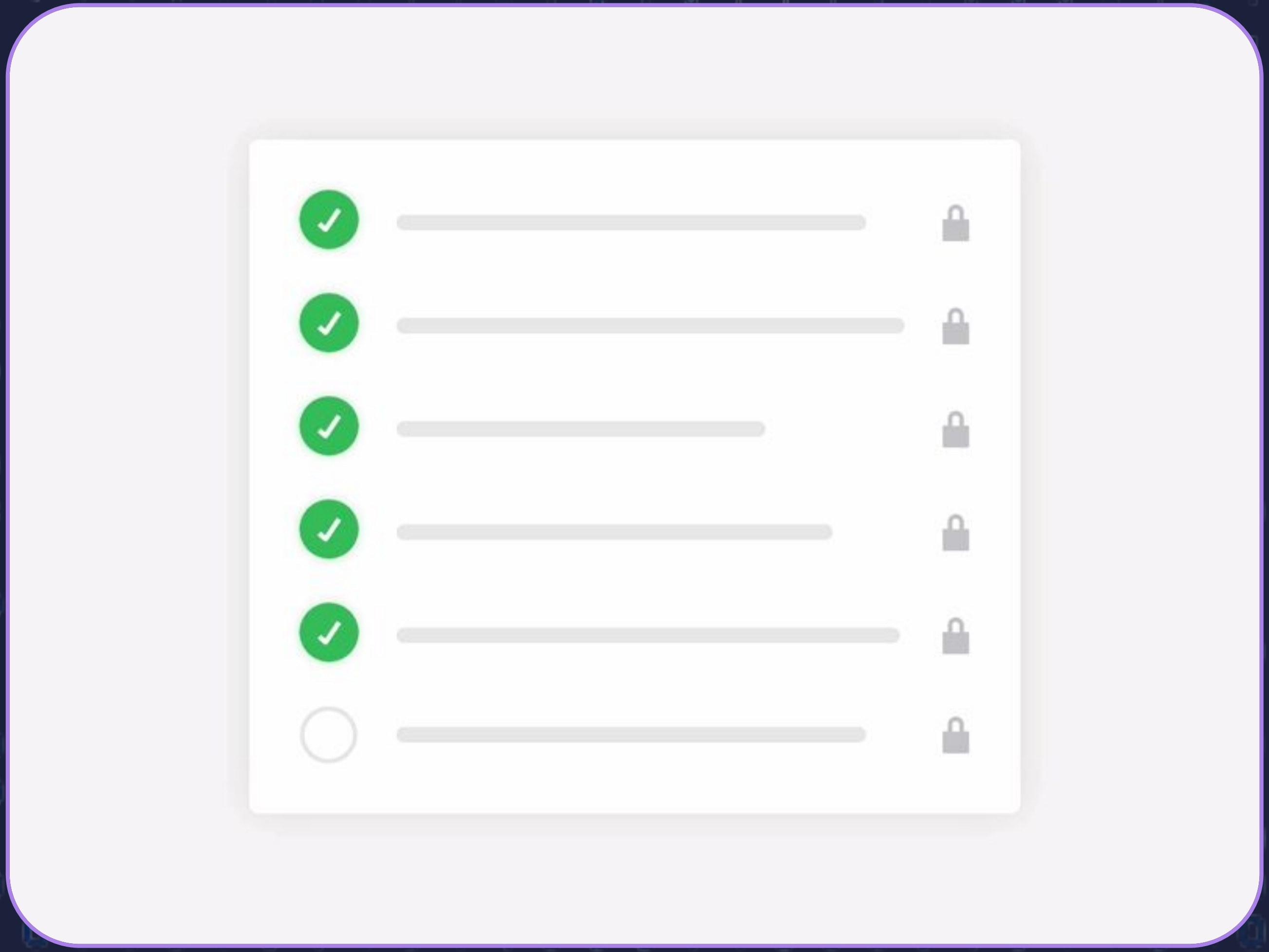
SKILLS

Spark

Apache Spark is an open-source, distributed processing system used for big data workloads

Lecture CheckList

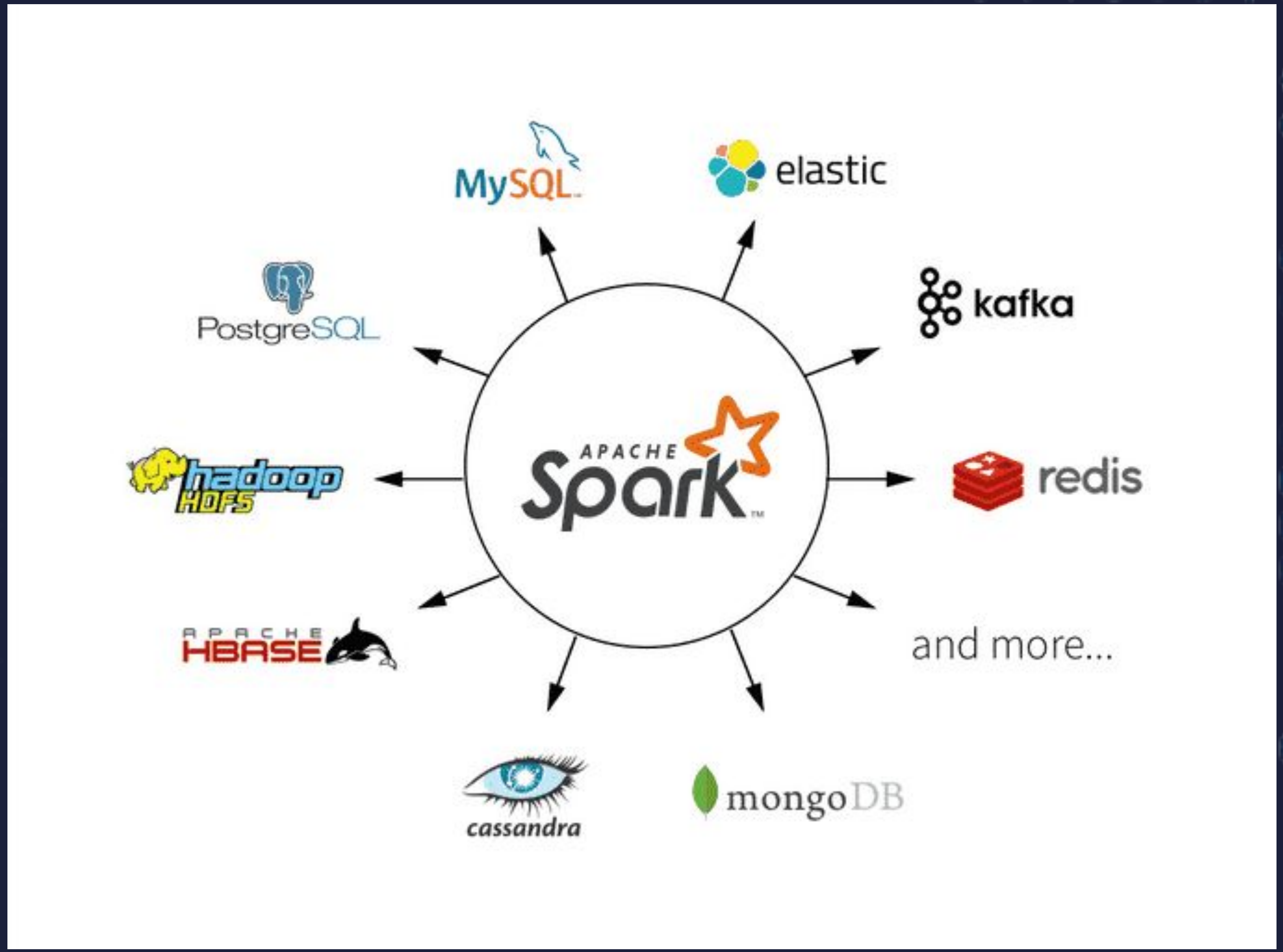
1. Introduction spark
2. Spark Installation
3. Spark Rdd
4. Spark Architecture
5. Spark Ml Lib
6. Spark Nlp
7. Spark Linear Regression
8. Spark Logistic Regression
9. Spark Decision Tree
10. Spark Naive Bayes



