云计算期末项目

1451907 胡炜

1552704 杨灏诚

1552711 徐哲涵

1552713 陈铠勋

目录

[项目简介 2](#_Toc503445072)

[搭建过程 2](#_Toc503445073)

[1、搭建hadoop 2](#_Toc503445074)

[2、利用zookeeper搭建一个HA架构。 3](#_Toc503445075)

[具体操作过程 4](#_Toc503445076)

[配置zoo.cfg 4](#_Toc503445077)

[配置hadoop相关文件 5](#_Toc503445078)

[启动zookeeper集群 8](#_Toc503445079)

[启动journalnode集群 9](#_Toc503445080)

[格式化zkfc,让在zookeeper中生成ha节点 9](#_Toc503445081)

[格式化hdfs 9](#_Toc503445082)

[启动namenode 9](#_Toc503445083)

[把Namenode的数据同步到slave2上 10](#_Toc503445084)

[启动mast2上的namenode作为standby 10](#_Toc503445085)

[启动datanode 10](#_Toc503445086)

[启动yarn 10](#_Toc503445087)

[启动ZKFC 10](#_Toc503445088)

[通过jps查询各个机器的进程。 11](#_Toc503445089)

[测试HA的高可用性 11](#_Toc503445090)

[安装hive并且将数据导入hive 12](#_Toc503445091)

[添加前端界面 13](#_Toc503445092)

[分析与结论 13](#_Toc503445093)

# 项目简介

本项目通过搭建hadoop，利用zookeeper和hive，实现了一个高可用的有关销售系统销售模块的查询分析系统。

# 搭建过程

## 1、搭建hadoop

Hadoop简介：Hadoop是一个能够对大量数据进行分布式处理的软件框架，以一种可靠、高效、可伸缩的方式进行数据处理，实现了一个分布式文件系统（HDFS）。HDFS有高容错性的特点，并且设计用来部署在低廉的硬件上；而且它提供高吞吐量来访问应用程序的数据，适合那些有着超大数据集的应用程序。HDFS放宽了POSIX的要求，可以以流的形式访问文件系统中的数据。

Hadoop的框架最核心的设计就是：HDFS和MapReduce。HDFS为海量的数据提供了存储，则MapReduce为海量的数据提供了计算。

特点：可靠、高效、可伸缩、高容错、低成本。

我们搭建的hadoop是基于3台物理机（master、slave1、slave2）组成的真正意义上的分布式系统。其中我们的master即是namenode也是datanode。它既可以控制各个节点同时自身也存储数据。

