一、NoSql（非关系型数据库）

NoSQL：NoSQL = Not Only SQL 非关系型数据库 NoSQL，泛指非关系型的数据库。随着互联网web2.0网站的兴起，传统的关系数据库在应付web2.0网站，特别是超大规模和高并发的SNS类型的web2.0纯动态网站已经显得力不从心，暴露了很多难以克服的问题，而非关系型的数据库则由于其本身的特点得到了非常迅速的发展。NoSQL数据库的产生就是为了解决大规模数据集合多重数据种类带来的挑战，尤其是大数据应用难题。

1.1 优点

- 高可扩展性
- 分布式计算
- 低成本
- 架构的灵活性，半结构化数据 没有复杂的关系

1.2 缺点

- 没有标准化
- 有限的查询功能（到目前为止）

1.3 分类

类型	代表	特点
列存储	Hbase、Cassandra、Hypertable	顾名思义，是按列存储数据的。最大的特点是方便存储结构化和半结构化数据，方便做数据压缩，对针对某一列或者某几列的查询有非常大的IO优势。
文档存储	MongoDB、CouchDB	文档存储一般用类似json的格式存储，存储的内容是文档型的。这样也就有有机会对某些字段建立索引，实现关系数据库的某些功能。
key-value存储	Berkeley DB、MemcacheDB、Redis	可以通过key快速查询到其value。一般来说，value可以是任何格式。
图存储	Neo4J、FlockDB	图形关系的最佳存储。使用传统关系数据库来解决的话性能低下，而且设计使用不方便。
对象存储	db4o、Versant	通过类似面向对象语言的语法操作数据库，通过对象的方式存取数据。
xml数据库	Berkeley DB XML、BaseX	高效的存储XML数据，并支持XML的内部查询语法，比如XQuery,Xpath。

二、Redis与网站架构

一个普通的问题列表需求

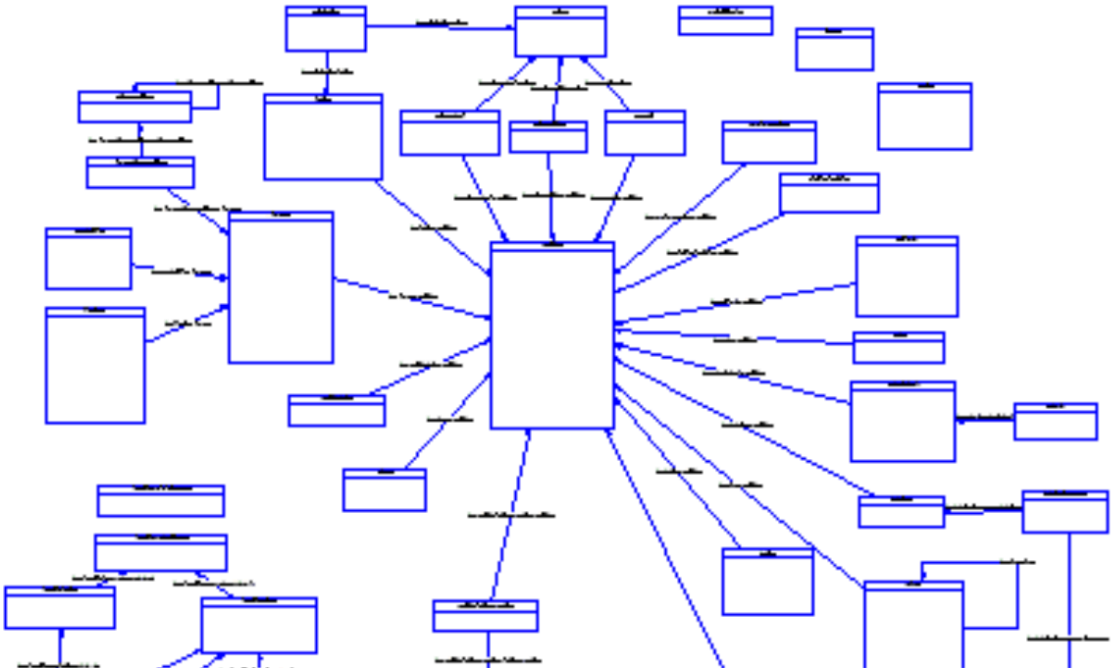
- 问题本身的数据(标题，投票等等)
- 问题的作者数据(另 张单独的 张数据表，通过某个键值关联)
- 问题的标签(本身单独一张数据表，通过一个中间关系表与问题产生 对多的关系)

