# Big Data Hadoop Training

Session 12 Assignment 2 Solution:

**1. How are worker, executor and task related to each other?**

A)

* **Worker** nodes are machines that run executors. There will be one JVM (= 1 UNIX Process) per Worker. Each Worker can spawn one or more Executors.
* **Spark Executors** are worker nodes' processes in charge of running individual tasks in a given Spark job. They run in child JVM (= 1 UNIX process). They execute one or more task using threads in a ThreadPool.
* **Task** is a unit of work to execute, in an executor thread.

Thus, we can state that workers contain executors for driving out tasks (computations).

**2. What are the key features of Spark?**

A) **Apache Spark** is a fast and general-purpose cluster computing system. Spark is a framework with Scheduling, Monitoring and Distributing applications. It has DAG (Directed Acyclic Graph) based execution and its fundamental data structure is Resilient Distributed Datasets (RDD). It is an immutable distributed collection of objects. **Apache Spark** has following features.

* **Speed** − Spark helps to run an application in Hadoop cluster, up to 100 times faster in memory, and 10 times faster when running on disk. This is possible by reducing number of read/write operations to disk. It stores the intermediate processing data in memory. It has **In-Memory** processing (Data Caching).
* **Supports multiple languages** − Spark provides built-in APIs in Java, Scala, or Python. Therefore, you can write applications in different languages. Spark comes up with 80 high-level operators for interactive querying. It has a rich set of APIs and provides strong ecosystem tool support.
* **Advanced Analytics** − Spark not only supports ‘Map’ and ‘reduce’. It also supports SQL queries, Streaming data, Machine learning (ML), and Graph algorithms. Therefore, Spark is a unified platform.

**3. What is Spark Driver?**

A)

* A **Spark driver** is the process that creates and owns an instance of SparkContext.
* It is our Spark application that launches the main method in which the instance of SparkContext is created.
* It is the cockpit of jobs and tasks execution (using DAGScheduler and Task Scheduler).
* It hosts Web UI for the environment.
* It connects to the cluster manager to allocate resources across applications.
* It splits a Spark application into tasks and schedules them to run on executors.
* A driver is where the task scheduler lives and spawns tasks across workers.
* A driver coordinates workers and overall execution of tasks.

**4. What are the benefits of Spark over MapReduce?**

A)

* Spark is easy to program and doesn’t require any abstractions whereas MapReduce is difficult to program and needs abstractions.
* Spark has interactive mode and for MapReduce there is no in-built interactive mode for Hadoop MapReduce except Pig and Hive.
* Spark makes it possible to perform Streaming, Batch Processing and Machine Learning all in the same cluster whereas Hadoop MapReduce is used for generating reports that help in finding answers to historical queries.
* Spark can execute batch processing jobs near about 10 to 100 times faster than Hadoop MapReduce because of its In-Memory processing capability
* Spark ensures lower latency computations by caching the partial results across its memory of distributed workers while MapReduce is disk oriented completely.
* Writing Spark code is always compact when compared to writing Hadoop MapReduce pipelines which are complex and lengthy process.

Thus, we can see many benefits with Spark over MapReduce.

**5. What is Spark Executor?**

A)

* **Spark Executors** are worker nodes' processes in charge of running individual tasks in a given Spark job.
* They are launched at the beginning of a Spark application and typically run for the entire lifetime of an application.
* Once they have run the task they send the results to the driver.
* They also provide in-memory storage for RDDs that are cached by user programs through Block Manager.
* They run in child JVM (= 1 UNIX process). They execute one or more task using threads in a ThreadPool.