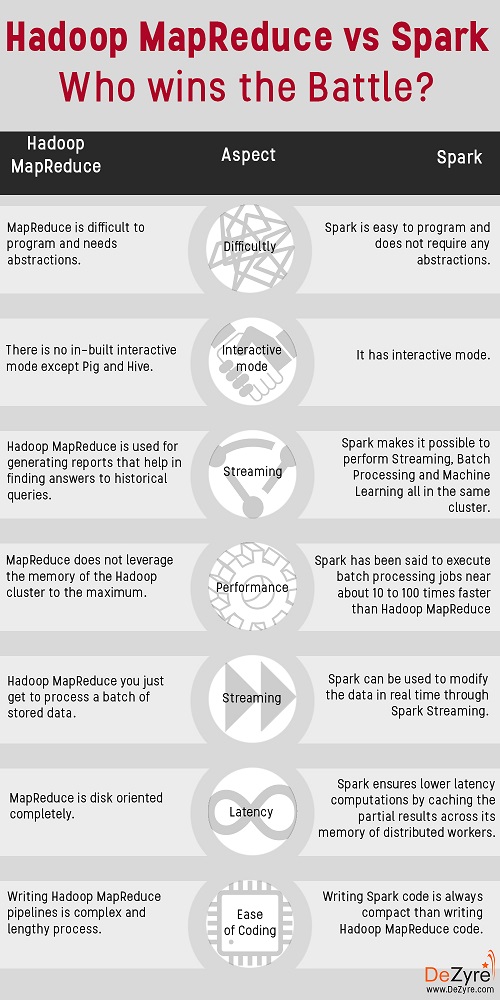
SPARK

**1) Compare Spark vs Hadoop MapReduce**

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Hadoop MapReduce** | **Apache Spark** |
| Memory | Does not leverage the memory of the hadoop cluster to maximum. | Let's save data on memory with the use of RDD's. |
| Disk usage | MapReduce is disk oriented. | Spark caches data in-memory and ensures low latency. |
| Processing | Only batch processing is supported | Supports real-time processing through spark streaming. |
| Installation | Is bound to hadoop. | Is not bound to Hadoop. |
| **Spark vs Hadoop** | | |



Simplicity, Flexibility and Performance are the major advantages of using Spark over Hadoop.

Spark is 100 times faster than Hadoop for big data processing as it stores the data in-memory, by placing it in Resilient Distributed Databases (RDD).

Spark is easier to program as it comes with an interactive mode.

It provides complete recovery using lineage graph whenever something goes wrong.

Refer [Spark vs Hadoop](https://www.dezyre.com/article/hadoop-mapreduce-vs-apache-spark-who-wins-the-battle/83)

**2) What is Shark?**

Most of the data users know only SQL and are not good at programming. Shark is a tool, developed for people who are from a database background - to access Scala MLib capabilities through Hive like SQL interface. Shark tool helps data users run Hive on Spark - offering compatibility with Hive metastore, queries and data.

**3) List some use cases where Spark outperforms Hadoop in processing.**

Sensor Data Processing –Apache Spark’s ‘In-memory computing’ works best here, as data is retrieved and combined from different sources.

Spark is preferred over Hadoop for real time querying of data

Stream Processing – For processing logs and detecting frauds in live streams for alerts, Apache Spark is the best solution.

**4) What is a Sparse Vector?**

A sparse vector has two parallel arrays –one for indices and the other for values. These vectors are used for storing non-zero entries to save space.

**5) What is RDD?**

RDDs (Resilient Distributed Datasets) are basic abstraction in Apache Spark that represent the data coming into the system in object format. RDDs are used for in-memory computations on large clusters, in a fault tolerant manner. RDDs are read-only portioned, collection of records, that are –

Immutable – RDDs cannot be altered.

Resilient – If a node holding the partition fails the other node takes the data.

**6) Explain about transformations and actions in the context of RDDs.**

Transformations are functions executed on demand, to produce a new RDD. All transformations are followed by actions. Some examples of transformations include map, filter and reduceByKey.

Actions are the results of RDD computations or transformations. After an action is performed, the data from RDD moves back to the local machine. Some examples of actions include reduce, collect, first, and take.

**7) What are the languages supported by Apache Spark for developing big data applications?**

Scala, Java, Python, R and Clojure

**8) Can you use Spark to access and analyse data stored in Cassandra databases?**

Yes, it is possible if you use Spark Cassandra Connector.

**9) Is it possible to run Apache Spark on Apache Mesos?**

Yes, Apache Spark can be run on the hardware clusters managed by Mesos.

**10) Explain about the different cluster managers in Apache Spark**

The 3 different clusters managers supported in Apache Spark are:

YARN

Apache Mesos -Has rich resource scheduling capabilities and is well suited to run Spark along with other applications. It is advantageous when several users run interactive shells because it scales down the CPU allocation between commands.

Standalone deployments – Well suited for new deployments which only run and are easy to set up.

**11) How can Spark be connected to Apache Mesos?**

To connect Spark with Mesos-

Configure the spark driver program to connect to Mesos. Spark binary package should be in a location accessible by Mesos. (or)

Install Apache Spark in the same location as that of Apache Mesos and configure the property ‘spark.mesos.executor.home’ to point to the location where it is installed.

**12) How can you minimize data transfers when working with Spark?**

Minimizing data transfers and avoiding shuffling helps write spark programs that run in a fast and reliable manner. The various ways in which data transfers can be minimized when working with Apache Spark are:

Using Broadcast Variable- Broadcast variable enhances the efficiency of joins between small and large RDDs.

Using Accumulators – Accumulators help update the values of variables in parallel while executing.

The most common way is to avoid operations ByKey, repartition or any other operations which trigger shuffles.

**13)  Why is there a need for broadcast variables when working with Apache Spark?**

These are read only variables, present in-memory cache on every machine. When working with Spark, usage of broadcast variables eliminates the necessity to ship copies of a variable for every task, so data can be processed faster. Broadcast variables help in storing a lookup table inside the memory which enhances the retrieval efficiency when compared to an RDD lookup ().

**14)  Is it possible to run Spark and Mesos along with Hadoop?**

Yes, it is possible to run Spark and Mesos with Hadoop by launching each of these as a separate service on the machines. Mesos acts as a unified scheduler that assigns tasks to either Spark or Hadoop.

**15)  What is lineage graph?**

The RDDs in Spark, depend on one or more other RDDs. The representation of dependencies in between RDDs is known as the lineage graph. Lineage graph information is used to compute each RDD on demand, so that whenever a part of persistent RDD is lost, the data that is lost can be recovered using the lineage graph information.

**16) How can you trigger automatic clean-ups in Spark to handle accumulated metadata?**

You can trigger the clean-ups by setting the parameter ‘spark.cleaner.ttl’ or by dividing the long running jobs into different batches and writing the intermediary results to the disk.

**17) Explain about the major libraries that constitute the Spark Ecosystem**

**Spark MLib**- Machine learning library in Spark for commonly used learning algorithms like clustering, regression, classification, etc.

**Spark Streaming**– This library is used to process real time streaming data.

**Spark GraphX**– Spark API for graph parallel computations with basic operators like joinVertices, subgraph, aggregateMessages, etc.

**Spark SQL**– Helps execute SQL like queries on Spark data using standard visualization or BI tools.

**18) What are the benefits of using Spark with Apache Mesos?**

It renders scalable partitioning among various Spark instances and dynamic partitioning between Spark and other big data frameworks.

**19) What is the significance of Sliding Window operation?**

Sliding Window controls transmission of data packets between various computer networks. Spark Streaming library provides windowed computations where the transformations on RDDs are applied over a sliding window of data. Whenever the window slides, the RDDs that fall within the particular window are combined and operated upon to produce new RDDs of the windowed DStream.

**20) What is a DStream?**

Discretized Stream is a sequence of Resilient Distributed Databases that represent a stream of data. DStreams can be created from various sources like Apache Kafka, HDFS, and Apache Flume. DStreams have two operations –

Transformations that produce a new DStream.

Output operations that write data to an external system.

**21) When running Spark applications, is it necessary to install Spark on all the nodes of YARN cluster?**

Spark need not be installed when running a job under YARN or Mesos because Spark can execute on top of YARN or Mesos clusters without affecting any change to the cluster.

**22) What is Catalyst framework?**

Catalyst framework is a new optimization framework present in Spark SQL. It allows Spark to automatically transform SQL queries by adding new optimizations to build a faster processing system.

**23) Name a few companies that use Apache Spark in production.**

Pinterest, Conviva, Shopify, Open Table

**24) Which spark library allows reliable file sharing at memory speed across different cluster frameworks?**

Tachyon

**Work On Interesting** [Data Science Projects](https://www.dezyre.com/data-science-projects?utm_source=TextLink&utm_medium=ProjectsLink&utm_campaign=Blog_SparkInterviewQuestions_208) **using Spark to build an impressive project portfolio!**

**25) Why is BlinkDB used?**

BlinkDB is a query engine for executing interactive SQL queries on huge volumes of data and renders query results marked with meaningful error bars. BlinkDB helps users balance ‘query accuracy’ with response time.

**26) How can you compare Hadoop and Spark in terms of ease of use?**

Hadoop MapReduce requires programming in Java which is difficult, though Pig and Hive make it considerably easier. Learning Pig and Hive syntax takes time. Spark has interactive APIs for different languages like Java, Python or Scala and also includes Shark i.e. Spark SQL for SQL lovers - making it comparatively easier to use than Hadoop.

**27) What are the common mistakes developers make when running Spark applications?**

Developers often make the mistake of-

Hitting the web service several times by using multiple clusters.

Run everything on the local node instead of distributing it.

Developers need to be careful with this, as Spark makes use of memory for processing.

**28) What is the advantage of a Parquet file?**

Parquet file is a columnar format file that helps –

Limit I/O operations

Consumes less space

Fetches only required columns.

**29) What are the various data sources available in SparkSQL?**

Parquet file

JSON Datasets

Hive tables

**30) How Spark uses Hadoop?**

Spark has its own cluster management computation and mainly uses Hadoop for storage.

For the complete list of big data companies and their salaries- [**CLICK HERE**](https://docs.google.com/forms/d/1LFuWEKQKCLR231qR9WE5PZakJj77fTDIW6ox5328HFM/viewform)

**31) What are the key features of Apache Spark that you like?**

Spark provides advanced analytic options like graph algorithms, machine learning, streaming data, etc

It has built-in APIs in multiple languages like Java, Scala, Python and R

It has good performance gains, as it helps run an application in the Hadoop cluster ten times faster on disk and 100 times faster in memory.

**32) What do you understand by Pair RDD?**

Special operations can be performed on RDDs in Spark using key/value pairs and such RDDs are referred to as Pair RDDs. Pair RDDs allow users to access each key in parallel. They have a reduceByKey () method that collects data based on each key and a join () method that combines different RDDs together, based on the elements having the same key.

**33) Which one will you choose for a project –Hadoop MapReduce or Apache Spark?**

The answer to this question depends on the given project scenario - as it is known that Spark makes use of memory instead of network and disk I/O. However, Spark uses large amount of RAM and requires dedicated machine to produce effective results. So the decision to use Hadoop or Spark varies dynamically with the requirements of the project and budget of the organization.

**34) Explain about the different types of transformations on DStreams?**

Stateless Transformations- Processing of the batch does not depend on the output of the previous batch. Examples – map (), reduceByKey (), filter ().

Stateful Transformations- Processing of the batch depends on the intermediary results of the previous batch. Examples –Transformations that depend on sliding windows.

**35) Explain about the popular use cases of Apache Spark**

Apache Spark is mainly used for

Iterative machine learning.

Interactive data analytics and processing.

Stream processing

Sensor data processing

**36) Is Apache Spark a good fit for Reinforcement learning?**

No. Apache Spark works well only for simple machine learning algorithms like clustering, regression, classification.

**37) What is Spark Core?**

It has all the basic functionalities of Spark, like - memory management, fault recovery, interacting with storage systems, scheduling tasks, etc.

**38) How can you remove the elements with a key present in any other RDD?**

Use the subtractByKey () function

**39) What is the difference between persist() and cache()**

persist () allows the user to specify the storage level whereas cache () uses the default storage level.

