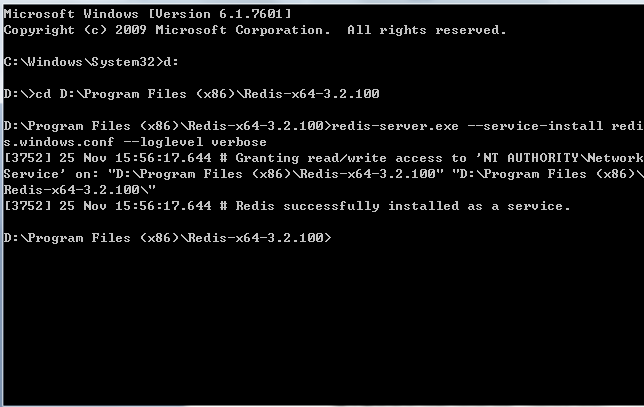
Redis介绍for windows

1.Redis安装<https://github.com/MSOpenTech/redis/releases>，

计算机生成了可选文字: 区亘二习一叫卜启动相关
redis一src一口．..
visualmanager一
兽redis·zip
遨。edi、一win一3.2.100
2017/9/2412:15
2016/7八7:52
.zlp
文件夹
文件夹
文件夹
WinRARZlp压缩…5,102KB
WinRARZlp压缩…2.504KB
oo一5414

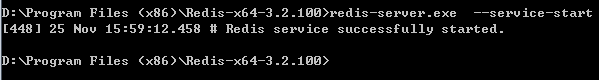
2.cmd命令界面进入相应的安装目录，使用默认服务安装

进入Redis安装包目录，注册服务：redis-server.exe --service-install redis.windows.conf --loglevel verbose

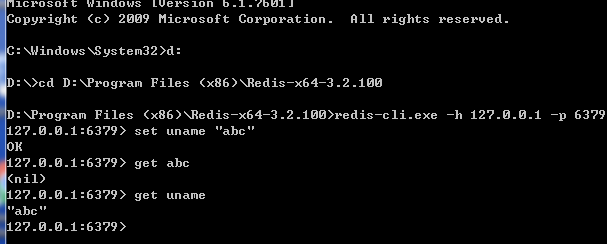


计算机生成了可选文字: 应用程序进程
服务
联网用户
名称
Ras人uto
描述
RemoteA
1犬态
已停止
工作组
net写VCS
Redis
Redis

 启动服务：redis-server.exe  --service-start



     客户端调用: redis-cli.exe -h 127.0.0.1 -p 6379



     停止服务：redis-server.exe  --service-stop

C:\Users\ADMINI~1\AppData\Local\Temp\msohtmlclip1\02\clip_image006.png

     卸载服务: redis-server.exe  --service-uninstall

C:\Users\ADMINI~1\AppData\Local\Temp\msohtmlclip1\02\clip_image007.png

2.Redis主从服务安装。

安装主从服务，其实就是把上面Redis安装文件包，拷贝到相应目录，修改主、从服务器配置文件中IP、Port，同时从服务器要指定主服务器 的IP、Port,按照Redis自定义服务安装中命令进行服务安装、服务启动、服务关闭、服务卸载即可使用。

     我本地主从服务器安装包都还是在D:\Program Files (x86)目录，Redis-x64-3.2.100为主服务器中,用的本地IP:127.0.0.1,生产环境大家可以根据自己实际情况进行设置。

     主服务器redis.windows.conf修改如下：

     主服务器redis.windows.conf修改如下：

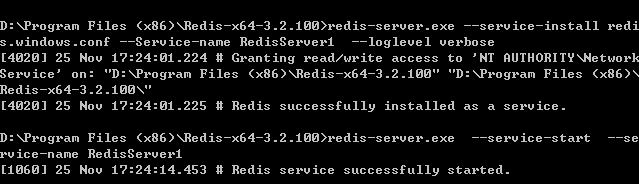
      port  6379

     从服务器redis.windows.conf修改如下：

      port  6380

     slaveof  127.0.0.1  6379

 安装、启动主服务器：



     安装、启动从服务器：



2.关系型数据和非关系型数据库的区别？

1.关系型数据库

关系型数据库，是指采用了关系模型来组织数据的数据库。

简单来说，关系模型指的就是二维表格模型，而一个关系型数据库就是由二维表及其之间的联系所组成的一个数据组织。

关系模型中常用的概念：

* + 关系：可以理解为一张二维表，每个关系都具有一个关系名，就是通常说的表名
  + 元组：可以理解为二维表中的一行，在数据库中经常被称为记录
  + 属性：可以理解为二维表中的一列，在数据库中经常被称为字段
  + 域：属性的取值范围，也就是数据库中某一列的取值限制
  + 关键字：一组可以唯一标识元组的属性，数据库中常称为主键，由一个或多个列组成
  + 关系模式：指对关系的描述。其格式为：关系名(属性1，属性2， ... ... ，属性N)，在数据库中成为表结构

