

## **BIG DATA ANALYTICS HOMEWORK 4**

### **1. Compare Hadoop and Spark.**

- Hadoop is designed to handle batch processing efficiently. Spark is designed to handle real-time data efficiently.
- Hadoop is a high latency computing framework, which does not have an interactive mode.
- Spark is a low latency computing and can process data interactively.

### **2. What is Apache Spark?**

- Apache Spark is an open-source, distributed processing system used for big data workloads.
- It utilizes in-memory caching, and optimized query execution for fast analytic queries against data of any size.

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

- Apache Spark is an open-source, distributed processing system used for big data workloads.
- It utilizes in-memory caching, and optimized query execution for fast analytic queries against data of any size.

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

- Scala, Java, Python and R. Among these languages, Scala and Python have interactive shells for Spark.
- The Scala shell can be accessed through spark-shell and the Python shell through 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?**

- The primary difference between Spark and MapReduce is that Spark processes and retains data in memory for subsequent steps, whereas MapReduce processes data on disk.

- As a result, for smaller workloads, Spark's data processing speeds are up to 100x faster than MapReduce.

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

- RDD was the primary user-facing API in Spark since its inception.
- At the core, an RDD is an immutable distributed collection of elements of your data, partitioned across nodes in your cluster that can be operated in parallel with a low-level API that offers transformations and actions.

## **7. How do we create RDDs in Spark?**

### **There are two ways to create RDDs:**

parallelizing an existing collection in your driver program, or referencing a dataset in an external storage system, such as a shared filesystem, HDFS, HBase, or any data source offering a Hadoop InputFormat.

## **8. What is Executor Memory in a Spark application?**

- An executor is a process that is launched for a Spark application on a worker node.
- Each executor memory is the sum of yarn overhead memory and JVM Heap memory.
- JVM Heap memory comprises of: RDD Cache Memory. Shuffle Memory

## **9. What do you understand by Transformations in Spark?**

- Spark Transformation is a function that produces new RDD from the existing RDDs.
- It takes RDD as input and produces one or more RDD as output. Each time it creates new RDD when we apply any transformation.
- Thus, the so input RDDs, cannot be changed since RDD are immutable in nature.

## **10. Define Actions in Spark.**

- Actions are RDD's operation, that value returns back to the spark driver programs, which kick off a job to execute on a cluster.
- Transformation's output is an input of Actions.
- reduce, collect, takeSample, take, first, saveAsTextfile, saveAsSequenceFile, countByKey, foreach are common actions in Apache spark.

