

实验7 HBase的安装与使用

实验7 HBase的安装与使用

一、版本介绍

二、单机安装

三、hbase伪分布式安装+ookeeper安装

1. 首先尝试内置zookeeper运行

步骤1.1 修改hbase-site.xml配置文件

步骤1.2使用 `./start-hbase.sh` 启动

2. zookeeper安

装

步骤2.1: 下载ZooKeeper

步骤2.2: tar文件解压

步骤2.3: 创建配置文件

步骤2.4: 启动ZooKeeper服务器

步骤2.5: 启动 CLI

3. 使用外置zookeeper启动hbase

步骤3.1 修改配置文件

步骤3.2 运行

4. 伪分布式运行截图

四、shell指令及运行截图

2.1 设计并创建合适的表;

2.2 查询选修Computer Science的学生的成绩;

2.3 增加新的列族和新列Contact:Email,并添加数据;

2.4 删除学号为2015003的学生的选课记录;

2.5 删除所创建的表。

1-5的运行截图

五、在IDEA中用java运行hbase

配置pom.xml文件

代码编写:

IDEA中运行截图

六、遇到的问题小结

一、版本介绍

hadoop: 3.3.0

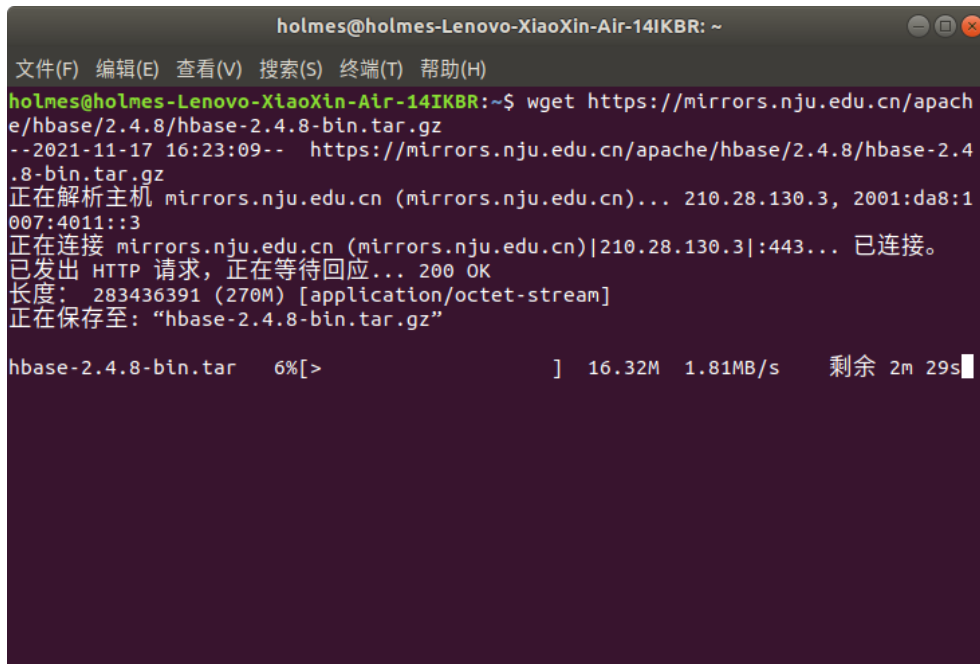
hbase: 2.4.8

zookeeper: 3.5.9

二、单机安装

1. 安装于/usr/local处并修改名称

```
1 cd /usr/local
2 wget https://mirrors.nju.edu.cn/apache/hbase/2.4.8/hbase-2.4.8-bin.tar.gz
3 tar xzvf hbase-1 2.4.8-bin.tar.gz
4 mv /usr/local/hbase-2.4.8 hbase
5 gedit /usr/local/hbase/conf/hbase-env.sh
```



```
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR:~$ wget https://mirrors.nju.edu.cn/apache/hbase/2.4.8/hbase-2.4.8-bin.tar.gz
--2021-11-17 16:23:09-- https://mirrors.nju.edu.cn/apache/hbase/2.4.8/hbase-2.4.8-bin.tar.gz
正在解析主机 mirrors.nju.edu.cn (mirrors.nju.edu.cn)... 210.28.130.3, 2001:da8:1007:4011::3
正在连接 mirrors.nju.edu.cn (mirrors.nju.edu.cn)|210.28.130.3|:443... 已连接。
已发出 HTTP 请求, 正在等待回应... 200 OK
长度: 283436391 (270M) [application/octet-stream]
正在保存至: "hbase-2.4.8-bin.tar.gz"

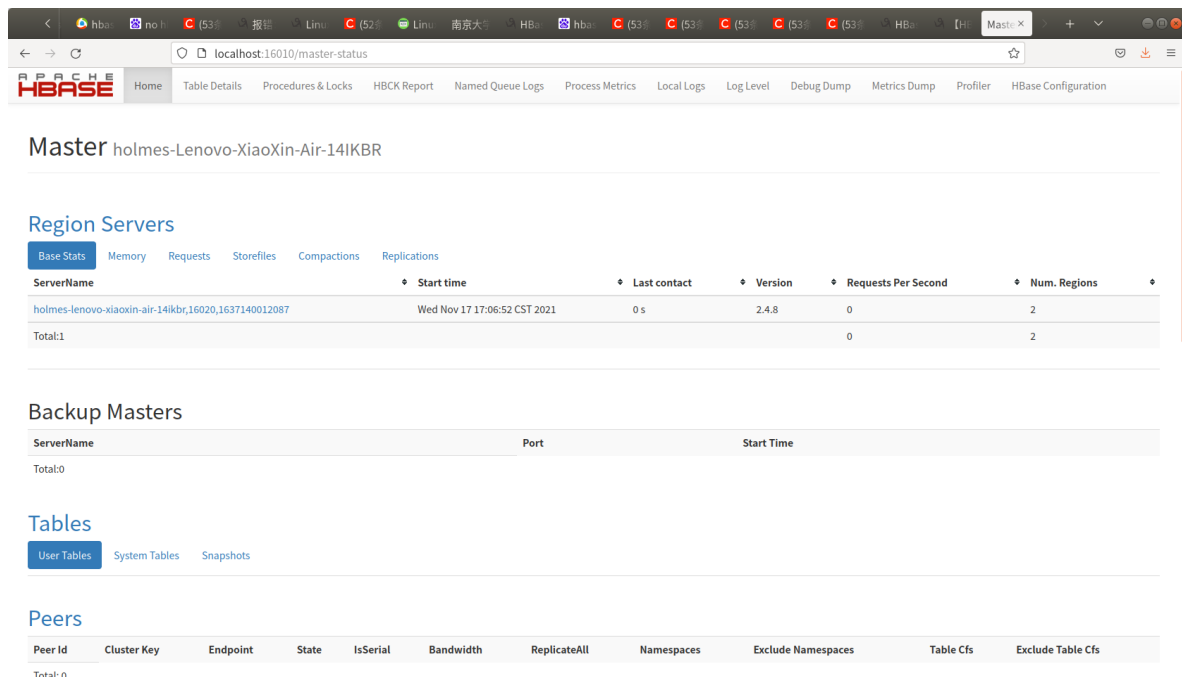
hbase-2.4.8-bin.tar  6%[>                ] 16.32M  1.81MB/s  剩余 2m 29s
```

