



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.passapply.com/ccd-410.html>

100% Passing Guarantee
100% Money Back Assurance

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

- ⚙️ **Instant Download** After Purchase
- ⚙️ **100% Money Back** Guarantee
- ⚙️ **365 Days** Free Update
- ⚙️ **800,000+** Satisfied Customers





QUESTION 1

What is the disadvantage of using multiple reducers with the default HashPartitioner and distributing your workload across your cluster?

- A. You will not be able to compress the intermediate data.
- B. You will no longer be able to take advantage of a Combiner.
- C. By using multiple reducers with the default HashPartitioner, output files may not be in globally sorted order.
- D. There are no concerns with this approach. It is always advisable to use multiple reducers.

Correct Answer: C

Multiple reducers and total ordering

If your sort job runs with multiple reducers (either because `mapreduce.job.reduces` in `mapred-site.xml` has been set to a number larger than 1, or because you've used the `-r` option to specify the number of reducers on the command-line), then by default Hadoop will use the HashPartitioner to distribute records across the reducers. Use of the HashPartitioner means that you can't concatenate your output files to create a single sorted output file. To do this you'll need total ordering,

Reference: Sorting text files with MapReduce

QUESTION 2

Determine which best describes when the reduce method is first called in a MapReduce job?

- A. Reducers start copying intermediate key-value pairs from each Mapper as soon as it has completed. The programmer can configure in the job what percentage of the intermediate data should arrive before the reduce method begins.
- B. Reducers start copying intermediate key-value pairs from each Mapper as soon as it has completed. The reduce method is called only after all intermediate data has been copied and sorted.
- C. Reduce methods and map methods all start at the beginning of a job, in order to provide optimal performance for map-only or reduce-only jobs.
- D. Reducers start copying intermediate key-value pairs from each Mapper as soon as it has completed. The reduce method is called as soon as the intermediate key-value pairs start to arrive.

Correct Answer: B

Reference: 24 Interview Questions and Answers for Hadoop MapReduce developers , When are the reducers started in a MapReduce job?

QUESTION 3

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 4

You write MapReduce job to process 100 files in HDFS. Your MapReduce algorithm uses TextInputFormat: the mapper applies a regular expression over input values and emits key- values pairs with the key consisting of the matching text, and the value containing the filename and byte offset. Determine the difference between setting the number of reduces to one and settings the number of reducers to zero.

- A. There is no difference in output between the two settings.
- B. With zero reducers, no reducer runs and the job throws an exception. With one reducer, instances of matching patterns are stored in a single file on HDFS.
- C. With zero reducers, all instances of matching patterns are gathered together in one file on HDFS. With one reducer, instances of matching patterns are stored in multiple files on HDFS.
- D. With zero reducers, instances of matching patterns are stored in multiple files on HDFS. With one reducer, all instances of matching patterns are gathered together in one file on HDFS.

Correct Answer: D

*

It is legal to set the number of reduce-tasks to zero if no reduction is desired.

In this case the outputs of the map-tasks go directly to the FileSystem, into the output path set by `setOutputPath(Path)`. The framework does not sort the map-outputs before writing them out to the FileSystem.

*

Often, you may want to process input data using a map function only. To do this, simply set `mapreduce.job.reduces` to zero. The MapReduce framework will not create any reducer tasks. Rather, the outputs of the mapper tasks will be the final output of the job.

Note:



Reduce

In this phase the `reduce(WritableComparable, Iterator, OutputCollector, Reporter)` method is called for each pair in the grouped inputs.

The output of the reduce task is typically written to the `FileSystem` via `OutputCollector.collect (WritableComparable, Writable)`.

Applications can use the `Reporter` to report progress, set application-level status messages and update `Counters`, or just indicate that they are alive.

The output of the Reducer is not sorted.

QUESTION 5

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

[CCD-410 PDF Dumps](#)

[CCD-410 VCE Dumps](#)

[CCD-410 Study Guide](#)