

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.leads4pass.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

You are developing a MapReduce job for sales reporting. The mapper will process input keys representing the year (IntWritable) and input values representing product identifies (Text). Identify what determines the data types used by the Mapper for a given job.

- A. The key and value types specified in the JobConf.setMapInputKeyClass and JobConf.setMapInputValuesClass methods
- B. The data types specified in HADOOP_MAP_DATATYPES environment variable
- C. The mapper-specification.xml file submitted with the job determine the mapper's input key and value types.
- D. The InputFormat used by the job determines the mapper's input key and value types.

Correct Answer: D

The input types fed to the mapper are controlled by the InputFormat used. The default input format, "TextInputFormat," will load data in as (LongWritable, Text) pairs. The long value is the byte offset of the line in the file. The Text object holds the string contents of the line of the file.

Note: The data types emitted by the reducer are identified by setOutputKeyClass() and setOutputValueClass(). The data types emitted by the reducer are identified by setOutputKeyClass() and setOutputValueClass().

By default, it is assumed that these are the output types of the mapper as well. If this is not the case, the methods setMapOutputKeyClass() and setMapOutputValueClass() methods of the JobConf class will override these.

Reference: Yahoo! Hadoop Tutorial, THE DRIVER METHOD

QUESTION 2

You've written a MapReduce job that will process 500 million input records and generated 500 million keyvalue pairs. The data is not uniformly distributed. Your MapReduce job will create a significant amount of intermediate data that it needs to transfer between mappers and reduces which is a potential bottleneck. A custom implementation of which interface is most likely to reduce the amount of intermediate data transferred across the network?

- A. Partitioner
- B. OutputFormat
- C. WritableComparable
- D. Writable
- E. InputFormat

F. Combiner

Correct Answer: F

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.

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

QUESTION 3

You want to populate an associative array in order to perform a map-side join. You've decided to put this information in a text file, place that file into the DistributedCache and read it in your Mapper before any records are processed.

Identify which method in the Mapper you should use to implement code for reading the file and populating the associative array?

A. combine

B. map

C. init

D. configure

Correct Answer: B

Reference: org.apache.hadoop.filecache , Class DistributedCache

QUESTION 4

You wrote a map function that throws a runtime exception when it encounters a control character in input data. The input supplied to your mapper contains twelve such characters total, spread across five file splits. The first four file splits each have two control characters and the last split has four control characters.

Identify the number of failed task attempts you can expect when you run the job with `mapred.max.map.attempts` set to 4:

A. You will have forty-eight failed task attempts

B. You will have seventeen failed task attempts

C. You will have five failed task attempts

D. You will have twelve failed task attempts

E. You will have twenty failed task attempts

Correct Answer: E

There will be four failed task attempts for each of the five file splits.

Note:

When the jobtracker is notified of a task attempt that has failed (by the tasktracker's heartbeat call), it will reschedule execution of the task. The jobtracker will try to avoid rescheduling the task on a tasktracker where it has previously failed. Furthermore, if a task fails four times (or more), it will not be retried further. This value is configurable: the maximum number of attempts to run a task is controlled by the `mapred.map.max.attempts` property for map tasks and `mapred.reduce.max.attempts` for reduce tasks. By default, if any task fails four times (or whatever the maximum number of attempts is configured to), the whole job fails.

QUESTION 5

What types of algorithms are difficult to express in MapReduce v1 (MRv1)?

- A. Algorithms that require applying the same mathematical function to large numbers of individual binary records.
- B. Relational operations on large amounts of structured and semi-structured data.
- C. Algorithms that require global, sharing states.
- D. Large-scale graph algorithms that require one-step link traversal.
- E. Text analysis algorithms on large collections of unstructured text (e.g, Web crawls).

Correct Answer: C

See 3) below.

Limitations of Mapreduce where not to use Mapreduce While very powerful and applicable to a wide variety of problems, MapReduce is not the answer to every problem. Here are some problems I found where MapReduce is not suited and some papers that address the limitations of MapReduce.

1.

Computation depends on previously computed values

If the computation of a value depends on previously computed values, then MapReduce cannot be used. One good example is the Fibonacci series where each value is summation of the previous two values. i.e., $f(k+2) = f(k+1) + f(k)$. Also, if the data set is small enough to be computed on a single machine, then it is better to do it as a single `reduce(map(data))` operation rather than going through the entire map reduce process.

2.

Full-text indexing or ad hoc searching

The index generated in the Map step is one dimensional, and the Reduce step must not generate a large amount of data or there will be a serious performance degradation. For example, CouchDB's MapReduce may not be a good fit for full-text indexing or ad hoc searching. This is a problem better suited for a tool such as Lucene.

3.

Algorithms depend on shared global state

Solutions to many interesting problems in text processing do not require global synchronization. As a result, they can be expressed naturally in MapReduce, since map and reduce tasks run independently and in isolation. However, there are many examples of algorithms that depend crucially on the existence of shared global state during processing, making them difficult to implement in MapReduce (since the single opportunity for global synchronization in MapReduce is the barrier between the map and reduce phases of processing)

Reference: Limitations of Mapreduce where not to use Mapreduce

[CCD-410 Practice Test](#)

[CCD-410 Exam Questions](#)

[CCD-410 Braindumps](#)