## 2、利用zookeeper搭建一个HA架构。

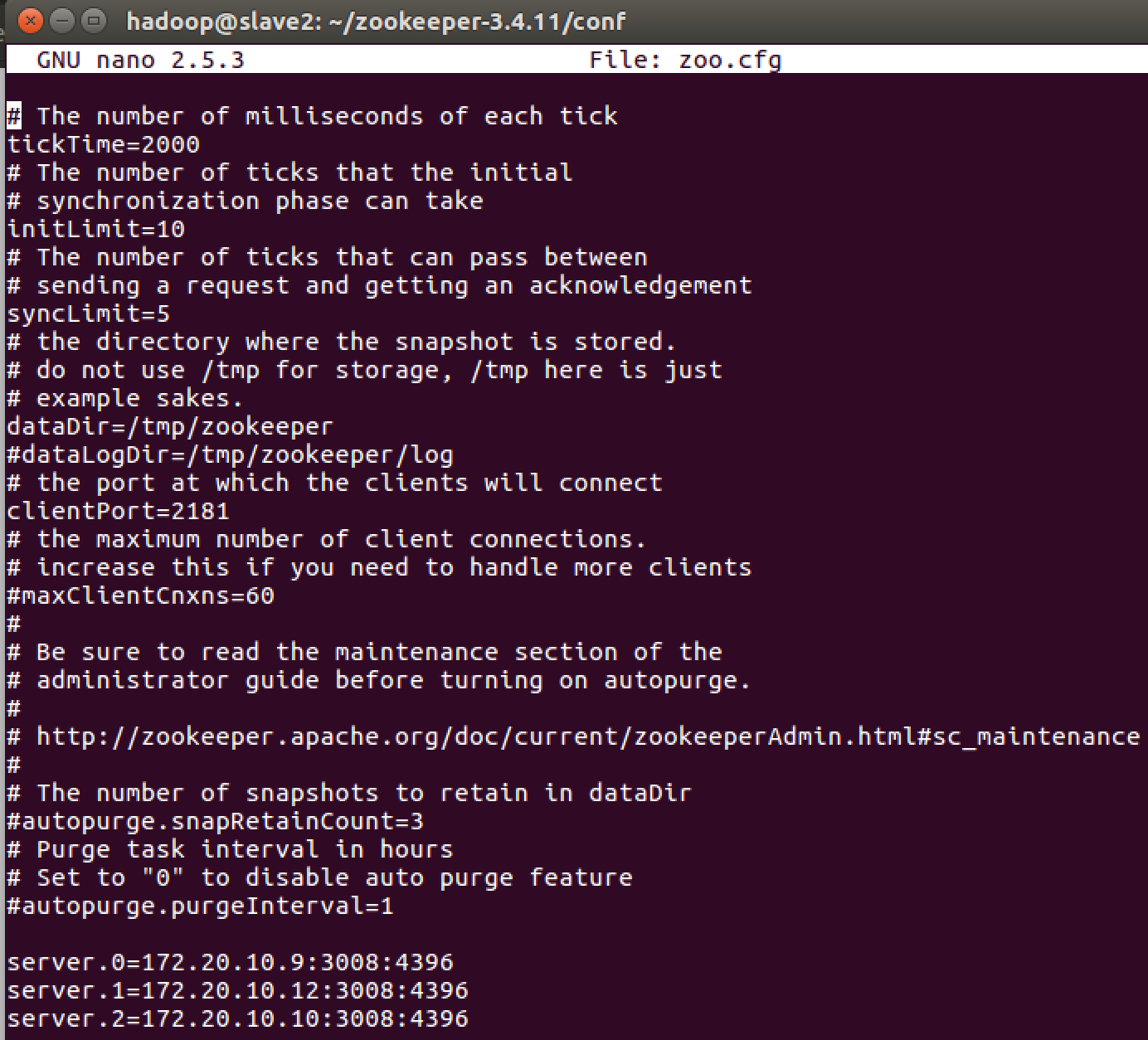
在上一部搭建hadoop之后，我们发现只有单个namenode的架构方式其可靠性不是特别强，虽然已经可以通过备份到多个datanode的方法来处理当某个datanode的物理机宕机的时候仍然可以让文件不丢失，但是如果namenode一但宕机，则整个系统便无法工作了，所以为了处理以上情况，我们需要引入一台新的namenode，在平时处于待机状态，当主namenode宕机时，它可以代替此namenode继续工作，这样的系统是高可用的。为此我们需要使用zookeeper。

Zookeeper简介：Zookeeper 是为分布式应用程序提供高性能协调服务的工具集合，也是Google的Chubby一个开源的实现，是Hadoop 的分布式协调服务。Zookeeper可以用来保证数据在ZK集群之间的数据的事务性一致。Zookeeper作为Hadoop项目中的一个子项目,是 Hadoop集群管理的一个必不可少的模块,它主要用来解决分布式应用中经常遇到的数据管理问题，如集群管理、统一命名服务、分布式配置管理、分布式消息队列、分布式锁、分布式协调等。在Hadoop中，它管理Hadoop集群中的NameNode，还有在Hbase中Master Election、Server 之间状态同状步等。