**40) What are the various levels of persistence in Apache Spark?**

Apache Spark automatically persists the intermediary data from various shuffle operations, however it is often suggested that users call persist () method on the RDD in case they plan to reuse it. Spark has various persistence levels to store the RDDs on disk or in memory or as a combination of both with different replication levels.

The various storage/persistence levels in Spark are -

MEMORY\_ONLY

MEMORY\_ONLY\_SER

MEMORY\_AND\_DISK

MEMORY\_AND\_DISK\_SER, DISK\_ONLY

OFF\_HEAP

**41) How Spark handles monitoring and logging in Standalone mode?**

Spark has a web based user interface for monitoring the cluster in standalone mode that shows the cluster and job statistics. The log output for each job is written to the work directory of the slave nodes.

**42) Does Apache Spark provide check pointing?**

Lineage graphs are always useful to recover RDDs from a failure but this is generally time consuming if the RDDs have long lineage chains. Spark has an API for check pointing i.e. a REPLICATE flag to persist. However, the decision on which data to checkpoint - is decided by the user. Checkpoints are useful when the lineage graphs are long and have wide dependencies.

**43) How can you launch Spark jobs inside Hadoop MapReduce?**

Using SIMR (Spark in MapReduce) users can run any spark job inside MapReduce without requiring any admin rights.

**44) How Spark uses Akka?**

Spark uses Akka basically for scheduling. All the workers request for a task to master after registering. The master just assigns the task. Here Spark uses Akka for messaging between the workers and masters.

**45) How can you achieve high availability in Apache Spark?**

Implementing single node recovery with local file system

Using StandBy Masters with Apache ZooKeeper.

**46) Hadoop uses replication to achieve fault tolerance. How is this achieved in Apache Spark?**

Data storage model in Apache Spark is based on RDDs. RDDs help achieve fault tolerance through lineage. RDD always has the information on how to build from other datasets. If any partition of a RDD is lost due to failure, lineage helps build only that particular lost partition.

**47) Explain about the core components of a distributed Spark application.**

Driver- The process that runs the main () method of the program to create RDDs and perform transformations and actions on them.

Executor –The worker processes that run the individual tasks of a Spark job.

Cluster Manager-A pluggable component in Spark, to launch Executors and Drivers. The cluster manager allows Spark to run on top of other external managers like Apache Mesos or YARN.

**48) What do you understand by Lazy Evaluation?**

Spark is intellectual in the manner in which it operates on data. When you tell Spark to operate on a given dataset, it heeds the instructions and makes a note of it, so that it does not forget - but it does nothing, unless asked for the final result. When a transformation like map () is called on a RDD-the operation is not performed immediately. Transformations in Spark are not evaluated till you perform an action. This helps optimize the overall data processing workflow.

**49)  Define a worker node.**

A node that can run the Spark application code in a cluster can be called as a worker node. A worker node can have more than one worker which is configured by setting the SPARK\_ WORKER\_INSTANCES property in the spark-env.sh file. Only one worker is started if the SPARK\_ WORKER\_INSTANCES property is not defined.

**50) What do you understand by SchemaRDD?**

An RDD that consists of row objects (wrappers around basic string or integer arrays) with schema information about the type of data in each column.

**51) What are the disadvantages of using Apache Spark over Hadoop MapReduce?**

Apache spark does not scale well for compute intensive jobs and consumes large number of system resources. Apache Spark’s in-memory capability at times comes a major roadblock for cost efficient processing of big data. Also, Spark does have its own file management system and hence needs to be integrated with other cloud based data platforms or apache hadoop.

**52) Is it necessary to install spark on all the nodes of a YARN cluster  while running Apache Spark on YARN ?**

No , it is not necessary because Apache Spark runs on top of YARN.

**53) What do you understand by Executor Memory in a Spark application?**

Every spark application has same fixed heap size and fixed number of cores for a spark executor. The heap size is what referred to as the Spark executor memory which is controlled with the spark.executor.memory property of the –executor-memory flag. Every spark application will have one executor on each worker node. The executor memory is basically a measure on how much memory of the worker node will the application utilize.

**54) What does the Spark Engine do?**

Spark engine schedules, distributes and monitors the data application across the spark cluster.

**55) What makes Apache Spark good at low-latency workloads like graph processing and machine learning?**

Apache Spark stores data in-memory for faster model building and training. Machine learning algorithms require multiple iterations to generate a resulting optimal model and similarly graph algorithms traverse all the nodes and edges.These low latency workloads that need multiple iterations can lead to increased performance. Less disk access and  controlled network traffic make a huge difference when there is lots of data to be processed.

**56) Is it necessary to start Hadoop to run any Apache Spark Application ?**

Starting hadoop is not manadatory to run any spark application. As there is no seperate storage in Apache Spark, it uses Hadoop HDFS but it is not mandatory. The data can be stored in local file system, can be loaded from local file system and processed.

**57) What is the default level of parallelism in apache spark?**

If the user does not explicitly specify then the number of partitions are considered as default level of parallelism in Apache Spark.

**58) Explain about the common workflow of a Spark program**

The foremost step in a Spark program involves creating input RDD's from external data.

Use various RDD transformations like filter() to create new transformed RDD's based on the business logic.

persist() any intermediate RDD's which might have to be reused in future.

Launch various RDD actions() like first(), count() to begin parallel computation , which will then be optimized and executed by Spark.

**59) In a given spark program, how will you identify whether a given operation is Transformation or Action ?**

One can identify the operation based on the return type -

i) The operation is an action, if the return type is other than RDD.

ii) The operation is transformation, if the return type is same as the RDD.

**60) What according to you is a common mistake apache spark developers make when using spark ?**

Maintaining the required size of shuffle blocks.

Spark developer often make mistakes with managing directed acyclic graphs (DAG's.)

**61) Suppose that there is an RDD named DeZyrerdd that contains a huge list of numbers.  The following spark code is written to calculate the average -**

**def DeZyreAvg(x, y):  
return (x+y)/2.0;  
avg = DeZyrerdd.reduce(DeZyreAvg);**

**What is wrong with the above code and how will you correct it ?**

Average function is neither commutative nor associative. The best way to compute average is to first sum it and then divide it by count as shown below -

def sum(x, y):  
return x+y;  
total =DeZyrerdd.reduce(sum);  
avg = total / DeZyrerdd.count();

However, the above code could lead to an overflow if the total becomes big. So, the best way to compute average is divide each number by count and then add up as shown below -

cnt = DeZyrerdd.count();  
def divideByCnt(x):  
return x/cnt;  
myrdd1 = DeZyrerdd.map(divideByCnt);  
avg = DeZyrerdd.reduce(sum);

**Q. Say I have a huge list of numbers in a file in HDFS. Each line has one number.And I want to com**

**Spark SQL Interview Questions**

**1) Explain the difference between Spark SQL and Hive.**

Spark SQL is faster than Hive.

Any Hive query can easily be executed in Spark SQL but vice-versa is not true.

Spark SQL is a library whereas Hive is a framework.

It is not mandatory to create a metastore in Spark SQL but it is mandatory to create a Hive metastore.

Spark SQL automatically infers the schema whereas in Hive schema needs to be explicitly declared..

**Spark Streaming Interview Questions**

**1) Name some sources from where Spark streaming component can process real-time data.**

Apache Flume, Apache Kafka, Amazon Kinesis

**2) Name some companies that are already using Spark Streaming.**

Uber, Netflix, Pinterest.

**3) What is the bottom layer of abstraction in the Spark Streaming API ?**

DStream.

**4) What do you understand by receivers in Spark Streaming ?**

Receivers are special entities in Spark Streaming that consume data from various data sources and move them to Apache Spark. Receivers are usually created by streaming contexts as long running tasks on various executors and scheduled to operate in a round robin manner with each receiver taking a single core.

We invite the big data community to share the most frequently asked Apache Spark Interview questions and answers, in the comments below - to ease big data job interviews for all prospective analytics professionals.

**5) How will you calculate the number of executors required to do real-time processing using Apache Spark? What factors need to be connsidered for deciding on the number of nodes for real-time processing?**

The number of nodes can be decided by benchmarking the hardware and considering multiple factors such as optimal throughput (network speed), memory usage, the execution frameworks being used (YARN, Standalone or Mesos) and considering the other jobs that are running within those execution frameworks along with spark.

**6) What is the difference between Spark Transform in DStream and map ?**

tranform function in spark streaming allows developers to use Apache Spark transformations on the underlying RDD's for the stream. map function in hadoop is used for an element to element transform and can be implemented using transform.Ideally , map works on the elements of Dstream and transform allows developers to work with RDD's of the DStream. map is an elementary transformation whereas transform is an RDD transformation.

[Add to Bookmark](https://www.edureka.co/blog/interview-questions/top-apache-spark-interview-questions-2016/)  Email this Post  72.3K [**1**](https://www.edureka.co/blog/interview-questions/top-apache-spark-interview-questions-2016/#disqus_thread)

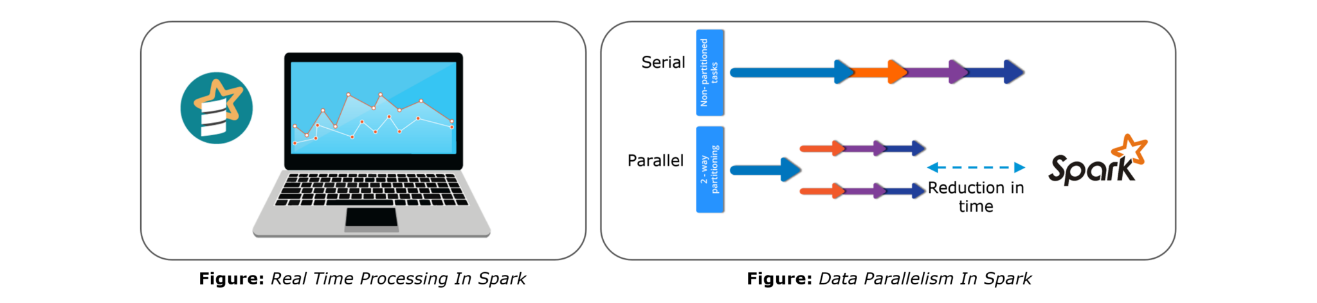
2017 has been the year of Big Data – the year when big data and analytics made tremendous progress through innovative technologies, data-driven decision making and outcome-centric analytics. Worldwide revenues for big data and business analytics (BDA) will grow from $130.1 billion in 2016 to more than $203 billion in 2020 (source IDC). Prepare with these top**Apache Spark Interview Questions** to get an edge in the burgeoning Big Data market where global and local enterprises, big or small, are looking for quality Big Data and Hadoop experts.

As a big data professional, it is essential to know the right buzzwords, learn the right technologies and prepare the right answers to commonly asked Spark interview questions. With questions and answers around Spark Core, Spark Streaming,Spark SQL, GraphX, MLlib among others, this blog is your gateway to your next Spark job.

**Apache Spark Interview Questions And Answers**

**1. What is Apache Spark?**

[***Apache Spark***](https://www.edureka.co/blog/spark-tutorial/) is an open-source cluster computing framework for real-time processing. It has a thriving open-source community and is the most active Apache project at the moment. Spark provides an interface for programming entire clusters with implicit data parallelism and fault-tolerance.

Spark is of the most successful projects in the Apache Software Foundation. Spark has clearly evolved as the market leader for Big Data processing. Many organizations run Spark on clusters with thousands of nodes. Today, Spark is being adopted by major players like Amazon, eBay, and Yahoo!

**2. Compare Hadoop and Spark.**

We will compare Hadoop MapReduce and Spark based on the following aspects:

Apache Spark vs. Hadoop

|  |  |  |
| --- | --- | --- |
| **Feature Criteria** | **Apache Spark** | **Hadoop** |
| **Speed** | 100 times faster than Hadoop | Decent speed |
| **Processing** | Real-time & Batch processing | Batch processing only |
| **Difficulty** | Easy because of high level modules | Tough to learn |
| **Recovery** | Allows recovery of partitions | Fault-tolerant |
| **Interactivity** | Has interactive modes | No interactive mode except Pig & Hive |

**Table:** Apache Spark versus Hadoop

Let us understand the same using an interesting analogy.

