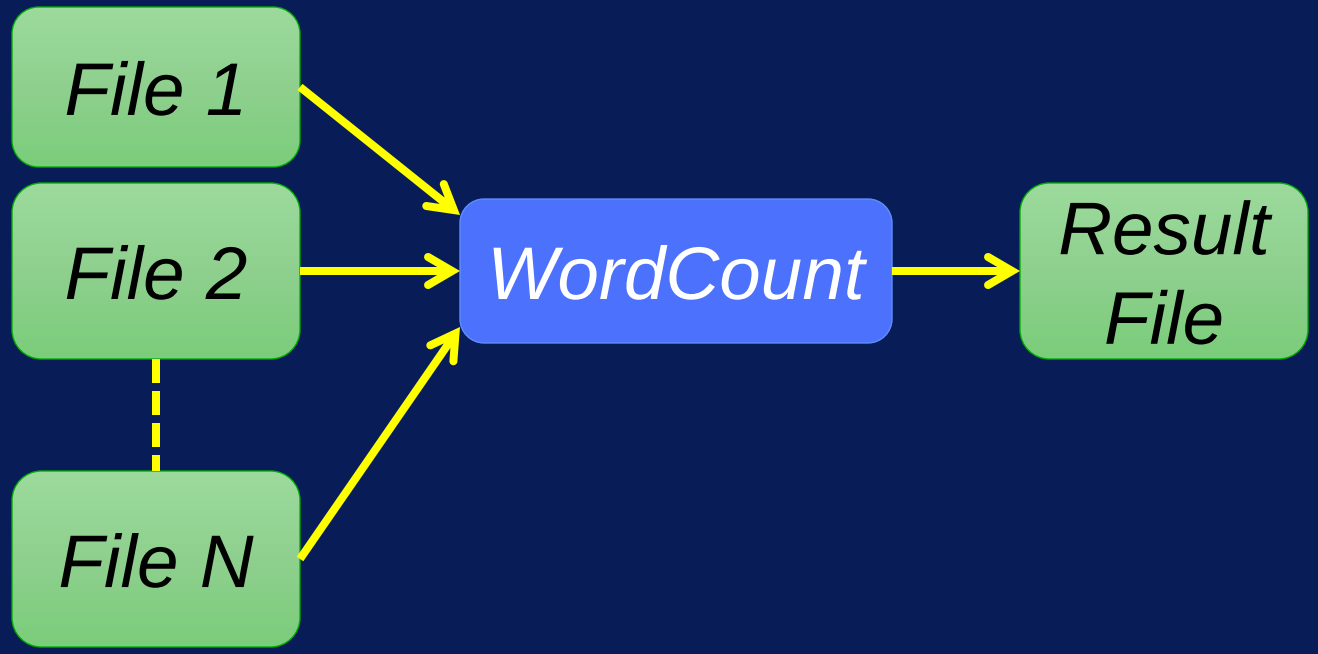
Big Data Processing Pipelines:

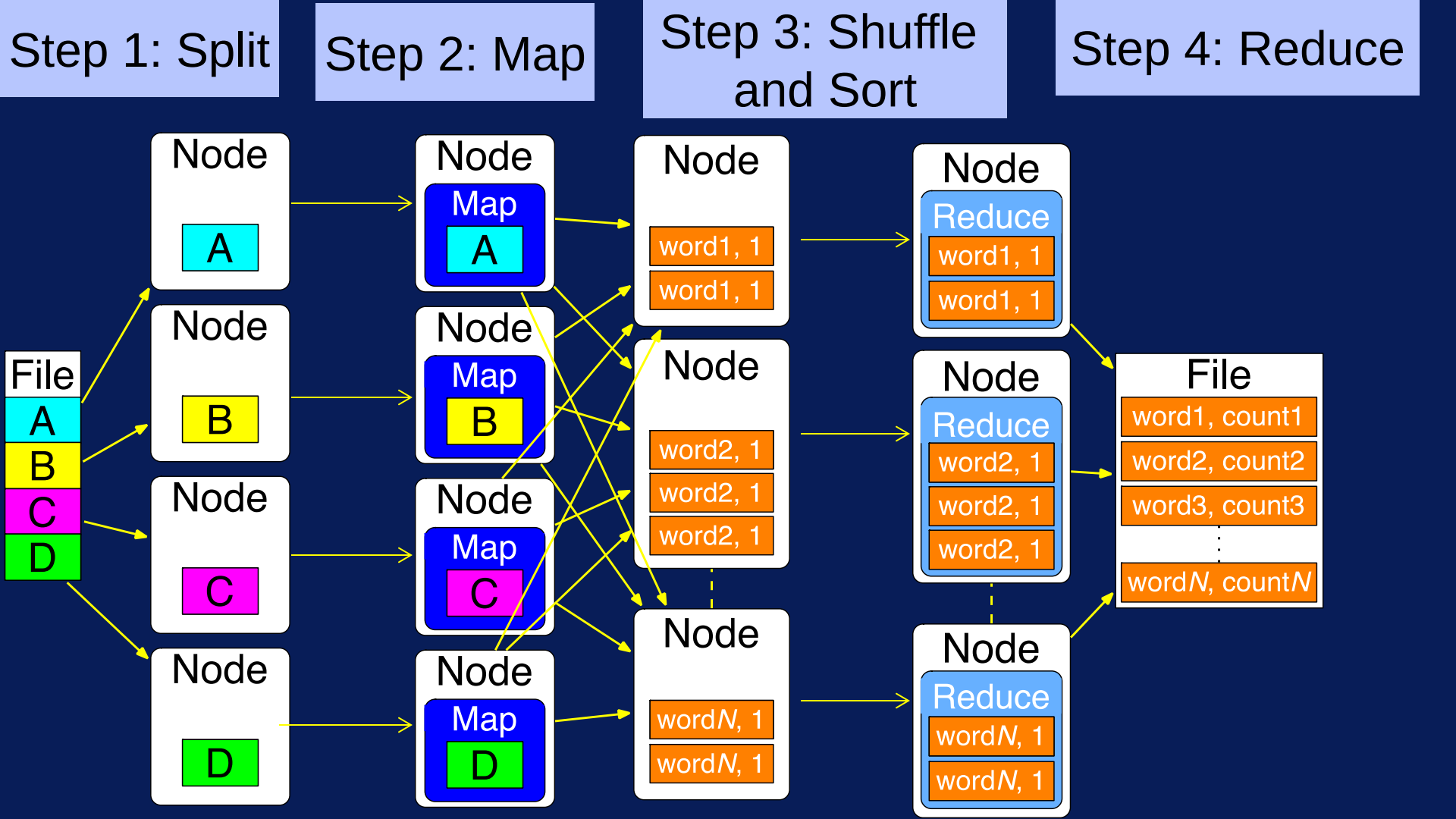
A Dataflow Approach

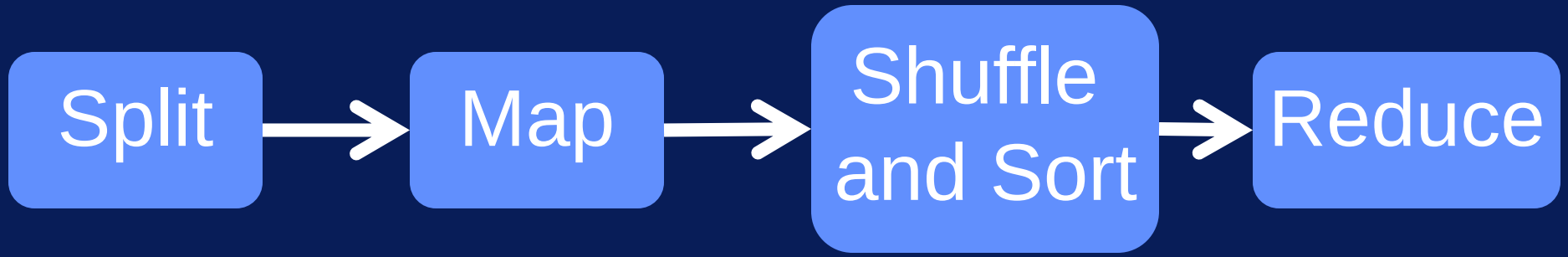
After this video you will be able to..

- Summarize what dataflow means and its role in data science
- Explain “split->do->merge” big data pipeline with examples
- Define the terms data parallel

Example MapReduce Application: WordCount

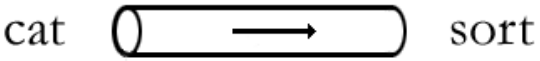



Step 1: Split
Step 2: Map
Step 3: Shuffle and Sort
Step 4: Reduce
File
A
B
C
D
Node
A
Node
B
Node
C
Node
D
Node
Map
A
Node
Map
B
Node
Map
C
Node
Map
D
Node
word1, 1
word1, 1
Node
word2, 1
word2, 1
word2, 1
Node
wordN, 1
wordN, 1
Node
Reduce
word1, 1
word1, 1
Node
Reduce
word2, 1
word2, 1
word2, 1
Node
Reduce
wordN, 1
wordN, 1
File
word1, count1
word2, count2
word3, count3
wordN, countN


Split
Map
Shuffle and Sort
Reduce

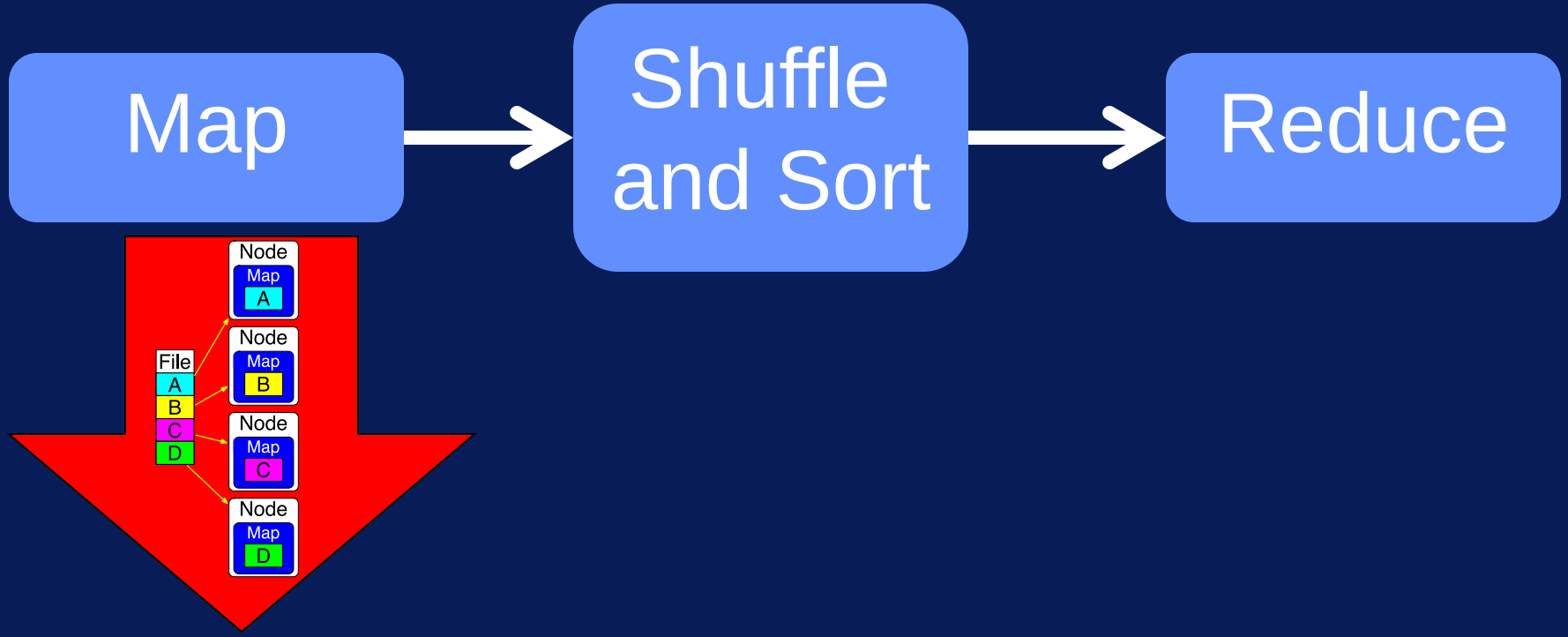
Represents a large number of applications.

