**尚品汇商城**

版本：V 2.0

**库存单元（sku）管理**

# 一、业务介绍

## 1 SPU与SKU

SPU(Standard Product Unit)：标准化产品单元。是商品信息聚合的最小单位，是一组**可复用、易检索**的标准化信息的集合，该集合描述了一个产品的特性。

SKU=Stock Keeping Unit（库存量单位）。即库存进出计量的基本单元，可以是以件，盒，托盘等为单位。SKU这是对于大型连锁超市DC（配送中心）物流管理的一个必要的方法。现在已经被引申为产品统一编号的简称，每种产品均对应有唯一的SKU号。

比如，咱们购买一台iPhoneX手机，iPhoneX手机就是一个SPU，但是你购买的时候，不可能是以iPhoneX手机为单位买的，商家也不可能以iPhoneX为单位记录库存。必须要以什么颜色什么版本的iPhoneX为单位。比如，你购买的是一台银色、128G内存的、支持联通网络的iPhoneX ，商家也会以这个单位来记录库存数。那这个更细致的单位就叫库存单元（SKU）。



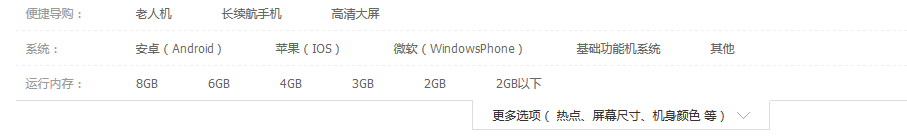
销售属性与平台属性

销售属性，就是商品详情页右边，可以通过销售属性来定位一组spu下的哪款sku。可以让当前的商品详情页，跳转到自己的“兄弟”商品。

一般每种商品的销售属性不会太多，大约1-4种。整个电商的销售属性种类也不会太多，大概10种以内。比如：颜色、尺寸、版本、套装等等。不同销售属性的组合也就构成了一个spu下多个sku的结构。



平台属性， 就是之前分类下面，辅助搜索的，类似于条件的属性。



销售属性与平台属性各自独立。一个SPU会决定一个商品都有哪些销售属性，比如iPhonx会有颜色、版本、内存的销售属性，某个T桖衫只有尺寸这个销售属性。

而某个商品有什么平台属性，由他的3级分类决定。比如笔记本包括：运行内存、cpu、显卡、硬盘、屏幕尺寸等等。

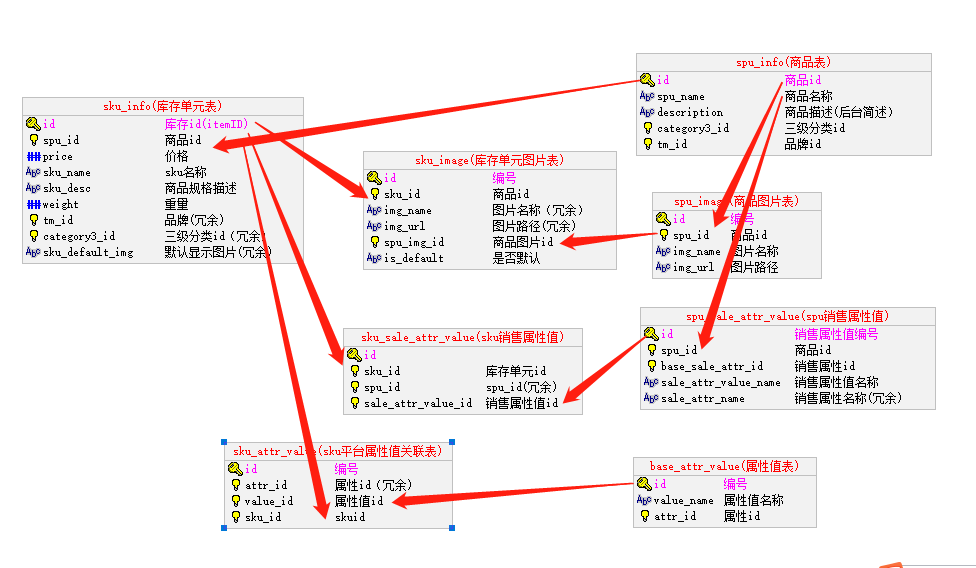
SKU与SPU的图片资源

另外同一个SPU下的SKU可以共用一些资源，比如商品图片，海报等等。毕竟同一种商品，大部分图片都是共用的只有因为颜色尺寸等，很少的差别。那么一般来说商品图片都是在新增SPU时上传的，在新增SKU时从该SPU已上传的图片中选择。

