Redis操作手册

一、Redis入门

1.在于Redis交互前，我们可以使用redis-server +配置文件路径的方式来启动Redis服务。

```
[root@superbox ~]# redis-server /Redis/redis-5.0.14/redis.conf
```

如果配置文件redis.conf的bind将ip修改为远程ip，则登录需要配合参数-h使用，如下图所示：

```
[root@superbox ~]# redis-cli -h 192.168.3.78
192.168.3.78:6379>
```

2.外部程序使用 TCP 套接字和 Redis 特定协议与 Redis 通信。该协议在 Redis 客户端库中针对不同的编程语言实现。为了让Redis更加简单，我们可以在命令行发送PING命令，如果返回PONG，则代表Redis工作正常。

```
192.168.3.78:6379> ping
PONG
192.168.3.78:6379>
```

3.Redis的数据类型。

(1)Strings(字符串)

1)Strings基本命令

获取和设置字符串

- SET存储一个字符串值。
- SETNX仅当键不存在时才存储字符串值。用于实现锁。
- GET检索字符串值。
- MGET在单个操作中检索多个字符串值。

字符串概述：

> Redis 字符串存储字节序列，包括文本、序列化对象和二进制数组。因此，字符串是最基本的 Redis 数据类型。它们通常用于缓存，但它们支持额外的功能，让您也可以实现计数器并执行按位操作。
>
> Redis 是一个数据结构服务器。Redis 的核心是提供一系列原生数据类型，可解决从缓存到队列再到事件处理的各种问题。

2）在 Redis 中存储然后检索字符串：

```
# 储存字符串
> SET a 8
```

```
192.168.3.78:6379> SET a 8
OK
192.168.3.78:6379>
```

```
# 检索字符串
> GET a
```

```
192.168.3.78:6379> GET a
"8"
```

获取和设置字符串：

```
SET存储一个字符串值。
SETNX仅当键不存在时才存储字符串值。用于实现锁。
GET检索字符串值。
MGET在单个操作中检索多个字符串值。
```

3）存储一个序列化的 JSON 字符串并将其设置为从现在起 100 秒后过期：

```
> SET ticket:27 "\"{'username': 'priya', 'ticket_id': 321}\"" EX 100
# 使用TTL查询key状态
> TTL ticket:27
(integer) 57
```

增加一个计数器：

```
> INCR views:page:2
(integer) 1
> INCRBY views:page:2 10
(integer) 11

# 管理计数器
INCRBY以原子方式递增（并在传递负数时递减）存储在给定键处的计数器。
浮点计数器存在另一个命令：INCRBYFLOAT。
```

(2)Lists(列表)

1)Lists基本命令

- LPUSH将一个新元素添加到列表的头部；RPUSH添加到尾巴。
- LPOP从列表的头部移除并返回一个元素；RPOP做同样的事情，但来自列表的尾部。
- LLEN返回列表的长度。
- LMOVE原子地将元素从一个列表移动到另一个列表。
- LTRIM将列表减少到指定的元素范围。

Redis 列表是字符串值的链表。Redis 列表经常用于：

> 1.实现堆栈和队列。
> 2.为后台工作系统构建队列管理。

2)将列表视为队列（先进先出）：

```
> LPUSH work:queue:ids 101
(integer) 1
> LPUSH work:queue:ids 237
(integer) 2
> RPOP work:queue:ids
"101"
> RPOP work:queue:ids
"237"
```

3)将列表视为堆栈（先进后出）：

```
> LPUSH work:queue:ids 101
(integer) 1
> LPUSH work:queue:ids 237
(integer) 2
> LPOP work:queue:ids
"237"
> LPOP work:queue:ids
"101"
```

4)检查列表的长度：

```
> LLEN work:queue:ids
(integer) 0
```

5)从一个列表中原子地弹出一个元素并推送到另一个

```
> LPUSH board:todo:ids 101
(integer) 1
> LPUSH board:todo:ids 273
(integer) 2
> LMOVE board:todo:ids board:in-progress:ids LEFT LEFT
"273"
> LRANGE board:todo:ids 0 -1
1) "101"
> LRANGE board:in-progress:ids 0 -1
1) "273"
```

6)要创建一个永远不会超过 100 个元素的上限列表，您可以LTRIM在每次调用后调用LPUSH：

```
> LPUSH notifications:user:1 "You've got mail!"
(integer) 1
> LTRIM notifications:user:1 0 99
OK
> LPUSH notifications:user:1 "Your package will be
delivered at 12:01 today."
(integer) 2
> LTRIM notifications:user:1 0 99
OK
```

