Hadoop分布式文件存储系统

目录

应用背景

1990年，硬盘1370M，传输速度4.4.MB/s。5分钟读取完成。

2010年，硬盘1TB，传输速度100MB/s。2.5小时读取完成。

试想，如果我们有100块硬盘，每块硬盘存储1%的数据，并行读取。
那么我们需要多少时间读完所有数据呢？2分钟

应用背景

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（partition）并存储到若干台单独的计算机上。管理网络中跨多台计算机存储的文件系统称为分布式文件系统（distributed filesystem）。该系统架构于网络之上，势必会引入网络编程的复杂性，因此分布式文件系统比普通磁盘文件系统更为复杂。例如，使文件系统能够容忍节点故障且不丢失任何数据，就是一个极大的挑战。

Hadoop 有一个称为 HDFS 的分布式系统，全称为 Hadoop Distributed Filesystem 在非正式文档或旧文档以及配置文件中，有时也简称为 DFS ，它们是一回事儿。

HDFS Hadoop 的旗舰级文件系统，同时也是本章的重点，但实际上 Hadoop 一个综合性的文件系统抽象。

HDFS简介

一句话(官方)：分布式存储系统HDFS（ Hadoop Distributed File System）。

其实就是一个文件系统，类似于linux的文件系统。有目录，目录下可以存储文件。但它又是一个分布式的文件系统。

基本原理：

- 将文件切分成等大的数据块，分别存储到多台机器上。
- 每个数据块存在多个备份。
- 将数据切分、容错、负载均衡等功能透明化。
- 可将HDFS看成是一个巨大、具有容错性的磁盘。

HDFS实现原理

基本原理

- 将文件切分成等大的数据块，分别存储到多台机器上。
- 每个数据块存在多个备份。
- 将数据切分、容错、负载均衡等功能透明化。
- 可将HDFS看成是一个巨大、具有容错性的磁盘。

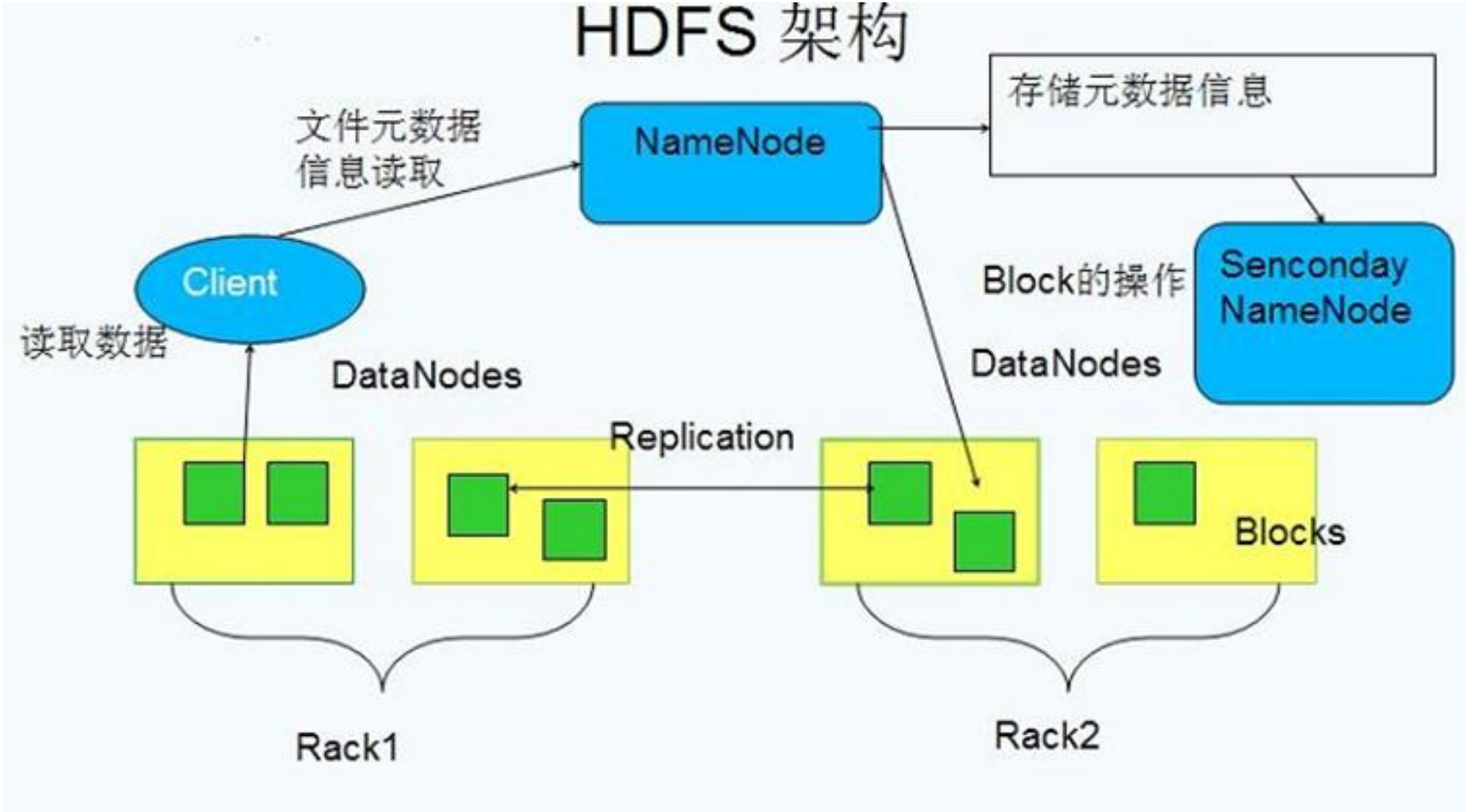
优点

- 处理超大文件。
- 流式的访问数据。
- 运行于廉价的商用机器集群上 。

缺点

- 不适合存储大量小文件。
- 不适合低延迟数据访问 。
- 不支持多用户写入及任意修改文件 。

HDFS实现原理


HDFS 架构
文件元数据
信息读取
NameNode
存储元数据信息
Client
读取数据
Block的操作
Senconday
NameNode
DataNodes
DataNodes
Replication
Blocks
Rack1
Rack2

HDFS实现原理-namenode和datanode

HDFS 集群有两类节点，并以管理者-工作者模式运行，即一个 namenode(管理者)和多个 datanode(工作者)。

namenode

namenode管理文件系统的命名空间。它维护着文件系统树及整棵树内所有的文件和目录。这些信息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(fs-image)和编辑日志(edit-logs)文件。
namenode 也记录着每个文件中各个块所在的数据节点信息，但它并不永久保存块的位置信息，因为这些信息会在系统启动时由数据节点重建。

datanode

是文件系统的工作节点。它们根据需要存储并检索数据块(受客户端或namenode 调度)，并且定期向 namenode 发送它们所存储的块的列表。

客户端(Client)

Client代表用户通过namenode和datanode访问整个文件系统。客户端提供一个类似于POSIX（可移植操作系统界面）的文件系统接口，因此用户在编程时无需知道namenode和datanode也可实现其功能。

Namenode备份机制

- 文件系统元数据持久状态的文件备份：一般的配置是，将持久状态写入本地磁盘的同时，写入一个远程挂载的网络文件系统 (NFS)。
- nameNodeHA。

HDFS实现原理

数据块

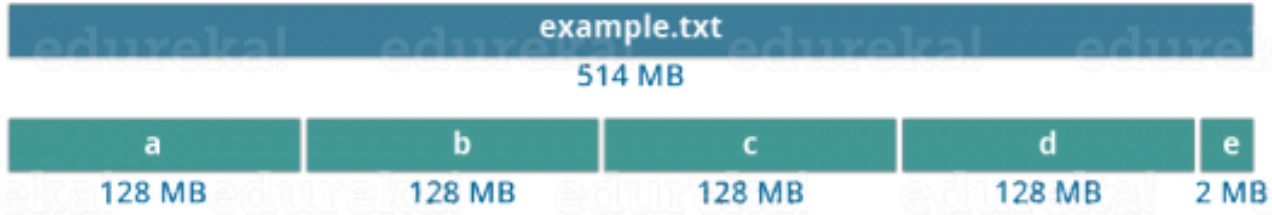
- 每个磁盘都有默认的数据块大小，这是磁盘进行数据读/写的最小单位。构建于单个磁盘之上的文件系统通过磁盘块来管理该文件系统中的块，该文件系统块的大小可以是磁盘块的整数倍。
- HDFS 同样也有块 (block) 的概念，但是大得多，默认为 128 MB 。与单一磁盘上的文件系统相似， HDFS 上的文件也被划分为块大小的多个分块 (chunk) ，作为独立的存储单元。但与其他文件系统不同的是， HDFS 中小于一个块大小的文件不会占据整个块的空间。

优点

- 一个大文件不用存储于整块磁盘上，可以分布式存储。
- 使用块抽象而非整个文件作为存储单元，大大简化了存储子系统的设计。这对于故障种类繁多的分布式系统尤为重要。

与磁盘管理相似，HDFS提供了fsck命令可以显示块信息。

hdfs fsck / -files -blocks



HDFS实现原理

NameNode

- NameNode是HDFS架构中的主节点。

功能

- 管理各个从节点的状态(DataNode)。
- 记录存储在HDFS上的所有数据的元数据信息。例如：block存储的位置，文件大小，文件权限，文件层级等等。这些信息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(FsImage): fsimage是HDFS文件系统存于硬盘中的元数据检查点，里面记录了自最后一次检查点之前HDFS文件系统中所有目录和文件的序列化信息
  编辑日志(edit-logs)文件:保存了自最后一次检查点之后所有针对HDFS文件系统的操作，比如：增加文件、重命名文件、删除目录等等。
- 记录了存储在HDFS上文件的所有变化，例如文件被删除，namenode会记录到editlog中。
- 接受DataNode的心跳和各个datanode上的block报告信息，确保DataNode是否存活。
- 负责处理所有块的复制因子。
- 如果DataNode节点宕机，NameNode会选择另外一个DataNode均衡复制因子，并做负载均衡。