关系型数据库的优点：

* + 容易理解：二维表结构是非常贴近逻辑世界的一个概念，关系模型相对网状、层次等其他模型来说更容易理解
  + 使用方便：通用的SQL语言使得操作关系型数据库非常方便
  + 易于维护：丰富的完整性(实体完整性、参照完整性和用户定义的完整性)大大减低了数据冗余和数据不一致的概率

2. 关系型数据库瓶颈

* + 高并发读写需求

网站的用户并发性非常高，往往达到每秒上万次读写请求，对于传统关系型数据库来说，硬盘I/O是一个很大的瓶颈

* + 海量数据的高效率读写

网站每天产生的数据量是巨大的，对于关系型数据库来说，在一张包含海量数据的表中查询，效率是非常低的

* + 高扩展性和可用性

在基于web的结构当中，数据库是最难进行横向扩展的，当一个应用系统的用户量和访问量与日俱增的时候，数据库却没有办法像web server和app server那样简单的通过添加更多的硬件和服务节点来扩展性能和负载能力。对于很多需要提供24小时不间断服务的网站来说，对数据库系统进行升级和扩展是非常痛苦的事情，往往需要停机维护和数据迁移。

对网站来说，关系型数据库的很多特性不再需要了：

* + 事务一致性

关系型数据库在对事物一致性的维护中有很大的开销，而现在很多web2.0系统对事物的读写一致性都不高

* + 读写实时性

对关系数据库来说，插入一条数据之后立刻查询，是肯定可以读出这条数据的，但是对于很多web应用来说，并不要求这么高的实时性，比如发一条消息之后，过几秒乃至十几秒之后才看到这条动态是完全可以接受的

* + 复杂SQL，特别是多表关联查询

任何大数据量的web系统，都非常忌讳多个大表的关联查询，以及复杂的数据分析类型的复杂SQL报表查询，特别是社交类型（SNS）的网站，从需求以及产品阶级角度，就避免了这种情况的产生。往往更多的只是单表的主键查询，以及单表的简单条件分页查询，SQL的功能极大的弱化了

在关系型数据库中，导致性能欠佳的最主要原因是多表的关联查询，以及复杂的数据分析类型的复杂SQL报表查询。为了保证数据库的ACID特性，我们必须尽量按照其要求的范式进行设计，关系型数据库中的表都是存储一个格式化的数据结构。每个元组字段的组成都是一样，即使不是每个元组都需要所有的字段，但数据库会为每个元组分配所有的字段，这样的结构可以便于标语表之间进行链接等操作，但从另一个角度来说它也是关系型数据库性能瓶颈的一个因素。

1.非关系型数据库

指非关系型的，分布式的，且一般不保证遵循ACID原则的数据存储系统。

以键值对存储，且结构不固定，每一个元组可以有不一样的字段，每个元组可以根据需要增加一些自己的键值对，这样就不会局限于固定的结构，可以减少一些时间和空间的开销。使用这种方式，用户可以根据需要去添加自己需要的字段，这样，为了获取用户的不同信息，不需要像关系型数据库中，要对多表进行关联查询。仅需要根据id取出相应的value就可以完成查询。但非关系型数据库由于很少的约束，他也不能够提供像SQL所提供的where这种对于字段属性值情况的查询。

4. 关系型数据库  V.S.  非关系型数据库

关系型数据库的最大特点就是事务的一致性：传统的关系型数据库读写操作都是事务的，具有ACID的特点，这个特性使得关系型数据库可以用于几乎所有对一致性有要求的系统中，如典型的银行系统。

但是，在网页应用中，尤其是SNS应用中，一致性却不是显得那么重要，用户A看到的内容和用户B看到同一用户C内容更新不一致是可以容忍的，或者说，两个人看到同一好友的数据更新的时间差那么几秒是可以容忍的，因此，关系型数据库的最大特点在这里已经无用武之地，起码不是那么重要了。

相反地，关系型数据库为了维护一致性所付出的巨大代价就是其读写性能比较差，而像微博、facebook这类SNS的应用，对并发读写能力要求极高，关系型数据库已经无法应付(在读方面，传统上为了克服关系型数据库缺陷，提高性能，都是增加一级memcache来静态化网页，而在SNS中，变化太快，memchache已经无能为力了)，因此，必须用新的一种数据结构存储来代替关系数据库。

关系数据库的另一个特点就是其具有固定的表结构，因此，其扩展性极差，而在SNS中，系统的升级，功能的增加，往往意味着数据结构巨大变动，这一点关系型数据库也难以应付，需要新的结构化数据存储。

于是，非关系型数据库应运而生，由于不可能用一种数据结构化存储应付所有的新的需求，因此，非关系型数据库严格上不是一种数据库，应该是一种数据结构化存储方法的集合。

必须强调的是，数据的持久存储，尤其是海量数据的持久存储，还是需要一种关系数据库这员老将。

5. 非关系型数据库分类

由于非关系型数据库本身天然的多样性，以及出现的时间较短，因此，不想关系型数据库，有几种数据库能够一统江山，非关系型数据库非常多，并且大部分都是开源的。

这些数据库中，其实实现大部分都比较简单，除了一些共性外，很大一部分都是针对某些特定的应用需求出现的，因此，对于该类应用，具有极高的性能。依据结构化方法以及应用场合的不同，主要分为以下几类：

