**简介**

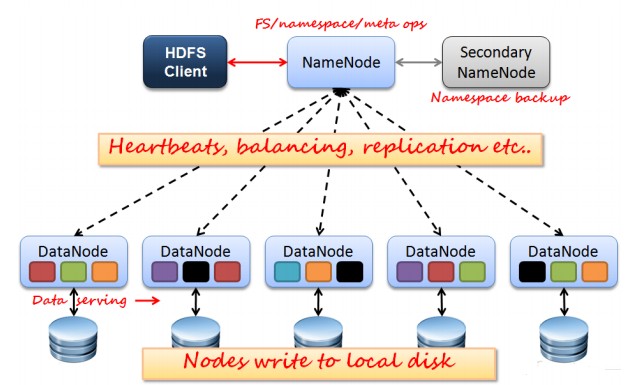
**HDFS**（Hadoop Distributed File System ）Hadoop分布式文件系统。是根据google发表的论文翻版的。论文为GFS（Google File System）Google 文件系统。

**HDFS有很多特点**：

**①**保存多个副本，且提供容错机制，副本丢失或宕机自动恢复。默认存3份。

**②**运行在廉价的机器上。

**③**适合大数据的处理。多大？多小？HDFS默认会将文件分割成block，64M为1个block。然后将block按键值对存储在HDFS上，并将键值对的映射存到内存中(namenode)。如果小文件太多，那内存的负担会很重。



如上图所示，HDFS也是按照Master和Slave的结构。分NameNode、SecondaryNameNode、DataNode这几个角色。

***NameNode***：是Master节点，是管理者。管理数据块映射；处理客户端的读写请求；配置副本策略；管理HDFS的名称空间；

NameNode保存的metadata包括

文件ownership和permission

文件包含的block信息

Block保存在那些DataNode节点上（这部分数据并非保存在NameNode磁盘上的，它是在DataNode启动时上报给NameNode的，Name接收到之后将这些信息保存在内存中）

NameNode的metadata信息在NameNode启动后加载到内存中

Metadata存储到磁盘上的文件名称为fsimage

Block的位置信息不会保存在fsimage中

Edits文件记录了客户端操作fsimage的日志，对文件的增删改等。

用户对fsimage的操作不会直接更新到fsimage中去，而是记录在edits中

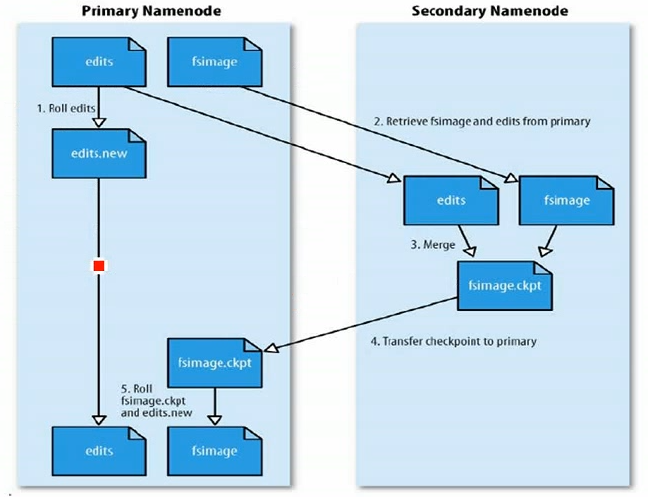
***SecondaryNameNode***：分担namenode的工作量；是NameNode的冷备份；合并fsimage和fsedits然后再发给namenode。

合并fsimage和fsedits文件，然后发送并替换NameNode的fsimage文件，同时自己留下一个副本。这个副本可供NameNode毁灭之后的部分文件恢复。

1可以通过配置fs.checkpoint.period修改合并间隔时间，默认1小时

2也可以通过配置edits日志文件的大小，fs.checkpoint.size规定edits文件的最大值，来让SecondaryNameNode来知道什么时候该进行合并操作了。默认是64M

