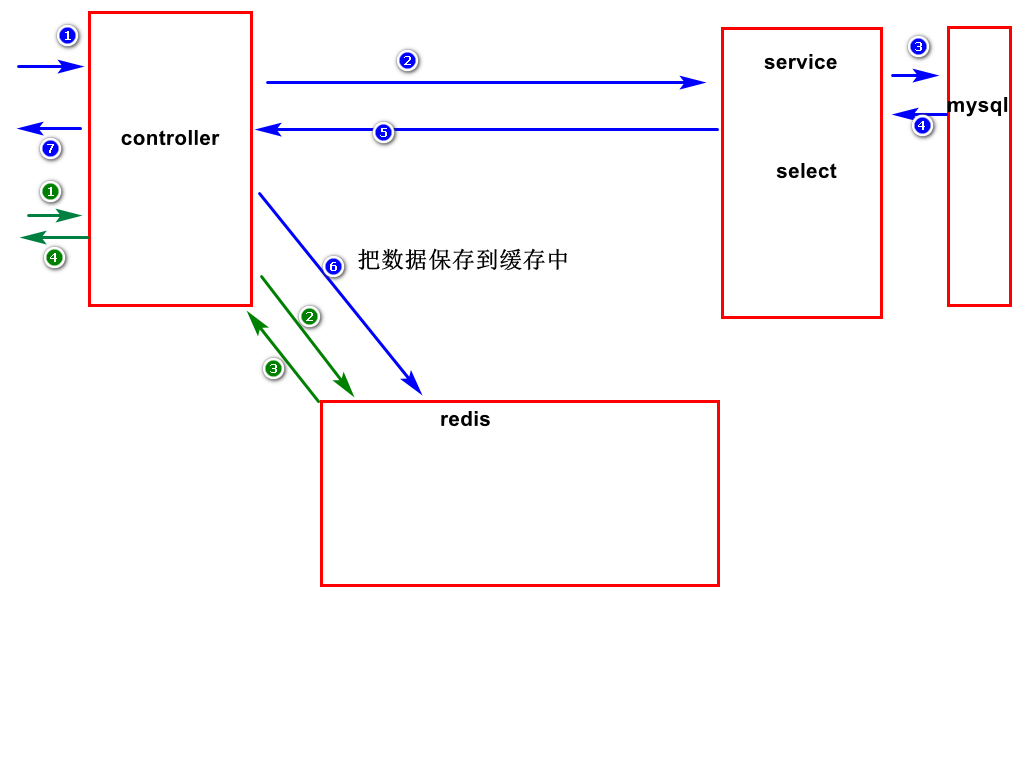
# Redis

## 为什么要用redis

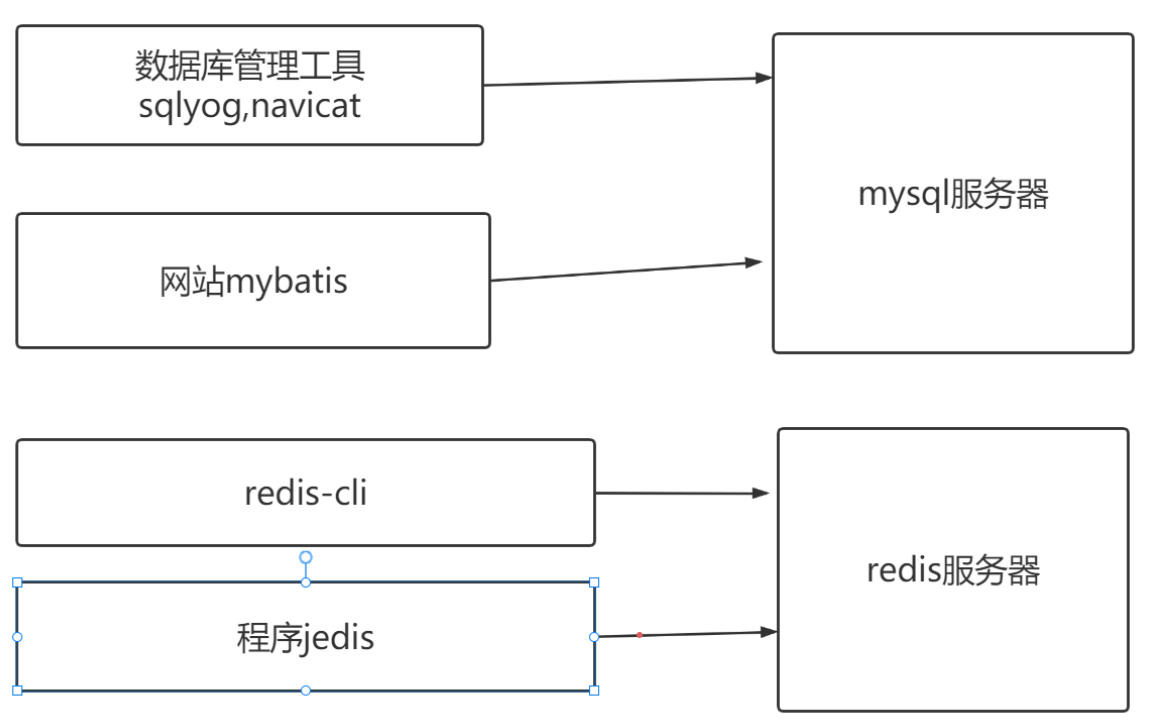
增，修改，删除，查询

查询多，用户不要直接从数据库中查

先从数据库中查，把查到数据放到redis,用户从redis中查速度快。



# Windows上安装redis

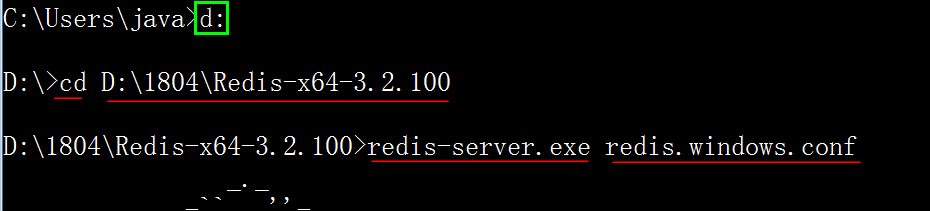


Redis-x64-3.2.100

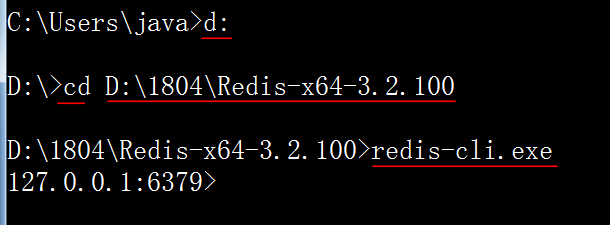
Windows下解压缩安装包，

