**文件编号：20170207**

**版本号：1.1**

**密 级：企业使用**

**stormGod使用文档**

**文档控制**

**发布**

|  |  |
| --- | --- |
| **文档名称** | **使用文档** |
| **文档版本号** | **1.1** |
| **程序版本号** | **1.5.3.2** |
| **版本日期** | **2017-02-16** |
| **作者** | **李高文** |
| **发行部门** |  |
| **发行日期** |  |

**版本控制**

|  |  |  |
| --- | --- | --- |
| **版本** | **日期** | **说明** |
| **V1.0** | **2017-02-16** | **开始文档的编写** |
|  |  |  |
|  |  |  |

**批准**

|  |  |  |
| --- | --- | --- |
| **头衔** | **签字** | **日期** |
|  |  |  |
|  |  |  |
|  |  |  |

1. 关于 stormGod 5

1.1 系统简介 5

1.2 基本参数 5

1.2.1 硬件 5

1.2.2 软件 5

1.3 系统主要功能 5

2. 系统性能测试 5

2.1 硬件和软件配置 5

2.2 性能测试 6

2.2.1 全内存模式测试 6

2.2.2 全硬盘模式测试 7

3. 程序启动 8

3.1 服务端 8

3.2 客户端 8

4. 各程序使用说明 8

4.1 服务程序 8

4.1.1整体架构 8

4.1.2配置文件说明 8

4.1.3服务端算法简介 9

4.1.4服务端接口 10

4.2 界面客户端 15

4.3 应用静态库 16

4.4 管理端 18

4.2.1 限定条件查询KEY 19

4.2.2 限定条件查询VALUE 23

4.2.3 限定条件查询KEY和VALUE 24

4.2.4 查询最小KEY 26

4.2.5 查询最大KEY 27

4.2.6 限定条件查询KEY数量 27

4.2.7 限定条件删除 27

4.2.8 限定条件导出数据到文件 27

4.2.9 从文件导入数据 28

4.2.10 正常结束进程 28

4.2.11 将内存数据写入到硬盘 28

4.2.12 修改线程数量 28

4.2.13 增加文件最大值 28

4.2.14 修改文件递增值 29

4.2.15 显示内存使用情况 29

4.2.16 显示硬盘使用情况 29

4.2.17 显示长连接和短连接线程数量 30

4.2.18 显示KEY数量 30

# 关于 stormGod

## 系统简介

这是个哈希表处理程序，向服务端插入KEY（键）和VALUE（值），KEY是唯一的，VALUE是任意的，可以根据KEY来查询、更新、删除KEY和VALUE对。KEY的最大长度是512B，VALUE的最大长度是4096B，对服务端来说，KEY和VALUE都是字符串。

整个stormGod，分为以下部分。

1. **stormGodServer.exe**：这个是服务端，作为整个系统的核心部分，负责处理所有客户端的各种请求， 包括数据处理，自身资源管理。
2. **stormGodClient.a**：给C语言应用程序用作编译链接的静态库，里面包括连接服务程序、单条数据处理、批量处理数据。
3. **stormGodUser.exe**：用来从服务端获取数据，只支持单条数据处理。
4. **stormGodManager.exe**：用来管理服务程序的资源，以及批量处理数据，可以执行限定条件。

## 基本参数

### 硬件

现在这套软件是在x64平台上测试的，只适用x64的CPU。

### 软件

这套软件是在windows7 64位上测试，只适用这个操作系统。

## 系统主要功能

这个系统，是用来实现KEY-VALUE对的操作，单条更新、单条查询、单条删除、批量更新、批量查询、批量删除、混合批量操作、限定条件查询KEY和VALUE、查询最小KEY值、查询最大KEY值、限定条件查询KEY数量、限定条件删除、限定条件导出数据到文件、从文件导入数据。另外，还有增加了对服务程序的控制以及查看相关的资源，正常结束进程、将内存数据写入到映射文件、修改长连接和短连接的线程数量、增加存储文件最大值、修改存储文件递增值、显示内存使用情况、显示硬盘使用情况、显示长连接和短连接的线程数量、显示KEY的总数量。

这里的KEY的最大长度是512B、VALLUE的最大长度是4096B、长连接和短连接侦听线程的最大数量是1000。

# 系统性能测试

## 硬件和软件配置



这个是用“鲁大师”软件看到的。简单的说就是 i7-4810MQ+32GB+WIN7 64。

## 性能测试

下面的测试结果都是把测试工具和服务端都放到同一部计算机上，我这里用自己写的客户端来测试，分别测试长连接单条插入和更新、短连接单条插入和更新、长连接批量插入和更新、短连接批量插入和更新。

全内存模式：服务端中的所有数据都在内存，不存在硬盘的读写。

短连接：每次都是先向服务端发起连接请求，发送请求数据，关闭和服务端的连接。

长连接：只向服务端执行一次连接请求，随后执行数据操作，然后才关闭连接。

单条操作：每次向服务端发送一条请求数据。

批量操作：每次向服务端发送多条请求数据。

测试单位：时间是毫秒，速度是条/秒，单条操作的测试数量是线程数\*每线程数量，批量操作的测试数量是线程数\*批量次数\*每次批量数量，结果取5次平均值。

### 全内存模式测试









### 全硬盘模式测试

由于客户端发起的数据都是进入内存，不能直接测试写入硬盘的速度，只能通过间接来实现。

插入时，先通过客户端插入数据，再通过管理端的一个命令，将内存所有数据插入到硬盘，取管理端显示的时间差。

查询时，先将内存所有数据都插入到硬盘，这样，执行查询时，内存中没有数据，从硬盘中查取，取客户端的时间差。

由于全硬盘模式比较慢，这里取3次测试平均值。并且，由于KEY长度的变化，对速度影响够大，分别测试KEY长度为256B和56B



# 程序启动

## 服务端

修改配置文件 keyValue.ini，双击stormGodServer.exe即可。或者新建配置文件，在dos命令行下面执行"server 配置文件"即可。

## 客户端

客户端包括两个可执行程序和一个静态库。stormGodUser.exe，双击该文件，根据提示输入服务端的IP地址和端口，即可执行数据操作，也可以在dos命令行下面执行"stormGodUser.exe 服务端IP 服务端端口"，服务端端口指的是服务端中的短连接，即配置文件中的SHORT\_LISTEN值。stormGodManager.exe的执行同stormGodUser.exe一样，但端口要指向配置文件文件中的MANAGER\_LISTEN值。stormGodClient.a，主要通过调用函数来实现，使用方法在下面会详细说明。