而海报几乎是所有SPU下的SKU都一样。

## 2 数据库表结构

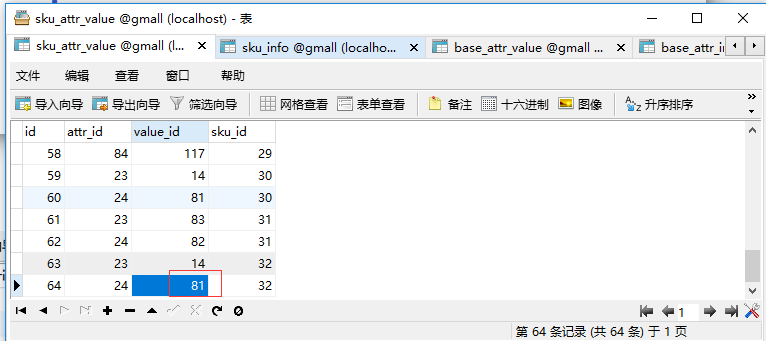
根据以上的需求，以此将SPU关联的数据库表结构设计为如下：



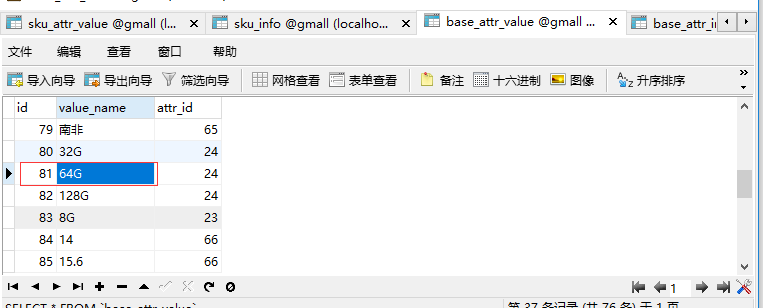
sku\_info



sku\_attr\_value: 平台属性关联表



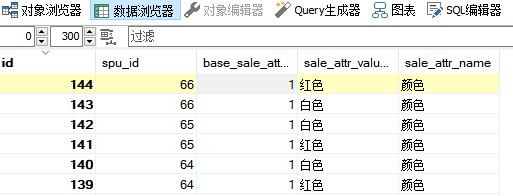
base\_attr\_value 平台属性值表



base\_attr\_info平台属性表



spu\_sale\_attr\_value：销售属性值表



## 3 数据准备：

平台属性添加：



商品spu管理—添加小米手机信息



添加销售属性信息



# 二 开发增加功能【为spu添加sku】

参考接口文档《电商管理系统API接口文档 .doc》sku接口

## 1 页面





## 2 难点分析

1、要动态生成平台属性和销售属性的下拉菜单

2、要生成图片列表以供用户选择

## 3 解决动态生成下拉菜单：

用ajax根据spuid（商品id）查询销售属性及销售属性值的列表数据。 然后通过js把数据展开成html，变为多下拉菜单。

## 4 保存

### 4.1图片加载功能

功能分析：图片列表是根据spuId得来，涉及到的数据库表spu\_image

|  |
| --- |
| 接口  */\*\*  \* 根据spuId 查询spuImageList  \** ***@param spuId*** *\** ***@return*** *\*/* List<SpuImage> getSpuImageList(Long spuId); |
| 实现类  @Override **public** List<SpuImage> getSpuImageList(Long spuId) {  QueryWrapper<SpuImage> queryWrapper = **new** QueryWrapper<>();  queryWrapper.eq(**"spu\_id"**, spuId);  **return spuImageMapper**.selectList(queryWrapper); } |
| 控制器[SkuManageController ]  @Api(tags = **"商品SKU接口"**) @RestController @RequestMapping(**"admin/product"**) **public class** SkuManageController {   @Autowired  **private** ManageService **manageService**;   */\*\*  \* 根据spuId 查询spuImageList  \** ***@param spuId*** *\** ***@return*** *\*/* @GetMapping(**"spuImageList/{spuId}"**)  **public** Result<List<SpuImage>> getSpuImageList(@PathVariable(**"spuId"**) Long spuId) {  List<SpuImage> spuImageList = **manageService**.getSpuImageList(spuId);  **return** Result.*ok*(spuImageList);  }  } |

### 4.2 平台属性功能：动态加载

1. 在BaseAttrInfo 实体类中添加BaseAttrValue 的实体类集合

|  |
| --- |
| @TableField(exist = **false**) **private** List<BaseAttrValue> **attrValueList**; |

平台属性与一级分类、二级分类和三级分类Id有关系

1. 在前面讲解平台属性时已经有过相应接口，直接使用，如下：

|  |
| --- |
| */\*\*  \* 根据分类Id 获取平台属性数据  \* 接口说明：  \* 1，平台属性可以挂在一级分类、二级分类和三级分类  \* 2，查询一级分类下面的平台属性，传：category1Id，0，0； 取出该分类的平台属性  \* 3，查询二级分类下面的平台属性，传：category1Id，category2Id，0；   \* 取出对应一级分类下面的平台属性与二级分类对应的平台属性  \* 4，查询三级分类下面的平台属性，传：category1Id，category2Id，category3Id；  \* 取出对应一级分类、二级分类与三级分类对应的平台属性  \*  \** ***@param category1Id*** *\** ***@param category2Id*** *\** ***@param category3Id*** *\** ***@return*** *\*/*  List<BaseAttrInfo> getAttrInfoList(Long category1Id, Long category2Id, Long category3Id);  } |
