**15Architecture\_05大数据+Hadoop安装与配置+HDFS**

**一 大数据**

**1.1 大数据的由来**

随着计算机技术的发展,互联网的普及,信息的积累已经到了一个非常庞大的地步,信息的增长也在不断的加快,随着互联网\物联网建设的加快,信息更是爆炸,收集\检索\统计这些信息越发困难,必须使用新的技术来解决这些问题

**1.2 什么是大数据**

1.2.1 定义

大数据指无法在一定时间范围内用常规软件工具进行捕捉\管理和处理的数据集合,需要新处理模式才能有更强的决策力\洞察发现力和流程优化能力的海量\高增长率和多样化的信息资产

**是指从各种各样类型的数据中,快速获得有价值的信息[核心能力]**

1.2.2 大数据能做什么

企组织利用相关数据分析帮助他们降低成本\提高效率\开发新产品\做出更明智的业务决策等

把数据集合并后进行分析得出的信息和数据关系性,用来察觉商业趋势\判定研究质量\避免疾病扩散\打击犯罪或测定即时交通路况等

大规模并行处理数据库,数据挖掘电网,分布式文件系统或数据库,云计算平和可扩展的存储系统等

**1.3 大数据特性**



5V特性

Volume(大体量):可从数百TB到数十数百PB,甚至EB的规模

Variety[vəˈraɪəti](多样性):大数据包括各种格式和形态的数据

Velocity[vəˈlɒsəti](时效性):很多大数据需要在一定的时间限度下得到及时处理

Veracity[vəˈræsəti](准确性):处理的结果要保证一定的准确性

Value(大价值):大数据包含很多深度的价值,大数据分析挖掘和利用将带来巨大的商业价值

**1.4 大数据与Hadoop**

Hadoop是什么

是一种分析和处理海量数据的软件平台

是一款开源软件,使用Java开发

可以提供一个分布式基础架构

Hadoop特点: 高可靠性,高扩展性,高效性,高容错性,低成本

**二 Hadoop起源**

2003年开始Google陆续发表了3篇论文:GFS,MapReduce,BigTable

GFS:

是一个可扩展的分布式文件系统,用于大型的\分布式的\对大量数据进行访问的应用

可以运行于廉价的普通硬件上,提供容错功能

MapReduce:

MapReduce是针对分布式并行计算的一套编程模型,由map和reduce组成,map是映射,把指令分发到多个worker上,reduce是规约,把worker计算出的结果合并

BigTable:

