**一、**  
1、hadoop集群搭建过程，写出步骤。  
2、hadoop集群运行过程中启动那些线程，各自的作用是什么？  
  
3、/tmp/hadoop-root/dfs/name    the path is not exists or is not accessable.  
    NameNode main中报错，该怎么解决。（大意这样 一个什么异常）  
  
4、工作中编写mapreduce用到的语言，编写一个mapreduce程序。  
  
5、hadoop命令  
   1）杀死一个job任务   （杀死50030端口的进程即可）  
   2）删除/tmp/aaa文件目录  
   3）hadoop集群添加或删除节点时，刷新集群状态的命令  
  
6、日志的固定格式：  
    a,b,c,d  
    a,a,f,e  
    b,b,d,f  
   使用一种语言编写mapreduce任务，统计每一列最后字母的个数。  
  
7、hadoop的调度器有哪些，工作原理。  
  
8、mapreduce的join方法有哪些？  
  
9、Hive元数据保存的方法有哪些，各有什么特点？  
  
10、java实现非递归二分法算法。  
11、mapreduce中Combiner和Partition的作用。  
  
12、用linux实现下列要求：

1. ip             username
2. a.txt
3. 210.121.123.12 zhangsan
4. 34.23.56.78    lisi
5. 11.56.56.72    wanger
6. .....
7. b.txt
8. 58.23.53.132   liuqi
9. 34.23.56.78    liba
10. .....

复制代码

  a.txt,b.txt中至少100万行。  
  1）a.txt,b.txt中各自的ip个数，ip的总个数。  
  2）a.txt中存在的ip而b.txt中不存在的ip。  
  3）每个username出现的总个数，每个username对应的ip个数。  
  
13、大意是 hadoop中java、streaming、pipe处理数据各有特点。  
  
14、如何实现mapreduce的二次排序。  
  
  
**二、**  
15、面试官上来就问hadoop的调度机制，  
  
16、机架感知，  
  
17、MR数据倾斜原因和解决方案，  
  
18、集群HA  
  
**三、**  
19、如果让你设计，你觉得一个分布式文件系统应该如何设计，考虑哪方面内容；  
每天百亿数据入hbase，如何保证数据的存储正确和在规定的时间里全部录入完毕，  
不残留数据。  
  
20、对于hive，你写过哪些UDF函数，作用是什么  
  
21、hdfs的数据压缩算法  
  
22、mapreduce的调度模式  
  
23、hive底层与数据库交互原理  
  
24、hbase过滤器实现原则  
  
25、对于mahout，如何进行推荐、分类、聚类的代码二次开发分别实现那些借口  
  
  
**四、**  
26、请问下，直接将时间戳作为行健，在写入单个region时候会发生热点问题，为什么呢？

**尽信书不如无书，尽信答案不如无答案，下面只供参考：**  
  
**1、hadoop运行的原理?**  
hadoop主要由三方面组成:  
1、HDFS  
2、MapReduce  
3、Hbase  
Hadoop框架中最核心的设计就是：MapReduce和HDFS。MapReduce的思想是由Google的一篇论文所提及而被广为流传的， 简单的一句话解释MapReduce就是“任务的分解与结果的汇总”。HDFS是Hadoop分布式文件系统（Hadoop Distributed File System）的缩写 ，为分布式计算存储提供了底层支持。  
MapReduce从它名字上来看就大致可以看出个缘由，两个动词Map和Reduce，“Map（展开）”就是将一个任务分解成为多个任 务，“Reduce”就是将分解后多任务处理的结果汇总起来，得出最后的分析结果。这不是什么新思想，其实在前面提到的多线程，多任务的设计就可以找到这 种思想的影子。不论是现实社会，还是在程序设计中，一项工作往往可以被拆分成为多个任务，任务之间的关系可以分为两种：一种是不相关的任务，可以并行执 行；另一种是任务之间有相互的依赖，先后顺序不能够颠倒，这类任务是无法并行处理的。回到大学时期，教授上课时让大家去分析关键路径，无非就是找最省时的 任务分解执行方式。在分布式系统中，机器集群就可以看作硬件资源池，将并行的任务拆分，然后交由每一个空闲机器资源去处理，能够极大地提高计算效率，同时 这种资源无关性，对于计算集群的扩展无疑提供了最好的设计保证。（其实我一直认为Hadoop的卡通图标不应该是一个小象，应该是蚂蚁，分布式计算就好比 蚂蚁吃大象，廉价的机器群可以匹敌任何高性能的计算机，纵向扩展的曲线始终敌不过横向扩展的斜线）。任务分解处理以后，那就需要将处理以后的结果再汇总起 来，这就是Reduce要做的工作。  
  
