[1.1 公司/客户管理 9](#_Toc28701)

[1.2 SPU标准产品管理 14](#_Toc22407)

[1.4 库存管理 15](#_Toc9828)

[1.5 入库管理 16](#_Toc31943)

[1.6 出库管理 17](#_Toc25076)

[1.7 盘点管理 17](#_Toc16156)

[1.8 预约管理 18](#_Toc28808)

[1.9 维修单 19](#_Toc29900)

**接口文档**

文档地址：https://localhost:8089/api/doc/index.htm

#### 请求数据返回

{ "data": [ ], "code": 1000, "message": "成功", "url": null }

说明：code

/// <summary>

/// 成功

/// </summary>

[Description("成功")]

Success = 1000,

/// <summary>

/// 失败

/// </summary>

[Description("失败")]

Failure = 1001,

/// <summary>

/// 验证码过期

/// </summary>

[Description("验证码过期！")]

Login\_ValidaterCodeExpire = 1150,

/// <summary>

/// 验证码错误

/// </summary>

[Description("验证码错误！")]

Login\_ValidaterCodeError = 1151,

/// <summary>

/// 登录名或密码错误

/// </summary>

[Description("登录名或密码错误！")]

Login\_NameOrPwdError = 1152,

/// <summary>

/// 用户已被禁用

/// </summary>

[Description("该用户已被禁用！")]

Login\_UserDisabled = 1153,

/// <summary>

/// 检测到重复的请求

/// </summary>

[Description("检测到重复的请求！")]

Request\_Repeat = 1199,

/// <summary>

/// 长时间未操作，会话过期

/// </summary>

[Description("会话过期")]

Session\_Expire = 1100,

/// <summary>

/// 该用户被其他用户踢下线，请重新登录

/// </summary>

[Description("该用户被其他用户踢下线，请重新登录")]

Session\_BeKicked = 1101,

/// <summary>

/// 相同用户登录

/// </summary>

[Description("相同用户登录，是否踢掉？")]

Session\_SameUserLogin = 1102,

/// <summary>

/// 相同用户登录

/// </summary>

[Description("会话无效")]

Session\_Invalid = 1103,

/// <summary>

/// 无操作权限

/// </summary>

[Description("无操作权限")]

Session\_NotPermission = 1104,

/// <summary>

/// 请先登录

/// </summary>

[Description("请先登录")]

Please\_Login = 1105,

/// <summary>

/// 删除缓存错误

/// </summary>

[Description("删除缓存错误")]

Cache\_Error\_Del = 1200,

/// <summary>

/// 删除缓存错误

/// </summary>

[Description("重新加载缓存错误")]

Cache\_Error\_ReLoad = 1201,

#region 业务层编码

/// <summary>

/// 名称重复

/// </summary>

[Description("名称重复")]

Duplicate\_Name = 100001,

/// <summary>

/// 存在相同数据

/// </summary>

[Description("存在相同数据")]

SameData = 100002,

/// <summary>

/// 数据已更新

/// </summary>

[Description("数据已更新")]

Data\_Updated = 100003,

message:

错误信息；

data:

成功访问后的返回值。

**用户模块**

1. 注册

路径：/passport/register

请求方式：Post

请求数据：（qs.stringfy）

{

UserName: ‘’,

Pwd: ‘’

}

返回数据：

{

"code": 101 || 202, // 成功 || 失败 （后面的同理）

"message": "error message", // 成功，message为空字符串

//（后面的同理）

"extend": ""

}

1. 登录

路径：/passport/login

请求方式：post

请求数据：

{

UserName: “”,

Pwd: “”

}

返回数据：

{

“code”: 101 || 102 || 202,

“message”: “error message” || “no condition message”, // 当code === 102 时，条件不 // 足信息也在这里（后面的同理）

“extend”:

{

“id”: 1,

“UserName”: “”,

“Email”: “”,

“Phone”: “”,

“RoleList”: [

{

“id”: 1,

“RoleName”: “”,

“Description”: “”,

“AuthorityList”: [

{

“id”: 1,

“AuthorityName”: “”,

“Description”: “”,

“Icon”: “”, // 菜单图标

“Resource”: “”, // 资源路径

“ParentId”: 0, // 父节点id，一级菜单为0

“Level”: 1, // 菜单级别

}

]

