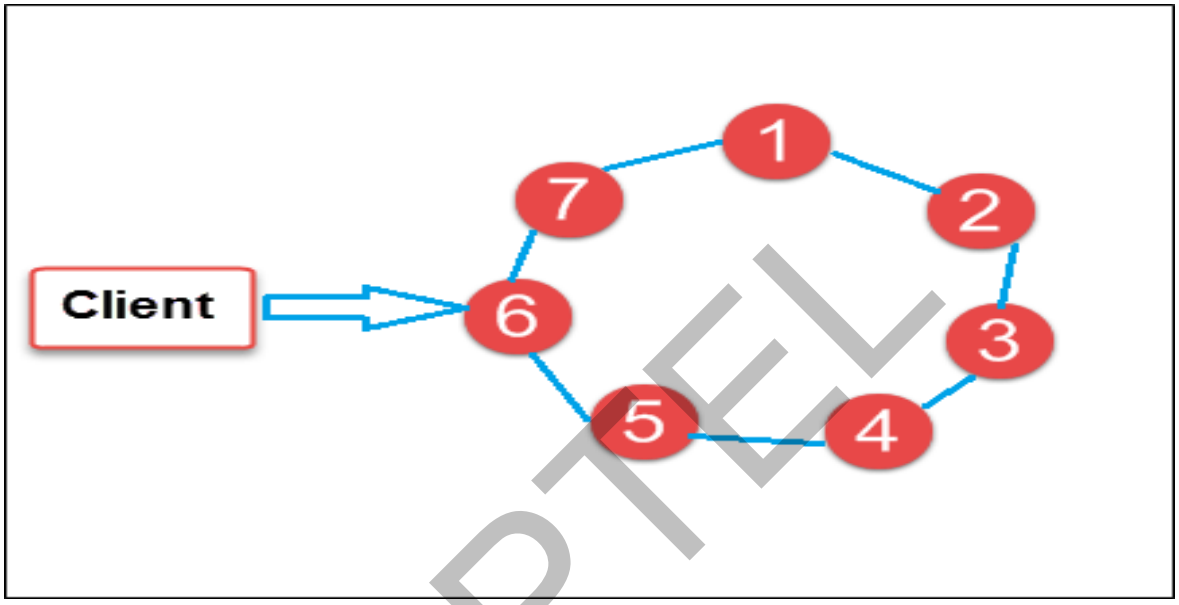
Cassandra

- **Apache Cassandra** is highly scalable, distributed and high-performance NoSQL database. Cassandra is designed to handle a huge amount of data.

- Cassandra handles the huge amount of data with its distributed architecture.

- Data is placed on different machines with more than one replication factor that provides high availability and no single point of failure.

Cassandra

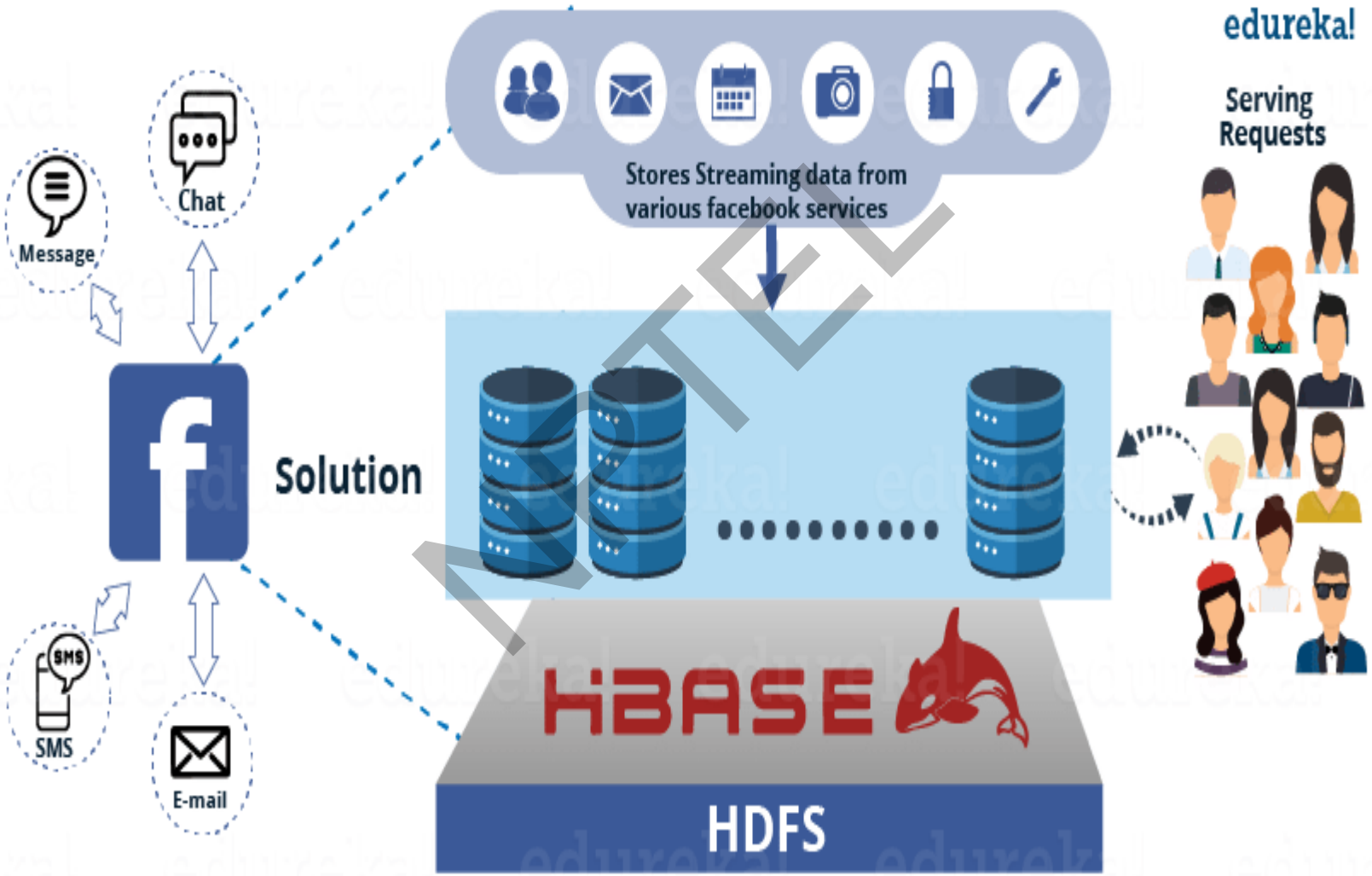


In the image above, circles are Cassandra nodes and lines between the circles shows distributed architecture, while the client is sending data to the node

HBase

- **HBase** is an open source, distributed database, developed by Apache Software foundation.
- Initially, it was Google Big Table, afterwards it was re-named as HBase and is primarily written in Java.
- HBase can store massive amounts of data from terabytes to petabytes.

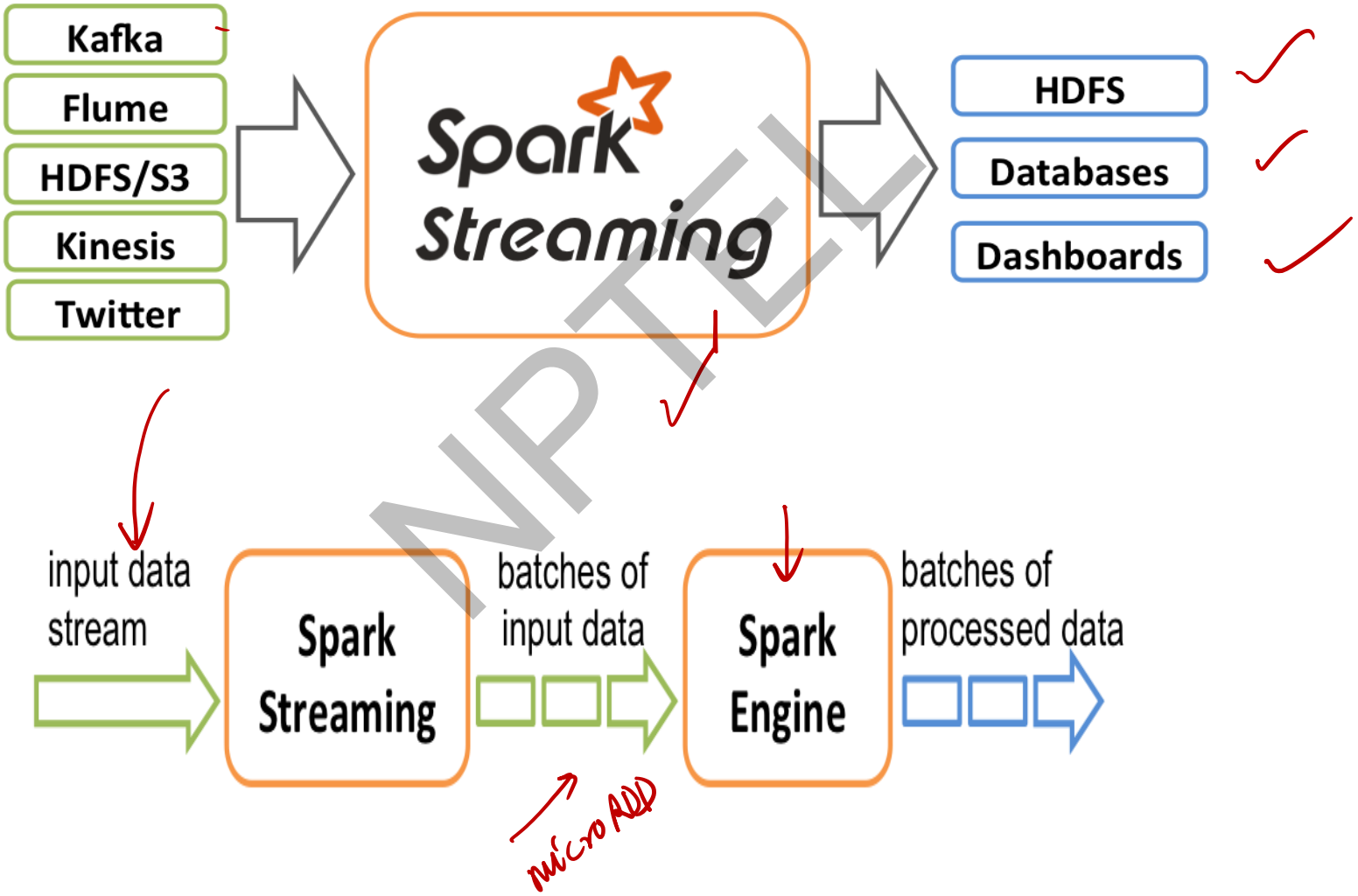
HBase


edureka!
Serving Requests
Stores Streaming data from various facebook services
Chat
Message
Solution
SMS
E-mail
HBASE
HDFS

Spark Streaming

- **Spark Streaming** is an extension of the core Spark API that enables scalable, high-throughput, fault-tolerant stream processing of live data streams.
- Streaming data input from HDFS, Kafka, Flume, TCP sockets, Kinesis, etc.
- Spark ML (Machine Learning) functions and GraphX graph processing algorithms are fully applicable to streaming data .

Spark Streaming



Kafka, Streaming Ecosystem

- **Apache Kafka** is an open-source stream-processing software platform developed by the Apache Software Foundation written in Scala and Java.

- Apache Kafka is an open source distributed streaming platform capable of handling trillions of events a day, Kafka is based on an abstraction of a distributed commit log

Kafka

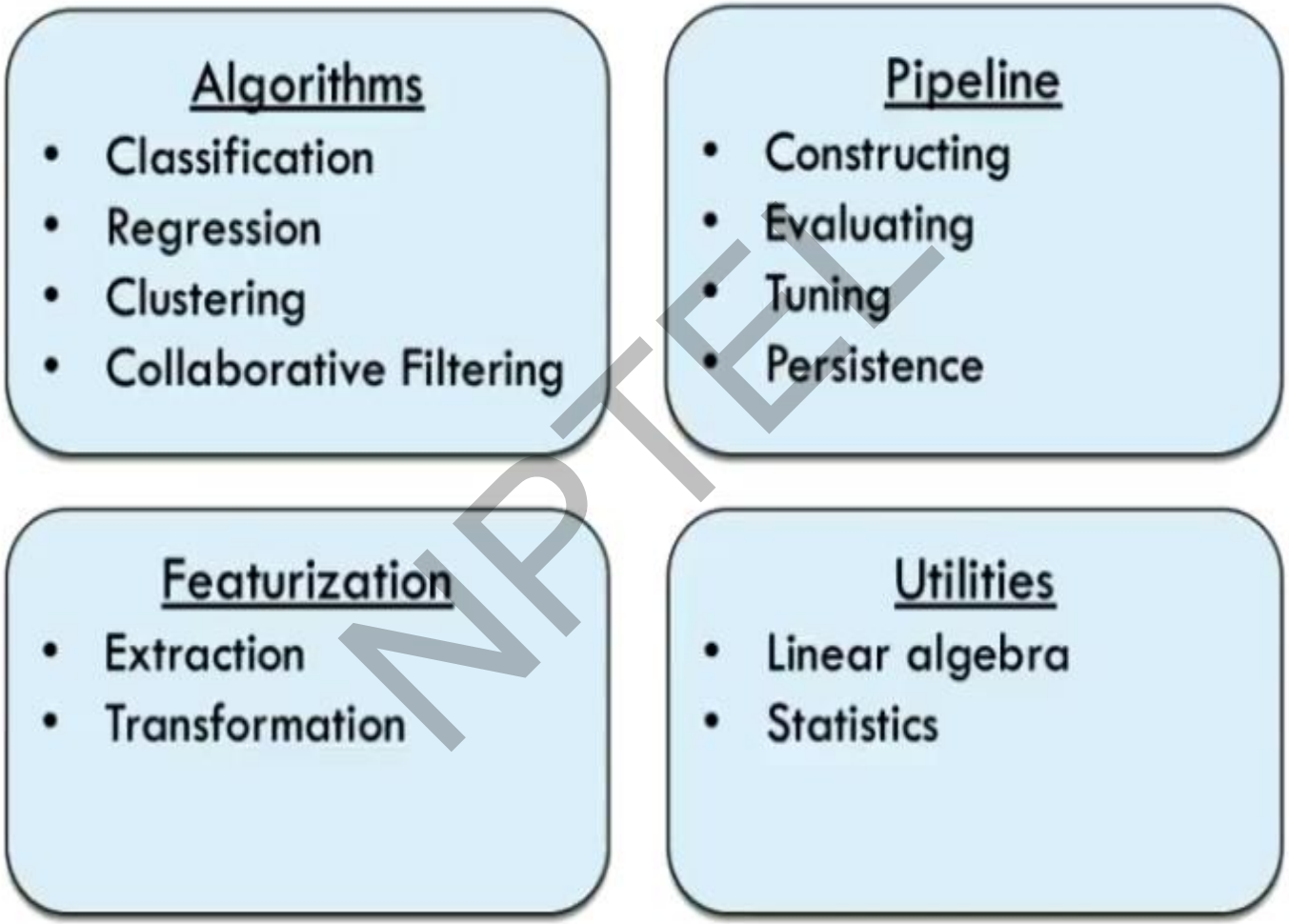

DATA STREAMS
kafka
APACHE
Spark
STRUCTURED
STREAMING
Console
Memory
Files

Spark MLlib

- **Spark MLlib** is a distributed machine-learning framework on top of Spark Core.

- MLlib is Spark's scalable machine learning library consisting of common learning algorithms and utilities, including classification, regression, clustering, collaborative filtering, dimensionality reduction.

