ĐẠI HỌC BÁCH KHOA

25 YEARS ANNIVERSARY SOICT

HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

IT4142E
Introduction to Data Science

Chapter 7: Introduction to Big data Analysis

Lecturer:
Muriel VISANI: murielv@soict.hust.edu.vn

Acknowledgements:
Khoat Than
Viet-Trung Tran

Department of Information Systems
School of Information and Communication Technology - HUST

Contents of the course

- Chapter 1: Overview
- Chapter 2: Data scraping
- Chapter 3: Data cleaning, pre-processing and integration
- Chapter 4: Introduction to Exploratory Data Analysis
- Chapter 5: Introduction to Data visualization
- Chapter 6: Introduction to Machine Learning
    - Performance evaluation
- Chapter 7: Introduction to Big Data Analysis
- Chapter 8: Applications to Image and Video Analysis

Goals of this chapter

Goal	Description of the goal
M1	**Understand and be able to design and manage the systems which are based on Data Science (DS)**
M1.2	Identify, compare, and categorize the data type and systems in practice
M1.3	Be able to design systems based on DS in their future organizations

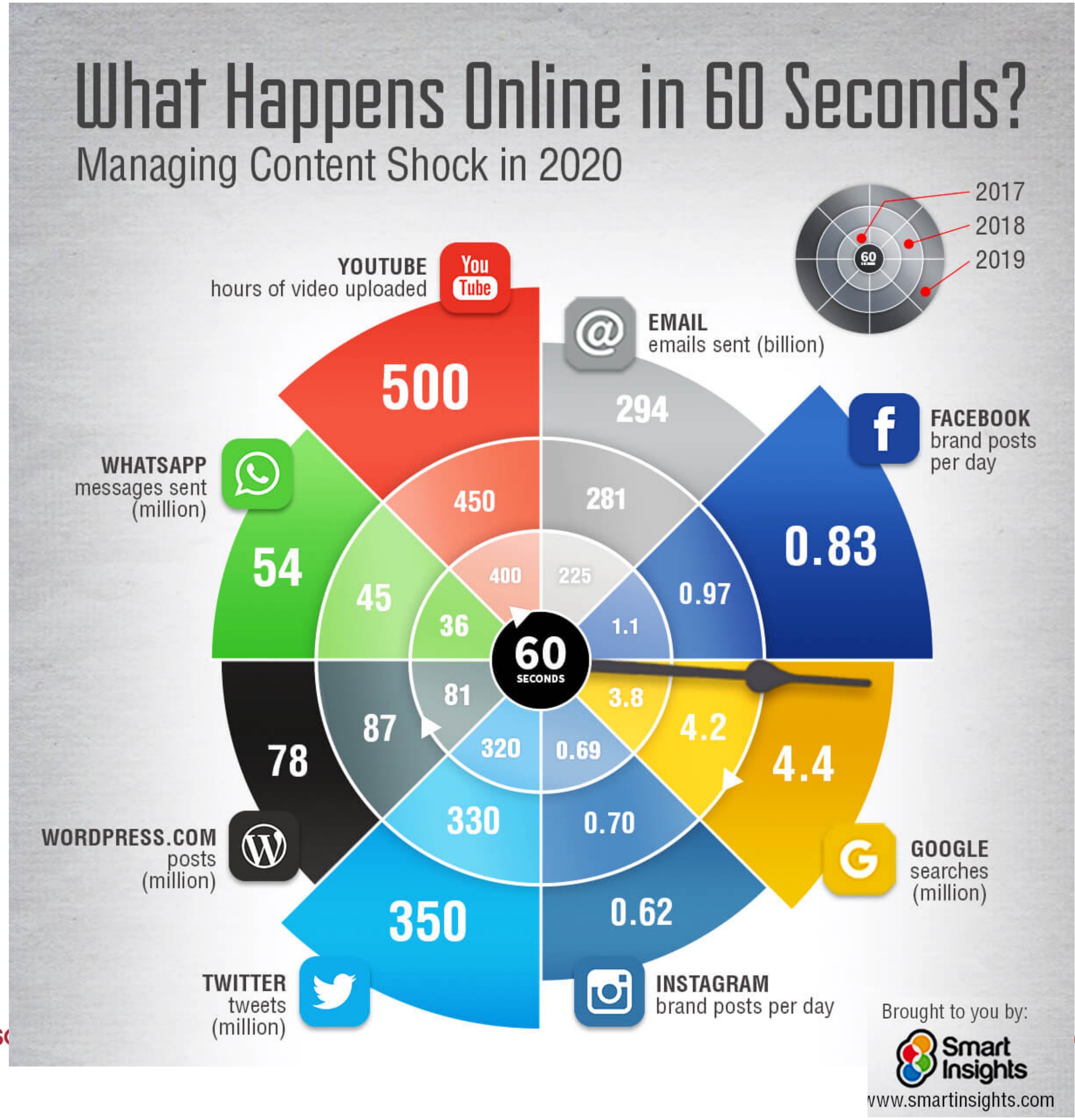
Contents of this chapter

- **Big data analysis**
  - Recall: definitions, numbers and challenges
  - Technological solutions
    - Hadoop
    - Spark

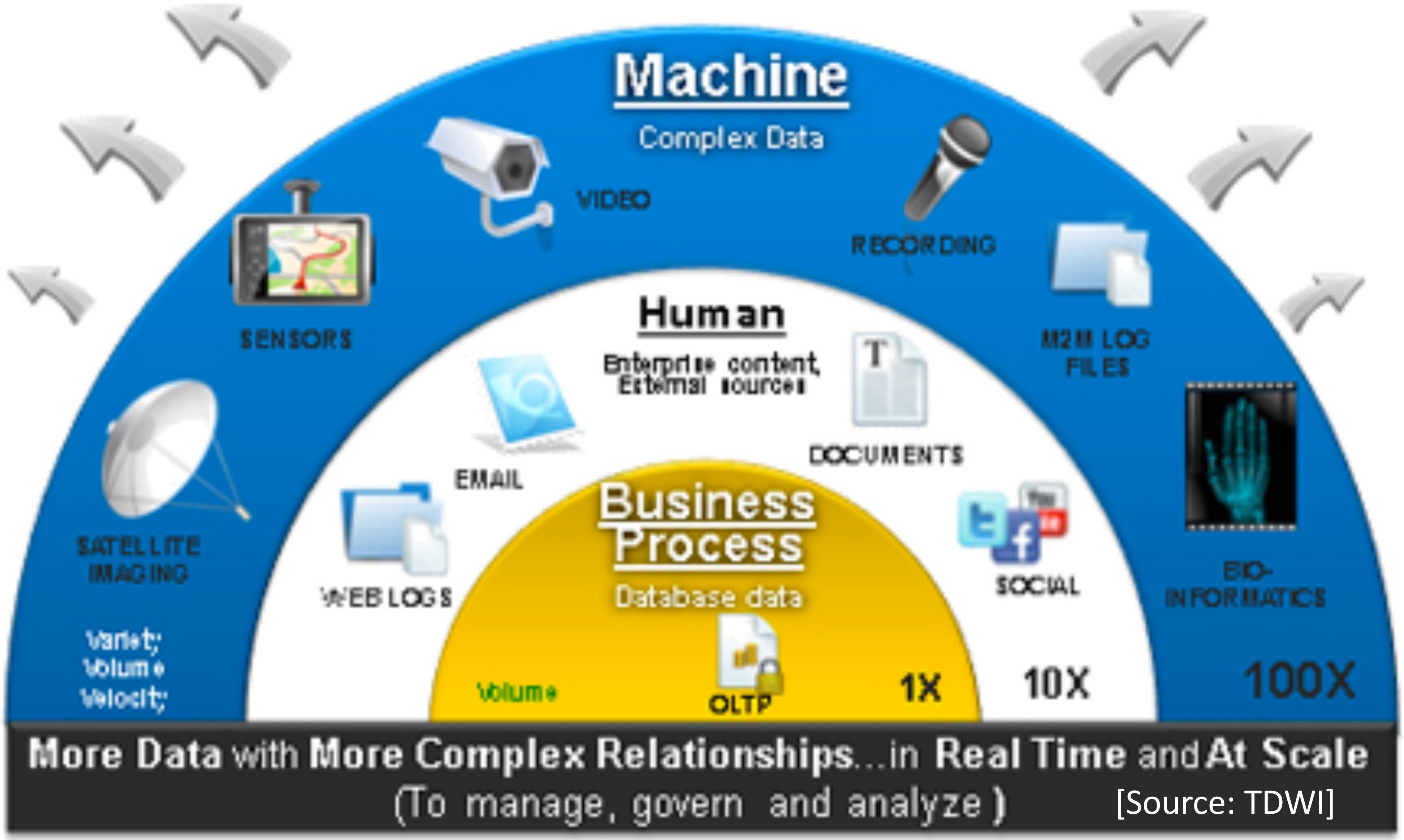
Big data analysis

Recall: definitions, numbers and challenges

Big data today: some numbers



Big data – today: sources



[Source: TDWI]

Big data – today: tentative definition

- Big data usually refers to a set of data that is VERY large
  - In terms of number of variables and/or
  - In terms of number of records
- This data might be structured, semi-structured or unstructured
  - Notion of data complexity
- Big data often comes from multiple sources
  - Raises the problems related to data homogeneity
- Big data is most often used for
  - Exploratory Data Analysis
  - Machine Learning

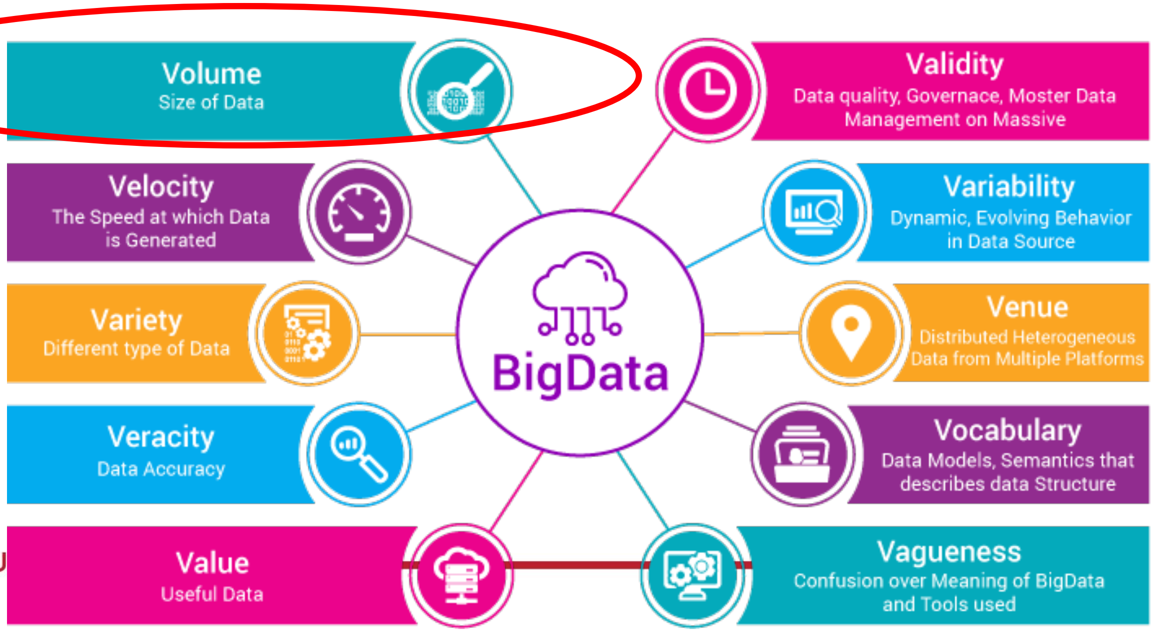
The 10 Vs of Big data

- In this course, we will focus mainly on the 3 Vs of Volume, Velocity and Variety

