redis

3.2.11version

# redis的优势

## 存储结构

1. 字符类型
2. 散列类型
3. 列表类型
4. 集合类型
5. 有序集合

## 功能

1. 可以为每个key设置超时时间；
2. 可以通过列表类型来实现分布式队列的操作
3. 支持发布订阅的消息模式

## 简单

1. 提供了很多命令与redis进行交互

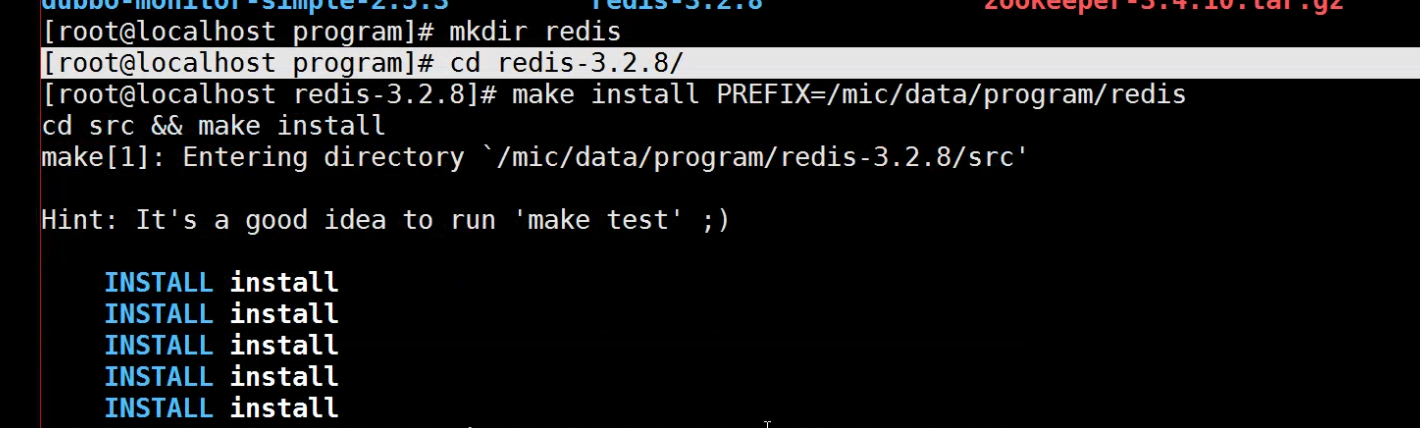
# redis的应用场景

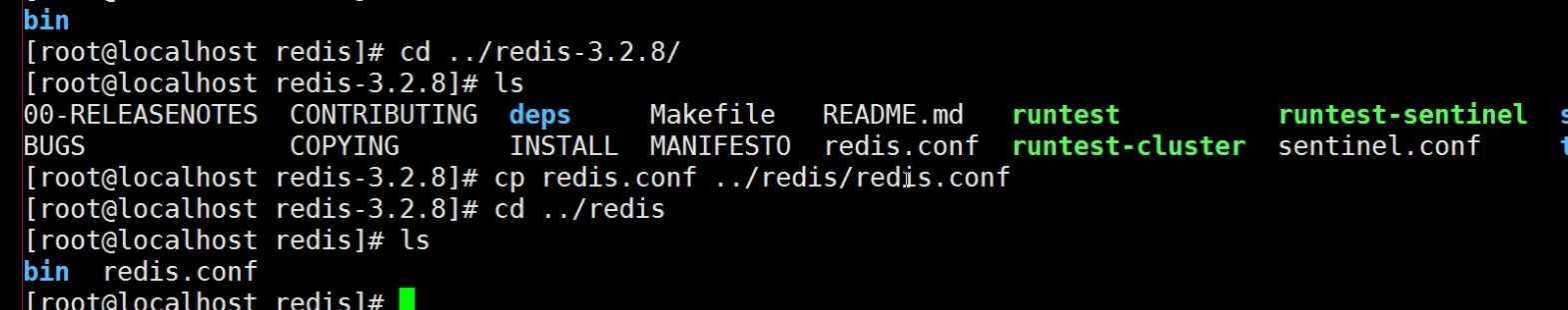
1. 数据缓存（商品数据、新闻、热点数据）
2. 单点登录
3. 秒杀、抢购
4. 网站访问排名…
5. 应用的模块开发

