1. What is Apache Spark?

Wikipedia defines Apache Spark “an open source cluster computing framework originally developed in the AMPLab at University of California, Berkeley but was later donated to the Apache Software Foundation where it remains today. In contrast to Hadoop’s two-stage disk-based MapReduce paradigm, Spark’s multi-stage in-memory primitives provides performance up to 100 times faster for certain applications. By allowing user programs to load data into a cluster’s memory and query it repeatedly, Spark is well-suited to machine learning algorithms.”

Spark is essentially a fast and flexible data processing framework. It has an advanced execution engine supporting cyclic data flow with in-memory computing functionalities. Apache Spark can run on Hadoop, as a standalone system or on the cloud. Spark is capable of accessing diverse data sources including HDFS, HBase, Cassandra among others

2. Explain the key features of Spark.

• Spark allows Integration with Hadoop and files included in HDFS.

• It has an independent language (Scala) interpreter and hence comes with an interactive language shell.

• It consists of RDD’s (Resilient Distributed Datasets), that can be cached across computing nodes in a cluster.

• It supports multiple analytic tools that are used for interactive query analysis, real-time analysis and graph processing. Additionally, some of the salient features of Spark include:

**Lighting fast processing:**When it comes to Big Data processing, speed always matters, and Spark runs Hadoop clusters way faster than others. Spark makes this possible by reducing the number of read/write operations to the disc. It stores this intermediate processing data in memory.

**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. This allows users to combine all these capabilities in a single workflow.

**Real-time stream processing:** Spark can handle real-time streaming. MapReduce primarily handles and processes previously stored data even though there are other frameworks to obtain real-time streaming.  Spark does this in the best way possible.

3. What is “RDD”?

RDD stands for Resilient Distribution Datasets: a collection of fault-tolerant operational elements that run in parallel. The partitioned data in RDD is immutable and is distributed in nature.

4. How does one create RDDs in Spark?

In Spark, parallelized collections are created by calling the SparkContext “parallelize” method on an existing collection in your driver program.

                val data = Array(4,6,7,8)

                val distData = sc.parallelize(data)

Text file RDDs can be created using SparkContext’s “textFile” method. Spark has the ability to create distributed datasets from any storage source supported by Hadoop, including your local file system, HDFS, Cassandra, HBase, [Amazon S3](http://wiki.apache.org/hadoop/AmazonS3), among others. Spark supports text files, “[SequenceFiles](http://hadoop.apache.org/common/docs/current/api/org/apache/hadoop/mapred/SequenceFileInputFormat.html)”, and any other Hadoop “I[nputFormat](http://hadoop.apache.org/docs/stable/api/org/apache/hadoop/mapred/InputFormat.html)” components.

                 val inputfile = sc.textFile(“input.txt”)

5. What does the Spark Engine do?

Spark Engine is responsible for scheduling, distributing and monitoring the data application across the cluster.

6. Define “Partitions”.

A “Partition” is a smaller and logical division of data, that is similar to the “split” in Map Reduce. Partitioning is the process that helps derive logical units of data in order to speed up data processing.

Here’s an example: val someRDD = sc.parallelize( 1 to 100, 4)

Here an RDD of 100 elements is created in four partitions, which then distributes a dummy map task before collecting the elements back to the driver program.

7. What operations does the “RDD” support?

 Transformations

 Actions

8. Define “Transformations” in Spark.

“Transformations” are functions applied on RDD, resulting in a new RDD. It does not execute until an action occurs. ***map***() and ***filer***() are examples of “transformations”, where the former applies the function assigned to it on each element of the RDD and results in another RDD. The ***filter***() creates a new RDD by selecting elements from the current RDD.

9. Define “Action” in Spark.

An “action” helps in bringing back the data from the RDD to the local machine. Execution of “action” is the result of all transformations created previously. ***reduce***() is an action that implements the function passed again and again until only one value is left. On the other hand, the ***take***() action takes all the values from the RDD to the local node.

10. What are the functions of “Spark Core”?

The “SparkCore” performs an array of critical functions like memory management, monitoring jobs, fault tolerance, job scheduling and interaction with storage systems.

It is the foundation of the overall project. It provides distributed task dispatching, scheduling, and basic input and output functionalities. RDD in Spark Core makes it fault tolerance. RDD is a collection of items distributed across many nodes that can be manipulated in parallel. Spark Core provides many APIs for building and manipulating these collections.

11. What is an “RDD Lineage”?

Spark does not support data replication in the memory. In the event of any data loss, it is rebuilt using the “RDD Lineage”. It is a process that reconstructs lost data partitions.