# 各程序使用说明

## 服务程序

### 4.1.1整体架构

整体采取统一资源分配结构，分为内存数据和硬盘数据，两种数据的总和是整个服务端的数据，另外，内存数据要映射到硬盘，在出现突发状况时，可以快速把数据写到硬盘。

插入数据的时候，先插入到内存，内存空间不足才把主体数据插入到硬盘，这种做法的结果是，先插入的数据在内存，后来的数据在硬盘，也可以通过操作把内存数据释放到硬盘。查询数据的时候，先查询内存数据，有就直接返回，没有就从硬盘里查询。

### 4.1.2配置文件说明

配置文件的名称可以自己定义，也可以用提供的配置文件来修改。如果自己定义的配置文件名称，只能在dos命令下面启动服务程序，而如果用提供的配置文件来修改，就可以双击stormGodServer.exe。

配置文件用“#”开头来注释，表示这行是注释，以dos或者unix换行符来结束。如果有多行注释，用多个“#”开头就可以了。

设置的样式是“字段名称=字段值”，等于号前后可以有空格。

下面是各字段的说明。

USER字段：这个是表示程序所有者的用户名，在管理端执行某些命令用到。如果是空值，就用默认的“root”名称。

PASS字段：这个是表示程序所有者的密码，在管理端执行某些命令用到。如果是空值，就用默认的“root”名称。

MANAGER\_LISTEN字段：管理侦听端口，也就是管理端连接的端口。下面管理端在连接服务端时，IP地址就是服务器的地址，端口就用这个。如果是空值，用默认的“10002”。注意，确定该端口没被其他程序占用，并且该端口必须是数字，范围是1~65535。

SHORT\_LISTEN字段：短连接侦听端口，下面的应用程序静态库在连接服务端时，如果是短连接，IP地址就是服务器的地址，端口就用这个。如果是空值，用默认的“10000”。注意，确定该端口没被其他程序占用，并且该端口必须是数字，范围是1~65535。

SHORT\_THREADS字段：短连接的线程数量，这个值决定并发数量，如果是10，表示同时可以处理10个连接，该值加上下面的LONG\_THREADS字段的值，不能超出1000。如果为空，用默认的数量100。

LONG\_LISTEN字段：长连接侦听端口，下面的应用程序静态库在连接服务端时，如果是长连接，IP地址就是服务器的地址，端口就用这个。如果是空值，用默认的“10001”。注意，确定该端口没被其他程序占用，并且该端口必须是数字，范围是1~65535。

LONG\_THREADS字段：长连接的线程数量，这个值决定并发数量，如果是10，表示同时可以处理10个连接，该值加上上面的SHORT\_THREADS字段的值，不能超出1000。如果为空，用默认的数量100。

KEY\_RAM\_SIZE字段：用来放 KEY 的内存大小，单位 MB，这个字段决定了KEY在内存的数量，当插入的KEY的总数量超出这个值，KEY就转入硬盘，降低数据处理速度，这个值越大，在内存上存放的数据越多，数据在内存的概率越大。如果为空，用默认值1024，表示KEY内存的大小是1024MB，也就是1GB。

VALUE\_RAM\_SIZE字段：用来放 VALUE 的内存大小，单位 MB，这个字段决定了VALUE在内存的数量，当插入的VALUE的总数量超出这个值，VALUE就转入硬盘，降低数据处理速度，这个值越大，在内存上存放的数据越多，数据在内存的概率越大。如果为空，用默认值1024，表示KEY内存的大小是1024MB，也就是1GB。

KEY\_FILE\_PATH字段：KEY 文件所在的目录，如果为空，默认是程序执行所在的目录。

KEY\_FILE\_NAME字段：KEY 文件名称，如果为空，默认是keyValue.key。

VALUE\_FILE\_PATH字段：VALUE 文件所在的目录，如果为空，默认是程序执行所在的目录。

VALUE\_FILE\_NAME字段：VALUE 文件名称，如果为空，默认是keyValue.val。

FILE\_UPPER\_SIZE字段：KEY 文件和 VALUE 文件步增值大小，单位 MB，如果为空，用默认值16。

FILE\_MAX\_SIZE字段：KEY 文件和 VALUE 文件最大值，单位 MB，如果为空，用默认值16。

SELECT\_TEMP\_PATH字段：查询缓冲区文件所在的目录，这个值是在管理端执行批量查询时的临时文件，如果为空，默认是程序执行所在的目录。

### 4.1.3服务端算法简介

数据结构用的是B\*树的修改版，这种修改，最大限度利用空间，可以保证空间的利用率在50%以上，也就是当前的计算值是100万，至少可以容下50万。每级数量32，最多10级，也就是支持最大的总数量就是32^10，也就是大概是1E，10^15，1000万亿。

这种数据结构，时间复杂度是n，以32为底的对数。当数据量为32时，插入、查询耗时1个单位。当数据量为1024时，耗时2个单位。数据量是10亿时，耗时6个单位。

存储资源分配用的是分段法，能有效利用空间，内存分配和硬盘分配都一样，只是一个把参数指向内存，另一个把参数指向硬盘。

这个服务端的最大特点是用了并发事务算法，事务是指在在一个过程里面，必须完成一个任务，然后才可以完成另一个任务，并发是指多个任务可以同时执行。插入数据是读写操作，改变整体数据分布，需要插入完一个数据后，才可以跟着插入第二个数据。查询是只读操作，不改变整体数据分布，可以并发。用并发事务算法，多个数据可以同时插入，提高了插入数据的速度，从测试结果看，也提高了查询速度。