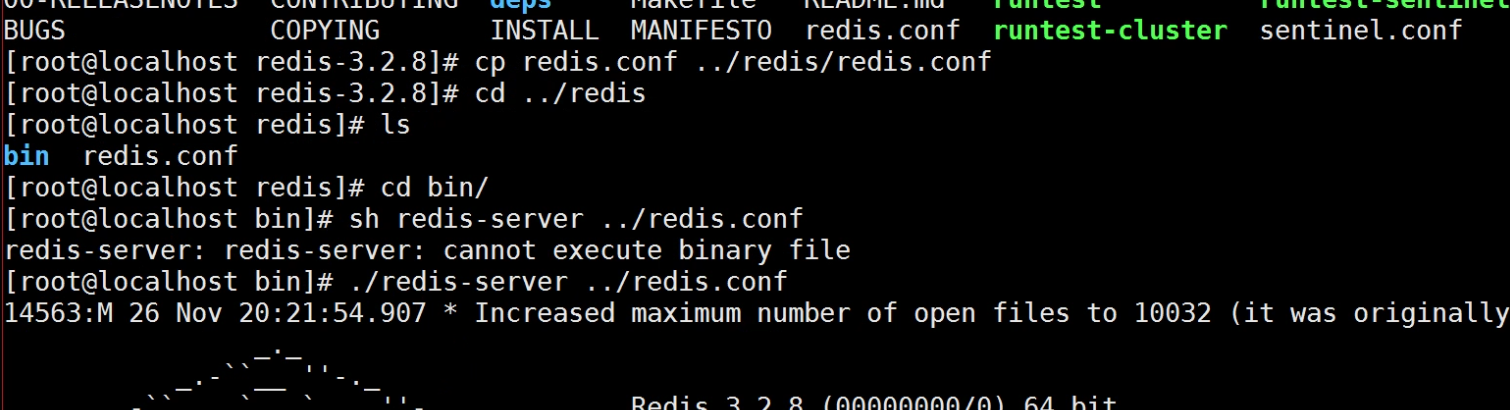
# redis的安装

**redis.io/commands**

1. 下载redis安装包
2. tar -zxvf 安装包
3. 在redis目录下 执行 make
4. 可以通过make test测试编译状态
5. make install [prefix=/path]完成安装







## 启动停止redis

./redis-server ../redis.conf 不是shell脚本 是个二进制文件 所以用./

./redis-cli shutdown

以后台进程的方式启动，修改redis.conf daemonize =yes

连接到redis的命令

./redis-cli -h 127.0.0.1 -p 6379 访问远程

./redis-cli 访问本地

## 其他命令说明

Redis-server 启动服务

Redis-cli 访问到redis的控制台

redis-benchmark 性能测试的工具

redis-check-aof aof文件进行检测的工具

redis-check-dump rdb文件检查工具

redis-sentinel sentinel 服务器配置

# 多数据支持

默认支持16个数据库；可以理解为一个命名空间 redis.conf里面可以设置数据库

跟关系型数据库不一样的点

1. redis不支持自定义数据库名词
2. 每个数据库不能单独设置授权
3. 每个数据库之间并不是完全隔离的。 可以通过flushall命令清空redis实例面的所有数据库中的数据

通过 select dbid 去选择不同的数据库命名空间 。 dbid的取值范围默认是0 -15

# 使用入门

1. 获得一个符合匹配规则的键名列表

keys pattern [? / \* /[]]

keys mic:hobby 如果直接keys 如果数据量大会影响性能

1. 判断一个键是否存在 ， EXISTS key
2. type key 去获得这个key的数据结构类型

# 各种数据结构的使用

## 字符类型

一个字符类型的key默认存储的最大容量是512M

赋值和取值

SET key value

GET key

**递增数字  
incr key**

**错误的演示**

**int value= get key;**

**value =value +1;**

**set key value;**

**key的设计 key value设计不要太长 数据量大占内存**

**对象类型:对象id:对象属性:对象子属性**

**建议对key进行分类，同步在wiki统一管理**

**短信重发机制：sms:limit:mobile 138。。。。。 expire**

**incryby key increment 递增指定的整数**

**decr key 原子递减**

**append key value 向指定的key追加字符串**

**strlen key 获得key对应的value的长度**

**mget key key.. 同时获得多个key的value 建议使用 一次命令 减少网络传输**

**mset key value key value key value …**

setnx

## 列表类型

**list, 可以存储一个有序的字符串列表 底层 双向链表**

LPUSH/RPUSH： 从左边或者右边push数据

LPUSH/RPUSH key value value …

｛17 20 19 18 16｝

llen num 获得列表的长度

lrange key start stop ; 索引可以是负数， -1表示最右边的第一个元素

lrem key count value 删除数据

lset key index value 设置某一个索引的值 有则覆盖

LPOP/RPOP : 取数据

**应用场景：可以用来做分布式消息队列**

## 散列类型

hash key value 不支持数据类型的嵌套

比较适合存储对象

person

age 18

sex 男

name mic

..