Spark MLlib Component

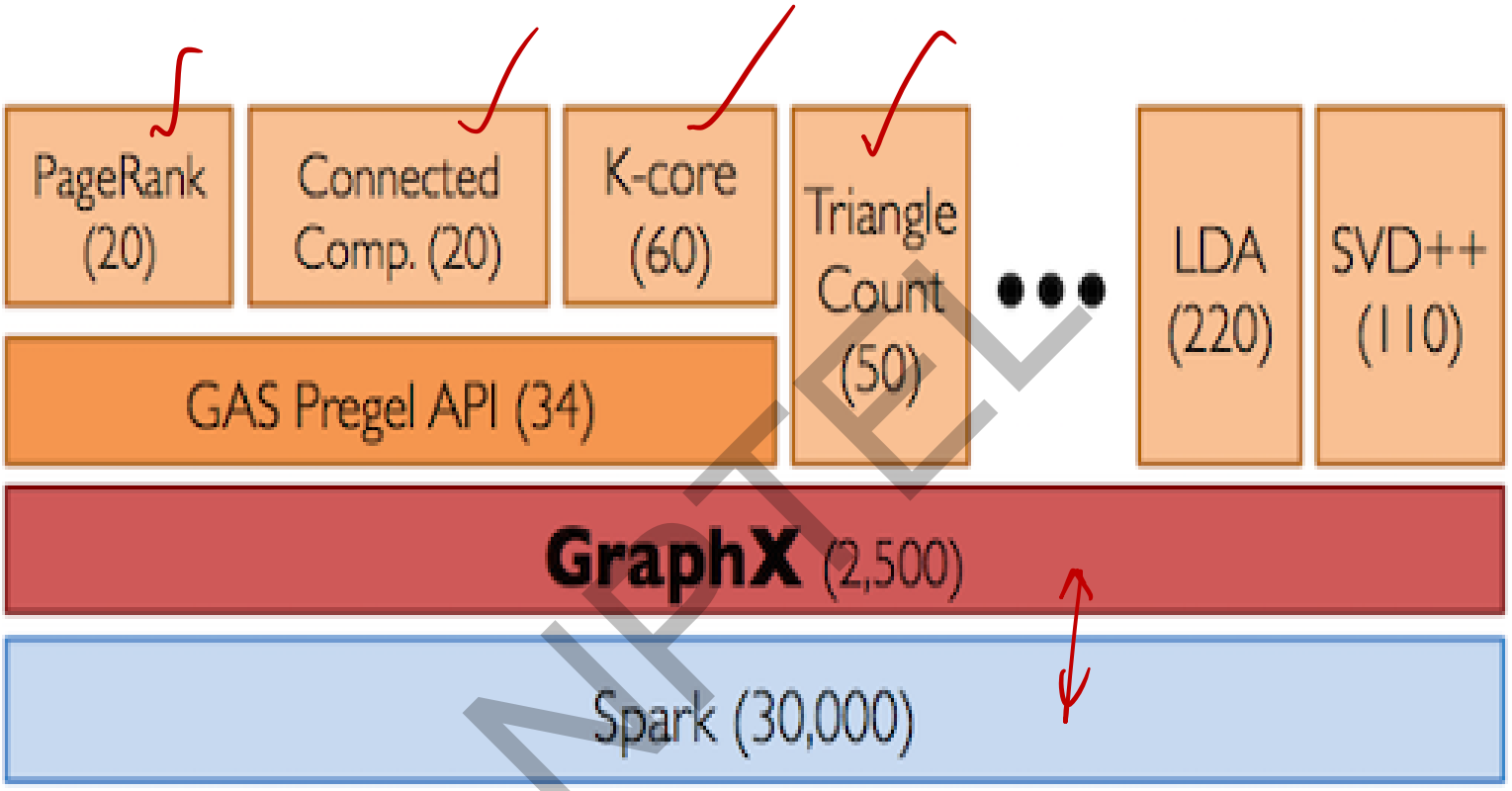


Spark GraphX

- **GraphX** is a new component in Spark for graphs and graph-parallel computation. At a high level, GraphX extends the Spark RDD by introducing a new graph abstraction.

- GraphX reuses Spark RDD concept, simplifies graph analytics tasks, provides the ability to make operations on a directed multigraph with properties attached to each vertex and edge.

Spark GraphX



GraphX is a thin layer on top of the Spark general-purpose dataflow framework (lines of code).

Conclusion

- In this lecture, we given a brief overview of following Big Data Enabling Technologies:
  - Apache Hadoop
  - Hadoop Ecosystem
  - HDFS Architecture
  - YARN
  - NoSQL
  - Hive
  - Map Reduce
  - Apache Spark
  - Zookeeper
  - Cassandra
  - Hbase
  - Spark Streaming
  - Kafka
  - Spark MLlib
  - GraphX

Hadoop Stack for Big Data

Dr. Rajiv Misra
Dept. of Computer Science & Engg.
Indian Institute of Technology Patna
rajivm@iitp.ac.in

Preface

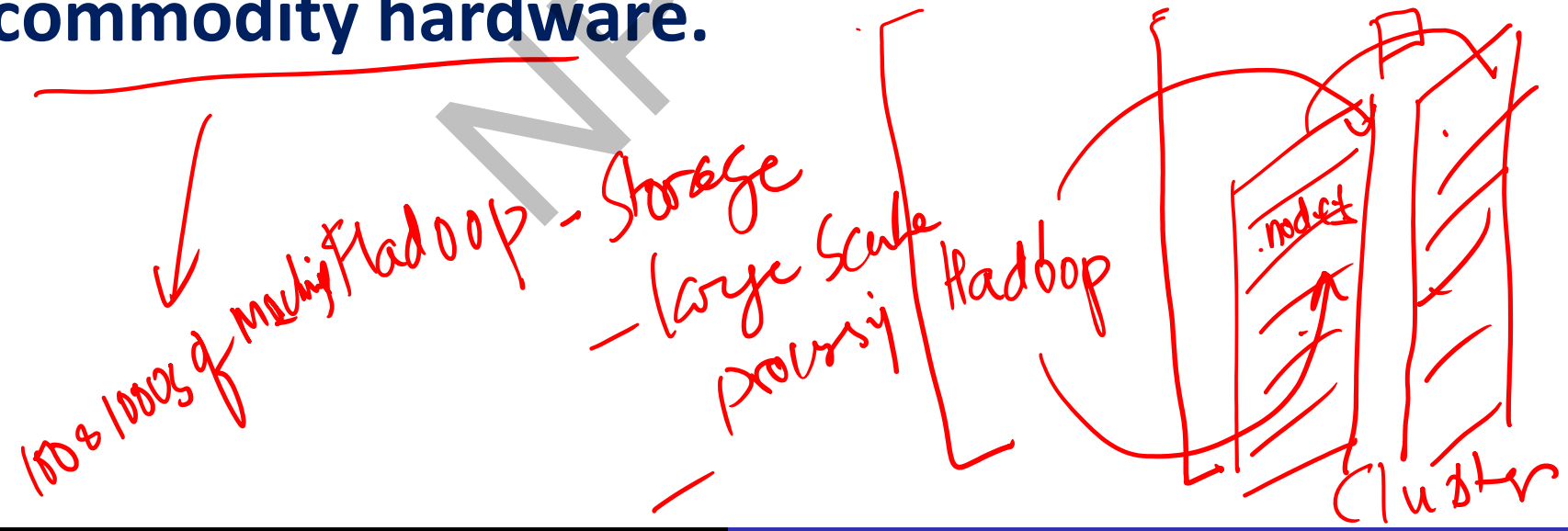
Content of this Lecture:

- In this lecture, we will provide insight into Hadoop technologies opportunities and challenges for Big Data.
- We will also look into the Hadoop stack and applications and technologies associated with Big Data solutions.

Hadoop Beginnings

What is Hadoop ?

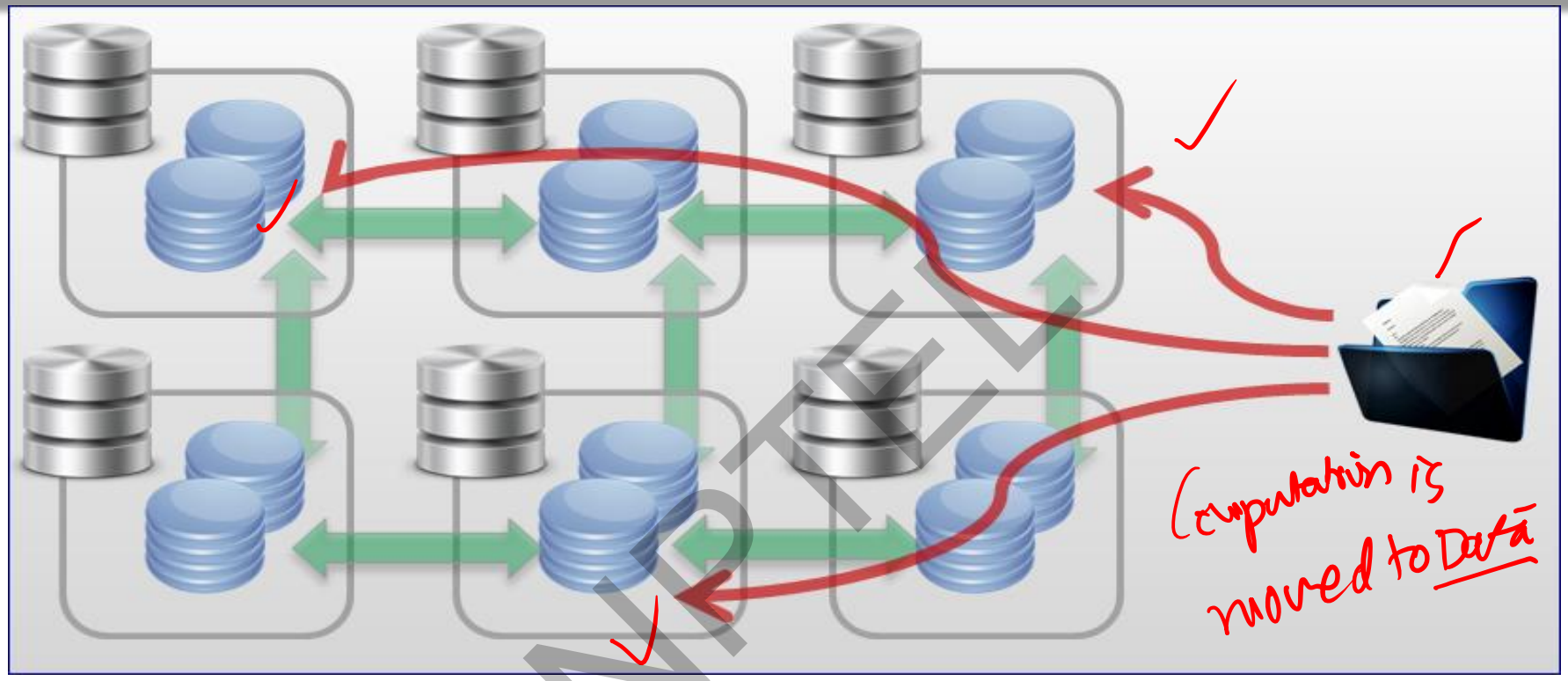
- **Apache Hadoop is an open source software framework for storage and large scale processing of the data-sets on clusters of commodity hardware.**



Hadoop Beginnings

- Hadoop was created by Doug Cutting and Mike Cafarella in 2005
- It was originally developed to support distribution of the Nutch Search Engine Project.
- Doug, who was working at Yahoo at the time, who is now actually a chief architect at Cloudera, has named this project after his son's toy elephant, Hadoop.

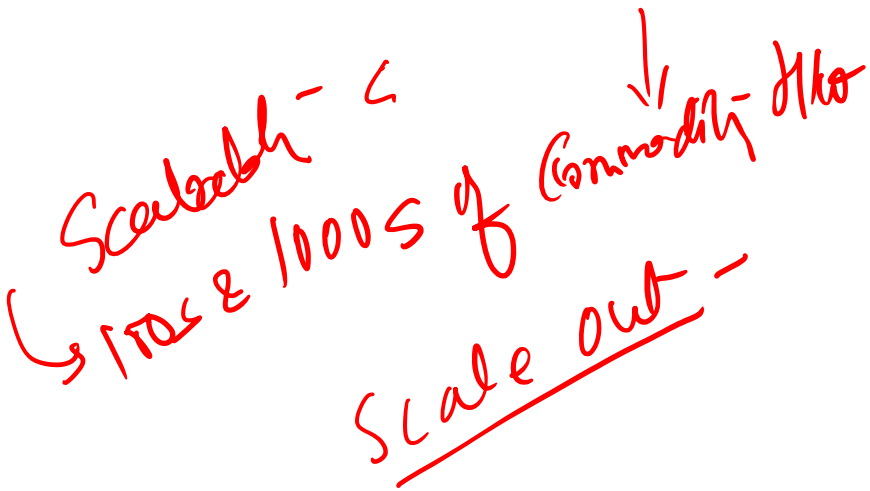
Moving Computation to Data



- Hadoop started out as a simple batch processing framework.
- The idea behind Hadoop is that instead of moving data to computation, we move computation to data.

Scalability

- Scalability's at it's core of a Hadoop system.
- We have cheap computing storage.
- We can distribute and scale across very easily in a very cost effective manner.



Reliability

- Hardware Failures Handles Automatically!

failure is norm!

- If we think about an individual machine or rack of machines, or a large cluster or super computer, they all fail at some point of time or some of their components will fail. These failures are so common that we have to account for them ahead of the time.
- And all of these are actually handled within the Hadoop framework system. So the Apache's Hadoop MapReduce and HDFS components were originally derived from the Google's MapReduce and Google's file system. Another very interesting thing that Hadoop brings is a new approach to data.

New Approach to Data: Keep all data

- A new approach is, we can keep all the data that we have, and we can take that data and analyze it in new interesting ways. We can do something that's called schema and read style.
- And we can actually allow new analysis. We can bring more data into simple algorithms, which has shown that with more granularity, you can actually achieve often better results in taking a small amount of data and then some really complex analytics on it.

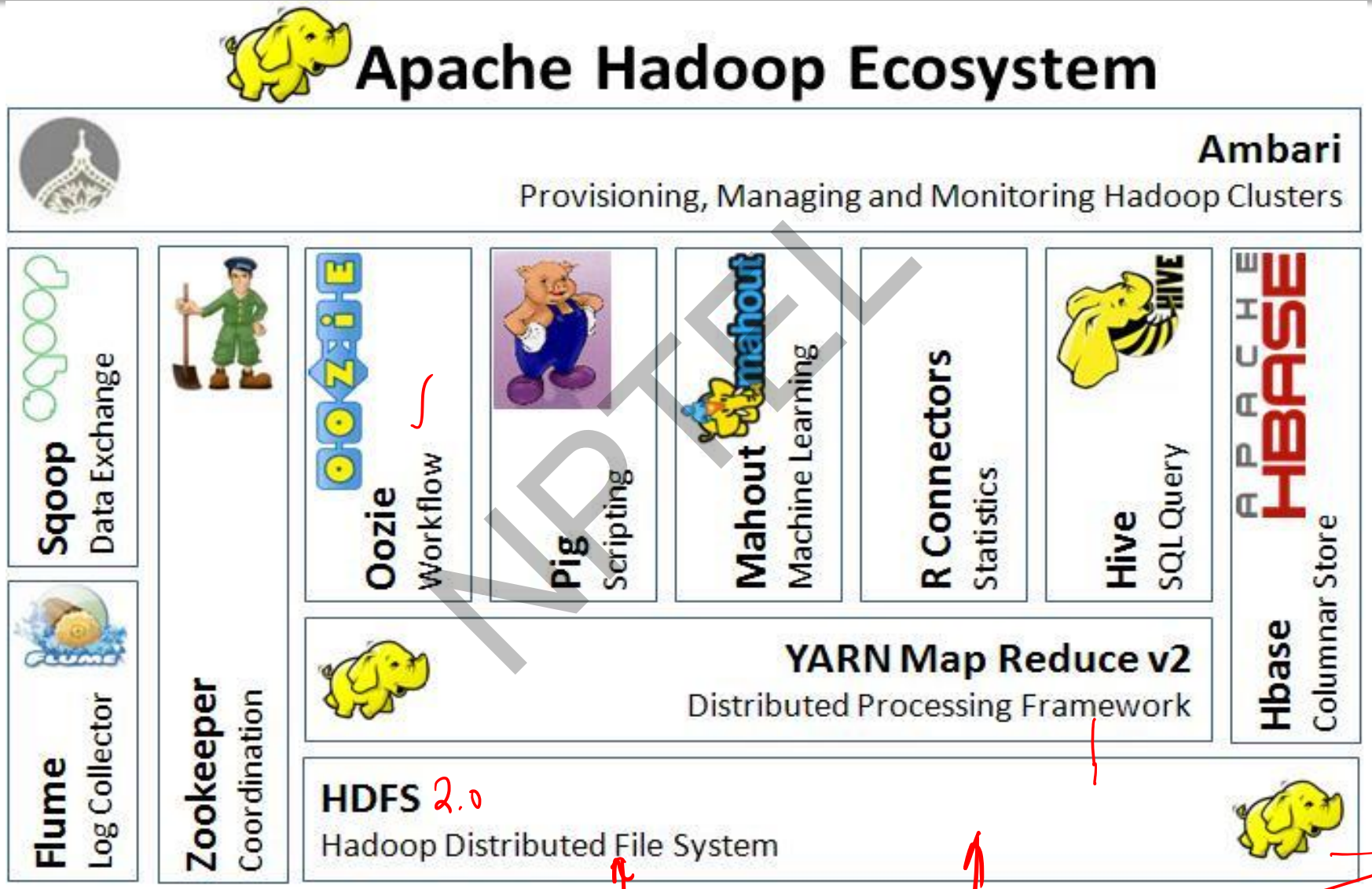
Apache Hadoop Framework & its Basic Modules

