1 学习目标

- 了解商品规格数据结构设计思路
- 了解SPU和SKU数据结构设计思路
- 实现规格组和规格CRUD

2 商品规格数据结构

b2c类型的商城是一个全品类的电商网站，因此商品的种类繁多，每一件商品，其属性又有差别。为了更准确描述商品及细分差别，抽象出两个概念：SPU和SKU，了解一下：

2.1 SPU和SKU

- SPU：Standard Product Unit （标准产品单位），一组具有共同属性的商品集
- SKU：Stock Keeping Unit（库存量单位），SPU商品集因具体特性不同而细分的每个商品

以图为例来看：

- 上图的 华为Mate 30 5G就是一个商品集（SPU）
- 当选择颜色 版本 购买方式后,出来的才是一个真正的商品(SKU)

可以看出

- SPU是一个抽象的商品集(集合)概念
- SKU才是具体需要销售的商品,每一个SKU的价格,库存可能都会不一样,用户购买的是SKU而不是SPU

2.2 数据库设计分析

2.2.1 思考并发现问题

弄清楚了SPU和SKU的概念区分，接下来我们一起思考一下该如何设计数据库表。

首先来看SPU，大家一起思考下SPU应该有哪些字段来描述？

```
id:主键
title：标题
description：描述
specification：规格
packaging_list：包装
after_service：售后服务
comment：评价
category_id：商品分类
brand_id：品牌
```

似乎并不复杂，但是大家仔细思考一下，商品的规格字段你如何填写？

主体	入网型号	TAS-AN00
	产品名称	HUAWEI Mate 30 5G
	上市年份	2019年
	上市月份	11月
基本信息	机身长度（mm）	160.8
	机身重量（g）	196克（含电池） 备注：实际重量依配置、制造工艺、测量方法的不同可能有所差异。
	机身材质工艺	其他
	机身宽度（mm）	76.1
	机身材质分类	玻璃后盖
	机身厚度（mm）	8.4
	运营商标志或内容	无
主芯片	CPU品牌	以官网信息为准
存储	最大存储扩展容量	256GB
屏幕	屏幕材质类型	OLED
	屏幕色彩	1670万色
	屏幕刷新率	其他
	主屏幕尺寸（英寸）	6.62英寸 备注：显示屏采用圆角设计，按照标准矩形测量时，屏幕的对角线长度是6.62英寸（实际可视区域略小）
	触摸屏类型	电容屏

不同商品的规格不一定相同，数据库中要如何保存？

下图为联想电脑的规格

主体	型号	联想小新Pro
	认证型号	Lenovo 小新 Pro-13 2020
显示器	屏幕类型	IPS；其他
	屏幕尺寸	13.3英寸
	物理分辨率	2560 x 1600
	显示比例	宽屏16：10
机器规格	产品尺寸（mm）	296.9mm × 208.5mm × 15.95mm
	产品净重（kg）	1.28kg
内存	插槽数量	其他
	内存频率	2666
	内存类型	DDR4
	最大支持容量	16GB
音效系统	内置麦克风	内置麦克风

再看下SKU，大家觉得应该有什么字段？

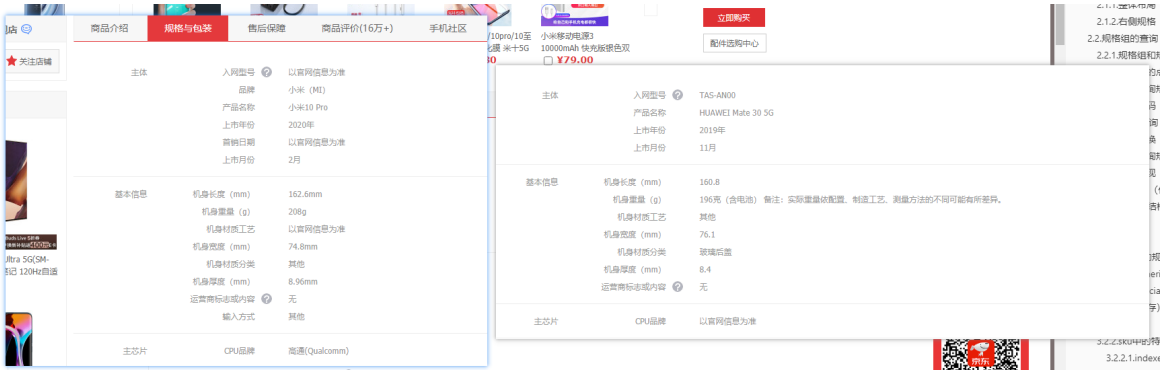
```
id：主键
spu_id：关联的spu
price：价格
images：图片
stock：库存
颜色？
内存？
硬盘？
```

