第4章 分布式文件系统HDFS

高琰

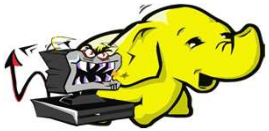
目录

01

分布式文件系统

Hadoop和HDFS的关系

Hadoop实现了一个分布式文件系统（Hadoop Distributed File System），简称HDFS。



对外部客户机而言，HDFS 就像一个传统的分级文件系统。可以创建、删除、移动或重命名文件，等等。很多时候，我们就叫它DFS（Distributed File System）。

为什么需要分布式文件系统

•**海量信息**:有证券交易所的每天1TB的信息；社交网站上2PB的图片；计算机每天产生的100GB到10TB的机器日志，缓存文件，RFID检测器等等。

1990年，一块普通磁盘存储1370MB数据，传输速度4.4MB/S，读取整个磁盘需要5min。

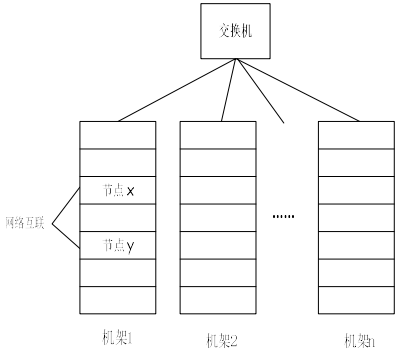
2011年，普及的1TB一块磁盘，其传输速度是100MB/S，读取整个磁盘需要100min左右。写的速度大概是读的3倍

当数据集的大小超过一台独立物理计算机的存储能力时，就有必要对它进行分区并存储到若干台单独的计算机上。

计算机集群结构

•分布式文件系统把文件分布存储到多个计算机节点上，成千上万的计算机节点构成计算机集群

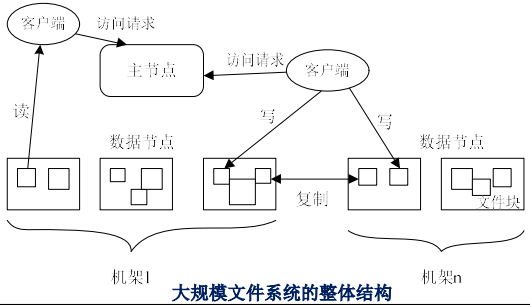
•与之前使用多个处理器和专用高级硬件的并行化处理装置不同的是，目前的分布式文件系统所采用的计算机集群，都是由普通硬件构成的，这就大大降低了硬件上的开销



计算机集群的基本架构

分布式文件系统的结构

分布式文件系统在物理结构上是由计算机集群中的多个节点构成的，这些节点分为两类，一类叫“主节点”(Master Node)或者也被称为“名称结点”(NameNode)，另一类叫“从节点”（Slave Node）或者也被称为“数据节点”(DataNode)



大规模文件系统的整体结构

02

HDFS简介

HDFS的优点

- **最高效的访问模式是 一次写入、多次读取(流式数据访问)**
  - HDFS存储的数据集作为Hadoop的分析对象。在数据集生成后，长时间在此数据集上进行各种分析。每次分析都将涉及该数据集的大部分数据甚至全部数据，因此读取整个数据集的时间延迟比读取第一条记录的时间延迟更重要。
  - 它被设计适合于批量处理的情形，而不是在于与用户的交互性。因此，它更注重于高的传输率，而不是低延迟性。也因此，而放宽了POSIX的语法要求。

HDFS的优点

- **运行在普通廉价的服务器上**

  HDFS设计理念之一就是让它能运行在普通的硬件之上，即便硬件出现故障，也可以通过容错策略来保证数据的高可用。因此，故障检测与快速自动故障恢复是HDFS设计的核心目标；

- **大数据集**

  运行在HDFS上的应用具有很大的数据集。HDFS上的一个典型文件　大小一般都在G字节至T字节。一个单一的HDFS实例应该能支撑数以千万计的文件

- **简单的文件模型**

  DFS应用需要一个“一次写入多次读取”的文件访问模型。一个文件经过创建、写入和关闭之后就不需要改变。这一假设简化了数据一致性问题，并且使高吞吐量的数据访问成为可能。Map/Reduce应用或者网络爬虫应用都非常适合这个模型。目前还有计划在将来扩充这个模型，使之支持文件的附加写操作。