0 得票	0 回答	8 浏览	以神之名 · 5 分钟前提问 material-ui中是怎么引入Collapse的 material-ui
0 得票	3 回答	77 浏览	i_D_S · 6 分钟前回答 全局安装 vue-cli 最后一步运行 npm run dev 报错 javascript
0 得票	5 回答	155 浏览	HenryYong · 7 分钟前回答 如何使用纯css实现下拉框? css
0 得票	0 回答	15 浏览	caryhgq · 12 分钟前提问 mongodb 数据更新疑问? javascript mongodb
0 得票	0 回答	27 浏览	HenryYong · 18 分钟前提问 请问如何优雅地在axios中访问vue的实例? axios vue-cli vue.js
0 得票	1 回答	24 浏览	chekun · 19 分钟前回答 如何用bash表达 grep 'png' test.txt 有结果，输出ok? bash
0 得票	8 回答	997 浏览	松树没有籽儿 · 19 分钟前回答 vue文件在webstorm上格式报错。实例正常运行没有报错 vue.js

一条sql语句解决问题 too young too simple

```
select www_name,user_pw,c_name,join_time,user_ad,end_time,user_id,user_type,user_tel,user_email_one,birthday fro
user where www_name="111111" and c_name="111111" and birthday="1980-07-01" and user_ad="1111111" and
user_tel="11111111" and user_email_one="111@111.111" and login_time="11" and join_time>="2007-07-29" and
join_time<="" and user_living=2 and user_living=1 and user_id in(select user_id from (select
user_id,login_num/(period_diff(extract(year_month from now()),extract(year_month from join_time))+1) as piju from us
group by user_id) user where piju>=1 and piju<=20) and user_id in(select user_id from (select user_id,year(now())-
year(birthday) as nl from user group by user_id) user where nl>=20 and nl<=40) and user_ad like "%111111%" and
user_id in(select user_id from (select user_id,count(goods_id) as mj from goods where end_time>="2007-05-29" and
end_time<="2008-01" group by user_id) goods where mj>=10 and mj<=20) and user_id in(select user_id from (select
user_id,count(goods_id) as cgmj from goods where end_time>="2007-05-29" and end_time<="2008-01" and g_ok=1
group by user_id) goods where cgmj>=1 and cgmj<=20) and user_id in(select user_id from (select
user_id,count(goods_id) as sbmj from goods where end_time>="2007-05-29" and end_time<="2008-01" and g_ok=2
group by user_id) goods where sbmj>=1 and sbmj<=20) and user_id in(select user_id from (select distinct user_id,(sele
sum(goods_total) as zs from goods where user_id=g.user_id and end_time>="2007-05-29" and end_time<="2008-01"
group by user_id) as zs,(select sum(goods_total) as cg from goods where user_id=g.user_id and end_time>="2007-05-
29" and end_time<="2008-01" and g_ok=1 group by user_id)as cg from goods as g) as g where cg/zs>=0.001 and
cg/zs<=0.9) and user_id in(select user_id from (select distinct user_id,(select sum(goods_total) as zs from goods where
user_id=g.user_id and end_time>="2007-05-29" and end_time<="2008-01" group by user_id) as zs,(select
sum(goods_total) as sb from goods where user_id=g.user_id and end_time>="2007-05-29" and end_time<="2008-01"
and g_ok=2 group by user_id)as sb from goods as g) as g where sb/zs>=0.01 and sb/zs<=0.9) and user_id in(select
```

多次查询让你怀疑人生



冗余字段过多会让你看起来很傻

为啥不试试Redis

2.2 与sql比较

大大减少了查询数量，提高了效率
redis的API更加人性化，再也不需要构建SQL语句，节省了SQL的解析时间

三、Redis

3.1简介

3.1什么是Redis?

Redis是一种基于键值对的NoSQL数据库，它提供了对多种数据类型（字符串、哈希、列表、集合、有序集合、位图等）的支持，能够满足很多应用场景的需求。Redis将数据放在内存中，因此读写性能是非常惊人的。与此同时，Redis也提供了持久化机制，能够将内存中的数据保存到硬盘上，在发生意外状况时数据也不会丢掉。此外，Redis还支持键过期、地理信息运算、发布订阅、事务、管道、Lua脚本扩展等功能，总而言之，Redis的功能和性能都非常强大，如果项目中要实现高速缓存和消息队列这样的服务，直接交给Redis就可以了。目前，国内外很多著名的企业和商业项目都使用了Redis，包括：Twitter、Github、StackOverflow、新浪微博、百度、优酷土豆、美团、小米、唯品会等。