2. 修改配置文件hbase-env.sh

```
1 export JAVA_HOME=/usr/lib/jvm/default-java
```

3. 单机启动

```
1 ./bin/start-hbase.sh
```



Master holmes-Lenovo-XiaoXin-Air-14IKBR

Region Servers

Base Stats Memory Requests Storefiles Compactions Replications

ServerName	Start time	Last contact	Version	Requests Per Second	Num. Regions
holmes-lenovo-xiaoxin-air-14ikbr,16020,1637140012087	Wed Nov 17 17:06:52 CST 2021	0 s	2.4.8	0	2
Total:1				0	2

Backup Masters

ServerName	Port	Start Time
Total:0		

Tables

User Tables System Tables Snapshots

Peers

Peer Id	Cluster Key	Endpoint	State	IsSerial	Bandwidth	ReplicateAll	Namespaces	Exclude Namespaces	Table Cfs	Exclude Table Cfs
Total:0										

三、hbase伪分布式安装+ookeeper安装

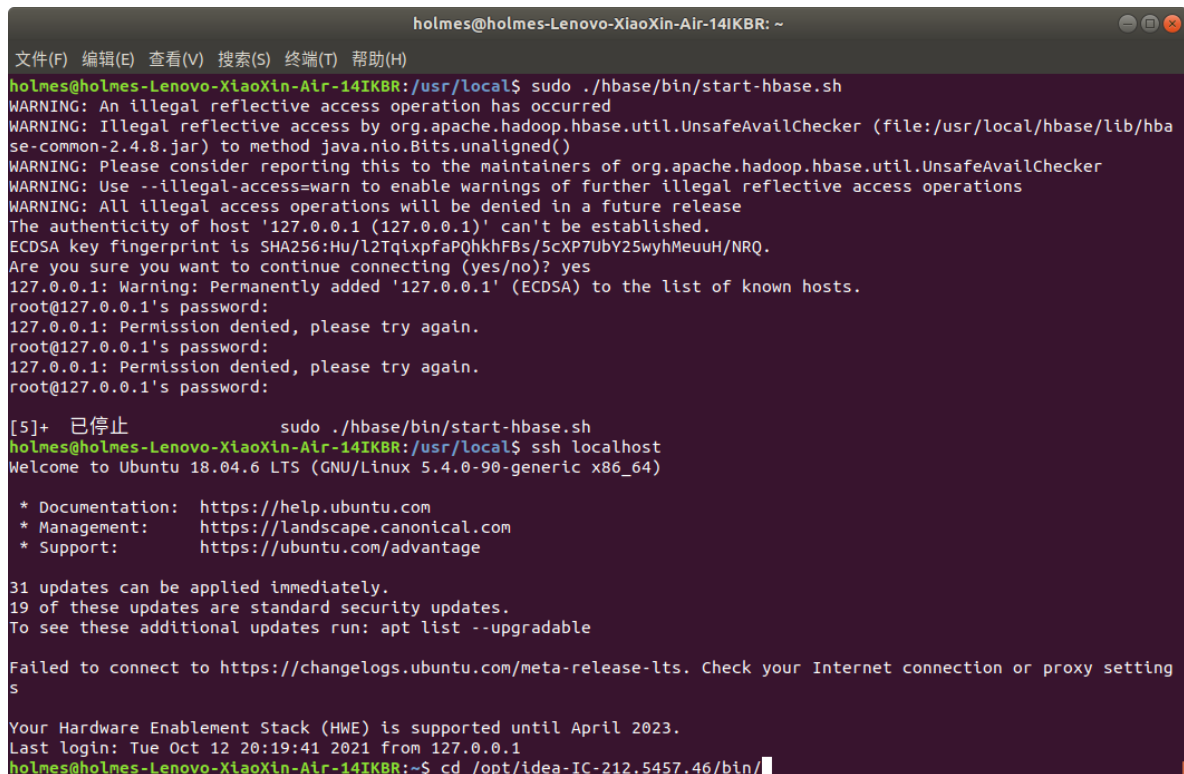
1. 首先尝试内置zookeeper运行

步骤1.1 修改hbase-site.xml配置文件

```
1  sudo vim /conf/hbase-site.xml
2
3  <property>
4  <name>hbase.cluster.distributed</name>
5  <value>true</value>
6  </property>
7  <property>
8  <name>hbase.rootdir</name>
9  <value>hdfs://hadoop-master:9000/hbase</value>
10 </property>
11 <property>
12 <name>hbase.tmp.dir</name>
13 <value>/home/<username>/hbase-tmp</value>
14 </property>
```

步骤1.2使用 `./start-hbase.sh` 启动

启动失败，报错如图所示，无法连接上localhost，但是运行ssh localhost可以正常连接。



```
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR:/usr/local$ sudo ./hbase/bin/start-hbase.sh
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.hbase.util.UnsafeAvailChecker (file:/usr/local/hbase/lib/hbase-common-2.4.8.jar) to method java.nio.Bits.unaligned()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.hbase.util.UnsafeAvailChecker
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:Hu/l2TqlxpfaPQhkhFBs/5cXP7Uby25wyhMeuuH/NRQ.
Are you sure you want to continue connecting (yes/no)? yes
127.0.0.1: Warning: Permanently added '127.0.0.1' (ECDSA) to the list of known hosts.
root@127.0.0.1's password:
127.0.0.1: Permission denied, please try again.
root@127.0.0.1's password:
127.0.0.1: Permission denied, please try again.
root@127.0.0.1's password:

[5]+ 已停止      sudo ./hbase/bin/start-hbase.sh
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR:/usr/local$ ssh localhost
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-90-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

31 updates can be applied immediately.
19 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy setting
s

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Tue Oct 12 20:19:41 2021 from 127.0.0.1
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR:~$ cd /opt/idea-IC-212.5457.46/bin/
```