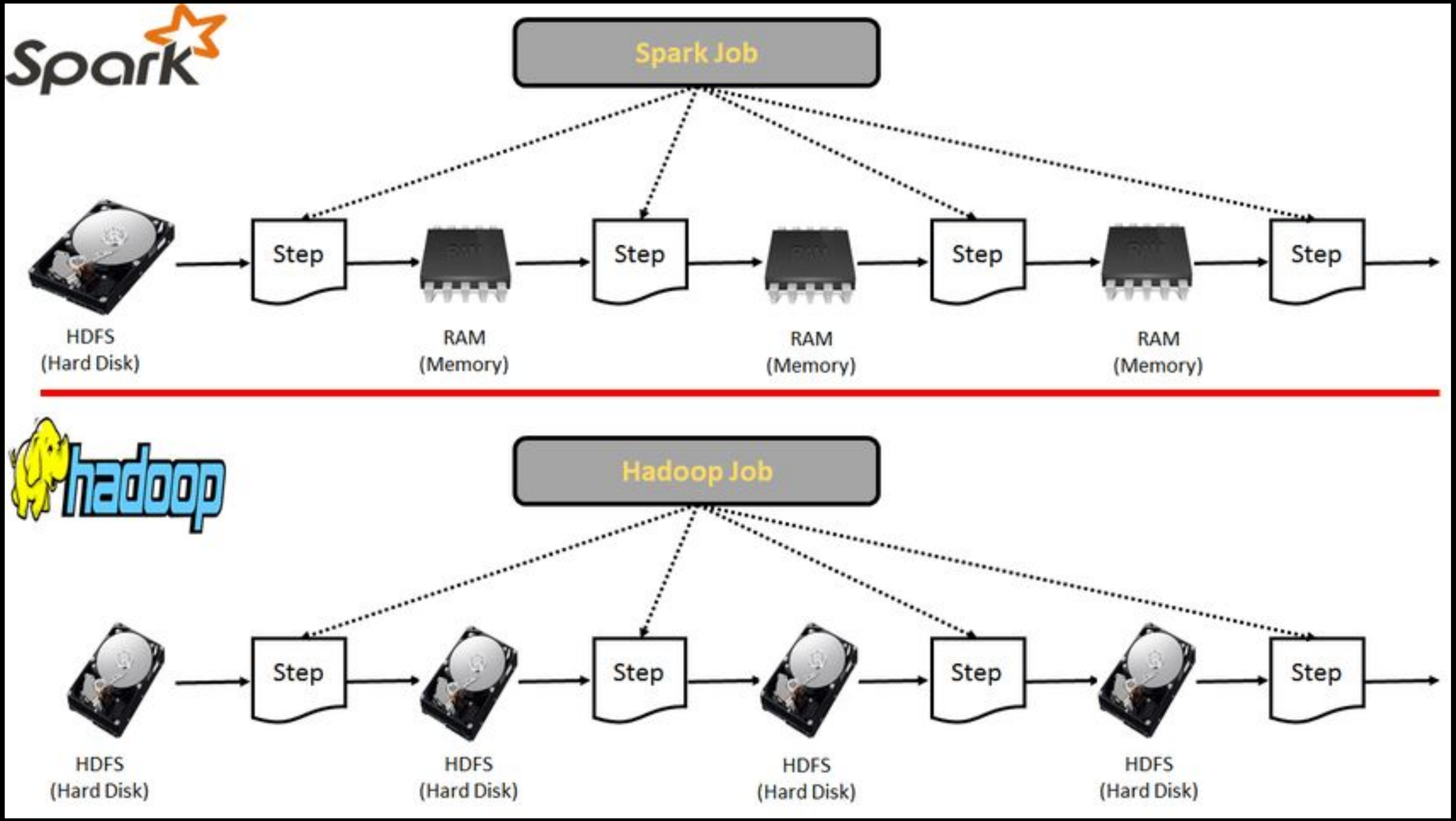
Spark Overview

Apache Spark is an open-source, distributed processing system used for big data workloads

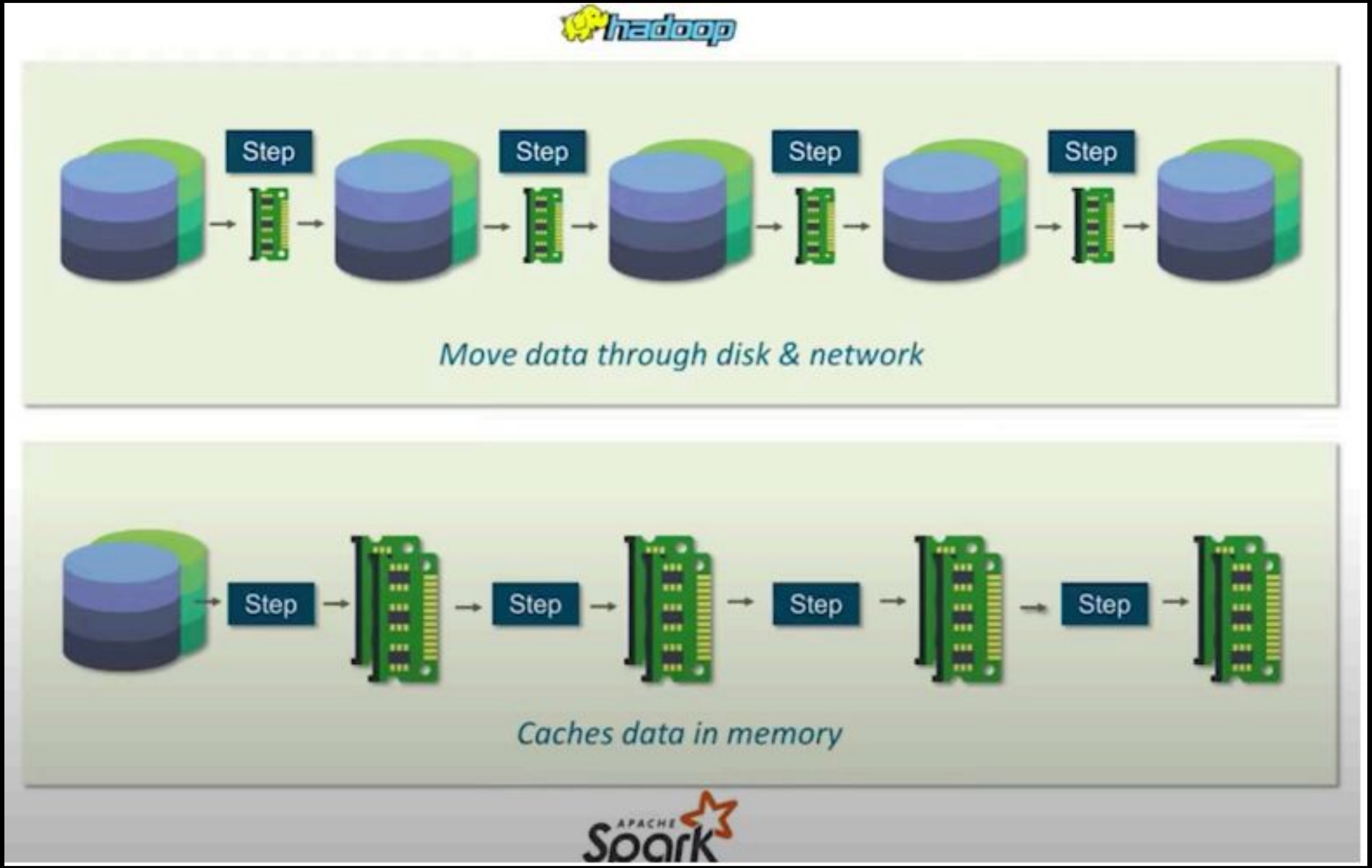
Apache Spark is an open-source, distributed processing system used for big data workloads. It utilizes in-memory caching and optimized query execution for fast queries against data of any size. Simply put, Spark is a fast and general engine for large-scale data processing.



