Q1:[**What is a Sequence File? | Hadoop**](http://www.atoziq.com/2012/12/what-is-sequencefile-hadoop.html)

A. A Sequence File contains a binary encoding of an arbitrary number of homogeneous writable objects.  
B. A Sequence File contains a binary encoding of an arbitrary number of heterogeneous writable objects.  
C. A Sequence File contains a binary encoding of an arbitrary number of Writable Comparable objects, in sorted order.  
D. A Sequence File contains a binary encoding of an arbitrary number key-value pairs. Each key must be the same type. Each value must be same type.

**Answer: D**

### Q2:[Is there a map input format? | Hadoop](http://www.atoziq.com/2012/12/is-there-map-input-format-hadoop.html)

A.  Yes, but only in Hadoop 0.22+.  
B.  Yes, there is a special format for map files.  
C.  No, but sequence file input format can read map files.  
D.  Both 2 and 3 are correct answers.

**Answers: C**

### Q3:[In a MapReduce job, you want each of you input files processed by a single map task. How do you configure a MapReduce job so that a single map task processes each input file regardless of how many blocks the input file occupies?](http://www.atoziq.com/2012/12/in-mapreduce-job-you-want-each-of-you.html)

A. Increase the parameter that controls minimum split size in the job configuration.  
B. Write a custom Map Runner that iterates over all key-value pairs in the entire file.  
C. Set the number of mappers equal to the number of input files you want to process.  
D. Write a custom FileInputFormat and override the method is Split table to always return false.

**Answer: B**

### Q4:[Which of the following best describes the workings of TextInputFormat? | Hadoop](http://www.atoziq.com/2012/12/which-of-following-best-describes.html)

A. Input file splits may cross line breaks. A line that crosses tile splits is ignored.  
B. The input file is split exactly at the line breaks, so each Record Reader will read a series of complete lines.  
C. Input file splits may cross line breaks. A line that crosses file splits is read by the Record Readers of both splits containing the broken line.  
D. Input file splits may cross line breaks. A line that crosses file splits is read by the Record Reader of the split that contains the end of the broken line.  
E. Input file splits may cross line breaks. A line that crosses file splits is read by the Record Reader of the split that contains the beginning of the broken line.

**Answer: D**

### Q5:[Which of the following statements most accurately describes the relationship between MapReduce and Pig?](http://www.atoziq.com/2012/12/which-of-following-statements-most.html)

A. Pig provides additional capabilities that allow certain types of data manipulation not possible with MapReduce.  
B. Pig provides no additional capabilities to MapReduce. Pig programs are executed as MapReduce jobs via the Pig interpreter.  
C. Pig programs rely on MapReduce but are extensible, allowing developers to do special-purpose processing not provided by MapReduce.  
D. Pig provides the additional capability of allowing you to control the flow of multiple MapReduce jobs.

**Answer: D**

### Q6:[You need to import a portion of a relational database every day as files to HDFS, and generate Java classes to Interact with your imported data. Which of the following tools should you use to accomplish this?](http://www.atoziq.com/2012/12/you-need-to-import-portion-of.html)

A. Pig  
B. Hue  
C. Hive  
D. Flume  
E. Sqoop  
F. Oozie  
G. fuse-dfs

**Answer: C,E**

### Q7:[You have an employee who is a Date Analyst and is very comfortable with SQL. He would like to run ad-hoc analysis on data in your HDFS duster. Which of the following is a data warehousing software built on top of Apache Hadoop that defines a simple SQL-like query language well-suited for this kind of user?](http://www.atoziq.com/2012/12/you-have-employee-who-is-date-analyst.html)

A. Pig  
B. Hue  
C. Hive  
D. Sqoop  
E. Oozie  
F. Flume  
G. Hadoop Streaming

**Answer: C**

### Q8:[Workflows expressed in Oozie can contain: | Hadoop](http://www.atoziq.com/2012/12/workflows-expressed-in-oozie-can.html)

A. Iterative repetition of MapReduce jobs until a desired answer or state is reached.  
B. Sequences of MapReduce and Pig jobs. These are limited to linear sequences of actions with exception handlers but no forks.  
C. Sequences of MapReduce jobs only; no Pig or Hive tasks or jobs. These MapReduce sequences can be combined with forks and path joins.  
D. Sequences of MapReduce and Pig. These sequences can be combined with other actions including forks, decision points, and path joins.