“Single cook cooking an entree is regular computing. Hadoop is multiple cooks cooking an entree into pieces and letting each cook her piece.Each cook has a separate stove and a food shelf. The first cook cooks the meat, the second cook cooks the sauce. This phase is called “Map”. A the end the main cook assembles the complete entree. This is called “Reduce”. For Hadoop, the cooks are not allowed to keep things on the stove between operations. Each time you make a particular operation, the cook puts results on the shelf. This slows things down.For Spark, the cooks are allowed to keep things on the stove between operations. This speeds things up. Finally, for Hadoop the recipes are written in a language which is illogical and hard to understand. For Spark, the recipes are nicely written.” – Stan Kladko*, Galactic Exchange.io*

[**GET STARTED WITH SPARK**](https://www.edureka.co/apache-spark-scala-training)

**3. Explain the key features of Apache Spark.**

The following are the key features of Apache Spark:

**Polyglot**

**Speed**

**Multiple Format Support**

**Lazy Evaluation**

**Real Time Computation**

**Hadoop Integration**

**Machine Learning**

Let us look at these features in detail:

**Polyglot**: Spark provides high-level APIs in Java, Scala, Python and R. Spark code can be written in any of these four languages. It provides a shell in Scala and Python. The Scala shell can be accessed through **./bin/spark-shell** and Python shell through **./bin/pyspark** from the installed directory.

**Speed**: Spark runs upto 100 times faster than Hadoop MapReduce for large-scale data processing. Spark is able to achieve this speed through controlled partitioning. It manages data using partitions that help parallelize distributed data processing with minimal network traffic.

**Multiple Formats**: Spark supports multiple data sources such as Parquet, JSON, Hive and Cassandra. The Data Sources API provides a pluggable mechanism for accessing structured data though Spark SQL. Data sources can be more than just simple pipes that convert data and pull it into Spark.

**Lazy Evaluation**: Apache Spark delays its evaluation till it is absolutely necessary. This is one of the key factors contributing to its speed. For transformations, Spark adds them to a DAG of computation and only when the driver requests some data, does this DAG actually gets executed.

**Real Time Computation**: Spark’s computation is real-time and has less latency because of its in-memory computation. Spark is designed for massive scalability and the Spark team has documented users of the system running production clusters with thousands of nodes and supports several computational models.

**Hadoop Integration**: Apache Spark provides smooth compatibility with Hadoop. This is a great boon for all the Big Data engineers who started their careers with Hadoop. Spark is a potential replacement for the MapReduce functions of Hadoop, while Spark has the ability to run on top of an existing Hadoop cluster using YARN for resource scheduling.

**Machine Learning**: Spark’s MLlib is the machine learning component which is handy when it comes to big data processing. It eradicates the need to use multiple tools, one for processing and one for machine learning. Spark provides data engineers and data scientists with a powerful, unified engine that is both fast and easy to use.

**4. What are the languages supported by Apache Spark and which is the most popular one?**

Apache Spark supports the following four languages: Scala, Java, Python and R. Among these languages, Scala and Python have interactive shells for Spark. The Scala shell can be accessed through **./bin/spark-shell**and the Python shell through **./bin/pyspark**. Scala is the most used among them because Spark is written in Scala and it is the most popularly used for Spark.

**5. What are benefits of Spark over MapReduce?**

Spark has the following benefits over MapReduce:

Due to the availability of in-memory processing, Spark implements the processing around 10 to 100 times faster than Hadoop MapReduce whereas MapReduce makes use of persistence storage for any of the data processing tasks.

Unlike Hadoop, Spark provides inbuilt libraries to perform multiple tasks from the same core like batch processing, Steaming, Machine learning, Interactive SQL queries. However, Hadoop only supports batch processing.

Hadoop is highly disk-dependent whereas Spark promotes caching and in-memory data storage.

Spark is capable of performing computations multiple times on the same dataset. This is called iterative computation while there is no iterative computing implemented by Hadoop.

**6. What is Yarn?**

Similar to Hadoop, Yarn is one of the key features in Spark, providing a central and resource management platform to deliver scalable operations across the cluster. Yarn is a distributed container manager, like Mesos for example, whereas Spark is a data processing tool. Spark can run on Yarn, the same way Hadoop Map Reduce can run on Yarn. Running Spark on Yarn necessitates a binary distribution of Spark as built on Yarn support.

**7. Do you need to install Spark on all nodes of Yarn cluster?**

No, because Spark runs on top of Yarn. Spark runs independently from its installation. Spark has some options to use YARN when dispatching jobs to the cluster, rather than its own built-in manager, or Mesos. Further, there are some configurations to run Yarn. They include master, deploy-mode, driver-memory, executor-memory, executor-cores, and queue.

**8. Is there any benefit of learning MapReduce if Spark is better than MapReduce?**

Yes, MapReduce is a paradigm used by many big data tools including Spark as well. It is extremely relevant to use MapReduce when the data grows bigger and bigger. Most tools like Pig and Hive convert their queries into MapReduce phases to optimize them better.

**9. Explain the concept of Resilient Distributed Dataset (RDD).**

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

Parallelized Collections: Here, the existing RDDs running parallel with one another.

Hadoop Datasets: They perform functions on each file record in HDFS or other storage systems.

RDDs are basically parts of data that are stored in the memory distributed across many nodes. RDDs are lazily evaluated in Spark. This lazy evaluation is what contributes to Spark’s speed.

**10. How do we create RDDs in Spark?**

Spark provides two methods to create RDD:

1. By parallelizing a collection in your Driver program.

2. This makes use of SparkContext’s ‘parallelize’

|  |  |
| --- | --- |
| 1  2  3 | method val DataArray = Array(2,4,6,8,10)    val DataRDD = sc.parallelize(DataArray) |

3. By loading an external dataset from external storage like HDFS, HBase, shared file system.

**11. What is Executor Memory in a Spark application?**

Every spark application has same fixed heap size and fixed number of cores for a spark executor. The heap size is what referred to as the Spark executor memory which is controlled with the spark.executor.memory property of the **–executor-memory** flag. Every spark application will have one executor on each worker node. The executor memory is basically a measure on how much memory of the worker node will the application utilize.

**12. Define Partitions in Apache Spark.**

As the name suggests, partition is a smaller and logical division of data similar to ‘split’ in MapReduce. It is a logical chunk of a large distributed data set. Partitioning is the process to derive logical units of data to speed up the processing process. Spark manages data using partitions that help parallelize distributed data processing with minimal network traffic for sending data between executors. By default, Spark tries to read data into an RDD from the nodes that are close to it. Since Spark usually accesses distributed partitioned data, to optimize transformation operations it creates partitions to hold the data chunks. Everything in Spark is a partitioned RDD.

**13. What operations does RDD support?**

RDD (Resilient Distributed Dataset) is main logical data unit in Spark. An RDD has distributed a collection of objects. Distributed means, each RDD is divided into multiple partitions. Each of these partitions can reside in memory or stored on the disk of different machines in a cluster. RDDs are immutable (Read Only) data structure. You can’t change original RDD, but you can always transform it into different RDD with all changes you want.

RDDs support two types of operations: transformations and actions.

Transformations: Transformations create new RDD from existing RDD like map, reduceByKey and filter we just saw. Transformations are executed on demand. That means they are computed lazily.

Actions: Actions return final results of RDD computations. Actions triggers execution using lineage graph to load the data into original RDD, carry out all intermediate transformations and return final results to Driver program or write it out to file system.

**14. What do you understand by Transformations in Spark?**

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

|  |  |
| --- | --- |
| 1  2  3 | val rawData=sc.textFile("path to/movies.txt")    val moviesData=rawData.map(x=>x.split("\t")) |

As we can see here, rawData RDD is transformed into moviesData RDD. Transformations are lazily evaluated.

**15. Define Actions in Spark.**

An action helps in bringing back the data from RDD to the local machine. An action’s execution is the result of all previously created transformations. Actions triggers execution using lineage graph to load the data into original RDD, carry out all intermediate transformations and return final results to Driver program or write it out to file system.

reduce() is an action that implements the function passed again and again until one value if left. take() action takes all the values from RDD to a local node.

|  |  |
| --- | --- |
| 1 | moviesData.saveAsTextFile(“MoviesData.txt”) |

As we can see here, moviesData RDD is saved into a text file called MoviesData.txt.

**16. Define functions of SparkCore.**

Spark Core is the base engine for large-scale parallel and distributed data processing. The core is the distributed execution engine and the Java, Scala, and Python APIs offer a platform for distributed ETL application development. SparkCore performs various important functions like memory management, monitoring jobs, fault-tolerance, job scheduling and interaction with storage systems. Further, additional libraries, built atop the core allow diverse workloads for streaming, SQL, and machine learning. It is responsible for:

Memory management and fault recovery

Scheduling, distributing and monitoring jobs on a cluster

Interacting with storage systems

**17. What do you understand by Pair RDD?**

Apache defines PairRDD functions class as

|  |  |
| --- | --- |
| 1 | class PairRDDFunctions[K, V] extends Logging with HadoopMapReduceUtil with Serializable |

Special operations can be performed on RDDs in Spark using key/value pairs and such RDDs are referred to as Pair RDDs. Pair RDDs allow users to access each key in parallel. They have a reduceByKey() method that collects data based on each key and a join() method that combines different RDDs together, based on the elements having the same key.

**18. Name the components of Spark Ecosystem.**

**Spark Core**: Base engine for large-scale parallel and distributed data processing

**Spark Streaming**: Used for processing real-time streaming data

**Spark SQL**: Integrates relational processing with Spark’s functional programming API

**GraphX**: Graphs and graph-parallel computation

**MLlib**: Performs machine learning in Apache Spark

**19. How is Streaming implemented in Spark? Explain with examples.**

Spark Streaming is used for processing real-time streaming data. Thus it is a useful addition to the core Spark API. It enables high-throughput and fault-tolerant stream processing of live data streams. The fundamental stream unit is DStream which is basically a series of RDDs (Resilient Distributed Datasets) to process the real-time data. The data from different sources like Flume, HDFS is streamed and finally processed to file systems, live dashboards and databases. It is similar to batch processing as the input data is divided into streams like batches.

****

**Figure:** Spark Interview Questions – Spark Streaming

**20. Is there an API for implementing graphs in Spark?**

GraphX is the Spark API for graphs and graph-parallel computation. Thus, it extends the Spark RDD with a Resilient Distributed Property Graph.

The property graph is a directed multi-graph which can have multiple edges in parallel. Every edge and vertex have user defined properties associated with it. Here, the parallel edges allow multiple relationships between the same vertices. At a high-level, GraphX extends the Spark RDD abstraction by introducing the Resilient Distributed Property Graph: a directed multigraph with properties attached to each vertex and edge.

To support graph computation, GraphX exposes a set of fundamental operators (e.g., subgraph, joinVertices, and mapReduceTriplets) as well as an optimized variant of the Pregel API. In addition, GraphX includes a growing collection of graph algorithms and builders to simplify graph analytics tasks.

**21. What is PageRank in GraphX?**

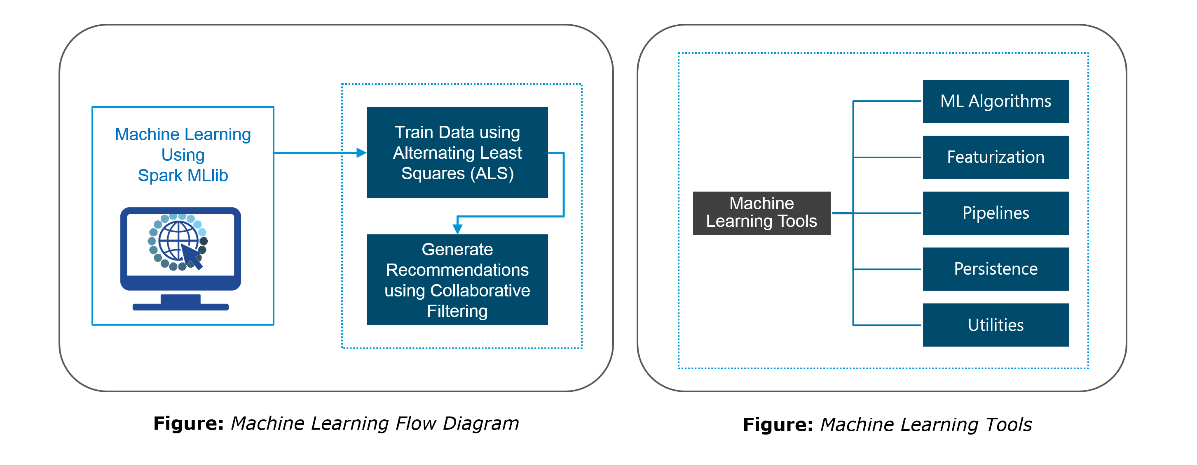
PageRank measures the importance of each vertex in a graph, assuming an edge from u to v represents an endorsement of v’s importance by u. For example, if a Twitter user is followed by many others, the user will be ranked highly.

