|  |
| --- |
| **zstorage云存储服务器操作设计说明** |

**修订历史**

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2015-01-01 | 1.0.1 | 完成模块设计 | 郑坚强 |
| 2015-01-01 | 1.0.1 | 完成模块编码 | 郑坚强 |
| 2015-03-27 | 1.0.1 | 完成文档 | 郑坚强 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**目录**

zstorage云存储服务器操作设计说明 1

版权申明 1

修订历史 2

目录 3

zstorage云存储服务器操作 5

设计说明书 5

1. 概述 ( Overview ) 5

1.1 范围 ( Scope ) 5

1.2 发布 ( Issue ) 5

1.3 综述 ( Brief ) 5

2. 定义 ( Definition ) 5

2.1 术语 ( Terminologies ) 5

2.2 约束 ( Constraints ) 5

3. 使用说明 5

3.1 基本用法 6

3.1.1 SET 6

3.1.2 GET 6

3.1.3 EXISTS 7

3.1.4 DEL 7

3.2 LIST队列 8

3.2.1 PUSH 8

3.2.2 POP 8

3.2.3 SIZE 9

3.2.4 LIMIT 9

3.2.5 UPDATE 10

3.2.6 SEEK 10

3.2.7 EXPIRE 11

3.2.8 TIME 11

3.2.9 DEL 12

3.3 MLIST键值队列 12

3.3.1 PUSH 12

3.3.2 SIZE 13

3.3.3 SEEK 13

3.3.4 EXPIRE 13

3.3.5 TIME 14

3.3.6 DEL 14

3.4 SEQ序列 15

3.4.1 PUSH 15

3.4.2 SIZE 15

3.4.3 INDEX 16

3.4.4 SEEK 16

3.4.5 QUERY 17

3.4.6 DELAY 17

3.4.7 LOOKDELAY 18

3.5 CLOUD云消息 18

3.5.1 PUSH 18

3.5.2 LIMIT 18

3.5.3 SIZE 19

3.5.4 INDEX 19

3.5.5 QUERY 20

3.5.6 ONLINE 20

3.5.7 READ 20

3.6 云服务器测试用例 21

3.6.1 测试环境 21

3.6.2 预览好友消息 21

**zstorage云存储服务器操作**

**设计说明书**

* **使用说明**

***独立模块***

**用法：**

TTL {id} KEY {keyword} ...

*id为序号，序号可任意表示，keyword为KEY的内容。*

***为了便于说明***,***以下示例中****id****都用0表示。***

**返回值说明：**



图3 返回值

|  |  |
| --- | --- |
| **参数** | **含义** |
| \*3 | 代表返回字段的个数为3：$1 $2 $3 |
| $1 7 | 对应id的值为7 |
| $2 OK | OK 代表命令执行成功 |
| $3 007 | 007 为关键字key\_test的值 |

