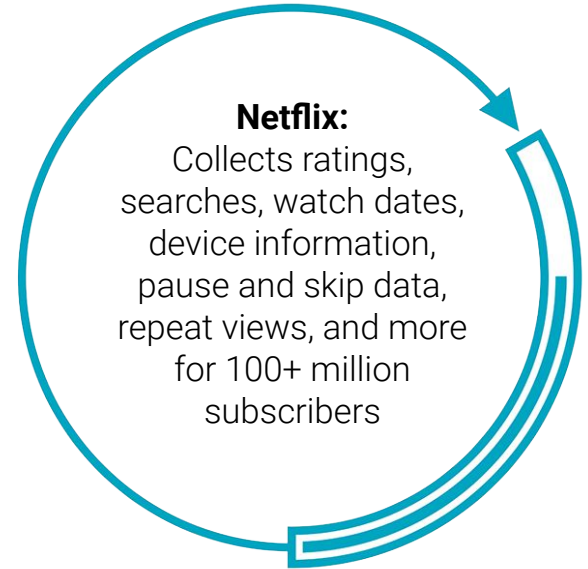
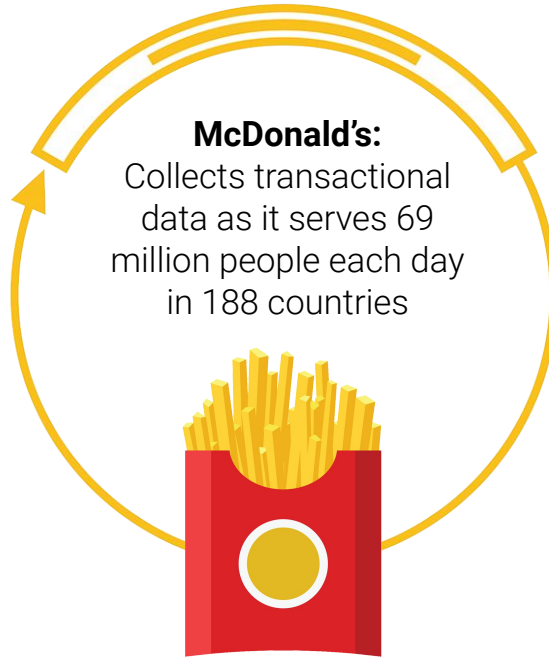
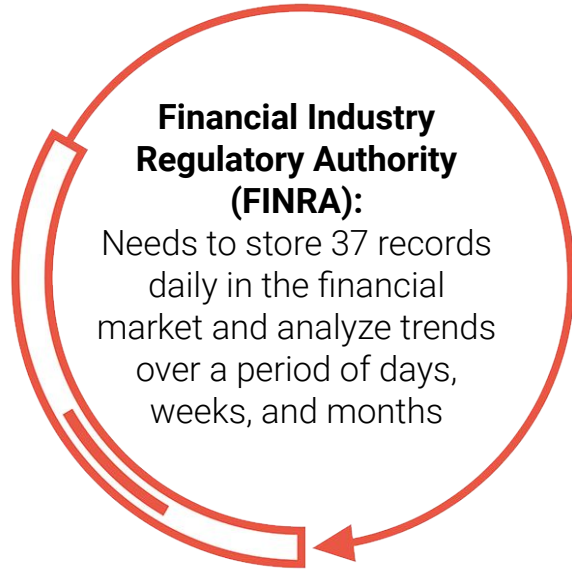


**Data Boot Camp**  
Lesson 22.1



# Big Data, Big Problems

---



# Class Objectives:

---

By the end of today's class you will:



Articulate the different components in the Hadoop ecosystem and their primary use cases.



