華中科技大學

大数据处理实验报告

实验二：HBase 的基本操作

姓　　名：刘日星

学　　号：X2020I1007

院　　系：计算机科学与技术

专　　业：计算机科学&金融

年　　级：CS1804 交换生

指导教师：石宣化

2021 年 12 月 8 日

一：实验目的

1、了解 HBase 的用途

2、掌握 HBase 的基本命令

二：实验要求

1、第四节中的实验内容要附上完整的实验过程截图以及必要的文字说明，每个人的 IP 地址等不同，不能直接套用样例的截图。

2、请同学们在完成报告后，将报告的 pdf 版本命名为：大数据实验二+姓名+学号.pdf，并在这周五前,发到邮箱:

aaaaltaaaa@126.com(石老师班)

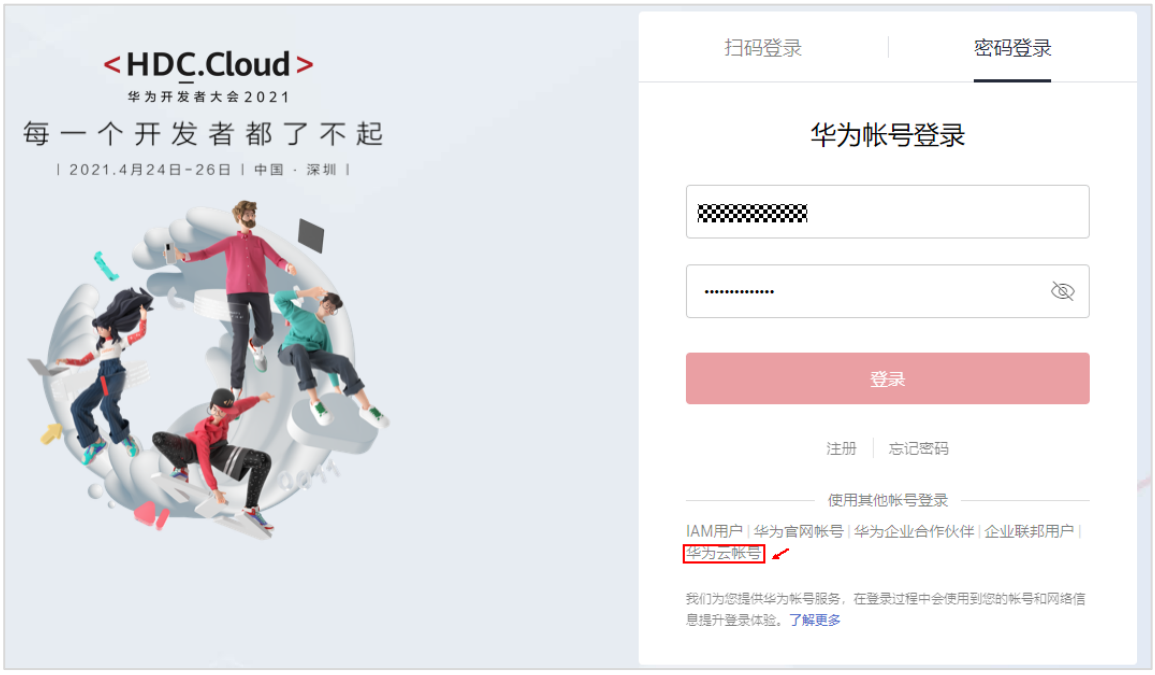
798792873@qq.com(郑老师班)

三：实验环境配置

步骤 1 登录华为云网站

https://www.huaweicloud.com

点击右上角登录，输入账号和密码

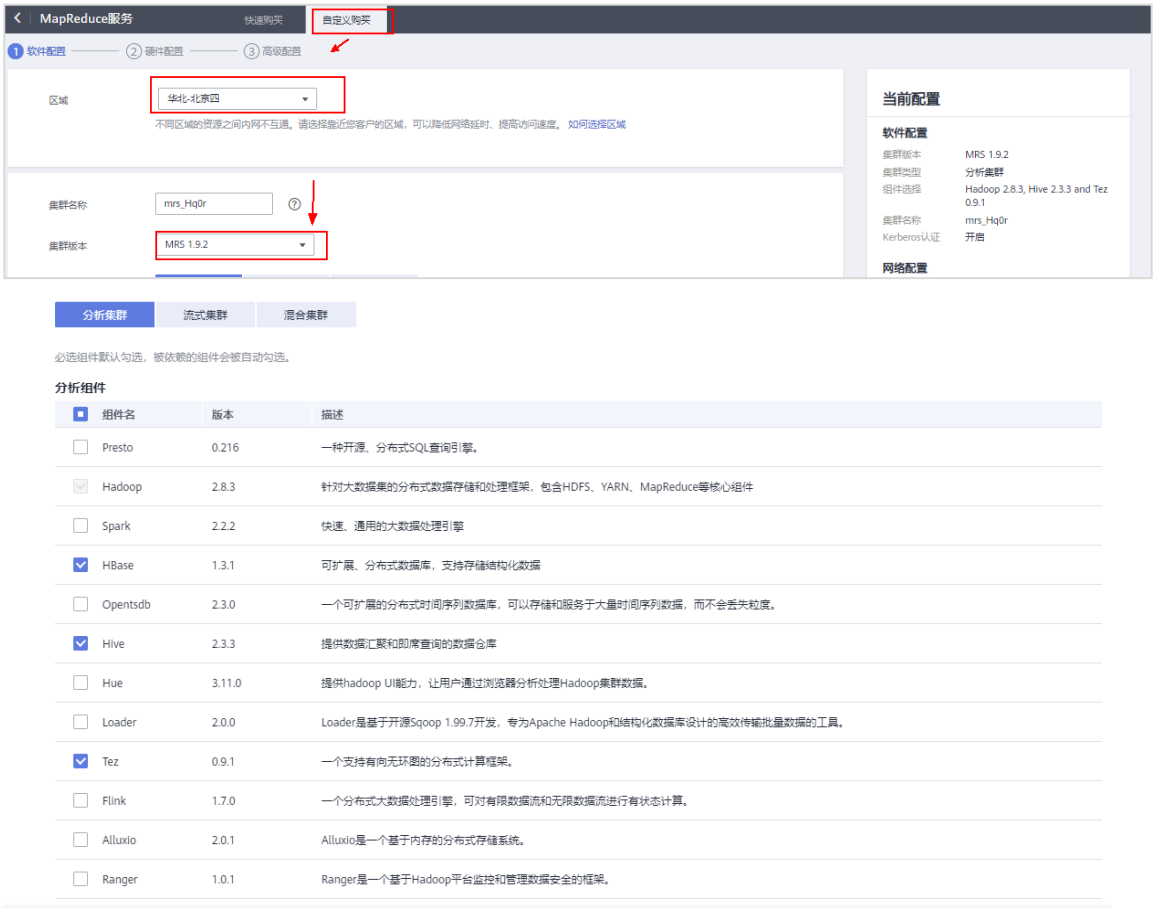


注意：华为云已统一登录入口，若仍不能登录则点击下方华为云账号进行登录。

步骤 2　点击“EI 企业智能”选择“MapReduce 服务”

步骤 3　点击“立即购买”