- **强大的跨平台兼容性**

HDFS不适用的场景

- 1）低延迟的场景

  HDFS具有很高的数据吞吐量的性能，代价是可能有较高的时间延迟。而当要求只有几十毫秒的延迟时，HBase表现会更好些。

- 2）大量的小文件

  hdfs适用较大的数据块。当有大量小文件时，如过亿的小文件，namenode的管理能力就不够了。

- 3）多用户写入，任意修改文件

  当有多个用户进行修改文件时，或者任意修改文件时，容易造成冲突，而比较低效。

03

HDFS相关概念

HDFS的概念

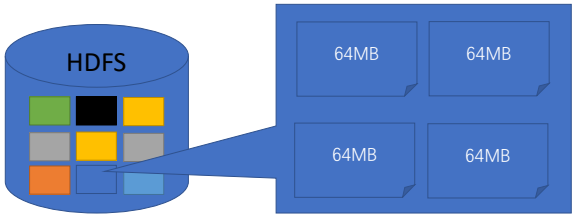
磁盘存储文件时，是按照数据块来存储的，也就是说，数据块是磁盘的读/写最小单位。数据块也称磁盘块。

构建于单个磁盘上的文件系统是通过磁盘块来管理文件系统，一般来说，文件系统块的大小是磁盘块的整数倍。

特别的，单个磁盘文件系统，小于磁盘块的文件会占用整个磁盘块。磁盘块的大小一般是512字节。

块（Block）

在HDFS中，也有块（block）这个概念，默认为64MB，每个块作为独立的存储单元。一个文件被分成多个块，以块作为存储单位。块的大小远远大于普通文件系统，可以最小化寻址开销



块的大小可以通过配置参数(dfs.blocksize)来规定，在现有的hadoop2.x版本中默认大小是128M。

块（Block）

当然不是设置每个块越大越好。

HDFS提供给MapReduce数据服务，而一般来说MapReduce的Map任务通常一次处理一个块中的数据，如果任务数太少（少于集群中节点的数量），就没有发挥多节点的优势，甚至作业的运行速度就会和单节点一样。

HDFS对与用户来说，可以直接看成是一个巨大的硬盘。所以，HDFS和文件系统相似，用fsck指令可以显示块信息。

```
% hadoop fsck / -files -blocks
```

块（Block）

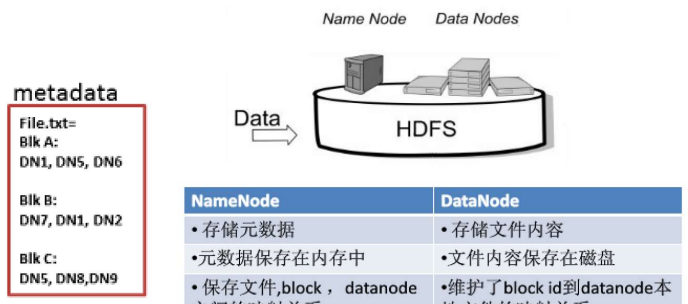
- **支持大规模文件存储**：文件以块为单位进行存储，一个大规模文件可以被分拆成若干个文件块，不同的文件块可以被分发到不同的节点上，因此，一个文件的大小不会受到单个节点的存储容量的限制，可以远远大于网络中任意节点的存储容量
- **简化系统设计**：首先，大大简化了存储管理，因为文件块大小是固定的，这样就可以很容易计算出一个节点可以存储多少文件块；其次，方便了元数据的管理，元数据不需要和文件块一起存储，可以由其他系统负责管理元数据
- **适合数据备份**：每个文件块都可以冗余存储到多个节点上，大大提高了系统的容错性和可用性