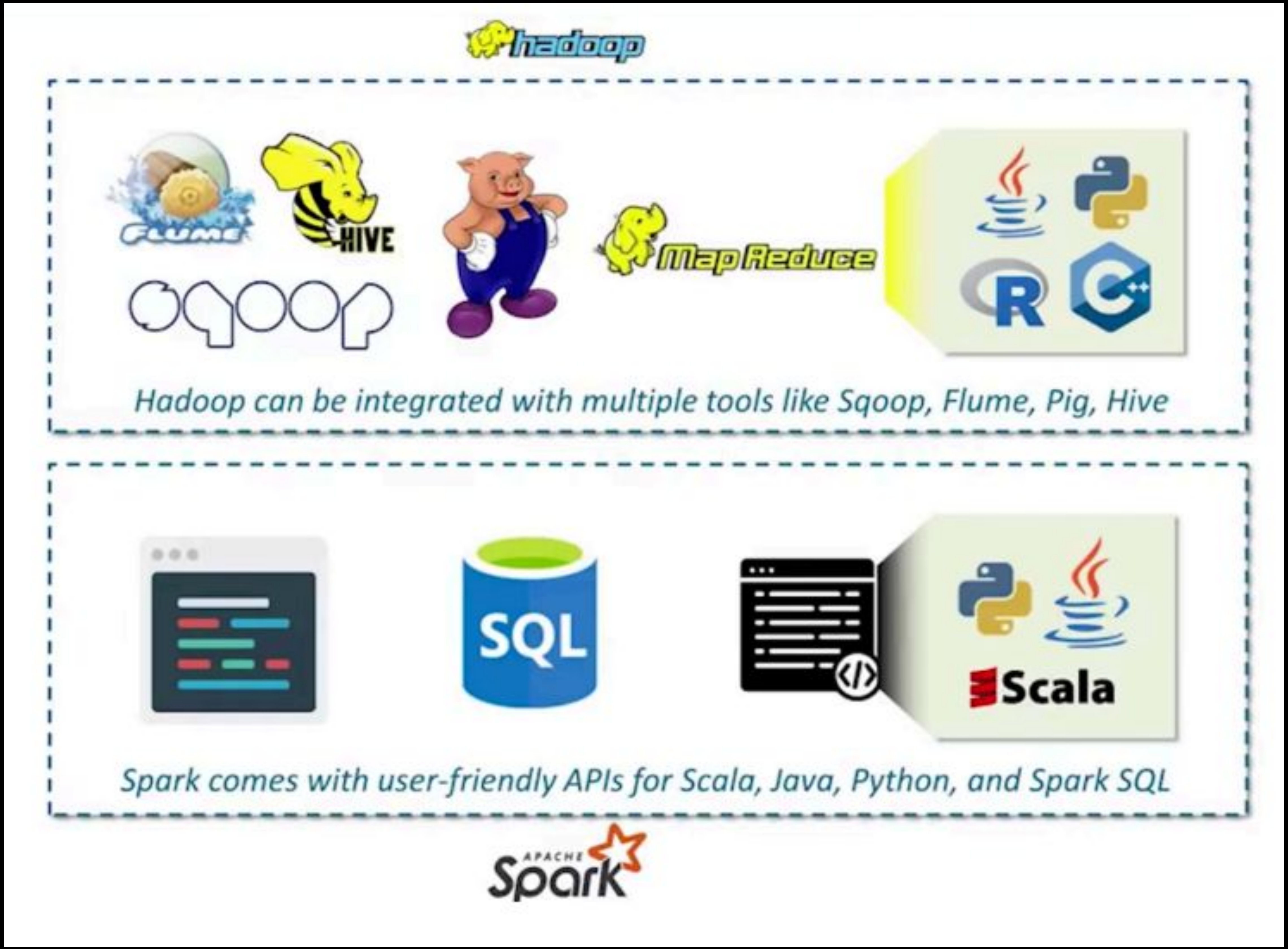
Difference Between Hadoop and Spark



Difference Between Hadoop and Spark



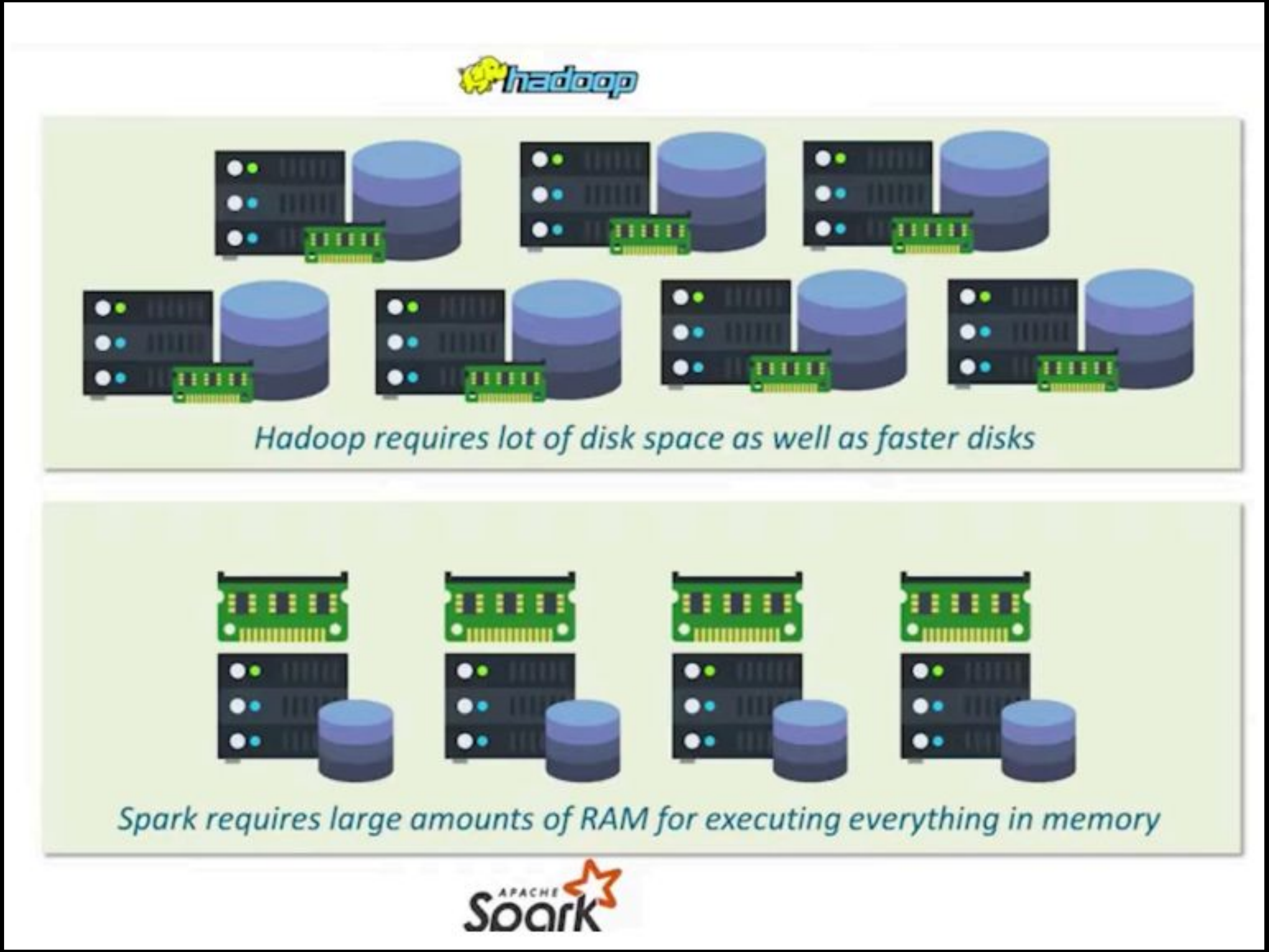
Difference Between Hadoop and Spark



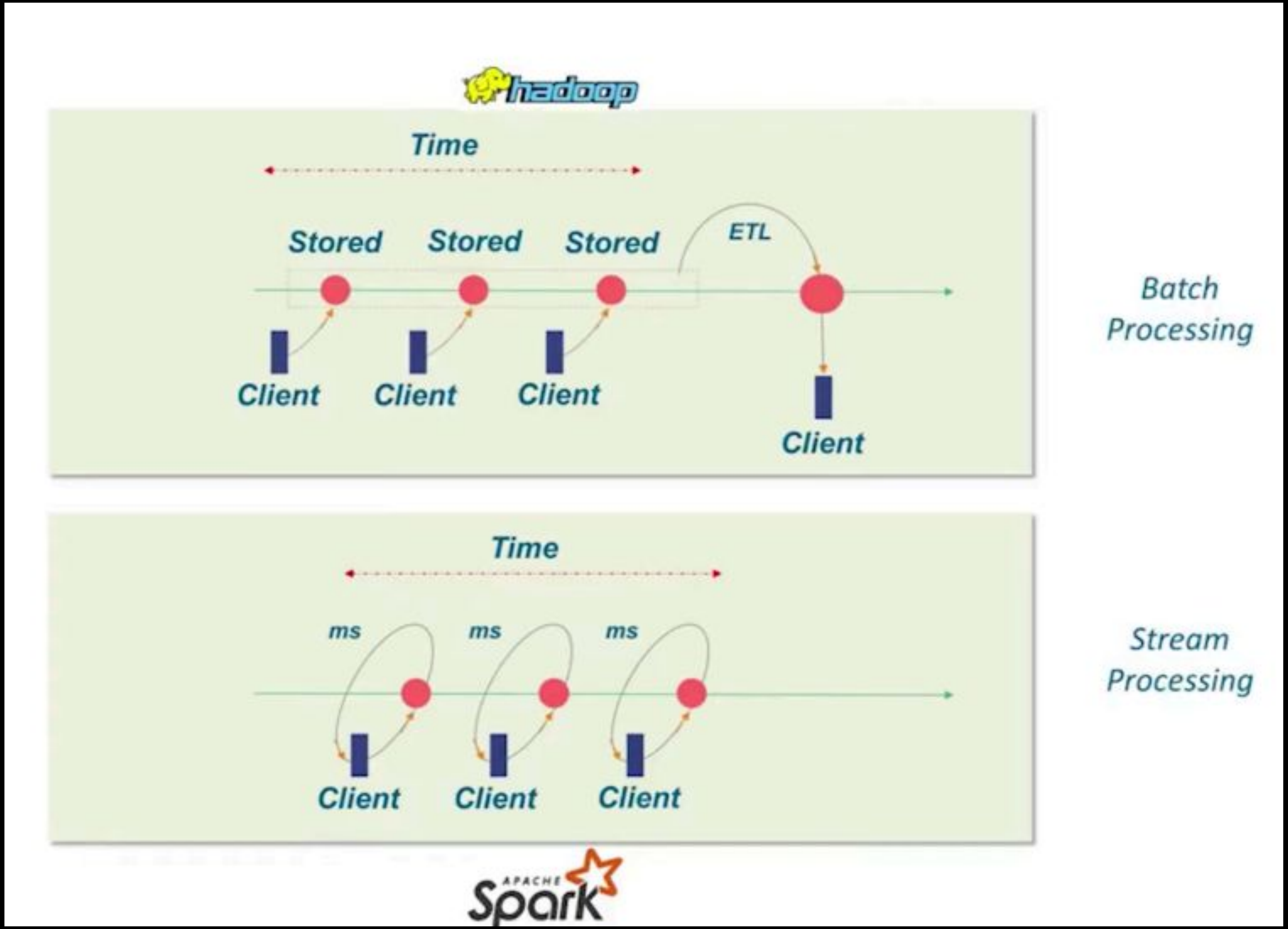
Hadoop can be integrated with multiple tools like Sqoop, Flume, Pig, Hive

