目录

[Redis学习笔记 2](#_Toc22884)

[linux系统下 装 redis 2](#_Toc22909)

[Redis 的启动，链接，关闭 3](#_Toc2997)

[Redis操作 3](#_Toc7984)

[Redis 高级应用 4](#_Toc24906)

[Redis持久化 5](#_Toc29837)

[Redis持久化 之 RDB方式 5](#_Toc125)

[Redis 持久化 之 AOF 方式 6](#_Toc21550)

[Java链接Redis 7](#_Toc22666)

[Redis 5种数据类型 8](#_Toc23388)

[Hash相关操作 8](#_Toc14498)

[List 操作 9](#_Toc28492)

[集合 set 操作 9](#_Toc8397)

[Redis集群 10](#_Toc2304)

[Redis单机多节点集群 10](#_Toc8883)

[Redis多机多节点集群 10](#_Toc17772)

# Redis学习笔记

Redis 是 NoSQL，属于非关系型数据库

Redis 是 key value的形式

具有高并发读写

海量数据高效率存储和访问

高可扩展性和高可用性

Redis支持主从同步。数据可以从主服务器向任意数量的从服务器上同步，从服务器可以是关联其他从服务器的主服务器。这使得Redis可执行单层树复制。存盘可以有意无意的对数据进行写操作。由于完全实现了发布/订阅机制，使得从数据库在任何地方同步树时，可订阅一个频道并接收主服务器完整的消息发布记录。同步对读取操作的可扩展性和数据冗余很有帮助。

## linux系统下 装 redis

由于redis是c++ 写的，所以redis需要c++ 环境的支持

yum -y install gcc-c++ : 安装gcc环境

1 wget <http://download.redis.io/releases/redis-3.2.9.tar.gz> ： 下载redis

2然后解压

3 编译 ： 进入 Redis 目录 执行 make命令 （用的是gcc 环境编译）

4 安装 redis ：make PREFIX=/usr/local/redis install :安装在 /usr/local/redis此目录

ls /usr/local/redis/ : 查看 redis目录，就一个bin目录

5执行 cp redis-3.2.9/sentinel.conf /usr/local/redis/ 将配置文件拷贝到 redis目录下

6执行 cp redis-3.2.9/redis.conf /usr/local/redis/ 将配置文件拷贝到 redis目录下

7 cd /usr/local/redis/ 执行 bin/redis-server ：前台运行redis

8 后台运行 redis

修改 redis.conf : daemonize yes

进入 /usr/local/redis 执行 bin/redis-server redis.conf ：后台运行redis

9 ps -ef | grep -i redis 查看redis 启动端口

bin/redis-cli shutdown ： 关闭redis （或者在redis终端执行 shutdown）

bin/redis-cli ： 链接redis

（链接redis后 在redis的控制终端可以 用shutdown 命令，关闭redis ，接着可以用 exit 或者 quit 退出 redis控制终端）

可以直接在 redis的控制 终端 按 ctrl +c 或者 exit 或者 quit 退出redis的控制终端 ，但是 不关闭redis

## Redis 的启动，链接，关闭

启动redis

cd /usr/local/redis/

bin/redis-server ：前台运行redis

bin/redis-server redis.conf ：后台运行redis

ps -ef | grep -i redis 查看redis 启动端口

bin/redis-cli shutdown ： 关闭redis

（或者在redis终端执行 shutdown save ：保存并关闭redis ）

bin/redis-cli ： 链接redis （链接redis后 在redis的控制终端可以 用shutdown 命令，关闭redis ，接着可以用 exit 或者 quit 退出 redis控制终端）

