第一轮：了解应用场景，从菜鸟教程（或其他文档）大概过一遍。

第二轮：实现demo。

第三路：实战开发应用。

第一轮

1. Redis的概念
   1. Redis是一个开源免费的使用ANSI C语言编写、遵守BSD协议、支持网络、可基于内存亦可持久化的日志型、Key-Value数据库，并提供多种语言的API。
   2. 又称数据结构服务器，因为value值可以是String（字符串），Hash（哈希），list（列表）,Set（无序集合），Sorted Set（有序不重复集合）。
2. Redis的优势
   1. **性能极高** – Redis能读的速度是110000次/s,写的速度是81000次/s 。
   2. **丰富的数据类型** – Redis支持二进制案例的 Strings, Lists, Hashes, Sets 及 Ordered Sets 数据类型操作。
   3. **原子** – Redis的所有操作都是原子性的，意思就是要么成功执行要么失败完全不执行。单个操作是原子性的。多个操作也支持事务，~~即原子性~~，通过MULTI和EXEC指令包起来。
   4. **丰富的特性** – Redis还支持 publish/subscribe, 通知, key 过期等等特性。
3. Redis安装
   1. Window平台安装：
      1. github下载对应的包，解压到C盘。
      2. 启动服务端：在cmd窗口进入解压目录（或配置C盘的路径到环境变量），输入redis-server.exe redis.windows.conf。
      3. 启动客户端：保持原来的cmd窗口，另起一个cmd窗口，进入解压目录，输入redis-cli.exe -h 127.0.0.1 -p 6379。连接上服务端后，在客户端窗口开始使用redis命令，设置键值对set myKey abc，取出键值对 get mykey。

（同mysql，CS架构）

1. Redis配置
   1. 位于Redis安装目录下的redis.conf
   2. 配置项说明：略
2. Redis数据类型
   1. String（字符串）
      1. string 类型是二进制安全的（即数据11110000加密成10001000），可以包含任何数据。比如jpg图片或者序列化的对象。最大能存储 512MB。
      2. 示例：

redis 127.0.0.1:6379> SET runoob "菜鸟教程"

OK

redis 127.0.0.1:6379> GET runoob

"菜鸟教程"

插入：

序列化对象与JSON的比较：

序列化(serialize)指将一个Java对象写入IO流中,对象的反序列化机制(Deserialize)则指从IO流中恢复该Java对象. (实现Serialize接口或者Externalizable接口)

JSON是结构化数据串行化的文本格式，作为XML的一种替代品，用于表示客户端与服务器间数据交换有效负载的格式。

serialize在编码后大概是json的两倍，数据量较大时serialize比json差一个数量级。但json无法处理对象方法等数据。

所以，对象存储传输用serialize，与对象无关的数据存储可以使用json。

* 1. Hash（哈希）
     1. 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 232 -1 键值对（40多亿）。
     2. 示例：（好用，类似js对象）

redis 127.0.0.1:6379> HMSET runoob field1 "Hello" field2 "World"

"OK"

redis 127.0.0.1:6379> HGET runoob field1

"Hello"

redis 127.0.0.1:6379> HGET runoob field2

"World"

* 1. List（列表）
     1. 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）。
     2. 示例

redis 127.0.0.1:6379> lpush runoob redis

(integer) 1

redis 127.0.0.1:6379> lpush runoob mongodb

(integer) 2

redis 127.0.0.1:6379> lpush runoob rabitmq

(integer) 3

redis 127.0.0.1:6379> lrange runoob 0 10

1) "rabitmq"

2) "mongodb"

3) "redis"

* 1. Set（集合
     1. Set 是 string 类型的无序集合。通过哈希表实现的，所以添加，删除，查找的复杂度都是 O(1)。
     2. 示例

redis 127.0.0.1:6379> sadd runoob redis

(integer) 1

redis 127.0.0.1:6379> sadd runoob mongodb

(integer) 1

redis 127.0.0.1:6379> sadd runoob rabitmq

(integer) 1

redis 127.0.0.1:6379> sadd runoob rabitmq

(integer) 0

redis 127.0.0.1:6379> smembers runoob

1) "redis"

2) "rabitmq"

3) "mongodb"