NameNode Metadata

- **Meta-data 存在内存中**
  - 整个Meta-data放入主内存
  - No demand paging of meta-data
- **Meta-data记录了**
  - 文件列表信息
  - 每个文件的块列表
  - 每个块对应的DataNode
  - 文件属性，如创建时间、创建者、几份副本等
- **Transaction Log (EditLog)**
  - 记录了文件系统的每个变化，如创建文件、删除文件、修改文件的副本数等
  - EditLog会被合并为FsImage并存入磁盘
- **Meta-data 磁盘故障**
  - NameNode可以维护多份数据

名称节点和数据节点

HDFS主要组件的功能



NameNode	DataNode
• 存储元数据	• 存储文件内容
•元数据保存在内存中	•文件内容保存在磁盘
• 保存文件,block， datanode之间的映射关系	•维护了block id到datanode本地文件的映射关系

名称节点和数据节点

名称节点的数据结构

•在HDFS中，名称节点（NameNode）负责管理分布式文件系统的命名空间（Namespace），保存了两个核心的数据结构，即FsImage和EditLog
  •FsImage用于维护文件系统树以及文件树中所有的文件和文件夹的元数据
  •操作日志文件EditLog中记录了所有针对文件的创建、删除、重命名等操作
•名称节点记录了每个文件中各个块所在的数据节点的位置信息

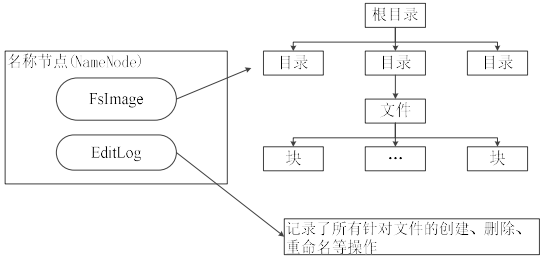


图4-3 名称节点的数据结构

名称节点和数据节点

FsImage文件

•FsImage文件包含文件系统中所有目录和文件inode的序列化形式。每个inode是一个文件或目录的元数据的内部表示，并包含此类信息：文件的复制等级、修改和访问时间、访问权限、块大小以及组成文件的块。对于目录，则存储修改时间、权限和配额元数据

•FsImage文件没有记录文件包含哪些块以及每个块存储在哪个数据节点。而是由名称节点把这些映射信息保留在内存中，当数据节点加入HDFS集群时，数据节点会把自己所包含的块列表告知给名称节点，此后会定期执行这种告知操作，以确保名称节点的块映射是最新的。

名称节点和数据节点

名称节点的启动

•在名称节点启动的时候，它会将FsImage文件中的内容加载到内存中，之后再执行EditLog文件中的各项操作，使得内存中的元数据和实际的同步，存在内存中的元数据支持客户端的读操作。

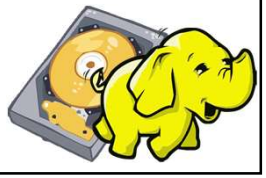
•一旦在内存中成功建立文件系统元数据的映射，则创建一个新的FsImage文件和一个空的EditLog文件

•名称节点起来之后，HDFS中的更新操作会重新写到EditLog文件中，

为什么？

名称节点和数据节点

因为FsImage文件一般都很大（GB级别的很常见），如果所有的更新操作都往FsImage文件中添加，这样会导致系统运行的十分缓慢，但是，如果往EditLog文件里面写就不会这样，因为EditLog 要小很多。每次执行写操作之后，且在向客户端发送成功代码之前，edits文件需要同步更新



