NoSql

为什么需要NoSQL？

NoSQL-----not only sql

high performance----高并发读写

huge Storage----海量数据的高效率存储和访问

high scalability && high avaliability----高可扩展性和高可用性

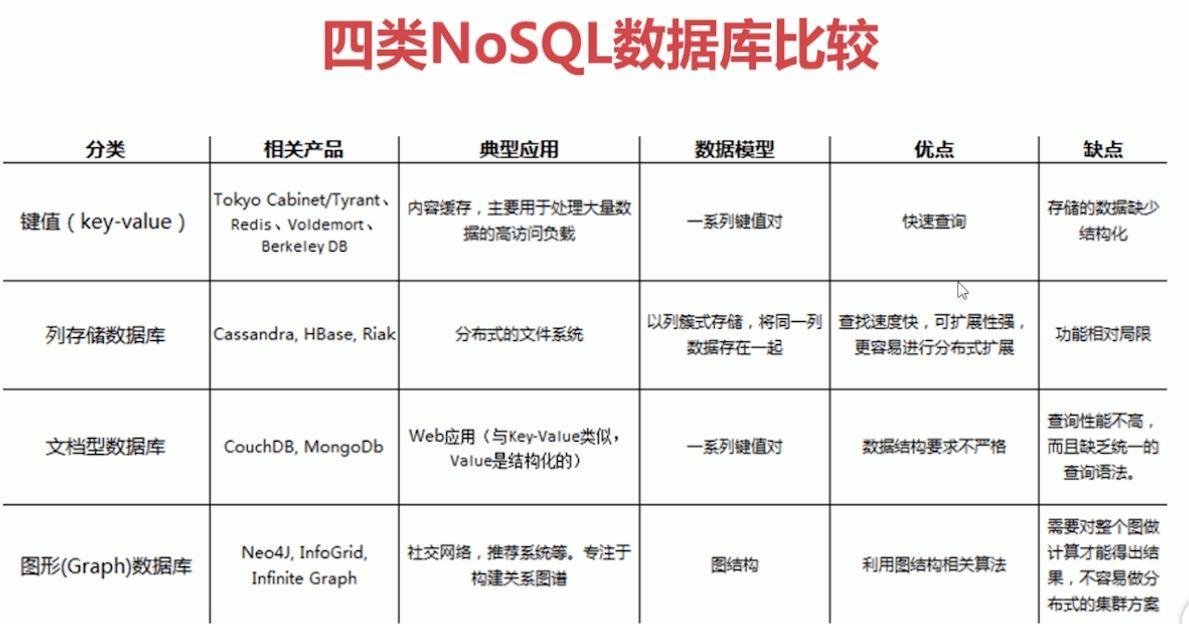
NoSql数据库的四大分类

键值（key-value）存储： redis

列存储: hbase

文档数据库: MongoDB

图形数据库: InfoGrid



NoSql特点：

易扩展：数据之间是没有关系的

灵活的数据模型：无需事先为需要存储的数据建立一些字段

大数据量、高性能：具有非常高的读写性能

高可用：不影响性能的情况下就可以很方便的事先高可用的框架

**Redis**

**高性能键值对数据库。支持的键值数据类型：**

**字符串类型**

**列表类型**

**有序集合类型**

**散列类型**

**集合类型**

**Redis的应用场景：**

**缓存**

**任务队列（秒杀、抢购）**

**网站访问统计**

**数据过期处理**

**应用排行榜**

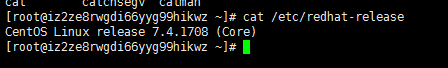
**分布式集群架构中的session分离**

**Redis安装**

**服务器**

****

**Linux版本**

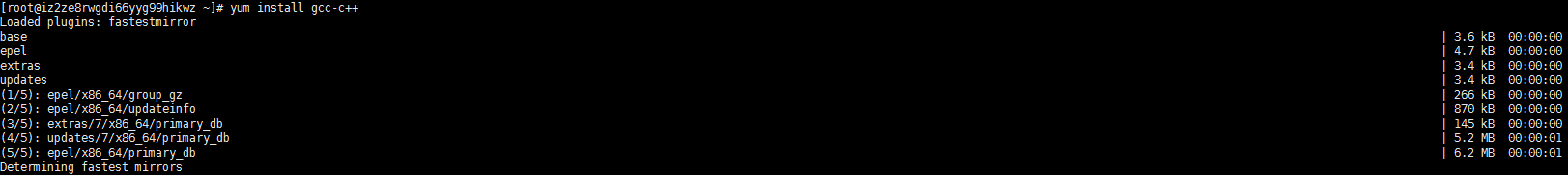
