**谷粒商城**

版本：V 1.0

**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）。



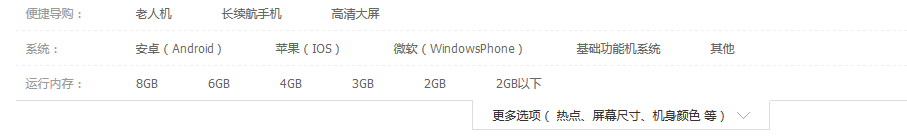
销售属性与平台属性

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。

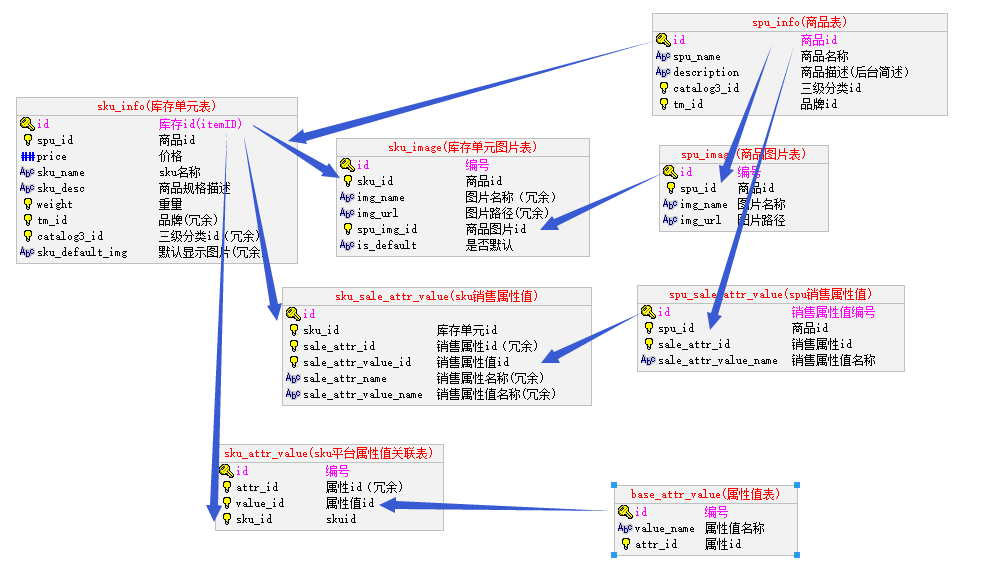
SKU与SPU的图片资源

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

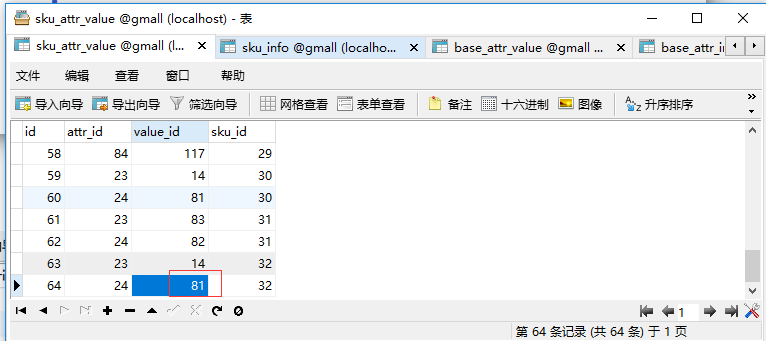
根据以上的需求，以此将SPU关联的数据库表结构设计为如下：



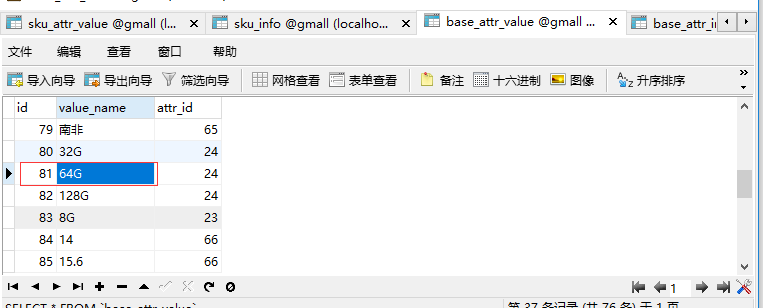
Sku\_info

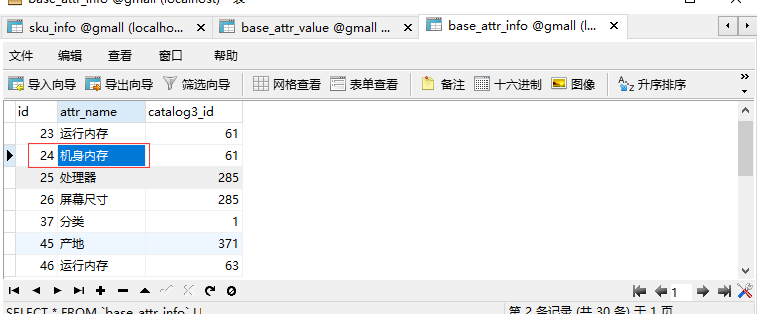


Sku\_attr\_value: 平台属性关联表

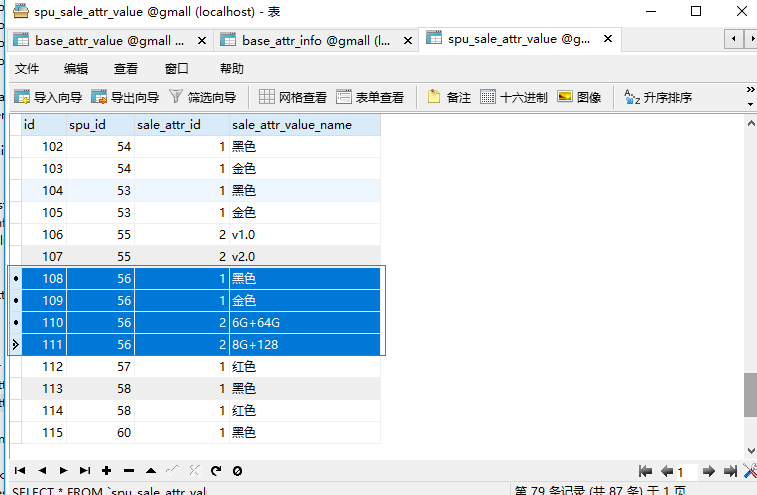


Base\_attr\_value



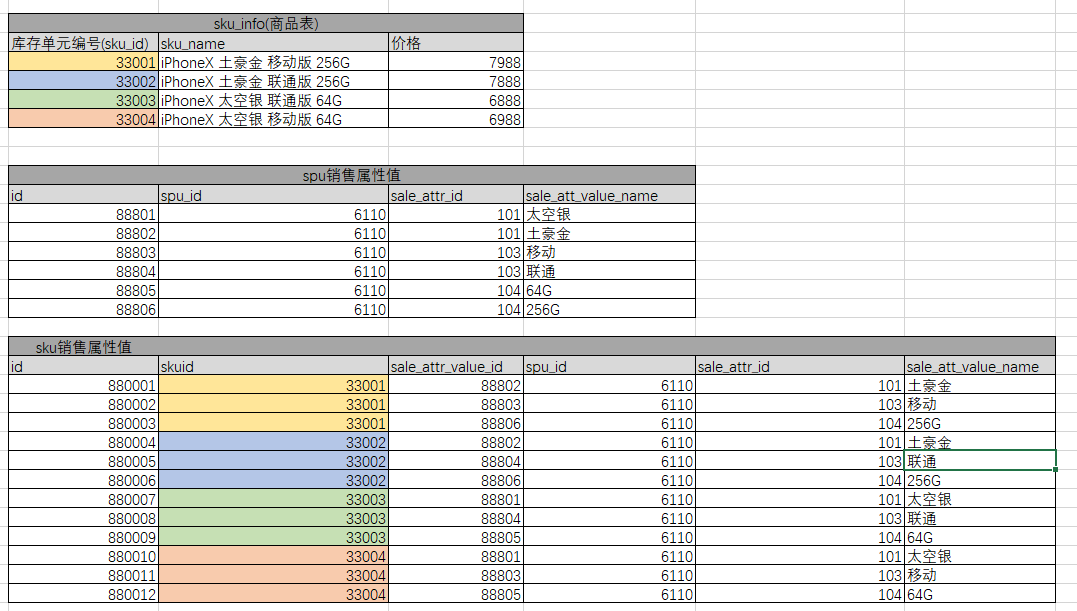