**Answer: D**

### Q9:[You need a distributed, scalable, data Store that allows you random, real-time read/write access to hundreds of terabytes of data. Which of the following would you use? | Hadoop](http://www.atoziq.com/2012/12/you-need-distributed-scalable-data.html)

A. Hue  
B. Pig  
C. Hive  
D. Oozie  
E. HBase  
F. Flume  
G. Sqoop

**Answer: E**

### Q10:[Which of the following utilities allows you to create and run MapReduce jobs with any executable or script as the mapper and/or the reducer?](http://www.atoziq.com/2012/12/which-of-following-utilities-allows-you.html)

A. Oozie  
B. Sqoop  
C. Flume  
D. Hadoop Streaming

**Answer: D**

### Q11:[You are running a Hadoop cluster with all monitoring facilities properly configured. Which scenario will go undetected.? | Hadoop](http://www.atoziq.com/2012/12/you-are-running-hadoop-cluster-with-all.html)

A. Map or reduce tasks that are stuck in an infinite loop.  
B. HDFS is almost full.  
C. The NameNode goes down.  
D. A Datanode is disconnected from the cluster.  
E. MapReduce jobs that are causing excessive memory swaps.

**Answer: C**

### Q12:[Which of the following scenarios makes HDFS unavailable? | Hadoop](http://www.atoziq.com/2012/12/which-of-following-scenarios-makes-hdfs.html)

A. Job Tracker failure  
B. Task Tracker failure  
C. Datanode failure(HDF)  
D. NameNode failure(single point of failure but HDFS will be accessible)  
E. Secondary NameNode failure

**Answer: C**

### Q13:[Which MapReduce stage serves as a barrier, where all previous stages must be completed before it may proceed? | Hadoop](http://www.atoziq.com/2012/12/which-mapreduce-stage-serves-as-barrier.html)

A. Combine  
B. Group (a.k.a. 'shuffle')  
C. Reduce  
D. Write

**Ans: B**

### Q14:[Which TACC resource has support for Hadoop MapReduce? | Hadoop](http://www.atoziq.com/2012/12/which-tacc-resource-has-support-for.html)

A. Ranger(Ranger was one of the largest computing system in the world for open science research)  
B. Longhorn(Hadoop cluster)  
C. Lone star  
D. Spur

**Ans:** B

### Q15:[The Combine stage, if present, must perform the same aggregation operation as Reduce. | Hadoop](http://www.atoziq.com/2012/12/the-combine-stage-if-present-must.html)

A. True  
B. False

**Ans: B**

### Q16:[What is the implementation language of the Hadoop MapReduce framework? | Hadoop](http://www.atoziq.com/2012/12/what-is-implementation-language-of.html)

A. Java  
B. C  
C. FORTRAN  
D. Python

**Ans: A**

### Q17:[Which of the following MapReduce execution frameworks focus on execution in shared-memory environments? | Hadoop](http://www.atoziq.com/2012/12/which-of-following-mapreduce-execution.html)

A. Hadoop  
B. Twister  
C. Phoenix

**Ans: C**

### Q18:[How can a distributed file system such as HDFS provide opportunities for optimization of a MapReduce operation? | Hadoop](http://www.atoziq.com/2012/12/how-can-distributed-filesystem-such-as.html)

A. Data represented in a distributed file system is already sorted.  
B. Distributed file systems must always be resident in memory, which is much faster than disk.  
C. Data storage and processing can be co-located on the same node, so that most input data relevant to Map or Reduce will be present on local disks or cache.  
D. A distributed file system makes random access faster because of the presence of a dedicated node serving file metadata.