| 对应controller  */\*\*  \* 根据分类Id 获取平台属性数据  \** ***@param category1Id*** *\** ***@param category2Id*** *\** ***@param category3Id*** *\** ***@return*** *\*/* @GetMapping(**"attrInfoList/{category1Id}/{category2Id}/{category3Id}"**)  **public** Result<List<BaseAttrInfo>> attrInfoList(@PathVariable(**"category1Id"**) Long category1Id,  @PathVariable(**"category2Id"**) Long category2Id,  @PathVariable(**"category3Id"**) Long category3Id) {  List<BaseAttrInfo> baseAttrInfoList = **manageService**.getAttrInfoList(category1Id, category2Id, category3Id);  **return** Result.*ok*(baseAttrInfoList);  }  } |

### 4.3 销售属性：

页面加载的SkuManageController

|  |
| --- |
| */\*\*  \* 根据spuId 查询销售属性集合  \** ***@param spuId*** *\** ***@return*** *\*/* @GetMapping(**"spuSaleAttrList/{spuId}"**) **public** Result<List<SpuSaleAttr>> getSpuSaleAttrList(@PathVariable(**"spuId"**) Long spuId) {  List<SpuSaleAttr> spuSaleAttrList = **manageService**.getSpuSaleAttrList(spuId);  **return** Result.*ok*(spuSaleAttrList); } |
| 接口  */\*\*  \* 根据spuId 查询销售属性集合  \** ***@param spuId*** *\** ***@return*** *\*/* List<SpuSaleAttr> getSpuSaleAttrList(Long spuId);  实现类  @Override **public** List<SpuSaleAttr> getSpuSaleAttrList(Long spuId) {  **return spuSaleAttrMapper**.selectSpuSaleAttrList(spuId); }  Mapper  @Mapper **public interface** SpuSaleAttrMapper **extends** BaseMapper<SpuSaleAttr> {  *// 根据spuId 查询销售属性集合* List<SpuSaleAttr> selectSpuSaleAttrList(Long spuId); }  *<?***xml version="1.0" encoding="UTF-8"***?>* **<!DOCTYPE mapper SYSTEM "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>*** <**mapper namespace="com.atguigu.gmall.product.mapper.SpuSaleAttrMapper"**>  <**resultMap id="spuSaleAttrMap" type="com.atguigu.gmall.model.product.SpuSaleAttr" autoMapping="true"**>   <**id property="id" column="id"**></**id**>  *<!--一对多-->* <**collection property="spuSaleAttrValueList" ofType="com.atguigu.gmall.model.product.SpuSaleAttrValue" autoMapping="true"**>  <**id property="id" column="sale\_attr\_value\_id"**></**id**>  </**collection**>  </**resultMap**>  <**sql id="spuSaleAttr"**> sa.id ,sa.spu\_id, sa.sale\_attr\_name,sa.base\_sale\_attr\_id,  sv.id sale\_attr\_value\_id,  sv.sale\_attr\_value\_name </**sql**> <**select id="selectSpuSaleAttrList" resultMap="spuSaleAttrMap"**>  select  <**include refid="spuSaleAttr"**></**include**>  from spu\_sale\_attr sa inner join spu\_sale\_attr\_value sv  on sa.spu\_id=sv.spu\_id and sa.base\_sale\_attr\_id=sv.base\_sale\_attr\_id  where sa.spu\_id=#{spu\_id} </**select**> </**mapper**> |