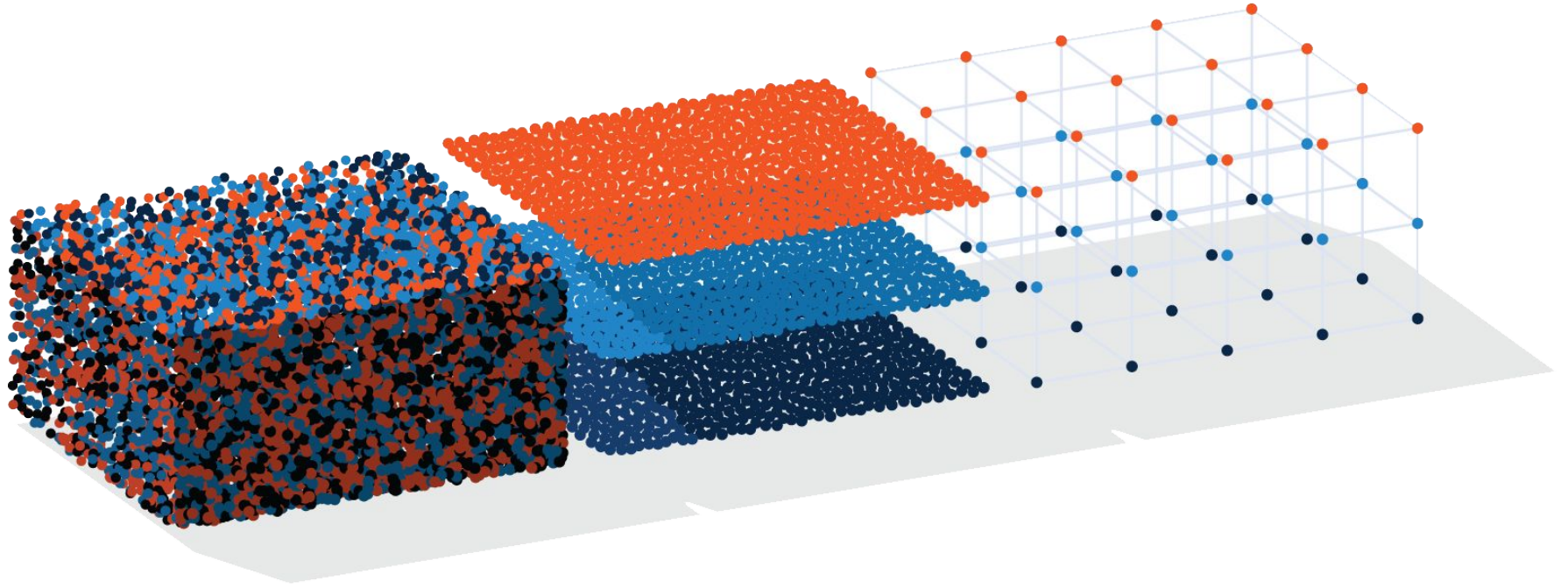
Run MapReduce jobs.



Use Spark DataFrames to process large datasets.

# What Is Big Data?

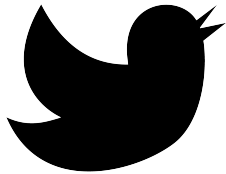
---



# What Is Big Data?

---

Big data includes stock exchange data, emails, and social media posts such as Facebook statuses and tweets.



It also includes lesser known things like supply chains, barcodes, and cell towers.



Big data has always existed, there just hasn't been a way to gather and analyze it.



# Four Vs of Big Data

---

01

**Volume: Size of the data** (terabytes of stock data)

02

**Velocity: How quickly data is coming in** (car sensors sending information every second)

03

**Variety: Different forms of data** (social media posts, comments, photos, etc.)

04

**Veracity: Uncertainty of data** (social media data may not be precise, come from bots, etc.)

# Big Data Overview

---

Visualization  
& Analytics

IBM Cognos  
SAP Business Objects  
MicroStrategy  
ZoomData  
Qlik  
PowerBI  
Tableau

Computations

Hadoop MapReduce  
HIVE  
TEZ  
Apache Drill  
Apache Storm  
Google BigQuery  
Flink  
Spark Streaming  
Kafka

Storage

IBM GPFS  
Hadoop  
MAPR Tables  
Apache HBASE  
KUDU  
MEMSQL

Distributions &  
Data Warehouse

Pivotal  
Oracle EXADATA  
TERADATA  
GreenPlum  
MAPR  
NETEZZA  
HortonWorks  
Cloudera  
Azure HDInsight  
Amazon EMR

**What are some issues that  
you might encounter when  
dealing with extremely  
large datasets?**



# Extremely Large Datasets

---

Issues you might encounter when dealing with extremely large datasets:



Need a place to store massive amounts of data



Need a way to access data quickly



Need backups for hardware failure



Need ways to analyze data quickly

# Hadoop Overview

# What is Hadoop?

---



The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.



*—Hadoop website*

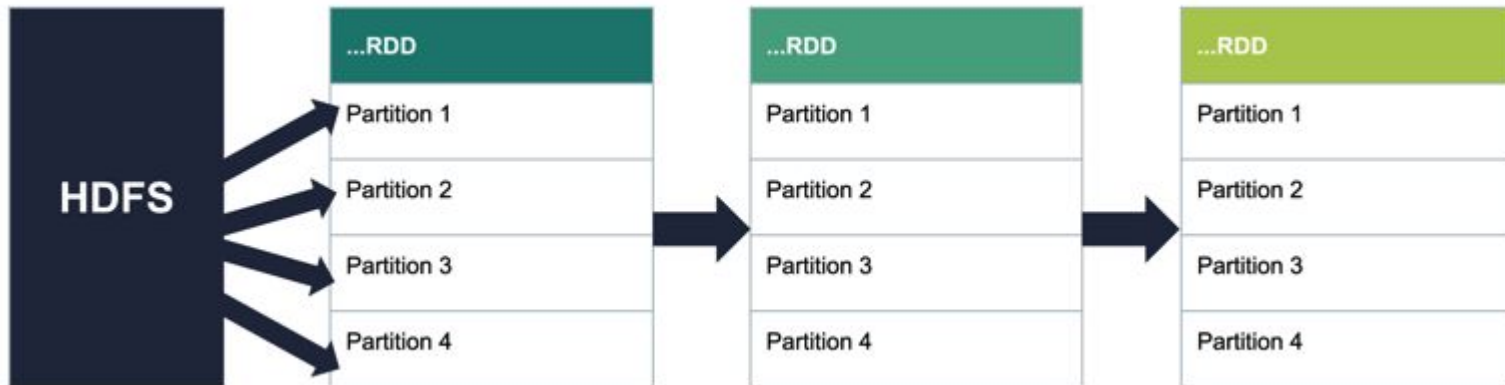
**Hadoop was named after a stuffed yellow elephant belonging to the son of Doug Cutting, the Hadoop creator.**



# Hadoop Distributed File System (HDFS)

---

HDFS is a file system that is used to store data across server clusters, and is **scalable**, **fault-tolerant**, and **distributed**.



