Hadoop-HDFS

macOS下Hadoop安装配置

https://www.jianshu.com/p/3859f57aa545

Hadoop读取node数据过程

HDFS客户端通过DistributedFileSystem实例通过RPC调用namenode namenode返回datanode地址信息，确定文件起始块的物理地址 DistributedFileSystem生成FSDataInputStream实例以便读取数据 FSDataInputStream生成DFSInputStream对象，该对象用read方法与datanode交互 DFSInputStream调用close方法关闭与datanode的连接

写入文件过程

1，用户将数据交给客户端，并配置 将数据切分块的大小和写入的datanode数

2，客户端写入数据到DataNode

3，完成后发送信号到NameNode（存储DataNode地址信息和HDFS目录结构、元数据存储）

4，关闭连接

具体步骤：

DistributedFileSystem用create方法新建文件
RPC调用namenode 检查各种权限 检查通过，创建文件 没通过，抛出异常
返回FSDataOutputStream实例，以便写入数据
FSDataOutputStream生成DFSOutputStream对象，用write方法与datanode交互
DFSOutputStream将数据分成多个数据包，写入内部数据队列
DataStreamer选出合适的一组datanode，并要求namenode分配数据
这组datanode连接起来形成"管线"，每个node称为一个节点 DFSOutputStream维护了一个新的确认队列（ack queue）接受到datanode的确认信息之后，删除数据队列里的数据包
有datanode发生故障： 关闭管线 把所有数据包加回到数据队列 将正常datanode做新标示，并传回namenode
将数据分配给剩下的datanode，等待坏掉的datanode恢复 DFSOutputStream调用close方法关闭与datanode的连接

4，Hadoop的数据复制

因为Hadoop设计出来是为了存储数据在廉价机器上，所以硬件不可靠，需要备份确保容错

HDFS 提供了数据复制机制。HDFS 将每一个文件存储为一系列块，每个块由多个副本来保证容错，块的大小和复制因子可以自行配置（默认情况下，块大小是 128M，默认复制因子是 3）

复制数据时Hadoop尽量保证多台服务器数据存放尽可能均匀，在读取时优先选择距离读取器最近的副本，如果跨越多个服务器，优先选择本地，可以减少带宽消耗和读取时延

HDFS容错及检测

DataNode由于在很垃圾的硬盘上存数据，经常会出以下三种问题：

1，通信故障

2，数据错误

3，节点挂了

解决节点挂了 – 心跳机制：DataNode定期向NameNode发送心跳信号，如果没接受到信号，则将其标记为死亡，不会将数据存储或者副本保存在死亡Node上
解决数据错误 – 数据完整性报告：datanode上的数据可能损坏，避免取到坏了的数据，HDFS创建数据时计算“校验和”，并存储文件的“校验和在namenode上，从datanode读取数据时进行验证
如果不匹配则向namenode报告，将这部分datanode标记为死亡node
hadoop fs -checksum可以检查校验和
支持数据备份
元数据稳定：namenode支持多副本同步
解决网络通信故障 – 应答机制 客户端想读取数据时先向DataNode发送一个应答信号，如果DataNode不回，几次之后标记为死亡
写入数据时DataNode挂掉处理方式： 写数据的时候，DataNode接受完数据会向NameNode发送一个响应，确认收到数据，如果没发代表Node挂掉，NameNode会跳过这个DataNode，调整存放数据到其他Node

NameNode存了什么

单个大文件会分成多块（默认128MB一块），存在多个节点上（默认三个），节点的选择不

同版本Hadoop不同，三个节点都存了大文件的全部
例如：300MB的文件存在3个节点上，每个节点存3份（2份128MB，1份44MB），读取的时候，节点一的一号文件，节点二的二号文件，节点三的三号文件一起读取，速度加快且文件不易丢失（相当于多了两个备份），当某个节点挂了，YARN的RM会找新的DN