}

],

“AuthorityList”: [ // 提取出来方便菜单处理

{

“id”: 1,

“AuthorityName”: “”,

“Description”: “”,

“Icon”: “”, // 菜单图标

“Resource”: “”, // 资源路径

“ParentId”: 0, // 父节点id，一级菜单为0

“Level”: 1, // 菜单级别, 从1级开始

}

]

}

}

1. 权限管理

3.1 添加权限

路径：/authority/add

请求方式：post

请求数据：

{

“AuthorityName”: “”,

“Description”: “”,

“Icon”: “”,

“Resource”: “”,

“ParentId”: 0,

“Level”: 1,

}

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”:

{

“id”: 1,

“AuthorityName”: “”,

“Description”: “”,

“Icon”: “”,

“Resource”: “”,

“ParentId”: 0,

“Level”: 1,

}

}

3.2 更新权限

路径：/authority/update

请求方式：post

请求数据：

{

“id”: 1,

“AuthorityName”: “”,

“Description”: “”,

“Icon”: “”,

“Resource”: “”,

“ParentId”: 0,

“Level”: 1,

}

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”: ""

}

3.3 删除权限

路径：/authority/delete

请求方式：delete

请求数据：

{

“id”: 1,

}

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”: ""

}

3.4 查询权限

查询所有

路径：/authority/getAll

请求方式：get

请求数据：

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”: [

{

“id”: 1,

“AuthorityName”: “”,

“Description”: “”,

“Icon”: “”,

“Resource”: “”,

“ParentId”: 0,

“Level”: 1,

}

]

}

条件查询

路径：/authority/search

请求方式：get

请求数据：

{

“SerachStr”: “”

}

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”: [

{

“id”: 1,

“AuthorityName”: “”,

“Description”: “”,

“Icon”: “”,

“Resource”: “”,

“ParentId”: 0,

“Level”: 1,

}

]

}

查询一级权限

路径：/authority/getLevel1

请求方式：get

请求数据：

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”: [

{

“id”: 1,

“AuthorityName”: “”,

“Description”: “”,

“Icon”: “”,

“Resource”: “”,

“ParentId”: 0,

“Level”: 1,

}

]

}

查询二级权限

路径：/authority/getLevel2

请求方式：get

请求数据：

{

“ParentId”: 1

}

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”: [

{

“id”: 1,

“AuthorityName”: “”,

“Description”: “”,

“Icon”: “”,

“Resource”: “”,

“ParentId”: 0,

“Level”: 2,

}

]

}

4 角色管理

4.1 添加角色

路径：/role/add

请求方式：post

请求数据：

{

“RoleName”: “”,

“Description”: “”,

“AuthorityListStr”: “1-2-3-4”, // 用 ‘-’ 拼接权限id 字符串

}

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”:

{

“id”: 1,

}

}

3.2 更新角色

路径：/role/update

请求方式：post

请求数据：

{

“id”: 1,

“RoleName”: “”,

“Description”: “”,

“AuthorityListStr”: “1-2-3-4”, // 用 ‘-’ 拼接权限id 字符串

}

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”: ""

}

3.3 删除权限

路径：/role/delete

请求方式：delete

请求数据：

{

“id”: 1,

}

返回数据

{

“code”: 101 || 202 || 303,

“message”: “”,

“extend”: ""

}

查询角色

## 公司/客户管理

3.1 客户添加

路径：/Client/AddClient

请求方式：post

请求数据：

{

"Company": "string",

"Contact": "string",

"Address": "string",

"Phone": "string",

"Email": "string",

"Type": "string",

"Description": "string"

}

返回数据：

{ "data": "3c0495a3b676483f91fd94f8a64270a8", // id

"code": 1000,

"message": "成功",

"url": null }

3.2 客户修改

路径：/Client/UpdateClient

请求方式：post

请求数据：

{

"id": "aa5027947997497bac57f5f2abef32e6",

"Company": "string",

"Contact": "string",

"Address": "string",

"Phone": "string",

"Email": "string",

"Type": "string",

"Description": "string"

