服务器相关命令

- ping： 检测连接是否存活
- echo： 在命令行打印一些内容
- quit、exit： 退出客户端
- shutdown： 退出服务器端
- info： 返回redis相关信息
- config get dir/* 实时传递接收的请求
- showlog： 显示慢查询
- select n： 切换到数据库n，redis默认有16个数据库（DB 0~DB 15），默认使用的第0个
- dbsize： 查看当前数据库大小
- move key n： 不同数据库之间数据是不能互通的，move移动键到指定数据库
- flushdb： 清空当前数据库中的键值对。
- flushall： 清空所有数据库的键值对。

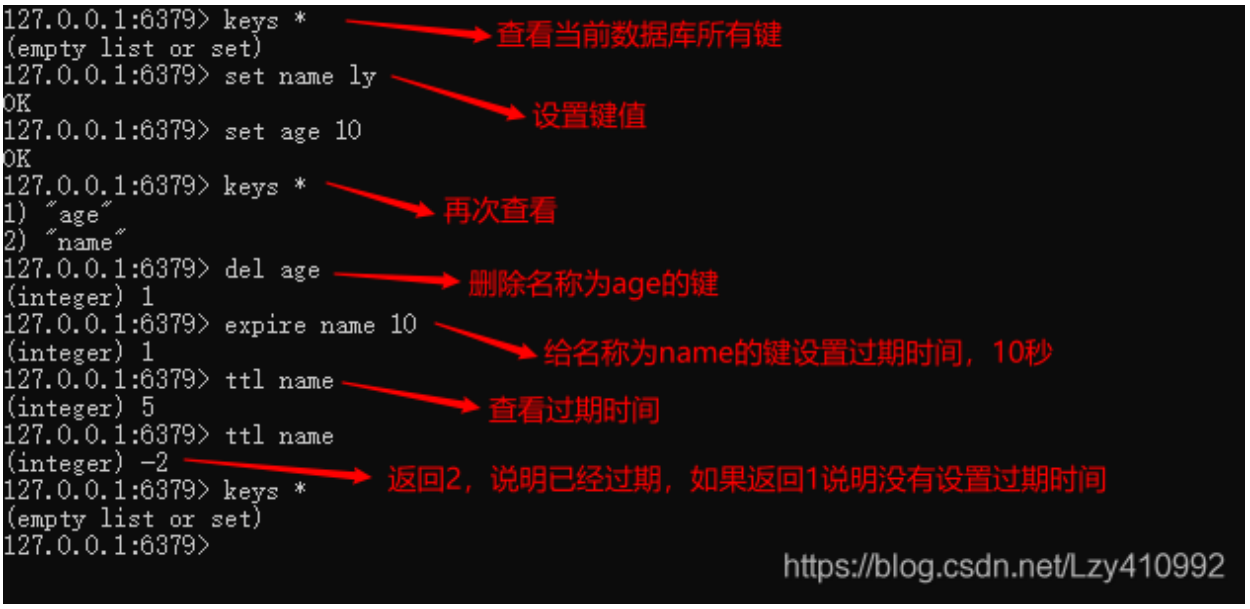
key相关命令

在redis中无论什么数据类型，在数据库中都是以key-value形式保存，通过进行对Redis-key的操作，来完成对数据库中数据的操作。

常用命令：

- keys * ：查看当前数据库中所有的key
- dbsize： 键总数
- exists key： 检查键是否存在
- del key [key ...]： 删除键
- expire key seconds： 键过期
- ttl key： 获取键的有效时长
- persist key： 移除键的过期时间
- type key： 键的数据结构类型
- randomkey： 随机返回数据库中一个键
- rename key1 key2 ： 重命名
- renamex key1 key2 ： 当key2不存在时，key1重命名

代码示例：



五大数据类型

Redis是一个开源（BSD许可），内存存储的数据结构服务器，可用作数据库，高速缓存和消息队列代理。其通过提供多种键值数据类型来适应不同场景下的存储需求，目前为止Redis支持的键值数据类型如下：

- 字符串类型： string
- 哈希类型： hash
- 列表类型： list
- 集合类型： set
- 有序集合类型： sortedset（zset）

String(字符串)

字符串类型是Redis最基础的数据结构，其它的几种数据结构都是在字符串类型基础上构建的，字符串的值可以是：字符串、数字、二进制，但其值最大不能超过512M。

使用场景： 缓存、计数器、对象存储缓存（共享session）、限速

常用命令：

- set key value： 设置一个key的value值
- setnx key value： 仅当key不存在时进行set
- setex key seconds value： set 键值对并设置过期时间
- mset key value [key value ...]： 设置多个key value