碰到难题了，不同的商品分类，可能属性是不一样的，比如手机有内存，衣服有尺码，我们是全品类的电商网站，这些不同的商品的不同属性，如何设计到一张表中？

2.2.2 分析规格参数

仔细查看每一种商品的规格你会发现：

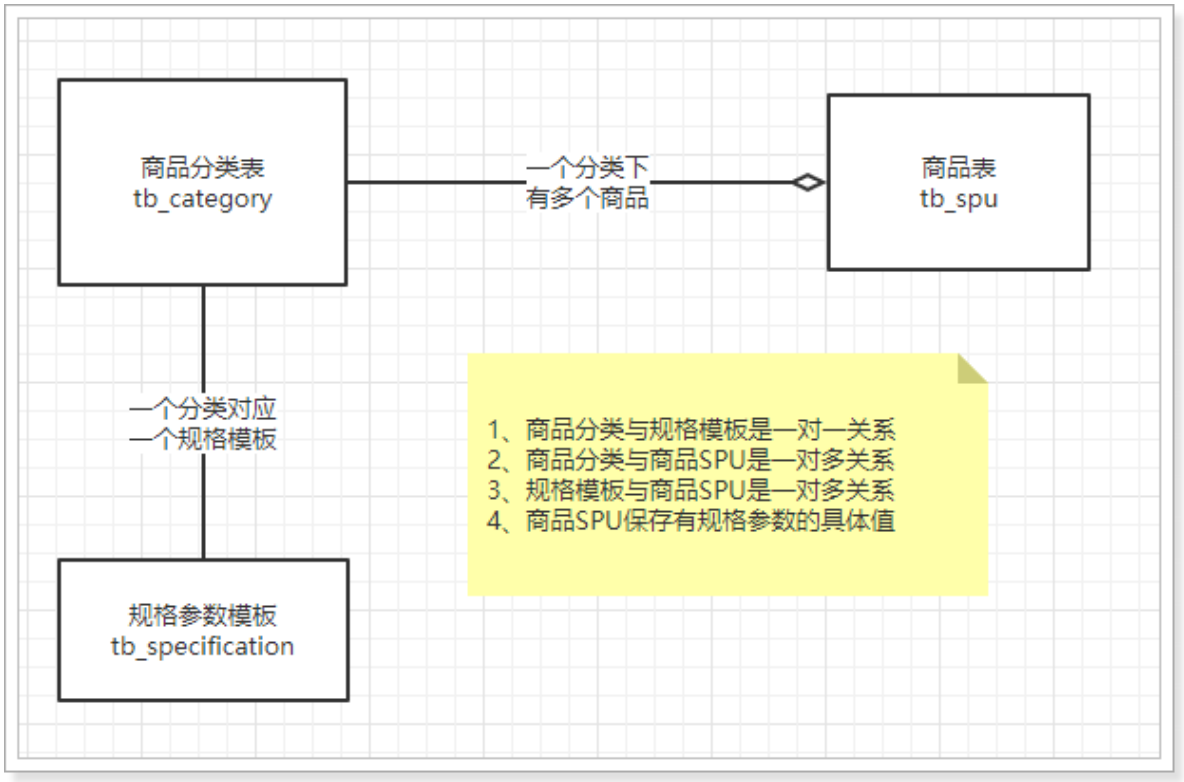
虽然商品规格千变万化，但是同一类商品（如手机）的规格是统一的，有图为证：



mate 30 主体跟小米不一样有可能是有写字段没有填,没有值的在页面就不展示了而已.不用纠结这个....

