### [redis集群环境搭建以及java中jedis客户端集群代码实现](http://ihenu.iteye.com/blog/2267881)

**博客分类：**

* [redis](http://ihenu.iteye.com/category/346231)

[redis](http://www.iteye.com/blogs/tag/redis)[linux](http://www.iteye.com/blogs/tag/linux)[rubygems](http://www.iteye.com/blogs/tag/rubygems)[java](http://www.iteye.com/blogs/tag/java)

      最近项目中用到了redis，所以就找了相关的资料学习了，然后搭建了redis集群，以及客户端的java代码实现。由于对linux不太熟悉，所以在Linux上搭建redis集群的时候碰到了很多问题，下面就大概总结下。

redis相关网站：

官网： [http://redis.io](http://ihenu.iteye.com/admin/blogs/ http:/redis.io" \t "http://ihenu.iteye.com/blog/_blank)     中文网：[http://www.redis.cn/](http://www.redis.cn/" \t "http://ihenu.iteye.com/blog/_blank)    文档：[http://redisdoc.com/](http://redisdoc.com/" \t "http://ihenu.iteye.com/blog/_blank)

操作系统：centos 6.3

redis版本：3.0.6

java客户端版本： jedis 2.7.2

redis客户端图形用户界面：RedisDesktopManager [下载地址](https://github.com/uglide/RedisDesktopManager/releases" \t "http://ihenu.iteye.com/blog/_blank)

**1.redis服务端集群搭建步骤：**

  1.下载redis安装包，进行解压安装

  2.安装ruby、rubygems install ruby ，安装ruby的原因是，在进行集群的时候，使用的是ruby语言工具实现的，所以在集群之前首先需要搭建ruby的环境

  3.在上述步骤完成之后，便可以搭建集群环境，redis提供了两种集群搭建方法，执行脚本方法(安装包下面的util包中)和手动搭建。

  注意：

  1.在集群的时候，如果是远端客户端访问redis服务端，那么在分片的时候，需要使用Ip进行分片，下面会详细说

  2.在创建每个节点的时候，不要只用redis-server ，使用绝对路径下的redis-server xxx

  具体的安装步骤如下：[http://blog.csdn.net/xu470438000/article/details/42971091](http://blog.csdn.net/xu470438000/article/details/42971091" \t "http://ihenu.iteye.com/blog/_blank)

  附件：在安装ruby的时候，需要gemredis，下载地址在下面。

**2.客户端(java)：**

注意：

1.本文的客户端使用的是java,官网中对于java客户端也提供了不少的client,但是本文使用的是官方推荐的jedis。

2.在项目开发中，一般情况下都会用到spring来管理应用，本文也是如此，spring 本身也提供了对redis的集成支持，具体的网址：http://projects.spring.io/spring-data-redis，

 但是好像目前spring-data-redis不提供集群的功能，所以本文没有使用它，而是使用了原装的jedis来进行开发，如果在项目中没有用到集群的功能，则可以使用spirng-data-redis。

下面是具体的代码实现

1.maven依赖

Java代码  [IMG_256](http://ihenu.iteye.com/blog/javascript:void())

1. <dependency>
2. <groupId>redis.clients</groupId>
3. <artifactId>jedis</artifactId>
4. <version>2.7.2</version>
5. </dependency>

2.applicationContext.xml中的配置

Java代码  [IMG_257](http://ihenu.iteye.com/blog/javascript:void())

1. <!-- jedis cluster config -->
2. <bean name="genericObjectPoolConfig" class="org.apache.commons.pool2.impl.GenericObjectPoolConfig" >
3. <property name="maxWaitMillis" value="-1" />
4. <property name="maxTotal" value="1000" />
5. <property name="minIdle" value="8" />
6. <property name="maxIdle" value="100" />
7. </bean>
9. <bean id="jedisCluster" class="com.besttone.subscribe.util.JedisClusterFactory">
10. <property name="addressConfig">
11. <value>classpath:redis-config.properties</value>
12. </property>
13. <property name="addressKeyPrefix" value="address" />
15. <property name="timeout" value="300000" />
16. <property name="maxRedirections" value="6" />
17. <property name="genericObjectPoolConfig" ref="genericObjectPoolConfig" />
18. </bean>