Summary
Design issues in Hadoop
1. Reliability
2. Scalable
3. Keeping all data ready on scheduling
4. moving computation to the data

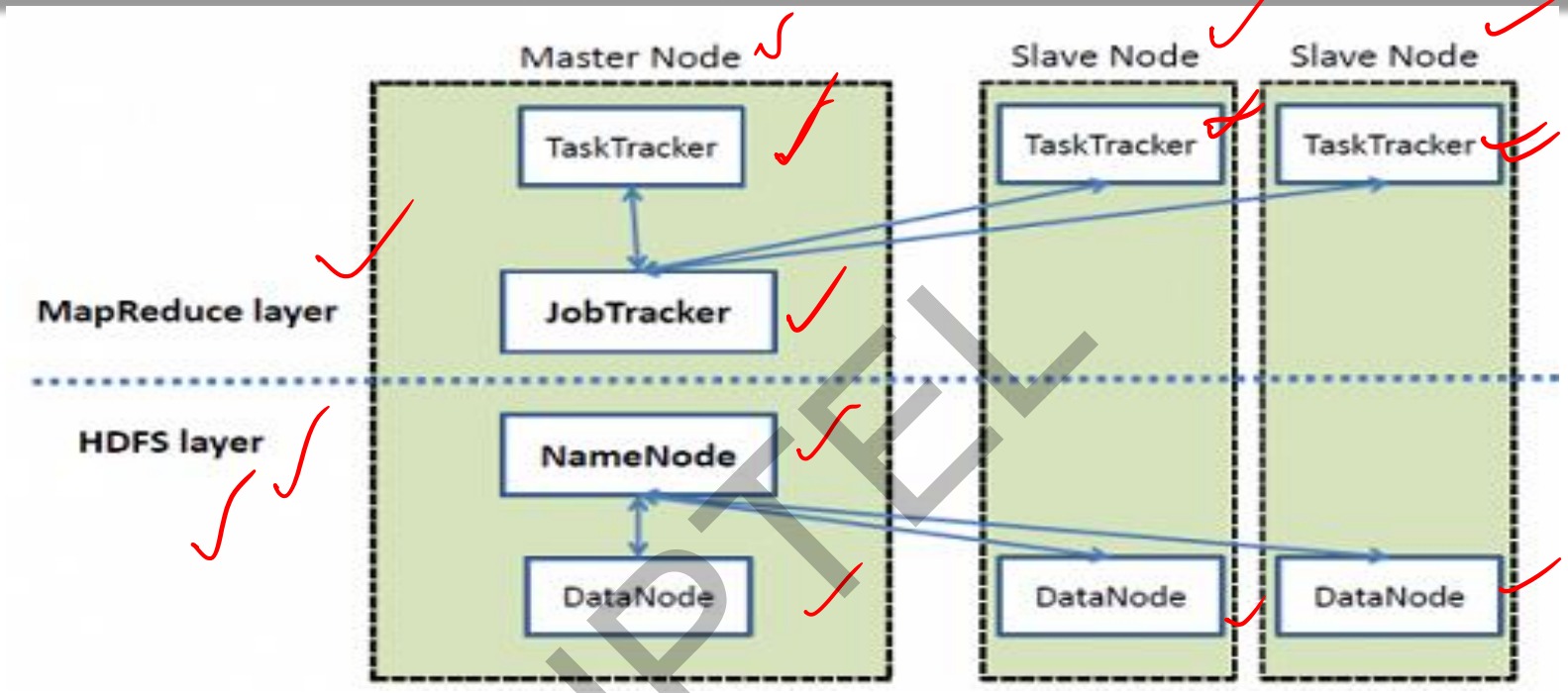
Apache Framework Basic Modules

- **Hadoop Common:** It contains libraries and utilities needed by other Hadoop modules.
- **Hadoop Distributed File System (HDFS):** It is a distributed file system that stores data on a commodity machine. Providing very high aggregate bandwidth across the entire cluster.
- **Hadoop YARN:** It is a resource management platform responsible for managing compute resources in the cluster and using them in order to schedule users and applications.
- **Hadoop MapReduce:** It is a programming model that scales data across a lot of different processes.

Apache Framework Basic Modules



High Level Architecture of Hadoop



- Two major pieces of Hadoop are: Hadoop Distribute the File System and the MapReduce, a parallel processing framework that will map and reduce data. These are both open source and inspired by the technologies developed at Google.

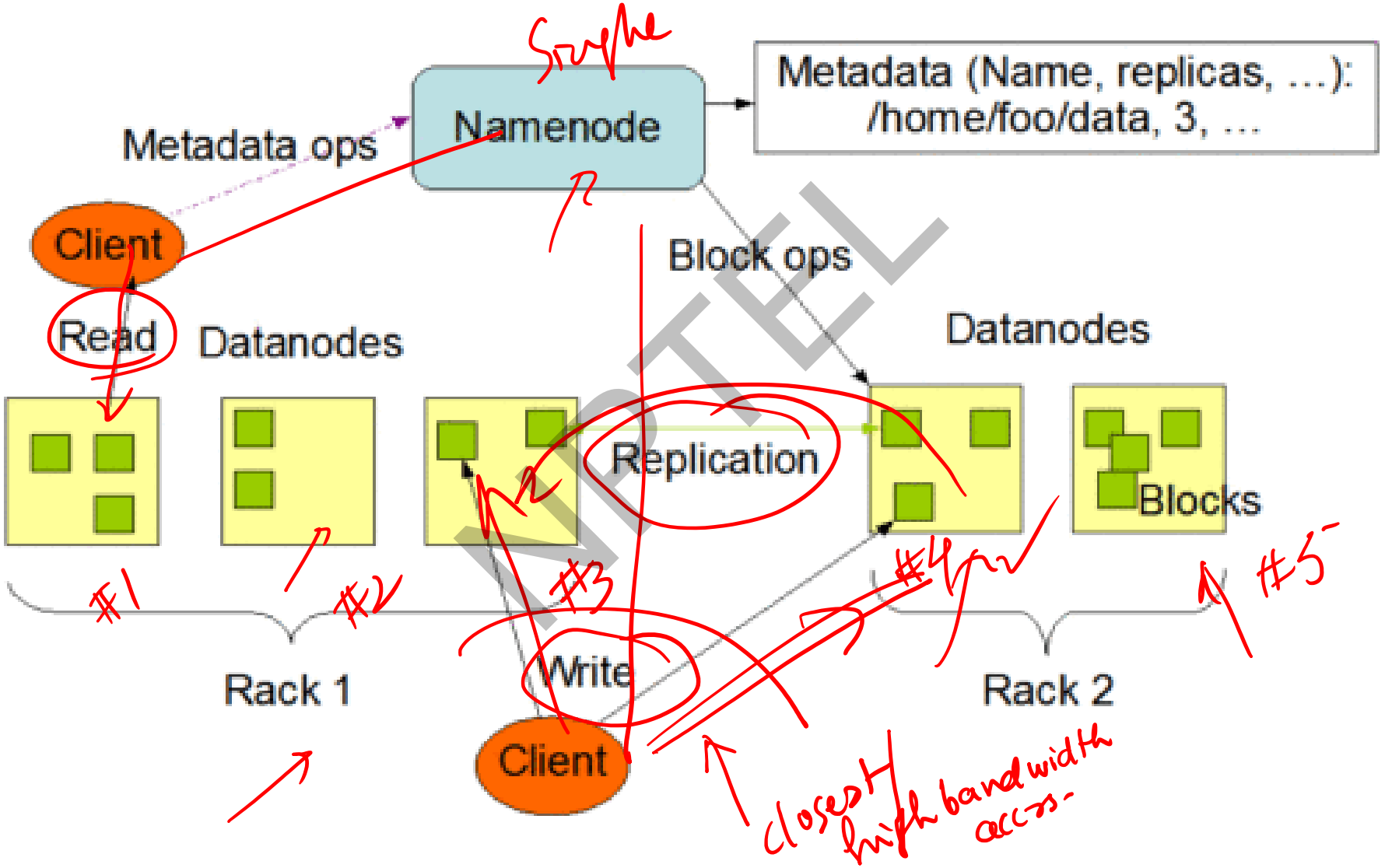
- If we talk about this high level infrastructure, we start talking about things like TaskTrackers and JobTrackers, the NameNodes and DataNodes.

HDFS
Hadoop distributed file system

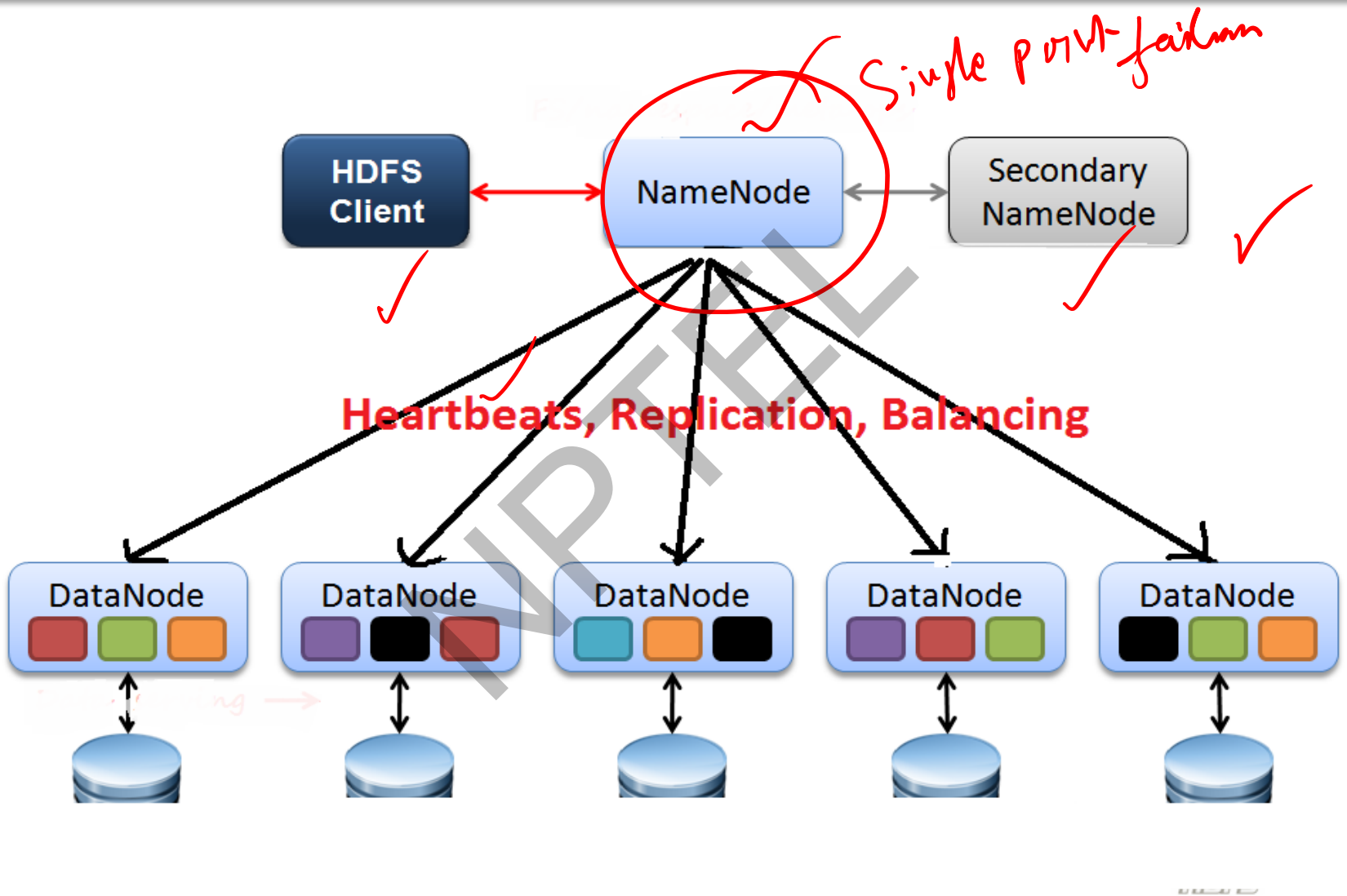
HDFS: Hadoop distributed file system

- Distributed, scalable, and portable file-system written in Java for the Hadoop framework.
- Each node in Hadoop instance typically has a single name node, and a cluster of data nodes that formed this HDFS cluster.
- Each HDFS stores large files, typically in ranges of gigabytes to terabytes, and now petabytes, across multiple machines. And it can achieve reliability by replicating the cross multiple hosts, and therefore does not require any range storage on hosts.

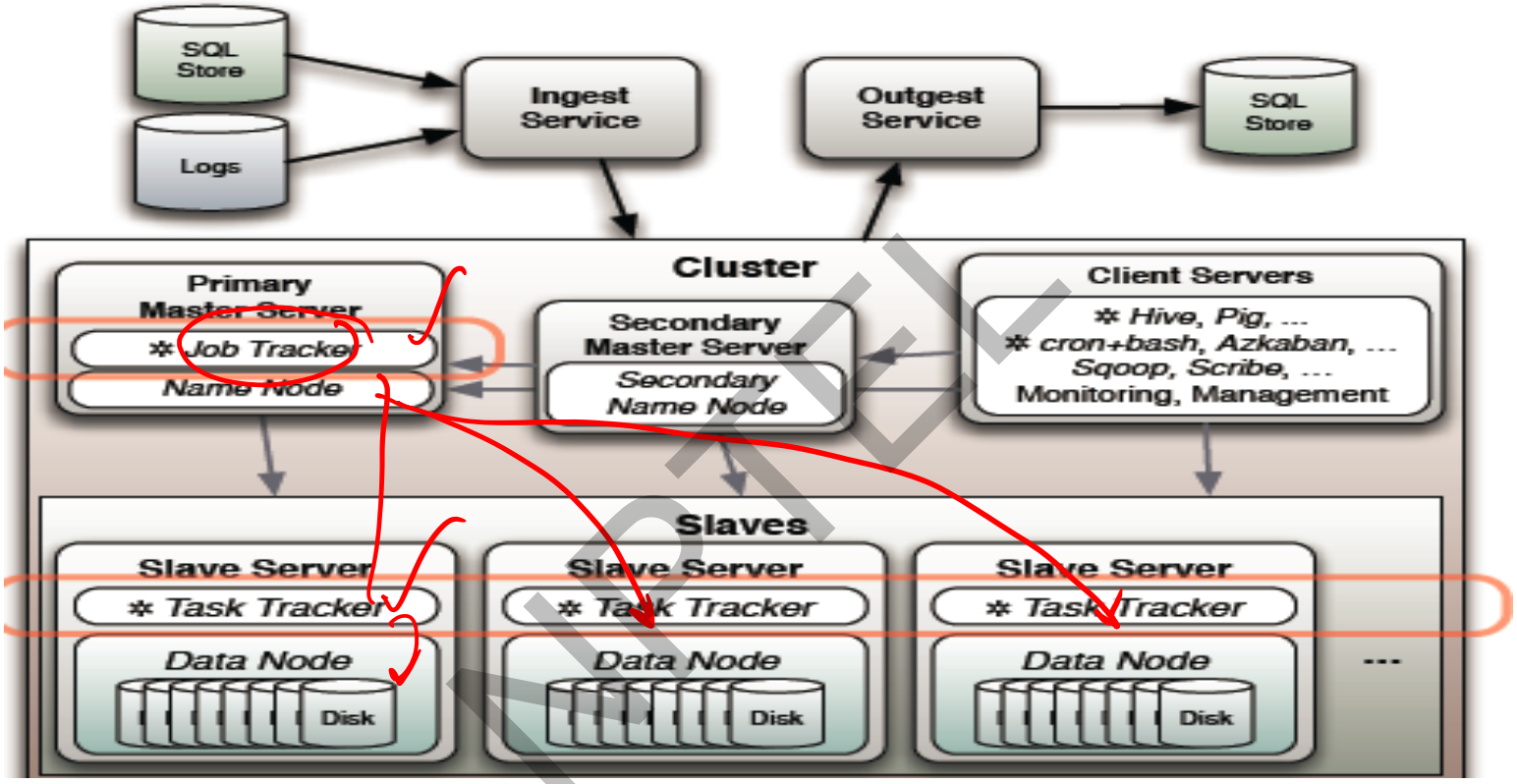
HDFS


Metadata ops
Namenode
Metadata (Name, replicas, …):
/home/foo/data, 3, …
Client
Block ops
Read
Datanodes
Datanodes
Replication
Blocks
Rack 1
Write
Rack 2
Client