也就是说，商品的规格参数应该是与分类绑定的。**每一个分类都有统一的规格参数模板，但不同商品其参数值可能不同。**

如下图所示：



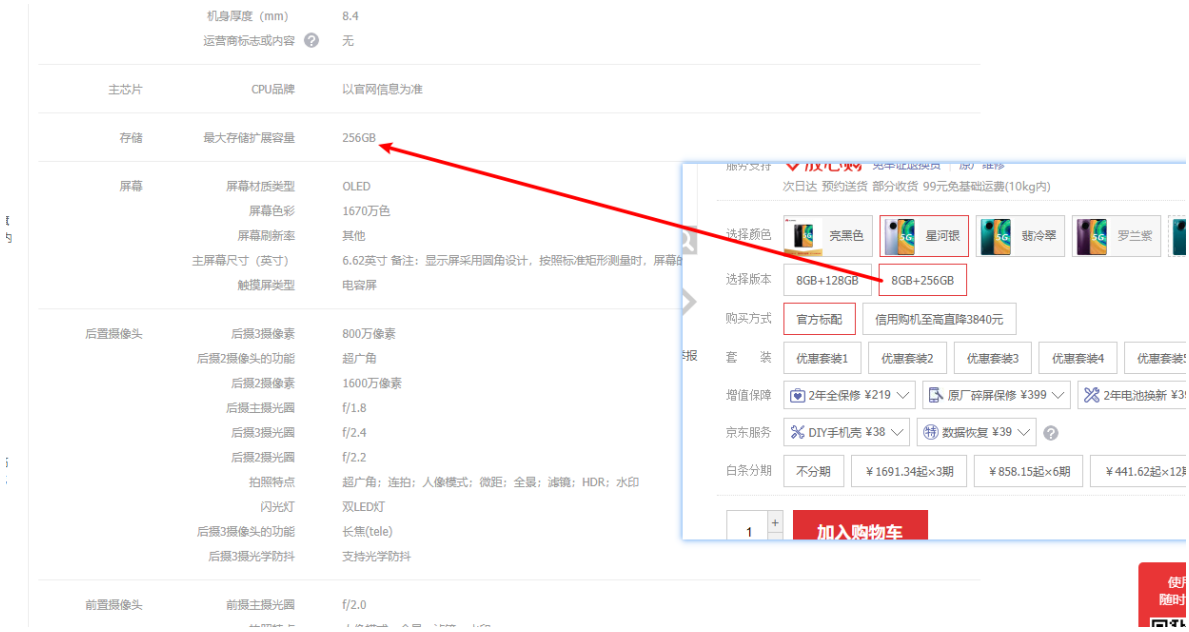
2.2.3 SKU的特有属性

SPU中会有一些特殊属性，用来区分不同的SKU，我们称为SKU特有属性。如华为mate30 的颜色、内存属性。

不同种类的商品，一个手机，一个衣服，其SKU属性不相同。

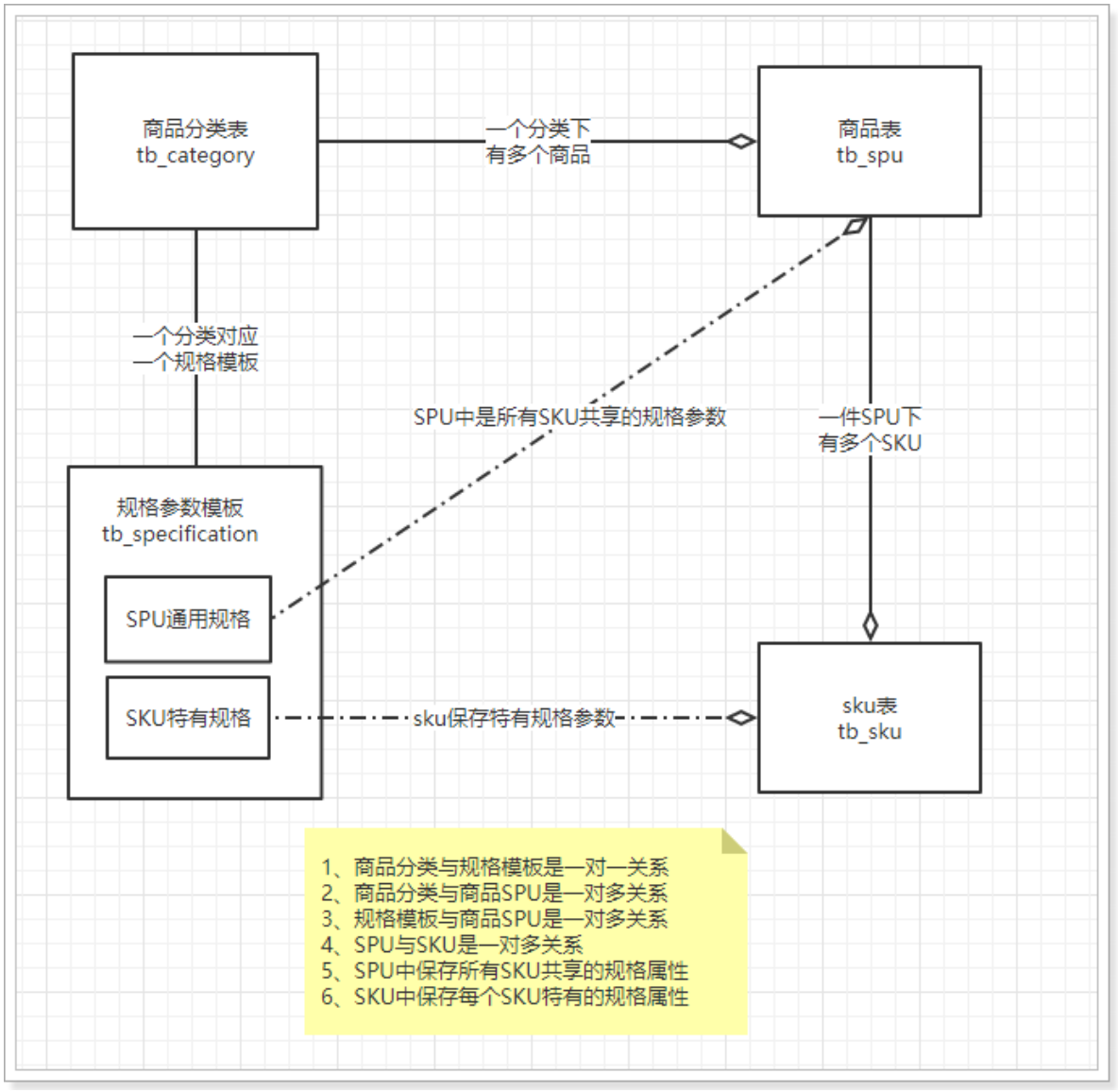
同一种类的商品，比如都是衣服，SKU属性基本是一样的，都是颜色、尺码等。

这样说起来，似乎SKU的特有属性也是与分类相关的？事实上，仔细观察你会发现，**SKU的特有属性是商品规格参数的一部分**：



也就是说，我们没必要单独对SKU的特有属性进行设计，它可以看做是规格参数中的一部分。这样规格参数中的属性可以标记成两部分：

- 所有sku共享的规格属性（称为全局属性/通用）
- 每个sku不同的规格属性（称为特有属性）



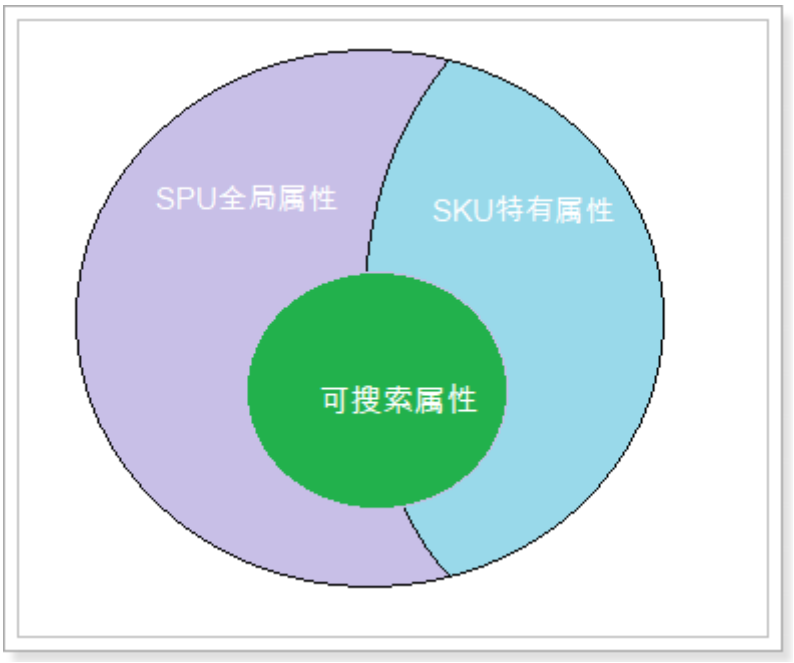
2.2.4 搜索属性

打开一个搜索页，我们来看看过滤的条件：

你会发现，过滤条件中的屏幕尺寸、运行内存、网路、机身内存、电池容量、CPU核数等，在规格参数中都能找到：

后置摄像头	后摄3摄像素	800万像素
	后摄2摄像头的功能	超广角
	后摄2摄像素	1600万像素
	后摄主摄光圈	f/1.8
	后摄3摄光圈	f/2.4
	后摄2摄光圈	f/2.2
	拍照特点	超广角；连拍；人像模式；微距；全景；滤镜；HDR；水印