BigTable是存储结构化数据库

BigTable建立在GFS,scheduler,lock service和mapreduce之上

每个table都是一个多维的稀疏图

GFS,mapreduce和big table三大技术被称为Google的三驾马车,虽然没有公布源码,但发布了这三个产品详细设计论

Yahoo资助的Hadoop,是按照这三篇论文的开源Java实现的,但在性能上Hadoop比Google要差很多

GFS---->HDFS

MapReduce---->MapReduce

Big Table----> Hbase

**三 Hadoop组件**

**3.1 Hadoop常用组件**

**HDFS:Hadoop分布式文件系统(核心组件)**

**MapReduce:分布式计算框架(核心组件)**

**Yarn:集群资源管理系统(核心组件)**

Zookeeper:分布式协作服务

Hbase:分布式列存储数据库

Hive:基于Hadoop的数据仓库

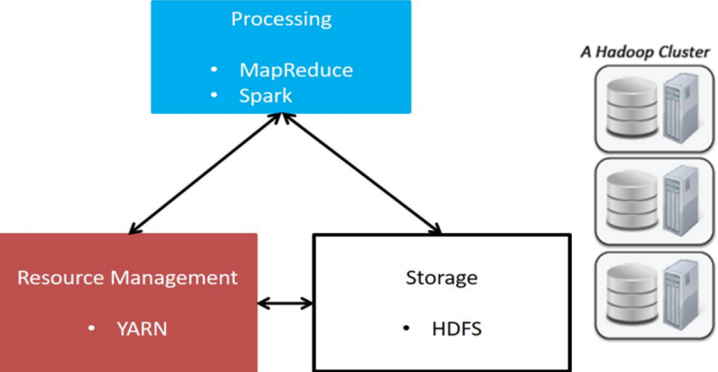
Sqoop:数据同步工具

Pig:基于Hadoop的数据流系统

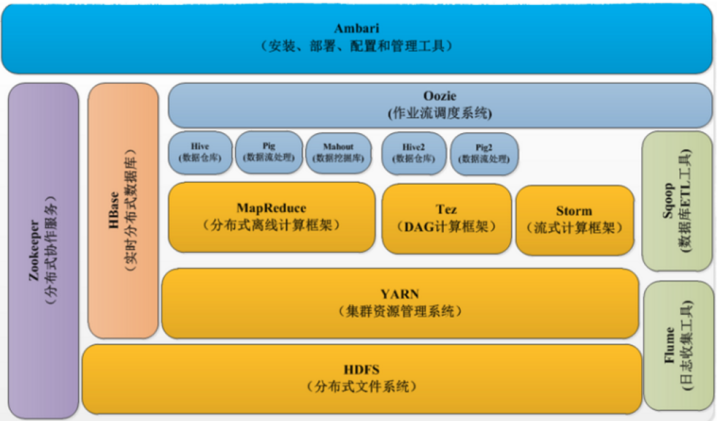
Mahout:数据挖掘算法库

Flume:日志收集工具

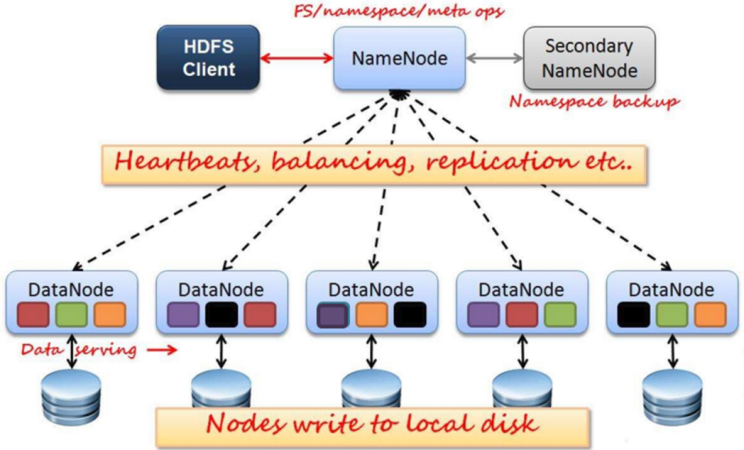
**3.2 Hadoop核心组件**



**3.3 Hadoop生态系统**



**3.4 HDFS结构**



**3.5 HDFS角色及概念**

Hadoop体系中数据存储管理的基础,是一个高度容错的系统,用于在低成本的通用硬件上运行

角色和概念

client namenode secondarynode datanode

**3.5.1 namenode:**

master节点,管理HDFS的名称空间和数据块映射信息,配置副本策略,处理所有客户端请求

**3.5.2 secondary namenode:**

定期合并fsimage和fsedits,推送给namenode

紧急情况像,可辅助恢复namenode

不是namenode的热备

**3.5.3 datanode:**

数据存储节点,存储实际的数据

汇报存储信息给namenode

**3.5.4 client:**

切分文件;访问HDFS;

