1、如何解除Hadoop系统的安全模式？

输入命令：hadoop dfsadmin -safemode leave

2、为什么HDFS不适合低延迟数据访问？

HDFS是为了处理大型数据集分析任务的，主要是为达到高的数据吞吐量而设计的，这就可能要求以高延迟作为代价。

3、为什么HDFS无法高效存储大量小文件？

因为namenode将文件系统的元数据存放在内存中，因此存储的文件数目受限于 namenode的内存大小。如果存放的文件数目过多的话会占用很大的内存。另外，HDFS适用于高吞吐量，而不适合低时间延迟的访问。如果同时存入大量的小文件会花费很长的时间。除此之外，数据是分块存储的，默认大小为64MB，存储大量文件大小小于块大小的文件也将造成磁盘空间的浪费。

4、为什么HDFS不支持多用户写入及任意修改文件？

因为HDFS采用流式读取的方式，不适合多用户写入，以及任意位置写入。因此在HDFS的一个文件中只有一个写入者，而且写操作只能在文件末尾完成，即只能执行追加操作。

在Apache Hadoop文档（<http://hadoop.apache.org/docs/r1.0.4/cn/hdfs_design.html）中的> “简单的一致性模型” 中写到：*HDFS应用需要一个“一次写入多次读取”的文件访问模型。一个文件经过创建、写入和关闭之后就不需要改变。这一假设简化了数据一致性问题，并且使高吞吐量的数据访问成为可能。Map/Reduce应用或者网络爬虫应用都非常适合这个模型。目前还有计划在将来扩充这个模型，使之支持文件的附加写操作。*

另外，由于数据分块存储且存在备份，如果在一个大文件的中间增加内容，则会影响到后面所有的数据块。

5、HDFS如何解决名称节点运行期间EditLog不断变大的问题？

使用SecondaryNameNode。

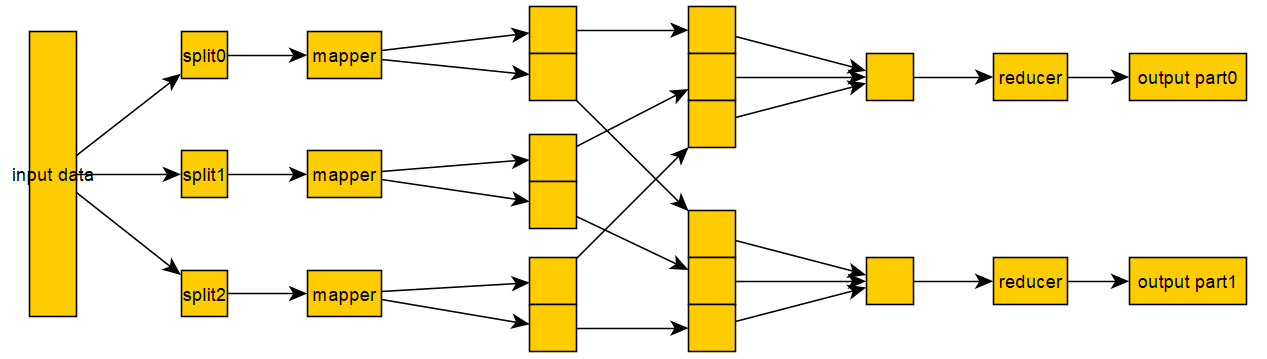
6、Map任务为什么要进行分区操作，如何分区？

分区操作能把map任务处理的结果发送给指定的reducer去执行，从而达到负载均衡，并且保证相同key的数据被同一个reducer处理。默认情况下，partitioner先计算key的哈希值，然后通过reducer的个数执行取模运算，这样能够将整个key空间平均分发给每个reducer,同时也能确保不同mapper产生的相同key能被分发到同一个reducer。

7、MapReduce中多个Reduce任务的结果会做进一步的处理么?为什么？

不会，默认直接落入本地磁盘，但是可以作为输入交给另一个MapReduce程序处理。

8、如果有两个Reduce，重新画出用户没有定义Combiner时的MapReduce过程示意图，并简要说明Shuffle过程。



Shuffle过程：

接收Mapper内存缓冲区数据，对数据先进行分区，然后按分区号排序，再按key排序，再将多个spill\_file做merge操作，一个Map Task最终只生成一个中间数据文件。之后Reduce进程启动一些数据copy线程(Fetcher)，获取已完成的Map Task的文件中属于自己的部分，先放入缓存，然后merge成一个文件，送至Reducer。