100 道常见 Hadoop 面试题及答案解析-第 1 版

目录

在目前生活中，随着移动互联网科技不断的发展和创新，如今无论是公司还是开发者个人而言，面试都是一项耗时耗钱的项目，从而小讲对于日常 Hadoop 面试中可能会遇到的问题进行了筛选与汇总。这个是第一版，下周带来第二版（更新版），请关注微信“大数据研习社”，每天一份有价值的大数据相关的学习资料。

1 单选题

1.1 下面哪个程序负责 HDFS 数据存储。

a)NameNode
b)Jobtracker
c)Datanode
d)secondaryNameNode
e)tasktracker
答案 C datanode

1.2 HDfS 中的 block 默认保存几份？

a)3 份
b)2 份
c)1 份
d)不确定
答案 A 默认 3 分

1.3 下列哪个程序通常与 NameNode 在一个节点启动？

a)SecondaryNameNode
b)DataNode
c)TaskTracker
d)Jobtracker
答案 D，此题分析：
hadoop 的集群是基于 master/slave 模式，namenode 和 jobtracker 属于 master，datanode 和 tasktracker 属于 slave，master 只有一个，而 slave 有多个
SecondaryNameNode 内存需求和 NameNode 在一个数量级上，所以通常 secondary NameNode（运行在单独的物理机器上）和 NameNode 运行在不同的机器上。
JobTracker 和 TaskTracker
JobTracker 对应于 NameNode
TaskTracker 对应于 DataNode
DataNode 和 NameNode 是针对数据存放来而言的
JobTracker 和 TaskTracker 是对于 MapReduce 执行而言的
mapreduce 中几个主要概念，mapreduce 整体上可以分为这么几条执行线索：obclient，JobTracker 与 TaskTracker。
1、JobClient会在用户端通过JobClient类将应用已经配置参数打包成jar文件存储到hdfs，并把路径提交到 Jobtracker,然后由 JobTracker 创建每一个 Task（即 MapTask 和

ReduceTask）并将它们分发到各个 TaskTracker 服务中去执行。
2、JobTracker 是一个 master 服务，软件启动之后 JobTracker 接收 Job，负责调度 Job 的每一个子任务 task 运行于 TaskTracker 上，并监控它们，如果发现有失败的 task 就重新运行它。一般情况应该把 JobTracker 部署在单独的机器上。
3、TaskTracker 是运行在多个节点上的 slaver 服务。TaskTracker 主动与 JobTracker 通信，接收作业，并负责直接执行每一个任务。TaskTracker 都需要运行在 HDFS 的 DataNode 上。

1.4 Hadoop 作者

a)Martin Fowler
b)Kent Beck
c)Doug cutting
答案 C Doug cutting

1.5 HDFS 默认 Block Size

a)32MB
b)64MB
c)128MB
答案：B
(因为版本更换较快，这里答案只供参考)

1.6 下列哪项通常是集群的最主要瓶颈：

a)CPU
b)网络
c)磁盘 IO
d)内存
答案：C 磁盘
该题解析：
首先集群的目的是为了节省成本，用廉价的 pc 机，取代小型机及大型机。小型机和大型机有什么特点？
1.cpu 处理能力强
2.内存够大
所以集群的瓶颈不可能是 a 和 d
3.网络是一种稀缺资源，但是并不是瓶颈。
4.由于大数据面临海量数据，读写数据都需要 io，然后还要冗余数据，hadoop 一般备 3 份数据，所以 IO 就会打折扣。