}

返回数据：

{ "data": true, "code": 1000, "message": "成功", "url": null }

3.3 客户删除

路径：/Client/DeleteClient

请求方式：post

请求数据：

{

"id": "aa5027947997497bac57f5f2abef32e6",

"Company": "string", // 添加日志

}

返回数据：

{ "data": true, "code": 1000, "message": "成功", "url": null }

3.4 查询所有，不分页

路径：/Client/GetAll

请求方式：get

请求数据：

返回数据：

{ "data":

[

{ "id": "a52232aea81d4b37bed1006fccd76557",

"Company": "xiao",

"Contact": null,

"Address": "dsf",

"Phone": "dd",

"Email": "dd",

"Type": "dd",

"Description": "xiao"

},

{ },

{ }

],

"code": 1000,

"message": "成功",

"url": null

}

3.5查询所有，分页

路径：Client/GetClientListWithPaging?PageIndex=2&PageSize=1

请求方式：get

请求数据：

{

“PageIndex”: , // 页号

“PageSize”: , // 每页记录条数

}

返回数据：

{

"data": {

"TotalCount": 2,

"Items": [

{

"id": "c9b6cfd49d024ba583418f4dd41a059a",

"Company": "xx",

"Contact": null,

"Address": "dd",

"Phone": "fdf",

"Email": "ee",

"Type": "ee",

"Description": "dfs"

}

]

},

"code": 1000,

"message": "成功",

"url": null

}

3.5 条件查询, 分页

路径：/Client/GetClientListPageBySearch?SearchStr=x&PageIndex=1&PageSize=44

请求方式：get

请求数据：

{

“SearchStr”: “”,

“PageIndex”: , // 页号

“PageSize”: , // 每页记录条数

}

返回数据：

{

"data": {

"TotalCount": 2,

"Items": [

{

"id": "a52232aea81d4b37bed1006fccd76557",

"Company": "xiao",

"Contact": null,

"Address": "dsf",

"Phone": "dd",

"Email": "dd",

"Type": "dd",

"Description": "xiao"

}

]

},

"code": 1000,

"message": "成功",

"url": null

}

3.6 条件查询，不分页

路径：/Client/GetClientListBySearch?SearchStr=s

请求方式：GET

请求数据：

{

“SearchStr”: “”

}

响应体：

{

"data": [

{

"id": "aa5027947997497bac57f5f2abef32e6",

"Company": "string",

"Contact": null,

"Address": "string",

"Phone": "string",

"Email": "string",

"Type": "string",

"Description": "string"

