

## 《数据库系统》课程设计报告

题目	图片销售管理系统		
小组成员信息			
姓名	学号	班级	分工
陈欣宇	21307347	人工智能与大数据	框架初始化, 编写代码报告
高宇	21307350	人工智能与大数据	ER 图初设计, 编写代码报告
陈华清	21307100	人工智能与大数据	数据初始化, 编写代码报告

提交时间: 2023 年 12 月 31 日

### 一. 开发环境与开发工具

-开发环境: windows

-开发工具: jdk1.8, Mysql8.0.31, Redis5.0.14.1, Maven3.6, Node16.19.1, IDEA2023.1, VS Code1.85.1

-客户端界面: 使用前后端分离的[若依框架](#)搭建 Web 界面

-数据库使用内网穿透(贝锐花生壳)映射一台主机端口进行协作使用

### 二. 系统需求分析(5 分)

此次课程设计实现以下功能:

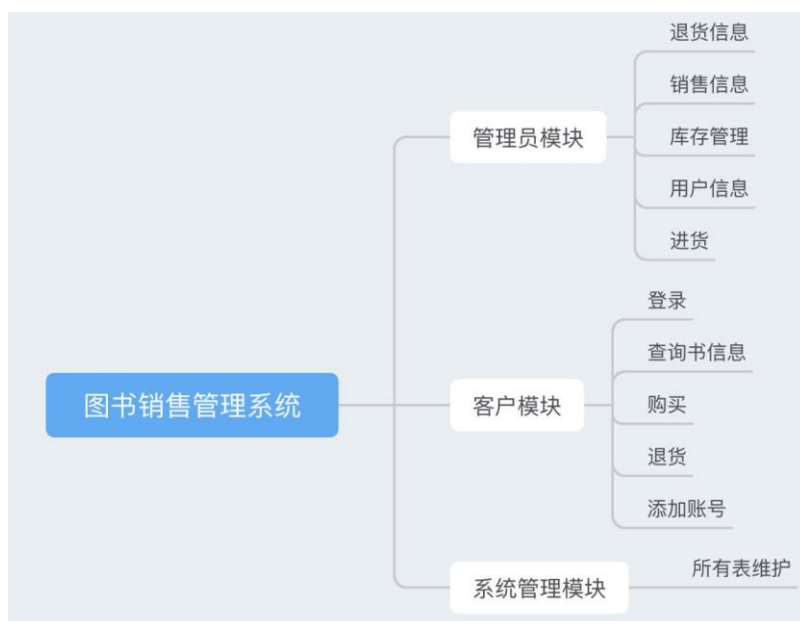
- 1) 进货:根据某种书籍的库存量及销售情况确定进货数量,根据供应商报价选择供应商。输出一份进货单并自动修改库存量,把本次进货的信息添加到进货库中。
- 2) 退货:顾客把已买的书籍退还给书店。输出一份退货单并自动修改库存量,把本次退货的信息添加到退货单中。
- 3) 销售:顾客搜索买书籍的信息,显示此书的库存量。得到销售记录并修改库存,同时把此次销售记录添加到销售表中。

系统数据字典: 根据以上需求,得到以下系统的数据结构

数据结构名	属性
图书 book	书号, 书名, 出版社
库存 stock	书号, 价格, 库存量
可进货表 sales	书号, 进货单号, 商家, 商家库存, 单价
进货单 supply	书号, 进货单号, 进货数量, 进货时间
销售记录 purchase	书号, 客户号, 销售数量, 是否退货, 销售时间
客户 consumer	客户号, 姓名, 电话, 地址, 性别

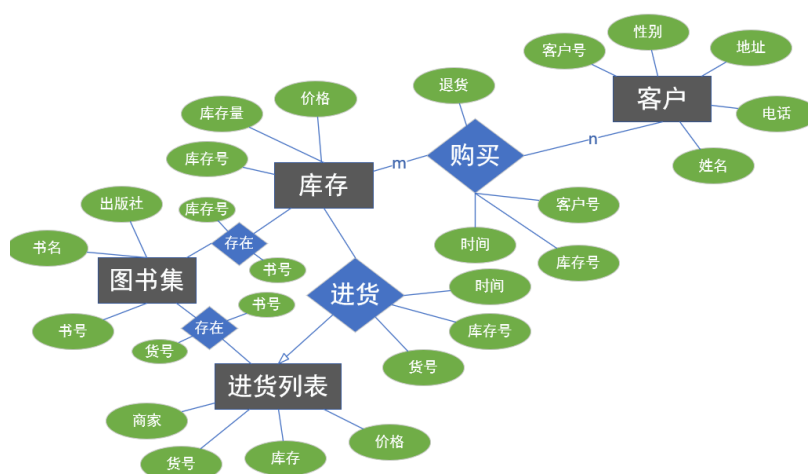
### 三. 功能需求分析 (10 分)