HDFS


Single point failure
HDFS Client
NameNode
Secondary NameNode
Heartbeats, Replication, Balancing
DataNode
DataNode
DataNode
DataNode
DataNode

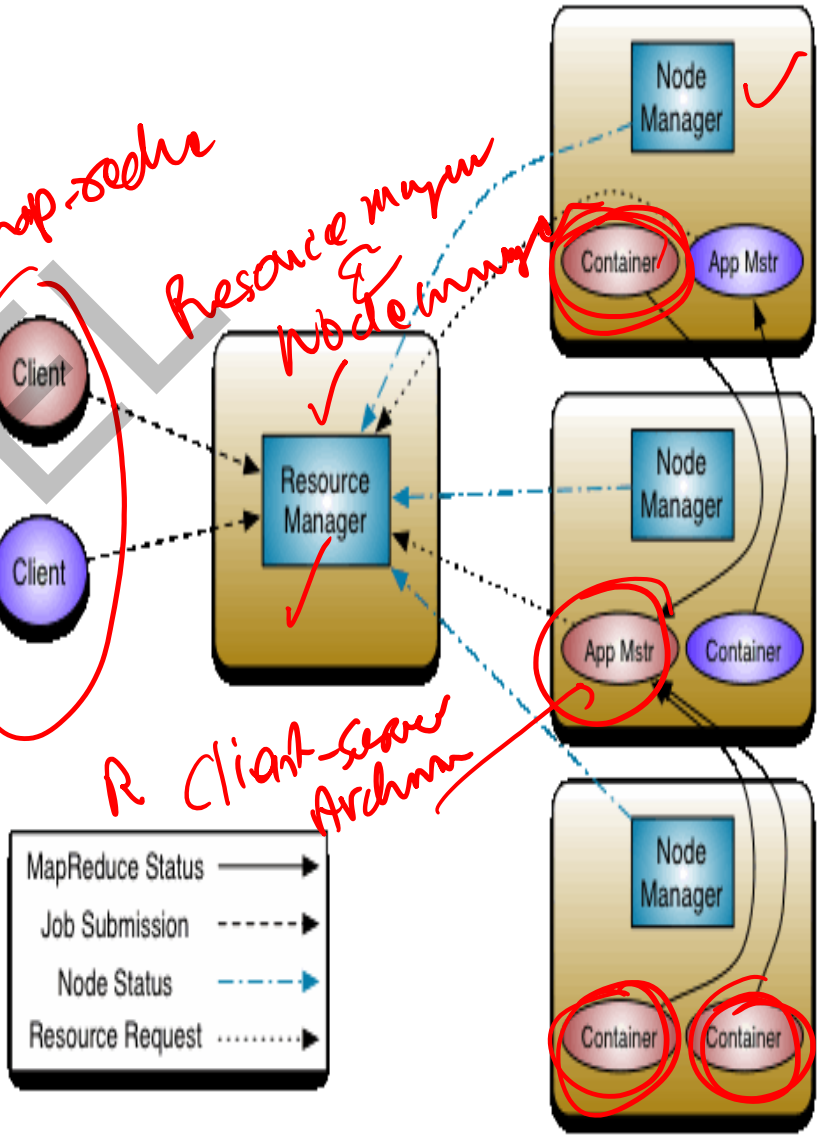
MapReduce Engine



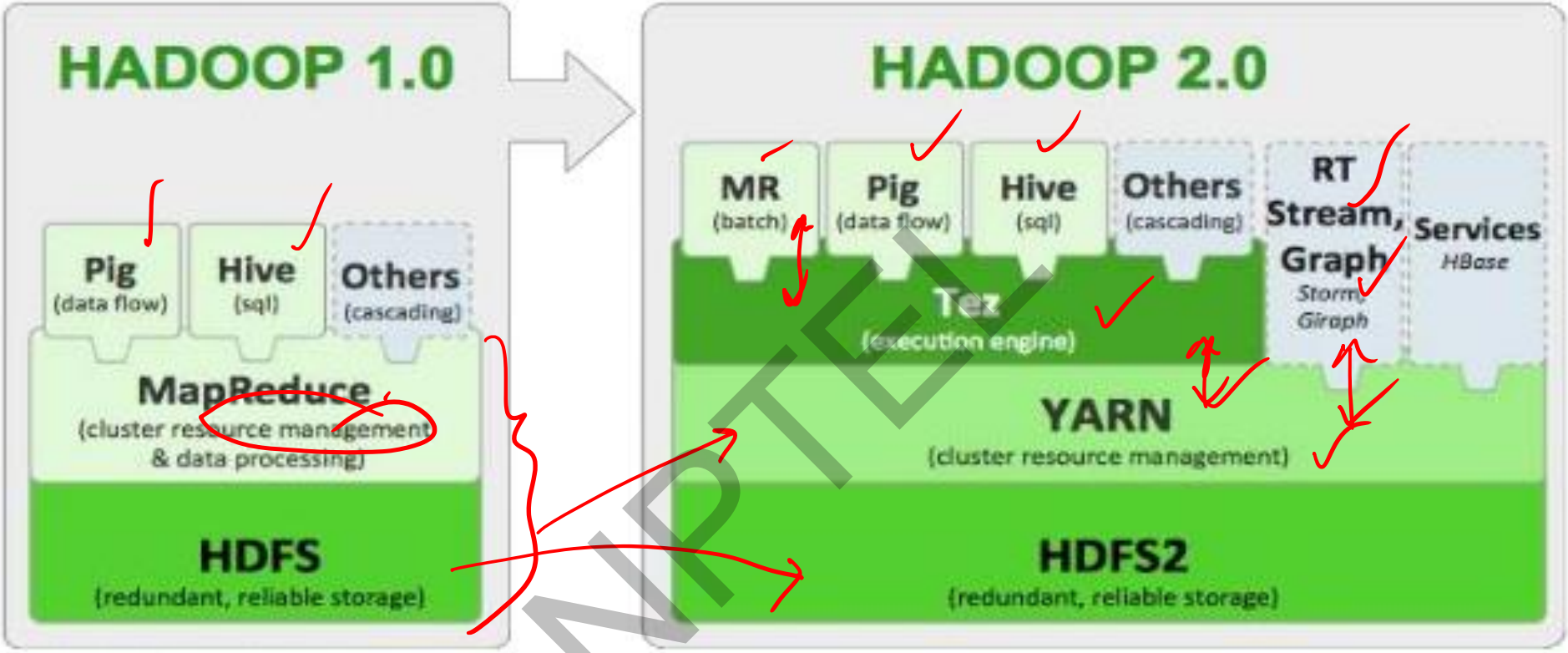
- The typical MapReduce engine will consist of a job tracker, to which client applications can submit MapReduce jobs, and this job tracker typically pushes work out to all the available task trackers, now it's in the cluster. Struggling to keep the word as close to the data as possible, as balanced as possible.

Apache Hadoop NextGen MapReduce (YARN)

- Yarn enhances the power of the Hadoop compute cluster, without being limited by the map produce kind of framework.
- It's scalability's great. The processing power and data centers continue to grow quickly, because the YARN research manager focuses exclusively on scheduling. It can manage those very large clusters quite quickly and easily.
- YARN is completely compatible with the MapReduce. Existing MapReduce application end users can run on top of the Yarn without disrupting any of their existing processes.



Hadoop 1.0 vs. Hadoop 2.0



- Hadoop 2.0 provides a more general processing platform, that is not constraining to this map and reduce kinds of processes.
- The fundamental idea behind the MapReduce 2.0 is to split up two major functionalities of the job tracker, resource management, and the job scheduling and monitoring, and to do two separate units. The idea is to have a global resource manager, and per application master manager.

What is Yarn ?

- Yarn enhances the power of the Hadoop compute cluster, without being limited by the map produce kind of framework.
- It's scalability's great. The processing power and data centers continue to grow quickly, because the YARN research manager focuses exclusively on scheduling. It can manage those very large clusters quite quickly and easily.
- YARN is completely compatible with the MapReduce. Existing MapReduce application end users can run on top of the Yarn without disrupting any of their existing processes.
- It does have a Improved cluster utilization as well. The resource manager is a pure schedule or they just optimize this cluster utilization according to the criteria such as capacity, guarantees, fairness, how to be fair, maybe different SLA's or service level agreements.

Scalability | **MapReduce Compatibility** | **Improved cluster utilization**

What is Yarn ?

- It supports other work flows other than just map reduce.
- Now we can bring in additional programming models, such as graph process or iterative modeling, and now it's possible to process the data in your base. This is especially useful when we talk about machine learning applications.
- Yarn allows multiple access engines, either open source or proprietary, to use Hadoop as a common standard for either batch or interactive processing, and even real time engines that can simultaneous acts as a lot of different data, so you can put streaming kind of applications on top of YARN inside a Hadoop architecture, and seamlessly work and communicate between these environments.

Fairness

Iterative Modeling

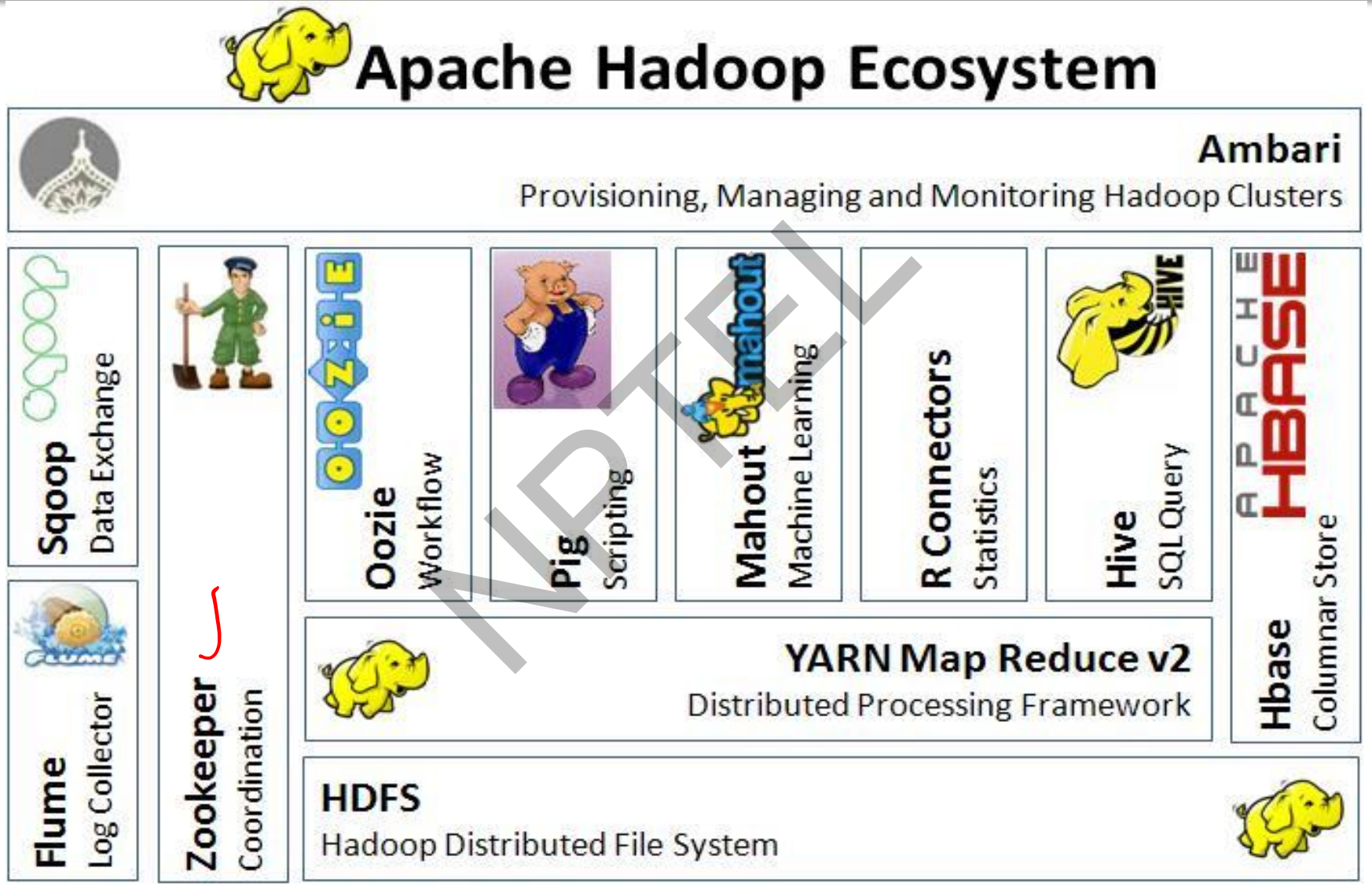
Machine Learning

Multiple Access Engines

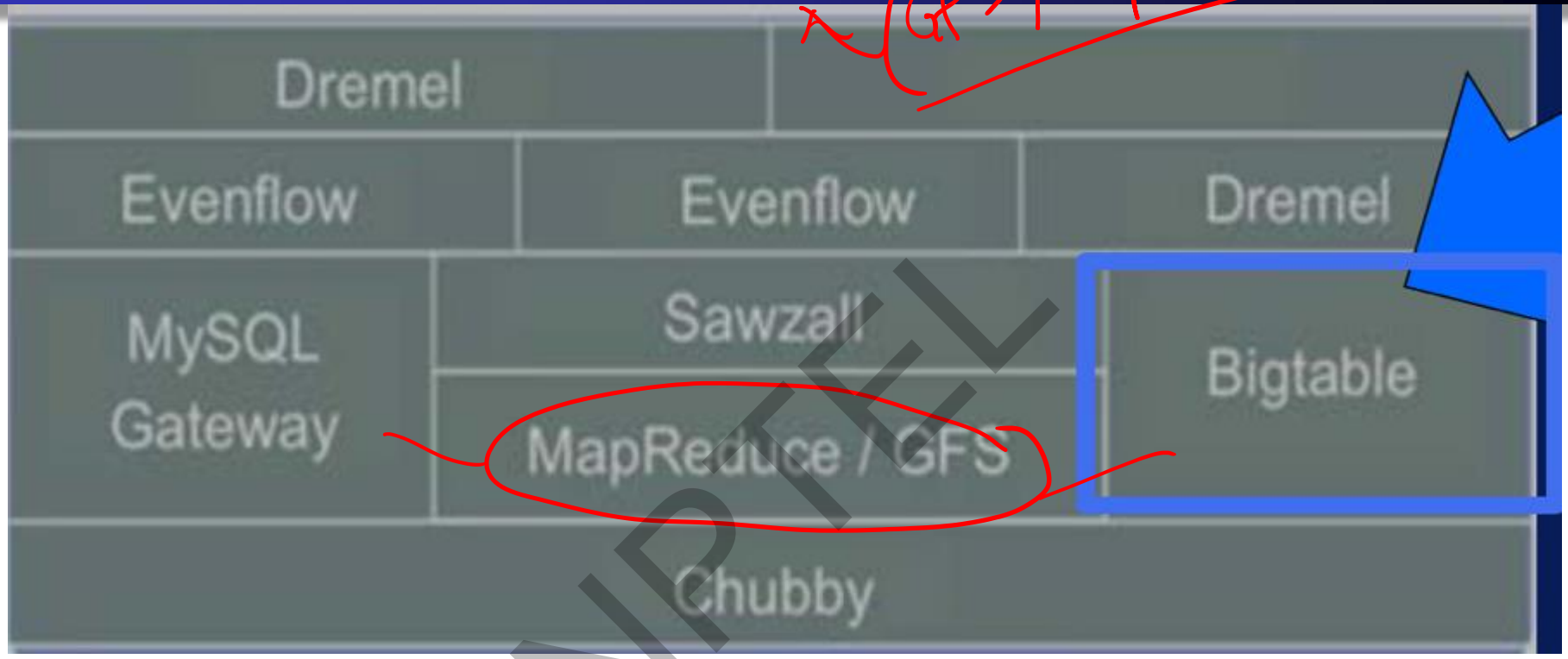
Supports Other Workloads

The Hadoop “Zoo”