# Fundamentals

---

01

MapReduce

02

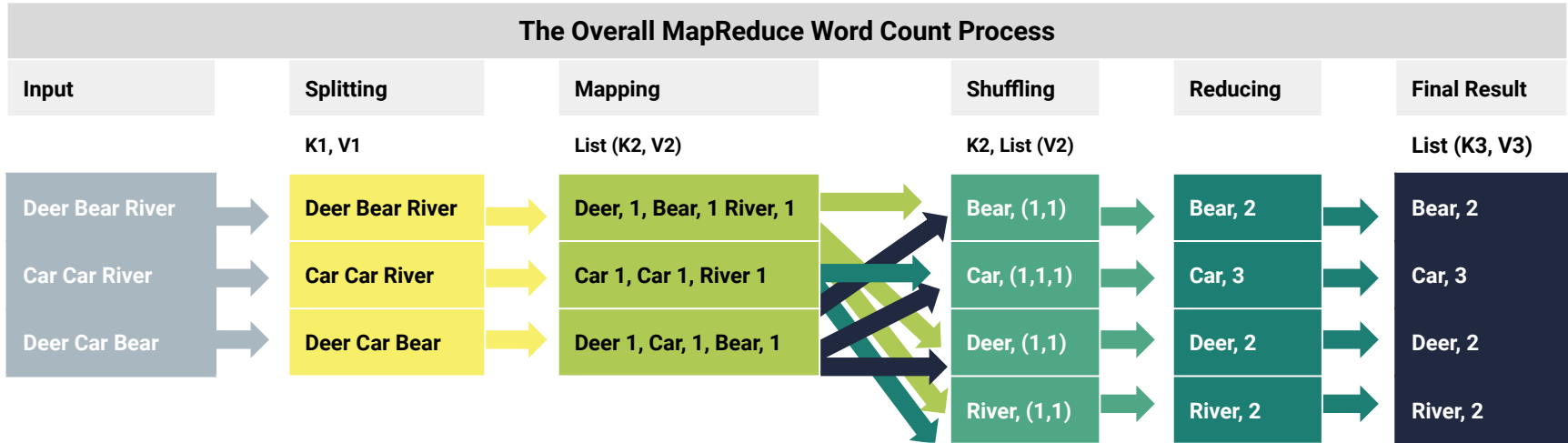
Hadoop Distributed  
File System (HDFS)

03

Yet Another  
Resource  
Negotiator (YARN)

# MapReduce

# Classic Word Count Example



# MapReduce History

---

It was originally the product of research done by Google.



Designed to solve a single problem: how to index all the information on the Internet