系统功能模块图



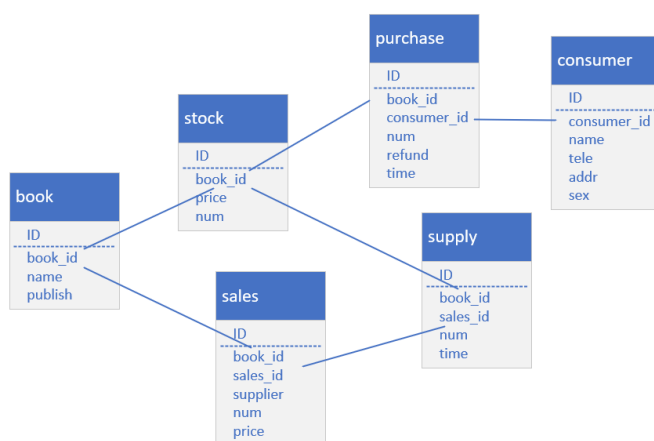
#### 四. 系统设计（25 分）

##### 数据概念结构设计（系统 ER 图）（10 分）



##### 数据库关系模式设计（10 分）

根据具体实现功能对 ER 图涉及字段进行简化，因为若依生成表要求 id 字段自动递增，故每个表中额外使用一个 id 字段



### 数据库物理结构设计（5 分）

book	ID	Book_id	Name	Publish
	Int	Char(5) (unique)	Varchar(45)	Varchar(45)
解释		书号	书名	出版社

Stock	ID	Book_id	Price	Num
	Int	Char(5) (unique)	Float	int
解释		书号	价格	库存量

Sales	ID	Book_id	Sales_id	Supplier	num	price
	Int	Char(5)	Char(5) (unique)	Char(30)	Int	float
解释		书号	单号	商家	商家库存	单价

Purchase	ID	Book_id	consumer_id	num	refund	time
	Int	Char(5)	Char(5)	Int	Char(5)	datetime
解释		书号	客户号	销售数量	是否退货	销售时间

Supply	ID	Book_id	Sales_id	Num	time
	Int	Char(5)	Char(5)	Int	datetime
解释		书号	单号	入货数量	入货时间

Consumer	ID	Consumer_id	Name	Tele	Addr	sex
	Int	Char(5) (unique)	char(20)	Char(20)	Varchar(45)	Char(5)
解释		客户号	姓名	电话	地址	性别

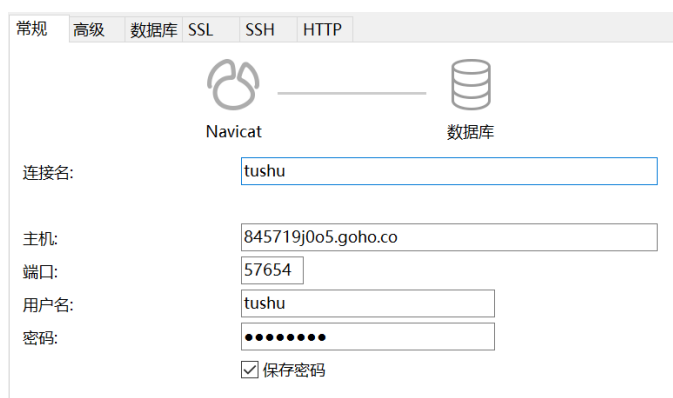
### 五. 系统功能的实现 (10 分)

主要功能模块的实现过程（简述）、运行界面

#### (1) 初始化框架

首先下载[若依框架](#)于本地，具体搭建运行过程不详细描述。

数据库：本地创建数据库 tushu，创建用户 tushu 供访问，在贝锐花生壳得到本地 3306 的映射网址 845719j0o5.goho.co 端口 57654，其他主机即可访问该数据库




The image shows the Navicat database connection configuration window. It includes tabs for '常规' (General), '高级' (Advanced), '数据库' (Database), 'SSL', 'SSH', and 'HTTP'. The '常规' tab is selected. The '连接名' (Connection Name) is 'tushu'. The '主机' (Host) is '845719j0o5.goho.co'. The '端口' (Port) is '57654'. The '用户名' (Username) is 'tushu'. The '密码' (Password) is masked with dots. There is a checkbox for '保存密码' (Save Password) which is checked. The interface also shows icons for 'Navicat' and '数据库' (Database).