**Ans: D**

### Q19:[What is the input to the Reduce function? | Hadoop](http://www.atoziq.com/2012/12/what-is-input-to-reduce-function-hadoop.html)

A. One key and a list of all values associated with that key.  
B. One key and a list of some values associated with that key.  
C. An arbitrarily sized list of key/value pairs.

**Ans: A**

### Q21:[Which MapReduce phase is theoretically able to utilize features of the underlying file system in order to optimize parallel execution? | Hadoop](http://www.atoziq.com/2012/12/which-mapreduce-phase-is-theoretically.html)

A. Split  
B. Map  
C. Combine

**Ans: A**

### Q22:[RPC means\_\_\_\_\_\_. | Hadoop](http://www.atoziq.com/2012/12/rpc-means-hadoop.html)

A. Remote processing call  
B. Remote process call  
C. Remote procedure call  
D. None of the above

**Answer: C**

### Q23: [Which method of the File System object is used for reading a file in HDFS | Hadoop](http://www.atoziq.com/2012/12/which-method-of-filesystem-object-is.html)

A. open()  
B. access()  
C. select()  
D. None of the above

**Answer: A**

### Q24: [How many states does Writable interface defines \_\_\_\_\_. | Hadoop](http://www.atoziq.com/2012/12/how-many-states-does-writable-interface.html)

A. Two  
B. Four  
C. Three  
D. None of the above

**Answer: A**

### Q25: [What are supported programming languages for Map Reduce? | Hadoop](http://www.atoziq.com/2012/12/what-are-supported-programming.html)

A.  The most common programming language is Java, but scripting languages are also supported via Hadoop streaming.   
B.  Any programming language that can comply with Map Reduce concept can be supported.   
C.  Only Java supported since Hadoop was written in Java.   
D.  Currently Map Reduce supports Java, C, C++ and COBOL.

**Answer: A**

### Q26:[How does Hadoop process large volumes of data? | Hadoop](http://www.atoziq.com/2012/12/how-does-hadoop-process-large-volumes.html)

A.  Hadoop uses a lot of machines in parallel. This optimizes data processing.   
B.  Hadoop was specifically designed to process large amount of data by taking advantage of MPP hardware   
C.  Hadoop ships the code to the data instead of sending the data to the code.   
D.  Hadoop uses sophisticated caching techniques on namenode to speed processing of data

**Answer: C**

### Q27: [What are sequence files and why are they important? | Hadoop](http://www.atoziq.com/2012/12/what-are-sequence-files-and-why-are.html)

A.  Sequence files are binary format files that are compressed and are split able. They are often used in high-performance map-reduce jobs   
B.  Sequence files are a type of the file in the Hadoop framework that allow data to be sorted   
C.  Sequence files are intermediate files that are created by Hadoop after the map step   
D.  Both B and C are correct

**Answer: A**

### Q28: [What are map files and why are they important? | Hadoop](http://www.atoziq.com/2012/12/what-are-map-files-and-why-are-they.html)

A.  Map files are stored on the namenode and capture the metadata for all blocks on a particular rack. This is how Hadoop is "rack aware"   
B.  Map files are the files that show how the data is distributed in the Hadoop cluster.   
C.  Map files are generated by Map-Reduce after the reduce step. They show the task distribution during job execution   
D.  Map files are sorted sequence files that also have an index. The index allows fast data look up.

**Answer: D**

### Q29: [How can you use binary data in MapReduce? | Hadoop](http://www.atoziq.com/2012/12/how-can-you-use-binary-data-in.html)

A.  Binary data can be used directly by a map-reduce job. Often binary data is added to a sequence file.   
B.  Binary data cannot be used by Hadoop framework. Binary data should be converted to a Hadoop compatible format prior to loading.   
C.  Binary can be used in map-reduce only with very limited functionality. It cannot be used as a key for example.   
D.  Hadoop can freely use binary files with map-reduce jobs so long as the files have headers

**Answer: A**

### Q30:[What is map - side join? | Hadoop](http://www.atoziq.com/2012/12/what-is-map-side-join-hadoop.html)

A.  Map-side join is done in the map phase and done in memory   
B.  Map-side join is a technique in which data is eliminated at the map step   
C.  Map-side join is a form of map-reduce API which joins data from different locations   
D.  None of these answers are correct