- msetnx key1 value1 [key2 value2...]： 批量设置键值对，仅当参数中所有的key都不存在时执行，原子性操作，一起成功，一起失败
- get key： 返回key的value
- mget key [key ...]： 批量获取多个key保存的值
- exists key [key ...]： 查询一个key是否存在
- decr/incr key： 将指定key的value数值进行+1/-1(仅对于数字)
- incrby/decrbyB key n： 按指定的步长对数值进行加减
- incrbyfloat key n： 为数值加上浮点型数值
- append key value： 向指定的key的value后追加字符串
- strlen key： 返回key的string类型value的长度。
- getset key value： 设置一个key的value，并获取设置前的值，如果不存在则返回null
- setrange key offset value： 设置指定位置的字符
- getrange key start end： 获取存储在key上的值的一个子字符串

代码示例：

List(列表)

Redis列表是简单的字符串列表，按照插入顺序排序。你可以添加一个元素到列表的头部（左边）或者尾部（右边），也可以获取指定范围指定下标的元素等。一个列表最多可以包

含 232 － 1 个元素（4294967295，每个列表超过40亿个元素）。

两个特点：

1. 列表中的元素是有序的，可以通过索引下标获取某个元素霍某个某个范围内的元素列表
2. 列表中的元素可以是重复的

使用场景： 消息队列、栈、文章列表等。

常用指令：

添加操作

- lpush/rpush key value1[value2...]： 从左边/右边向列表中PUSH值(一个或者多个)
- lpushx/rpushx key value： 向已存在的列名中push值（一个或者多个），list不存在 lpushx失败
- linsert key before|after pivot value： 在指定列表元素的前/后 插入value

查找操作

- lindex key index： 通过索引获取列表元素
- lrange key start end： 获取list 起止元素 （索引从左往右 递增）
- llen key： 查看列表长度

删除操作

- lpop/rpop key： 从最左边/最右边移除值 并返回
- lrem key count value： count >0：从头部开始搜索 然后删除指定的value 至多删除count个 count < 0：从尾部开始搜索... count = 0：删除列表中所有的指定value。
- ltrim key start end： 通过下标截取指定范围内的列表
- rpoplpush source destination： 将列表的尾部(右)最后一个值弹出，并返回，然后加到另一个列表的头部