名称节点和数据节点

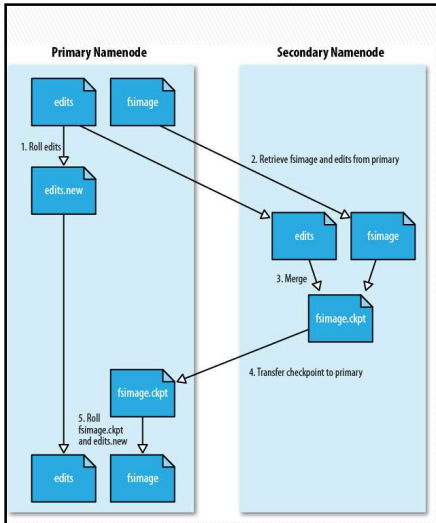
名称节点运行期间EditLog不断变大的问题

•在名称节点运行期间，HDFS的所有更新操作都是直接写到EditLog中，久而久之， EditLog文件将会变得很大

•虽然这对名称节点运行时候是没有什么明显影响的，但是，当名称节点重启的时候，名称节点需要先将FsImage里面的所有内容映像到内存中，然后再一条一条地执行EditLog中的记录，当EditLog文件非常大的时候，会导致名称节点启动操作非常慢，而在这段时间内HDFS系统处于安全模式，一直无法对外提供写操作，影响了用户的使用

如何解决？答案是：SecondaryNameNode第二名称节点

第二名称节点是HDFS架构中的一个组成部分，它是用来保存名称节点中对HDFS 元数据信息的备份，并减少名称节点重启的时间。SecondaryNameNode一般是单独运行在一台机器上



SecondaryNameNode的工作情况：
（1）SecondaryNameNode会定期和NameNode通信，请求其停止使用EditLog文件，暂时将新的写操作写到一个新的文件edit.new上来，这个操作是瞬间完成，上层写日志的函数完全感觉不到差别；
（2）SecondaryNameNode通过HTTP GET方式从NameNode上获取到FsImage和EditLog文件，并下载到本地的相应目录下；
（3）SecondaryNameNode将下载下来的FsImage载入到内存，然后一条一条地执行EditLog文件中的各项更新操作，使得内存中的FsImage保持最新；这个过程就是EditLog和FsImage文件合并；
（4）SecondaryNameNode执行完（3）操作之后，会通过post方式将新的FsImage文件发送到NameNode节点上
（5）NameNode将从Secondary-NameNode接收到的新的FsImage替换旧的FsImage文件，同时将edit.new替换EditLog文件，通过这个过程EditLog就变小了

名称节点和数据节点

数据节点（DataNode）

•数据节点是分布式文件系统HDFS的工作节点，负责数据的存储和读取，会根据客户端或者是名称节点的调度来进行数据的存储和检索，并且向名称节点定期发送自己所存储的块的列表

•每个数据节点中的数据会被保存在各自节点的本地Linux文件系统中

04

HDFS体系结构

HDFS体系结构概述

HDFS采用了主从（Master/Slave）结构模型，一个HDFS集群包括一个名称节点（NameNode）和若干个数据节点（DataNode）。名称节点作为中心服务器，负责管理文件系统的命名空间及客户端对文件的访问。集群中的数据节点一般是一个节点运行一个数据节点进程，负责处理文件系统客户端的读/写请求，在名称节点的统一调度下进行数据块的创建、删除和复制等操作。

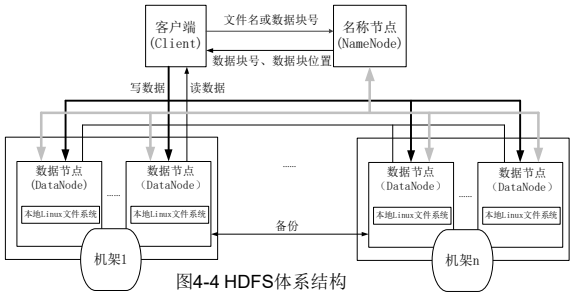
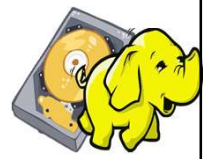


图4-4 HDFS体系结构

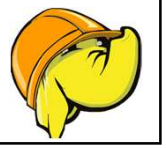
HDFS命名空间管理

- HDFS的命名空间包含目录、文件和块
- 在HDFS1.0体系结构中，在整个HDFS集群中只有一个命名空间，并且只有唯一一个名称节点，该节点负责对这个命名空间进行管理
- 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
- HDFS使用的是传统的分级文件体系，因此，用户可以像使用普通文件系统一样，创建、删除目录和文件，在目录间转移文件，重命名文件等
- 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。