* **基本用法**

* **SET**

**用法：**

TTL {id} KEY {keyword} SET {value}

设置关键字keyword的值为value,成功返回OK。

TTL {id} KEY {keyword} SET {value} EXPIRE {time\_value}

设置关键字keyword的值为value,time\_value秒后自动清除,成功返回OK。

**示例：**

TTL 0 KEY key\_test SET 101

设置关键字key\_test的值为101,成功返回OK（如图3.1.1所示）。

TTL 0 KEY key\_test SET 101 EXPIRE 20

设置关键字key\_test的值为101,20秒后自动清除 (key\_test不再存在)。



图3.1.1 SET示例

* **GET**

**用法：**

TTL {id} KEY {keyword} GET

获取关键字keyword所对应值。

**示例：**

TTL 0 KEY key\_test GET

获取关键字key\_test所对应值（如图3.1.2所示）。



图3.1.2 GET示例

* **EXISTS**

**用法：**

TTL {id} KEY {keyword} EXISTS

查看关键字keyword是否存在，存在返回1，反之返回0。

**示例：**

TTL 0 KEY key\_test EXISTS

查看关键字key\_test是否存在，存在返回1，反之返回0（如图3.1.3所示）。



图3.1.3 EXISTS示例

* **DEL**

**用法：**

TTL {id} KEY {keyword} DEL

删除关键字keyword,成功返回OK。

**示例：**

TTL 0 KEY key\_test DEL

删除关键字key\_test,成功返回OK。



图3.1.4 DEL示例

* **LIST队列**
* **PUSH**

**用法：**

TTL {id} KEY {list\_name} LIST PUSH {value}

往队列list\_name中压入一个值value，成功返回OK。

**示例：**

TTL 0 KEY list\_test LIST PUSH 001

往队列list\_test中压入一个值001，成功返回OK。



图3.2.1 PUSH示例

* **POP**

**用法：**

TTL {id} KEY {list\_name} LIST POP

弹出队列list\_name中最前面一个值，并在队列中移除,成功返回删除的值。

**示例：**

TTL 0 KEY list\_test LIST POP

弹出队列list\_test中最前面一个值001，并在队列中移除。



图3.2.2 POP示例

* **SIZE**

**用法：**

TTL {id} KEY {list\_name}LIST SIZE

获取队列list\_name的长度,成功返回队列的长度。

**示例：**

TTL 0 KEY list\_test LIST SIZE

获取队列list\_test的长度，返回队列的长度。



图3.2.3 SIZE示例

* **LIMIT**

**用法：**

TTL {id} KEY {list\_name} LIST LIMIT {num}

num<0时返回队列list\_name的所有值，反之返回队列的前num个值。

**示例：**

TTL 0 KEY list\_test LIST LIMIT -1

获取队列list\_test所有值。

TTL 0 KEY list\_test LIST LIMIT 2

获取队列list\_test的前2个值。



图3.2.4 LIMIT示例

* **UPDATE**

**用法：**

TTL {id} KEY {list\_name} LIST UPDATE {var\_1} {var\_2} {...}

把队列list\_name更新为最新数据var\_1,var\_2等，老数据清空,成功返回OK。

**示例：**

TTL 0 KEY list\_test LIST UPDATE 01 02 03

把队列list\_test更新为最新数据01,02,03，老数据清空。



图3.2.5 UPDATE示例

* **SEEK**

**用法：**

TTL {id} KEY {list\_name} LIST SEEK {start\_num} LIMIT {sum}

从队列list\_name的第***start\_num+1***条开始，获取sum(>0)条最新记录。

**示例：**

TTL 0 KEY list\_test LIST SEEK 0 LIMIT 3

从队列list\_test的第1条开始，获取3条最新记录。



图3.2.6 SEEK示例

* **EXPIRE**

**用法：**

TTL {id} KEY {list\_name} LIST EXPIRE {time}

设置队列list\_name在time秒钟后自动删除，成功返回OK。

**示例：**

TTL 0 KEY list\_test LIST EXPIRE 60

设置队列list\_test在60秒后自动删除。



图3.2.7 EXPIRE示例

* **TIME**

**用法：**

TTL {id} KEY {list\_name} LIST TIME

获取队列list\_name的生命周期（单位秒）,成功返回剩余的秒数。

**示例：**

TTL 0 KEY list\_test LIST TIME

获取队列list\_test的生命周期（单位秒）。



图3.2.8 TIME示例

* **DEL**

