Redis6-06- 集群

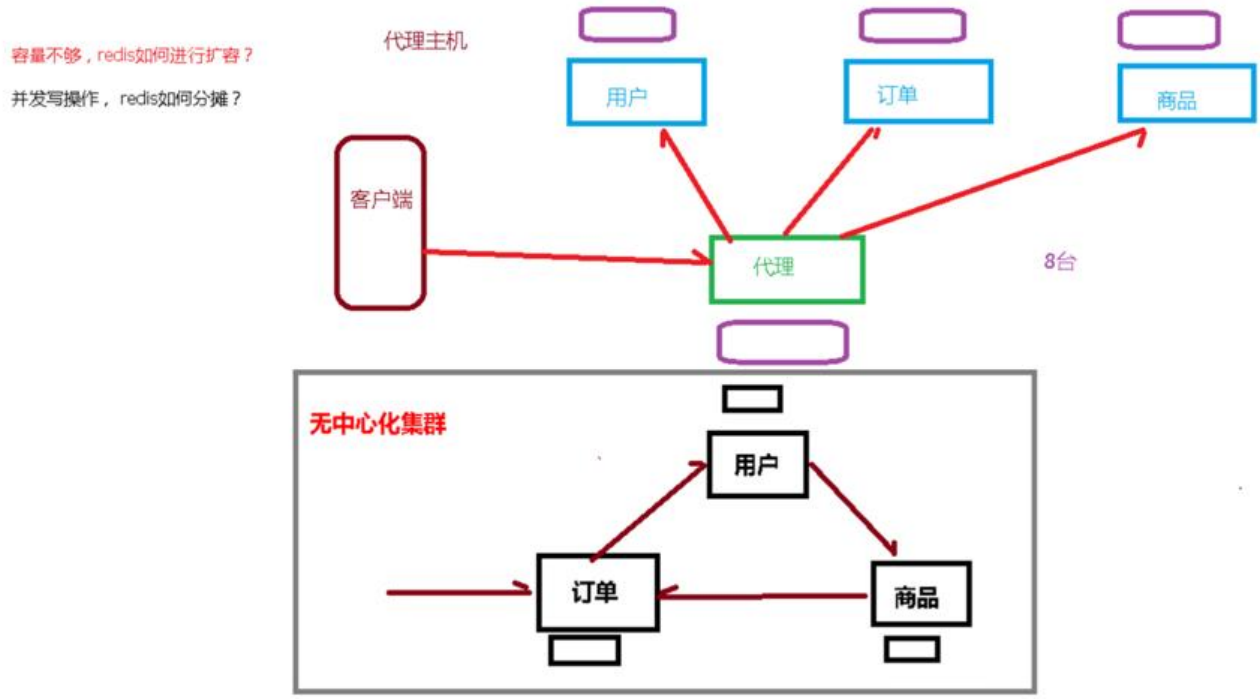
作者：Anileh
原文链接：https://ld246.com/article/1652274650015
来源网站：链滴

Redis集群

1.问题引入

当容量不足时，Redis如何进行扩容?

并发写操作，Redis如何进行分摊?



2. Redis cluster配置修改

cluster-enable yes：打开集群模式

cluster-config-file nodes-6379.conf：设定节点配置文件

cluster-node-timeout 15000：设定节点失联时间，超过此时间集群自动进行主从切换

```
include /home/bigdata/redis.conf
port 6379
pidfile "/var/run/redis_6379.pid"
dbfilename "dump6379.rdb"
dir "/home/bigdata/redis_cluster"
logfile "/home/bigdata/redis_cluster/redis_err_6379.log"
cluster-enabled yes
cluster-config-file nodes-6379.conf
cluster-node-timeout 15000
```

3.实验

创建6个实例： 6379,6380,6381,6389,6390,6391

（1）配置基本信息

开启daemonize yes

Pid文件名字

指定端口

Log文件名字

Dump.rdb名字

Appendonly 关掉或者换名字

（2）redis cluster配置修改

cluster-enable yes：打开集群模式

cluster-config-file nodes-6379.conf：设定节点配置文件

cluster-node-timeout 15000：设定节点失联时间，超过此时间集群自动进行主从切换

```
include /home/bigdata/redis.conf
port 6379
pidfile "/var/run/redis_6379.pid"
dbfilename "dump6379.rdb"
dir "/home/bigdata/redis_cluster"
logfile "/home/bigdata/redis_cluster/redis_err_6379.log"
cluster-enabled yes
cluster-config-file nodes-6379.conf
cluster-node-timeout 15000
```