日志如图所示

```
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR: /usr/local/hbase/logs
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
2021-11-17 17:58:00,195 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
Cnxn: Opening socket connection to server localhost/127.0.0.1:2181. Will not att
empt to authenticate using SASL (unknown error)
2021-11-17 17:58:00,196 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
Cnxn: Socket error occurred: localhost/127.0.0.1:2181: 拒绝连接
2021-11-17 17:58:01,297 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
Cnxn: Opening socket connection to server localhost/127.0.0.1:2181. Will not att
empt to authenticate using SASL (unknown error)
2021-11-17 17:58:01,299 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
Cnxn: Socket error occurred: localhost/127.0.0.1:2181: 拒绝连接
2021-11-17 17:58:02,400 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
Cnxn: Opening socket connection to server localhost/127.0.0.1:2181. Will not att
empt to authenticate using SASL (unknown error)
2021-11-17 17:58:02,400 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
Cnxn: Socket error occurred: localhost/127.0.0.1:2181: 拒绝连接
2021-11-17 17:58:03,501 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
Cnxn: Opening socket connection to server localhost/127.0.0.1:2181. Will not att
empt to authenticate using SASL (unknown error)
2021-11-17 17:58:03,502 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
Cnxn: Socket error occurred: localhost/127.0.0.1:2181: 拒绝连接
2021-11-17 17:58:04,604 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
Cnxn: Opening socket connection to server localhost/127.0.0.1:2181. Will not att
empt to authenticate using SASL (unknown error)
2021-11-17 17:58:04,605 INFO [main-SendThread(127.0.0.1:2181)] zookeeper.Client
```

尝试搜索多种解决方法均无效，听从老师建议：安装zookeeper

2. zookeeper安装

步骤2.1：下载ZooKeeper

这里注意的是需要下载.bin的版本，否则将无法运行，需要重新安装。

下载正确后进行下一步骤

步骤2.2：tar文件解压

使用以下命令解压 tar 文件 -

```
1 $ tar -zxf zookeeper-3.5.9.bin.tar.gz
```

步骤2.3：创建配置文件

使用命令 vi 打开 conf/zoo.cfg 配置文件，并将以下所有参数设置为开始点。

```
1 $ vi conf/zoo.cfg
2 tickTime = 2000
3 dataDir = /path/to/zookeeper/data
4 clientPort = 2181
5 initLimit = 5
6 syncLimit = 2
```

当配置文件已经保存成功后，再返回到终端。创建以上的dataDir文件和datalogDir文件

现在，就可以启动zookeeper服务器。

步骤2.4：启动ZooKeeper服务器

执行以下命令 -

```
1 $ bin/zkServer.sh start
```

执行此命令后，你会得到一个响应如下 -

```
1 $ JMX enabled by default
2 $ Using config: /Users/../../zookeeper-3.4.6/bin/../../conf/zoo.cfg
3 $ Starting zookeeper ... STARTED
```

步骤2.5: 启动 CLI

输入以下命令 进去zookeeper界面，正常运行

```
1 $ bin/zkCli.sh
```

3. 使用外置zookeeper启动hbase

步骤3.1 修改配置文件

```
1 export HBASE_MANAGES_ZK=false
```

步骤3.2 运行

```
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR: /usr/local/zookeeper
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
opping infoServer
2021-11-17 20:47:26,229 INFO [master/localhost:16000] handler.ContextHandler: Stoppe
d o.a.h.t.o.e.j.w.WebAppContext@365afe87{master/,null,STOPPED}{file:///usr/local/hbase
/hbase-webapps/master}
2021-11-17 20:47:26,239 INFO [master/localhost:16000] server.AbstractConnector: Stop
ped ServerConnector@664632e9{HTTP/1.1, (http/1.1)}{0.0.0.0:16010}
2021-11-17 20:47:26,239 INFO [master/localhost:16000] server.session: node0 Stopped
scavenging
2021-11-17 20:47:26,240 INFO [master/localhost:16000] handler.ContextHandler: Stoppe
d o.a.h.t.o.e.j.s.ServletContextHandler@468173fa{static,/static,file:///usr/local/hba
se/hbase-webapps/static/,STOPPED}
2021-11-17 20:47:26,241 INFO [master/localhost:16000] handler.ContextHandler: Stoppe
d o.a.h.t.o.e.j.s.ServletContextHandler@6707a4bf{logs,/logs,file:///usr/local/hbase/l
ogs/,STOPPED}
2021-11-17 20:47:26,243 INFO [master/localhost:16000] regionserver.HRegionServer: ab
orting server localhost,16000,1637153240553
2021-11-17 20:47:26,263 INFO [master/localhost:16000] regionserver.HRegionServer: st
opping server localhost,16000,1637153240553; all regions closed.
2021-11-17 20:47:26,263 INFO [master/localhost:16000] hbase.ChoreService: Chore serv
ice for: master/localhost:16000 had [] on shutdown
2021-11-17 20:47:26,268 WARN [master/localhost:16000] master.ActiveMasterManager: Fa
iled get of master address: java.io.IOException: Can't get master address from ZooKe
eper; znode data == null
2021-11-17 20:47:26,377 INFO [main-EventThread] zookeeper.ClientCnxn: EventThread sh
ut down for session: 0x1000068e61e0001
2021-11-17 20:47:26,377 INFO [master/localhost:16000] zookeeper.ZooKeeper: Session:
0x1000068e61e0001 closed
2021-11-17 20:47:26,378 INFO [master/localhost:16000] regionserver.HRegionServer: Ex
iting; stopping=localhost,16000,1637153240553; zookeeper connection closed.
2021-11-17 20:47:26,378 ERROR [main] master.HMasterCommandLine: Master exiting
java.lang.RuntimeException: HMaster Aborted
    at org.apache.hadoop.hbase.master.HMasterCommandLine.startMaster(HMasterComma
ndLine.java:261)
    at org.apache.hadoop.hbase.master.HMasterCommandLine.run(HMasterCommandLine.j
ava:149)
    at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:76)
    at org.apache.hadoop.hbase.util.ServerCommandLine.doMain(ServerCommandLine.ja
va:149)
    at org.apache.hadoop.hbase.master.HMaster.main(HMaster.java:2958)
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR:/usr/local/hbase/logs$ cd ..
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR:/usr/local/hbase$ ls
bin      conf      hbase-webapps  lib      logs      README.txt
CHANGES.md  docs      LEGAL          LICENSE.txt  NOTICE.txt  RELEASENOTES.md
holmes@holmes-Lenovo-XiaoXin-Air-14IKBR:/usr/local/hbase$
```

于是，为了解决这个问题。我开始了一段漫长且较为痛苦的路程。包括但不限于各种修改hbase的配置文件，各种修改zookeeper的配置文件，hosts主机的配置等等，但是无一例外的日志毫无变化，完全没有效果。非常的痛苦。