Big Data Pipelines

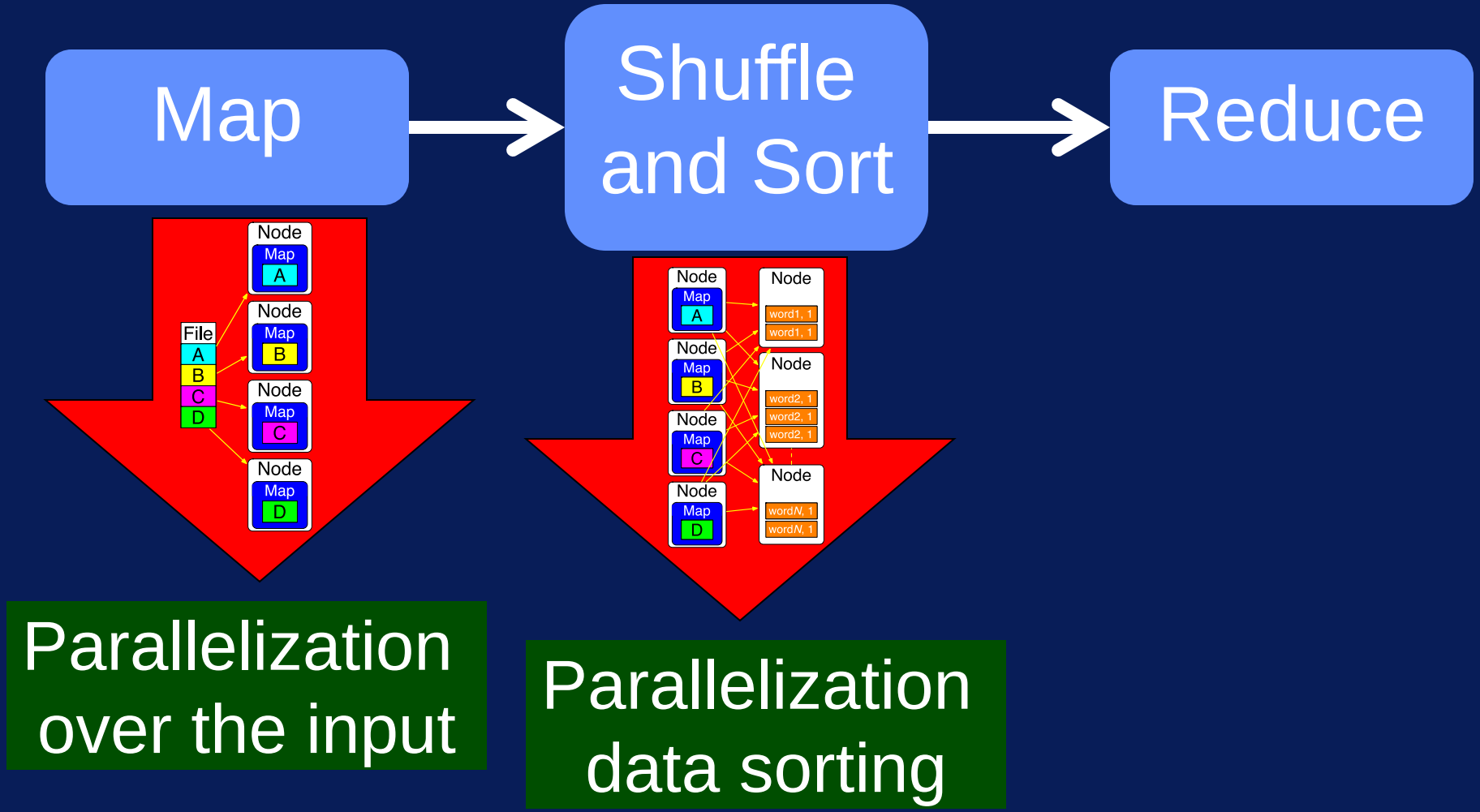


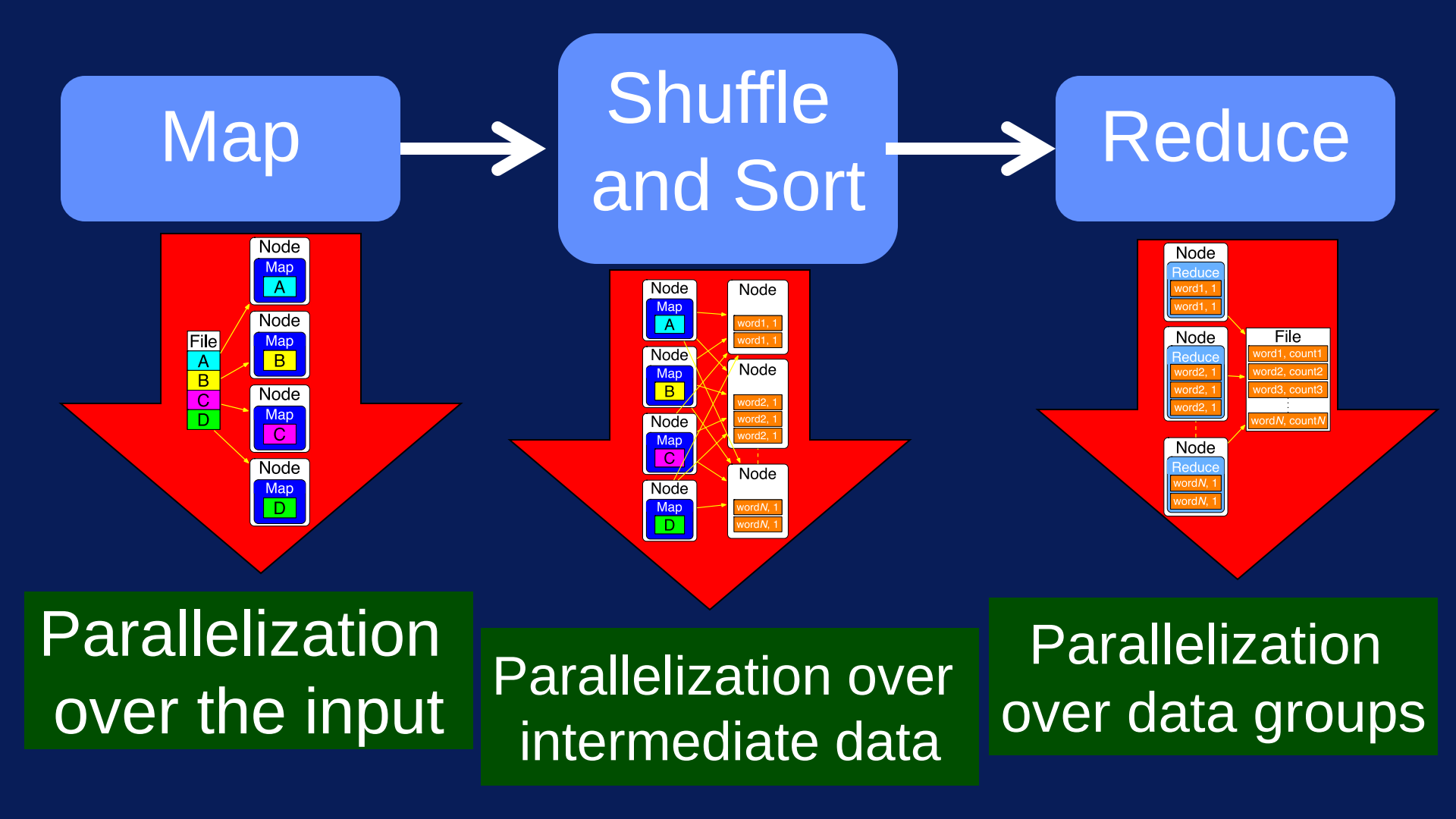
A UNIX pipe provides one-way communication
between two processes on the same computer

