Open | DSE

Architecting Big Data
Solutions with Apache Spark
Lecture 1: Introduction to the Course

Ekhtiar Syed

Data Engineer/Scientist

Signify Research (formerly known as Philips Lighting)

Course Objective

- In this course we will learn how to build Big Data Solutions with Apache Spark.
- This is mostly a notebook lab-oriented course with a few slides to cover the theory part. However, there are more theories inside the notebook labs themselves.
- You can find the latest version of the course in our GitHub repository: https://github.com/open-dse/architect_big_data_solutions_spark under our open data science and engineering education initiative.
- We will use Databricks as our infrastructure to run our notebooks on. Databricks provides you a single node spark cluster for free.

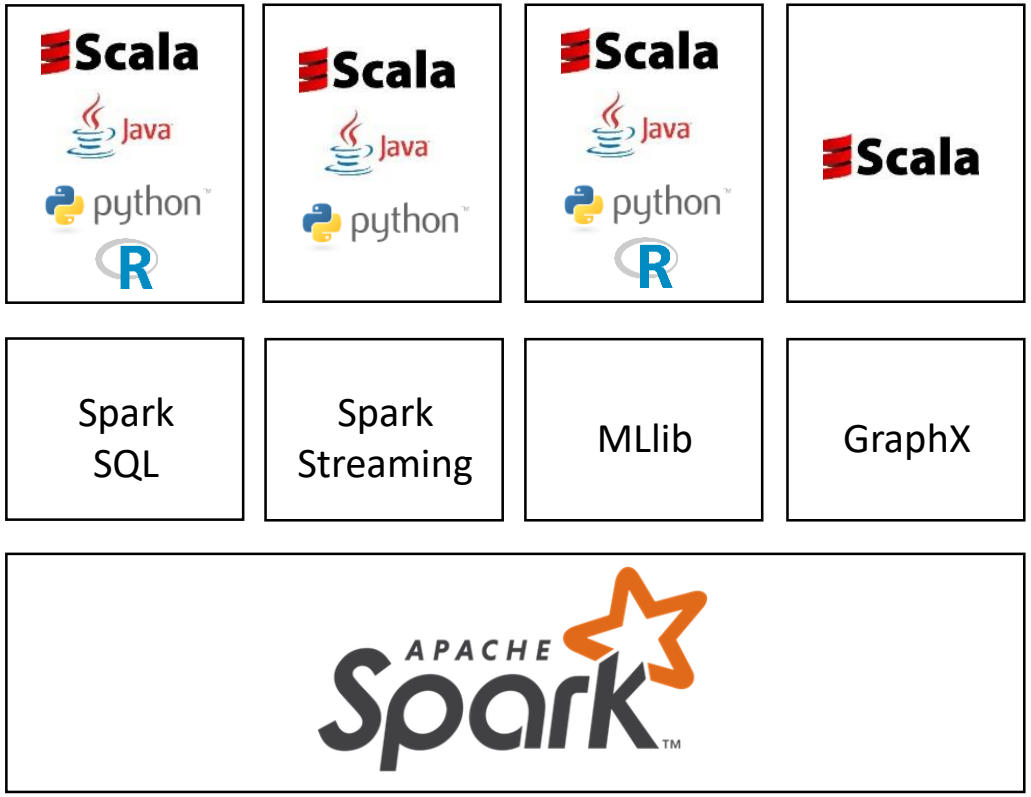
What Is Apache Spark?

spark.apache.org

- Spark is a fast and general engine for large-scale data processing.
- **Easy to Use:** Spark offers a rich application programming interface (API) for developing big data applications.
- **Fast:** Spark takes advantage of in-memory compute to provide fast data processing capabilities in a distributed environment.
- **General Purpose:** Spark provides a unified integrated platform for different types of data processing jobs.
- **Scalable:** The data processing capacity of a Spark cluster can be increased by just adding more nodes to a cluster.
- **Fault Tolerant:** Spark automatically handles the failure of a node in a cluster. Failure of a node may degrade performance, but will not crash an application.