参考：http://hadoop.apache.org/docs/r2.6.0/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html

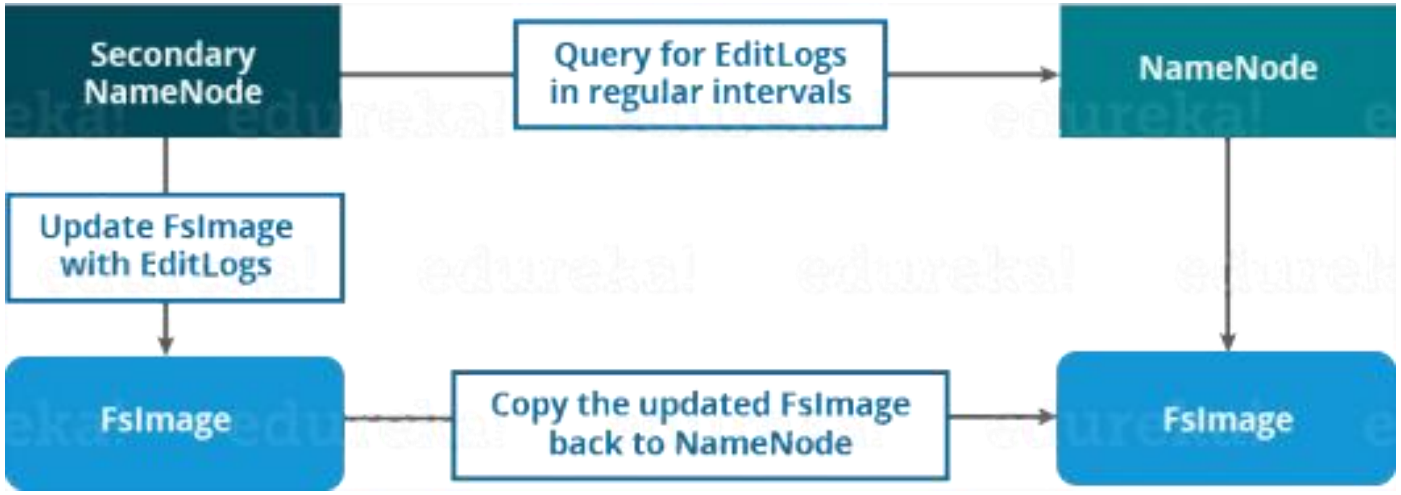
HDFS实现原理

Sencondary NameNode

- SNameNode是NameNode的助手，不要将其理解成是NameNode的备份。Secondary NameNode的整个目的在HDFS中提供一个Checkpoint Node，所以也被叫做checkpoint node。

功能

- 定时的从NameNode获取EditLogs,并更新到FsImage上。
- 一旦它有新的fsimage文件，它将其拷贝回NameNode上，NameNode在下次重启时回使用这个新的fsimage文件，从而减少重启的时间。



注意：关于NameNode是什么时候将改动写到edit logs中的？这个操作实际上是由DataNode的写操作触发的，当我们往DataNode写文件时，DataNode会跟NameNode通信，告诉NameNode什么文件的第几个block放在它那里，NameNode这个时候会将这些元数据信息写到edit logs文件中。

HDFS实现原理

DataNode

- DataNode是HDFS架构的从节点，管理各自节点的Block信息。

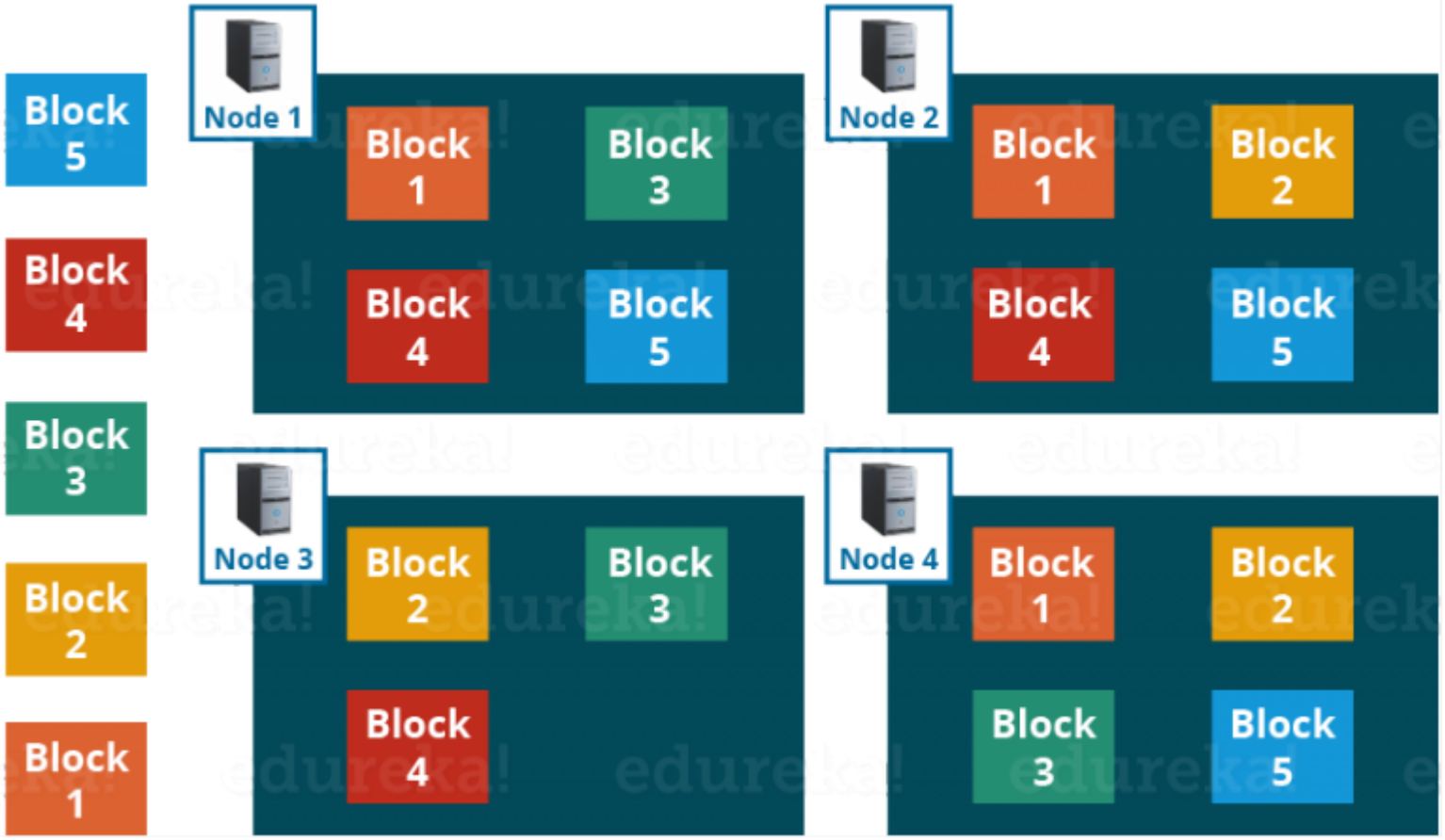
功能

- 多个DataNode分别运行在独立的节点上。
- 数据实际是存储到DataNode上面。
- DataNode执行客户端级别的读写请求。
- 向NameNode发送心跳(默认是3s)，报告各自节点的健康状况。

