# Hadoop 学习

## HDFS 学习

### 1. ：概念

**数据块**

数据块 ：每一个磁盘都有自己默认的磁盘大小，这是磁盘进行读写的最小单位；

HDFS也有数据块的概念，默认64m， hadoop2中是128m。对分布式系统块进行抽象会带来很对的好处。

最明显的好处就是一个文件可以大于任意的一个磁盘的容量。文件的所有块并不是存储在一个磁盘，而是存储在任一磁盘中。

使用抽象块 ，而非整个文件系统大大简化了 存储系统的设计，

DATA\_NODE, NAME\_NODE

对于任一一个而言进行对应的

HDFS集群是以管理者和工作者模式运行。其中DATA\_NODE 就是工作结点。--NAME\_NODE 就是对应的管理结点。

Name\_node 是系统管理者的命名空间。

他维护者这个文件系统树以及这个文件系统数对应的所有文件和目录。这些信息一一两个文件形式。以一两个文件形式 永远保存在本地磁盘。

对应的也记录者一个文件的所有块的信息，但不是永久保存。

通过 datanode和namenode交互来访问整个文件系统。

Datanode是文件系统的工作结点。根据需要检索数据块，并且定期向namenode 发送数据块的列表。

对namenode实现容错非常重要，

对namenode实现容错非常重要，Hadoop为此要两种机制：   
\* 第一种机制是备份那些组成文件系统元数据持久状态的文件。Hadoop通过配置使namenode在多个文件系统上实时同步的保存元数据的持久状态（原子操作）。一般配置是将持久状态写入一个远程挂载的网络文件系统(NFS)。   
\* 另一种可行的方法就是运行一个辅助的namenode，但他不能作为namenode使用。辅助namenode的作用是定期通过编辑日志合并命名空间镜像，以防止编辑日志过大。辅助namenode的保存状态滞后主namenode，当主namenode失效时，把存储在NFS上的namenode元数据复制到辅助namenode上，并作为主namenode。

）HDFS高可用性，namenode是唯一存储元数据与文件到数据块映射的的地方。   
如何恢复一个失效的namenode？   
\* 系统管理员要启动一个拥有文件系统元数据副本的新的namenode，并配置datanode和客户端以便使用这个新的namenode。   
\* 新的namenode响应服务的条件：   
一是将命名空间的映像导入内存中；   
二是重做编辑日志；三是接受足够多的来自datanode的数据块报告并退出安全模式。   
\* 故障切换与规避：一个称为故障转移控制器的系统中有一个新实体管理这将活动namenode转移为备用namenode的转换过程。每个namenode运行一个轻量级的故障转移控制器，其工作就是监视宿主namenode是否失效并在namenode失效时进行故障切换，可以组织两个namenode进行有序切

上图中展现了整个HDFS三个重要角色：NameNode、DataNode和Client。NameNode可以看作是分布式文件系统中的管理者，主要负责管理文件系统的命名空间、集群配置信息和存储块的复制等。NameNode会将文件系统的Meta-data存储在内存中，这些信息主要包括了文件信息、每一个文件对应的文件块的信息和每一个文件块在DataNode的信息等。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。Client就是需要获取分布式文件系统文件的应用程序。

NameNode

* 1. Namenode 上保存着 HDFS 的名字空间。对于任何对文件系统元数据产生修改的操作， Namenode 都会使用一种称为 EditLog 的事务日志记录下来。例如，在 HDFS 中创建一个文件， Namenode 就会在 Editlog 中插入一条记录来表示；同样地，修改文件的副本系数也将往 Editlog 插入一条记录。 Namenode 在本地操作系统的文件系统中存储这个 Editlog 。整个文件系统的名 字空间，包括数据块到文件的映射、文件的属性等，都存储在一个称为 FsImage 的文件中，这 个文件也是放在 Namenode 所在的本地文件系统上。
  2. Namenode 在内存中保存着整个文件系统的名字空间和文件数据块映射 (Blockmap) 的映像 。这个关键的元数据结构设计得很紧凑，因而一个有 4G 内存的 Namenode 足够支撑大量的文件 和目录。当 Namenode 启动时，它从硬盘中读取 Editlog 和 FsImage ，将所有 Editlog 中的事务作 用在内存中的 FsImage 上，并将这个新版本的 FsImage 从内存中保存到本地磁盘上，然后删除 旧的 Editlog ，因为这个旧的 Editlog 的事务都已经作用在 FsImage 上了。这个过程称为一个检查 点 (checkpoint) 。在当前实现中，检查点只发生在 Namenode 启动时，在不久的将来将实现支持 周期性的检查点。

