# redis介绍

Nosql 非关系型数据库

百科

Redis

可基于内存 , 可持久化的日志型、Key-Value数据库

去取数据的时候，指定key即可。不能用sql语句。

Redis是一个开源的内存数据库，它以键值对的形式存储数据。由于数据存储在内存中，因此Redis的速度很快，但是每次重启Redis服务时，其中的数据也可能会丢失，因此，Redis也提供了持久化存储机制，将数据以某种形式保存在文件中，每次重启时，可以自动从文件加载数据到内存当中.

## Windows

绿色版 （建议使用）



安装版

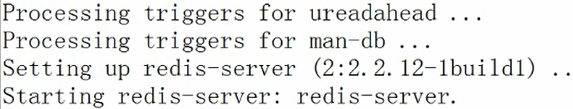


建议redis安装的目录增加到环境变量。

D:\Program Files\redis

## Linux





启动

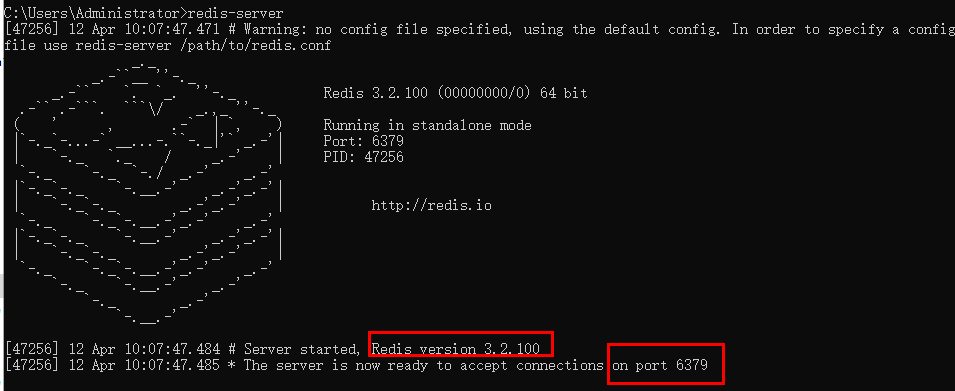






启动









# 配置介绍

# redis.conf

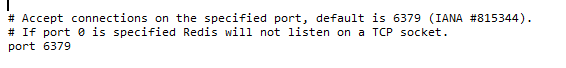
## bind



外网使用不了

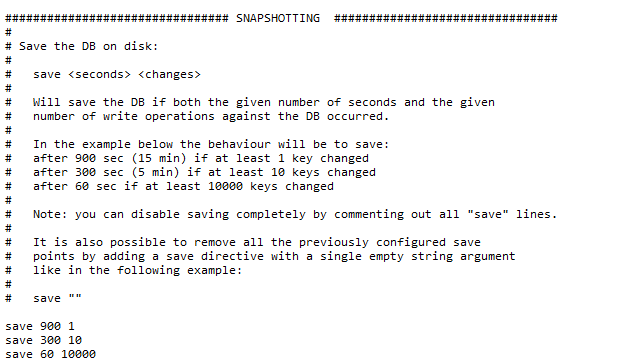
改成0.0.0.0

## 端口号



通常不需要修改

## 保存时间



## 存储策略

### RDB

Redis DataBase

Redis 默认的存储策略



### AOF

Append-only file

保存策略

# 基本使用

## 命令行下使用

## Demo

Redis 是key value形式的存储系统（nosql数据库）

我们所有存的东西，都需要有个key

Redis Vs HashMap

只是我们之前value 可以存很多对象，

Redis里value 默认支持常见的数据结构。

HashMap<String ,Object> 可以put任何对象。

而redis通过key去保存的值，只能是特定的一些结构。

操作不同的数据结构的时候，redis提供了不同的命令。

### 字符串（value的类型）

跟之前的map 非常类似。Value是字符串。

* [SET](http://doc.redisfans.com/string/set.html) key value
* [GET](http://doc.redisfans.com/string/get.html) key
* [INCR](http://doc.redisfans.com/string/incr.html) 可以对应的key的数值（整型的数值）加一
* [INCRBY](http://doc.redisfans.com/string/incrby.html) 给数值加上一个步长
* [SETEX](http://doc.redisfans.com/string/setex.html) expire 过期
* [SETNX](http://doc.redisfans.com/string/setnx.html) not exist key不存在的时候再去赋值