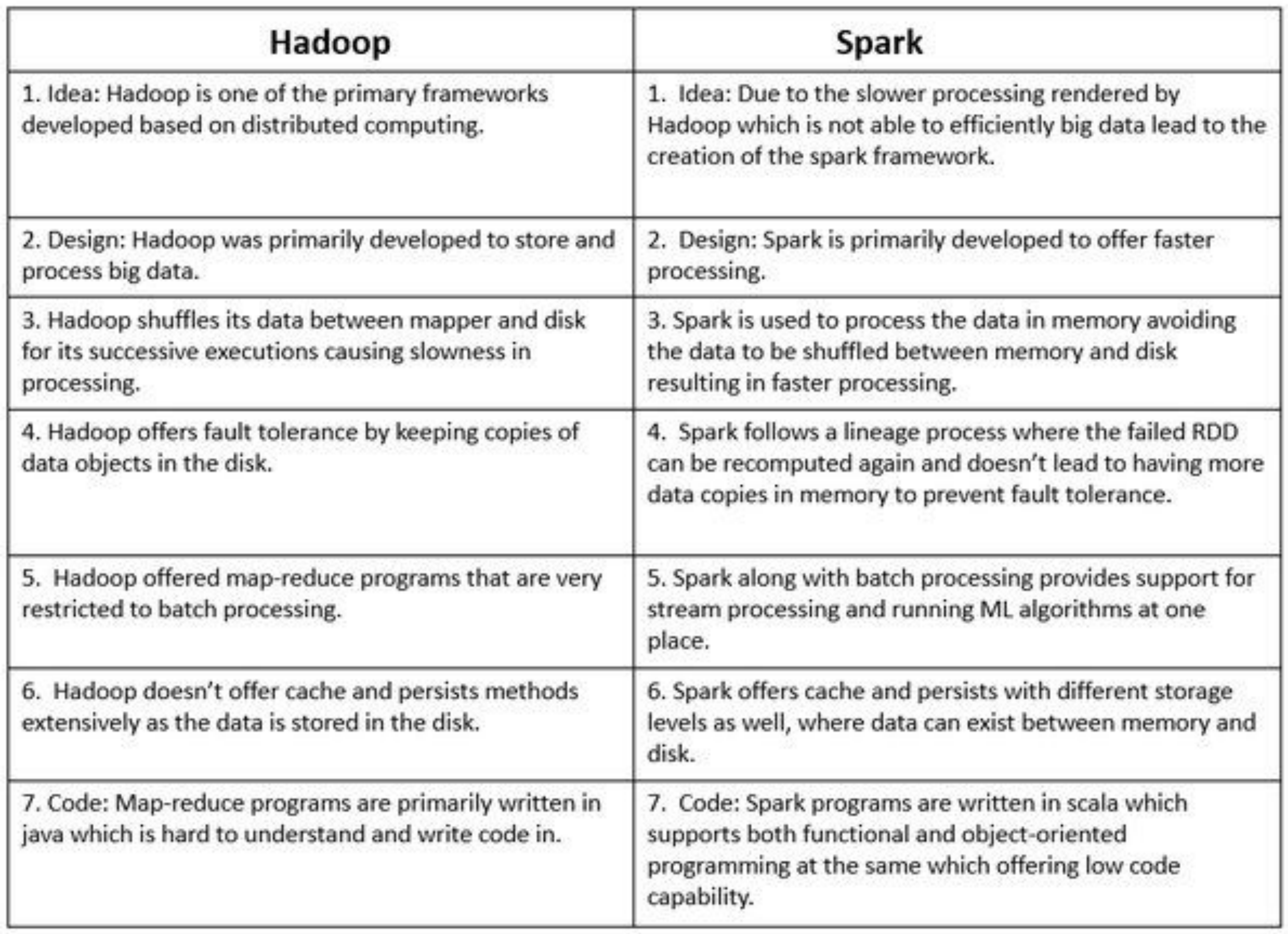
Spark comes with user-friendly APIs for Scala, Java, Python, and Spark SQL

Difference Between Hadoop and Spark

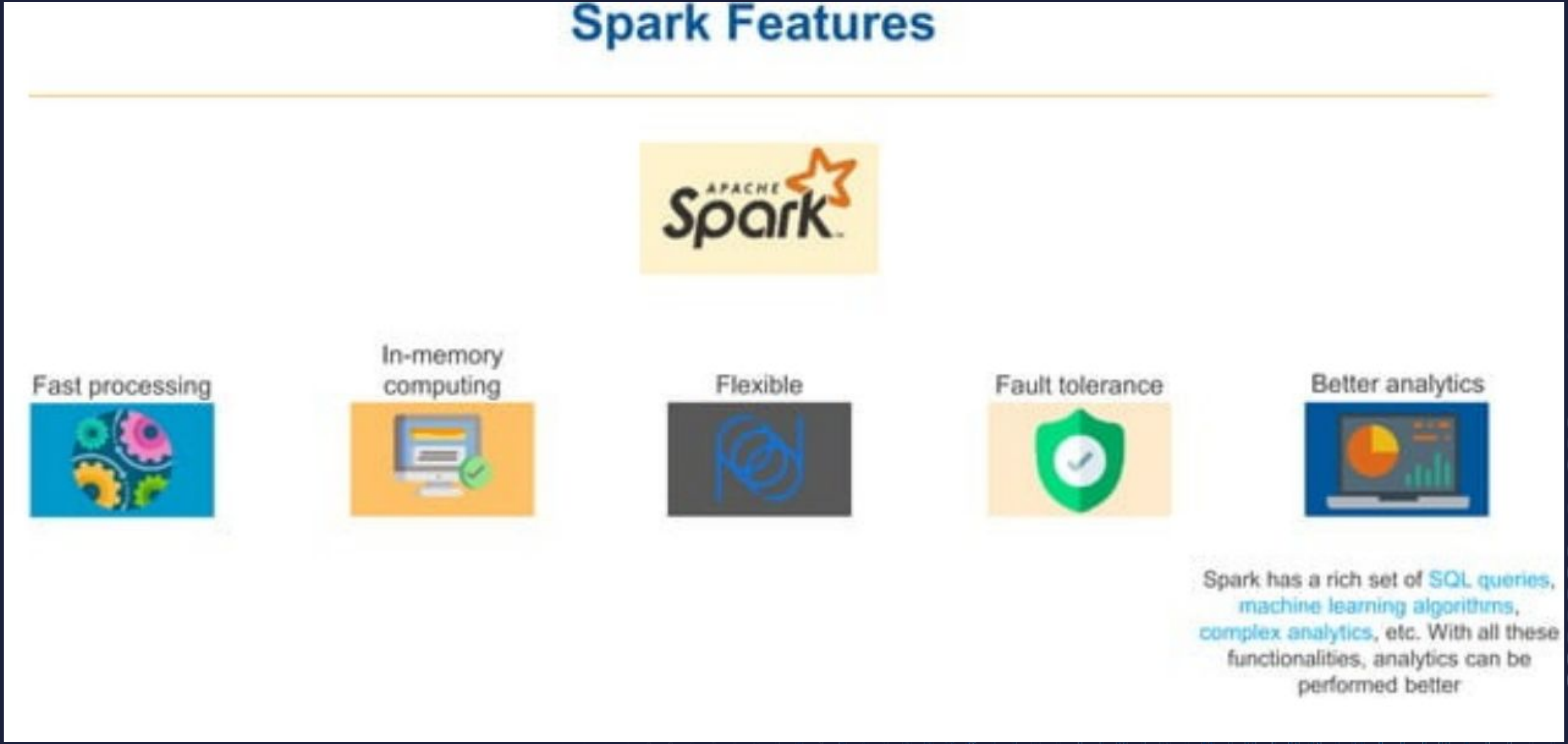

hadoop
Hadoop requires lot of disk space as well as faster disks
Spark requires large amounts of RAM for executing everything in memory
APACHE Spark

Difference Between Hadoop and Spark





Hadoop	Spark
1. Idea: Hadoop is one of the primary frameworks developed based on distributed computing.	1. Idea: Due to the slower processing rendered by Hadoop which is not able to efficiently big data lead to the creation of the spark framework.
2. Design: Hadoop was primarily developed to store and process big data.	2. Design: Spark is primarily developed to offer faster processing.
3. Hadoop shuffles its data between mapper and disk for its successive executions causing slowness in processing.	3. Spark is used to process the data in memory avoiding the data to be shuffled between memory and disk resulting in faster processing.
4. Hadoop offers fault tolerance by keeping copies of data objects in the disk.	4. Spark follows a lineage process where the failed RDD can be recomputed again and doesn't lead to having more data copies in memory to prevent fault tolerance.
5. Hadoop offered map-reduce programs that are very restricted to batch processing.	5. Spark along with batch processing provides support for stream processing and running ML algorithms at one place.
6. Hadoop doesn't offer cache and persists methods extensively as the data is stored in the disk.	6. Spark offers cache and persists with different storage levels as well, where data can exist between memory and disk.
7. Code: Map-reduce programs are primarily written in java which is hard to understand and write code in.	7. Code: Spark programs are written in scala which supports both functional and object-oriented programming at the same which offering low code capability.