在资源管理器中找到redis根目录，输入cmd

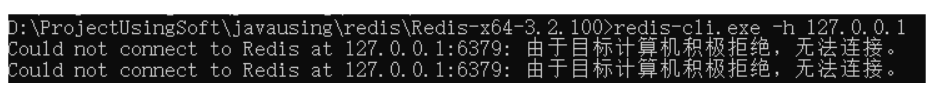
## 启动服务器端



## 启动客户端



关闭服务器会报下面的错误



# Redis支持数据类型

mysql关系型数据库，主外键

redis主要解决的是查询慢

redis中的所有数据数据都是放在hashMap中

hashMap的查找速度是最快的

key value

​ value的类型可以是string,list,set,hashMap

string（字符串）

list（列表）

set（列表）

hash（哈希）

## String

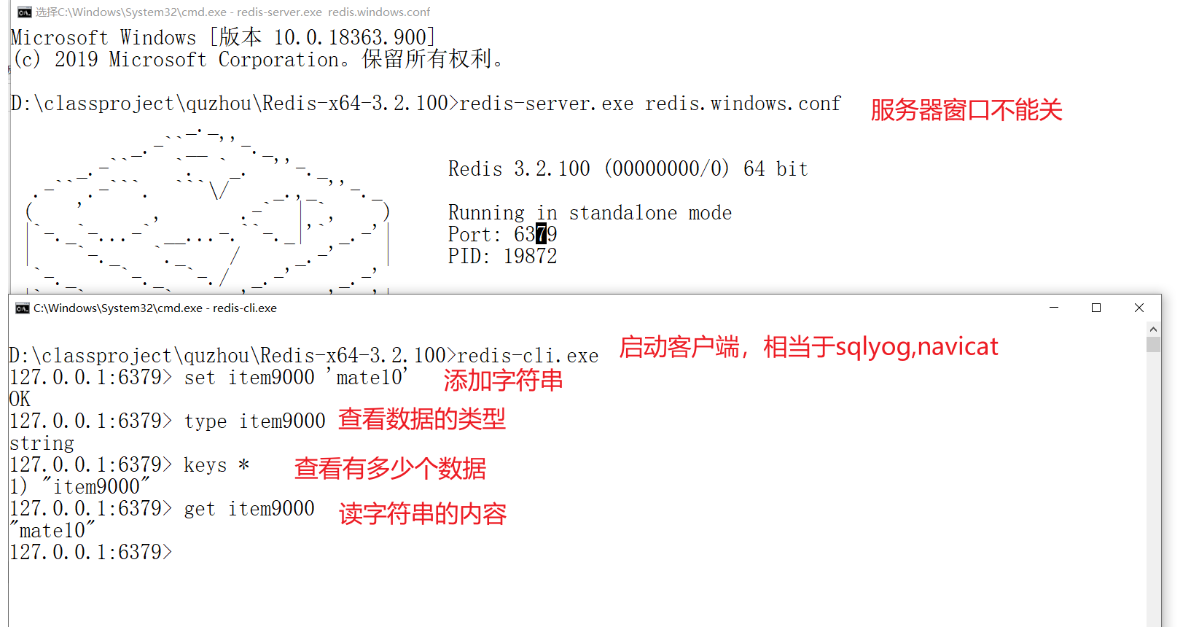