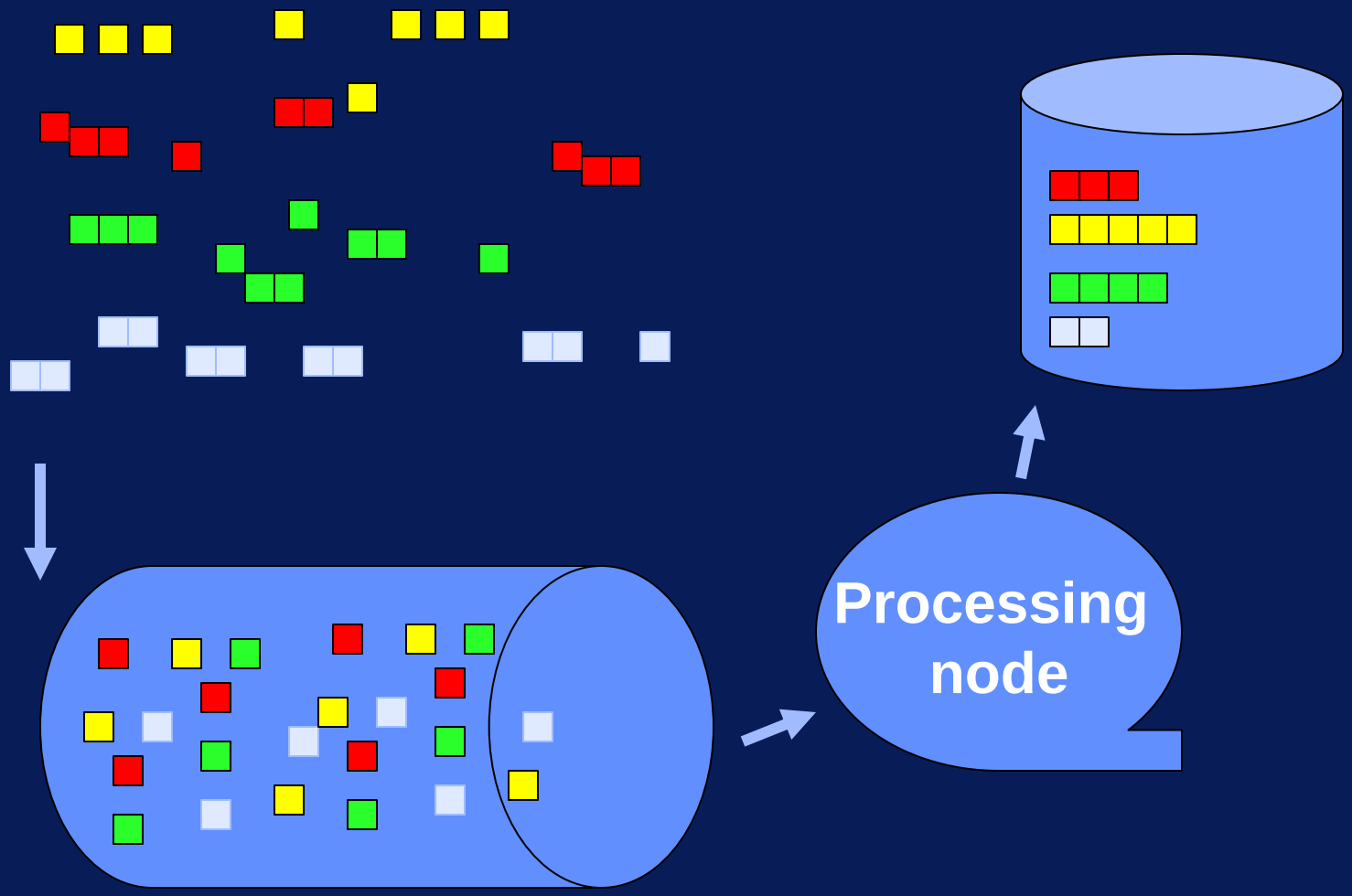

Map
Shuffle and Sort
Reduce

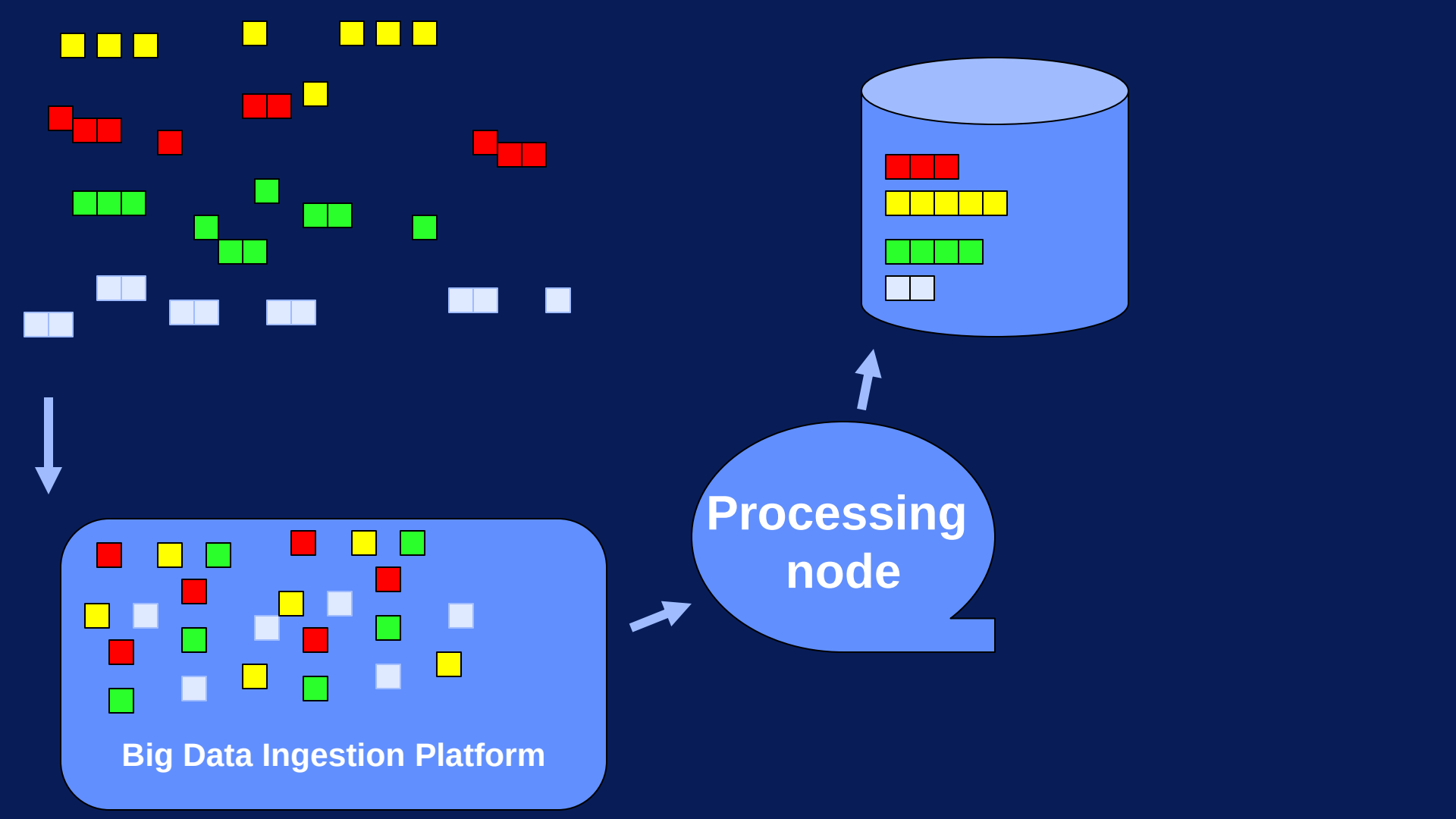


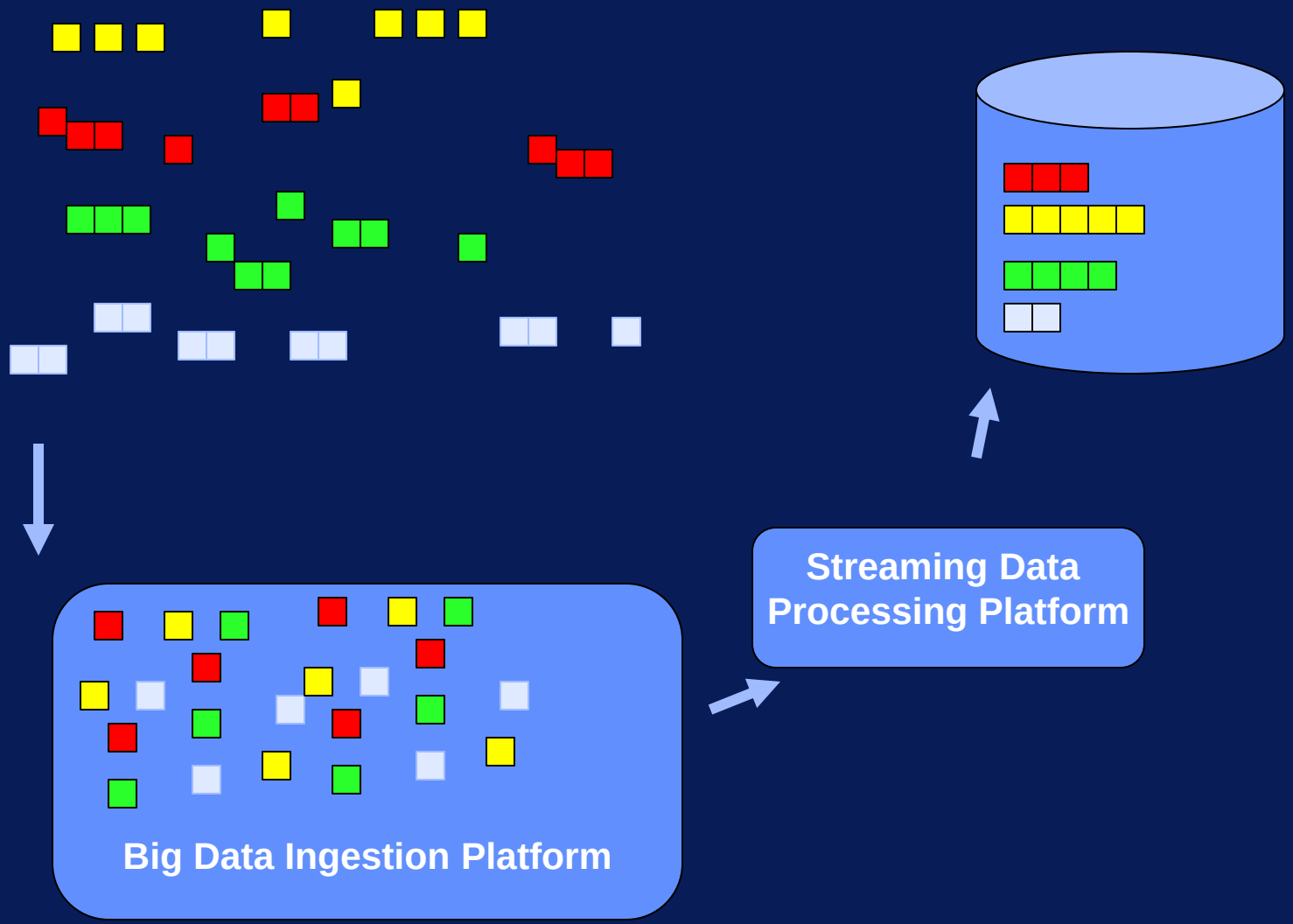
Parallelization over the input

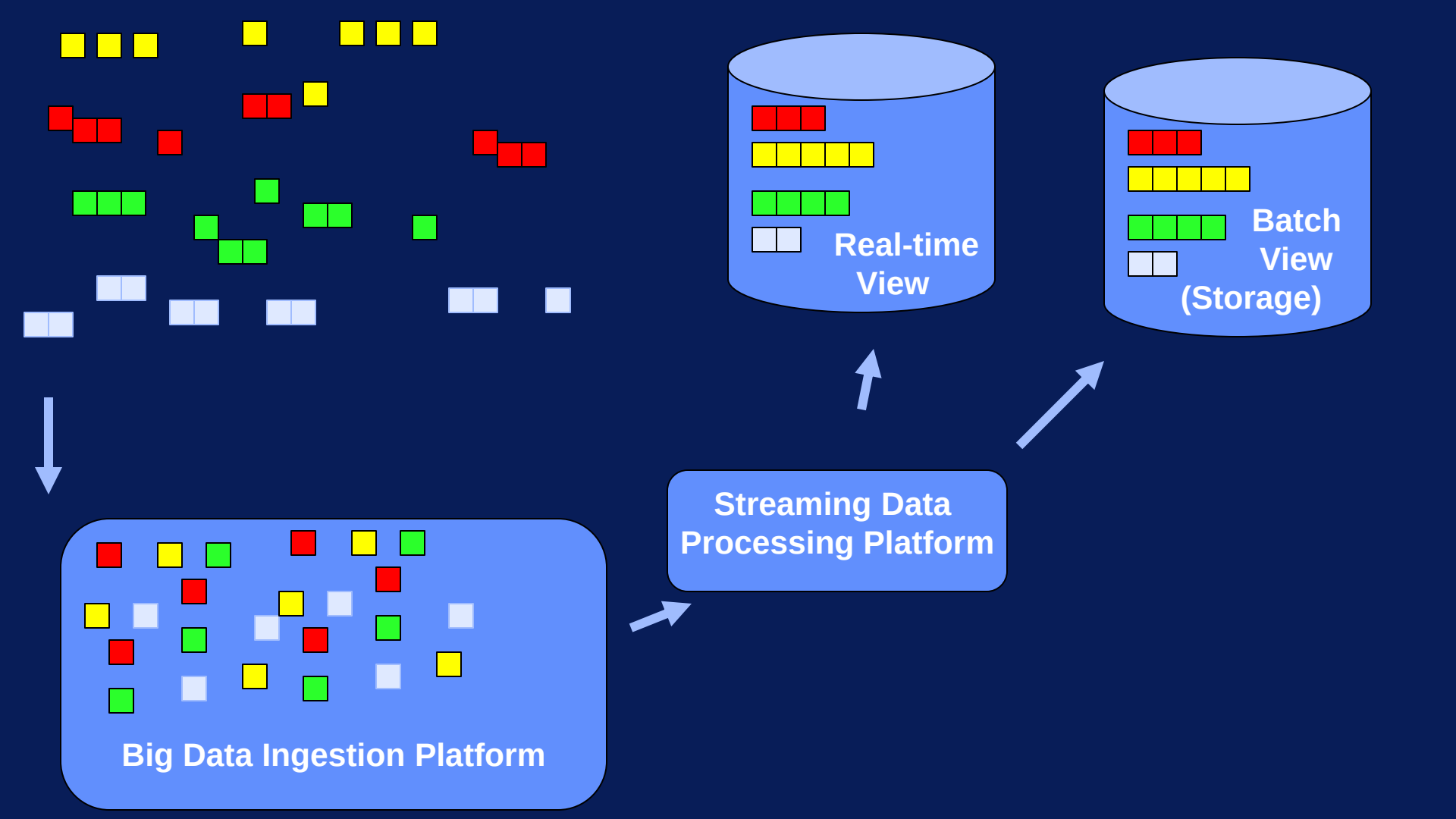

Map
Shuffle and Sort
Reduce
File
A
B
C
D
Node
Map
A
Node
Map
B
Node
Map
C
Node
Map
D
Node
Map
A
Node
Map
B
Node
Map
C
Node
Map
D
Node
word1, 1
word1, 1
Node
word2, 1
word2, 1
word2, 1
Node
wordN, 1
wordN, 1
Parallelization over the input
Parallelization data sorting


Map
Shuffle and Sort
Reduce
File
A
B
C
D
Node
Map
A
Node
Map
B
Node
Map
C
Node
Map
D
Node
Map
A
Node
Map
B
Node
Map
C
Node
Map
D
Node
word1, 1
word1, 1
Node
word2, 1
word2, 1
word2, 1
Node
wordN, 1
wordN, 1
Node
Reduce
word1, 1
word1, 1
Node
Reduce
word2, 1
word2, 1
word2, 1
Node
Reduce
wordN, 1
wordN, 1
File
word1, count1
word2, count2
word3, count3
wordN, countN
Parallelization over the input
Parallelization over intermediate data
Parallelization over data groups


Processing node


Big Data Ingestion Platform
Processing node


Streaming Data Processing Platform
Big Data Ingestion Platform


Real-time View
Batch View (Storage)
Streaming Data Processing Platform
Big Data Ingestion Platform