	闪光灯	双LED灯
	后摄3摄像头的功能	长焦(tele)
	后摄3摄光学防抖	支持光学防抖
前置摄像头	前摄主摄光圈	f/2.0
	拍照特点	人像模式；全景；滤镜；水印
电池信息	电池是否可拆卸	电池不可拆卸
	充电器	10V/4A；5V/2A；9V/2A
	无线充电	支持无线充电；支持反向无线充电
网络支持	副SIM卡4G网络	4G TD-LTE(移动)；4G FDD-LTE(联通、电信)
	最大支持SIM卡数量	2个
	网络频率（2G/3G）	以官网信息为准
	5G网络	移动5G；联通5G；电信5G
	4G网络	4G FDD-LTE(联通)；4G TD-LTE(移动)；4G FDD-LTE(联通、电信)
	双卡机类型	双卡双待
	高清语音通话（VOLTE）	移动VOLTE；联通VOLTE；电信VOLTE
	3G/2G网络	3G WCDMA(联通)；3G CDMA2000(电信)；2G GSM(移动/联通)；2G CDMA(电信)
	SIM卡类型	Nano SIM
	网络模式(5G)	独立组网；非独立组网
	副SIM卡3G/2G网络	3G WCDMA(联通)；3G CDMA2000(电信)；2G GSM(移动/联通)；2G CDMA(电信)

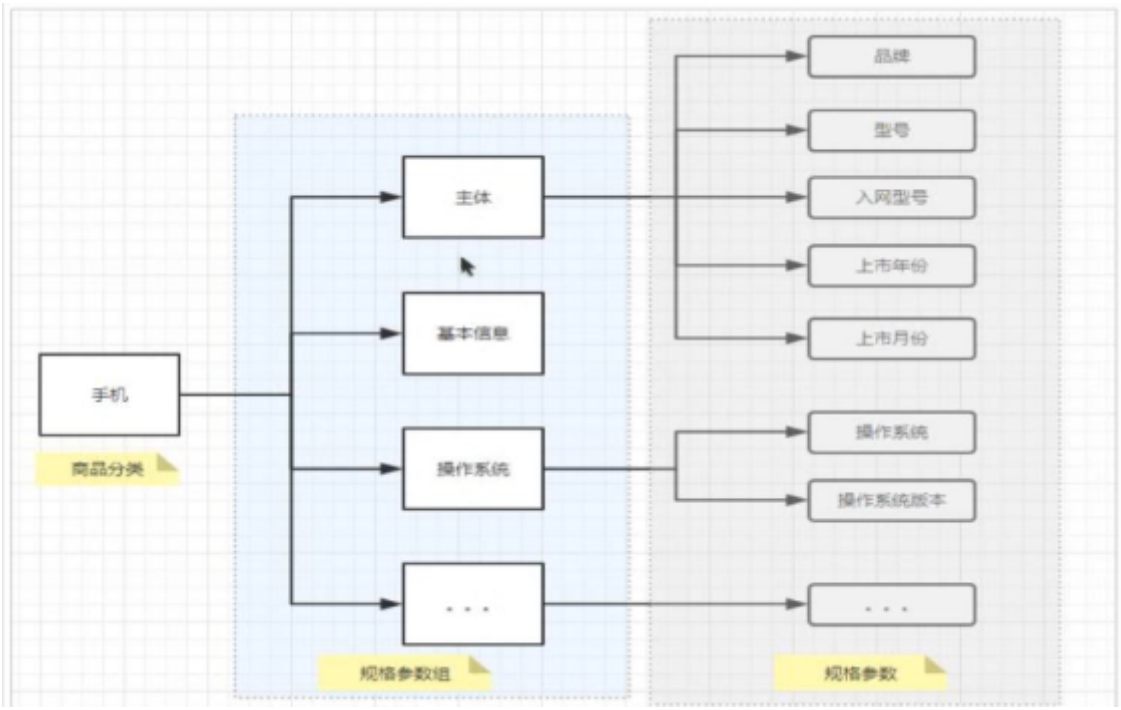
也就是说，规格参数中的数据，将来会有一部分作为搜索条件来使用。我们可以在设计时，将这部分属性标记出来，将来做搜索的时候，作为过滤条件。要注意的是，无论是SPU的全局属性，还是SKU的特有属性，都有可能作为搜索过滤条件的，并不冲突，而是有一个交集：