## 5点击保存按钮

### 5.1建立数据库表，sku\_info，sku\_attr\_value，sku\_sale\_attr\_value，sku\_image 并添加实体类

|  |
| --- |
| @Data @ApiModel(description = **"SkuInfo"**) @TableName(**"sku\_info"**) **public class** SkuInfo **extends** BaseEntity {    **private static final long *serialVersionUID*** = 1L;    @ApiModelProperty(value = **"商品id"**)  @TableField(**"spu\_id"**)  **private** Long **spuId**;   @ApiModelProperty(value = **"价格"**)  @TableField(**"price"**)  **private** BigDecimal **price**;   @ApiModelProperty(value = **"sku名称"**)  @TableField(**"sku\_name"**)  **private** String **skuName**;   @ApiModelProperty(value = **"商品规格描述"**)  @TableField(**"sku\_desc"**)  **private** String **skuDesc**;   @ApiModelProperty(value = **"重量"**)  @TableField(**"weight"**)  **private** String **weight**;   @ApiModelProperty(value = **"品牌(冗余)"**)  @TableField(**"tm\_id"**)  **private** Long **tmId**;   @ApiModelProperty(value = **"三级分类id（冗余)"**)  @TableField(**"category3\_id"**)  **private** Long **category3Id**;   @ApiModelProperty(value = **"默认显示图片(冗余)"**)  @TableField(**"sku\_default\_img"**)  **private** String **skuDefaultImg**;   @ApiModelProperty(value = **"是否销售（1：是 0：否）"**)  @TableField(**"is\_sale"**)  **private** Integer **isSale**;   @TableField(exist = **false**)  List<SkuImage> **skuImageList**;   @TableField(exist = **false**)  List<SkuAttrValue> **skuAttrValueList**;   @TableField(exist = **false**)  List<SkuSaleAttrValue> **skuSaleAttrValueList**; } |
| @Data @ApiModel(description = **"Sku销售属性值"**) @TableName(**"sku\_sale\_attr\_value"**) **public class** SkuSaleAttrValue **extends** BaseEntity {    **private static final long *serialVersionUID*** = 1L;    @ApiModelProperty(value = **"库存单元id"**)  @TableField(**"sku\_id"**)  **private** Long **skuId**;   @ApiModelProperty(value = **"spu\_id(冗余)"**)  @TableField(**"spu\_id"**)  **private** Long **spuId**;   @ApiModelProperty(value = **"销售属性值id"**)  @TableField(**"sale\_attr\_value\_id"**)  **private** Long **saleAttrValueId**;  } |
| @Data @ApiModel(description = **"Sku图片"**) @TableName(**"sku\_image"**) **public class** SkuImage **extends** BaseEntity {    **private static final long *serialVersionUID*** = 1L;    @ApiModelProperty(value = **"商品id"**)  @TableField(**"sku\_id"**)  **private** Long **skuId**;   @ApiModelProperty(value = **"图片名称（冗余）"**)  @TableField(**"img\_name"**)  **private** String **imgName**;   @ApiModelProperty(value = **"图片路径(冗余)"**)  @TableField(**"img\_url"**)  **private** String **imgUrl**;   @ApiModelProperty(value = **"商品图片id"**)  @TableField(**"spu\_img\_id"**)  **private** Long **spuImgId**;   @ApiModelProperty(value = **"是否默认"**)  @TableField(**"is\_default"**)  **private** String **isDefault**;  } |
| @Data @ApiModel(description = **"Sku平台属性值"**) @TableName(**"sku\_attr\_value"**) **public class** SkuAttrValue **extends** BaseEntity {    **private static final long *serialVersionUID*** = 1L;    @ApiModelProperty(value = **"属性id（冗余)"**)  @TableField(**"attr\_id"**)  **private** Long **attrId**;   @ApiModelProperty(value = **"属性值id"**)  @TableField(**"value\_id"**)  **private** Long **valueId**;   @ApiModelProperty(value = **"skuid"**)  @TableField(**"sku\_id"**)  **private** Long **skuId**;  } |

