Emergency data sources

Recap

- Introduction to EM
- Introduction to Bigdata
- Presentations by you
- Essay topics
- Programming project topics
- Practical session
    - Apache Spark installation
    - Exercises?

Overview

- Individual essay and programming projects
- Big data architecture and technologies
- EM data sources
  - open data
  - linked open data
  - government data
- Practical session
  - Running Hadoop and MapReduce
  - Querying and analyzing government open data source by using Spark cluster
- Presentations by you

Individual essays

- The essay shall present and discuss selected theory, technology and tools related to big data technologies and EM, backed by scholarly and other references
  - counts 30% of final grade
  - presentations: November 22nd
  - **deadline: December 5th 1400**
  - deadline is passed for selection of the essay topic.
- Encouraged:
  - more than a paper
  - social media contrib's (Wikipedia, Wikidata...)

Some possible Essay Themes:

- Privacy in emergency big data
- Visualization of big data
- Big data ethics.
- Big Data - is it trustworthy?
- Participatory Sensing: A further step. Sensing through Social Media feeds.
- Sentiment Analysis: machine learning approach.
- Redefining communication- the role of social media in disaster management
- Social media and disasters: Current uses, future options, and policy considerations.
- How social media enhances emergency situation awareness?
- Discovering Big data Technologies for EM
- Crisis Analytics: Big Data Driven Crisis Response
- Opportunities and challenges of bid data use in EM?

Chosen Essay Topics

- Visualization of big data. **Kari Raudstein, Marius, F. Lillevik**
- Big Data - is it trustworthy? **Marius, F. Lillevik**
- Big data ethics. **Malene Gilje Fonnes, and Taume Dery**
- Participatory Sensing: A further step. Sensing through Social Media feeds. **Espen Holst Wiik**
- Redefining communication- the role of social media in disaster management.
- How social media enhances emergency situation awareness? **Magnus Lyngseth Vestby**
- Discovering Big data Technologies for EM. **Andreas Iden**
- Opportunities and challenges of bid data use in EM? **Kristoffer Torkildsen Marthinsen**
- Data infrastructure for emergency systems. **Johnny Bjånesøy**
- Ethics of health data collection during a pandemic. **Roar Håkonsen Helland**
- Social and ethical challenges of mobile data for disaster management. **Eirik Sjøvoll**
- The use of big data in healthcare. **Yngve Kristoffersen**
- Twitter data for disaster management: to predict, warn and respond to crisis situations. **Sara Stegane**
- How does social media enhance situation awareness? Government's responsibility in using such technologies? Pros/cons? Past, present, future? **Sebastian Øverhaug**

Not chosen topics

- Privacy in emergency big data
- Big Data - is it trustworthy?
- Sentiment Analysis: machine learning approach.
- Social media and disasters: Current uses, future options, and policy considerations.
- Crisis Analytics: Big Data Driven Crisis Response.

Student group programming project

- The project shall develop an application that can be used for emergency management. Development and run-time platform is free choice, as is programming language. The project should be carried out in groups of three and not more. Working individually or in pairs is not recommended.

  - Counts 40% of final grade.

  - Final presentation: Friday November 23rd

  - Submission deadline: Tuesday November 27th, 1400

  - ***Optional deadline:*** *Thursday September 21st*

  - **Deadline for selection of the project and group: Oct 10th, 2018, kl.14.00**

Project Topics

1) Integrating different social media applications for providing information awareness either to victims/first responders.
2) Detecting emerging hot topics over the social media before, during and after emergency management.
3) Social media data analytics for disaster management.
4) Deep Learning for emergency management Using Social Media Information.
5) Analysis of Post-disaster Twitter Communication: A case study of….
6) Develop an information visualization tool to identify the disaster risk.

Integrating different social media data sources for Information awareness.

- Developing an application for integrating different social media data sources for information awareness.
    - take twitter, facebook, snapchat, Instagram etc.,
    - Integration (combining data)
    - present it in a unified way on GUI / App
    - use API to collect data from social media data sources

Developing an application to detect emerging hot topics over the social media before, during and after emergency management.

- Developing an application for detecting the disaster trends in various regions.
    - Using different machine learning for detection

Social media data analytics for emergency management

- Develop a system capable of processing, analyzing, and extracting useful information from Twitter during an emergency to understand the public sentiment.

  e.g. Usecase: Kerala floods 2018.

Deep Learning for emergency management Using Social Media Information.

- Classification of social media information by using different machine learning algorithms.

Analysis of Post-disaster Twitter Communication: A case study

- Develop an application to understand/classify the communication tasks that are performed by using Twitter during post–disaster phase.

Information visualization tool to identify the disaster risk.

- Develop an information visualization tool to identify the disaster risk.

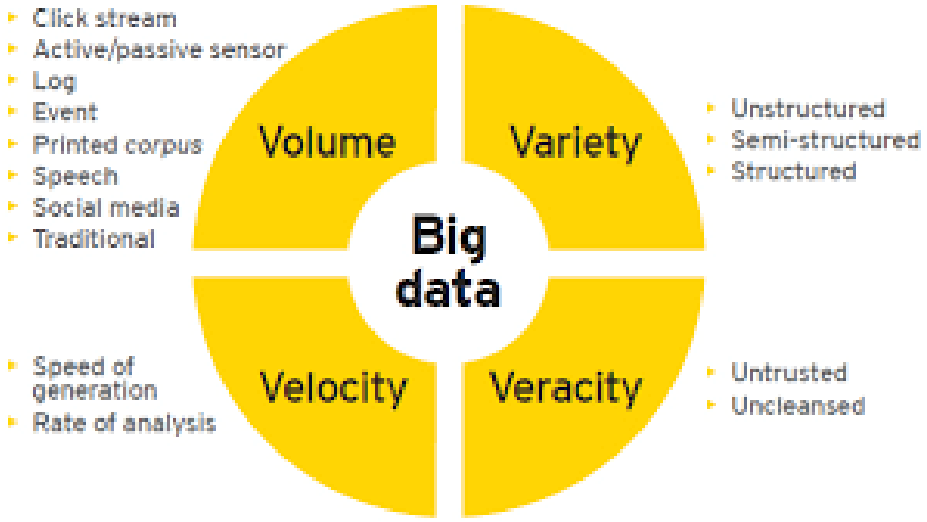
Should we find a coordinated project task?

Overview

- Individual essay and programming projects
- Big data architecture and technologies
- Available data sources for EM
  - open data
  - linked open data
  - government data
- Presentations by you
- Practical session
  - Running Hadoop and MapReduce
  - Querying and analyzing government open data source by using Spark cluster

Characteristics of Big data:

- The “three V's” (3V):
  - volume, velocity, variety – at once
    - old days: you could only have two of the three
  - also two more: veracity, value



Data vs Information

- **Data**: Data is raw.

  - It simply exists and has no significance beyond its existence (in and of itself).

  - It can exist in any form, usable or not. It does not have meaning of itself.

  - In computer parlance, a spreadsheet generally starts out by holding data.

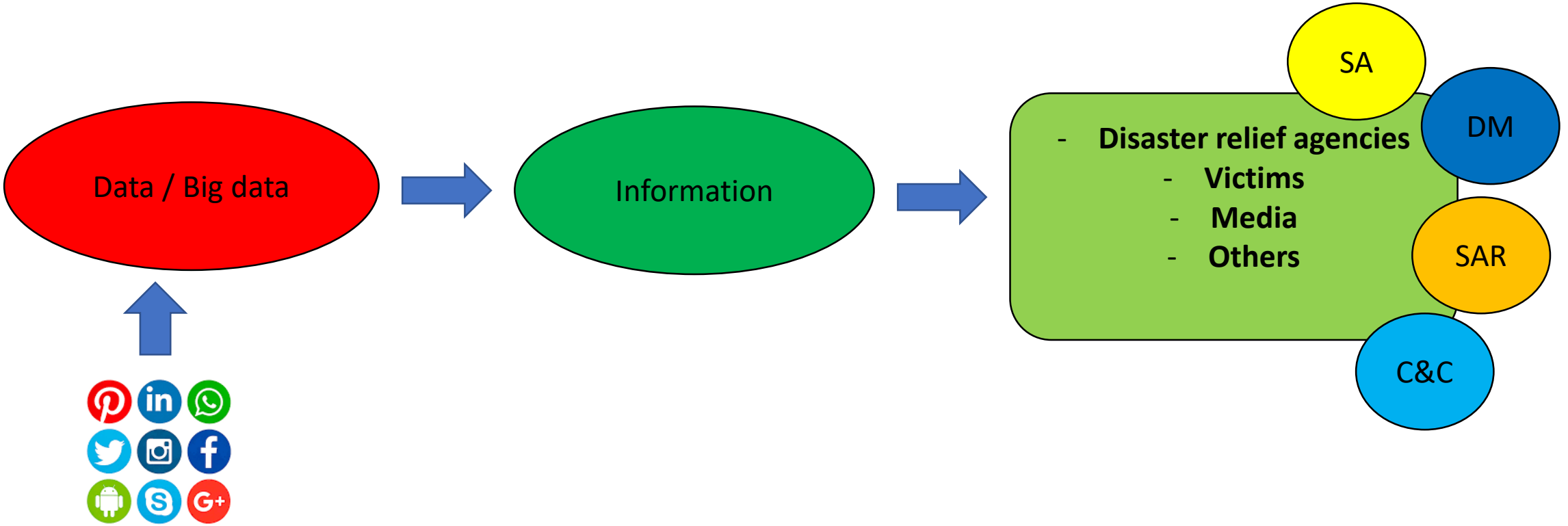
- **Information:** It is data that has been given meaning by way of relational connection.

  - This "meaning" can be useful, but does not have to be.

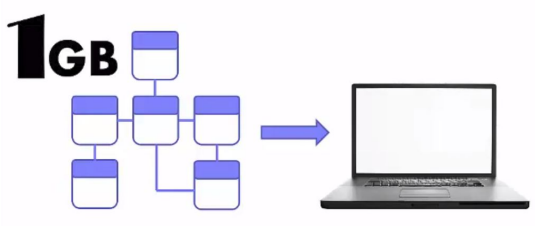
  - In computer parlance, a relational database makes information from the data stored within it.

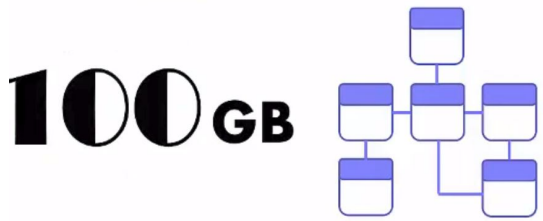
Data for Disaster management

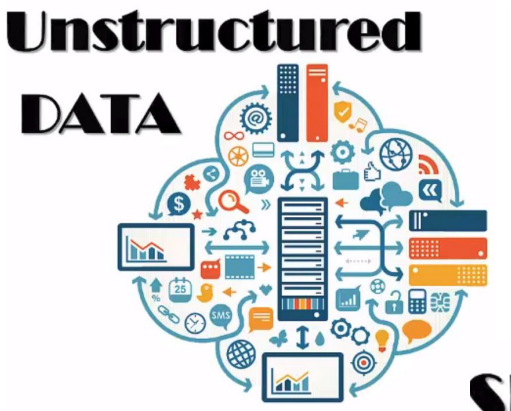
- SA: situational awareness
- DM: Decision Making
- SAR: Search and rescue operation
- C&C: Coordination and collaboration

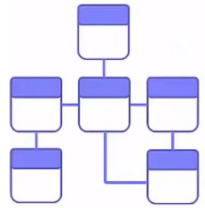


Why Big data?