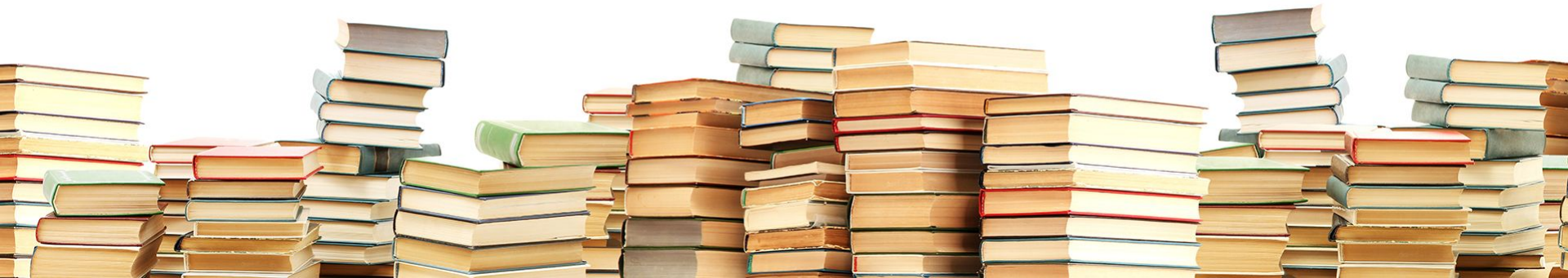


## Example: Counting the Number of Books in a Library

---

**Map:** You count this half of the library, and I'll count the other.

**Reduce:** We get together and add up our counts.



# The Map Part of MapReduce



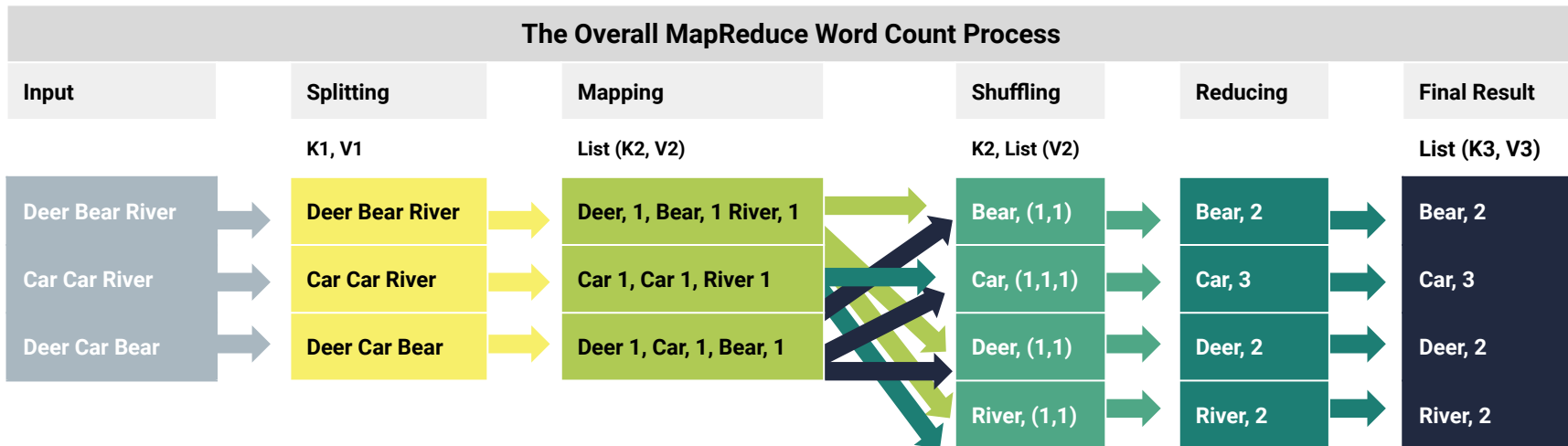
Execute the `Map()` function on data.



Execute on each node.

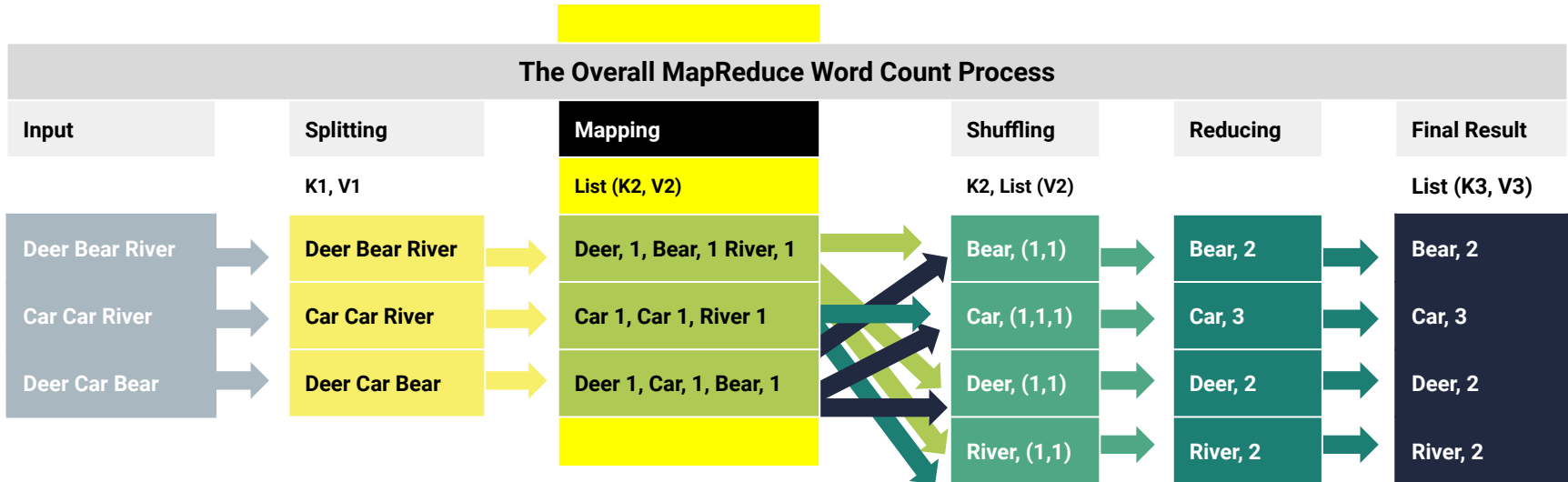


Output `<key, value>` pairs on each node.