Spark Features

Fast processing

In-memory computing

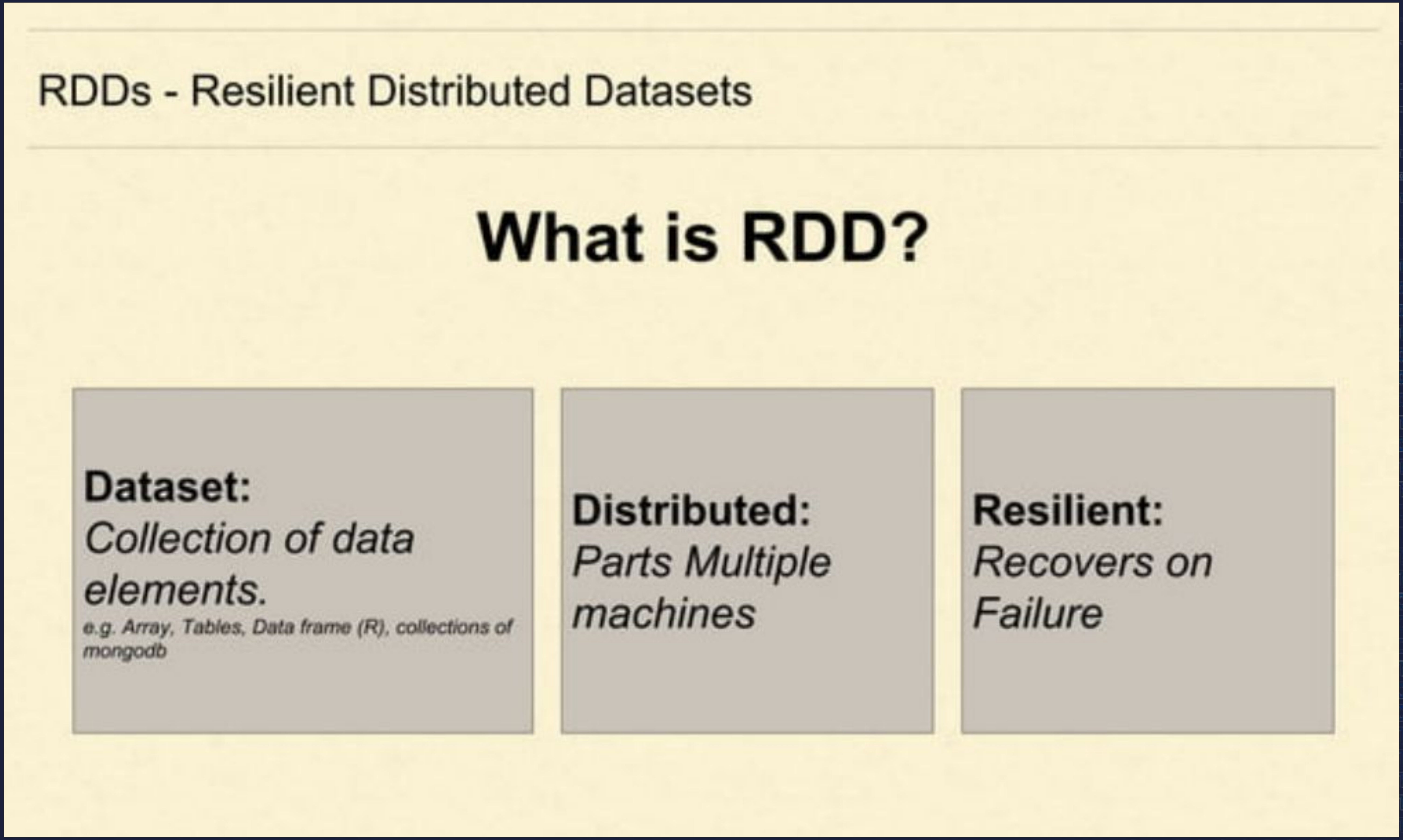
Flexible

Fault tolerance

Better analytics

Spark has a rich set of SQL queries, machine learning algorithms, complex analytics, etc. With all these functionalities, analytics can be performed better

Resilient distributed datasets - RDDs



RDDs - Resilient Distributed Datasets

What is RDD?

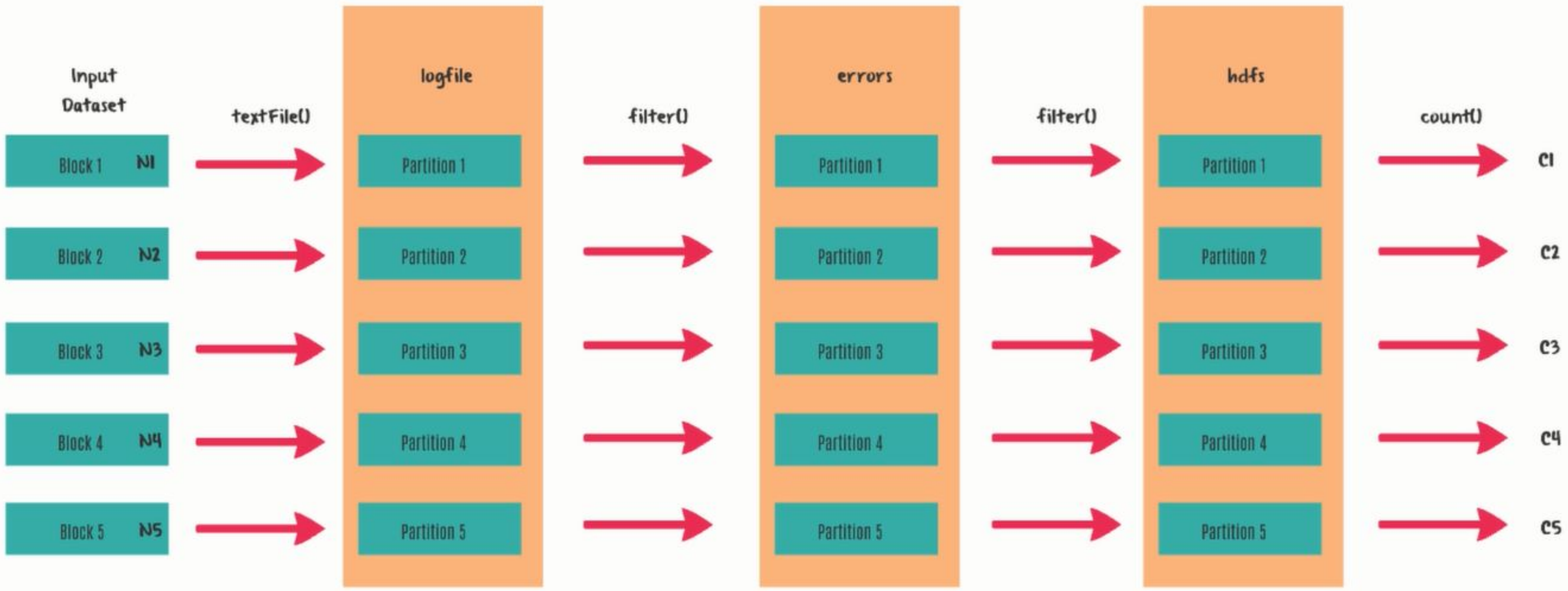
Dataset:
Collection of data elements.
e.g. Array, Tables, Data frame (R), collections of mongodb