### 4.1.4服务端接口

如果要写客户端来对接服务端，可以根据下面的接口来做。长度是字符串，不够位数，可以左补0，也可以左补空格。处理结果，1B，其ASC2值为32（对应字符“0”）时表示成功，值为33（对应字符“1”）时表示失败。

另外，数据总长度不包括长度域本身。

更新操作：服务端接收到KEY，如果KEY存在，做更新操作，如果不存在，做插入操作。

查询操作：服务端收到KEY，查找对应的VALUE返回。如果查询失败，处理结果后面就没有数据。

删除操作：服务端收到KEY，执行删除数据操作。

#### 4.1.4.1非管理端操作

##### 4.1.4.1.1短连接、长连接

短连接：连接服务端的SHORT\_LISTEN端口，客户端关闭连接作为结束。

长连接：连接服务端的LONG\_LISTEN端口，还要每隔3600秒内发送一个长度为0的报文，也就是数据长度域为0，来做心跳报文，进行状态更新，客户端关闭连接作为结束。

##### 4.1.4.1.2接口

每个操作都有固定的关键字，单条更新（0001）、单条删除（0002）、单条查询（0003）、批量更新（0004）、批量删除（0005）、批量查询（0006）、混合批量（0007）、混合批量里面的更新（1001）、混合批量里面的删除（1002）、混合批量里面的查询（1003）。

###### 4.1.4.1.2.1单条更新

发送：数据总长度（10B）+0001+KEY长度（4B）+KEY+VALUE长度（4B）+VALUE

接收：处理结果（1B）

###### 4.1.4.1.2.2单条查询

发送：数据总长度（10B）+0003+KEY长度（4B）+KEY

接收：处理结果（1B）+VALUE长度（4B）+VALUE

###### 4.1.4.1.2.3单条删除

发送：数据总长度（10B）+0002+KEY长度（4B）+KEY

接收：处理结果（1B）

###### 4.1.4.1.2.4批量更新

发送：数据总长度（10B）+0004+（ KEY长度（4B）+KEY + VALUE长度 + VALUE）\*N个KEY-VALUE对

接收：（分段数据长度（10B）+ 处理结果（1B）\*N个）\*M段，直到接收到的分段数据长度为0结束。

###### 4.1.4.1.2.5批量查询

发送：数据总长度（10B）+0006+（ KEY长度（4B）+KEY）\*N个。

接收：（分段数据长度（10B）+ （处理结果（1B）+VALUE长度（4B） + VALUE）\*N个）\*M段，直到接收到的分段数据长度为0结束。

###### 4.1.4.1.2.6批量删除

发送：数据总长度（10B）+0005+（ KEY长度（4B）+KEY）\*N个。

接收：（分段数据长度（10B）+ 处理结果（1B）\*N个）\*M段，直到接收到的分段数据长度为0结束。

###### 4.1.4.1.2.7混合批量

发送：数据总长度（10B）+0007+混合操作类型（4B）+KEY长度（4B）+KEY+VALUE长度（4B）+VALUE。如果混合操作类型是更新，有KEY和VALUE，如果是查询和删除，只有KEY。

接收：（分段数据长度（10B）+ （处理结果（1B）+VALUE长度（4B） + VALUE）\*N个）\*M段，直到接收到的分段数据长度为0结束。位置对应发送的顺序，如果是查询，有VALUE，更新和删除，只有处理结果。

#### 4.1.4.2管理端操作

管理端连接服务端的MANAGER\_LISTEN端口，是短连接，支持后面的操作。限定条件查询KEY操作（2101）、限定条件查询VALUE操作（2102）、限定条件查询KEY和VALUE操作（2103）、查询最小KEY操作（2104）、查询最大KEY操作（2105）、限定条件查询KEY数量操作（2106）、限定条件删除操作（2201）、限定条件导出数据到文件操作（2301）、从文件导入数据操作（2401）、正常结束进程操作（2501）、将内存数据写入到硬盘操作（2601）、增加线程数量操作（2701）、增加文件最大值操作（2702）、修改文件递增值操作（2703）、显示内存使用情况操作（2801）、显示硬盘使用情况操作（2802）、显示长连接和短连接线程数量操作（2803）、显示key数量操作（2804）。

管理端只对从属数据操作，即对硬盘的数据进行限定搜索，在执行条件限定时，可以先执行将内存数据写入到硬盘操作，再执行对应的操作。

条件限定，类似于SQL中的WHERE关键字，但这里只支持简单的条件限定，可以用 WHERE [KEY1， KEY2）来说明，“[”是数学里面的符号，数学术语称为闭区间，放在左边表示大于或等于，放在右边的符号为“]”，表示小于或等于。“)”也是数学里面的符号，数学术语称为开区间，放在右边表示小于，放在左边的符号为“(”，表示大于。这个条件限定表示，查询的结果大于等于KEY1，并且小于KEY2。这里，对查询结果的条件限定，只有9中情况。分别是，没有限定条件、[KEY1，KEY2]、[KEY1，KEY2)、(KEY1，KEY2)、(KEY1，KEY2]、[VALUE1，VALUE2]、[VALUE1，VALUE2)、(VALUE1，VALUE2)、(VALUE1，VALUE2]。