GraphX comes with static and dynamic implementations of PageRank as methods on the PageRank Object. Static PageRank runs for a fixed number of iterations, while dynamic PageRank runs until the ranks converge (i.e., stop changing by more than a specified tolerance). GraphOps allows calling these algorithms directly as methods on Graph.

[**LEARN SPARK FROM EXPERTS**](https://www.edureka.co/apache-spark-scala-training)

**22. How is machine learning implemented in Spark?**

MLlib is scalable machine learning library provided by Spark. It aims at making machine learning easy and scalable with common learning algorithms and use cases like clustering, regression filtering, dimensional reduction, and alike.

****

**23. Is there a module to implement SQL in Spark? How does it work?**

Spark SQL is a new module in Spark which integrates relational processing with Spark’s functional programming API. It supports querying data either via SQL or via the Hive Query Language. For those of you familiar with RDBMS, Spark SQL will be an easy transition from your earlier tools where you can extend the boundaries of traditional relational data processing.

Spark SQL integrates relational processing with Spark’s functional programming. Further, it provides support for various data sources and makes it possible to weave SQL queries with code transformations thus resulting in a very powerful tool.

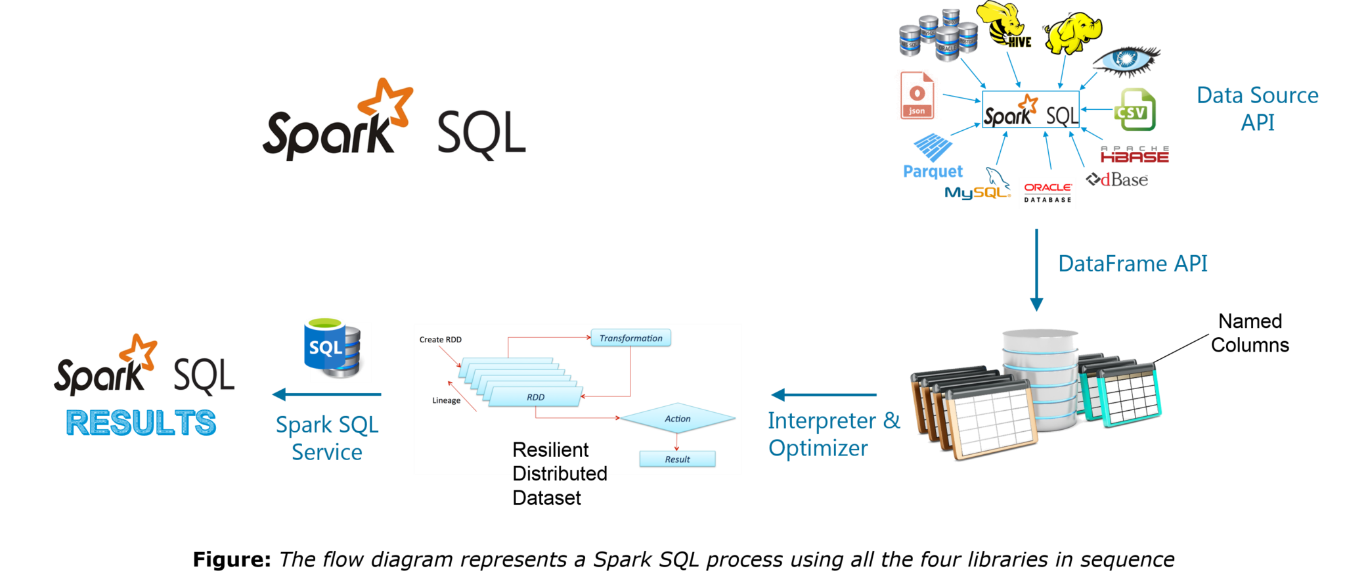
The following are the four libraries of Spark SQL.

Data Source API

DataFrame API

Interpreter & Optimizer

SQL Service

**24. What is a Parquet file?**

Parquet is a columnar format file supported by many other data processing systems. Spark SQL performs both read and write operations with Parquet file and consider it be one of the best big data analytics formats so far.

Parquet is a columnar format, supported by many data processing systems. The advantages of having a columnar storage are as follows:

Columnar storage limits IO operations.

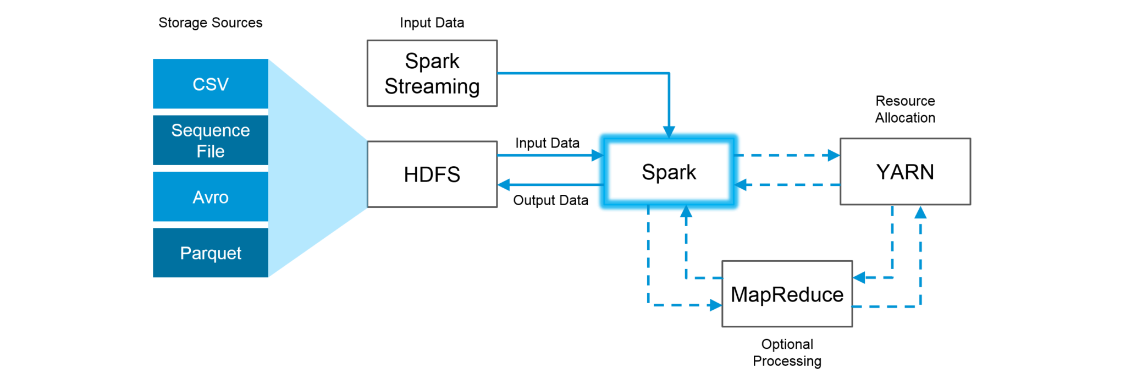
It can fetch specific columns that you need to access.

Columnar storage consumes less space.

It gives better-summarized data and follows type-specific encoding.

**25. How can Apache Spark be used alongside Hadoop?**

The best part of Apache Spark is its compatibility with Hadoop. As a result, this makes for a very powerful combination of technologies. Here, we will be looking at how Spark can benefit from the best of Hadoop. Using Spark and Hadoop together helps us to leverage Spark’s processing to utilize the best of Hadoop’s HDFS and YARN.

**Figure:**Using Spark and Hadoop

Hadoop components can be used alongside Spark in the following ways:

**HDFS**: Spark can run on top of HDFS to leverage the distributed replicated storage.

**MapReduce**: Spark can be used along with MapReduce in the same Hadoop cluster or separately as a processing framework.

**YARN**: Spark applications can also be run on YARN (Hadoop NextGen).

**Batch & Real Time Processing**: MapReduce and Spark are used together where MapReduce is used for batch processing and Spark for real-time processing.

**26. What is RDD Lineage?**

Spark does not support data replication in the memory and thus, if any data is lost, it is rebuild using RDD lineage. RDD lineage is a process that reconstructs lost data partitions. The best is that RDD always remembers how to build from other datasets.

**27. What is Spark Driver?**

Spark Driver is the program that runs on the master node of the machine and declares transformations and actions on data RDDs. In simple terms, a driver in Spark creates SparkContext, connected to a given Spark Master.  
The driver also delivers the RDD graphs to Master, where the standalone cluster manager runs.

**28. What file systems does Spark support?**

The following three file systems are supported by Spark:

Hadoop Distributed File System (HDFS).

Local File system.

Amazon S3

**29. List the functions of Spark SQL.**

Spark SQL is capable of:

Loading data from a variety of structured sources.

Querying data using SQL statements, both inside a Spark program and from external tools that connect to Spark SQL through standard database connectors (JDBC/ODBC). For instance, using business intelligence tools like Tableau.

Providing rich integration between SQL and regular Python/Java/Scala code, including the ability to join RDDs and SQL tables, expose custom functions in SQL, and more.

**30. What is Spark Executor?**

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

**31. Name types of Cluster Managers in Spark.**

The Spark framework supports three major types of Cluster Managers:

**Standalone**: A basic manager to set up a cluster.

**Apache Mesos**: Generalized/commonly-used cluster manager, also runs Hadoop MapReduce and other applications.

**Yarn**: Responsible for resource management in Hadoop.

**32. What do you understand by worker node?**

Worker node refers to any node that can run the application code in a cluster. The driver program must listen for and accept incoming connections from its executors and must be network addressable from the worker nodes.

Worker node is basically the slave node. Master node assigns work and worker node actually performs the assigned tasks. Worker nodes process the data stored on the node and report the resources to the master. Based on the resource availability, the master schedule tasks.

**33. Illustrate some demerits of using Spark.**

The following are some of the demerits of using Apache Spark:

Since Spark utilizes more storage space compared to Hadoop and MapReduce, there may arise certain problems.

Developers need to be careful while running their applications in Spark.

Instead of running everything on a single node, the work must be distributed over multiple clusters.

Spark’s “in-memory” capability can become a bottleneck when it comes to cost-efficient processing of big data.

Spark consumes a huge amount of data when compared to Hadoop.

**34. List some use cases where Spark outperforms Hadoop in processing.**

**Sensor Data Processing**: Apache Spark’s “In-memory” computing works best here, as data is retrieved and combined from different sources.

**Real Time Processing**: Spark is preferred over Hadoop for real-time querying of data. e.g. Stock Market Analysis,Banking, Healthcare, Telecommunications, etc.

**Stream Processing**: For processing logs and detecting frauds in live streams for alerts, Apache Spark is the best solution.

**Big Data Processing**:Spark runs upto 100 times faster than Hadoop when it comes to processing medium and large-sized datasets.

**35. What is a Sparse Vector?**

A sparse vector has two parallel arrays; one for indices and the other for values. These vectors are used for storing non-zero entries to save space.

|  |  |
| --- | --- |
| 1 | Vectors.sparse(7,Array(0,1,2,3,4,5,6),Array(1650d,50000d,800d,3.0,3.0,2009,95054)) |

The above sparse vector can be used instead of dense vectors.

|  |  |
| --- | --- |
| 1 | val myHouse = Vectors.dense(4450d,2600000d,4000d,4.0,4.0,1978.0,95070d,1.0,1.0,1.0,0.0) |

**36. Can you use Spark to access and analyze data stored in Cassandra databases?**

Yes, it is possible if you use Spark Cassandra Connector.To connect Spark to a Cassandra cluster, a Cassandra Connector will need to be added to the Spark project. In the setup, a Spark executor will talk to a local Cassandra node and will only query for local data. It makes queries faster by reducing the usage of the network to send data between Spark executors (to process data) and Cassandra nodes (where data lives).

**37. Is it possible to run Apache Spark on Apache Mesos?**

Yes, Apache Spark can be run on the hardware clusters managed by Mesos. In a standalone cluster deployment, the cluster manager in the below diagram is a Spark master instance. When using Mesos, the Mesos master replaces the Spark master as the cluster manager. Mesos determines what machines handle what tasks. Because it takes into account other frameworks when scheduling these many short-lived tasks, multiple frameworks can coexist on the same cluster without resorting to a static partitioning of resources.

**38. How can Spark be connected to Apache Mesos?**

To connect Spark with Mesos:

Configure the spark driver program to connect to Mesos.

Spark binary package should be in a location accessible by Mesos.

Install Apache Spark in the same location as that of Apache Mesos and configure the property ‘spark.mesos.executor.home’ to point to the location where it is installed.