|  |
| --- |
| 创建对应数据库表实体类，以及接口mapper @Mapper **public interface** SkuInfoMapper **extends** BaseMapper<SkuInfo> { } |
| @Mapper **public interface** SkuImageMapper **extends** BaseMapper<SkuImage> { } |
| @Mapper **public interface** SkuAttrValueMapper **extends** BaseMapper<SkuAttrValue> { } |
| @Mapper **public interface** SkuSaleAttrValueMapper **extends** BaseMapper<SkuSaleAttrValue> { } |

### 5.2 接口，实现类，控制器

SkuManageController

|  |
| --- |
| */\*\*  \* 保存sku  \** ***@param skuInfo*** *\** ***@return*** *\*/* @PostMapping(**"saveSkuInfo"**) **public** Result saveSkuInfo(@RequestBody SkuInfo skuInfo) {  *// 调用服务层* **manageService**.saveSkuInfo(skuInfo);  **return** Result.*ok*(); } |
| **接口**  */\*\*  \* 保存数据  \** ***@param skuInfo*** *\*/* **void** saveSkuInfo(SkuInfo skuInfo); |
| **实现类**  @Override @Transactional **public void** saveSkuInfo(SkuInfo skuInfo) {  */\*  skuInfo 库存单元表 --- spuInfo！  skuImage 库存单元图片表 --- spuImage!  skuSaleAttrValue sku销售属性值表{sku与销售属性值的中间表} --- skuInfo ，spuSaleAttrValue  skuAttrValue sku与平台属性值的中间表 --- skuInfo ，baseAttrValue  \*/* **skuInfoMapper**.insert(skuInfo);  List<SkuImage> skuImageList = skuInfo.getSkuImageList();  **if** (skuImageList != **null** && skuImageList.size() > 0) {   *// 循环遍历* **for** (SkuImage skuImage : skuImageList) {  skuImage.setSkuId(skuInfo.getId());  **skuImageMapper**.insert(skuImage);  }  }   List<SkuSaleAttrValue> skuSaleAttrValueList = skuInfo.getSkuSaleAttrValueList();  *// 调用判断集合方法* **if** (!CollectionUtils.*isEmpty*(skuSaleAttrValueList)) {  **for** (SkuSaleAttrValue skuSaleAttrValue : skuSaleAttrValueList) {  skuSaleAttrValue.setSkuId(skuInfo.getId());  skuSaleAttrValue.setSpuId(skuInfo.getSpuId());  **skuSaleAttrValueMapper**.insert(skuSaleAttrValue);  }  }   List<SkuAttrValue> skuAttrValueList = skuInfo.getSkuAttrValueList();  **if** (!CollectionUtils.*isEmpty*(skuAttrValueList)) {  **for** (SkuAttrValue skuAttrValue : skuAttrValueList) {  skuAttrValue.setSkuId(skuInfo.getId());  **skuAttrValueMapper**.insert(skuAttrValue);  }  } } |