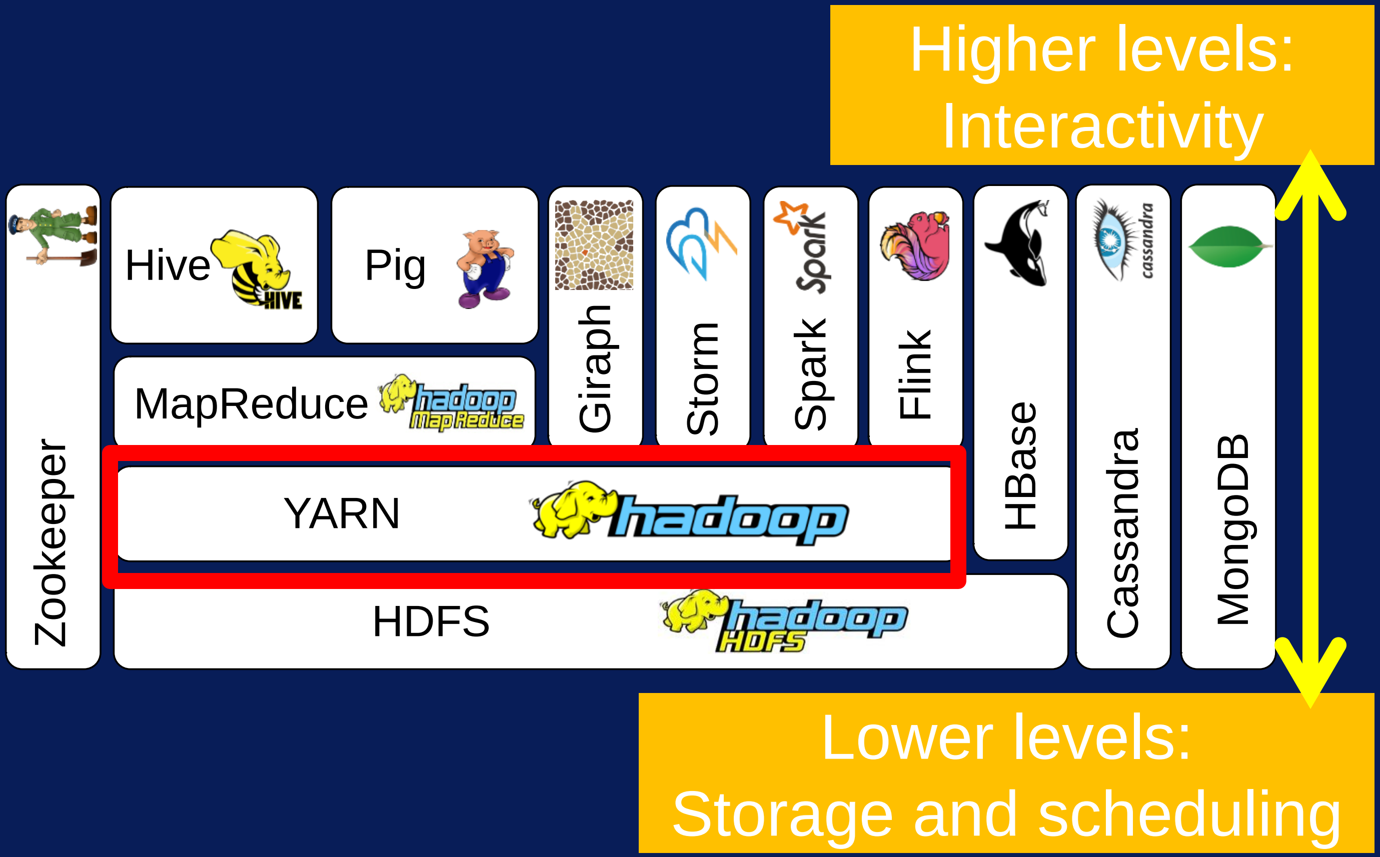
# Overview of Big Data Processing Systems

# After this video you will be able to..

- Recall the Hadoop Ecosystem
- Draw a layer diagram with three layers for data storage, data processing and workflow management
- Summarize an evaluation criteria for big data processing systems
- Explain the properties of Hadoop, Spark, Flink, Beam and Storm