[Source: houseofbots.com]

The 10 Vs of Big data: Volume

- Volume is probably the best known characteristic of big data
- More than 90% of all today's data was created in the past 2 years
- Poses challenges in terms of:
  - Exploratory Data Analysis (see Chapter 4)
  - Data visualization (see Chapter 5)

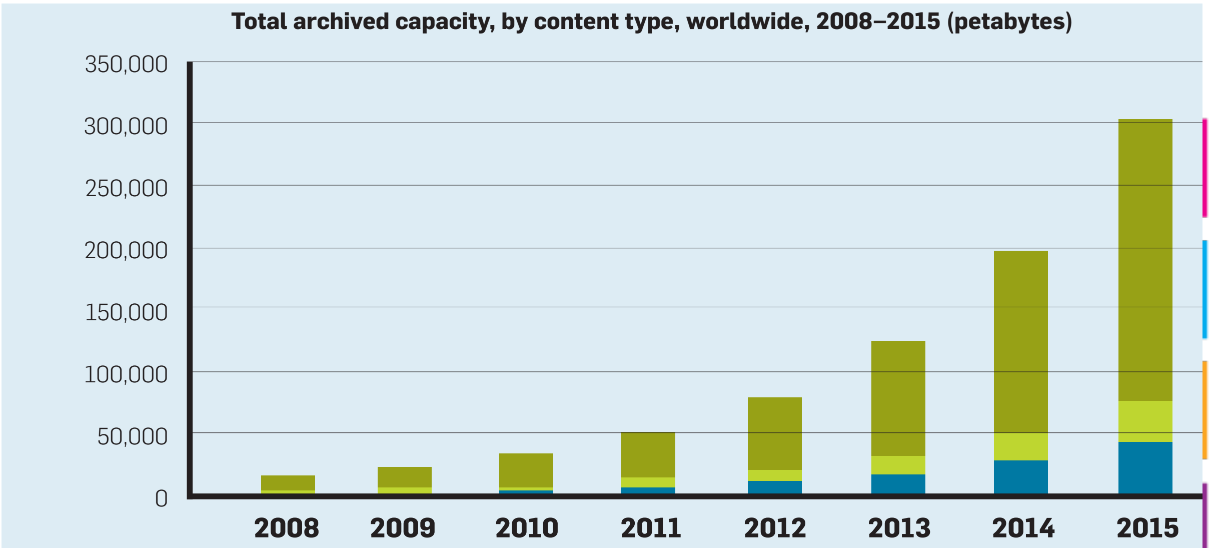


The 10 Vs of Big data: Velocity

- Velocity refers to the speed at which data is being generated, produced, created, or refreshed
  - It is ever-increasing, contributing to exponentional growth in the data volume!
  - It poses several challenges in terms of data integration (*cf* Chapter 3)
- It can be extended to the speed at which the data can be processed (notion of **complexity**)

The 10 Vs of Big data: Variety

- Variety refers to the different kinds of data one has to hande:
  - **Structured** data: from OLTP datasets of Excel files for instance
  - **Unstructured** data increases extremely fast: texts, images, tags, links, likes, emotions, …



	2008	2009	2010	2011	2012	2013	2014	2015
Unstructured	11,430	16,737	25,127	39,237	59,600	92,536	147,885	226,716
Database	1,952	2,782	4,065	6,179	9,140	13,824	21,532	32,188
Email	1,652	2,552	4,025	6,575	10,411	16,796	27,817	44,091

(Vasant Dhar, CACM, 2013)

Summary

- “Big data is a term for data sets that are so large or complex that traditional data processing application software is inadequate to deal with them” (wikipedia)

Big data analysis

Technological solutions

Technological solutions

- In this course, we will focus mainly on free solutions
- First attempt: Apache Lucene, 1999
- The most popular frameworks **today** are
  - Apache Hadoop (since 2006 – started as a Yahoo project)
  - Apache Spark (since 2012 – started at the AMPLab at UC Berkeley)

Hadoop *v.s.* Spark

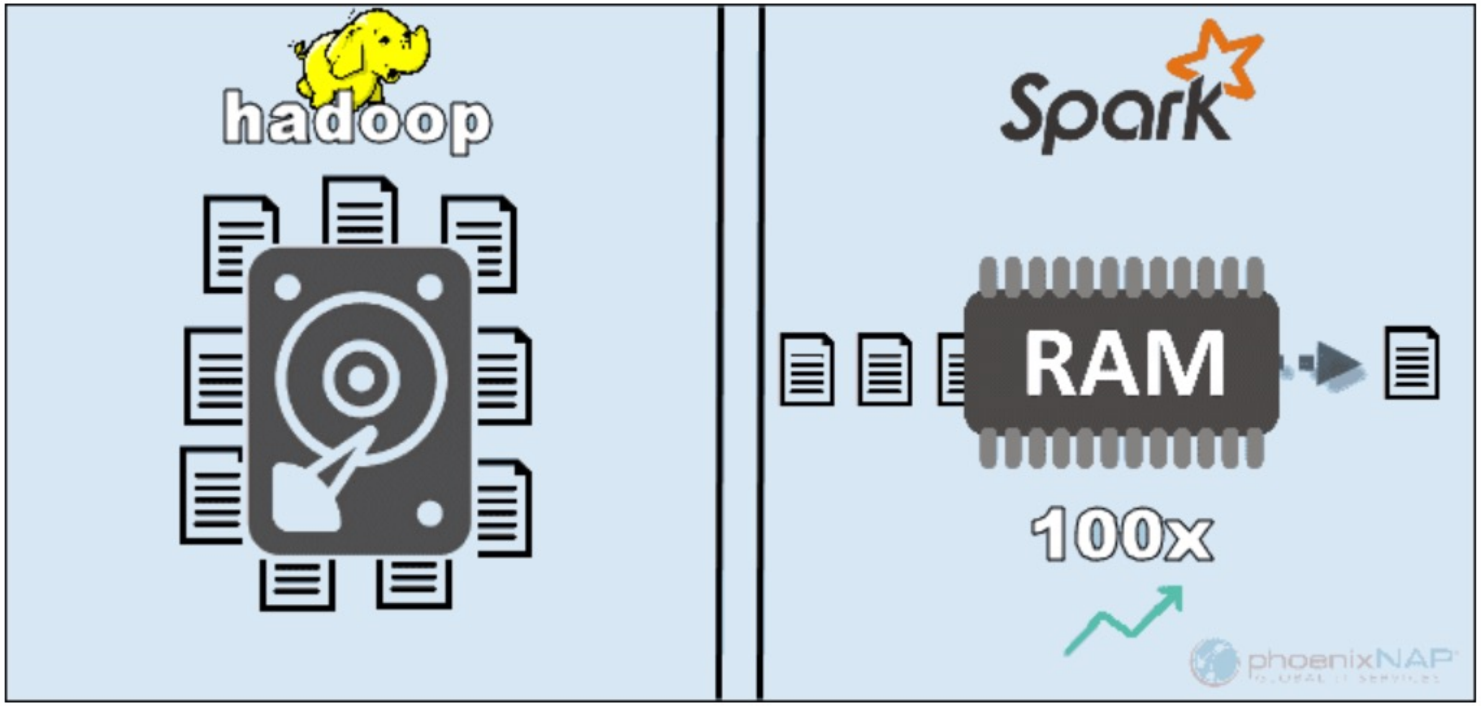
- Until a few years ago, Hadoop had a bigger market share than Spark
- Nowadays, Spark is catching up, and might even overtake Hadoop
- But, these two technologies should be seen as **complementary** instead of competitors
    - More and more often used together

Hadoop *v.s.* Spark

- **Hadoop**: distributed processing, core components:
  - Hadoop Distributed File System (**HDFS**)
    - **stores** files in a Hadoop-native format and **parallelizes** them across a cluster
  - **MapReduce**
    - algorithm that **processes** the data in parallel
  - Many more components…
- **Spark**: also for distributed processing, core components:
  - **Spark Core**: for scheduling, task dispatching, input and output operations, fault recovery, etc.
  - Works in-memory (in RAM) using **RDD**, Resilient Distributed Dataset