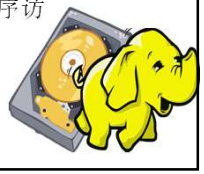
通信协议

- HDFS是一个部署在集群上的分布式文件系统，因此，很多数据需要通过网络进行传输
- 所有的HDFS通信协议都是构建在TCP/IP协议基础之上的
- 客户端通过一个可配置的端口向名称节点主动发起TCP连接，并使用客户端协议与名称节点进行交互
- 名称节点和数据节点之间则使用数据节点协议进行交互
- 客户端与数据节点的交互是通过RPC（Remote Procedure Call）来实现的。在设计上，名称节点不会主动发起RPC，而是响应来自客户端和数据节点的RPC请求



客户端

- 客户端是用户操作HDFS最常用的方式，HDFS在部署时都提供了客户端
- HDFS客户端是一个库，暴露了HDFS文件系统接口，这些接口隐藏了HDFS实现中的大部分复杂性
- 严格来说，客户端并不算是HDFS的一部分
- 客户端可以支持打开、读取、写入等常见的操作，并且提供了类似Shell的命令行方式来访问HDFS中的数据
- 此外，HDFS也提供了Java API，作为应用程序访问文件系统的客户端编程接口



HDFS体系结构的局限性

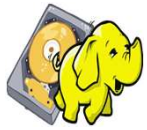
HDFS只设置唯一一个名称节点，这样做虽然大大简化了系统设计，但也带来了一些明显的局限性，具体如下：

（1）**命名空间的限制**：名称节点是保存在内存中的，因此，名称节点能够容纳的对象（文件、块）的个数会受到内存空间大小的限制。

（2）**性能的瓶颈**：整个分布式文件系统的吞吐量，受限于单个名称节点的吞吐量。

（3）**隔离问题**：由于集群中只有一个名称节点，只有一个命名空间，因此，无法对不同应用程序进行隔离。

（4）**集群的可用性**：一旦这个唯一的名称节点发生故障，会导致整个集群变得不可用。



05

HDFS存储原理

冗余数据保存-数据复制

作为一个分布式文件系统，为了保证系统的容错性和可用性，HDFS采用了多副本方式对数据进行冗余存储，通常一个数据块的多个副本会被分布到不同的数据节点上。这种多副本方式具有以下几个优点：

（1）**加快数据传输速度**

（2）**容易检查数据错误**

（3）**保证数据可靠性**

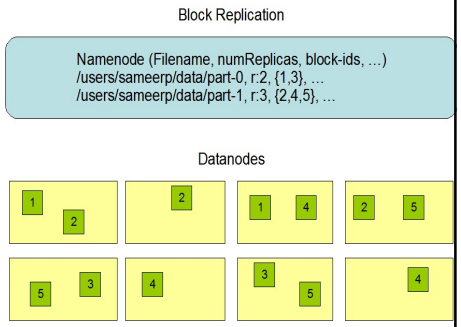


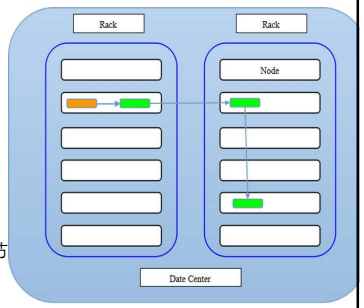
图4-5 HDFS数据块多副本存储

复制策略

- 副本的存放是HDFS可靠性和性能的关键。优化的副本存放策略是HDFS区分于其他大部分分布式文件系统的重要特性。HDFS采用一种称为**机架感知(rack-aware)的策略**来改进数据的可靠性、可用性和网络带宽的利用率。

- **机架感知：**
  NameNode能感知机架id，选择较优方式

- **备份策略：有3份或以上：**
  - 一份放在本地节点上
  - 第二份放在另外一个机架的节点上
  - 第三份放在跟第二份同机架的不同节
  - 其他的随机放置