在我准备短暂的放弃后，出现了一点转机。

我开始停止进程，奇怪的是，我使用stop-hbase.sh指令后，长时间无法关闭，网上搜索因为有进程在运行。

同时日志中写Master existing，但是我的web端无法正常打开，且jps中没有任何相关进程。

于是我选择强行关机停止所有进程，第二天重启打开hbase后，仍然是同样的日志，但是我使用hbase shell指令可以进入到如下图的界面。

```

4734 SecondaryNameNode
7 holmes@holmes-Lenovo-XiaoXin-Air-14IKBR:/usr/local/hbase/bin$ hbase shell
/usr/local/hadoop/libexec/hadoop-functions.sh: 行 2366: HADOOP_ORG.APACHE.HADOOP.HBASE.UTIL.GETJAVAPROPERT
USER: 错误的替换
/usr/local/hadoop/libexec/hadoop-functions.sh: 行 2461: HADOOP_ORG.APACHE.HADOOP.HBASE.UTIL.GETJAVAPROPERT
_OPTS: 错误的替换
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/s
lf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/client-facing-thirdparty/slf4j-log4j12-1.7.30.jar!/o
rg/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.hbase.util.UnsafeAvailChecker (file:/usr/local/hba
se/lib/hbase-common-2.4.8.jar) to method java.nio.Bits.unaligned()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.hbase.util.UnsafeAvailChec
er
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
HBase Shell
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
For Reference, please visit: http://hbase.apache.org/2.0/book.html#shell
Version 2.4.8, rf844d09157d9dce6c54fcd53975b7a45865ee9ac, Wed Oct 27 08:48:57 PDT 2021
Took 0.0019 seconds
hbase:001:0> status

ERROR: KeeperErrorCode = NoNode for /hbase/master

For usage try 'help "status"'

Took 0.0965 seconds
hbase:002:0>

```

此时出现了新的报错 **NoNode for /hbase/master**，虽然是报错，但是终于！！是新的报错了！！（没错，长时间被一个报错折磨的人就是这么容易满足，即使不能直接success，但是遇到另一个报错也会小小的激动一下子55）

搜索该报错后，发现是datanode有问题，查看datanode的启动日志，果然有问题！



```
rack/127.0.0.1:9866
2021-11-22 09:58:42,382 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockReportLeaseManager: Registered DN 751ecce2-ee35-4182-8f3e-b6fb599d4eb8 (127.0.0.1:9866).
2021-11-22 09:58:42,430 INFO org.apache.hadoop.hdfs.server.blockmanagement.DatanodeDescriptor: Adding new storage ID DS-2899d862-8ec1-4367-a9db-b577efe3c5ff for DN 127.0.0.1:9866
2021-11-22 09:58:42,464 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockStateChange: BLOCK* processReport 0x24c0f19a823dbd41: Processing first storage report for DS-2899d862-8ec1-4367-a9db-b577efe3c5ff from datanode 751ecce2-ee35-4182-8f3e-b6fb599d4eb8
2021-11-22 09:58:42,470 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: initializing replication queues
2021-11-22 09:58:42,471 INFO org.apache.hadoop.hdfs.StateChange: STATE* Safe mode extension entered.
The reported blocks 78 has reached the threshold 0.9990 of total blocks 79. The minimum number of live datanodes is not required. In safe mode extension. Safe mode will be turned off automatically in 29 seconds.
2021-11-22 09:58:42,472 INFO org.apache.hadoop.hdfs.StateChange: BLOCK* processReport 0x24c0f19a823dbd41: from storage DS-2899d862-8ec1-4367-a9db-b577efe3c5ff node DatanodeRegistration(127.0.0.1:9866, datanodeUuid=751ecce2-ee35-4182-8f3e-b6fb599d4eb8, infoPort=9864, infoSecurePort=0, ipcPort=9867, storageInfo=lv=-57;cid=CID-7440a7ad-6d5b-4a01-880d-98e3e785481d;nsid=217071438;c=1633417862071), blocks: 79, hasStaleStorage: false, processing time: 9 msecs, invalidatedBlocks: 0
2021-11-22 09:58:42,476 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Total number of blocks = 79
2021-11-22 09:58:42,476 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Number of invalid blocks = 0
2021-11-22 09:58:42,476 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Number of under-replicated blocks = 2
2021-11-22 09:58:42,477 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Number of over-replicated blocks = 0
2021-11-22 09:58:42,477 INFO org.apache.hadoop.hdfs.server.blockmanagement.BlockManager: Number of blocks being written = 0
2021-11-22 09:58:42,477 INFO org.apache.hadoop.hdfs.StateChange: STATE* Replication Queue initialization scan for invalid, over- and under-replicated blocks completed in 6 msec
2021-11-22 09:59:02,484 INFO org.apache.hadoop.hdfs.StateChange: STATE* Safe mode ON, in safe mode extension.
The reported blocks 79 has reached the threshold 0.9990 of total blocks 79. The minimum number of live datanodes is not required. In safe mode extension. Safe mode will be turned off automatically in 9 seconds.
2021-11-22 09:59:12,489 INFO org.apache.hadoop.hdfs.StateChange: STATE* Safe mode is OFF
2021-11-22 09:59:12,490 INFO org.apache.hadoop.hdfs.StateChange: STATE* Leaving safe mode after 32 secs
2021-11-22 09:59:12,490 INFO org.apache.hadoop.hdfs.StateChange: STATE* Network topology has 1 racks and 1 datanodes
2021-11-22 09:59:12,490 INFO org.apache.hadoop.hdfs.StateChange: STATE* UnderReplicatedBlocks has 2 blocks
2021-11-22 09:59:41,151 INFO org.apache.hadoop.ipc.Server: IPC Server handler 4 on default port 9000, call Call#2 Retry#0 org.apache.hadoop.hdfs.protocol.ClientProtocol.mkdirs from 127.0.0.1:42962: org.apache.hadoop.security.AccessControlException: Permission denied: user=root, access=WRITE, inode="/" :holmes:supergroup:drwxr-xr-x
```

搜索该报错后得知用户名不一致导致该报错，对应的修改方法是配置hdfs-site.xml文件，查看hdfs-default.xml可以看到这一句：

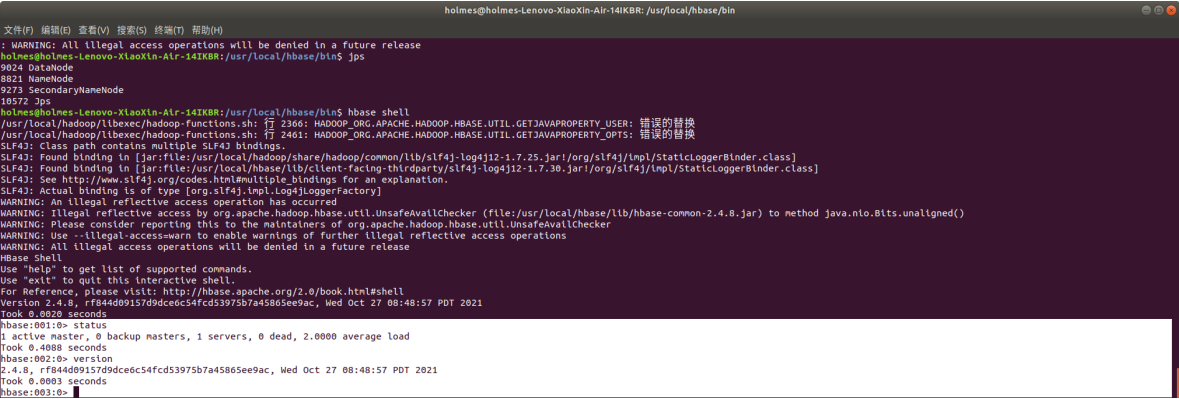
```
1 <property>
2   <name>dfs.permissions.enabled</name>
3   <value>true</value>
4   <description>
5     If "true", enable permission checking in HDFS.
6     If "false", permission checking is turned off,
7     but all other behavior is unchanged.
8     Switching from one parameter value to the other does not change the mode,
9     owner or group of files or directories.
10  </description>
11 </property>
```