Redis是REmote DIctionary Server的缩写，它是一个用ANSI C编写的高性能的key-value存储系统；Redis 是完全开源免费的，遵守BSD协议，是一个高性能的key-value数据库，与其他的key-value存储系统相比，Redis有以下一些特点（也是优点）：

- Redis的读写性能极高（ Redis能读的速度是110000次/s,写的速度是81000次/s ），并且有丰富的特性（发布/订阅、事务、通知等）。
- Redis支持数据的持久化（RDB和AOF两种方式），可以将内存中的数据保存在磁盘中，重启的时候可以再次加载进行使用。
- Redis支持多种数据类型，包括：string、hash、list、set，zset、bitmap、hyperloglog等。
- Redis支持主从复制（实现读写分析）以及哨兵模式（监控master是否宕机并自动调整配置）。
- Redis支持分布式集群，可以很容易的通过水平扩展来提升系统的整体性能。
- Redis基于TCP提供的可靠传输服务进行通信，很多编程语言都提供了Redis客户端支持。

3.2 Redis的应用场景

- 高速缓存 - 将不常变化但又经常被访问的热点数据放到Redis数据库中，可以大大降低关系型数据库的压力，从而提升系统的响应性能。

- 排行榜 - 很多网站都有排行榜功能，利用Redis中的列表和有序集合可以非常方便的构造各种排行榜系统。
- 商品秒杀/投票点赞 - Redis提供了对计数操作的支持，网站上常见的秒杀、点赞等功能都可以利用Redis的计数器通过+1或-1的操作来实现，从而避免了使用关系型数据的`update`操作。
- 分布式锁 - 利用Redis可以跨多台服务器实现分布式锁（类似于线程锁，但是能够被多台机器上的多个线程或进程共享）的功能，用于实现一个阻塞式操作。
- 消息队列 - 消息队列和高速缓存一样，是一个大型网站不可缺少的基础服务，可以实现业务解耦和非实时业务削峰等特性，这些我们都会在后面的项目中为大家展示。

3.3 持久化存储

在运行情况下， Redis 以数据结构的形式将数据维持在内存中， 为了让这些数据在 Redis 重启之后仍然可用， Redis 分别提供了 RDB 和 AOF 两种持久化模式。

- RDB [RDB 将数据库的快照（snapshot）以二进制的方式保存到磁盘中。]

  在 Redis 运行时， RDB 程序将当前内存中的数据库快照保存到磁盘文件中，在 Redis 重启动时， RDB 程序可以通过载入 RDB 文件来还原数据库的状态。

  RDB 功能最核心的是 rdbSave 和 rdbLoad 两个函数， 前者用于生成 RDB 文件到磁盘， 而后者则用于将 RDB 文件中的数据重新载入到内存中：

  RDB 本质上是个文件 每隔一段时间 在redis配置文件中进行设置 将内存中的数据存入文件中 如果数据过大 也容易造成数据丢失

- AOF [则以协议文本的方式，将所有对数据库进行过写入的命令（及其参数）记录到 AOF 文件，以此达到记录数据库状态的目的。]

  AOF 将命令追加到文件中 将原有的内容替换掉 记录到 AOF 文件， 以此达到记录数据库状态的目的， 为了方便起见， 我们称呼这种记录过程为同步。

3.5 安装redis

```
sudo apt install redis-server
安装完成后，Redis服务器会自动启动，我们检查Redis服务器程序
ps -aux|grep redis
或者
netstat -nlt | grep 6379

#看见 port 6379  就成功启动了redis服务
```

文件名称	作用
redis-server	redis 服务端
redis-cli	redis 客户端
redis-benchmark	redis性能测试工具
redis-check-aof	aof修复工具
redis-check-rdb	rdb
redis-sentinel	哨兵服务器 2.8版本之后才有

【redis-sentinel】监控你管理的作用来提高集群的高可用性

```
redis-cli客户端使用方式:
redis-cli
-p  #端口
-h  #主机
链接上
redis-cli -p 6379
127.0.0.1:6379> ping
PONG
127.0.0.1:6379>   #ping之后 pong来了就是成功了
离开客户端请输入quit

服务管理
启动/停止/重启redis有三种方式
1) systemctl start/stop/restart  redis-server.service
2) service  redis-server start|restart|stop
3) cd /etc/init.d
  ./redis-server  start/stop/restart
4) redis-server /root/redis-5.0.4/redis.conf  #通过指定的配置文件来修改Redis的配置。
```