**Answer: A**

### Q31: [What is reduce - side join? | Hadoop](http://www.atoziq.com/2012/12/what-is-reduce-side-join-hadoop.html)

A.  Reduce-side join is a technique to eliminate data from initial data set at reduce step   
B.  Reduce-side join is a technique for merging data from different sources based on a specific key. There are no memory restrictions   
C.  Reduce-side join is a set of API to merge data from different sources.   
D.  None of these answers are correct

**Answer: B**

### Q32: [What is HIVE? | Hadoop](http://www.atoziq.com/2012/12/what-is-hive-hadoop_21.html)

A.  Hive is a part of the Apache Hadoop project that provides SQL like interface for data processing   
B.  Hive is one component of the Hadoop framework that allows for collecting data together into an external repository   
C.  Hive is a way to add data from local file system to HDFS   
D.  HIVE is part of the Apache Hadoop project that enables in-memory analysis of real-time streams of data

**Answer: A**

### Q33: [What is PIG? | Hadoop](http://www.atoziq.com/2012/12/what-is-pig-hadoop.html)

A.  Pig is a subset fo the Hadoop API for data processing   
B.  Pig is a part of the Apache Hadoop project that provides C-like scripting language interface for data processing   
C.  Pig is a part of the Apache Hadoop project. It is a "PL-SQL" interface for data processing in Hadoop cluster   
D.  PIG is the third most popular form of meat in the US behind poultry and beef.

**Answer: B**

### Q34: [How can you disable the reduce step? | Hadoop](http://www.atoziq.com/2012/12/how-can-you-disable-reduce-step-hadoop.html)

A.  The Hadoop administrator has to set the number of the reducer slot to zero on all slave nodes. This will disable the reduce step.   
B.  It is impossible to disable the reduce step since it is critical part of the Map-Reduce abstraction.   
C.  A developer can always set the number of the reducers to zero. That will completely disable the reduce step.   
D.  While you cannot completely disable reducers you can set output to one. There needs to be at least one reduce step in Map-Reduce abstraction.

**Answer: C**

### Q35: [Why would a developer create a map-reduce without the reduce step? | Hadoop](http://www.atoziq.com/2012/12/why-would-developer-create-map-reduce.html)

A.  Developers should design Map-Reduce jobs without reducers only if no reduce slots are available on the cluster.   
B.  Developers should never design Map-Reduce jobs without reducers. An error will occur upon compile.   
C.  There is a CPU intensive step that occurs between the map and reduce steps. Disabling the reduce step speeds up data processing.   
D.  It is not possible to create a map-reduce job without at least one reduce step. A developer may decide to limit to one reducer for debugging purposes.

**Answer: C**

### Q36: [What is the default input format? | Hadoop](http://www.atoziq.com/2012/12/what-is-default-input-format-hadoop.html)

A.  The default input format is xml. Developer can specify other input formats as appropriate if xml is not the correct input.   
B.  There is no default input format. The input format always should be specified.   
C.  The default input format is a sequence file format. The data needs to be preprocessed before using the default input format.   
D.  The default input format is TextInputFormat with byte offset as a key and entire line as a value.

**Answer: D**

### Q37: [How can you overwrite the default input format? | Hadoop](http://www.atoziq.com/2012/12/how-can-you-overwrite-default-input.html)

A.  In order to overwrite default input format, the Hadoop administrator has to change default settings in config file.   
B.  In order to overwrite default input format, a developer has to set new input format on job config before submitting the job to a cluster.   
C.  The default input format is controlled by each individual mapper and each line needs to be parsed individually.   
D.  None of these answers are correct.