向初始化框架的数据库中新建数据表



表 book:


名	类型	长度	小数点	不是 null	虚拟	键	注释
▶id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	编号
book_id	char	5		<input type="checkbox"/>	<input type="checkbox"/>		书号
name	varchar	45		<input type="checkbox"/>	<input type="checkbox"/>		书名
publish	varchar	45		<input type="checkbox"/>	<input type="checkbox"/>		出版社

<

默认:

☒ 自动递增

表 stock:


名	类型	长度	小数点	不是 null	虚拟	键	注释
▶id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	编号
book_id	char	5		<input type="checkbox"/>	<input type="checkbox"/>		书号
price	float			<input type="checkbox"/>	<input type="checkbox"/>		单价
num	int			<input type="checkbox"/>	<input type="checkbox"/>		库存量

<

默认:

☒ 自动递增

表 sales:


名	类型	长度	小数点	不是 null	虚拟	键	注释
▶id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	编号
book_id	char	5		<input type="checkbox"/>	<input type="checkbox"/>		书号
sales_id	char	5		<input type="checkbox"/>	<input type="checkbox"/>		单号
supplier	char	30		<input type="checkbox"/>	<input type="checkbox"/>		商家
num	int			<input type="checkbox"/>	<input type="checkbox"/>		库存
price	float			<input type="checkbox"/>	<input type="checkbox"/>		单价

<

默认:

☒ 自动递增

表 supply:


名	类型	长度	小数点	不是 null	虚拟	键	注释
▶id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	编号
book_id	char	5		<input type="checkbox"/>	<input type="checkbox"/>		书号
sales_id	char	5		<input type="checkbox"/>	<input type="checkbox"/>		单号
num	int			<input type="checkbox"/>	<input type="checkbox"/>		进货数量
time	datetime			<input type="checkbox"/>	<input type="checkbox"/>		进货时间

<

默认:

☒ 自动递增

表 purchase:


名	类型	长度	小数点	不是 null	虚拟	键	注释
▶id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	编号
book_id	char	5		<input type="checkbox"/>	<input type="checkbox"/>		书号
consumer_id	char	5		<input type="checkbox"/>	<input type="checkbox"/>		客户号
num	int			<input type="checkbox"/>	<input type="checkbox"/>		销售数量
refund	char	5		<input type="checkbox"/>	<input type="checkbox"/>		是否退货
time	datetime			<input type="checkbox"/>	<input type="checkbox"/>		销售时间

<

默认:

☒ 自动递增

表 consumer:

名	类型	长度	小数点	不是 null	虚拟	键	注释
▶id	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	编号
consumer_id	char	5		<input type="checkbox"/>	<input type="checkbox"/>		客户号
name	char	20		<input type="checkbox"/>	<input type="checkbox"/>		名字
tele	char	20		<input type="checkbox"/>	<input type="checkbox"/>		电话
addr	varchar	45		<input type="checkbox"/>	<input type="checkbox"/>		地址
sex	char	5		<input type="checkbox"/>	<input type="checkbox"/>		性别