**2、mapreduce的原理?**  
Hadoop中的MapReduce是一个使用简易的软件框架，基于它写出来的应用程序能够运行在由上千个商用机器组成的大型集群上，并以一种可靠容错的式并  行处理上T级别的数据集。  
一个MapReduce作业（job）通常会把输入的数据集切分为若干独立的数据块，由map任务（task）以完全并行的方式处理它们。框架会对map的输出先进行排序，然后把结果输入给reduce任务。通常作业的输入和输出都会被存储在文件系统中。整个框架负责任务的调度和监控，以及重新执行已经失败的任务。  
通常，MapReduce框架和分布式文件系统是运行在一组相同的节点上的，也就是说，计算节点和存储节点通常在一起。这种配置允许框架在那些已经存好数据的节点上高效地调度任务，这可以使整个集群的网络带宽被非常高效地利用。  
MapReduce框架由一个单独的master JobTracker和每个集群节点一个slave TaskTracker共同组成。master负责调度构成一个作业的所有任务，这些任务分布在不同的slave上，master监控它们的执行，重新执行已经失败的任务。而slave仅负责执行由master指派的任务  
  
**3、HDFS存储的机制?**  
HDFS的三个实体  
数据块  
每个磁盘都有默认的数据块大小,这是磁盘进行读写的基本单位.构建于单个磁盘之上的文件系统通过磁盘块来管理该文件系统中的块.该文件系统中的块一般为磁盘块的整数倍.磁盘块一般为512字节.HDFS也有块的概念,默认为64MB(一个map处理的数据大小).HDFS上的文件也被划分为块大小的多个分块,与其他文件系统不同的是,HDFS中小于一个块大小的文件不会占据整个块的空间.  
HDFS用块存储带来的第一个明显的好处一个文件的大小可以大于网络中任意一个磁盘的容量,数据块可以利用磁盘中任意一个磁盘进行存储.第二个简化了系统的设计,将控制单元设置为块,可简化存储管理,计算单个磁盘能存储多少块就相对容易.同时也消除了对元数据的顾虑,如权限信息,可以由其他系统单独管理.  
DataNode节点  
DataNode是HDFS文件系统的工作节点,它们根据需要存储并检索数据块,受NameNode节点调度.并且定期向NameNode发送它们所存储的块的列表  
NameNode节点  
NameNode管理HDFS文件系统的命名空间,它维护着文件系统树及整棵树的所有的文件及目录.这些文件以两个文件形式永久保存在本地磁盘上(命名空间镜像文件和编辑日志文件).NameNode记录着每个文件中各个块所在的数据节点信息但并不永久保存这些块的位置信息,因为这些信息在系统启动时由数据节点重建.  
没有NameNode,文件系统将无法使用.如提供NameNode服务的机器损坏,文件系统上的所有文件丢失,我们就不能根据DataNode的块来重建文件.因此,对NameNode的容错非常重要.第一种机制,备份那些组成文件系统元数据持久状态的文件.通过配置使NameNode在多个文件系统上保存元数据的持久状态或将数据写入本地磁盘的同时,写入一个远程挂载的网络文件系统.当然这些操作都是原子操作.第二种机制是运行一个辅助的NameNode,它会保存合并后的命名空间镜像的副本,并在Name/Node发生故障时启用.但是辅助NameNode保存.态总是滞后于主力节点,所以在主节点全部失效后难免丢失数据.在这种情况下,一般把存储在远程挂载的网络文件系统的数据复制到辅助NameNode并作为新的主NameNode运行  
  
