**Redis笔记**

**持久化**

提供两种方法，适用于断电数据保护

RDB，周期性将内存中同步到硬盘

AOF，（1s/++次）的记录日志进行持久化操作。

面向互联网方案：

1.主从

主->从 写用主 读用从

2.哨兵

一台机器用来监视，将挂掉的主 用从来升级替代

3.集群

水平扩展，实现单点高可用，并且实现数据分布

总结：

**数据类型**

**Key:** 所有元素的名字

**String:** key , value

**Hash:** hashkey [ key , value ]

**List:** 队列 listkey [ val1 , val2 , …]

**Set:** 有序不重交集并集差集 setkey [ val1 , val2 , …]

**zset:** 做rank排名，zsetkey [ ( score1 , val1 ) , ( score2 , val2 ) , ( … ) , …]

[**命令**](http://doc.redisfans.com/)

**KEY类型**

**keys \*** 显示所有key

**exists** 是否存在

**expire** 设置某个key的过期时间，使用ttl查看剩余时间

**persist** 取消过期时间

**select** 数据库为0-15 （16个） 默认进入0

**rename** 改名

**echo** 打印命令

**dbsize** 查看key数量

**info** 获取数据库信息

**config get parameter**

[config get](http://doc.redisfans.com/server/config_get.html#id1) 命令用于取得运行中的 Redis 服务器的配置参数

**move key db**

将当前数据库的 key 移动到给定的数据库 db 当中。

**String类型**

基础类型

**set [key] [value]**  设置属性 多次设置覆盖

set name Ghaya

**setnx [key][value]** 假如存在value则不设置

set name abc // get name -> Ghaya

**setex [key] [time][value]** 设置存在时间数据，过时清空，可覆盖

setex name 10 abc //10 秒后消失name数据

**mset [key1][value1] [key2][value2]** 批量设置

mset k1 111 k2 222

**getset[key][balue]** 先获取值再设置

getset name abc // ghaya abc

**get [key]** 获取属性

get name // Ghaya

**meet [key1][key2]**  批量获取

mget k1 k2

**setrange [key][start][string]** 替换字符串

setrange email 5 www // "0123456789" -> "01234www89"