****

**SSH客户端**

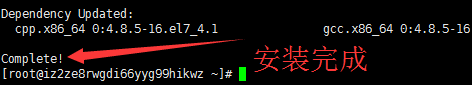
****

**安装环境**

**redis是C语言开发，安装redis需要先将官网下载的源码进行编译，编译依赖gcc环境。首先需要在线安装gcc . yum install gcc-c++**



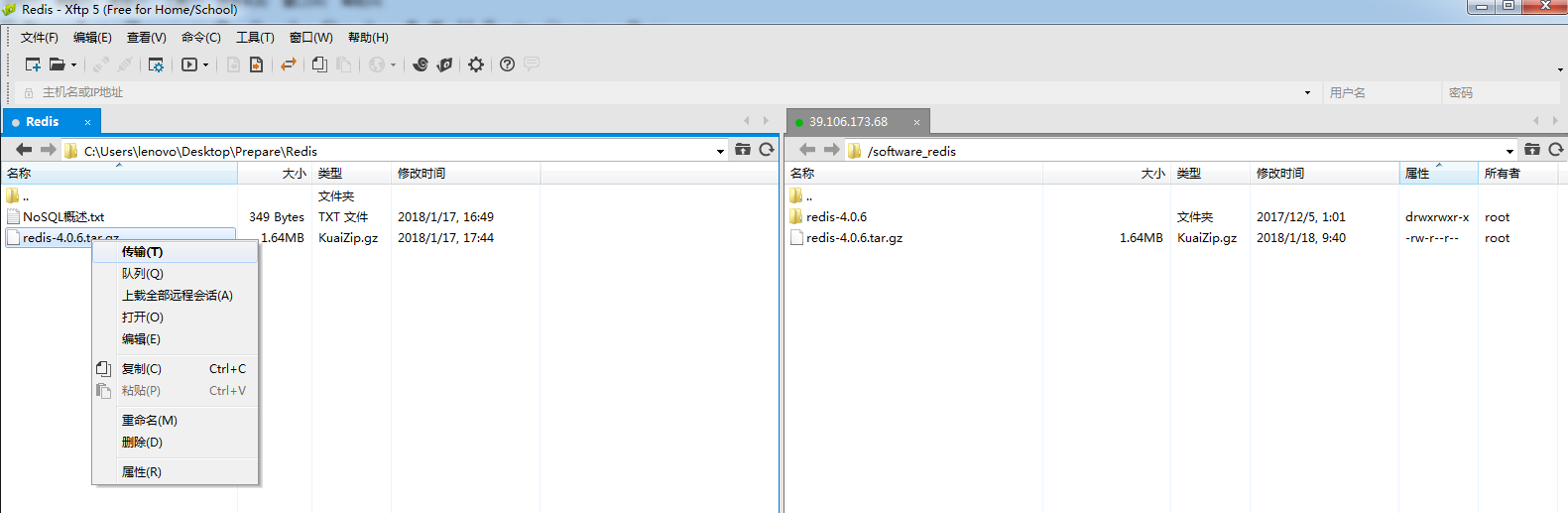
C:\Users\lenovo\Documents\Tencent Files\295873321\Image\C2C\GM2M4@`31JU_JT(PEQ0MB[4.png



**gcc安装完成后开始安装redis**

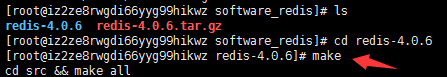
****

**将该压缩包通过xftp上传到linux服务器root目录下并解压**

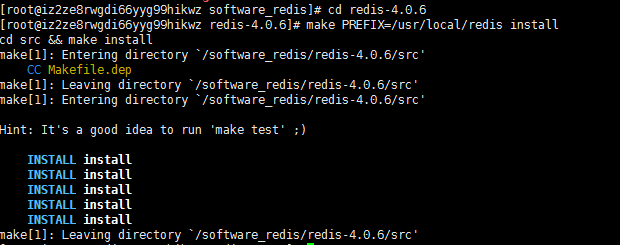
****

****

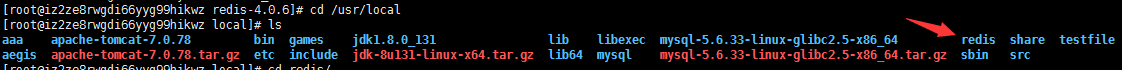
**进入解压后的目录下执行make命令，做基本的编译**

****

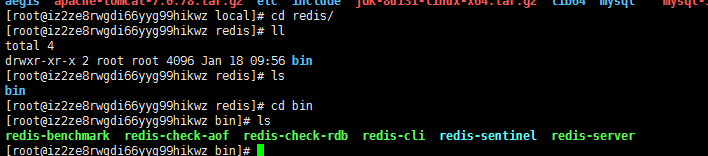
**执行make PREFIX=/usr/local /redis install进行安装**