合并过程如下：



***DataNode****：Slave*节点，奴隶，干活的。负责存储client发来的数据块block；执行数据块的读写操作。

***热备份***：b是a的热备份，如果a坏掉。那么b马上运行代替a的工作。

***冷备份***：b是a的冷备份，如果a坏掉。那么b不能马上代替a工作。但是b上存储a的一些信息，减少a坏掉之后的损失。

***fsimage***:元数据镜像文件（文件系统的目录树。）

***edits***：元数据的操作日志（针对文件系统做的修改操作记录）

**namenode内存中存储的是=fsimage+edits。**

SecondaryNameNode负责定时默认1小时，从namenode上，获取fsimage和edits来进行合并，然后再发送给namenode。减少namenode的工作量。

**HDFS优点**

高容错性

数据自动保存多个副本

副本丢失后，自动回复

适合批处理

移动的计算和操作

数据位置暴漏给计算框架

适合大数据处理

GB、TB、PB甚至更大

百万规模以上的文件数量

10K+节点

可构建在廉价的机器上

通过副本提高可靠性

提供了容错和恢复机制

**HDFS缺点**

低延迟数据访问

毫秒级读取

低延迟与高吞吐量

小文件存取

占用NameNode内存空间

寻址时间超过读取时间

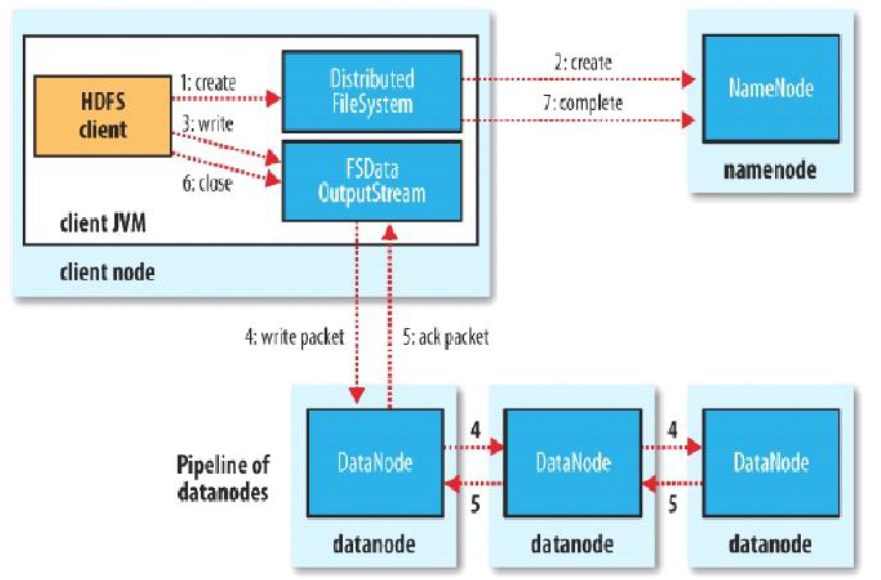
并发写入、文件随即修改

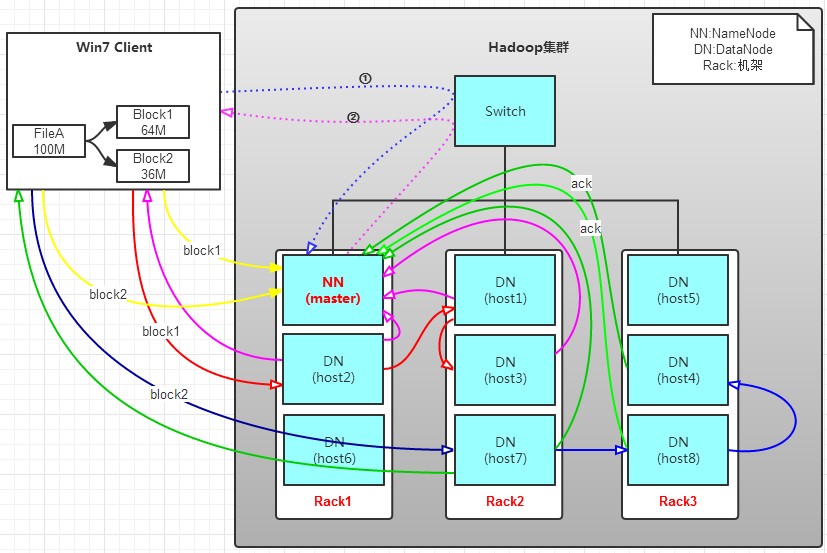
一个文件同时只能由一个写入者

仅支持append

**工作原理**

**写操作：**





有一个文件FileA，100M大小。Client将FileA写入到HDFS上。

HDFS按默认配置。

HDFS分布在三个机架上Rack1，Rack2，Rack3。

**a.** Client将FileA按64M分块。分成两块，block1和Block2;

**b.** Client向nameNode发送写数据请求，如图蓝色虚线①------>。

**c.** NameNode节点，记录block信息。并返回可用的DataNode，如粉色虚线②--------->。

    Block1: host2,host1,host3

    Block2: host7,host8,host4

    原理：

        NameNode具有RackAware机架感知功能，这个可以配置。

        若client为DataNode节点，那存储block时，规则为：副本1，同client的节点上；副本2，不同机架节点上；副本3，同第二个副本机架的另一个节点上；其他副本随机挑选。

        若client不为DataNode节点，那存储block时，规则为：副本1，随机选择一个节点上；副本2，不同副本1，机架上；副本3，同副本2相同的另一个节点上；其他副本随机挑选。