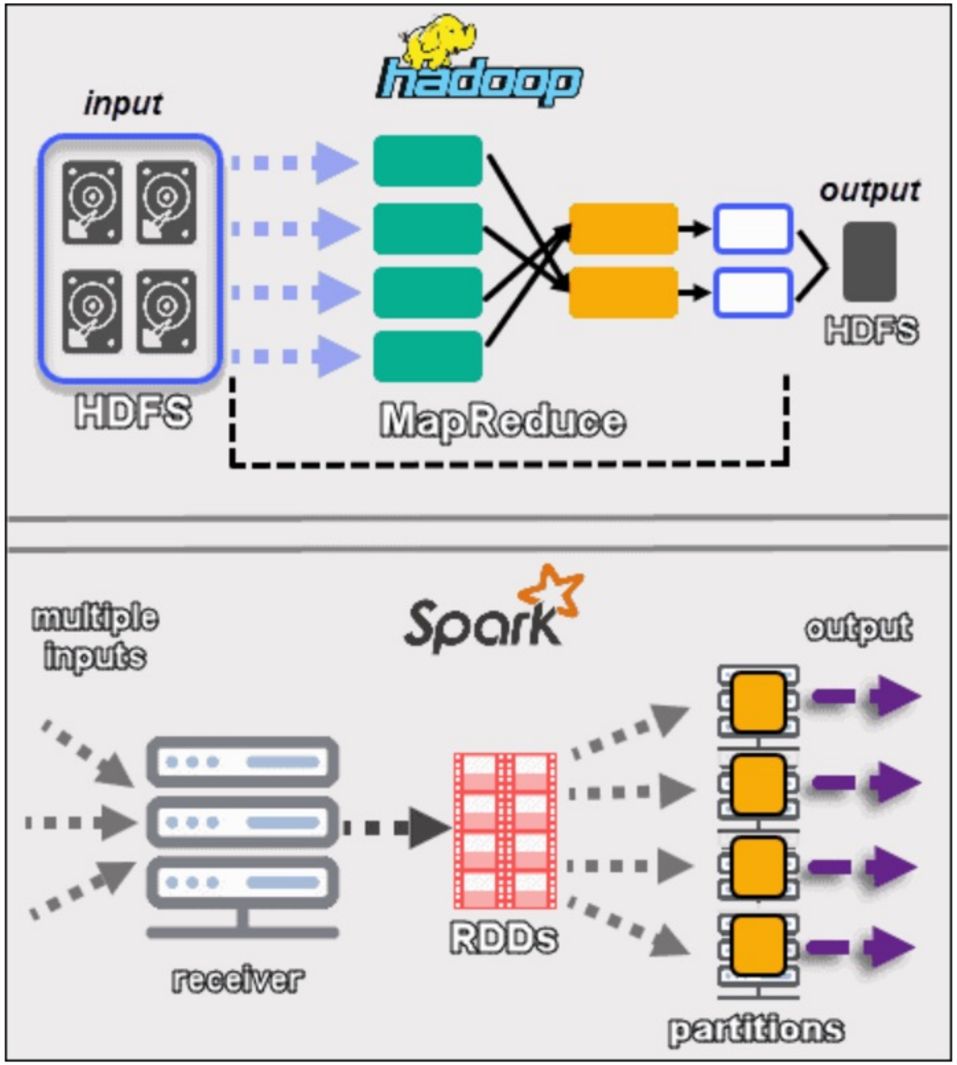
Hadoop *v.s.* Spark

- In (very) short:



[https://phoenixnap.com/kb/hadoop-vs-spark]

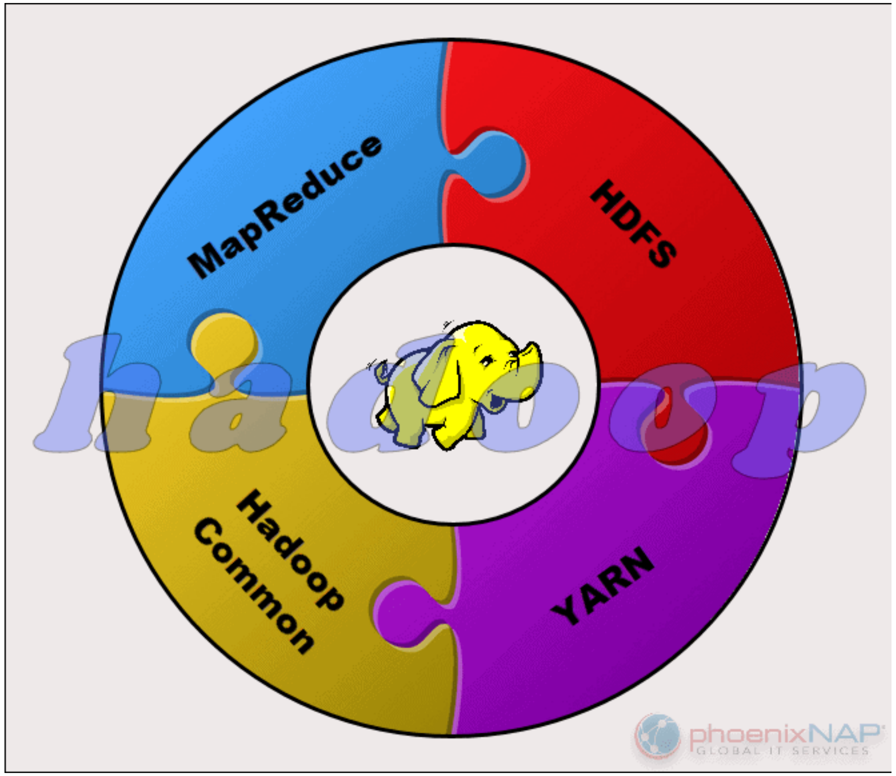
Hadoop *v.s.* Spark



[https://phoenixnap.com/kb/hadoop-vs-spark]

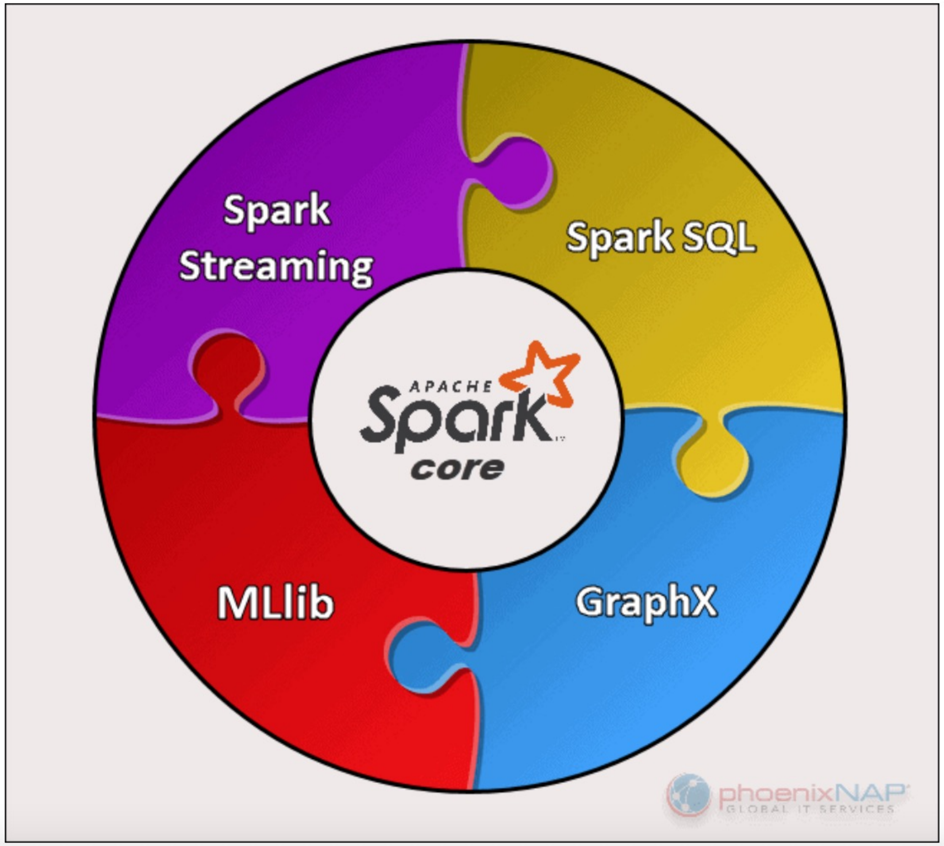
Focus on Hadoop – main modules

- **HDFS –** can store both structured and unstructured data
  - Compatible with most storage hardware (from consumer-grade HDDs to enterprise drives)
- **MapReduce –** processing component
  - Assigns data fragments from HDFS to separate map tasks in the cluster
  - MapReduce processes the chunks in parallel and combines the pieces into the desired result
- **YARN –** Yet Another Resource Negotiator
  - Manage computing resources and job scheduling
- **Hadoop Common (*a.k.a.* Hadoop Core)**
  - Set of common libraries and utilities that all other modules depend on
    - including **Mahoot** (for ML)



Focus on Spark – main modules

- **Spark Core**
  - Scheduling, task dispatching, I / O operations, fault recovery, etc
  - Other functionalities are built on top of it
- **Spark Streaming**
  - Processing live data streams
  - Data can originate from many different sources, including Kafka, Kinesis, Flume…
- **Spark SQL**
  - Gathers information about the structured data and how the data is processed.
- **Machine Learning Library (MLlib)**
  - Provides many ML **scalable** algorithms.
- **GraphX**
  - Set of APIs for graph analytics tasks.
    - *E.g.* social network analysis



Hadoop *v.s.* Spark

- Spark's processing speed is **most often** better than Hadoop:
  - No waste of time for input-output concerns
  - Enables optimizations between processing steps
- But, Spark may suffer from RAM overhead memory leaks
  - So, if the size of data is larger than the available RAM, Hadoop should be preferred
- Most often
  - Hadoop is prefereed to Spark for **batch processing**
  - Spark is prefereed to Hadoop for **streaming processing**
- When summing up all costs, Hadoop is cheaper

Hadoop *v.s.* Spark

Hadoop	*Category for Comparison*	Spark
Slower performance, uses disks for storage and depends on disk read and write speed.	Performance	Fast in-memory performance with reduced disk reading and writing operations.
An open-source platform, less expensive to run. Uses affordable consumer hardware. Easier to find trained Hadoop professionals.	Cost	An open-source platform, but relies on memory for computation, which considerably increases running costs.
Best for batch processing. Uses MapReduce to split a large dataset across a cluster for parallel analysis.	Data Processing	Suitable for iterative and live-stream data analysis. Works with RDDs and DAGs to run operations.
A highly fault-tolerant system. Replicates the data across the nodes and uses them in case of an issue.	Fault Tolerance	Tracks RDD block creation process, and then it can rebuild a dataset when a partition fails. Spark can also use a DAG to rebuild data across nodes.
Easily scalable by adding nodes and disks for storage. Supports tens of thousands of nodes without a known limit.	Scalability	A bit more challenging to scale because it relies on RAM for computations. Supports thousands of nodes in a cluster.

[https://phoenixnap.com/kb/hadoop-vs-spark]

Hadoop *v.s.* Spark

Hadoop	*Category for Comparison*	Spark
Extremely secure. Supports LDAP, ACLs, Kerberos, SLAs, etc.	Security	Not secure. By default, the security is turned off. Relies on integration with Hadoop to achieve the necessary security level.
More difficult to use with less supported languages. Uses Java or Python for MapReduce apps.	Ease of Use and Language Support	More user friendly. Allows interactive shell mode. APIs can be written in Java, Scala, R, Python, Spark SQL.
Slower than Spark. Data fragments can be too large and create bottlenecks. Mahout is the main library.	Machine Learning	Much faster with in-memory processing. Uses MLlib for computations.
Uses external solutions. YARN is the most common option for resource management. Oozie is available for workflow scheduling.	Scheduling and Resource Management	Has built-in tools for resource allocation, scheduling, and monitoring.