string是redis最基本的类型，能存放字符串，整数，浮点数 ，能存json,json能转成对象，图片

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类型最大能存储512MB。

在windows上只有redis-cli.exe可以连上linux中的redis

redis-cli.exe -h 192.168.64.140 -p 6379



## List

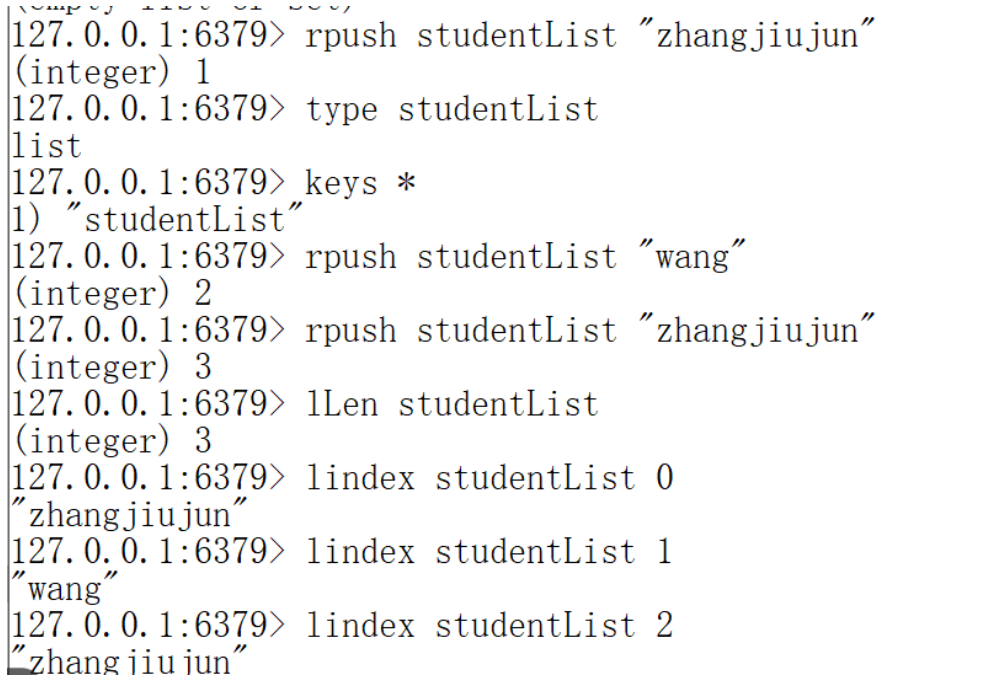
list是列表，可以存放字符串，整数，浮点数，按照插入顺序排序。你可以添加一个元素到列表的头部（左边）或者尾部（右边）。