**Answer: B**

### Q38: [What are the common problems with map-side join? | Hadoop](http://www.atoziq.com/2012/12/what-are-common-problems-with-map-side.html)

A.  The most common problem with map-side joins is introducing a high level of code complexity. This complexity has several downsides: increased risk of bugs and performance degradation. Developers are cautioned to rarely use map-side joins.   
B.  The most common problem with map-side joins is lack of the available map slots since map-side joins require a lot of mappers.   
C.  The most common problems with map-side joins are out of memory exceptions on slave nodes.   
D.  The most common problem with map-side join is not clearly specifying primary index in the join. This can lead to very slow performance on large datasets.

**Answer: C**

### Q39: [Which is faster: Map-side join or Reduce-side join? Why?](http://www.atoziq.com/2012/12/which-is-faster-map-side-join-or-reduce.html)

A.  Both techniques have about the the same performance expectations.   
B.  Reduce-side join because join operation is done on HDFS.   
C.  Map-side join is faster because join operation is done in memory.   
D.  Reduce-side join because it is executed on a the namenode which will have faster CPU and more memory.

**Answer: C**

### Q40: [Will settings using Java API overwrite values in configuration files? | Hadoop](http://www.atoziq.com/2012/12/will-settings-using-java-api-overwrite.html)

A.  No. The configuration settings in the configuration file takes precedence   
B.  Yes. The configuration settings using Java API take precedence   
C.  It depends when the developer reads the configuration file. If it is read first then no.   
D.  Only global configuration settings are captured in configuration files on namenode. There are only a very few job parameters that can be set using Java API.

**Answer: B**

### Q41: [What is AVRO? | Hadoop](http://www.atoziq.com/2012/12/what-is-avro-hadoop.html)

A.  Avro is a java serialization library   
B.  Avro is a java compression library   
C.  Avro is a java library that create split table files   
D.  None of these answers are correct

**Answer: A**

### Q42: [Can you run Map - Reduce jobs directly on Avro data? | Hadoop](http://www.atoziq.com/2012/12/can-you-run-map-reduce-jobs-directly-on.html)

A.  Yes, Avro was specifically designed for data processing via Map-Reduce   
B.  Yes, but additional extensive coding is required   
C.  No, Avro was specifically designed for data storage only   
D.  Avro specifies metadata that allows easier data access. This data cannot be used as part of map-reduce execution, rather input specification only.

**Answer: A**

### Q43:[What is distributed cache? | Hadoop](http://www.atoziq.com/2012/12/what-is-distributed-cache-hadoop.html)

A.  The distributed cache is special component on namenode that will cache frequently used data for faster client response. It is used during reduce step.   
B.  The distributed cache is special component on datanode that will cache frequently used data for faster client response. It is used during map step.   
C.  The distributed cache is a component that caches java objects.   
D.  The distributed cache is a component that allows developers to deploy jars for Map-Reduce processing.

**Answer: D**

|  |  |
| --- | --- |
|  | Problem here was that I was doing the following:  Configuration conf = new Configuration();  Job job = new Job(conf, "word count");  DistributedCache.addCacheFile(new URI("/user/hduser/cacheFile/Cache1"), conf);  Since the Job constructor makes an internal copy of the conf instance, adding the cache file afterwards doesn't affect things. Instead, I should do this:  Configuration conf = new Configuration();  DistributedCache.addCacheFile(new URI("/user/hduser/cacheFile/Cache1"), conf);  Job job = new Job(conf, "wordcount"); |

### Q44: [What is the best performance one can expect from a Hadoop cluster? | Hadoop](http://www.atoziq.com/2012/12/what-is-best-performance-one-can-expect.html)

A.  The best performance expectation one can have is measured in seconds. This is because Hadoop can only be used for batch processing   
B.  The best performance expectation one can have is measured in milliseconds. This is because Hadoop executes in parallel across so many machines   
C.  The best performance expectation one can have is measured in minutes. This is because Hadoop can only be used for batch processing   
D.  It depends on on the design of the map-reduce program, how many machines in the cluster, and the amount of data being retrieved

**Answer: A**

### Q45: [What is writable? | Hadoop](http://www.atoziq.com/2012/12/what-is-writable-hadoop.html)

A.  Writable is a java interface that needs to be implemented for streaming data to remote servers.   
B.  Writable is a java interface that needs to be implemented for HDFS writes.   
C.  Writable is a java interface that needs to be implemented for MapReduce processing.   
D.  None of these answers are correct.