Apache Spark Modules

Along with it's core modules, Spark mainly offers the following four modules:



Spark SQL: Enables the use of SQL statements or DataFrame API inside Spark applications

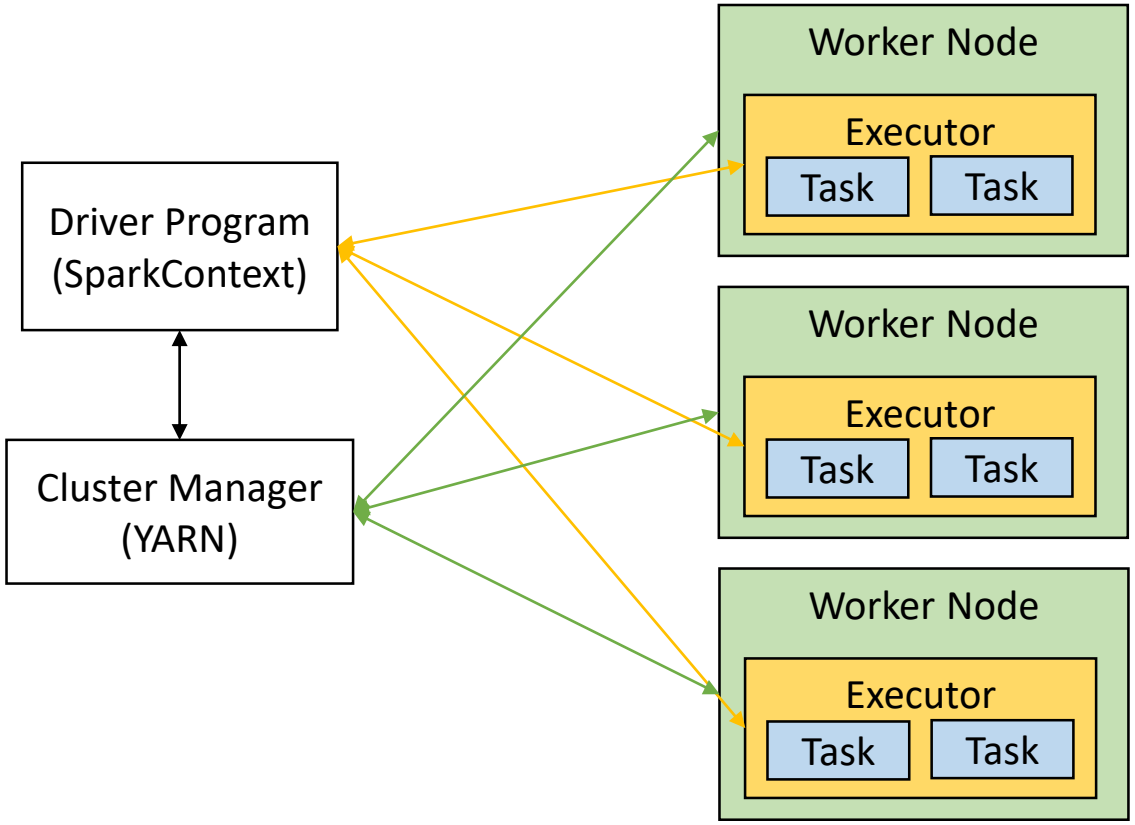
Spark Streaming: Enables processing of live data streams

Mlib: Enables development of machine learning applications

GraphX: Enables graph processing and supports a growing library of graph algorithms