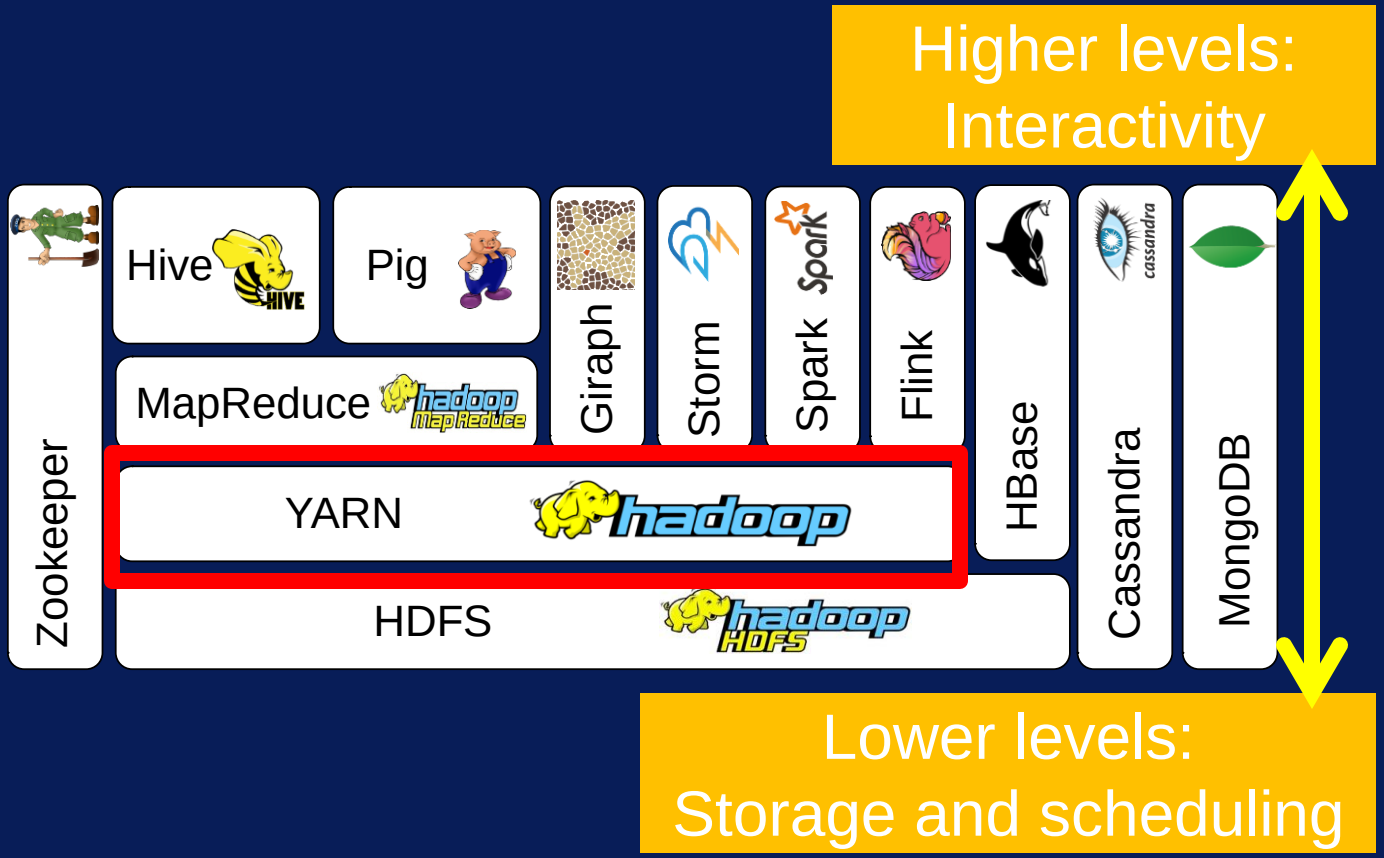

Split
Do Something
Merge

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



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

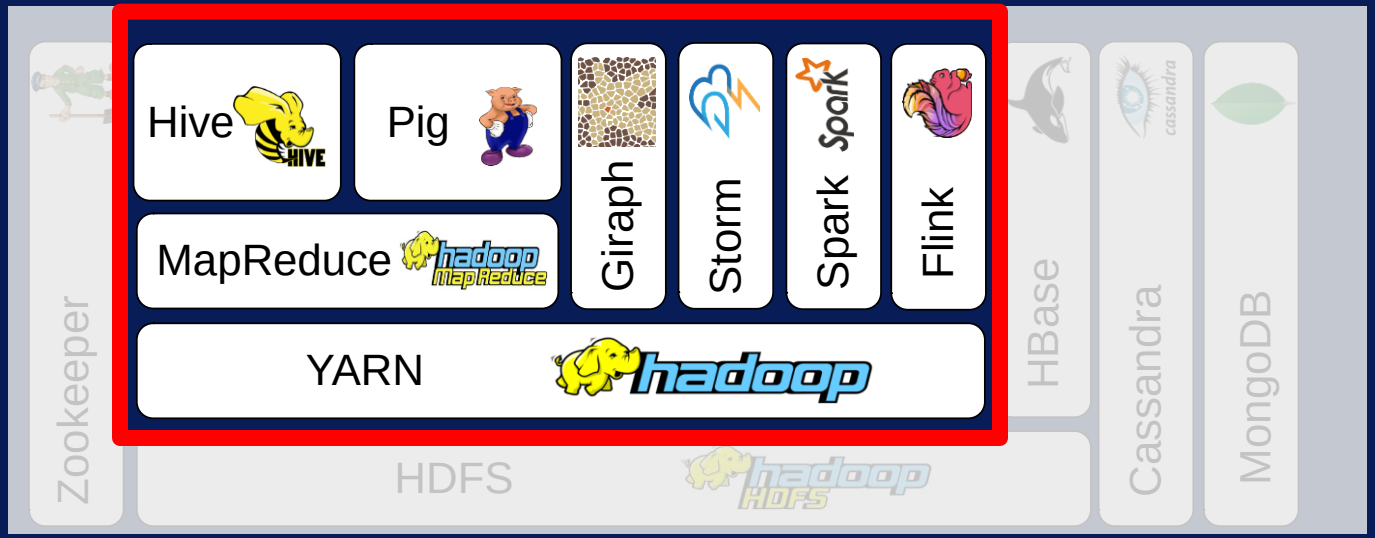
Another way to look at the Hadoop Ecosystem

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

OOZIE

Apache Zookeeper

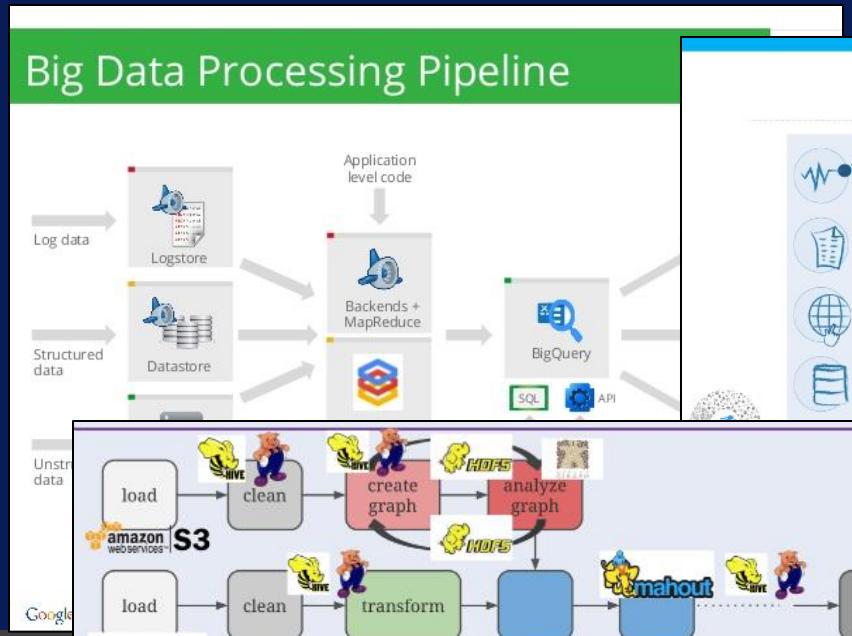
Another way to look at the Hadoop Ecosystem

