CCD-410$^{Q\&As}$

Cloudera Certified Developer for Apache Hadoop (CCDH)

Pass Cloudera CCD-410 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.pass2lead.com/ccd-410.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Cloudera Official Exam Center

QUESTION 1

Can you use MapReduce to perform a relational join on two large tables sharing a key? Assume that the two tables are formatted as comma-separated files in HDFS.

A. Yes.

B. Yes, but only if one of the tables fits into memory

C. Yes, so long as both tables fit into memory.

D. No, MapReduce cannot perform relational operations.

E. No, but it can be done with either Pig or Hive.

Correct Answer: A

Note:

*

Join Algorithms in MapReduce A) Reduce-side join B) Map-side join C) In-memory join / Striped Striped variant variant / Memcached variant

*

Which join to use? / In-memory join > map-side join > reduce-side join / Limitations of each? In-memory join: memory Map-side join: sort order and partitioning

Reduce-side join: general purpose

QUESTION 2

Which describes how a client reads a file from HDFS?

A. The client queries the NameNode for the block location(s). The NameNode returns the block location

(s) to the client. The client reads the data directory off the DataNode(s).

B. The client queries all DataNodes in parallel. The DataNode that contains the requested data responds directly to the client. The client reads the data directly off the DataNode.

C. The client contacts the NameNode for the block location(s). The NameNode then queries the DataNodes for block locations. The DataNodes respond to the NameNode, and the NameNode redirects the client to the DataNode that holds the requested data block(s). The client then reads the data directly off the DataNode.

D. The client contacts the NameNode for the block location(s). The NameNode contacts the DataNode that holds the requested data block. Data is transferred from the DataNode to the NameNode, and then from the NameNode to the client.

Correct Answer: A

Reference: 24 Interview Questions and Answers for Hadoop MapReduce developers, How the Client communicates

with HDFS?

QUESTION 3

The Hadoop framework provides a mechanism for coping with machine issues such as faulty configuration or impending hardware failure. MapReduce detects that one or a number of machines are performing poorly and starts more copies of a map or reduce task. All the tasks run simultaneously and the task finish first are used. This is called:

A. Combine

B. IdentityMapper

C. IdentityReducer

D. Default Partitioner

E. Speculative Execution

Correct Answer: E

Speculative execution: One problem with the Hadoop system is that by dividing the tasks across many nodes, it is possible for a few slow nodes to rate-limit the rest of the program. For example if one node has a slow disk controller, then it may be reading its input at only 10% the speed of all the other nodes. So when 99 map tasks are already complete, the system is still waiting for the final map task to check in, which takes much longer than all the other nodes.

By forcing tasks to run in isolation from one another, individual tasks do not know where their inputs come from. Tasks trust the Hadoop platform to just deliver the appropriate input. Therefore, the same input can be processed multiple times in parallel, to exploit differences in machine capabilities. As most of the tasks in a job are coming to a close, the Hadoop platform will schedule redundant copies of the remaining tasks across several nodes which do not have other work to perform. This process is known as speculative execution. When tasks complete, they announce this fact to the JobTracker. Whichever copy of a task finishes first becomes the definitive copy. If other copies were executing speculatively, Hadoop tells the TaskTrackers to abandon the tasks and discard their outputs. The Reducers then receive their inputs from whichever Mapper completed successfully, first.

Reference: Apache Hadoop, Module 4: MapReduce

Note:

*

Hadoop uses "speculative execution." The same task may be started on multiple boxes. The first one to

finish wins, and the other copies are killed.

Failed tasks are tasks that error out.

*

There are a few reasons Hadoop can kill tasks by his own decisions:

a) Task does not report progress during timeout (default is 10 minutes)

b) FairScheduler or CapacityScheduler needs the slot for some other pool (FairScheduler) or queue

(CapacityScheduler).

c) Speculative execution causes results of task not to be needed since it has completed on other place.

Reference: Difference failed tasks vs killed tasks

QUESTION 4

All keys used for intermediate output from mappers must:

A. Implement a splittable compression algorithm.

B. Be a subclass of FileInputFormat.

C. Implement WritableComparable.

D. Override isSplitable.

E. Implement a comparator for speedy sorting.

Correct Answer: C

The MapReduce framework operates exclusively on pairs, that is, the framework views the input to the job as a set of pairs and produces a set of pairs as the output of the job, conceivably of different types.

The key and value classes have to be serializable by the framework and hence need to implement the Writable interface. Additionally, the key classes have to implement the WritableComparable interface to facilitate sorting by the framework.

Reference: MapReduce Tutorial

QUESTION 5

Given a directory of files with the following structure: line number, tab character, string:

Example: 1 abialkjfjkaoasdfjksdlkjhqweroij 2 kadfjhuwqounahagtnbvaswslmnbfgy 3 kjfteiomndscxeqalkzhtopedkfsikj

You want to send each line as one record to your Mapper. Which InputFormat should you use to complete the line: conf.setInputFormat (____.class) ; ?

