*Spark*

**A C A D G I L D Page 1**

Session 6: In-Memory Management

Assignment 2*Spark*

**A C A D G I L D Page 2**

*Session 6: In-Memory Management*

*Assignment 2*

**Table of Contents**

1. Introduction .......................................................................................................................................... 3

2. Objective ............................................................................................................................................... 3

3. Prerequisites ......................................................................................................................................... 3

4. Associated Data Files ............................................................................................................................ 3

5. Problem Statement ............................................................................................................................... 3

6. Approximate Time to Complete Task ................................................................................................... 3 *Spark*

**A C A D G I L D Page 3**

**1. Introduction**

In this assignment you need to answer the question given.

**2. Objective**

This assignment will help you to understand Big Data basics.

**3. Prerequisites**

None

**4. Associated Data Files**

N/A

**5. Problem Statement**

1. What is difference cache and persist operation?

Ans:

Spark can persist (or cache) a dataset in memory across operations

The cache is fault-tolerant: if any partition of an RDD is lost, it will automatically be

Recomputed using the transformation that originally created it.

Memory\_only

MEMORY\_AND\_DISK

MRMORY\_ONLY\_SER

MEMORY\_AND\_DISK\_SER

2. What is the equivalent of Accumulators in MapReduce?

ANS :Accumulator is a variable which has to accumulate the associative operations like sum, count etc.

If accumulators are created with name,they will display in SPARK UI. This is useful to understand the progress of running stages.

Val acc=sc.accumulator(0)

3. What is the equivalent of Broadcast variables in MapReduce?

ANS :Spark actions are executed through a set of stages ,separated by distributed “Shuffle” Operation.

Spark automatically broadcast the common data needed by tasks within each stage.

The data broadcasted this way is cached in serialized and deserialized before running each task.

Val broadcastvar= sc.broadcast(Array(1,2,3,4))

Broadcastvar.value

4. How spark allocates the available memory during processing?

One of Spark’s selling points is that it takes 10-100x less time to finish a similar job written as Hadoop Map-Reduce. The trick is to store data reliably in in-memory – this makes repeatedly accessing it (ie. for iterative algorithms) incomparably faster.

In every Big Data application, memory is an extremely indispensable resource. Before proceeding further, it will be good to gain some insight regarding the possible impact of choosing the proper storage for our data.

**An executor** is the Spark application’s JVM process launched on a **worker node**. It runs tasks in threads and is responsible for keeping relevant partitions of data. Each process has an allocated **heap** with available memory (executor/driver).

5. How to recover the data from cached RDDs?

 Store the data in a distributed storage system like HDFS, etc. This will store your data in a distributed manner. You have to choose the right file system depending on your requirement (on-premise, or in cloud, etc.)

 Run Spark on the data in the HDFS file. Create an RDD from the file (see spark documentation), filter out the part of the data you actually needs (example, only the lines containing "error" in a large log file), and cache the necessary part in memory (so that subsequent queries are faster)

**6. Approximate Time to Complete Task**

30 min