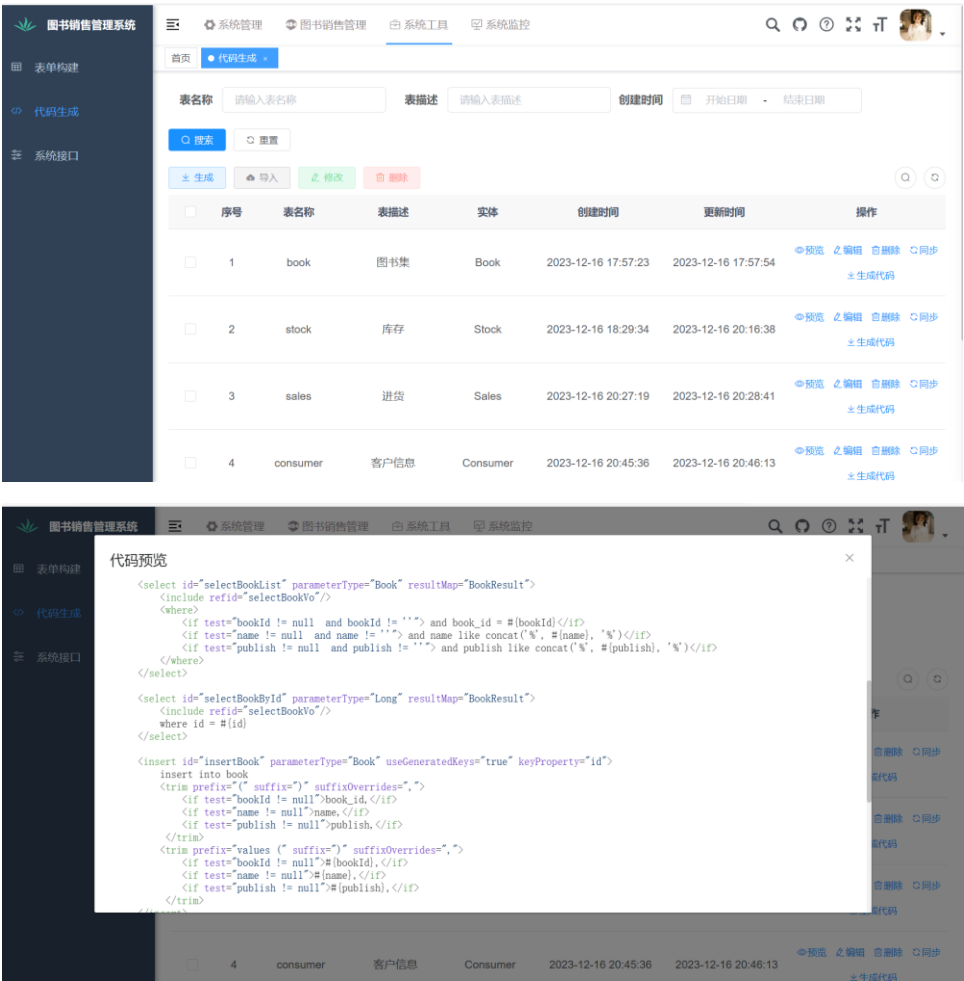
<

默认:

☒ 自动递增

(2) Web 界面生成表

使用代码生成功能将所有表的代码生成，如图将表导入->编辑->生成代码，将代码放入文件包中，



Web 界面生成表效果：



在生成表之后，通过修改对应表的 Mapper.xml 和 domain 文件，以及修改 vue 文件的排版，能够实现多表关联查询，如下图，库存表 stock 中能够根据“书号”显示“书名”和“出版社”的信息，在这也不展开描述具体细节。管理员**库存管理**或是客户**查询书本信息**的功能也顺带实现。

书号

请输入书号

书名

请输入书名

出版社

请输入出版社

单价

请输入单价

库存量

请输入库存量

🔍 搜索

🔄 重置

+ 新增

🔗 修改

🗑 删除

📤 导出

🔍

🔄

<input type="checkbox"/>	书号	书名	出版社	单价	库存量	操作
<input type="checkbox"/>	s1	解忧杂货店	南海出版公司	45	65	<a href="#">🔗 修改</a> <a href="#">🛒 购买</a> <a href="#">🗑 删除</a>
<input type="checkbox"/>	s2	巨人的陨落：世纪三部曲	江苏凤凰文艺出版社	39	82	<a href="#">🔗 修改</a> <a href="#">🛒 购买</a> <a href="#">🗑 删除</a>
<input type="checkbox"/>	s4	房思琪的初恋乐园	北京联合出版公司	50	79	<a href="#">🔗 修改</a> <a href="#">🛒 购买</a> <a href="#">🗑 删除</a>
<input type="checkbox"/>	s3	我的前半生	新世界出版社	40	21	<a href="#">🔗 修改</a> <a href="#">🛒 购买</a> <a href="#">🗑 删除</a>