**39. How can you minimize data transfers when working with Spark?**

Minimizing data transfers and avoiding shuffling helps write spark programs that run in a fast and reliable manner. The various ways in which data transfers can be minimized when working with Apache Spark are:

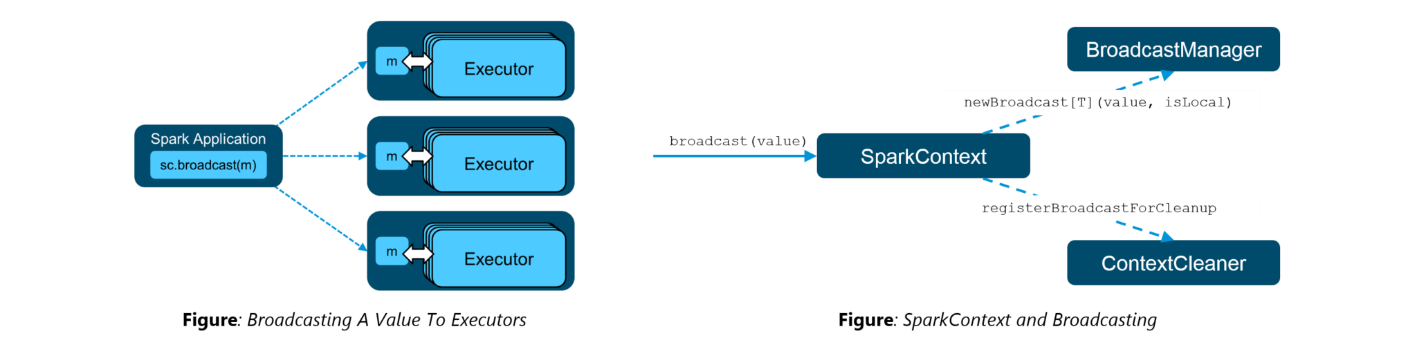
Using Broadcast Variable- Broadcast variable enhances the efficiency of joins between small and large RDDs.

Using Accumulators – Accumulators help update the values of variables in parallel while executing.

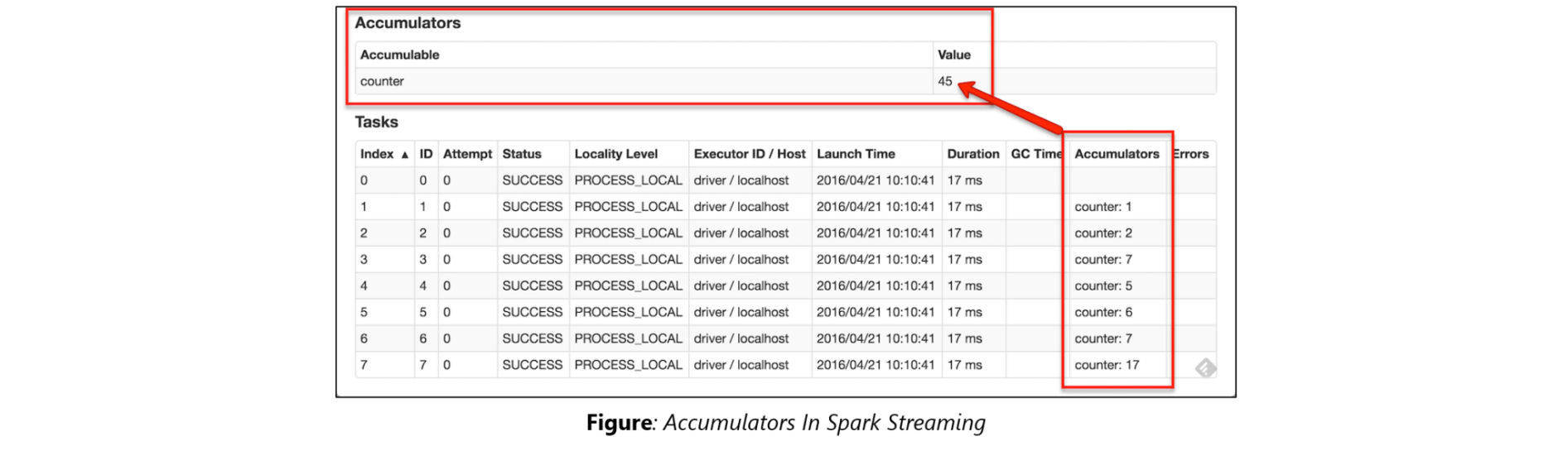
The most common way is to avoid operations ByKey, repartition or any other operations which trigger shuffles.

**40. What are broadcast variables?**

Broadcast variables allow the programmer to keep a read-only variable cached on each machine rather than shipping a copy of it with tasks. They can be used to give every node a copy of a large input dataset in an efficient manner. Spark also attempts to distribute broadcast variables using efficient broadcast algorithms to reduce communication cost.

**41. Explain accumulators in Apache Spark.**

Accumulators are variables that are only added through an associative and commutative operation. They are used to implement counters or sums. Tracking accumulators in the UI can be useful for understanding the progress of running stages. Spark natively supports numeric accumulators. We can create named or unnamed accumulators.

****

**42. Why is there a need for broadcast variables when working with Apache Spark?**

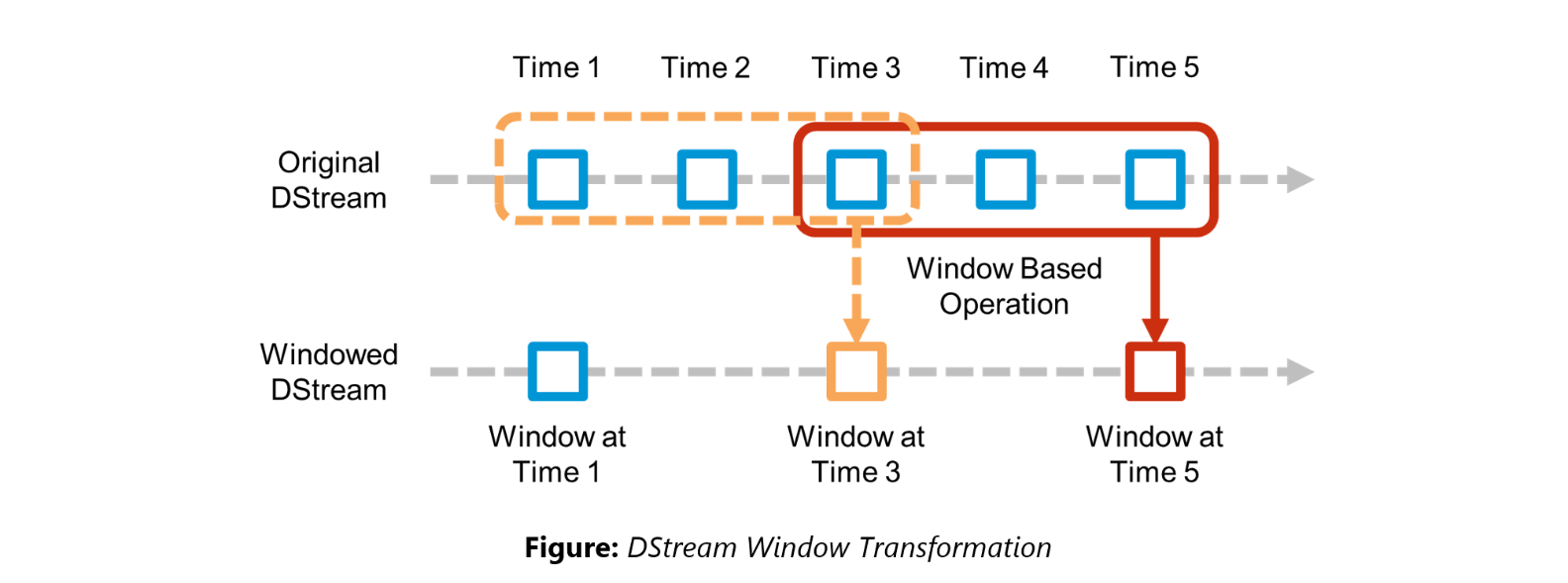
Broadcast variables are read only variables, present in-memory cache on every machine. When working with Spark, usage of broadcast variables eliminates the necessity to ship copies of a variable for every task, so data can be processed faster. Broadcast variables help in storing a lookup table inside the memory which enhances the retrieval efficiency when compared to an RDD lookup().

**43. How can you trigger automatic clean-ups in Spark to handle accumulated metadata?**

You can trigger the clean-ups by setting the parameter ‘spark.cleaner.ttl’ or by dividing the long running jobs into different batches and writing the intermediary results to the disk.

**44. What is the significance of Sliding Window operation?**

Sliding Window controls transmission of data packets between various computer networks. Spark Streaming library provides windowed computations where the transformations on RDDs are applied over a sliding window of data. Whenever the window slides, the RDDs that fall within the particular window are combined and operated upon to produce new RDDs of the windowed DStream.

**45. What is a DStream in Apache Spark?**

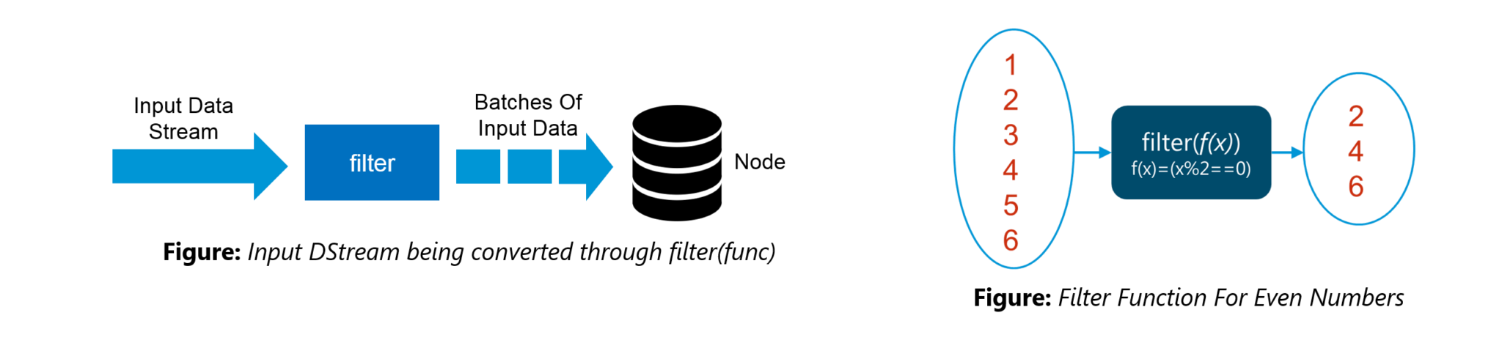
***Discretized Stream***(DStream) is the basic abstraction provided by Spark Streaming. It is a continuous stream of data. It is received from a data source or from a processed data stream generated by transforming the input stream. Internally, a DStream is represented by a continuous series of RDDs and each RDD contains data from a certain interval. Any operation applied on a DStream translates to operations on the underlying RDDs.

DStreams can be created from various sources like Apache Kafka, HDFS, and Apache Flume. DStreams have two operations:

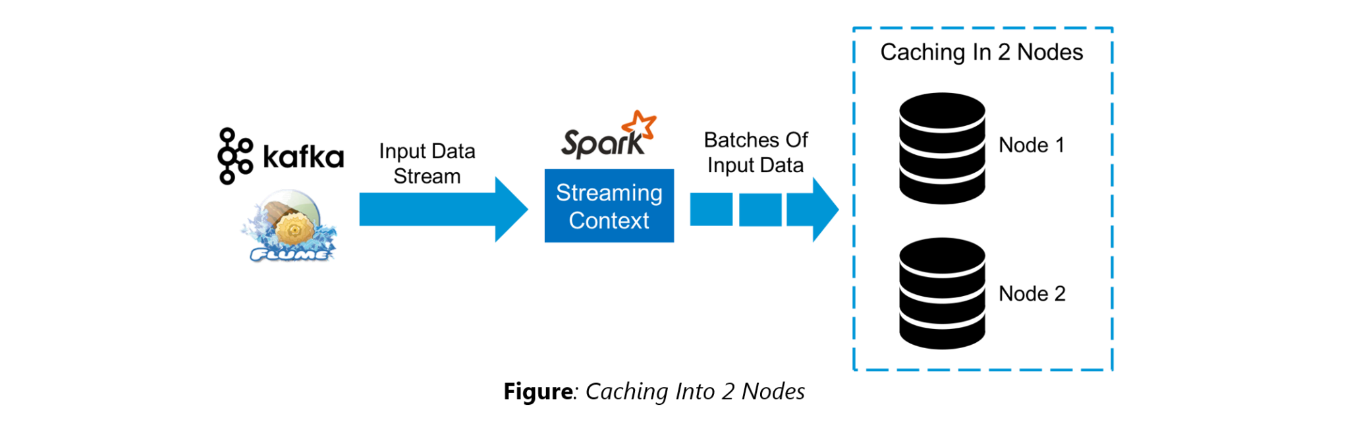
Transformations that produce a new DStream.