使用rpush命令向列表头部存储数据

使用lpop命令取出数据

使用lLen取长度

使用lIndex取某个值



list可以重复

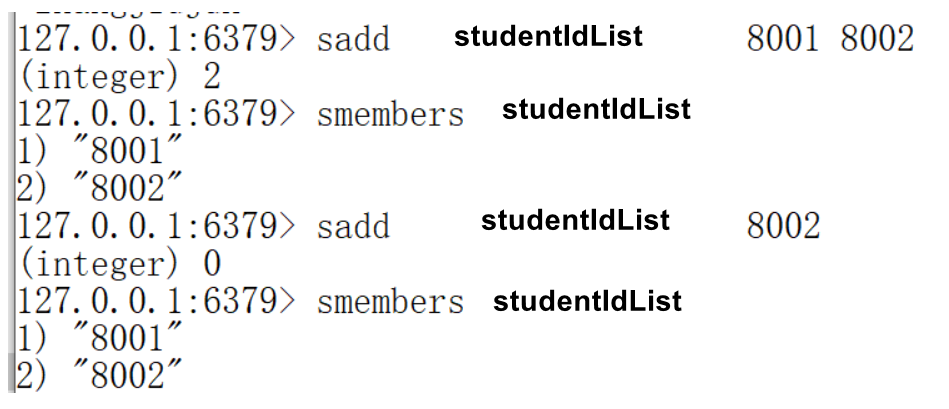
## set

Set是string类型的无序集合。

无序集合中元素不允许重复。

sadd命令向集合中添加数据（无序），使用smembers命令读取集合中的所有数据

用集合放学号，学号不能重复



## Hash

**hash 是一个键名对集合。**

**前面学习过value可以是string,list,set，value也可以是hash.**

需求：在redis中保存学生信息，学生信息包含number,studentName,mobile,age

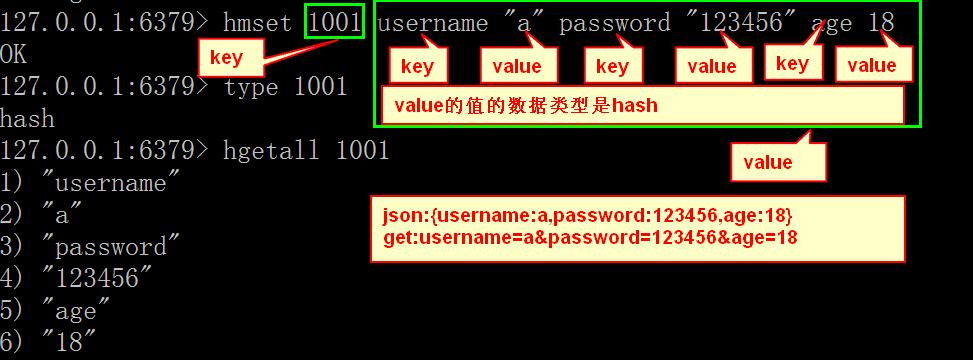
| **key** | **value** |
| --- | --- |
| 88 | number:88, studentName:zhangjiujun mobile:1369, age:18 |
| 99 | number:99, studentName:wang mobile:1369, age:18 |

login?username=a&password=1

hmset 88 number 88 studentName zhangjiujun

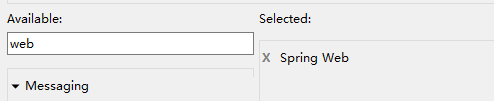
命令 key key value key value

**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。**



# Springboot和redis整合

## 创建项目



<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-redis</artifactId>  
</dependency>

## 写数据

application.yml

server:  
 port: 1314  
spring:  
 redis:  
 host: 127.0.0.1  
 port: 6379

public class Item {  
 String itemId;  
 String itemName;

//构造方法

public Item(){

}

public Item(String itemId, String itemName) {  
 this.itemId = itemId;  
 this.itemName = itemName;  
}

public String getItemId() {  
 return itemId;  
 }  
  
 public void setItemId(String itemId) {  
 this.itemId = itemId;  
 }  
  
 public String getItemName() {  
 return itemName;  
 }  
  
 public void setItemName(String itemName) {  
 this.itemName = itemName;  
 }  
}

@RestController  
public class RedisController {  
 @Autowired*//redis框架会自动创建StringRedisTemplate对象，放到spring容器中。* StringRedisTemplate stringRedisTemplate;  
  
  
 @RequestMapping("/insert")  
 public String insert() throws Throwable {  
  
 *//1,创建对象* Item item = new Item("1008", "电脑");  
 *//2,把对象转成字符串* ObjectMapper objectMapper = new ObjectMapper();  
 String itemString = objectMapper.writeValueAsString(item);  
 *//3,把字符串保存到redis  
 //redis-cli有set,get的命令，  
 //stringRedisTemplate中有对应的方法，方法名和命令名一样* stringRedisTemplate.opsForValue().set("1008", itemString);  
  