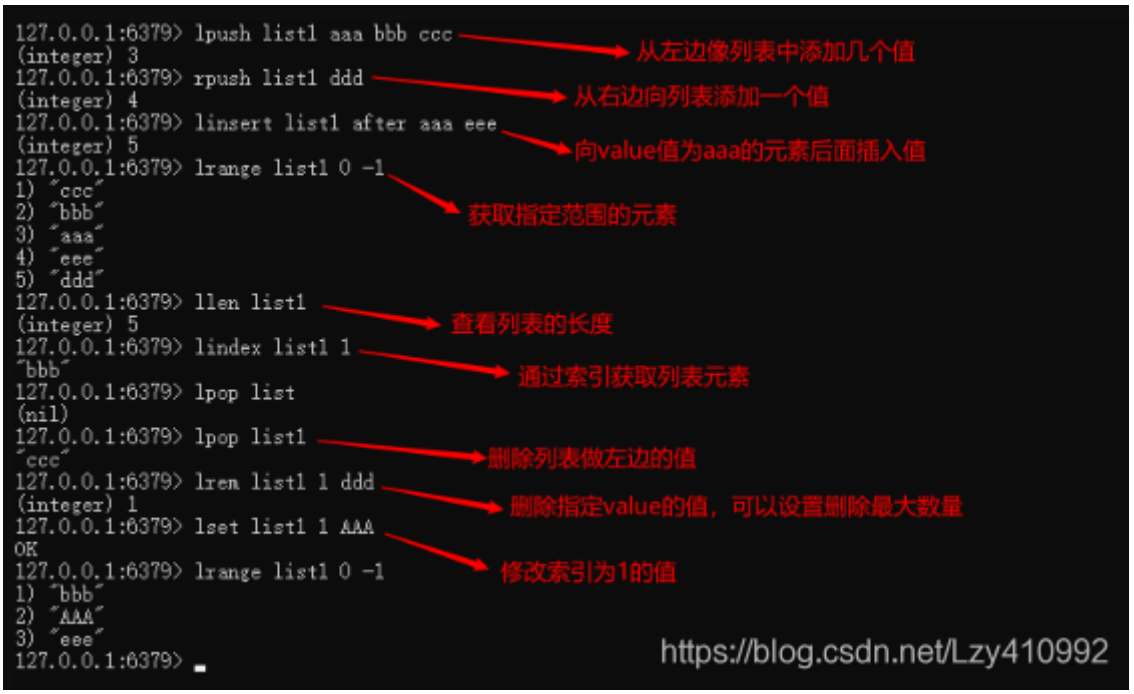
修改操作

- lset key index value： 通过索引为元素设值

阻塞操作

- blpop/brpop key1[key2] timout： 移出并获取列表的第一个/最后一个元素，如果列表没有元素会阻塞列表直到等待超时或发现可弹出元素为止。
- brpoplpush source destination timeout： 和rpoplpush功能相同，如果列表没有元素会阻塞列表直到等待超时或发现可弹出元素为止。

代码示例：



Set(集合)

Redis的Set是string类型的无序集合，我们不能通过索引获取元素。集合成员是唯一的，这就意味着集合中不能出现重复的数据。Redis中集合是通过哈希表实现的，所以添加，删除，查找的复杂度都是0(1)。集合中最大的成员数为 232 - 1 (4294967295，每个集合可存储40多亿个成员)。

应用场景： 标签（tag）

常用命令：

集合内操作

- sadd key member1[member2...]： 向集合中无序增加一个/多个成员
- srem key member1[member2...]： 移除集合中一个/多个成员
- scard key： 获取集合的成员数
- smembers key： 返回集合中所有的成员
- sismember key member： 查询member元素是否是集合的成员，若存在返回1，不存在返回0
- srandmember key [count]： 随机返回集合中count个成员，count缺省值为1
- spop key [count]： 随机移除并返回集合中count个成员，count缺省值为1

集合间操作

- sinter key1 [key2...]： 返回所有集合的交集
- sinterstore destination key1[key2...]： 在SINTER的基础上，存储结果到集合中。覆盖
- sunion key1 [key2...]： 返回所有集合的并集
- sunionstore destination key1 [key2...]： 在SUNION的基础上，存储结果到及和张。覆盖
- sdiff key1[key2...]： 返回所有集合的差集 key1- key2 - ...
- sdiffstore destination key1[key2...]： 在SDIFF的基础上，将结果保存到集合中。覆盖
- smove source destination member： 将source集合的成员member移动到destination集合
- sscan key [MATCH pattern] [COUNT count]： 在大量数据环境下，使用此命令遍历集合中元素，每次遍历部分

代码示例：

Hash（哈希）

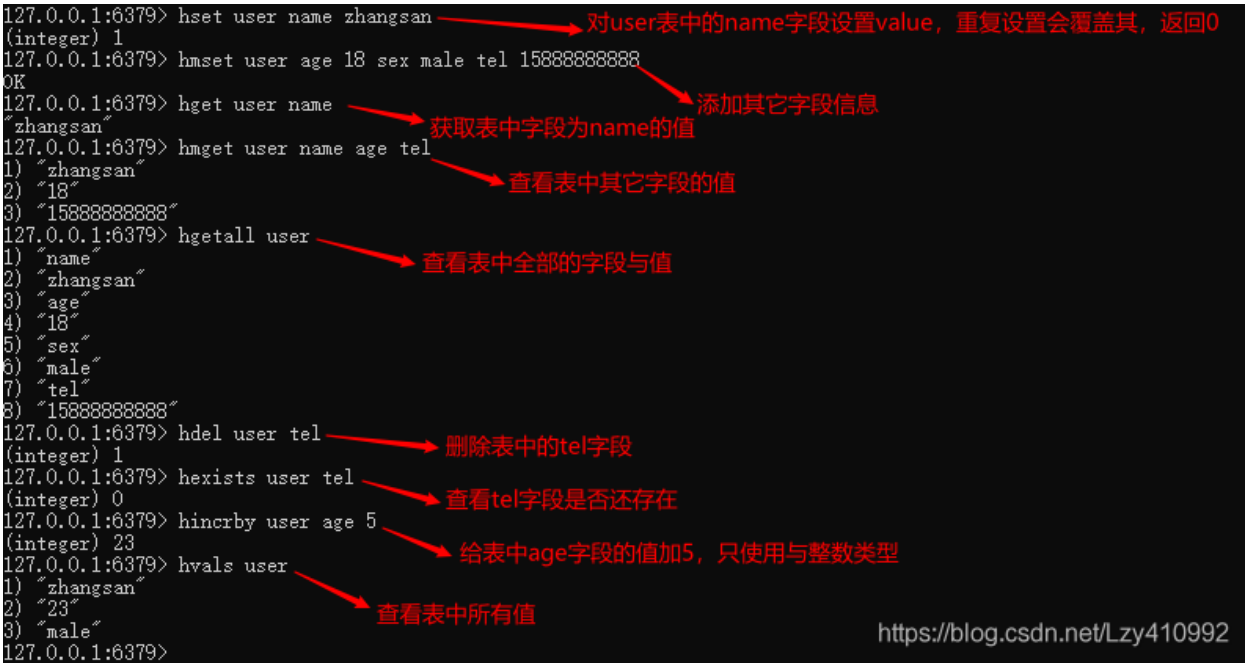
几乎所有的编程语言都提供了哈希（hash）结构，Redis中 hash 是一个string类型的field和value的映射表value={{field1,value1},{field2,value2}…}，可以将一个Hash表作为一个对象进行存储，表中存放对象的信息。