**d.** client向DataNode发送block1；发送过程是以流式写入。

    流式写入过程，

**1>**将64M的block1按64k的package划分;

**2>**然后将第一个package发送给host2;

**3>**host2接收完后，将第一个package发送给host1，同时client想host2发送第二个package；

**4>**host1接收完第一个package后，发送给host3，同时接收host2发来的第二个package。

**5>**以此类推，如图红线实线所示，直到将block1发送完毕。

**6>**host2,host1,host3向NameNode，host2向Client发送通知，说“消息发送完了”。如图粉红颜色实线所示。

**7>**client收到host2发来的消息后，向namenode发送消息，说我写完了。这样就真完成了。如图黄色粗实线

**8>**发送完block1后，再向host7，host8，host4发送block2，如图蓝色实线所示。

**9>**发送完block2后，host7,host8,host4向NameNode，host7向Client发送通知，如图浅绿色实线所示。

**10>**client向NameNode发送消息，说我写完了，如图黄色粗实线。。。这样就完毕了。

**分析，**通过写过程，我们可以了解到：

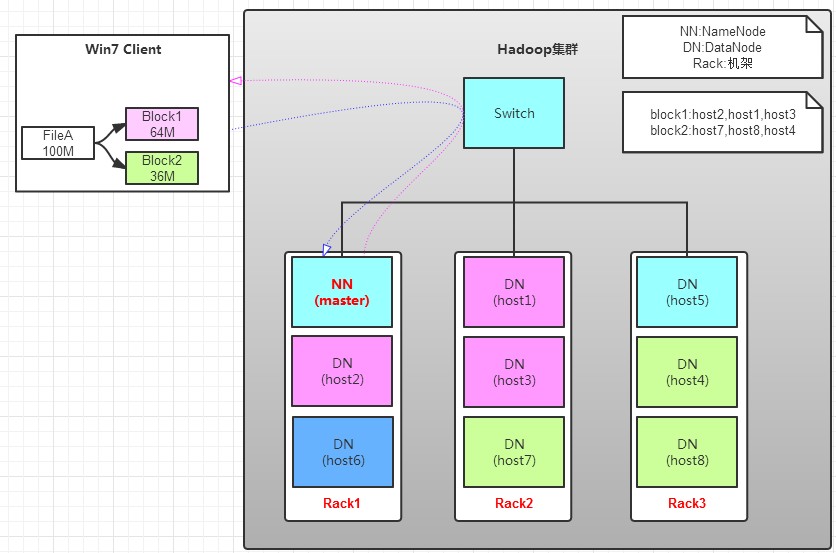
**①**写1T文件，我们需要3T的存储，3T的网络流量带宽。

**②**在执行读或写的过程中，NameNode和DataNode通过HeartBeat进行保存通信，确定DataNode活着。如果发现DataNode死掉了，就将死掉的DataNode上的数据，放到其他节点去。读取时，要读其他节点去。

**③**挂掉一个节点，没关系，还有其他节点可以备份；甚至，挂掉某一个机架，也没关系；其他机架上，也有备份。

**读操作：**





读操作就简单一些了，如图所示，client要从datanode上，读取FileA。而FileA由block1和block2组成。

那么，读操作流程为：

**a.** client向namenode发送读请求。

**b.** namenode查看Metadata信息，返回fileA的block的位置。

    block1:host2,host1,host3

    block2:host7,host8,host4

**c.** block的位置是有先后顺序的，先读block1，再读block2。而且block1去host2上读取；然后block2，去host7上读取；

上面例子中，client位于机架外，那么如果client位于机架内某个DataNode上，例如,client是host6。那么读取的时候，遵循的规律是：

**优选读取本机架上的数据**。

**HDFS文件权限**

与Linux文件权限类似