12. What is a “Spark Driver”?

“Spark Driver” is the program that runs on the master node of the machine and declares transformations and actions on data RDDs. The driver also delivers RDD graphs to the “Master”, where the standalone cluster manager runs.

13. What is SparkContext?

“SparkContext” is the main entry point for Spark functionality. A “SparkContext” represents the connection to a Spark cluster, and can be used to create RDDs, accumulators and broadcast variables on that cluster.

14. What is Hive on Spark?

Hive is a component of Hortonworks’ Data Platform (HDP). Hive provides an SQL-like interface to data stored in the HDP. Spark users will automatically get the complete set of Hive’s rich features, including any new features that Hive might introduce in the future.

The main task around implementing the Spark execution engine for Hive lies in query planning, where Hive operator plans from the semantic analyzer which is translated to a task plan that Spark can execute. It also includes query execution, where the generated Spark plan gets actually executed in the Spark cluster.

15. Name a few commonly used Spark Ecosystems.

 Spark SQL (Shark)

 Spark Streaming

 GraphX

 MLlib

 SparkR

16. What is “Spark Streaming”?

Spark supports stream processing, essentially an extension to the Spark API. This allows stream processing of live data streams. The data from different sources like Flume and HDFS is streamed and processed to file systems, live dashboards and databases. It is similar to batch processing as the input data is divided into streams like batches.  
Business use cases for Spark streaming: Each Spark component has its own use case. Whenever you want to analyze data with the latency of less than 15 minutes and greater than 2 minutes i.e. near real time is when you use Spark streaming

17. What is “GraphX” in Spark?

“GraphX” is a component in Spark which is used for graph processing. It helps to build and transform interactive graphs.

18. What is the function of “MLlib”?

“MLlib” is Spark’s machine learning library. It aims at making machine learning easy and scalable with common learning algorithms and real-life use cases including clustering, regression filtering, and dimensional reduction among others.

19. What is “Spark SQL”?

Spark SQL is a Spark interface to work with structured as well as semi-structured data. It has the capability to load data from multiple structured sources like “textfiles”, JSON files, Parquet files, among others. Spark SQL provides a special type of RDD called SchemaRDD. These are row objects, where each object represents a record.

Here’s how you can create an SQL context in Spark SQL:

        SQL context: scala> var sqlContext=new SqlContext

        HiveContext: scala> var hc = new HIVEContext(sc)

20. What is a “Parquet” in Spark?

**“**Parquet” is a columnar format file supported by many data processing systems. Spark SQL performs both read and write operations with the “Parquet” file.

21. What is an “Accumulator”?

“Accumulators” are Spark’s offline debuggers. Similar to “Hadoop Counters”, “Accumulators” provide the number of “events” in a program.

Accumulators are the variables that can be added through associative operations. Spark natively supports accumulators of numeric value types and standard mutable collections. “AggregrateByKey()” and “combineByKey()” uses accumulators.

22. Which file systems does Spark support?

 Hadoop Distributed File System (HDFS)

 Local File system

 S3

23. What is “YARN”?

“YARN” is a large-scale, distributed operating system for big data applications. It is one of the key features of Spark, providing a central and resource management platform to deliver scalable operations across the cluster.

24. List the benefits of Spark over MapReduce.

 Due to the availability of in-memory processing, Spark implements the processing around 10-100x faster than Hadoop MapReduce.

 Unlike MapReduce, Spark provides in-built libraries to perform multiple tasks form the same core; like batch processing, steaming, machine learning, interactive SQL queries among others.

 MapReduce is highly disk-dependent whereas Spark promotes caching and in-memory data storage

 Spark is capable of iterative computation while MapReduce is not.

Additionally, Spark stores data in-memory whereas Hadoop stores data on the disk. Hadoop uses replication to achieve fault tolerance while Spark uses a different data storage model, resilient distributed datasets (RDD). It also uses a clever way of guaranteeing fault tolerance that minimizes network input and output.

25. What is a “Spark Executor”?

When “SparkContext” connects to a cluster manager, it acquires an “Executor” on the cluster nodes. “Executors” are Spark processes that run computations and store the data on the worker node. The final tasks by “SparkContext” are transferred to executors.

26. List the various types of “Cluster Managers” in Spark.

The Spark framework supports three kinds of Cluster Managers:

* Standalone
* Apache Mesos
* YARN

27. What is a “worker node”?

“Worker node” refers to any node that can run the application code in a cluster.

28. Define “PageRank”.

“PageRank” is the measure of each vertex in a graph.

29. Can we do real-time processing using Spark SQL?

30. What is the biggest shortcoming of Spark?Spark utilizes more storage space compared to Hadoop and MapReduce.

Also, Spark streaming is not actually streaming, in the sense that some of the window functions cannot properly work on top of micro batching.