3.JedisClusterFactory实现类

Java代码  [IMG_258](http://ihenu.iteye.com/blog/javascript:void())

1. public class JedisClusterFactory implements FactoryBean<JedisCluster>, InitializingBean {
3. private Resource addressConfig;
4. private String addressKeyPrefix ;
6. private JedisCluster jedisCluster;
7. private Integer timeout;
8. private Integer maxRedirections;
9. private GenericObjectPoolConfig genericObjectPoolConfig;
11. private Pattern p = Pattern.compile("^.+[:]\\d{1,5}\\s\*$");
13. @Override
14. public JedisCluster getObject() throws Exception {
15. return jedisCluster;
16. }
18. @Override
19. public Class<? extends JedisCluster> getObjectType() {
20. return (this.jedisCluster != null ? this.jedisCluster.getClass() : JedisCluster.class);
21. }
23. @Override
24. public boolean isSingleton() {
25. return true;
26. }


30. private Set<HostAndPort> parseHostAndPort() throws Exception {
31. try {
32. Properties prop = new Properties();
33. prop.load(this.addressConfig.getInputStream());
35. Set<HostAndPort> haps = new HashSet<HostAndPort>();
36. for (Object key : prop.keySet()) {
38. if (!((String) key).startsWith(addressKeyPrefix)) {
39. continue;
40. }
42. String val = (String) prop.get(key);
44. boolean isIpPort = p.matcher(val).matches();
46. if (!isIpPort) {
47. throw new IllegalArgumentException("ip 或 port 不合法");
48. }
49. String[] ipAndPort = val.split(":");
51. HostAndPort hap = new HostAndPort(ipAndPort[0], Integer.parseInt(ipAndPort[1]));
52. haps.add(hap);
53. }
55. return haps;
56. } catch (IllegalArgumentException ex) {
57. throw ex;
58. } catch (Exception ex) {
59. throw new Exception("解析 jedis 配置文件失败", ex);
60. }
61. }
63. @Override
64. public void afterPropertiesSet() throws Exception {
65. Set<HostAndPort> haps = this.parseHostAndPort();
67. jedisCluster = new JedisCluster(haps, timeout, maxRedirections,genericObjectPoolConfig);
69. }
70. public void setAddressConfig(Resource addressConfig) {
71. this.addressConfig = addressConfig;
72. }
74. public void setTimeout(int timeout) {
75. this.timeout = timeout;
76. }
78. public void setMaxRedirections(int maxRedirections) {
79. this.maxRedirections = maxRedirections;
80. }
82. public void setAddressKeyPrefix(String addressKeyPrefix) {
83. this.addressKeyPrefix = addressKeyPrefix;
84. }
86. public void setGenericObjectPoolConfig(GenericObjectPoolConfig genericObjectPoolConfig) {
87. this.genericObjectPoolConfig = genericObjectPoolConfig;
88. }
90. }