COORDINATION AND WORKFLOW MANAGEMENT

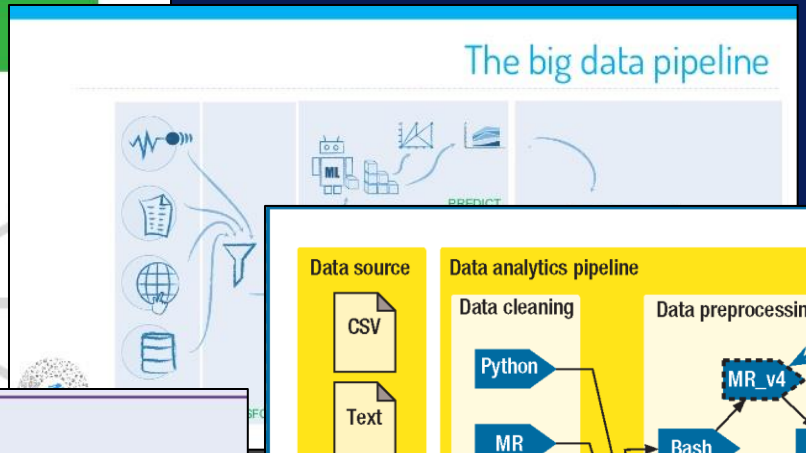
DATA INTEGRATION AND PROCESSING

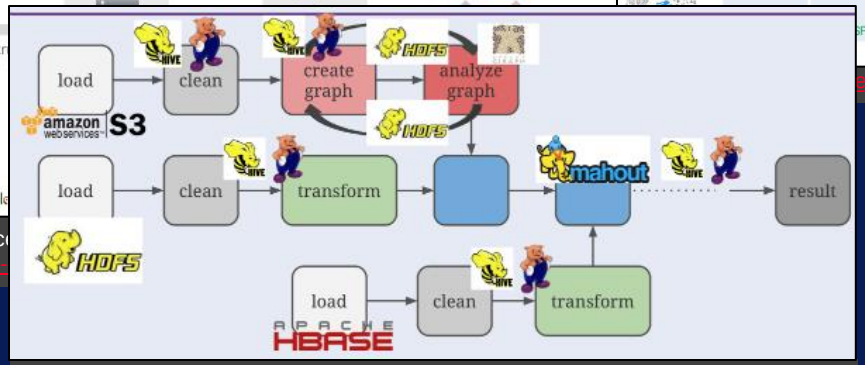
DATA MANAGEMENT AND STORAGE

Example Big Data Processing Pipelines

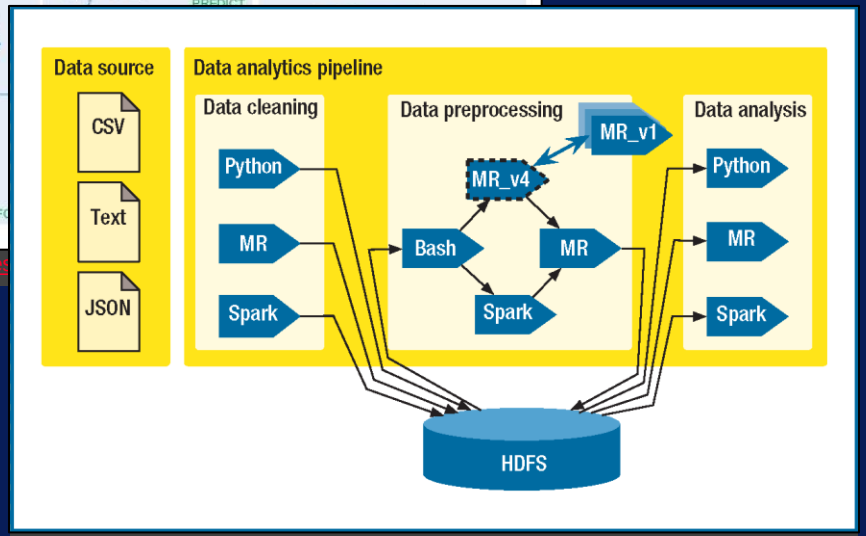


Source: cloud





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

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	

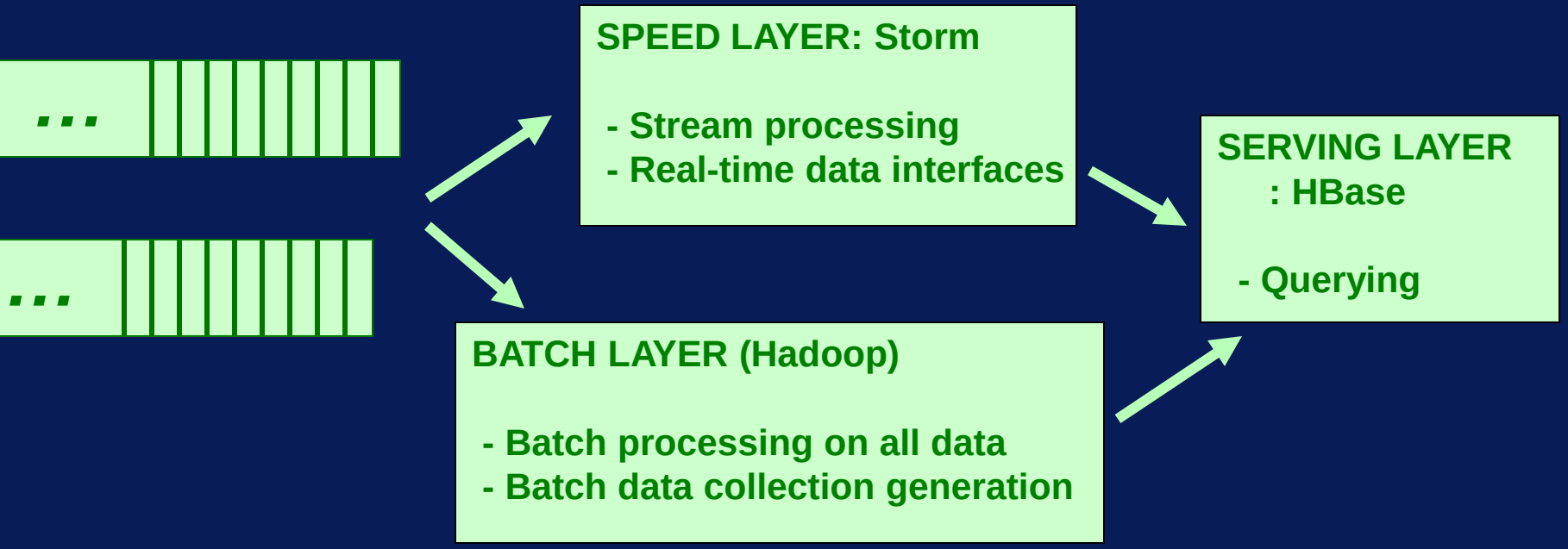
Beam

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

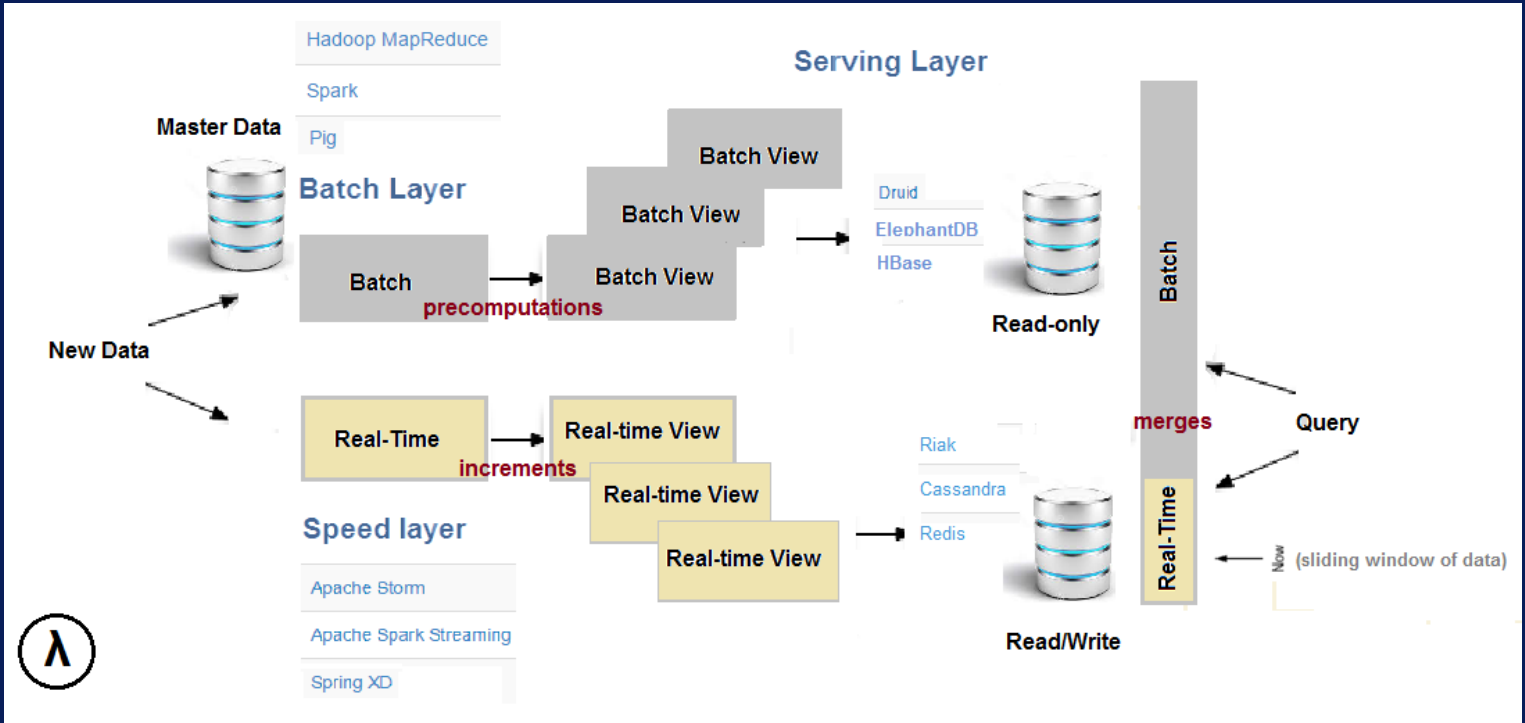
Storm

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

Lambda Architecture:
A Hybrid Data Processing Architecture



Lambda Architecture:
A Hybrid Data Processing Architecture



Introduction to Apache Spark

After this video you will be able to..

- List the main motivations for the development of Spark
- Draw the Spark stack as a layer diagram
- Explain the functionality of the components in the Spark stack

Why Spark?

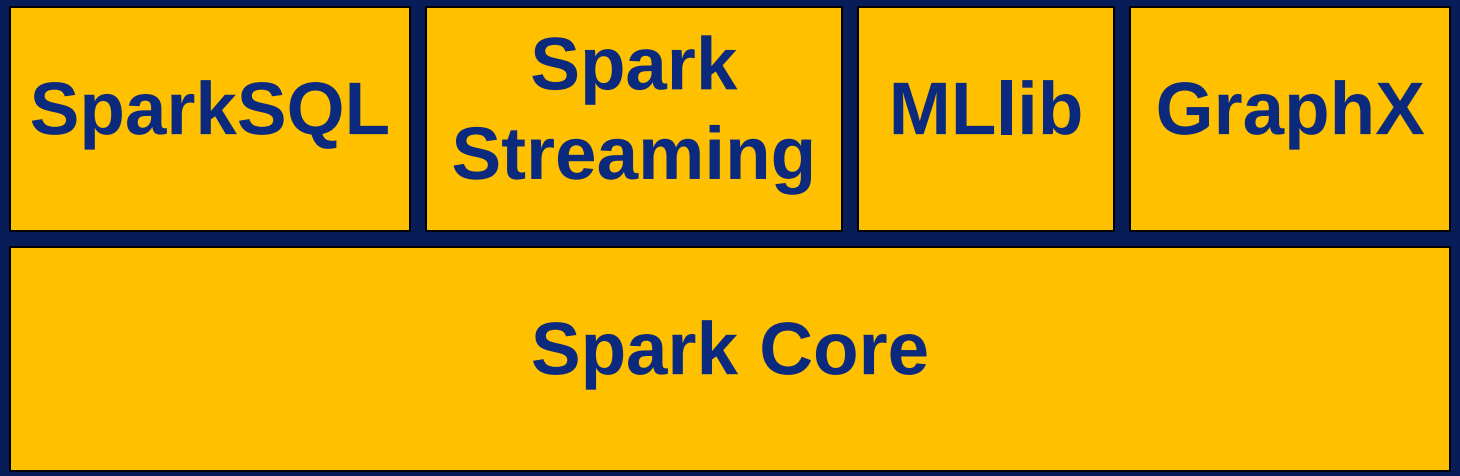
Hadoop MapReduce Shortcomings

- Only for Map and Reduce based computations
- Relies on reading data from HDFS
- Native support for Java only
- No interactive shell support
- No support for streaming

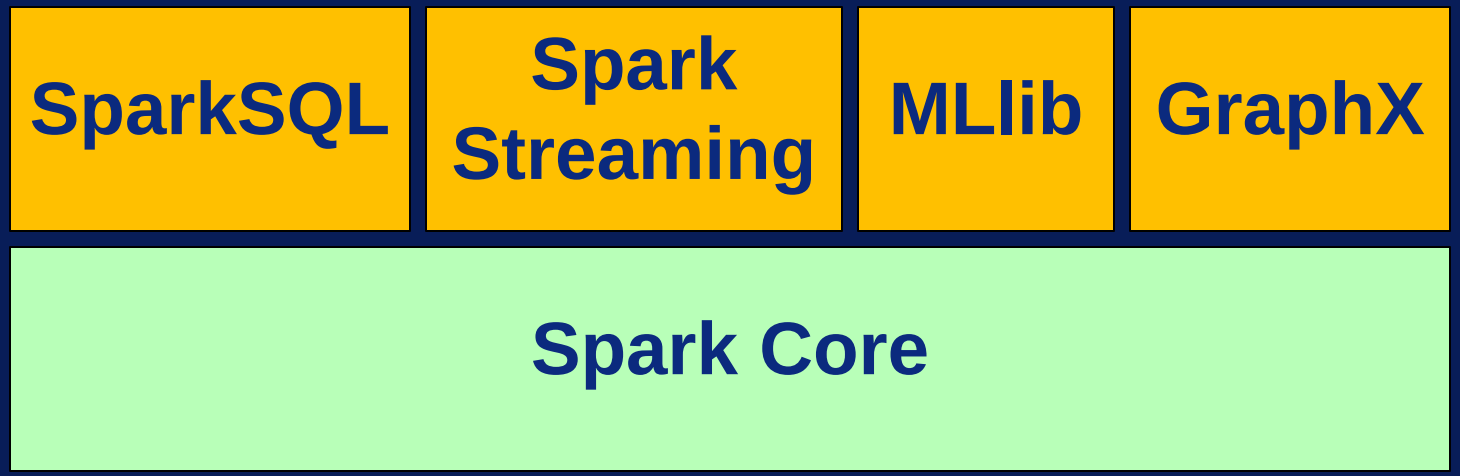
Basics of Data Analysis with Spark

- Expressive programing model
- In-memory processing
- Support for diverse workloads
- Interactive shell