1.7 关于 SecondaryNameNode 哪项是正确的？

a)它是 NameNode 的热备
b)它对内存没有要求

c)它的目的是帮助 NameNode 合并编辑日志，减少 NameNode 启动时间

d)SecondaryNameNode 应与 NameNode 部署到一个节点。

答案 C

2 多选题

2.1 下列哪项可以作为集群的管理？

a)Puppet

b)Pdsh

c)Cloudera Manager

d)Zookeeper

答案：ABD

2.2 配置机架感知的下面哪项正确：

a)如果一个机架出问题，不会影响数据读写

b)写入数据的时候会写到不同机架的 DataNode 中

c)MapReduce 会根据机架获取离自己比较近的网络数据

答案 ABC

2.3 Client 端上传文件的时候下列哪项正确？

a)数据经过 NameNode 传递给 DataNode

b)Client 端将文件切分为 Block，依次上传

c)Client 只上传数据到一台 DataNode，然后由 NameNode 负责 Block 复制工作

答案 B，该题分析：

Client 向 NameNode 发起文件写入的请求。

NameNode 根据文件大小和文件块配置情况，返回给 Client 它所管理部分 DataNode 的信息。

Client 将文件划分为多个 Block，根据 DataNode 的地址信息，按顺序写入到每一个 DataNode 块中。

2.4 下列哪个是 Hadoop 运行的模式：

a)单机版

b)伪分布式

c)分布式

答案 ABC

2.5 Cloudera 提供哪几种安装 CDH 的方法？

a)Cloudera manager

b)Tarball

c)Yum

d)Rpm

答案：ABCD

3 判断题

3.1 Ganglia 不仅可以进行监控，也可以进行告警。(正确)

分析：此题的目的是考 Ganglia 的了解。严格意义上来讲是正确。ganglia 作为一款最常用的 Linux 环境中的监控软件，它擅长的的是从节点中按照用户的需求以较低的代价采集数据。但是 ganglia 在预警以及发生事件后通知用户上并不擅长。最新的 ganglia 已经有了部分这方面的功能。但是更擅长做警告的还有 Nagios。Nagios，就是一款精于预警、通知的软件。通过将 Ganglia 和 Nagios 组合起来，把 Ganglia 采集的数据作为 Nagios 的数据源，然后利用 Nagios 来发送预警通知，可以完美的实现一整套监控管理的系统。

3.2 Block Size 是不可以修改的。(错误)

分析：它是可以被修改的 Hadoop 的基础配置文件是 hadoop-default.xml，默认建立一个 Job 的时候会建立 Job 的 Config，Config 首先读入 hadoop-default.xml 的配置，然后再读入 hadoop-site.xml 的配置 (这个文件初始的时候配置为空)，hadoop-site.xml 中主要配置需要覆盖的 hadoop-default.xml 的系统级配置。

3.3 Nagios 不可以监控 Hadoop 集群，因为它不提供 Hadoop 支持。(错误)

分析：Nagios 是集群监控工具，而且是云计算三大利器之一

3.4 如果 NameNode 意外终止，SecondaryNameNode 会接替它使集群继续工作。(错误)

分析：SecondaryNameNode 是帮助恢复，而不是替代，如何恢复，可以查看

3.5 Cloudera CDH 是需要付费使用的。(错误)

分析：第一套付费产品是 Cloudera Enterpris，Cloudera Enterprise 在美国加州举行的 Hadoop 大会 (Hadoop Summit) 上公开，以若干私有管理、监控、运作工具加强 Hadoop 的功能。收费采取合约订购方式，价格随用的 Hadoop 丛集大小变动。

3.6 Hadoop 是 Java 开发的，所以 MapReduce 只支持 Java 语言编写。(错误)

分析：rhadoop 是用 R 语言开发的，MapReduce 是一个框架，可以理解是一种思想，可以使用其他语言开发。

3.7 Hadoop 支持数据的随机读写。(错)

分析：lucene 是支持随机读写的，而 hdfs 只支持随机读。但是 HBase 可以来补救。HBase 提供随机读写，来解决 Hadoop 不能处理的问题。HBase 自底层设计开始即聚焦于各种可伸缩性问题：表可以很“高”，有数十亿个数据行；也可以很“宽”，有数百万个列；水平分区并在上千个普通商用机节点上自动复制。表的模式是物理存储的直接反映，使系统有可能提高高效的数据结构的序列化、存储和检索。

3.8 NameNode 负责管理 metadata，client 端每次读写请求，它都会从磁盘中读取或则会写入 metadata 信息并反馈 client 端。(错误)

此题分析：

NameNode 不需要从磁盘读取 metadata，所有数据都在内存中，硬盘上的只是序列化的结果，只有每次 namenode 启动的时候才会读取。

1）文件写入

Client 向 NameNode 发起文件写入的请求。

NameNode 根据文件大小和文件块配置情况，返回给 Client 它所管理部分 DataNode 的信息。

Client 将文件划分为多个 Block，根据 DataNode 的地址信息，按顺序写入到每一个 DataNode 块中。

2）文件读取

Client 向 NameNode 发起文件读取的请求。

3.9 NameNode 本地磁盘保存了 Block 的位置信息。（ 个人认为正确，欢迎提出其它意见）

分析：DataNode 是文件存储的基本单元，它将 Block 存储在本地文件系统中，保存了 Block 的 Meta-data，同时周期性地将所有存在的 Block 信息发送给 NameNode。NameNode 返回文件存储的 DataNode 的信息。