HDFS实现原理

复制因子

- HDFS为我们提供了可靠的存储，就是因为这个复制因子。默认复制因子是3。

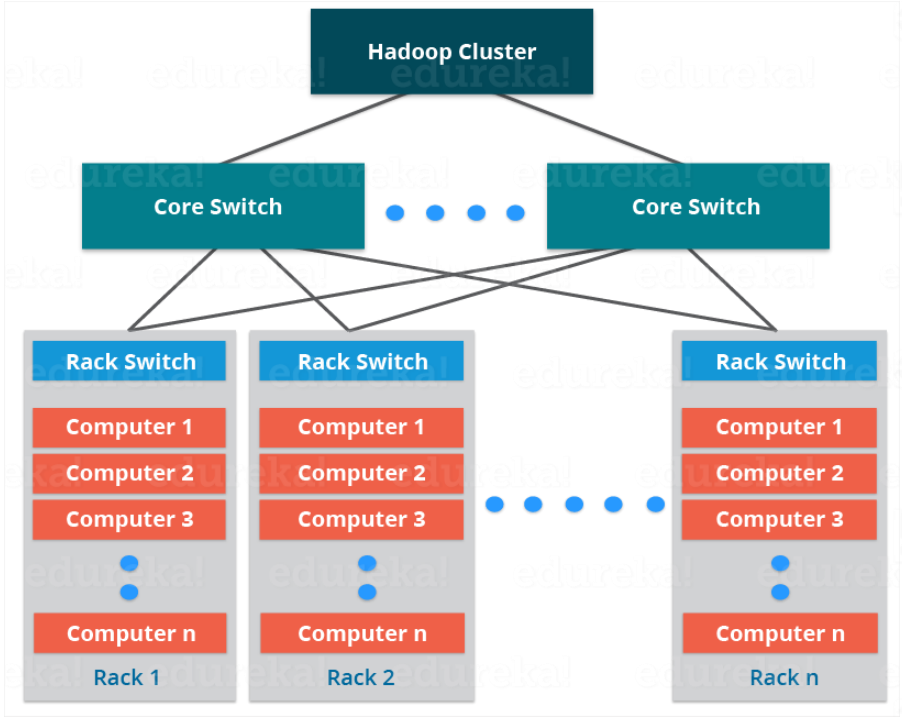


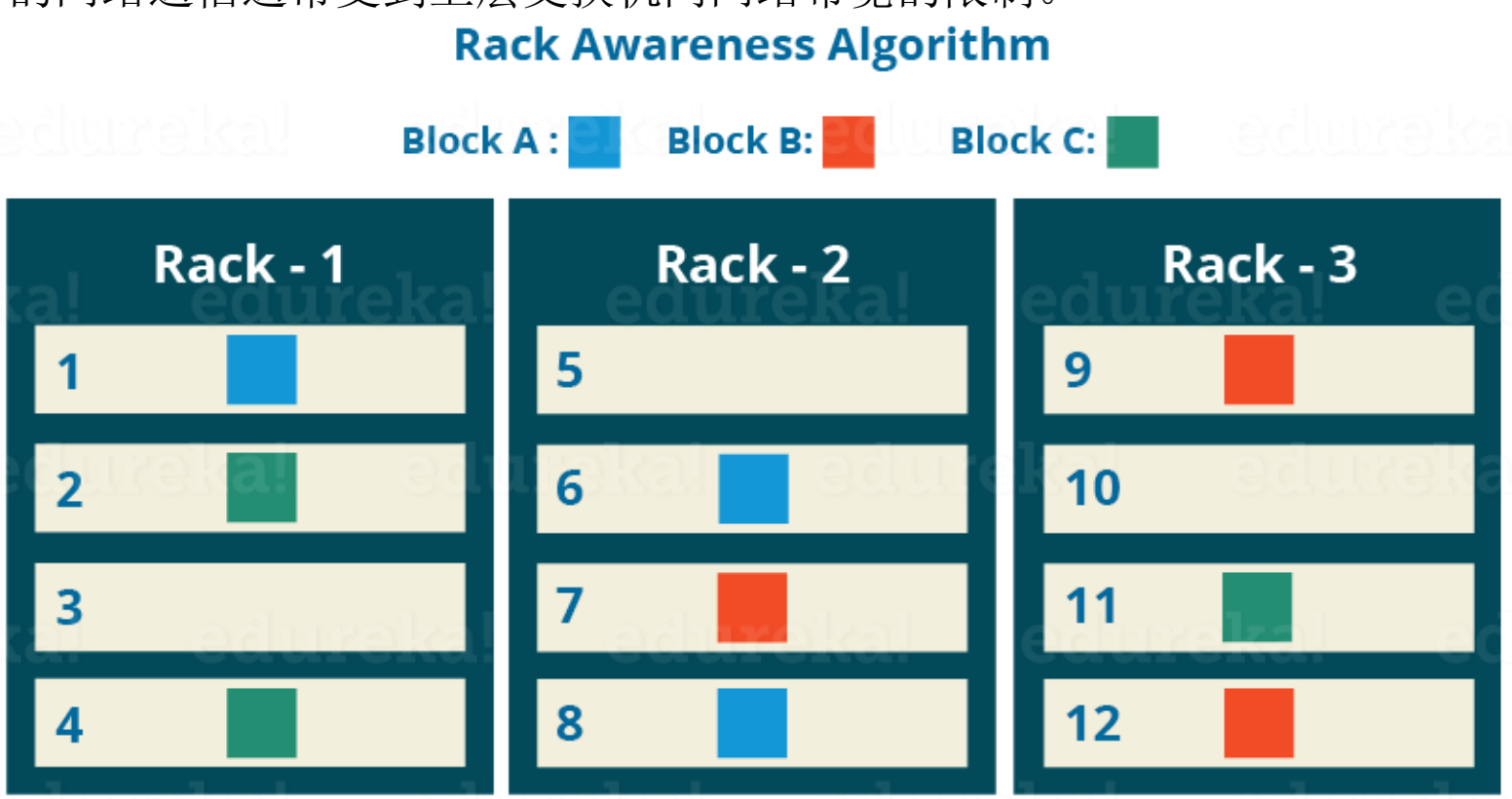
注意：
DataNode会定时发送心跳给NameNode，汇报各自节点的Block信息。NameNode收集到这些信息后，会对超出复制因子的Block删除，复制因子不足的Block做添加。

HDFS实现原理

机架感知

- 分布式的集群通常包含非常多的机器，由于受到机架槽位和交换机网口的限制，通常大型的分布式集群都会跨好几个机架，由多个机架上的机器共同组成一个分布式集群。机架内的机器之间的网络速度通常都会高于跨机架机器之间的网络速度，并且机架之间机器的网络通信通常受到上层交换机间网络带宽的限制。



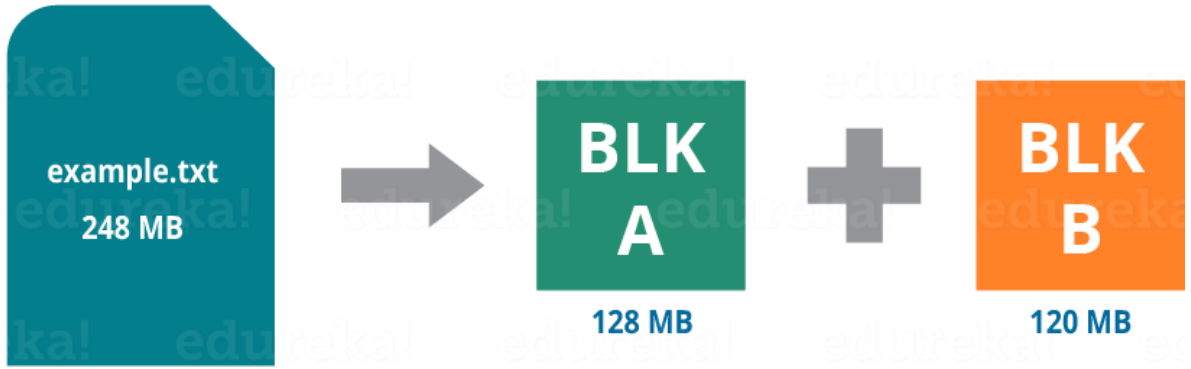


HDFS读写流程-写数据

假设我们有一个example.txt的文件，大小为248M。要将其写入到HDFS中。

假设我们使用HDFS设置的块大小为128M(默认值)。那么client会将此文件拆分成两个块。

第一个块是120MB(BlOCK 1),第二个块是120MB(BLOCK 2)。

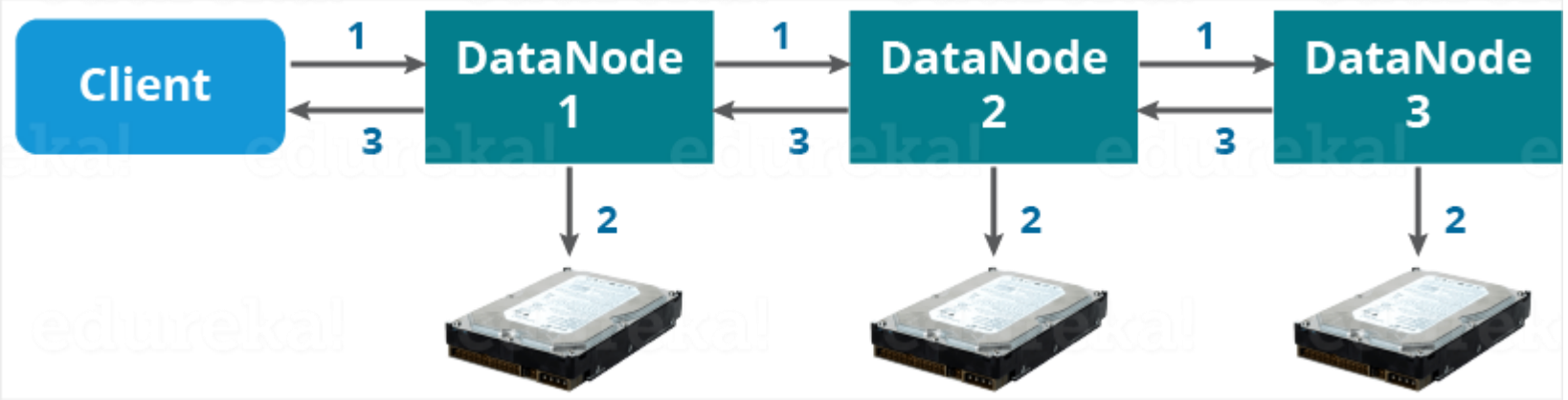


HDFS读写流程-写数据

- 首先，client请求namenode，要将多个块写入到HDFS。例如这里的Block A和Block B。
- NameNode会给client赋予写权限，并为client提供可以写入数据的DataNode的IP地址。Namenode在选择可写入数据的dataNode的规则是结合了DN的健康状态、复制因子、机架感知等因素随机选择的DN。假如复制因子是3(默认值)，那么会为每个block返回三个IP地址。例如NN为client提供了以下的IP地址列表。

  Block A, list A = {IP of DataNode 1, IP of DataNode 4, IP of DataNode 6}
  Block B, set B = {IP of DataNode 3, IP of DataNode 7, IP of DataNode 9}

- 整体的数据复制流程分一下三个阶段。1. 流水线建立；2. 复制数据；3. 关闭流水线；



HDFS读写流程-写数据之建立流水线(pipline)