应用场景： 用户信息缓存

常用命令：

- hset key field value： 将哈希表 key 中的字段 field 的值设为 value。重复设置同一个field会覆盖,返回0
- hmset key field1 value1 [field2 value2...]： 同时将多个 field-value (域-值)对设置到哈希表 key 中。
- hsetnx key field value： 只有在字段 field不存在时，设置哈希表字段的值。
- hget key field value： 获取存储在哈希表中指定字段的值
- hmget key field1 [field2...]： 获取所有给定字段的值
- hexists key field： 查看哈希表 key 中，指定的字段是否存在。
- hdel key field1 [field2...]： 删除哈希表key中一个/多个field字段
- hlen key： 获取哈希表中字段的数量
- hkeys key： 获取所有字段field
- hvals key： 获取哈希表中所有值value
- hgetall key： 获取在哈希表key 的所有字段和值
- hincrby key field n： 为哈希表 key 中的指定字段的整数值加上增量n，并返回增量后结果 一样只适用于整数型字段
- hincrbyfloat key field n： 为哈希表 key 中的指定字段的浮点数值加上增量n。
- hscan key cursor [MATCH pattern] [COUNT count]： 迭代哈希表中的键值对。

代码示例：



Zset（有序集合）

在有序集合中保留了不能有重复成员的特性，但其中的成员是可以排序的，每一个元素都会关联一个double类型的分数（score）作为排序依据，score相同时按字典顺序排序。redis正是通过分数来为集合中的成员进行从小到大的排序。

应用场景： 排行榜系统，成绩单，工资表

常用命令：

集合内

- zadd key score member1 [score2 member2]： 向有序集合添加一个或多个成员，或者更新已存在成员的分数
- zcard key： 获取有序集合的成员数
- zscore key member： 返回有序集中，成员的分数值
- zcount key min max： 计算在有序集合中指定区间score的成员数
- zlexcount key min max： 在有序集合中计算指定字典区间内成员数量
- zincrby key n member： 有序集合中对指定成员的分数加上增量 n
- zscan key cursor [MATCH pattern] [COUNT count]： 迭代有序集合中的元素（包括元素成员和元素分值）

范围查询

- zrank key member： 返回有序集合中指定成员的索引
- zrevrank key member： 返回有序集合中指定成员的索引，从大到小排序
- zrange key start end： 通过索引区间返回有序集合成指定区间内的成员
- zrevrange key start end： 通过索引区间返回有序集合成指定区间内的成员，分数从高到底
- zrangebylex key min max： 通过字典区间返回有序集合的成员
- zrevrangebylex key max min： 按字典顺序倒序返回有序集合的成员
- zrangebyscore key min max： 返回有序集中指定分数区间内的成员 -inf 和 +inf分别表示最小最大值，只支持开区间
- zrevrangebyscore key max min： 返回有序集中指定分数区间内的成员，分数从高到低排序

删除操作

- zrem key member1 [member2...]： 移除有序集合中一个/多个成员
- zremrangebylex key min max： 移除有序集合中给定的字典区间的所有成员

- zremrangebyrank key start stop： 移除有序集合中给定的排名区间的所有成员
- zremrangebyscore key min max： 移除有序集合中给定的分数区间的所有成员

集合间操作

- zinterstore destination numkeyskey1 [key2 ...]： 计算给定的一个或多个有序集的交集并将结果集存储在新的有序集合 key 中，numkeys：表示参与运算的集合数，将score相加作为结果的score
- zunionstore destination numkeys key1 [key2...]： 计算给定的一个或多个有序集的交集并将结果集存储在新的有序集合 key 中

代码示例：