**用法：**

TTL {id} KEY {list\_name} LIST DEL

删除队列list\_name，成功返回OK。

**示例：**

TTL 0 KEY list\_test LIST DEL

删除队列list\_test。



图3.2.9 DEL示例

* **MLIST键值队列**
* **PUSH**

**用法：**

TTL {id} KEY {mlist\_name} MLIST PUSH {mkey\_name} {mvalue}

往队列mlist\_name中压入键值对{mkey\_name,mvalue},成功返回OK。

**示例：**

TTL 0 KEY mlist\_test MLIST PUSH mk\_1 001

往队列mlist\_test中压入键值对{mk\_1,001}。



图3.3.1 PUSH示例

* **SIZE**

**用法：**

TTL {id} KEY {mlist\_name} MLIST SIZE

获取队列mlist\_name的大小。

**示例：**

TTL 0 KEY mlist\_test MLIST SIZE

获取队列mlist\_test的大小。



图3.3.2 SIZE示例

* **SEEK**

**用法：**

TTL {id} KEY {mlist\_name} MLIST SEEK {mkey\_name} {sum}

从队列mlist\_name中获取从***mkey\_name后一个记录开始***的sum（sum>0）个记录。

**示例：**

TTL 0 KEY mlist\_test MLIST SEEK mk\_1 3

从队列mlist\_test中获取从mk\_1***后一个记录开始***的3个记录。



图3.3.3 SEEK示例

* **EXPIRE**

**用法：**

TTL {id} KEY {mlist\_name} MLIST EXPIRE {time}

设置队列mlist\_name,time秒后自动删除,成功返回OK。

**示例：**

TTL 0 KEY mlist\_test MLIST EXPIRE 30

设置队列mlist\_test,30秒后自动删除。



图3.3.4 EXPIRE示例

* **TIME**

**用法：**

TTL {id} KEY {mlist\_name} MLIST TIME

获取队列mlist\_name的生命周期（单位秒）,返回剩余的秒数。

**示例：**

TTL 0 KEY mlist\_test MLIST TIME

获取队列mlist\_test的生命周期（单位秒）。



图3.3.5 TIME示例

* **DEL**

**用法：**

TTL {id} KEY {mlist\_name} MLIST DEL

删除队列mlist\_name，成功返回OK。

**示例：**

TTL 0 KEY mlist\_test MLIST DEL

删除队列mlist\_test。



图3.3.6 DEL示例

* **SEQ序列**
* **PUSH**

**用法：**

TTL {id} KEY {seq\_name} SEQ PUSH {value}

将内容value压入序列seq\_name中，返回当前序号编号。

**示例：**

TTL 0 KEY seq\_test SEQ PUSH v001

将内容v001压入序列seq\_test中，返回当前序号编号。



图3.4.1 PUSH示例

* **SIZE**

**用法：**

TTL {id} KEY {seq\_name} SEQ SIZE

获取当前序列seq\_name的大小。

**示例：**

TTL 0 KEY seq\_test SEQ SIZE

获取当前序列seq\_test的大小。



图3.4.2 SIZE示例

* **INDEX**

**用法：**

TTL {id} KEY {seq\_name} SEQ INDEX

获取当前序列seq\_name的编号。

**示例：**

TTL 0 KEY seq\_test SEQ INDEX

获取当前序列seq\_test的编号。



图3.4.3 INDEX示例

* **SEEK**

**用法：**

TTL {id} KEY {seq\_name} SEQ SEEK {keyId} {count}

从序列keyId(1、2、3 ... ***0代表为最后一个记录的位置***)位置获取count个记录数。count为正时向下获取， count为负时向上获取。

**示例：**

TTL 0 KEY seq\_test SEQ SEEK 1 2

从序列1位置向下获取2个记录数。

TTL 0 KEY seq\_test SEQ SEEK 3 -2

从序列3位置向上获取2个记录数。

TTL 0 KEY seq\_test SEQ SEEK 0 -3

从序列的最后一个位置开始向上获取3个记录数。



图3.4.4 SEEK示例

* **QUERY**

**用法：**

TTL {id} KEY {seq\_name} SEQ QUERY {yyyy-mm-dd hh:mm:ss}

查询时间yyyy-mm-dd hh:mm:ss之后离查询时间最近的一条消息的ID。

**示例：**

TTL 0 KEY seq\_test SEQ QUERY 2015-3-27 10:24:02

查询2015-3-27 10:21:02之后离查询时间最近的一条消息ID。



图3.4.5 QWERY示例

* **DELAY**

**用法：**

TTL {id} KEY {seq\_name} SEQ DELAY {time}

设置SEQ窗口seq\_name time秒为超时。