将这里的value改为false，写到hdfs-site.xml中即可。

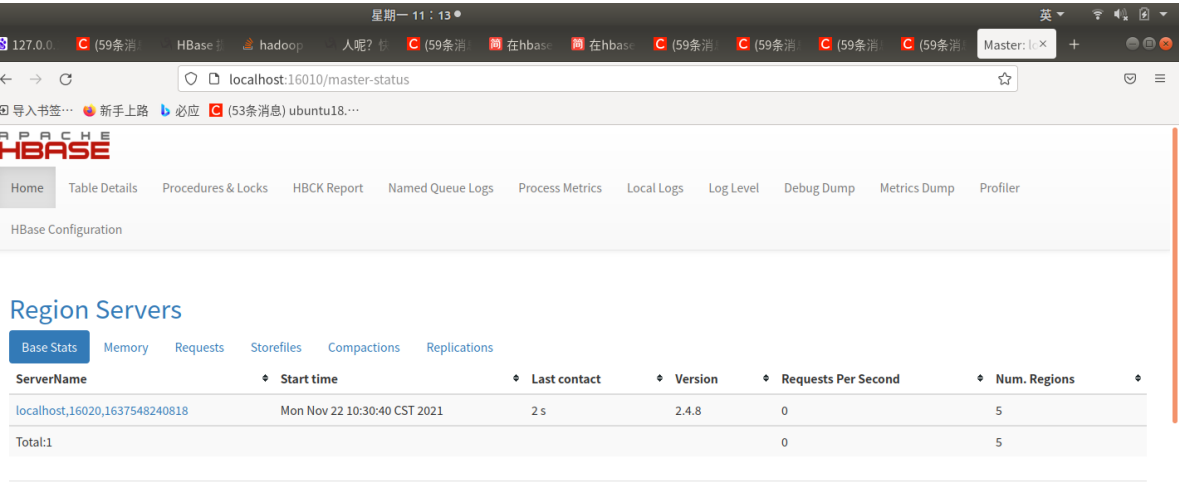
然后，重启namenode，重启hbase，hbase正常运行如下图。[我的实验3历史从此进入新纪元。](#)

4. 伪分布式运行截图

终端运行截图



16010端口截图



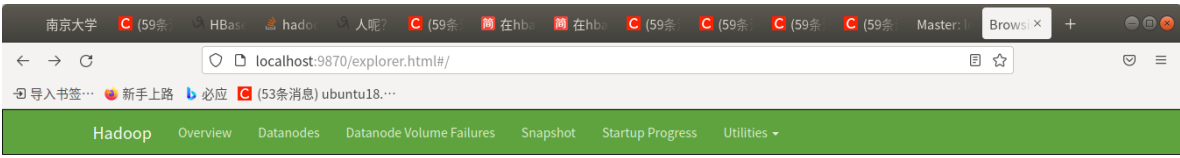
Backup Masters

ServerName	Port	Start Time
Total:0		

Tables

User Tables	System Tables	Snapshots									
3 table(s) in set. [Details] . Click count below to see list of regions currently in 'state' designated by the column title. For 'Other' Region state, browse to hbase:meta and adjust filter on 'Meta Entries' to query on states other than those listed here. Queries may take a while if the <i>hbase:meta</i> table is large.											
Namespace	Name	State	Regions								Description
			OPEN	OPENING	CLOSED	CLOSING	OFFLINE	SPLIT	Other		
default	course	ENABLED	1	0	0	0	0	0	0	'course', {NAME => 'info'}	
default	sc	ENABLED	1	0	0	0	0	0	0	'sc', {NAME => 'info'}	

9870端口截图



Browse Directory

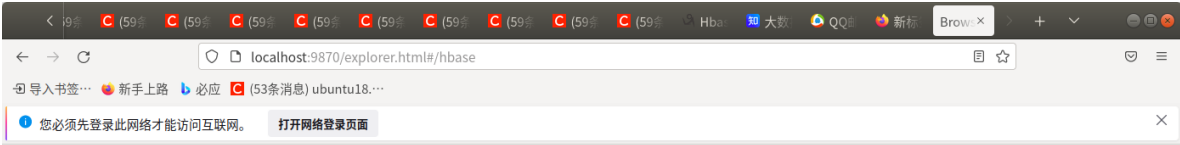
/ Go! [Icons]

Show 25 entries Search: [Search Box]

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 22 10:30	0	0 B	hbase	<input type="checkbox"/>
<input type="checkbox"/>	drwxrwx---	holmes	supergroup	0 B	Oct 12 21:04	0	0 B	tmp	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	holmes	supergroup	0 B	Nov 14 20:19	0	0 B	user	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	holmes	supergroup	0 B	Oct 29 20:10	0	0 B	usr	<input type="checkbox"/>

Showing 1 to 4 of 4 entries Previous 1 Next

Hadoop, 2021.



/hbase Go! [Icons]

Show 25 entries Search: [Search Box]

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 22 10:30	0	0 B	.hbck	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 22 10:30	0	0 B	.tmp	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 22 10:30	0	0 B	MasterData	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 22 11:45	0	0 B	WALS	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 22 11:45	0	0 B	archive	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 22 10:30	0	0 B	corrupt	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 22 10:30	0	0 B	data	<input type="checkbox"/>
<input type="checkbox"/>	-rw-r--r--	root	supergroup	42 B	Nov 22 10:30	3	128 MB	hbase.id	<input type="checkbox"/>
<input type="checkbox"/>	-rw-r--r--	root	supergroup	7 B	Nov 22 10:30	3	128 MB	hbase.version	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 22 10:30	0	0 B	mobdir	<input type="checkbox"/>
<input type="checkbox"/>	drwxr-xr-x	root	supergroup	0 B	Nov 23 14:26	0	0 B	oldWALS	<input type="checkbox"/>
<input type="checkbox"/>	drwx--x--x	root	supergroup	0 B	Nov 22 10:30	0	0 B	staging	<input type="checkbox"/>

Showing 1 to 12 of 12 entries Previous 1 Next

Hadoop, 2021.