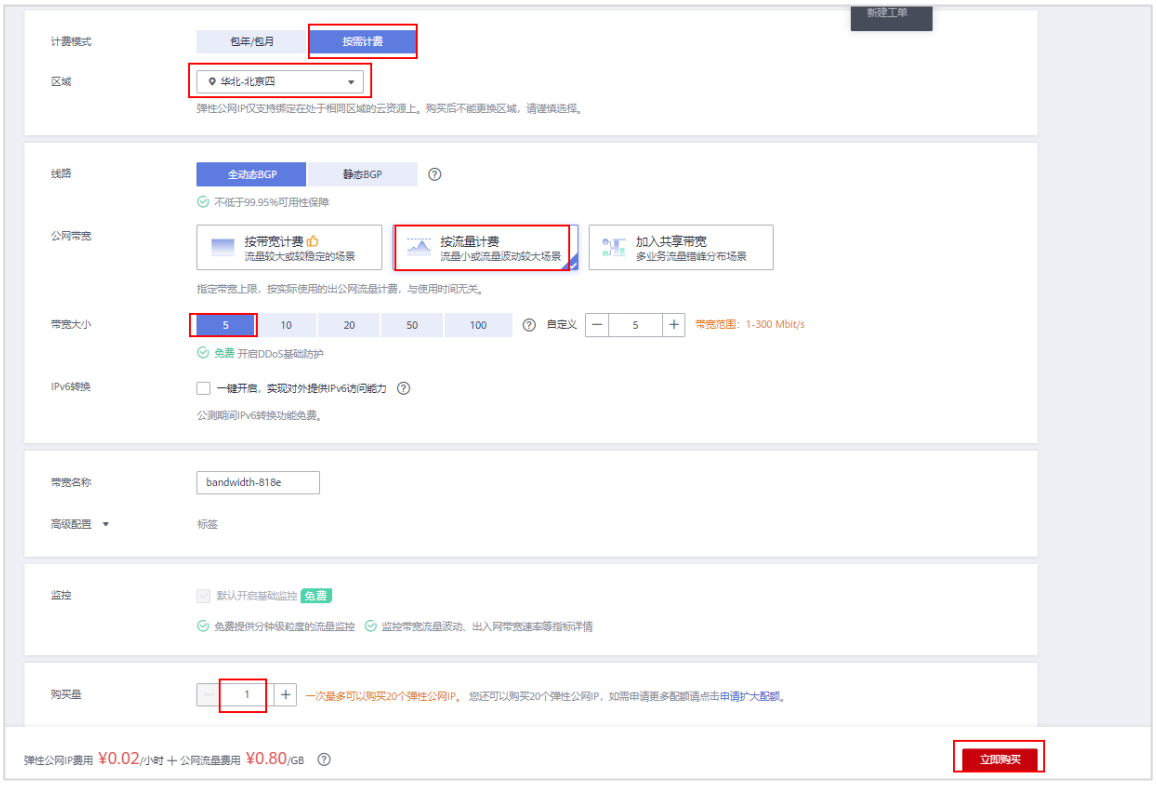
选择“自定义购买”



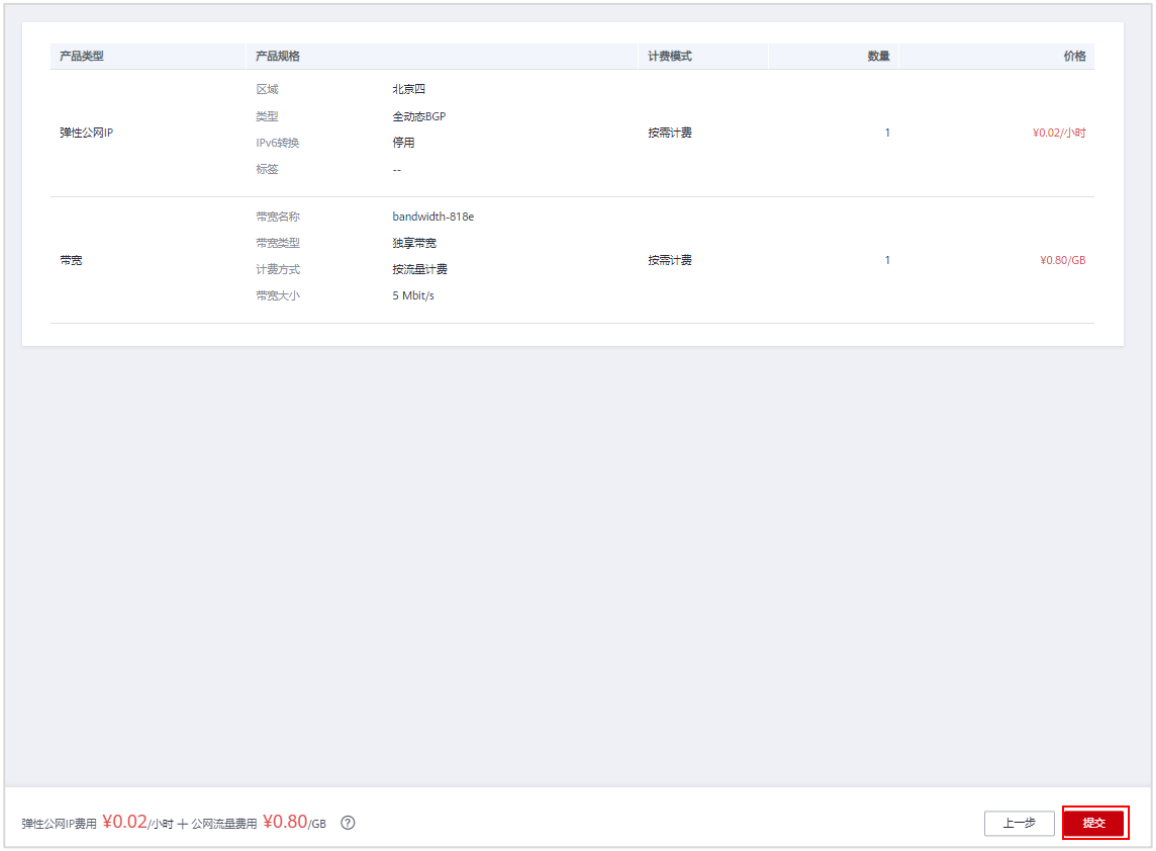
点击下一步，进入硬件配置

选择“按需计费”，“可用区 2”，点击“弹性公网 IP”，如下图：

点击“购买弹性公网 IP”，选择“按需计费”，“按流量计费”，“5M”,点击“立即购买”，如下图：



点击“提交”，如下图：



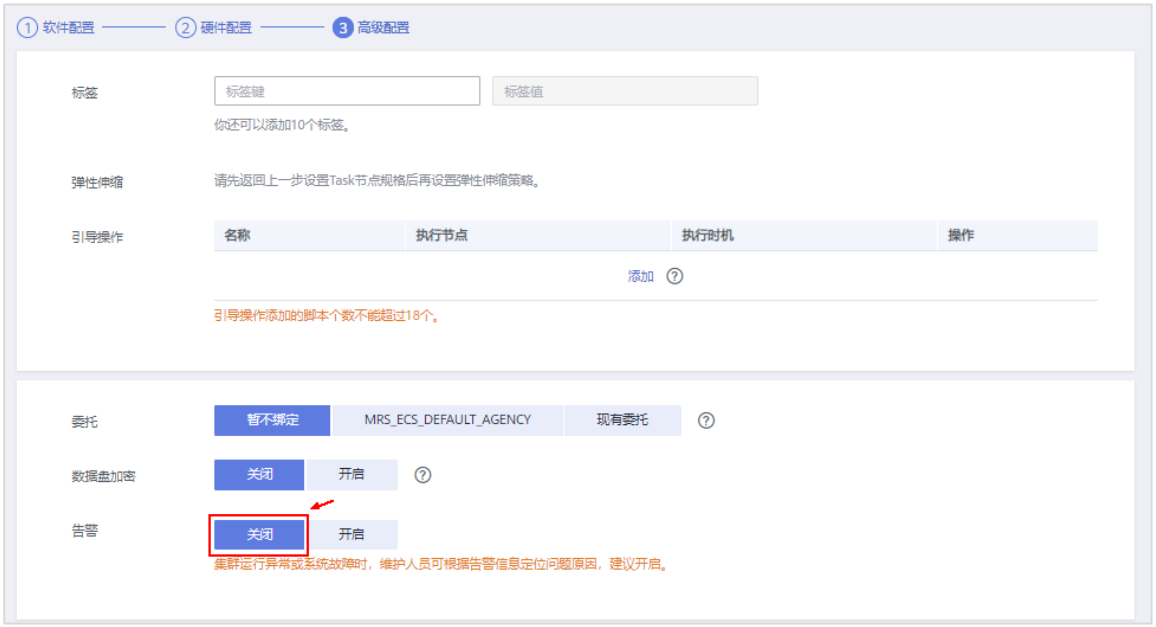
购买成功，如下图：

返回 MapReduce 服务自定义购买界面绑定 EIP

选择“鲲鹏计算”，关闭高可用，调整 core 节点数为 1,如下图：

点击“下一步”