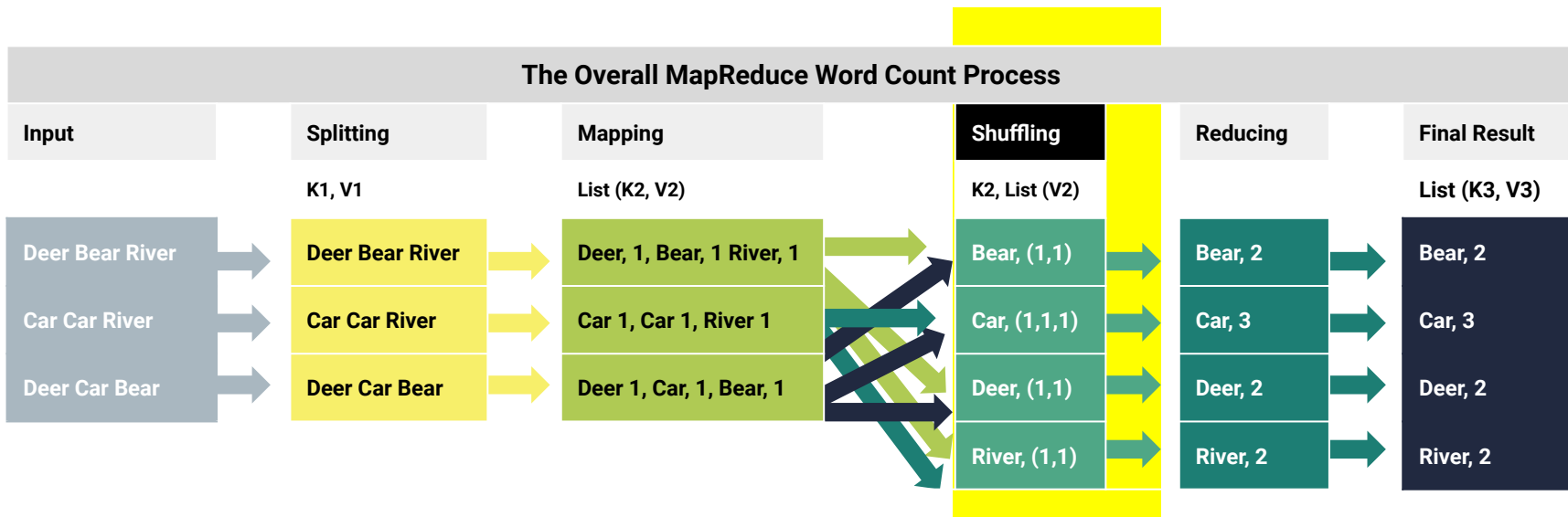
# The Map in MapReduce

The mapping step takes a small piece of the input and maps the data to key-value pairs. A common example is sending a line of text to a mapper function, and the mapper generates a key-value pair for each word. The values will be added up later in the reducing step.



# The Shuffle in MapReduce

The shuffle step groups the keys together. Each value found for a key is appended to the list of values.



# What Is a Job?

---



A job is defined by a class that inherits from `MRJob`.



This class contains methods that define the steps of your job.



Can translate the job and run it locally or on a Hadoop cluster.



Note: the shuffle step is handled behind the scenes

```
from mrjob.job import MRJob  
  
class WordCounter(MRJob):
```

# The Reduce Part of MapReduce

---



Execute the `Reduce()` function on data.



Execute on some node.



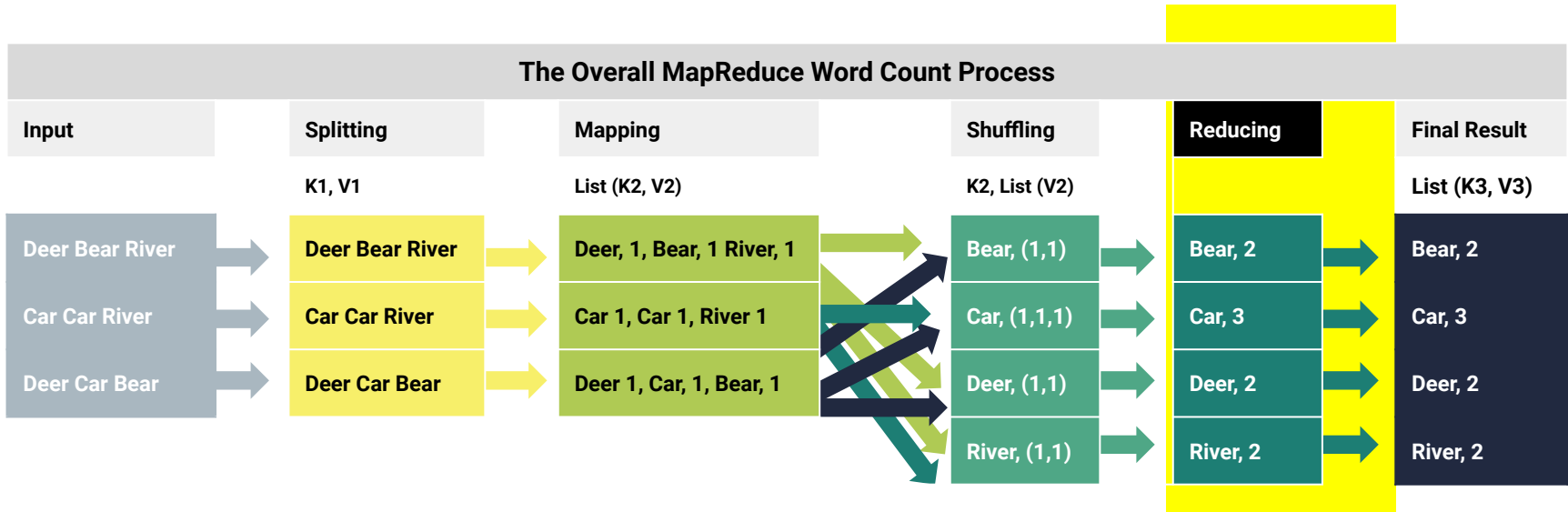
Aggregate sets of `<key, value>` pairs on some nodes.