我们在之前搭建的hadoop中，给slave2加入了一个namenode，当master宕机，此时slave2可以接替master成为namenode。

# 具体操作过程

## 配置zoo.cfg



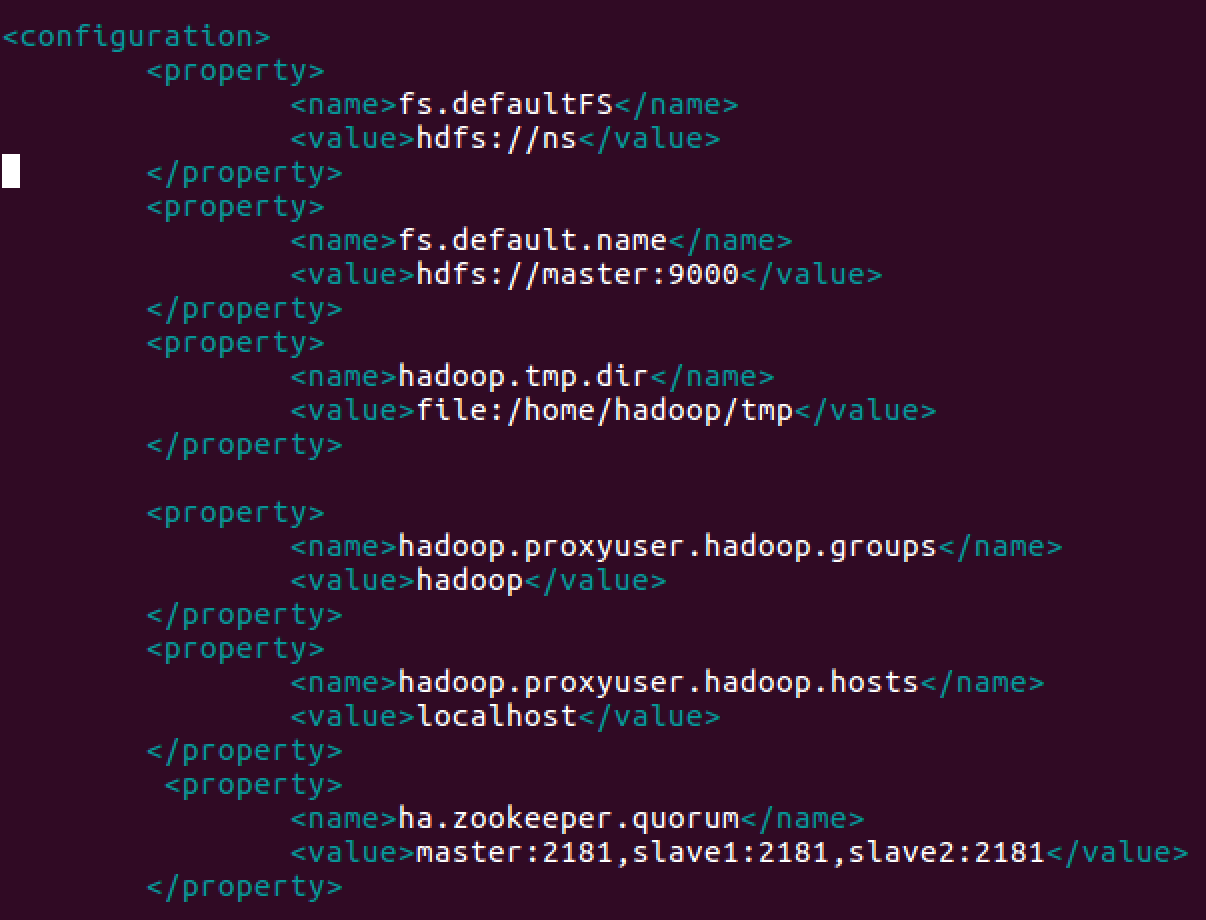
## 配置hadoop相关文件

core-site.xml

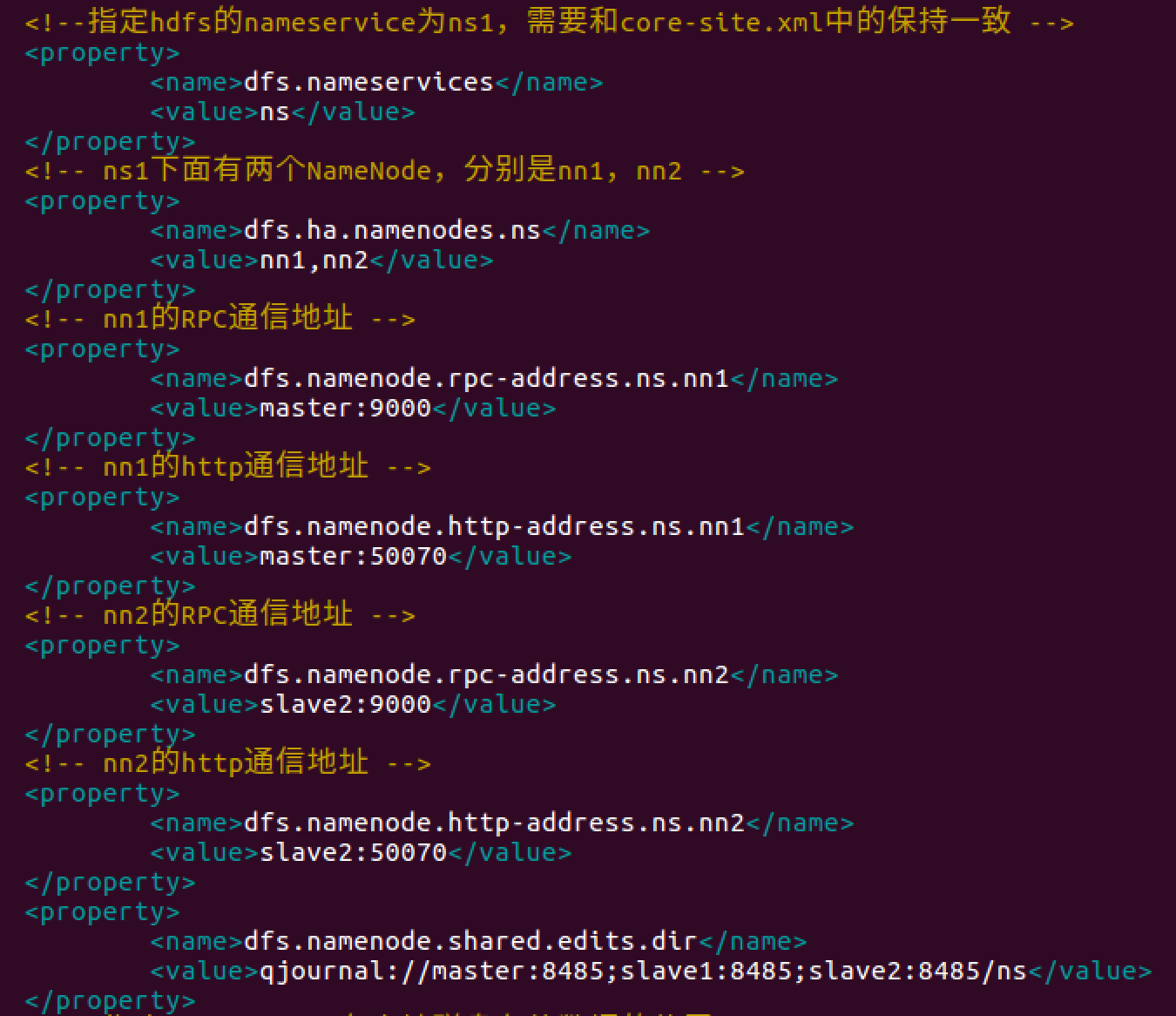
指定hdfs的nameservice为ns

指定hadoop数据临时存放目录