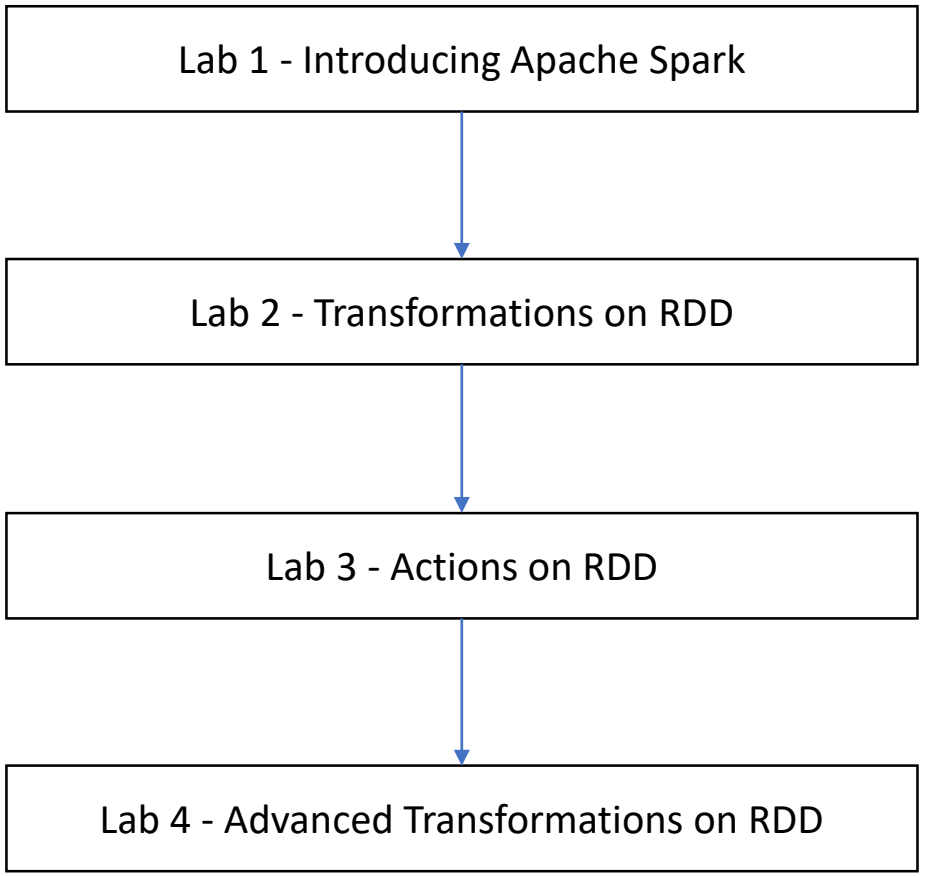
Spark Under The Hood

Spark allows you to write code to process data in a distributed environment. You write the code on a driver node, and spark abstracts the difficult task of distributing the problem on multiple nodes in the following way:



- Spark has five key entities: driver program, cluster manager, workers, executors, and tasks.
- Worker nodes provide compute resources to a Spark application and runs as distributed process.
- Spark uses a cluster manager to acquire cluster resources for executing a job. Spark currently supports standalone, Mesos, and YARN.
- Driver program is an application that uses Spark as a library. A driver program can launch one or more jobs on a Spark cluster.
- An executor is a Java virtual machine process that Spark creates on each worker for an application.
- Task is the smallest unit of work that Spark sends to an executor.
- Driver node orchestrates worker nodes to execute the “graph of operations” in a lazy way.

Spark Core & RDD



- Spark's core revolves around the concept of RDD, which stands for resilient distributed dataset
- **RDD** is the building block for all the other modules of Spark
- In simple terms, if we load data spark will store it as an RDD
- We will learn what we can do with RDDs in the four labs illustrated on the left
- We will use Databricks for running all our notebooks, you can create a free account by clicking on the "Community Edition" from https://databricks.com/try-databricks
- Make sure you choose "Community Edition" and not Free Trial, as it will require credit card
- After signing up, you can load the notebooks in the course repository in Databricks environment by "Importing" the notebook from your local filesystem

Let’s Start The Practical Part!
Load Up Lab 1, 2, 3, 4