r:read；w:write；x：execute

如果Linux系统用用户xxx使用hadoop命令创建一个文件，那么，在hdfs中这个文件的owner就是xxx

HDFS的权限目的是将控制权交出去，本身只判断用户和权限，至于用户是不是真的，不管。

**HDFS安全模式**

NameNode启动的时候，首先讲fsimage载入内存，然后按照fsedits中的各项操作修改内存中的fsimage。

当元数据文件在内存中创建完成之后，在NameNode上创建一个新的fsimage替换原fsimage，同时创建一个空的fsedits文件（无需SecodaryNameNode参与）

这时，NameNode是运行在安全模式的。即对外（客户端）只读，所以此段时间内对hdfs的写入、删除、重命名都会失败。

然后NameNode收集各个DataNode的报告，当block达到最小副本数以上时，会被认为“安全”的了，在一定比例的数据块被确定为“安全”后，再过若干事件后，安全模式结束。

当检测到副本数不足的数据块时，该块会被复制，直到达到最小副本数。

Hdfs中数据块的位置并不是由namenode维护的，而是以块列表的形式存储在datanode中的。

**HDFS中常用到的命令**

1、hdfs fs

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | hdfs fs -ls /  hdfs fs -lsr  hdfs fs -mkdir /user/hadoop  hdfs fs -put a.txt /user/hadoop/  hdfs fs -get /user/hadoop/a.txt /  hdfs fs -cp src dst  hdfs fs -mv src dst  hdfs fs -cat /user/hadoop/a.txt  hdfs fs -rm /user/hadoop/a.txt  hdfs fs -rmr /user/hadoop/a.txt  hdfs fs -text /user/hadoop/a.txt  hdfs fs -copyFromLocal localsrc dst 与hadoop fs -put功能类似。  hdfs fs -moveFromLocal localsrc dst 将本地文件上传到hdfs，同时删除本地文件。 |

2、hdfs fsadmin

|  |  |
| --- | --- |
| 1  2  3 | hdfs dfsadmin -report  hdfs dfsadmin -safemode enter | leave | get | wait  hdfs dfsadmin -setBalancerBandwidth 1000 |

3、hdfs fsck

4、start-balancer.sh

负载均衡,可以使DataNode节点上选择策略重新平衡DataNode上的数据块的分布