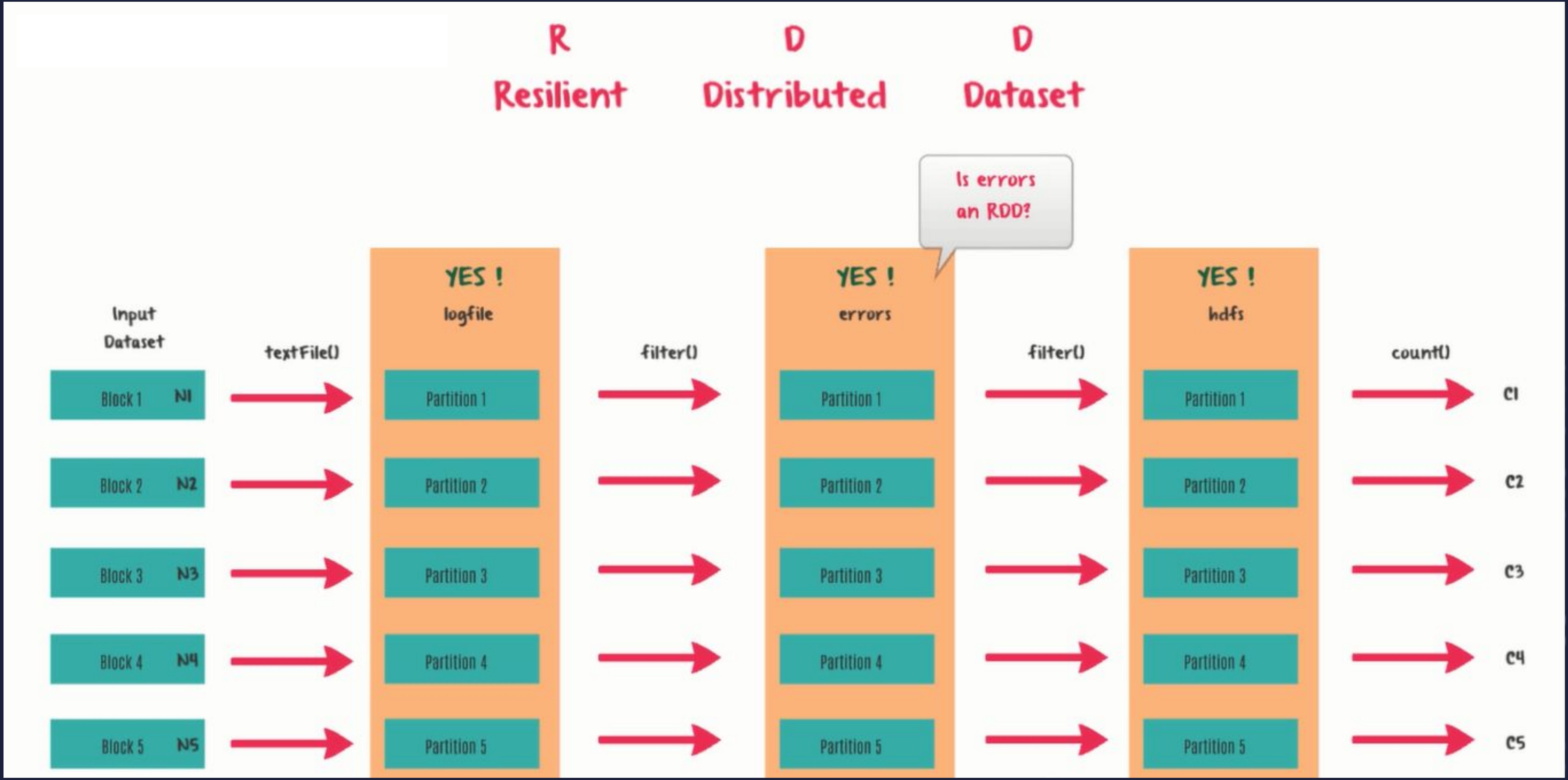
Distributed:
Parts Multiple machines

Resilient:
Recovers on Failure

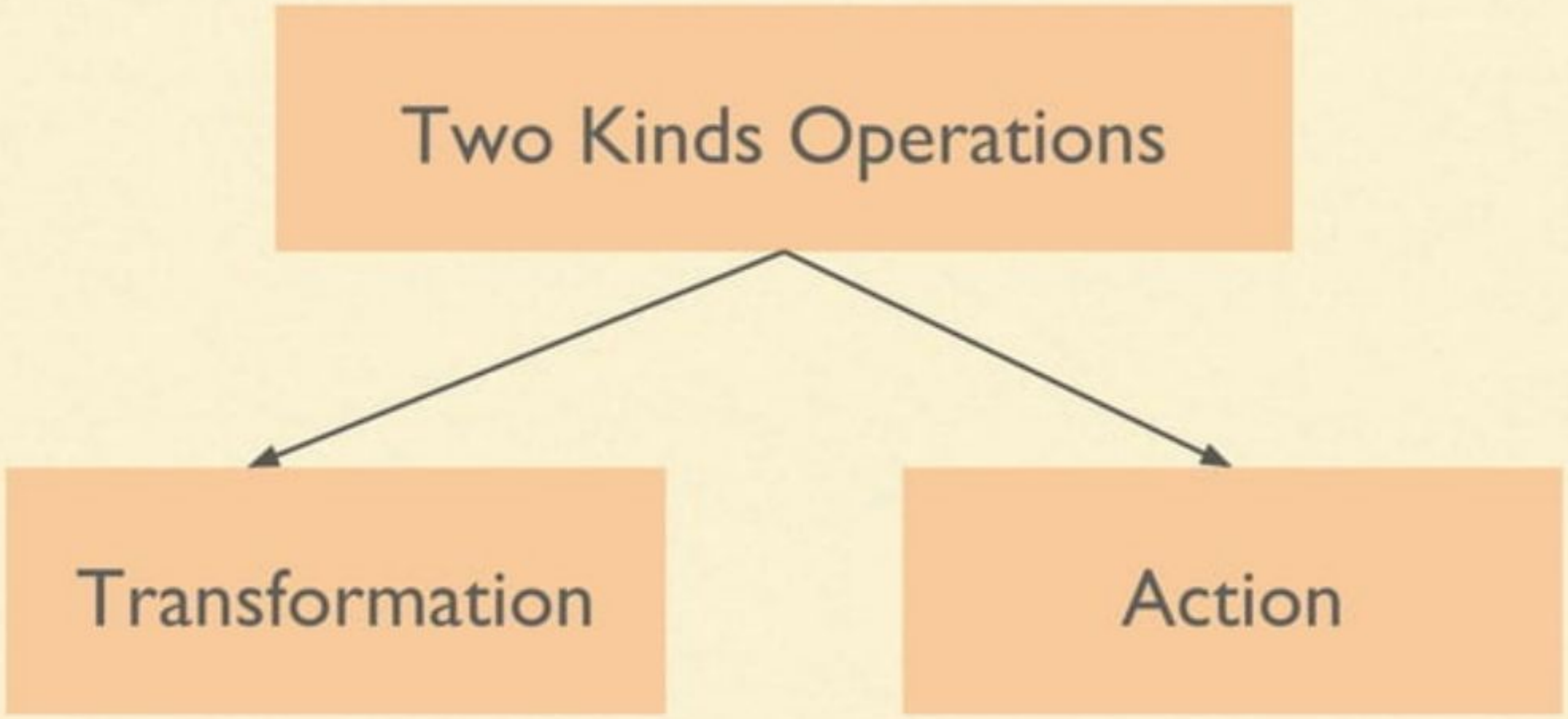
What is RDD?

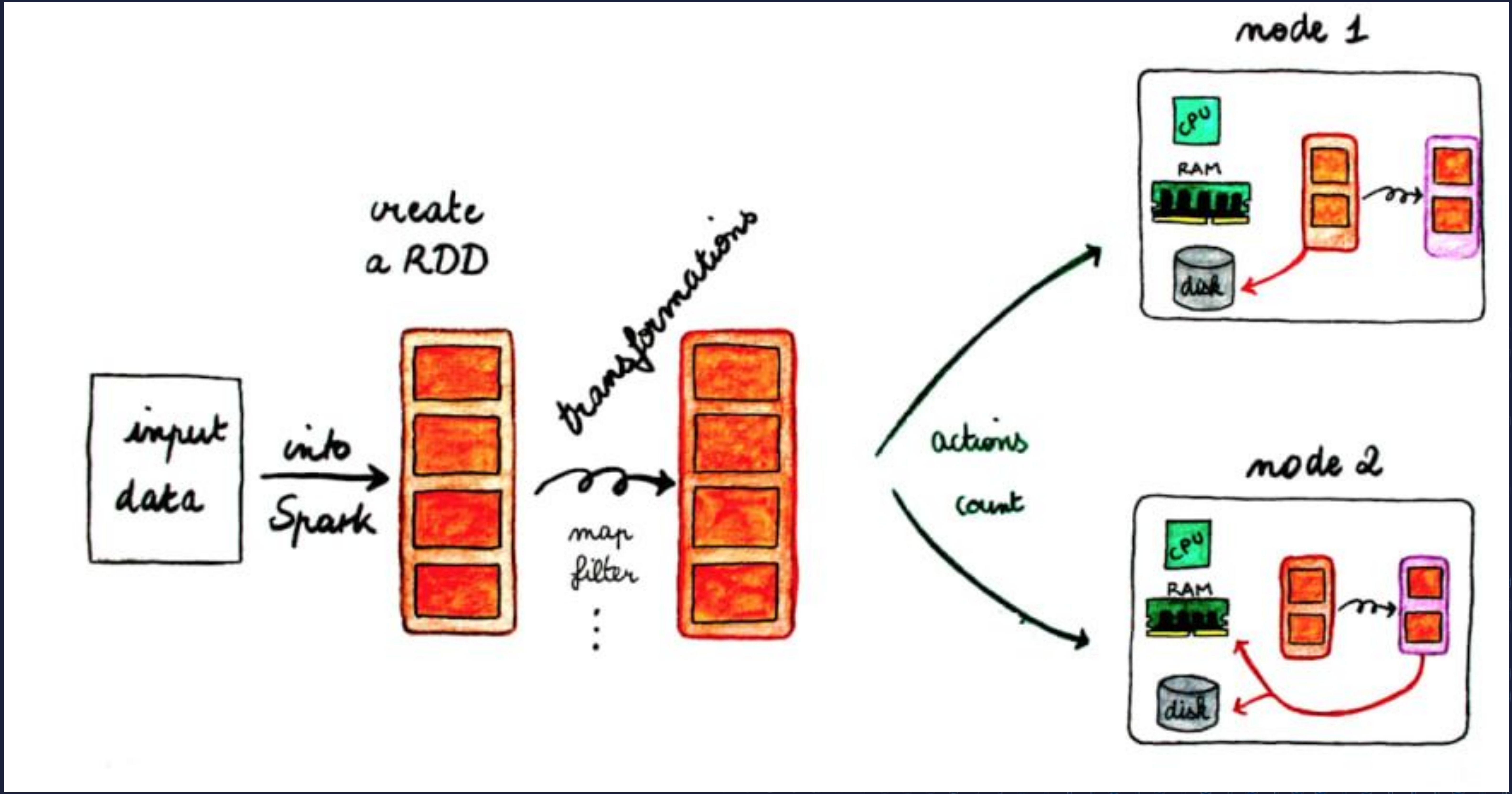
- ❑ Spark tightly controls what you do with the dataset. You cannot work with data in Spark without RDD.
- ❑ To refer your dataset in Spark you will use the functions provided by Spark and those functions will create a RDD behind the scenes.
- ❑ Can you join two datasets in spark? Of course you can using the join function provided by Spark and internally the join function will result in a RDD.
- ❑ Can you do aggregation in your dataset? Of course you can, Spark provides functions for that too and spark will create a RDD behind the scenes any time you attempt to transform the data.
- ❑ In short, Spark controls any operations that you do with your dataset, to perform any meaningful operation with your data you need to use functions provided by Spark and it will result in RDD.
- ❑ This way Spark can keep track of everything that you are trying to do with the dataset and this is referred to as lineage and this helps Spark deal with tolerating failures effectively, which by the way is a very big deal in in-memory computing.


