UNDERSTANDING SPARK ESSENTIAL & ARCHITECTURE

CLASS-IV

Instructor: Palash Gupta

Re-Cap

Hello Everyone!

Last few class we went though following topics in general.

- Concept of Big Data and what it?
- Hadoop Key components
- Understanding HDFS, MAP Reduce, YARN
- Setting up HDFS – Distributed Storage Technique
- Working with HDFS
- Starting deep dive with Distributed Computing

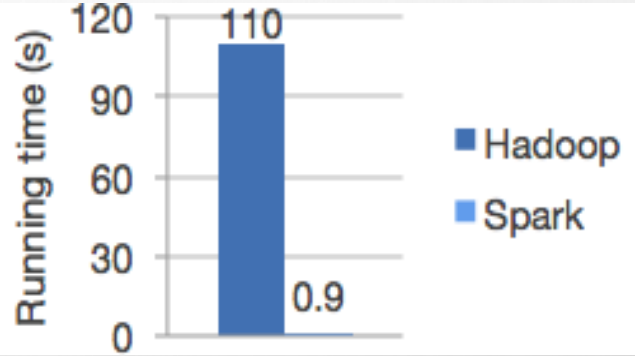
What we are going to Cover today?

- Introduction to Apache Spark
- Why Spark where we have Hadoop?
- Spark Architecture
- Introduction to Spark Component
- Introduction to Spark RDD, Dataset, DataFrame and DAG
- Understanding Spark Execution Model

Introduction to Apache Spark

- Apache Spark™ is a unified analytics engine for large-scale data processing.
- It runs 10x faster in disk and 100x faster in-memory than Hadoop.
- Originally developed at the University of California & later donated to Apache Foundation.
- It supports both batch and real time processing.

https://spark.apache.org/



Logistic regression in Hadoop and Spark

Why Spark where we have Hadoop?