# One possible layer diagram for Hadoop tools



Higher levels: Interactivity

Zookeeper

Hive

Pig

MapReduce

Giraph

Storm

Spark

Flink

YARN

HBase

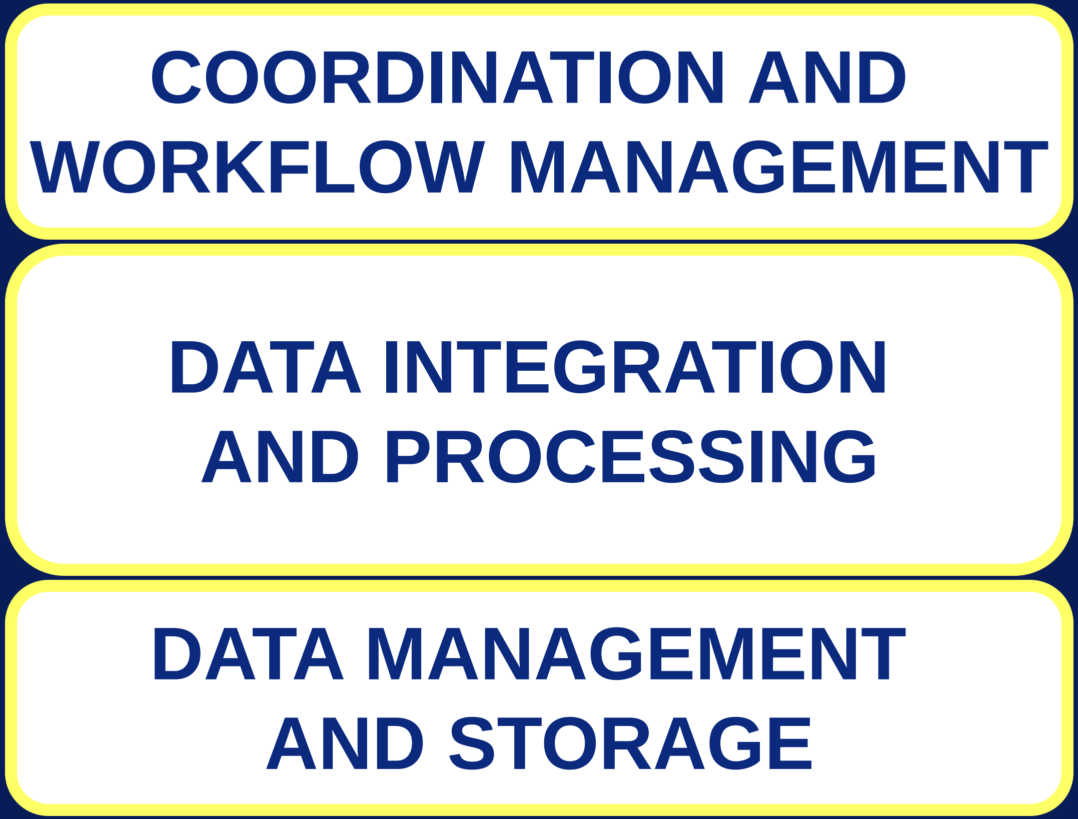
Cassandra

MongoDB

HDFS

Lower levels: Storage and scheduling

# Another way to look at the Hadoop Ecosystem

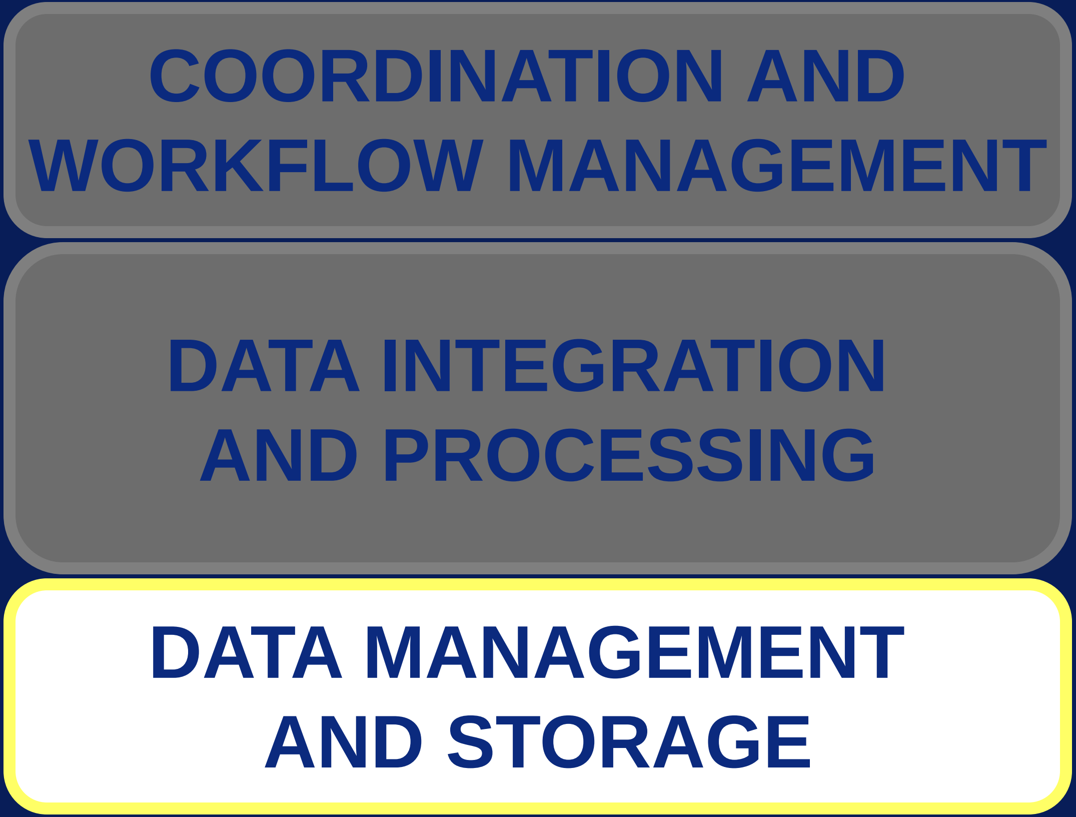


COORDINATION AND WORKFLOW MANAGEMENT

DATA INTEGRATION AND PROCESSING

DATA MANAGEMENT AND STORAGE

# Another way to look at the Hadoop Ecosystem



COORDINATION AND WORKFLOW MANAGEMENT