Apache Hadoop Ecosystem

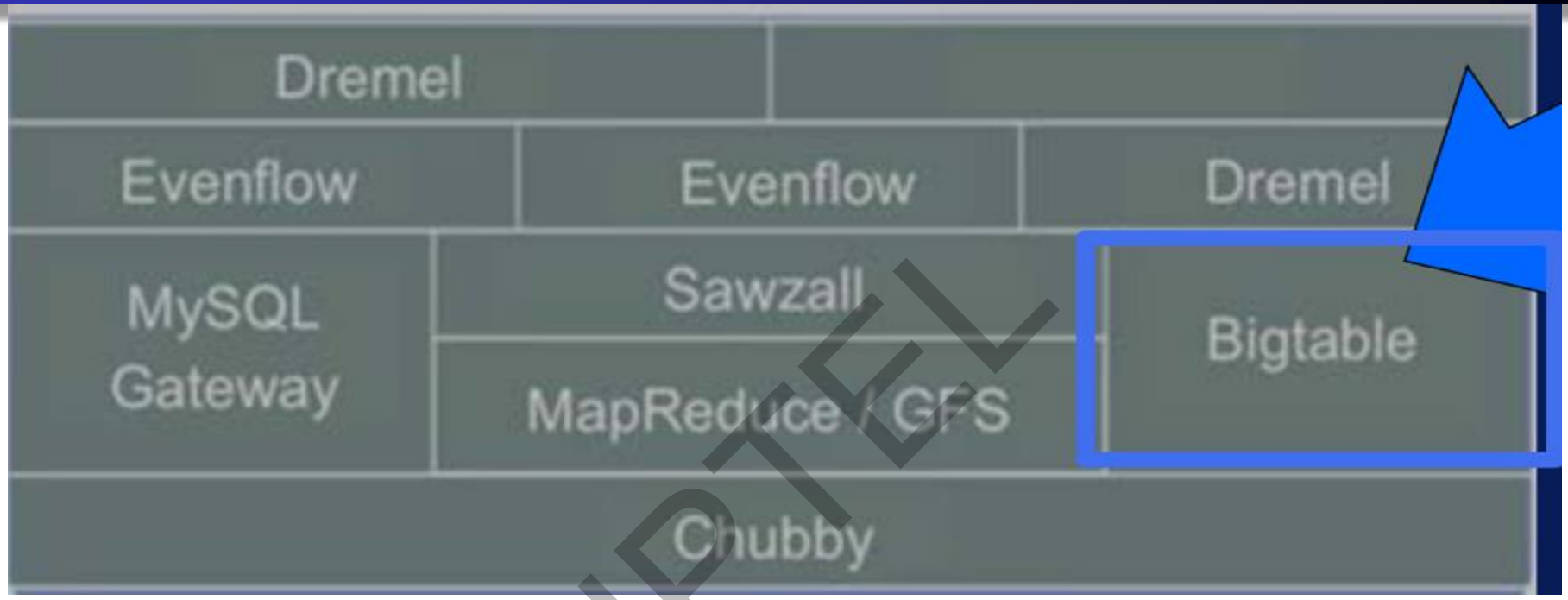


Original Google Stack



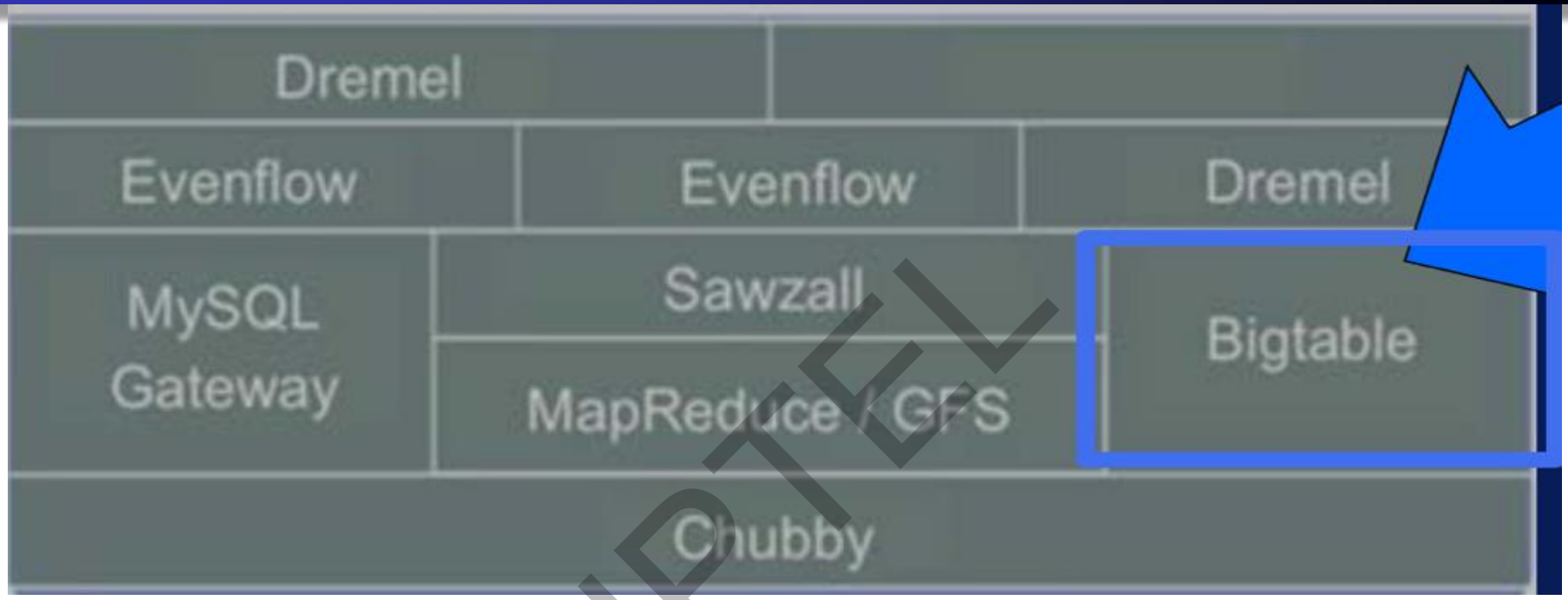
- Had their original MapReduce, and they were storing and processing large amounts of data.
- Like to be able to access that data and access it in a SQL like language. So they built the SQL gateway to adjust the data into the MapReduce cluster and be able to query some of that data as well.

Original Google Stack



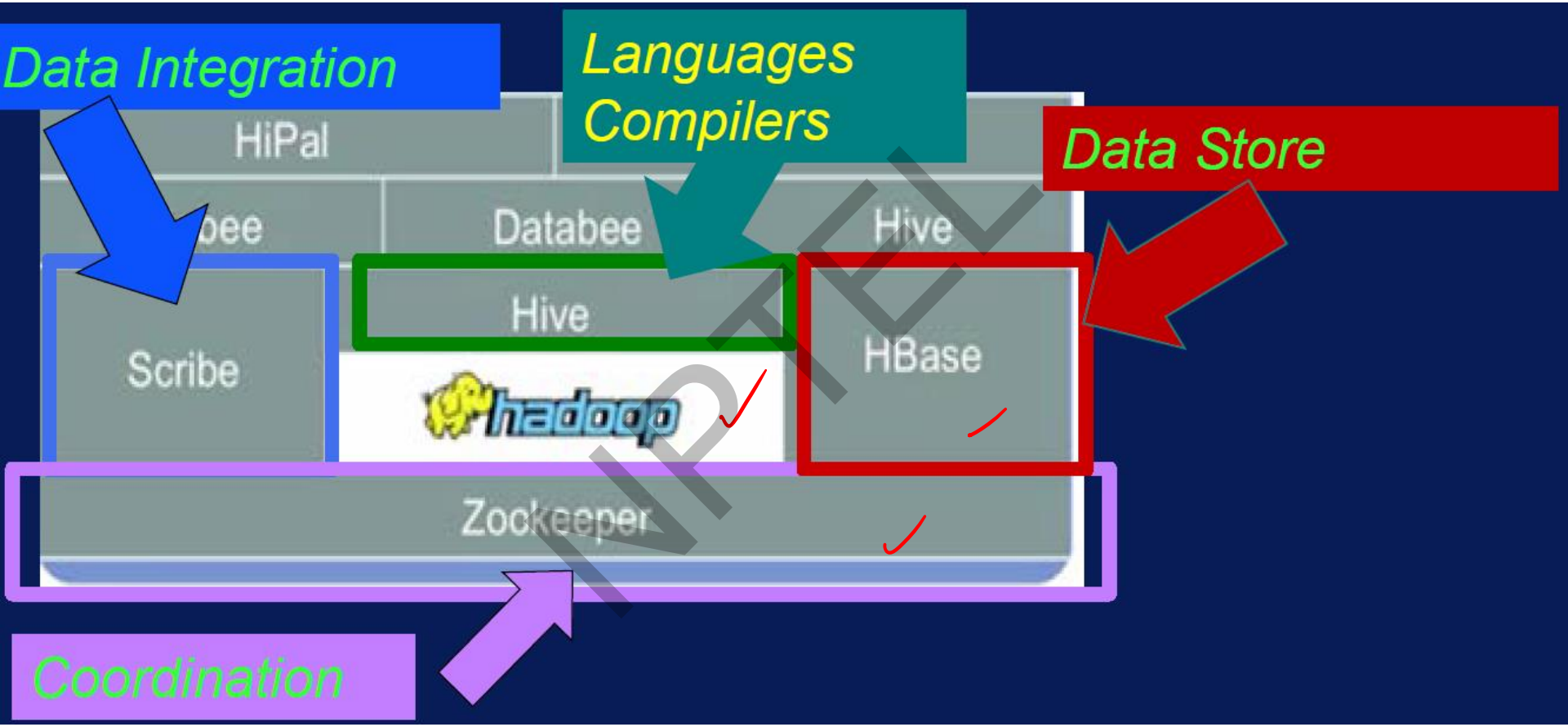
- Then, they realized they needed a high-level specific language to access MapReduce in the cluster and submit some of those jobs. So Sawzall came along.
- Then, Evenflow came along and allowed to chain together complex work codes and coordinate events and service across this kind of a framework or the specific cluster they had at the time.

Original Google Stack

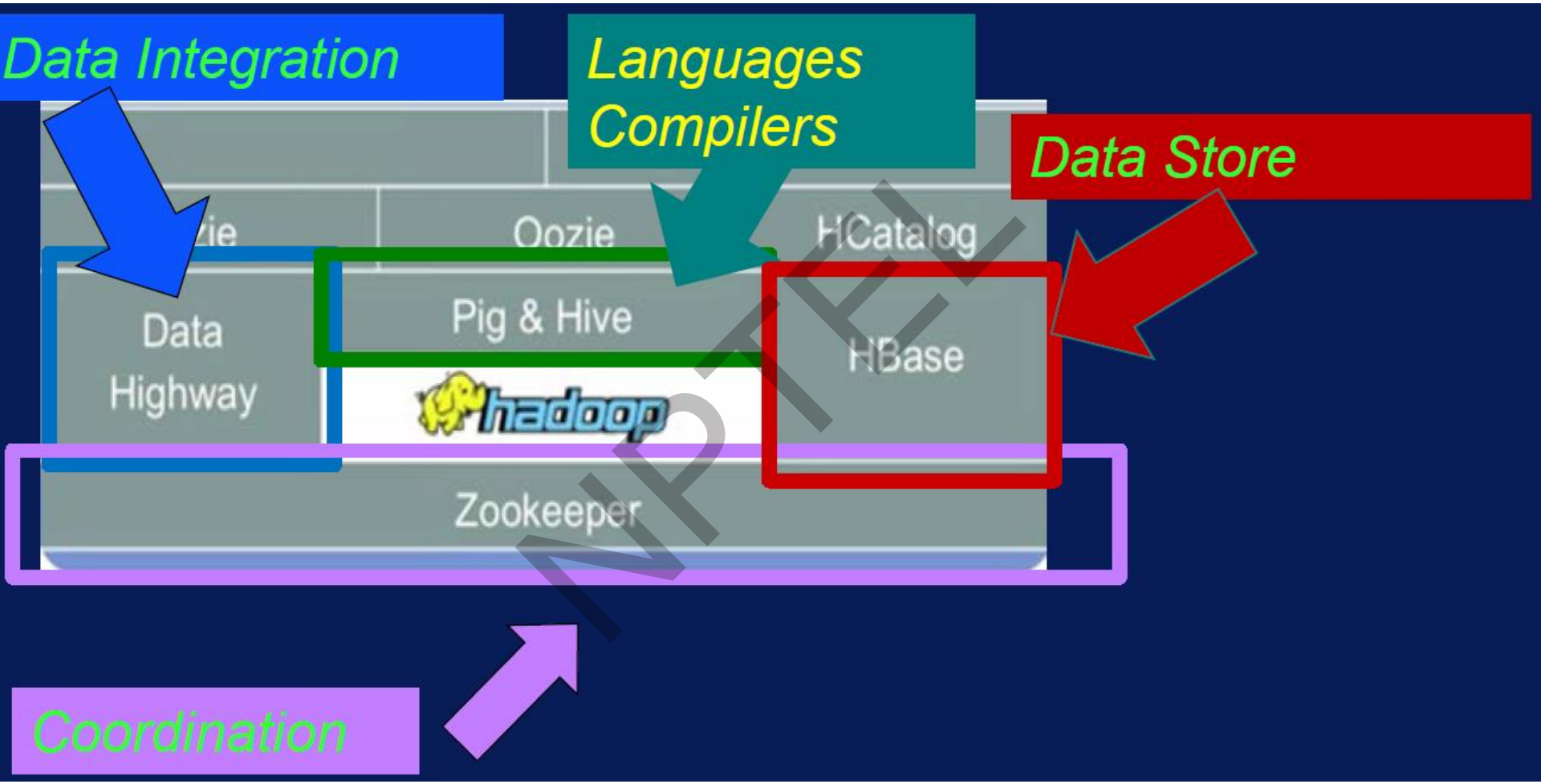


- Then, Dremel came along. Dremel was a columnar storage in the metadata manager that allows us to manage the data and is able to process a very large amount of unstructured data.
- Then Chubby came along as a coordination system that would manage all of the products in this one unit or one ecosystem that could process all these large amounts of structured data seamlessly.