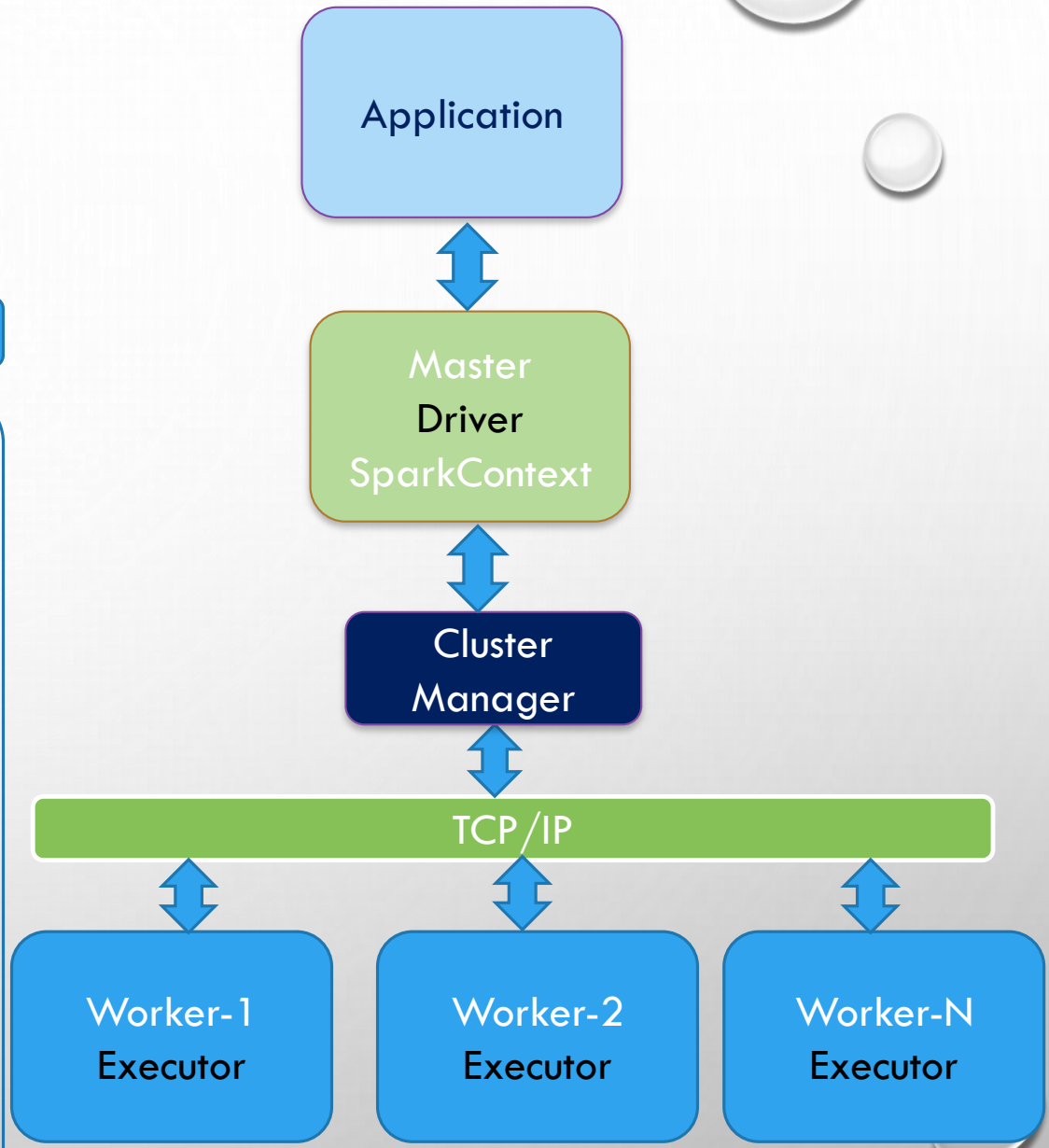
❑ **Why we would need Spark?**

- Map reduce only do batch processing, however Spark supports both batch processing, real time as well as streaming & many more.
- Spark is much faster than Hadoop
- Spark gives much more support for Programming languages staring from Scala, Python, Java and R. In Hadoop Map Reduce job it supports mainly Java.
- It supports interactive mode for Scala, Python, R and SQL Shells.
- Runs Everywhere & can access diverse data sources.
- Easy to write.
- Powerful Caching


```
from pyspark.sql import SQLContext

sqlContext = SQLContext(sc)

data_frame =
sqlContext.read.format('com.databricks.spark.csv').options(header='
true').load('/home/hadoop/development/Pre_NSN_Traffic_BTS_RAW
_2G.csv')

df.show()
```