[https://phoenixnap.com/kb/hadoop-vs-spark]

Why are Hadoop and Spark complementary?

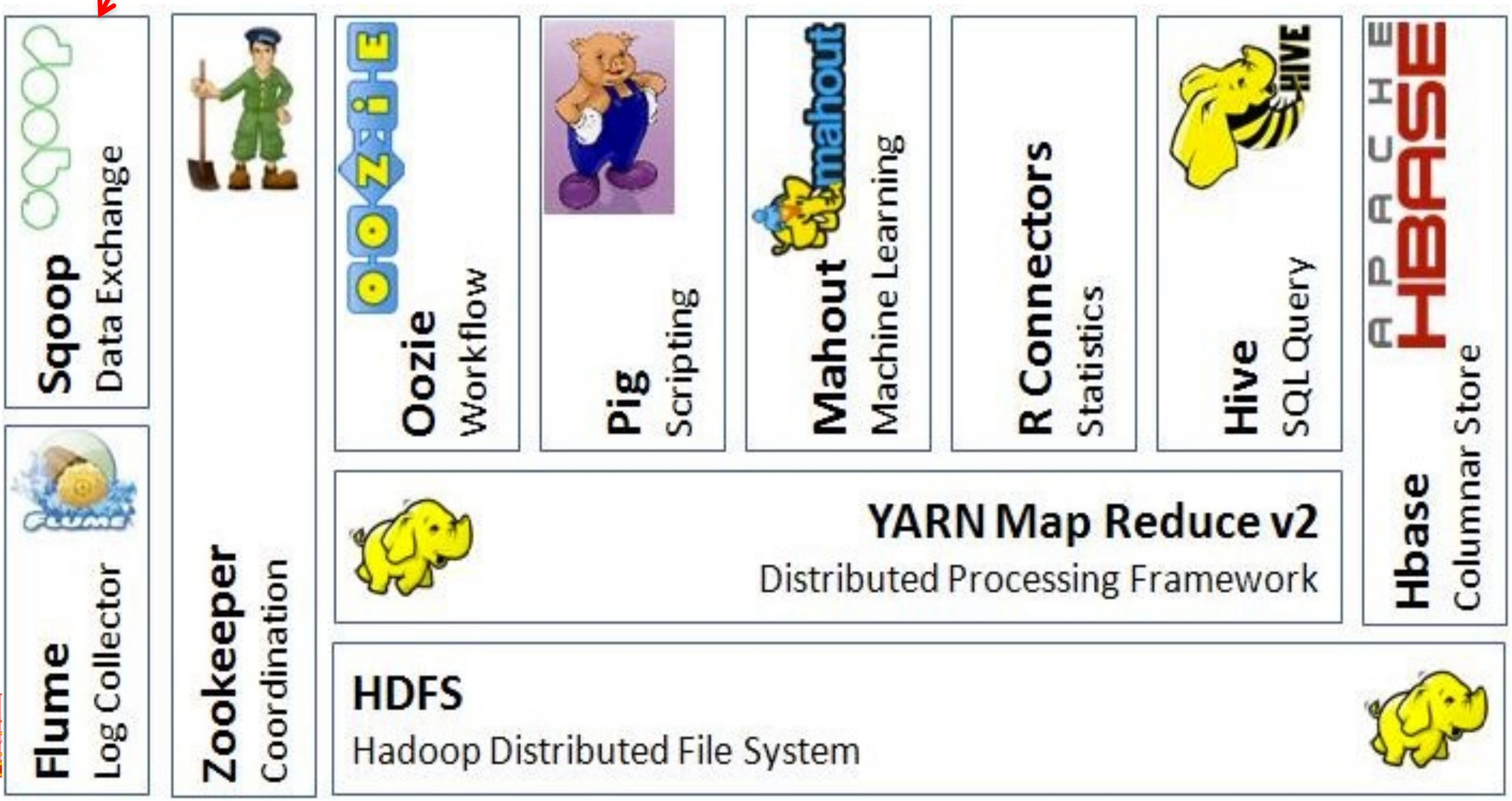
- Spark can run in stand-alone mode, with a Hadoop cluster as a data source
  - *(or on a cloud or cluster manager such as Apache Mesos, and other platforms)*
- Spark usually relies on Hadoop for ensuring its security
- Many enterprises switch from Hadoop to Spark (and *vice-versa)* depending on the task
  - Hadoop for batch analysis
  - Spark (sometimes over Hadoop HDFS) for stream analysis

Big data analysis

Hadoop

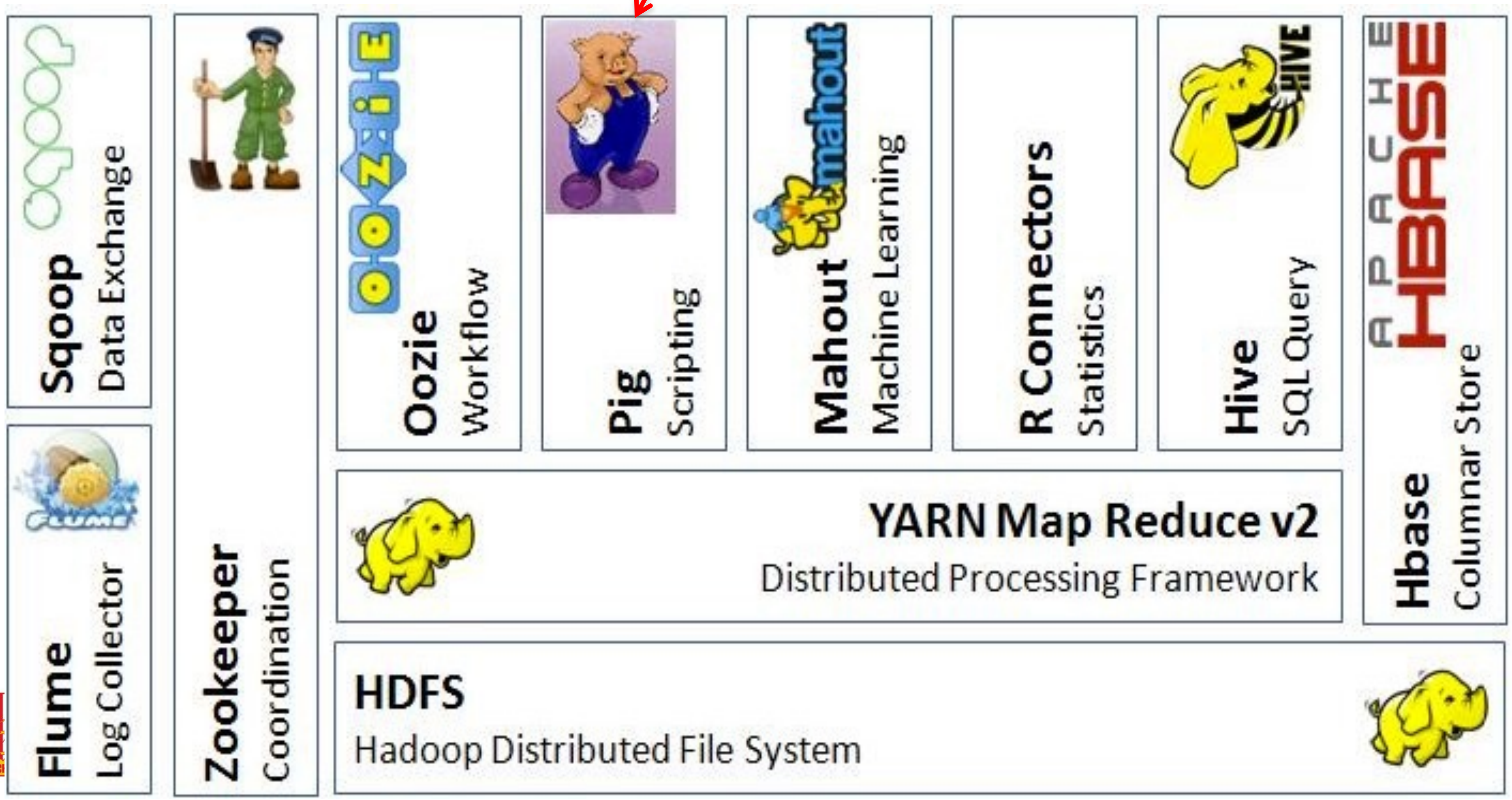
Overview of Hadoop ecosystem

Sqoop: for efficiently transferring bulk data between Apache Hadoop and structured datastores (*e.g.* relational databases) – both ways