}

],

"code": 1000,

"message": "成功",

"url": null

}

**分类管理**

1. 一级分类管理

添加分类

更新分类

删除分类

查询分类 -- 所有

批量添加

批量删除 -- 把二级分类也删除 -- 二级分类属性

1. 二级分类管理

添加分类

更新分类

删除分类

查询分类 -- 所有

批量添加

批量删除 -- 二级分类属性

**属性管理**

属性添加

属性删除

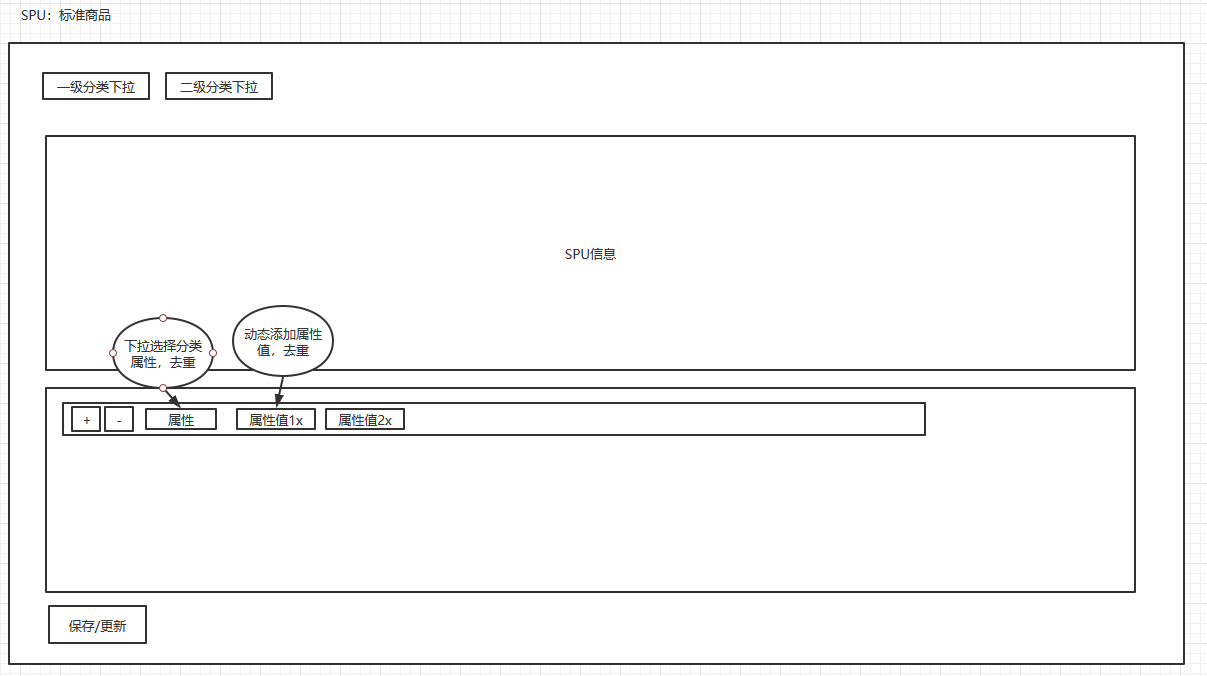
属性更新

查询分类属性 -- 所有

批量添加

批量删除

## SPU标准产品管理

****

1. 添加spu

添加属性 (不用提交id)

添加属性值 (不用提交id)