### LIST

* [LPUSH](http://doc.redisfans.com/list/lpush.html)



后面的元素放在栈顶 （）

* [LPOP](http://doc.redisfans.com/list/lpop.html)

返回第一个元素，并且在列表上删除该元素 （栈顶）

* [LLEN](http://doc.redisfans.com/list/llen.html)

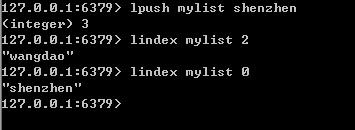
返回当前的list列表的长度



* [LINDEX](http://doc.redisfans.com/list/lindex.html)

返回当前的list的指定index下标的元素。没有返回nil

0表示栈顶的元素



* [LINSERT](http://doc.redisfans.com/list/linsert.html)

插入的位置是按照index的顺序

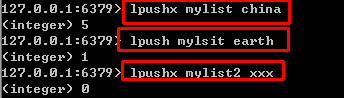
Before的话得注意 index的值





* [LPUSHX](http://doc.redisfans.com/list/lpushx.html)

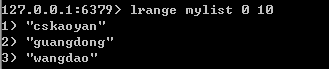
如果list存在，再去push



* [LRANGE](http://doc.redisfans.com/list/lrange.html)

可以方便的查看某个index范围内的lislt的值

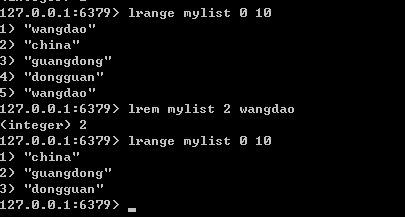
输入的index是从0开始，显示的标号是从1开始的。



* [LREM](http://doc.redisfans.com/list/lrem.html)

删除list里的指定的前几个（指定value的）元素

比如 lrem my12list 2 123 意思是删除前两个 123



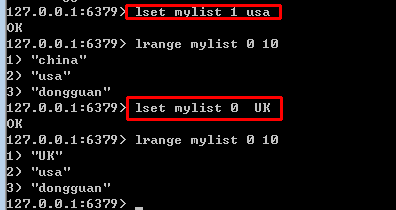


Value表示删除的元素

Count表示删除几个该元素 从列表的第一个开始找

删除指定位置的元素：没有

* [LSET](http://doc.redisfans.com/list/lset.html)
* 设置指定的位置的元素的值 （修改）
* 输入的index是从0开始，显示的标号是从1开始的。



### Hash （二维表）

User{

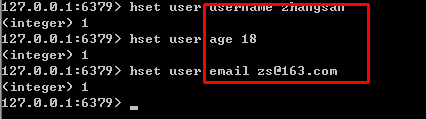
Username:zhangsan  
 age :18

Email:zs@163.colm

}

值哈希表 key value

* [HSET](http://doc.redisfans.com/hash/hset.html)



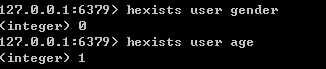
* [HGET](http://doc.redisfans.com/hash/hget.html)

给一个hash表的key，返回一个值



* [HEXISTS](http://doc.redisfans.com/hash/hexists.html)

表里的key是否存在

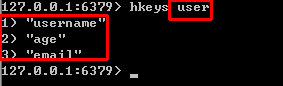


* [HGETALL](http://doc.redisfans.com/hash/hgetall.html)

返回hash表里的所有元素



* [HKEYS](http://doc.redisfans.com/hash/hkeys.html)



* [HLEN](http://doc.redisfans.com/hash/hlen.html)

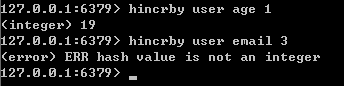
有几个元素（几行数据）



* [HVALS](http://doc.redisfans.com/hash/hvals.html)



* [HINCRBY](http://doc.redisfans.com/hash/hincrby.html)



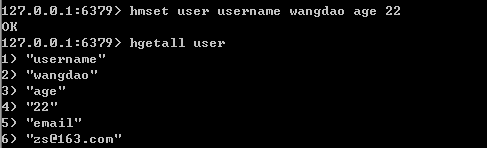
* [HMGET](http://doc.redisfans.com/hash/hmget.html)

Many

给一个hash表的多个key，返回多个值



* [HMSET](http://doc.redisfans.com/hash/hmset.html)



* [HSETNX](http://doc.redisfans.com/hash/hsetnx.html)