（4）合并成集群

相应地更改配置文件内容，然后启动redis服务

```
[root@zy myredis]# ps -ef|grep redis
root      14256  13631  0 20:59 pts/1    00:00:00 grep --color=auto redis
[root@zy myredis]# ll
总用量 76
-rw-r--r--. 1 root root   181 12月 21 20:50 redis6379.conf
-rw-r--r--. 1 root root   181 12月 21 20:51 redis6380.conf
-rw-r--r--. 1 root root   181 12月 21 20:51 redis6381.conf
-rw-r--r--. 1 root root   181 12月 21 20:51 redis6389.conf
-rw-r--r--. 1 root root   181 12月 21 20:52 redis6390.conf
-rw-r--r--. 1 root root   181 12月 21 20:52 redis6391.conf
-rw-r--r--. 1 root root 46705 12月 21 13:57 redis.conf
-rw-r--r--. 1 root root   335 12月 21 17:07 sentinel.conf
[root@zy myredis]# redis-server redis6379.conf
[root@zy myredis]# redis-server redis6380.conf
[root@zy myredis]# redis-server redis6381.conf
[root@zy myredis]# redis-server redis6389.conf
[root@zy myredis]# redis-server redis6390.conf
[root@zy myredis]# redis-server redis6391.conf
[root@zy myredis]# ps -ef|grep redis
root      14261      1  0 20:59 ?        00:00:00 redis-server *:6379 [cluster]
root      14265      1  0 20:59 ?        00:00:00 redis-server *:6380 [cluster]
root      14269      1  0 20:59 ?        00:00:00 redis-server *:6381 [cluster]
root      14273      1  0 20:59 ?        00:00:00 redis-server *:6389 [cluster]
root      14277      1  0 20:59 ?        00:00:00 redis-server *:6390 [cluster]
root      14281      1  0 20:59 ?        00:00:00 redis-server *:6391 [cluster]
root      14285  13631  0 20:59 pts/1    00:00:00 grep --color=auto redis
```