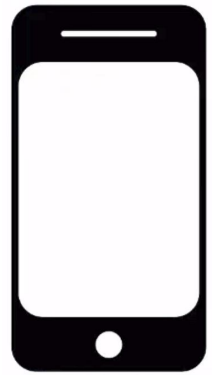






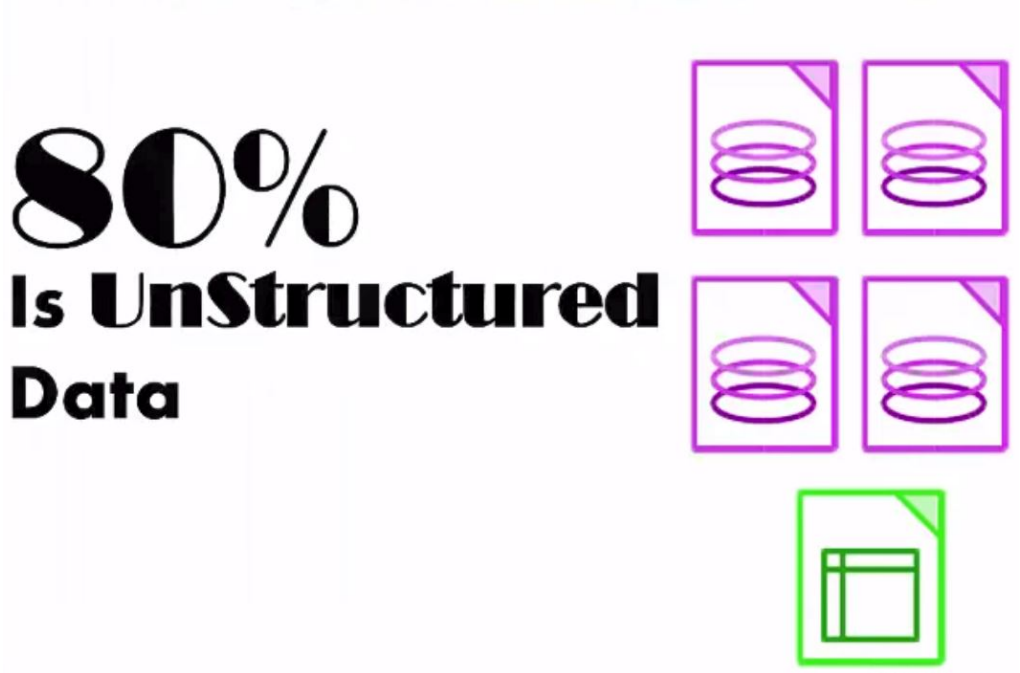


Parallel Processing
Parallel Processing
Parallel Processing
Parallel Processing




7
TeraByte


600
TeraByte
https://code.facebook.com/posts/229861827208629/scaling-the-facebook-data-warehouse-to-300-pb/


80%
Is UnStructured
Data

Major challenges with Big Data

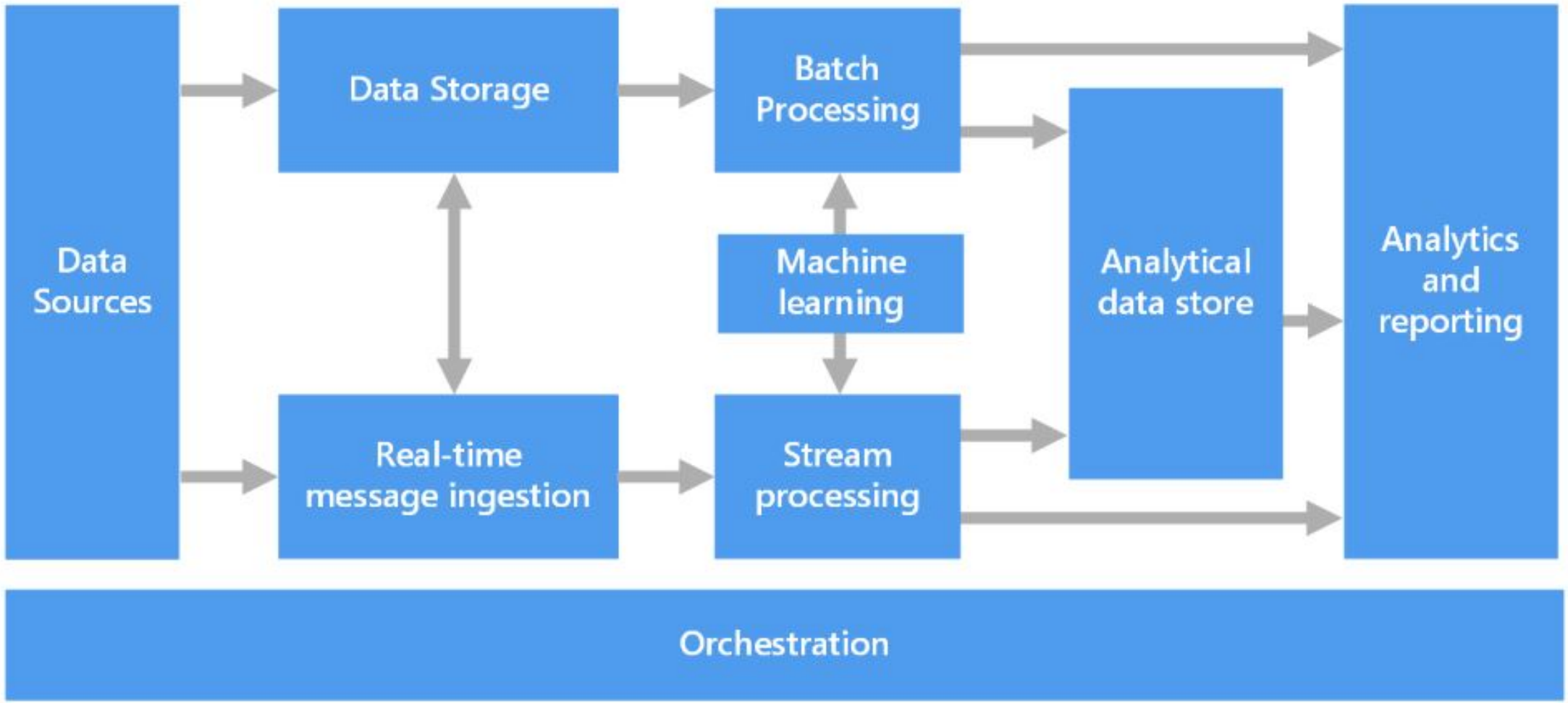
- ***Storing the colossal amount of data:***
  - Storing huge data in a traditional system is not possible.
    - the storage will be limited to one system
    - the data is increasing at a tremendous rate.
- ***Storing heterogeneous data:***
  - Data is in various formats i.e. unstructured, semi-structured and structured.
- ***Processing speed:***
  - processing the huge amount of data is quite high.
  - data to be processed is too large.

Big data architecture

- Handles the ingestion, processing, and analysis of data that is too large or complex for traditional database systems.
- Big data architectures used for solving:
  - Expectations with the data has changed.
  - The cost of storage has fallen dramatically,
  - Facing an advanced analytics problem
- Big data solutions typically involve one or more of the following types of workload:
  - Batch processing of big data sources at rest.
  - Real-time processing of big data in motion.
  - Interactive exploration of big data.
  - Predictive analytics and machine learning.
- Big data architectures are needed when:
  - Store and process data in volumes too large for a traditional database.
  - Transform unstructured data for analysis and reporting.
  - Capture, process, and analyze unbounded streams of data in real time, or with low latency.

Source: https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

Components of a big data architecture



Source: https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

Big Data Technologies

Ingestion	Storage/Retention	Processing	Access
Ingestion Architecture: • Scalable, Extensible to capture streaming and batch data. • Provide capability to business logic, filters, validation, data quality, routing, etc. business requirements.	**Data Storage:** • Depending on the requirements data is placed into Hadoop HDFS, Hive, Hbase, Elastic Search or In-memory. • Metadata management • Policy-based Data Retention is provided.	**Data Processing:** • Processing is provided for both batch and near-realtime use cases • Provision Workflows for repeatable Data processing • Provide Late Data Arrival Handling	**Visualization and APIs:** • Dashboard and applications that provides valuable business insights • Data will be made available to consumers using API, MQ Feed and DB access
Technology Stack: • Apache Flume • Apache Kafka • Apache Storm • Apache Sqoop • NFS Gateway	**Technology Stack:** • HDFS • Hive Tables • Hbase/MapR DB • Elastic Search	**Technology Stack:** • Map Reduce • Hive • Spark • Storm • Drill	**Technology Stack:** • Qlik/Tableau/Spotfire • REST APIs • Apache Kafka • JDBC

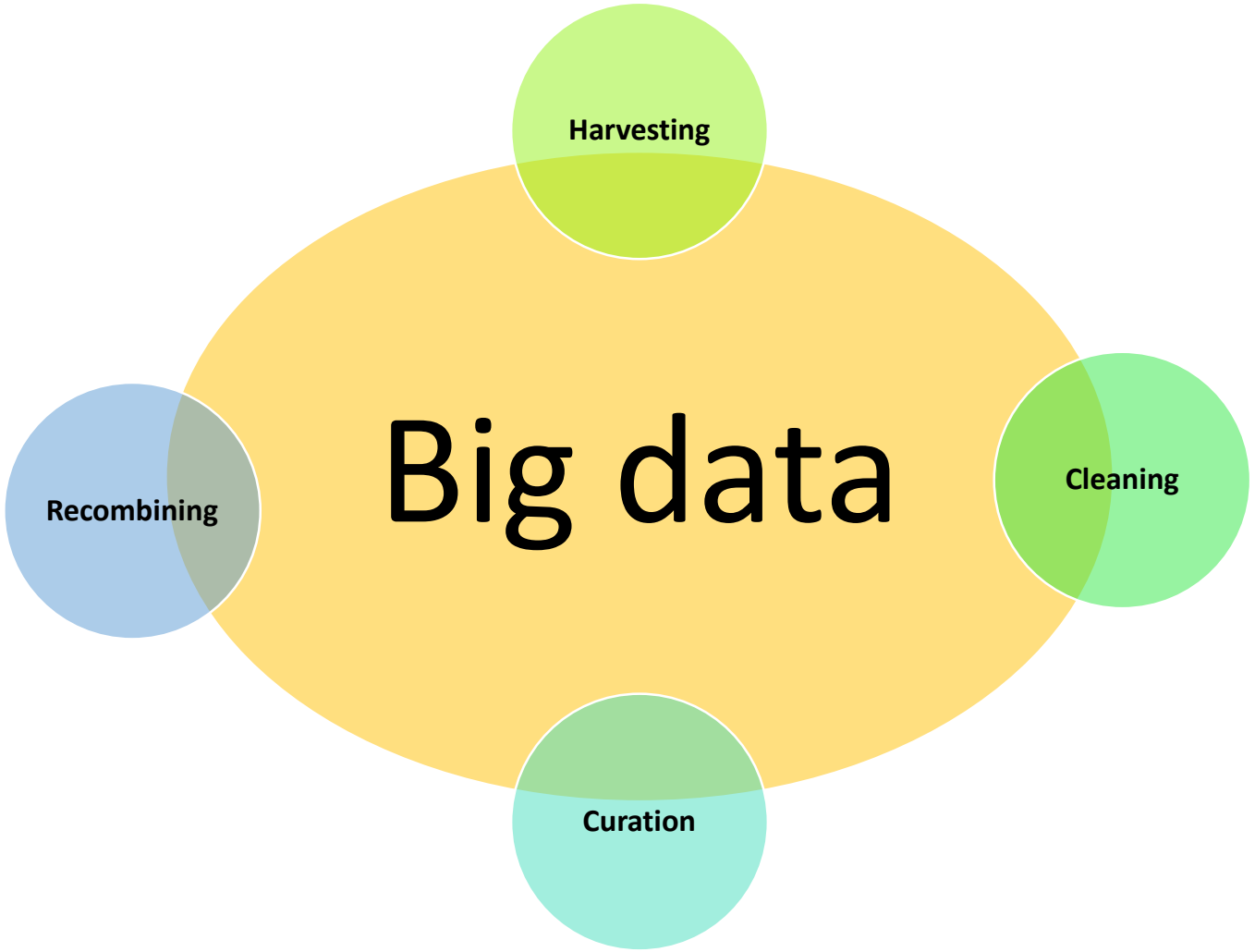
Management, Monitoring, Governance

Ambari, Cloudera Manager, Cloudera Navigator, MapR MCS

Batch Processing

- Data at rest.
  - the source data is loaded into data storage, either by the source application itself or by an orchestration workflow. The data is then processed in-place by a parallelized job, which can also be initiated by the orchestration workflow. The processing may include multiple iterative steps before the transformed results are loaded into an analytical data store, which can be queried by analytics and reporting components.
    - For example, the logs from a web server might be copied to a folder and then processed overnight to generate daily reports of web activity.
- Batch processing is used in a variety of scenarios,
  - from simple data transformations to a more complete ETL (extract-transform-load) pipeline.
  - batch processing may operate over very large data sets, where the computation takes significant time.
  - batch processing typically leads to further interactive exploration, provides the modeling ready data for machine learning, or writes the data to a data store that is optimized for analytics and visualization.
    - One example of batch processing is transforming a large set of flat, semi-structured CSV or JSON files into a schematized and structured format that is ready for further querying. Typically the data is converted from the raw formats used for ingestion (such as CSV) into binary formats that are more performant for querying because they store data in a columnar format, and often provide indexes and inline statistics about the data.

Big data technologies usage

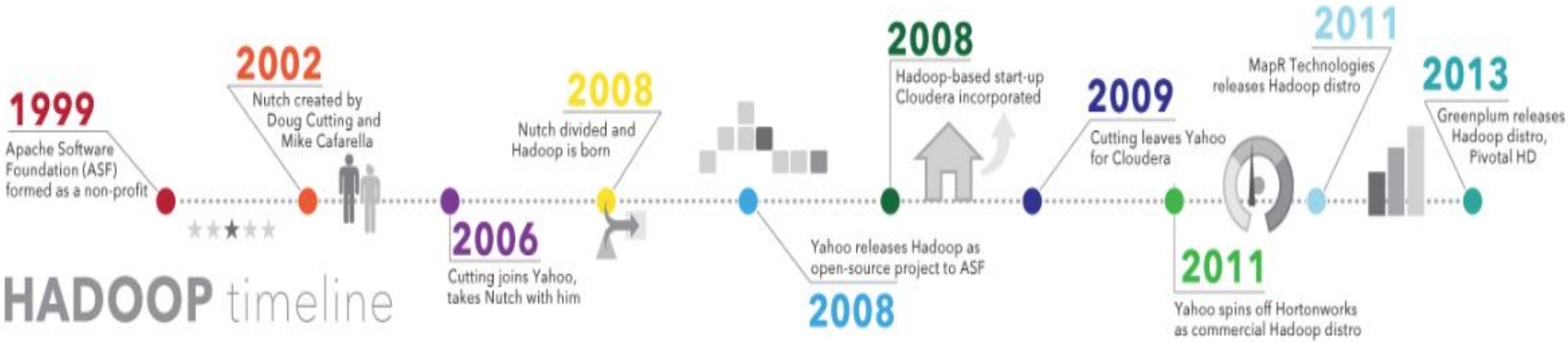


Hadoop?