Not exist 不存在会创建一个key，然后设置值



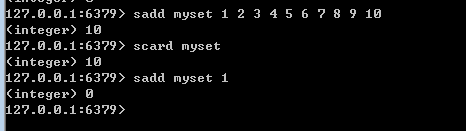
存在，不改变该key 的值



### Set（无序集合）

* [SADD](http://doc.redisfans.com/set/sadd.html)

往set集合里增加一个或者多个元素



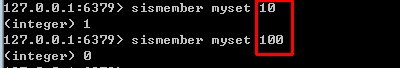
* [SMEMBERS](http://doc.redisfans.com/set/smembers.html)

列出这个集合里的所有元素



* [SISMEMBER](http://doc.redisfans.com/set/sismember.html)

是否存在在当前的集合中



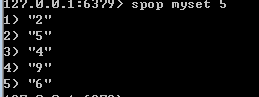
* [SCARD](http://doc.redisfans.com/set/scard.html)

集合的长度



* [SPOP](http://doc.redisfans.com/set/spop.html) （弹出并从集合删除）

取出集合里的5个元素



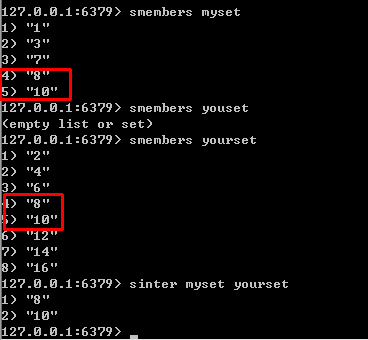
* [SRANDMEMBER](http://doc.redisfans.com/set/srandmember.html)

随机取出 5个元素（不删除）



* [SINTER](http://doc.redisfans.com/set/sinter.html)

求出两个集合的相交部分

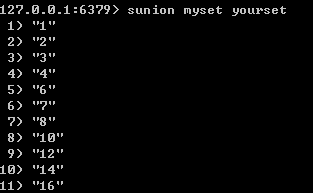


* [SINTERSTORE](http://doc.redisfans.com/set/sinterstore.html)

把相交的集合保存起来

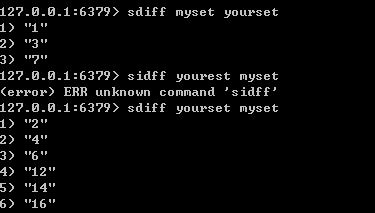
* [SUNION](http://doc.redisfans.com/set/sunion.html)

求两个集合的并集



* [SUNIONSTORE](http://doc.redisfans.com/set/sunionstore.html)
* [SDIFF](http://doc.redisfans.com/set/sdiff.html)

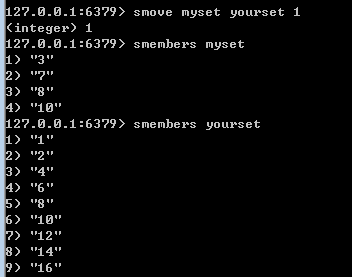
参数1 减去参数1和2的交集



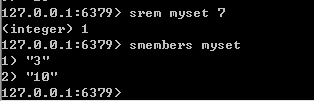
* [SDIFFSTORE](http://doc.redisfans.com/set/sdiffstore.html)

同上，SDIFFSTORE destination key1 [key2] 返回给定所有集合的差集并存储在 destination 中

* [SMOVE](http://doc.redisfans.com/set/smove.html)



* [SREM](http://doc.redisfans.com/set/srem.html) 删除



你的关注的人

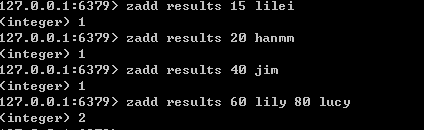
他关注的人

的共同部分 sinter 就是共同关注的

### SortSet

有序的集合，它是可以给一个值 赋予一个分值，根据分值来排序

* [ZADD](http://doc.redisfans.com/sorted_set/zadd.html)



* [ZCARD](http://doc.redisfans.com/sorted_set/zcard.html)

长度



* [ZSCORE](http://doc.redisfans.com/sorted_set/zscore.html)



* [ZCOUNT](http://doc.redisfans.com/sorted_set/zcount.html) （闭区间）



* [ZINCRBY](http://doc.redisfans.com/sorted_set/zincrby.html)