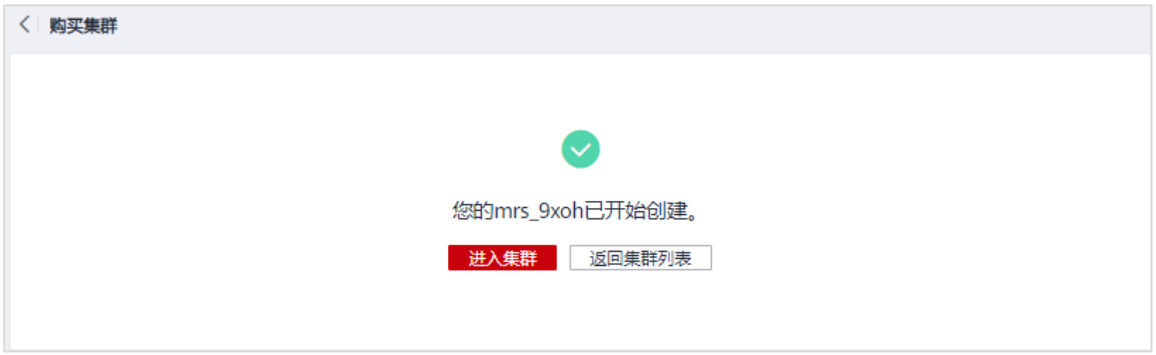
高级配置项参考如下：



点击“确认授权”

点击“立即购买”

步骤 4 点击“返回集群列表”，如下图:



创建过程需要等待几分钟，待状态变为“运行中”集群创建完成

步骤 5 点击集群名称

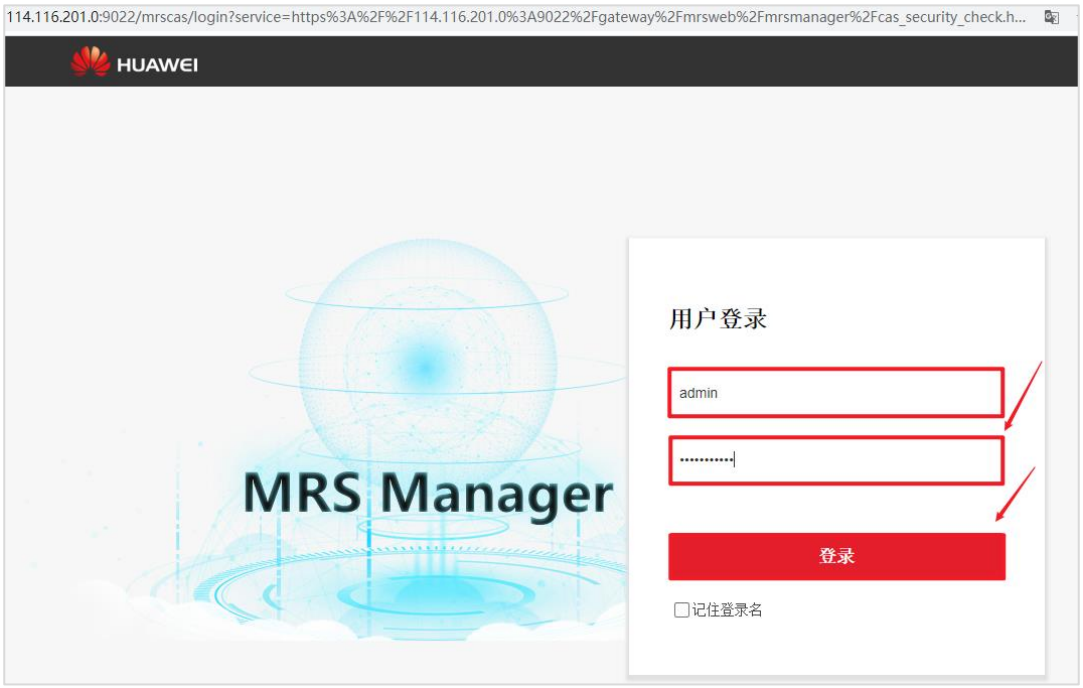
步骤 6 点击“前往 Manager”

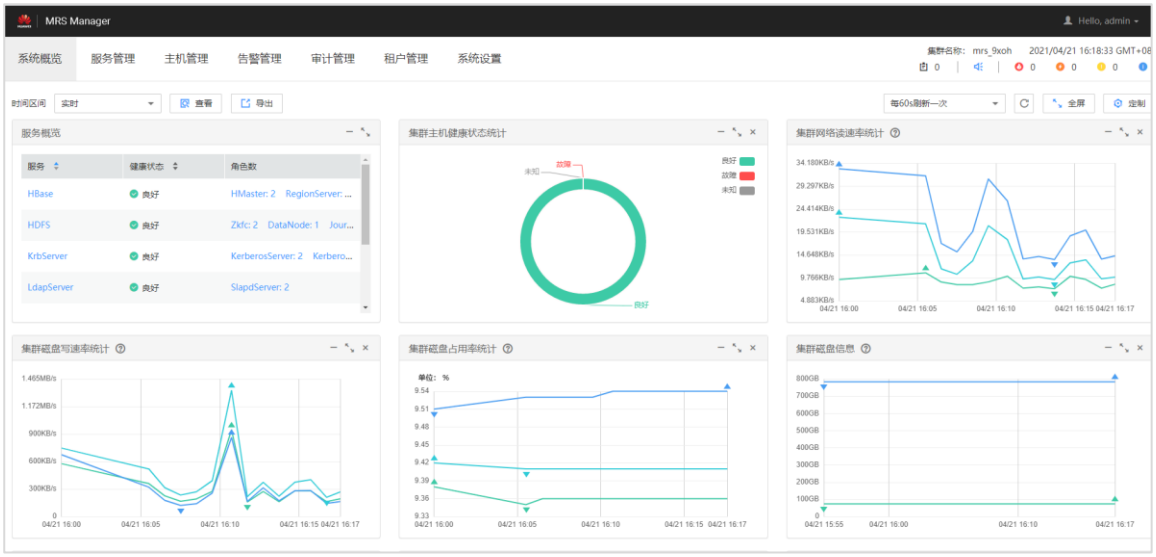
参照下图进入 Manager

点击“高级”,如下图:

点击“继续向前”，如下图：

输入用户名 admin 及密码,点击“登录”，进入 MRS Manager



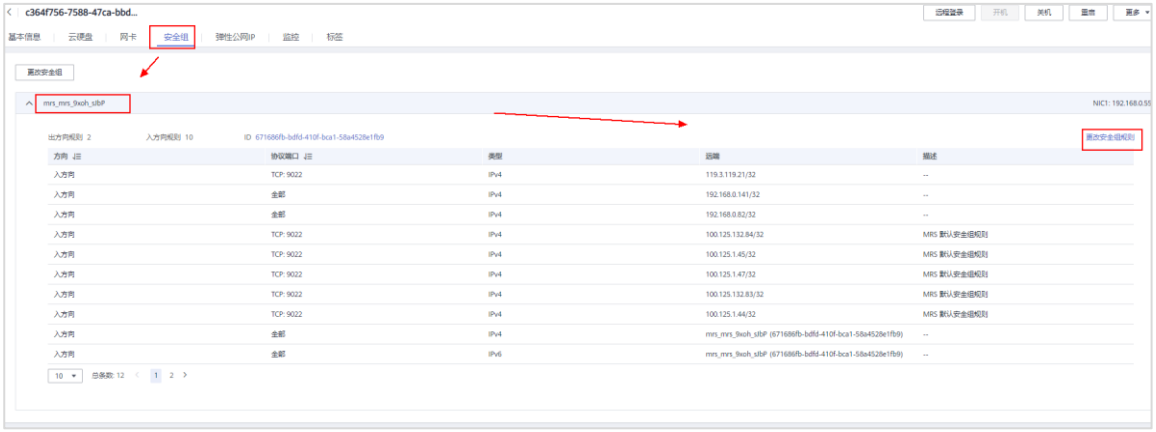


步骤 7　配置安全组

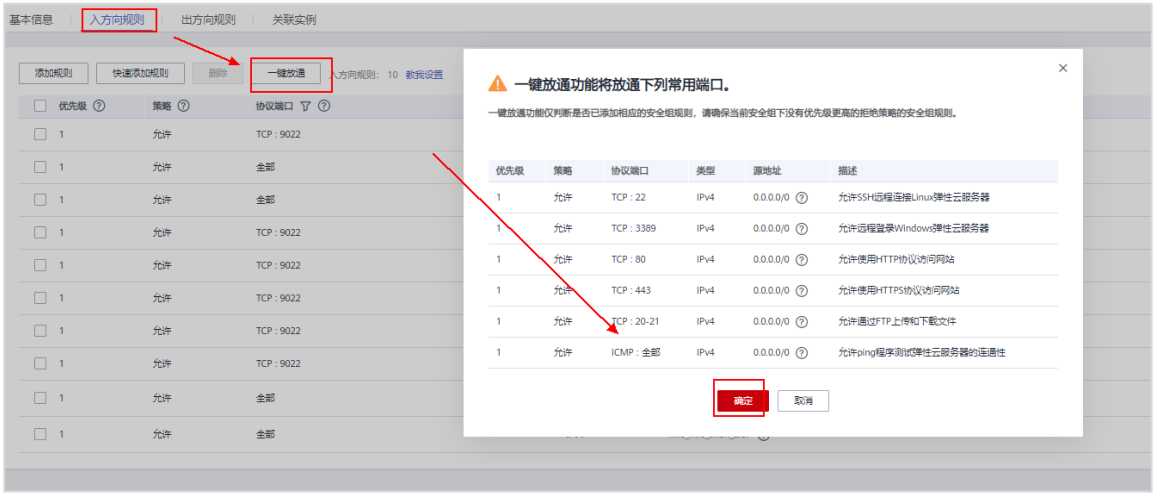
点击集群名称

选择“节点管理”，点击含有“master1”的节点

在弹出页面中选择“安全组”，点击“更改安全组规则”，如下图所示：



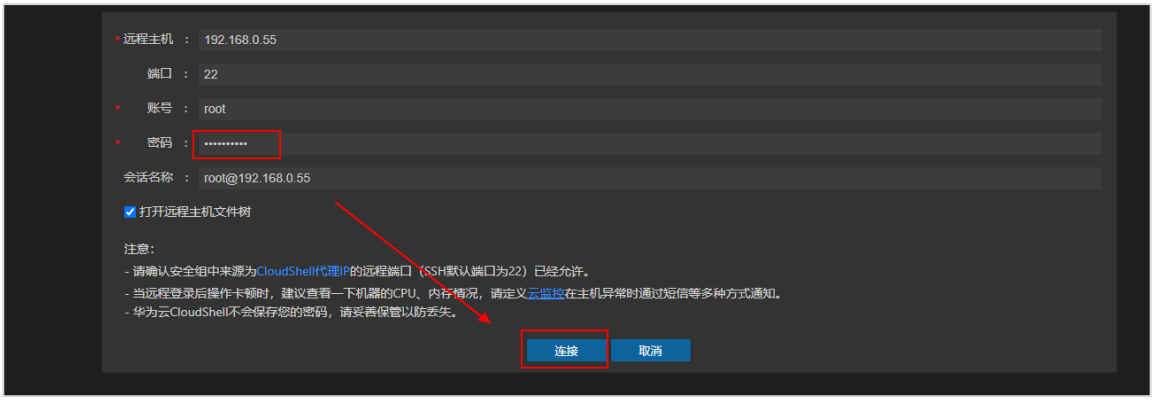
选择“入方向规则”，点击“一键放通”，确认即可。

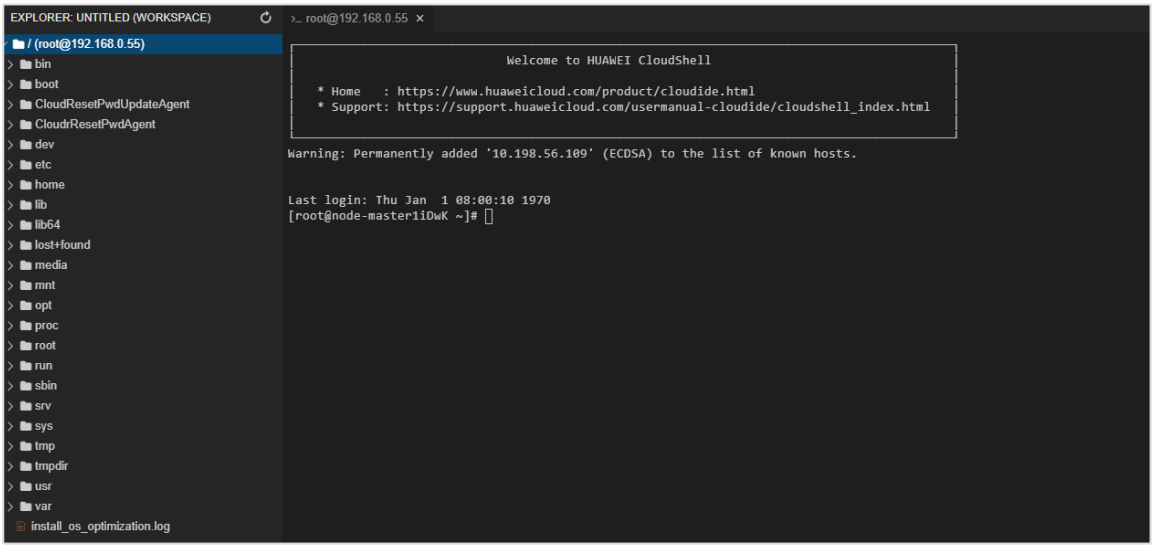


步骤 8　远程登录 master 节点

在安全组配置项，点击右上方“远程登录”，选择 cloudshell 登录。

输入密码，点击连接即可。



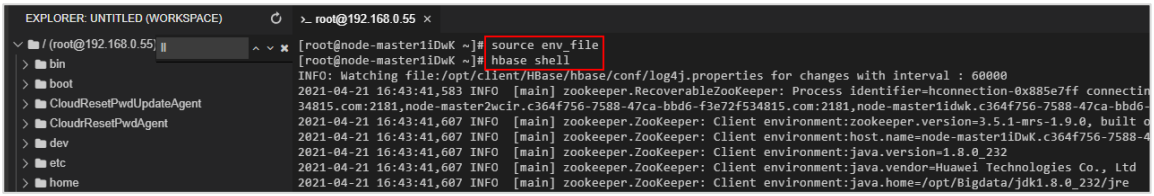


步骤 9 设置环境变量

执行命令：

source env_file

hbase shell

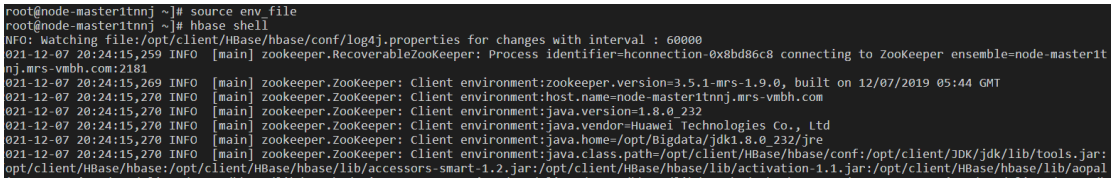


四：实验内容及步骤、实验的详细记录、实验结果分析

请附上实验过程截图（截图需包含指令）以及必要的文字分析

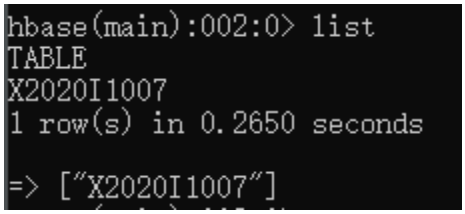
4.1 准备数据(20’)