共 4 条

10条/页

<

1

>

前往

1

页

### (3) 进货功能

在介绍如何实现之前首先说明本系统本身具备的功能。

- ① 角色管理：能够创建角色，本次设置了图书管理员和客户两个角色，用于权限判定以及界面区分

<input type="checkbox"/>	角色编号	角色名称	权限字符	状态	创建时间	操作
<input type="checkbox"/>	1	超级管理员	admin	<input checked="" type="checkbox"/>	2023-12-15 15:34:09	
<input type="checkbox"/>	2	普通角色	common	<input checked="" type="checkbox"/>	2023-12-15 15:34:09	<a href="#">🔗 修改</a> <a href="#">🗑 删除</a> <a href="#">» 更多</a>
<input type="checkbox"/>	100	图书管理员	libadmin	<input checked="" type="checkbox"/>	2023-12-17 11:18:01	<a href="#">🔗 修改</a> <a href="#">🗑 删除</a> <a href="#">» 更多</a>
<input type="checkbox"/>	101	客户	consumer	<input checked="" type="checkbox"/>	2023-12-17 11:19:33	<a href="#">🔗 修改</a> <a href="#">🗑 删除</a> <a href="#">» 更多</a>

- ② 用户管理：创建用户账号，账号能够指定角色，使得不同账号登录看到的界面不同，在同一界面上看到的按钮不同。

+ 新增

🔗 修改

🗑 删除

📶 导入

📶 导出

🔍

🔄

☰

<input type="checkbox"/>	用户号	用户昵称	状态	创建时间	操作
<input type="checkbox"/>	admin	若依	<div><div></div></div>	2023-12-15 15:34:09	
<input type="checkbox"/>	ry	若依	<div><div></div></div>	2023-12-15 15:34:09	<a href="#">🔗 修改</a> <a href="#">🗑 删除</a> <a href="#">» 更多</a>
<input type="checkbox"/>	tushu	管理员	<div><div></div></div>	2023-12-17 13:16:16	<a href="#">🔗 修改</a> <a href="#">🗑 删除</a> <a href="#">» 更多</a>
<input type="checkbox"/>	k0003	王五	<div><div></div></div>	2023-12-17 14:37:53	<a href="#">🔗 修改</a> <a href="#">🗑 删除</a> <a href="#">» 更多</a>
<input type="checkbox"/>	k0001	张三	<div><div></div></div>	2023-12-17 15:00:23	<a href="#">🔗 修改</a> <a href="#">🗑 删除</a> <a href="#">» 更多</a>

正式说明进货功能实现：效果图如下，通过管理员角色登入进到“进货”目录下，原先的“修改”“删除”按钮被“进货”按钮替代

图书销售系统

图书集

库存

进货

客户信息

购买记录

入货单

首页

进货

书号

请输入书号

书名

请输入书名

出版社

请输入出版社

单号

请输入单号

商家

请输入商家

库存

请输入库存

单价

请输入单价

搜索

重置

	单号	书号	书名	出版社	商家	库存	单价	操作
<input type="checkbox"/>	d1	s1	解忧杂货店	南海出版公司	以淳	8	40	<a href="#">去进货</a>
<input type="checkbox"/>	d2	s1	解忧杂货店	南海出版公司	新华文轩	90	39	<a href="#">去进货</a>
<input type="checkbox"/>	d3	s2	巨人的陨落：世纪三部曲	江苏凤凰文艺出版社	盛世峰图书	48	39	<a href="#">去进货</a>
<input type="checkbox"/>	d4	s4	房思琪的初恋乐园	北京联合出版公司	新华	100	45	<a href="#">去进货</a>
<input type="checkbox"/>	d5	s3	我的前半生	新世界出版社	中信书店	79	50	<a href="#">去进货</a>

共 5 条

10条/页

1

前往 1 页

点击“进货”弹出对话框：通过滑动框选择进货数量

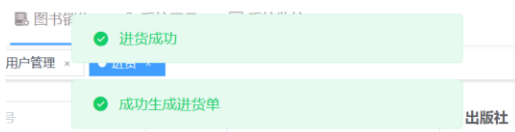


进货详情对话框，包含以下输入项：

- 单号: d1
- 书号: s1
- 商家: 以淳
- 单价: 40
- 商家库存: 8
- 进货量: 0 (滑动框)

底部按钮: 确定入货 (蓝色), 取消 (灰色)