2.3 规格参数表

2.3.1 整体数据结构

观察页面的规格参数结构：



可以看到规格参数是分组的，每一组都有多个参数键值对。不过对于规格参数的模板而言，其值现在是不确定的，不同的商品值肯定不同，模板中只要保存组信息、组内参数信息即可。

因此我们设计了两张表：

- tb_spec_group：组，与商品分类关联
- tb_spec_param：参数名，与组关联，一对多

2.3.2 规格组

规格参数分组表：tb_spec_group

```
CREATE TABLE `tb_spec_group`  (
  `id` bigint(0) NOT NULL AUTO_INCREMENT COMMENT '主键',
  `cid` bigint(0) NOT NULL COMMENT '商品分类id，一个分类下有多个规格组',
  `name` varchar(50) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL COMMENT
'规格组的名称',
  PRIMARY KEY (`id`) USING BTREE,
  INDEX `key_category`(`cid`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 15 CHARACTER SET = utf8 COLLATE =
utf8_general_ci COMMENT = '规格参数的分组表，每个商品分类下有多个规格参数组' ROW_FORMAT
= Dynamic;
```

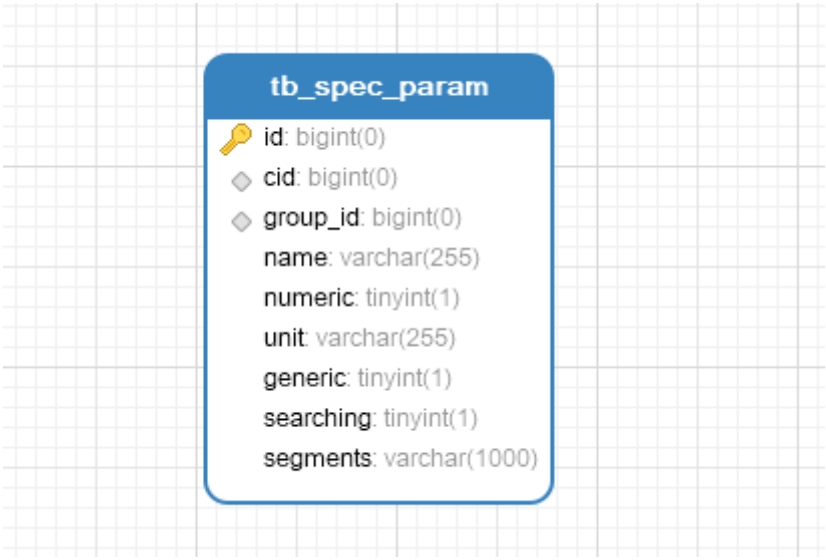
规格组有3个字段：

- id：主键
- cid：商品分类id，一个分类下有多个模板
- name：该规格组的名称。

2.3.3 规格参数

规格参数表：tb_spec_param

```
CREATE TABLE `tb_spec_param`  (
  `id` bigint(0) NOT NULL AUTO_INCREMENT COMMENT '主键',
  `cid` bigint(0) NOT NULL COMMENT '商品分类id', -- 反三范式,提升查询效率(少关联)
  `group_id` bigint(0) NOT NULL,
  `name` varchar(255) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL
COMMENT '参数名',
  `numeric` tinyint(1) NOT NULL COMMENT '是否是数字类型参数，true或false',
  `unit` varchar(255) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT ''
COMMENT '数字类型参数的单位，非数字类型可以为空',
  `generic` tinyint(1) NOT NULL COMMENT '是否是sku通用属性，true或false',
  `searching` tinyint(1) NOT NULL COMMENT '是否用于搜索过滤，true或false',
  `segments` varchar(1000) CHARACTER SET utf8 COLLATE utf8_general_ci NULL
DEFAULT '' COMMENT '数值类型参数，如果需要搜索，则添加分段间隔值，如CPU频率间隔：0.5-1.0',
  PRIMARY KEY (`id`) USING BTREE,
  INDEX `key_group`(`group_id`) USING BTREE,
  INDEX `key_category`(`cid`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 24 CHARACTER SET = utf8 COLLATE =
utf8_general_ci COMMENT = '规格参数组下的参数名' ROW_FORMAT = Dynamic;
```



按道理来说，我们的规格参数就只需要记录参数名、组id、商品分类id即可。但是这里却多出了很多字段，为什么?