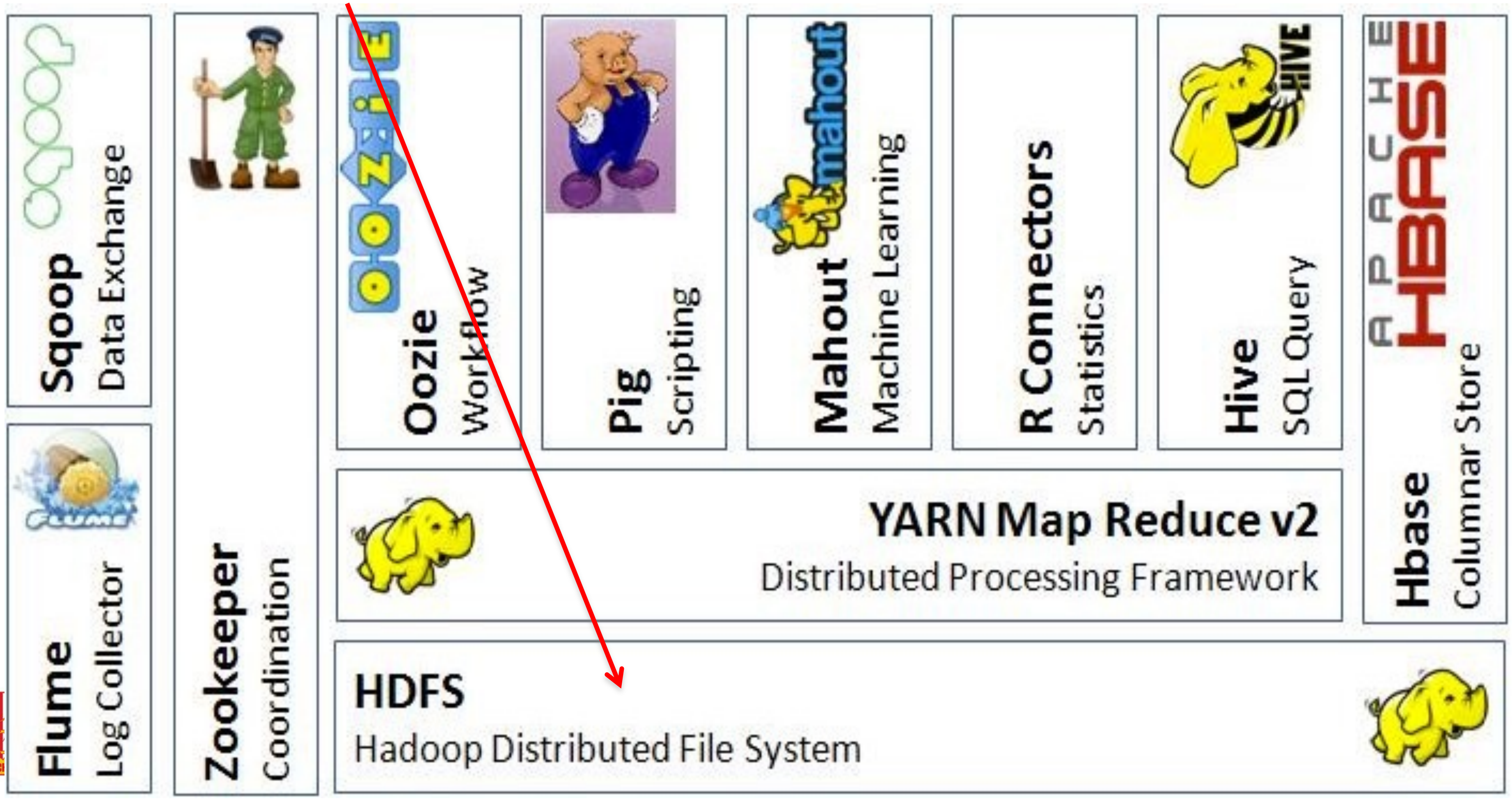
Overview of Hadoop ecosystem

Pig: for data integration / processing; especially good at joining and transforming data



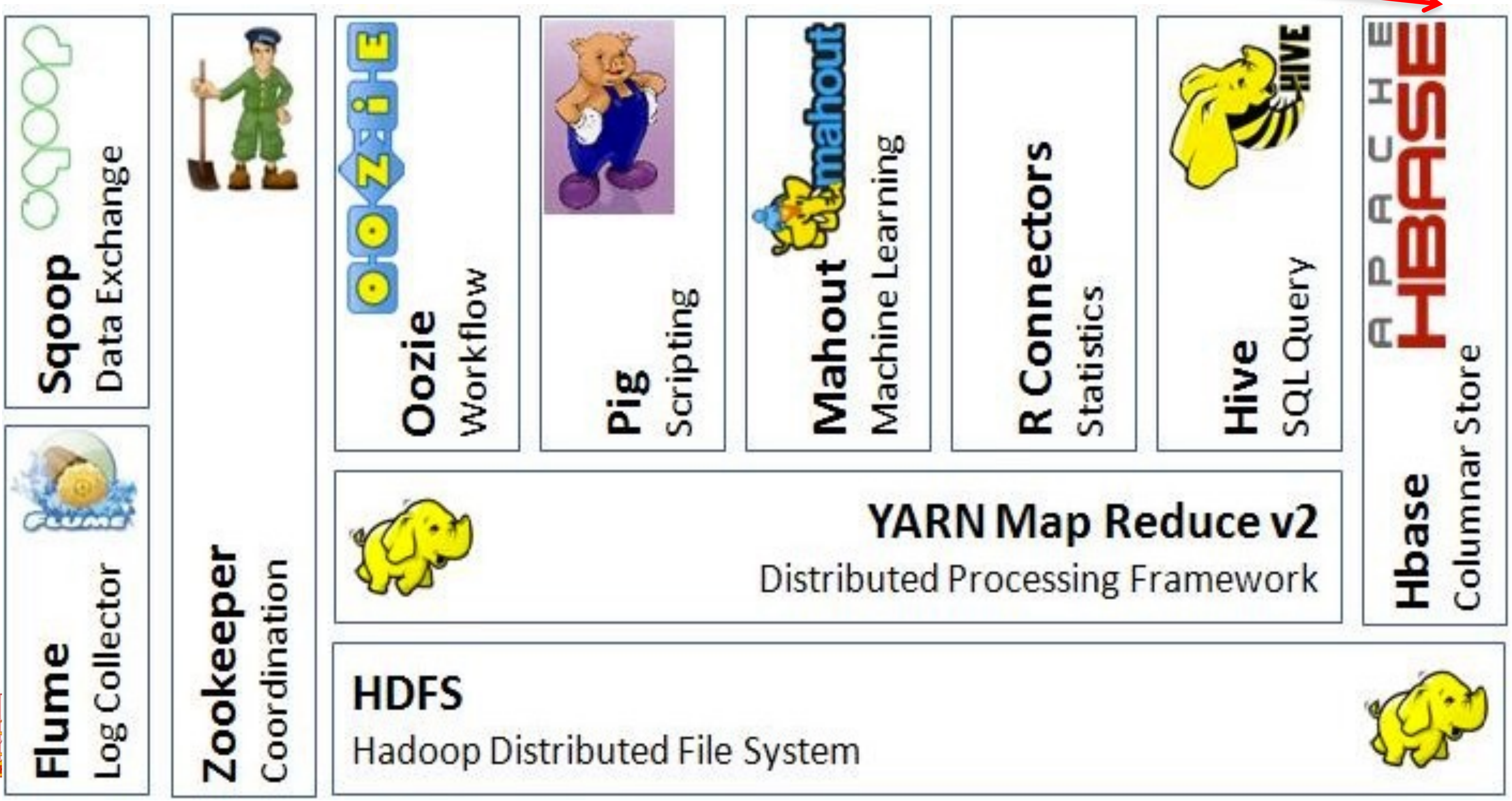
Overview of Hadoop ecosystem

HDFS: distributed file storage. All files passed into HDFS are split into blocks. Each block is replicated a certain number of times across the cluster, ensuring fault tolerance



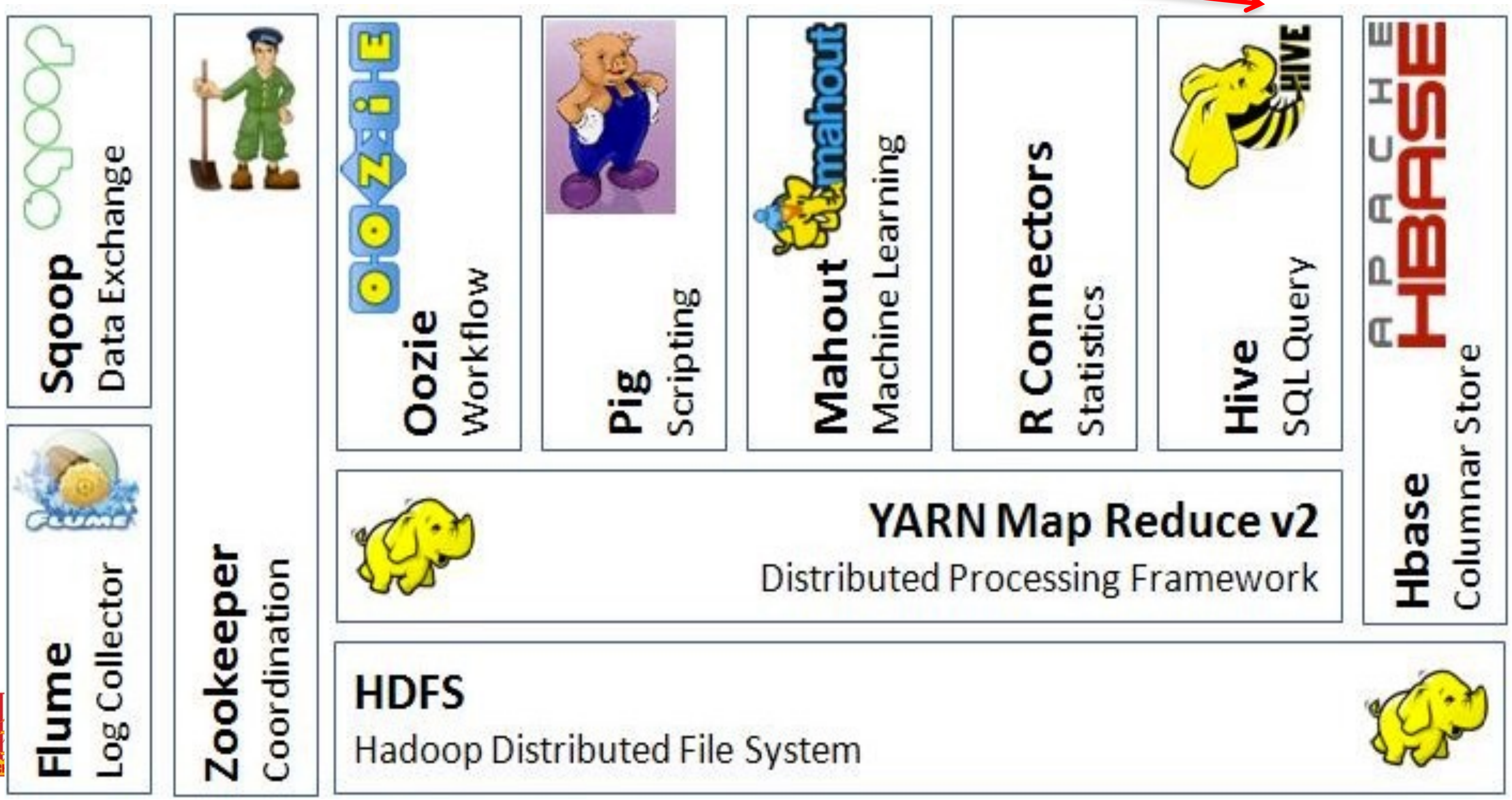
Overview of Hadoop ecosystem

HBase: distributed column-oriented data store built on top of HDFS (considered as the Hadoop database, organized logically into data tables, but queried using NoSQL)



