Assignment-17.6:

Explain the following terms in detail

-- What are the uses of counters

-- MR Unit testing is based on

-- How testing is useful in industry

-- Mapreduce Task Counters,File system counters,Job Counter

-- Raw comparator VS Writable Comparator

-- Partitioner, Sort comparator, Group comparator

a.) What are the uses of counters:

Hadoop MapReduce Counter provides a way to measure the progress or the number of operations that occur within MapReduce programs. Basically, MapReduce framework provides a number of built-in counters to measure basic I/O operations, such as FILE\_BYTES\_READ/WRITTEN and Map/Combine/Reduce input/output records.

Hadoop inbuilt counters:

1) Task Counters

- Map input records (MAP\_INPUT\_RECORDS):- The number of input records consumed by all the maps in the job. Incremented every time a record is read from a RecordReader and passed to the maps map ( ) method by the framework.

-Split raw bytes (SPLIT\_RAW\_BYTES):- The number of bytes of input-split objects read by maps. These objects represent the split metadata (that is, the offset and length within a file) rather than the split data itself, so the total size should be small.

2) Job Counters

- Launched map tasks (TOTAL\_LAUNCHED\_MAPS):-The number of map tasks that were launched. Includes tasks that were started speculatively

-Launched reduce tasks (TOTAL\_LAUNCHED\_REDUCES) The number of reduce tasks that were launched. Includes tasks that were started speculatively.

b.) MR Unit testing is based on

MrUnit Testing is based on Junit which is a testing framework for java. Junit has two methods of testing methods manual testing and automated testing. J unit uses the technique of first testing and the coding so it emphasizes on first creating the test data that can be tested once the code is ready for implementation.

Features of Junit

* JUnit is an open source framework, which is used for writing and running tests.
* Provides annotations to identify test methods.
* Provides assertions for testing expected results.
* Provides test runners for running tests.
* JUnit tests allow you to write codes faster, which increases quality.
* JUnit is elegantly simple. It is less complex and takes less time.
* JUnit tests can be run automatically and they check their own results and provide immediate feedback. There's no need to manually comb through a report of test results.
* JUnit tests can be organized into test suites containing test cases and even other test suites.

c.) How testing is useful in industry

Testing is the process of evaluating a system or its component(s). It is basically used to check whether the developed code meets the requirements or not.

A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item.

Testing can be conducted by the developer which is called as the unit testing or as in many it companies the testing is done by separate section which does the intense testing on the developed code so as it meets the clients expectations.

An early start to testing reduces the cost and time to rework and produce error-free software that is delivered to the client. But it depends on the product being developed and the expectations of the client based on the above criteria different SDLC are mentioned.

Different types of SDLC are

V-Shaped Model

Iterative Model.

Spiral Model.

Big Bang Model.

Agile Model.

d.) Mapreduce Task Counters, File system counters, Job Counter

**Task Counter** – They gather information about the specific task over the execution of the task and the results then summed up at the end of the job. Task counters are maintained by each task. The value of the task counter are sent periodically to the tasktracker.

e.g.**PHYSICAL\_MEMORY\_BYTES,VIRTUAL\_MEMORY\_BYTES,** and **COMMITTED\_HEAP\_BYTE.**

**Job Counters**- Job counters are maintained by jobtracker, which measures the job level statistics. Job counter maintains the statistics related to the particular job, and it is maintained by application master. The value of the job counters do not change during the execution of the job.

e.g.

Launched map tasks (TOTAL\_LAUNCHED\_MAPS)The number of map tasks that were launched.

Launched reduce tasks (TOTAL\_LAUNCHED\_REDUCES) The number of reduce tasks that were launched. Includes tasks that were started speculatively.

**File System counters** - File system maintains different aspects related to data read and written into the hdfs there are three counters in this type.

1. FILE\_BYTES\_WRITTEN.
2. HDFS\_BYTES\_READ.
3. HDFS\_BYTES\_WRITTEN.

e.) Raw comparator VS Writable Comparator

**Raw Comparator:**

This interface permits implementers to compare records read from a stream without deserializing them into objects, thereby avoiding any overhead of object creation.

For example, the comparator for IntWritables implements the raw compare() method by reading an integer from each of the byte arrays b1 and b2 and comparing them directly from the given start positions (s1 and s2) and lengths (l1 and l2).

**Writable Comparator:**

WritableComparator is a general-purpose implementation of RawComparator for

WritableComparable classes.

It provides two main functions.

First, it provides a default implementation of the raw compare() method that deserializes the objects to be compared from the stream and invokes the object compare() method.

Second, it acts as a factory for RawComparator instances (that Writable implementations have registered).

f.) Partitioner, Sort comparator, Group comparator

**Group Comparator** – It decides which map output keys will be united (grouped) into one key, and of course all collections of values will be grouped too. Usually it takes a first key as the only one for summary collection.

**Partitioner** – It is used to decide the which key should go to which reducer, by default it uses the hash code of the object to decide the reducer but one can override the partitioner to send particular to particular reducer. This is mostly used in case of composite key, secondary sort

**Sort Comparator-** Used to define how map output keys are sorted,SortComparator decides how map output keys are sorted. If the property mapred.output.key.comparator.class is set, either explicitly or by calling setSortComparatorClass() on Job, then an instance of that class is used. (In the old API the equivalent method is setOutputKeyComparatorClass() on JobConf.)