还记得我们之前的分析吧，规格参数中有一部分是 SKU的通用属性，一部分是SKU的特有属性，而且其中会有一些将来用作搜索过滤，这些信息都需要标记出来。

- 通用属性
  - 用一个布尔类型字段来标记是否为通用：
    - generic来标记是否为通用属性：
      - true：代表通用属性
      - false：代表sku特有属性
- 搜索过滤
  - 与搜索相关的有两个字段：
    - searching：标记是否用作过滤
      - true：用于过滤搜索
      - false：不用于过滤
    - segments：某些数值类型的参数，在搜索时需要按区间划分，这里提前确定好划分区间
      - 比如电池容量，0~2000mAh，2000mAh~3000mAh，3000mAh~4000mAh
- 数值类型
  - 某些规格参数可能为数值类型，这样的数据才需要划分区间，我们有两个字段来描述：
    - numberic：是否为数值类型
      - true：数值类型
      - false：不是数值类型
    - unit：参数的单位

3 商品规格参数管理

3.1 加载分类信息

3.1.1 vue项目

3.1.1.1 index.js line:29

```
route("/item/specification",'/item/specification/Specification',"Specification")
,
```

3.1.1.2 specification/Specification.vue

```
<!--修改成我们自己的url-->
 <v-tree url="/category/list" :isEdit="false" @handleClick="handleClick" />
```

3.2 规格组(查询)

3.2.1 vue项目

3.2.1.1 SpecGroup.vue

line：72

```
    loadData(){
        this.$http.get("/specgroup/getSpecGroupInfo",{
            params:{
                cid:this.cid
            }
        })
        .then((resp) => {
            this.groups = resp.data.data;
        })
        .catch(() => {
            this.groups = [];
        })
    },
```

3.2.2 mingrui-shop-service-api-xxx

3.2.2.1 entity包下新建SpecGroupEntity

```
import lombok.Data;

import javax.persistence.Id;
import javax.persistence.Table;

/**
 * @ClassName SepcGroupEntity
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/23
 * @Version V1.0
 **/
@Table(name = "tb_spec_group")
@Data
public class SpecGroupEntity {

    @Id
    private Integer id;

    private Integer cid;

    private String name;
}
```

3.2.2.2 dto包下新建SpecGroupDTO

```
import com.baidu.shop.base.BaseDTO;
import com.baidu.shop.validate.group.MingruiOperation;
import io.swagger.annotations.ApiModel;
import io.swagger.annotations.ApiModelProperty;
import lombok.Data;

import javax.validation.constraints.NotEmpty;
import javax.validation.constraints.NotNull;

/**
 * @ClassName SepcGroupDTO
```

```
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/23
 * @Version V1.0
 **/
@ApiModel(value = "规格组数据传输DTO")
@Data
public class SpecGroupDTO extends BaseDTO {

    @ApiModelProperty(value = "主键", example = "1")
    @NotNull(message = "主键不能为空", groups = {MingruiOperation.Update.class})
    private Integer id;

    @ApiModelProperty(value = "类型id", example = "1")
    @NotNull(message = "类型id不能为空", groups = {MingruiOperation.Add.class})
    private Integer cid;

    @ApiModelProperty(value = "规格组名称")
    @NotEmpty(message = "规格组名称不能为空", groups =
{MingruiOperation.Add.class})
    private String name;
}
```

3.2.2.3 service包下新建SpecificationService

```
import com.baidu.shop.base.Result;
import com.baidu.shop.dto.SpecGroupDTO;
import com.baidu.shop.entity.SpecGroupEntity;
import io.swagger.annotations.Api;
import io.swagger.annotations.ApiOperation;
import org.springframework.web.bind.annotation.*;

/**
 * @ClassName SpecificationServiece
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/23
 * @Version V1.0
 **/
@Api(tags = "规格接口")
public interface SpecificationService {

    @ApiOperation(value = "通过条件查询规格组")
    @GetMapping(value = "specgroup/getSpecGroupInfo")
    Result<List<SpecGroupEntity>> getSepcGroupInfo(SpecGroupDTO specGroupDTO);

}
```

3.2.3 mingrui-shop-service-xxx

3.2.3.1 mapper包下新建SpecGroupMapper

```
import com.baidu.shop.entity.SpecGroupEntity;
import tk.mybatis.mapper.common.Mapper;

/**
 * @ClassName SpecGroupMapper
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/23
 * @Version V1.0
 **/
public interface SpecGroupMapper extends Mapper<SpecGroupEntity> {
}
```

3.2.3.2 service.impl包下新建SpecificationServiceImpl

```
import com.baidu.shop.base.BaseApiService;
import com.baidu.shop.base.Result;
import com.baidu.shop.dto.SpecGroupDTO;
import com.baidu.shop.entity.SpecGroupEntity;
import com.baidu.shop.mapper.SpecGroupMapper;
import com.baidu.shop.service.SpecificationService;
import com.baidu.shop.utils.ObjectUtil;
import org.springframework.util.StringUtils;
import org.springframework.web.bind.annotation.RestController;
import tk.mybatis.mapper.entity.Example;
import javax.annotation.Resource;
import java.util.List;

/**
 * @ClassName SpecificationServiceImpl
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/23
 * @Version V1.0
 **/
@RestController
public class SpecificationServiceImpl extends BaseApiService implements
SpecificationService {

    @Resource
    private SpecGroupMapper specGroupMapper;

    @Override
    public Result<List<SpecGroupEntity>> getSepcGroupInfo(SpecGroupDTO
specGroupDTO) {

        //通过分类id查询数据
        Example example = new Example(SpecGroupEntity.class);

        if(ObjectUtil.isNotNull(specGroupDTO.getCid()))
example.createCriteria().andEqualTo("cid",specGroupDTO.getCid());

        List<SpecGroupEntity> list = specGroupMapper.selectByExample(example);
        return this.setResultSuccess(list);
    }
```

```
}
```

3.3 规格组(新增,删除,修改)