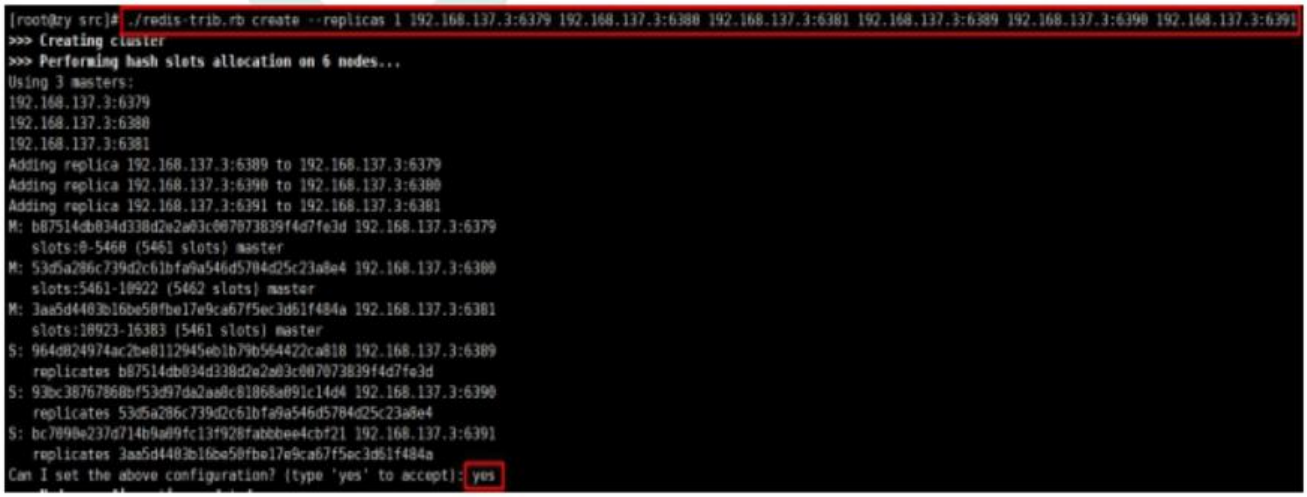
组合之前，请**确保所有redis实例启动**后，nodes-xxxx.conf文件都生成正常。

```
[root@zy myredis]# ll
总用量 100
-rw-r--r--. 1 root root     0 12月 21 20:59 appendonly.aof
-rw-r--r--. 1 root root   112 12月 21 20:59 nodes-6379.conf
-rw-r--r--. 1 root root   112 12月 21 20:59 nodes-6380.conf
-rw-r--r--. 1 root root   112 12月 21 20:59 nodes-6381.conf
-rw-r--r--. 1 root root   112 12月 21 20:59 nodes-6389.conf
-rw-r--r--. 1 root root   112 12月 21 20:59 nodes-6390.conf
-rw-r--r--. 1 root root   112 12月 21 20:59 nodes-6391.conf
-rw-r--r--. 1 root root   181 12月 21 20:50 redis6379.conf
-rw-r--r--. 1 root root   181 12月 21 20:51 redis6380.conf
-rw-r--r--. 1 root root   181 12月 21 20:51 redis6381.conf
-rw-r--r--. 1 root root   181 12月 21 20:51 redis6389.conf
-rw-r--r--. 1 root root   181 12月 21 20:52 redis6390.conf
-rw-r--r--. 1 root root   181 12月 21 20:52 redis6391.conf
-rw-r--r--. 1 root root 46705 12月 21 13:57 redis.conf
-rw-r--r--. 1 root root   335 12月 21 17:07 sentinel.conf
```

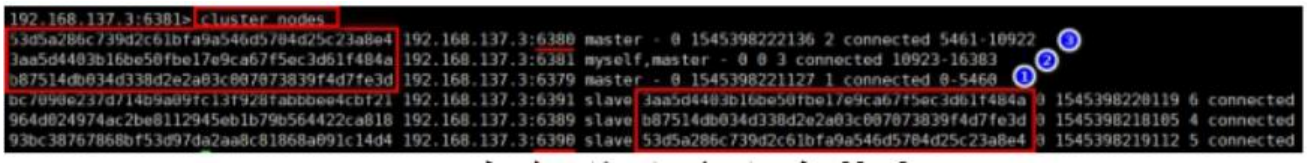
合并：cd /opt/redis-6.2.1/src

redis-cli --cluster create --cluster-replicas 1 192.168.11.101:6379 192.168.11.101:6380 192.168.11.101:6381 192.168.11.101:6389 192.168.11.101:6390 192.168.11.101:6391

--replicas 1 采用最简单的方式配置集群，一台主机，一台从机，正好三组。



可以通过cluster nodes查看集群信息



4. 集群slots

一个Redis集群包干16384个插槽(hash slot)，数据库中的每个键都属于这些插槽

集群使用公式**CRC16(key) %16384**来计算key属于那个值，其中CRC16(key)用于计算key的CRC16验和

集群中的每个节点负责处理一部分插槽。 举个例子， 如果一个集群可以有主节点， 其中:

节点 A 负责处理 0 号至 5460 号插槽。

节点 B 负责处理 5461 号至 10922 号插槽。

节点 C 负责处理 10923 号至 16383 号插槽。

5. 在集群中录入值

在redis-cli每次录入、查询键值，redis都会计算出该key的所在插槽，如果不是该客户端对应服务器插槽，redis会报错，并告知应前往的redis实例地址和端口。

redis-cli客户端提供了** -c 参数实现自动重定向**

如 redis-cli -c –p 6379 登入后，再录入、查询键值对可以自动重定向。

不在一个slot下的键值，是不能使用mget,mset等多键操作。

```
192.168.137.3:6381> mset k1 v1 k2 v2 k3 v3
(error) CROSSSLOT Keys in request don't hash to the same slot
```