4.1.1 进入 hbase shell(5’)



4.1.2 创建一个表，表名为学号，列族名为 cf1（ create ）(5’)

4.1.3 显示所有的表（ list ）(5’)



4.1.4 向表中增加两行数据（ put ）(5’)

行键	列族	列名	单元格的值
20200001	cf1	name	tom
20200001	cf1	gender	male
20200001	cf1	age	20
20200002	cf1	name	hanmeimei
20200002	cf1	gender	female
20200002	cf1	age	19

```
hbase(main):003:0> put 'X2020I1007','20200001','cf1:name','tom'
0 row(s) in 0.1030 seconds

hbase(main):004:0> put 'X2020I1007','20200001','cf1:gender','male'
0 row(s) in 0.0360 seconds

hbase(main):005:0> put 'X2020I1007','20200001','cf1:age','20'
0 row(s) in 0.0150 seconds

hbase(main):006:0> put 'X2020I1007','20200002','cf1:name','hanmeimei'
0 row(s) in 0.0210 seconds

hbase(main):007:0> put 'X2020I1007','20200002','cf1:gender','female'
0 row(s) in 0.0100 seconds

hbase(main):008:0> put 'X2020I1007','20200002','cf1:age','19'
0 row(s) in 0.0090 seconds
```

输入表格内容 4.1.1

```
hbase(main):009:0> scan 'X2020I1007'
ROW                           COLUMN+CELL
 20200001                     column=cf1:age, timestamp=1638881911382, value=20
 20200001                     column=cf1:gender, timestamp=1638881892668, value=male
 20200001                     column=cf1:name, timestamp=1638881860545, value=tom
 20200002                     column=cf1:age, timestamp=1638881970430, value=19
 20200002                     column=cf1:gender, timestamp=1638881954302, value=female
 20200002                     column=cf1:name, timestamp=1638881936338, value=hanmeimei
2 row(s) in 0.0510 seconds
```

显示表格内容 4.1.1(1)

4.2 查询数据(30')

4.2.1 查找表中，列族名为 cf1 的数据（ scan ）(3')

```
hbase(main):010:0> scan 'X2020I1007',{COLUMN => 'cf1'}
ROW                           COLUMN+CELL
 20200001                     column=cf1:age, timestamp=1638881911382, value=20
 20200001                     column=cf1:gender, timestamp=1638881892668, value=male
 20200001                     column=cf1:name, timestamp=1638881860545, value=tom
 20200002                     column=cf1:age, timestamp=1638881970430, value=19
 20200002                     column=cf1:gender, timestamp=1638881954302, value=female
 20200002                     column=cf1:name, timestamp=1638881936338, value=hanmeimei
2 row(s) in 0.0230 seconds
```

4.2.2 查找表中，列族名为 cf1,列名为 name 的数据（scan）(3′)

```
hbase(main):011:0> scan 'X2020I1007',{COLUMN => 'cf1:name'}
ROW                           COLUMN+CELL
 20200001                     column=cf1:name, timestamp=1638881860545, value=tom
 20200002                     column=cf1:name, timestamp=1638881936338, value=hanmeimei
2 row(s) in 0.0170 seconds
```

4.2.3 查找表中，行键为 20200001 的行（get）(3′)

```
hbase(main):012:0> get 'X2020I1007','20200001'
COLUMN                        CELL
 cf1:age                      timestamp=1638881911382, value=20
 cf1:gender                   timestamp=1638881892668, value=male
 cf1:name                     timestamp=1638881860545, value=tom
1 row(s) in 0.0440 seconds
```

4.2.4 查找表中,行键为 20200001,列族为 cf1,列名为 name 的数据（get）(3′)

```
hbase(main):013:0> get 'X2020I1007','20200001','cf1:name'
COLUMN                        CELL
 cf1:name                     timestamp=1638881860545, value=tom
1 row(s) in 0.0090 seconds
```

4.2.5 查看起始行键为 20200001,终止行键为 20200002(不包括)，限制长度为 2 的数据(scan)(3′)

```
hbase(main):014:0> scan 'X2020I1007',{STARTROW => '20200001', ENDROW => '20200002', MAXLENGTH => 2}
ROW                           COLUMN+CELL
 20200001                     column=cf1:age, ti
 20200001                     column=cf1:gender, ti
 20200001                     column=cf1:name, ti
1 row(s) in 0.0730 seconds
```

4.2.6 查看有数据值为 20 的行(scan)(3’)

```
hbase(main):016:0> scan "X2020I1007", FILTER => "ValueFilter(=,'binary:20')"
ROW                           COLUMN+CELL
 20200001                     column=cf1:age, timestamp=1638881911382, value=20
1 row(s) in 0.0470 seconds
```

4.2.7 查看有数据值为 tom 的行(scan)(3’)

```
hbase(main):019:0> scan "X2020I1007", FILTER => "ValueFilter(=,'substring:tom')"
ROW                           COLUMN+CELL
 20200001                     column=cf1:name, timestamp=1638881860545, value=tom
1 row(s) in 0.0180 seconds
```

4.2.8 查看列名为 gender 的列(scan)(3’)

```
hbase(main):020:0> scan 'X2020I1007',{COLUMN => 'cf1:gender'}
ROW                           COLUMN+CELL
 20200001                     column=cf1:gender, timestamp=1638881892668, value=male
 20200002                     column=cf1:gender, timestamp=1638881954302, value=female
2 row(s) in 0.0250 seconds
```

4.2.9 查看列名为 name，值为 hanmeimei 的行(scan)(3’)

```
hbase(main):022:0> scan 'X2020I1007',{COLUMN => 'cf1:name', FILTER => "ValueFilter(=, 'substring:hanmeimei')"}
ROW                           COLUMN+CELL
 20200002                     column=cf1:name, timestamp=1638881936338, value=hanmeimei
1 row(s) in 0.0170 seconds
```

4.2.10 查看表的属性（ desc ）(3’)

```
hbase(main):023:0> desc 'X2020I1007'
Table X2020I1007 is ENABLED
X2020I1007
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOC
K_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE =
> '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.0450 seconds
```

4.3 修改数据(20’)

4.3.1 改变表的VERSIONS为5以显示更多的历史版本(alter) (3’)

```
hbase(main):031:0> alter 'X2020I1007',{NAME=>'cf1', VERSIONS => 5}
Updating all regions with the new schema...
0/1 regions updated.
0/1 regions updated.
1/1 regions updated.
Done.
0 row(s) in 3.8580 seconds
```

4.3.2 添加行键 20200001，列族 cf1，列名 name 的多个历史版本（ put ）(3’)

行键	列族	列名	单元格的值
20200001	cf1	name	LiSi
20200001	cf1	name	ZhangSan'
20200002	cf1	name	WangWu

```
hbase(main):032:0> put "X2020I1007","20200001","cf1:name","LiSi"
0 row(s) in 0.0430 seconds

hbase(main):033:0> scan "X2020I1007"
ROW                     COLUMN+CELL
 20200001               column=cf1:age, timestamp=1638881911382, value=20
 20200001               column=cf1:gender, timestamp=1638881892668, value=male
 20200001               column=cf1:name, timestamp=1638884636047, value=LiSi
 20200002               column=cf1:age, timestamp=1638881970430, value=19
 20200002               column=cf1:gender, timestamp=1638881954302, value=female
 20200002               column=cf1:name, timestamp=1638881936338, value=hanmeimei
2 row(s) in 0.0350 seconds

hbase(main):034:0> put "X2020I1007","20200001","cf1:name","ZhangSan"
0 row(s) in 0.0290 seconds

hbase(main):035:0> put "X2020I1007","20200001","cf1:name","WangWu"
0 row(s) in 0.0100 seconds
```