过程：判断是否有重复属性和属性值，属性/属性值是否已经存在

解决前端下拉过滤，后台判断

1. 删除spu

删除属性值 删除属性表 删除spu

1. 更新

先删再添加属性与属性值

提交是否重复，是否已经存在

1. 查询

条件查询 -- 分页 -- 名字查询，分类查询

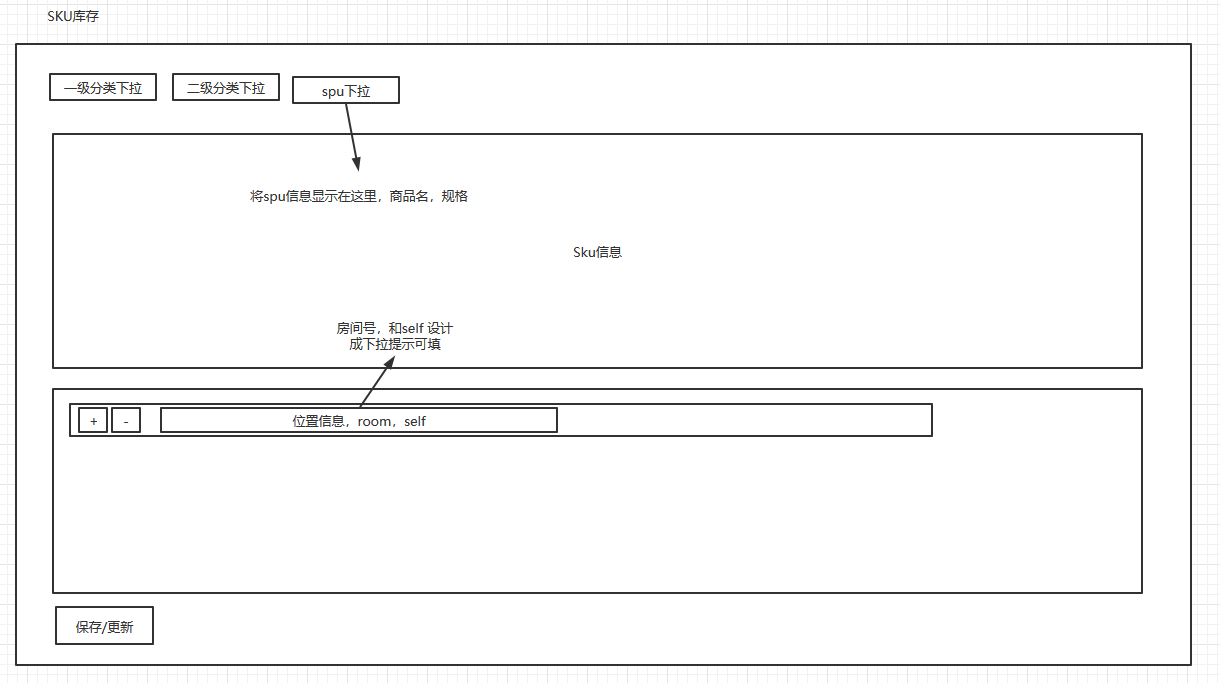
是否双条件查询 -- 分类限定，名字搜索，品牌（待定，暂定不要）

5. 是否可以查询所有 -- 分页

1. 是否可以批量删除 -- 待定

## 

## 库存管理

****

1. 添加库存

下拉选项：品牌（待定），分类，关联spu，属性值，选完后过滤 避免选择重复

单位（下拉提示，可选可填）

总数量，

\*\*\*\*生成库存编号规则：（待定）

可以有多个位置：数量，房间号，货架号

注意: totalCount--总数, SkuName(SpuName), 由前端提交过来

判断是否已经存在：品牌，关联spu，属性值

库存名也要提交

1. 删除库存 -- 绑定其他表删除注意-- 生成库存操作日志
2. 更新库存 -- 生成库存操作日志

属性值，位置，先删再添加

1. 查看库存

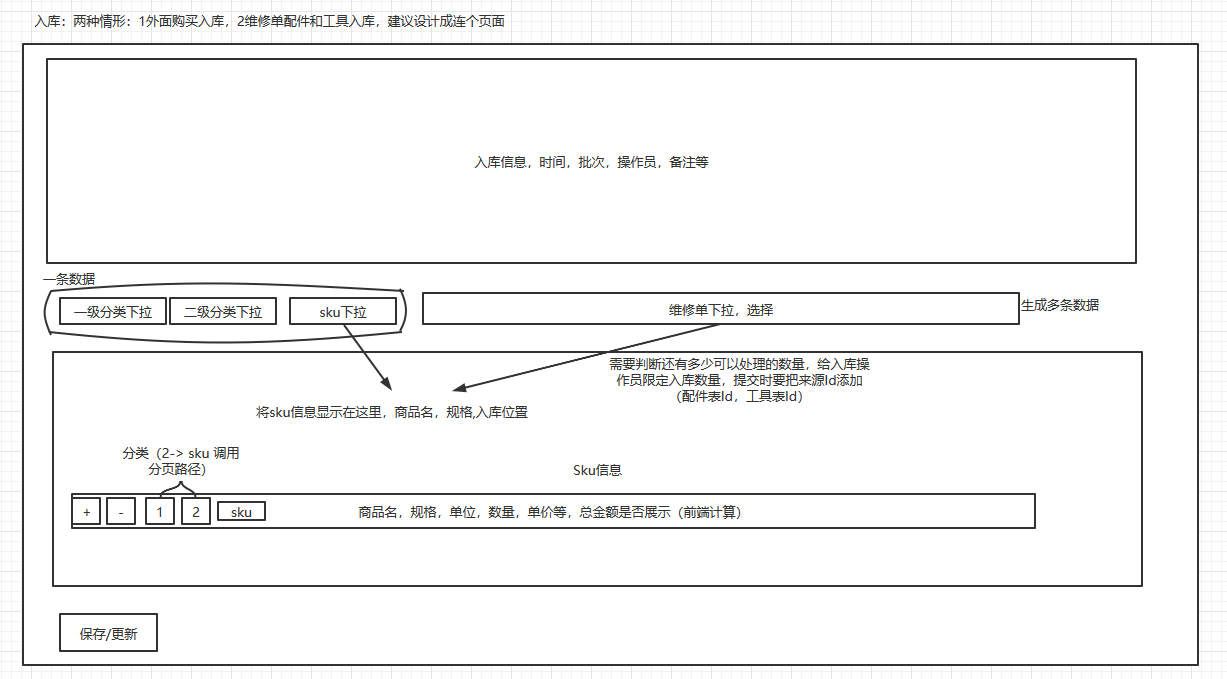
分页查询

查询所有

条件查询 --

1. 查看库存操作日志 -- 定时任务删除记录 -- 是否记录出入库记录

## 入库管理

****

1. 添加

必要字段：

操作员，入库日期，批次，供应商，库存List不为空（位置可以后面补上）

下拉：分类，spuId，属性（规格）， -- 多个

位置

1. 即时添加：可以添加多个 - 数量 单价 -- 暂时使用

2.添加分配表，可以后续分配

批次，供应商

入库单号生成策略：年+月+日+供应商（三位001）+批次（01）

1. 更新（备注入库单）

允许更新的字段：备注 -- 标注错误信息，Description，数据错误由库存管理员直接修改库存

1. 查询