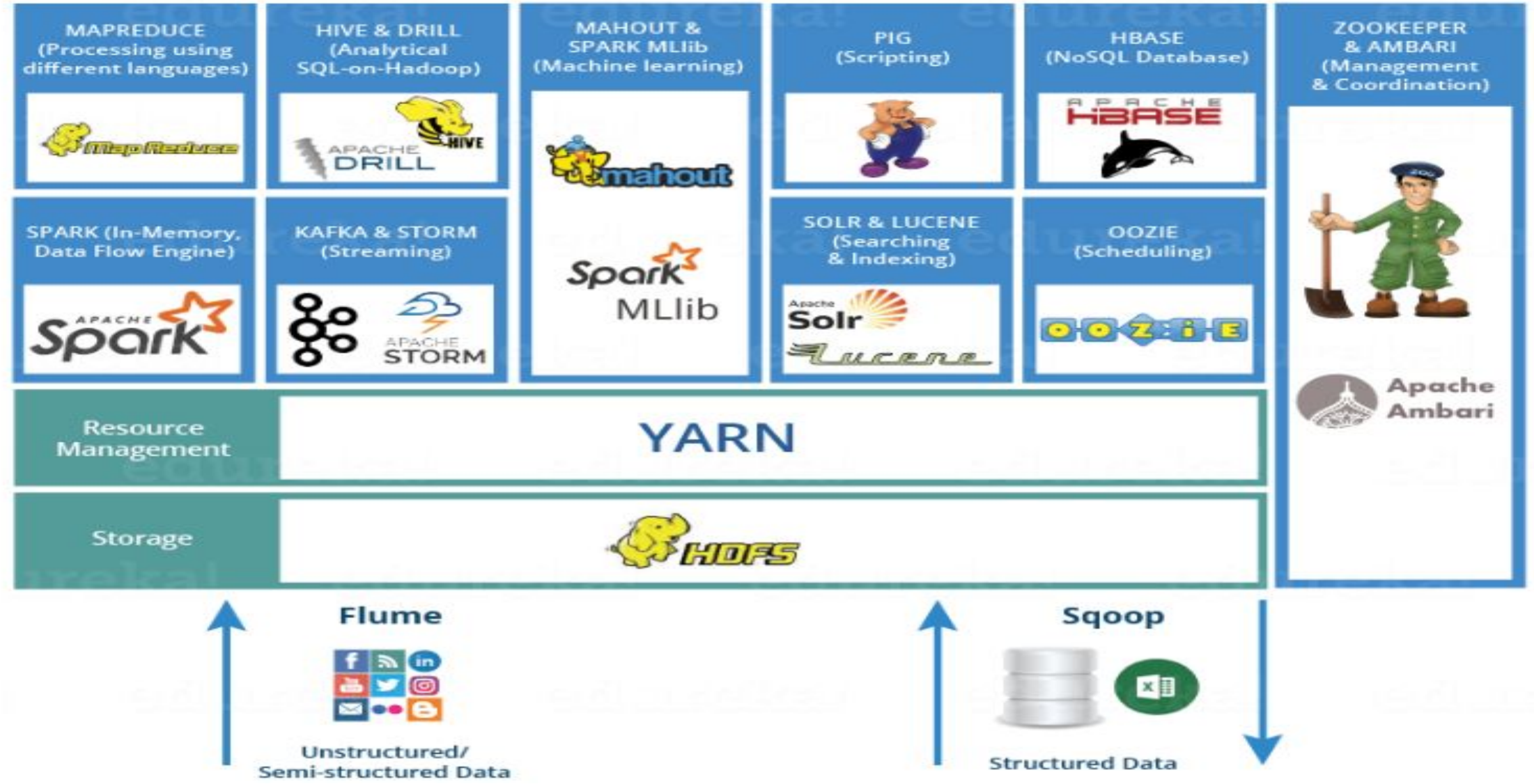
What is Hadoop?

- Hadoop is a running software framework
  - for storing data and running applications using the MapReduce algorithm.
  - massively distributed computing
    - on terabytes of data and beyond
    - over thousands of computing nodes (or on a laptop)
  - mostly written in Java

Hadoop History

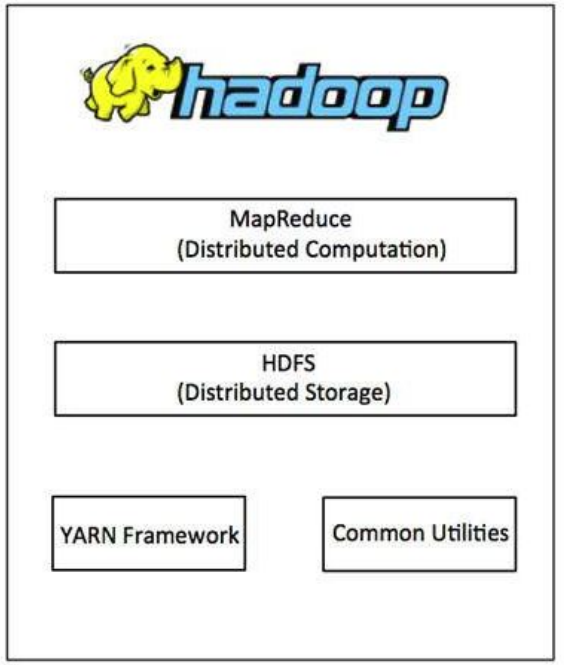


Hadoop ecosystem



Source: https://www.edureka.co/blog/hadoop-ecosystem

Hadoop Framework



- Hadoop framework components:
  - ***HDFS*** – Hadoop Distributed File System
  - **MapReduce** – distributed computing model
  - **YARN** – job tracking and process monitoring
  - **Common** – libraries and utilities (.jar-files)
  - – most can be run separately
  - – part of a bigger ecology of big-data technologies

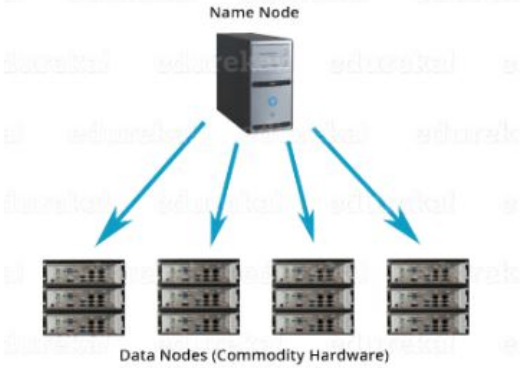
Source: http://hadoop.apache.org/

Why is Hadoop important?

What are the challenges of using Hadoop?

- MapReduce programming is not a good match for all problems.
- There's a widely acknowledged talent gap.
- Data security.
- Full-fledged data management and governance.

Hadoop Distributed File System (HDFS)



- The backbone of Hadoop Ecosystem
- Distributed, scalable, portable file system / data store
  - – optimized for mostly immutable files
  - - to store different types of large data sets
- Creates a level of abstraction over the resources
- Storing our data across various nodes and maintaining the log file about the stored data (metadata).
- Two core components:
  - *NameNode*
    - Doesn't store the actual data, contains metadata.
    - Requires less storage and high computational resources.
  - *DataNode*
    - data is stored, hence requires more storage resources.
    - commodity hardware in the distributed environment.

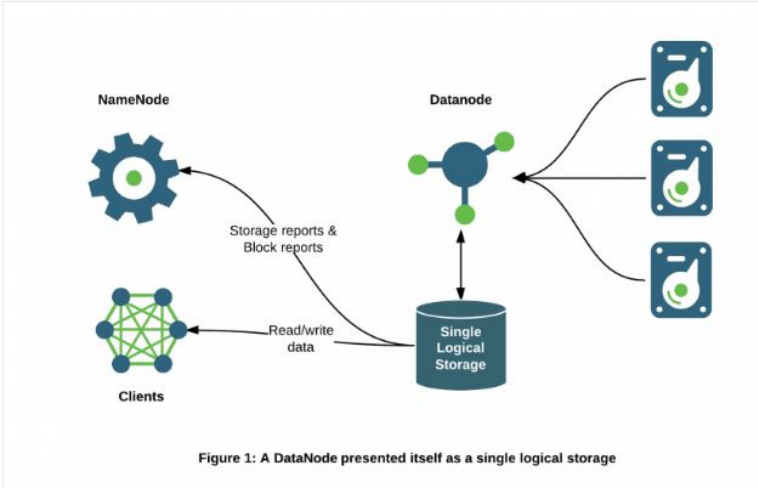
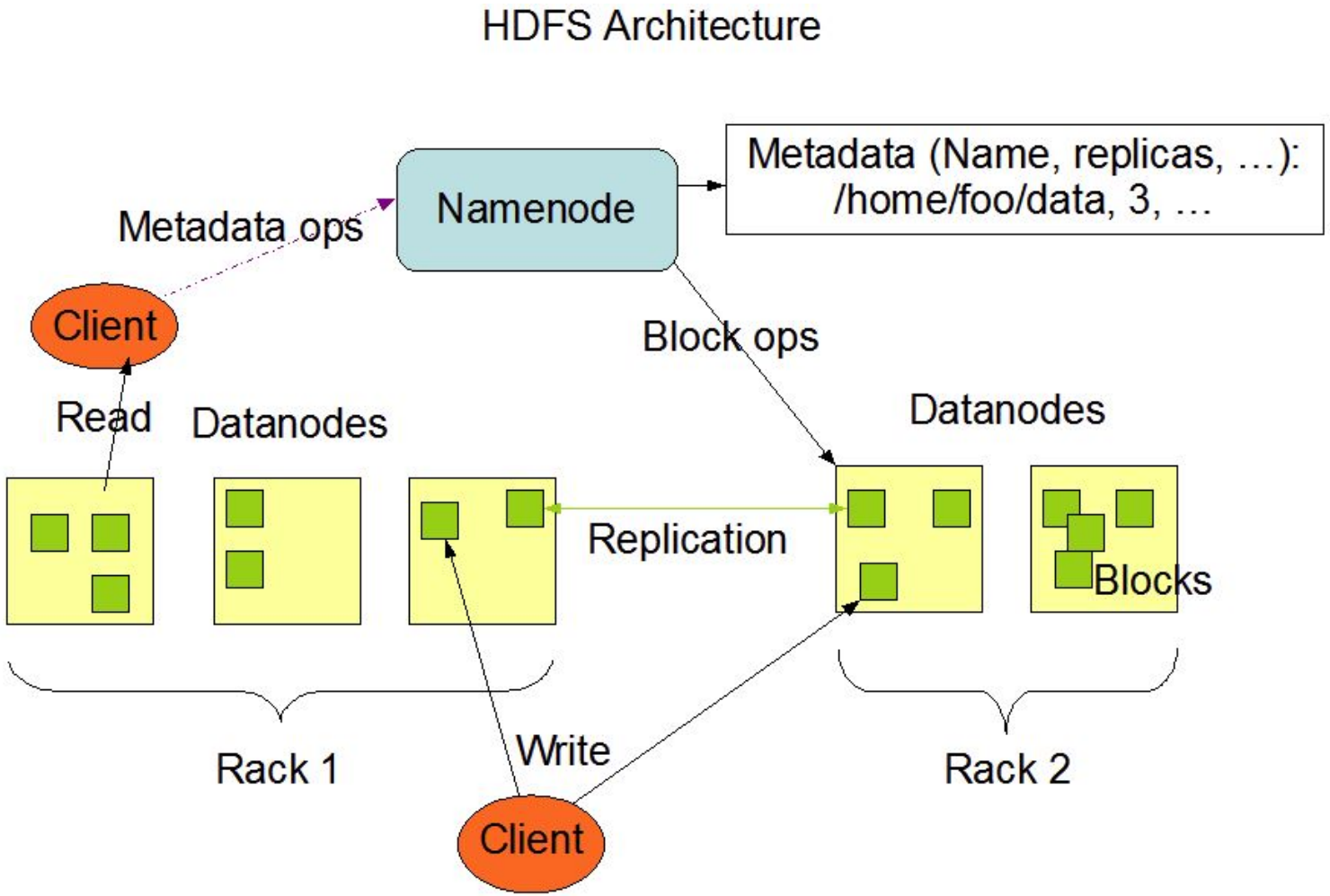


Figure 1: A DataNode presented itself as a single logical storage

HDFS architecture



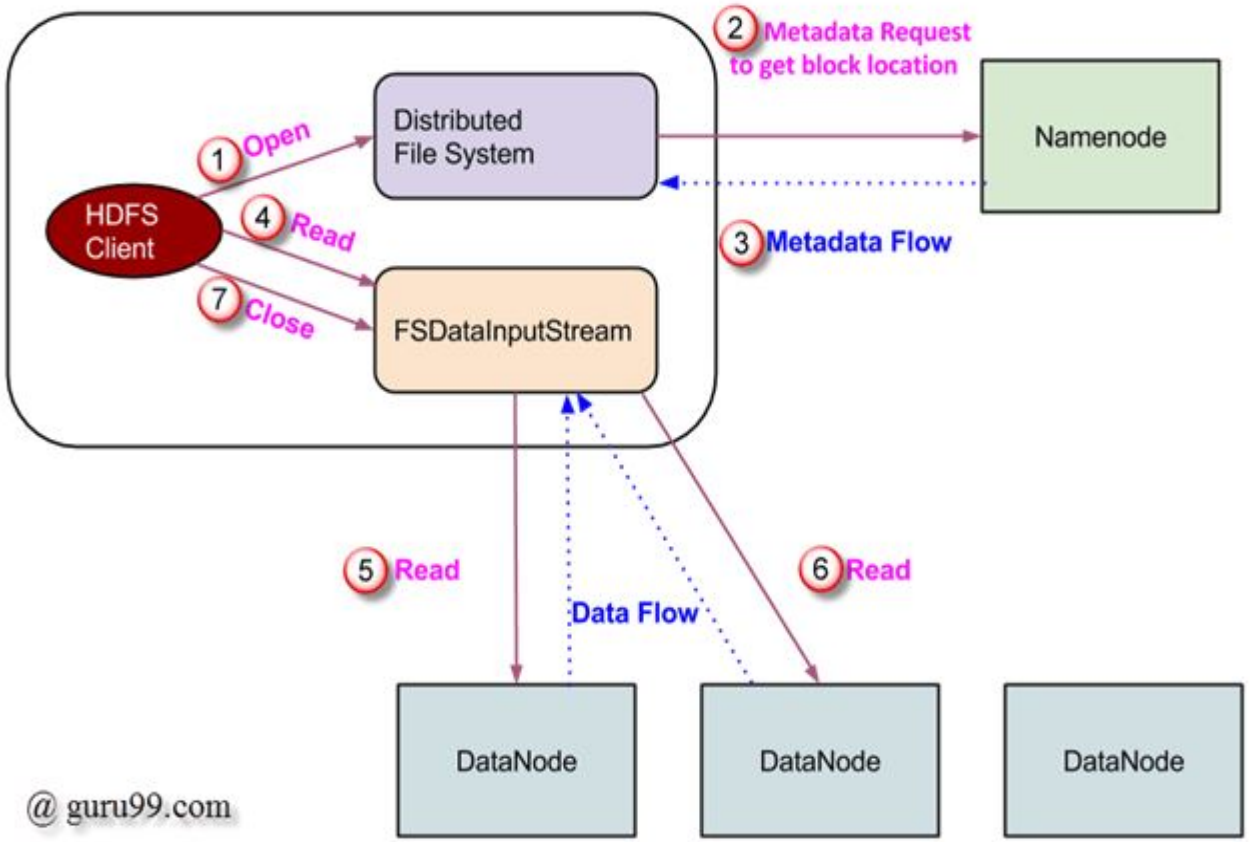
Source: http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html

Hadoop Distributed File System (HDFS)

- Read/write operations in HDFS operate at a block level.
- Data files in HDFS are broken into block-sized chunks, which are stored as independent units i.e., default block-size is 64 MB.
- HDFS operates on a concept of data replication.