hset key field value hmset person age 18

hget key filed

hmset key filed value [filed value …] 一次性设置多个值

hmget key field field … 一次性获得多个值

hgetall key 获得hash的所有信息，包括key和value hgetall person

hexists key field 判断字段是否存在。 存在返回1. 不存在返回0

hincryby

hsetnx

hdel key field [field …] 删除一个或者多个字段

## 集合类型

set 跟list 不一样的点。 集合类型不能存在重复的数据。而且是无序的

sadd key member [member ...] 增加数据； 如果value已经存在，则会忽略存在的值，并且返回成功加入的元素的数量 sadd mic 1 2 3

srem key member 删除元素

smembers key 获得所有数据

sdiff key key … 对多个集合执行差集运算 sdiff mic(1 2 3) ff (2 3 4 5) == 1(前集合减后集合)

sunion 对多个集合执行并集操作, 同时存在在两个集合里的所有值

## 有序集合

zadd key score member score：优先权 eg:zadd teacher 10 mic 8 james 12 tom ; zrange teacher 0 3 withscores

zrange key start stop [withscores] 去获得元素。 withscores是可以获得元素的分数 zrange teacher 0 3 withscores

如果两个元素的score是相同的话，那么根据(0<9<A<Z<a<z) 方式从小到大

网站访问的前10名。

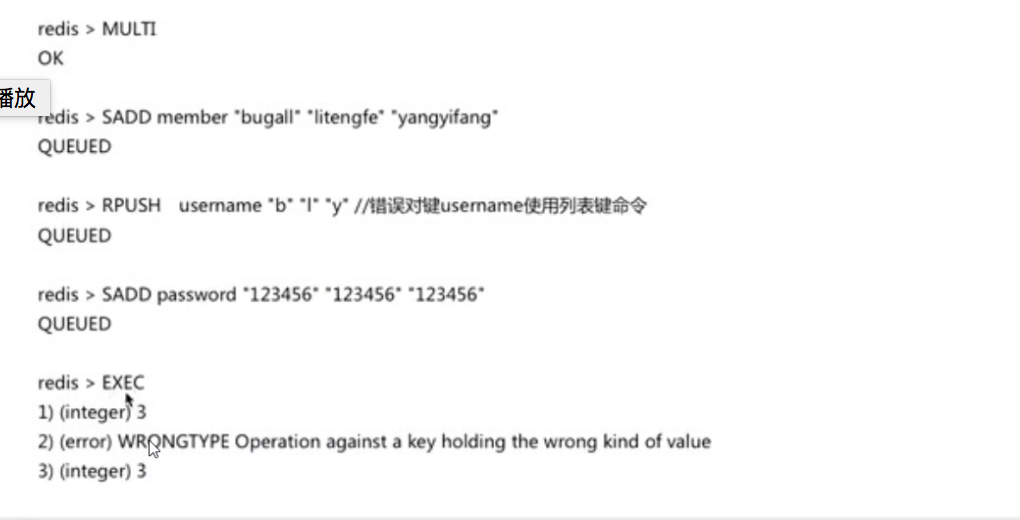
# redis的事务处理

MULTI 去开启事务

QUEUED 命令先放到队列里面 mysql的事务好像是放在log里（我自己这样认为的 后期好好看下mysql）

EXEC 去执行事务 ~~没有类似关系型数据库的rollback 但是exec出错会自动回滚（待定）~~

事务好像是没法回滚的 大部分是能回滚的 就是有特殊情况 运行时错误不会回滚



# 过期时间

expire key seconds

set demo5 value

expire demo5 10

ttl 获得key的过期时间

ttl demo5

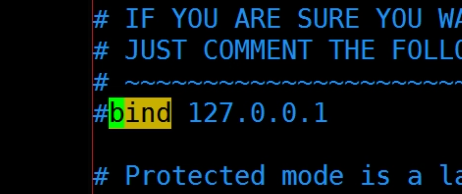
# 发布订阅 阻塞的订阅

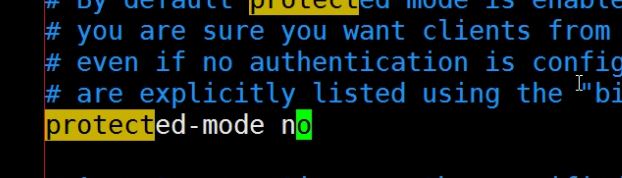
publish channel message

subscribe channel [ …]

codis . twmproxy redis分片 应用层和redis之间加代理

redis会部署在一个授信的内网环境里 然后bind绑定一个ip,如果注释掉的话 意味着可以通话外网来访问





# redis实现分布式锁

数据库可以做 activemq

缓存 -redis setnx

zookeeper