Output a combined list.

# The Reduce in MapReduce

The reducing step reduces the list of the values for a key to a single value. In this example, the values are added to get the count of each word.



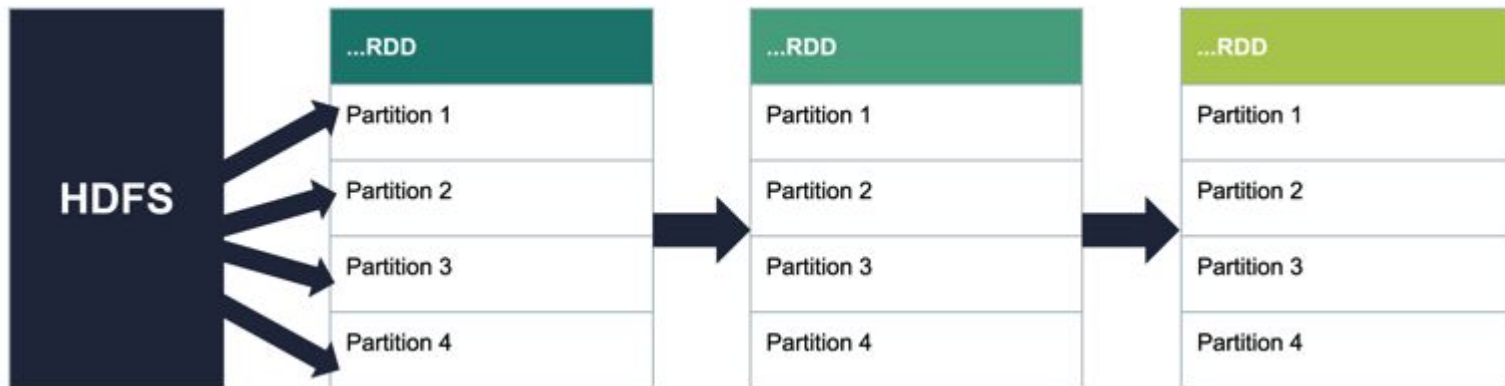
# Hadoop Distributed File System



# Hadoop Distributed File System (HDFS)

---

- Partitioned large data sets
- Store files across a network of machines
- Evolved from the GFS, or Google File System



# Hadoop Distributed File System (HDFS)

---

## Pros



- 1 Handles terabytes of data
- 2 Write once, read many times
- 3 Uses commodity hardware

