1.讲述HDFS上传文件和读文件的流程?

> HDFS 上传流程，举例说明一个256M的文件上传过程

(1)由客户端Client向NameNode节点发出请求;

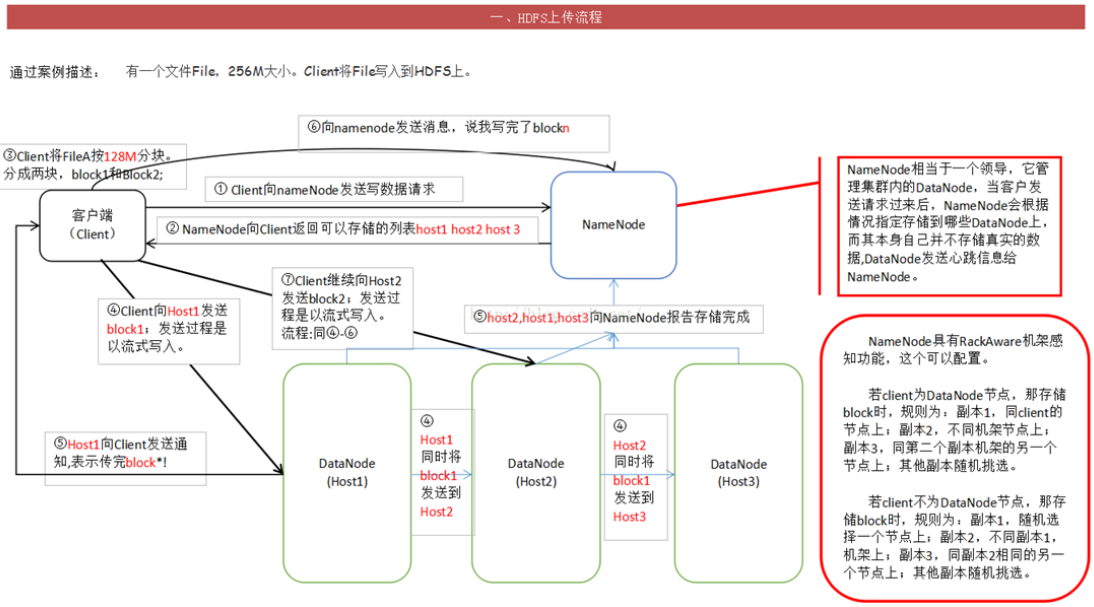
(2)NameNode向Client返回可以存数据的DataNode列表，这里遵循机架感应原则(把副本分别放在不同的机架，甚至不同的数据中心);

(3)客户端首先根据返回的信息先将文件分块(Hadoop2.X版本每一个block为 128M，而之前的版本为 64M);

(4)通过NameNode返回的DataNode信息，将文件块以流失写入方式直接发送给DataNode，同时复制到其他两台机器(默认一份数据，有两个副本);

(5)数据块传送完成以后，dataNode向Client通信，同时向NameNode报告;

(6)依照上面(4)到(5)的原理将所有的数据块都上传，结束后向 NameNode 报告 表明已经传完所有的数据块。



2.HDFS在上传文件的时候，如果其中一个块突然损坏了怎么办?

其中一个块坏了，只要有其它块存在，会自动检测还原。

3.NameNode的作用?

namenode总体来说是管理和记录恢复功能。比如管理datanode，保持心跳，如果超时则排除。对于上传文件都有镜像images和edits,这些可以用来恢复。

4.NameNode在启动的时候会做哪些操作?

NameNode启动的时候，会加载fsimage

Fsimage加载过程完成的操作主要是为了:

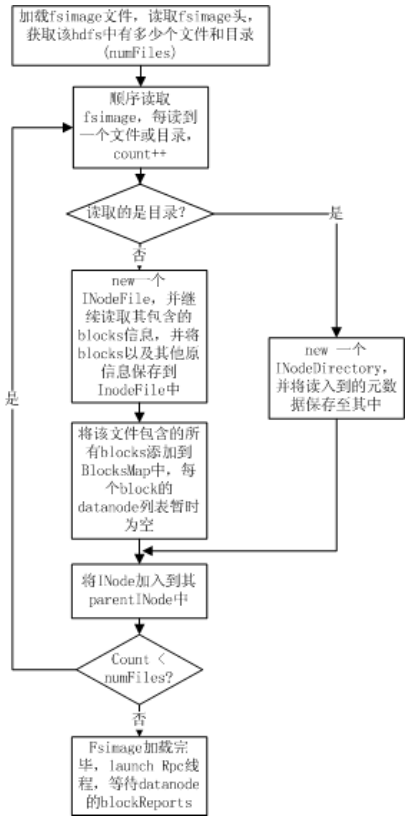
(1)从fsimage中读取该HDFS中保存的每一个目录和每一个文件

(2)初始化每个目录和文件的元数据信息

(3)根据目录和文件的路径，构造出整个namespace在内存中的镜像

(4)如果是文件，则读取出该文件包含的所有blockid，并插入到BlocksMap中。

整个加载流程如下图所示:



如上图所示，namenode在加载fsimage过程其实非常简单，就是从fsimage中不停的顺序读取文件和目录的元数据信息，并在内存中构建整个namespace，同时将每个文件对应的blockid保存入BlocksMap中，此时BlocksMap中每个block对应的datanodes列表暂时为空。当fsimage加载完毕后，整个HDFS的目录结构在内存中就已经初始化完毕，所缺的就是每个文件对应的block对应的datanode列表信息。这些信息需要从datanode的blockReport中获取，所以加载fsimage完毕后，namenode进程进入rpc等待状态，等待所有的datanodes发送blockReports。

5.NameNode的HA?

NameNode的HA一个备用，一个工作，且一个失败后，另一个被激活。他们通过journal node来实现共享数据。

6.Hadoop的作业提交流程?

Hadoop2.x Yarn作业提交（客户端）

http://www.aboutyun.com/forum.php?mod=viewthread&tid=9498

Hadoop2.x Yarn作业提交（服务端）

http://www.aboutyun.com/forum.php?mod=viewthread&tid=9496

7.Hadoop怎么分片?

1、分块:

HDFS存储系统中，引入了文件系统的分块概念（block），块是存储的最小单位，HDFS定义其大小为64MB。与单磁盘文件系统相似，存储在 HDFS上的文件均存储为多个块，不同的是，如果某文件大小没有到达64MB，该文件也不会占据整个块空间。在分布式的HDFS集群上，Hadoop系统 保证一个块存储在一个datanode上。

HDFS的namenode只存储整个文件系统的元数据镜像，这个镜像由配置dfs.name.dir指定，datanode则存有文件的metainfo和具体的分块，存储路径由dfs.data.dir指定。

2、分片:

hadoop的作业在提交过程中，需要把具体的输入进行分片。具体的分片细节由InputSplitFormat指定。分片的规则为FileInputFormat.class中的getSplits()方法指定:

```
1      long splitSize = computeSplitSize(goalSize, minSize, blockSize);
```

```
    computeSplitSize:
    Math.max(minSize, Math.min(goalSize, blockSize));
```

其中goalSize为“InputFile大小”/“我们在配置文件中定义的mapred.map.tasks”值，minsize为mapred.min.split.size，blockSize为64，所以，这个算式为取分片大小不大于block，并且不小于在mapred.min.split.size配置中定义的最小Size。

当某个分块分成均等的若干分片时，会有最后一个分片大小小于定义的分片大小，则该分片独立成为一个分片。

8.如何减少Hadoop Map端到Reduce端的数据传输量?

减少传输量，可以让map处理完，让同台的reduce直接处理，理想情况下，没有数据传输。

9.Hadoop的Shuffle?

1、hadoop：map端保存分片数据，通过网络收集到reduce端

2、Shuffle产生的意义是什么?

完整地从map task端拉取数据到reduce 端；在跨节点拉取数据时，尽可能地减少对带宽的不必要消耗；减少磁盘IO对task执行的影响；每个map task都有一个内存缓冲区，存储着map的输出结果，当缓冲区快满的时候需要将缓冲区的数据该如何处理?

每个map task都有一个内存缓冲区，存储着map的输出结果，当缓冲区快满的时候需要将缓冲区的数据以一个临时文件的方式存放到磁盘，当整个map task结束后再对磁盘中这个map task产生的所有临时文件做合并，生成最终的正式输出文件，然后等待reduce task来拉数据。

10.哪些场景才能使用Combiner呢?