原文链接：Redis6-06- 集群

可以通过{}来定义组的概念，从而使key中{}内相同内容的键值对放到一个slot中去。

```
192.168.137.3:6381> mset k1{cust} v1 k2{cust} v2 k3{cust} v3
-> Redirected to slot [4847] located at 192.168.137.3:6379
OK
```

6. 查询集群中的值

CLUSTER GETKEYSINSLOT <slot> <count> <slot><count>　　返回 count 个 slot 槽中的键。

```
192.168.137.3:6379> cluster keyslot cust
(integer) 4847
192.168.137.3:6379> cluster countkeysinslot 4847
(integer) 3
192.168.137.3:6379> cluster getkeysinslot 4847 10
1) "k1{cust}"
2) "k2{cust}"
3) "k3{cust}"
```

7. 故障恢复

- 如果主节点下线，从节点自动升级为主节点，注意超时时间：15秒

```
127.0.0.1:6380> cluster nodes
b87514db034d338d2e2a03c007073839f4d7fe3d 192.168.137.3:6379 master,fail - 1545399235089 1545399231958 1 disconnected
964d024974ac2be8112945eb1b79b564422ca818 192.168.137.3:6389 master - 0 1545399381385 7 connected 0-5460
bc7090e237d714b9a09fc13f928fabbbee4cbf21 192.168.137.3:6391 slave 3aa5d4403b16be50fbe17e9ca67f5ec3d61f484a 0 1545399382313 6 connected
53d5a286c739d2c61bfa9a546d5704d25c23a8e4 192.168.137.3:6380 myself,master - 0 0 2 connected 5461-10922
93bc38767868bf53d97da2aa8c81868a091c14d4 192.168.137.3:6390 slave 53d5a286c739d2c61bfa9a546d5704d25c23a8e4 0 1545399383320 5 connected
3aa5d4403b16be50fbe17e9ca67f5ec3d61f484a 192.168.137.3:6381 master - 0 1545399379288 3 connected 10923-16383
```

- 主节点恢复后将变为从机

```
127.0.0.1:6380> cluster nodes
b87514db034d338d2e2a03c007073839f4d7fe3d 192.168.137.3:6379 slave 964d024974ac2be8112945eb1b79b564422ca818 0 1545399515482 7 connected
964d024974ac2be8112945eb1b79b564422ca818 192.168.137.3:6389 master - 0 1545399513467 7 connected 0-5460
bc7090e237d714b9a09fc13f928fabbbee4cbf21 192.168.137.3:6391 slave 3aa5d4403b16be50fbe17e9ca67f5ec3d61f484a 0 1545399511450 6 connected
53d5a286c739d2c61bfa9a546d5704d25c23a8e4 192.168.137.3:6380 myself,master - 0 0 2 connected 5461-10922
93bc38767868bf53d97da2aa8c81868a091c14d4 192.168.137.3:6390 slave 53d5a286c739d2c61bfa9a546d5704d25c23a8e4 0 1545399516488 5 connected
3aa5d4403b16be50fbe17e9ca67f5ec3d61f484a 192.168.137.3:6381 master - 0 1545399514474 3 connected 10923-16383
```

- 如果某一段插槽的所有主从节点全部宕机，redis服务是否能够继续?

cluster-require-full-coverage yes：整个集群都挂掉

cluster-require-full-coverage no：该插槽的数据全无法使用也无法储存

7. 集群的Jedis开发

如果连接的不是主机，集群会自动切换到主机。主机写、从机读。

无中心化主从集群：无论从哪台主机写数据，其他主机上都能读到数据。

```
public class JedisClusterTest {
    public static main(String[] args){
```

```
        Set<HostAndPort> set = new HashSet<HostAndPort>();
        set.add(new HostAndPort("192.168.31.211", 6379);
        JedisCluster jedisCluster = new JedisCluster(set);
        jedisCluster.set("k1", "v1");
        System.out.println(jedisCluster.get("k1");
    }
}
```

8. Redis 集群的评价

好处：

- 实现扩容
- 分摊压力
- 无中心化配置相对简单

不足：

- 不支持多键操作
- 不支持多键的Redis事务和lua脚本