**redis**

REmote DIctionary Server(Redis) 是一个由 Salvatore Sanfilippo 写的 key-value 存储系统，是跨平台的**非关系型数据库**。

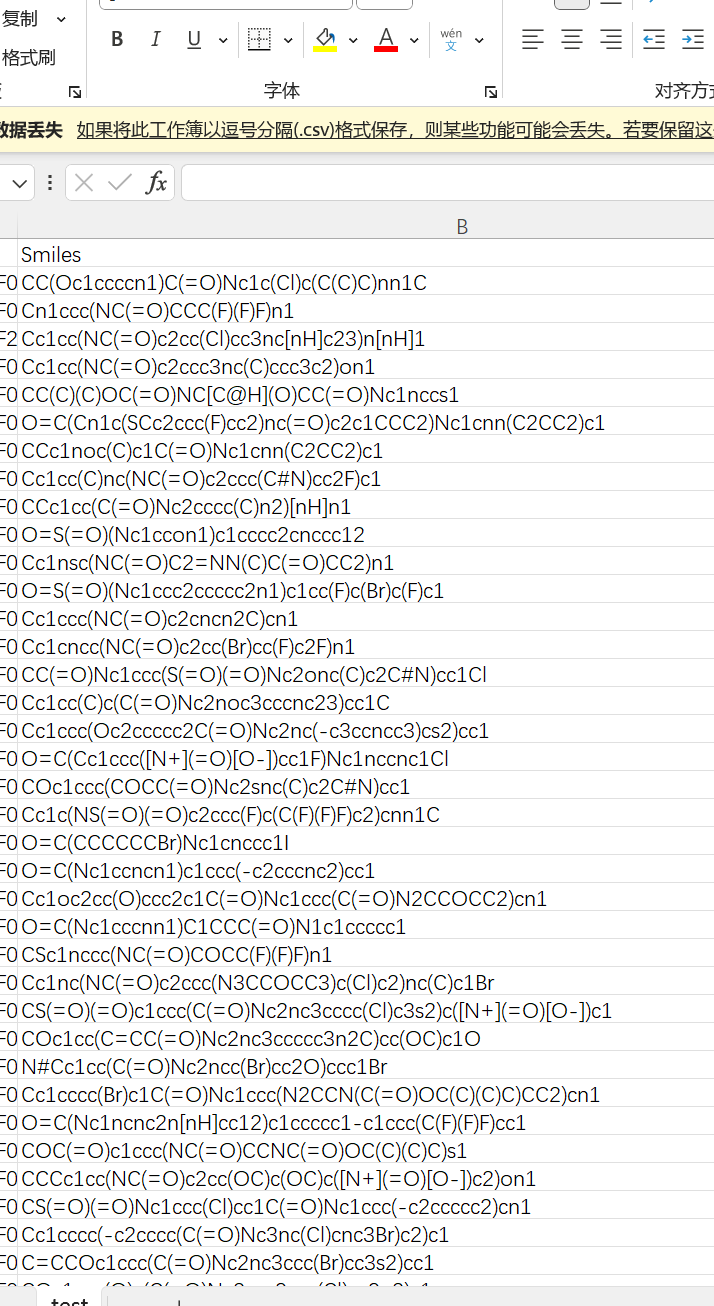
Redis 是一个开源的使用 ANSI C 语言编写、遵守 BSD 协议、支持网络、可基于内存、分布式、可选持久性的键值对**(Key-Value)**存储数据库，并提供多种语言的 API。

Redis 通常被称为数据结构服务器，因为值（value）可以是字符串(String)、哈希(Hash)、列表(list)、集合(sets)和有序集合(sorted sets)等类型。

**高速读写** – Redis能读的速度是110000次/s,写的速度是81000次/s 。

**具体场景**

**我有一个具有100000条数据的表格文件，现在高速地要对每一条数据进行处理，并保存。**

****

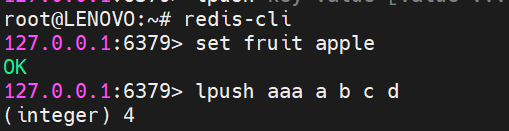
如果直接操作这个文件，速度会比较慢。

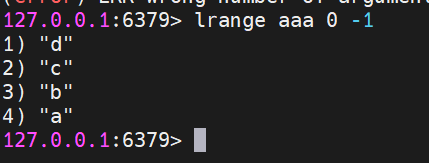
**我们可以借助redis**

1. **文件数据全部写入redis,以队列地形式储存，**
2. **通过redis队列分发处理**
3. **再存入另一个redis。(一个redis又读又写可能会内存爆炸)**
4. **最后写回文件。**

**一些简单的例子：**

**通过redis-cli客户端操作：**



****

**Java，Python的redis库**

r = redis.Redis(host=redisHost2，port=6379)

**就可以通过r使用r.set（） r.lpush（）...这些方法，像上面客户端一样来读写数据了。**

**Redis的数据类型**

**Redis五种基本数据类型**

**String字符串**

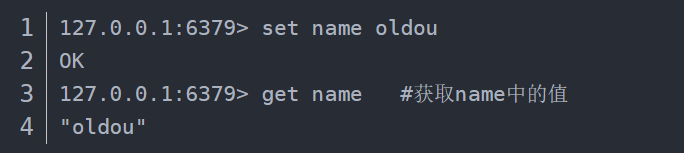
**Redis 字符串是字节序列。Redis 字符串是二进制安全的，这意味着他们有一个已知的长度没有任何特殊字符终止，所以你可以存储任何东西，512 M为上限，主要的还是操作键值对。**