范围查询

条件查询

分页查询

1. 删除

5. 批量删除

## 出库管理

1. 添加出库

必要字段 -- 出库日期，批次，库存list（包含位置 -- 前端可以选择显示）

下拉：分类， spu， sku，位置

出库单生成策略：年+月+日+客户（三位001）+批次（01） -- 待定

-- 判断库存是否足够数量

-- 出库后是否触发报警 --

1. 备注出库单

3. 查询

分页查询

查询所有

条件查询 --

1. 删除

5. 批量删除

## 盘点管理

1. 添加盘点单

下拉：分类，sku，位置 -- AddressId

必要参数： CheckSkuList， 盘点日期，操作员

生成盘点单号： 年月日+ 操作员

// 1. 查重

// 2. 判断该盘点单是否已经存在

// 3. 计算盘点总金额，盘点相差数据 -- 前端计算

添加盘点SMS\_CHECK

添加SMS\_CHECK\_SKU

1. 更新，备注，处理 -- 是否可以改为未处理，暂定不可以，只能改为处理完成

// 1. 修改每项的备注状态

// 2. 是否取消总状态

// 2. 是否修改总状态，一键标注解决 -- 另一路径

2.2 更新整个

删除SMS\_CHECK\_SKU

添加SMS\_CHECK\_SKU

更新SMS\_CHECK

1. 查询

条件查询 -- 操作员，盘点单号,库存是否需要（暂不需要）

日期范围查询 --

查询全部 -- 分页， 不分页

有差错，未处理查询 -- 分页

有差错，已处理查询 -- 分页

没有差错 -- 分页

有差错（不分是否解决） -- 分页

1. 删除

5. 批量删除

## 预约管理

1 预约添加

下拉：公司，车型，

必要字段：预约时间，(联系人，联系电话 -- 暂定可以不要 )，车牌号

生成预约编号：预约年月日+公司编号(三位后三位（不足补）)+车牌后两位+当天公司维修次序（三位） --- 查询数据库 -- 不用查重？？ -- 车牌号判断？？ -- 是否存在同一天同一辆车两次维修？？