* 1. Sort set（zset:有序集合）
     1. 也是string类型元素的集合,且不允许重复的成员。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。
     2. 示例

redis 127.0.0.1:6379> zadd runoob 0 redis

(integer) 1

redis 127.0.0.1:6379> zadd runoob 0 mongodb

(integer) 1

redis 127.0.0.1:6379> zadd runoob 0 rabitmq

(integer) 1

redis 127.0.0.1:6379> zadd runoob 0 rabitmq

(integer) 0

redis 127.0.0.1:6379> > ZRANGEBYSCORE runoob 0 1000

1) "mongodb"

2) "rabitmq"

3) "redis"

1. 各数据类型应用产场景



1. Redis数据类型命令（类似SQL）
   1. 与键相关命令
      1. DEL key //key存在时删除
      2. DUMP key //系列化给定key，并返回序列化值
      3. EXIXTX key //检查给定key是否存在
      4. EXPIRE key seconds //给key设置过期时间
      5. KEYS pattern //查找符合模式的key
      6. MOVE key db //将key移动给定的数据库db中
      7. RANDOMKEY //从当前数据库随机返回一个key
      8. TYPE key //返回key所存储的值的类型
   2. 与字符串相关命令
      1. SET key value //设置key的值
      2. GET kry //获取key的值
      3. GETRANGE key start end //返回key中字符串值的子字符
      4. GETBIT key offset //对key所存储的字符串值，获取指定偏移量的位
      5. STRLEN key /返回key所存储字符串值的长度

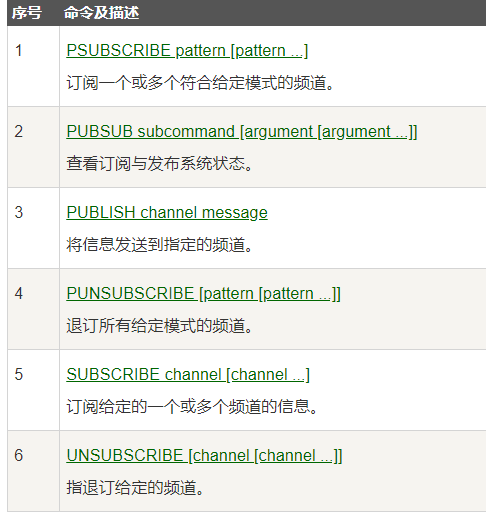
…………….

* 1. 与Hash相关命令
     1. HMSET key field1 value1 [field2 value2] //同时将多个field-value对设置到哈希表中
     2. HAETALL key //获取在哈希表中指定key中的所有字段和值
     3. HEDL key filed1[field2] //删除一个或多个哈希表字段
     4. HEXISTS key field //查看哈希表key中，指定的字段是否存在
  2. 与List相关命令
  3. 与Set相关命令
  4. 与Sorted set相关命令

1. Redis发布订阅
   1. 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。



* 1. 与发布订阅相关命令：



1. Redis事务
   1. 前面错了，redis单个操作是原子性的，但事务不是！！！事务可以理解为一个打包的批量执行脚本，但批量指令并非原子化的操作，中间某条指令的失败不会导致前面已做指令的回滚，也不会造成后续的指令不做。
   2. 一个事务经历三个过程：开始事务——命令入队——执行事务。以 MULTI 开始一个事务， 然后将多个命令入队到事务中， 最后由 EXEC 命令触发事务
   3. 与事务相关命令：
      1. DISCARD //取消事务块所有的命令
      2. EXEC //执行事务块所有的命令
      3. MULTI //标记一个事务块的开始
      4. UNWATCH //取消对所有key的监视
      5. WATCH key [key ...] //监视一个(或多个) key ，如果在事务执行之前这个(或这些) key 被其他命令所改动，那么事务将被打断。
2. Redis脚本（包含动态SQL的功能）（shell脚本也是C语法+linux命令）
   1. Redis 脚本使用 Lua 解释器来执行脚本。 Redis 2.6 版本通过内嵌支持 Lua 环境。
   2. Redis事务的缺陷：命令错误，set写成sett，整个事务无法执行；运行时错误，sadd写成zadd，语法没毛病，但已经入库，不支持回滚，需手动修复；命令之间无法实现逻辑关系。
   3. Lua脚本的好处：lua脚本是作为一个整体执行的.所以中间不会被其他命令插入; 可以把多条命令一次性打包,所以可以有效减少网络开销; 脚本可以常驻在redis内存中,所以在使用的时候,可以直接拿来复用.也减少了代码量.
   4. 与lua脚本相关命令