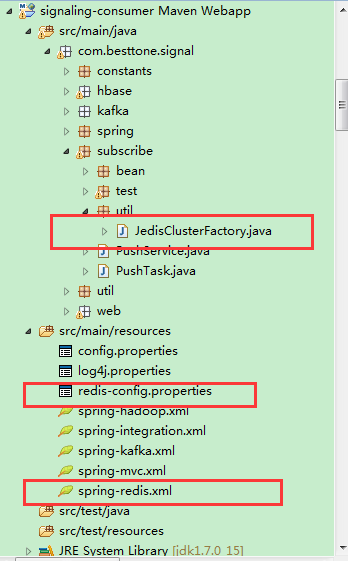
4.redis-config.properties文件

   这是一个集群环境，六个节点(不同端口)，三个master ,三个slaver

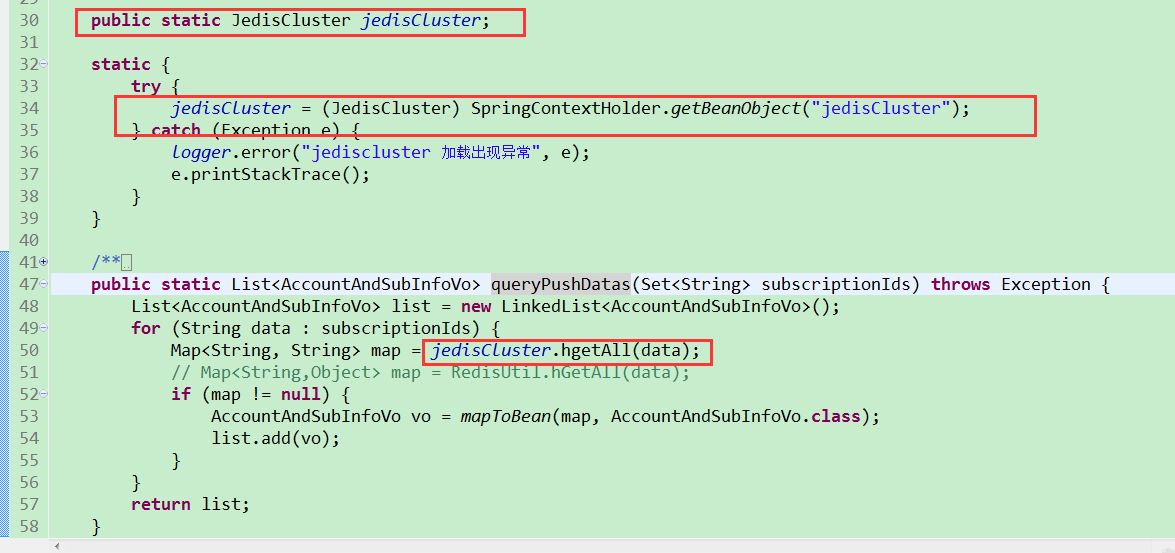
Java代码  [IMG_259](http://ihenu.iteye.com/blog/javascript:void())

1. address1=192.168.30.139:7000
2. address2=192.168.30.139:7001
3. address3=192.168.30.139:7002
4. address4=192.168.30.139:7003
5. address5=192.168.30.139:7004
6. address6=192.168.30.139:7005

5.项目目录图

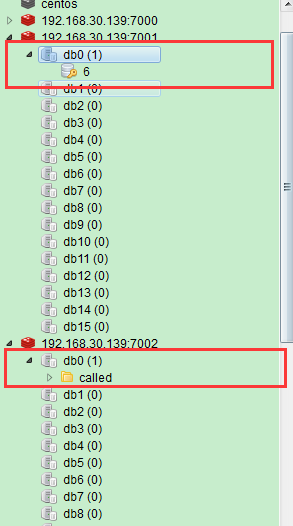


6.代码中使用(此代码为从redis中获取相关信息)

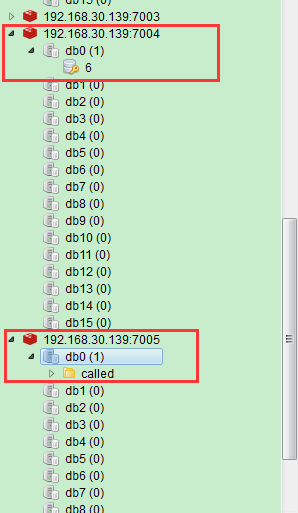


ok,运行之后，会发现redis会根据不同的key,把它们放入到不同的节点中，如下图

7.三个master节点中的数据



 8.三个slave节点中的数据



实践过程中碰到的问题：

1.在一切准备好了之后，在操作redis的时候，却报错误：**Too many Cluster redirections**

由于，我是windows开发环境，在本机开了一个虚拟机，然后在虚拟机中搭建的linux集群环境，本机的ip和虚拟机中的ip不相同，所以报这个错误，

解决方法：在redis集群搭建过程中，在为每个节点分hash槽的时候，执行如下代码(其中，xxx为集群环境中的ip)：

Java代码  [IMG_264](http://ihenu.iteye.com/blog/javascript:void())

1. ./redis-trib.rb create --replicas 1 xxx.xxx.xxx.xxx:7000  xxx.xxx.xxx.xxx:7001 xxx.xxx.xxx.xxx:7002 xxx.xxx.xxx.xxx:7003 xxx.xxx.xxx.xxx:7004 xxx.xxx.xxx.xxx:7005./redis-trib.rb create --replicas 1 127.0.0.1:7000  127.0.0.1:7001 127.0.0.1:7002 127.0.0.1:7003 127.0.0.1:7004 127.0.0.1:7005