Input Dataset
textFile()
logfile
filter()
errors
filter()
hdfs
count()
Block 1 N1
Block 2 N2
Block 3 N3
Block 4 N4
Block 5 N5
Partition 1
Partition 2
Partition 3
Partition 4
Partition 5
C1
C2
C3
C4
C5


R
Resilient
D
Distributed
D
Dataset
Is errors an RDD?
Input Dataset
Block 1 N1
Block 2 N2
Block 3 N3
Block 4 N4
Block 5 N5
textFile()
YES !
logfile
Partition 1
Partition 2
Partition 3
Partition 4
Partition 5
filter()
YES !
errors
Partition 1
Partition 2
Partition 3
Partition 4
Partition 5
filter()
YES !
hdfs
Partition 1
Partition 2
Partition 3
Partition 4
Partition 5
count()
C1
C2
C3
C4
C5

RDD Operations





input data
into Spark
create a RDD
transformations
map
filter
actions
count
node 1
CPU
RAM
disk
node 2
CPU
RAM
disk

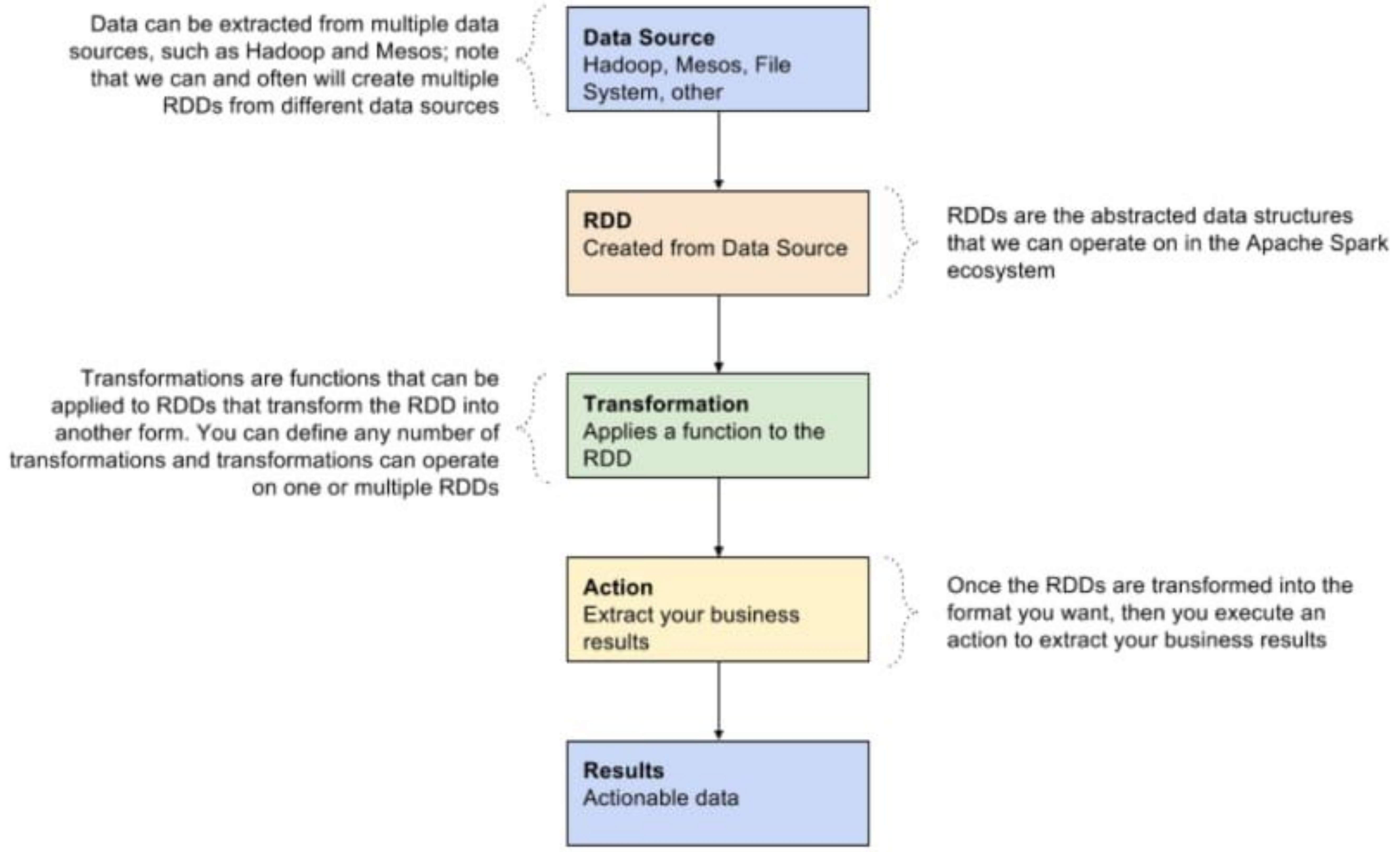

Data can be extracted from multiple data sources, such as Hadoop and Mesos; note that we can and often will create multiple RDDs from different data sources
Data Source
Hadoop, Mesos, File System, other
RDD
Created from Data Source
RDDs are the abstracted data structures that we can operate on in the Apache Spark ecosystem
Transformations are functions that can be applied to RDDs that transform the RDD into another form. You can define any number of transformations and transformations can operate on one or multiple RDDs
Transformation
Applies a function to the RDD
Action
Extract your business results
Once the RDDs are transformed into the format you want, then you execute an action to extract your business results
Results
Actionable data

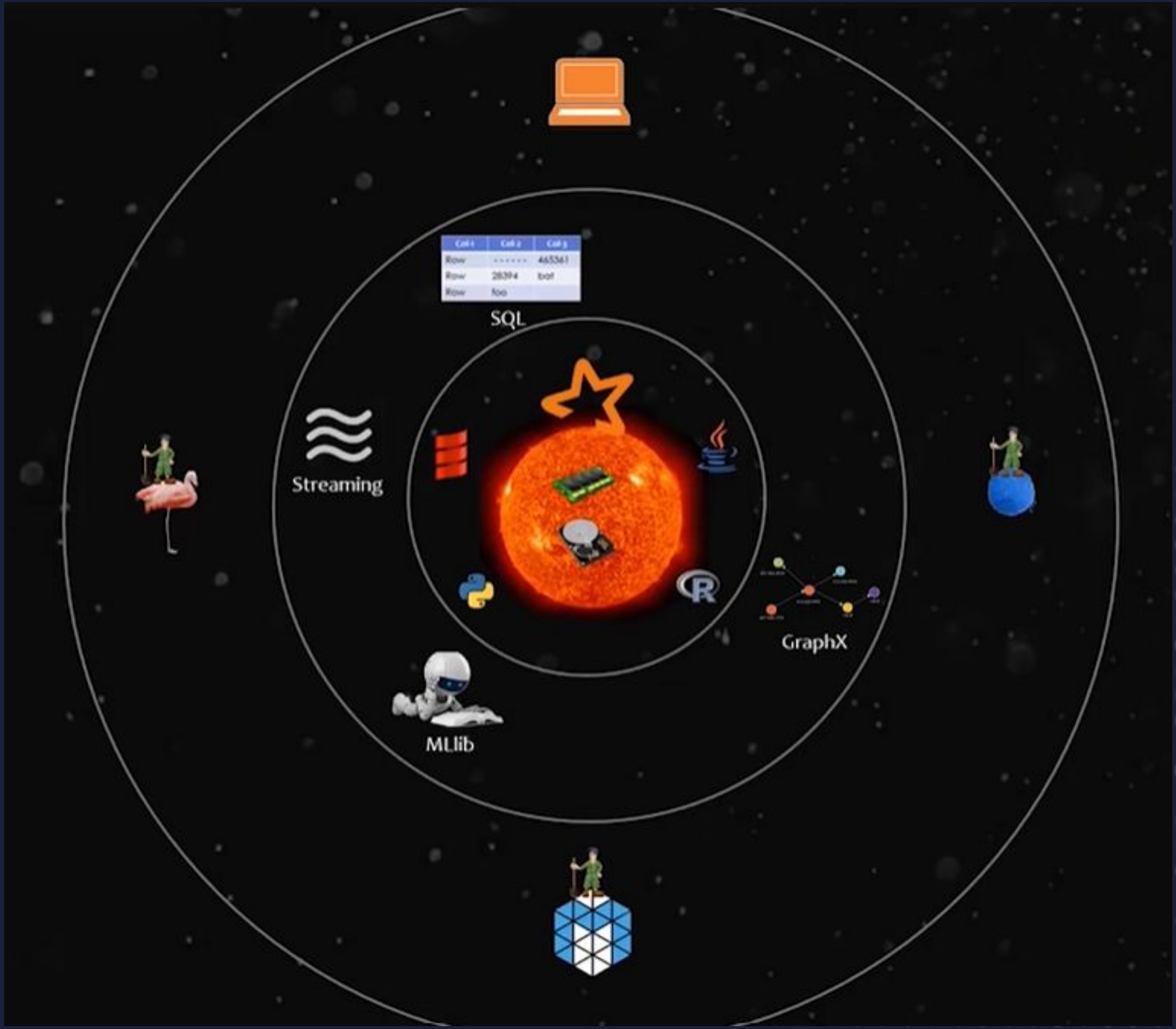
Figure 1. Apache Spark Application Process Flow