1. Combiner的输出是Reducer的输入，Combiner绝不能改变最终的计算结果。所以从我的想法来看，Combiner只应该用于那种Reduce的输入key/value与输出key/value类型完全一致，且不影响最终结果的场景。比如累加，最大值等。Combiner的使用一定得慎重，如果用好，它对job执行效率有帮助，反之会影响reduce的最终结果。
2. 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。
3. combiner的目的是减少map网络流量。combiner的对象是对于map。combiner具有和reduce相似的功能。只不过combiner合并对象，是对于一个map。reduce合并对象，是对于多个map。

11.HMaster的作用?

1. 为region server分配region.
2. 负责region server的负载均衡。
3. 发现失效的region server并重新分配其上的region.
4. Gfs上的垃圾文件回收。
5. 处理schema更新请求。

12.如何实现hadoop的安全机制?

(1)共享hadoop集群:

a: 管理人员把开发人员分成了若干个队列，每个队列有一定的资源，每个用户及用户组只能使用某个队列中指定资源。

b: HDFS上有各种数据，公用的，私有的，加密的。不用的用户可以访问不同的数据。

(2) HDFS安全机制

client获取namenode的初始访问认证(使用kerberos)后，会获取一个delegation token，这个token可以作为接下来访问HDFS或提交作业的认证。同样，读取block也是一样的。

(3) mapreduce安全机制

所有关于作业的提交或者作业运行状态的追踪均是采用带有Kerberos认证的RPC实现的。授权用户提交作业时，JobTracker会为之生成一个delegation token，该token将被作为job的一部分存储到HDFS上并通过RPC分发给各个TaskTracker，一旦job运行结束，该token失效。

(4) DistributedCache是安全的。

DistribuedCache分别两种，一种是shared，可以被所有作业共享，而private的只能被该用户的作业共享。

(5) RPC安全机制

在Hadoop RP中添加了权限认证授权机制。当用户调用RPC时，用户的login name会通过RPC头部传递给RPC，之后RPC使用Simple Authentication and Security Layer（SASL）确定一个权限协议（支持Kerberos和DIGEST-MD5两种），完

成RPC授权。

13.hadoop的调度策略的实现，你们使用的是哪种策略，为什么?

(1)默认情况下hadoop使用的FIFO, 先进先出的调度策略。按照作业的优先级来处理。

(2)计算能力调度器(Capacity Scheduler) 支持多个队列，每个队列可配置一定的资源量，每个队列采用FIFO, 为了防止同一个用户的作业独占资源，那么调度器会对同一个用户提交的作业所占资源进行限定，首先按以下策略选择一个合适队列：计算每个队列中正在运行的任务数与其应该分得的计算资源之间的比值，选择一个该比值最小的队列；然后按以下策略选择该队列中一个作业：按照作业优先级和提交时间顺序选择，同时考虑用户资源量限制和内存限制。

(3)公平调度器(Fair Scheduler) 支持多队列多用户，每个队列中的资源量可以配置，同一队 列中的作业公平共享队列中所有资源。

(4)异构集群的调度器LATE

(5)实时作业的调度器Deadline Scheduler和Constraint-based Scheduler

14.数据倾斜怎么处理?

数据倾斜有很多解决方案，本例子简要介绍一种实现方式，假设表A 和表B连接，表A 数据倾斜，只有一个key倾斜，首先对A进行采样，统计出最倾斜的key，将A表分隔为A1 只有倾斜 key， A2 不包含倾斜key， 然后分别与 表B 连接。最后将结果合并， union

15.评述hadoop运行原理?

1、有hdfs 负责数据存放，是Hadoop的分布式文件存储系统

2、将大文件分解为多个Block，每个Block保存多个副本。提供容错机制，副本丢失或者宕机时自动恢复。默认每个Block保存3个副本，64M为1个Block。由mapreduce负责计算，Map（映射）和Reduce（归约）

16.简单说一下hadoop的map-reduce编程模型

(1)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。

(2)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。

(3)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

(4)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。

举例说明：将一副牌的分成四种花色。

17.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

18.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。

19.Hadoop平台集群配置、环境变量设置?

1、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。

2、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。

3、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

4、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

5、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。

20.Hadoop性能调优?

1、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）

2、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
```

21.Hadoop如何实现高并发?

（1）分段加锁机制+内存双缓冲机制

（2）多线程并发吞吐量的百倍优化

（3）缓冲数据批量刷磁盘+网络优化

参考：https://www.codercto.com/a/38917.html