与namenode交互,获取文件位置信息

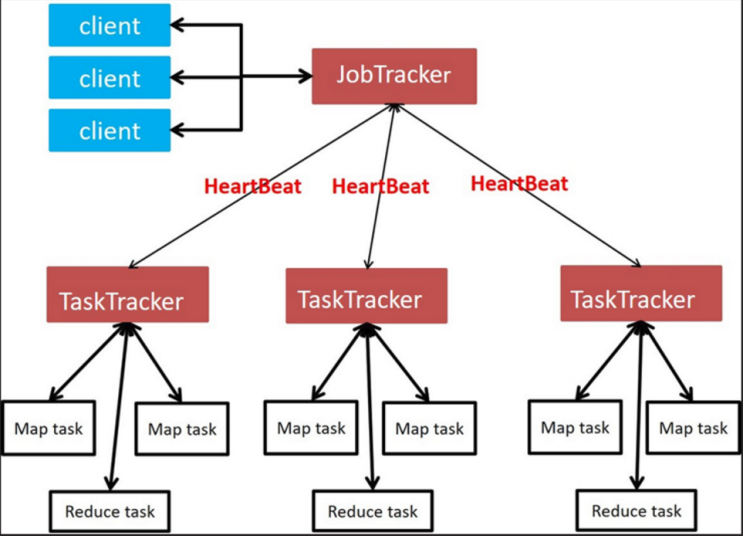
与datanode交互,读取和写入数据

**3.5.5 block:**

每块大小缺省128M

每块可以多个副本

**3.6 MapReduce结构**



**3.7 MapReduce结构及概念**

源自于Google的mapreduce论文,Java实现的分布式计算框架

角色和概念: job tracker task tracker map task reducer task

**3.7.1 jobtracker**

master节点只有一个

管理所有作业/任务的监控\错误处理等

将任务分解成一系列任务,并分派给tasktracker

**3.7.2 tasktracker**

slave节点,一般是多台

运行map task和reduce task

并与jobtracker交互,汇报任务状态

**3.7.3 map task**

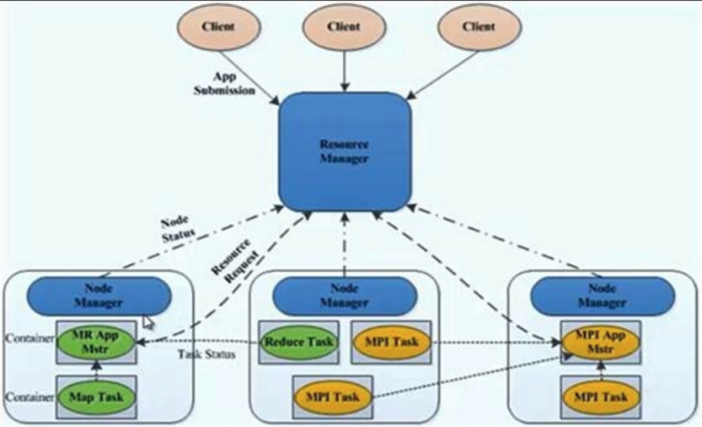
解析每条数据记录,传递给用户编写map()并执行,将输出结果写入本地磁盘

如果为map-only作业,直接写入HDFS

**3.7.4 reduce task**

从map task的执行结果中,远程读取输入数据,对数据进行排序,将数据按照分组传递给用户编写的reduce函数执行

**3.8 Yarn结构**



**3.9 Yarn角色及概念**

yarn是Hadoop的一个通用的资源管理系统

**3.9.1 yarn角色:**

resourcemanager

nodemanager

applicationmaster

container

client

**3.9.2 resourcemanager**

处理客户端请求

启动/监控applicationmaster

监控nodemanager

资源分配与调度

**3.9.3 nodemanager**

单个节点上的资源管理

处理来自resourcemanager的命令

处理来自applicationmaster的命令

**3.9.4 container**

对任务运行环境的抽象,封装了CPU\内存等

多维资源以及环境变量\启动命令等任务运行相关信息资源分配与调度

**3.9.5 applicationmaster**

数据切分

为应用程序申请资源,并分配给内部任务

任务监控与容错

**3.9.6 client**

用户与yarn交互的客户端程序

提交应用程序\监控应用程序状态,杀死应用程序等

**3.9.7 yarn的核心思想**

将jobtracker和tasktacker进行分离,它由下面几大构成组件

resourcemanager一个全局的资源管理器

nodemanager每个节点(RM)代理

applicationmaster表示每个应用

每一个applicationmaster有多个container在nodemanager上运行

**四 Hadoop安装与配置**

**4.1 Hadoop模式**

Hadoop的部署模式有三种: 单机 伪分布式 完全分布式

**4.2 单机模式**

room9pc01 ~]# base-vm nn01 192.168.1.60

nn01 ~]# yum -y install java-1.8.0-openjdk java-1.8.0-openjdk-devel

room9pc01 ~]# scp /linux-soft/04/hadoop/hadoop-2.7.7.tar.gz

[root@192.168.1.60:/root](mailto:root@192.168.1.60:/root)

