Hiredis的使用

一、Hiredis的安装与使用

1、下载hiredis软件包，`https://github.com/redis/hiredis.git`或者使用git下载到本地
git clone `https://github.com/redis/hiredis.git`（已经将源文件下载到了本地，可以直接使用）

2、进行解压与安装，步骤如下

```
tar -xzvf hiredis.tar.gz
cd hiredis
make
sudo make install
```

3、更新动态库配置文件
sudo ldconfig

4、头文件与实现文件路径
按照上面步骤安装之后，hiredis的头文件会存在/usr/local/include下面，hiredis的库文件存在/usr/local/lib下面

5、编译方式
g++ xxx.cc -o xxx -I /usr/local/include/hiredis -lhiredis或者直接g++ xxx.cc -lhiredis需要链接hiredis的库文件

6、后续在代码中引用hiredis的头文件，可以直接使用

```
#include <hiredis/hiredis.h>
```

二、Hiredis的重要API

2.1、连接redis数据库

```
redisContext* redisConnect(const char *ip, int port)
```

```
//redisContext不是线程安全的
typedef struct redisContext
{
    int err; /*错误标志，正确连接标志为0，出错时设置为非零常量*/
    char errstr[128]; /*存放错误信息的字符串*/
    int fd;
    int flags;
    char *obuf; /* Write buffer */
    redisReader *reader; /* Protocol reader */
} redisContext;
```

2.2、发送请求命令

第一个参数为连接数据库返回的值，剩余的是可变参数，类似printf。此函数的返回值是void *，但是一般会强制转换为redisReply类型，便于做进一步处理。

```
void *redisCommand(redisContext *c, const char *format...)
```

如果命令执行错误，返回值为NULL，redisContext的err字段被设置为**非零常量**。如果，错误发生，原先的redisContext就不能重复使用，需要重新建立一个新的连接。如果成功执行命令，则标准返回一个redisReply类型，该类型结构如下：

```
typedef struct redisReply
{
    int type; /* 测试收到什么样的回返回 REDIS_REPLY_* */
    long long integer; /* type是REDIS_REPLY_INTEGER类型，integer保存返回的值*/
    int len; /* 保存str类型的长度 */
    char *str; /* type是REDIS_REPLY_ERROR和REDIS_REPLY_STRING，str保存返回的值
*/
    size_t elements; /* type是REDIS_REPLY_ARRAY，保存返回多个元素的数量 */
    struct redisReply **element; /* 返回多个元素以redisReply对象的形式存放 */
} redisReply;

//type还可以是REDIS_REPLY_NIL，表示返回了一个零对象，没有数据可以访问。
```

通过redisReply结构体中的type变量可以确定命令执行的情况.

```
#define REDIS_REPLY_STRING 1    //字符串
#define REDIS_REPLY_ARRAY 2     //数组，例如mget返回值
#define REDIS_REPLY_INTEGER 3   //数字类型
#define REDIS_REPLY_NIL 4       //空
#define REDIS_REPLY_STATUS 5    //状态，例如set成功返回的‘OK’
#define REDIS_REPLY_ERROR 6     //执行失败
```

- REDIS_REPLY_STATUS：

返回执行结果为状态的命令。比如set命令的返回值的类型是REDIS_REPLY_STATUS，然后只有当返回信息是"OK"时，才表示该命令执行成功。可以通过reply->str得到文字信息，通过reply->len得到信息长度。

- REDIS_REPLY_ERROR：

返回错误。错误信息可以通过reply->str得到文字信息，通过reply->len得到信息长度。

- REDIS_REPLY_INTEGER：

返回整型标识。可以通过reply->integer变量得到类型为long long的值。

- REDIS_REPLY_NIL:

返回nil对象，说明不存在要访问的数据。

- REDIS_REPLY_STRING:

返回字符串标识。可以通过reply->str得到具体值，通过reply->len得到信息长度。

- REDIS_REPLY_ARRAY:

返回数据集标识。数据集中元素的数目可以通过reply->elements获得，每个元素是个redisReply对象，元素值可以通过reply->element[..index..].*形式获得，用在获取多个数据结果的操作。

2.3、释放资源

释放redisCommand执行后返回的的redisReply所占用的内存。

```
void freeReplyObject(void *reply)
```

释放redisConnect()所产生的连接

```
void redisFree(redisContext *c)
```