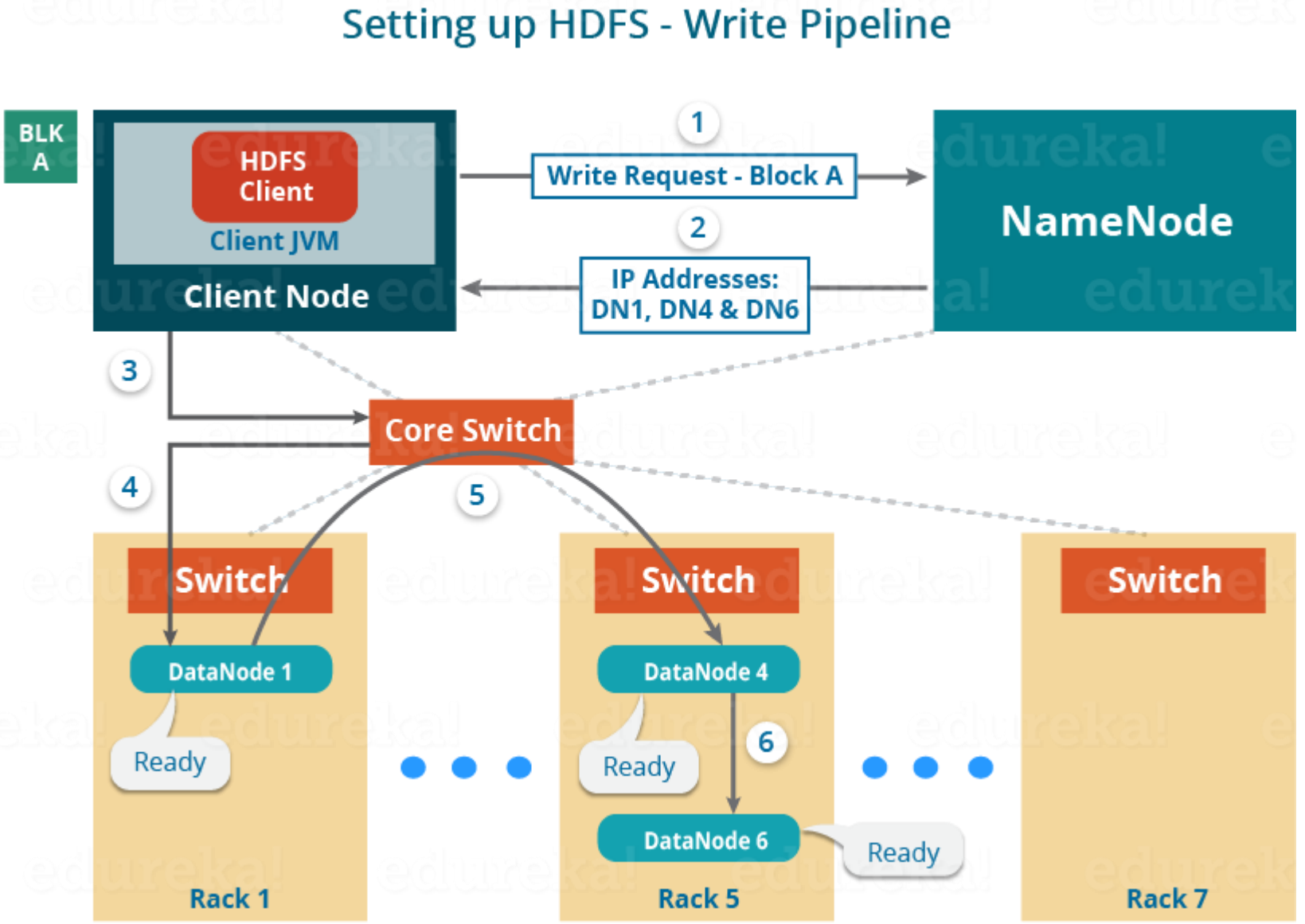
- 在写入数据之前，client首先要确认namenode提供的ip列表是否准备好了接收数据。Client通过连接各个块的ip列表来为每个块创建流水线。
- 我们以Block1为例。

For Block A, list A = {

IP of DataNode 1,

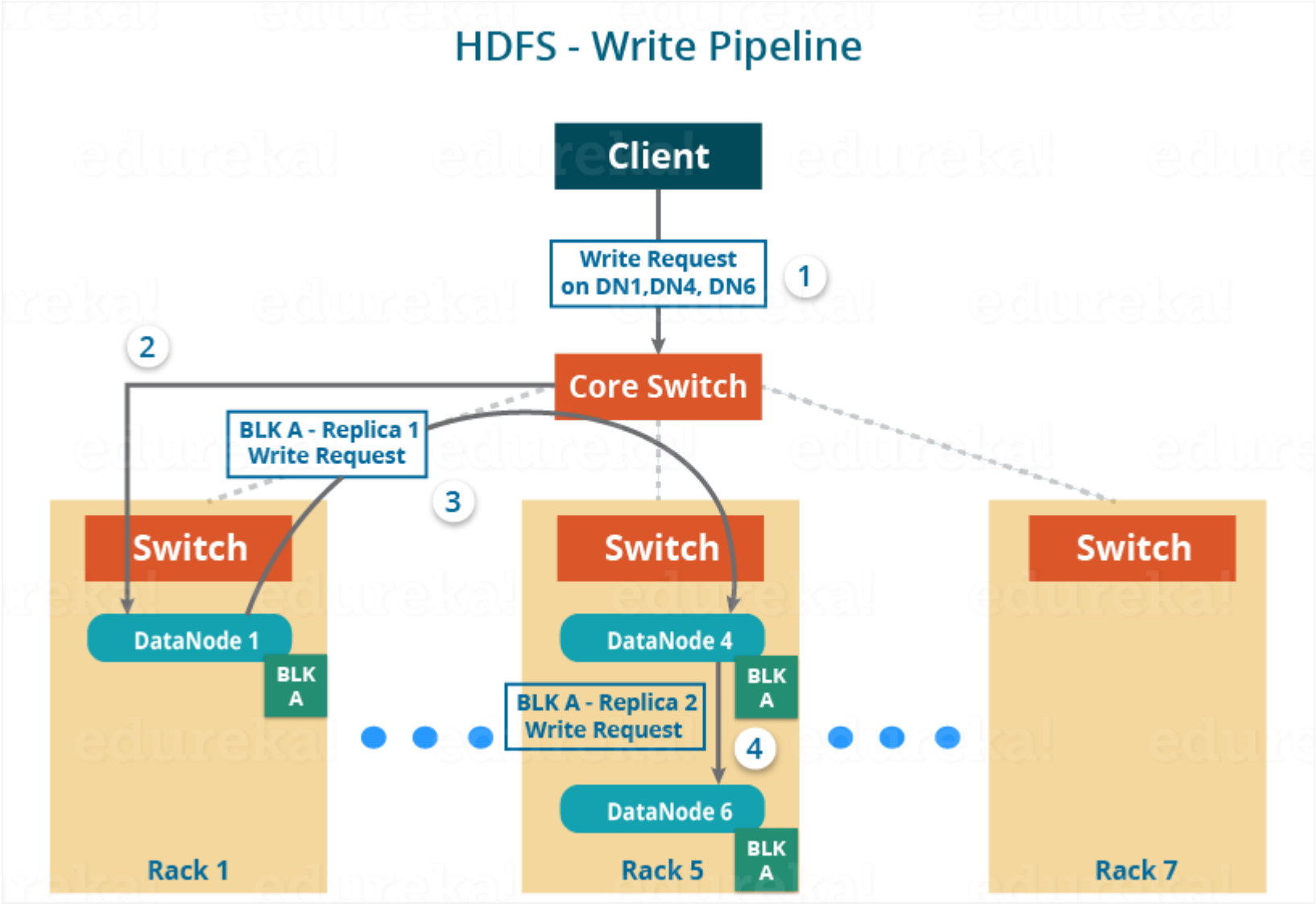
IP of DataNode 4,

IP of DataNode 6}



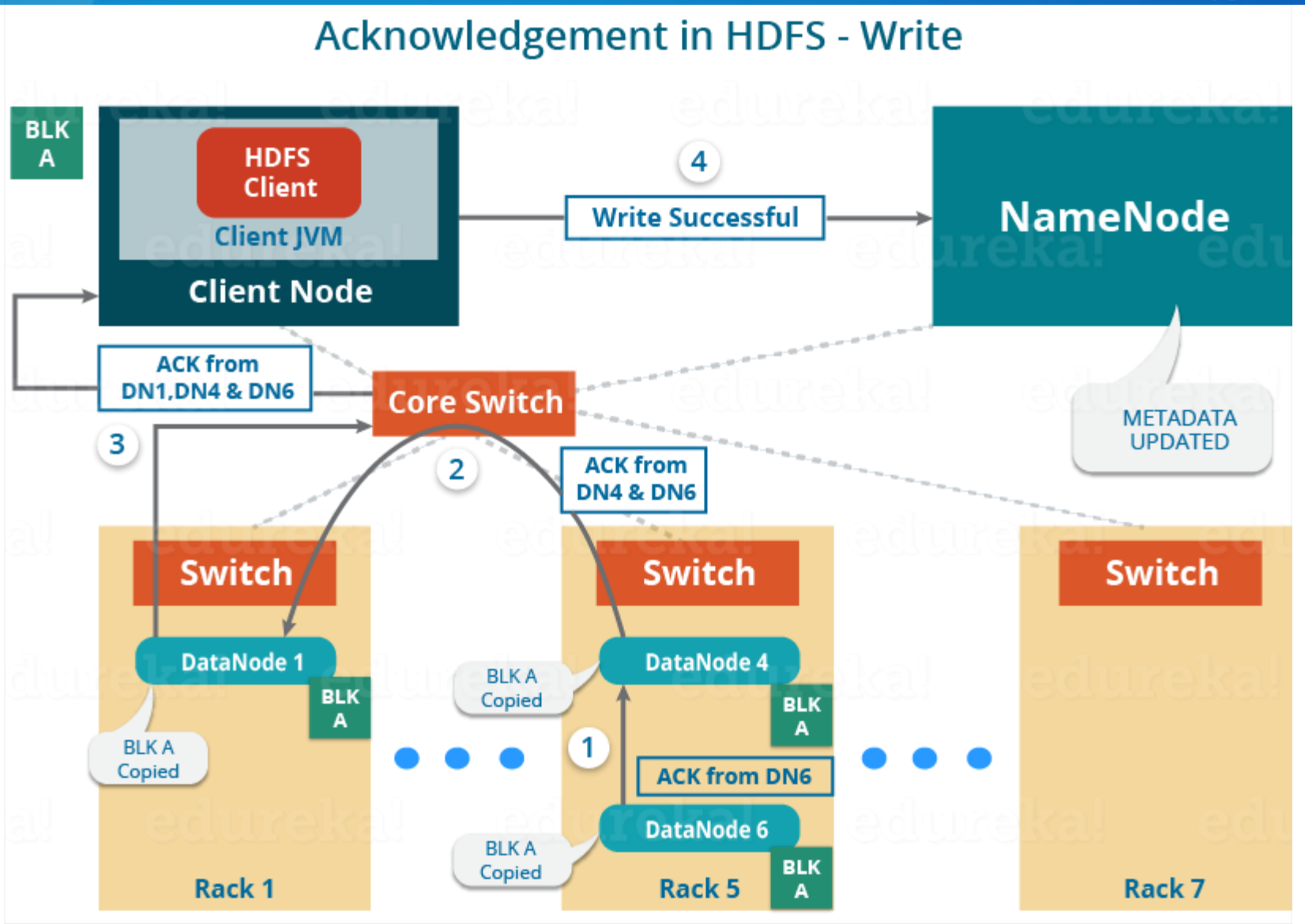
HDFS读写流程-写数据之数据复制(data Streaming)