**incr/decr incrby/decrby** 递增/递减

incr age // 0 1 2

incr age 3 // 0 3 6

**append** 字符串追加

append name 123456 // name123456

**strlen** 字符串长度

strlen name // 10

Hash类型

**hset [hash][key] [value]** 设置

**hmset [hash] [key1] [value1] [key2] [value2]** 批量设置

**hsetnx** 存在则不设置

**hget [hash][key] / hmget [hash] [key1] [key2]** 获取/批量获取

**hincrby/** **hdecrby** 集合递增和递减

**hexists** 判断存在

**hlen** 返回hash集合里所有的键数值

**hdel**  删除指定hash下的field

**hkeys /hvals/hgetall**  返回hash的所有字段 、value、所有

List类型

有序，一般用作队列应用

**lpush key value [value ...] //**lpush list1 a b c

将一个或多个值 value 插入到列表 key 的表头

**rpush key value [value ...] //**rpush list1 a b c

将一个或多个值 value 插入到列表 key 的表尾

**linsert key BEFORE|AFTER pivot value**

将值 value 插入到列表 key 当中，位于值 pivot 之前或之后。

**lrange key start stop**

返回列表 key 中指定区间内的元素，区间以偏移量 start 和 stop 指定。 -1 表示列表的最后一个元素

**lset key index value**

将列表 key 下标为 index 的元素的值设置为 value 。

**lrem key count value**

根据参数 count 的值，移除列表中与参数 value 相等的元素。

**ltrim key start stop**

对一个列表进行修剪(trim)，就是说，让列表只保留指定区间内的元素，不在指定区间之内的元素都将被删除。

举个例子，执行命令 LTRIM list 0 2 ，表示只保留列表 list 的前三个元素，其余元素全部删除。

**lpop key**

移除并返回列表 key 的头元素。

**rpop key**

移除并返回列表 key 的尾元素。

**rpoplpush source destination**

（原子性）将列 source 中的尾元素弹出，返回。将 source 弹出的元素插入到列表destination ，作为 destination 的头元素。

**llen key**

返回列表 key 的长度。

**SET类型**

string类型的无序集合，不允许重复，取交集并集差集

**sadd key member [member ...]**

将一个或多个 member 元素加入到集合 key 当中，已经存在于集合的 member 元素将被忽略。

**srem key member [member ...]**

移除集合 key 中的一个或多个 member 元素，不存在的 member 元素会被忽略。

**smembers key**

返回集合 key 中的所有成员。

**spop key**

移除并返回集合中的一个随机元素。

**sdiff key [key ...]**

返回一个集合的全部成员，该集合是所有给定集合之间的差集。

**sdiffstore destination key [key ...]**

这个命令的作用和 [sdiff](http://doc.redisfans.com/set/sdiff.html" \l "sdiff) 类似，但它将结果保存到 destination 集合，而不是简单地返回结果集。

**sinter key [key ...] 取交集**

返回一个集合的全部成员，该集合是所有给定集合的交集。

**sinterstore destination key [key ...] 取交集返回**

这个命令类似于 [SINTER](http://doc.redisfans.com/set/sinter.html#sinter) 命令，但它将结果保存到 destination 集合，而不是简单地返回结果集。

**sunion key [key ...] 取并集**

返回一个集合的全部成员，该集合是所有给定集合的并集。

**sunionstore destination key [key ...] 取并集返回**

这个命令类似于 [sunion](http://doc.redisfans.com/set/sunion.html" \l "sunion) 命令，但它将结果保存到 destination 集合，而不是简单地返回结果集。

**smove source destination member**

（原子性）将 member 元素从 source 集合移动到 destination 集合。

**scard key**

返回集合 key 的基数(集合中元素的数量)。

**sismember key member**

判断 member 元素是否集合 key 的成员。

**srandmember key [count]**

如果命令执行时，只提供了 key 参数，那么返回集合中的一个随机元素。从 Redis 2.6 版本开始， [srandmember](http://doc.redisfans.com/set/srandmember.html" \l "srandmember) 命令接受可选的 count 参数：

**ZSet类型**

set增强有序

应用：rank 排行，海量数据进行排行

**zadd key score member [[score member] [score member] ...] 添加**

// zadd zset1 1 one 2 two 3 three

将一个或多个 member 元素及其 score 值加入到有序集 key 当中。

**zrange key start stop [WITHSCORES] 范围查询递增(从小到大)**

返回有序集 key 中，指定区间内的成员。

**zrevrange key start stop [WITHSCORES] 范围查询递减(从大到小)**

返回有序集 key 中，指定区间内的成员。

**zrem key member [member ...] 移除成员**

移除有序集 key 中的一个或多个成员，不存在的成员将被忽略。

**zincrby key increment member 增值/减值（负数）**

为有序集 key 的成员 member 的 score 值加上增量减量 increment 。

当 key 不存在，或 member 不是 key 的成员

zincrby key increment member 等同于 zadd key increment member 。

**zrangebyscore key min max [WITHSCORES] [LIMIT offset count]**

**按分排序查询**

zrangebyscore zset1 -inf +inf withscores limit 3 1

zset1有序集合中，最小到最大，分数，从3开始 查1个

返回有序集 key 中，所有 score 值介于并等于 min 和 max 之间的成员，

按 score 值递增(从小到大)次序排列

**zremrangebyrank key start stop 下标范围删除**

移除有序集 key 中，指定排名(rank)区间内的所有成员。

**zremrangebyscore key min max 分数范围删除**

移除有序集 key 中，所有 score 值介于等于 min 和 max 之间的成员。

**zrank key member**

返回有序集 key 中成员 member 的排名。其中有序集成员按 score 值递增(从小到大)顺序排列。

**zrevrank key member**

返回有序集 key 中成员 member 的排名。其中有序集成员按 score 值递减(从大到小)排序。

**zrangebyscore key min max [WITHSCORES] [LIMIT offset count]**

返回有序集 key 中，所有 score 值介于等于 min 和 max 之间的成员。有序集成员按 score 值递增(从小到大)次序排列。

**zcard key 查集合中个数**

返回有序集 key 的基数。

**zcount key min max 查范围个数**

返回有序集 key 中， score 值在 min 和 max 之间(默认包括 score 值等于 min 或 max )的成员的数量。