【redis的配置文件】

配置文件在/etc/redis/redis.conf

```
sudo vim /etc/redis/redis.conf
# 常用配置项
requirepass 你的密码    # 服务器远程连接密码
bind 127.0.0.1      # 绑定ip
port 5379  # 指定端口
daemonize yes # 是否以守护进程执行，如果以守护进程执行，不会在命令行下阻塞
dbfilename dump.rdb    #数据文件
dir /var/lib/redis     #数据文件存储路径


#如果指定了密码，启用客户端时需要加上-a 密码
redis-cli -a 密码
```

四、redis数据类型

Redis支持五种数据类型：string（字符串），hash（哈希），list（列表），set（集合）及zset(sorted set：有序集合)。

- redis常用命令请参考：http://redis.cn/commands.html

4.1 string

是最简单的类型，你可以理解为一个key 对应一个value。string 类型是二进制安全的。意思是redis 的string 可以包含任何数据，比如jpg 图片或者序列化的对象。string类型是Redis最基本的数据类型，一个键最大能存储512MB。常用于：缓存页面、session共享、计数

- 设置键

```
命令：SET key value  #设置单键值对
>set h1 100  #设置h1的值为100

命令：mset key  value [key  value] #设置多个键值对
>mset name '王宝强' age 30 gender '男'

命令：setex   key seconds  value #设置键值及过期时间（秒单位）
>setex  age 100 20 #设置年龄的值为20，过期时间100秒
```

- 获取键

```
命令：get  key  #获取单个键
>get h1

命令：mget key1 key2  key3  #获取多个键
>mget name age sex
```

- 查看过期时间

```
命令：ttl key
>ttl a1 #查看a1的过期时间
```

- 运算

```
原来的值必须是数值字符串
命令：incr key  #将对应的key 加1
命令：decr key  #将对应的key值减1
命令：incrby  key  num   #将对应的key加指定值
命令：decrby  key  num   #将对应的key的值减去指定值
```

- 其它操作

```
命令：append key  value  #追加值，redis中值都是字符串，追加就是字符拼接
>append name 'hello'  #如果原来的值是tom,那么现在就是tomhello

命令：strlen  key  #获取值得长度
```

4.2 hash

Redis hash 是一个键值(key=>value)对集合。Redis hash是一个string类型的field和value的映射表，hash特别适合用于存储对象。每个 hash 可以存储 2的32次方-1 键值对（40多亿）。存储形式：

key = {name:'tom',age: 18}

- 设置值

```
命令：hset  key  field   value  #设置key所指对象的指定属性的值
命令：hmset key  field   value  [field value] #设置key所指对象的多个属性值
命令：hsetnx  key  field  value  #当field字段不存在时 设置key所指对象的field属性值

hset person name '二狗子'
hmset person age 20 sex '男'
hsetnx person maried '未婚'
```

- 获取值

```
命令：  hget key field  #获取key指定的对象的属性值
命令：  hmget  key  field [field]  #获取key指定对象的多个属性值
命令：  hgetall key   #获取key所指对象的所有属性的名称和值
命令：  hkeys   key   #获取key所指对象的所有属性名
命令：  hvals   key   #获取key所指对象是的所有属性值
命令：  hlen key      #获取key所指对象的属性个数
```

- 其它操作

```
命令：hincrby  key   field   increment  #为key所指对象的指定字段的整数值加上increment
命令：hincrbyfloat  key  field  increment #为key所指对象的指定字段的实数值加上increment
命令：hexists key  field  #判断当前的字段是否存在在（在返回1 否则返回0）
命令：hdel  key  field  [field] #删除字段和值
```

4.3 list

redis 列表是简单的字符串列表，按照插入顺序排序。你可以添加一个元素到列表的头部（左边）或者尾部（右边）。列表最多可存储 2的32次方 - 1 元素 (4294967295, 每个列表可存储40多亿)。

常应用于：1、对数据量大的集合数据删减 2、任务队列

- 添加数据

```
命令：lpush  key   value [value]    #头部插入数据
命令：lpushx  key  value            #如果列表存在则在列表头部插入数据
命令：rpush  key  value  [value]    #在列表尾部添加数据
命令：rpushx  key  value            #如果列表存在，则在尾部添加数据
命令：linsert key  before|after  value  value  #在指定值前或后插入数据
命令：lset  key  index  value       #设定指定索引元素的值
注意：索引的值从左边开始，向右增加，左边第一个是0，从右边向左索引编号为：-1 -2...
```