可以直接在 redis的控制 终端 按 ctrl +c 或者 exit 或者 quit 退出redis的控制终端 ，但是 不关闭redis

## Redis操作

clear : 清屏

incr key： 将key的值加一

desc key ：将key的值减一

Incrby key 3 : 指定增长量增长

descby key 3 :指定增长量减少

set key value :增加键值对

exists key ： 判断一个key是否存在，存在返回1 否则返回0

del key1 key2 ... : 删除一系列key 成功返回1

type key: 返回key的数据类型（string set 等等。。）

keys patten: 返回匹配的可以列表

randomkey ： 随机获得一个已经存在的key

rename oldkey newkey ：该key的名字

dbsize ：返回数据库key的总数

Redis 时间相关命令

expire key 10 ：设置key的过期时间 （10 s后删除该key）

ttl key: 查询key剩余生存时间 ： 单位 s

flushdb ：清除数据库所有的键

flushall ：清除数据库所有的键

Redis 设置相关命令

config get requirepass ：（查看密码）

config set requirepass 123： 设置密码为123 （临时设置，如果要持久化就要修改文件）

auth password ：认证

CONFIG RESETSTAT ： 重置报告

INFO [section] ：查询Redis相关信息。 INFO命令可以查询Redis几乎所有的信息，其命令选项有如下

server: Redis server的常规信息

clients: Client的连接选项

memory: 存储占用相关信息

persistence: RDB and AOF 相关信息

stats: 常规统计

replication: Master/slave请求信息

cpu: CPU 占用信息统计

cluster: Redis 集群信息

keyspace: 数据库信息统计

all: 返回所有信息

default: 返回常规设置信息

## Redis 高级应用

Redis的高级应用，包括：安全性设置，主从复制，事务处理， 持久化机制， 虚拟内存的使用。

设置密码的两种方式 ：

1 config set requirepass “password”。 这种是设置密码是暂时性的

2 配置redis.conf 中设置requirepass属性，后面为密码。

输入认证的方式也有两种

1登录时可以 redis-cli -a password

2 登录后使用 auth password

**主从复制**

通过主从复制可以允许多个slave server拥有和master server相同的数据库副本。

从服务器只能读，不能写。

Redis主从复制特点：

1、master可以拥有多个slave。

2、多个slave可以连接同一个master外，还可以连接到其他的slave。（当master宕机后，相连的slave转变为master）

3、主从复制不会阻塞master，再同步数据时，master可以继续处理client请求。

4、提高了系统的可伸缩性。

Redis主从复制的过程：

1、Slave与master建立连接，发送sync同步命令。

2、 Master会启动一个后台进程，将数据库快照保存到文件中，同时Master主进程会开始收集新的写命令并缓存。

3、 后台完成保存后，就将此文件发送给Slave。

4、 Slave将此文件保存到磁盘上。

**持久化机制**

Redis是一个支持持久化的内存数据库（数据存储在内存之中所以速度非常快），Redis需要经常将内存中的数据同步到磁盘来保证持久化。

Redis支持两种持久化方式：

1、snapshotting（快照），将数据存放到文件里，默认方式。

是将内存中的数据已快照的方式写入到二进制文件中，默认文件dump.rdb，可以通过配置设置自动做快照持久化的方式。可配置Redis在n秒内如果超过m个key被修改就自动保存快照。

save 900 1 #900秒内如果超过1个key被修改，则发起快照保存

save 300 10 #300秒内如果超过10个key被修改，则快照保存

2、 Append-only file（缩写为aof），将读写操作存放到文件中。

由于快照方式在一定间隔时间做一次，所以如果Redis意外down掉的话，就会丢失最后一次快照后的所有修改。

aof比快照方式有更好的持久化性，是由于使用aof时，redis会将每一个收到的写命令都通过write函数写入到文件中当redis启动时会通过重新执行文件中保存的写命令来在内存中重新建立整个数据库的内容。

由于os会在内核中缓存write做的修改，所以可能不是立即写到磁盘上，这样aof方式的持久化也还是有可能会丢失一部分数据。可以通过配置文件告诉redis我们想要通过fsync函数强制os写入到磁盘的时机。