- 流水线建立好只有，client将会向流水线中写入数据。
- 注意：Client只会将block A想DN1复制。其他节点复制是在DN之间完成的。



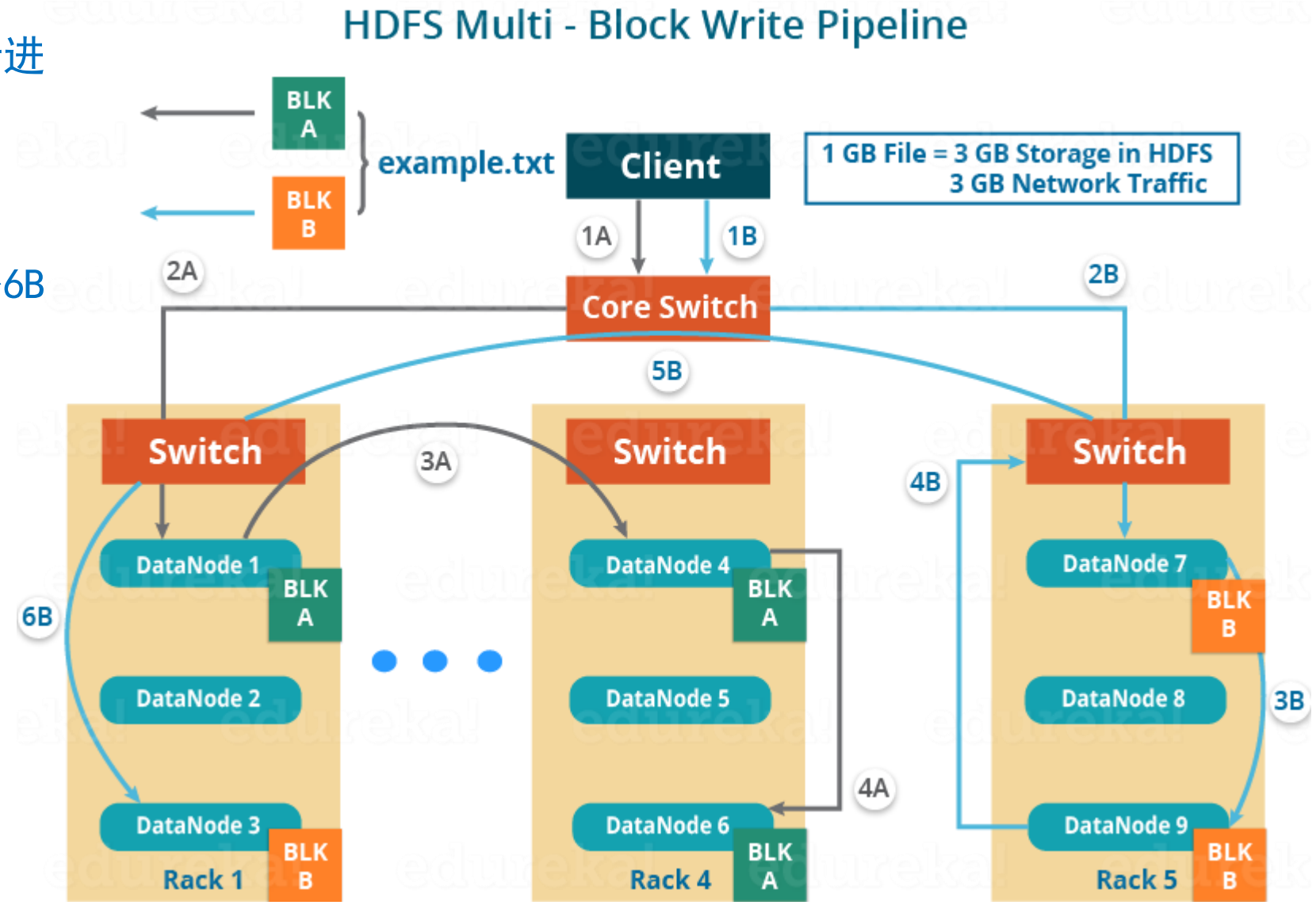
HDFS读写流程-写数据之关闭流水线

- 当数据复制到所有的DN完成之后，按照ip地址列表相反的方向，依次反馈写入成功的信息。
- DN6->DN4->DN1
- DN1将确认信息反馈给client，client再将确认信息反馈给NN，NN更新元数据信息，client关闭pipline。



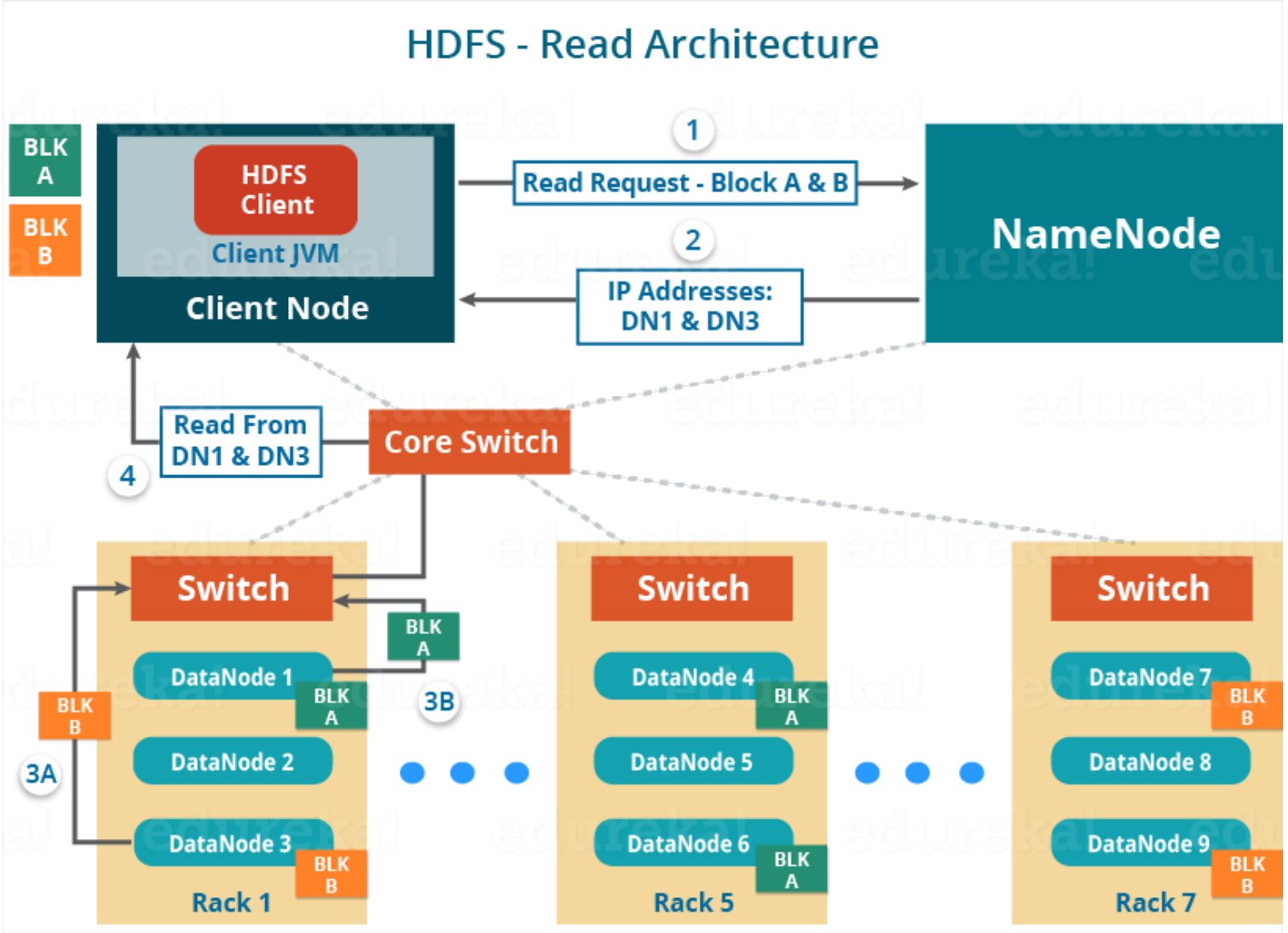
HDFS读写流程-多个Block同时写入

- Block A和Block B的写入是并行进行的。
- Block A:1A->2A->3A
- Block B:1B->2B->3B->4B->5B->6B



HDFS读写流程-文件读取

- Client请求namenode要读取exaple.txt文件。
- NN根据自己的元数据信息，反馈给client一个DataNode的列表(存储Block A和B)。
- Client连接DN，读取BlockA,Block B的数据。
- Client合并block A和Block B的数据。



HDFS filesystem shell

[cat]

使用方法：hadoop fs -cat URI [URI ...]