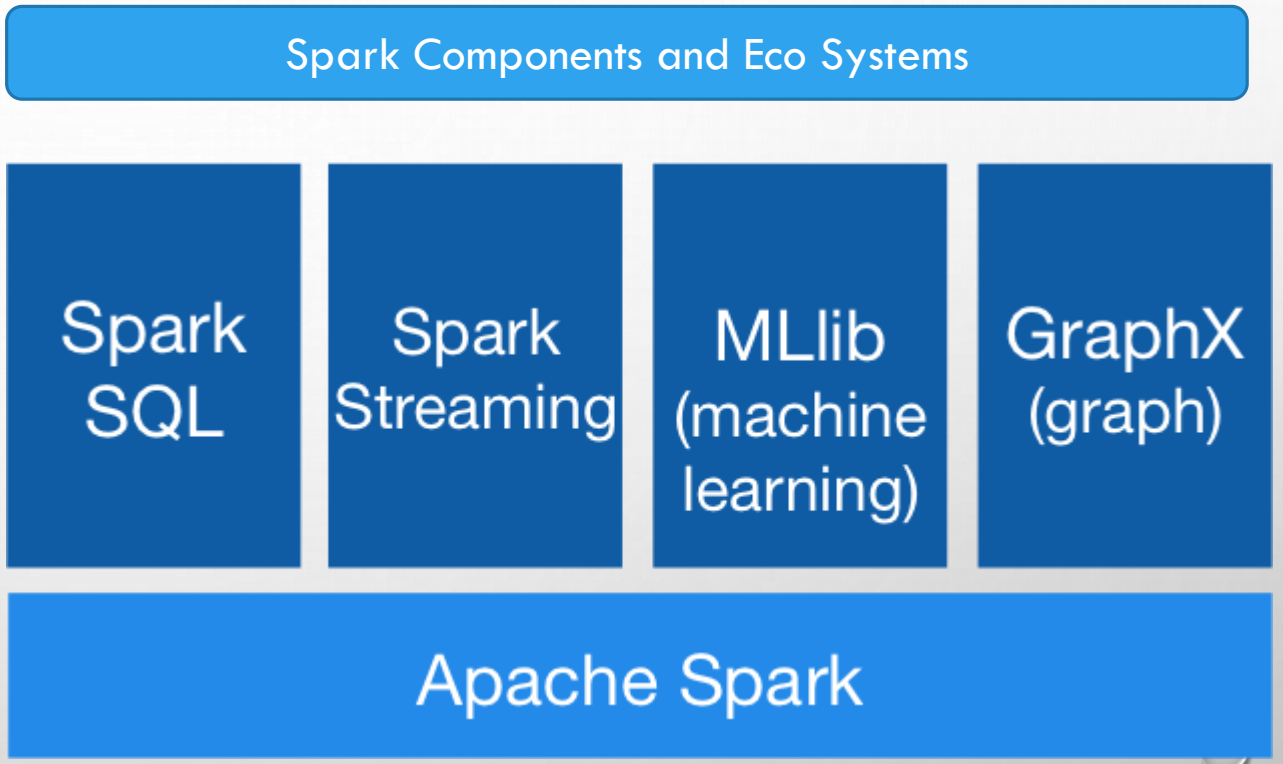
Spark Architecture

Driver	SC	Cluster Manager	Worker
○ Application hosted in Driver. ○ Driver basically owns your code to be run. ○ It creates a JVM for the code.	○ It is the gateway of Spark like you use database connection.	○ Allocates resources to Driver Program to run the tasks. ○ Schedule the Spark Application.	○ It is including executors and tasks to be executed. ○ It executes the task assigned by the executors.



Introduction to Spark Components

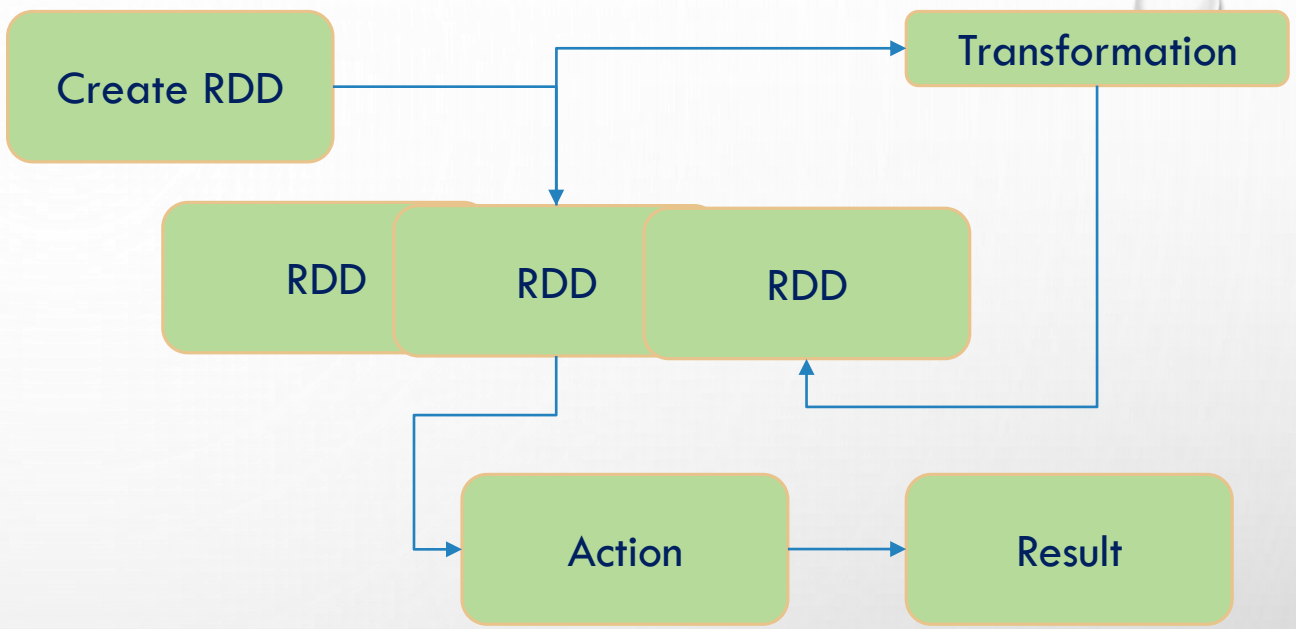
- Spark SQL:
  - It is giving SQL like query capabilities over verities of data sources utilizing RDD/DataFrame/DataSet.
- Spark Streaming:
  - It allows developer to take streaming data as input as well output within same Spark application.
- Spark MLib:
  - It supports running machine learning algorithm over large data set e.g. Summary Statistics, Clustering, Correlations, feature extraction etc.
- Spark GraphX:
  - Graph Construction and Transformation by working both on Graph or non-Graph sources.
- Spark Core:
  - Basic I/O Functions, Scheduling & Monitoring and base engine.