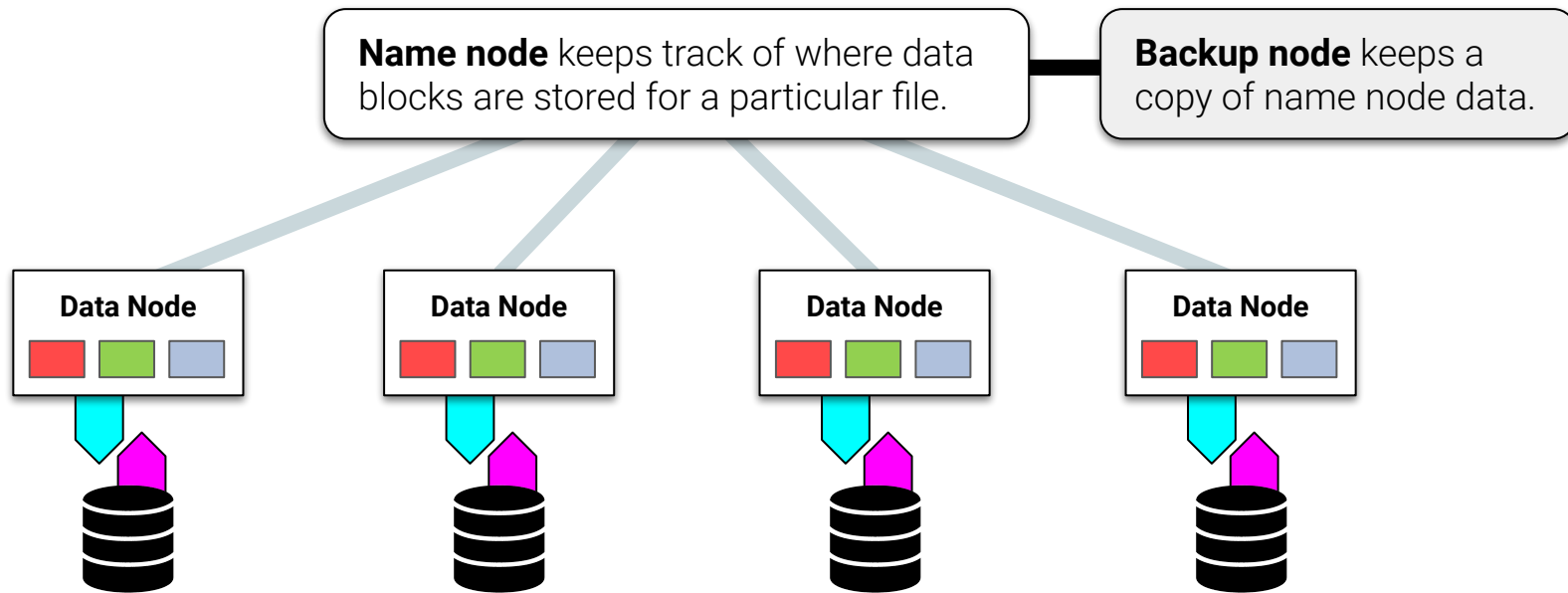


## Cons

- 1 Not good to low latency access
- 2 Bad for lots of small files
- 3 Not for multiple writers

# Node

The client will request data through the name node. The name node will locate that data across its data node, and then return data back to the client.



Data nodes store the data and return it on request.

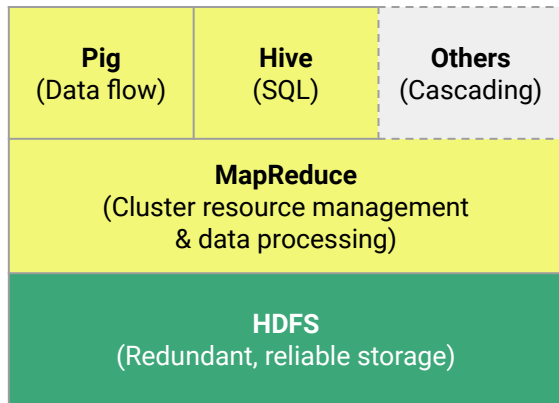
# **Yet Another Resource Manager (YARN)**

# YARN Overview

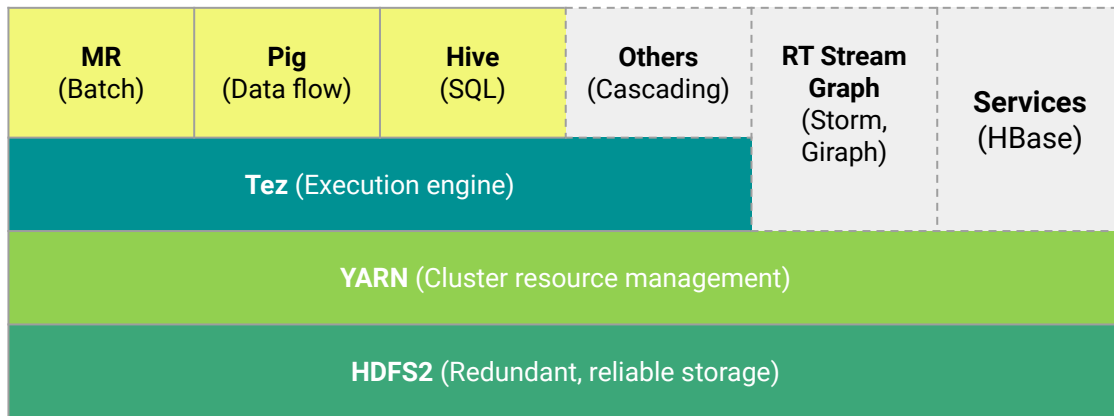
Apache Hadoop YARN (Yet Another Resource Negotiator) is a cluster management technology.

YARN is one of the key features in the second-generation Hadoop 2 version of the Apache Software Foundation's open-source distributed processing framework.

## HADOOP 1.0



## HADOOP 2.0





mrjob

# What Is MRjob?

---



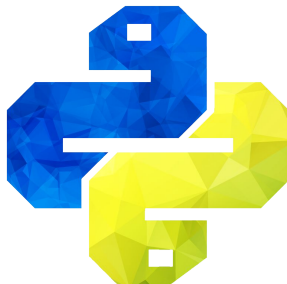
mrjob is a Python library built by Yelp.



It lets you write MapReduce jobs in Python and run them on different platforms.



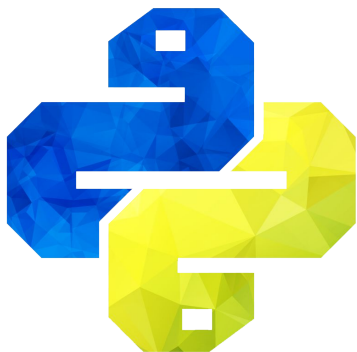
It also lets you stay in Python while making it easier to process and analyze big data.