确认入货后提示成功：并在 supply 表中插入对应的进货记录



查看入货单：观察到入货单更新入货条目，库存表中对应书条目库存量也增加

图书集

库存

进货

客户信息

购买记录

入货单

退货单

单号

请输入单号

入货数量

请输入入货数量

入货时间

请选择入货时间

搜索

重置

新增

修改

删除

导出

	编号	书号	书名	出版社	单号	入货数量	入货时间	操作
<input type="checkbox"/>	2	s1	解忧杂货店	南海出版公司	d1	5	2023-12-17	<a href="#">修改</a> <a href="#">删除</a>
<input type="checkbox"/>	3	s1	解忧杂货店	南海出版公司	d1	6	2023-12-17	<a href="#">修改</a> <a href="#">删除</a>
<input type="checkbox"/>	4	s3	我的前半生	新世界出版社	d5	21	2023-12-17	<a href="#">修改</a> <a href="#">删除</a>
<input type="checkbox"/>	5	s1	解忧杂货店	南海出版公司	d1	3	2023-12-17	<a href="#">修改</a> <a href="#">删除</a>
<input type="checkbox"/>	6	s2	巨人的陨落：世纪三部曲	江苏凤凰文艺出版社	d3	33	2023-12-17	<a href="#">修改</a> <a href="#">删除</a>
<input type="checkbox"/>	7	s1	解忧杂货店	南海出版公司	d1	2	2023-12-17	<a href="#">修改</a> <a href="#">删除</a>

实现过程：主要操作该目录 sales 的前端 vue 文件：根据定义的 character 角色判断是否显示该按钮，以及定义 handlesupply 为“进货”按钮触发函数

```
<template slot-scope="scope">
  <el-button
    size="mini"
    type="text"
    icon="el-icon-edit"
    @click="handleUpdate(scope.row)"
    v-hasPermi="['tushudb:sales:edit']"
    v-if="character==='超级管理员'"
  >修改</el-button>
  <el-button
    size="mini"
    type="text"
    icon="el-icon-edit"
    @click="handlesupply(scope.row)"
    v-hasPermi="['tushudb:sales:edit']"
    v-if="character==='超级管理员' || character==='图书管理员'"
  >进货</el-button>
```

进货按钮函数通过 supplyopen=true 打开弹出

```
/** 进货按钮操作 */
handlesupply(row){
  this.reset();
  const id = row.id || this.ids
  getSales(id).then(response =>{
    this.form = response.data;
    this.supplyopen = true;
    this.title = "进货详情";
  });
},
```

弹窗内容：将选择入货量赋值给 form.supplynum，最后调用 submitsupply 确认入货

```
<el-dialog :title="title" :visible.sync="supplyopen" width="500px" append-to-body>
  <el-form ref="form" :model="form" :rules="rules" label-width="80px">
    <el-form-item label="单号" prop="salesId">
      <el-input v-model="form.salesId" readonly />
    </el-form-item>
    <el-form-item label="书号" prop="bookId">
      <el-input v-model="form.bookId" readonly />
    </el-form-item>
    <el-form-item label="商家" prop="supplier">
      <el-input v-model="form.supplier" readonly />
    </el-form-item>
    <el-form-item label="单价" prop="price">
      <el-input v-model="form.price" readonly />
    </el-form-item>
    <el-form-item label="商家库存" prop="num">
      <el-input v-model="form.num" readonly />
    </el-form-item>
    <el-form-item label="进货量" prop="supplynum">
      <el-slider :max='form.num' :step='1' v-model="form.supplynum"></el-slider>
    </el-form-item>
  </el-form>
  <div slot="footer" class="dialog-footer">
    <el-button type="primary" @click="submitsupply" 确定入货</el-button>
    <el-button @click="cancel">取消</el-button>
  </div>
```

Submitsupply 函数实现：将 form.num 也就是 sales 表对应行的库存量降掉前面赋值的 form.supplynum，定义 form.time=new Date()得到当前时间，调用 updateSales 更新可进货表，调用 addSupply 将得到的进货信息插入 supply 表中。

```
/** 确认进货按钮 */
submitsupply(){
  this.form.num = this.form.num - this.form.supplynum;
  this.form.time = new Date();
  console.log(this.form);
  this.$refs["form"].validate(valid => {
    if (valid) {
      if (this.form.id != null) {
        updateSales(this.form).then(response => {
          this.$modal.msgSuccess("进货成功");
          this.supplyopen = false;
          this.getList();
        });
      }
      this.form.id = null;
      this.form.num = this.form.supplynum;
      addSupply(this.form).then(response => {
        this.$modal.msgSuccess("成功生成进货单");
      });
    }
  });
},
```