DATA INTEGRATION AND PROCESSING

DATA MANAGEMENT AND STORAGE

# DATA MANAGEMENT AND STORAGE

hadoop HDFS

Lucene

redis

VERTICA

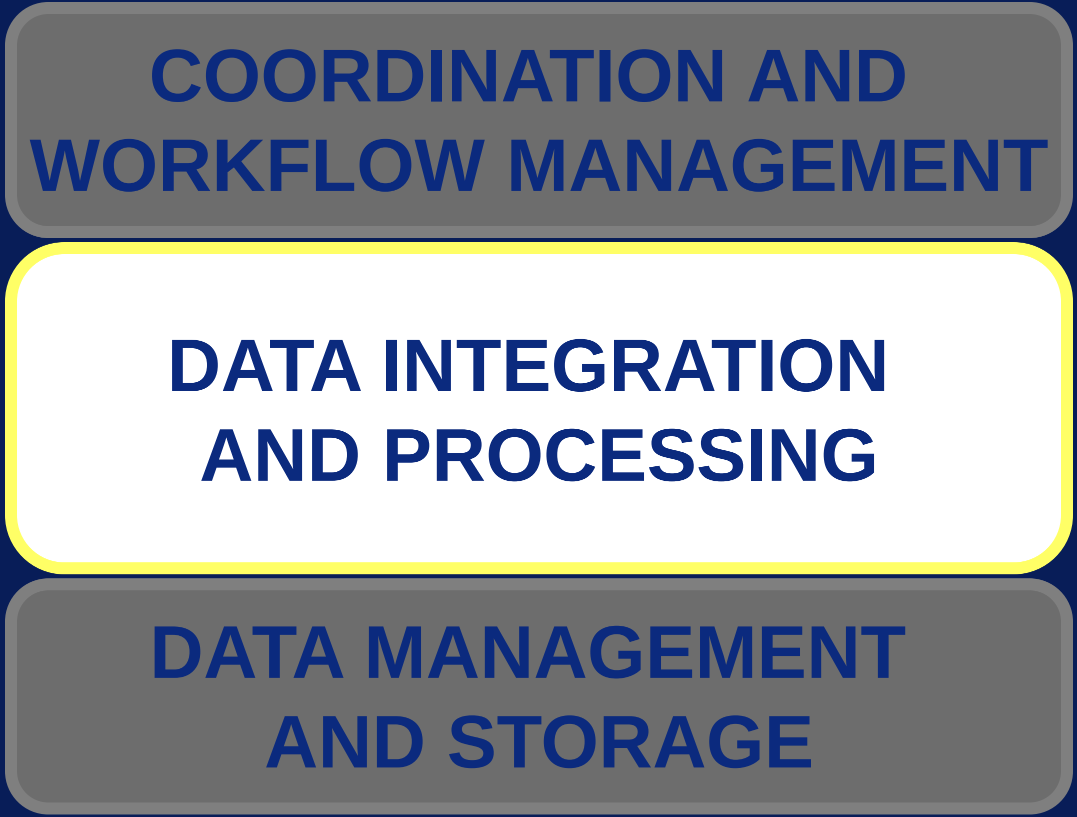
AEROSPIKE

Solr

Gephi

mongoDB

# Another way to look at the Hadoop Ecosystem

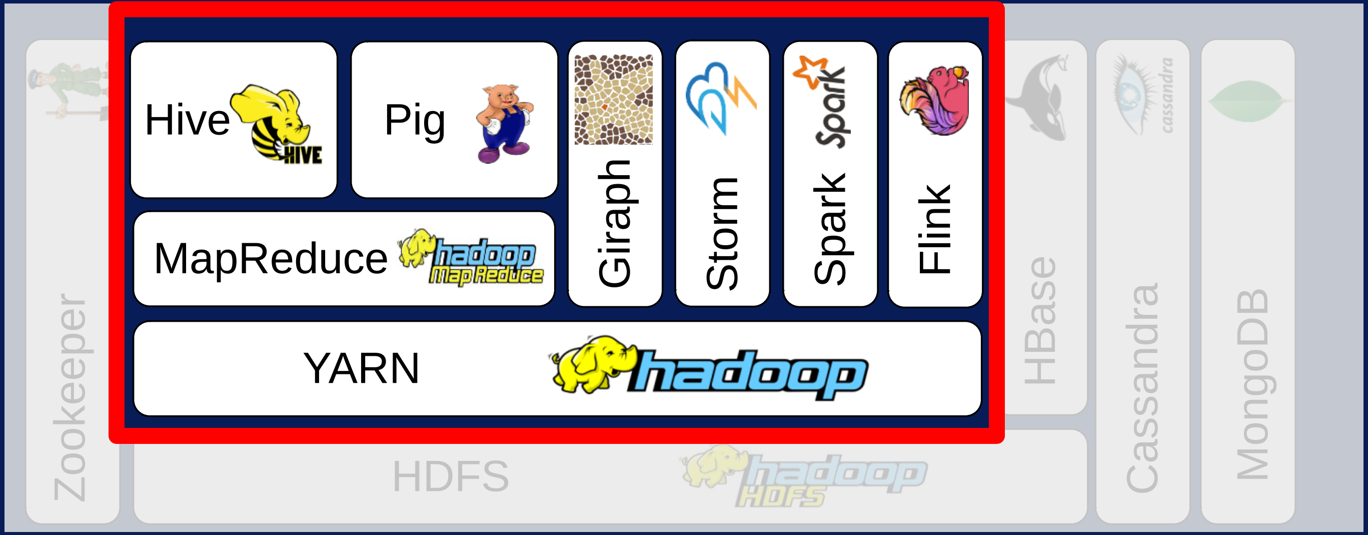


COORDINATION AND WORKFLOW MANAGEMENT

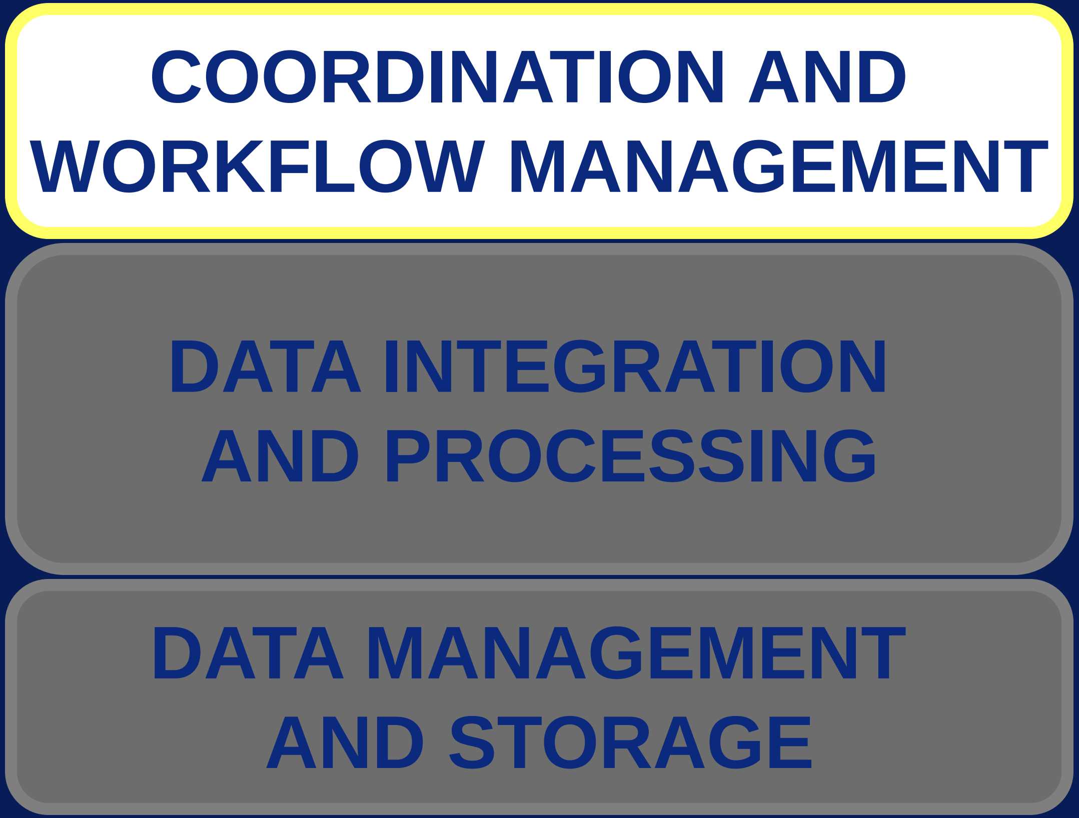
DATA INTEGRATION AND PROCESSING

DATA MANAGEMENT AND STORAGE

# DATA INTEGRATION AND PROCESSING



# Another way to look at the Hadoop Ecosystem

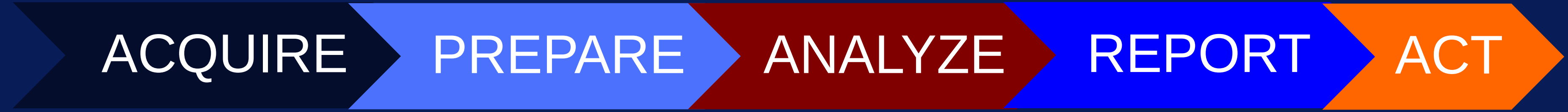