**Answer: C**

### Q46: [The Hadoop API uses basic Java types such as Long Writable, Text, IntWritable. They have almost the same features as default java classes. What are these writable data types optimized for?](http://www.atoziq.com/2012/12/the-hadoop-api-uses-basic-java-types.html)

A.  Writable data types are specifically optimized for network transmissions   
B.  Writable data types are specifically optimized for file system storage   
C.  Writable data types are specifically optimized for map-reduce processing   
D.  Writable data types are specifically optimized for data retrieval

**Answer: A**

### Q47: [Can a custom type for data Map-Reduce processing be implemented? | Hadoop](http://www.atoziq.com/2012/12/can-custom-type-for-data-map-reduce.html)

A.  No, Hadoop does not provide techniques for custom data types.   
B.  Yes, but only for mappers.   
C.  Yes, custom data types can be implemented as long as they implement writable interface.   
D.  Yes, but only for reducers.

**Answer: C**

### Q48: [What happens if mapper output does not match reducer input? | Hadoop](http://www.atoziq.com/2012/12/what-happens-if-mapper-output-does-not.html)

A.  Hadoop API will convert the data to the type that is needed by the reducer.   
B.  Data input/output inconsistency cannot occur. A preliminary validation check is executed prior to the full execution of the job to ensure there is consistency.   
C.  The java compiler will report an error during compilation but the job will complete with exceptions.   
D.  A real-time exception will be thrown and map-reduce job will fail.

**Answer: D**

### Q49: [Can you provide multiple input paths to a map-reduce jobs? | Hadoop](http://www.atoziq.com/2012/12/can-you-provide-multiple-input-paths-to.html)

A.  Yes, but only in Hadoop 0.22+.   
B.  No, Hadoop always operates on one input directory.   
C.  Yes, developers can add any number of input paths.   
D.  Yes, but the limit is currently capped at 10 input paths.

**Answer:  C**

### Q50: [Which of the following is not monitored by monitors of Hadoop cluster? | Hadoop](http://www.atoziq.com/2012/12/can-you-provide-multiple-input-paths-to.html)

A.  Map or reduce tasks that are stuck in an infinite loop.  
B.   HDFS is almost full.  
C.  The Name Node goes down.  
D.  A Data Node is disconnected from the cluster.

**Answer:  D**

### Q51:Does AVRO data can be processed and used by MapReduce jobs? |Hadoop

A.  Yes, Avro was specifically designed for data processing via Map-Reduce   
B.   Yes, but additional extensive coding is required   
C.  No, Avro was specifically designed for data storage only   
D.  Avro specifies metadata that allows easier data access. This data cannot be used as part of map-reduce execution, rather input specification only.

**Answer:  A**

### Some Hadoop commands

### To execute the jar in HDFS and process the MapReduce Job

hadoop jar /u01/training/student33/Dictionary.jar com.msat.mapreduce.Dictionary1 /user/student33/fulldictionary.txt /user/student33/Dictionary1

**To see the content in HDFS**

hdfs dfs -ls /user/student33/

**Copy the file from Linux to HDFS**

hdfs dfs -copyFromLocal /u01/training/student33/fulldictionary.txt

**To see the output of MapReduce Job**

hdfs dfs -cat /user/student33/Dictionary/part-r-00000

**HDFS’s fsck command understands blocks**

hadoop fsck / -files -blocks

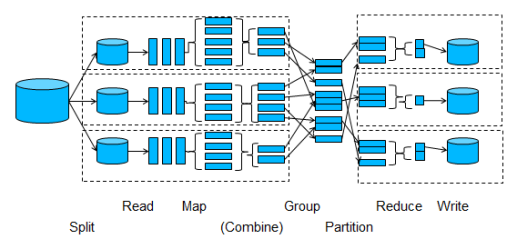
### C:\Users\sk104955\Desktop\CCDH-Certification\hadoop_jobtracker.png

### C:\Users\sk104955\Desktop\CCDH-Certification\hadoop_jobtracker_arch.png

### C:\Users\sk104955\Desktop\CCDH-Certification\o_Picture 6.png

### Map Reduce Execution Stage

### Reference: http://www.cac.cornell.edu/Ranger/MapReduce/dfs.aspx



This is a schematic of various steps that *may* be managed by a MapReduce execution framework. Three processing nodes, indicated by the dashed boxes, execute *Map* and *Reduce* operations in parallel.