* [ZRANGE](http://doc.redisfans.com/sorted_set/zrange.html)

根据排名给出范围



* [ZRANGEBYSCORE](http://doc.redisfans.com/sorted_set/zrangebyscore.html)

根据指定的分值范围去查找



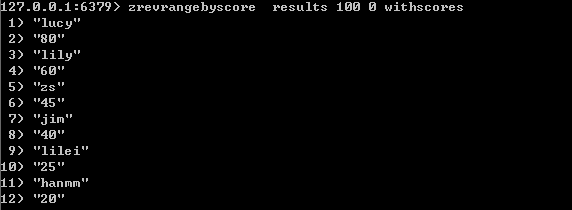
* [ZRANK](http://doc.redisfans.com/sorted_set/zrank.html) （排名从0开始）



* [ZREVRANGE](http://doc.redisfans.com/sorted_set/zrevrange.html)
* 先全部反转，在去对应的index的value

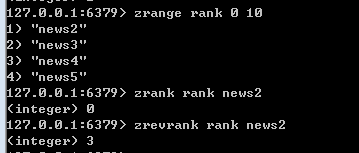


* [ZREVRANGEBYSCORE](http://doc.redisfans.com/sorted_set/zrevrangebyscore.html)



* [ZREVRANK](http://doc.redisfans.com/sorted_set/zrevrank.html)

返回一个逆序的排名值

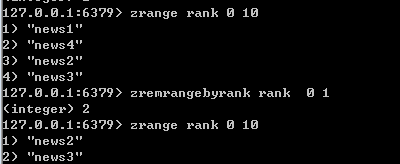


* [ZREM](http://doc.redisfans.com/sorted_set/zrem.html)

删除

* [ZREMRANGEBYRANK](http://doc.redisfans.com/sorted_set/zremrangebyrank.html)

删除指定排名的成员



* [ZREMRANGEBYSCORE](http://doc.redisfans.com/sorted_set/zremrangebyscore.html)

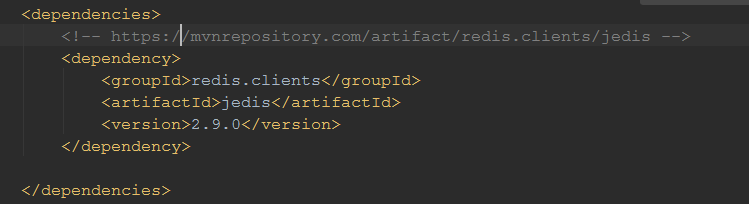
删除指定分数区间的成员



## 代码里使用

1 jedis

增加依赖

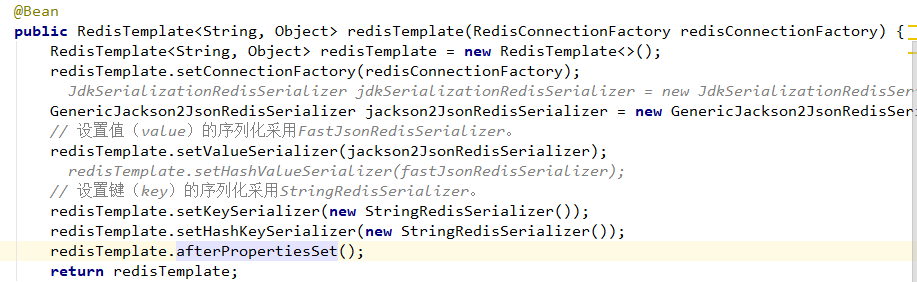


2 RedisTemplate

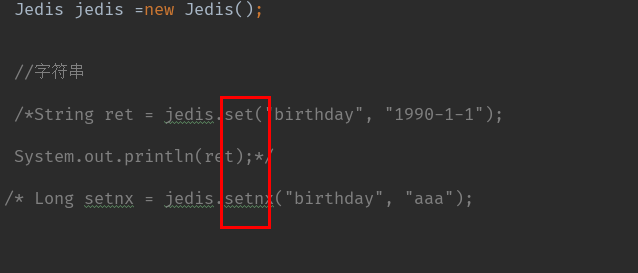
导包

<**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-data-redis</**artifactId**>  
</**dependency**>

设置序列化



### 基本使用



### 优化一下

# 项目实战

点赞点踩

用哪种数据结构实现

Set

Key ： newsid\_like

集合里放的是什么？ { 1 }

newsid\_dislike {

}