Client 读取文件信息。

3.10 DataNode 通过长连接与 NameNode 保持通信。（有分歧 ）

这个有分歧：具体正在找这方面的有利资料。下面提供资料可参考。

首先明确一下概念：

（1）.长连接

Client 方与 Server 方先建立通讯连接，连接建立后不断开，然后再进行报文发送和接收。这种方式下由于通讯连接一直存在，此种方式常用于点对点通讯。

（2）.短连接

Client 方与 Server 每进行一次报文收发交易时才进行通讯连接，交易完毕后立即断开连接。此种方式常用于一点对多点通讯，比如多个 Client 连接一个 Server.

3.11 Hadoop 自身具有严格的权限管理和安全措施保障集群正常运行。（错误 ）

hadoop 只能阻止好人犯错，但是不能阻止坏人干坏事

3.12 Slave 节点要存储数据，所以它的磁盘越大越好。（ 错误）

分析：一旦 Slave 节点宕机，数据恢复是一个难题

3.13 hadoop dfsadmin –report 命令用于检测 HDFS 损坏块。（错误 ）

3.14 Hadoop 默认调度器策略为 FIFO（正确 ）

3.15 集群内每个节点都应该配 RAID，这样避免单磁盘损坏，影响整个节点运行。（错误 ）

分析：首先明白什么是 RAID，可以参考百科磁盘阵列。这句话错误的地方在于太绝对，具体情况具体分析。题目不是重点，知识才是最重要的。因为 hadoop 本身就具有冗余能力，所以如果不是很严格不需要都配备 RAID。具体参考第二题。

3.16 因为 HDFS 有多个副本，所以 NameNode 是不存在单点问题的。（错误 ）

3.17 每个 map 槽就是一个线程。（错误 ）

分析：首先我们知道什么是 map 槽，map 槽->map slotmap slot 只是一个逻辑值（org.apache.hadoop.mapred.TaskTracker.TaskLauncher.numFreeSlots），而不是对应着一个线程或者进程

3.18 Mapreduce 的 input split 就是一个 block。(错误)

3.19 NameNode 的 Web UI 端口是 50030，它通过 jetty 启动的 Web 服务。(错误)

3.20 Hadoop 环境变量中的 HADOOP_HEAPSIZE 用于设置所有 Hadoop 守护线程的内存。它默认是 200 GB。(错误)

hadoop 为各个守护进程
（namenode,secondarynamenode,jobtracker,datanode,tasktracker）统一分配的内存在 hadoop-env.sh 中设置，参数为 HADOOP_HEAPSIZE，默认为 1000M。

3.21 DataNode 首次加入 cluster 的时候，如果 log 中报告不兼容文件版本，那需要 NameNode 执行"Hadoop namenode -format"操作格式化磁盘。(错误)

分析：
首先明白介绍，什么 ClusterID
ClusterID
添加了一个新的标识符 ClusterID 用于标识集群中所有的节点。当格式化一个 Namenode，需要提供这个标识符或者自动生成。这个 ID 可以被用来格式化加入集群的其他 Namenode。
二次整理
有的同学问题的重点不是上面分析内容：内容如下：
这个报错是说明 DataNode 所装的 Hadoop 版本和其它节点不一致，应该检查 DataNode 的 Hadoop 版本

4 问答题（一）

4.1 Hadoop 集群可以运行的 3 个模式？

单机（本地）模式
伪分布式模式
全分布式模式

4.2 单机（本地）模式中的注意点？

在单机模式（standalone）中不会存在守护进程，所有东西都运行在一个 JVM 上。这里同样没有 DFS，使用的是本地文件系统。单机模式适用于开发过程中运行 MapReduce 程序，这也是最少使用的一个模式。

4.3 伪分布模式中的注意点？

伪分布式（Pseudo）适用于开发和测试环境，在这个模式中，所有守护进程都在同一台机器上运行。

4.4 VM 是否可以称为 Pseudo？

不是，两个事物，同时 Pseudo 只针对 Hadoop。

4.5 全分布模式又有什么注意点？

全分布模式通常被用于生产环境，这里我们使用 N 台主机组成一个 Hadoop 集群，Hadoop 守护进程运行在每台主机之上。这里会存在 Namenode 运行的主机，Datanode 运行的主机，以及 tasktracker 运行的主机。在分布式环境下，主节点和从节点会分开。