没有限定条件（0000）：表示查询全部数据。

KEY条件限定，左开右开（0001）：对应(KEY1，KEY2)，表示查询结果大于KEY1，并且小于KEY2。

KEY条件限定，左开右闭（0002）：对应(KEY1，KEY2]，表示查询结果大于KEY1，并且小于或者等于KEY2。

KEY条件限定，左闭右开（0003）：对应[KEY1，KEY2)，表示查询结果大于或者等于KEY1，并且小于KEY2。

KEY条件限定，左闭右闭（0004）：对应[KEY1，KEY2]，表示查询结果大于或者等于KEY1，并且小于或者等于KEY2。

VALUE条件限定，左开右开（0005）：对应(VALUE1，VALUE2)，表示查询结果大于VALUE1，并且小于VALUE2。

VALUE条件限定，左开右闭（0006）：对应(VALUE1，VALUE2]，表示查询结果大于VALUE1，并且小于或者等于VALUE2。

VALUE条件限定，左闭右开（0007）：对应[VALUE1，VALUE2)，表示查询结果大于或者等于VALUE1，并且小于VALUE2。

VALUE条件限定，左闭右闭（0008）：对应[VALUE1，VALUE2]，表示查询结果大于或者等于VALUE1，并且小于或者等于VALUE2。

##### 4.1.4.2.1限定条件查询KEY

###### 4.1.4.2.1.1第一次查询

发送：数据总长度（10B）+2101+查询ID（20B）+获取数量（20B）+获取开始位置（20B）+条件限定类型（4B）+条件开始值长度（4B）+条件开始值+条件结束值长度（4B）+条件结束值。这里的查询ID为20个0。

接收：总返回长度（10B）+操作结果（1B）+查询ID（20B）+返回数量（20B）+（KEY长度（4B）+KEY值）\*返回数量。这里的查询ID是服务端返回的，用作继续查询。

###### 4.1.4.2.1.2继续查询

发送：数据总长度（10B）+2101+查询ID（20B）+获取数量（20B）+获取开始位置（20B）。

接收：总返回长度（10B）+操作结果（1B）+查询ID（20B）+返回数量（20B）+（KEY长度（4B）+KEY值）\*返回数量。

##### 4.1.4.2.2限定条件查询VALUE

###### 4.1.4.2.2.1第一次查询

发送：数据总长度（10B）+2102+查询ID（20B）+获取数量（20B）+获取开始位置（20B）+条件限定类型（4B）+条件开始值长度（4B）+条件开始值+条件结束值长度（4B）+条件结束值。这里的查询ID为20个0。

接收：总返回长度（10B）+操作结果（1B）+查询ID（20B）+返回数量（20B）+（VALUE长度（4B）+VALUE值）\*返回数量。这里的查询ID是服务端返回的，用作继续查询。

###### 4.1.4.2.2.2继续查询

发送：数据总长度（10B）+2102+查询ID（20B）+获取数量（20B）+获取开始位置（20B）。

接收：总返回长度（10B）+操作结果（1B）+查询ID（20B）+返回数量（20B）+（VALUE长度（4B）+VALUE值）\*返回数量。

##### 4.1.4.2.3限定条件查询KEY和VALUE

###### 4.1.4.2.3.1第一次查询

发送：数据总长度（10B）+2103+查询ID（20B）+获取数量（20B）+获取开始位置（20B）+条件限定类型（4B）+条件开始值长度（4B）+条件开始值+条件结束值长度（4B）+条件结束值。这里的查询ID为20个0。

接收：总返回长度（10B）+操作结果（1B）+查询ID（20B）+返回数量（20B）+（KEY长度（4B）+KEY值+VALUE长度（4B）+VALUE值）\*返回数量。这里的查询ID是服务端返回的，用作继续查询。

###### 4.1.4.2.3.2继续查询

发送：数据总长度（10B）+2103+查询ID（20B）+获取数量（20B）+获取开始位置（20B）。

接收：总返回长度（10B）+操作结果（1B）+查询ID（20B）+返回数量（20B）+（KEY长度（4B）+KEY值+VALUE长度（4B）+VALUE值）\*返回数量。

##### 4.1.4.2.4查询最小KEY

发送：数据总长度（10B）+2104

接收：总返回长度（10B）+操作结果（1B）+KEY值长度（4B）+KEY值+VALUE值长度（4B）+VALUE值。如果操作结果成功，有KEY和VALUE，不成功就没有。

发送：数据总长度（10B）+2102+查询ID（20B）+获取数量（20B）+获取开始位置（20B）。

接收：总返回长度（10B）+操作结果（1B）+查询ID（20B）+返回数量（20B）+（VALUE长度（4B）+VALUE值）\*返回数量。

##### 4.1.4.2.5查询最大KEY

发送：数据总长度（10B）+2105

接收：总返回长度（10B）+操作结果（1B）+KEY值长度+KEY值+VALUE值长度+VALUE值。如果操作结果成功，有KEY和VALUE，不成功就没有。

##### 4.1.4.2.6限定条件查询KEY数量

发送：数据总长度（10B）+2106+条件限定类型（4B）+KEY开始值长度（4B）+KEY开始值+KEY结束值长度（4B）+KEY结束值。

接收：总返回长度（10B）+操作结果（1B）+数量值（20B）。如果操作结果成功，有数量值，不成功就没有。

##### 4.1.4.2.7限定条件删除

发送：数据总长度（10B）+2201+条件限定类型（4B）+KEY开始值长度（4B）+KEY开始值+KEY结束值长度（4B）+KEY结束值。

接收：总返回长度（10B）+操作结果（1B）+删除数量（20B）。如果操作结果成功，有删除数量值，不成功就没有。

##### 4.1.4.2.8限定条件导出数据到文件

发送：数据总长度（10B）+2301+导出到文件路径名称（256B）+条件限定类型（4B）+KEY开始值长度（4B）+KEY开始值+KEY结束值长度（4B）+KEY结束值。

接收：总返回长度（10B）+操作结果（1B）+导出数量（20B）。如果操作结果成功，有导出数量值，不成功就没有。

文件的数据格式：每个KEY或者VALUE放在单引号对里面，每个KEY-VALUE对用换行符来分开，如下面。

'key1' 'value1'

'key2' 'value2'

'key3' 'value3'

'key4' 'value4'

##### 4.1.4.2.9从文件导入数据

发送：数据总长度（10B）+2401+导入到文件路径名称（256B）+用户名（64B）+用户密码（64B）。

接收：总返回长度（10B）+操作结果（1B）+导入数量（20B）。如果操作结果成功，有导入数量值，不成功就没有。

文件的数据格式：每个KEY或者VALUE放在单引号对里面，每个KEY-VALUE对用换行符来分开，如下面。

'key1' 'value1'