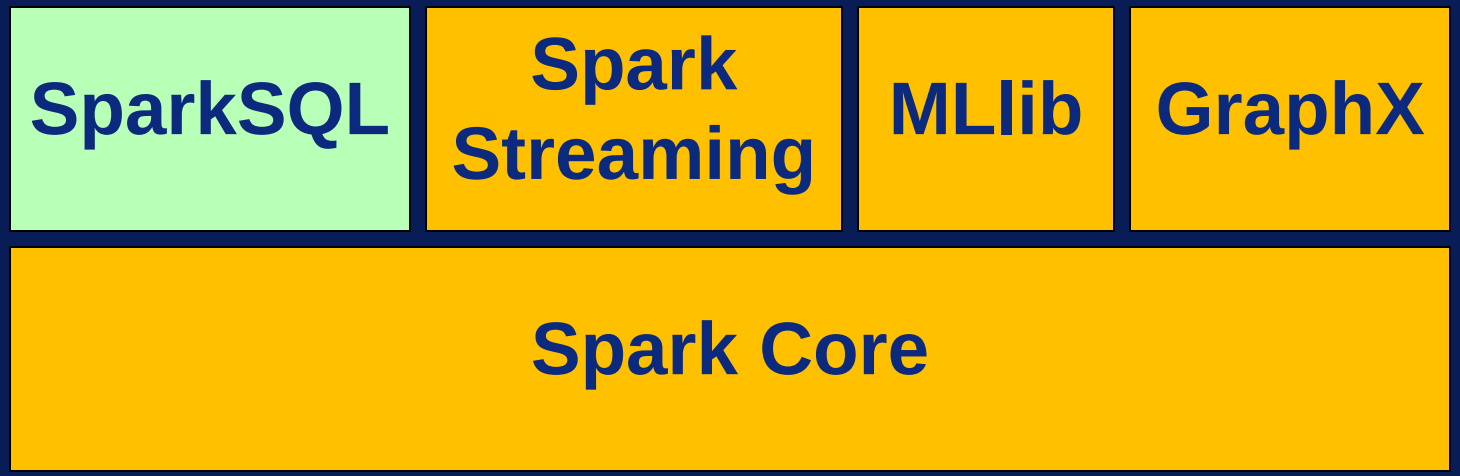
The Spark Stack



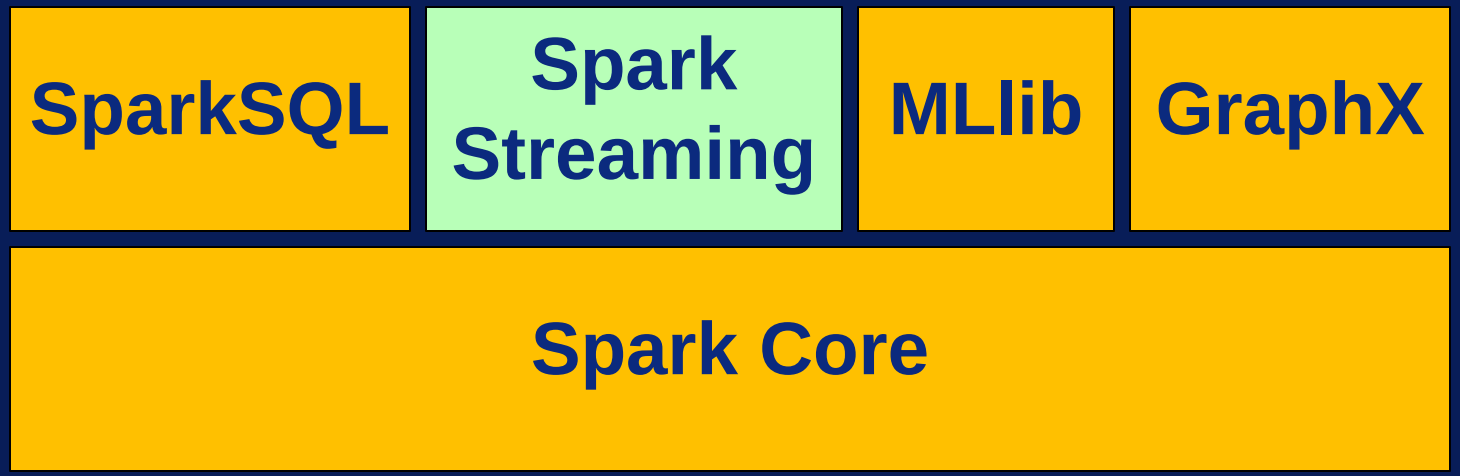
The Spark Stack



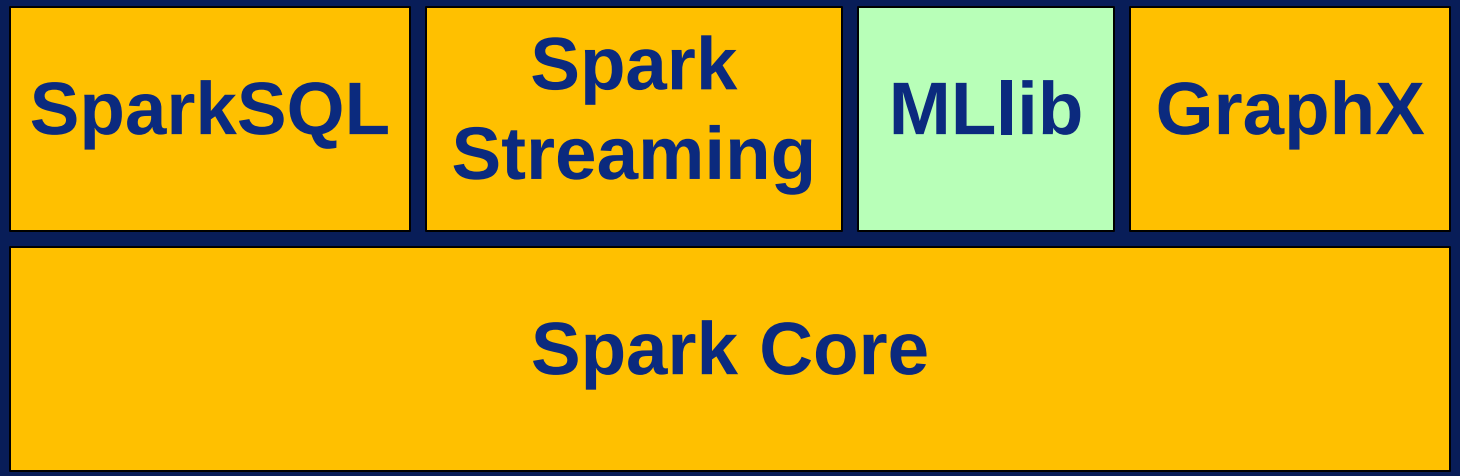
The Spark Stack



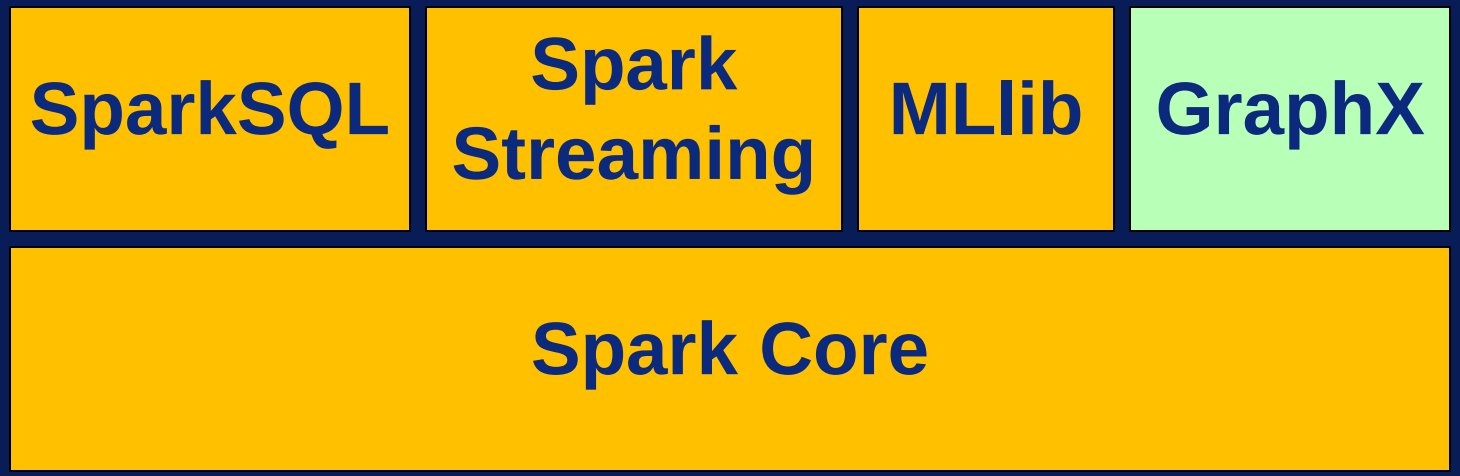
The Spark Stack

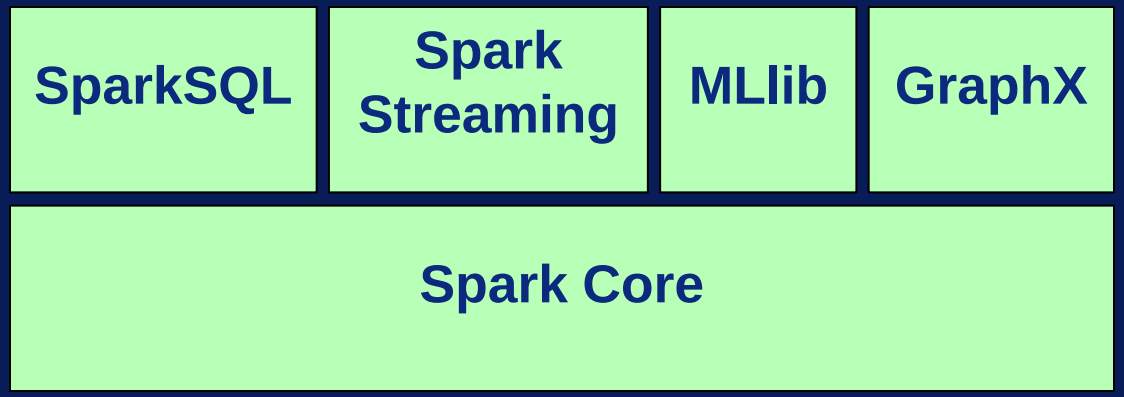


The Spark Stack

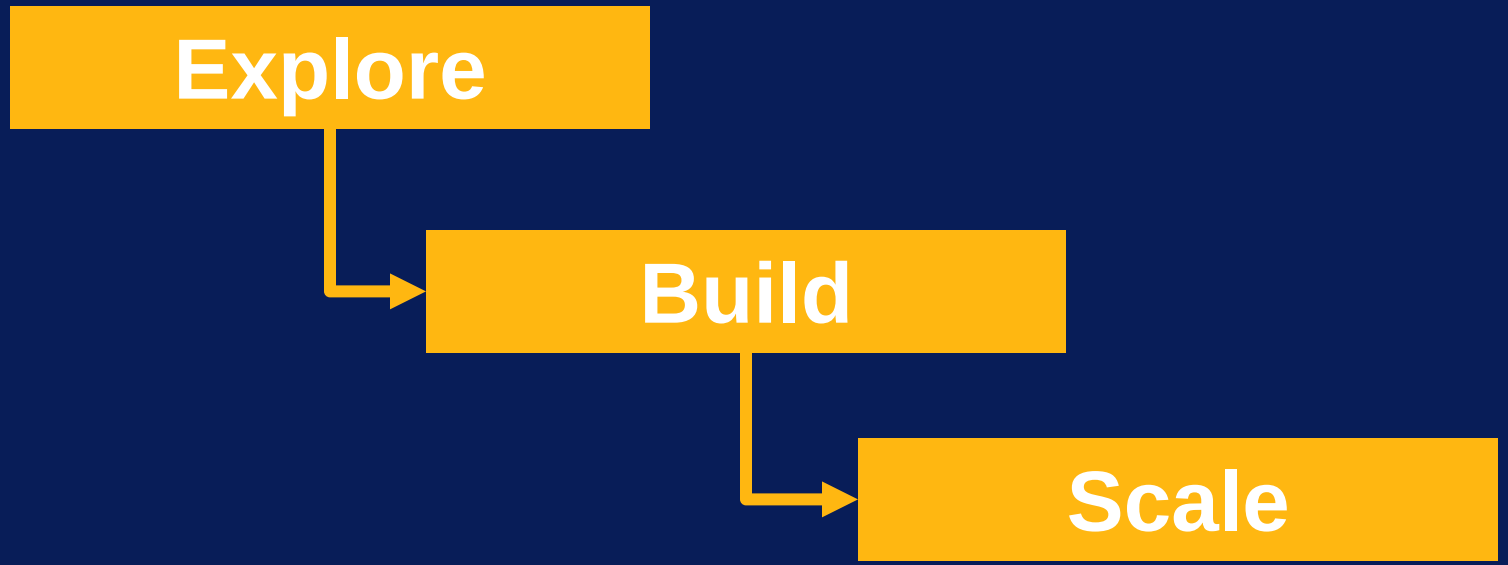


The Spark Stack




SparkSQL
Spark Streaming
MLlib
GraphX
Spark Core


Spark


Explore
Build
Scale

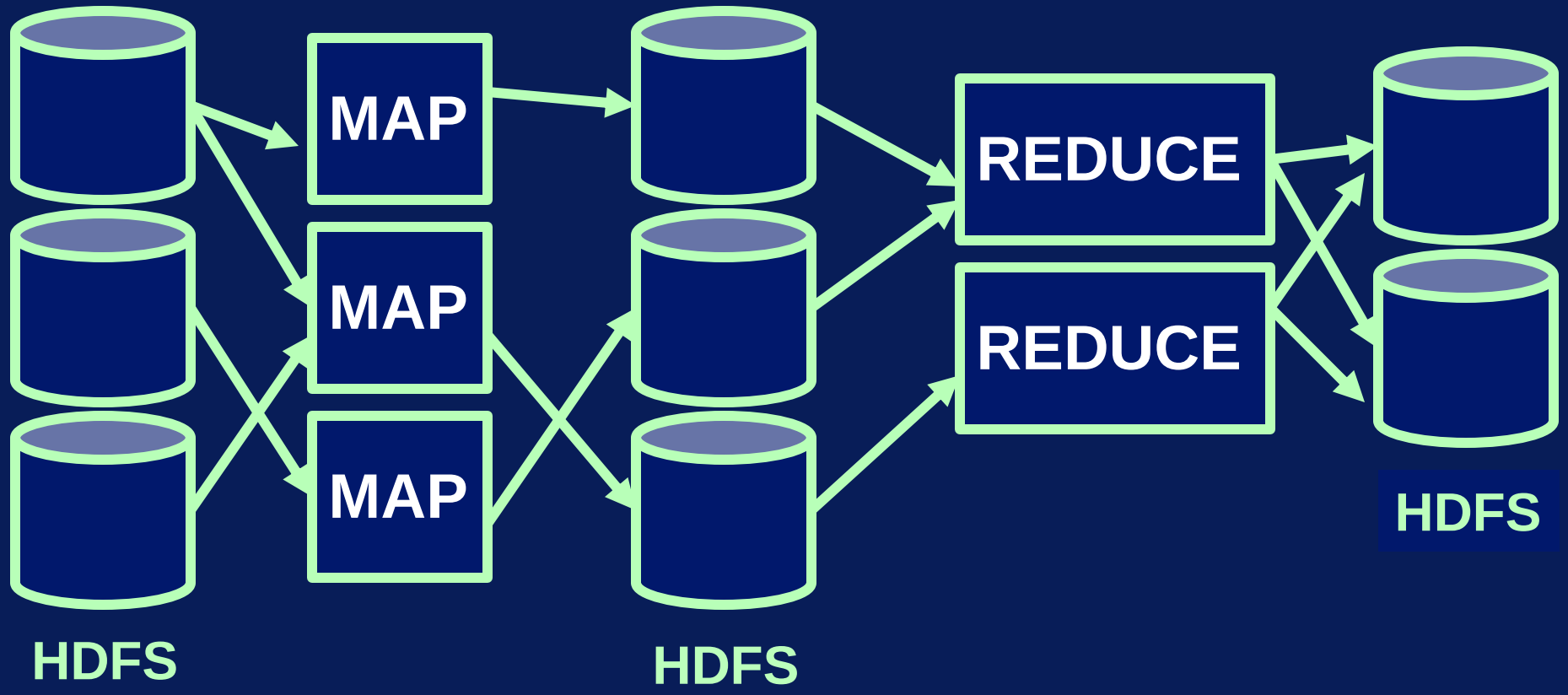
Getting Started with Spark: The Architecture and Basic Concepts

After this video you will be able to..

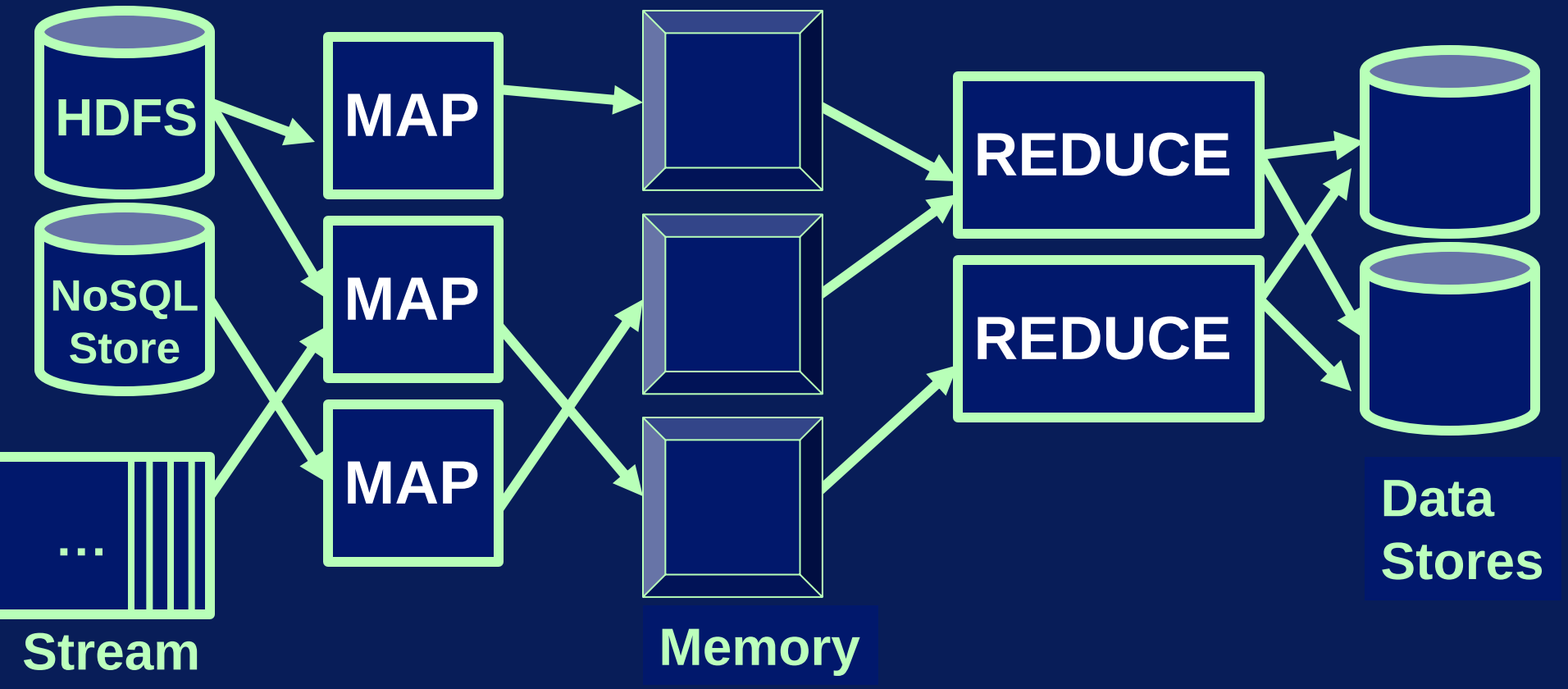
- Describe how Spark does in-memory processing using the RDD abstraction
- Explain the inner workings of the Spark architecture
- Summarize how Spark manages and executes code on Clusters

What does in memory processing mean?