3.3.1 vue项目

3.3.1.1 SpecGroup.vue

```
save(){
     this.$http({
      method: this.isEdit ? 'put' : 'post',
      url: 'specgroup/save',
      data: this.group
    }).then((resp) => {
      // 关闭窗口
      if(resp.data.code == 200){
          this.show = false;
          this.$message.success("保存成功! ");
          this.loadData();
      }else{
          this.$message.error(resp.data.msg);
      }

    }).catch(() => {
        this.$message.error("保存失败! ");
      });
},
deleteGroup(id){
    this.$message.confirm("确认要删除分组吗? ")
    .then(() => {
      this.$http.delete("/specgroup/delete?id=" + id)
          .then((resp) => {
              if(resp.data.code == 200){
                  this.$message.success("删除成功");
                  this.loadData();
              }else{
                  this.$message.error(resp.data.msg);
              }

          })
    })
},
```

3.3.2 mingrui-shop-service-api-xxx

3.3.2.1 SpecificationService

```
    @ApiOperation(value = "新增规格组")
    @PostMapping(value = "specgroup/save")
    Result<JSONObject> addGroupInfo(@Validated({MingruiOperation.Add.class})
@RequestBody SpecGroupDTO specGroupDTO);

    @ApiOperation(value = "修改规格组")
    @PutMapping(value = "specgroup/save")
    Result<JSONObject> editGroupInfo(@Validated({MingruiOperation.Update.class})
@RequestBody SpecGroupDTO specGroupDTO);

    @ApiOperation(value = "删除规格组")
    @DeleteMapping(value = "specgroup/delete")
    Result<JSONObject> deleteGroupInfo(Integer id);
```

3.3.3 mingrui-shop-service-xxx

3.3.3.1 SpecificationServiceImpl

```
    @Transactional
    @Override
    public Result<JSONObject> addGroupInfo(SpecGroupDTO specGroupDTO) {


 specGroupMapper.insertSelective(BaiduBeanUtil.copyProperties(specGroupDTO,SpecG
roupEntity.class));

        return this.setResultSuccess();
    }

    @Transactional
    @Override
    public Result<JSONObject> editGroupInfo(SpecGroupDTO specGroupDTO) {


 specGroupMapper.updateByPrimaryKeySelective(BaiduBeanUtil.copyProperties(specGr
oupDTO,SpecGroupEntity.class));
        return this.setResultSuccess();
    }

    @Transactional
    @Override
    public Result<JSONObject> deleteGroupInfo(Integer id) {

        specGroupMapper.deleteByPrimaryKey(id);

        return this.setResultSuccess();
    }
```

3.4 规格参数(查询)

3.4.1 vue项目

3.4.1.1 SpecParam.vue

line:125

```
   loadData() {
      this.$http
        .get("specparam/getSpecParamInfo",{
          params:{
            groupId:this.group.id
          }
        })
        .then(( resp ) => {
          resp.data.data.forEach(p => {
              p.segments = p.segments ? p.segments.split(",").map(s =>
s.split("-")) : [];
          })
          this.params = resp.data.data;
        })
        .catch(() => {
          this.params = [];
        });
    },
```

3.4.2 mingrui-shop-service-api-xxx

3.4.2.1 entity包下新建SpecParamEntity

```
import lombok.Data;

import javax.persistence.Column;
import javax.persistence.Id;
import javax.persistence.Table;

/**
 * @ClassName SpecParamEntity
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/23
 * @Version V1.0
 **/
@Table(name = "tb_spec_param")
@Data
public class SpecParamEntity {

    @Id
    private Integer id;

    private Integer cid;

    private Integer groupId;

    private String name;

    //numeric是mysql数据库的关键字,
    //SELECT  id,cid,group_id,name,numeric,unit,generic,searching,segments  FROM
tb_spec_param  WHERE (       (  group_id = 1 ) )
    //这句sql不会执行错误
    //加上``会当成普通字符串处理
    @Column(name = "`numeric`")
    private Integer numeric;
```

```
    private String unit;

    private Integer generic;

    private Integer searching;

    private String segments;
}
```

3.4.2.2 dto包下新建SpecParamDTO

```
import com.baidu.shop.base.BaseDTO;
import com.baidu.shop.validate.group.MingruiOperation;
import io.swagger.annotations.ApiModel;
import io.swagger.annotations.ApiModelProperty;
import lombok.Data;

import javax.validation.constraints.NotNull;

/**
 * @ClassName SpecParamDTO
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/23
 * @Version V1.0
 **/
@ApiModel(value = "规格参数数据传输DTO")
@Data
public class SpecParamDTO extends BaseDTO {

    @ApiModelProperty(value = "主键", example = "1")
    @NotNull(message = "主键不能为空", groups = {MingruiOperation.Update.class})
    private Integer id;

    @ApiModelProperty(value = "分类id", example = "1")
    private Integer cid;

    @ApiModelProperty(value = "规格组id", example = "1")
    private Integer groupId;

    @ApiModelProperty(value = "规格参数名称")
    private String name;

    @ApiModelProperty(value = "是否是数字类型参数，1->true或0->false", example =
"0")
    @NotNull(message = "是否是数字类型参数不能为空",groups =
{MingruiOperation.Add.class,MingruiOperation.Update.class})
    private Integer numeric;

    @ApiModelProperty(value = "数字类型参数的单位，非数字类型可以为空")
    private String unit;

    @ApiModelProperty(value = "是否是sku通用属性，1->true或0->false", example =
"0")
    @NotNull(message = "是否是sku通用属性不能为空",groups =
{MingruiOperation.Add.class,MingruiOperation.Update.class})
```

```
    private Integer generic;

    @ApiModelProperty(value = "是否用于搜索过滤，1->true或0->false", example = "0")
    @NotNull(message = "是否用于搜索过滤不能为空",groups =
{MingruiOperation.Add.class,MingruiOperation.Update.class})
    private Integer searching;

    @ApiModelProperty(value = "数值类型参数，如果需要搜索，则添加分段间隔值，如CPU频率间
隔：0.5-1.0")
    private String segments;
}
```