spu\_sale\_attr\_value：销售属性值表

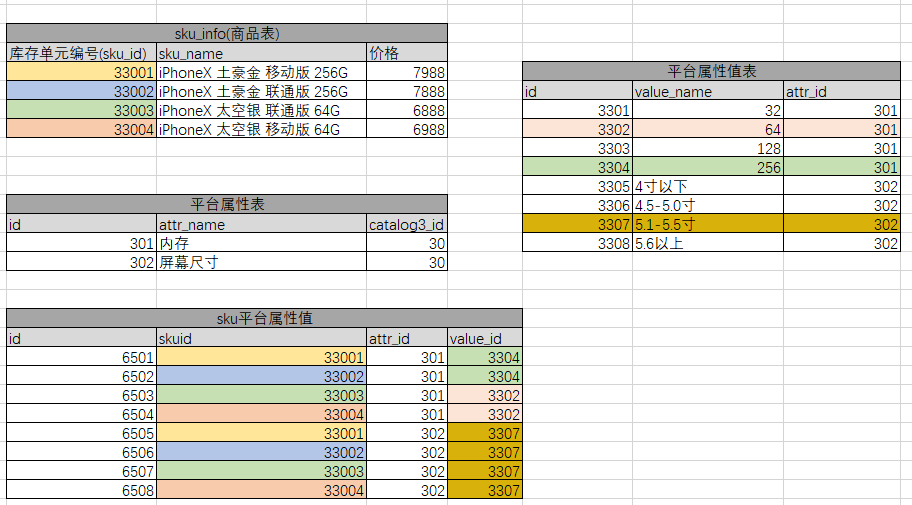


数据案例：

销售属性

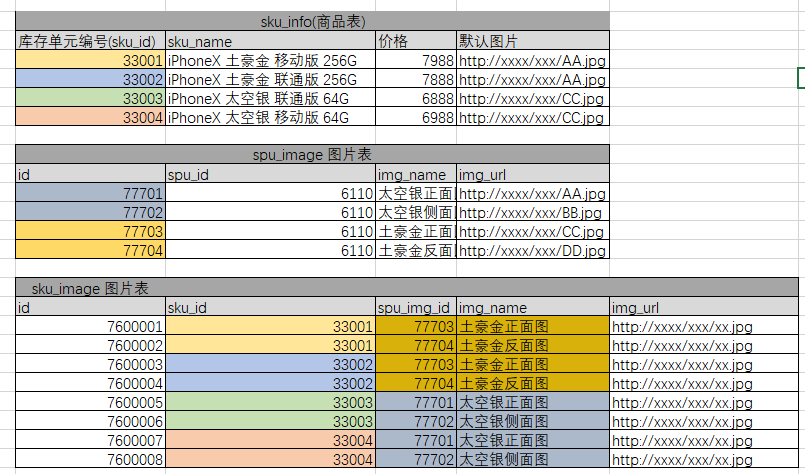


平台属性



图片：

Sku\_image 的数据都是从Spu\_image中选出来。



## 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】

## 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

### 图片加载功能

* 1. 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获取spuImage中的所有图片列表* List<SpuImage> getSpuImageList(String spuId); |
| 实现类  @Override **public** List<SpuImage> getSpuImageList(String spuId) {  SpuImage spuImage = **new** SpuImage();  spuImage.setSpuId(spuId);  **return spuImageMapper**.select(spuImage); } |
| 控制器[SkuManageController] @CrossOrigin  @RequestMapping(**"spuImageList"**) @ResponseBody **public** List<SpuImage> spuImageList(String spuId){  **return manageService**.getSpuImageList(spuId); } |

### 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

|  |
| --- |
| @Transient **private** List<BaseAttrValue> **attrValueList**; |

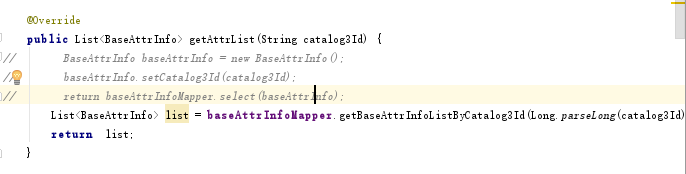
平台属性根据三级分类Id有关系

1. 建立接口，以及mapper

|  |
| --- |
| **public interface** BaseAttrInfoMapper **extends** Mapper<BaseAttrInfo> {  *// 根据三级分类id查询属性表* List<BaseAttrInfo> getBaseAttrInfoListByCatalog3Id(Long catalog3Id); } |