- 获取数据

```
命令：lpop  key                 #左侧出队并返回出队元素
命令：rpop  key                 #右侧出队并返回出队元素
命令：lindex   key    index     #返回指定索引的值
命令：lrange  key  start  end   #返回存储列表中的指定范围的元素
                                 [start,end]
命令：lrem key count value      #从列表里移除前 count 次出现的值为 value 的元素
         count > 0: 从头往尾移除值为 value 的元素。
         count < 0: 从尾往头移除值为 value 的元素。
         count = 0: 移除所有值为 value 的元素。
```

- 其它操作

```
命令：llen  key  #获取列表长度
命令：ltrim  key  start  stop  #裁剪列表 保留start到stop之间的元素，其它都删除
 ltrime  mylist  -3  -1  #从索引为-3到-2的保留， 以外的全部删除
```

4.4 set 无序的集合

Redis的Set是string类型的无序集合，元素具有唯一性 不重复。集合是通过哈希表实现的，所以添加，删除，查找的复杂度都是O(1)。

常应用于：对两个集合间的数据进行交集、并集、差集运算

- 添加元素

```
sadd  key  member [member]  #添加多个元素
```

- 获取元素

```
smembers   key   #获取集合中所有的元素
scard    key     #返回集合元素的个数
srandmember  key   [count]  #返回集合中随机元素的值，可以返回count个
```

- 其它操作

```
spop   key [count]   #移除集合中随机的count个元素,并返回
srem  key  member1  [member2]  #移除集合中 一个或者 多个 成员
sismember  key  member   #判断元素是否在集合中  存在返回1  不在返回0
```

- 集合操作

```
求多个集合的交集:   sinter  key  [key...]
求多个集合的差集 (注意比较顺序): sdiff   key  [key...]
求 多个集合的并集:    sunion  key [key....]
```

4.5 zset 有序从大到小排序

Redis zset 和 set 一样也是string类型元素的集合,且不允许重复的成员。不同的是每个元素都会关联一个double类型的分数。redis正是通过分数来为集合中的成员进行从小到大的排序。zset的成员是唯一的,但分数(score)却可以重复。

常应用于：排行榜

- 添加元素

```
zadd  key score  member [score member]    #添加多个元素
zincrby  key  increment   member          #对指定的成员增加权重
increment
```

- 获取元素

```
zrange  key  start  end    #返回指定范围的元素
zcard  key                 #返回元素的个数
zcount  key min max        #返回有序集合中权重在min和max之间的元素的
个数
zscore  key  member        #返回有序集合中 member(元素)  的权重
(score)
zrange key  start  end  withscores  #返回当前key中 所有的权重
(score)和元素(member)
```

4.6 数据库切换

```
redis默认带有16个数据库，编号从0-15。进入redis后默认数据库是0，可以使用
select num进行切换
客户端不显示中文的处理：打开客户端的时候添加参数:--raw
redis-cli --raw
```

4.7 其他

```
keys *  #查看所有的key
keys u* #查以u开始的key
keys  n???    查找以n为开头长度为4个的key
keys  n      查找 包含 n 的所有的key

支持的正则表达式:
- h?llo   匹配第二位为任意的字符
- h*llo     匹配第二位为任意字符 0个 或多个
- h[ab]llo  匹配第二位为 a或者b的字符的key
- hello  匹配第二位除了e字符以外的任意的key
- h[a-z]llo 匹配第二位为a-z的小写字母的key

exists key #判断键是否存在
type  key   #查看key对应的value的类型
del   key   #删除指定key
expire key 10  #设置过期时间，秒
persist  key   #移除key的过期时间
rename key newkey #修改key的名称(如果新的key的名字存在 则会把存在的key的
值 覆盖掉)
randomkey  #随机返回一个 key
move key  db  将键移动到指定库
```

```
flushdb  #清空当前库所有key
flushall #清空所有库里的key

exit #退出redis客户端
quit 退出客户端

查看服务器信息
info

dbsize 当前库中有多少key
```

五、redis备份和还原

redis支持持久化的方式有两种：RDB和AOF

- RDB备份
  - 查看备份目录

```
#1查看配置文件
cd /etc/redis
vim redis.conf

# 当后台进程执行save出错时，停止redis的写入操作。
stop-writes-on-bgsave-error yes
rdbcompression yes  # 将rdb文件进行压缩
rdbchecksum yes   # 对rdb文件进行校验
dbfilename dump.rdb  # rdb文件命名
dir /var/lib/redis  # rdb文件备份存储目录

#使用命令查看
进入redis客户端
127.0.0.1:6379> config get dir
1) "dir"
2) "/var/lib/redis"  #备份目录
```