4.6 Hadoop 是否遵循 UNIX 模式？

是的，在 UNIX 用例下，Hadoop 还拥有“conf”目录。

4.7 Hadoop 安装在什么目录下？

Cloudera 和 Apache 使用相同的目录结构，Hadoop 被安装在 cd/usr/lib/hadoop-0.20/。

4.8 Namenode、Jobtracker 和 tasktracker 的端口号是？

Namenode，70；Jobtracker，30；Tasktracker，60。

4.9 Hadoop 的核心配置是什么？

Hadoop 的核心配置通过两个 xml 文件来完成：1，hadoop-default.xml；2，hadoop-site.xml。这些文件都使用 xml 格式，因此每个 xml 中都有一些属性，包括名称和值，但是当下这些文件都已不复存在。

4.10 那当下又该如何配置？

Hadoop 现在拥有 3 个配置文件：1，core-site.xml；2，hdfs-site.xml；3，mapred-site.xml。这些文件都保存在 conf/子目录下。

4.11 RAM 的溢出因子是？

溢出因子(Spillfactor)是临时文件中储存文件的大小，也就是 Hadoop-temp 目录。

4.12 fs.mapr.working.dir 只是单一的目录？

fs.mapr.working.dir 只是一个目录。

4.13 hdfs-site.xml 的 3 个主要属性？

dfs.name.dir 决定的是元数据存储的路径以及 DFS 的存储方式（磁盘或是远端）

dfs.data.dir 决定的是数据存储的路径

fs.checkpoint.dir 用于第二 Namenode

4.14 如何退出输入模式？

退出输入的方式有：1，按 ESC；2，键入:q（如果你没有输入任何当下）或者键入:wq（如果你已经输入当下），并且按下 Enter。

4.15 当你输入 hadoopfsck/造成“connectionrefusedjavaexception’”时，系统究竟发生了什么？

这意味着 Namenode 没有运行在你的 VM 之上。

4.16 我们使用 Ubuntu 及 Cloudera，那么我们该去哪里下载 Hadoop，或者是默认就与 Ubuntu 一起安装？

这个属于 Hadoop 的默认配置 ,你必须从 Cloudera 或者 Edureka 的 dropbox 下载，然后在你的系统上运行。当然，你也可以自己配置，但是你需要一个 Linuxbox，Ubuntu 或者是 RedHat。在 Cloudera 网站或者是 Edureka 的 Dropbox 中有安装步骤。

4.17 “jps”命令的用处？

这个命令可以检查 Namenode、Datanode、TaskTracker、JobTracker 是否正常工作。

4.18 如何重启 Namenode？

点击 stop-all.sh，再点击 start-all.sh。

键入 sudohdfs（Enter），su-hdfs（Enter），/etc/init.d/ha（Enter），及/etc/init.d/hadoop-0.20-namenodestart（Enter）。