'key2' 'value2'

'key3' 'value3'

'key4' 'value4'

##### 4.1.4.2.10正常结束进程

发送：数据总长度（10B）+2501+用户名（64B）+用户密码（64B）。

接收：总返回长度（10B）+操作结果（1B）。

##### 4.1.4.2.11将内存数据写入到对应的映射文件

发送：数据总长度（10B）+2601+用户名（64B）+用户密码（64B）。

接收：总返回长度（10B）+操作结果（1B）。

##### 4.1.4.2.12修改线程数量

发送：数据总长度（10B）+2701+短连接线程数量（20B）+长连接线程数量（20B）+用户名（64B）+用户密码（64B）。

接收：总返回长度（10B）+操作结果（1B）。

##### 4.1.4.2.13增加文件最大值

发送：数据总长度（10B）+2702+文件最大值（20B）+用户名（64B）+用户密码（64B）。

接收：总返回长度（10B）+操作结果（1B）。

##### 4.1.4.2.14修改文件递增值

发送：数据总长度（10B）+2703+文件步增值（20B）+用户名（64B）+用户密码（64B）。

接收：总返回长度（10B）+操作结果（1B）。

##### 4.1.4.2.15显示内存使用情况

发送：数据总长度（10B）+2801。

接收：总返回长度（10B）+操作结果（1B）+ KEY文件缓冲区总数量（20B） + KEY文件缓冲区的使用偏移量（单位 B）（20B） + VALUE文件缓冲区总数量（20B） + VALUE文件缓冲区的使用偏移量（单位 B）（20B）。如果操作结果成功，有后面数据，不成功就没有。

##### 4.1.4.2.16显示硬盘使用情况

发送：数据总长度（10B）+2802。

接收：总返回长度（10B）+操作结果（1B）+ KEY文件最大值（20B）+KEY文件当前值（20B）+KEY文件步增值（20B）+ KEY文件使用偏移量（20B） + KEY文件待使用大小（20B）+VALUE文件最大值（20B）+VALUE文件当前值（20B）+VALUE文件步增值（20B）+ VALUE文件使用偏移量（20B）+VALUE文件待使用大小（20B）。如果操作结果成功，有后面数据，不成功就没有。

##### 4.1.4.2.17显示长连接和短连接线程数量

发送：数据总长度（10B）+2803。

接收：总返回长度（10B）+操作结果（1B）+短连接线程数量（20B）+长连接线程数量（20B）。如果操作结果成功，有后面数据，不成功就没有。

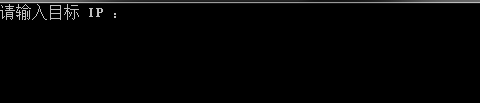
##### 4.1.4.2.18显示KEY数量

发送：数据总长度（10B）+2804。

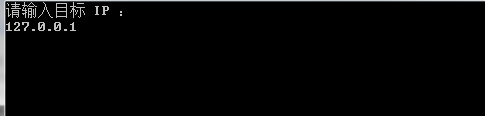
接收：总返回长度（10B）+操作结果（1B）+ KEY总数量（20B）。如果操作结果成功，有后面数据，不成功就没有。

## 界面客户端

双击界面客户端程序stormGodUser.exe，出现下面的界面。



输入服务端的IP地址，然后按回车键



输入端口，然后按回车键



在下面开始输入操作语句。这个客户端支持3种语句，SET KEY VALUE、GET KEY、DEL KEY。关键字SET、GET、DEL不区分大小写，KEY、VALUE这些变量区分大小写。

下面是一些执行结果。



输入语句“set key1 value1”后按回车键，显示“set data success”，表示设置数据成功。

输入语句“get key1”后按回车键，显示“value1”，表示key1对应的是value1。

输入语句“set key1 value2”后按回车键，显示“set data success”，表示设置数据成功。

输入语句“get key1”后按回车键，显示“value2”，表示key1对应的是value2。

输入语句“del key1”后按回车键，显示“del data success”，表示删除数据成功。

输入语句“get key1”后按回车键，显示“null”，表示key1不存在。

## 应用静态库

这个静态库的函数很多，但是，面对应用程序的函数不多，只有6个。编译的时候需要加“libws2\_32.a”所在的全路径，比如编译1.c，并且文件libws2\_32.a在1.c同一个目录下，就可以执行“gcc -o 1 1.c stormGodClient.a libws2\_32.a”来编译。

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名称: keyValueMalloc

函数功能: 申请内存

参 数：

第 一：申请大小 I

返 回：内存地址

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void \*keyValueMalloc( int iaSize );

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名称: keyValueFree

函数功能: 释放用keyValueMalloc分配的内存

参 数：

第 一：申请过的地址 I

返 回：

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void keyValueFree( void \*pcaAddress );

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名称: keyValueClientShort

函数功能: 短连接操作

参 数：

第 一：服务端IP I

第 二：服务端端口 I

第 三：数据地址 I/O

第 四：数据大小 I/O

返 回：0是成功，其他失败

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int keyValueClientShort( char \*pcaIp, char \*pcaPort, void \*\*pcaData, int \*iaDataLen );

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名称: keyValueClientLongInti

函数功能: 长连接初始化

参 数：

第 一：服务端IP I

第 二：服务端端口 I

返 回：PTHREAD ID，正数表示成功，负数表示失败

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

pthread\_t keyValueClientLongInti( char \*pcaIp, char \*pcaPort );

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名称: keyValueClientLongRls

函数功能: 释放长连接资源

参 数：

第 一：线程ID I

返 回：

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void keyValueClientLongRls(pthread\_t threadID);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名称: keyValueClientLong