A. SequenceFileAsTextInputFormat

B. SequenceFileInputFormat

C. KeyValueFileInputFormat

D. BDBInputFormat

Correct Answer: C

http://stackoverflow.com/questions/9721754/how-to-parse-customwritable-from-text-in-hadoop

QUESTION 6

You have the following key-value pairs as output from your Map task:

(the, 1) (fox, 1) (faster, 1) (than, 1) (the, 1) (dog, 1)

How many keys will be passed to the Reducer\\'s reduce method?

A. Six

B. Five

C. Four

D. Two

E. One

F. Three

Correct Answer: B

Only one key value pair will be passed from the two (the, 1) key value pairs.

QUESTION 7

Workflows expressed in Oozie can contain:

A. Sequences of MapReduce and Pig. These sequences can be combined with other actions including forks, decision points, and path joins.

B. Sequences of MapReduce job only; on Pig on Hive tasks or jobs. These MapReduce sequences can be combined with forks and path joins.

C. Sequences of MapReduce and Pig jobs. These are limited to linear sequences of actions with exception handlers but no forks.

D. Iterntive repetition of MapReduce jobs until a desired answer or state is reached.

Correct Answer: A

Oozie workflow is a collection of actions (i.e. Hadoop Map/Reduce jobs, Pig jobs) arranged in a control dependency DAG (Direct Acyclic Graph), specifying a sequence of actions execution. This graph is specified in hPDL (a XML Process Definition Language).

hPDL is a fairly compact language, using a limited amount of flow control and action nodes. Control nodes define the flow of execution and include beginning and end of a workflow (start, end and fail nodes) and mechanisms to control the workflow execution path (decision, fork and join nodes).

Workflow definitions Currently running workflow instances, including instance states and variables

Reference: Introduction to Oozie

Note: Oozie is a Java Web-Application that runs in a Java servlet-container - Tomcat and uses a database to store:

QUESTION 8

In a MapReduce job, you want each of your 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?

A. Increase the parameter that controls minimum split size in the job configuration.

B. Write a custom MapRunner 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 isSplitable to always return false.

Correct Answer: D

FileInputFormat is the base class for all file-based InputFormats. This provides a generic implementation of getSplits(JobContext). Subclasses of FileInputFormat can also override the isSplitable(JobContext, Path) method to ensure input-files are not split-up and are processed as a whole by Mappers.

Reference: org.apache.hadoop.mapreduce.lib.input, Class FileInputFormat

QUESTION 9

MapReduce v2 (MRv2/YARN) is designed to address which two issues?

A. Single point of failure in the NameNode.

B. Resource pressure on the JobTracker.

C. HDFS latency.

D. Ability to run frameworks other than MapReduce, such as MPI.

E. Reduce complexity of the MapReduce APIs.

F. Standardize on a single MapReduce API.

Correct Answer: BD

YARN (Yet Another Resource Negotiator), as an aspect of Hadoop, has two major kinds of benefits:

*

(D) The ability to use programming frameworks other than MapReduce. / MPI (Message Passing Interface) was mentioned as a paradigmatic example of a MapReduce alternative

*

Scalability, no matter what programming framework you use. Note:

*

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.

*

(B) The central goal of YARN is to clearly separate two things that are unfortunately smushed together in current Hadoop, specifically in (mainly) JobTracker:

/ Monitoring the status of the cluster with respect to which nodes have which resources available. Under YARN, this will be global. / Managing the parallelization execution of any specific job. Under YARN, this will be done separately for each job. The current Hadoop MapReduce system is fairly scalable -- Yahoo runs 5000 Hadoop jobs, truly concurrently, on a single cluster, for a total 1.5 2 millions jobs/cluster/month. Still, YARN will remove scalability bottlenecks

Reference: Apache Hadoop YARN Concepts and Applications

QUESTION 10

You want to count the number of occurrences for each unique word in the supplied input data. You\\'ve decided to implement this by having your mapper tokenize each word and emit a literal value 1, and then have your reducer increment a counter for each literal 1 it receives. After successful implementing this, it occurs to you that you could optimize this by specifying a combiner. Will you be able to reuse your existing Reduces as your combiner in this case and why or why not?

A. Yes, because the sum operation is both associative and commutative and the input and output types to the reduce method match.

B. No, because the sum operation in the reducer is incompatible with the operation of a Combiner.

C. No, because the Reducer and Combiner are separate interfaces.

D. No, because the Combiner is incompatible with a mapper which doesn\\'t use the same data type for both the key and value.

E. Yes, because Java is a polymorphic object-oriented language and thus reducer code can be reused as a combiner.

Correct Answer: A

Combiners are used to increase the efficiency of a MapReduce program. They are used to aggregate intermediate map output locally on individual mapper outputs. Combiners can help you reduce the amount of data that needs to be transferred across to the reducers. You can use your reducer code as a combiner if the operation performed is commutative and associative. The execution of combiner is not guaranteed, Hadoop may or may not execute a combiner. Also, if required it may execute it more then 1 times. Therefore your MapReduce jobs should not depend on the combiners execution.

Reference: 24 Interview Questions and Answers for Hadoop MapReduce developers, What are combiners? When should I use a combiner in my MapReduce Job?