COORDINATION AND WORKFLOW MANAGEMENT

DATA INTEGRATION AND PROCESSING

DATA MANAGEMENT AND STORAGE

# COORDINATION AND WORKFLOW MANAGEMENT



ACQUIRE

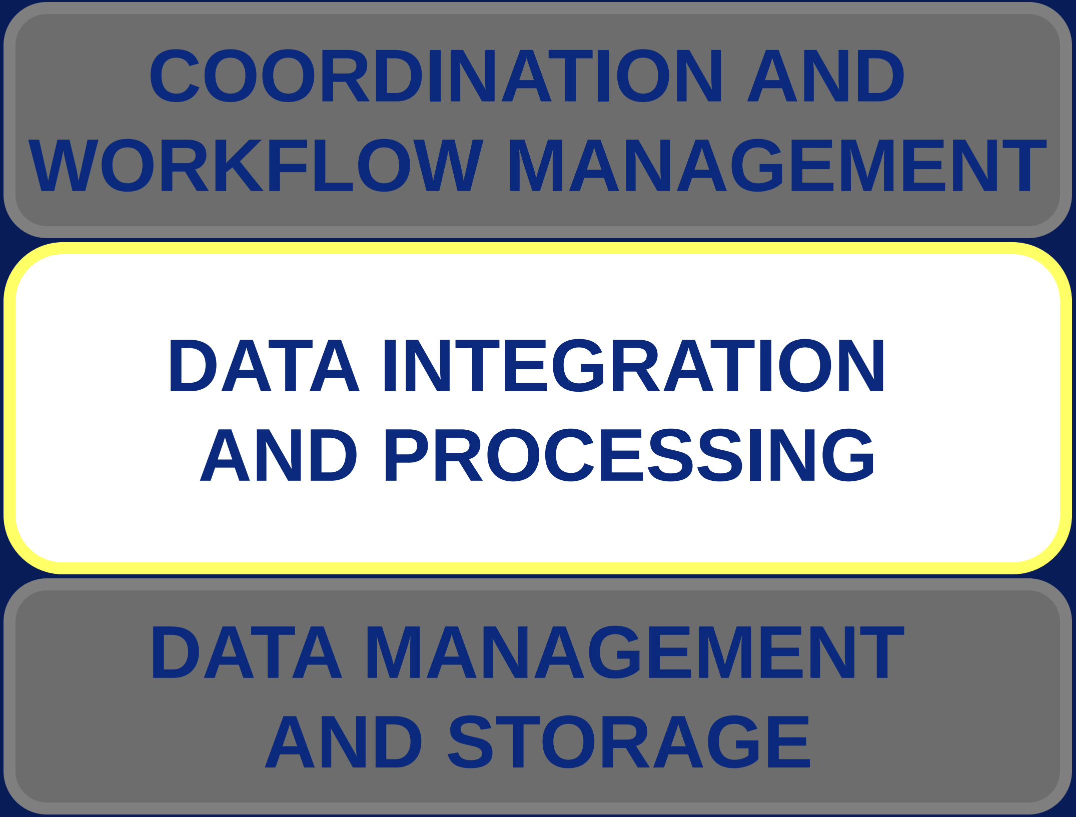
PREPARE

ANALYZE

REPORT

ACT

# Another way to look at the Hadoop Ecosystem

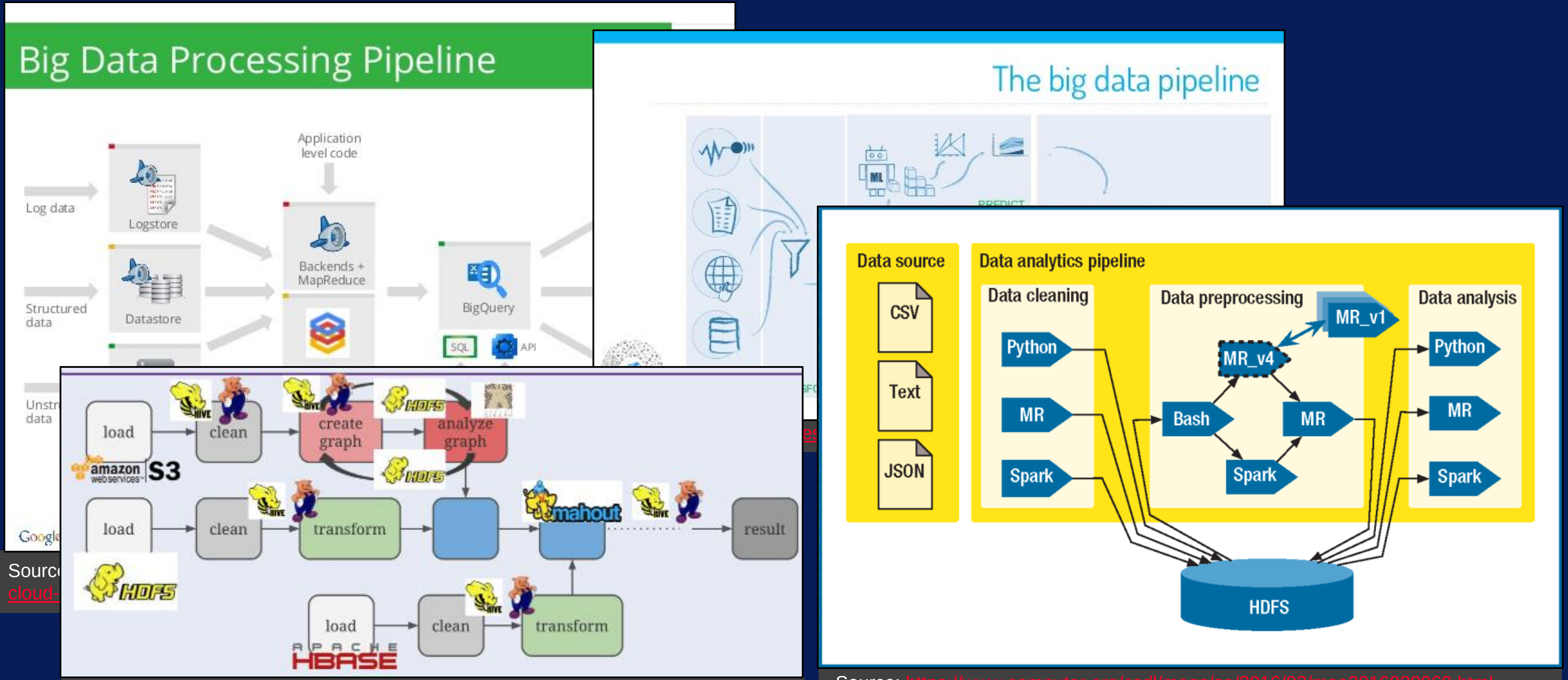


COORDINATION AND WORKFLOW MANAGEMENT

DATA INTEGRATION AND PROCESSING

DATA MANAGEMENT AND STORAGE

# Example Big Data Processing Pipelines



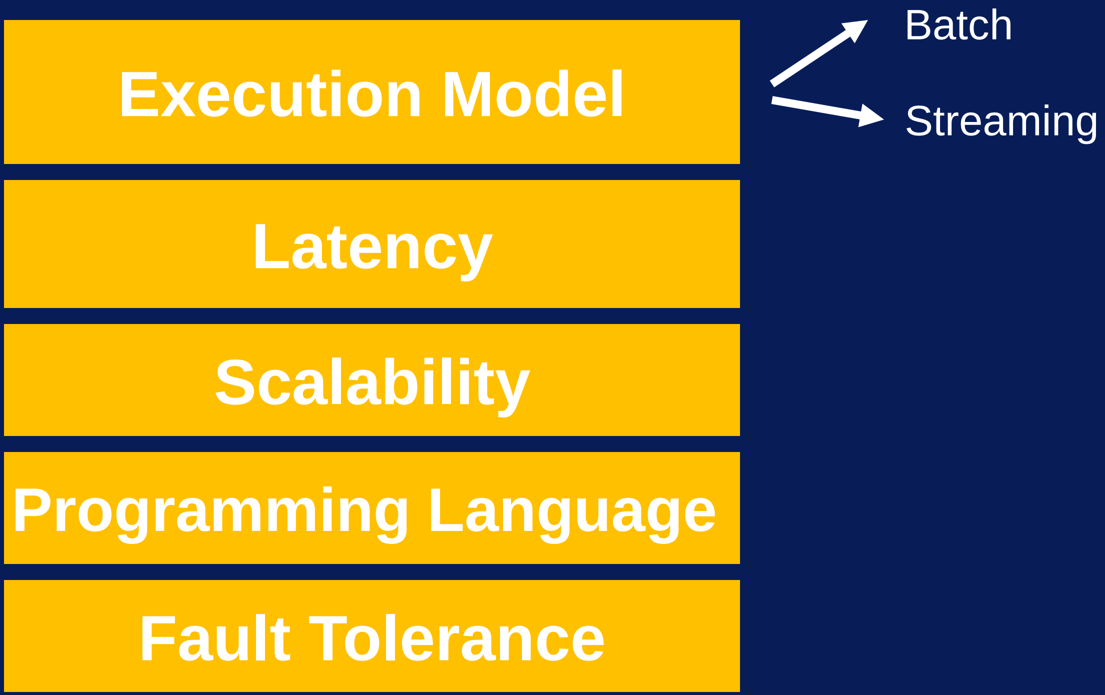
Sourc

cloud-