****

**以上步骤完成后就将redis安装到/usr/local目录下了**

****

**查看redis文件夹下的内容**

****

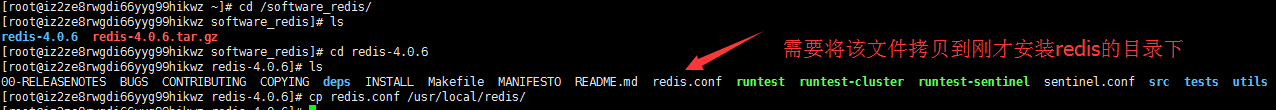
**redis-benchmark：性能测试的一个工具**

**redis-check-aof：aof文件修复的一个工具**

**redis-check-rdb：rdb的一个文件检查工具**

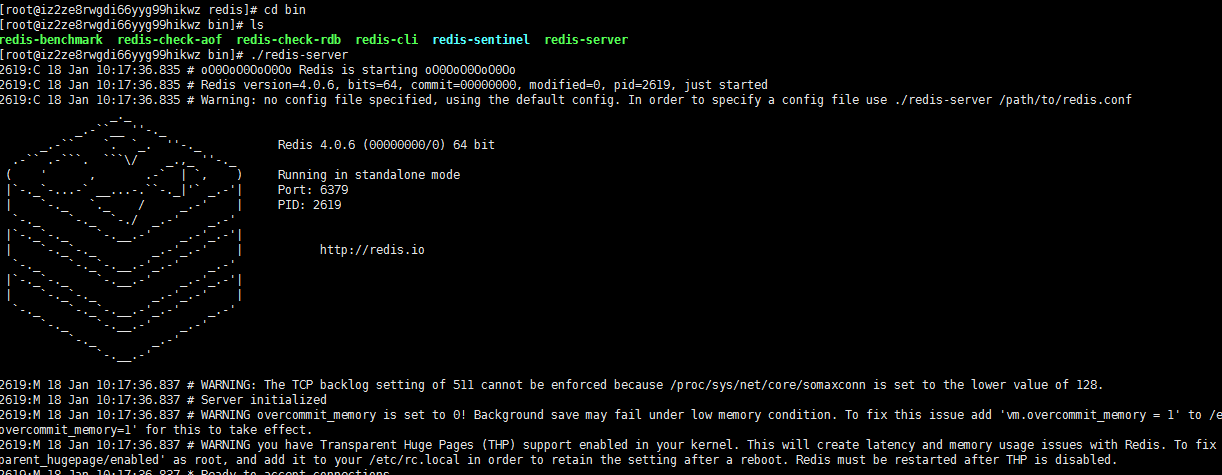
**redis-cli：命令行的一个客户端**

**redis-server：redis服务器启动的命令**

****

****

**此时已经安装好了，接下来我们启动redis**

****

**以上这种启动方式叫作前端启动，但是启动后当前窗口不能再执行其它命令**

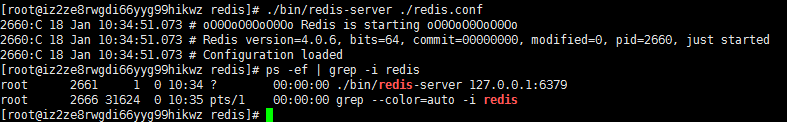
**Ctrl + C 停止前端启动模式**

**redis的启动方式一般是后端启动**

**修改配置文件: vi /usr/local/redis/redis.conf**

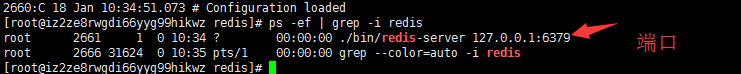
**将其中daemonize no 修改为 daemonize yes**

**然后启动redis**

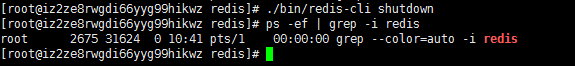
****

**./bin/redis-server ./redis.conf 意思是启动redis时候加载redis.conf配置文件**

**ps -ef|grep -i redis意思是查找redis**

****

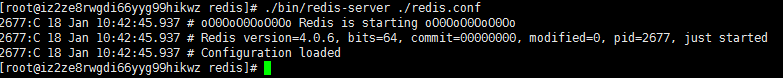
**如何关闭redis**

****

**./bin/redis-cli shutdown意思是停止redis**

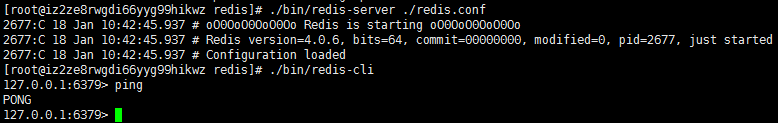
**ps -ef|grep -i redis再一次查看redis，发现已经没有了**

**再次启动redis**

****

**现在可以开始使用redis了**

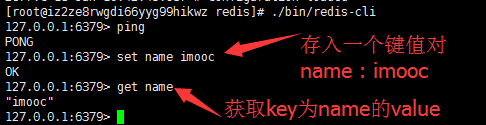
**执行客户端 ./bin/redis-cli**

****

**可以看到当前连接的是主机的6379端口了**

**输入命令ping 回复pong 说明连接是没有问题的**

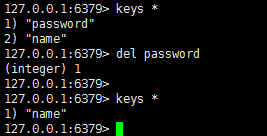
**现在就可以向redis中存入数据了**

****

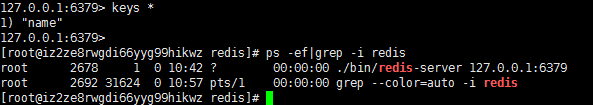
