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

The driver is the process where the main method runs. First it converts the user program into tasks and after that it schedules the tasks on the executors.

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.

When executors are started they register themselves with the driver and from so on they communicate directly.

The workers are in charge of communicating the cluster manager the availability of their resources.The workers is where the tasks are executed - executors. They should have resources and network connectivity sufficient to perform transformations and actions on the RDDs defined in the main program.

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

* Enables applications in hadoop clusters to run upto 100x faster in memory and 10x in disk
* Currently provides APIs in Scala, Java,R and Python
* Integrates well with the Hadoop ecosystem and data sources (HDFS, Amazon S3, Hive, HBase, Cassandra, etc.)
* Can run on clusters managed by Hadoop YARN or Apache Mesos, and can also run standalone
* interactive shell (REPL).  Using REPL, one can test the outcome of each line of code without first needing to code and execute the entire job
* Spark’s Analytic Suite – Spark comes with tools for interactive query analysis, large-scale graph processing and analysis and real-time analysis.
* Resilient Distributed Datasets (RDD’s) – RDD’s are distributed objects that can be cached in-memory, across a cluster of compute nodes. They are the primary data objects used in Spark.
* Distributed Operators – Besides MapReduce, there are many other operators one can use on RDD’s.

**3. What is Spark Driver?**

A Spark driver (application’s driver process) is a JVM process that hosts [SparkContext](https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-sparkcontext.html) for a Spark application. It is the master node in a Spark application.

It is the cockpit of jobs and tasks execution (using [DAGScheduler](https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-dagscheduler.html) and [Task Scheduler](https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-TaskScheduler.html)). It hosts [Web UI](https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-webui.html) for the environment.

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?**

* Executes jobs 10 to 100 times faster than Hadoop MapReduce.
* Programmers can modify the data in real-time through Spark streamingmwhere as Mapreduce helps is processing a batch of stored data.
* Programmers can perform streaming, batch processing and machine learning ,all in the same cluster.
* Unlike MR Spark is easy to program and does not require any abstractions.
* MR is disk oriented completely
* Writing MR pipeline is complex and lengthy process, whereas writing spark code is compact.

**5. What is Spark Executor?**

* Executor is a distributed agent that is responsible for executing [tasks](https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-taskscheduler-tasks.html).
* They 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