2.在一切搭建好后，我们使用redis-cli登陆，当命令：set msg XXX时，报错：

Java代码  [IMG_265](http://ihenu.iteye.com/blog/javascript:void())

1. (error) MOVED 6257 192.168.30.141:7001

 解决：在使用客户端登陆时：加上-c参数，即：

Java代码  [IMG_266](http://ihenu.iteye.com/blog/javascript:void())

1. redis-cli -c -h 192.168.30.141 -p 7000

 ok,以上满足使用，结束！！

**----------------------------------------------------------------------- 下面是我前一段时间看redis资料，总结的一些东东，记录下来 -------------------------------------------------------------------------------------------**

1.redis是什么？

redis（remote dictionary server）:是一个以key-value形式存储于内存中的数据库.提供了 String / List / Set / Sort Set /Hash 五种数据结构。

服务器在断电之后，仍然可以恢复到断电之前的状态。

资料： 官网 ： http://redis.io   中文网:     http://www.redis.cn/         相关文档： http://redisdoc.com/

2.redis特点?

线程模型：单线程-多路复用io模型

性能高：支持读 11万/秒  ，  写 8万/秒

存储： 内存 ;  RDB文件(二进制安全的真实数据) ;  AOF文件（客户端的命令集合）

事务： 支持事务（每个客户端串行执行命令，其他客户端处于阻塞状态）

发布/订阅模式：  功能？ 什么场景使用？？

3.redis数据类型

String:动态字符串(每个key都是一个String)

编码方式：int / raw() /embstr

应用场景：普通的string场景

List:列表结构，有序可重复的结构。它拥有队列的特性。

编码方式：ziplist / linkedlist (如果数据量较小，且是数字或者字符串，则内部结构为 ziplist)

应用场景：普通的集合数据

Set:集合结构，不重复的集合结构。

编码方式：intset(整数集合) / hashtable

应用场景：普通的非重复集合数据;支持取交集、取并集等操作

Sort Set:有序集合结构，和Set比较起来，它是有序的。

编码方式：ziplist / skiplist

应用场景：有序不重复的集合数据

Hash:哈希结构，存储多个key:value的结构,此种结构可以存储对象 ；  如  HMSET  user(key)  username value1 password value2

编码方式：ziplist / hashtable

应用场景： 从关系型数据库去出一条数据，就可以让入到此种结构中

4.内存优化

redis提供内存回收策略，根据使用的情况可以选择适当的回收策略

redis提供内存共享策略，服务器启动时，会自动创建0-9999的数字对象，其他地方使用，可以直接引用。

本质：对内存的操作，其实是在每一个redis对象结构内都有一个count的属性，该属性记录了这个对象被引用的次数，如果为0，那么在内存回收时将回收该空间。

save参数调整：当满足条件时，触发SAVE命令，持久化到RDB文件

appendonly参数： 默认no ,若yes，则开启AOF文件持久化； BGREWRITEAOF  命令 持久化。其中appendsync参数调整具体的持久化策略，默认为每秒

内存回收策略：

5.发布订阅模式

6.数据过期设置

可以根据业务需求，将某些数据进行日期设置

7.事务

单线程处理所有客户端发来的请求，所以当有一个客户端在执行，其他客户端只能处于阻塞态。只有当前客户端请求完毕，其他客户端才能请求

8.数据存储

RDB文件模式(快照)：该模式存储的是真实数据，SAVE /BGSAVE 命令 可以将内存中的数据存储到磁盘文件中。SAVE和BGSAVE区别在于，SAVE是同步命令，

            即当执行该命令，其他客户端处于阻塞状态；而BGSAVE 命令则是开启一个子进程处理，不会影响主进程操作。

AOF文件模式：该模式存储的是服务器执行的命令集合，BGREWRITEAOF 命令。 该模式是appendonly参数控制，若打开，则会将数据同步到aof文件中

特点：该模式下，会在服务器端开辟一段缓冲内存来存储最近时间单位的命令，所以该点要注意。同样，它也是子进程进行执行

注意：AOF模式的更新频率比RDB高，若开启AOF模式的情况下，优先载入AOF文件内容

9.数据恢复策略

若RDB模式开启：重启服务器只加载rdb文件内容

若AOF模式开启：重启服务器只加载aof文件内容

若两者都开启：只加载aof文件内容

10主从复制

功能：数据备份，读写分离(测试环境，主服务器写，从服务器读)