Source: https://www.mapr.com/blog/distributed-stream-and-graph-processing-apache-flink

Source: https://www.computer.org/csdl/mags/so/2016/02/mso2016020060.html

# Categorization of Big Data Processing Systems



Execution Model

Batch

Streaming

Latency

Scalability

Programming Language

Fault Tolerance

# Big Data Processing Systems

POWERED BY APACHE™

APACHE STORM™

Distributed · Resilient · Real-time

Flink

hadoop Map Reduce

Spark

beam

# MapReduce

| | |
|---|---|
| **Execution Model** | Batch processing using disk storage |
| **Latency** | High-latency |
| **Scalability** | |
| **Programming Language** | Java |
| **Fault Tolerance** | Replication |

# Spark

| | |
|---|---|
| **Execution Model** | Batch and stream processing using disk or memory storage |
| **Latency** | Low-latency for small micro-batch size |
| **Scalability** | |
| **Programming Language** | Scala, Python, Java, R |
| **Fault Tolerance** | |

# Flink

| | |
|---|---|
| Execution Model | Batch and stream processing using disk or memory storage |
| Latency | Low-latency |
| Scalability | |
| Programming Language | Java and Scala |
| Fault Tolerance | |

# Beam

| | |
|---|---|
| Execution Model | Batch and stream processing |
| Latency | Low-latency |
| Scalability | |
| Programming Language | Java and Scala |
| Fault Tolerance | |

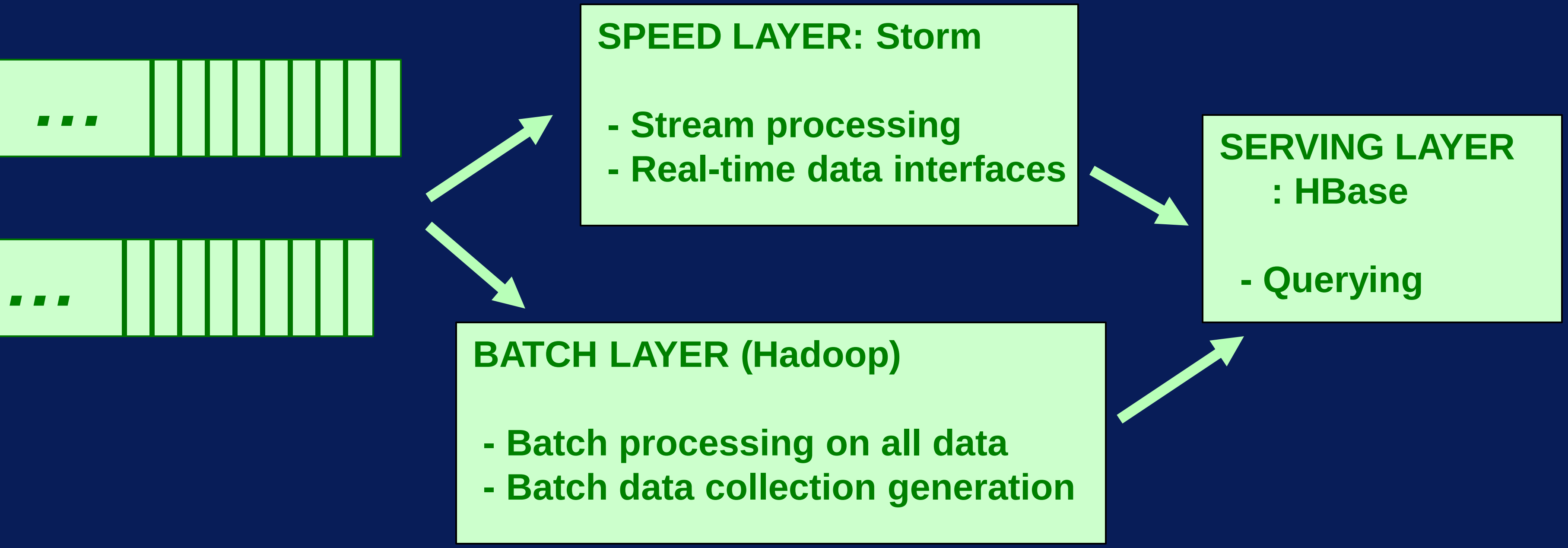
# Storm

APACHE
STORM™
Distributed · Resilient · Real-time

| | |
|---|---|
| **Execution Model** | Stream processing |
| **Latency** | Very low-latency |
| **Scalability** | |
| **Programming Language** | Many programming languages |
| **Fault Tolerance** | |