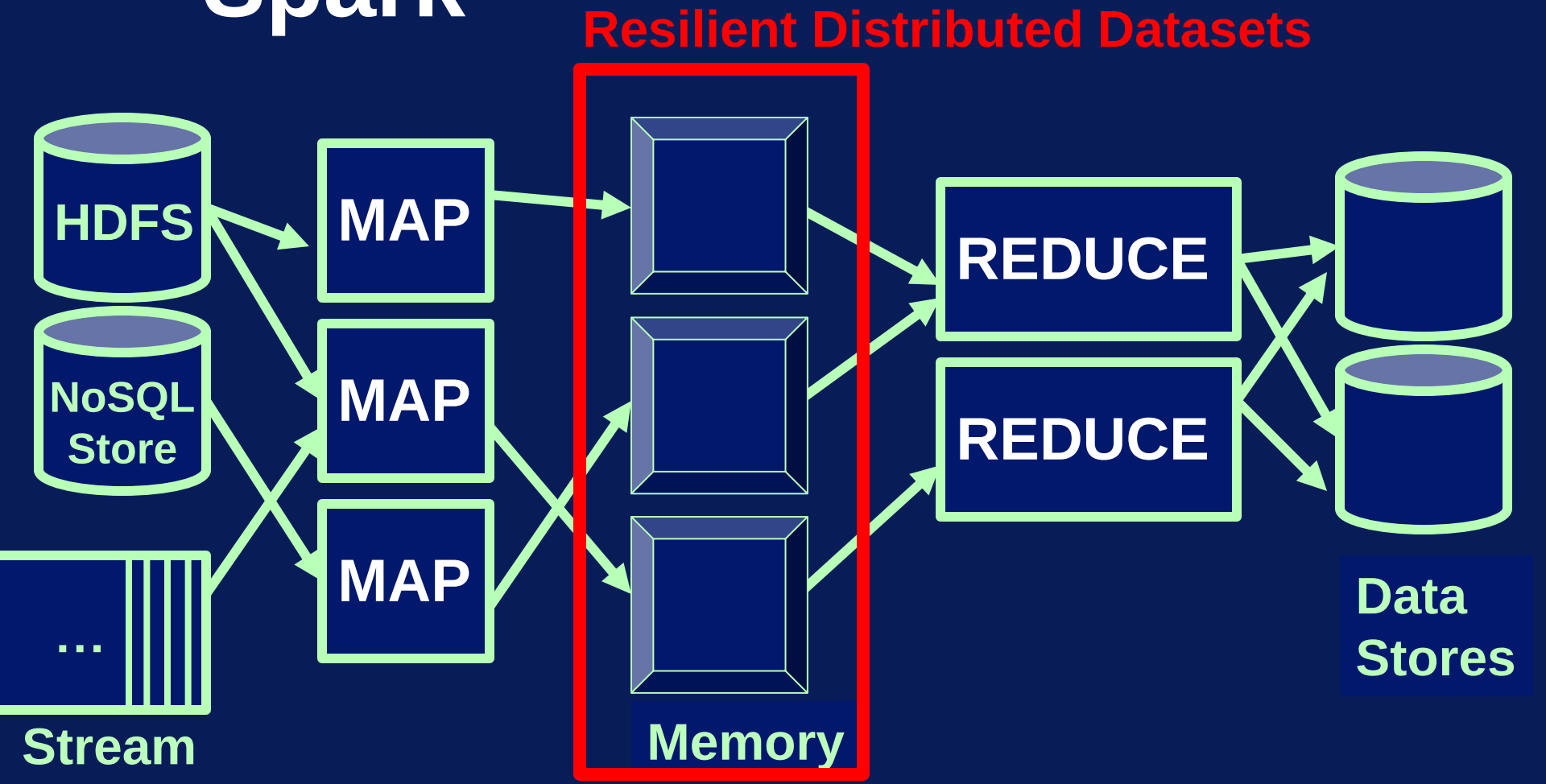
MapReduce



Spark


HDFS
NoSQL Store
...
Stream
MAP
MAP
MAP
Memory
REDUCE
REDUCE
Data Stores

Spark


Resilient Distributed Datasets
HDFS
NoSQL Store
...
Stream
MAP
MAP
MAP
Memory
REDUCE
REDUCE
Data Stores

Resilient Distributed Datasets

Dataset

Data storage created from: HDFS, S3, HBase, JSON, text, Local hierarchy of folders