函数功能: 长连接操作

参 数：

第 一：线程ID I

第 二：数据地址 I/O

第 三：数据大小 I/O

返 回：0是成功，其他失败

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int keyValueClientLong( pthread\_t pthreadID, void \*\*pcaData, int \*iaDataLen );

pcaData变量用作输入输出，调用keyValueClientShort、keyValueClientLong之前，要先用 keyValueMalloc来分配，调用之后要用keyValueFree来释放。iaDataLen表示pcaData的总长度。

pcaData变量的输入部分，支持下面7种，set、get、del、setlist、getlist、list是关键字。

单条更新：set key value

单条删除：del key

单条查询：get key

批量更新：setlist key1 value1; key2 value2; ……用分号作为分割符

批量删除：dellist key1; key2; ……用分号作为分割符

批量查询：getlist key1; key2; ……用分号作为分割符

批量混合：list set key value; get key; del key; ……用分号作为分割符

pcaData变量的输出部分如下。

单条更新：处理结果（1B）

单条删除：处理结果（1B）

单条查询：处理结果（1B）+VALUE

批量更新：处理结果（1B）\*N

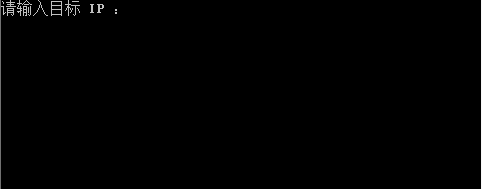
批量删除：处理结果（1B）\*N

批量查询：处理结果（1B）+（KEY长度+KEY+VALUE长度+VALUE）\*N

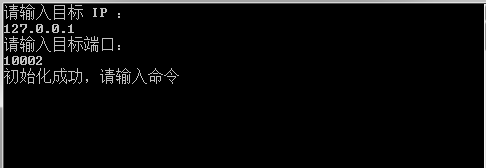
批量混合：（处理结果（1B）+VALUE长度（4B） + VALUE）\*N

## 管理端

双击stormGodServer.exe就会出现如下界面。



根据提示，输入服务端IP和端口



到这里，就可以开始输入命令了。这里支持上面4.1.4.2提到的。

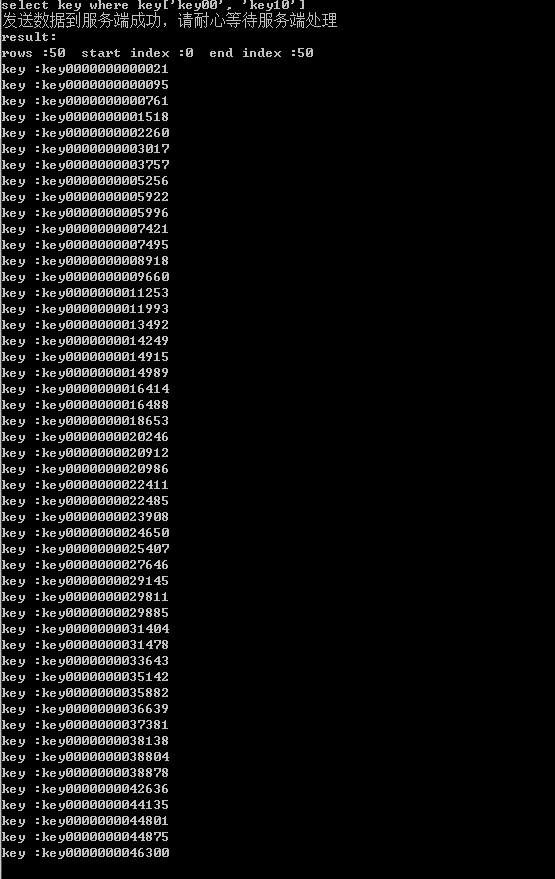
管理端输入的语句，为了演示方便，大小字母为关键字，小写字母为变量。实际上，关键字也可以小写字母，大小写字母混合。关键字之间需要空格分开，单引号之间的值表示变量的值，符合[、(、)、]分别表示大于等于、大于、小于、小于等于。

限定查询语句，第一次获取结果后，可以输入键N来取下一批，输入L键获取上一批，输入F键结束。每批数据取50条。

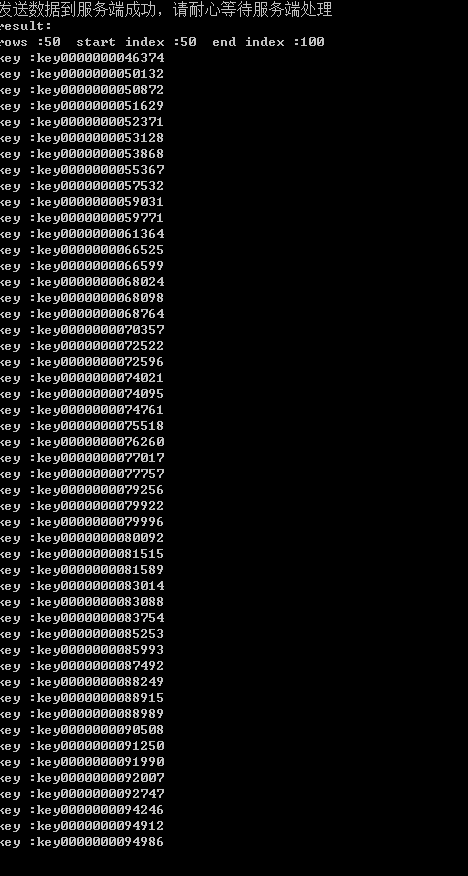
### 限定条件查询KEY

示例：SELECT KEY WHERE KEY['a','b']