4.19 Fsck 的全名？

全名是：FileSystemCheck。

4.20 如何检查 Namenode 是否正常运行？

如果要检查 Namenode 是否正常工作，使用命令/etc/init.d/hadoop-0.20-namenodestatus 或者就是简单的 jps。

4.21 mapred.job.tracker 命令的作用？

可以让你知道哪个节点是 JobTracker。

4.22 /etc/init.d 命令的作用是？

/etc/init.d 说明了守护进程（服务）的位置或状态，其实是 LINUX 特性，和 Hadoop 关系不大。

4.23 如何在浏览器中查找 Namenode？

如果你确实需要在浏览器中查找 Namenode，你不再需要 localhost:8021，Namenode 的端口号是 50070。

4.24 如何从 SU 转到 Cloudera？

从 SU 转到 Cloudera 只需要键入 exit。

4.25 启动和关闭命令会用到哪些文件？

Slaves 及 Masters。

4.26 Slaves 由什么组成？

Slaves 由主机的列表组成，每台 1 行，用于说明数据节点。

4.27 Masters 由什么组成？

Masters 同样是主机的列表组成，每台一行，用于说明第二 Namenode 服务器。

4.28 hadoop-env.sh 是用于做什么的？

hadoop-env.sh 提供了 Hadoop 中.JAVA_HOME 的运行环境。

4.29 Master 文件是否提供了多个入口？

是的你可以拥有多个 Master 文件接口。

4.30 hadoop-env.sh 文件当下的位置？

hadoop-env.sh 现在位于 conf。

4.31 在 Hadoop_PID_DIR 中，PID 代表了什么？

PID 代表了“ProcessID”。

4.32 /var/hadoop/pids 用于做什么？

/var/hadoop/pids 用来存储 PID。

4.33 hadoop-metrics.properties 文件的作用是？

hadoop-metrics.properties 被用做“Reporting”，控制 Hadoop 报告，初始状态是“nottoreport”。

4.34 Hadoop 需求什么样的网络？

Hadoop 核心使用 Shell（SSH）来驱动从节点上的服务器进程，并在主节点和从节点之间使用 password-lessSSH 连接。

4.35 全分布式环境下为什么需求 password-lessSSH？

这主要因为集群中通信过于频繁，JobTracker 需要尽可能快的给 TaskTracker 发布任务。

4.36 这会导致安全问题吗？

完全不用担心。Hadoop 集群是完全隔离的，通常情况下无法从互联网进行操作。与众不同的配置，因此我们完全不需要在意这种级别的安全漏洞，比如说通过互联网侵入等等。Hadoop 为机器之间的连接提供了一个相对安全的方式。

4.37 SSH 工作的端口号是？

SSH 工作的端口号是 NO.22，当然可以通过它来配置，22 是默认的端口号。

4.38 SSH 中的注意点还包括？

SSH 只是个安全的 shell 通信，可以把它当做 NO.22 上的一种协议，只需要配置一个密码就可以安全的访问。

4.39 为什么 SSH 本地主机需要密码？

在 SSH 中使用密码主要是增加安全性，在某些情况下也根本不会设置密码通信。

4.40 如果在 SSH 中添加 key，是否还需要设置密码？

是的，即使在 SSH 中添加了 key，还是需要设置密码。

4.41 假如 Namenode 中没有数据会怎么样？

没有数据的 Namenode 就不能称之为 Namenode，通常情况下，Namenode 肯定会有数据。

4.42 当 JobTracker 宕掉时，Namenode 会发生什么？

当 JobTracker 失败时，集群仍然可以正常工作，只要 Namenode 没问题。

4.43 是客户端还是 Namenode 决定输入的分片？

这并不是客户端决定的，在配置文件中以及决定分片细则。

4.44 是否可以自行搭建 Hadoop 集群？

是的，只要对 Hadoop 环境足够熟悉，你完全可以这么做。

4.45 是否可以在 Windows 上运行 Hadoop？

你最好不要这么做，RedHatLinux 或者是 Ubuntu 才是 Hadoop 的最佳操作系统。在 Hadoop 安装中，Windows 通常不会被使用，因为会出现各种各样的问题。因此，Windows 绝对不是 Hadoop 的推荐系统。

5 问答题（二）

5.1 写出以下执行命令

1）如何杀死一个 job

先 Hadoop job -list 得到 jobid　　杀死 job： hadoop job -kill jobid

2）删除 hdfs 上的/tmp/xxx 目录 hadoop fs -rm -r /tmp/xxx

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

加入新节点时：

Hadoop-daemon.sh start datanode

Hadoop-daemon.sh start tasktracker

删除节点时

Hadoop maradmin -refreshnodes

Hadoop dfsadmin -refreshnodes

5.2 简述一下 hdfs 的数据压缩算法，工作中用的是那种算法，为什么？

1、在 HDFS 之上将数据压缩好后，再存储到 HDFS

2、在 HDFS 内部支持数据压缩，这里又可以分为几种方法：

2.1、压缩工作在 DataNode 上完成，这里又分两种方法：

2.1.1、数据接收完后，再压缩

这个方法对 HDFS 的改动最小，但效果最低，只需要在 block 文件 close 后，调用压缩工具，将 block 文件压缩一下，然后再打开 block 文件时解压一下即可，几行代码就可以搞定

2.1.2、边接收数据边压缩，使用第三方提供的压缩库

效率和复杂度折中方法，Hook 住系统的 write 和 read 操作，在数据写入磁盘之前，先压缩一下，但 write 和 read 对外的接口行为不变，比如：原始大小为 100KB 的数据，压缩后大小为 10KB，当写入 100KB 后，仍对调用者返回 100KB，而不是 10KB