**Split**

Divide the input data into parallel streams and deliver it to the processing nodes. Where data resides in parallel or distributed filesystems, the framework may leverage native parallel capabilities of the filesystem (e.g. striping or sharding) to deliever the split data to processing nodes in an efficient manner. Some frameworks implement *split* in a communications layer using publish/subscribe message queuing or MPI.

**Read**

Parse the input data into individual units *Di*. In the "word count" example from before, each unit was a line of ASCII text. In practice, data may be read into arbitrarily simple or complex units such as arrays of integers, pdf documents, SQL rows, etc.

**Map**

This is the user-supplied *Map* function. This function must be able to accept *Di* produced by *read* as input.

**Combine**

This is a potential optimization stage where some intermediate key/value pairs may be aggregated prior to *Reduce*, if it is possible to do so. It accepts an arbitrary list of intermediate key/value pairs grouped by key, performs an aggregation operation on the values (if possible), and emits a smaller number of key/value pairs as output. In other words, it can opportunistically pre-aggregate some of the intermediate values *Vi* in order to reduce the workload on *Reduce*.

**Group**

Groups all intermediate results produced by *Map* (possibly refined by *Combine*) and assembles all associated values into a list. This stage necessarily forms a *barrier* in the MapReduce process. Group cannot complete and emit the grouped results to a later stage in Map Reduce until all intermediate values produced by *Map* have been produced and all earlier stages have been completed.

**Partition**

Where there are multiple nodes executing *Reduce* in parallel, the *Partition* stage decides which intermediate keys to send to a particular *Reduce* node. Many frameworks use a simple hash to evenly distribute keys to *Reduce* nodes. In situations where the data volume is skewed (i.e. certain keys may have more values associated with them than others), custom *Partition* logic may be provided in order to make the distribution of work to *Reduce* nodes more fair and even.

**Reduce**

This is the user-supplied *Reduce* function.

**Write**

Given lists of output values from *Reduce*, *list(Vf)*, the *Write* stage serializes these values to storage.

## Apache Hadoop NextGen MapReduce (YARN)

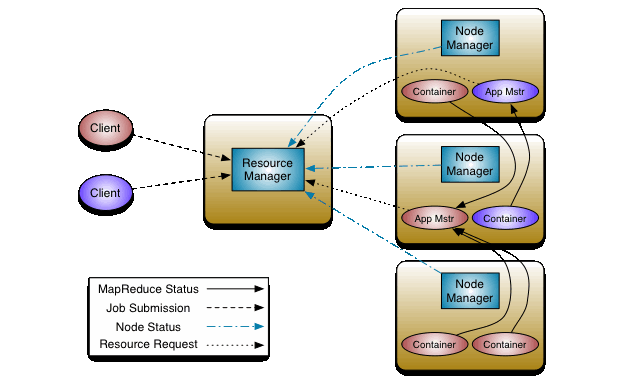
YARN meets the scalability shortcomings of “classic” MapReduce by splitting the responsibilities of the jobtracker into separate entities. The jobtracker takes care of both job scheduling (matching tasks with tasktrackers) and task progress monitoring (keeping track of tasks and restarting failed or slow tasks, and doing task bookkeeping such as maintaining counter totals).YARN separates these two roles into two independent daemons: a *resource manager* to manage the use of resources across the cluster, and an *application master* to manage the lifecycle of applications running on the cluster.

MapReduce has undergone a complete overhaul in hadoop-0.23 and we now have, what we call, MapReduce 2.0 (MRv2) or YARN.