数据复制-安全模式

- Namenode启动后会进入一个称为安全模式的特殊状态。处于安全模式的Namenode是不会进行数据块的复制的。Namenode从所有的 Datanode接收心跳信号和块状态报告。
- 块状态报告包括了某个Datanode所有的数据块列表。每个数据块都有一个指定的最小副本数。
- 当Namenode检测确认某个数据块的副本数目达到这个最小值，那么该数据块就会被认为是副本安全(safely replicated)的；在一定百分比（这个参数可配置）的数据块被Namenode检测确认是安全之后（加上一个额外的30秒等待时间），Namenode将退出安全模式状态。接下来它会确定还有哪些数据块的副本没有达到指定数目，并将这些数据块复制到其他Datanode上。

数据存取策略

数据读取

- HDFS提供了一个API可以确定一个数据节点所属的机架ID，客户端也可以调用API获取自己所属的机架ID

- 当客户端读取数据时，从名称节点获得数据块不同副本的存放位置列表，列表中包含了副本所在的数据节点，可以调用API来确定客户端和这些数据节点所属的机架ID，当发现某个数据块副本对应的机架ID和客户端对应的机架ID相同时，就优先选择该副本读取数据。如果没有发现，就随机选择一个副本读取数据。如果一个HDFS集群跨越多个数据中心，那么客户端也将首先读本地数据中心的副本。

数据写入

Staging

- 客户端创建文件的请求其实并没有立即发送给Namenode，事实上，在刚开始阶段HDFS客户端会先将文件数据缓存到本地的一个临时文件。应用程序的写操作被透明地重定向到这个临时文件。
- 当这个临时文件累积的数据量超过一个数据块的大小，客户端才会联系Namenode。Namenode将文件名插入文件系统的层次结构中，并且分配一个数据块给它。然后返回Datanode的标识符和目标数据块给客户端。
- 接着客户端将这块数据从本地临时文件上传到指定的Datanode上。
- 当文件关闭时，在临时文件中剩余的没有上传的数据也会传输到指定的Datanode上。然后客户端告诉Namenode文件已经关闭。此时Namenode才将文件创建操作提交到日志里进行存储。如果Namenode在文件关闭前宕机了，则该文件将丢失。

数据写入

流水线复制

- 当客户端向HDFS文件写入数据的时候，一开始是写到本地临时文件中。假设该文件的副本系数设置为3，当本地临时文件累积到一个数据块的大小时，客户端会从Namenode获取一个Datanode列表用于存放副本。
- 然后客户端开始向第一个Datanode传输数据，第一个Datanode一小部分一小部分(4 KB)地接收数据，将每一部分写入本地仓库，并同时传输该部分到列表中第二个Datanode节点。第二个Datanode也是这样，一小部分一小部分地接收数据，写入本地仓库，并同时传给第三个Datanode。
- 最后，第三个Datanode接收数据并存储在本地。因此，Datanode能流水线式地从前一个节点接收数据，并在同时转发给下一个节点，数据以流水线的方式从前一个Datanode复制到下一个。

数据错误与恢复

HDFS具有较高的容错性，可以兼容廉价的硬件，它把硬件出错看作一种常态，而不是异常，并设计了相应的机制检测数据错误和进行自动恢复，主要包括以下几种情形：**名称节点出错、数据节点出错和数据出错。**

1. 名称节点出错

名称节点保存了所有的元数据信息，其中，最核心的两大数据结构是FsImage和Editlog，如果这两个文件发生损坏，那么整个HDFS实例将失效。因此，HDFS设置了备份机制，把这些核心文件同步复制到备份服务器SecondaryNameNode上。当名称节点出错时，就可以根据备份服务器SecondaryNameNode中的FsImage和Editlog数据进行恢复。

数据错误与恢复

2. 数据节点出错

•每个数据节点会定期向名称节点发送“心跳”信息，向名称节点报告自己的状态

•当数据节点发生故障，或者网络发生断网时，名称节点就无法收到来自一些数据节点的心跳信息，这时，这些数据节点就会被标记为“宕机”，节点上面的所有数据都会被标记为“不可读”，名称节点不会再给它们发送任何I/O请求