7）额外补充：

```
# 限制：Redis 列表的最大长度为 2^32 - 1 (4,294,967,295) 个元素。
# 阻塞命令
# 列表支持几个阻塞命令。例如：
BLPOP从列表的头部删除并返回一个元素。如果列表为空，则命令会阻塞，直到元素可用或达到指定的超时。
BLMOVE原子地将元素从源列表移动到目标列表。如果源列表为空，则该命令将阻塞，直到有新元素可用。
```

(3)Sets(无序集合)

1）Redis 集是唯一字符串（成员）的无序集合。您可以使用 Redis 集高效地：

> 1.跟踪唯一项目（例如，跟踪访问给定博客文章的所有唯一 IP 地址）。
> 2.表示关系（例如，具有给定角色的所有用户的集合）。
> 3.执行常见的集合运算，例如交集、并集和差集。

Sets基本命令

- SADD将新成员添加到集合中。
- SREM从集合中删除指定的成员。
- SISMEMBER测试一个字符串的集合成员资格。
- SINTER返回两个或多个集合共有的成员集合（即交集）。
- SCARD返回集合的大小（也称为基数）。

2)

(4)Hashes(哈希)

1)Redis 哈希是结构为字段值对集合的记录类型。可以使用散列来表示基本对象并存储计数器分组等。

Hashes基本命令

- HSET设置散列上一个或多个字段的值。
- HGET返回给定字段的值。
- HMGET返回一个或多个给定字段的值。
- HINCRBY将给定字段的值增加提供的整数。

2)

(5)Sorted sets(排序集合)

1)Redis 排序集是由相关分数排序的唯一字符串（成员）的集合。当多个字符串具有相同的分数时，这些字符串按字典顺序排列。排序集的一些用例包括：

> 排行榜。例如，您可以使用排序集轻松维护大型在线游戏中最高分的有序列表。
> 速率限制器。特别是，您可以使用排序集来构建滑动窗口速率限制器，以防止过多的 API 请求。

Sorted sets基本命令

- ZADD将新成员和相关分数添加到排序集中。如果成员已经存在，则更新分数。
- ZRANGE返回在给定范围内排序的有序集合的成员。
- ZRANK返回提供的成员的排名，假设排序是按升序排列的。
- ZREVRANK返回提供的成员的排名，假设排序集是按降序排列的。

2)

(6)Streams(流)

1. Redis 流数据类型是在 Redis 5.0 中引入的。Streams 对日志数据结构进行建模，但也实现了几个操作来克服典型的仅附加日志的一些限制。其中包括 O(1) 时间内的随机访问和复杂的消费策略，例如消费群体。

   Redis 流是一种数据结构，其作用类似于仅附加日志。您可以使用流实时记录和同步事件。Redis 流用例的示例包括：

> 1.事件溯源（例如，跟踪用户操作、点击等）
> 2.传感器监控（例如，现场设备的读数）
> 3.通知（例如，将每个用户的通知记录存储在单独的流中）
> 4.Redis 为每个流条目生成一个唯一的 ID。

Streams基本命令

- XADD向流中添加新条目。
- XREAD读取一个或多个条目，从给定位置开始并及时向前移动。
- XRANGE返回两个提供的条目 ID 之间的条目范围。
- XLEN返回流的长度。

(7)Geospatial indexes(地理空间索引)

1)Redis 地理空间索引可让您存储坐标并搜索它们。此数据结构对于查找给定半径或边界框内的附近点很有用。

Geospatial indexes基本命令

- GEOADD将位置添加到给定的地理空间索引（请注意，使用此命令，经度位于纬度之前）。
- GEOSEARCH返回具有给定半径或边界框的位置。

(8)Bitmaps(位图)

1)Redis 位图是字符串数据类型的扩展，可让您将字符串视为位向量。您还可以对一个或多个字符串执行按位运算。位图用例的一些示例包括：

> 对于集合的成员对应于整数 0-N 的情况，有效的集合表示。
> 对象权限，每个位代表一个特定的权限，类似于文件系统存储权限的方式。

(9)HyperLogLog(超级日志)

1)HyperLogLog 是一种估计集合基数的数据结构。作为一种概率数据结构，HyperLogLog 以完美的准确性换取了高效的空间利用。Redis HyperLogLog 实现最多使用 12 KB，并提供 0.81% 的标准误差

HyperLogLog基本命令

- PFADD将项目添加到 HyperLogLog。
- PFCOUNT返回集合中项目数的估计值。
- PFMERGE将两个或多个 HyperLogLog 合并为一个。

(10)Extensions(位域)