# Lambda Architecture:

## A Hybrid Data Processing Architecture



**SPEED LAYER: Storm**

- **Stream processing**
- **Real-time data interfaces**

**BATCH LAYER (Hadoop)**

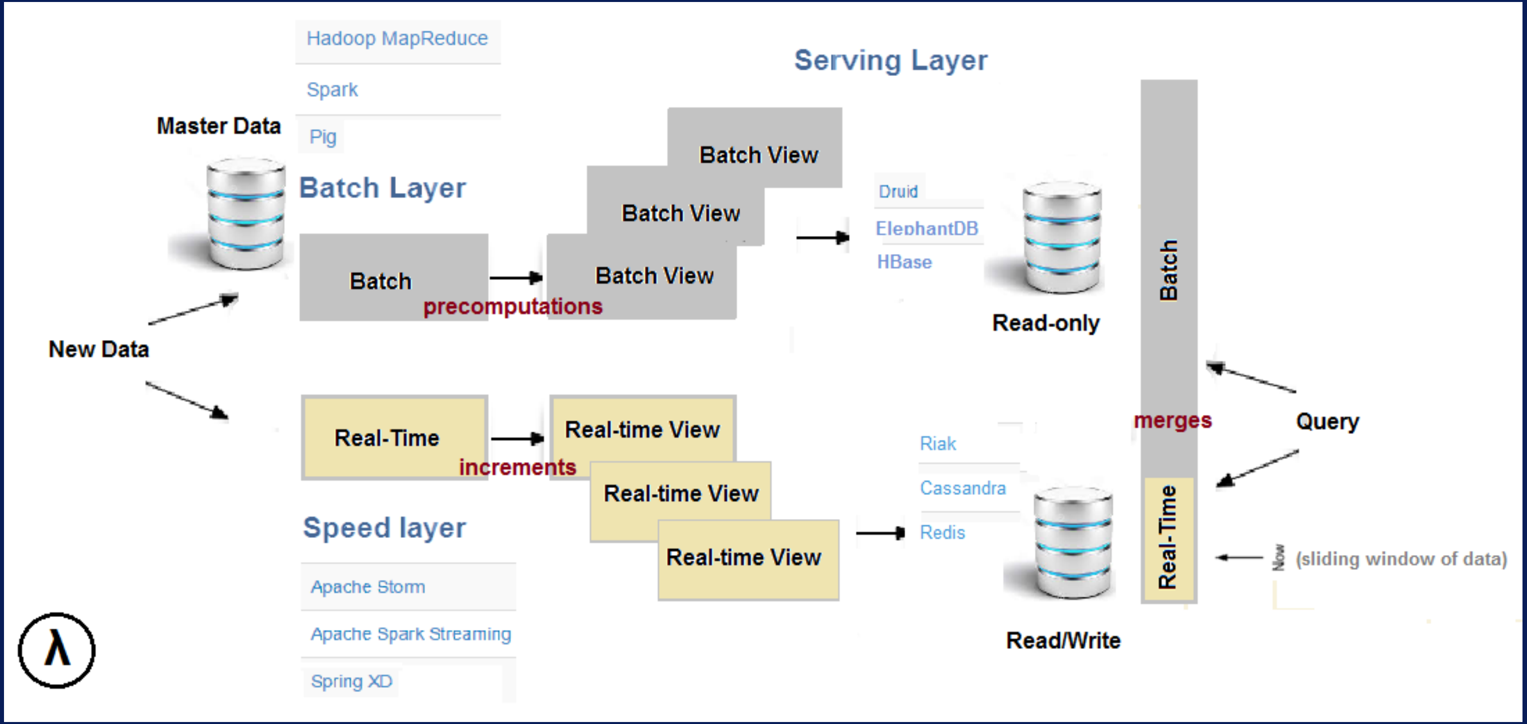
- **Batch processing on all data**
- **Batch data collection generation**

**SERVING LAYER : HBase**

- **Querying**

# Lambda Architecture:

## A Hybrid Data Processing Architecture



Hadoop MapReduce
Spark
Pig
Serving Layer
Master Data
Batch Layer
Batch View
Batch View
Batch View
Batch
precomputations
Druid
ElephantDB
HBase
Read-only
Batch
New Data
Query
merges
Real-Time
Real-time View
Real-time View
Real-time View
increments
Riak
Cassandra
Redis
Speed layer
Apache Storm
Apache Spark Streaming
Spring XD
Real-Time
Now
(sliding window of data)
Read/Write
λ