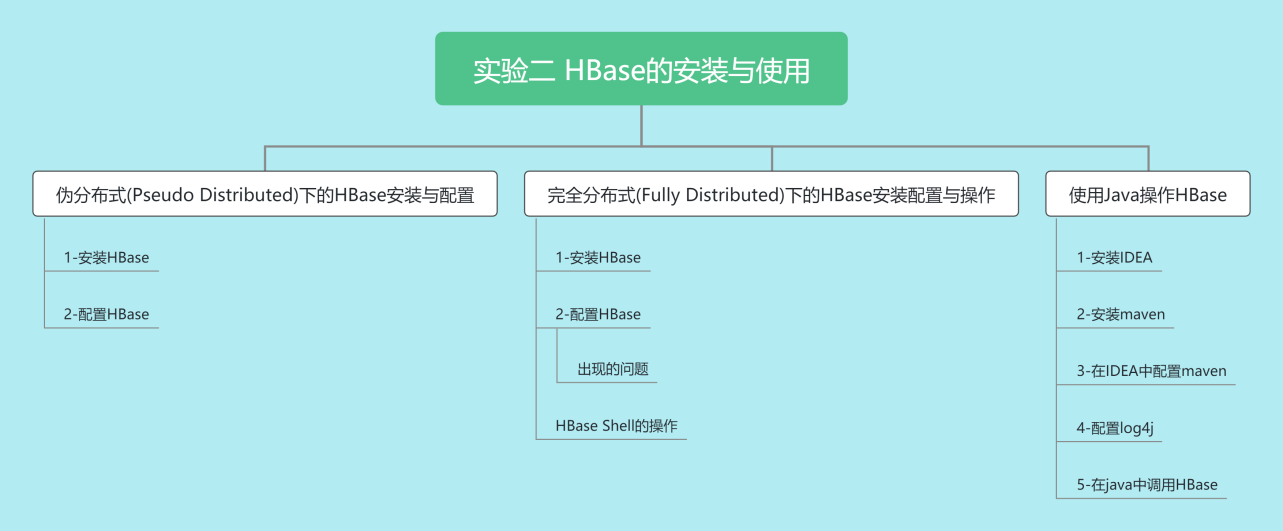
# **实验二** HBase的安装与使用

**组号：24**

**组员：**

**思维导图：**



**大纲（可点击快速跳转）：**

**[实验二](#实验二-hbase的安装与使用)** [HBase的安装与使用](#实验二-hbase的安装与使用)  
 [伪分布式(Pseudo Distributed)下的HBase安装与配置](#伪分布式pseudo-distributed下的hbase安装与配置)  
 [1-安装HBase](#X9062c8090972e51a92d95d4a7074e6a3cec903d)  
 [2-配置HBase](#X07c8169c3ab0a8329bc6933b98578349a562bbd)  
 [完全分布式(Fully Distributed)下的HBase安装配置与操作](#完全分布式fully-distributed下的hbase安装配置与操作)  
 [1-安装HBase](#X988ba03725bf33389736dead97d38e1cba1d0eb)  
 [2-配置HBase](#Xe7f5c3899ac3544990bf34535e44c1e32f0c72c)  
 [HBase Shell的操作](#hbase-shell的操作)  
 [使用Java操作HBase](#使用java操作hbase)  
 [1-安装IDEA](#X464aad53c82a70a80c94b80162dd69fdb997aa1)  
 [2-安装maven](#Xe2029a430d446aad2b7cf7cc087ed8eaf6ac436)  
 [3-在IDEA中配置maven](#X136951df47bac5400a3f46aeeb00ed98687cd3a)  
 [4-配置log4j](#X3a7a9b2c9fa82e2f6e91dae79cd591b83824911)  
 [5-在java中调用HBase](#Xb7b25f0fda5339a7658df7cdc5a1044e7f9a978)  
 [Build Success](#build-success)

## 伪分布式(Pseudo Distributed)下的HBase安装与配置

实验环境如下：

Ubuntu 22.04 虚拟机  
JDK 1.8  
Hadoop 2.10.1  
HBase 2.4.5

### 1-安装HBase

与hadoop2.10.1对应的HBase版本之一是2.4.5 。从<https://archive.apache.org/dist/hbase/2.4.5/hbase-2.4.5-bin.tar.gz>下载对应的版本

截图

下载相关文件，并将其放在想安装的位置。为了方便，选择与上一份实验手册相同的安装位置： /opt/hbase 。

tar xzf hbase-2.4.5-bin.tar.gz  
sudo mv ./hbase-2.4.5 /opt/hbase

**截图：**

### 2-配置HBase

为了方便操作，为HBase配置环境变量：

vim /etc/profile

在profile末尾添加以下两行配置：

export HBASE\_HOME=/opt/hbase  
export PATH=$PATH:$HBASE\_HOME/bin

**截图：**

更新环境变量：

source /etc/profile

**截图：**

修改配置文件 /opt/hbase/conf/hbase-site.xml ：

截图：

注意， hbase.rootdir 属性的值应为 hdfs://localhost:9000/hbase ； dfs.replication 可以设置为1( 非必须)；因为是伪分布式，所以 hbase.cluster.distributed 可以为 true 。