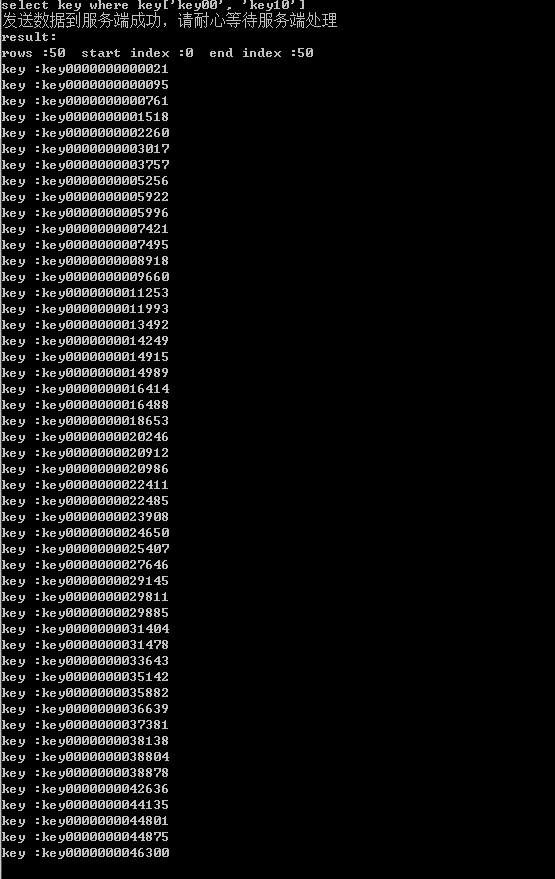
执行结果如下



输入N键再取下一批数据



输入L键取上一批数据，也就是刚开始执行时的数据

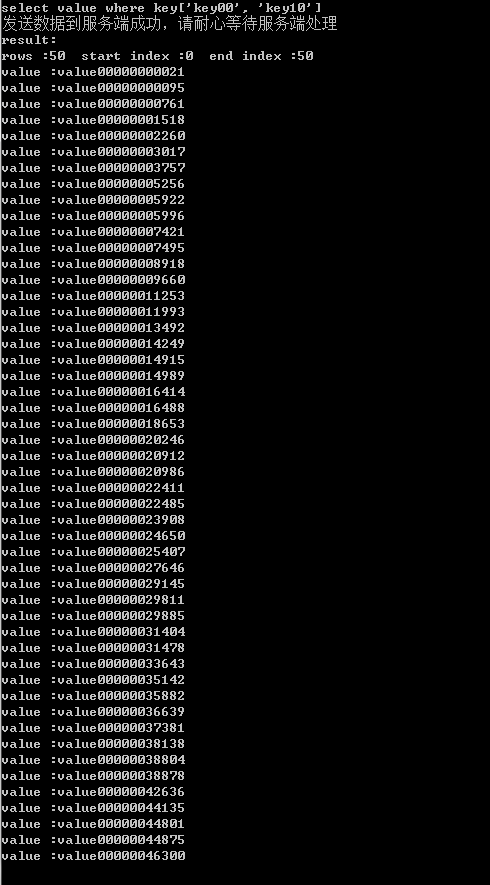


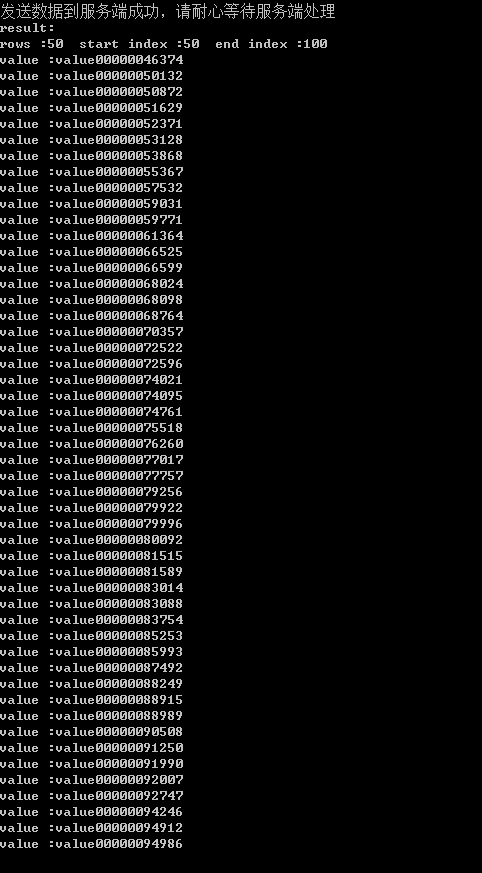
输入F键结束，显示“查询完成”

### 限定条件查询VALUE

示例：SELECT VALUE WHERE KEY['a','b']