4.3.3 查看所有行键为 20200001，列簇为 cf1 的多版本数据（get）(3′)

```
hbase(main):036:0> get 'X2020I1007','20200001',{COLUMN=>'cf1:name', VERSIONS=>3}
COLUMN                          CELL
 cf1:name                       timestamp=1638884889033, value=WangWu
 cf1:name                       timestamp=1638884880182, value=ZhangSan
 cf1:name                       timestamp=1638884636047, value=LiSi
1 row(s) in 0.0130 seconds
```

这里说明一下，因手快了直接查看了 name 部分的多版本数据

4.3.4 删除行键为 20200002，列名为 age，的数据（delete）(3′)

```
hbase(main):037:0> delete 'X2020I1007','20200002','cf1:age'
0 row(s) in 0.0350 seconds
```

4.3.5 删除行键为 20200002 的行（deleteall）(4′)

```
hbase(main):038:0> deleteall 'X2020I1007','20200002'
0 row(s) in 0.0280 seconds
```

4.3.6 删除整个表（disable，drop）(4′)

```
hbase(main):039:0> disable 'X2020I1007'
2021-12-07 21:53:42,981 INFO  [main] client.HBaseAdmin: Started disable of X2020I1007
2021-12-07 21:53:45,221 INFO  [main] client.HBaseAdmin: Disabled X2020I1007
0 row(s) in 2.3010 seconds

hbase(main):040:0> drop 'X2020I1007'
2021-12-07 21:54:01,505 INFO  [main] client.HBaseAdmin: Deleted X2020I1007
0 row(s) in 1.3550 seconds
```

4.4 Region 初探(20’)

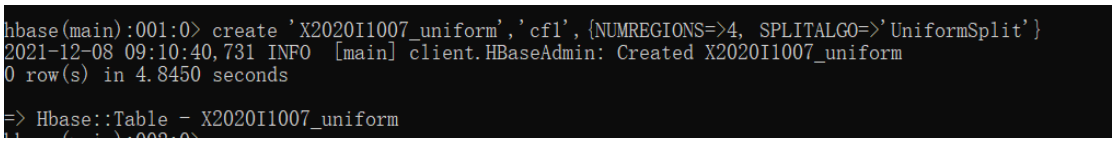
HBase 默认建表时只有一个 region，这个 region 的 rowkey 是没有边界的，即没有 startkey，也没有 endkey。在数据写入时，所有数据都会写入这个默认的 region，随着数据量的不断增加,此 region 已经不能承受不断增长的数据量,会进行 split,分成 2 个 region。在此过程中，会产生两个问题：

1.数据往一个 region 上写,会有写热点问题。

2.region split 会消耗宝贵的集群 I/O 资源。

基于此我们可以在建表的时候，创建多个空 region，并确定每个 region 的起始和终止 rowky，这样只要我们的 rowkey 设计能均匀的命中各个 region，就不会存在写热点问题，自然 split 的几率也会大大降低。hbase 提供了两种 pre-split 算法：HexStringSplit 和 UniformSplit，前者适用于十六进制字符的 rowkey，后者适用于随机字节数组的 rowkey。以 rowkey 切分，随机分为 4 个 region。

4.4.1 创建具有四个 region 的表，表名为”学号_uniform”，pre-split 算法选择 UniformSplit（create）(5’)



4.4.2 创建具有四个 region 的表，表名为”学号_num”指定 region以行键10000000,20000000,30000000划分(create) (5’)

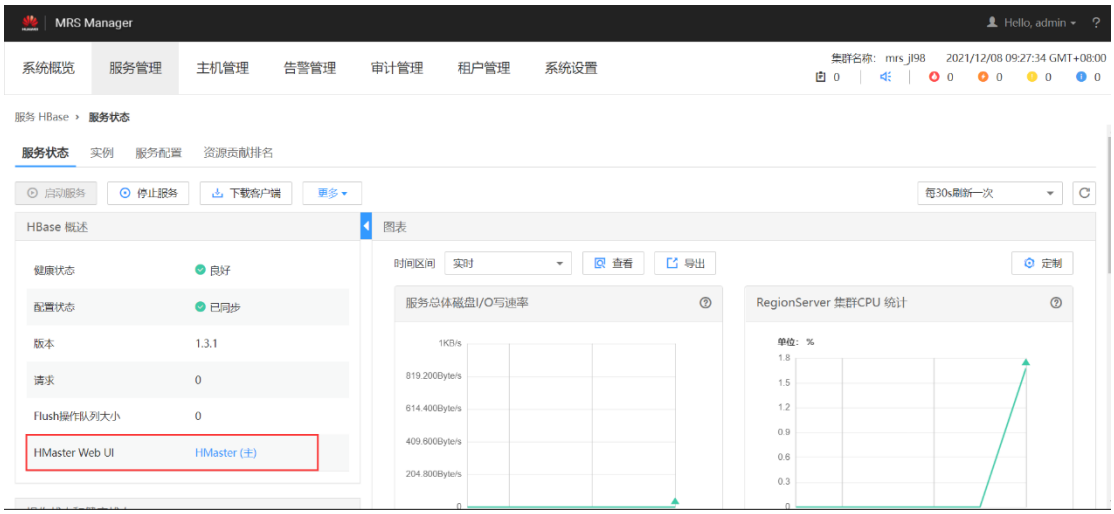
```
hbase(main):002:0> create 'X2020I1007_num','cf1',SPLITS=>['10000000','20000000','30000000']
2021-12-08 09:20:29,958 INFO  [main] client.HBaseAdmin: Created X2020I1007_num
0 row(s) in 4.2940 seconds

=> Hbase::Table - X2020I1007_num
hbase(main):003:0>
```

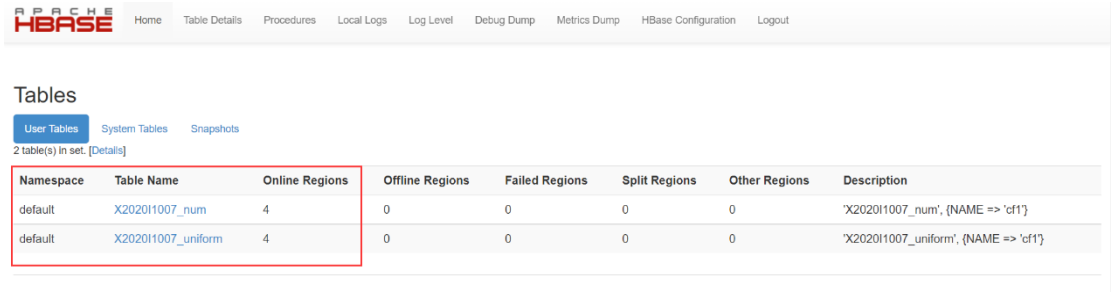
4.4.3 在 Manager 中查看 HBase(5’)