Output operations that write data to an external system.

There are many DStream transformations possible in Spark Streaming. Let us look at **filter(*func*)**. filter(*func*) returns a new DStream by selecting only the records of the source DStream on which func returns true.

**46. Explain Caching in Spark Streaming.**

DStreams allow developers to cache/ persist the stream’s data in memory. This is useful if the data in the DStream will be computed multiple times. This can be done using the persist() method on a DStream. For input streams that receive data over the network (such as Kafka, Flume, Sockets, etc.), the default persistence level is set to replicate the data to two nodes for fault-tolerance.

****

**47. When running Spark applications, is it necessary to install Spark on all the nodes of YARN cluster?**

Spark need not be installed when running a job under YARN or Mesos because Spark can execute on top of YARN or Mesos clusters without affecting any change to the cluster.

**48. What are the various data sources available in Spark SQL?**

Parquet file, JSON datasets and Hive tables are the data sources available in Spark SQL.

**49. What are the various levels of persistence in Apache Spark?**

Apache Spark automatically persists the intermediary data from various shuffle operations, however, it is often suggested that users call persist () method on the RDD in case they plan to reuse it. Spark has various persistence levels to store the RDDs on disk or in memory or as a combination of both with different replication levels.

The various storage/persistence levels in Spark are:

MEMORY\_ONLY: Store RDD as deserialized Java objects in the JVM. If the RDD does not fit in memory, some partitions will not be cached and will be recomputed on the fly each time they’re needed. This is the default level.

MEMORY\_AND\_DISK: Store RDD as deserialized Java objects in the JVM. If the RDD does not fit in memory, store the partitions that don’t fit on disk, and read them from there when they’re needed.

MEMORY\_ONLY\_SER: Store RDD as *serialized* Java objects (one byte array per partition).

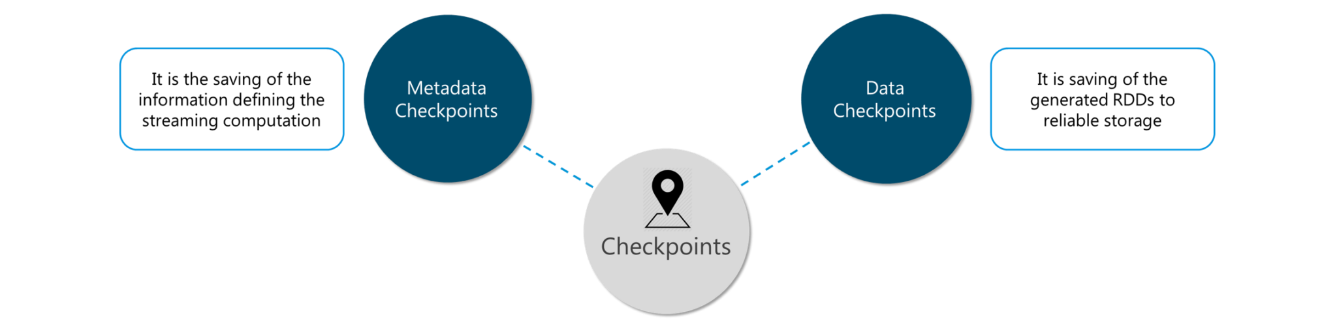
MEMORY\_AND\_DISK\_SER: Similar to MEMORY\_ONLY\_SER, but spill partitions that don’t fit in memory to disk instead of recomputing them on the fly each time they’re needed.

DISK\_ONLY: Store the RDD partitions only on disk.

OFF\_HEAP: Similar to MEMORY\_ONLY\_SER, but store the data in off-heap memory.

**50. Does Apache Spark provide checkpoints?**

Checkpoints are similar to checkpoints in gaming. They make it run 24/7 and make it resilient to failures unrelated to the application logic.

**Figure:** Spark Interview Questions – Checkpoints

Lineage graphs are always useful to recover RDDs from a failure but this is generally time-consuming if the RDDs have long lineage chains. Spark has an API for checkpointing i.e. a REPLICATE flag to persist. However, the decision on which data to checkpoint – is decided by the user. Checkpoints are useful when the lineage graphs are long and have wide dependencies.

**51. How Spark uses Akka?**

Spark uses Akka basically for scheduling. All the workers request for a task to master after registering. The master just assigns the task. Here Spark uses Akka for messaging between the workers and masters.

**52. What do you understand by Lazy Evaluation?**

Spark is intellectual in the manner in which it operates on data. When you tell Spark to operate on a given dataset, it heeds the instructions and makes a note of it, so that it does not forget – but it does nothing, unless asked for the final result. When a transformation like map() is called on an RDD, the operation is not performed immediately. Transformations in Spark are not evaluated till you perform an action. This helps optimize the overall data processing workflow.

**53. What do you understand by SchemaRDD in Apache Spark RDD?**

SchemaRDD is an RDD that consists of row objects (wrappers around the basic string or integer arrays) with schema information about the type of data in each column.

SchemaRDD was designed as an attempt to make life easier for developers in their daily routines of code debugging and unit testing on SparkSQL core module. The idea can boil down to describing the data structures inside RDD using a formal description similar to the relational database schema. On top of all basic functions provided by common RDD APIs, SchemaRDD also provides some straightforward relational query interface functions that are realized through SparkSQL.

Now, it is officially renamed to DataFrame API on Spark’s latest trunk.

**54. How is Spark SQL different from HQL and SQL?**

Spark SQL is a special component on the Spark Core engine that supports SQL and Hive Query Language without changing any syntax. It is possible to join SQL table and HQL table to Spark SQL.

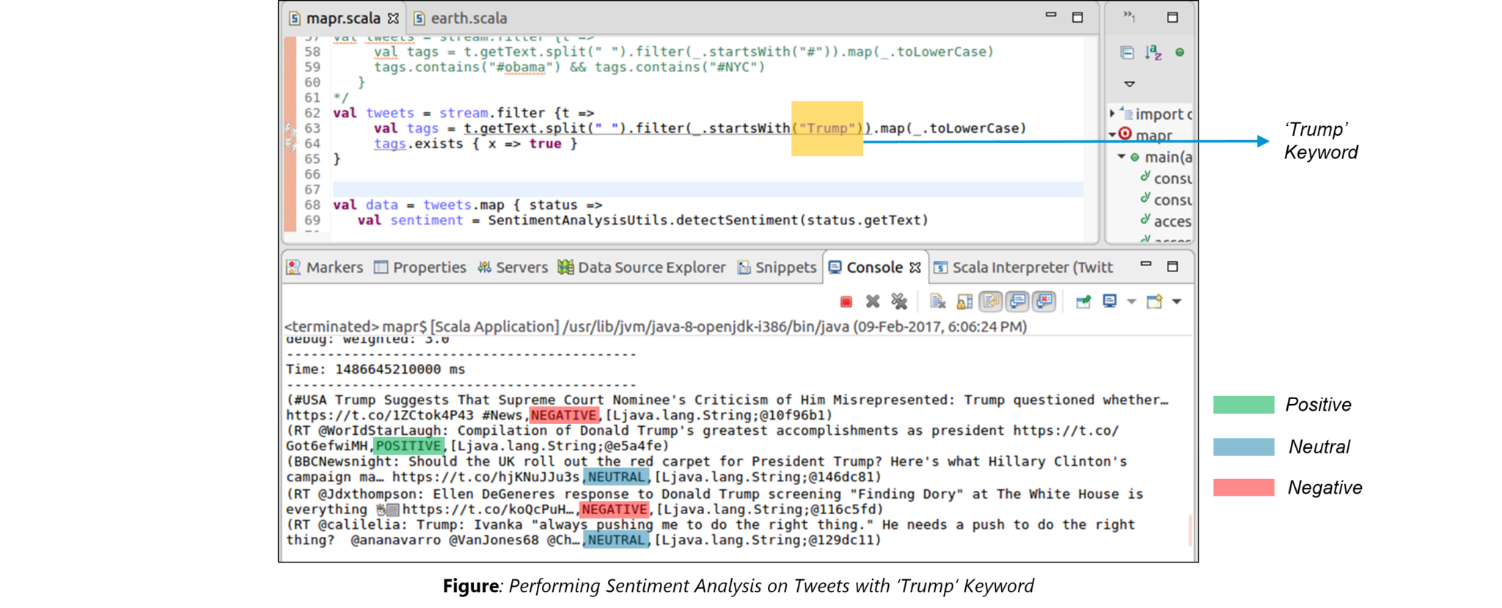
**55. Explain a scenario where you will be using Spark Streaming.**

When it comes to Spark Streaming, the data is streamed in real-time onto our Spark program.

Twitter Sentiment Analysis is a real-life use case of Spark Streaming. Trending Topics can be used to create campaigns and attract a larger audience. It helps in crisis management, service adjusting and target marketing.

Sentiment refers to the emotion behind a social media mention online. Sentiment Analysis is categorizing the tweets related to a particular topic and performing data mining using Sentiment Automation Analytics Tools.

Spark Streaming can be used to gather live tweets from around the world into the Spark program. This stream can be filtered using Spark SQL and then we can filter tweets based on the sentiment. The filtering logic will be implemented using MLlib where we can learn from the emotions of the public and change our filtering scale accordingly.



**1. Compare MapReduce and Spark?**

|  |  |  |
| --- | --- | --- |
| **Criteria** | **MapReduce** | **Spark** |
| Processing Speeds | Good | Excellent (up to 100 times faster) |
| Data caching | Hard disk | In-memory |
| Perform iterative jobs | Average | Excellent |
| Independent of Hadoop | No | Yes |
| Machine learning applications | Average | Excellent |

**2. What is Apache Spark?**

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

**Go through this insightful blog to learn in detail about**[**what is Apache Spark**](https://intellipaat.com/blog/what-is-apache-spark/)**?**

**3. Explain key features of Spark.**

**Allows Integration with Hadoop and files included in HDFS.**

**Spark has an interactive language shell as it has an independent Scala (the language in which Spark is written) interpreter.**

**Spark consists of RDD’s (Resilient Distributed Datasets), which can be cached across computing nodes in a cluster.**

**Spark supports multiple analytic tools that are used for interactive query analysis , real-time analysis and graph processing**

Learn more about the Spark key features in this [Apache Spark Tutorial](https://intellipaat.com/tutorial/spark-tutorial/).

**4. Define RDD?**

RDD is the acronym for Resilient Distribution Datasets – a fault-tolerant collection of operational elements that run parallel. The partitioned data in RDD is immutable and distributed. There are primarily two types of RDD:

**Parallelized Collections :** The existing RDD’s running parallel with one another.

**Hadoop datasets :** perform function on each file record in HDFS or other storage system

**5. What does a Spark Engine do?**

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

**Find out more about what the Spark Engine does in this**[Apache Spark Video](https://intellipaat.com/apache-spark-scala-training/#course-preview)**.**

**6. Define Partitions?**

As the name suggests, partition is a smaller and logical division of data  similar to ‘split’ in MapReduce. Partitioning is the process to derive logical units of data to speed up the processing process. Everything in Spark is a partitioned RDD.

**7. What operations RDD support?**

Transformations.

Actions

**8. What do you understand by Transformations in Spark?**

Transformations are functions applied on RDD, resulting into another RDD. It does not execute until an action occurs. map() and filer() are examples of transformations, where the former applies the function passed to it on each element of RDD and results into another RDD. The filter() creates a new RDD by selecting elements form current RDD that pass function argument.

**9. Define Actions.**

An action helps in bringing back the data from RDD to the local machine. An action’s execution is the result of all previously created transformations. reduce() is an action that implements the function passed again and again until one value if left. take() action takes all the values from RDD to local node.