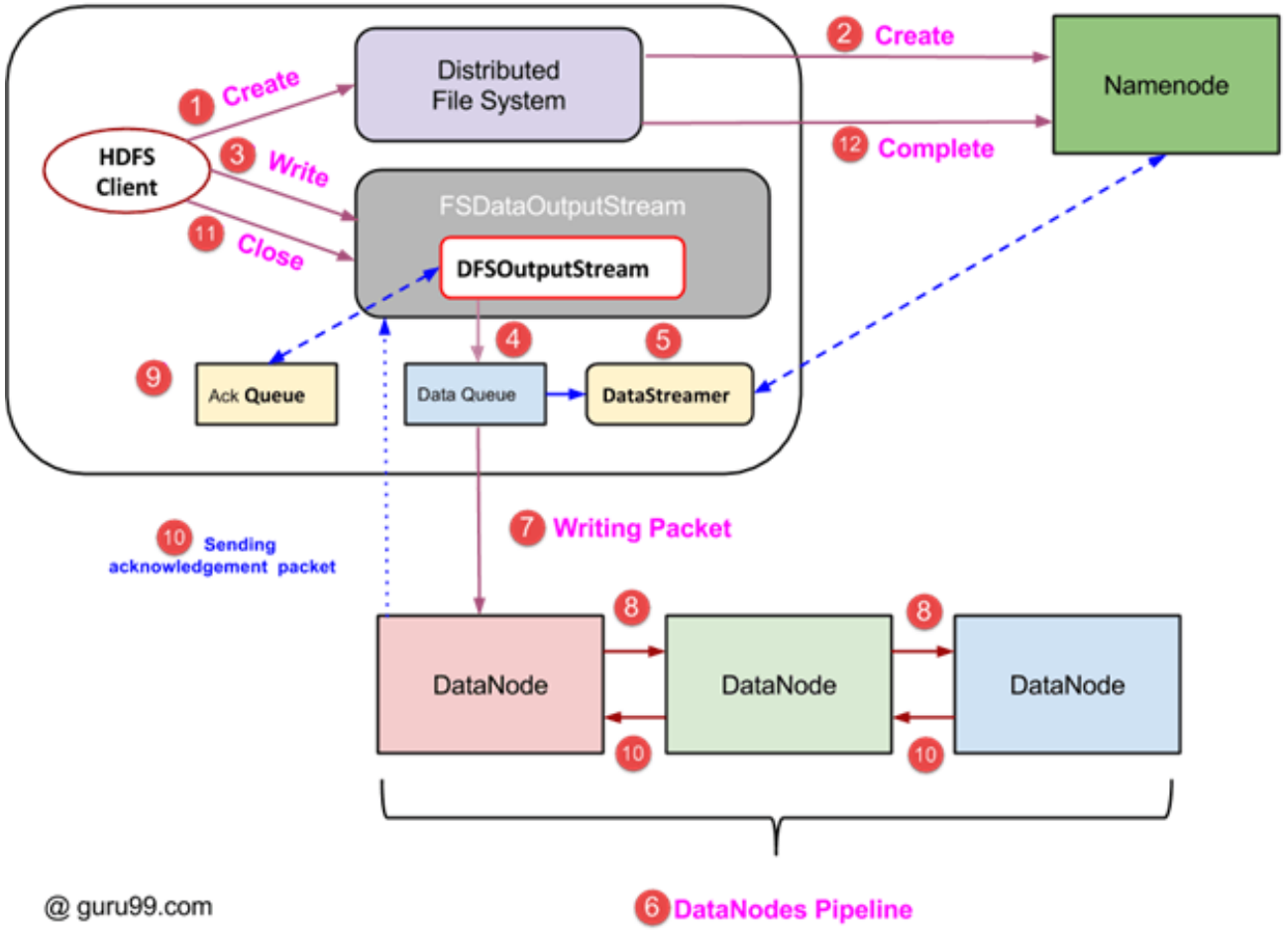
Hadoop Distributed File System (HDFS)

- **Read operations in HDFS:**



Hadoop Distributed File System (HDFS)

- **Write Operations In HDFS:**

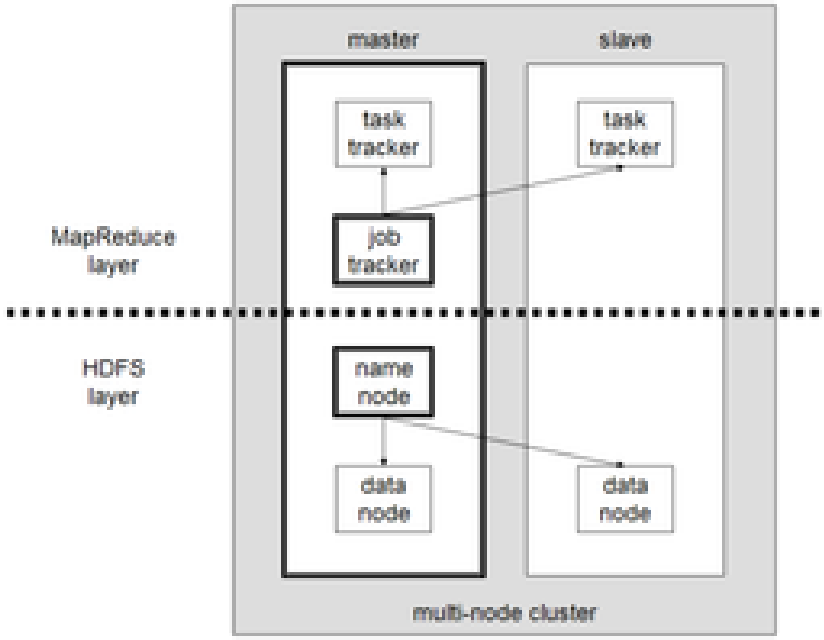


Hadoop cluster

- A collection of independent components connected through a dedicated network to work as a single centralized data processing resource.
- It is a special type of computational cluster designed specifically for storing and analyzing huge amounts of structured/unstructured data in a distributed computing environment.
- It distributes data analysis workload across various cluster nodes that work collectively to process the data in parallel.
- Also known as “Shared Nothing” systems .

Hadoop cluster Architecture

- Hadoop cluster consists of three components :
- Master node:
  - responsible for storing data in HDFS and executing parallel computation the stored data using MapReduce.
  - runs the job tracker and name node (in Hadoop 1)
  - and can be a slave too
- Slave/Worker Node:
  - responsible for storing the data and performing computations.
  - every slave/worker node runs both a TaskTracker
  - dataNode service to communicate with the Master node in the cluster.
  - the DataNode service is secondary to the NameNode
  - the TaskTracker service is secondary to the JobTracker.



 Source: https://www.dezyre.com/article/hadoop-cluster-overview-what-it-is-and-how-to-setup-one/356

Hadoop cluster Architecture

- Client Nodes:

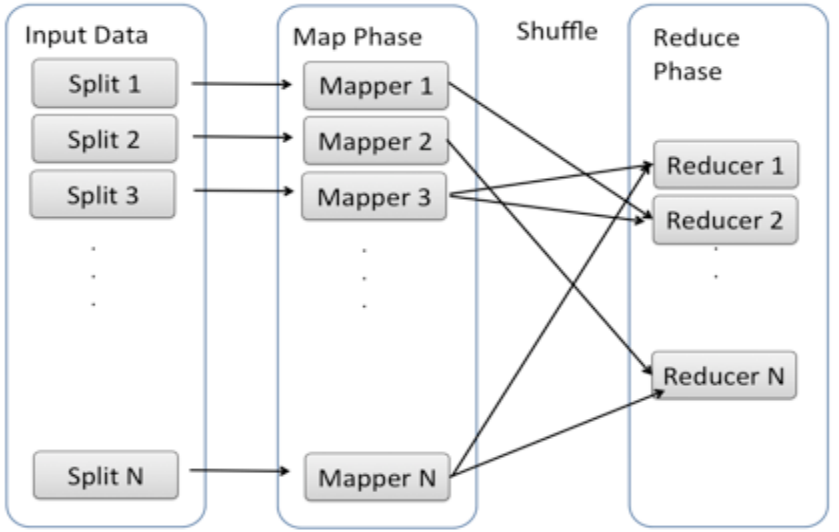
  - responsible for loading all the data into the hadoop cluster.

  - It submits mapreduce jobs describing on how data need to be processed and then the output is retrieved by the client node once the job processing is completed.

 Source: https://www.dezyre.com/article/hadoop-cluster-overview-what-it-is-and-how-to-setup-one/356

- Single JobTracker:
  - YARN ResourceManager (in Hadoop 2)
  - receives MapReduce jobs from client(application)-s
  - starts a YARN MRAppManager per application
  - pushes the work to task trackers close to the data
  - (simple) scheduling and rescheduling
- Multiple TaskTrackers:
  - YARN NodeManager
  - has task slots available (called containers)
  - spawns (lots of) separate JVMs
- ...so what are these tasks?

MapReduce



- Hadoop MapReduce is a software framework:
  - Designed by Google as a programming model
  - process vast amounts of data (multi-terabyte data-sets)
  - in parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.
- MapReduce program contains two tasks:
  - **Map**
  - **Reduce**

MapReduce programming model

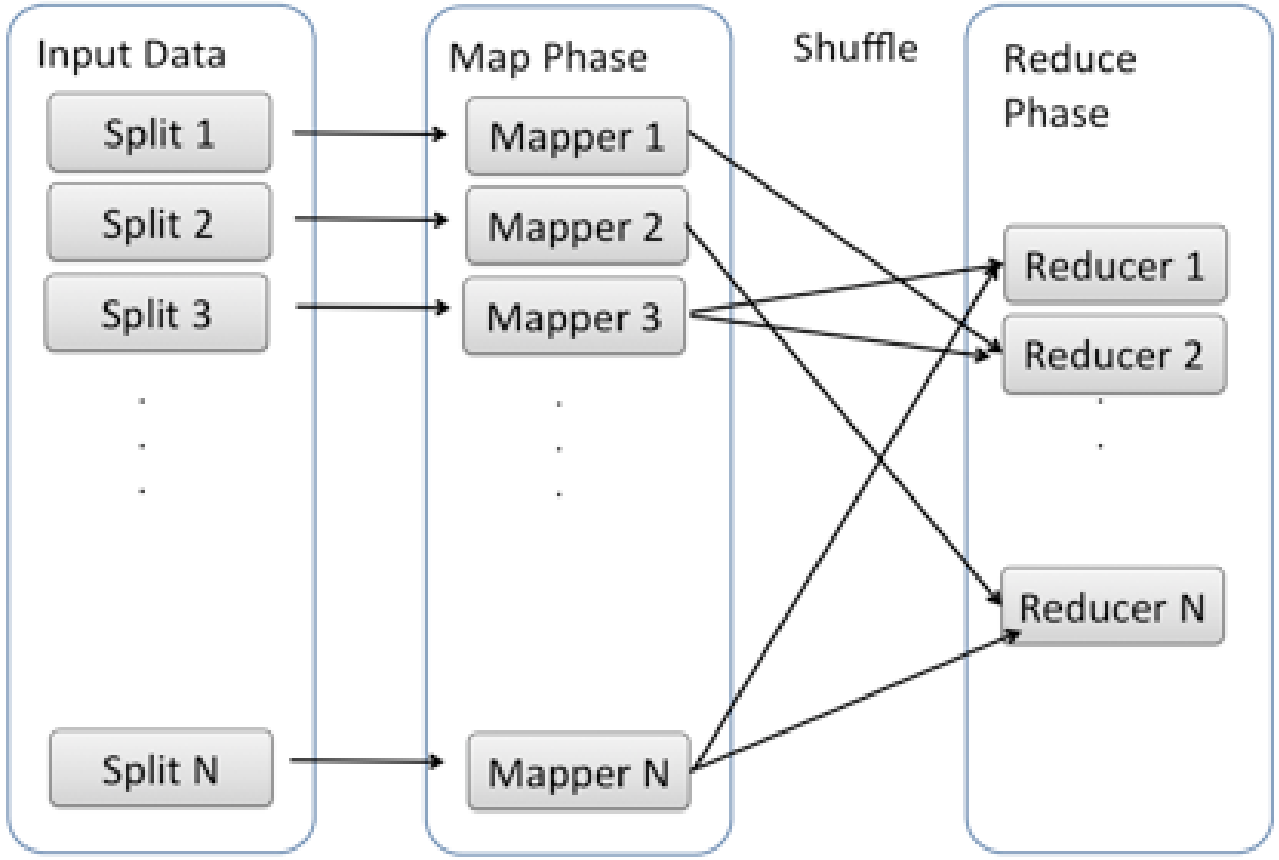
- Two main tasks with an intermediate step:

  – **Map:** filters and sorts local input data

  – **Shuffle:** redistributes intermediate data across nodes

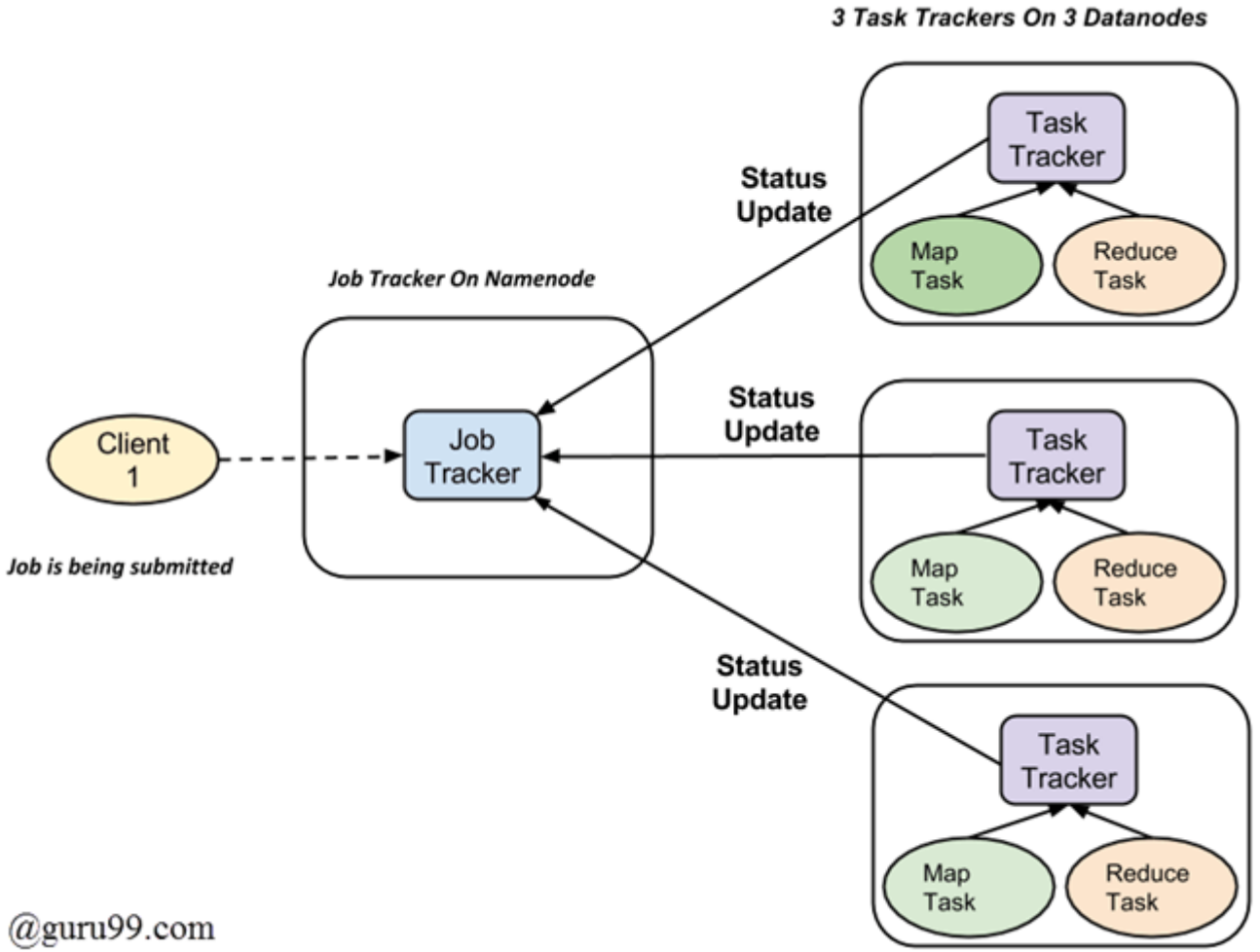
  – **Reduce:** summarizes the data in each node

  ...all three steps are parallel (and more can be added)



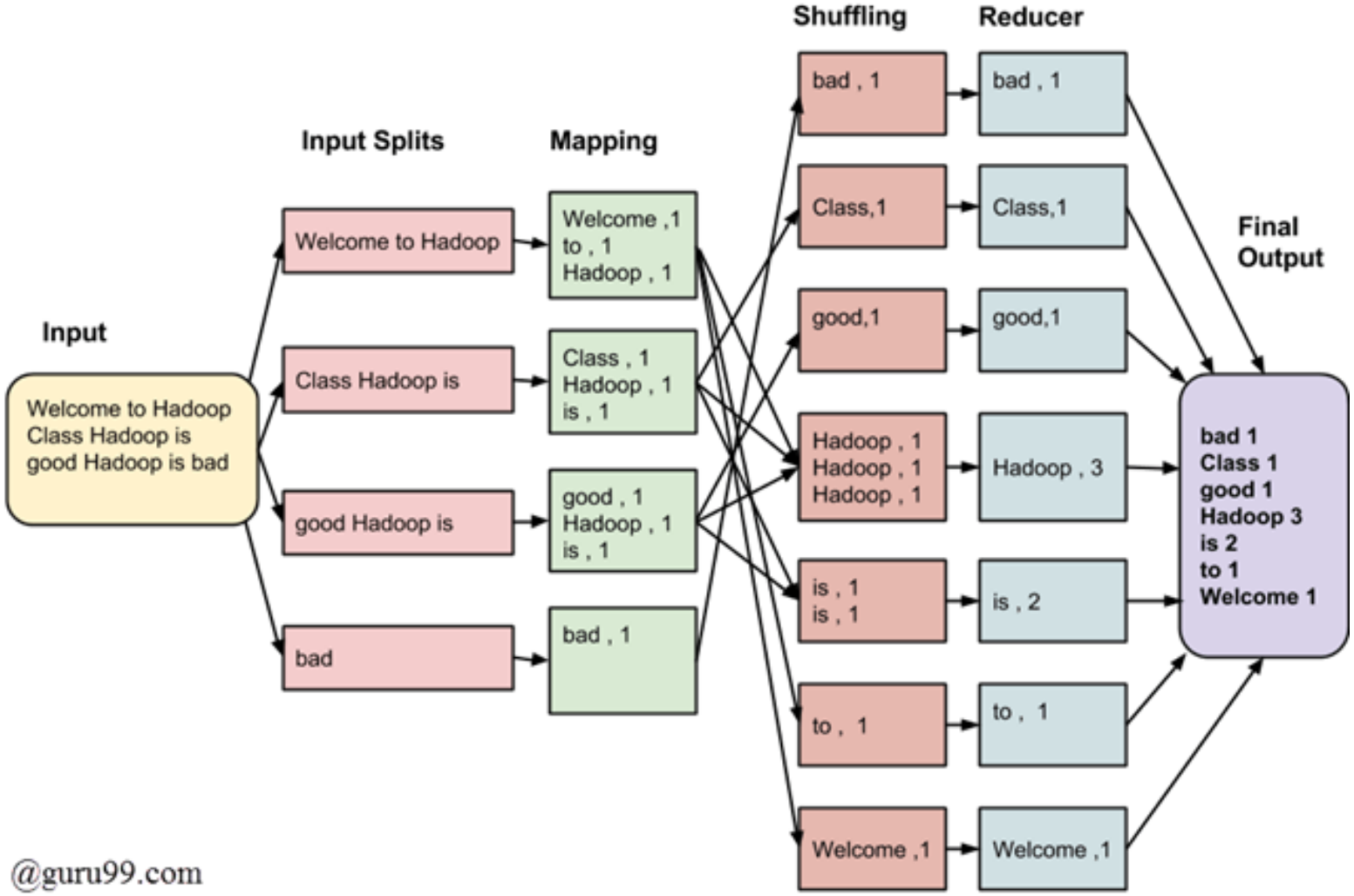
How MapReduce Organizes Work?

- The complete execution process:
- **Jobtracker** : Acts like a **master** (responsible for complete execution of submitted job)
- **Multiple Task Trackers** : Acts like **slaves,** each of them performing the job.



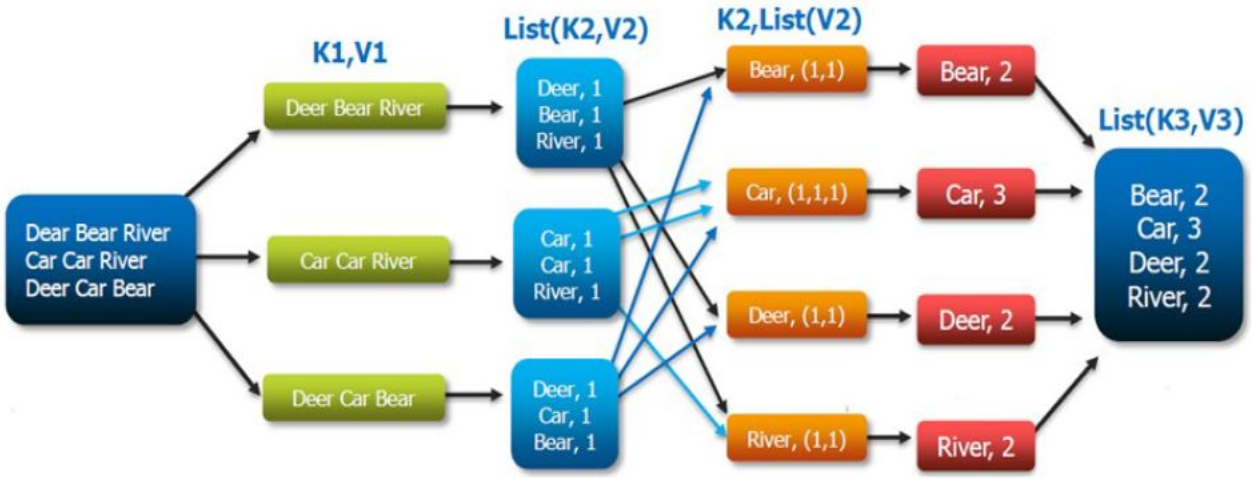
How MapReduce works

Consider you have following input data for your MapReduce Program
- Welcome to Hadoop Class
- Hadoop is good
- Hadoop is bad



Data are *(key, value)*-pairs

- Map(k1,v1) → list(k2,v2)
  - *Map(<file_id>,<text>) → list(<word>,<count>)*
- Reduce(k2, list (v2)) → (k2, v3 *(or list, or nothing)*)
  - *Reduce(<word>, list (<count>)) → (<word>, <count>)*
  - *associativity* and *commutativity are* helpful



Example

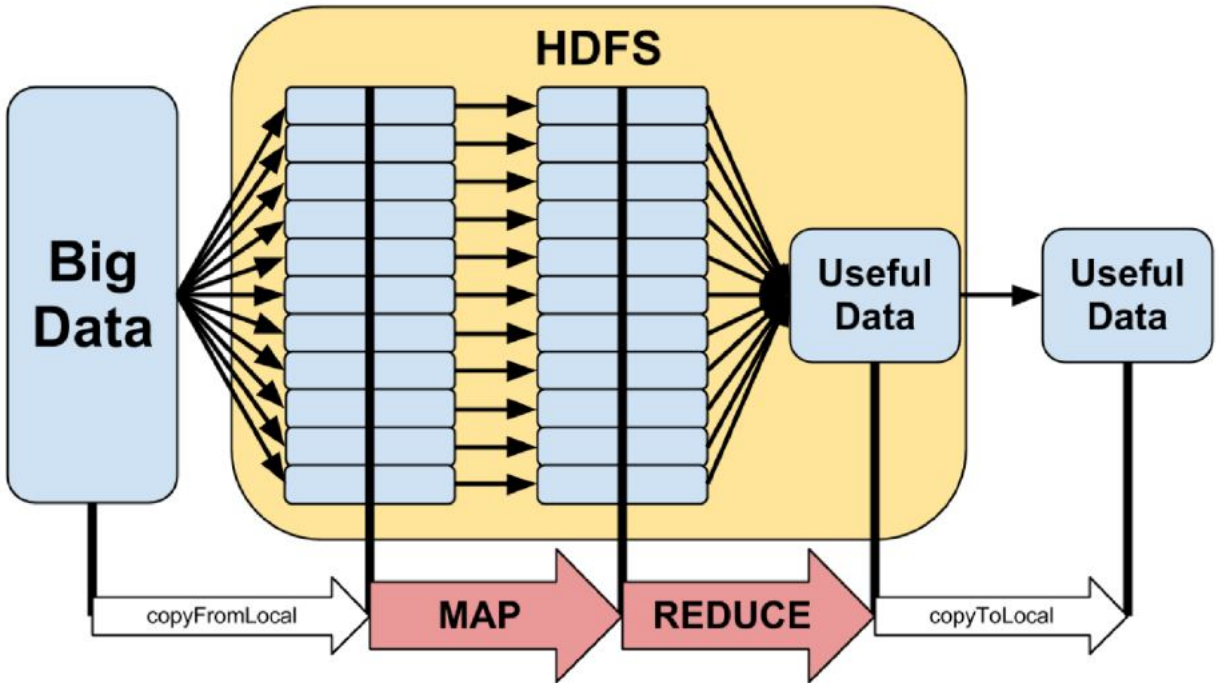
- Pseudocode:

```
function map(String name, String document):
// name: document name
// document: document contents
for each word w in document:
emit (w, 1)

function reduce(String word, Iterator partialCounts):
// word: a word
// partialCounts: a list of aggregated partial counts
sum = 0
for each pc in partialCounts:
sum += pc
emit (word, sum)
```

MapReduce and HDFS

- MapReduce can run on other file systems...
- HDFS can support other computing models...



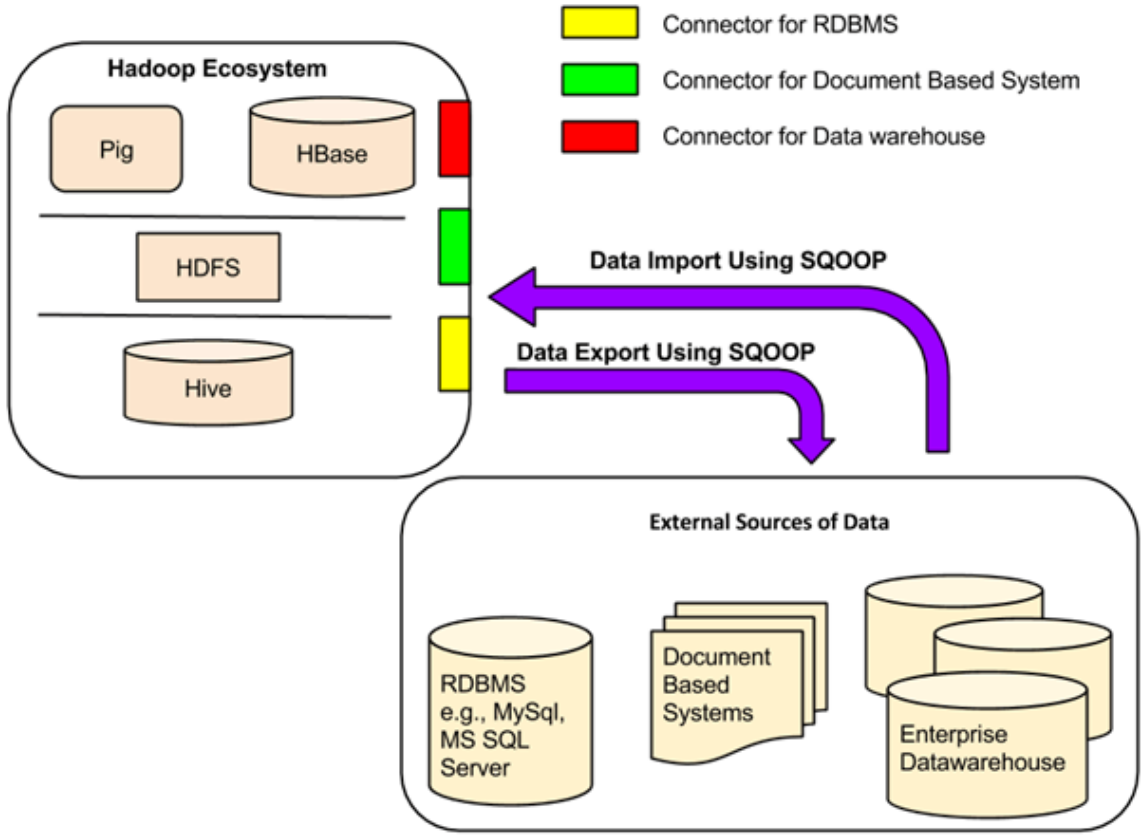
Data Load into Hadoop

- Work with enormous data in several different forms and load such heterogeneous data into Hadoop, different tools have been developed.