**String的数据结构是简单的Key-Value模型，Value可以是字符串，也可以是数字。**

**常用命令**

**添加元素（SET命令）**

**获取值 （GET命令）**

****

**mset/mget命令**

**同时设置/获取一个或者多个键值对，主要就是批量设置和获取键值对。**

**同时设置多个值，如果其中有一个存在，那么就都创建失败。**

**要么一起成功，要么一起失败。**

**incr命令**

**让当前键值以 1 的数量递增，并返回递增后的值。相当于Java中的自增，每次使用改命令都能让变量自增1。**

**incrby命令**

**可以指定参数一次增加的数值，并返回递增后的值，(原来的num是2，加10之后变为12**

**decr命令**

**可以指定参数一次递减的数值，并返回递减后的值，每次使用该命令都自减1，相当于Java中的自减。**

**append命令**

**向键值的末尾追加 value。如果键不存在则将该键的值设置为 value。**

**返回值是追加后字符串的总长度。**

**strlen命令 ：获取字符串长度**

**\*\*getrange命令：\*\*截去指定索引的字符串**

**setrange命令：从指定索引开始替换字符串**

**Hash散列表**

**Redis 的哈希是键值对的集合。**

**Redis 的哈希值是字符串字段和字符串值之间的映射，因此它们被用来表示对象，还有用户信息之类的，经常变动的信息。**

**Hash更适合用于对象的存储，String更适合字符串存储。**

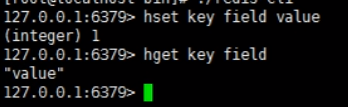
**常用命令**

**（1）hset命令：存储一个哈希键值对的集合**

**格式为：hset key field value**

**（2）hget命令：获取一个哈希键的值**

**格式为：hget key field**



**（3）hmset： 存储一个或多个哈希是键值对的集合**

**格式为：hmset key field1 value1 …fieldN keyN**

**（4）hmget： 获取多个指定的键的值**

**格式为：hmget key field1 … fieldN**

**(5)hexists： 判断哈希表中的字段名是否存在 如果存在返回 1 否则返回 0**

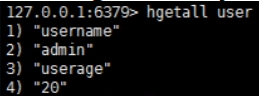
**格式为：hexists key field**

**（6）hdel： 删除一个或多个字段**

**格式为：hdel key field**

**（7）hgetall： 获取一个哈希是键值对的集合**

**格式为：hgetall key**



**（8）hvals： 只返回字段值**

**格式为：hvals key**

**（9）hkeys： 只返回字段名**

**格式为：hkeys key**

**（10）hlen： 返回 key 的 hash 的元素个数**

**格式为：hlen key**

**这里是因为user下有两个属性 username和userage**

**（11）hincrby key field value: 指定增量value**

**（12）hsetnx key field value: 如果该键不存在就创建，如果该键存在就创建失败。**

**List链表**

**Redis 的链表是简单的字符串列表，排序插入顺序。您可以添加元素到 Redis 的列表的头部或尾部**

**Lpush：表示的是向链表的左添加，也就是向链表的头添加；**

**Rpush：表示的是向链表的右添加，也就是向链表的尾添加；**

**常用命令**

**（1）lpush key value： 向链表左侧添加—头插法**

**（2）rpush key value： 向链表右侧添加–尾插法**

**（3）lpop key： 从左边移出一个元素，就是从最左边的那个节点剔除掉。**

**（4）rpop key： 从右边移出一个元素，就是从最右边的那个节点剔除掉。**

**（5）lrange key start end lrange: 命令将返回索引从 start 到 stop 之间的所有元素。Redis 的列表起始索引为 0。**

**如果是要获取全部的元素：lrange key 0 -1**



**（6）llen key： 返回链表中元素的个数 相当于关系型数据库中 select count(\*)**

**（7）lindex key indexnumber :lindex 命令用来返回指定索引的元素，索引从 0 开始，如果是负数表示从右边开始计算的索引，最右边元素的索引是-1。如果要将列表类型当做数组来用，lindex 命令是必不可少的。**

**（8）lset key indexnumber value： 是另一个通过索引操作列表的命令，它会将索引为 index的元素赋值为 value，原来的值会被覆盖。如果该列表不存在就会报错。所以使用这个命令之前先使用exists判断一下。**

**（9）lrem key count value: 移除key链表中count个元素的value值，精确匹配，如果链表中可以有多个重复的值，这里的count指的是可以删除多个相同key的值。**

**（10）ltrim list startIndex endIndex ：通过下标截取指定的长度，这个时候list已经改变了，只剩下截断的元素。**

**（11）rpoplpush source destination ：移除当前的source链表中的最后一个元素，并且将该元素移动到destination链表当中。**

**（12）linsert key BEFORE|AFTER pivot value：在key列表的pivot元素的前/后面插入元素value。**

**实际上list是一个链表，before node after，left ，right都可以插入值；**

**如果key不存在，就创建新的链表；**

**如果key存在就创建新的值；**

**如果移除了所有的值，空链表，也就代表不存在；**

**在两边插入或者改动值，效率最高，中间元素相对来说效率会低一点。**

**应用场景**

**消息队列：利用List的PUSH操作，将任务存在List中，然后工作线程再用POP操作将任务取出进行执行。**

### 此外，还有Set集合，SortedSet( 有序集合)。