库存量增加实现使用了触发器，当入货表插入新入货单时，库存表进行更新，为 supply 定义触发器如下：

名	触发	插入	更新	删除
supply_stock	AFTER	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

定义

```
1 begin
2 IF(new.book_id not in (select book_id from stock))THEN
3   insert into stock(book_id,num) VALUES(new.book_id,new.num);
4 ELSEIF (1) THEN
5   update stock set stock.num=stock.num+new.num
6   where stock.book_id=new.book_id;
7 end if;
8 end
```

至此入货功能实现，效果已进行展示。

(4) 销售功能

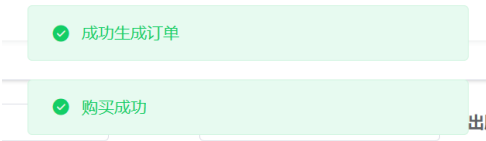
销售功能对应了客户的购买操作，主要在库存表 stock 上实现同入货功能相似操作，先展示效果图，以下为客户界面



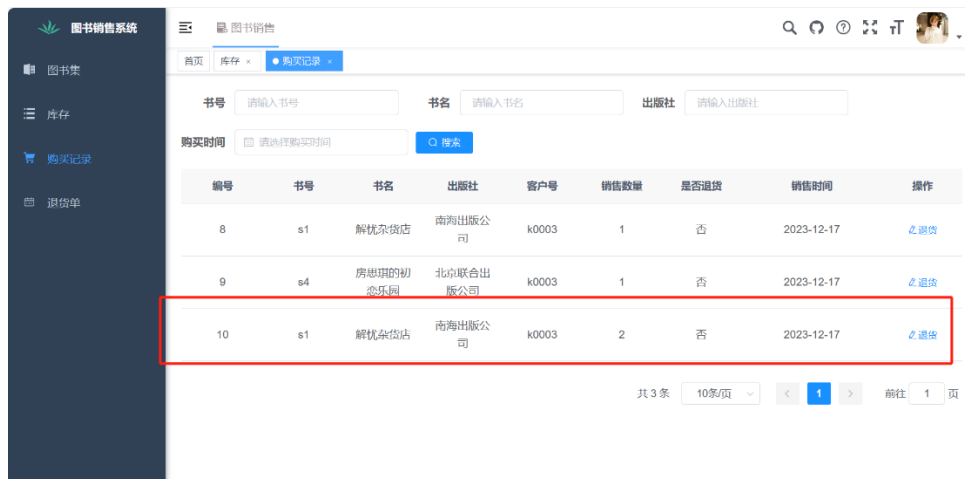
点击购买弹出窗口，选择购买数量，确认购买



购买成功提示



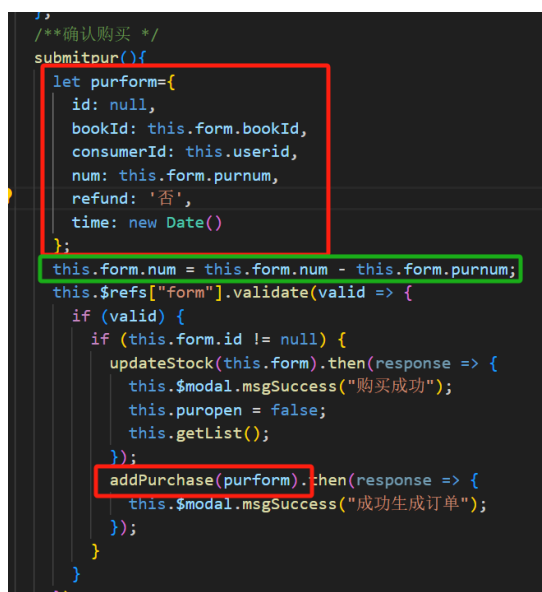
生成购买记录：对应库存中图书的库存量下降



实现过程：在目录 stock 的前端 vue 文件进行编写代码，以上文同样方式，根据用户角色为“客户”，得到该只存在“购买”按钮的界面，调用 handlepur 函数弹出窗口，窗口实现如下