3.4.2.3 SpecificationService

```
    @ApiOperation(value = "查询规格参数")
    @GetMapping(value = "specparam/getSpecParamInfo")
    public Result<List<SpecParamEntity>> getSpecParamInfo(SpecParamDTO
specParamDTO);
```

3.4.3 mingrui-shop-service-xxx

3.4.3.1 mapper包下新建SpecParamMapper

```
import com.baidu.shop.entity.SpecParamEntity;
import tk.mybatis.mapper.common.Mapper;

/**
 * @ClassName SpecParamMapper
 * @Description: TODO
 * @Author shenyaqi
 * @Date 2020/8/23
 * @Version V1.0
 **/
public interface SpecParamMapper extends Mapper<SpecParamEntity> {
}
```

3.4.3.2 SpecificationServiceImpl

```
    @Resource
    private SpecParamMapper specParamMapper;

   @Override
    public Result<List<SpecParamEntity>> getSpecParamInfo(SpecParamDTO
specParamDTO) {

        List<SpecParamEntity> list = null;
        if (ObjectUtil.isNotNull(specParamDTO)) {

            Example example = new Example(SpecParamEntity.class);

            Example.Criteria criteria = example.createCriteria();

            //此处省略了一些代码,备课实在不想写了,只写了关键的一步
            if (ObjectUtil.isNotNull(specParamDTO.getGroupId())) {
```

```
                criteria.andEqualTo("groupId", specParamDTO.getGroupId());
            }

            list = specParamMapper.selectByExample(example);
        }

        return this.setResultSuccess(list);
    }
```

3.5 规格参数(增,删,改)

3.5.1 vue项目

3.5.1.1 SpecParam.vue

```
  methods: {
    loadData() {
      this.$http
        .get("specgroup/getSpecParamInfo",{
          params:{
            groupId:this.group.id
          }
        })
        .then((resp) => {
          resp.data.data.forEach(p => {
              p.segments = p.segments ? p.segments.split(",").map(s =>
s.split("-")) : [];
          })
          this.params = resp.data.data;
        })
        .catch(() => {
          this.params = [];
        });
    },
    editParam(param) {
        this.param = param;
        this.isEdit = true;
        this.show = true;
    },
    addParam() {
      this.param = {
          cid: this.group.cid,
          groupId: this.group.id,
          segments:[],
          numeric:false,
          searching:false,
          generic:false}
      this.show = true;
    },
    deleteParam(id) {
        this.$message.confirm("确认要删除该参数吗？")
        .then(() => {
            this.$http.delete("specparam/del?id=" + id)
            .then(() => {
                this.$message.success("删除成功");
```

```
                this.loadData();
            })
            .catch(() => {
                this.$message.error("删除失败");
            })
        })
    },
    formatBoolean(boo) {
      return boo ? "是" : "否";
    },
    save(){
        const p = {};
        Object.assign(p, this.param);
        p.segments = p.segments.map(s => s.join("-")).join(",")
        this.$http({
            method: this.isEdit ? 'put' : 'post',
            url: '/specparam/save',
            data: p,
        }).then(() => {
            // 关闭窗口
            this.show = false;
            this.$message.success("保存成功! ");
            this.loadData();
          }).catch(() => {
              this.$message.error("保存失败! ");
            });
    }
  }
```

3.5.2 mingrui-shop-service-api-xxx

3.5.2.1 SpecificationService

```
    @ApiOperation(value = "新增规格参数")
    @PostMapping(value = "specparam/save")
    Result<JSONObject> addParam(@Validated({MingruiOperation.Add.class})
@RequestBody SpecParamDTO specParamDTO);

    @ApiOperation(value = "修改规格参数")
    @PutMapping(value = "specparam/save")
    Result<JSONObject> editParam(@Validated({MingruiOperation.Update.class})
@RequestBody SpecParamDTO specParamDTO);


    @ApiOperation(value = "删除规格参数")
    @DeleteMapping(value = "specparam/del")
    Result<JSONObject> delParam(Integer id);
```

3.5.3 mingrui-shop-service-xxx

3.5.3.1 SpecificationServiceImpl

```
    @Transactional
    @Override
    public Result<JSONObject> addParam(SpecParamDTO specParamDTO) {
```

```
specParamMapper.insertSelective(BaiduBeanUtil.copyProperties(specParamDTO,SpecPa
ramEntity.class));

        return this.setResultSuccess();
    }

    @Transactional
    @Override
    public Result<JSONObject> editParam(SpecParamDTO specParamDTO) {


specParamMapper.updateByPrimaryKeySelective(BaiduBeanUtil.copyProperties(specPar
amDTO,SpecParamEntity.class));

        return this.setResultSuccess();
    }

    @Transactional
    @Override
    public Result<JSONObject> delParam(Integer id) {

        specParamMapper.deleteByPrimaryKey(id);

        return this.setResultSuccess();
    }
```