数据块列表： 数据块1:存在DN1，DN2，DN3 数据块2:存在DN1，DN5，DN6

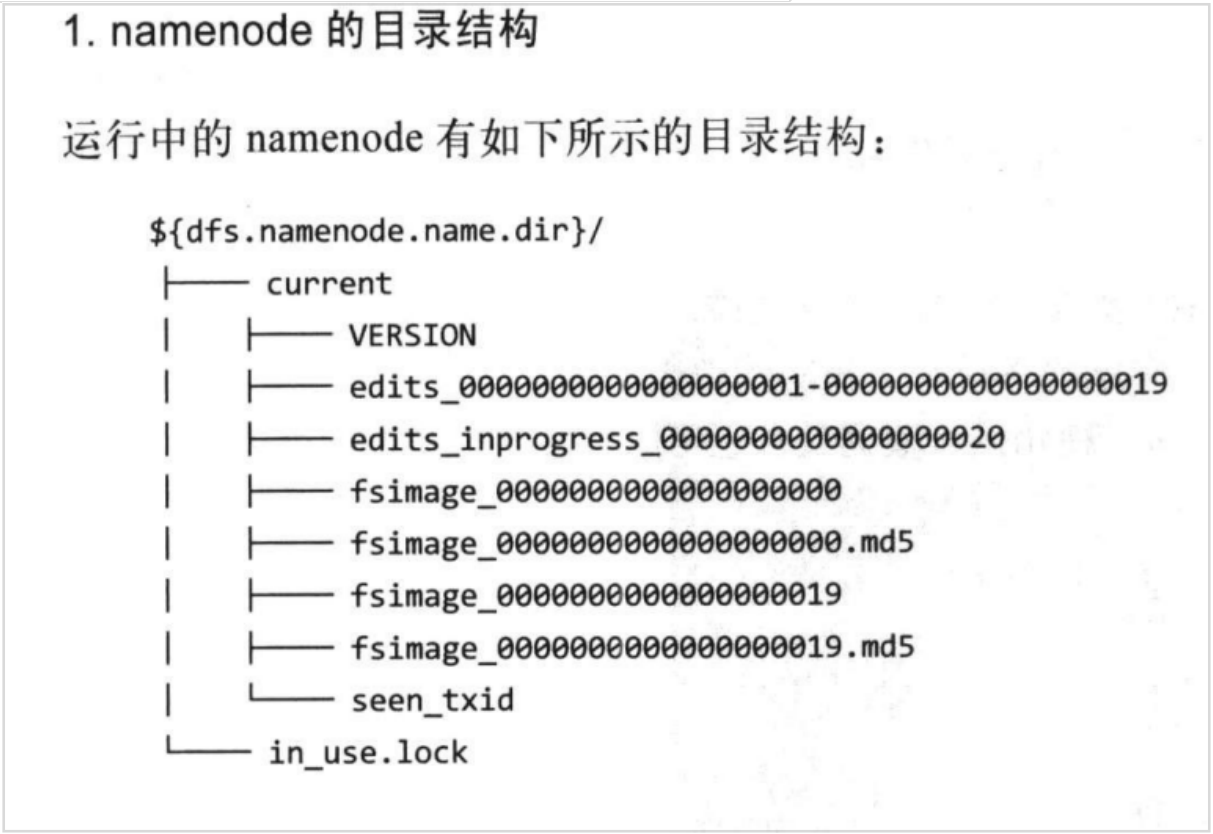
DataNode列表： DN1:存了 数据块1，数据块2 。。。 DN2: DN3:

NameNode定期检查数据块列表，看看每个数据块有没有被备份，如果没有指定DataNode互相备份

备份选择：优先选择不同机架上的相邻DataNode，如果找不到这样的，会随机选择

HDFS文件目录

```
# $HADOOP_HOME/hdfs/tmp下
# hadoop行时产生的数据也可以保存在tmp下，由core-site.xml控制
```



- VERSION版本相关信息

```
#Fri Aug 28 04:25:30 GMT 2020
namespaceID=2061912758
clusterID=CID-d2818055-3a99-4d90-996d-930d71212c93
cTime=0
storageType=NAME_NODE
blockpoolID=BP-1808700271-192.168.88.161-1598584786641
layoutVersion=-63
~
```

- storageType代表保存的是什么组件的数据
- clusterID代表NameNode和DataNode的ID，两者需要一致
- blockpoolID代表了NameNode下文件池（所有文件）的唯一ID

current下的所有文件

```
[root@initstudy current]# ls
edits_0000000000000000001-0000000000000000001  edits_0000000000000000012-0000000000000000013
edits_0000000000000000002-0000000000000000002  edits_0000000000000000014-0000000000000000015
edits_0000000000000000003-0000000000000000003  edits_inprogress_0000000000000000016
edits_0000000000000000004-0000000000000000004  fsimage_0000000000000000013
edits_0000000000000000005-0000000000000000005  fsimage_0000000000000000013.md5
edits_0000000000000000006-0000000000000000006  fsimage_0000000000000000015
edits_0000000000000000007-0000000000000000008  fsimage_0000000000000000015.md5
edits_0000000000000000009-0000000000000000010  seen_txid
edits_0000000000000000011-0000000000000000011  VERSION
```

edits_XXX代表多个日志文件的段，同一时间只有一个保持可写状态
Fsimage代表NameNode的检查点，包含所有文件的备份信息

Edit日志数据会无限增长，不利于NameNode重启更新，辅助NameNode帮助NameNode记录日志和检查点数据，方便迅速挂掉重启

SecondNameNode

- SNN请求NN停止使用当前edit文件，将新的操作更新到新edit文件,并更新seen_txid文件
- SNN获取NN最新的edit和fsimage文件，采用HTTP GET
- 将fsimage加载到内存，执行edit事务，并生成新的fsimage
- 将fsimage发回NN
- NN重新命名fsimage文件，便于以后使用

```
# 手动创立检查点：hdfs dfsadmin -saveNamespace
```

检查点设置

- 默认每隔一小时设置一次
- 如果未到一小时，但是累计了100万个事务，也会创建一次