2.2、压缩工作交给 DFSClient 做，DataNode 只接收和存储

这个方法效果最高，压缩分散地推给了 HDFS 客户端，但 DataNode 需要知道什么时候一个 block 块接收完成了。

推荐最终实现采用 2.2 这个方法，该方法需要修改的 HDFS 代码量也不大，但效果最高。

1、 Datanode 在什么情况下不会备份？

单节点的情况下不会备份！

2、 datanode 首次加入 cluster 的时候，如果 log 报告不兼容文件版本，那需要 namenode

3、 执行格式化操作，这样处理的原因是？这个说法是错误的！

添加了一个新的标识符 ClusterID 用于标识集群中所有的节点。当格式化一个 Namenode，需要提供这个标识符或者自动生成。这个 ID 可以被用来格式化加入集群的其他 Namenode 应该检查 hadoop 的版本是不是与其他的 hadoop 版本一致！

5.3 三个 datanode，当有一个 datanode 出现错误会怎样？

Datanode 以数据块作为容错单位 通常一个数据块会备份到三个 datanode 上，如果一个 datanode 出错，则回去其他备份数据块的 datanode 上读取，并且会把这个 datanode 上的数据块再复制一份 以达到备份的效果！

5.4 hdfs 原理，以及各个模块的职责？

Client：切分文件；访问或通过命令行管理 HDFS；与 NameNode 交互，获取文件位置信息；与 DataNode 交互，读取和写入数据。

NameNode：Master 节点，只有一个，管理 HDFS 的名称空间和数据块映射信息；配置副本策略；处理客户端请求。
DataNode :Slave 节点 ,存储实际的数据 ;执行数据块的读写 ;汇报存储信息给 NameNode。
Secondary NameNode 辅助 NameNode ,分担其工作量 ;定期合并 fsimage 和 fsedits，推送给 NameNode；紧急情况下，可辅助恢复 NameNode，但 Secondary NameNode 并非 NameNode 的热备
Hdfs 文件读取

1.首先调用 FileSystem 对象的 open 方法，其实是一个 DistributedFileSystem 的实例

2.DistributedFileSystem 通过 rpc 获得文件的第一批个 block 的 locations，同一 block 按照重复数会返回多个 locations，这些 locations 按照 hadoop 拓扑结构排序，距离客户端近的排在前面.

3.前两步会返回一个 FSDataInputStream 对象，该对象会被封装成 DFSInputStream 对象 ,DFSInputStream 可以方便的管理 datanode 和 namenode 数据流。客户端调用 read 方法，DFSInputStream 最会找出离客户端最近的 datanode 并连接。

4.数据从 datanode 源源不断的流向客户端。

5.如果第一块的数据读完了 ,就会关闭指向第一块的 datanode 连接 ,接着读取下一块。这些操作对客户端来说是透明的，客户端的角度看来只是读一个持续不断的流。

6.如果第一批 block 都读完了 ,DFSInputStream 就会去 namenode 拿下一批 blocks 的 location，然后继续读，如果所有的块都读完，这时就会关闭掉所有的流
Hdfs 的文件写入
1.客户端通过调用 DistributedFileSystem 的 create 方法创建新文件
2.DistributedFileSystem 通过 RPC 调用 namenode 去创建一个没有 blocks 关联的新文件，创建前，namenode 会做各种校验，比如文件是否存在，客户端有无权限去创建等。如果校验通过，namenode 就会记录下新文件，否则就会抛出 IO 异常.
3.前两步结束后会返回 FSDataOutputStream 的对象，和读文件的时候相似，FSDataOutputStream 被封装成 DFSOutputStream，DFSOutputStream 可以协调 namenode 和 datanode。客户端开始写数据到 DFSOutputStream,DFSOutputStream 会把数据切成一个个小 packet，然后排成队列 data quene。
4.DataStreamer 会去处理接受 data quene ,他先问询 namenode 这个新的 block 最适合存储的在哪几个 datanode 里，比如重复数是 3，那么就找到 3 个最适合的 datanode，把他们排成一个 pipeline.DataStreamer 把 packet 按队列输出到管道的第一个 datanode 中，第一个 datanode 又把 packet 输出到第二个 datanode 中，以此类推。
5.DFSOutputStream 还有一个对列叫 ack quene，也是有 packet 组成，等待 datanode 的收到响应，当 pipeline 中的所有 datanode 都表示已经收到的时候，这时 akc quene 才会把对应的 packet 包移除掉。
6.客户端完成写数据后调用 close 方法关闭写入流
7.DataStreamer 把剩余得包都刷到 pipeline 里然后等待 ack 信息 ,收到最后一个 ack 后，