1. 创建mapper文件建立BaseAttrInfoMapper.xml 对应的接口BaseAttrInfoMapper.java   
   mybatis 中有包扫描形 package name=”com.atguigu.mapper”
   1. 定义xml的时候，尽量使接口名称跟xml名称保持一致

|  |
| --- |
|  |
| *<?***xml version="1.0" encoding="UTF-8"***?>* **<!DOCTYPE mapper SYSTEM "http://mybatis.org/dtd/mybatis-3-mapper.dtd" *>***  <**mapper namespace="com.atguigu.gmall110.manage.mapper.BaseAttrInfoMapper"**>  <**resultMap id="baseAttrInfoMap" type="com.atguigu.gmall110.bean.BaseAttrInfo" autoMapping="true"**>  <**id column="id" property="id"**></**id**>  <**collection ofType="com.atguigu.gmall110.bean.BaseAttrValue" property="attrValueList" autoMapping="true"**>  <**result property="id" column="attr\_value\_id"**></**result**>  </**collection**>  </**resultMap**>  <**select id="getBaseAttrInfoListByCatalog3Id" resultMap="baseAttrInfoMap"**>  SELECT  bai.id,  bai.attr\_name,  bai.catalog3\_id,  bav.id attr\_value\_id,  bav.value\_name,  bav.attr\_id FROM  base\_attr\_info bai INNER JOIN base\_attr\_value bav ON bai.id = bav.attr\_id WHERE  catalog3\_id = #{0}  </**select**> </**mapper**> |

1. 改造public List<BaseAttrInfo> getAttrList(String catalog3Id) 方法
2. 
3. 完成controller

|  |
| --- |
|  |

注意：在application.properties 配置文件中添加

|  |
| --- |
| **mybatis.mapper-locations**=**classpath:mapper/\*Mapper.xml mybatis.configuration.mapUnderscoreToCamelCase**=**true** |

### 销售属性：

页面加载的controller

|  |
| --- |
| @RequestMapping(**"spuSaleAttrList"**) @ResponseBody **public** List<SpuSaleAttr> getspuSaleAttrList(String spuId){  List<SpuSaleAttr> spuSaleAttrList = **manageService**.getSpuSaleAttrList(spuId);  **return** spuSaleAttrList; } |
| 接口  List<SpuSaleAttr> getSpuSaleAttrList(String spuId);  实现类  @Override **public** List<SpuSaleAttr> getSpuSaleAttrList(String spuId) {  **return spuSaleAttrMapper**.selectSpuSaleAttrList(spuId); }  Mapper  **public interface** SpuSaleAttrMapper **extends** Mapper<SpuSaleAttr> {  **public** 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.manage.mapper.SpuSaleAttrMapper"**>   <**select id ="selectSpuSaleAttrList" parameterType="long" resultMap="spuSaleAttrMap"**>  select sa.id ,sa.spu\_id, sa.sale\_attr\_name,sa.sale\_attr\_id,  sv.id sale\_attr\_value\_id,  sv.sale\_attr\_value\_name  from spu\_sale\_attr sa inner join spu\_sale\_attr\_value sv  on sa.spu\_id=sv.spu\_id and sa.sale\_attr\_id=sv.sale\_attr\_id  where sa.spu\_id=#{spu\_id}  </**select**>  <**resultMap id="spuSaleAttrMap" type="com.atguigu.gmall.bean.SpuSaleAttr" autoMapping="true"**>  <**result property="id" column="id"** ></**result**>  <**collection property="spuSaleAttrValueList" ofType="com.atguigu.gmall.bean.SpuSaleAttrValue" autoMapping="true"**>  <**result property="id" column="sale\_attr\_value\_id"** ></**result**>  </**collection**>  </**resultMap**> </**mapper**> |