用法和上面的一样，下面分别是第一次查询、获取下一批的结果。

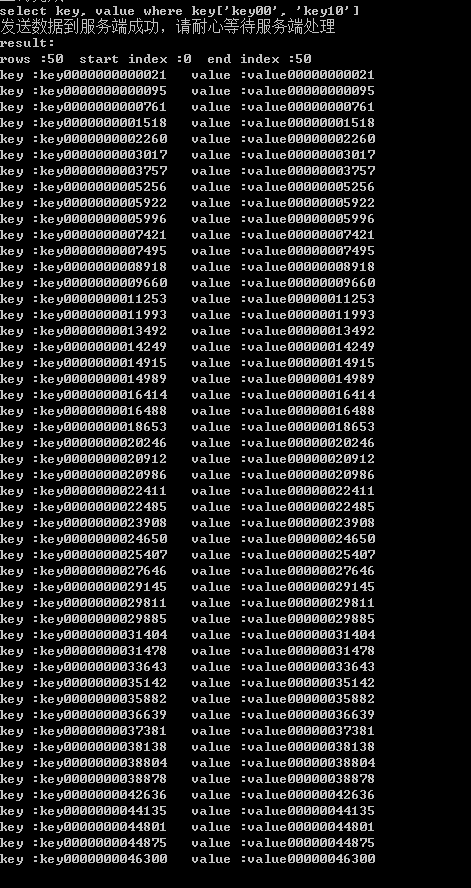


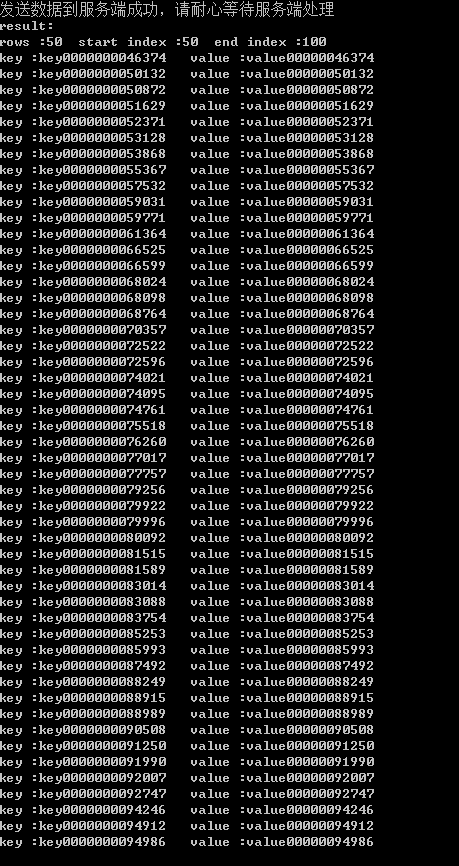


### 限定条件查询KEY和VALUE

示例：SELECT KEY, VALUE WHERE KEY['a','b']

用法和上面的一样，下面分别是第一次查询、获取下一批的结果。

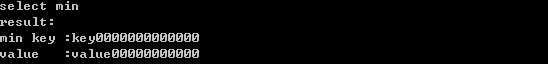




### 查询最小KEY

示例：SELECT MIN

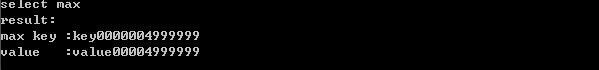
执行结果如下



### 查询最大KEY

示例：SELECT MAX

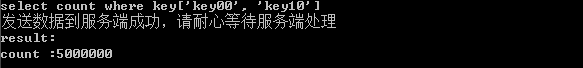
执行结果如下



### 限定条件查询KEY数量

示例：SELECT COUNT WHERE KEY ['key00', 'key10']

执行结果如下

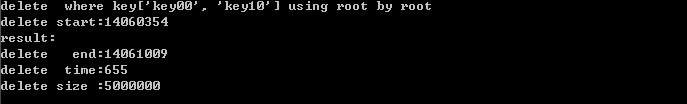


### 限定条件删除

示例：DELETE WHERE KEY ['key00', 'key10'] USING user BY password

USING 用户名 BY 密码，这两个值在启动配置文件有。

执行结果如下



实际上只会显示delete size:5000000，我这里添加了delete start、delete end、delete time显项，是为了看批量删除效率，清理500万个KEY-VALUE对，也就是1000万个数据，只花了655毫秒，程序对数据的回收速度达到763万个KEY-VALUE对每秒，效率还是很高的。

### 限定条件导出数据到文件

示例：UNLOAD TO file WHERE KEY ['a','b']

执行结果如下

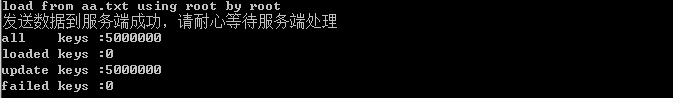
H:\keyValue\使用文档\u1.PNG

打开文件“aa.txt”，可以看到500万行数据

### 从文件导入数据

示例：LOAD FROM file USING user BY password

执行结果如下



### 正常结束进程

示例：EXIT USING user BY password

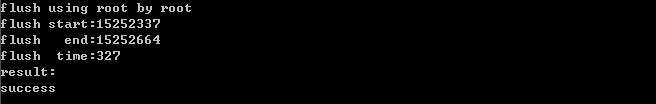
执行结果如下



### 将内存数据写入到硬盘

示例：FLUSH USING user BY password

执行结果如下



实际只显示 success，我增加了其他是为了看效率如何。

### 修改线程数量

示例：ALTER THREADS TO size1, size2 USING user BY password

第一个size1是短连接线程的数量，第二个size2是长连接线程的数量。

执行结果如下

H:\keyValue\使用文档\a1.PNG

此时，可以用命令 show threads命令，或者从系统任务管理器看到，总线程数量=1个管理侦听线程+1个异步日志线程+1个主线程+短连接线程+长连接线程

### 增加文件最大值

示例：ALTER DISCS TO size USING user BY password

这里的size的单位是MB

执行结果如下

H:\keyValue\使用文档\a2.PNG

执行完可以用 show discs命令看结果

### 修改文件递增值

示例：ALTER UPPERS TO size USING user BY password

这里的size的单位是MB

执行结果如下

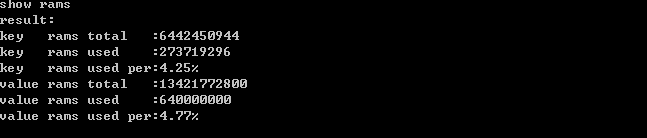
H:\keyValue\使用文档\a3.PNG

执行完可以用 show discs命令看结果

### 显示内存使用情况

示例：SHOW RAMS

执行结果如下

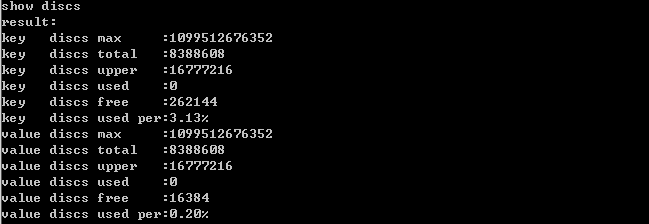


除了百分比，其他单位是B，也就是字节。

### 显示硬盘使用情况

示例：SHOW DISCS

执行结果如下

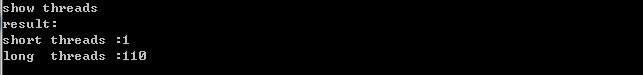


除了百分比，其他的单位是B，也就是字节。

### 显示长连接和短连接线程数量

示例：SHOW THREADS

执行结果如下



### 显示KEY数量

示例：SHOW KEYS

执行结果如下

H:\keyValue\使用文档\sk1.PNG