* + 面向高性能并发读写的key-value数据库：

key-value数据库的主要特点即使具有极高的并发读写性能，Redis,Tokyo Cabinet,Flare就是这类的代表

* + 面向海量数据访问的面向文档数据库：

这类数据库的特点是，可以在海量的数据中快速的查询数据，典型代表为MongoDB以及CouchDB

* + 面向可扩展性的分布式数据库：

这类数据库想解决的问题就是传统数据库存在可扩展性上的缺陷，这类数据库可以适应数据量的增加以及数据结构的变化

**数据结构与对象**

Redis数据库里面每个键值对(k-v)都是由对象组成。

* + 数据库键总是一个字符串对象(string object)。
  + 值则可以是字符串对象、列表对象(list object)、哈希对象(hash object)、集合对象(set object)、有序集合对象(sorted set object)。

**SET msg "hello world"**  key:msg(string object) value: hello world (string object)

**MSET msg "123" msg1 "234"** 设置多个key

**RPUSH numbers 1 3 5 6 9** key: numbers(string object) value: list object ,继续执行可以向表尾添加值

**LPUSH 和RPUSH一样，不一样是每次添加都是在首部**

**KEYS pattern** 查看键，pattern 是正则

**dbsize**获取键总数

**exists**检查键是否存在 存在返回1 不存在返回0

**del key**删除键  删除成功返回1 删除了不存在的键返回0 支持同时删除多个键 del key1 key2 key3

**EXPIRE key seconds** 为给定 key 设置生存时间，当 key 过期时(生存时间为 0 )，它会被自动删除。例如，expire msg 10,如果在键没有被删除的时候再次执行此命令将重新计时。

**Ttl key** (time to live)查看key的过期时间,以秒为单位，如果没有设置过期时间的key 返回-1 , 查看没有的key返回-2

**PTTl key** 以毫秒为单位查看过期时间

**Persist** 移除key的过期时间，将key变为持久态。移除成功返回1,如果 key 不存在或 key 没有设置生存时间，返回 0 。

**EXPIREAT key timestamp** EXPIREAT 的作用和 EXPIRE 类似，都用于为 key 设置生存时间。不同在于 EXPIREAT 命令接受的时间参数是 UNIX 时间戳(unix timestamp)。

**Select db** 使用数据库，如select 1 使用数据库1

**Move key db** 将key移动到数据库db,例如 move msg 2 将msg移动到数据库2中，成功返回1,失败返回0

* + **OBJECT REFCOUNT <key>** 返回给定 key 引用所储存的值的次数。此命令主要用于除错。
  + **OBJECT ENCODING <key>**返回给定 key 锁储存的值所使用的内部表示(representation)。
  + **OBJECT IDLETIME <key>** 返回给定 key 自储存以来的空闲时间(idle， 没有被读取也没有被写入)，以秒为单位。
  + Randomkey 随机从当前数据库中返回一个key,数据库为空时返回null
  + Rename key newkey 如果不存在key会返回no such key , 当newkey存在，rename会把原来的key覆盖newkey
  + Renamenx key newkey 如果newkey存在，则返回0,不能覆盖已有的key, 当key不存在时返回错误
  + Sort key desc 倒叙排列 Sort key正序排列 ,默认排序对象为数字，当需要对字符串进行排序时，需要显示的在sort命令之后添加alpha修饰符.

计算机生成了可选文字: 。．redis一爪aster
:dbo:：取sgX
TTI‘一l
{Ren,e』｝0Deiete}IReioadvalue)}setTTL)
见dbl(0)
元dbZ(0)
必曲3(0)
必曲4(0)
元dbs(0)
［口曲6(o)
三db7(0)
＿曲s(0)
二dbg(0)
二db10(0)
二db11(0)
二db12(0)
_db13(0)
二db14(0)
三db15(D)
'.redis一slavel
redis一爪aster:
STRD暇：msg
丽习
b尹二二5
VieWas:
川刊n引引引引引叼引l
dbo(0)
dbl(0)
Jk勺，n、

计算机生成了可选文字: 。．redis一兀aster
。函曲。（2/2)
奋
…！
LIST:nUmberS
SIZe:
5：几：一1!Ren.e{{0Deiete}}setTTL{
Va1Ue
一一一
乡dbl(0)
二dbZ(0)
三db3(0)
二db4(0)
二dbs(0)
叻db6(0)
函曲7(o)
函dbs(0)
函曲9(o)
几曲10(o)
函曲11(o)
函曲12(o)
函曲13(o)
几曲l吸（o)
函曲15(o)
redis一SlaVel
函曲。（o)
L民．dbl(0)
习
0De工eterow
洛ReloadValue
.．几〔乙勺」
万earchonpage二
叮．〔己
W
0
崖卜』卜口以旧
一
Value：工z"i七．到s
｝勺1｛。J
[Jt一一一＿一一一J
VieW
JSON
口U
l一一

计算机生成了可选文字: redls一爪aster:0>set爪59"hello"
介OK"
redis一爪aster:0＞路叮SHn切山ers12378
份5"