四、shell指令及运行截图

- 为了将三张表放于一张表中，设计如下所示表格。

	Info			class1				Class2/3同理	Contact
<u>Rowkey</u>	S_Name	S_Sex	S_Age	C_No	C_Name	C_Credit	C_Score		Email
2015001	Li Lei	Male	23	123001	Math	2.0	86	Li lei@163.com
2015002	HanMeimei	Female	22					hmm@163.com
2015003	Zhang San	Male	24	123001	Math	2.0	98	zs@163.com

2.1 设计并创建合适的表;

(以2015001为例)

```

1  hbase:035:0> put 'student','2015001','info:S_Name','Li Lei'
2  Took 0.0245 seconds
      hbase:036:0> put 'student','2015002','info:S_Name','Han Meimei'
3  Took 0.0044 seconds
      hbase:037:0> put 'student','2015003','info:S_Name','Zhang San'
4  Took 0.0051 seconds
      hbase:038:0> put 'student','2015001','info:S_Sex','male'
5  Took 0.0060 seconds
      hbase:039:0> put 'student','2015002','info:S_Sex','female'
6  Took 0.0048 seconds
      hbase:040:0> put 'student','2015003','info:S_Sex','male'
7  Took 0.0048 seconds
      hbase:041:0> put 'student','2015001','info:S_Age','23'
8  Took 0.0045 seconds
      hbase:042:0> put 'student','2015002','info:S_Age','22'
9  Took 0.0053 seconds
      hbase:043:0> put 'student','2015003','info:S_Age','24'
10 hbase:045:0> put 'student','2015001','class1:C_No','123001'
11 Took 0.0270 seconds
12 hbase:046:0> put 'student','2015001','class3:C_No','123003'
13 Took 0.0064 seconds
      hbase:047:0> put 'student','2015001','class1:C_Name','Math'
14 Took 0.0054 seconds
      hbase:048:0> put 'student','2015001','class1:C_Credit','2.0'
15 Took 0.0054 seconds
      hbase:049:0> put 'student','2015001','class1:C_Score','86'
16 Took 0.0050 seconds
      hbase:050:0> put 'student','2015001','class3:C_Score','69'
17 Took 0.0055 seconds
      hbase:051:0> put 'student','2015001','class3:C_Name','English'
18 Took 0.0050 seconds
      hbase:052:0> put 'student','2015001','class3:C_Credit','3.0'

```

```
19 Took 0.0087 seconds
```

2.2 查询选修Computer Science的学生的成绩;

```
1 hbase:018:0> scan 'student',{COLUMN=>'class2:C_Score'}
2 ROW COLUMN+CELL
3 2015002 column=class2:C_Score, timestamp=2021-11-22T11:47:42.638, value=77
4 2015003 column=class2:C_Score, timestamp=2021-11-22T11:51:08.092, value=95
```

2.3 增加新的列族和新列Contact:Email,并添加数据;

```
1 hbase:020:0> put 'student','2015001','Contact:Email','lilei@qq.com'
2 Took 0.0075 seconds

3 hbase:021:0> put 'student','2015002','Contact:Email','hmm@qq.com'
4 Took 0.0047 seconds

5 hbase:022:0> put
   'student','2015003','Contact:Email','zs@qq.com'
6 Took 0.0074 seconds

7 hbase:023:0> scan 'student',{COLUMN=>'Contact'}
```

2.4 删除学号为2015003的学生的选课记录;

```
1 hbase:029:0> delete 'student','2015003','class1:C_No'
2 Took 0.0066 seconds
3 hbase:030:0> delete 'student','2015003','class1:C_Name'
4 Took 0.0069 seconds
5 hbase:031:0> delete 'student','2015003','class1:C_Credit'
6 Took 0.0068 seconds
7 hbase:032:0> delete 'student','2015003','class1:C_Score'
8 Took 0.0058 seconds
9 hbase:033:0> delete 'student','2015003','class2:C_Score'
10 Took 0.0053 seconds
11 hbase:034:0> delete 'student','2015003','class2:C_Credit'
12 Took 0.0318 seconds
13 hbase:035:0> delete 'student','2015003','class2:C_Name'
14 Took 0.0140 seconds

15 hbase:036:0> delete 'student','2015003','class2:C_No'
16 Took 0.0052 seconds

17 hbase:037:0> delete 'student','2015003','class3:C_No'
18 Took 0.0066 seconds

19 hbase:038:0> delete 'student','2015003','class3:C_Name'
20 Took 0.0117 seconds

21 hbase:039:0> delete 'student','2015003','class3:C_Score'
```

```

22 Took 0.0055 seconds

23 hbase:040:0> delete 'student','2015003','class3:C_Credit'
24 Took 0.0047 seconds

25 hbase:041:0> get 'student','2015003'
26 COLUMN                                CELL
    Contact:Email                        timestamp=2021-11-22T12:09:11.937,
value=zs@qq.com                        info:S_Age
timestamp=2021-11-22T11:40:39.806, value=24                info:S_Name
                                timestamp=2021-11-22T11:39:16.226, value=Zhang San
27 info:S_Sex                            timestamp=2021-11-22T11:40:11.784, value=male
28

```

2.5 删除所创建的表。

```
1 hbase:042:0> disable 'student'
2 Took 1.2243 seconds

3 hbase:043:0> drop 'student'
4 Took 0.3454 seconds

5 hbase:044:0> exists 'student'
6 Table student does not existTook 0.0071 seconds

7 => false
```