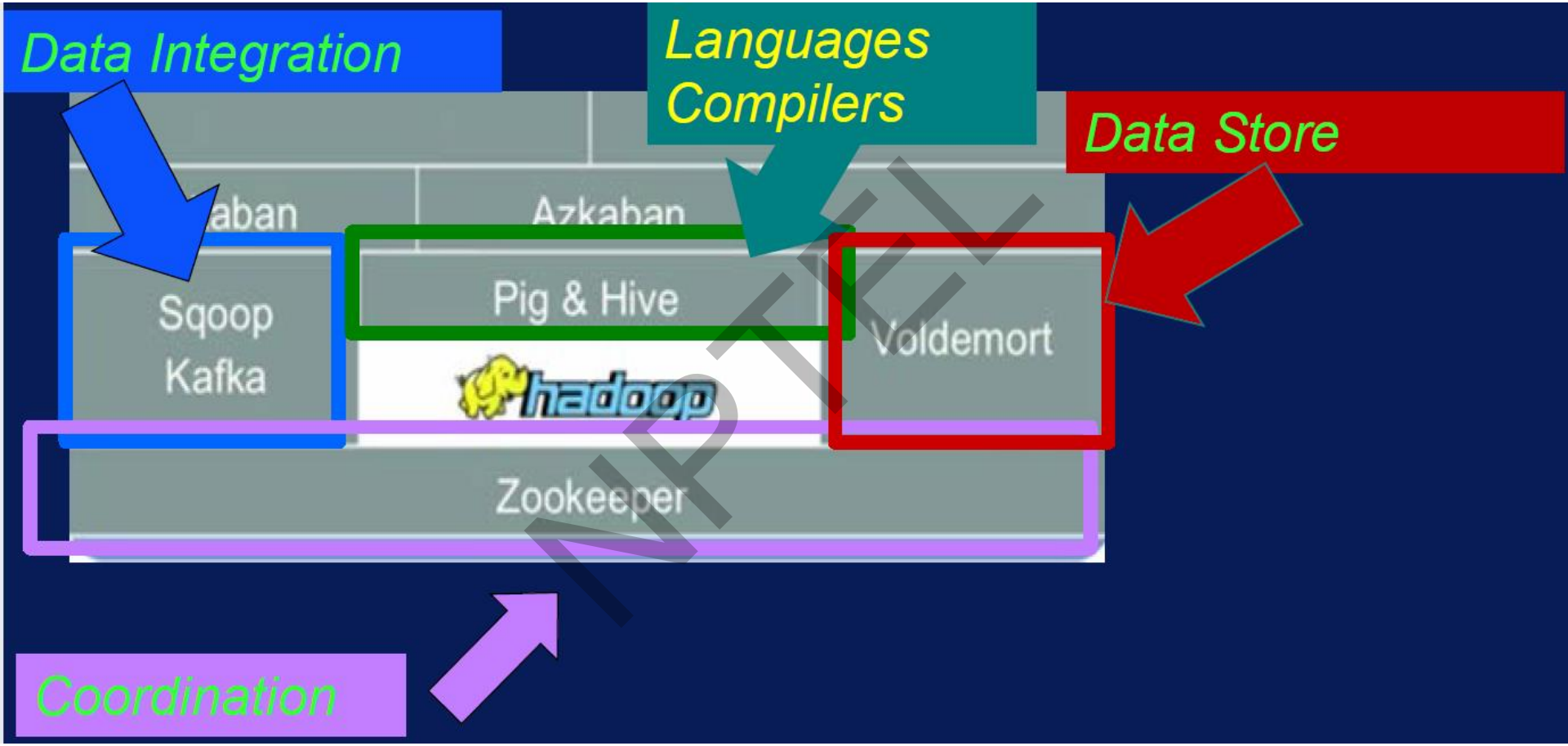
Facebook's Version of the Stack



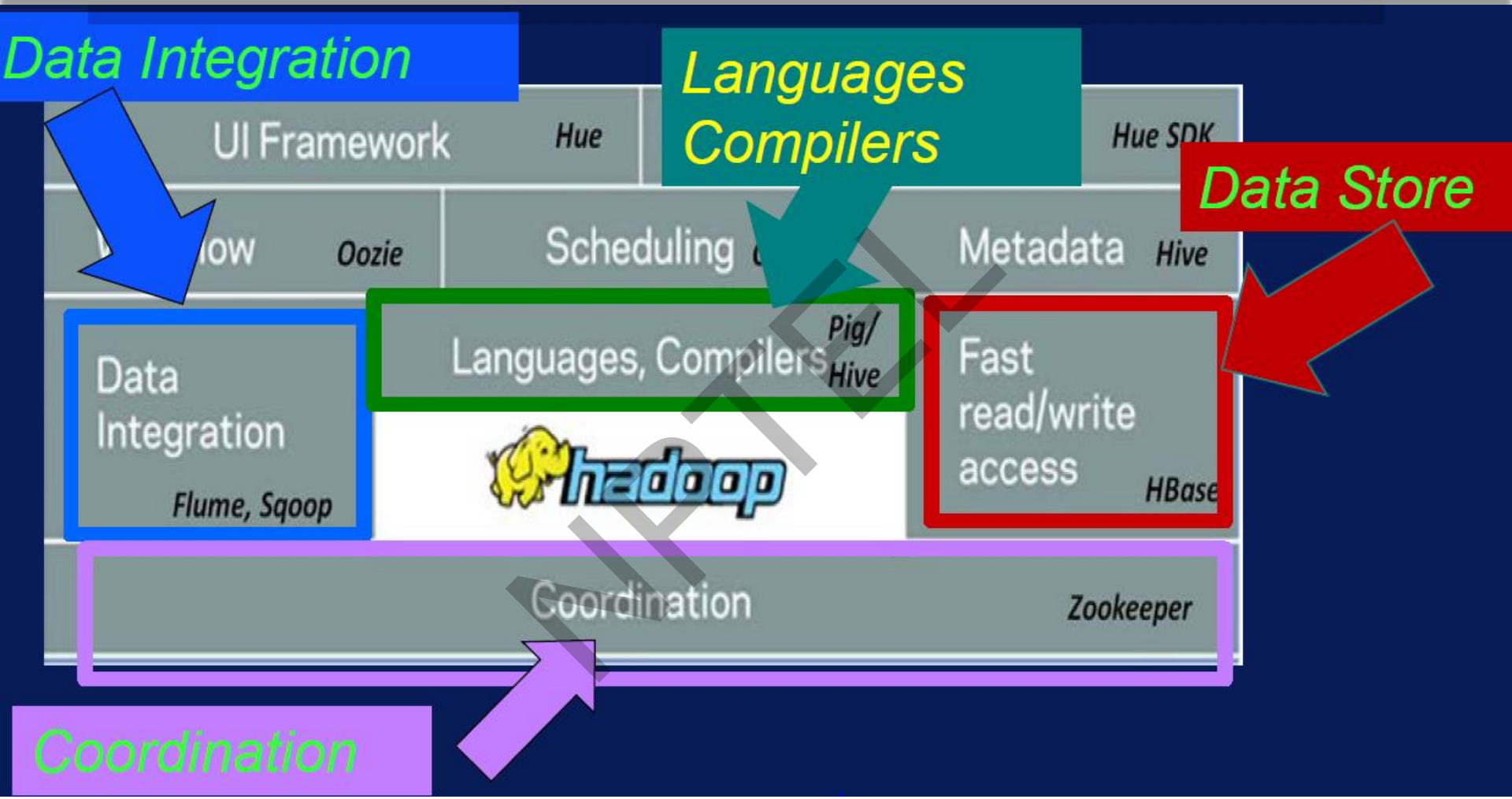
Yahoo Version of the Stack



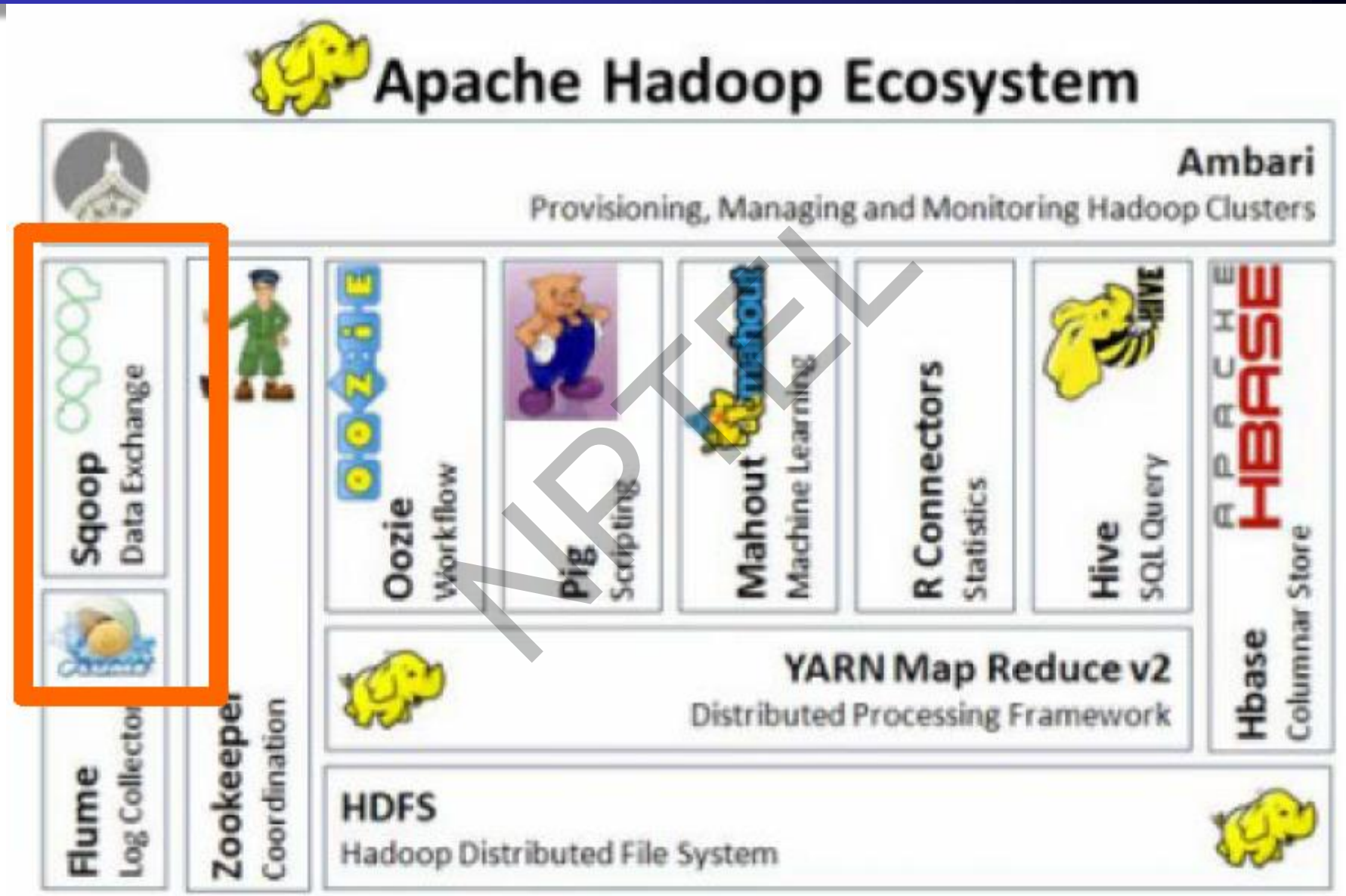
LinkedIn's Version of the Stack



Cloudera's Version of the Stack

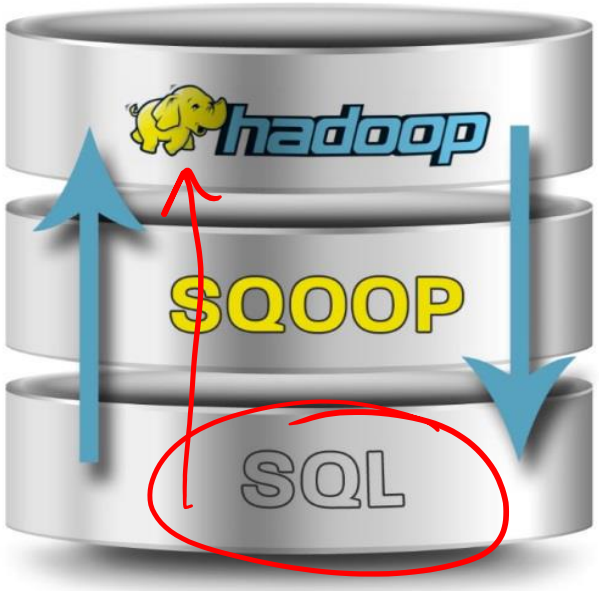


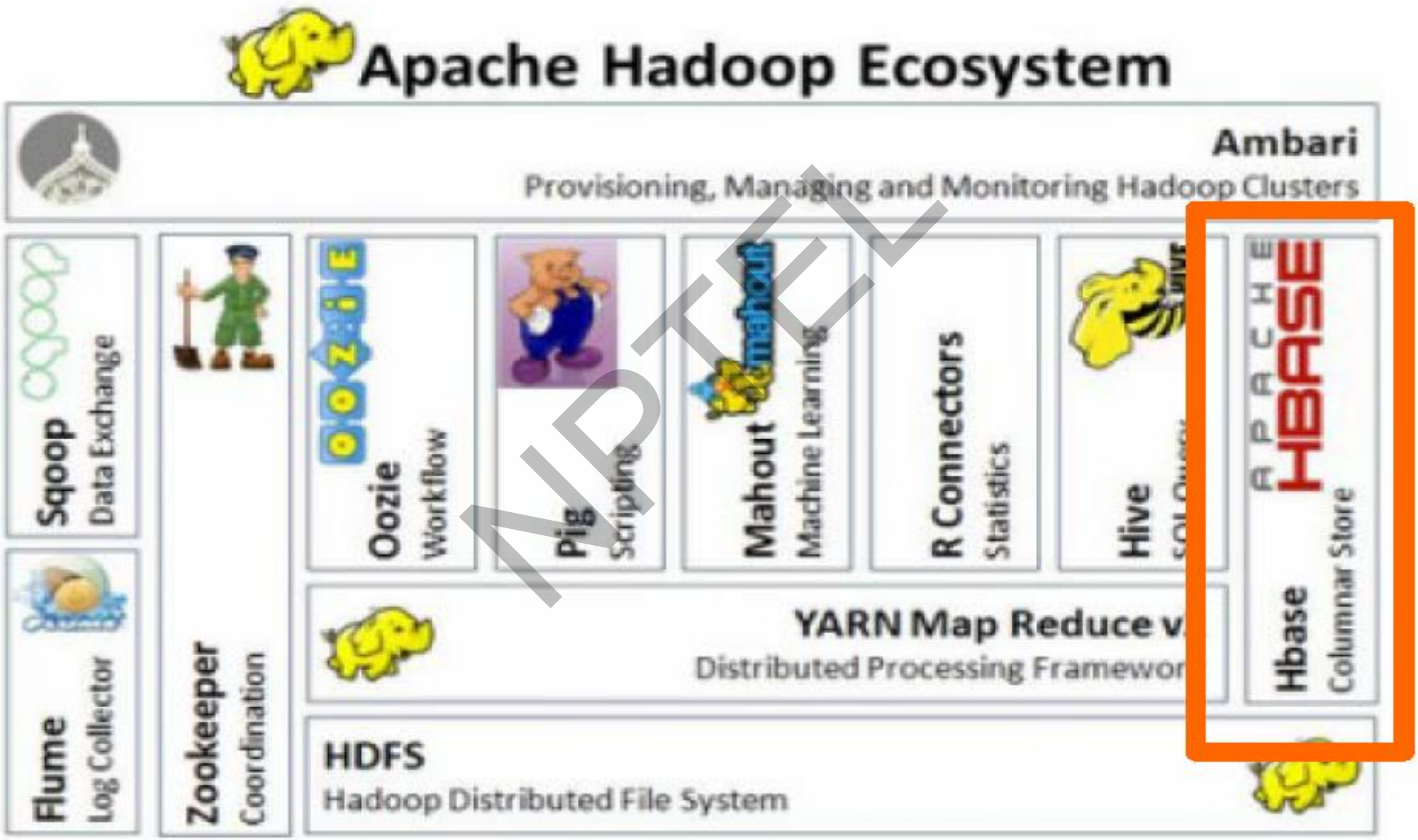
Hadoop Ecosystem
Major Components


Apache Hadoop Ecosystem
Ambari
Provisioning, Managing and Monitoring Hadoop Clusters
Sqoop
Data Exchange
Flume
Log Collector
Zookeeper
Coordination
Oozie
Workflow
Pig
Scripting
Mahout
Machine Learning
R Connectors
Statistics
Hive
SQL Query
APACHE HBASE
Hbase
Columnar Store
YARN Map Reduce v2
Distributed Processing Framework
HDFS
Hadoop Distributed File System

Apache Sqoop

- Tool designed for efficiently transferring bulk data between Apache Hadoop and structured datastores such as relational databases