说明：将路径指定文件的内容输出到*stdout*

- hadoop fs -cat /dataAnalysis/duliming/input/mapreduceInput/wordCount.txt
- hadoop fs -cat file:///usr/local/usrJar/duliming/DataAnalysis/output/AEP/20161203094310.csv

[appendToFile]

使用方法：hadoop fs -appendToFile <localsrc> ... <dst>

说明：添加(追加)一个或多个源文件到目标文件中。或者将标准输入中的数据写入目标文件。

hadoop fs -appendToFile /usr/local/usrJar/duliming/pcbin/pcbin_month/000002_0 /usr/local/usrJar/duliming/pcbin/pcbin_month/000003_0 /dataAnalysis/duliming/output/mergeFile.csv

hadoop fs -appendToFile - /dataAnalysis/duliming/output/stdinMergeFile.csv

[chgrp]

使用方法：hadoop fs -chgrp [-R] GROUP URI [URI ...]

说明：改变hdfs文件所属组。修改该权限的用户必须拥有此目录权限或者父用户。 -R是否递归

hadoop fs -chgrp hdfs /dataAnalysis/duliming/output/mergeFile.csv

HDFS filesystem shell

[chmod]

使用方法：hadoop fs -chmod [-R] <MODE[,MODE]... | OCTALMODE> URI [URI ...]

说明：修改文件权限。-R表示是否递归。修改者必须拥有该目录权限，或者是拥有者的父用户。

Example:

hadoop fs -chmod 777 /dataAnalysis/duliming/output/mergeFile.csv

[chown]

使用方法：hadoop fs -chown [-R] [OWNER][:[GROUP]] URI [URI]

说明：修改文件拥有者。修改者必须拥有该文件或者是其父用户。-R表示递归。

Example:

hadoop fs -chown hdfs /dataAnalysis/duliming/output/mergeFile.csv

[copyFromLocal]

使用方法：hadoop fs -copyFromLocal <localsrc> URI

说明：拷贝本地文件到HDFS。类似于put命令，但可以拷贝目录。-f参数表示覆盖原来已存在目录。

Example:

hadoop fs -copyFromLocal /usr/local/usrJar/duliming/DataAnalysis/output/AEP /dataAnalysis/duliming/output/

HDFS filesystem shell

[copyToLocal]

使用方法：hadoop fs -copyToLocal [-ignorecrc] [-crc] URI <localdst>

说明：拷贝HDFS文件到本地，类似于get命令，但可以拷贝目录。

Example:

Hadoop fs –copyToLocal /dataAnalysis/duliming/output/ /usr/local/usrJar/duliming/DataAnalysis/output/AEP

[count]

使用方法：hadoop fs -count [-q] [-h] [-v] <paths>

说明：统计目录下文件数，空间占用情况。

-h:输出格式化后的信息。 -v：输出表头

Example:

hadoop fs -count -q -h -v /dataAnalysis/duliming/output

-count	-count -q	输出列	说明
	√	QUOTA	命名空间quota(剩余能创建的文件数目)
	√	REMAINING_QUATA	剩余的命名空间quota(剩余能创建的文件数目)
	√	SPACE_QUOTA	物理空间的quota （限制磁盘空间占用大小）
	√	REMAINING_SPACE_QUOTA	剩余的物理空间
√	√	DIR_COUNT	目录数目
√	√	FILE_COUNT	文件数目
√	√	CONTENT_SIZE	目录逻辑空间大小
√	√	PATHNAME	路径

HDFS filesystem shell

[cp]

使用方法：hadoop fs -cp [-f] [-p | -p[topax]] URI [URI ...] <dest>

说明：将文件从源路径复制到目标路径。这个命令允许有多个源路径，此时目标路径必须是一个目录。 -f 如果目标目录已存在，则覆盖之前的目录。

Example:

hadoop fs -cp -f /dataAnalysis/duliming/output/wordcount /dataAnalysis/duliming/output/wordcount1

[df]

使用方法：hadoop fs -df [-h] URI [URI ...]

显示目录空闲空间，-h：转换为更加易读的方式，比如67108864用64M代替。

Example:

hadoop fs -df -h /dataAnalysis

[expunge]

使用方法： hadoop fs –expunge

说明：清空回收站

HDFS filesystem shell

[get]

使用方法：hadoop fs -get [-ignorecrc] [-crc] <src> <localdst>

说明：复制文件到本地文件系统。可用-ignorecrc选项复制CRC校验失败的文件。使用-crc选项复制文件以及CRC信息。Example:

hadoop fs -get /user/hadoop/file localfile

hadoop fs -get hdfs://host:port/user/hadoop/file localfile

[getmerge]

使用方法：hadoop fs -getmerge <src> <localdst> [addnl]

接受一个源目录和一个目标文件作为输入，并且将源目录中所有的文件连接成本地目标文件。addnl是可选的，用于指定在每个文件结尾添加一个换行符。

[ls]

使用方法：hadoop fs -ls <args>

说明：如果是文件，则按照如下格式返回文件信息：

文件名 <副本数> 文件大小 修改日期 修改时间 权限 用户ID 组ID

如果是目录，则返回它直接子文件的一个列表，就像在Unix中一样。目录返回列表的信息如下：

目录名 <dir> 修改日期 修改时间 权限 用户ID 组ID

Example:

hadoop fs -ls /user/hadoop/file1 /user/hadoop/file2 hdfs://host:port/user/hadoop/dir1 /nonexistentfile

HDFS filesystem shell

[lsr]

使用方法：hadoop fs -lsr <args>

说明：ls命令的递归版本。类似于Unix中的ls -R。

[mkdir]

使用方法：hadoop fs -mkdir <paths>

接受路径指定的uri作为参数，创建这些目录。其行为类似于Unix的mkdir -p，它会创建路径中的各级父目录。

Example:

hadoop fs -mkdir /user/hadoop/dir1 /user/hadoop/dir2

hadoop fs -mkdir hdfs://host1:port1/user/hadoop/dir hdfs://host2:port2/user/hadoop/dir

[mv]

使用方法： hadoop fs -mv URI [URI …] <dest>

说明：将文件从源路径移动到目标路径。这个命令允许有多个源路径，此时目标路径必须是一个目录。不允许在不同的文件系统间移动文件。

hadoop fs -mv /user/hadoop/file1 /user/hadoop/file2

hadoop fs -mv hdfs://host:port/file1 hdfs://host:port/file2 hdfs://host:port/file3 hdfs://host:port/dir1

HDFS filesystem shell

[put]

使用方法： hadoop fs -put <localsrc> ... <dst>

说明：从本地文件系统中复制单个或多个源路径到目标文件系统。也支持从标准输入中读取输入写入目标文件系统。

Example:

hadoop fs -put localfile /user/hadoop/hadoopfile

hadoop fs -put localfile1 localfile2 /user/hadoop/hadoopdir

hadoop fs -put - hdfs://host:port/hadoop/hadoopfile （从标准输入中读取输入。）

[rm]

使用方法： hadoop fs -rm URI [URI …]

删除指定的文件。只删除非空目录和文件。-r 递归删除。

Example:

hadoop fs -rm hdfs://host:port/file /user/hadoop/emptydir

[setrep]

使用方法： hadoop fs -setrep [-R] [-w] <numReplicas> <path>

说明：改变一个文件的副本系数。-R选项用于递归改变目录下所有文件的副本系数。 -w选项指定，改请求等待操作执行结束。

hadoop fs -setrep -w 3 -R /user/hadoop/dir1

HDFS filesystem shell

[stat]

使用方法： hadoop fs -tail [-f] URI

说明：返回指定路径的统计信息。
%b:文件大小
%F:文件类型
%g:所属组
%o:block大小
%n:文件名
%r:复制因子数
%u:文件所有者
%Y,%y:修改日期
Example:
hadoop fs -stat "%F %u:%g %b %y %n" /file

[tail]

使用方法： hadoop fs -tail [-f] URI

说明：将文件尾部1K字节的内容输出到stdout。支持-f选项，行为和Unix中一致。 。
Example:
hadoop fs -tail pathname

HDFS filesystem shell

[text]

使用方法： hadoop fs -text <src>

说明：类似于cat。将源文件输出为文本格式。允许的格式是zip和TextRecordInputStream。

[touchz]

使用方法：hadoop fs -touchz URI [URI ...]

说明：创建一个0字节的空文件。

Example:

Hadoop fs -touchz pathname

[truncate]

使用方法：hadoop fs -truncate [-w] <length> <paths>

说明：文件截断，-w要求该命令等待恢复完成。

Example:

hadoop fs -truncate 55 /user/hadoop/file1 /user/hadoop/file2

hadoop fs -truncate -w 127 hdfs://nn1.example.com/user/hadoop/file1

HDFS filesystem shell

[usage]

使用方法： hadoop fs -usage command

说明：返回命令的帮助信息。

[find]

使用方法：hadoop fs -find <path> ... <expression> ...