**4、举一个简单的例子说明mapreduce是怎么来运行的 ?**  
  
**5、面试的人给你出一些问题,让你用mapreduce来实现？**  
比如:现在有10个文件夹,每个文件夹都有1000000个url.现在让你找出top1000000url。  
  
**6、hadoop中Combiner的作用?**  
1、combiner最基本是实现本地key的聚合，对map输出的key排序，value进行迭代。如下所示：  
map: (K1, V1) → list(K2, V2)  
combine: (K2, list(V2)) → list(K2, V2)  
reduce: (K2, list(V2)) → list(K3, V3)  
2、combiner还具有类似本地的reduce功能.  
例如hadoop自带的wordcount的例子和找出value的最大值的程序，combiner和reduce完全一致。如下所示：  
map: (K1, V1) → list(K2, V2)  
combine: (K2, list(V2)) → list(K3, V3)  
reduce: (K3, list(V3)) → list(K4, V4)  
3、如果不用combiner，那么，所有的结果都是reduce完成，效率会相对低下。使用combiner，先完成的map会在本地聚合，提升速度。  
4、对于hadoop自带的wordcount的例子，value就是一个叠加的数字，所以map一结束就可以进行reduce的value叠加，而不必要等到所有的map结束再去进行reduce的value叠加。  
combiner使用的合适，可以在满足业务的情况下提升job的速度，如果不合适，则将导致输出的结果不正确。

**1.**简要描述如何安装配置一个apache开源版hadoop，描述即可，列出步骤更好  
  
**2.**请列出正常工作的hadoop集群中hadoop都需要启动哪些进程，他们的作用分别是什么？  
  
**3.**启动hadoop报如下错误，该如何解决？  
error  org.apache.hadoop.hdfs.server.namenode.NameNode   
  
org.apache.hadoop.hdfs.server.common.inconsistentFSStateExceptio  
  
n Directory /tmp/hadoop-root/dfs/name is in an inconsistent   
  
state storage direction does not exist or is not accessible?  
  
**4.**请写出以下执行命令  
1）杀死一个job?  
2)删除hdfs上的/tmp/aaa目录  
3加入一个新的存储节点和删除一个计算节点需要刷新集群状态命令？  
  
**5.**请列出你所知道的hadoop调度器，并简要说明其工作方法？  
**6.**请列出在你以前工作中所使用过的开发mapreduce的语言？  
**7.**当前日志采样格式为

1. a,b,c,d
2. b,b,f,e
3. a,a,c,f

复制代码

请用你最熟悉的语言编写一个mapreduce，并计算第四列每个元素出现的个数  
  
  
**8.**你认为用Java，Streaming,pipe方式开发mapreduce,各有哪些优缺点？  
**9.**hive有哪些方式保存元数据，各有哪些特点？  
**10.**请简述hadoop怎么样实现二级排序？  
**11.**简述hadoop实现join的几种方法？  
**12.**请用Java实现非递归二分查找？  
**13.**请简述mapreduce中，combiner，partition作用？  
**14.**某个目录下有两个文件a.txt和b.txt,文件格式为（ip，username）,  
  
列如：  
  
a.txt  
127.0.0.1 zhangsan  
127.0.0.1 wangxiaoer  
127.0.0.2 lisi  
127.0.0.3 wangwu  
  
b.txt  
127.0.0.4 lixiaolu  
127.0.0.1 lisi  
  
每个文件至少100万行，请使用Linux命令完成如下工作：  
  
1）每个文件各自的ip数  
2）出现在b.txt而没有出现在a.txt的ip  
3）每个user出现的次数以及每个user对应的ip数

**1、简答说一下hadoop的map-reduce编程模型**

  首先map task会从本地文件系统读取数据，转换成key-value形式的键值对集合，使用的是hadoop内置的数据类型，比如longwritable、text等，将键值对集合输入mapper进行业务处理过程，将其转换成需要的key-value在输出之后会进行一个partition分区操作，默认使用的是hashpartitioner，可以通过重写hashpartitioner的getpartition方法来自定义分区规则；之后会对key进行进行sort排序，grouping分组操作将相同key的value合并分组输出，在这里可以使用自定义的数据类型，重写WritableComparator的Comparator方法来自定义排序规则，重写RawComparator的compara方法来自定义分组规则，之后进行一个combiner归约操作，其实就是一个本地段的reduce预处理，以减小后面shufle和reducer的工作量，reduce task会通过网络将各个数据收集进行reduce处理，最后将数据保存或者显示，结束整个job。

**2、hadoop的TextInputFormat作用是什么，如何自定义实现**

InputFormat会在map操作之前对数据进行两方面的预处理

1、是getSplits，返回的是InputSplit数组，对数据进行split分片，每片交给map操作一次

2、是getRecordReader，返回的是RecordReader对象，对每个split分片进行转换为key-value键值对格式传递给map，常用的InputFormat是TextInputFormat，使用的是LineRecordReader对每个分片进行键值对的转换，以行偏移量作为键，行内容作为值，自定义类继承InputFormat接口，重写createRecordReader和isSplitable方法在createRecordReader中可以自定义分隔符。