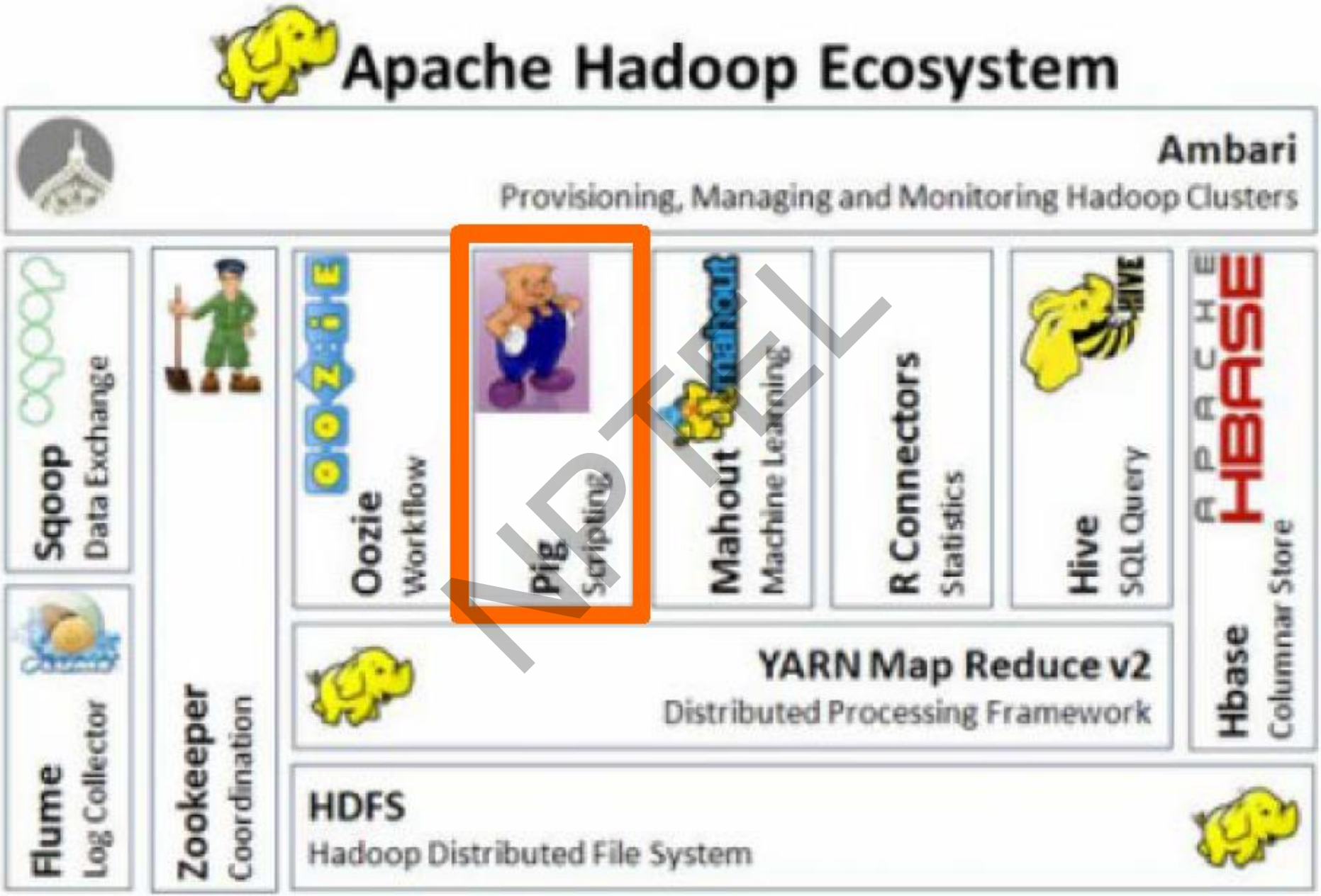



Apache Hadoop Ecosystem
Ambari
Provisioning, Managing and Monitoring Hadoop Clusters
Sqoop
Data Exchange
Flume
Log Collector
Zookeeper
Coordination
Oozie
Workflow
Pig
Scripting
Mahout
Machine Learning
R Connectors
Statistics
Hive
APACHE
HBASE
Hbase
Columnar Store
YARN Map Reduce v
Distributed Processing Framewor
HDFS
Hadoop Distributed File System

HBASE

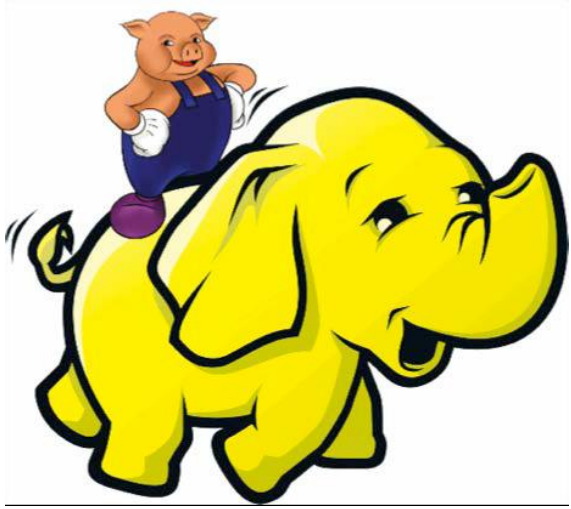
- Hbase is a key component of the Hadoop stack, as its design caters to applications that require really fast random access to significant data set.

- Column-oriented database management system
- Key-value store
- Based on Google Big Table
- Can hold extremely large data
- Dynamic data model
- Not a Relational DBMS

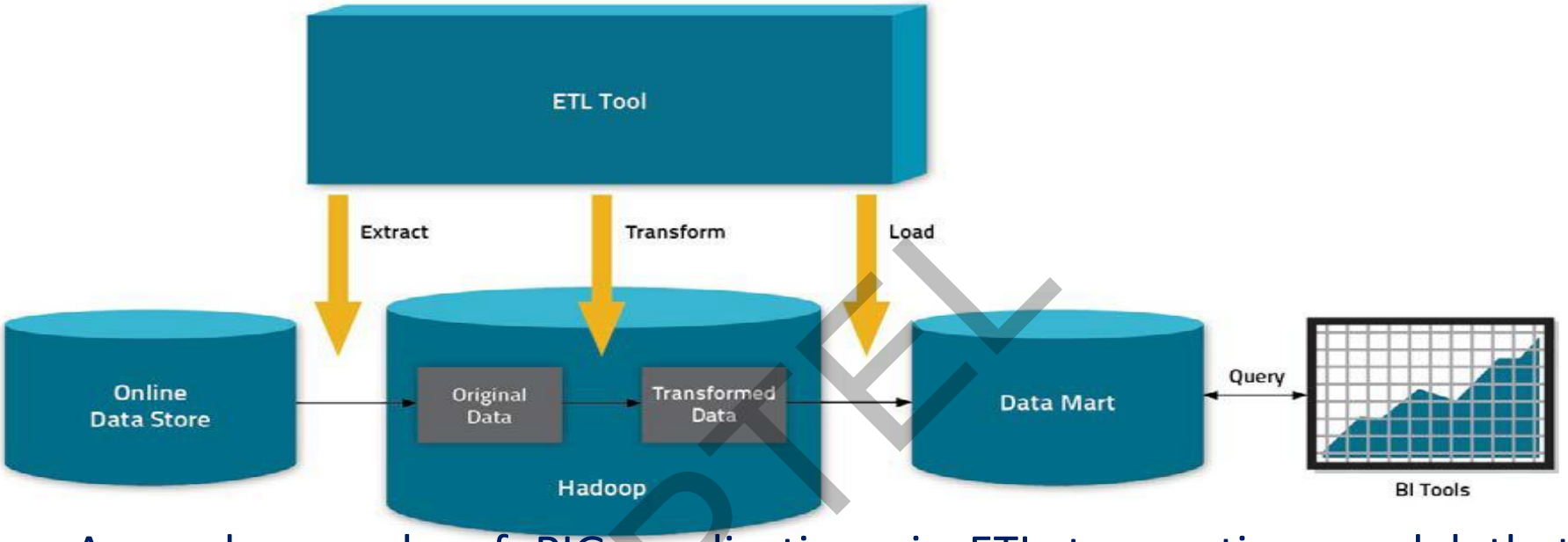

Apache Hadoop Ecosystem
Ambari
Provisioning, Managing and Monitoring Hadoop Clusters
Sqoop
Data Exchange
Flume
Log Collector
Zookeeper
Coordination
Oozie
Workflow
Pig
Scripting
Mahout
Machine Learning
R Connectors
Statistics
Hive
SQL Query
Hbase
Columnar Store
YARN Map Reduce v2
Distributed Processing Framework
HDFS
Hadoop Distributed File System

PIG

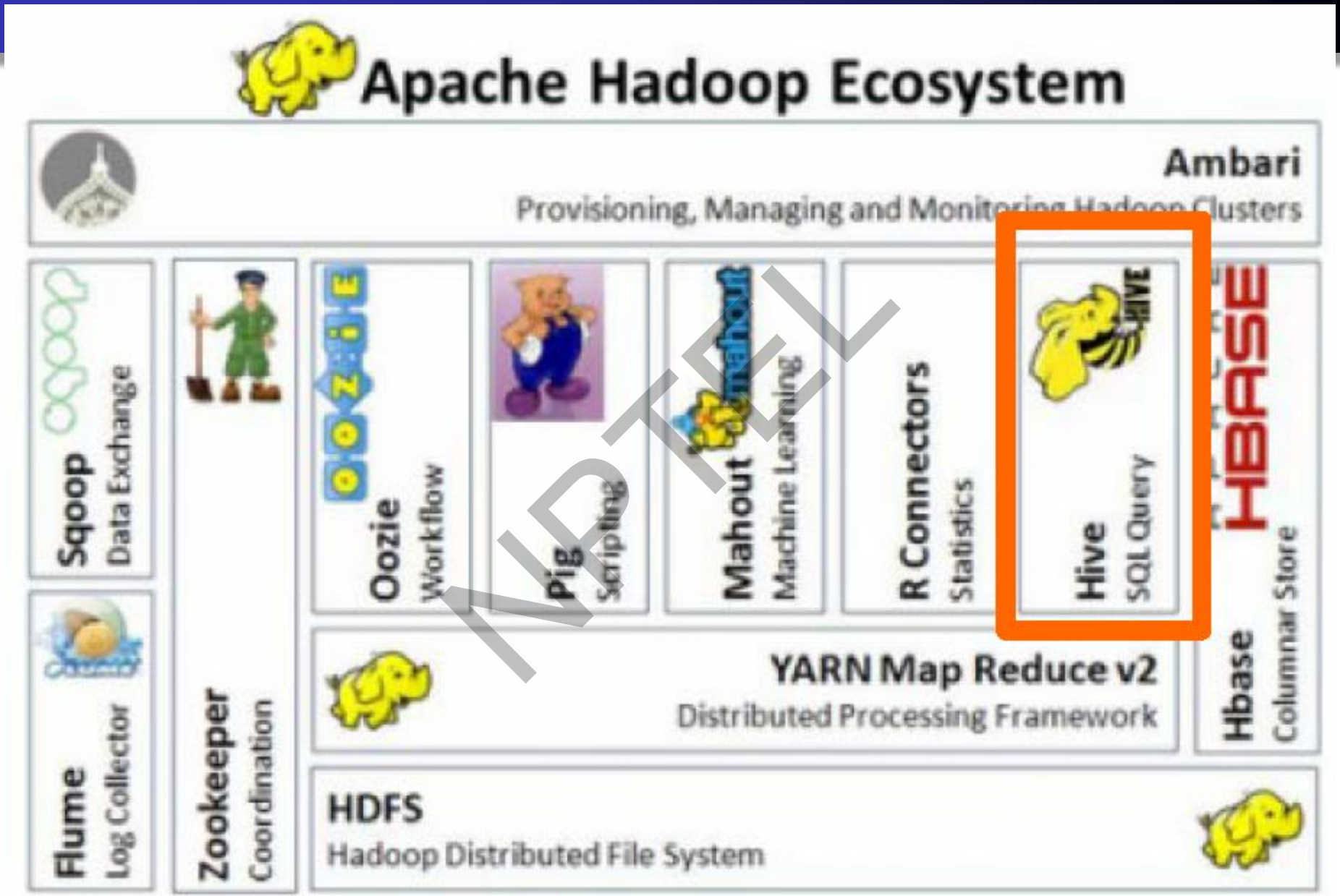
- High level programming on top of Hadoop MapReduce
- The language: Pig Latin
- Data analysis problems as data flows
- Originally developed at Yahoo 2006



PIG for ETL

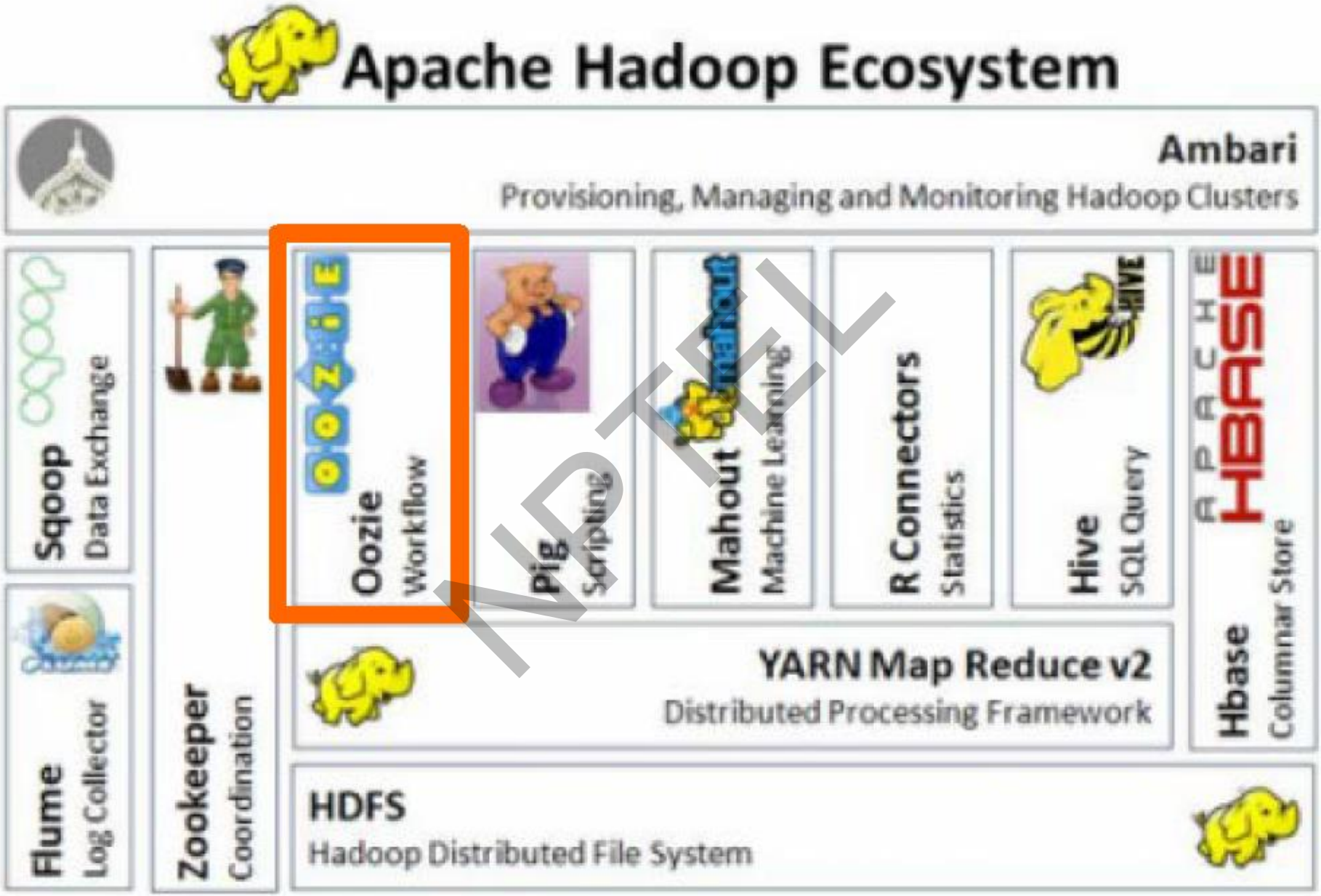


- A good example of PIG applications is ETL transaction model that describes how a process will extract data from a source, transporting according to the rules set that we specify, and then load it into a data store.
- PIG can ingest data from files, streams, or any other sources using the UDF: a user-defined functions that we can write ourselves.
- When it has all the data it can perform, select, iterate and do kinds of transformations.