MRS Manager 界面，点击“HBase”服务

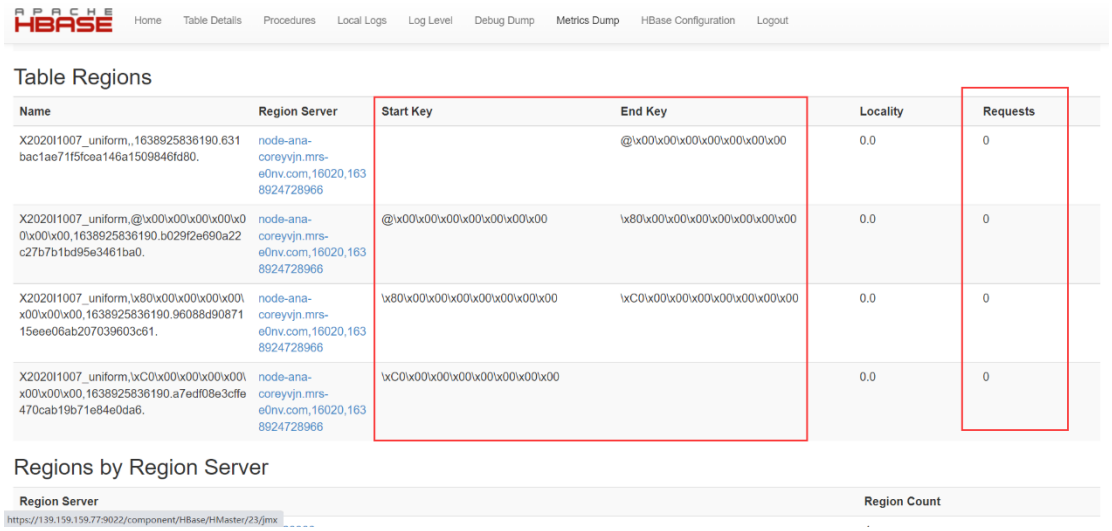
点击 HMaster(主)进入 HBase UI

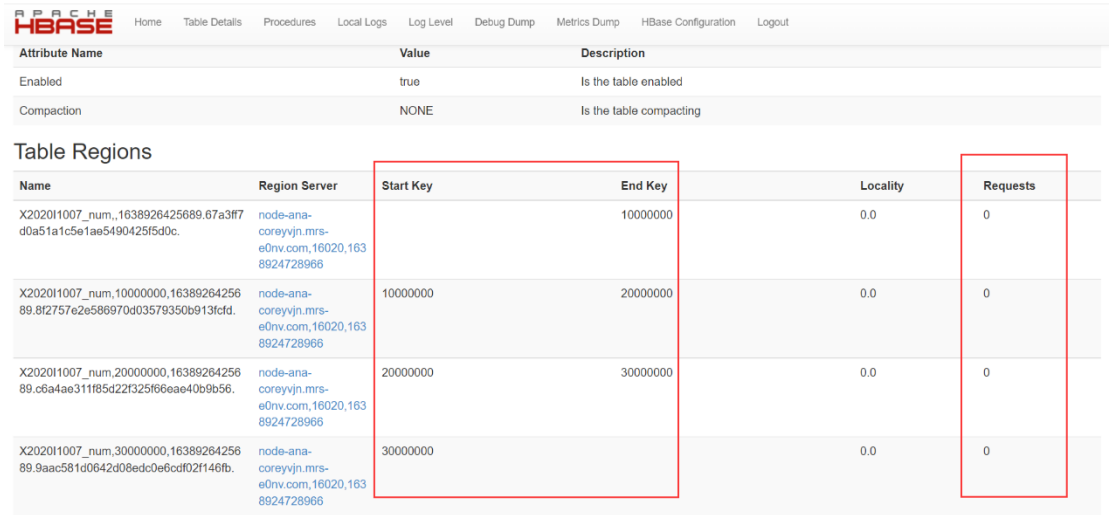


“User Tables”下点击创建好的表名“cx_table_stu02”，如“下图”：



查看该表分区情况，如下图：





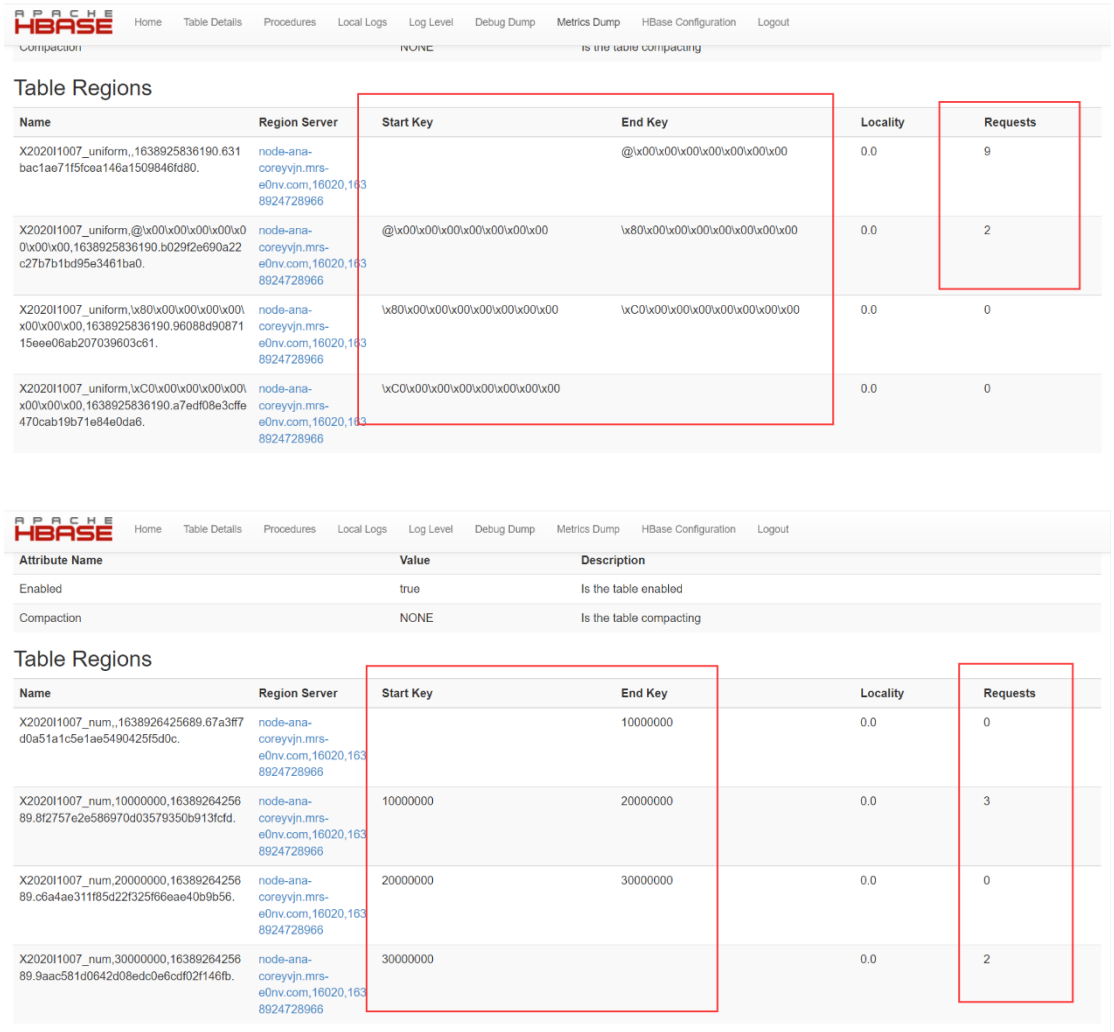
4.4.4 根据两个表的 End key 和 Start Key，选择适当的行键往两个表的不同 region 中添加任意两个数据，使得每个表至少有两个不同 region 中 Requests 不为 0（put）(5’)

```
hbase(main):004:0> put 'X2020I1007_num','10000000','cf1:student_id','82265544'
0 row(s) in 0.1310 seconds

hbase(main):005:0> put 'X2020I1007_num','10000000','cf1:gender','male'
0 row(s) in 0.0130 seconds

hbase(main):006:0> put 'X2020I1007_num',{STARTROW=>'',ENDROW=>'10000000'},'cf1:student_id','
2248866'
0 row(s) in 0.0190 seconds

hbase(main):007:0> scan 'X2020I1007_num'
ROW                    COLUMN+CELL
 10000000              column=cf1:gender, timestamp=1638930988191, value=male
 10000000              column=cf1:student_id, timestamp=1638930957663, value=82265544
 STARTROWSTOPROW10000000 column=cf1:student_id, timestamp=1638931146766, value=12248866
2 row(s) in 0.0290 seconds
```



APACHE HBASE Home | Table Details | Procedures | Local Logs | Log Level | Debug Dump | Metrics Dump | HBase Configuration | Logout