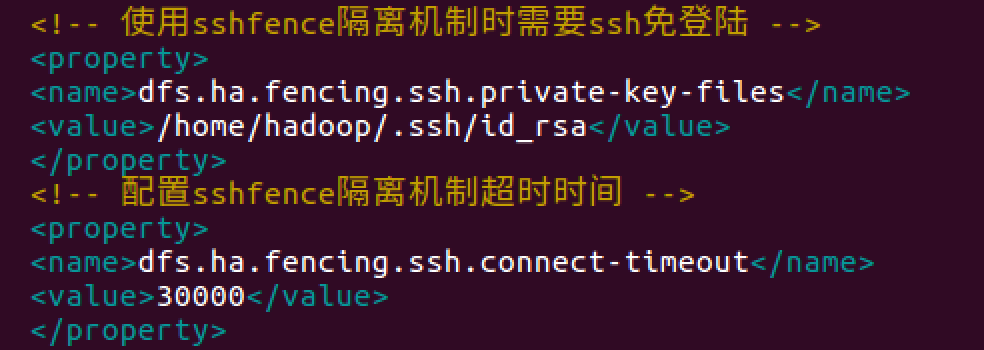
指定zookeeper地址



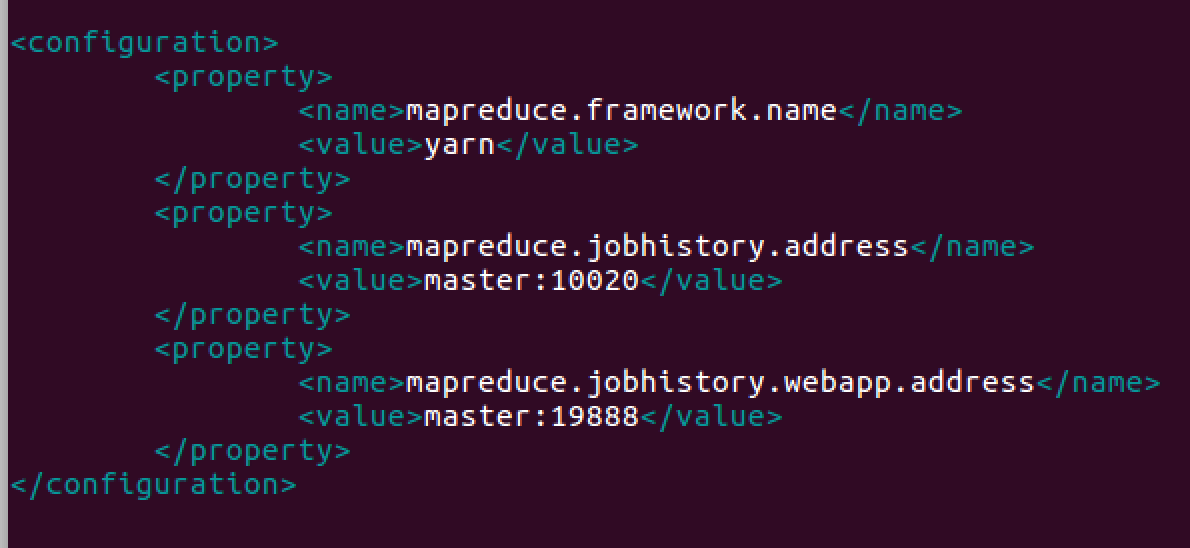
hdfs-site.xml







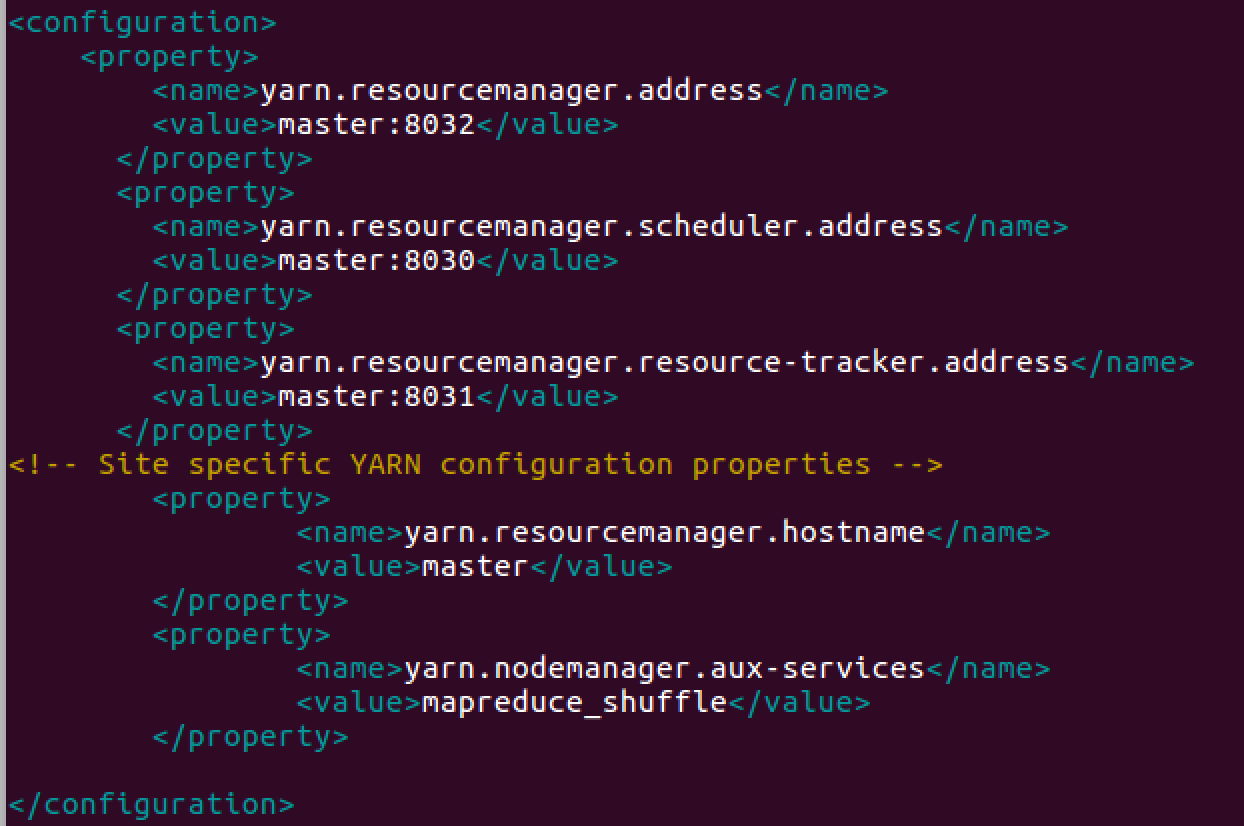
mapred-site.xml



yarn-site.xml

指定nodemanager启动时加载server的方式为shuffle serve

指定resourcemanager地址



在相关配置文件配置完成后，将配置复制到slave1,slave2。至此，Hadoop的配置就完成了。

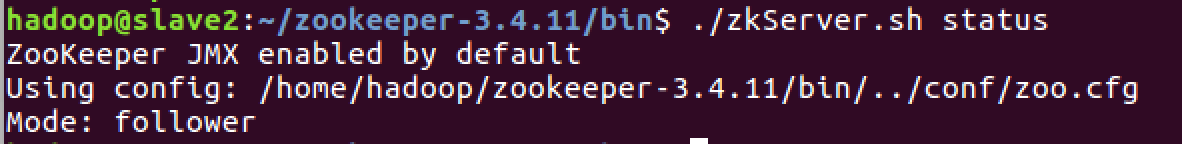
## 启动zookeeper集群

分别在master，slave1,slave2上执行命令启动zookeeper集群。

 sh zkServer.sh start

验证集群zookeeper集群是否启动，分别在mast、slave1、slave2上执行如下命令验证zookeeper集群是否启动，集群启动成功，有两个follower节点跟一个leader节点