Apache Hadoop Ecosystem
Ambari
Provisioning, Managing and Monitoring Hadoop Clusters
Sqoop
Data Exchange
Flume
Log Collector
Zookeeper
Coordination
Oozie
Workflow
Pig
Scripting
Mahout
Machine Learning
R Connectors
Statistics
Hive
SQL Query
Hbase
Columnar Store
YARN Map Reduce v2
Distributed Processing Framework
HDFS
Hadoop Distributed File System

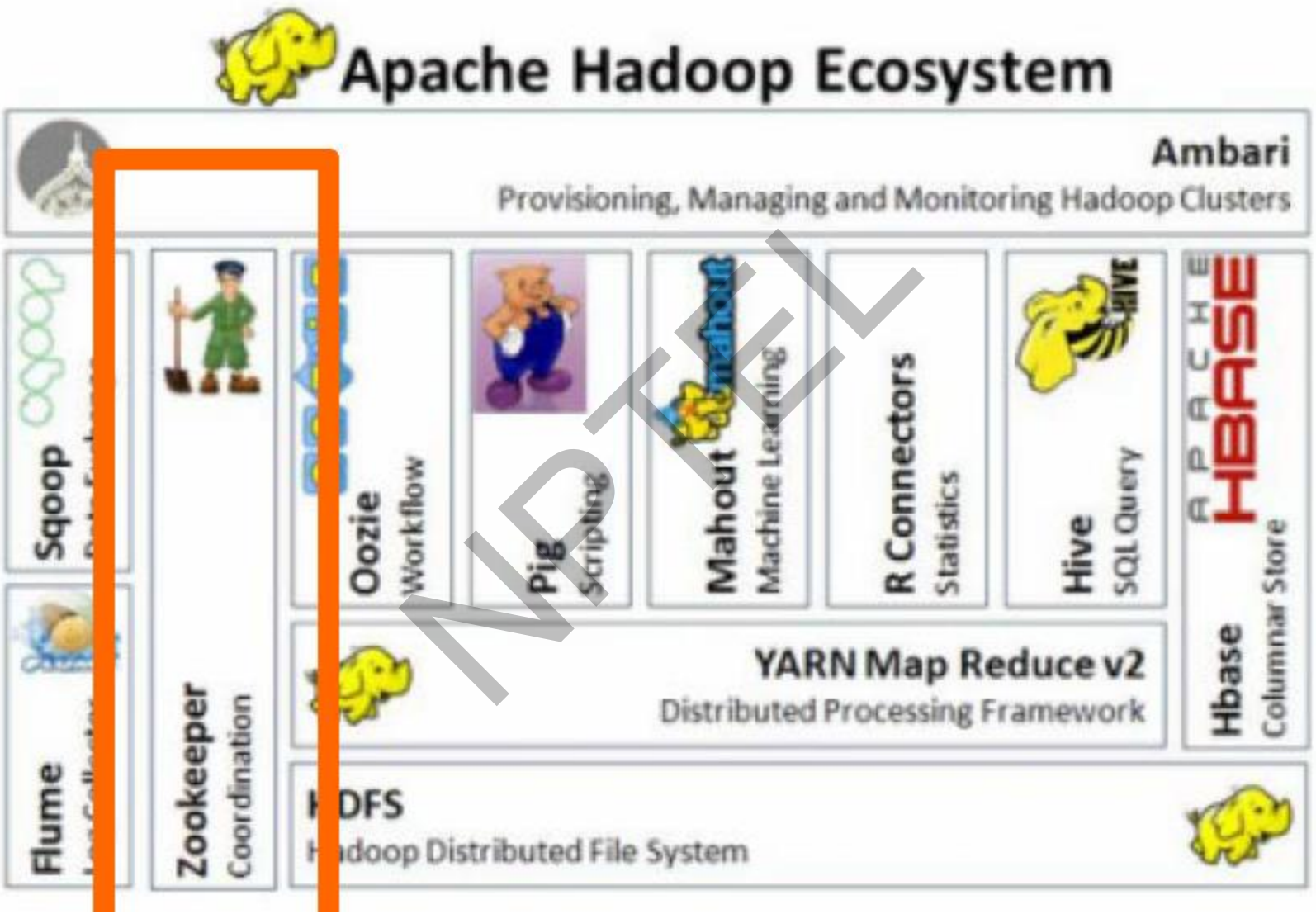
Apache Hive

- Data warehouse software facilitates querying and managing large datasets residing in distributed storage
- SQL-like language!
- Facilitates querying and managing large datasets in HDFS
- Mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL


Apache Hadoop Ecosystem
Ambari
Provisioning, Managing and Monitoring Hadoop Clusters
Sqoop
Data Exchange
Oozie
Workflow
Pig
Scripting
Mahout
Machine Learning
R Connectors
Statistics
Hive
SQL Query
APACHE HBASE
Hbase
Columnar Store
Flume
Log Collector
Zookeeper
Coordination
YARN Map Reduce v2
Distributed Processing Framework
HDFS
Hadoop Distributed File System

Oozie

- Workflow scheduler system to manage Apache Hadoop jobs
- Oozie Coordinator jobs!
- Supports MapReduce, Pig, Apache Hive, and Sqoop, etc.

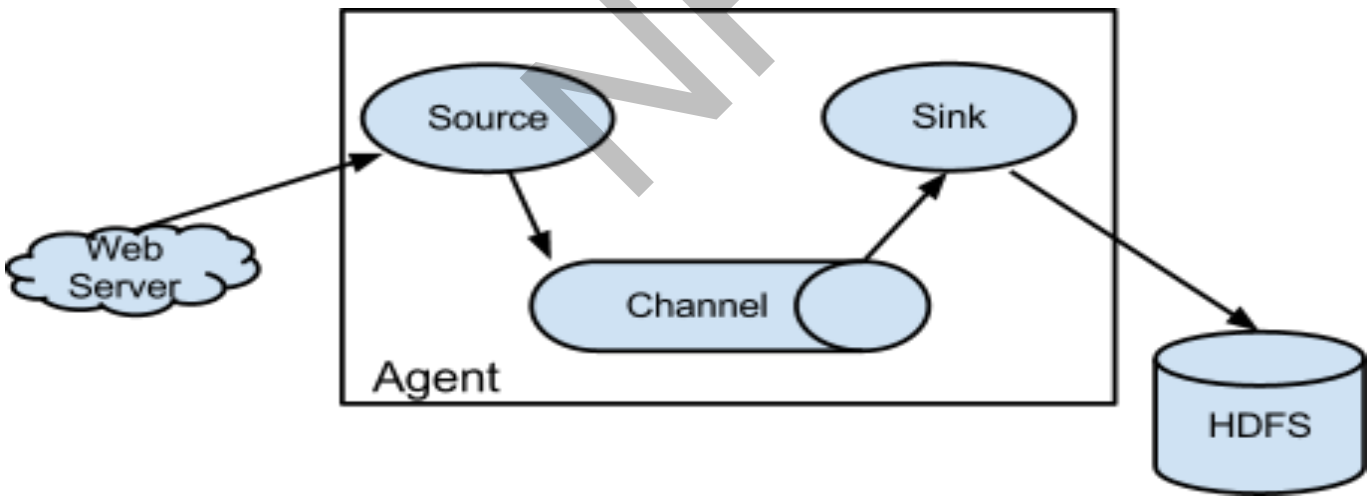

Apache Hadoop Ecosystem
Ambari
Provisioning, Managing and Monitoring Hadoop Clusters
Sqoop
Oozie
Workflow
Pig
Scripting
Mahout
Machine Learning
R Connectors
Statistics
Hive
SQL Query
APACHE
HBASE
Hbase
Columnar Store
Flume
Zookeeper
Coordination
YARN Map Reduce v2
Distributed Processing Framework
HDFS
Hadoop Distributed File System

Zookeeper

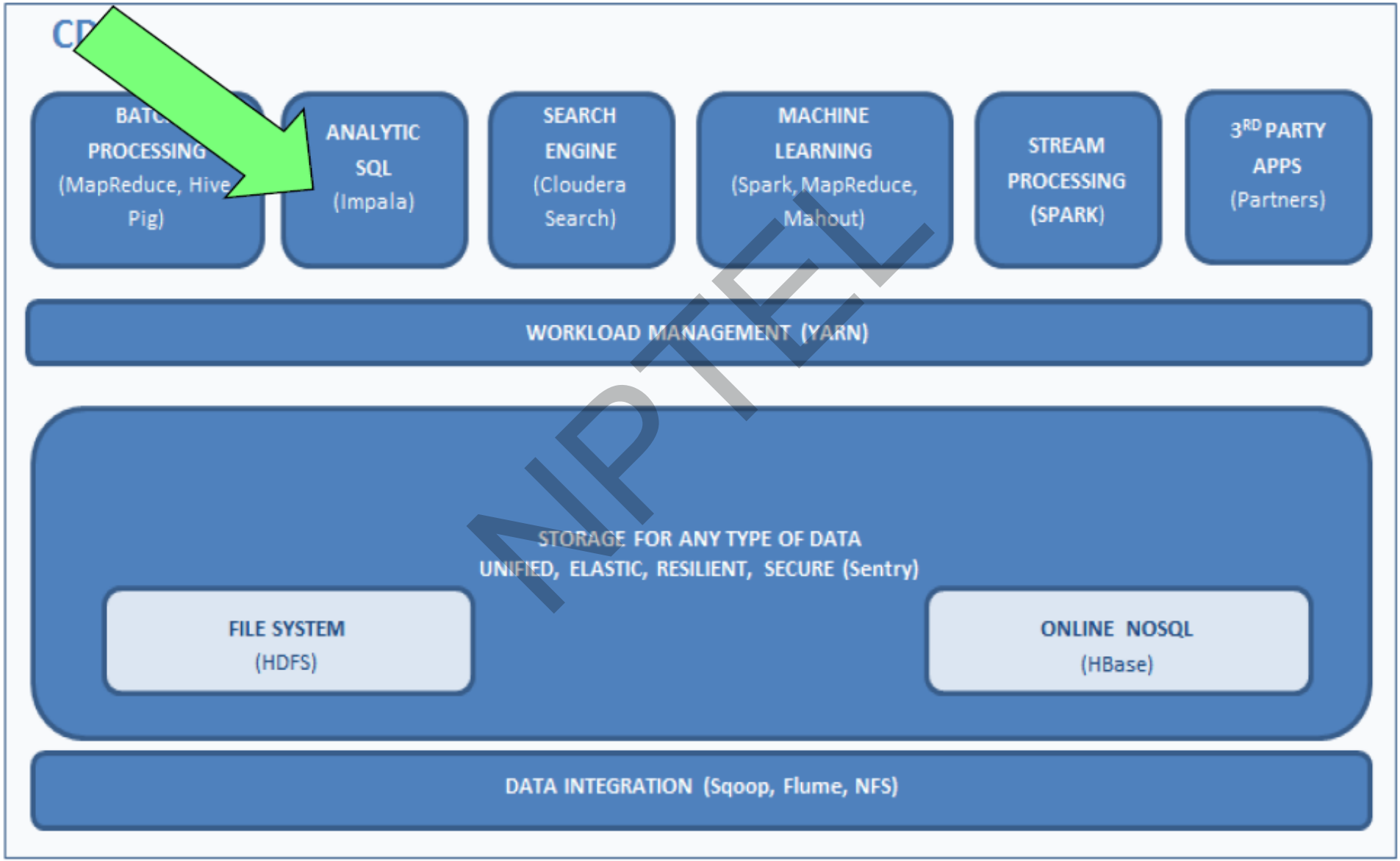
- Provides operational services for a Hadoop cluster group services
- Centralized service for: maintaining configuration information naming services
- Providing distributed synchronization and providing group services

Flume

- Distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data — data ingestion
- It has a simple and very flexible architecture based on streaming data flows. It's quite robust and fall tolerant, and it's really tunable to enhance the reliability mechanisms, fail over, recovery, and all the other mechanisms that keep the cluster safe and reliable.
- It uses simple extensible data model that allows us to apply all kinds of online analytic applications.



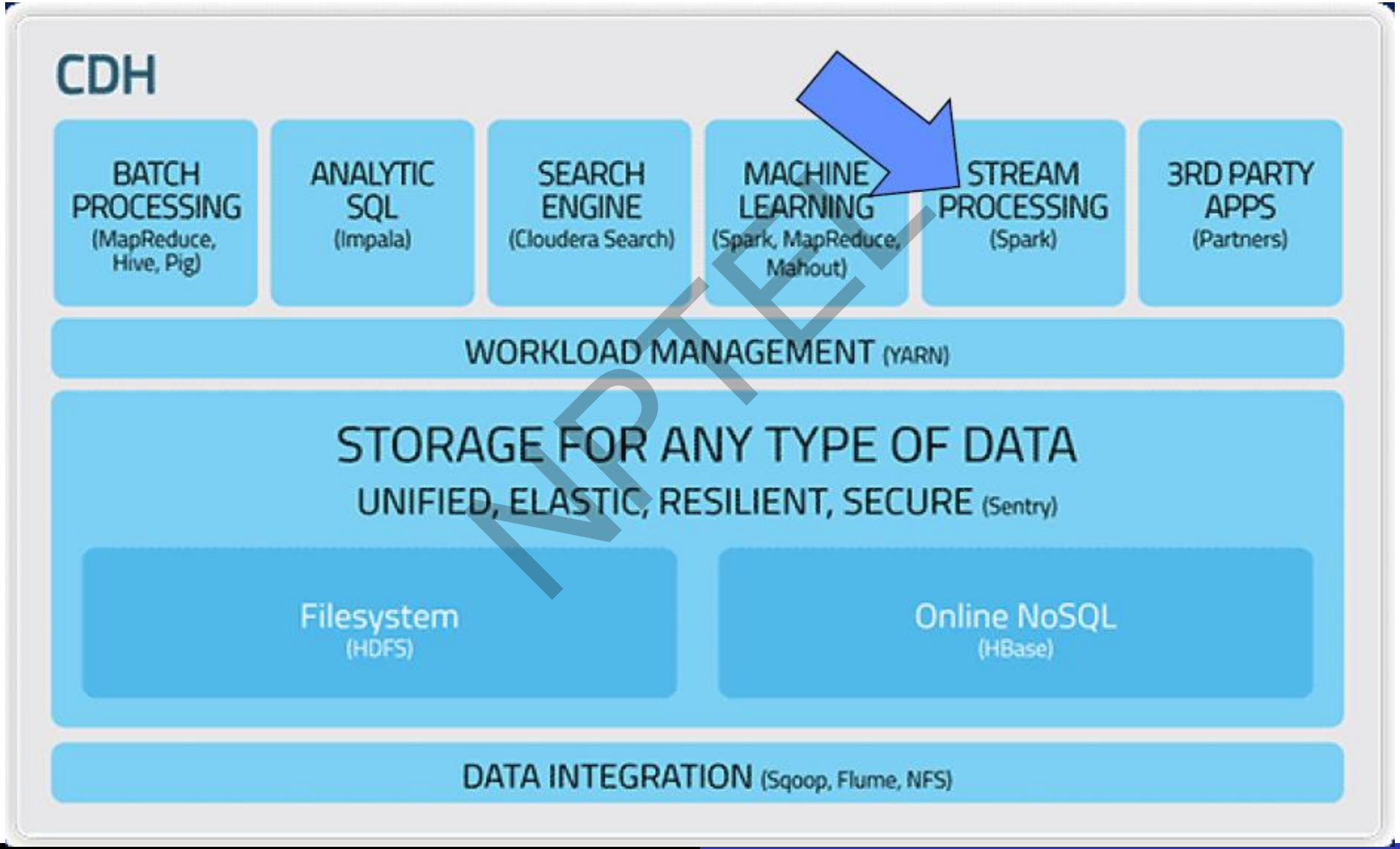
Additional Cloudera Hadoop Components Impala



Impala

- Cloudera, Impala was designed specifically at Cloudera, and it's a query engine that runs on top of the Apache Hadoop. The project was officially announced at the end of 2012, and became a publicly available, open source distribution.
- Impala brings scalable parallel database technology to Hadoop and allows users to submit low latencies queries to the data that's stored within the HDFS or the Hbase without acquiring a ton of data movement and manipulation.
- Impala is integrated with Hadoop, and it works within the same power system, within the same format metadata, all the security and reliability resources and management workflows.
- It brings that scalable parallel database technology on top of the Hadoop. It actually allows us to submit SQL like queries at much faster speeds with a lot less latency.

Additional Cloudera Hadoop Components Spark The New Paradigm



Spark

- Apache Spark™ is a fast and general engine for large-scale data processing
- Spark is a scalable data analytics platform that incorporates primitives for in-memory computing and therefore, is allowing to exercise some different performance advantages over traditional Hadoop's cluster storage system approach. And it's implemented and supports something called Scala language, and provides unique environment for data processing.
- Spark is really great for more complex kinds of analytics, and it's great at supporting machine learning libraries.
- It is yet again another open source computing frame work and it was originally developed at MP labs at the University of California Berkeley and it was later donated to the Apache software foundation where it remains today as well.

Spark Benefits

- In contrast to Hadoop's two stage disk based MapReduce paradigm Multi-stage in-memory primitives provides performance up to 100 times faster for certain applications.
- Allows user programs to load data into a cluster's memory and query it repeatedly
- Spark is really well suited for these machined learning kinds of applications that often times have iterative sorting in memory kinds of computation.
- Spark requires a cluster management and a distributed storage system. So for the cluster management, Spark supports standalone native Spark clusters, or you can actually run Spark on top of a Hadoop yarn, or via patching mesas.
- For distributor storage, Spark can interface with any of the variety of storage systems, including the HDFS, Amazon S3.

Conclusion

- In this lecture, we have discussed the specific components and basic processes of the Hadoop architecture, software stack, and execution environment.