**示例：**

TTL 0 KEY seq\_test SEQ DELAY 60

设置SEQ窗口60秒为超时。



图3.4.6 DELAY示例

* **LOOKDELAY**

**用法：**

TTL {id} KEY seq\_123 SEQ LOOKDELAY

查看当前窗口的延时（单位为秒）。

**示例：**

TTL 0 KEY seq\_test SEQ LOOKDELAY

查看当前窗口的延时的秒数。



图3.4.7 LOOKDELAY示例

* **CLOUD云消息**
* **PUSH**

**用法：**

TTL {id} KEY {cloud\_name} CLOUD PUSH {keyword} {value}

在云容器cloud\_name的子容器keyword中压入一个元素value。

**示例：**

TTL 0 KEY cloud\_1 CLOUD PUSH k1 0001

在云容器cloud\_1的子容器k1中压入一个元素0001。



图3.5.1 PUSH示例

* **LIMIT**

**用法：**

TTL {id} KEY {cloud\_name} CLOUD LIMIT {keyword} {index} {num}

在云容器cloud\_name的子容器keyword 中index(>=0 ***0代表最后一条记录的位置***)位置开始向下查询num条记录。

**示例：**

TTL 0 KEY cloud\_1 CLOUD LIMIT k1 1 3

在云容器cloud\_1的子容器k1 中1位置开始向下查询3条记录。

TTL 0 KEY cloud\_1 CLOUD LIMIT k1 0 -3

在云容器cloud\_1的子容器k1 中最后一个位置开始向上查询3条记录。



图3.5.2 LIMIT示例

* **SIZE**

**用法：**

TTL {id} KEY {cloud\_name} CLOUD SIZE {keyword}

获取云容器cloud\_name的子容器keyword中的记录数。

**示例：**

TTL 0 KEY cloud\_1 CLOUD SIZE k1

获取云容器cloud\_1的子容器k1中的记录数。



图3.5.3 SIZE示例

* **INDEX**

**用法：**

TTL {id} KEY {cloud\_name} CLOUD INDEX {keyword}

获取当前序列cloud\_name的编号。

**示例：**

TTL 0 KEY cloud\_1 CLOUD INDEX k1

获取当前序列k1的编号。



图3.5.4 INDEX示例

* **QUERY**

**用法：**

TTL {id} KEY {cloud\_name} CLOUD QUERY {keyword} {yyyy-mm-dd} {num}

在云容器cloud\_name的子容器keyword 中yyyy-mm-dd时间点位置向下查询num条记录。

**示例：**

TTL 0 KEY cloud\_1 CLOUD QUERY k1 2015-3-24 3

在云容器cloud\_1的子容器keyword中2015-3-24时间点位置向下查询3条记录。



图3.5.5 QUERY示例

* **ONLINE**

**用法：**

TTL {id} KEY {cloud\_name} CLOUD ONLINE {id,id,id,...}

云容器专用协议，用于IM上线。

**示例：**

TTL 0 KEY cloud10000 CLOUD ONLINE 10001,10002,10003,

(假设10000为自己的呀呀id号，10001 10002 10003为好友的id号)

上边指令代表的意思是自己上线时把好友的消息从服务器拉出来。

* **READ**

**用法：**

TTL {id} KEY {cloud\_name} CLOUD READ {keyword} {id}

如果消息已经阅读过，把消息编号id对应的内容写入云存储服务器。

**示例：**

TTL 0 KEY cloud\_1 CLOUD READ k1 2

如果消息已经阅读过，把消息编号2对应的内容写入云存储服务器。

* **云服务器测试用例**

**3.6.1 测试环境**  
 云服务器 ： telnet chat.yunva.com 7777

**3.6.2 预览好友消息**

**用法：**

TTL 0 KEY CLOUD{我的用户ID} CLOUD LIMIT {好友用户ID} {当前位置} {条数}   
 ***我的ID(1241304) 好友的ID(1241305)***

**示例：**  
 1)从最后开始拉取我的好友聊天记录10条。   
 TTL 0 KEY CLOUD1241304 CLOUD LIMIT 1241305 0 -10   
  
 2)从5位置向下拉10条消息。   
 TTL 0 KEY CLOUD1241304 CLOUD LIMIT 1241305 5 10   
  
 3)从最后开始拉取我的系统消息10条。   
 TTL 0 KEY CLOUD1241304 CLOUD LIMIT SYSTEM 0 -10   
  
 4)从最后开始拉取我的推送消息10条。    
 TTL 0 KEY CLOUD1241304 CLOUD LIMIT PUSH 0 -10   
  
**3.6.3 预览群消息**

**用法：**  
 TTL 0 KEY GROUP\_CLOUD{群ID} CLOUD LIMIT {群ID} {当前位置} {条数}

***群的ID(100491)***

**示例：**

TTL 0 KEY GROUP\_CLOUD100491 CLOUD LIMIT 100491 0 -10