•这时，有可能出现一种情形，即由于一些数据节点的不可用，会导致一些数据块的副本数量小于冗余因子

•名称节点会定期检查这种情况，一旦发现某个数据块的副本数量小于冗余因子，就会启动数据冗余复制，为它生成新的副本

•HDFS和其它分布式文件系统的最大区别就是可以调整冗余数据的位置

数据错误与恢复

3. 数据出错

- **原因：**网络传输和磁盘错误
- **措施：**
  - 客户端在读取到数据后，会采用md5和sha1对数据块进行校验，以确定读取到正确的数据
  - 在文件被创建时，客户端就会对每一个文件块进行信息摘录，并把这些信息写入到同一个路径的隐藏文件里面
  - 当客户端读取文件的时候，会先读取该信息文件，然后，利用该信息文件对每个读取的数据块进行校验，如果校验出错，客户端就会请求到另外一个数据节点读取该文件块，并且向名称节点报告这个文件块有错误，名称节点会定期检查并且重新复制这个块

06

HDFS使用

HDFS使用

Hadoop提供了关于HDFS在Linux操作系统上进行文件操作的常用Shell命令以及Java API。同时还可以利用Web界面查看和管理Hadoop文件系统

备注：Hadoop安装成功后，已经包含HDFS和MapReduce，不需要额外安装。而HBase等其他组件，则需要另外下载安装。

在学习HDFS编程实践前，我们需要启动Hadoop。执行如下命令：

```
$ cd /usr/local/hadoop
$ ./bin/hdfs namenode -format    #格式化hadoop的hdfs文件系统
$ ./sbin/start-dfs.sh #启动hadoop
```

HDFS常用命令

HDFS有很多shell命令，其中，fs命令可以说是HDFS最常用的命令
利用该命令可以查看HDFS文件系统的目录结构、上传和下载数据、创建文件等。该命令的用法为：

hadoop fs [genericOptions] [commandOptions]

备注：Hadoop中有三种Shell命令方式：
•hadoop fs适用于任何不同的文件系统，比如本地文件系统和HDFS文件系统
hadoop dfs只能适用于HDFS文件系统
hdfs dfs跟hadoop dfs的命令作用一样，也只能适用于HDFS文件系统

HDFS常用命令

实例：

显示<path>指定的文件的详细信息：％hadoop fs –ls <path>

在HDFS内创建创建<path>指定的文件夹：％hadoop fs –mkdir <path>

在HDFS内删除<path>指定的文件夹：％hadoop fs –rmr <path>

```
administrator@ubuntu:~/hadoop/hadoop-1.2.1/bin$ ./hadoop fs -mkdir hdfs://127.0.0.1:9000/tempDir

administrator@ubuntu:~/hadoop/hadoop-1.2.1/bin$ ./hadoop fs -ls hdfs://127.0.0.1:9000/
Found 4 items
drwxr-xr-x   - administrator supergroup          0 2015-04-26 16:30 /hbase
drwxr-xr-x   - administrator supergroup          0 2015-04-26 15:44 /home
drwxr-xr-x   - administrator supergroup          0 2015-04-26 16:46 /tempDir
drwxr-xr-x   - administrator supergroup          0 2015-04-26 15:55 /user
```

HDFS常用命令

实例：

hadoop fs -cat <path>:将<path>指定的文件的内容输出到标准输出（stdout）

hadoop fs -copyFromLocal <localsrc> <dst>:将本地源文件<localsrc>复制到路径<dst>指定的文件或文件夹中

```
administrator@ubuntu:~/hadoop/hadoop-1.2.1/bin$ ./hadoop fs -copyFromLocal /home/administrator/t
empfile/* hdfs://127.0.0.1:9000/tempDir
administrator@ubuntu:~/hadoop/hadoop-1.2.1/bin$ ./hadoop fs -ls  hdfs://127.0.0.1:9000/tempDir/
Found 8 items
-rw-r--r--   1 administrator supergroup         18 2015-04-26 16:48 /tempDir/file1.txt
-rw-r--r--   1 administrator supergroup         14 2015-04-26 16:48 /tempDir/file1.txt~
-rw-r--r--   1 administrator supergroup         18 2015-04-26 16:48 /tempDir/file2.txt
-rw-r--r--   1 administrator supergroup         18 2015-04-26 16:48 /tempDir/file3.txt
-rw-r--r--   1 administrator supergroup         18 2015-04-26 16:48 /tempDir/file4.abc
-rw-r--r--   1 administrator supergroup         18 2015-04-26 16:48 /tempDir/file5.abc
-rw-r--r--   1 administrator supergroup         17 2015-04-26 16:48 /tempDir/testFile
-rw-r--r--   1 administrator supergroup          0 2015-04-26 16:48 /tempDir/testFile~
administrator@ubuntu:~/hadoop/hadoop-1.2.1/bin$ ./hadoop fs -cat  hdfs://127.0.0.1:9000/tempDir/
*
this is file1.txt
this is file1
this is file2.txt
this is file3.txt
this is file4.abc
this is file5.abc
welcome to DBLab
```