Spark Data Frame

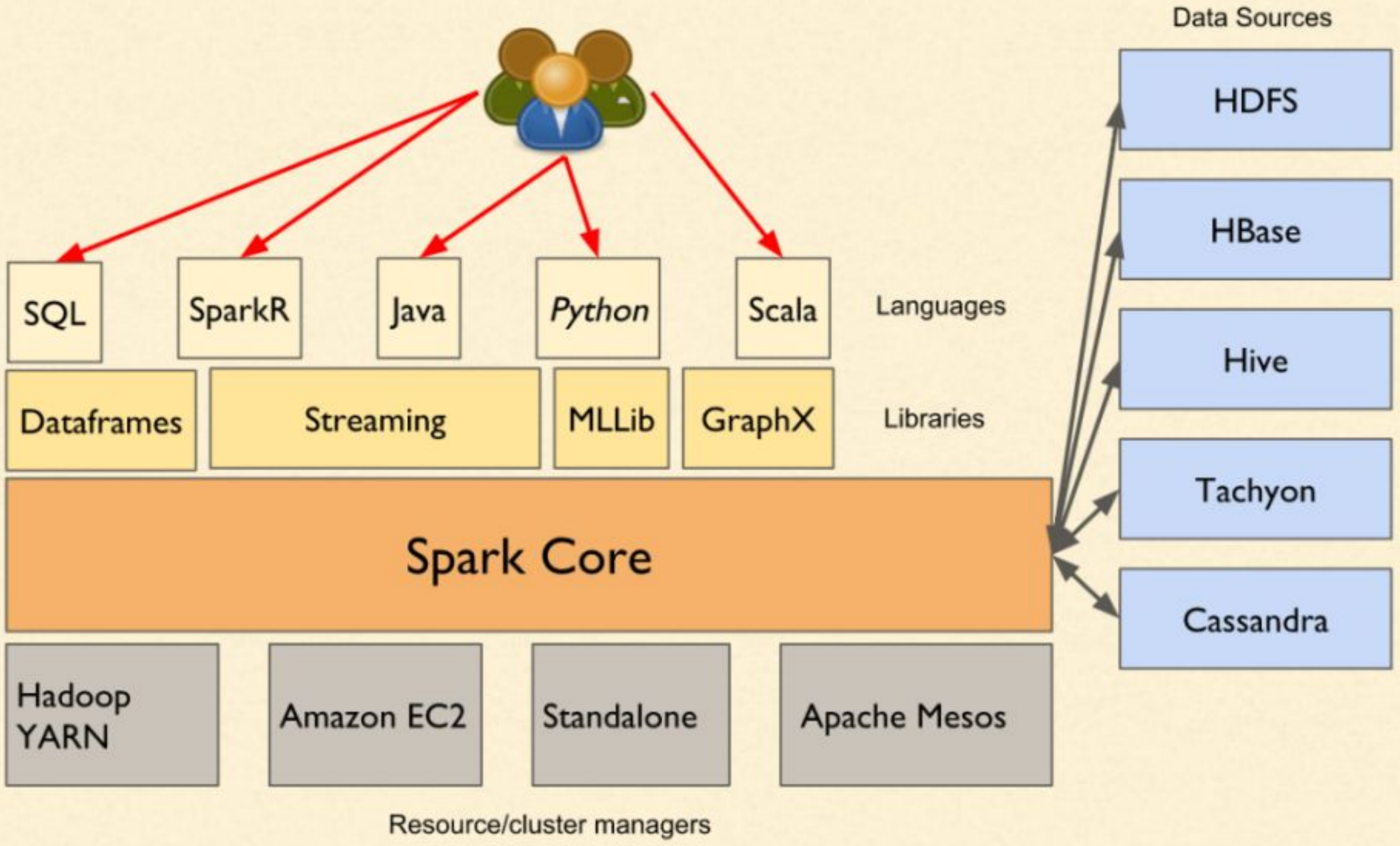
A Data Frame is a *Dataset* organized into named columns. It is conceptually equivalent to a table in a relational database or a data frame in R/Python, but with richer optimizations under the hood. Data Frames can be constructed from a wide array of sources such as: structured data files, tables in Hive, external databases, or existing RDDs.

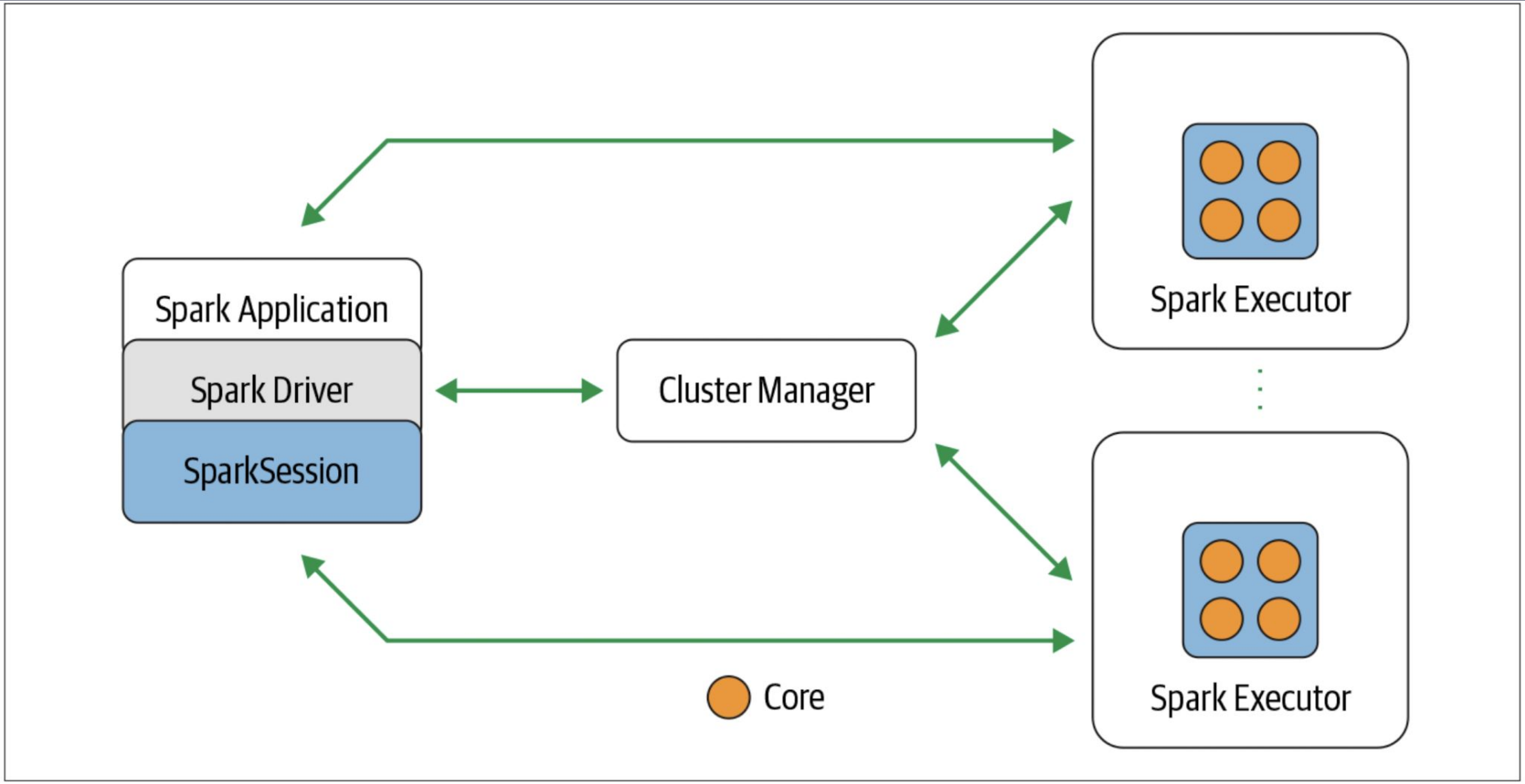
PW SKILLS

Spark Architecture


SQL
Streaming
GraphX
MLlib

Spark Architecture




Spark Application
Spark Driver
SparkSession
Cluster Manager
Spark Executor
Spark Executor
Core

SKILLS

Spark Ml Lib

Spark MLlib is used to perform machine learning in Apache Spark. MLlib consists popular algorithms and utilities.

MLlib Overview:

spark.mllib contains the original API built on top of RDDs. It is currently in maintenance mode.
spark.ml provides higher level API built on top of Data Frames for constructing ML pipelines. spark.ml is the primary Machine Learning API for Spark at the moment.

Spark MLlib Tools

Spark MLlib provides the following tools:

- **ML Algorithms:** ML Algorithms form the core of MLlib. These include common learning algorithms such as classification, regression, clustering and collaborative filtering.

- ❑ **Featurization:** Featurization includes feature extraction, transformation, dimensionality reduction and selection.
- ❑ **Pipelines:** Pipelines provide tools for constructing, evaluating and tuning ML Pipelines.
- ❑ **Persistence:** Persistence helps in saving and loading algorithms, models and Pipelines.
- ❑ **Utilities:** Utilities for linear algebra, statistics and data handling.

MLlib Algorithms

The popular algorithms and utilities in Spark MLlib are:

- ❑ Basic Statistics
- ❑ Regression
- ❑ Classification
- ❑ Recommendation System
- ❑ Clustering
- ❑ Dimensionality Reduction
- ❑ Feature Extraction
- ❑ Optimization

SKILLS

Spark Nlp

SKILLS

THANK YOU