说明：查找满足表达式的文件和文件夹。没有配置path的话，默认的就是全部目录/；如果表达式没有配置，则默认为-print。

-name pattern 不区分大小写，对大小写不敏感

-iname pattern 对大小写敏感。

-print 打印。

-print0 打印在一行。

Example:

。

Example:hadoop fs -find / -name test –print

[getfacl]

使用方法：hadoop fs -getfacl [-R] <path>

说明：获取文件的acl权限。-R指定递归查找

hadoop fs -getfacl -R /dir

HDFS dfsadmin

bin/hadoop dfsadmin命令支持一些和HDFS管理相关的操作。bin/hadoop dfsadmin -help 命令能列出所有当前支持的命令。

用法： **hadoop dfsadmin [GENERIC_OPTIONS]**

[-report]

[-safemode enter | leave | get | wait]

[-refreshNodes]

[-finalizeUpgrade]

[-upgradeProgress status | details | force]

[-metasave filename]

[-setQuota <quota> <dirname>...<dirname>]

[-clrQuota <dirname>...<dirname>]

[-help [cmd]]

说明：

HDFS dfsadmin

命令选项	描述
-report:	报告文件系统的基本信息和统计信息。
-safemode enter | leave | get | wait	安全模式维护命令。安全模式是Namenode的一个状态，这种状态下，Namenode 1. 不接受对名字空间的更改(只读) 2. 不复制或删除块 Namenode会在启动时自动进入安全模式，当配置的块最小百分比数满足最小的副本数条件时，会自动离开安全模式。安全模式可以手动进入，但是这样的话也必须手动关闭安全模式。
-refreshNodes	重新读取hosts和exclude文件，更新允许连到Namenode的或那些需要退出或入编的Datanode的集合。
-finalizeUpgrade	终结HDFS的升级操作。Datanode删除前一个版本的工作目录，之后Namenode也这样做。这个操作完结整个升级过程。
-upgradeProgress status | details | force	请求当前系统的升级状态，状态的细节，或者强制升级操作进行。

HDFS dfsadmin

命令选项	描述
-metasave filename	保存Namenode的主要数据结构到hadoop.log.dir属性指定的目录下的<filename>文件。 对于下面的每一项，<filename>中都会一行内容与之对应 1. Namenode收到的Datanode的心跳信号 2. 等待被复制的块 3. 正在被复制的块 4. 等待被删除的块
-setQuota <quota> <dirname>...<dirname>	为每个目录 <dirname>设定配额<quota>。目录配额是一个长整型整数，强制限定了目录树下的名字个数。 命令会在这个目录上工作良好，以下情况会报错： 1. N不是一个正整数，或者 2. 用户不是管理员，或者 3. 这个目录不存在或是文件，或者 4. 目录会马上超出新设定的配额。
-clrQuota <dirname>...<dirname>	为每一个目录<dirname>清除配额设定。 命令会在这个目录上工作良好，以下情况会报错： 1. 这个目录不存在或是文件，或者 2. 用户不是管理员。 如果目录原来没有配额不会报错。

HDFS dfsadmin

Example:
hdfs –safemode enter|leave|get|wait
hdfs dfsadmin -getDatanodeInfo idh101:8010(获取datanode信息，该命令用于检测datanode是否存活)
hdfs dfsadmin -shutdownDatanode idh101:8010 [upgrade]
hdfs dfsadmin -rollingUpgrade <query|prepare|finalize>
参照： hadoop dfsadmin -setSpaceQuota 1g /user/seamon/
hadoop dfsadmin -clrSpaceQuota /user/seamon

http://hadoop.apache.org/docs/r1.0.4/cn/commands_manual.html#dfsadmin
http://hadoop.apache.org/docs/r2.7.3/hadoop-project-dist/hadoop-hdfs/HDFSCommands.html#dfsadmin
http://hadoop.apache.org/docs/r2.7.3/hadoop-project-dist/hadoop-hdfs/HdfsRollingUpgrade.html

HDFS DistCp

DistCp（分布式拷贝）是用于大规模集群内部和集群之间拷贝的工具。 它使用Map/Reduce实现文件分发，错误处理和恢复，以及报告生成。 它把文件和目录的列表作为map任务的输入，每个任务会完成源列表中部分文件的拷贝。 由于使用了Map/Reduce方法，这个工具在语义和执行上都会有特殊的地方。

使用方法：

hadoop distcp hdfs://nn1:8020/foo/bar \ hdfs://nn2:8020/bar/foo

命令行中可以指定多个源目录：

```
hadoop distcp hdfs://nn1:8020/foo/a \
              hdfs://nn1:8020/foo/b \
              hdfs://nn2:8020/bar/foo
```

或者使用-f选项，从文件里获得多个源：

```
hadoop distcp -f hdfs://nn1:8020/srclist \
                hdfs://nn2:8020/bar/foo
```

其中srclist 的内容是

```
  hdfs://nn1:8020/foo/a
  hdfs://nn1:8020/foo/b
```

HDFS DistCp

选项

选项索引

标识	描述	备注
-p[rbugp]	Preserve r: replication number b: block size u: user g: group p: permission	修改次数不会被保留。并且当指定 -update 时，更新的状态不会 被同步，除非文件大小不同（比如文件被重新创建）。
-i	忽略失败	就像在 附录中提到的，这个选项会比默认情况提供关于拷贝的更精确的统计， 同时它还将保留失败拷贝操作的日志，这些日志信息可以用于调试。最后，如果一个map失败了，但并没完成所有分块任务的尝试，这不会导致整个作业的失败。
-log <logdir>	记录日志到 <logdir>	DistCp为每个文件的每次尝试拷贝操作都记录日志，并把日志作为map的输出。 如果一个map失败了，当重新执行时这个日志不会被保留。
-m <num_maps>	同时拷贝的最大数目	指定了拷贝数据时map的数目。请注意并不是map数越多吞吐量越大。
-overwrite	覆盖目标	如果一个map失败并且没有使用-i选项，不仅仅那些拷贝失败的文件，这个分块任务中的所有文件都会被重新拷贝。 就像下面提到的，它会改变生成目标路径的语义，所以 用户要小心使用这个选项。
-update	如果源和目标的大小不一样则进行覆盖	像之前提到的，这不是"同步"操作。 执行覆盖的唯一标准是源文件和目标文件大小是否相同；如果不同，则源文件替换目标文件。 像 下面提到的，它也改变生成目标路径的语义， 用户使用要小心。
-f <urilist_uri>	使用<urilist_uri> 作为源文件列表	这等价于把所有文件名列在命令行中。 urilist_uri 列表应该是完整合法的URI。

参照：http://hadoop.apache.org/docs/r1.0.4/cn/distcp.html

HDFS getConf

说明：用于获取hdfs配置信息。

Example:

hdfs getconf -namenodes
hdfs getconf -secondaryNameNodes
hdfs getconf -backupNodes
hdfs getconf -includeFile
hdfs getconf -excludeFile
hdfs getconf -nnRpcAddresses
hdfs getconf -confKey dfs.namenode.name.dir
hdfs getconf -confKey dfs.datanode.data.dir
hdfs getconf -confKey dfs.replication

参照：http://hadoop.apache.org/docs/r2.7.3/hadoop-project-dist/hadoop-hdfs/HDFSCommands.html#getconf

COMMAND_OPTION	Description
-namenodes	gets list of namenodes in the cluster.
-secondaryNameNodes	gets list of secondary namenodes in the cluster.
-backupNodes	gets list of backup nodes in the cluster.
-includeFile	gets the include file path that defines the datanodes that can join the cluster.
-excludeFile	gets the exclude file path that defines the datanodes that need to decommissioned.
-nnRpcAddresses	gets the namenode rpc addresses
-confKey [key]	gets a specific key from the configuration

Gets configuration information from the configuration directory, post-processing.

HDFS oev

用法： hdfs oev [OPTIONS] -i INPUT_FILE -o OUTPUT_FILE

说明：命令hdfs oev用于查看edits文件。

-i,–inputFile <arg> 输入edits文件，如果是xml后缀，表示XML格式，其他表示二进制。

-o,–outputFile <arg> 输出文件，如果存在，则会覆盖。

可选参数：

-p,–processor <arg> 指定转换类型: binary (二进制格式), xml (默认， XML格式),stats (打印edits文件的静态统计信息)

-f,–fix-txids 重置输入edits文件中的transaction IDs

-r,–recover 使用recovery模式，跳过eidts中的错误记录。

-v,–verbose 打印处理时候的输出。

Example:

hdfs oev -i /data1/hadoop/hdfs/name/current/edits_0000000000019382469-0000000000019383915 -o /home/hadoop/edits.xml

未指定-p选项，默认转换成xml格式，查看edits.xml文件。输出的xml文件中记录了文件路径（PATH），修改时间（MTIME）、添加时间（ATIME）、客户端名称（CLIENT_NAME）、客户端地址（CLIENT_MACHINE）、权限（PERMISSION_STATUS）等非常有用的信息。

当edits文件破损进而导致HDFS文件系统出现问题时，可以通过将原有的binary文件转换为xml文件，并手动编辑xml文件然后转回binary文件来实现。

参照： http://lxw1234.com/archives/2015/08/442.htm

HDFS oiv

用法： hdfs oiv [OPTIONS] -i INPUT_FILE

说明：命令hdfs oiv用于将fsimage文件转换成其他格式的，如文本文件、XML文件。

-i,–inputFile <arg>　　输入FSImage文件。

-o,–outputFile <arg> 输出转换后的文件，如果存在，则会覆盖。

可选参数：

-p,–processor <arg>　将FSImage文件转换成哪种格式： (Ls|XML|FileDistribution).默认为Ls.

-h,–help　　显示帮助信息

Example:

hdfs oiv -i /data1/hadoop/dfs/name/current/fsimage_0000000000019372521 -o /home/hadoop/fsimage.txt

由于未指定-p选项，默认为Ls，出来的结果和执行hadoop fs –ls –R一样。

hdfs oiv -i /data1/hadoop/dfs/name/current/fsimage_0000000000019372521 -o /home/hadoop/fsimage.xml**-p XML**

指定-p XML，将fsimage文件转换成XML格式，查看fsimage.xml

XML文件中包含了fsimage中的所有信息，比如inodeid、type、name、修改时间、权限、大小等等。

参照：http://lxw1234.com/archives/2015/08/440.htm

HDFS fsck

用法： hdfs fsck <path>
[-list-corruptfileblocks |[-move | -delete | -openforwrite] [-files [-blocks [-locations | -racks]]] [-includeSnapshots] [-storagepolicies] [-blockId <blk_Id>]

说明：检查HDFS上文件和目录的健康状态、获取文件的block信息和位置信息等。

-list-corruptfileblocks:查看文件中损坏的块
-move:将损坏的文件移动至/lost+found目录
-delete:删除损坏的文件
-files:检查并列出所有文件状态
-openforwrite:检查并打印正在被打开执行写操作的文件
-blocks:打印文件的Block报告（需要和-files一起使用）
-locations:打印文件块的位置信息（需要和-files -blocks一起使用。）
-racks:打印文件块位置所在的机架信息