  - Sqoop and **Flume** .

What is SQOOP in Hadoop?

- Designed to support bulk import of data into HDFS
  - from structured data stores
  - relational databases, enterprise data warehouses, and NoSQL systems.
- Sqoop Connectors
  - Data transfer between Sqoop and external storage system



What is FLUME in Hadoop?

- Used for moving massive quantities of streaming data into HDFS

  - E.g., . Collecting log data present in log files from web servers and aggregating it in HDFS for analysis

- Flume supports multiple sources like –

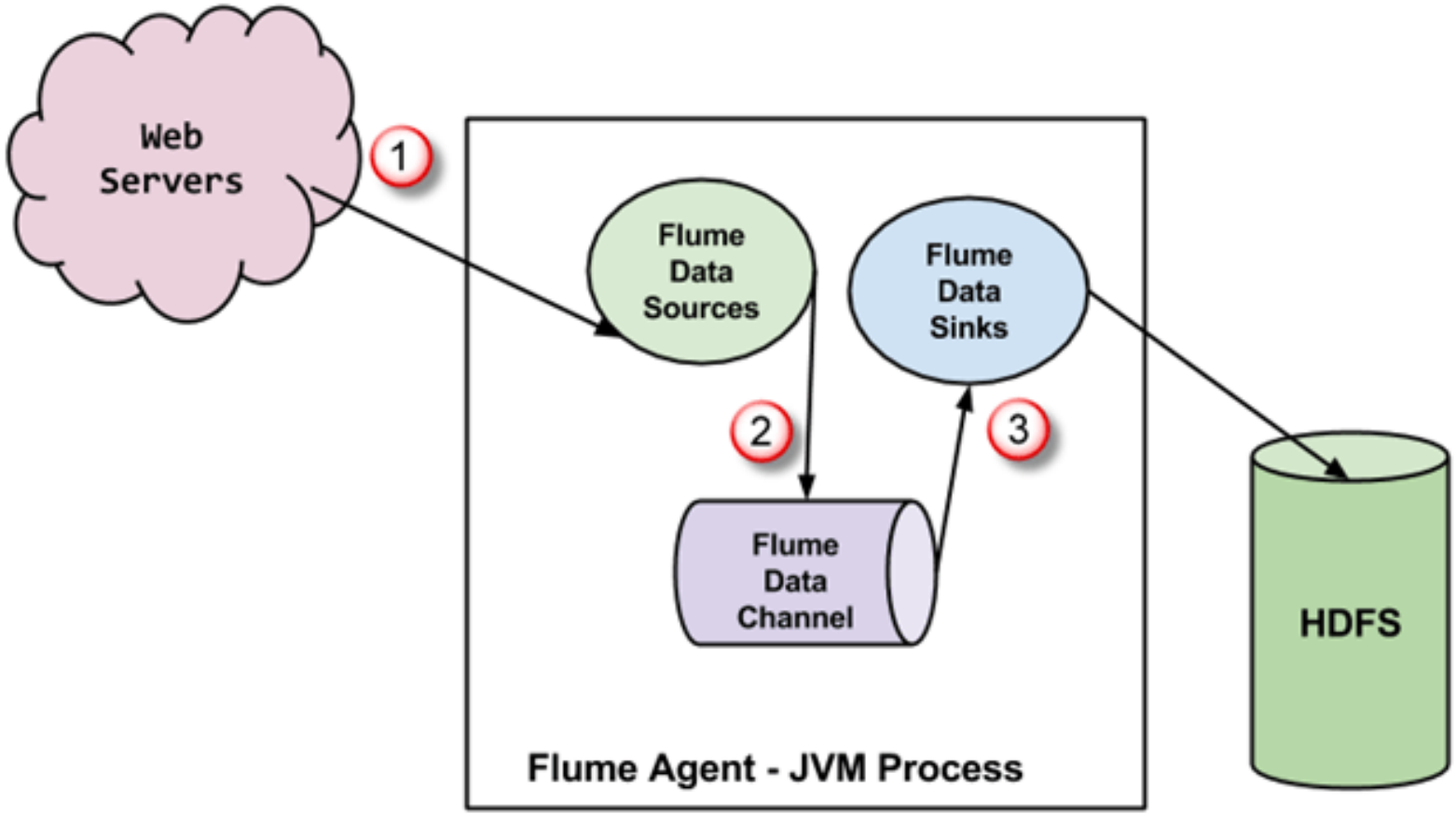
  - 'tail' (which pipes data from local file and write into HDFS via Flume, similar to Unix command 'tail')

  - System logs

  - Apache log4j (enable Java applications to write events to files in HDFS via Flume).

Data Flow in Flume

A **Flume agent** is a **JVM** process which has 3 components -**Flume Source**, **Flume Channel** and **Flume Sink**- through which events propagate after initiated at an external source .



Technology Comparison

Sqoop	Flume	HDFS
Sqoop is used for importing data from structured data sources such as RDBMS.	Flume is used for moving bulk streaming data into HDFS.	HDFS is a distributed file system used by Hadoop ecosystem to store data.
Sqoop has a connector based architecture. Connectors know how to connect to the respective data source and fetch the data.	Flume has an agent based architecture. Here, code is written (which is called as 'agent') which takes care of fetching data.	HDFS has a distributed architecture where data is distributed across multiple data nodes.
HDFS is a destination for data import using Sqoop.	Data flows to HDFS through zero or more channels.	HDFS is an ultimate destination for data storage.
Sqoop data load is not event driven.	Flume data load can be driven by event.	HDFS just stores data provided to it by whatsoever means.
In order to import data from structured data sources, one has to use Sqoop only, because its connectors know how to interact with structured data sources and fetch data from them.	In order to load streaming data such as tweets generated on Twitter or log files of a web server, Flume should be used. Flume agents are built for fetching streaming data.	HDFS has its own built-in shell commands to store data into it. HDFS can not import streaming data

Flume Tutorial

- https://www.guru99.com/create-your-first-flume-program.html

Limitations of Hadoop

- Hadoop can perform **only batch processing**, and data will be accessed only in a sequential manner. That means one **has to search the entire dataset** even for the simplest of jobs.
- A huge dataset when processed **results in another huge data set**, which should also be processed sequentially. At this point, a new solution is needed to access any point of data in a single unit of time (random access).

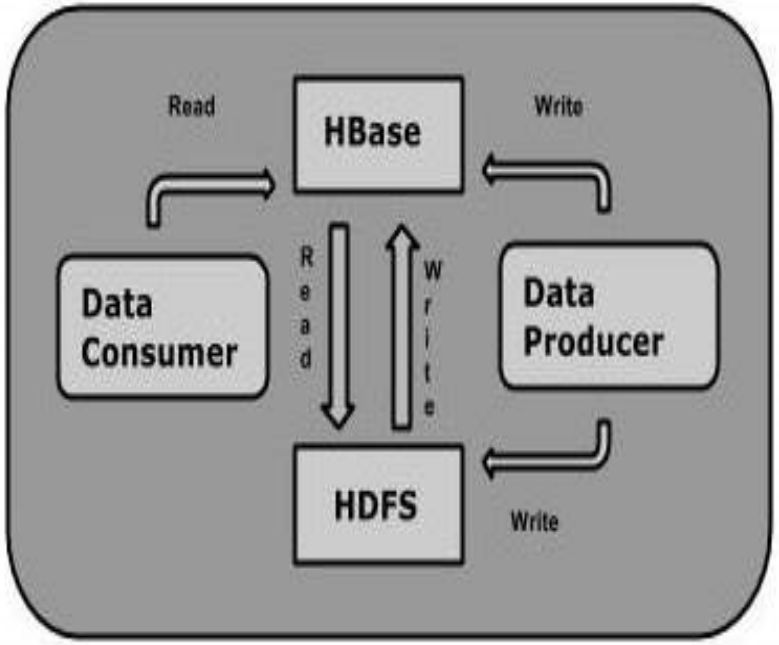
  - Hadoop Random Access Databases: HBase, Cassandra, couchDB, Dynamo, and MongoDB

What is HBase?

- It is a distributed column-oriented database built on top of the Hadoop file system.
- It is a data model that is similar to Google's big table designed to provide quick random access to huge amounts of structured data.
- The fault tolerance provided by the Hadoop File System (HDFS).
- It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in the Hadoop File System.
- One can store the data in HDFS either directly or through HBase.
- Data consumer reads/accesses the data in HDFS randomly using HBase.
- It sits on top of the Hadoop File System and provides read and write access.

Hbase and HDFS

HDFS	HBase
HDFS is a distributed file system suitable for storing large files.	HBase is a database built on top of the HDFS.
HDFS does not support fast individual record lookups.	HBase provides fast lookups for larger tables.
It provides high latency batch processing; no concept of batch processing.	It provides low latency access to single rows from billions of records (Random access).
It provides only sequential access of data.	HBase internally uses Hash tables and provides random access, and it stores the data in indexed HDFS files for faster lookups.

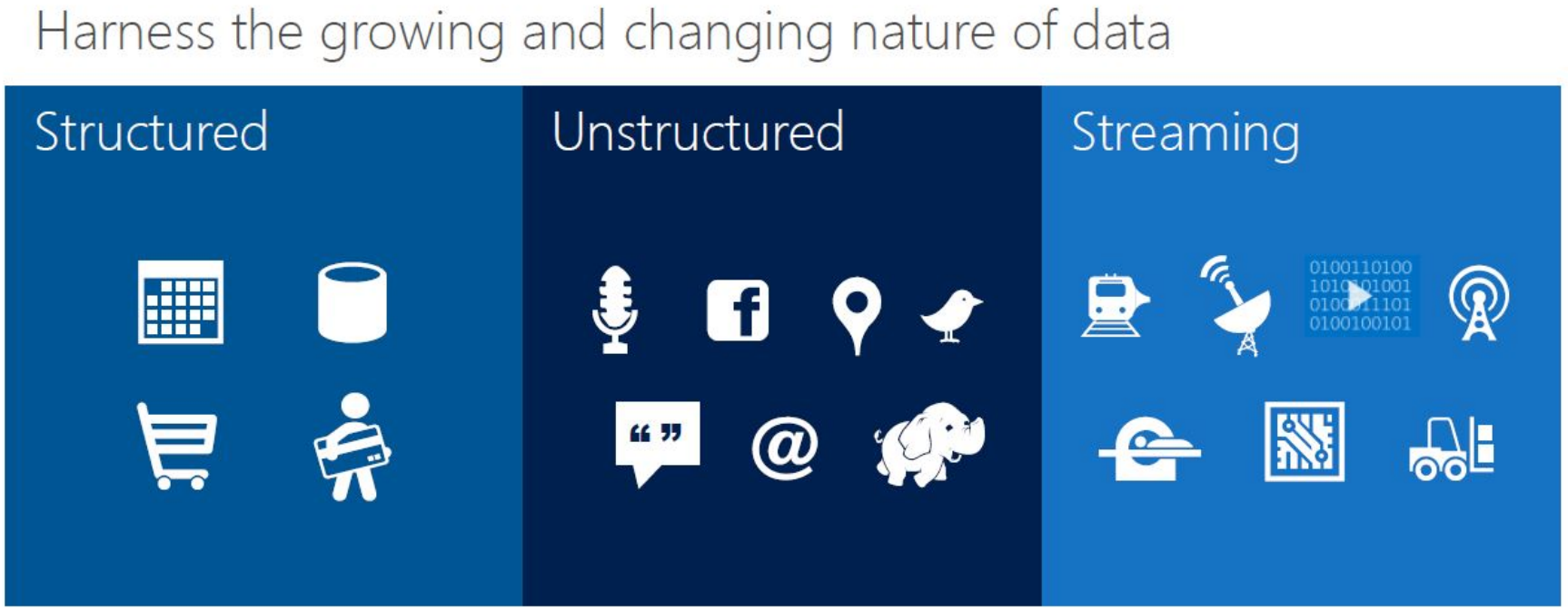


Emergency Data sources

- data.norge.no (Open Public Data in Norway)
  - ...or other public data sources (EU, other...)
- There are lots of open data out there
  - data.gov
  - GIS datasets

  https://researchguides.dartmouth.edu/gisdata/disasterdata
  - Social media
  - but not so much of it is in semantic formats