Introduction to Spark RDD, Dataset, DataFrame and DAG

❑ RDD:

- RDD is Resilient Distributed Dataset
- It is a collection of items distributed across multiple nodes that can be manipulated in parallel
- It is the basic building block of Spark
- RDD can be cached and persisted
- It can be reconstructed using DAG
- RDD can partitioned in multiple cluster nodes
- It is type safe, lazy and immutable
- They are slow in non-JVM platform e.g. Python
- It is very easy to build an inefficient RDD transformation chain. If you do with this in RDD then you will not get that efficiency of optimizer




```
>>>numbers = sc.parallelize([14,21,88,99,455])
>>>import math
>>>log_values = numbers.map(lambda n : math.log10(n))
>>>log_values.collect()
```

e.g. users.join(trips, users("id") == trips("uid")).filter(trip("tripdate") > "2020-04-01").

Introduction to Spark RDD, Dataset, DataFrame and DAG Contd.

❑ DataFrame:

- DataFrame provides higher-level abstraction allowing you to use query to manipulate data.
- It can be created from RDD & different Spark supported data source.
- It has type – so you will get error for type issue
- It is lazy and can be mutable
- DataFrame queries are optimized
- Finally everything changes to RDD

```
from pyspark.sql import SQLContext

sqlContext = SQLContext(sc)

df =
sqlContext.read.format('com.databricks.spark.csv').options(header='true').load("hdfs://nakparsdev-1.nak-mci.ir:9000/parsdev/CM_Dump.txt")

df.show()
```

❑ DataSet:

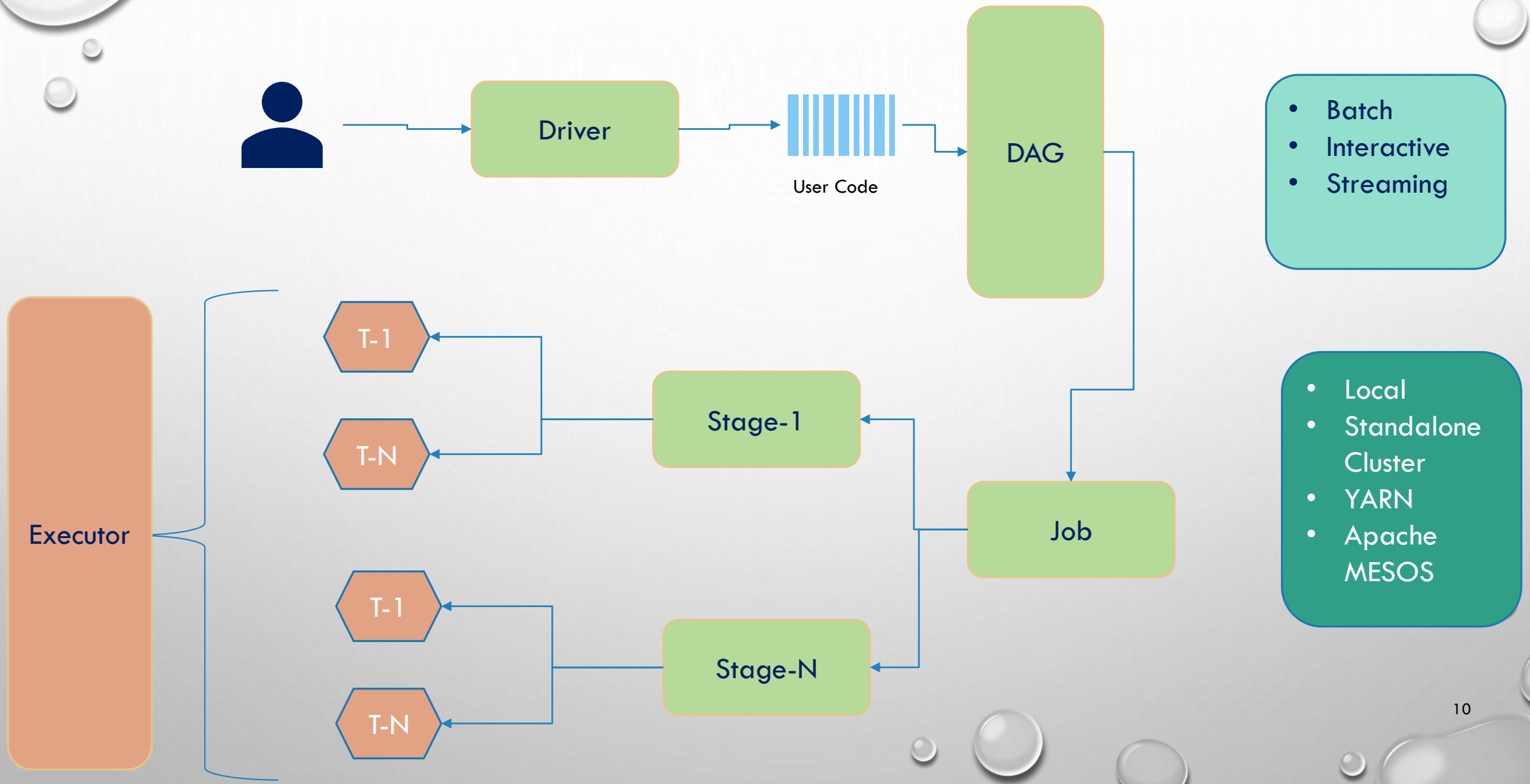
- It is extension of DataFrame API
- You can run SQL still on DataSets
- It is conceptually like RDD and certain level of type safety is there
- It is lazy and can be mutable
- DataSets queries are optimized
- Use special fast in-memory encoding for serialization/de-serialization
- Finally everything changes to RDD

```
val lines = sc.textFile("hdfs://path/to/some/ebook.txt")
val words = lines.flatMap(_.split("""\s+""")).filter(_.nonEmpty)
val counts = words.groupBy(_.toLowerCase).map { case (w, all) => (w, all.size) }
```

Datasets:

```
val lines = sqlContext.read.text("hdfs://path/to/some/ebook.txt").as[String]
val words = lines.flatMap(_.split("""\s+""")).filter(_.nonEmpty)
val counts = words.groupBy(_.toLowerCase).count()
```

Understanding Spark Execution Model



QUESTION & ANSWER

THANKS FOR ATTENDING THE CLASS & YOUR CO-OPERATION

References

- http://elephantscale.com/
- https://www.hadoop.apache.org
- https://www.guru99.com/
- https://techvidvan.com/tutorials
- https://www.tutorialspoint.com/
- https://spark.apache.org/