配置文件中的可配置参数：

appendonly yes //启用aof持久化方式

#appendfsync always //收到写命令就立即写入磁盘，最慢，但是保证了数据的完整持久化

appendfsync everysec //每秒中写入磁盘一次，在性能和持久化方面做了很好的折中

#appendfsync no //完全依赖os，性能最好，持久化没有保证

在redis-cli的命令中，SAVE命令是将数据写入磁盘中。

> help save

>save

## Redis持久化

### Redis持久化 之 RDB方式

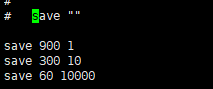
在指定的时间间隔写入硬盘 ,此方式 是 Redis默认的持久化方式

这种持久化方式，比较方便，因为就是一个文件 ，在/usr/local/redis

目录下的 dump.rdb

缺点 ：若服务器宕机，在指定时间没将数据写入文件，就损失较大

redis.conf 文件下，下图内容的含义



save 900 1： 表示每900秒内，至少有一个key发生变化，就持久化

save 300 10 : 表示每300 秒内，至少有10个key发生变化，就持久化

save 60 10000 : 表示每60 秒内，至少有10000个key发生变化，就持久化

TIM截图20181015200908

dbfilename : 持久化数据文件的名称

演示 数据库的备份和恢复

1 删除dump.rdb 文件

2 后台启动redis，进入redis

3 设置几个key value

4 执行 shutdown save 紧接着执行 exit

5 这时候 dump.conf 文件又有了

6 将 dump.conf 文件剪切到 另外一个目录 备份

7 再 启动 链接 redis ，keys \* 已经没有数据了

8 在把 dump.conf 拷贝到 /usr/local/redis 目录下 恢复

9 数据 又回来了 这就是 redis的 备份和恢复

### Redis 持久化 之 AOF 方式

将以日志，记录每一个操作，服务器启动后就构建数据库

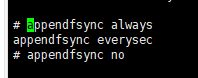
优势 ： 安全性相比RDB方式高很多

劣势 ：效率相比RDB低很多

在redis.conf 下 又这么一个默认配置

no : 默认关闭 aof方式 ，改为yes 开启

在redis.conf文件下 AOF 方式的三种同步策略



appendfsync always ：只要发生修改，立即同步（推荐使用，安全性最高）

appendfsync everysec ： 每秒同步一次

appendfsync no ： 不同步

所以要 将 always 打开 其他的注释

在 启动，链接 redis 会在 /usr/local/redis 目录下，增加 appendonly.aof文件

AOF方式数据的恢复 演示

先不演示了，即使删除了dump.rdb，数据也能恢复

## Java链接Redis

一般用Jedis

依赖

<dependency>

<groupId>redis.clients</groupId>

<artifactId>jedis</artifactId>

<version>2.9.0</version>

</dependency>

示例代码 设置 key为 age 值为 18

Jedis jedis =new Jedis("192.168.221.200",6379); //创建客户端 设置IP和端口

jedis.set("age","18");

String age = jedis.get("age");

jedis.close(); //释放链接资源

现在用java链接 redis 还有问题

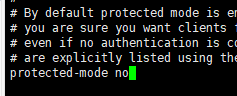
1 需配置下防火墙，开一个6379 端口的权限

firewall-cmd --zone=public --add-port=6379/tcp --permanent

firewall-cmd --reload

2 修改配置文件 /usr/local/redis/redis.conf 将 bind 127.0.0.1 注释

并修改 protected-mode no



关闭 redis

重启redis

-------------------------------------------------------

第2种 设置redis 链接密码

protected-mode 的值 还改为yes

关闭 重启 链接

然后 config set requirepass 1234

a uth 1234 ： 验证密码

Java代码 ： 增加jedis.auth(“1234”)

//quit 退出redis 控制命令 进入 linux 命令 exit :也是

Redis链接池 ： 不用连接池的话，创建一个链接 再释放一个链接，在操作时还得创建链接，这样频繁的访问 数据库，会对数据库有不好的影响

Java示例代码

JedisPoolConfig config=new JedisPoolConfig();

config.setMaxTotal(100); //设置最大链接数

config.setMaxIdle(10); //设置最大空闲链接数

JedisPool pool =new JedisPool(config,"192.168.221.200",6379);

Jedis jedis=null;

try {

jedis= pool.getResource();

jedis.auth("1234");

jedis.set("age","18");

String age = jedis.get("age");

System.out.println(age);

} catch (Exception e) {

e.printStackTrace();

}finally {

if(jedis!=null){

jedis.close();//将链接放回pool

}

if(pool!=null){

pool.close();

}

}

## Redis 5种数据类型

1 字符串 （String）

2 字符串列表 (list)

3 有序字符串集合 （sorted set）

4 哈希 （hash）

5 字符串集合 （set）

Hash 的 值 是一个 map集合，

### Hash相关操作

hset h1 username tom

hset h1 password 123

hget h1 username : 获取 hash中username的值

hmset h2 username tom password 123 : 同时设置多个键值对

hegetall h2 : 获取 h2 哈希中的所有键值对

hdel h2 username : 删除 （可以同时删除多个）

hincrby h1 age 2 ： 增加key的值

hexists h1 age : 判断 h1 hash中 是否存在相应的key 存在返回 1

hlen h1 :查看h1 hash中 有多少 键值对

hgetall h1 : 获取 h1中所有的 键值对

hkeys h1: 获取 h1 中所有的 key

hvals h1 : 获取所有的值

### List 操作

lpush l1 a b c d : 定义 列表 l1 并添加 abcd 元素 从左到右push

rpush l2 a b c d : 定义列表l2 并添加 abcd 元素 从右到左push

lrange l1 start end : 获取 start到 end 的 值 索引支持负数

lpop : 从左侧 弹出集合元素

rpop ： 从右侧弹出集合元素

llen l1 : 查看元素个数

lpush l1 xx : 从列表头部 插入元素xx

rpush l1 xx : 从列表尾部 插入元素xx

lset l1 index xxx : 在指定位置上设置值

linsert l1 before xxx yyy : 在 xxx 前面 插入元素

linsert l1 afterxxx yyy : 在 xxx 后面 插入元素 yyy

rpoplpush l1 l2 : 将列表l1 最后一个元素 弹出 并 将这个元素插到 l2 元素最前面

### 集合 set 操作

集合元素不可以重复

sadd set1 a b c : 定义set1集合，并赋值

smembers set1 : 获取集合元素

sismember set1 a : 查询元素a 是否存在 集合中，存在返回1，不存在返回0

sdiff set1 set2 : 获取 差集 set1 - set2

sinter set1 set2 : 计算两个集合的交集

sunion set1 set2 : 计算两个集合的并集

scard set1 ： 计算元素的总个数

srandmember set1 : 从集合 set1中随机取一个元素

srandmember set1 2 ： 随机取 2 个元素

sdiffstore r1 set1 set2 : 将差集结果存放到 r1当中

sinterstore r set1 set2 : 将2个集合的交集 结果存放到r中

sunionstore r set1 set2 : 将...