编辑hbase-env.sh文件，添加以下配置，来配置区域服务器的路径，并启动自动管理zookeeper：

**截图：**

启动hbase进行测试

start-all.sh  
start-hbase.sh  
hbase shell

截图：

运行 jps 指令，如果出现 HQuorumPeer ， HRegionServer 和 HMaster ，则证明启动成功：

截图：

## 完全分布式(Fully Distributed)下的HBase安装配置与操作

### 1-安装HBase

同伪分布，多两台slave

### 2-配置HBase

编辑/etc/profile同伪分布。

HBase默认写到 /${java.io.tmpdir}/HBase-${user.name} 。$ {.io.tmpdir}通常映射到 /tmp 目录下。可以修改配置文件/opt/hbase/conf/hbase-site.xml 以规定其映射目录；同时需要修改配置文件来指定内置zookeeper的运行文件保存位置：

**截图：**

<property>
  
 <name>hbase.rootdir</name>
  
 <value>hdfs://M2020212185:9000/hbase</value>
  
</property>
  
<property>
  
 <name>hbase.cluster.distributed</name>
  
 <value>true</value>
  
</property>
  
<property>
  
 <name>hbase.zookeeper.quorum</name>
  
 <value>localhost</value>
  
</property>
  
<property>
  
 <name>dfs.replication</name>
  
 <value>3</value>
  
</property>
  
<property>
  
 <name>hbase.zookeeper.property.clientPort</name>
  
 <value>2181</value>
  
</property>
  
<property>
  
 <name>hbase.zookeeper.property.dataDir</name>
  
 <value>/opt/hbase/zookeeper</value>
  
</property>

hbase.rootdir 属性的配置应在之前配置的HDFS的 fs.defaultFS 属性上添加hbase目录；在dfs.replication 属性中设置副本数量。默认情况下，将副本设置为 1。在完全分布式模式下，存在多个数据节点，因此我们可以通过设置 dfs.replication 为超过1的值。

编辑hbase-env.sh文件，添加以下配置，来配置区域服务器的路径，并启动自动管理zookeeper：

同伪分布。

启动hbase进行测试，指令如下：

start-all.sh  
start-hbase.sh  
hbase shell

**截图：**

### HBase Shell的操作

我们创建一个名为test的表，并创建一个名为data的列族(Column Family)，指令如下:

create 'test', 'data'

**截图**：

注意：HBase的表中会有一个系统默认的属性作为行键，无需自行创建，默认为put命令操作中表名后第一个数据。

为了证明新表已成功创建，运行 list 命令，输出用户空间中的所有表：

**截图**：

以上过程会将表 test 保存到HDFS上，位置为 /hbase/data/default 即我们在 hbase-site.xml 上配置的 hbase.rootdir 。打开 http://127.0.0.1:50070/ ，选择 utilities/Browse the file system查看：

**截图**：

我们将数据插入到数据列族中的三个不同行和列中，获取第一行，然后列出表内容，指令如下：

put 'test', 'row1', 'data:1', 'value1'  
put 'test', 'row2', 'data:2', 'value2'  
put 'test', 'row3', 'data:3', 'value3'  
get 'test', 'row1'  
scan 'test'

截图

对 row1 的 data1 进行修改：

put 'test', 'row1', 'data:1', 'new value'  
get 'test', 'row1'

**截图**：

如果想删除表，指令如下：

disable 'test'  
drop 'test'  
list

**截图**：

如图，test表已经被删除了。

在HDFS的 /hbase/data/default 目录下的test也被删除了。

离开HBase Shell的指令为 exit ：

HBase的关闭方法为：

stop-hbase.sh

**截图**：

## 使用Java操作HBase

### 1-安装IDEA

sudo snap install intellij-idea-community --classic

**截图**：

### 2-安装maven

使用以下命令下载并解压maven3.9.0：

wget https://dlcdn.apache.org/maven/maven-3/3.8.7/binaries/apache-maven-3.8.7-bin.tar.gz  
tar -zxvf ~/apache-maven-3.8.7-bin.tar.gz

这里会404，进官网发现没有3.8.7，改为3.8.8：

wget https://dlcdn.apache.org/maven/maven-3/3.8.8/binaries/apache-maven-3.8.8-bin.tar.gz

**截图**：

修改/apache-maven-3.8.8/conf/settings.xml，在其中添加如下配置：

<mirror>  
 <id>alimaven</id>  
 <mirrorOf>central</mirrorOf>  
 <name>aliyun maven</name>  
 <url>http://maven.aliyun.com/nexus/content/repositories/central/</url>  
</mirror>  
<mirror>  
 <id>aliyunmaven</id>  
 <mirrorOf>\*</mirrorOf>  
 <name>阿里云公共仓库</name>  
 <url>https://maven.aliyun.com/repository/public</url>  
</mirror>  
<localRepository>/opt/maven</localRepository>

**截图**：

以上配置添加了一个镜像以提高下载速度，并指定了一个本地的maven仓库路径。

### 3-在IDEA中配置maven

首先创建一个java项目，并选择maven为构建工具：

**截图：**

在IDEA中打开 File/Settings/Build,Execution,Deployment/BuildTools/Maven ，分别设置其中的Maven home path 、 User setting file 和 Local repository 为你的**maven安装目录**、安装目录下的 /apache-maven-3.8.7/conf/settings.xml 和**自己的本地仓库路径**。

**截图**：

### 4-配置log4j

在java项目的resources文件夹下添加 log4j.properties 文件：

内容如下：

**截图**：

### 5-在java中调用HBase

实例代码Main.java如下：

**截图**

### Build Success

截图