**查看redis中所有key**

****

**如果要删除一个key**

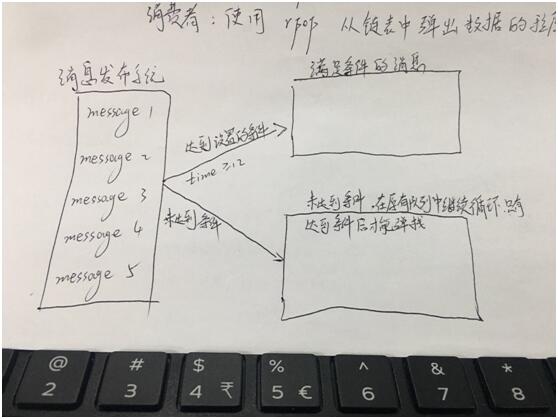
****

**ctrl + c 退出tab**

****

**Redis五大数据类型讲解**  
Redis共有五大数据类型String、hash、list、set、sorted set，它们都遵循key---value的数据结构。  
 **String**  
set 赋值  
get 取值  
getset 取值再赋值  
incr key 该key值增加1，必须整型  
decr key 该key值减小1，必须整型  
incrby key x 该key值对应的值基础上增加x  
decrby key x 该key值对应的值基础上减小x  
append key xxx 在该key所对应的value值拼接字符串xxx  
del key 删除该key  
 **Hash**  
hset myhash username bob 赋值key---myhash一个属性username 值为bob  
hget myhash username  获取key为myhash的属性username的值  
hgetall myhash 获取key为myhash的所有的属性的值  
hmset student username jack age 18 赋值key--student属性username为jack  
  age为18  
hmget student username age 获取key--student的属性username和age的值  
hdel student username 删除key--student的属性username  
del student 删除整个key--student  
hexists student username 查询key--student是否有username属性  
hlen student 查看key--student的长度  
hincrby student age 5  key--student的属性age增加5

**list**  
ArrayList:数组,根据索引查询,速度很快,插入删除涉及到元素位移,比较慢  
LinkedList:双向链表,每个元素都记录了前后元素的指针，插入删除只是改变指针,速度较快。  
redis链表经常会用于消息队列的一些服务,来完成多个程序之间的消息互通。  
lpush mylist 1 2 3 创建一个key--mylist,往里面插入数据  
lrange mylist 0 -1 查看key--mylist里面的数据查询到倒数第一个  
rpoplpush:业务场景  
生产者:使用lpush往链表中插入数据的程序  
消费者:使用rpop从链表中弹出数据的程序