**zset 类型（排序set）**

zadd sort1 1 a 2 b 3 c : 向 sort1 集合中添加元素 a,b,c 评分是 1,2,3。排序是根据评分大小排序的

zrange sort1 0 -1 : 查看集合 sort1 中元素

zscore sort1 a : 获取 a的评分

zrem sort1 a : 删除元素

zcard sort1 : 查看集合里元素的个数

zrange sort1 0 -1 withscores : 把评分也查出来

zrevrange sort1 0 -1 withscores : 降序排列

zremrangebyrank sort1 0 2 : 根据排名删除元素，删除 索引 前3 个元素

zrangebyscore sort1 18 23 withscores :根据评分范围查询元素

zrangebyscore sort1 18 23 withscores limit 0 2 :分页查询

Zincrby sort1 5 a : 给指定元素加分

zcount sort1 20 23 : 查找指定评分范围的元素个数

## 

## Redis集群

### Redis单机多节点集群

**第一步**

Redis安装包里有个集群工具，要复制到/usr/local/bin里面去

cp /root/redis-3.2.9/src/redis-trib.rb /usr/local/bin

**第二步 ： 修改配置，创建节点**

假如 搞6个节点，3主3从

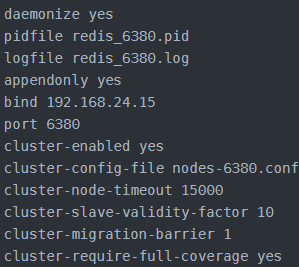
端口规定分别为 7001 - 1006

在root 目录下 ... 如下图的步骤



先将 /root/redis-3.2.9/redis.conf 复制到7001 目录下，然后修改 7001目录下的 redis.conf

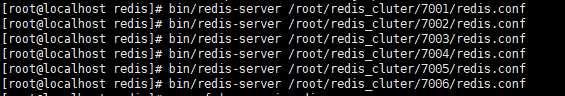




在将 redis.conf 拷贝到 7002-7006 目录，分别编辑，将7001 的地方改成 对应的即可

**第三步：启动6个节点的redis**

执行 bin/redis-server 命令,指定 相应的redis.conf 后台启动

****

**bin/redis-server /root/redis\_cluter/7001/redis.conf**

**bin/redis-server /root/redis\_cluter/7002/redis.conf**

**bin/redis-server /root/redis\_cluter/7003/redis.conf**

**bin/redis-server /root/redis\_cluter/7004/redis.conf**

**bin/redis-server /root/redis\_cluter/7005/redis.conf**

**bin/redis-server /root/redis\_cluter/7006/redis.conf**

**第四步：创建集群**

Redis官方提供了redis-trib.rb工具，第一步已经放到bin目录下了，但在使用之前需要安装**ruby**，以及redis和 ruby的 连接

yum -y install ruby ruby-devel rubygems rpm-build

gem install redis : 安装连接 出现下面错误 下面的地址解决

[https://www.cnblogs.com/PatrickLiu/p/8454579.htmlTIM截图20181015222003](https://www.cnblogs.com/PatrickLiu/p/8454579.html)

依次执行 下面命令解决：

1 ：gpg --keyserver hkp://keys.gnupg.net --recv-keys 409B6B1796C275462A1703113804BB82D39DC0E37D2BAF1CF37B13E2069D6956105BD0E739499BDB

2：curl -sSL https://get.rvm.io | bash -s stable

3： find / -name rvm -print

4： source /usr/local/rvm/scripts/rvm

5：rvm install 2.4.1 安装一个ruby版本

6：rvm use 2.4.1 使用一个 ruby版本

7：rvm use 2.4.1 --default 设置默认版本

8：rvm remove 2.3.4 卸载一个已知 版本

ruby --version ： 查看ruby版本

gem install redis ： 安装redis

创建连接 （创建集群）

redis-trib.rb create --replicas 1 localhost:7001 localhost:7002 localhost:7003 localhost:7004 localhost:7005 localhost:7006

创建集群

**第五步 ：集群数据测试**

先连接任意一个节点，然后添加一个key。

redis-cli是redis默认的客户端工具，启动时加上-c 参数，-p 指定端口，就可以连接到集群。

bin/redis-cli -c -p 7001

然后 set key value时 报如下图 的错误

TIM截图20181016213020

pkill -9 redis ：关闭所有redis节点

### Redis多机多节点集群