通知 datanode 把文件标示为已完成。

5.5 哪个进程通常与 namenode 在一个节点启动？并做分析

JobTracker

hadoop 的集群是基于 master/slave 模式，namenode 和 jobtracker 属于 master，datanode 和 tasktracker 属于 slave，master 只有一个，而 slave 有多个

SecondaryNameNode 内存需求和 NameNode 在一个数量级上，所以通常 secondary NameNode（运行在单独的物理机器上）和 NameNode 运行在不同的机器上。

JobTracker 和 TaskTracker

JobTracker 对应于 NameNode

TaskTracker 对应于 DataNode

DataNode 和 NameNode 是针对数据存放来而言的

JobTracker 和 TaskTracker 是对于 MapReduce 执行而言的

mapreduce 中几个主要概念，mapreduce 整体上可以分为这么几条执行线索：

jobclient，JobTracker 与 TaskTracker。

1、JobClient会在用户端通过JobClient类将应用已经配置参数打包成jar文件存储到hdfs，并把路径提交到 Jobtracker,然后由 JobTracker 创建每一个 Task（即 MapTask 和 ReduceTask）

并将它们分发到各个 TaskTracker 服务中去执行

2、JobTracker 是一个 master 服务，软件启动之后 JobTracker 接收 Job，负责调度 Job 的每一个子任务 task 运行于 TaskTracker 上，

并监控它们，如果发现有失败的 task 就重新运行它。一般情况应该把 JobTracker 部署在单独的机器上。

3、TaskTracker 是运行在多个节点上的 slaver 服务。TaskTracker 主动与 JobTracker 通信，接收作业，并负责直接执行每一个任务。

TaskTracker 都需要运行在 HDFS 的 DataNode 上。

5.6 hdfs 的体系结构？

HDFS 采用了主从（Master/Slave）结构模型，一个 HDFS 集群是由一个 NameNode 和若干个 DataNode 组成的。其中 NameNode 作为主服务器，管理文件系统的命名空间和客户端对文件的访问操作；集群中的 DataNode 管理存储的数据。HDFS 允许用户以文件的形式存储数据。从内部来看，文件被分成若干个数据块，而且这若干个数据块存放在一组 DataNode 上。NameNode 执行文件系统的命名空间操作，比如打开、关闭、重命名文件或目录等，它也负责数据块到具体 DataNode 的映射。DataNode 负责处理文件系统客户端的文件读写请求，并在 NameNode 的统一调度下进行数据块的创建、删除和复制工作。NameNode 和 DataNode 都被设计成可以在普通商用计算机上运行。这些计算机通常运行的是 GNU/Linux 操作系统。HDFS 采用 Java 语言开发，因此任何支持 Java 的机器

都可以部署 NameNode 和 DataNode。一个典型的部署场景是集群中的一台机器运行一个 NameNode 实例，其他机器分别运行一个 DataNode 实例。当然，并不排除一台机器运行多个 DataNode 实例的情况。集群中单一的 NameNode 的设计则大大简化了系统的架构。NameNode 是所有 HDFS 元数据的管理者，用户数据永远不会经过 NameNode。

5.7 HDFS，replica 如何定位

```
//查找某个文件在 HDFS 集群的位置
public static void getFileLocal() throws IOException{

    //返回 FileSystem 对象
    FileSystem fs = getFileSystem();

    //文件路径
    Path path =
new Path("hdfs://single.hadoop.dajiangtai.com:9000/middle/weibo/weibo.txt");

    //获取文件目录
    FileStatus filestatus = fs.getFileStatus(path);
    //获取文件块位置列表
    BlockLocation[] blkLocations =
fs.getFileBlockLocations(filestatus, 0, filestatus.getLen());
    //循环输出块信息
    for(int i=0;i< blkLocations.length;i++){
        String[] hosts = blkLocations[i].getHosts();
        System.out.println("block_"+i+"_location:"+hosts[0]);
    }
```

5.8 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

5.9 hdfs 的 client 端，复制到第三个副本时宕机， hdfs 怎么恢复保证下次写第三副本?

Datanode 会定时上报 block 块的信息给 namenode ，namenode 就会得知副本缺失，然后 namenode 就会启动副本复制流程以保证数据块的备份！

5.10 block 块信息是先写 dataNode 还是先写 nameNode?