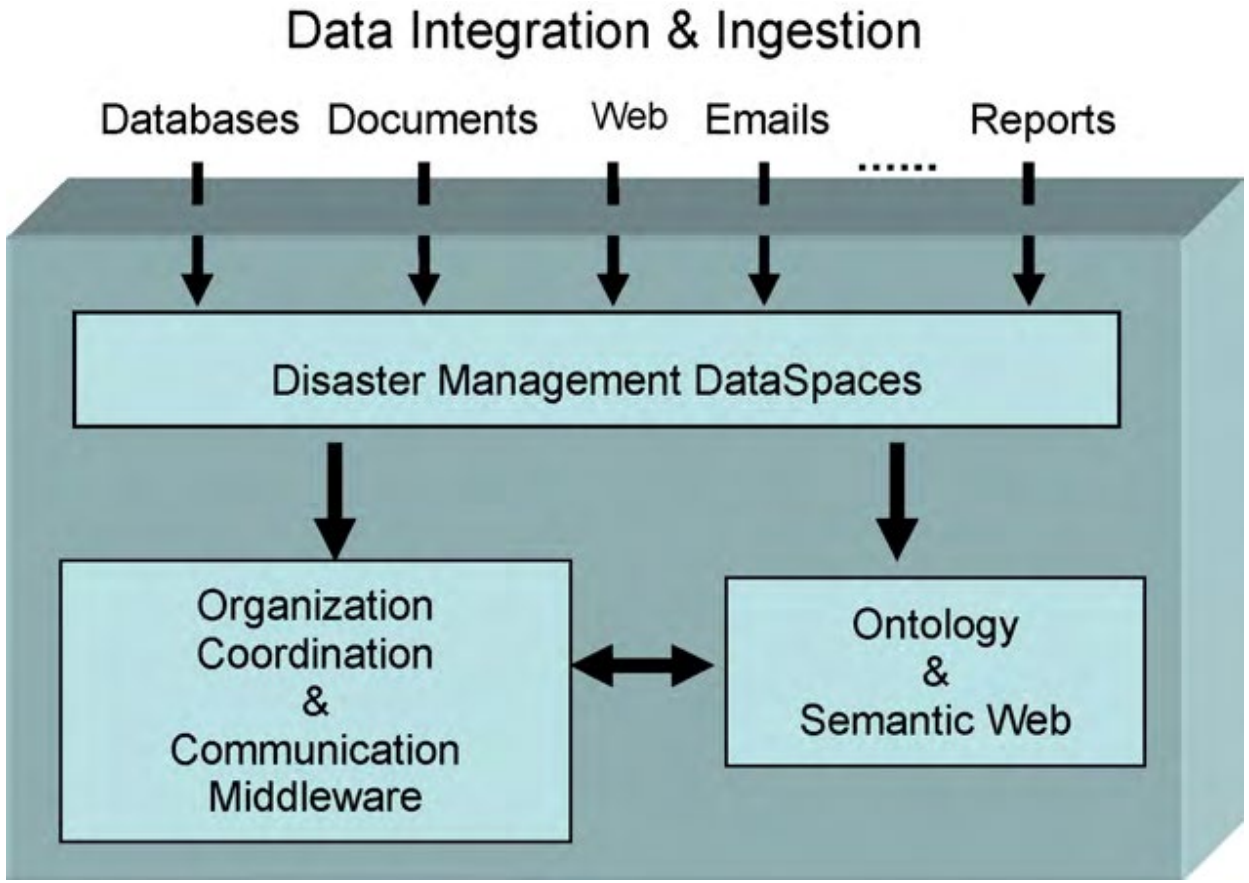
Addressing the problem of big data variety



- Challenge is combining transactional data stored in relational databases with less structured data
- Big Data = All Data
- Get the right information to the right people at the right time in the right format

Why big data integration is difficult?

- Disconnected data
- Unable to handle complexity
- Impediment to big data success is having too many silos.



Big data integration challenges

- Integrating diverse data sources is the ability to import different data formats into a common representation.
- When the sources are large it is not possible to read an entire source into main memory.

Problems With the Relational Approach

Inflexible Data Model

- Everything modeled up front
- Schema complexity
- Difficult to make changes later
- Fixed to a specific business purpose
- Lots of expensive ETL
- Inability to store unstructured data
- Mismatch for modern app development

Inability to Model Relationships

- No standard for modeling people, places, things
- Lack of context within taxonomies/ontologies

Inability to Query Heterogeneous Data

- Inability to handle complex queries across varied data

Limited Scalability

- Scale up, not out

Semantics

- Semantics plays a substantial role in the integration of Big Data.
- Semantics is the study of meaning.
- In the context of big data, semantics is the technique of creating meaning full assertions about data objects.

  - all meaningful assertions can be structured as a three item list consisting of an identified data object , a data value, and a descriptor for the data value.

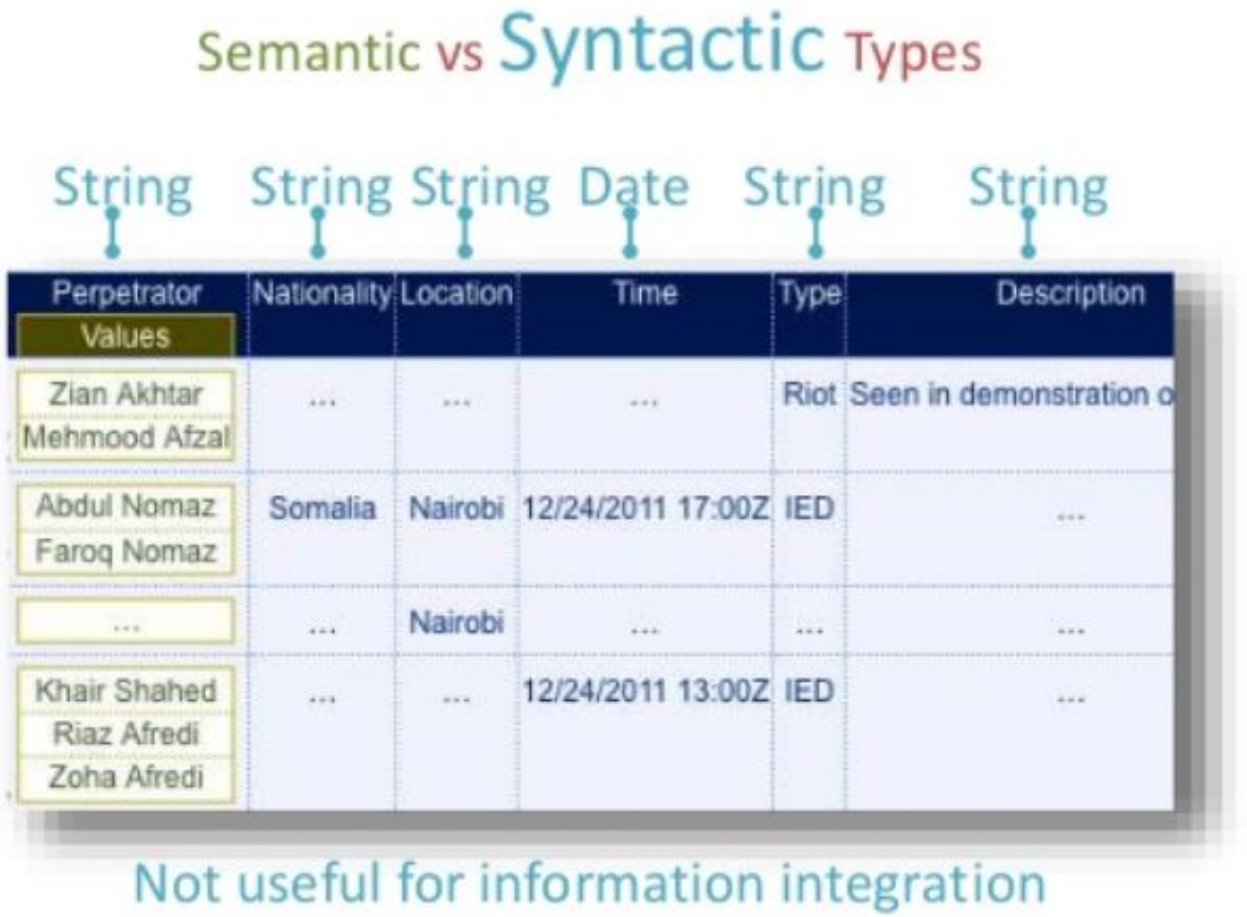
  e.g., <james><height in feet inches><5’11’’>

  e.g., every cell in a spreadsheet is a data value that has a descriptor (the column header) and a subject (the row identifier).

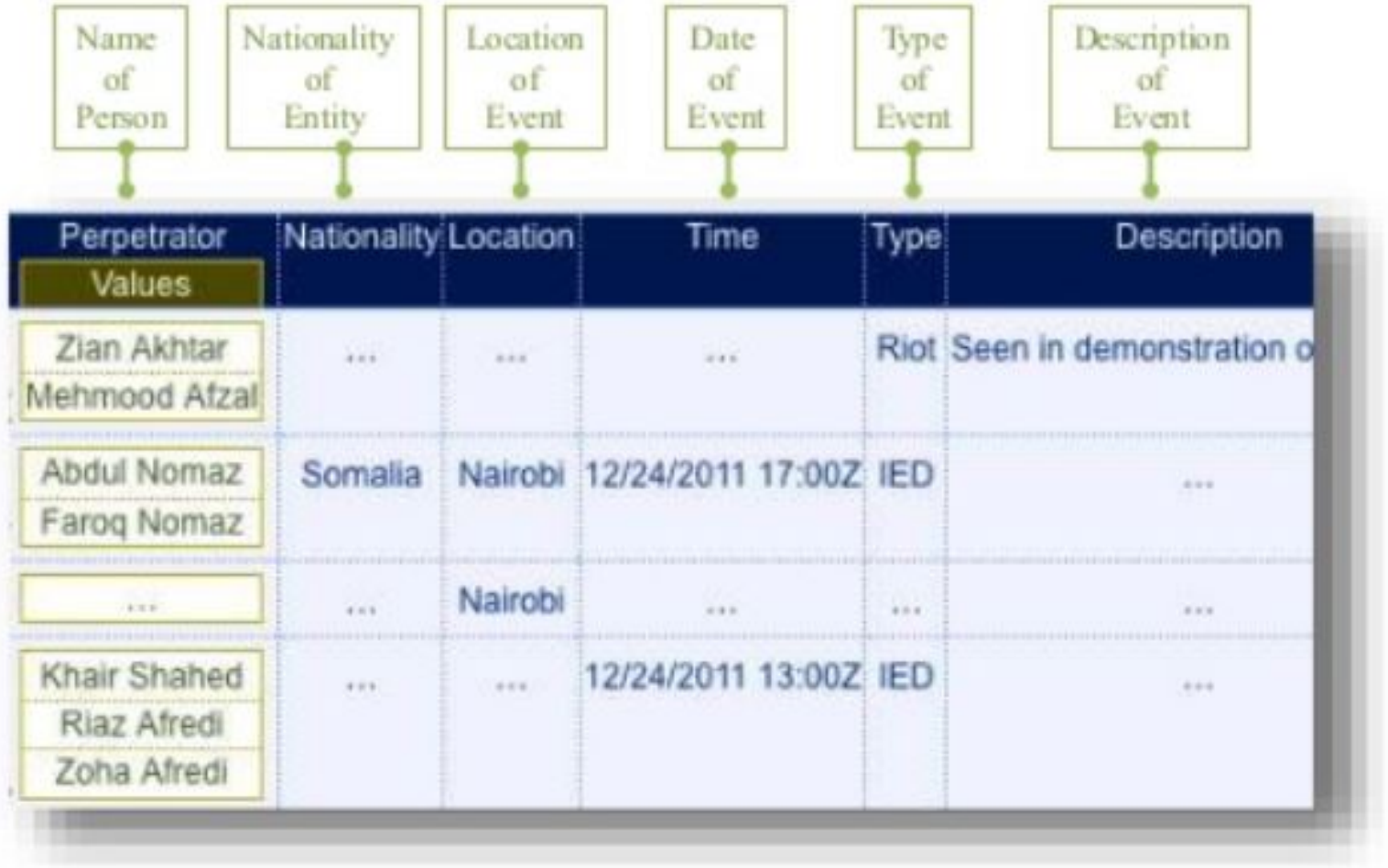
  <row identifier><column header><content of cell>

Semantic Types

- Meaning of data in columns:



Semantic vs Syntactic Types



Semantic Technology

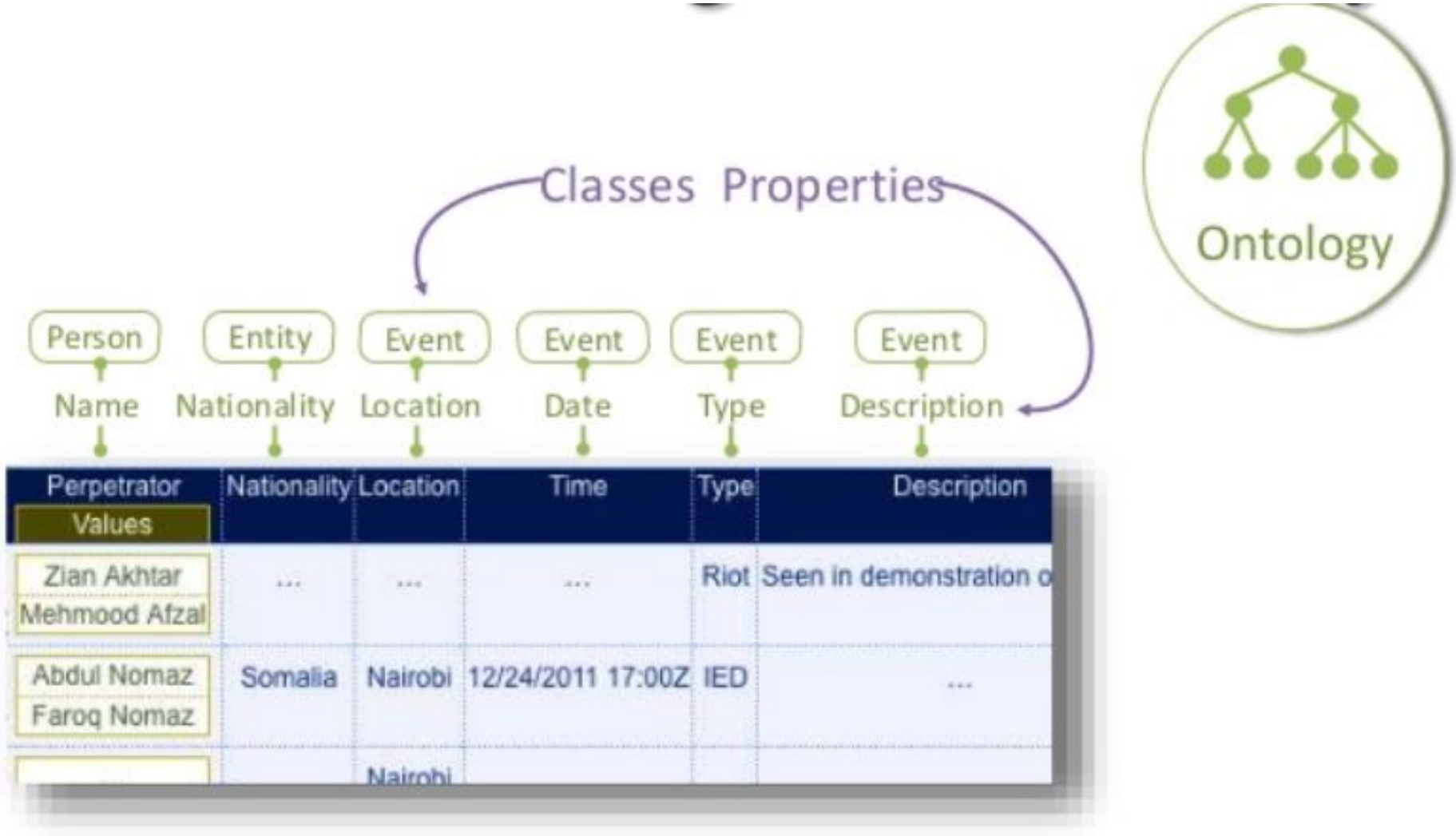
- Semantic Technology uses formal semantics to give meaning to the disparate and raw data.