The fundamental idea of MRv2 is to split up the two major functionalities of the JobTracker, resource management and job scheduling/monitoring, into separate daemons. The idea is to have a global ResourceManager (*RM*) and per-application ApplicationMaster (*AM*). An application is either a single job in the classical sense of Map-Reduce jobs or a DAG of jobs.

The ResourceManager and per-node slave, the NodeManager (*NM*), form the data-computation framework. The ResourceManager is the ultimate authority that arbitrates resources among all the applications in the system.

The per-application ApplicationMaster is, in effect, a framework specific library and is tasked with negotiating resources from the ResourceManager and working with the NodeManager(s) to execute and monitor the tasks.



The ResourceManager has two main components: Scheduler and ApplicationsManager.

The Scheduler is responsible for allocating resources to the various running applications subject to familiar constraints of capacities, queues etc. The Scheduler is pure scheduler in the sense that it performs no monitoring or tracking of status for the application. Also, it offers no guarantees about restarting failed tasks either due to application failure or hardware failures. The Scheduler performs its scheduling function based the resource requirements of the applications; it does so based on the abstract notion of a resource *Container* which incorporates elements such as memory, cpu, disk, network etc. In the first version, only memory is supported.

The Scheduler has a pluggable policy plug-in, which is responsible for partitioning the cluster resources among the various queues, applications etc. The current Map-Reduce schedulers such as the CapacityScheduler and the FairScheduler would be some examples of the plug-in.

The CapacityScheduler supports hierarchical queues to allow for more predictable sharing of cluster resources

The ApplicationsManager is responsible for accepting job-submissions, negotiating the first container for executing the application specific ApplicationMaster and provides the service for restarting the ApplicationMaster container on failure.

The NodeManager is the per-machine framework agent who is responsible for containers, monitoring their resource usage (cpu, memory, disk, network) and reporting the same to the ResourceManager/Scheduler.

The per-application ApplicationMaster has the responsibility of negotiating appropriate resource containers from the Scheduler, tracking their status and monitoring for progress.

MRV2 maintains **API compatibility** with previous stable release (hadoop-0.20.205). This means that all Map-Reduce jobs should still run unchanged on top of MRv2 with just a recompile.

# Hive

[](http://hive.apache.org/)Hive is a data warehousing infrastructure based on the Hadoop. Hive is designed to enable easy data summarization using ad-hoc queries on large volumes of data. When data is loaded in hive, it creates schema but the original data is stored in HDFS. When SQL queries are submitted to hive, it internally converts it to MapReduce job. It enables users familiar with SQL to do ad-hoc querying on unstructured data present in HDFS. On the other hand, Hive also allows traditional MapReduce programmers to be able to plug in their custom mappers and reducers. Hive does not offer real-time queries and row level updates. It is best used for batch jobs over large sets of immutable data.

# Pig

[](http://pig.apache.org/)Pig is a platform for analyzing large data sets that consists of a high level language for expressing data analysis programs, with infrastructure to execute these programs. It is a very simple language, to write programs, when we execute these programs; it internally gets converted into MapReduce Job. Pig programming structure is amenable to substantial parallelization. The way in which tasks are encoded permits the system to optimize their execution automatically, so programmers need not to focus on efficiency.

# Sqoop

[](http://sqoop.apache.org/)Sqoop is a tool designed to transfer data between Hadoop and relational database management system (RDBMS). We can use Sqoop to import data from a RDBMS such as MySQL or Oracle into the Hadoop Distributed File System (HDFS), transform the data in Hadoop by running MapReduce job, and then export the data back into an RDBMS. Sqoop automates this process, relying on the database to describe the schema for the data to be imported. Sqoop uses MapReduce to import and export the data, which provides parallel operation as well as fault tolerance.

# Flume

Apache Flume is a distributed, reliable, and available system for efficiently collecting, aggregating and moving large amounts of log data from many different sources to a centralized data store. The use of Apache Flume is not only restricted to log data aggregation. Since data sources are customizable, Flume can be used to transport massive quantities of event data including but not limited to network traffic data, social-media-generated data, email messages and pretty much any data source possible.