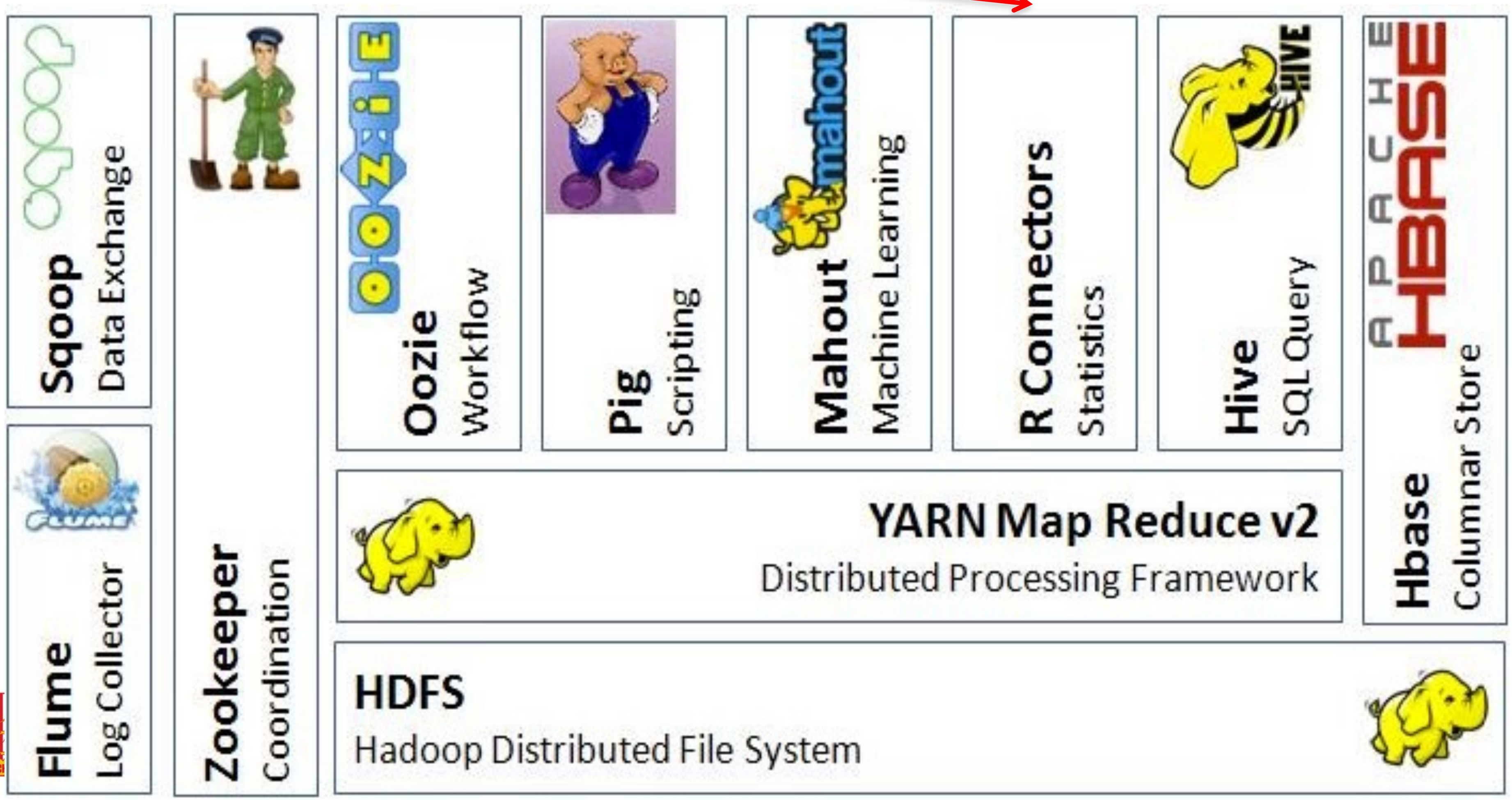
Overview of Hadoop ecosystem

Hive interpreter runs on the client machine; Uses HiveQL: SQL-like language.
HiveQL scripts are turned into MapReduce jobs that are submitted to the cluster



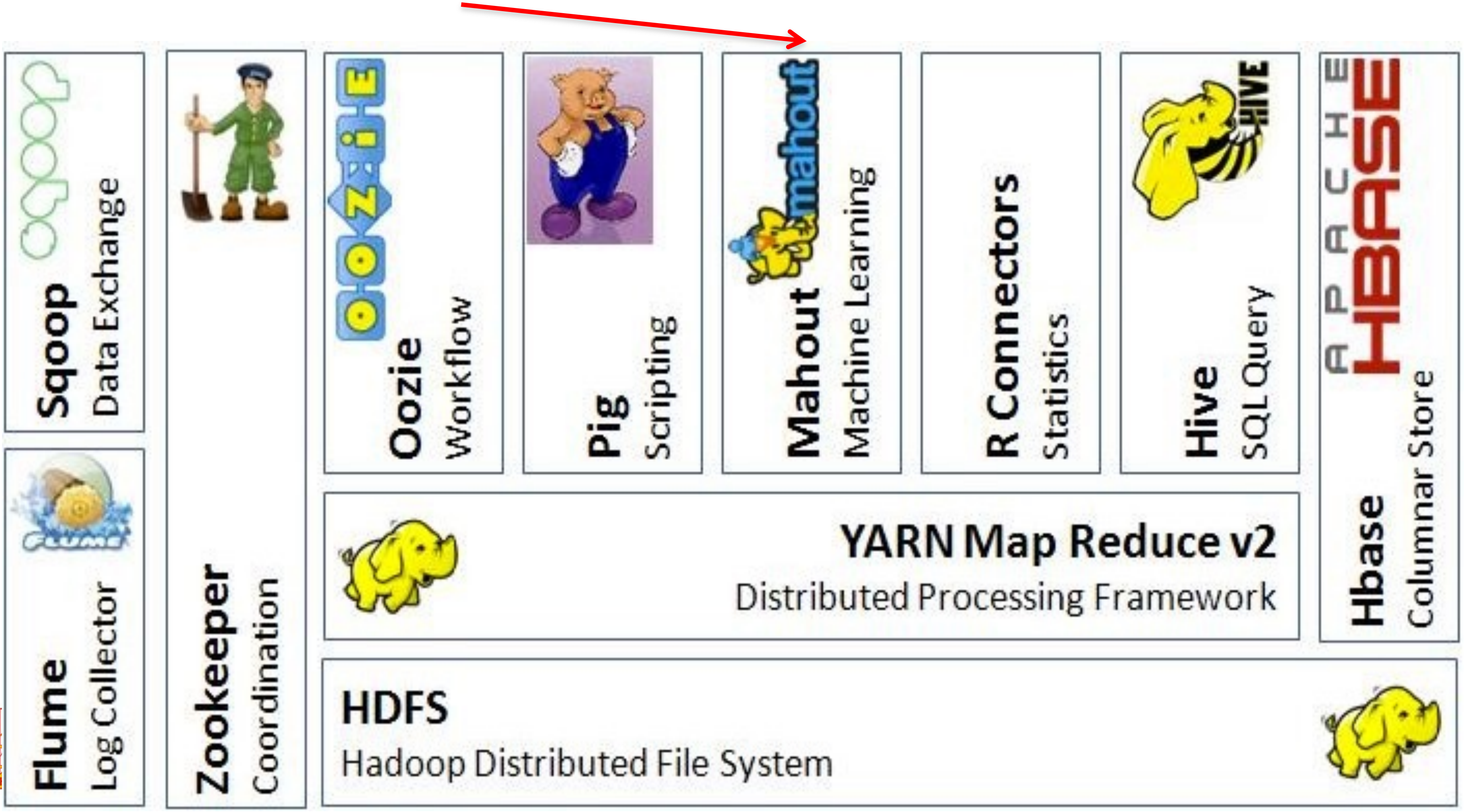
Overview of Hadoop ecosystem

R connectors: using R language with Hadoop to apply statistical computations *(e.g. Exploratory Data Analysis)* on data sets stored in HDFS



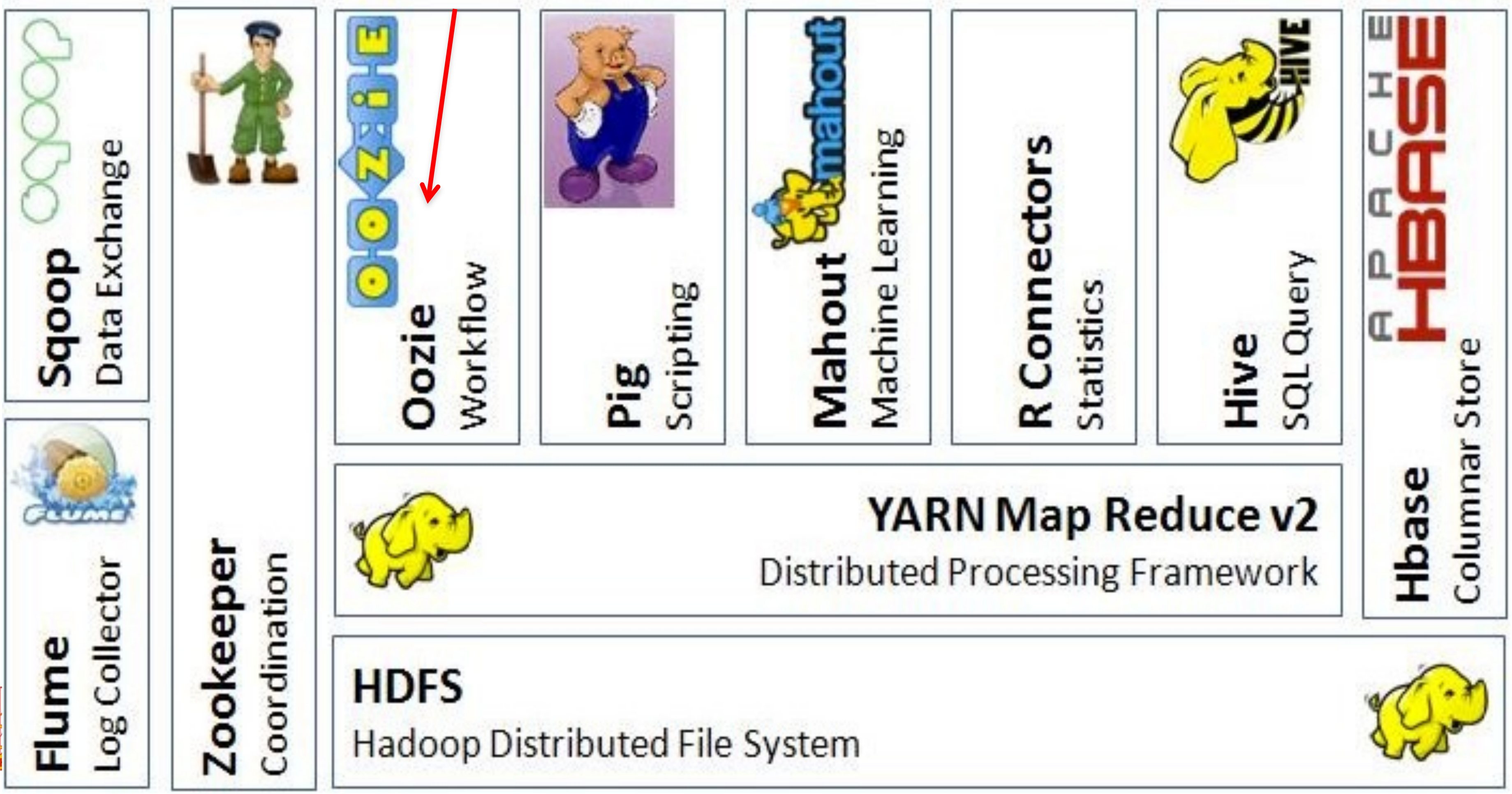
Overview of Hadoop ecosystem

Mahout: relies on MapReduce to perform mostly clustering, classification, and association