sh zkServer.sh status



## 启动journalnode集群

sbin/hadoop-daemons.sh start journalnode

## 格式化zkfc,让在zookeeper中生成ha节点

在master上执行如下命令，完成格式化

hdfs zkfc –formatZK

## 格式化hdfs

hadoop namenode –format

## 启动namenode

sbin/hadoop-daemon.sh start namenode

## 把Namenode的数据同步到slave2上

在slave2中操作：

hdfs namenode –bootstrapStandby

## 启动mast2上的namenode作为standby

sbin/hadoop-daemon.sh start namenode

## 启动datanode

sbin/hadoop-daemons.sh start datanode

## 启动yarn

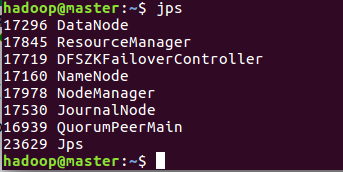
sbin/start-yarn.sh

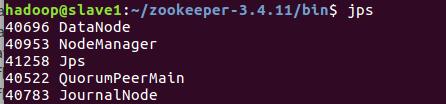
## 启动ZKFC

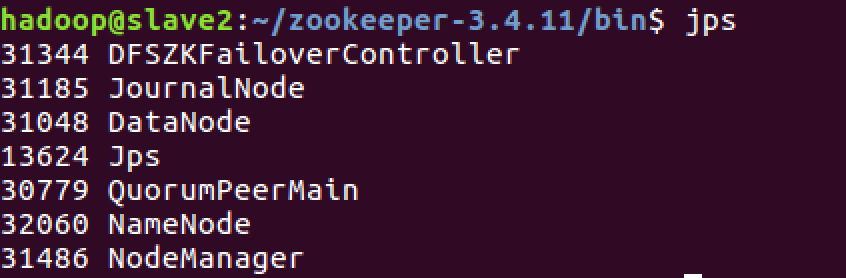
sbin/hadoop-daemons.sh start zkfc

至此所有便启动了Hadoop的HA。

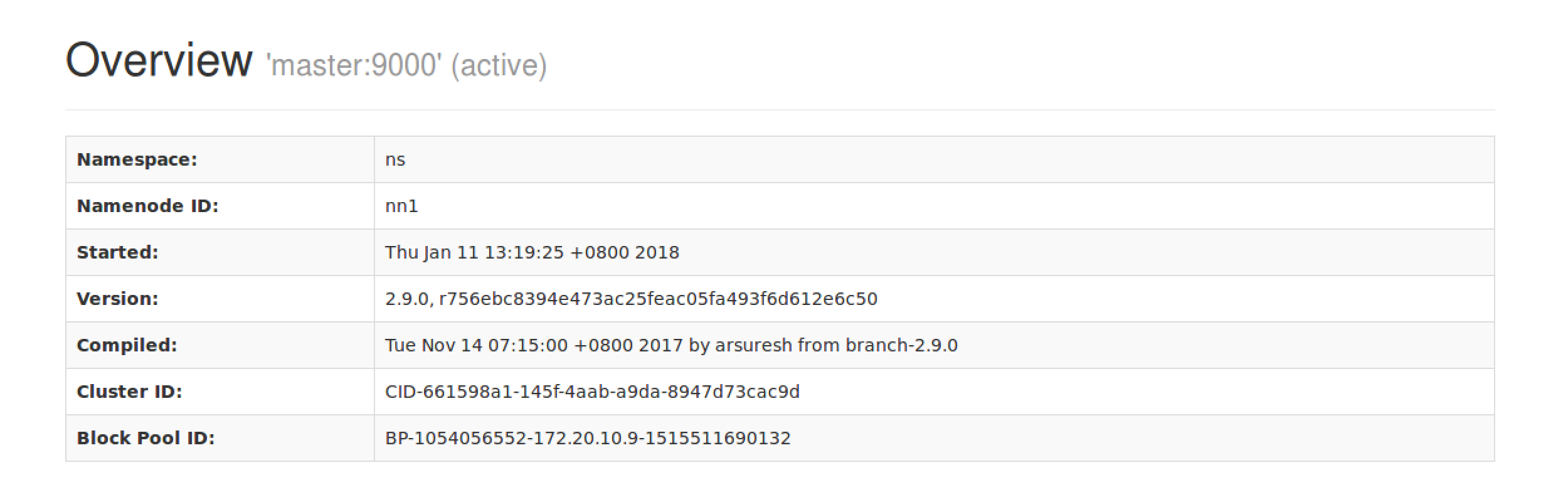
## 通过jps查询各个机器的进程。

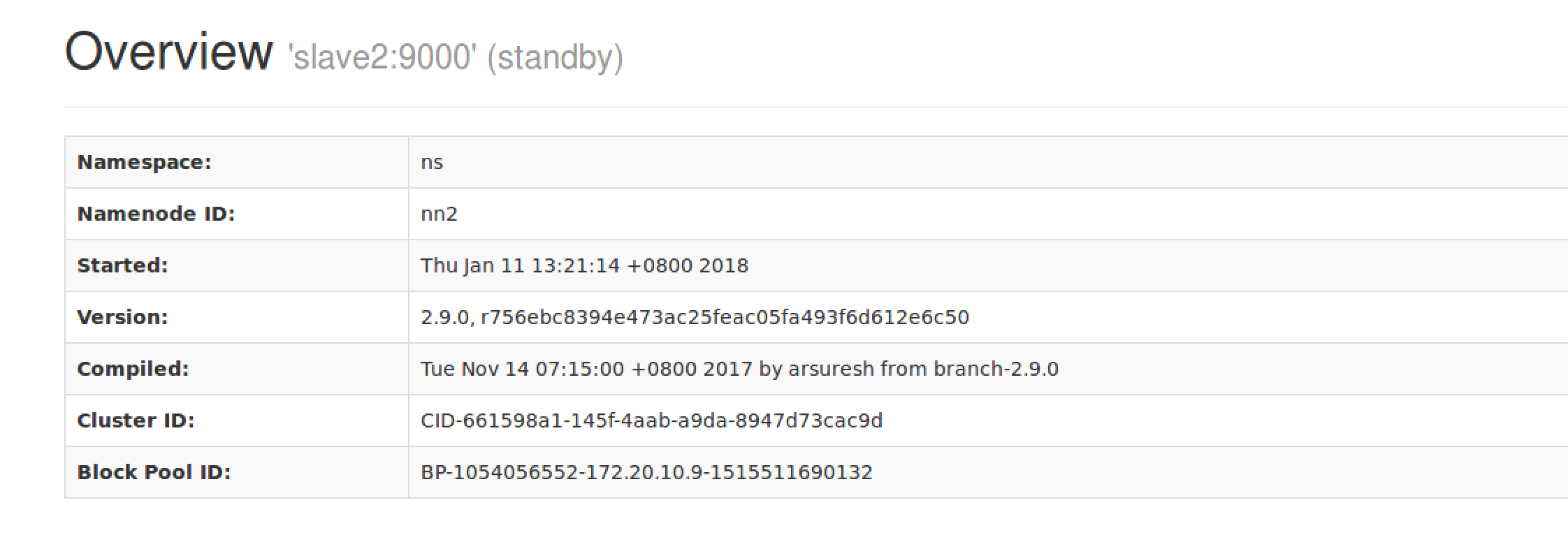






## 测试HA的高可用性





可以看到有两个namenode, master的namenode是active 状态。

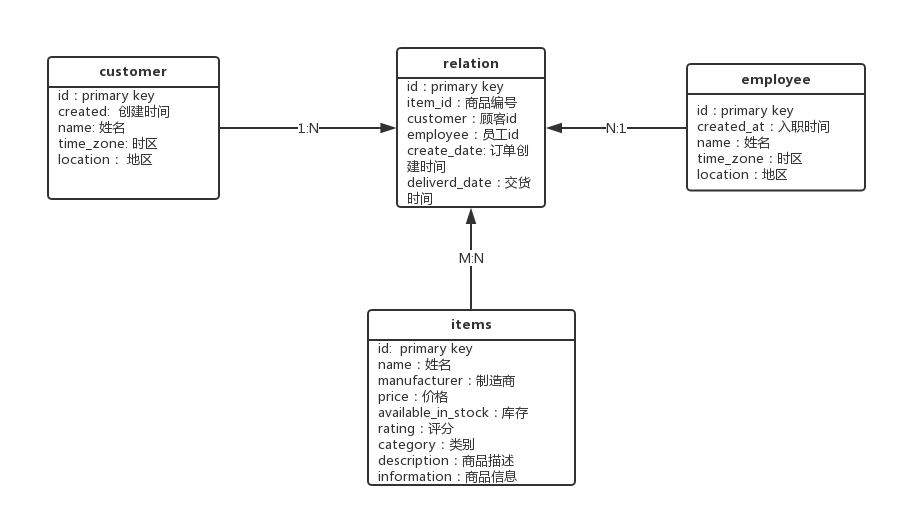
我们断掉master的网络连接或者在master中执行 sbin/hadoop-daemon.sh stop namenode

