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

* **Worker:** A worker node is any node that can run application code in the cluster.
* **Task**: A task is a unit of work that can be run on a partition of a distributed dataset and gets executed on a single executor.
* **Executor**: An executor is a single JVM process which is launched for an application on a ***worker node***. Executor runs ***tasks*** and keeps data in memory or disk storage across them. Each application has its own executors. A single node can run multiple executors and executors for an application can span multiple worker nodes. An executor stays up for the duration of the Spark Application and runs the tasks in multiple threads.

2. What are the key features of Spark?

* **Fast Processing:** Spark enables applications in Hadoop clusters to run up to 100x faster in memory, and 10x faster even when running on disk.

Spark makes it possible by reducing number of read/write to disc. It stores this intermediate processing data in-memory. It uses the concept of a Resilient Distributed Dataset (RDD), which allows it to transparently store data on memory and persist it to disc only it’s needed. This helps to reduce most of the disc read and write, the main time consuming factors of data processing.

* **Ease of Use:** Spark lets you quickly write applications in Java, Scala, or Python. This helps developers to create and run their applications on their familiar programming languages.
* **Support for Sophisticated Analytics:** In addition to simple 'map' and 'reduce' operations, Spark supports SQL queries, streaming data and complex analytics such as machine learning and graph algorithms.
* **Real Time Stream Processing:** Spark can handle real time streaming. Map-reduce mainly handle and process the data stored already and it has to depend on other frameworks with their integration to handle streaming.
* **Ability to Integrate with Hadoop and Existing Hadoop Data:** Spark can run independently. Apart from that it can run on Hadoop 2’s YARN cluster manager and can read any existing Hadoop data. It can read from any Hadoop data sources for example HBase, HDFS etc.

3. What is Spark Driver?

* Driver program contains application's main () function and is responsible for launching various parallel operations on the cluster.
* It is the process which is running the user code which in turn creates the SparkContext object, creates RDDs and performs transformation and action operation on RDD.
* Driver program access Apache Spark through a SparkContext object which represents a connection to computing cluster (From Spark 2.0 onwards we can access SparkContext object through SparkSession).
* Driver program is responsible for converting user program into the unit of physical execution called task.
* It also defines distributed datasets on the cluster and we can apply different operations on Dataset (transformation and action).
* Spark program creates a logical plan called Directed Acyclic graph which is converted to physical execution plan by the driver when driver program runs.

4. What are the benefits of Spark over MapReduce?

* Spark is easy to program and don't require that much hand coding whereas MapReduce is not that easy in terms of programming and requires lots of hand coding
* It has interactive mode whereas in MapReduce there is no built-in interactive mode, MapReduce is developed for batch processing.
* For data processing Spark can use streaming, machine learning, and batch processing whereas Hadoop MapReduce can use the batch engine. Spark is general purpose cluster computation engine.
* Spark executes batch processing jobs about 10 to 100 times faster than Hadoop MapReduce.
* Spark uses an abstraction called RDD which makes Spark feature rich, whereas map reduce doesn't have any abstraction
* Spark uses lower latency by caching partial/complete results across distributed nodes whereas MapReduce is completely disk-based.

5. What is Spark Executor?

An executor is a single JVM process which is launched for an application on a *worker node*. Executor runs tasks and keeps data in memory or disk storage across them. Each application has its own executors. A single node can run multiple executors and executors for an application can span multiple worker nodes. An executor stays up for the duration of the Spark Application and runs the tasks in multiple threads.