Table Regions

Name	Region Server	Start Key	End Key	Locality	Requests
X2020I1007_uniform,,1638925836190.631bac1ae71f5fcea146a1509846fd80.	node-ana-coreyvjn.mrs-e0nv.com,16020,1638924728966		@\x00\x00\x00\x00\x00\x00\x00	0.0	9
X2020I1007_uniform,@\x00\x00\x00\x00\x00\x00\x00,1638925836190.b029f2e690a22c27b7b1bd95e3461ba0.	node-ana-coreyvjn.mrs-e0nv.com,16020,1638924728966	@\x00\x00\x00\x00\x00\x00\x00	\x80\x00\x00\x00\x00\x00\x00\x00	0.0	2
X2020I1007_uniform,\x80\x00\x00\x00\x00\x00\x00,1638925836190.96088d9087115eee06ab207039603c61.	node-ana-coreyvjn.mrs-e0nv.com,16020,1638924728966	\x80\x00\x00\x00\x00\x00\x00\x00	\xC0\x00\x00\x00\x00\x00\x00\x00	0.0	0
X2020I1007_uniform,\xC0\x00\x00\x00\x00\x00\x00,1638925836190.a7edf08e3cffe470cab19b71e84e0da6.	node-ana-coreyvjn.mrs-e0nv.com,16020,1638924728966	\xC0\x00\x00\x00\x00\x00\x00\x00		0.0	0

APACHE HBASE Home | Table Details | Procedures | Local Logs | Log Level | Debug Dump | Metrics Dump | HBase Configuration | Logout

Attribute Name	Value	Description
Enabled	true	Is the table enabled
Compaction	NONE	Is the table compacting

Table Regions

Name	Region Server	Start Key	End Key	Locality	Requests
X2020I1007_num,,1638926425689.67a3ff7d0a51a1c5e1ae5490425f5d0c.	node-ana-coreyvjn.mrs-e0nv.com,16020,1638924728966		10000000	0.0	0
X2020I1007_num,10000000,1638926425689.8f2757e2e586970d03579350b913fcfd.	node-ana-coreyvjn.mrs-e0nv.com,16020,1638924728966	10000000	20000000	0.0	3
X2020I1007_num,20000000,1638926425689.c6a4ae311f85d22f325f66eae40b9b56.	node-ana-coreyvjn.mrs-e0nv.com,16020,1638924728966	20000000	30000000	0.0	0
X2020I1007_num,30000000,1638926425689.9aac581d0642d08edc0e6cdf02f146fb.	node-ana-coreyvjn.mrs-e0nv.com,16020,1638924728966	30000000		0.0	2

4.4.5 删除所有表(5’)

```
hbase(main):010:0> disable 'X2020I1007_uniform'
2021-12-08 10:44:23,456 INFO  [main] client.HBaseAdmin: Started disable of X2020I1007_uniform
2021-12-08 10:44:27,734 INFO  [main] client.HBaseAdmin: Disabled X2020I1007_uniform
0 row(s) in 4.3950 seconds

hbase(main):011:0> drop 'X2020I1007_uniform'
2021-12-08 10:44:45,719 INFO  [main] client.HBaseAdmin: Deleted X2020I1007_uniform
0 row(s) in 1.2560 seconds

hbase(main):012:0> disable 'X2020I1007_num'
2021-12-08 10:44:59,270 INFO  [main] client.HBaseAdmin: Started disable of X2020I1007_num
2021-12-08 10:45:03,537 INFO  [main] client.HBaseAdmin: Disabled X2020I1007_num
0 row(s) in 4.3520 seconds

hbase(main):013:0> drop 'X2020I1007_num'
2021-12-08 10:45:16,403 INFO  [main] client.HBaseAdmin: Deleted X2020I1007_num
0 row(s) in 1.2550 seconds

hbase(main):014:0> exists 'X2020I1007_uniform'
Table X2020I1007_uniform does not exist


0 row(s) in 0.0140 seconds

hbase(main):015:0> exists 'X2020I1007_num'
Table X2020I1007_num does not exist
0 row(s) in 0.0130 seconds
```

4.5 hive 初探(10’)

4.5.1 准备 file1.txt，内容为”hello hust”，file2.txt，内容为”hello 学号”（vim）(3’)

```
[root@node-master1tUtV ~]# vi file1.txt
[root@node-master1tUtV ~]# vi file2.txt
```

```
hello hust
~
~
~
~
~
~
~
~
```

```
hello X2020I1007
~
~
~
~
~
~
~
~
```

4.5.2 将创建的文件移动到 HDFS 中/test 文件夹内（见上次实验）(3’)

```
[root@node-master1tUtV ~]# hdfs dfs -mkdir /test
[root@node-master1tUtV ~]# hdfs dfs -put file1.txt /test
[root@node-master1tUtV ~]# hdfs dfs -put file2.txt /test
[root@node-master1tUtV ~]# hdfs dfs -ls /test
Found 2 items
-rw-r--r--   1 root ficommon         11 2021-12-08 11:18 /test/file1.txt
-rw-r--r--   1 root ficommon         17 2021-12-08 11:18 /test/file2.txt
[root@node-master1tUtV ~]#
```

4.5.3 在 hive 中创建表，tablename 替换为学号(1’)

create table tablename(line string);

```
hive> create table X2020I1007(line string);
OK
Time taken: 2.139 seconds
```

4.5.4 加载 hdfs 中的数据到 hive 中(1’)

load data inpath 'hdfs:///test' overwrite into table tablename;

```
hive> load data inpath 'hdfs:///test' overwrite into table X2020I1007
    > ;
Loading data to table default.x2020i1007
chgrp: changing ownership of 'hdfs://hacluster/user/hive/warehouse/x2020i1007': User null does not belong to hive
OK
Time taken: 0.732 seconds
```

4.5.5 通过 HiveQL 语句创建词频统计表(1’)

```
create table word_count as
select word,count(1) as count from
    (select explode(split(line, ' '))as word from tablename) w
    group by word
    order by word;
```

```
hive> create table word_count as
    > select word,count(1) as count from
    > (select explode(split(line, ' '))as word from X2020I1007) w
    > group by word
    > order by word;
```

```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.3 sec   HDFS Read: 8324 HDFS Write: 172
SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 3.67 sec   HDFS Read: 5323 HDFS Write: 102
 SUCCESS
Total MapReduce CPU Time Spent: 8 seconds 970 msec
OK
Time taken: 80.444 seconds
```

4.5.6 通过 HiveQL 语句创建词频统计表(1')

select * from word_count;

```
hive> select * from word_count;
OK
2021-12-08 11:41:46,905 INFO  [46f2ed3d-26f0-4cb2-b8b5-a96b90155240 main] mapred.FileInputFormat: Total input files to process : 1
X2020I1007      1
hello   2
hust    1
Time taken: 0.148 seconds, Fetched: 3 row(s)
hive>
```

五：实验总结(10')

本次实验目的在于实现 MapReduce 中的 Hbase 分布式数据库的功能和基于 Hadoop 的 Hive 数据仓库分析系统的操作。实验前期，总是因为华为云的 MapReduce 大数据并行处理平台出问题，一是卡在集群创建安装时候；二是卡在无法打开 MapReduce MRS 的集群管理 manager;三是进入 Cloud Shell 云平台后,在 Hbase Shell 模式下创建表报 error。到后面使用命令提示符才能顺利进入 Hbase Shell 开始实验内容。后面就是抓紧时间把实验进度赶上来。希望华为云以后不要再出各种各样的问题，太闹心和浪费时间以及金钱了。总的来说，在完成本次实验后，我感到受益匪浅，不仅对 Hbase 分布式存储系统的运用和运作原理有了更深入的了解，同时还学到了新的知识“Hive 数据仓库分析系统操作和运作模式”，受益匪浅。我会在接下来的实验里再接再厉。也感谢助教的指导，否则我是不可能自己找到方法去解决华为云各种出错的问题。

附录：

1 容易出错的地方:多打或少打空格；英文引号打成成文分号。

2 HBase 基本命令:http://c.biancheng.net/view/3587.html

3 牢记删除资源