nn01 ~]# tar -xf hadoop-2.7.7.tar.gz

nn01 ~]# mv hadoop-2.7.7 /usr/local/hadoop

nn01 ~]# cd /usr/local/hadoop/

nn01 hadoop]# ./bin/hadoop #执行时报错

nn01 hadoop]# cd /usr/local/hadoop/etc/hadoop/

nn01 hadoop]# vim hadoop-env.sh

25 exportJAVA\_HOME=

"/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-2.b14.el7.x86\_64/jre"

33 export HADOOP\_CONF\_DIR="/usr/local/hadoop/etc/hadoop"

nn01 hadoop]# ./bin/hadoop version #执行成功

nn01 hadoop]# mkdir input

nn01 hadoop]# cp /etc/passwd ./input/

nn01 hadoop]# ./bin/hadoop jar

share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.7.jar wordcount input output

nn01 hadoop]# cat output/part-r-00000

**4.3 伪分布式**

伪分布式的安装和完全分布式类似,区别是所有角色安装在一台机器上,使用本地磁盘,一般生产环境都会使用完全分布式,伪分布式一般是用来学习和测试

伪分布式的配置和完全分布式配置类似

**4.4 Hadoop配置文件及格式,用于5.4及之后的内容**

文件格式

hadoop-env.sh

JAVA\_HMOE

HADOOP\_CONF\_DIR

xml文件配置格式

<property>

<name>关键字</name>

<value>变量值</value>

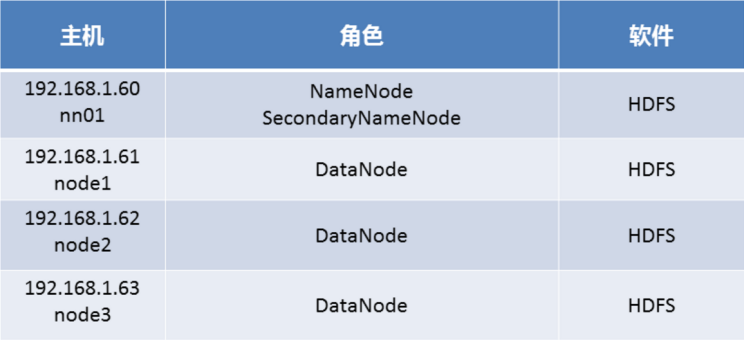
<description>描述</description>

</property>

**五 HDFS分布式文件系统**

**5.1 完全分布式**

系统规划



**5.2 搭建完全分布式**

room9pc01 ~]# base-vm node61\62\63

禁用selinux和firewalld

node61\62\63 ~]# yum -y install

java-1.8.0-openjdk java-1.8.0-openjdk-devel

java安装验证

node61\62\63 ~]# java -version

node61\62\63 ~]# jps

nn01 ~]# vim /etc/hosts

192.168.1.60 nn01

192.168.1.61 node61

192.168.1.62 node62

192.168.1.63 node63

nn01 ~]# scp /etc/hosts [root@192.168.1.61\62\63:/etc/hosts](mailto:root@192.168.1.61/62/63:/etc/hosts)

