1、步骤为：

（1）通过命令行登录到Hadoop的NameNode服务器上。

（2）执行以下命令，以退出安全模式：hdfs dfsadmin -safemode leave

如果需要强制退出安全模式，可以添加 -force 参数： hdfs dfsadmin -safemode leave -force

（3）执行命令后，系统会输出一条消息，告诉你是否已成功退出安全模式。

（4）检查Hadoop系统是否已经退出安全模式，可以使用以下命令查看： hdfs dfsadmin -safemode get 如果输出结果为“Safe mode is OFF”，则表示系统已经成功退出安全模式。

2、原因：

（1）写入延迟高：HDFS的写入操作需要先将数据写入到本地磁盘，再复制到多个节点上进行备份。因此，HDFS的写入延迟比较高，无法满足低延迟数据访问的要求。

（2）数据一致性难以保证：由于数据备份和同步需要一定的时间，因此HDFS不能保证实时的数据一致性，可能会导致数据访问时数据不一致的情况发生。

（3）适用于大文件存储：HDFS适用于存储大文件，对小文件存储效率较低。当存储大量小文件时，HDFS需要维护大量的元数据，会导致元数据的负载过高，影响系统性能。

（4）不支持原子性操作：HDFS不支持原子性操作，如文件的追加和删除等操作，可能会导致数据访问的不一致性。

3、原因：

（1） 存储元数据开销大：HDFS需要维护大量的元数据，如文件名、权限、时间戳等，存储大量小文件时，需要为每个文件维护一个元数据，这会导致存储空间的浪费。

（2）存储效率低：HDFS是按照数据块的方式存储文件，每个数据块的大小一般为128MB或256MB，如果存储小文件，则会造成空间的浪费。当一个小文件占用一个数据块时，其余的空间将被浪费，这会导致存储效率低下。

（3）读写效率低：由于HDFS需要访问大量的元数据，所以读写小文件时需要频繁地访问磁盘和网络，这会导致读写效率低下，影响系统的性能。

4、原因：

（1） 一致性：HDFS的设计目标是提供高可靠性和数据一致性。在多用户并发地写入和任意修改文件的情况下，可能会导致数据不一致，因此HDFS限制了这些操作。

（2）写入性能：在多用户并发地写入的情况下，需要进行锁定和同步，这会导致写入性能降低，不符合HDFS的设计目标。

（3）安全性：如果允许多用户并发地写入和任意修改文件，可能会导致数据的安全性问题，例如数据的篡改或者丢失。

（4）数据可靠性：HDFS提供了数据备份和容错机制来保证数据的可靠性，如果多个用户并发地写入和修改文件，将会影响数据备份和容错机制的实现。

5、方法：

（1）Checkpoint：Checkpoint是一种名称节点的备份机制，可以将名称节点的命名空间和元数据备份到另一个位置。在进行Checkpoint备份时，HDFS会将EditLog文件中的已提交操作复制到新的EditLog文件中，并将旧的EditLog文件截断。这样可以有效地减少EditLog文件的大小，同时也可以保证数据的安全性。

（2）EditLog Roll：EditLog Roll是一种策略，用于控制EditLog文件的大小。当EditLog文件的大小达到预设的阈值时，HDFS会自动创建一个新的EditLog文件，并将当前的EditLog文件命名为旧的EditLog文件。这样可以避免EditLog文件不断增长的问题，同时也可以更好地管理和维护EditLog文件。

（3） EditLog压缩：在进行Checkpoint备份时，HDFS还可以对EditLog文件进行压缩，这样可以进一步减小EditLog文件的大小。HDFS使用一种称为Hadoop SequenceFile的格式来压缩EditLog文件，该格式可以高效地存储和检索序列化的键值对。

6、Map任务需要对输入数据进行处理，并将其转换成键值对。在进行Map操作时，会产生大量的键值对，这些键值对需要被分配到不同的Reducer节点进行处理。为了避免所有键值对都被发送到同一个Reducer节点，Map任务需要进行分区操作，将不同的键值对分配到不同的Reducer节点上进行处理，从而实现负载均衡，提高MapReduce的效率。

分区操作的实现方式是通过对键值对的键进行哈希操作，将哈希值和键值对发送到不同的Reducer节点。具体实现过程中，需要在Map任务中定义一个分区函数（Partitioner），该函数将键值对作为输入，并返回一个整数值，该值将被用于将键值对分配到不同的Reducer节点。通常情况下，分区函数可以简单地对键进行哈希操作，并取模得到一个整数值，然后将这个整数值作为分区的编号，从而将键值对分配到对应的Reducer节点上。

7、在MapReduce中，多个Reduce任务的结果通常需要进一步处理，因为MapReduce框架本身只能处理简单的聚合操作，例如求和、计数等。而在实际应用中，通常需要进行更复杂的数据分析和处理，例如排序、分组、连接等操作，这些操作无法直接在Reduce阶段中完成。

因此，MapReduce通常会将Reduce任务的结果写入到分布式文件系统中（例如HDFS），然后通过其他的数据处理框架（例如Hive、Pig等）对数据进行进一步的处理和分析。这些数据处理框架提供了更丰富的操作，可以对大规模的数据进行高效的处理和分析，从而满足更加复杂的数据处理需求。