**10. Define functions of SparkCore?**

Serving as the base engine, SparkCore performs various important functions like memory management, monitoring jobs, fault-tolerance, job scheduling and interaction with storage systems.

Download Spark Interview Questions asked by top MNCs in 2018

**GET PDF**

**11. What is RDD Lineage?**

Spark does not support data replication in the memory and thus, if any data is lost, it is rebuild using RDD lineage. RDD lineage is a process that reconstructs lost data partitions. The best is that RDD always remembers how to build from other datasets.

**12. What is Spark Driver?**

Spark Driver is the program that runs on the master node of the machine and declares transformations and actions on data RDDs. In simple terms, driver in Spark creates SparkContext, connected to a given Spark Master.  
The driver also delivers the RDD graphs to Master, where the standalone cluster manager runs.

**Are you interested in the comprehensive**[*Apache Spark and Scala Videos*](https://intellipaat.com/apache-spark-scala-training/#course-preview)**to take your career to the next level?**

**13. What is Hive on Spark?**

Hive contains significant support for Apache Spark, wherein Hive execution is configured to Spark:

hive> set spark.home=/location/to/sparkHome;

hive> set hive.execution.engine=spark;

Hive on Spark supports Spark on yarn mode by default.

**14. Name commonly-used Spark Ecosystems.**

Spark SQL (Shark)- for developers.

Spark Streaming for processing live data streams.

GraphX for generating and computing graphs.

MLlib (Machine Learning Algorithms).

SparkR to promote R Programming in Spark engine.

**15. Define Spark Streaming.**

Spark supports stream processing – an extension to the Spark API , allowing stream processing of live data streams. The data from different sources like Flume, HDFS is streamed and finally processed to file systems, live dashboards and databases. It is similar to batch processing as the input data is divided into streams like batches.

Learn in detail about [Top four Spark use cases](https://intellipaat.com/blog/top-4-apache-spark-use-cases/) including Spark streaming.

**16. What is GraphX?**

Spark uses GraphX for graph processing to build and transform interactive graphs. The GraphX component enables programmers to reason about structured data at scale.

**17. What does MLlib do?**

MLlib is scalable machine learning library provided by Spark. It aims at making machine learning easy and scalable with common learning algorithms and use cases like clustering, regression filtering, dimensional reduction, and alike.

Our in-depth[Scala Certification](https://intellipaat.com/apache-spark-scala-training/" \l "certification" \t "_blank) Course can give your career a big boost!

**18. What is Spark SQL?**

SQL Spark, better known as Shark is a novel module introduced in Spark to work with structured data and perform structured data processing. Through this module, Spark executes relational SQL queries on the data. The core of the component supports an altogether different RDD called SchemaRDD, composed of rows objects and schema objects defining data type of each column in the row. It is similar to a table in relational database.

**19. What is a Parquet file?**

Parquet is a columnar format file supported by many other data processing systems. Spark SQL performs both read and write operations with Parquet file and consider it be one of the best big data analytics format so far.

**20. What file systems Spark support?**

Hadoop Distributed File System (HDFS).

Local File system.

S3

**21. What is Yarn?**

Similar to Hadoop, Yarn is one of the key features in Spark, providing a central and resource management platform to deliver scalable operations across the cluster . Running Spark on Yarn necessitates a binary distribution of Spar as built on Yarn support.

**22. List the functions of Spark SQL.?**

Spark SQL is capable of:

Loading data from a variety of structured sources.

Querying data using SQL statements, both inside a Spark program and from external tools that connect to Spark SQL through standard database connectors (JDBC/ODBC). For instance, using business intelligence tools like Tableau.

Providing rich integration between SQL and regular Python/Java/Scala code, including the ability to join RDDs and SQL tables, expose custom functions in SQL, and more.

**23. What are benefits of Spark over MapReduce?**

Due to the availability of in-memory processing, Spark implements the processing around 10-100x faster than Hadoop MapReduce. MapReduce makes use of persistence storage for any of the data processing tasks.

Unlike Hadoop, Spark provides in-built libraries to perform multiple tasks form the same core like batch processing, Steaming, Machine learning, Interactive SQL queries. However, Hadoop only supports batch processing.

Hadoop is highly disk-dependent whereas Spark promotes caching and in-memory data storage.

Spark is capable of performing computations multiple times on the same dataset. This is called iterative computation while there is no iterative computing implemented by Hadoop.

[Read more](https://intellipaat.com/blog/spark-vs-map-reduce/) in this blog about the comparison of Spark and MapReduce.

**24. Is there any benefit of learning MapReduce, then?**

Yes, MapReduce is a paradigm used by many big data tools including Spark as well. It is extremely relevant to use MapReduce when the data grows bigger and bigger. Most tools like Pig and Hive convert their queries into MapReduce phases to optimize them better.

**25. What is Spark Executor?**

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

**26. Name types of Cluster Managers in Spark.**

The Spark framework supports three major types of Cluster Managers:

**Standalone :** a basic manager to set up a cluster.

**Apache Mesos :** generalized/commonly-used cluster manager, also runs Hadoop MapReduce and other applications.

**Yarn :** responsible for resource management in Hadoop

**27. What do you understand by worker node?**

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

**28. What is PageRank?**

A unique feature and algorithm in graph, PageRank is the measure of each vertex in the graph. For instance, an edge from u to v represents endorsement of v’s importance by u. In simple terms, if a user at Instagram is followed massively, it will rank high on that platform.

**29. Do you need to install Spark on all nodes of Yarn cluster while running Spark on Yarn?**

No because Spark runs on top of Yarn.

**30. Illustrate some demerits of using Spark.**

Since Spark utilizes more storage space compared to Hadoop and MapReduce, there may arise certain problems. Developers need to be careful while running their applications in Spark. Instead of running everything on a single node, the work must be distributed over multiple clusters.

**31. How to create RDD?**

Spark provides two methods to create RDD:• By parallelizing a collection in your Driver program. This makes use of SparkContext’s ‘parallelize’ methodval IntellipaatData = Array(2,4,6,8,10)  
val distIntellipaatData = sc.parallelize(IntellipaatData)• By loading an external dataset from external storage like HDFS, shared file system.

**Q1 Name a few commonly used Spark Ecosystems.**

Answer: Spark SQL (Shark)

Spark Streaming

GraphX

MLlib

SparkR

**Q2 What is “Spark SQL”?**

Answer: 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 “text files”, 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.

**Q3 Can we do real-time processing using Spark SQL?**

Answer: Not directly but we can register an existing RDD as a SQL table and trigger SQL queries on top of that.

**Q4 Explain about the major libraries that constitute the Spark Ecosystem**

Answer: Spark MLib- Machine learning library in Spark for commonly used learning algorithms like clustering, regression, classification, etc.

Spark Streaming – This library is used to process real time streaming data.

Spark GraphX – Spark API for graph parallel computations with basic operators like join Vertices, subgraph, aggregate Messages, etc.

Spark SQL – Helps execute SQL like queries on Spark data using standard visualization or BI tools.

**Q5 What is Spark SQL?**

Answer: SQL Spark, better known as Shark is a novel module introduced in Spark to work with structured data and perform structured data processing. Through this module, Spark executes relational SQL queries on the data. The core of the component supports an altogether different RDD called SchemaRDD, composed of rows objects and schema objects defining data type of each column in the row. It is similar to a table in relational database.

**Q6 What is a Parquet file?**

Answer: Parquet is a columnar format file supported by many other data processing systems. Spark SQL performs both read and write operations with Parquet file and consider it be one of the best big data analytics format so far.

**Q7 List the functions of Spark SQL.**

Answer: Spark SQL is capable of:

Loading data from a variety of structured sources

Querying data using SQL statements, both inside a Spark program and from external tools that connect to Spark SQL through standard database connectors (JDBC/ODBC). For instance, using business intelligence tools like Tableau

Providing rich integration between SQL and regular Python/Java/Scala code, including the ability to join RDDs and SQL tables, expose custom functions in SQL, and more

**Q8 What is Spark?**

Answer: Spark is a parallel data processing framework. It allows to develop fast, unified big data application combine batch, streaming and interactive analytics.

Q9 What is Hive on Spark?

Answer: 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.

**Q10 What is a “Parquet” in Spark?**

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

**Q11 What is Catalyst framework?**

Answer: Catalyst framework is a new optimization framework present in Spark SQL. It allows Spark to automatically transform SQL queries by adding new optimizations to build a faster processing system.

**Q12 Why is BlinkDB used?**

Answer: BlinkDB is a query engine for executing interactive SQL queries on huge volumes of data and renders query results marked with meaningful error bars. BlinkDB helps users balance ‘query accuracy’ with response time.

**Q13 How can you compare Hadoop and Spark in terms of ease of use?**

Answer: Hadoop MapReduce requires programming in Java which is difficult, though Pig and Hive make it considerably easier. Learning Pig and Hive syntax takes time. Spark has interactive APIs for different languages like Java, Python or Scala and also includes Shark i.e. Spark SQL for SQL lovers – making it comparatively easier to use than Hadoop.

**Q14 What are the various data sources available in SparkSQL?**

Answer: Parquet file

JSON Datasets

Hive tables

SparkSQL is a Spark component that supports querying data either via SQL or via the Hive Query Language. It originated as the Apache Hive port to run on top of Spark (in place of MapReduce) and is now integrated with the Spark stack. In addition to providing support for various data sources, it makes it possible to weave SQL queries with code transformations which results in a very powerful tool. Below is an example of a Hive compatible query:

**Q15 What are benefits of Spark over MapReduce?**

Answer: • Due to the availability of in-memory processing, Spark implements the processing around 10-100x faster than Hadoop MapReduce. MapReduce makes use of persistence storage for any of the data processing tasks.

Unlike Hadoop, Spark provides in-built libraries to perform multiple tasks form the same core like batch processing, Steaming, Machine learning, Interactive SQL queries. However, Hadoop only supports batch processing.

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

Spark is capable of performing computations multiple times on the same dataset. This is called iterative computation while there is no iterative computing implemented by Hadoop.

**Q16 How SparkSQL is different from HQL and SQL?**

Answer: SparkSQL is a special component on the spark Core engine that support SQL and Hive Query Language without changing any syntax. It’s possible to join SQL table and HQL table.

\*\*\*\*\*\*\*\*\*\*ALL The Best\*\*\*\*\*\*\*\*\*

Apache Spark SQL interview questions and answers

Question: 1. What are the multiple data sources supported by Spark SQL?

Answer: Apache Spark SQL is a popular ecosystem or interfaces to work with structured or semi-structured data. The multiple data sources supported by Spark SQL includethe text file, JSON file, Parquet file etc.

**[Read:   R Programming Language Interview Questions & Answers](https://www.janbasktraining.com/blog/r-interview-questions-answers/" \t "_blank)**

Question: 2. Does Spark SQL help in big data analytics through external tools too?

Answer: Yes, Spark SQL helps in big data analytics through external tools too. Let us see how it is done actually –

It access data using SQL statements in both ways either data is stored inside Spark program or data needs to access through external tools that are connected to Spark SQL through database connectors like JDBC or ODBC.

It provides rich integration between a database and regular coding with RDDs and SQL tables. It is also able to expose custom SQL functions as needed.

Question: 3. How is Spark SQL superior from others – HQL and SQL?

Answer: Spark SQL is advance database component able to support multiple database tools without changing their syntax. This is the way how Spark SQL accommodates both HQL and SQL superiorly.

Question: 4. Do real-time data processing is possible with Spark SQL?

Answer: Real-time data processing is not possible directly but obviously, we can make it happen by registering existing RDD as SQL table and trigger the SQL queries on priority.

Question: 5. Why is Parquet file format taken best choice for various data processing systems?

Answer: Parquet is popular columnar file format compatible with almost all data processing systems. This is the reason why it is taken as one of the best choices for big data analytics so far. Spark SQL interface is able to perform read and write operation on Parquet file and it can be accessed quickly whenever required.

Question: 6. Spark SQL is parallel or distributed data processing framework?

Answer: Spark SQL is parallel data processing framework where batch streaming and interactive data analytics is performed altogether.