# What Will We Use mrjob For?

---

Writing MapReduce  
jobs in pure Python



Running jobs on our  
Hadoop cluster





# **mrjob fundamentals**

# Basic Example

---

```
from MRJob.job import MRJob

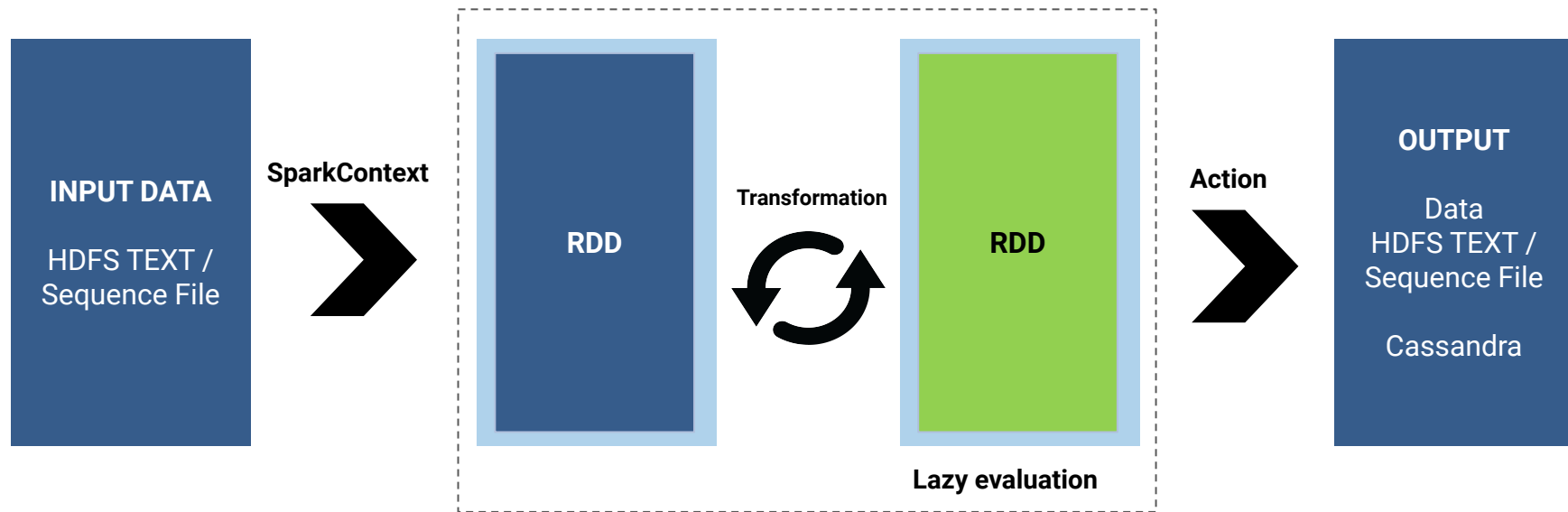
class WordCounter(MRJob):
    def mapper(self, _, line):
        """Maps every word in a line."""
        for word in line.split():
            yield word, 1

    def reducer(self, word, values):
        """Reduces the list of values."""
        yield word, sum(values)
```

# Spark (PySpark)

# What Is Spark?

Apache Spark is a unified analytics engine for large-scale data processing. It lets you write applications in Java, Scala, Python, R, and SQL and runs on Hadoop, stand-alone, or in the cloud (and many other platforms). Spark can be 100 times faster than Hadoop.

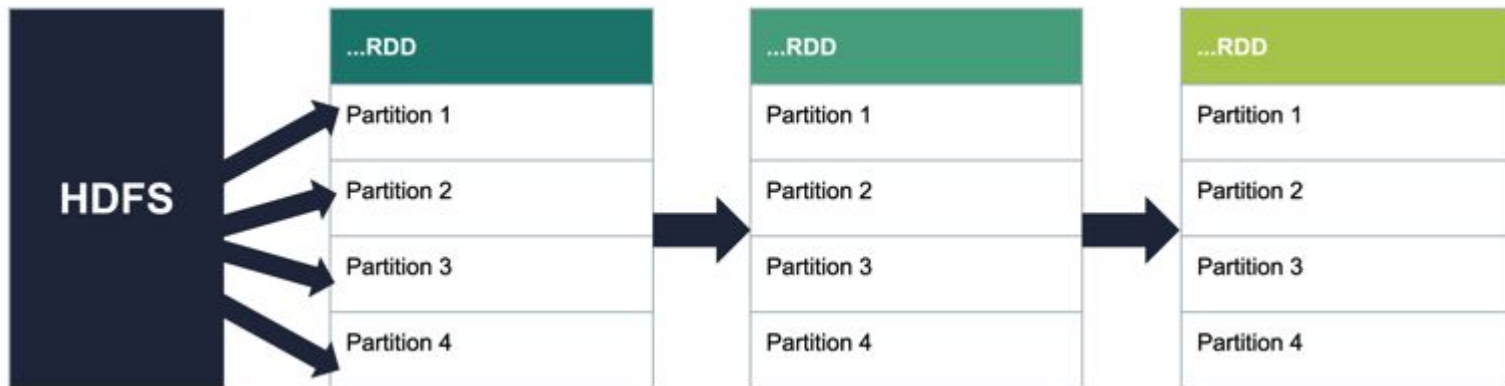


# Resilient Distributed Dataset

---

A resilient distributed dataset (RDD) is the basic abstraction in Spark.

It represents an immutable, partitioned collection of elements that can be operated on in parallel.



# Spark DataFrames

---



Spark DataFrames are similar to Pandas.



They hold data in a column and row format.



Each column represents a variable or feature.



Each row represents one data point.



Unlike Pandas, Spark DataFrames can scale to handle petabytes of data on clusters of servers or in the cloud.