**3、hadoop和spark的都是并行计算，那么他们有什么相同和区别**

两者都是用mr模型来进行并行计算，hadoop的一个作业称为job，job里面分为map task和reduce task，每个task都是在自己的进程中运行的，当task结束时，进程也会结束

spark用户提交的任务成为application，一个application对应一个sparkcontext，app中存在多个job，每触发一次action操作就会产生一个job，这些job可以并行或串行执行，每个job中有多个stage，stage是shuffle过程中DAGSchaduler通过RDD之间的依赖关系划分job而来的，每个stage里面有多个task，组成taskset有TaskSchaduler分发到各个executor中执行，executor的生命周期是和app一样的，即使没有job运行也是存在的，所以task可以快速启动读取内存进行计算，hadoop的job只有map和reduce操作，表达能力比较欠缺而且在mr过程中会重复的读写hdfs，造成大量的io操作，多个job需要自己管理关系spark的迭代计算都是在内存中进行的，API中提供了大量的RDD操作如join，groupby等，而且通过DAG图可以实现良好的容错

**4、为什么要用flume导入hdfs，hdfs的构架是怎样的**

flume可以实时的导入数据到hdfs中，当hdfs上的文件达到一个指定大小的时候会形成一个文件，或者超过指定时间的话也形成一个文件，文件都是存储在datanode上面的，namenode记录着datanode的元数据信息，而namenode的元数据信息是存在内存中的，所以当文件切片很小或者很多的时候会卡死

**5、map-reduce程序运行的时候会有什么比较常见的问题**

比如说作业中大部分都完成了，但是总有几个reduce一直在运行，这是因为这几个reduce中的处理的数据要远远大于其他的reduce，可能是因为对键值对任务划分的不均匀造成的数据倾斜，解决的方法可以在分区的时候重新定义分区规则对于value数据很多的key可以进行拆分、均匀打散等处理，或者是在map端的combiner中进行数据预处理的操作

**6、简单说一下hadoop和spark的shuffle过程**

hadoop：map端保存分片数据，通过网络收集到reduce端

spark：spark的shuffle是在DAGSchedular划分Stage的时候产生的，TaskSchedule要分发Stage到各个worker的executor，减少shuffle可以提高性能

**7、Hive中存放是什么？**

表。存的是和hdfs的映射关系，hive是逻辑上的数据仓库，实际操作的都是hdfs上的文件，HQL就是用sql语法来写的mr程序。

**8、Hive与关系型数据库的关系？**

没有关系，hive是数据仓库，不能和数据库一样进行实时的CURD操作。是一次写入多次读取的操作，可以看成是ETL工具。

**9、Flume工作机制是什么？**

核心概念是agent，里面包括source、chanel和sink三个组件。

source运行在日志收集节点进行日志采集，之后临时存储在chanel中，sink负责将chanel中的数据发送到目的地。只有成功发送之后chanel中的数据才会被删除。首先书写flume配置文件，定义agent、source、chanel和sink然后将其组装，执行flume-ng命令。

**10、Sqoop工作原理是什么？**

hadoop生态圈上的数据传输工具。可以将关系型数据库的数据导入非结构化的hdfs、hive或者bbase中，也可以将hdfs中的数据导出到关系型数据库或者文本文件中。使用的是mr程序来执行任务，使用jdbc和关系型数据库进行交互。

import：原理：通过指定的分隔符进行数据切分，将分片传入各个map中，在map任务中在每行数据进行写入处理没有reduce。

export：原理：根据要操作的表名生成一个java类，并读取其元数据信息和分隔符对非结构化的数据进行匹配，多个map作业同时执行写入关系型数据库

**11、Hbase行健列族的概念，物理模型，表的设计原则？**

行健：是hbase表自带的，每个行健对应一条数据。列族：是创建表时指定的，为列的集合，每个列族作为一个文件单独存储，存储的数据都是字节数组，其中的数据可以有很多，通过时间戳来区分。物理模型：整个hbase表会拆分为多个region，每个region记录着行健的起始点保存在不同的节点上，查询时就是对各个节点的并行查询，当region很大时使用.META表存储各个region的起始点，-ROOT又可以存储.META的起始点。

rowkey

的设计原则：各个列簇数据平衡，长度原则、相邻原则，创建表的时候设置表放入regionserver缓存中，避免自动增长和时间，使用字节数组代替string，最大长度64kb，最好16字节以内，按天分表，两个字节散列，四个字节存储时分毫秒。列族的设计原则：尽可能少（按照列族进行存储，按照region进行读取，不必要的io操作），经常和不经常使用的两类数据放入不同列族中，列族名字尽可能短。