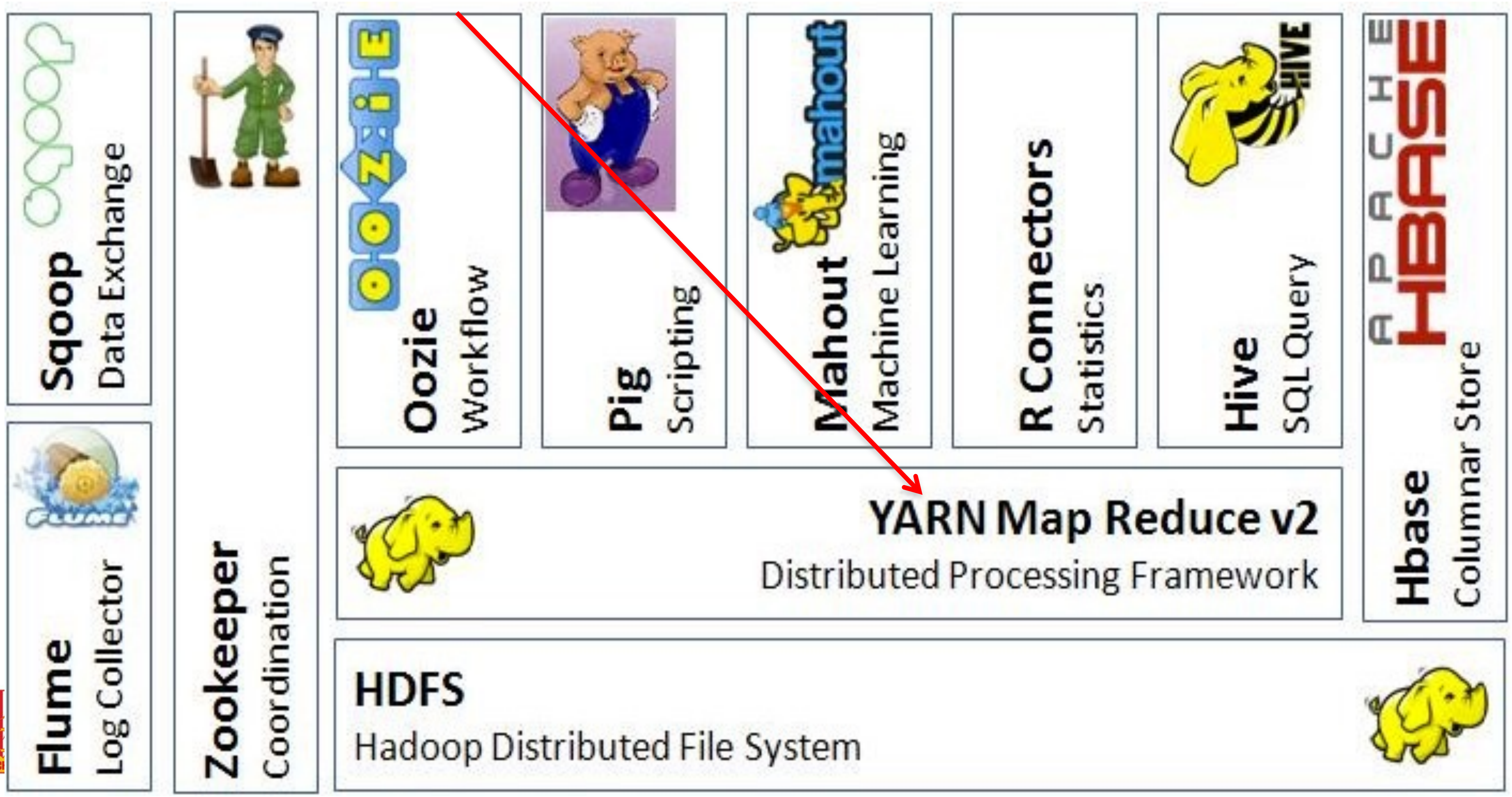
Overview of Hadoop ecosystem

Oozie manages Apache Hadoop jobs using Directed Acyclical Graphs (DAGs) of actions, including executing MapReduce jobs, running Pig or Hive scripts, executing standard Java or shell programs, sending e-mails, etc.



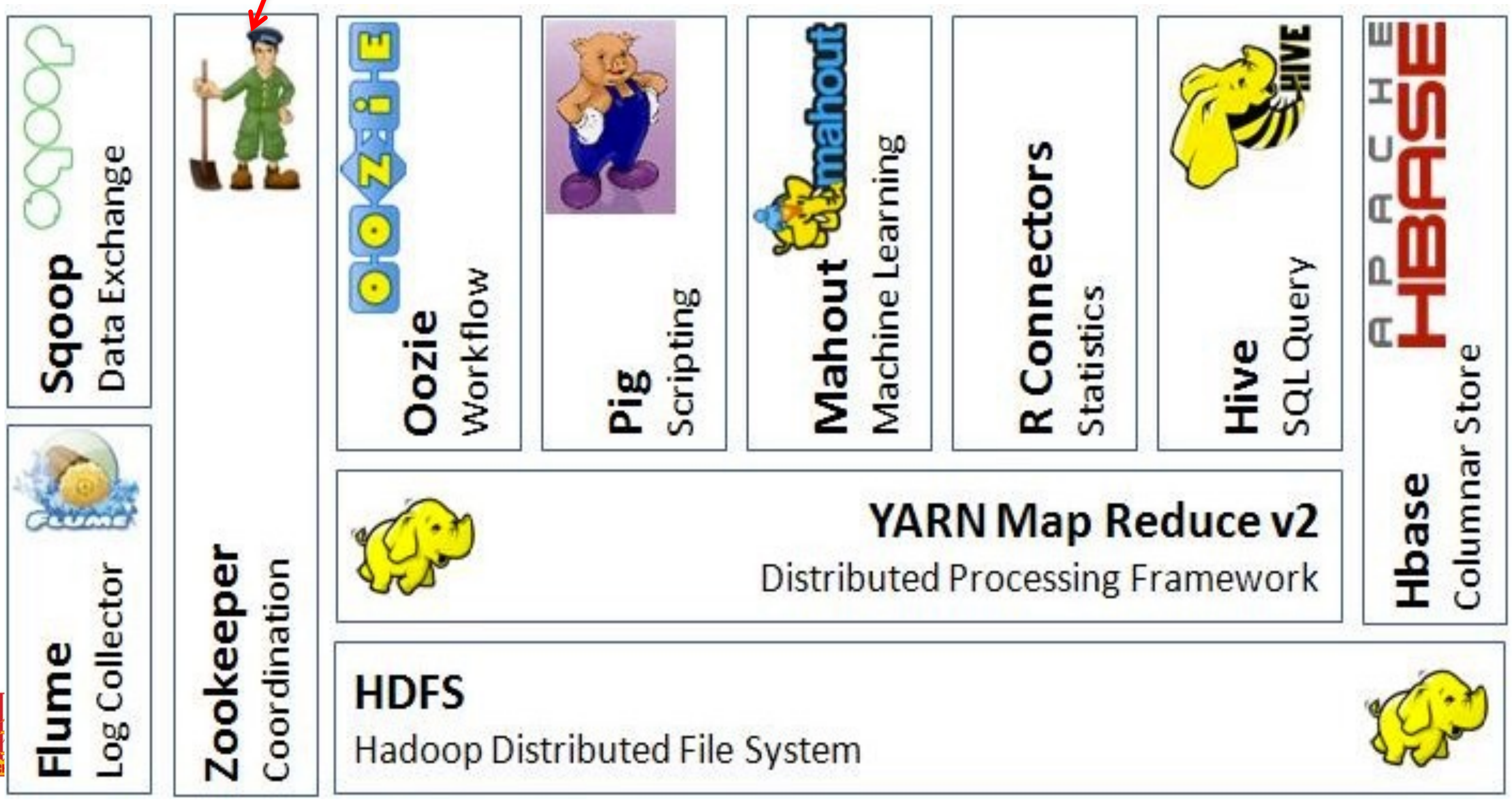
Overview of Hadoop ecosystem

YARN manages the allocation of relevant and necessary resources (memory and CPU cores) to Map Reduce and non-Map Reduce data processing task