HDFS NameSpace

* 1. HDFS 支持传统的层次型文件组织结构。用户或者应用程序可以创建目 录，然后将文件保存在这些目录里。文件系统名字空间的层次结构和大多数 现有的文件系统类似：用户可以创建、删除、移动或重命名文件。当前， HDFS 不支持用户磁盘配额和访问权限控制，也不支持硬链接和软链接。但 是 HDFS 架构并不妨碍实现这些特性。
  2. Namenode 负责维护文件系统命名空间，任何对文件系统名字空间或属 性的修改都将被 Namenode 记录下来。应用程序可以设置 HDFS 保存的文件 的副本数目。文件副本的数目称为文件的副本系数，这个信息也是由 Namenode 保存的。

DataNode

* 1. Datanode 将 HDFS 数据以文件的形式存储在本地的文件系统中，它并不知道有 关 HDFS 文件的信息。它把每个 HDFS 数据块存储在本地文件系统的一个单独的文件 中。 Datanode 并不在同一个目录创建所有的文件，实际上，它用试探的方法来确定 每个目录的最佳文件数目，并且在适当的时候创建子目录。在同一个目录中创建所 有的本地文件并不是最优的选择，这是因为本地文件系统可能无法高效地在单个目 录中支持大量的文件。
  2. 当一个 Datanode 启动时，它会扫描本地文件系统，产生一个这些本地文件对应 的所有 HDFS 数据块的列表，然后作为报告发送到 Namenode ，这个报告就是块状态 报告。

Secondary NameNode

* 1. Secondary NameNode 定期合并 fsimage 和 edits 日志，将 edits 日志文件大小控制在一个限度下

### 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。

### 文件读取：

* Client向NameNode发起文件读取的请求。
* NameNode返回文件存储的DataNode的信息。
* Client读取文件信息。

### 文件Block复制：

* NameNode发现部分文件的Block不符合最小复制数或者部分DataNode失效。
* 通知DataNode相互复制Block。
* DataNode开始直接相互复制。

### HDFS的几个设计特点：

* Block的放置：默认不配置。一个Block会有三份备份，一份放在NameNode指定的DataNode，另一份放在与指定 DataNode非同一Rack上的DataNode，最后一份放在与指定DataNode同一Rack上的DataNode上。备份无非就是为了数据安全，考虑同一Rack的失败情况以及不同Rack之间数据拷贝性能问题就采用这种配置方式。
* 心跳检测DataNode的健康状况，如果发现问题就采取数据备份的方式来保证数据的安全性。
* 数据复制（场景为DataNode失败、需要平衡DataNode的存储利用率和需要平衡DataNode数据交互压力等情况）：这里先说一下，使用HDFS的balancer命令，可以配置一个Threshold来平衡每一个DataNode磁盘利用率。例如设置了Threshold为 10%，那么执行balancer命令的时候，首先统计所有DataNode的磁盘利用率的均值，然后判断如果某一个DataNode的磁盘利用率超过这个均值Threshold以上，那么将会把这个DataNode的block转移到磁盘利用率低的DataNode，这对于新节点的加入来说十分有用。
* 数据交验：采用CRC32作数据交验。在文件Block写入的时候除了写入数据还会写入交验信息，在读取的时候需要交验后再读入。
* NameNode是单点：如果失败的话，任务处理信息将会纪录在本地文件系统和远端的文件系统中。
* 数据管道性的写入：当客户端要写入文件到DataNode上，首先客户端读取一个Block然后写到第一个DataNode上，然后由第一个DataNode传递到备份的DataNode上，一直到所有需要写入这个Block的NataNode都成功写入，客户端才会继续开始写下一个 Block。
* 安全模式：在分布式文件系统启动的时候，开始的时候会有安全模式，当分布式文件系统处于安全模式的情况下，文件系统中的内容不允许修改也不允许删除，直到安全模式结束。安全模式主要是为了系统启动的时候检查各个DataNode上数据块的有效性，同时根据策略必要的复制或者删除部分数据块。运行期通过命令也可以进入安全模式。在实践过程中，系统启动的时候去修改和删除文件也会有安全模式不允许修改的出错提示，只需要等待一会儿即可。