Or created transforming another RDD

Resilient Distributed Datasets

Distributed

Distributed across the cluster of machines

Divided in partitions, atomic chunks of data

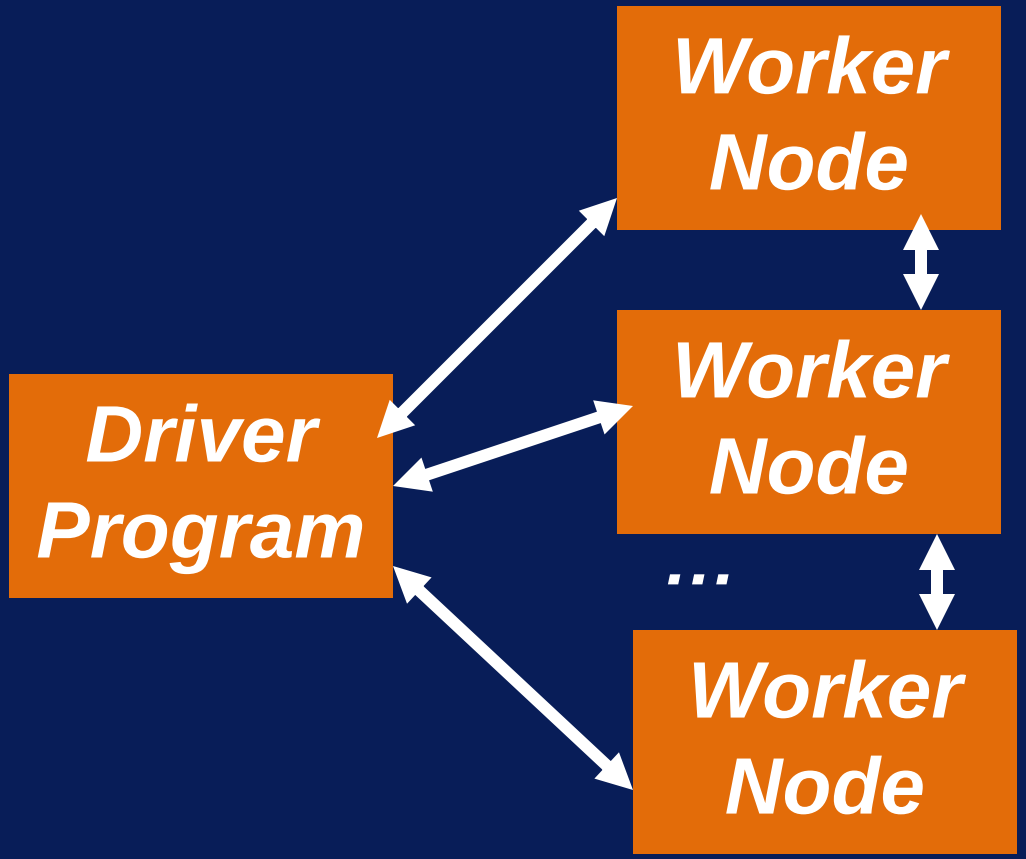
Resilient Distributed Datasets

Resilient

Recover from errors, e.g. node failure, slow processes

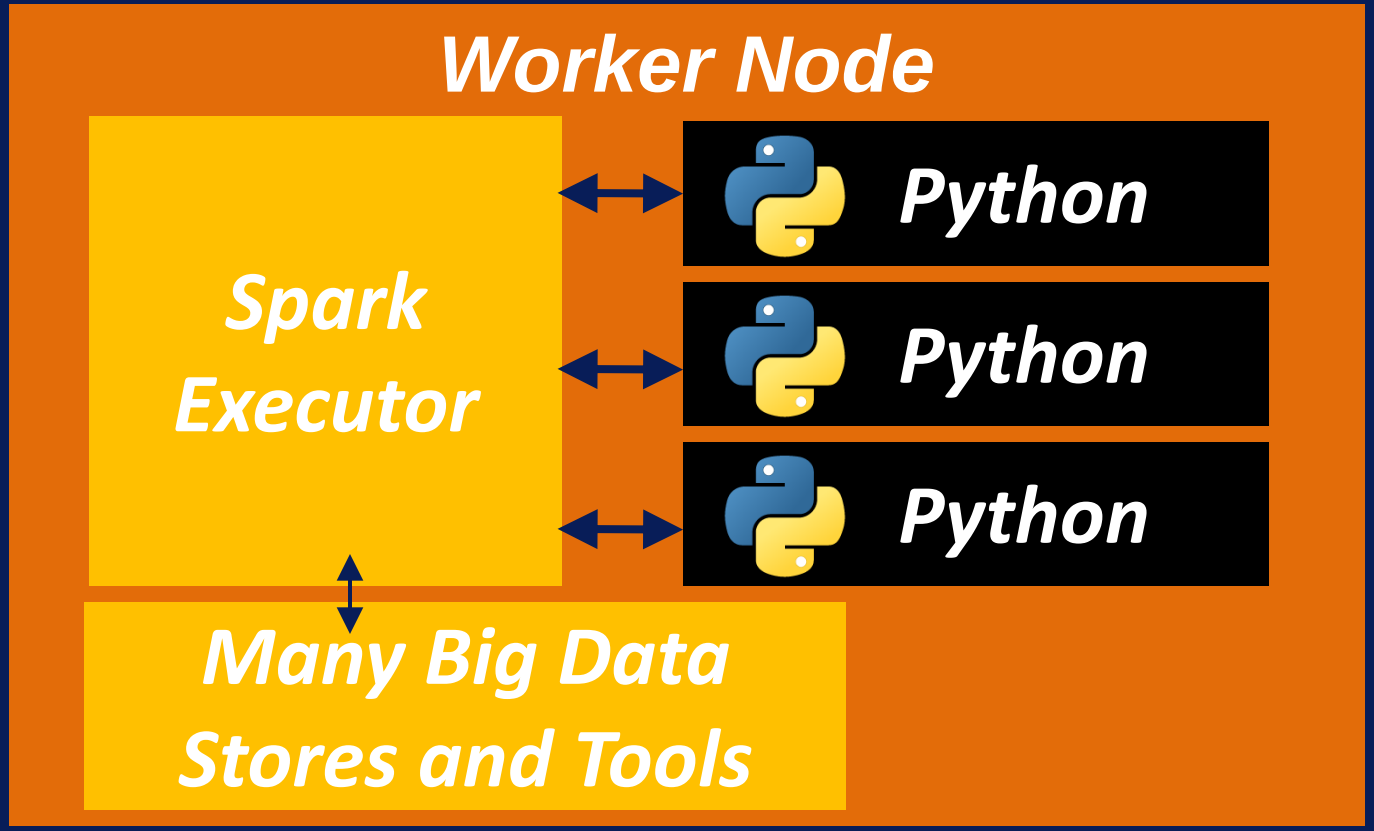
Track history of each partition, re-run

Spark Architecture

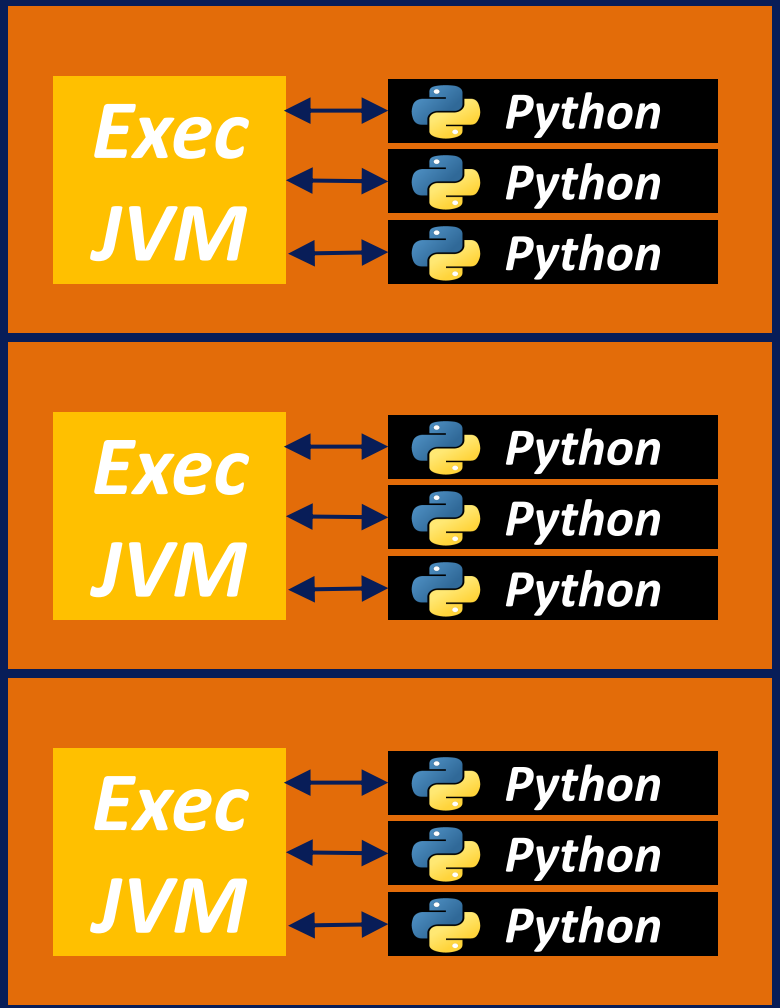

Worker Node
Worker Node
Driver Program
...
Worker Node

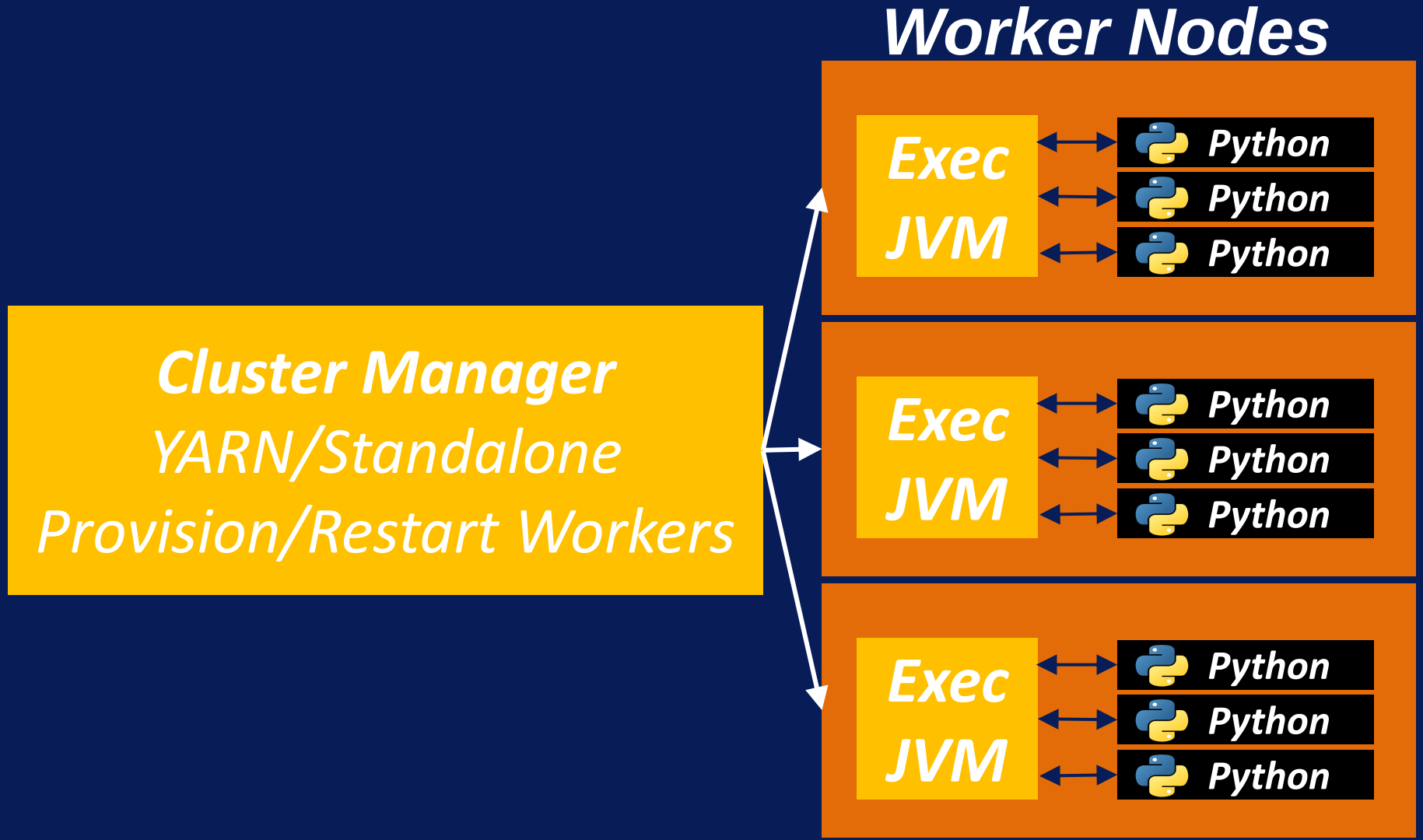
Driver Program

```
In [1]: lines = sc.textFile("hdfs:/user/cloudera/words.txt")
```


Worker Node
Spark Executor
Python
Python
Python
Many Big Data Stores and Tools

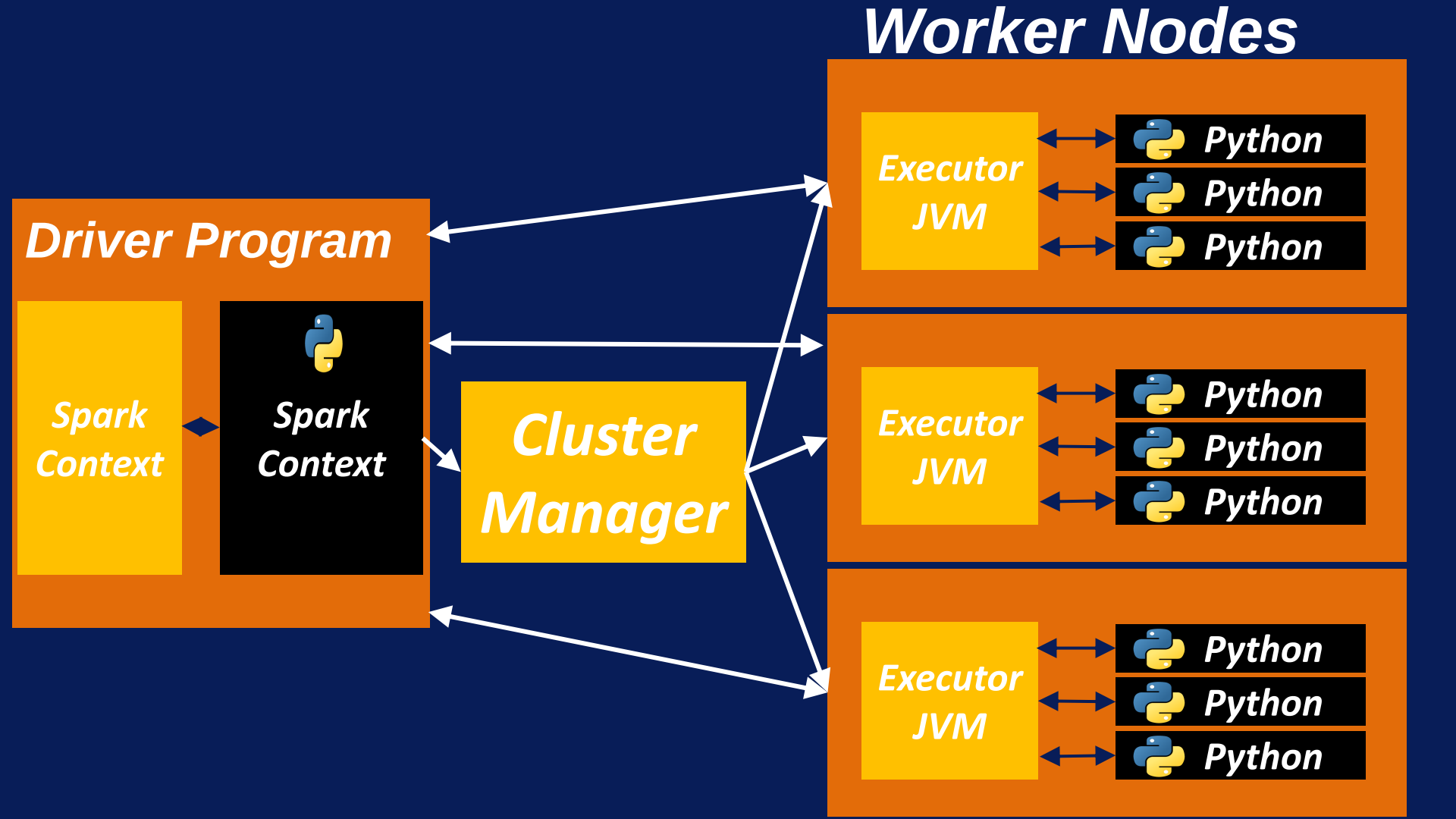
Worker Nodes



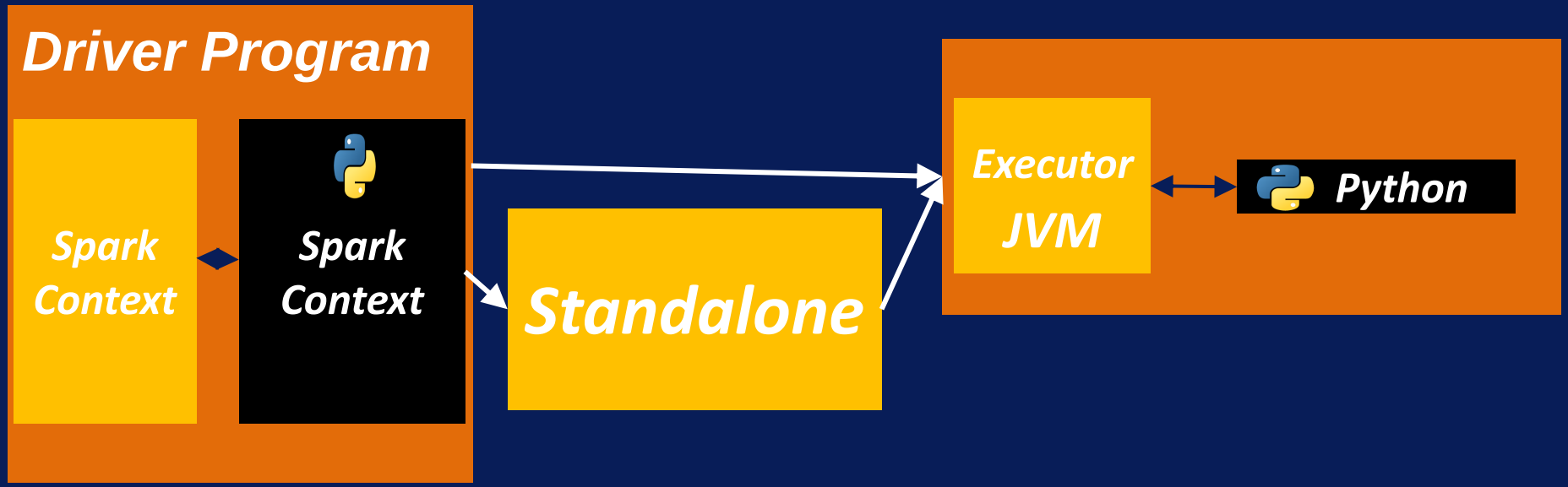

Worker Nodes
Cluster Manager
YARN/Standalone
Provision/Restart Workers
Exec JVM
Python
Python
Python
Exec JVM
Python
Python
Python
Exec JVM
Python
Python
Python

Which cluster manager?

http://www.agildata.com/apache-spark-cluster-managers-yarn-mesos-or-standalone/


Worker Nodes
Driver Program
Spark Context
Spark Context
Cluster Manager
Executor JVM
Python
Python
Python
Executor JVM
Python
Python
Python
Executor JVM
Python
Python
Python

Cloudera VM


Driver Program
Spark Context
Spark Context
Standalone
Executor JVM
Python