 *//返回redis* return "insert ok";  
 }

127.0.0.1:6379> keys \*

127.0.0.1:6379> type 1008

string

127.0.0.1:6379> get 1008

"{\"itemId\":\"1008\",\"itemName\":\"\xe7\x94\xb5\xe8\x84\x91\"}"

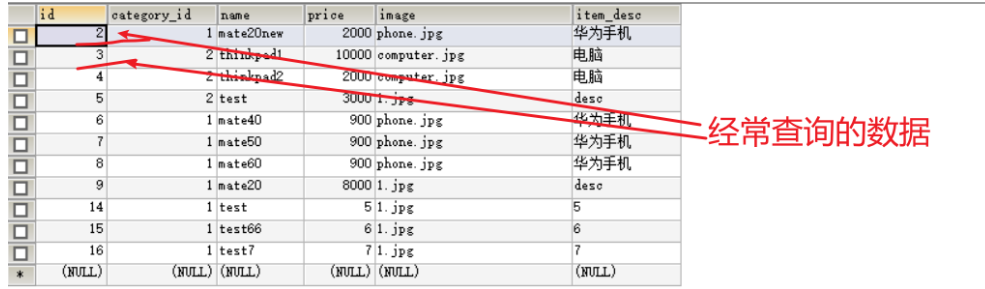
## 读数据

@RequestMapping("/select")  
 public Item select() throws Exception{  
  
 String string = stringRedisTemplate.opsForValue().get("1008");  
 ObjectMapper objectMapper=new ObjectMapper();  
 Item item=objectMapper.readValue(string,Item.class);  
 return item;  
 }

# 商城中使用redis

在商城项目中继续完成redis操作，不创建新项目。

item表有经常查询的商品，比如id为2，3



## 添加依赖

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-redis</artifactId>  
</dependency>

## 修改application.yml

spring:  
 redis:  
 host: 127.0.0.1  
 port: 6379  
 datasource:   
*# 有的mysql如5.0 下面必须改成com.mysql.jdbc.Driver* driver-class-name: com.mysql.cj.jdbc.Driver   
 url: jdbc:mysql://localhost:13306/mall?useUnicode=true&characterEncoding=UTF-8&serverTimezone=GMT%2b8  
 username: root   
 password: 'root'

## ItemController

直接把商品以字符串格式存入到redis中，读取更方便

public class ItemController {  
 @Autowired  
 StringRedisTemplate stringRedisTemplate;  
   
 @RequestMapping("/api/v1/item/selectItemWriteToRedis")  
 *//打开sqlyog，选一件热销商品放到redis中* public ServerResult selectItemWriteToRedis(Integer itemId) throws Throwable{  
 *//从数据库查询热销商品* Item item= itemService.selectByItemId(itemId);  
 *//把item保存到redis中* ObjectMapper objectMapper=new ObjectMapper();  
 String string=objectMapper.writeValueAsString(item);  
 stringRedisTemplate.opsForValue().set(itemId.toString(),string);  
 return new ServerResult(0,"写redis成功",null);  
 }

返回的msg中加from,告诉浏览器数据来之redis还是来自mysql

@RequestMapping("/api/v1/item/selectByItemId")  
 public ServerResult selectByItemId(Integer itemid) throws Throwable{  
 *//先从redis中查* String string=stringRedisTemplate.opsForValue().get(itemid.toString());  
 Item item=null;

String from=””  
 if (ObjectUtils.*isEmpty*(string)){  
 *//如果redis中没有数据，再到数据库中查* item=itemService.selectByItemId(itemid);  
 }else{  
 *//从redis中查到了，把redis中的字符串转成对象* ObjectMapper objectMapper=new ObjectMapper();  
 item=objectMapper.readValue(string,Item.class);  
 }  
 if (ObjectUtils.*isEmpty*(item)){  
 return new ServerResult(0,"查询失败",null);  
 }  
 else {  
 return new ServerResult(0,"查询成功",item);  
 }  
  
 }

## 保存用户信息到redis中

登录成功后，保存用户信息到redis中

## 分布式场景不使用session原因

Sessin信息保存在web服务器中，下一个请求会到另一台服务器，要使用redis保存登录的信息

浏览器端把useid保存到 浏览器cookie中，