Example:

hdfs fsck /hivedata/warehouse/liuxiaowen.db/lxw_product_names/ -list-corruptfileblocks
hdfs fsck /hivedata/warehouse/liuxiaowen.db/lxw_product_names/part-00168 –move
hdfs fsck /hivedata/warehouse/liuxiaowen.db/lxw_product_names/part-00168 –delete
hdfs fsck /hivedata/warehouse/liuxiaowen.db/lxw_product_names/ -files

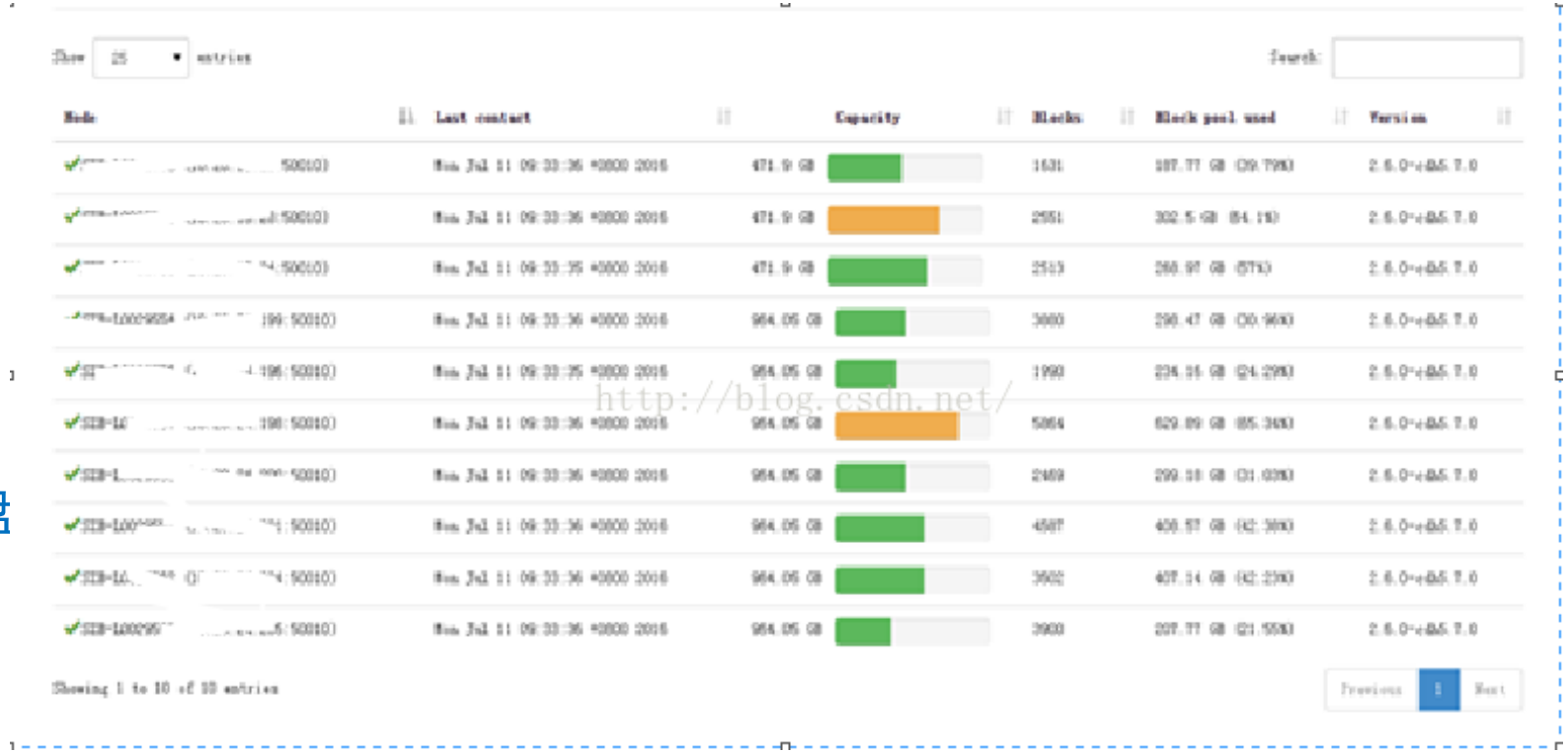
参照：http://lxw1234.com/archives/2015/08/452.htm

HDFS balancer

集群磁盘数据不均衡导致的原因有很多情况。

- 添加新的DataNode节点。
- 人为干预，修改block副本数。
- 各个机器磁盘大小不一致。
- 长时间运行大量的delete操作等。

Hadoop提供了Balancer工具来改善磁盘不均衡的问题。



HDFS balancer

用法： hdfs balancer [-threshold <threshold>] [-policy <policy>] [-exclude [-f <hosts-file> | <comma-separated list of hosts>]] [-include [-f <hosts-file> | <comma-separated list of hosts>]] [-idleiterations <idleiterations>]

说明：用于平衡hadoop集群中各datanode中的文件块分布，以避免出现部分datanode磁盘占用率高的问题。

-threshold <threshold>:表示的平衡的阀值，取值范围在0%到100%之间。每个Datanode中空间使用率与HDFS集群总的空间使用率的差距百分比。

-policy <policy>: 平衡策略，默认DataNode。应用于重新平衡 HDFS 存储的策略。默认DataNode策略平衡了 DataNode 级别的存储。这类似于之前发行版的平衡策略。BlockPool 策略平衡了块池级别和 DataNode 级别的存储。BlockPool 策略仅适用于 Federated HDFS 服务。

-exclude/include:参数-exclude和-include是用来选择balancer时，可以指定哪几个DataNode之间重分布，也可以从HDFS集群中排除哪几个节点不需要重分布

-idleiterations <iterations>:迭代检测的次数。

Example:

hdfs balancer –threshold 10

HDFS 快照

HDFS快照是一个只读的基于时间点文件系统拷贝。快照可以是整个文件系统的也可以是一部分。常用来作为数据备份，防止用户错误和容灾。
在datanode 上面的blocks 不会复制，做Snapshot 的文件是纪录了block的列表和文件的大小，但是没有数据的复制
Snapshot 并不会影响HDFS 的正常操作：修改会按照时间的反序记录，这样可以直接读取到最新的数据。快照数据是当前数据减去修改的部分计算出来的。
快照会存储在snapshottable的目录下。snapshottable下存储的snapshots 最多为65535个。没有限制snapshottable目录的数量。管理员可以设置任何的目录成为snapshottable。如果snapshottable里面存着快照，那么文件夹不能删除或者改名。

快照常用操作

hdfs dfsadmin -allowSnapshot <path>：快照目录建立。如果这个操作成果，那么目录会变成snapshottable
hdfs dfsadmin -disallowSnapshot <path>：文件夹里面的所有快照在失效快照前必须被删除，如果没有该目录会被建立。
hdfs dfsadmin -createSnapshot <path> [<snapshotName>]：snapshottable目录创建一个快照。这个操作需要snapshottable目录的权限。
hdfs dfsadmin -deleteSnapshot <path> <snapshotName>：从一个snapshottable目录删除的快照。这个操作需要snapshottable目录的权限。
hdfs dfsadmin -renameSnapshot <path> <oldName> <newName>：重命名快照。这个命令也需要snapshottable目录的权限。
hdfs lsSnapshottableDir：获取当前用户的所有snapshottable。
hdfs snapshotDiff <path> <fromSnapshot> <toSnapshot>：得到两个快照之间的不同。需要两个目录的权限。
参照：http://blog.csdn.net/linlinv3/article/details/44564313

HDFS Java API

1.创建maven项目，加入如下依赖。

```
<dependencies>
<dependency>
<groupId>org.apache.hadoop</groupId>
<artifactId>hadoop-client</artifactId>
<version>2.7.3</version>
</dependency>
</dependencies>
```

2.添加hdfs-site.xml文件，core-site.xml文件到src/main/resources目录中。

3. FileSystem操作HDFS。

```
Configuration conf = new Configuration();

FileSystem fs = FileSystem.get(conf);

Path srcPath = new Path("/test");

fs.mkdirs(srcPath);// 创建test目录
```

作业

1.回顾以上所讲的知识点，hdfs架构组成，hdfs读写文件流程，副本策略，hdfs shell命令，谈谈自己的理解。

2.hdfs相关操作练习(每步骤运行结果截图并整理成word文档)。

1）在hdfs中创建目录/data/姓名/

2）将指定的文件夹内容通过hdfs shell命令上传到创建的hdfs目录中。

3）查看/data/姓名/目录下文件列表。

4）查看/data/姓名/目录下统计目录下文件数，空间占用情况。

5）递归将/data/姓名/下的文件拥有者修改为自己电脑的用户名。

6）递归将/data/姓名/下的文件acl权限修改为只能自己读写执行，组和其他用户只可以读。

7）在/data/姓名/目录下创建一个空的文件word.txt。

8）向/data/姓名/word.txt中追加文字。（内容自己定，最好是英文）。

9）hdfs命令查看/data/姓名/word.txt中的内容。

10）以上操作请使用java API再操作一遍。

总结

- HDFS架构及其实现原理。
  - ✓ 中心化设计，主从结构。NN+DN+SD+Client。
  - ✓ NN存储元数据信息，内存+持久化数据(fsimage+editlogs)
  - ✓ SD作为助手，定期合并editlogs到fsimage。
- HDFS写入/读取数据流程。
  - ✓ 建立流水线
  - ✓ 写入数据
  - ✓ 关闭流水线
- HDFS shell命令。
- HDFS JavaApi。

问题

- 回收站：

hdfs默认是没有开启回收站功能的，需要配置中开启。fs.trash.interval。
当我们通过Hadoop fs –rm 命令将文件删除之后，文件会先移动到回收站中。再某个时间周期内如果想恢复文件，需要自己去回收站将文件移动出来。

- Hadoop fs –ls后面不跟目录，那么它默认找的路径则是：/user/用户名/

- Hadoop fs –appendToFile追加文件时报错。配置此参数：dfs.support.append=true

THANKS