  - 备份
    - 命令行下执行：redis-cli save 阻塞主进程
    - 命令行下执行：redis-cli bgsave 不阻塞主进程

        - 查看备份时间，命令行下执行 time redis-cli save
    - 还原
        - 只要将备份的dump.rdb文件覆盖原来的文件就可以还原

六、主从复制

当数据量变得庞大的时候，读写分离还是很有必要的。同时避免一个redis服务宕机，导致应用宕机的情况，我们启用sentinel(哨兵)服务，实现主从切换的功能。redis提供了一个master,多个slave的服务。

角色	ip
master（主）	10.20.100.186
slave（从）	10.20.100.106

- master主配置

```
sudo vim /etc/redis/redis.conf
#找到# bind 127.0.0.1 修改为
bind 127.0.0.1 10.20.100.186

#找到# requirepass 将其修改为 这是修改的是服务器远程连接密码
requirepass 123
```

- slave从配置

```
bind 127.0.0.1 10.20.100.106  #10.20.100.106从服务器网络地址
daemonize yes    #是否是守护进程
slaveof 10.20.100.186 6379   #主服务器的ip和端口
masterauth 123    # 主服务器的密码
```

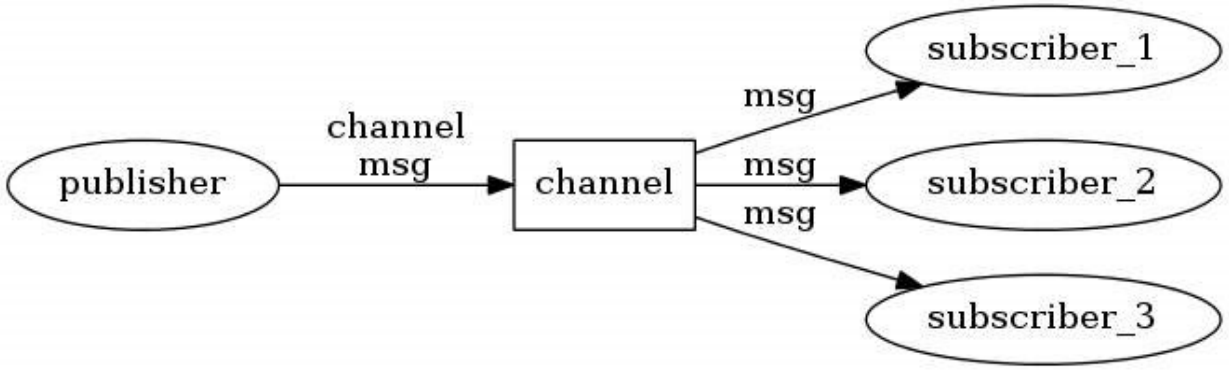
重启master和slave的服务，然后登录从服务器，执行：

```
python@ubuntu:/etc/redis$ redis-cli -p 6380 -a 123
127.0.0.1:6380> info replication
# Replication
role:slave
master_host:10.20.100.186
master_port:6379
```

```
master_link_status:up
master_last_io_seconds_ago:0
master_sync_in_progress:0
slave_repl_offset:1444
slave_priority:100
slave_read_only:1
connected_slaves:0
master_repl_offset:0
repl_backlog_active:0
repl_backlog_size:1048576
repl_backlog_first_byte_offset:0
repl_backlog_histlen:0
```

七. 发布和订阅

Redis提供了发布订阅功能，可以用于消息的传输，Redis的发布订阅机制包括三个部分，发布者，订阅者和Channel。



发布者和订阅者都是Redis客户端，Channel则为Redis服务器端，发布者将消息发送到某个的频道，订阅了这个频道的订阅者就能接收到这条消息。Redis的这种发布订阅机制与基于主题的发布订阅类似，Channel相当于主题。

- 订阅

  订阅者对一个或多个频道感兴趣，只需要接收感兴趣的的消息，不必知道谁发布的

  ```
  SUBSCRIBE 频道名称,频道名称..
  SUBSCRIBE  channel*
  ```

- 发布

```
publish 频道 消息
```

- 取消订阅

```
UNSUBSCRIBE 频道
UNSUBSCRIBE     #不带参数，取消所有订阅
```