-- 暂时采用不查重策略

2 预约修改

修改 -- 日期修改，预约编号是否重新生成 -- 不重新生成

不可修改项：预约编号

必要字段：Id，预约编号，预约时间，车牌号

3 预约删除

必要字段：Id

4 预约查询

获取所有 分页/不分

条件获取 分页

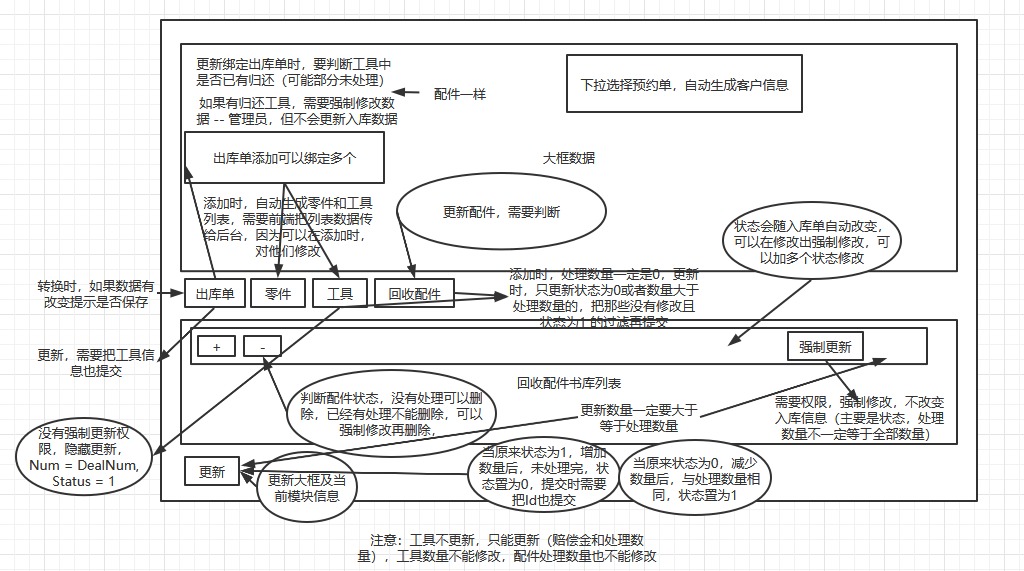
-- 预约时间范围，联系人，联系电话，车牌号，公司名

// 是否取消，0未处理，1已处理，2取消, -1 不开启处理查询 -- Status

5 批量删除

必要字段Id

## 维修单

****

是否可以取消 -- 暂不可以取消

直接在维修单修改配件/工具状态，不作入库处理

1 修维添加

（绑定出库表）工具表添加，旧配件添加（手动添加）

可以绑定多个出库表

必要字段：负责人，开始维修时间，

旧配件下拉选择，先选择分类，再选择库存 -- 规格，品牌

维修单号生成策略：= 预约单号 -- 前端提交

由于可能出现创建维修单时，就把工具配件等数据填写上，所以工具，配件由前端提交过来

2 删除

情景分析：

2.1 过时删除（操作完成）

2.2 添加错误删除 ？？？

解决：

回滚数据 -- 预约单状态回滚，工具，旧配件是否回滚操作

不能删除，提示，数据未完成 -- 暂时采用

// 查看是否已经签字/取消

// 查看是否工具/配件已经处理 -- 可以通过入库处理，也可以直接再维修单处修改状态

// 删除配件表 -- 不能删除，绑定入库信息

try

{

}catch(Exception e)

{

throw new MyServiceException("该配件已入库，需要绑定来源，请先将该库存和对应入库信息删除");

}

批量删除

3 更新

部分工具归还

1. 状态更新 - UpdateToolStatus

配件更新建议：只有出现配件删除添加才显示保存更新按钮，每条数据有更新按钮，只要配件已处理，不能再修改该配件（可以继续添加一样的）

情景分析：如果配件入库，是否可以修改/删除 -- 不能

如果存在部分入库，而想要继续添加配件，怎么解决

1. 单个添加
2. 批量添加 -- 采用
   1. 判断是否存在 添加

删除情形：

解决：前端把处理完的过滤，将没有处理的添加保存 -- 更新（另外操作，不兼大框修改）

后台，先删除状态未处理的配件，再添加新的数据

工具状态更新

（归还部分，再修改，造成数据错误，所以部分归还但没有处理完成的也要不能操作更新）

批量工具更新

旧配件状态更新

批量旧配件更新

2. 信息更新

更新整个（包括出库单，工具，配件）

判断是否已经有处理完成的工具/配件

更新维修单基本信息，但不更新关联信息

工具/旧配件入库（处理） -- 入库了是否还可以回滚 -- 采用不回滚，只要处理的数据，让其更新失败，或者管理员强制修改处理状态，在重新更新

4 查询

获取所有 -- 不分页

分页

是否存在未入库的旧配件，是否存在未解决的工具 -- 未归还/未赔款 -- 未标记解决

是否完成签字或取消

1. 获取还有未处理工具的维修单 -- ToolStatus 0 没全部处理完，1全部处理完，-1 不用理会
2. 获取还有未处理的旧配件的维修单 -- OldPartStatus 0 没全部处理完，1全部处理完，-1 不用理会
3. 获取未签字/取消的维修单 -- Status 0没有，1 签了，-1 不用理会

条件模糊查询：维修单号，员工，开始时间范围(只针对开始维修时间)，负责人，关联预约单： 车牌，公司，联系人，联系电话

工具：工具名，品牌，规格，数量，单位，单价，状态，备注，归还数量，赔偿金额

旧配件：配件名，品牌，规格，数量，单位，单价，状态，备注，处理数量

购买的配件：配件名，品牌，规格，数量，单位，单价，总价（前端看情况添加）

5 添加旧配件

问题：

入库更新，是否回滚旧配件和工具的处理状态和数量

现在采用 -- 不回滚 -- 数据需要管理员自己修改标记