1-5的运行截图

```
hohmes@hohmes-Lenovo-XiaoXin-Air-14IKBRB:/usr/local/hbase/bin
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)

hbase>!lsz<0>- put 'student','2015003','class2:C_No','123002'
took 0.0051 seconds
hbase>!lsc<0>- put 'student','2015003','class2:C_Name','Computer Science'
took 0.0117 seconds
hbase>!ldc<0>- put 'student','2015003','class2:C_Credit','5.0'
took 0.0076 seconds
hbase>!lis<0>- put 'student','2015003','class2:C_Score','95'
took 0.0045 seconds
hbase>!RIS<0>- scan 'student'
ROW
2015001      column=Class1:C_Credit,timestamp=2021-11-22T11:42:58.368,value=2.0
2015001      column=Class1:C_Name,timestamp=2021-11-22T11:42:41.448,value=male
2015001      column=Class1:C_No,timestamp=2021-11-22T11:41:53.659,value=123001
2015001      column=Class1:C_Score,timestamp=2021-11-22T11:43:58.262,value=86
2015001      column=Class1:C_Credit,timestamp=2021-11-22T11:44:39.945,value=3.0
2015001      column=Class1:C_Name,timestamp=2021-11-22T11:44:22.225,value=English
2015001      column=Class1:C_No,timestamp=2021-11-22T11:42:22.841,value=123003
2015001      column=Class1:C_Score,timestamp=2021-11-22T11:44:03.294,value=69
2015001      column=Info5_Age,timestamp=2021-11-22T11:40:19.188,value=23
2015001      column=Info5_Name,timestamp=2021-11-22T11:38:33.893,value=Lel
2015001      column=Info5_Sex,timestamp=2021-11-22T11:39:40.176,value=female
2015002      column=Class2:C_Credit,timestamp=2021-11-22T11:47:21.028,value=5.0
2015002      column=Class2:C_Name,timestamp=2021-11-22T11:46:19.769,value=Computer Science
2015002      column=Class2:C_No,timestamp=2021-11-22T11:46:08.692,value=123002
2015002      column=Class2:C_Score,timestamp=2021-11-22T11:47:42.638,value=77
2015002      column=Class3:C_Credit,timestamp=2021-11-22T11:48:56.351,value=3.0
2015002      column=Class3:C_Name,timestamp=2021-11-22T11:48:04.251,value=English
2015002      column=Class3:C_No,timestamp=2021-11-22T11:48:39.217,value=123003
2015002      column=Class3:C_Score,timestamp=2021-11-22T11:47:53.687,value=99
2015002      column=Info5_Age,timestamp=2021-11-22T11:40:25.905,value=22
2015002      column=Info5_Name,timestamp=2021-11-22T11:38:53.541,value=Han Meimei
2015002      column=Info5_Sex,timestamp=2021-11-22T11:39:57.525,value=female
2015003      column=Class1:C_Credit,timestamp=2021-11-22T11:49:49.326,value=2.0
2015003      column=Class1:C_Credit,timestamp=2021-11-22T11:49:39.368,value=2.0
2015003      column=Class1:C_No,timestamp=2021-11-22T11:49:27.907,value=123001
2015003      column=Class1:C_Score,timestamp=2021-11-22T11:50:01.491,value=98
2015003      column=Class2:C_Credit,timestamp=2021-11-22T11:50:54.145,value=5.0
2015003      column=Class2:C_Name,timestamp=2021-11-22T11:50:41.691,value=Computer Science
2015003      column=Class2:C_No,timestamp=2021-11-22T11:50:18.239,value=123002
2015003      column=Class2:C_Score,timestamp=2021-11-22T11:51:08.892,value=95
2015003      column=Info5_Age,timestamp=2021-11-22T11:40:39.406,value=24
2015003      column=Info5_Name,timestamp=2021-11-22T11:39:22.6,value=Zhang San
2015003      column=Info5_Sex,timestamp=2021-11-22T11:40:11.784,value=female
3 row(s)
took 0.0668 seconds
hbase>!RI7<0>- scan 'student',[COLUMN=>'Class2:C_Score']
ROW
org.apache.hadoop.hbase.regionserver.NoSuchColumnFamilyException: org.apache.hadoop.hbase.regionserver.NoSuchColumnFamilyException: Column family class2 does not exist in region student,,1637552284932.e75bc2dd95ccbbac129ad9dbd2e29b3c,in table 'student',(NAME => 'class1', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1', KEEP_DELETED CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER')
at org.apache.hadoop.hbase.regionserver.HRegion.checkFamily(Exception.java:819)
at org.apache.hadoop.hbase.regionserver.HRegion.getScanner(Region.java:3123)
at org.apache.hadoop.hbase.regionserver.HRegion.getScanner(Region.java:3123)
at org.apache.hadoop.hbase.regionserver.HBaseServices.lambda$scan$5(HBaseServices.java:320)
at org.apache.hadoop.hbase.regionserver.HBaseServices.scan(HBaseServices.java:342)
at org.apache.hadoop.hbase.shaded.protobuf.GeneratedMethodInvoker$ReflectionBasedInvoker.call(LockAndGetMethodInterceptor.java:623)
at org.apache.hadoop.hbase.shaded.protobuf.GeneratedMethodInvoker$ReflectionBasedInvoker.call(RpcServer.java:392)
at org.apache.hadoop.hbase.shaded.protobuf.GeneratedMethodInvoker$ReflectionBasedInvoker.call(RpcServer.java:392)
at org.apache.hadoop.hbase.shaded.protobuf.GeneratedMethodInvoker$ReflectionBasedInvoker.call(RpcServer.java:392)
at org.apache.hadoop.hbase.shaded.protobuf.GeneratedMethodInvoker$ReflectionBasedInvoker.call(RpcServer.java:392)
at org.apache.hadoop.hbase.shaded.protobuf.GeneratedMethodInvoker$ReflectionBasedInvoker.call(RpcServer.java:392)
at org.apache.hadoop.hbase.shaded.protobuf.GeneratedMethodInvoker$ReflectionBasedInvoker.call(RpcServer.java:392)
For usage try /help "scan".
```

```
hbase>!RIS<0>- scan 'student',[COLUMN=>'class2:C_Score']
ROW
2015001      column=Class1:C_Score,timestamp=2021-11-22T11:47:42.638,value=77
2015002      column=Class2:C_Score,timestamp=2021-11-22T11:51:08.892,value=95
2015003
took 0.0153 seconds
```



```

1  configuration = HBaseConfiguration.create();
2  configuration.set("hbase.zookeeper.quorum", "localhost");
3  configuration.set("hbase.zookeeper.property.clientPort", "2181");
4  try {
5      connection = ConnectionFactory.createConnection(configuration);
6      //3.1获得Admin接口
7      admin = connection.getAdmin();
8  } catch (IOException e) {
9      e.printStackTrace();
10 }

```

问题1: 使用createTable+insert函数

```

1  String familyNames[]={"info", "class1", "class2", "class3"};
2  createTable("student", familyNames);
3  //向表中插入2015001数据
4  insert("student", "2015001", "info", "S_Name", "Li Lei");
5  insert("student", "2015001", "info", "S_Sex", "male");
6  insert("student", "2015001", "info", "S_Age", "23");
7  insert("student", "2015001", "class1", "C_No", "123001");
8  insert("student", "2015001", "class1", "C_Name", "Math");
9  insert("student", "2015001", "class1", "C_Credit", "2.0");
10 insert("student", "2015001", "class1", "C_Score", "86");
11 insert("student", "2015001", "class3", "C_No", "123003");
12 insert("student", "2015001", "class3", "C_Name", "English");
13 insert("student", "2015001", "class3", "C_Credit", "3.0");
14 insert("student", "2015001", "class3", "C_Score", "69");
15
16 //展示函数关键部分
17 public static void insert(String tableName, String rowKey, String family, String
column, String value) throws IOException {
18     //获得Table接口,需要传入表名
19     Table table =connection.getTable(TableName.valueOf(tableName));
20     //确定rowkey
21     Put put = new Put(Bytes.toBytes(rowKey));
22     //确定列族和value
23     put.addColumn(Bytes.toBytes(family), Bytes.toBytes(column),
Bytes.toBytes(value));
24     //写入table
25     table.put(put);
26 }
27
28
29 public static void createTable(String tableName, String familyNames[]) throws
IOException {
30     //如果表存在退出
31 }
32 //通过HTableDescriptor类来描述一个表, HColumnDescriptor描述一个列族
33 HTableDescriptor tableDescriptor = new
HTableDescriptor(TableName.valueOf(tableName));
34 for (String familyName : familyNames) {
35     tableDescriptor.addFamily(new HColumnDescriptor(familyName));
36 }
37 //创建该表
38 admin.createTable(tableDescriptor);
39 }