配置SSH信任关系,要求不能出现输入yes的情况,每台机器都要能登录成功,包括本机

nn01 ~]# vim /etc/ssh/ssh\_config

35 # StrictHostKeyChecking ask 复制此行,粘贴到58行下,并修改

58 Host \*

59 **StrictHostKeyChecking no**

60 GSSAPIAuthentication yes

61 StrictHostKeyChecking no

nn01 ~]# ssh-keygen -f /root/.ssh/id\_rsa -N ''

nn01 ~]# ssh-copy-id 192.168.1.60\61\62\63

**5.2 HDFS完全分布式系统配置**

环境配置文件: hadoop-env.sh

核心配置文件: core-site.xml

HDFS配置文件: hdfs-site.xml

节点配置文件: slaves

官方文档:http://hadoop.apache.org/docs/

找到r2.7.7,点击进入,页面左边拉倒最下面,组态中,有模板:

core-default.xml

hdfs-default.xml

yarn-default.xml

文档中有name value description三列,对应4.4格式的xml文件配置格式中的关键字 变量值 描述

**5.3 环境配置文件: hadoop-env.sh**

nn01 ~]# vim **/usr/local/hadoop/etc/hadoop/hadoop-env.sh**

25 export JAVA\_HOME=

"/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-2.b14.el7.x86\_64/jre"

#openjdk的安装目录(rpm -ql java-1.8.0-openjdk查询获得)

33 export HADOOP\_CONF\_DIR="/usr/local/hadoop/etc/hadoop"

#hadoop配置文件的存放目录

**5.4 核心配置文件: core-site.xml**

fs.defaultFS: 定义默认文件系统

hadoop.tmp.dir: 定义核心数据目录

nn01 ~]# vim **/usr/local/hadoop/etc/hadoop/core-site.xml**

19 <configuration>

20 <property>

21 <name>fs.defaultFS</name>

22 <value>hdfs://nn01:9000</value>

23 </property>

24 <property>

25 <name>hadoop.tmp.dir</name>

26 <value>/var/hadoop</value>

27 </property>

28 </configuration>

**5.5 HDFS配置文件hdfs-site.xml**

namenode: 地址声明: dfs.namenode.http-address

secondarynamenode: 地址声明: dfs.namenode.secondary.http-address

dfs.replication:文件冗余份数:备份多少份

nn01 ~]# vim **/usr/local/hadoop/etc/hadoop/hdfs-site.xml**

19 <configuration>

20 <property>

21 <name>dfs.namenode.http-address</name>

22 <value>nn01:50070</value>

23 </property>

24 <property>

25 <name>dfs.namenode.secondary.http-address</name>

26 <value>nn01:50090</value>

27 </property>

28 <property>

29 <name>dfs.replication</name>

30 <value>2</value>

31 </property>

32 </configuration>

**5.6 节点配置文件slaves**

只写datanode节点的主机名称

nn01 ~]# vim /usr/local/hadoop/etc/hadoop/slaves

node61

node62

node63

**5.7 同步配置**

hadoop所有节点的配置参数完全一样,在一台配置好后,把配置文件同步到其他主机

nn01 ~]# scp -r /usr/local/hadoop/

[root@192.168.1.61\62\63:/usr/local/hadoop/](mailto:root@192.168.1.61/62/63:/usr/local/hadoop/)

**5.8 HDFS完全分布式配置**

**5.8.1 在本机创建/var/hadoop核心数据目录,见5.4**

nn01 ~]# mkdir /var/hadoop

**5.8.2 在namenode上执行格式化操作**

nn01 ~]# cd /usr/local/hadoop/

nn01 hadoop]# ./bin/hdfs namenode -format

每行都有INFO

SHUTDOWN\_MSG: Shutting down NameNode at nn01/192.168.1.60

**5.8.3 启动集群**

nn01 hadoop]# ./sbin/start-dfs.sh

**5.8.4 JPS验证角色**

**namenode验证**

nn01 hadoop]# jps

30531 SecondaryNameNode

30340 NameNode

30653 Jps

**datanode验证**

node61 ~]# jps

23410 DataNode

23492 Jps

node62 ~]# jps

23412 Jps

23339 DataNode

node63 ~]# jps

23332 DataNode

23405 Jps

**节点验证**

namenode上

nn01 hadoop]# ./bin/hdfs dfsadmin -report

Live datanodes (3):

Name: 192.168.1.63:50010 (node63)

...

Name: 192.168.1.61:50010 (node61)

...

Name: 192.168.1.62:50010 (node62)

排错:查看/usr/local/hadoop/logs/目录内的out日志