## 点击保存按钮

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

|  |
| --- |
| **public class** SkuInfo **implements** Serializable {   @GeneratedValue(strategy = GenerationType.***IDENTITY***)  @Id  @Column  String **id**;   @Column  String **spuId**;   @Column  BigDecimal **price**;   @Column  String **skuName**;   @Column  BigDecimal **weight**;   @Column  String **skuDesc**;   @Column  String **catalog3Id**;   @Column  String **skuDefaultImg**;   @Transient  List<SkuImage> **skuImageList**;   @Transient  List<SkuAttrValue> **skuAttrValueList**;   @Transient  List<SkuSaleAttrValue> **skuSaleAttrValueList**;  } |
| **public class** SkuSaleAttrValue **implements** Serializable {   @Id  @Column  String **id**;   @Column  String **skuId**;   @Column  String **saleAttrId**;   @Column  String **saleAttrValueId**;   @Column  String **saleAttrName**;   @Column  String **saleAttrValueName**;  } |
| **public class** SkuImage **implements** Serializable{   @Id  @Column  String **id**;  @Column  String **skuId**;  @Column  String **imgName**;  @Column  String **imgUrl**;  @Column  String **spuImgId**;  @Column  String **isDefault**;  } |
| **public class** SkuAttrValue **implements** Serializable {   @Id  @Column  String **id**;   @Column  String **attrId**;   @Column  String **valueId**;   @Column  String **skuId**;  } |

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

### 接口，实现类，控制器

|  |
| --- |
| @RequestMapping(**"saveSkuInfo"**) @ResponseBody **public** String saveSkuInfo(@RequestBody SkuInfo skuInfo){  **manageService**.saveSkuInfo(skuInfo);  **return "OK"**; } |
| **接口**  **public void** saveSkuInfo(SkuInfo skuInfo); |
| **实现类**  @Override  **public void** saveSkuInfo(SkuInfo skuInfo) {  *// sku\_info* **if** (skuInfo.getId()==**null** || skuInfo.getId().length()==0){  *// 设置id 为自增* skuInfo.setId(**null**);  **skuInfoMapper**.insertSelective(skuInfo);  }**else** {  **skuInfoMapper**.updateByPrimaryKeySelective(skuInfo);  }   *// sku\_img,* SkuImage skuImage = **new** SkuImage();  skuImage.setSkuId(skuInfo.getId());  **skuImageMapper**.delete(skuImage);   *// insert* List<SkuImage> skuImageList = skuInfo.getSkuImageList();  **if** (skuImageList!=**null** && skuImageList.size()>0){  **for** (SkuImage image : skuImageList) {  */\* "" 区别 null\*/* **if** (image.getId()!=**null** && image.getId().length()==0){  image.setId(**null**);  }  *// skuId 必须赋值* image.setSkuId(skuInfo.getId());  **skuImageMapper**.insertSelective(image);  }  } *// sku\_attr\_value,* SkuAttrValue skuAttrValue = **new** SkuAttrValue();  skuAttrValue.setSkuId(skuInfo.getId());  **skuAttrValueMapper**.delete(skuAttrValue);   *// 插入数据* List<SkuAttrValue> skuAttrValueList = skuInfo.getSkuAttrValueList();  **if** (skuAttrValueList!=**null** && skuAttrValueList.size()>0){  **for** (SkuAttrValue attrValue : skuAttrValueList) {  **if** (attrValue.getId()!=**null** && attrValue.getId().length()==0){  attrValue.setId(**null**);  }  *// skuId* attrValue.setSkuId(skuInfo.getId());  **skuAttrValueMapper**.insertSelective(attrValue);  }  } *// sku\_sale\_attr\_value,* SkuSaleAttrValue skuSaleAttrValue = **new** SkuSaleAttrValue();  skuSaleAttrValue.setSkuId(skuInfo.getId());  **skuSaleAttrValueMapper**.delete(skuSaleAttrValue); *// 插入数据* List<SkuSaleAttrValue> skuSaleAttrValueList = skuInfo.getSkuSaleAttrValueList();  **if** (skuSaleAttrValueList!=**null** && skuSaleAttrValueList.size()>0){  **for** (SkuSaleAttrValue saleAttrValue : skuSaleAttrValueList) {  **if** (saleAttrValue.getId()!=**null** && saleAttrValue.getId().length()==0){  saleAttrValue.setId(**null**);  }  *// skuId* saleAttrValue.setSkuId(skuInfo.getId());  **skuSaleAttrValueMapper**.insertSelective(saleAttrValue);  }  }  } |