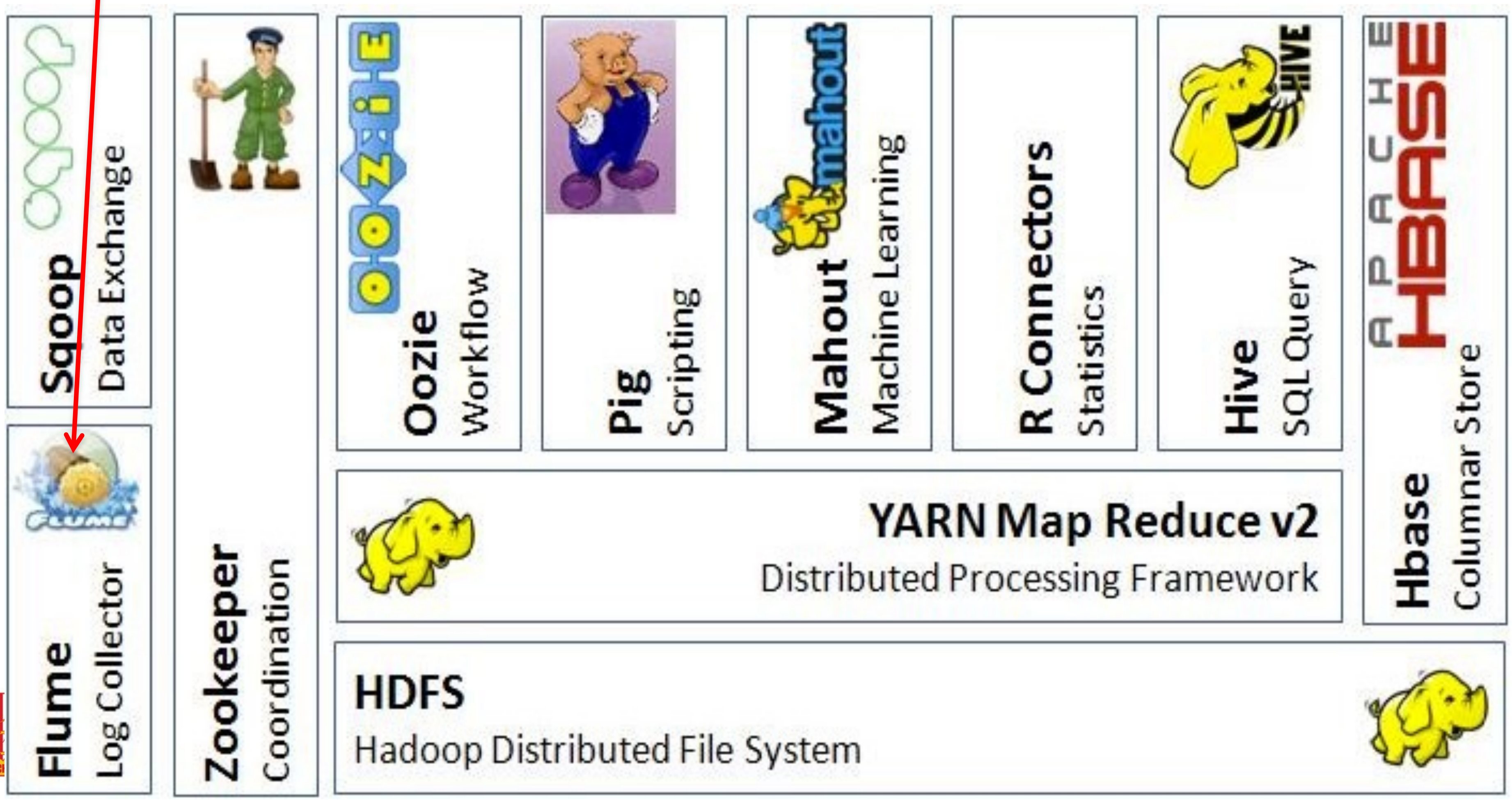
Overview of Hadoop ecosystem

Zookeeper: high-performance coordination service for distributed applications (configuration, coordination, ...)



Overview of Hadoop ecosystem

Apache Flume: distributed service for collecting, aggregating, and moving large amounts of log data (log collector)

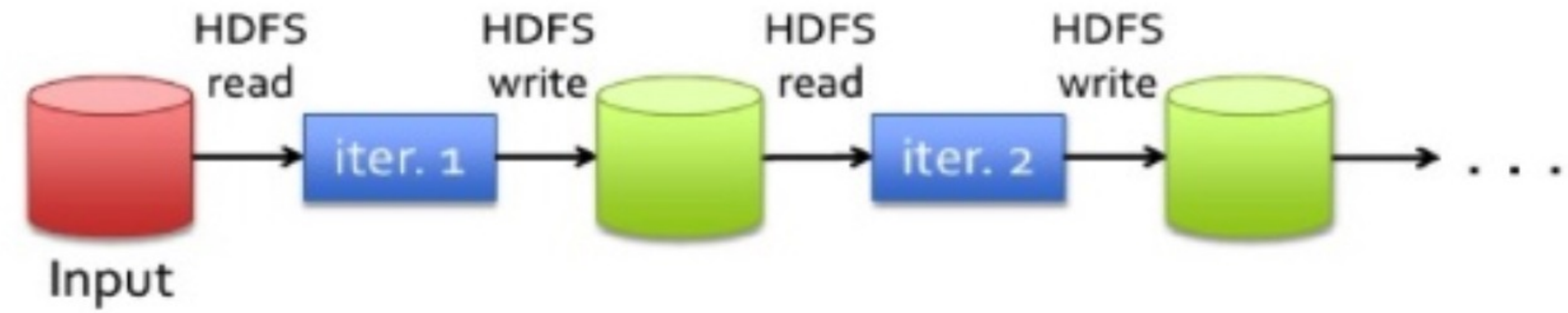


For more information on Hadoop

- Study the course from Prof. Viet-Trung Tran
  - L9-1 - Hadoop Course.pdf on Microsoft Teams

Limitations of Hadoop

- MapReduce Iterative jobs involve a lot of disk I/O for each repetition

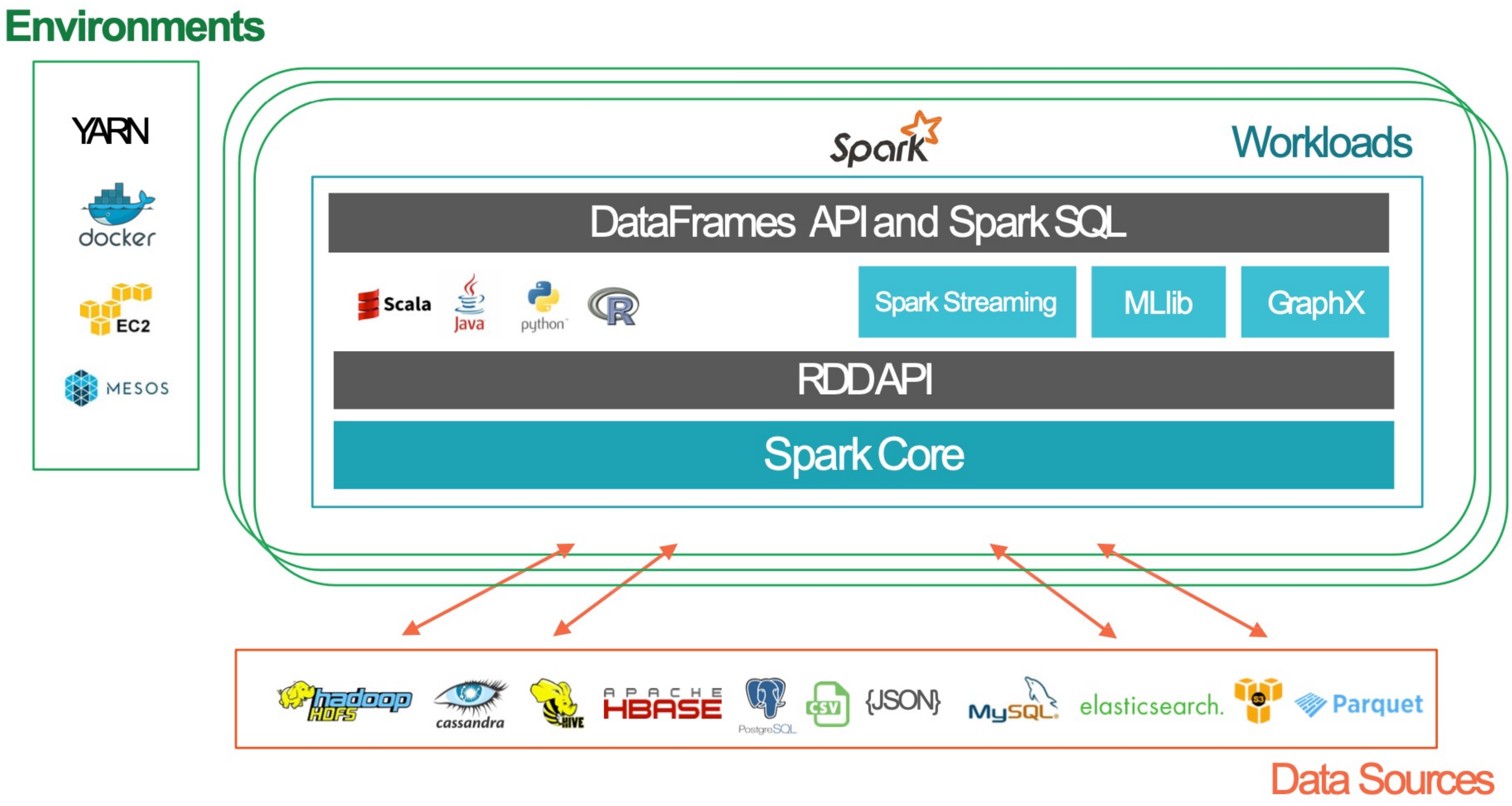


- This makes Hadoop slow, especially when dealing with data streams
  - ➢Spark

Big data analysis

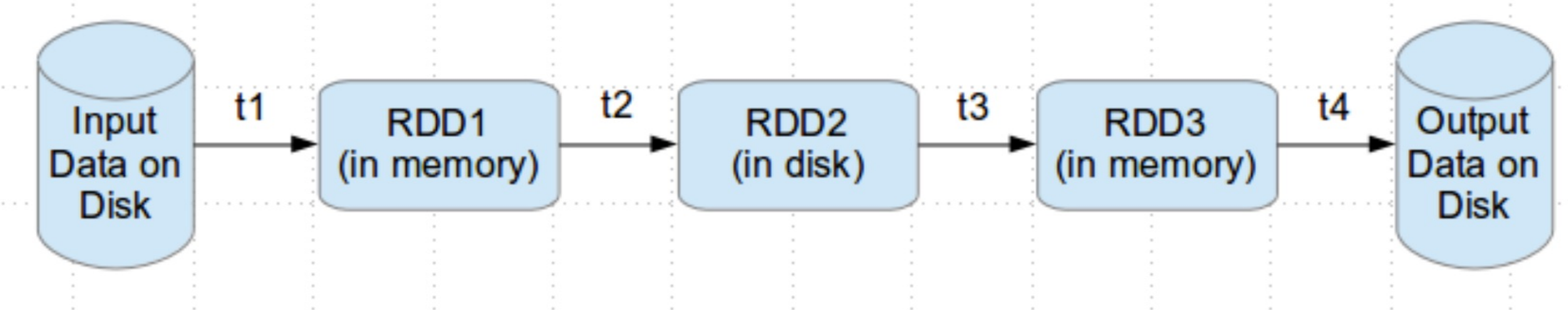
Spark

Overview of the Spark ecosystem



Resilient Distributed Dataset (RDD)

- RDDs are ***parallel data structures*** that let users
  - explicitly persist ***intermediate results in memory***
  - control their partitioning to optimize data placement
  - manipulate the data using ***a rich set of operators***
- RDDs are ***fault-tolerant,*** as they can be automatically rebuilt upon machine failure



Limitations of Spark

- More costly than Hadoop
  - RAM costs more than storage
  - Spark experts are fewer, so you'll need to pay them more
- Spark may suffer from RAM overhead memory leaks
  - So, if the size of data is larger than the available RAM, Hadoop should be preferred
- Most often
  - Hadoop is prefereed to Spark for batch processing
  - Spark is prefereed to Hadoop for streaming processing

For more information on Spark

- Study the course from Prof. Viet-Trung Tran
  - L9-2 - Spark Overview.pdf on Microsoft Teams

Questions

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attention!!!

soict.hust.edu.vn/ fb.com/groups/soict