  - builds relationships between data in various formats and sources, from one string to another, helping build context and creating links out of these relationships.

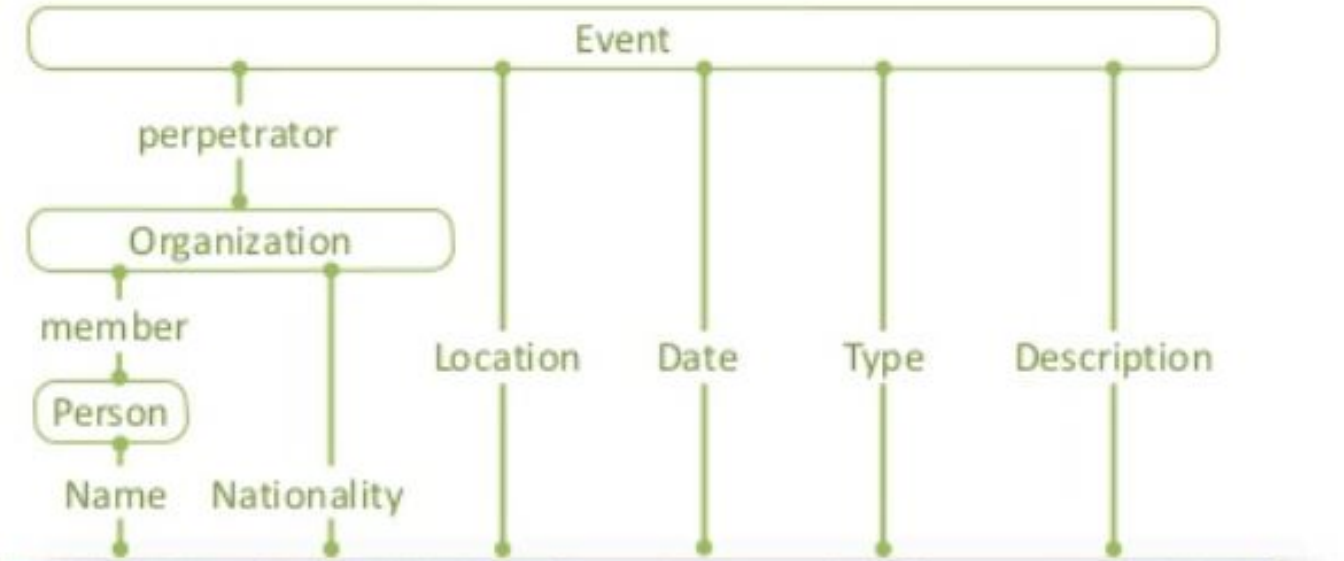
- Defined by the **World Wide Web Consortium (W3C)** international community.
- The purpose of this technology in the context of the Semantic Web is to create a '*universal medium for the exchange of data*' by smoothly interconnecting the global sharing of any kind of personal, commercial, scientific and cultural data.

- The main standards that Semantic Technology builds on are:
  - the Resource Description Framework (RDF),
  - SPARQL (SPARQL Protocol and RDF Query Language) and,
  - optionally, OWL (Web Ontology Language).
- **RDF** is the format Semantic Technology uses
  - to store data on the Semantic Web or in a semantic graph database.

- **SPARQL** is the semantic query language specifically designed to
  - query data across various systems and databases, and
  - to retrieve and process data stored in RDF format.
- **OWL** is the computational logic-based language
  - is designed to show the data schema and that represents rich and complex knowledge about hierarchies of things and the relations between them.
  - It is complementary to RDF and allows for formalizing a data schema/ontology in a given domain, separately from the data.


Ontology
Classes Properties
Person
Entity
Event
Event
Event
Event
Name
Nationality
Location
Date
Type
Description
Perpetrator
Values
Nationality
Location
Time
Type
Description
Zian Akhtar
Mehmood Afzal
...
...
...
Riot
Seen in demonstration
Abdul Nomaz
Faroq Nomaz
Somalia
Nairobi
12/24/2011 17:00Z
IED
...
Nairobi

Relationships specified in Terms of classes and properties



Perpetrator Values	Nationality	Location	Time	Type	Description
Zian Akhtar Mehmood Afzal	...	...	...	Riot	Seen in demonstration c
Abdul Nomaz Faroq Nomaz	Somalia	Nairobi	12/24/2011 17:00Z	IED	...
		Nairobi			

MapReduce and semantic technologies

- Not much explored so far:
  - the largest linked open datasets were smaller
  - focus on native triple stores
  - focus automated reasoning
- More interest in big data + semantics in recent years
  - more research papers since ~2014
  - *linked big data*
  - higher-capacity triple stores
  - Apache Jena Elephas (early beta):
- supports Hadoop MapReduce with Jena
  - https://jena.apache.org/documentation/hadoop/
  - also, e.g., Apache Giraph for graph processing

Use Case: Earthquake Detection using Spark

- **Problem Statement**: *To design a Real Time Earthquake Detection Model to send life saving alerts, which should improve its machine learning to provide near real-time computation results.*

- **Use Case – Requirements**:
  - Process data in real-time
  - Handle input from multiple sources
  - Easy to use system
  - Bulk transmission of alerts

Use Case – Dataset:

EARTHQUAKE ROC DATASET																	
	S Wave							P Wave									
Classification Index	First Activity	Time Taken	Acceleration	Building Strength	Velocity	Sa	Sd	First Activity	Time Taken	Acceleration	Building Strength	Velocity	Sa	Sd	Total Weight	Sum * ROC	ROC	AVG * ROC
0	3	11	14	19	39	42	55	64	67	73	75	76	80	83	701	618.168	0.881837	623.2843
0	3	6	17	27	35	40	57	63	69	73	74	76	81	103	724	638.4503	0.881837	623.2843
0	4	6	15	21	35	40	57	63	67	73	74	77	80	83	695	612.877	0.881837	623.2843
0	5	6	15	22	36	41	47	66	67	72	74	76	80	83	690	608.4678	0.881837	623.2843
0	2	6	16	22	36	40	54	63	67	73	75	76	80	83	693	611.1133	0.881837	623.2843
0	2	6	14	20	37	41	47	64	67	73	74	76	82	83	686	604.9405	0.881837	623.2843
0	1	6	14	22	36	42	49	64	67	72	74	77	80	83	687	605.8223	0.881837	623.2843
0	1	6	17	19	39	42	53	64	67	73	74	76	80	83	694	611.9952	0.881837	623.2843
0	2	6	18	20	37	42	48	64	71	73	74	76	81	83	695	612.877	0.881837	623.2843
1	5	11	15	32	39	40	52	63	67	73	74	76	78	83	708	624.3409	0.881837	623.2843
0	5	16	30	35	41	64	67	73	74	76	80	83			644	567.9033	0.881837	623.2843
0	5	6	15	20	37	40	50	63	67	73	75	76	80	83	690	608.4678	0.881837	623.2843
0	5	7	16	29	39	40	48	63	67	73	74	76	78	83	698	615.5225	0.881837	623.2843
0	1	11	18	20	37	42	59	62	71	72	74	76	80	83	706	622.5772	0.881837	623.2843
1	5	18	19	39	40	63	67	73	74	76	80	83			637	561.7304	0.881837	623.2843

Figure: *Use Case – Earthquake Dataset*

Use Case – Flow Diagram:

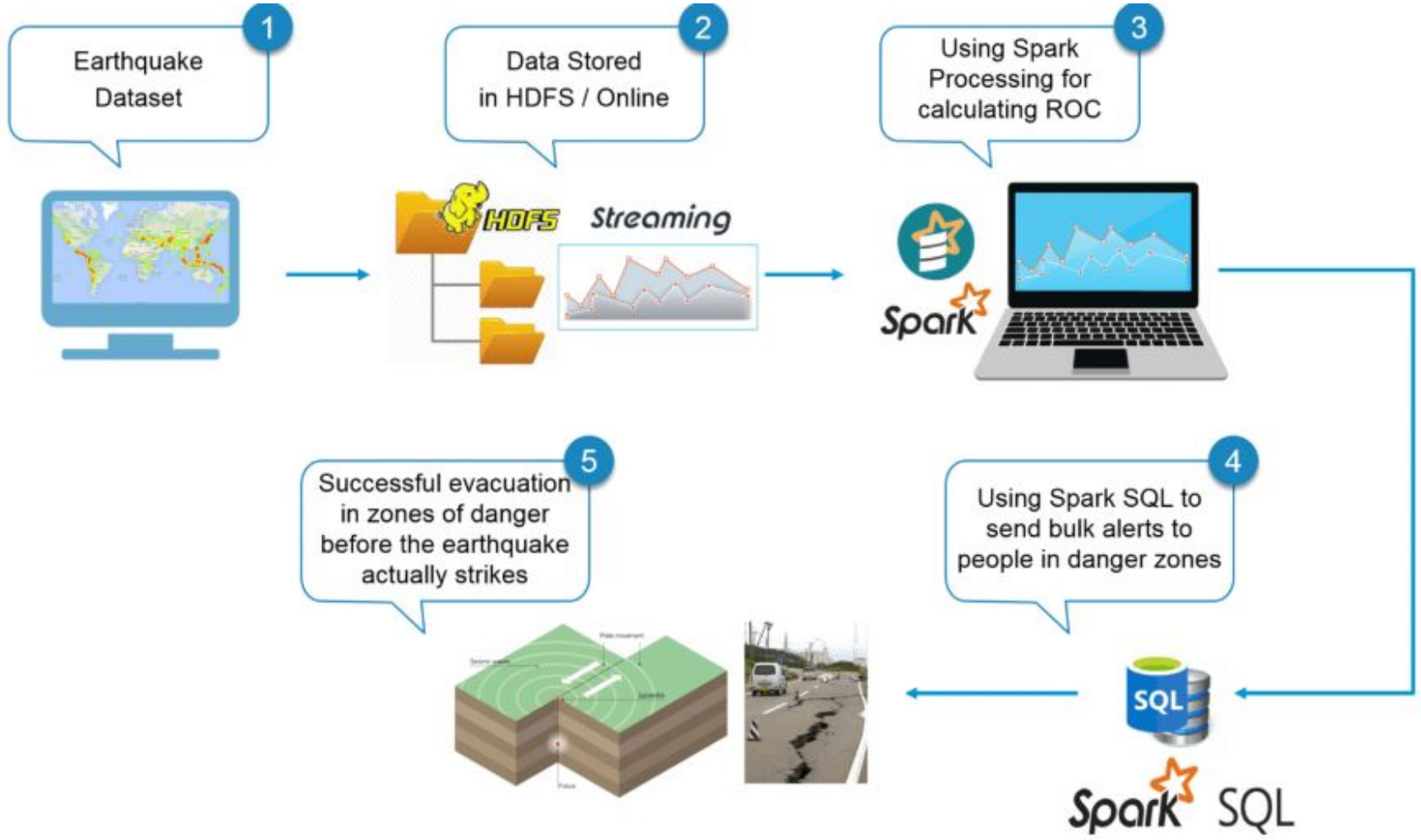


Figure: *Use Case – Flow diagram of Earthquake Detection using Apache Spark*

Use Case – Spark Implementation:

Find the Pseudo Code below:

```
//Importing the necessary classes
import org.apache.spark._
...
//Creating an Object earthquake
object earthquake {
 def main(args: Array[String]) {

//Creating a Spark Configuration and Spark Context
val sparkConf = new SparkConf().setAppName("earthquake").setMaster("local[2]")
val sc = new SparkContext(sparkConf)

//Loading the Earthquake ROC Dataset file as a LibSVM file
val data = MLUtils.loadLibSVMFile(sc, *Path to the Earthquake File* )

//Training the data for Machine Learning
val splits = data.randomSplit( *Splitting 60% to 40%* , seed = 11L)
val training = splits(0).cache()
val test = splits(1)

//Creating a model of the trained data
val numIterations = 100
val model = *Creating SVM Model with SGD* (  *Training Data* , *Number of Iterations* )

//Using map transformation of model RDD
val scoreAndLabels = *Map the model to predict features*

//Using Binary Classification Metrics on scoreAndLabels
val metrics = * Use Binary Classification Metrics on scoreAndLabels *(scoreAndLabels)
val auROC = metrics. *Get the area under the ROC Curve*()

//Displaying the area under Receiver Operating Characteristic
println("Area under ROC = " + auROC)
 }
}
```

Run the Tutorial:

https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

Check out the JavaDoc:

http://hadoop.apache.org/docs/r2.7.4/api/index.html

What to do in Two Weeks?

...and in the meantime :-)