Client 向 NameNode 发起文件写入的请求。

NameNode 根据文件大小和文件块配置情况，返回给 Client 它所管理部分 DataNode 的信息。

Client 将文件划分为多个 Block，根据 DataNode 的地址信息，按顺序写入到每一个 DataNode 块中。

5.11 Hive 的 join 有几种方式，怎么实现 join 的？

答：3 种 join 方式：

1）在 reduce 端进行 join，最常用的 join 方式。

Map 端的主要工作：为来自不同表(文件)的 key/value 对打标签以区别不同来源的记录。然后用连接字段作为 key，其余部分和新加的标志作为 value，最后进行输出。

reduce 端的主要工作：在 reduce 端以连接字段作为 key 的分组已经完成，我们只需要在每一个分组当中将那些来源于不同文件的记录(在 map 阶段已经打标志)分开，最后进行笛卡尔。

2）在 map 端进行 join，使用场景：一张表十分小、一张表很大：

在提交作业的时候先将小表文件放到该作业的 DistributedCache 中，然后从 DistributeCache 中取出该小表进行 join key / value 解释分割放到内存中(可以放大 Hash Map 等等容器中)。然后扫描大表，看大表中的每条记录的 join key /value 值是否能够在内存中找到相同 join key 的记录，如果有则直接输出结果

3）SemiJoin，semijoin 就是左边连接是 reducejoin 的一种变种，在 map 端过滤掉一些数据，在网络传输过程中，只传输参与连接的数据，减少了 shuffle 的网络传输量，其他和 reduce 的思想是一样的。

实现：将小表中参与 join 的 key 单独抽取出来通过 DistributeCache 分发到相关节点，在 map 阶段扫描连接表，将 join key 不在内存 hashset 的纪录过滤掉，让参与 join 的纪录通过 shuffle 传输到 reduce 端进行 join，其他和 reduce join 一样。

5.12 hive 内部表和外部表的区别？

内部表：建表时会在 hdfs 创建一个表的存储目录，增加分区的时候，会将数据复制到此 location 下，删除数据的时候，将表的数据和元数据一起删除。

外部表：一般会建立分区，增加分区的时候不会将数据移到此表的 location 下，删除数据的时候，只删除了表的元数据信息，表的数据不会删除。

5.13 hive 是如何实现分区的？

建表语句：

```
create table tablename (id) partitioned by (dt string)
```

增加分区：

```
alter table tablenname  add partition (dt = '2016-03-06')
```

删除分区：

```
alter table tablename  drop partition (dt = '2016-03-06')
```

5.14 hive 支持 not in 吗？

不支持，可以用 left join 实现此功能

5.15 Hive 有哪些方式保存元数据，各有哪些优缺点。

1)存储于 derby 数据库，此方法只能开启一个 hive 客户端，不推荐使用

2)存储于 mysql 数据库中，可以多客户端连接，推荐使用。

5.16 hive 如何优化

1）join 优化，尽量将小表放在 join 的左边，如果一个表很小可以采用 mapjoin

2)排序优化，order by 一个 reduce 效率低，distirbute by +sort by 也可以实现全局排序

3）使用分区，查询时可减少数据的检索，从而节省时间。

5.17 hive 能像关系数据库那样，建多个库吗？

可以建立多个库，多库多表都支持。

5.18 hive 中的压缩格式 RCFile、 TextFile、 SequenceFile 各有什么区别？

TextFile：默认格式，数据不做压缩，磁盘开销大，数据解析开销大

SequenceFile：Hadoop API 提供的一种二进制文件支持，使用方便，可分割，可压缩，支持三种压缩，NONE，RECORD，BLOCK

RCFILE 是一种行列存储相结合的方式。首先，将数据按行分块，保证同一个 record 在同一个块上，避免读一个记录读取多个 block。其次，块数据列式存储，有利于数据压缩和快

速的列存取。数据加载的时候性能消耗大，但具有较好的压缩比和查询响应。

5.19 hive 相对于 Oracle 来说有那些优点？

1）存储，hive 存储在 hdfs 上，oracle 存储在本地文件系统

2）扩展性，hive 可以扩展到数千节点，oracle 理论上只可扩展到 100 台左右

3）单表存储，数据量大 hive 可以分区分桶，oracle 数据量大只能分表。

5.20 Hive 的 sort by 和 order by 的区别

order by 会对输入数据做全局排序，只有一个 reduce，数据量较大时，很慢。

sort by 不是全局排序，只能保证每个 reduce 有序，不能保证全局有序，需设置 mapred.reduce.tasks>1