```

问题2: scanByColumnKey+printScanResults函数

```

1  scanByColumnKey("student", "class2", "C_Score");
2
3  public static void scanByColumnKey(String tableName, String family, String
qualifier) throws IOException {
4      Table table = connection.getTable(TableName.valueOf(tableName));
5      // 通过列键 (family:qualifier) 创建扫描器, 得到基于列键扫描的数据
6      scanner =
table.getScanner(Bytes.toBytes(family), Bytes.toBytes(qualifier));
7      printScanResults();
8  }
9
10 // 格式化打印扫描到的数据
11 private static void printScanResults() {
12     for (Result row : scanner) {
13         System.out.println(row);
14         for (Cell cell : row.listCells()) {
15             System.out.println(
16                 "RowKey:"
17                 + Bytes.toString(row.getRow())
18                 + " Family:"
19                 + Bytes.toString(CellUtil.cloneFamily(cell))
20                 + " Qualifier:"
21                 + Bytes.toString(CellUtil.cloneQualifier(cell))
22                 + " Value:"
23                 + Bytes.toString(CellUtil.cloneValue(cell)));
24         }
25     }
26 }
27 }

```

问题3: addColumnFamily+insert函数

```

1  addColumnFamily("student", "Contact");
2  insert("student", "2015001", "Contact", "Email", "lilei@qq.com");
3  insert("student", "2015002", "Contact", "Email", "hmm@qq.com");
4  insert("student", "2015003", "Contact", "Email", "zs@qq.com");
5
6  public static void addColumnFamily(String tableName, String columnName) throws
IOException {
7      // 获得Table接口, 供创建列描述使用
8      TableName tableName1 = TableName.valueOf(tableName);
9      // 获得列表描述
10     HColumnDescriptor hColumnDescriptor = new HColumnDescriptor(columnName);
11     // 添加列族
12     admin.addColumn(tableName1, hColumnDescriptor);
13 }

```

问题4: deleteColumn函数

```

1  // 以class1为例
2  deleteColumn("student", "2015003", "class1", "C_No");
3  deleteColumn("student", "2015003", "class1", "C_Name");
4  deleteColumn("student", "2015003", "class1", "C_Credit");
5  deleteColumn("student", "2015003", "class1", "C_Score");
6
7  public static void deleteColumn(String tableName, String rowKey, String
familyName, String columnName)
8  throws IOException{

```

```
9      Table table = connection.getTable(TableName.valueOf(tableName));
10      //确定要删除的对象
11      Delete deleteColumn = new Delete(Bytes.toBytes(rowKey));
12      deleteColumn.addColumn(Bytes.toBytes(familyName), Bytes.toBytes(columnName));
13      table.delete(deleteColumn);
14  }
```

问题5: dropTable函数

```
1      dropTable("student");
2
3      public static void dropTable(String tableName) throws IOException {
4          //删除之前要将表disable
5          if (!admin.isTableDisabled(TableName.valueOf(tableName))) {
6              admin.disableTable(TableName.valueOf(tableName));
7          }
8          admin.deleteTable(TableName.valueOf(tableName));
9
10     }
```

IDEA中运行截图

-----task1-----

createtable success!

insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015001 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015002 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.
insert recored 2015003 to table student ok.

-----task2-----

keyvalues={2015002/class2:C_Score/1637649131953/Put/vlen=2/seqid=0}
RowKey:2015002 Family:class2 Qualifier:C_Score Value:77
keyvalues={2015003/class2:C_Score/1637649131987/Put/vlen=2/seqid=0}
RowKey:2015003 Family:class2 Qualifier:C_Score Value:95

```
-----task3-----  
add column family Contact ok  
insert recored 2015001 to table student ok.  
insert recored 2015002 to table student ok.  
insert recored 2015003 to table student ok.
```

```
-----task4-----  
delete 2015003:class1:C_No susscess  
delete 2015003:class1:C_Name susscess  
delete 2015003:class1:C_Credit susscess  
delete 2015003:class1:C_Score susscess  
delete 2015003:class2:C_No susscess  
delete 2015003:class2:C_Name susscess  
delete 2015003:class2:C_Credit susscess  
delete 2015003:class2:C_Score susscess  
delete 2015003:class3:C_No susscess  
delete 2015003:class3:C_Name susscess  
delete 2015003:class3:C_Credit susscess  
delete 2015003:class3:C_Score susscess
```

```
-----task5-----  
deletetable student ok.  
  
Process finished with exit code 0
```

六、遇到的问题小结

- 总的来说，这次实验看起来好像不是很复杂的样子，但是此次配环境就我个人而言，还是比较艰难的。配置过程中遇到的问题在前文：三、hbase分布式安装中阐述，最大的问题就是忽略了hadoop启动的问题，一直纠结于hbase和zookeeper的配置，反而忽略了hbase所依赖的hadoop，如果能够早一些看hadoop启动的日志，应该会更快地找到问题所在。
- hbase的表格设计：由于习惯了mysql中表格的设计，最初的想法是设计三张表，但是hbase并没有像关系型数据库中表的连接功能，也就是说，第二题的根据课程名查看学生分数，不能使用一个指令完成，必须先从课程表中确认名为computer science的课程号，再去成绩表中读取。所以，我选择放弃刚刚创建好的三个表，转而**将三个表合为一个表，充分利用hbase存储的特性**，也就是如四开头所示的表，该表由于汇集了三张表的全部信息，查询信息会更加方便。
- shell中指令编写：主要遇到的问题是第四题，删除2015003学生的选课信息，起初我使用的指令是 `delete 'student', '2015003', 'class2'`，使用完之后用get指令查询2015003的信息，发现并没有被删除，继续查询指令得知，不指出列族名的删除无效，必须采用 `delete 'student', '2015003', 'class2:C_Credit'` 的指令才能真正删除信息。
- 用Java代码操作hbase：主要遇到的问题是，最初借鉴黄老师书上的代码，发现有些代码已经不再适用新版本。另外，犯了一个愚蠢的错误，我发现我的代码在第二题运行完之后，第三题就找不到master，在环境配置上找了很久的错误后，发现是因为我在第二题代码的最后自己断了连接！可以说是非常愚蠢了，所以...以后一定要先从代码找问题。