HDFS的Web界面

在配置好Hadoop集群之后，可以通过浏览器登录

“http://[*NameNodeIP*]:50070”访问HDFS文件系统

HDFS的java访问接口——FileSystem

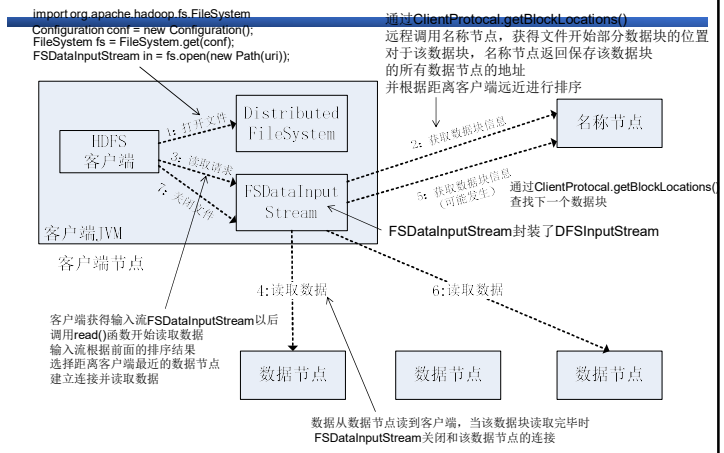
◆使用hadoop提供的javaAPI操作HDFS文件，最重要的是获取及使用FileSystem对象

◆FileSystem可以实现的操作包括：

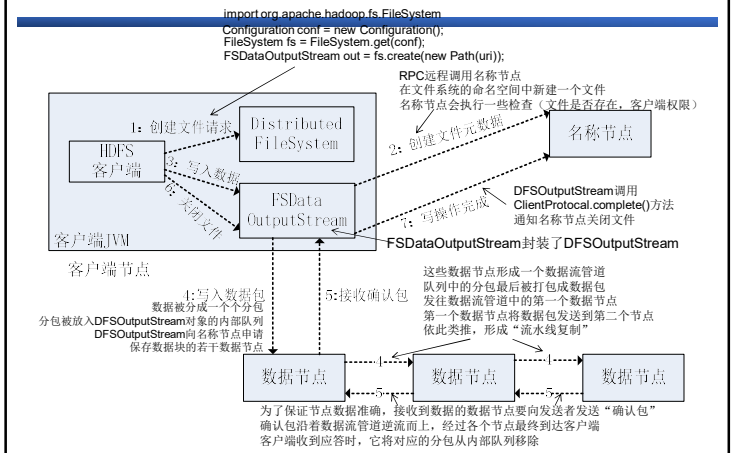
- 写文件 create
- 读取文件 open
- 删除文件delete
- 创建目录 mkdirs
- 删除文件或目录 delete
- 列出目录的内容 listStatus
- 显示文件系统的目录和文件的元数据信息 getFileStatus

用户代码操作HDFS进行文件及目录的操作，是直接调用FileSystem的方法完成的。

读数据的过程



写数据的过程



思考题

- hdfs的组成部分有哪些，分别解释一下
- hdfs的高可靠如何实现
- hdfs的常用shell命令有哪些
- hdfs的常用java api有哪些
- 请用shell命令实现目录、文件的增删改查

51