可以发现slave2 的namenode自动切换为active。

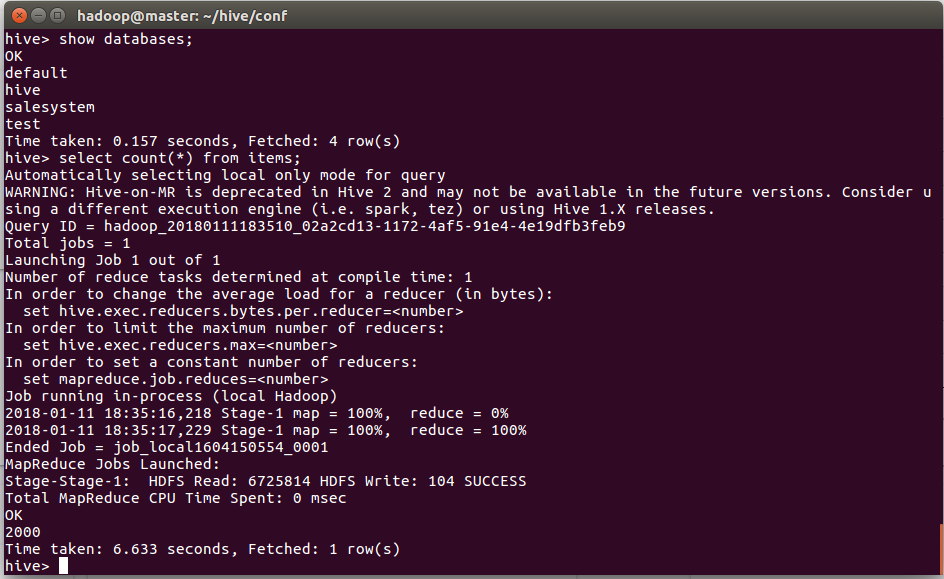


# 安装hive并且将数据导入hive

我们的数据从kaggle中获取，并且通过js脚本进行了ETL预处理。自行设计了数据库，总共建立了4章表：用户、订单、员工、商品。建立的数据库ER图如下所示：



执行搜索的截图：



# 添加前端界面

为了增强用户体验，我们还使用angular框架绘制了简单的用户界面，简化了用户使用该系统的操作。



# 分析与结论

在整个项目过程中，我们使用了两种数据存储形式：基于hive的数据库和本地的关系型数据库（postgresql）来分别存储整个销售系统的数据。通过比较两种数据库执行query语句的时间的出了以下的结论：

通过比较，我们发现了在本地的环境下，传统的数据库如Postgressql,有明显的性能和环境上的优势。但当数据量比较大时，单机数据库的性能会下降，而通过分布式的环境，则可以提高效率。