1. Redis连接
   1. 客户端连接redis服务
   2. 与连接相关命令



1. Redis服务器
   1. 服务器命令主要是用于管理 redis 服务。
   2. 与服务器相关命令



1. 其他高级功能————略
2. Java使用Redis
   1. 安装Redis服务
   2. 下载Redis驱动包
   3. 连接使用









1. 应用场景<https://www.jianshu.com/p/40dbc78711c8>
   1. 热点数据的缓存

由于访问速度快、支持数据类型丰富，最常用的功能是存储热点数据，结合expire设置过期时间再进行缓存更新操作。

* 1. 限时业务的应用

使用expire命令设置一个键的生存时间，到时间后redis删除它，利用这一特性可以运用在限时的优惠活动信息、手机验证码等业务信息。

* 1. 计数器相关问题

利用incrby命令可以实现原子性的递增，可以运用于高并发的秒杀活动、分布式序列号的生成、限制一个手机号发多少条短信、一个接口一分钟限制多少请求等等

* 1. 排行榜相关问题

关系型数据库在排行榜方面查询普遍偏慢，借助redis的SortedSet进行热点数据的排序。

* 1. 分布式锁

利用redis的setnx命令进行，setnx："set if not exists"就是如果不存在则成功设置缓存同时返回1，否则返回0 。

* 1. 延时操作

如订单超时

* 1. 分页、模糊查询

redis的set集合中提供了一个zrangebylex方法，利用这个特性，速度比mysql的like查询稍微快些，但减少大量的io操作，还是能提高性能。

* 1. 点赞、好友等相互关系的存储

set是可以自动排重的，当你需要存储一个列表数据，又不希望出现重复数据时，set是一个很好的选择。速度很快，十分高效。

* 1. 队列

redis有list push和list pop这样的命令，所以能够很方便的执行队列操作。

第二轮 demo阶段

1. 测试环境搭建
   1. 安装gcc：

yum -y install gcc-c++

* 1. 下载Redis压缩包上传或linux上下载，解压：

wget <http://download.redis.io/releases/redis-6.0.4.tar.gz> // yum -y install wget

tar xzf redis-5.0.0.tar.gz

* 1. 进入解压目录，编译：make
  2. Redis服务配置，vim Redis.conf （vim是vi的升级版）
     1. requirepass:配置redis登录密码，默认是不需要的
     2. bind：配置ip：默认为绑定的ip为127.0.0.1，为了远程访问，替换为~~服务器ip~~，修改为0.0.0.0！
     3. port:配置端口号，默认为6399
     4. protected-mode:设置守护进程，默认是开启的，需要改为非开启
     5. daemonize：是否在后台启动redis-server，默认表示为非在后台启动服务，将其设置为后台运行
     6. logfile：日志文件，修改为文件路径

创建文件：vim  /myredis/redis-5.0.4/myredis.log

* 1. 防火墙配置
     1. 查看防火墙状态 firewall-cmd --state
     2. 启动防火墙 systemctl start firewalld.service
     3. 关闭防火墙 ystemctl stop firewalld.service
     4. 向防火墙添加开放端口 firewall-cmd --permanent --zone=public --add-port=6399/tcp
     5. 加载配置 firewall-cmd –reload

注意：阿里云服务器或者腾讯云服务器还需要在后台设置开放端口

* 1. 启动redis

src/redis-server redis.conf //带上配置文件启动

src/redis-server& //也可直接后台启动

cat myredis.log //查看启动情况

ps aux | less //查看所有正在运行的进程

ps –A | grep redis //查看进程

输入q //退出ps

netstat -lntp | grep 6379 //查看端口

* 1. 简单使用

src/redis-cli //进入命令框

或

src/redis-cli -h 192.168.202.134 -p 6399 -a 123456

exit //退出

或者使用客户端连接使用

* 1. 关闭redis

src/redis-cli shutdown

或

redis-cli -h ip -p port -a password shutdown

1. Redis的java客户端(或者说驱动)：Jedis、Lettuce等等。（O2O项目用的是Jedis和Jedis连接池，mall项目用spring-data-redis）
2. Spting-data-redis：它依赖jedis或Lettuce，内部实现了对Lettuce和jedis两个客户端的封装，默认使用的是Lettuce，提供一套与客户端无关的api供应用使用，从而你在从一个redis客户端切换为另一个客户端，不需要修改业务代码。

所以，这里使用spring-data-redis。

1. Redis环境搭好了，可不知道拿来干嘛。。。仿造mall项目的例子也运用一下就算完成第一阶段吧。

根据mall项目中的用法有了灵感，可用于本项目后台用户的菜单管理，给seiVice建立cacheService来注入业务实现，消除卡顿感。