步骤：在从服务端器执行： slaveof <masterip> <masterport> 即可维持关系；配置文件中也可以

特点：

1.master可以有多个slave

2.除了多个slave连到相同的master外，slave也可以连接其他slave形成图状结构

3.主从复制不会阻塞master。也就是说当一个或多个slave与master进行初次同步数据时，master可以继续处理client发来的请求。相反slave在初次同步数据时则会阻塞不能处理client的请求。

4.主从复制可以用来提高系统的可伸缩性,我们可以用多个slave 专门用于client的读请求，比如sort操作可以使用slave来处理。也可以用来做简单的数据冗余

5.可以在master禁用数据持久化，只需要注释掉master 配置文件中的所有save配置，然后只在slave上配置数据持久化。

  6.主服务器可以关闭持久化功能（注释掉save参数）

11.sentinel(监测系统)

本质：是一个运行在特殊模式下的redis服务器。

功能：监控运行在多机上的主redis服务器，若有某一台主服务器出现故障，将自动把其他正常的从服务器切换为主服务器，代替出现故障主服务器的工作。

特点：

1.不发挥数据库的功能(所有对key以及数据类型操作的命令不能使用)

2.将会给监控的主服务器以及主服务器所属的从服务器发送命令，确认是否下线

3.会和监控同一个主服务器的其他sentinel服务器通信，作用是在共同判断所监控的主服务器的状态

4.根据多个sentinel判断的主服务器状态，来决定是否要进行主从切换，故障转移等

转移：sentinel监控的主服务器配置参数要在 sentinel.conf 文件中配置，启动时加载

具体配置安装步骤：

1.http://blog.csdn.net/pi9nc/article/details/17735653

2.http://blog.csdn.net/luyee2010/article/details/9385155

12.集群

功能：将众多的key-value集合存在多个节点上，当某一个节点出现障碍，不影响整个集群的功能。

涉及到的关键词：

节点：一个端口的redis服务便是一个节点

槽指派（集群将整个系统分为16384个hash槽）：这16384个槽位要全部分布在集群中的主节点上。

重新分片：若某个主节点故障了，将该主节点的槽位分配到其他可以用的主节点上。

上线/下线状态： 是否全部的槽位都分布在节点上。

特点：

1.如果某个节点要集群，必须要设置cluster-enabled yes

2.每个节点都有这16384个槽位所属的节点信息，如果值没有正确进入槽位，那么该节点会提示系统将信息放入正确槽位。重定向的过程会出现一个面向客户端隐藏的MOVED错误

3.集群在线状态也可以进行重新分片

4.集群中的主节点用户处理客户端命令，从节点用于复制主节点的数据，主节点下线时，从节点代替主节点的工作

//注意：目前官方提供的集群功能仍处于内测版本。

13.redis基准

redis自带的redis-benchmark 工具，支持各种参数进行性能测试

特点：

1.可以模拟多个客户端处理任意个请求

2.可以测试仅仅少数使用的命令等

注意：测试发现，linux环境下部署的redis服务器性能远高于windows下部署的redis服务器性能， 不在一个层级上面

14.关系数据库模型的转换

关系型数据库表结构：user表 (uid username password birthday )

在redis中可以这样存在：

1.主键： SET user:uid 1  、  GET user:1

2.其他字段：SET user:uid:username  GET user:5:username ( 5 是通过参数值传进来的)

3.表数据也可以存在hash结构中:      HMSET user:uid username value1 password value2 birthday value3

15.排序

16.管道

功能：客户端一次可以传送多个命令到服务器，减少往返时延。大大提高性能。

17.优化

redis提供一些简单的内存优化策略，如过期数据清除，内存数据共享，

18.持久化

* [redis-3.2.1.rar](http://dl.iteye.com/topics/download/dc77cd06-314f-361e-82a1-52872e4a01b4) (69.2 KB)
* 下载次数: 17
* [Redis介绍.ppt.png](http://dl.iteye.com/topics/download/4150b307-cd20-37bd-acd9-1905b296f1ab) (836 KB)
* 下载次数: 1
* [查看图片附件](http://ihenu.iteye.com/blog/2267881)