Question: 7. What is catalyst framework in Spark SQL?

Answer: Catalyst framework is advanced functionality in Spark SQL for automatic transformation of SQL queries by addition of optimized functions that help in processing data faster and accurately than your expectations.

Question: 8. How to balance query accuracy and response time in Spark SQL?

Answer: TO maintain query accuracy and response time in Spark SQL, you are advised to go with BlinkDB query engine. The engine renders queries with meaningful results and significant error to maintain the accuracy.

Question: 9. Which framework is more preferable in terms of usage either Hadoop or Spark?

Answer: The programming in Hadoop was really tough that has been made easier with Spark by usage of interactive APIs for the different programming language. Obviously, Spark is preferable choice than Hadoop in terms of usage.

Question: 10. Are there any benefits of Apache Spark over Hadoop MapReduce?

Answer: Spark has the ability to perform data processing 100 times faster than MapReduce. Also, Spark has in-built memory processing and libraries to perform multiple tasks together like batch processing, streaming, interactive processing etc. The above discussion makes sure than Apache Spark is surely better than any other data processing frameworks exist as of now.

Apache Scala Interview questions and answers

Question: 11. How Array and List can be differentiated in Scala?

Answer: The Array is a mutable data structure that is sequential in nature while Lists are immutable data structures that are recursive in nature.

Size of array is predefined while lists change its size based on operational requirements. In other words, Lists are variable in size while the array is fixed size data structure.

Question: 12. How to map data and forms together in Scala?

Answer: The most wonderful solution to map data and forms together in Scala is “apply” and “unapply” methods. As the name suggests, apply method is used to map data while unapply method can be used to unmap the data.The unapply method follows the reverse operation of theapply method.

**[Read:   Kafka Interview Questions and Answers](https://www.janbasktraining.com/blog/kafka-interview-questions-answers/" \t "_blank)**

Question: 13. Do private members of Companion classes can be accessed through companion objects in Scala?

Answer: Yes, it is possible that private members of Companion classes can be accessed through companion objects in Scala.

Question: 14. What is the significance of immutable design in Scala programming language?

Answer: Every time when working with concurrent programs and other similar equality issues then immutable design in Scala programming language works amazingly. It helps in resolving coding related issues and makes programming easy for Scala developers.

Question: 15. Howcan Auxiliary Constructors be defined in Scala?

Answer: The keywords “def” and “this” is used to declare secondary or auxiliary constructors in Scala programming language. They are designed to overload constructors similar to Java. This is necessary to understand the working of each constructor deeply so that right constructor can be invoked at the right time. Even declaration of constructor differs from each other in terms of data types or parameters.

Question: 16. How will you explain yield keyword in Scala?

Answer: Yield keyword can be used either before or after expressions. It is taken more useful when declared before expression. The return value from every expression will be stored as the collection.The returned value can either be used as a normal collection or iterate in another loop.

Question: 17. How can functions be invoked silently without passing all the parameters?

Answer: In case, when we want to invoke functions silently without passing all the parameters, we should use implicit parameters.The parameters that you want to use implicit, you need to provide default values for the same.

Question: 18. What do you mean by Scala Traits and how it can be used in Scala programming language?

Answer: Scala trait is an advanced class in Scala that enables usage of multiple inheritance and it can be extended to multiple classes together. In other words, one class can have multiple Scala traits based on requirement.

Traits are used commonly when you need dependency injection. You just need to initiate class with Scala traits and dependency will be injected immediately.

Question: 19. Is there any difference between parallelism and concurrency in Scala programming language?

Answer: Normal users are generally confused between two terms parallelism and concurrency in Scala programming language.Here, we will discuss in simple words how they are different from each other and their significance too. When processes are executed sequentially then it is termed as concurrency while processes are executed simultaneously then it is named as parallelism technology. There are several library functions available in Scala to achieve parallelism.

Question: 20. How are Monads useful for Scala developers?

Answer: If you want to understand Monads in simple words then it would not be wrong comparing them with a wrapper. As wrappers are used to protect any product and to make it attractive, Monads are used for the same purpose in Scala. They are used to wrap objects together and perform two important functions further. These functions are –

Identity through “unit” in Scala

Bind through “flatMap” in Scala

Apache coding interview questions and answers

Question: 21. Howcan Transformations be defined in Apache Spark?

Answer: Transformations are created early in programs and these are generally used along with RDD. These functions are applied on already existed RDD to make a new RDD. Transformations cannot be used without implementing actions in Apache Spark.

**[Read:   Hive Interview Question And Answers](https://www.janbasktraining.com/blog/hive-interview-question-answers/" \t "_blank)**

The most popular examples of transformation are amap () and filter () that helps to create new RDD by selecting elements in available RDD.

Question: 22. What is the meaning of“Actions” in Apache Spark?

Answer: The data is taken back to the local machine from RDD with the help of “actions” in Apache Spark. The popular example of the action is folded () passes value again and again until the time it is left only one.

The actions are executed with the assistance of transformations that are created early in programs. The most popular examples of transformation are amap () and filter () that helps to create new RDD by selecting elements in available RDD.

Question: 23. Define Spark Core and how it is useful for Scala Developers?

Answer: Spark Core in Apache Spark is used for memory management, job monitoring, tolerate faults, scheduling jobs and interactive storage features. RDD is anadvanced feature in Spark Core suitable for tolerating faults.RDD is a collection of distributed objects available across multiple nodes that are generally manipulated in parallel.

Question: 24. Define data streaming in Apache Spark?

Answer: No framework can come tothe top without thefunctionality of live data streaming or handling live events. This is the reason why Apache Spark has used most advanced techniques to allow the same. For this purpose, Apache uses complex algorithms and high-level functions like reduce, map, join or window etc. These functions push data to file systems and live dashboards further.

Question: 25. How can graphs be processed in Apache Spark?

Answer: Out of all, one attractive feature supported in Apache Spark includes graph processing. Spark uses advanced multimedia component GraphX to create or explore graphs used to explore data more wisely and accurately.

Question: 26. Is there any library function to support machine learning algorithms?

Answer: Spark MLib is popular library function in Apache Spark to support machine learning algorithms. The common learning algorithms and utilities included in MLib library functions are aregression, clustering, classification, dimensional reduction, low-level optimization, advance level pipelining APIs, and collaborative filtering etc. The main objective of machine learning algorithm is recommendations, predictions and similar other functions.

Question: 27. Which File System is supported by Apache Spark?

Answer: Apache Spark is an advanced data processing system that can access data from multiple data sources. It creates distributed datasets from file system you use for data storage. The popular file systems used by Apache Spark include HBase, Cassandra, HDFS, and Amazon S3 etc.

Question: 28. How many cluster modes are supported in Apache Spark?

Answer: The three popular cluster modes supported in Apache Spark include – Standalone, Apache Mesos, and YARN cluster managers. YARN is the cluster management technology in Apache Spark stands for yet another resource negotiator. The idea was taken from Hadoop where YARN technology was specially introduced to reduce the burden on MapReduce function.

Question: 29. Is there any cluster management technology in Apache Spark?

Answer: Yes, the cluster management technology in Apache Spark is popular with the name YARN technology. YARN stands for yet another resource negotiator. The idea was taken from Hadoop where YARN technology was specially introduced to reduce the burden on MapReduce function.

Question: 30. How can you create RDD in Apache Spark?

Answer: There are two popular techniques that can be used to create RDD in Apache Spark – First is Parallelize and other is text File method. Here is quick explanation how both methods can be used for RDD creation.

val x= Array(5,7,8,9)

val y= sc.parallelize(x)

val input = sc.textFile(“input.txt”);

\*\*\*\*\*\*\*\*\*\* ALL THE BEST \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**What is Apache Spark?**

Begin is a fast, easy to-use and versatile data dealing with structure. It has a moved execution engine supporting cyclic data stream and in-memory figuring. Begin can continue running on Hadoop, free or in the cloud and is fit for getting to diverse data sources including HDFS, HBase, Cassandra and others.

**Explain key features of Spark.**

**Key Features:**

Grants Integration with Hadoop and files included in HDFS.

Spark has a canny vernacular shell as it has an independent **[Scala](http://www.multisoftsystems.com/big-data/apache-spark-scala-training)** interpreter.

Spark reinforces different analytic tools that are used for interactive query analysis, real-time analysis and graph processing

Spark includes Resilient Distributed Datasets, which can be put away across computing nodes in a cluster.

**Define RDD?**

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

**There are primarily two types of RDD:**

**Parallelized Collections**: The current RDD’s running parallel with each other.

**Hadoop datasets**: Perform function on each file record in HDFS or other accumulating structure.

**What does a Spark Engine do?**

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

**Define Partitions?**

As the name suggests, partition is a more diminutive and sensible division of data similar to ‘split’ in MapReduce. Partitioning is the system to derive logical units of data to quicken the taking care of method. Everything in Spark is a partitioned RDD.

**What operations RDD support?**

Actions

Transformations

[](http://www.multisoftsystems.com/business-analytics/apache-spark-training)

**What do you understand by Transformations in Spark?**

Transformations are functions implemented on RDD, resulting into another RDD. It does not execute until an action occurs. map() and filer() are examples of transformations, where the former applies the function passed to it on each element of RDD and results into another RDD. The filter() creates a new RDD by selecting elements form current RDD that pass function argument.

**Define Actions.**

An action helps in passing on back the data from RDD to the local machine. An action’s execution is the delayed consequence of all effectively rolled out improvements. reduce () is a movement that realizes the limit disregarded and over until the point that one regard accepting left. take() move makes each one of the qualities from RDD to local node.

**Define functions of SparkCore?**

Serving as the base engine, SparkCore performs diverse basic limits like memory organization, monitoring jobs, adjustment to inner disappointment, job scheduling and correspondence with storage systems.

**What is RDD Lineage?**

Spark does not support data replication in the memory and in this way, if any data is lost, it is remake using RDD family history. RDD ancestry is a technique that reproduces lost data distributions. The best is that RDD always remembers how to function from various datasets.

**What is Spark Driver?**

Spark Driver is the program that continues running on the pro center of the machine and articulates changes and exercises on data RDDs. The driver in like manner passes on the RDD outlines to Master, where the standalone cluster manager runs.

**What is Hive on Spark?**

Hive contains significant support for [**Apache Spark**](http://www.multisoftsystems.com/business-analytics/apache-spark-training), wherein Hive execution is configured to Spark:

hive> set spark.home=/location/to/sparkHome;

hive> set hive.execution.engine=spark;

Hive on Spark supports Spark on yarn mode by default.

**Name commonly-used Spark Ecosystems**

GraphX for making and computing graphs.

MLlib (Machine Learning Algorithms).

Spark SQL (Shark) – for developers.

Spark Streaming for processing live data streams.

SparkR to promote R Programming in Spark engine.

**Define Spark Streaming.**

Spark supports stream processing – a development to the Spark API, allowing stream processing of live data streams. The data from various sources like Flume, HDFS is spouted and finally processed to file systems, live dashboards and databases.

**What is GraphX?**

Spark uses GraphX for outline taking care of to fabricate and change smart graphs. The GraphX part enables programming architects to reason about sorted out data at scale.

**What does MLlib do?**

MLlib is adaptable machine learning library offered by Spark. It goes for making machine learning basic and flexible with essential learning algorithms and use cases like clustering, backslide filtering, dimensional abatement, and alike.

**What is Spark SQL?**

SQL Spark, generally called Shark is a novel module familiar in Spark with work with composed data and performs structured data dealing with. Through this module, Spark executes social SQL inquiries on the data.

**What is a Parquet file?**

Parquet is a columnar format file maintained by various other data processing systems. Spark SQL performs both read and create operations with Parquet file and consider it be extraordinary among different gigantic data examination outline up to this point.

**What is Yarn?**

Like Hadoop, Yarn is one of the key components in Spark, giving a central and resource organization stage to pass on versatile operations across the cluster. Running Spark on Yarn requires a matched apportionment of Spar as built on Yarn support.

**Is there any benefit of learning MapReduce, then?**

Yes, MapReduce is a perspective used by various immense data gadgets including Spark moreover. It is to an extraordinary degree imperative to use MapReduce when the data winds up plainly more noteworthy and more prominent.