**Set**

无序不可重复的集合

sadd myset a b c创建一个key--myset,往里面插入数值a b csrem myset a b删除key--myset中的a b

smembers nyset查看key--myset中的所有元素

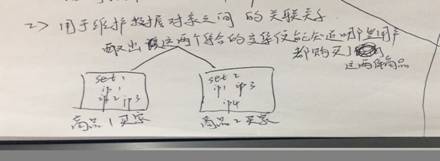
Sismember myset 1查看key-myset中有没有1,如果有返回1没有返回0

sdiff mysa1 mysa2查看key mysa1与mysa2的差集

sinter mya2 myb2查看key mya2与myb2的交集

sunion mya3 myb3查看mya3与myb3的并集

使用场景：1、跟踪一些唯一性数据，比如访问该博客的用户ip2、用于维护数据对象之间的关联关系：取出这两个集合的交集就能够知道哪些用户都购买了这两件商品



**Sorted Set**

**sorted set中的每一个成员都会有一个分数与之关联，Redis通过该分数对sorted set中成员进行排序，但是key值不允许出现重复，但是分数允许重复。**

**所以成员是默认一分数升序排列的。**

**zadd myset 70 zs 80 ls 90 ww 创建key--myset属性zs 分数70属性ls分数80属性ww分数80**

**zadd myset 100 zs 因为存在zs,所以会用分数100覆盖70**

**zscore myset zs 查看zs的分数**

**zcard myset 查看myset中的元素个数**

**zrem myset 删除key--myset**

**zrange myset 0 -1 查看myset中的所有属性**

**zrange myset 0 -1 withscores 查看myset中元素及分数（默认升序）**

**zrevrange myset 0 -1 withscores 查看myset中元素及分数（降序）**

**zremrangebyrank myset 0 4 删除排名前四的**

**zremrangebyscore myset 80 100删除分数在80到100之间的**

**zincrby myset 3 ls 给ls的分数加3**

**zcount myset 80 90 计算分数在80到90之间的个数**

**使用场景：游戏排名、微博的热点话题**

**Redis Key的操作**  
rename key key1 重命名key  
keys \* 查看所有key  
expire key 10s  设置key的过期时间  
type key 查看key的类型  
del key 删除key

**Redis特性**  
1、多数据库  
2、支持事务  
一个redis实例可以包含多个数据库,好比mysql中创建多个数据库一样，最多可以提供16个数据库。0-15  
select 0  查看第一个数据库  
select 1  查看第二个数据库  
move myset 1  将key--myset从第一个数据库移动到第二个数据库（数据库下标从0开始）  
事务：multi---------exec---------discard  
multi:开启事务。后面执行的命令都会存到命令队列中，知道执行exec,相当于关系型数据库中的begin transactin  
exec:提交，相当于commit  
discard:回滚，相当于rollback

**Redis持久化**  
1、RDB  
在指定的时间间隔内，将内存中的数据及快照写入到磁盘中（默认支持）  
2、AOF  
将以日志的形势记录服务器的每一个操作，在Redis服务器启动之初，它会读取文件重新构建该数据库来保证启动后数据库数据是完整的  
3、无持久化  
可以通过配置来禁用Redis服务器的持久化功能，这样redis只用来作缓冲  
4、同时使用RDB和AOF

**RDB**

**优势**：

1、采用RDB持久化，那么Redis数据库只含有一个文件dump.rdb,一小时保存一天的数据，一天保存30天的数据

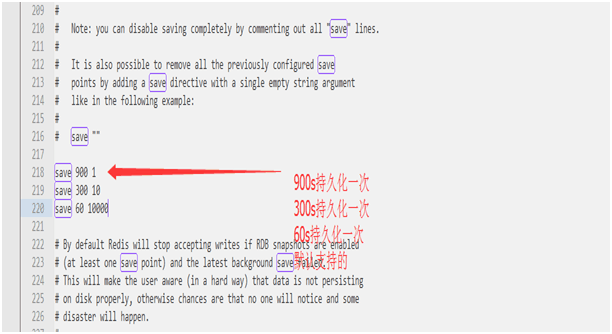
2、可以非常轻松的将每一个单独的文件压缩后再把它转移到其它的存储介质上

3、性能最大化，持久化开始后又分为几个进程，再由子进程完成持久化操作

**劣势**：

比如30s写入硬盘一次，25秒的时候redis服务器挂掉，数据会流失

**配置**：redis.conf文件 同目录下生成dump.rdb文件



**AOF**  
优势：  
3种同步策略。  
1、每秒同步  
2、每修改同步（效率低最安全）  
3、不同步  
AOF包含一个格式清晰易于理解的日志文件用于记录所有的修改操作，可以根据该日志文件进行数据重置  
劣势：运行效率低于RDB日志文件比RDB大  
配置：redis.conf   日志文件  appendonly.aof