**12、Spark Streaming和Storm有何区别？**

一个实时毫秒一个准实时亚秒，不过storm的吞吐率比较低。

**13、mllib支持的算法？**

大体分为四大类，分类、聚类、回归、协同过滤。

**15、Hadoop平台集群配置、环境变量设置？**

zookeeper：修改zoo.cfg文件，配置dataDir，和各个zk节点的server地址端口，tickTime心跳时间默认是2000ms，其他超时的时间都是以这个为基础的整数倍，之后再dataDir对应目录下写入myid文件和zoo.cfg中的server相对应。

hadoop：修改hadoop-env.sh配置java环境变量

core-site.xml：配置zk地址，临时目录等

hdfs-site.xml：配置nn信息，rpc和http通信地址，nn自动切换、zk连接超时时间等

yarn-site.xml：配置resourcemanager地址

mapred-site.xml：配置使用yarn、slaves、配置节点信息格式化nn和zk。

hbase：修改hbase-env.sh配置java环境变量和是否使用自带的zk

hbase-site.xml：配置hdfs上数据存放路径，zk地址和通讯超时时间、master节点

regionservers：配置各个region节点

zoo.cfg拷贝到conf目录下

spark：安装Scala，修改spark-env.sh配置环境变量和master和worker节点配置信息

环境变量的设置：直接在/etc/profile中配置安装的路径即可，或者在当前用户的宿主目录下，配置在.bashrc文件中，该文件不用source重新打开shell窗口即可，配置在.bash\_profile的话只对当前用户有效。

**16、Hadoop性能调优？**

调优可以通过系统配置、程序编写和作业调度算法来进行。

hdfs：的block.size可以调到128/256（网络很好的情况下，默认为64）调优的大头：mapred.map.tasks、mapred.reduce.tasks设置mr任务数（默认都是1）

mapred.tasktracker.map.tasks.maximum

每台机器上的最大map任务数

mapred.tasktracker.reduce.tasks.maximum

每台机器上的最大reduce任务数

mapred.reduce.slowstart.completed.maps

配置reduce任务在map任务完成到百分之几的时候开始进入这个几个参数要看实际节点的情况进行配置，reduce任务是在33%的时候完成copy，要在这之前完成map任务，（map可以提前完成）

mapred.compress.map.output,mapred.output.compress

配置压缩项，消耗cpu提升网络和磁盘io

合理利用combiner

注意重用writable对象

**17、Hadoop高并发？**

首先肯定要保证集群的高可靠性，在高并发的情况下不会挂掉，支撑不住可以通过横向扩展。

datanode

挂掉了使用hadoop脚本重新启动。

**27、kafka工作原理？**

producer向broker发送事件，consumer从broker消费事件。事件由topic区分开，每个consumer都会属于一个group。相同group中的consumer不能重复消费事件，而同一事件将会发送给每个不同group的consumer。

**28、ALS算法原理？**

答：对于user-product-rating数据，als会建立一个稀疏的评分矩阵，其目的就是通过一定的规则填满这个稀疏矩阵。

als

会对稀疏矩阵进行分解，分为用户-特征值，产品-特征值，一个用户对一个产品的评分可以由这两个矩阵相乘得到。通过固定一个未知的特征值，计算另外一个特征值，然后交替反复进行最小二乘法，直至差平方和最小，即可得想要的矩阵。

**29、kmeans算法原理？**

随机初始化中心点范围，计算各个类别的平均值得到新的中心点。重新计算各个点到中心值的距离划分，再次计算平均值得到新的中心点，直至各个类别数据平均值无变化。

**30、canopy算法原理？**

根据两个阈值来划分数据，以随机的一个数据点作为canopy中心。计算其他数据点到其的距离，划入t1、t2中，划入t2的从数据集中删除，划入t1的其他数据点继续计算，直至数据集中无数据。

**31、朴素贝叶斯分类算法原理？**

对于待分类的数据和分类项，根据待分类数据的各个特征属性，出现在各个分类项中的概率判断该数据是属于哪个类别的。

**32、关联规则挖掘算法apriori原理？**

一个频繁项集的子集也是频繁项集，针对数据得出每个产品的支持数列表，过滤支持数小于预设值的项，对剩下的项进行全排列，重新计算支持数，再次过滤，重复至全排列结束，可得到频繁项和对应的支持数。