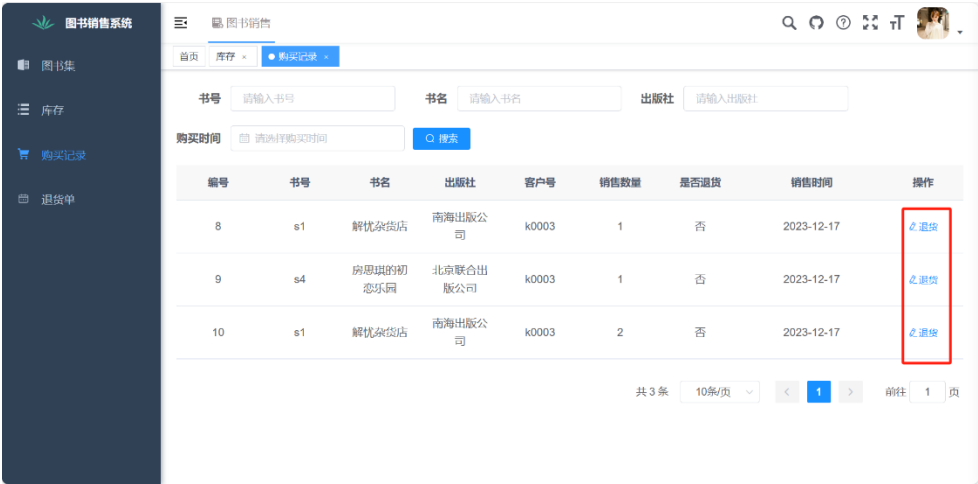
通过 submitpur 确认购买，定义一个 purform 类型，存储将要插入 purchase 表的信息，调用 addPurchase 将本次购买记录插入 purchase 表，而对于此处库存减掉的实现，因为正是在本 stock 表操作，不需要触发器，直接将 num 减掉对应购买量 purnum 调用 updateStock 即可。



至此购买功能实现结束

(5) 退货功能

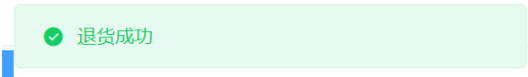
同上先看效果图：“客户”在购买记录中点击“退货”



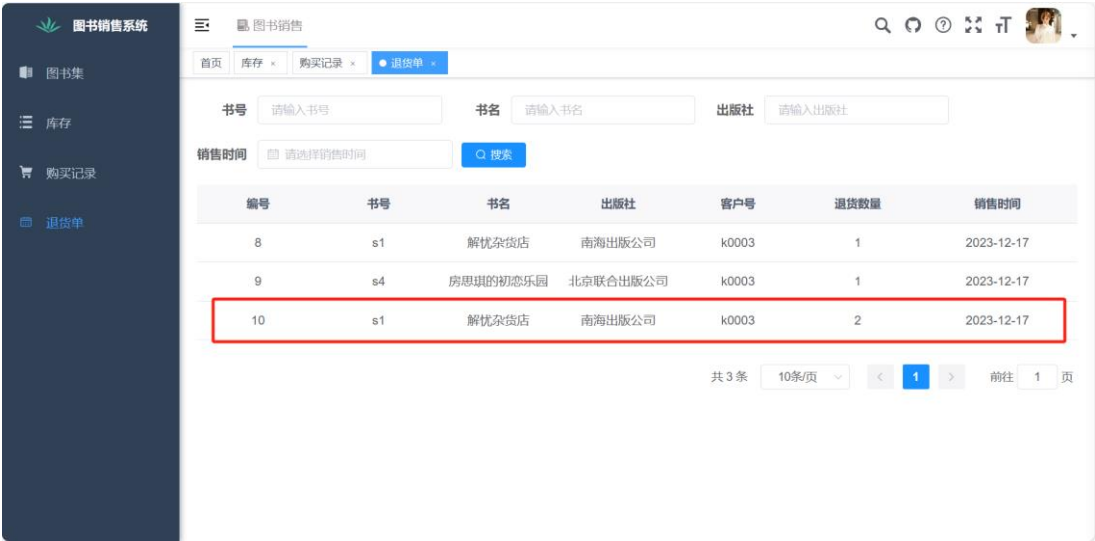
弹出窗口显示退货信息，点击“确认退货”



提示退货成功



然后就可以在退货单中查找到退货记录



实现过程：首先说明此处退货单和购买记录同用一张 purchase 表，根据前端 vue 生成两个菜单文件，在 web 界面初始化时将查询参数的 refund 是否退货置为“是”，即可把以及退货的购买记录筛选为退货记录。以及对于“管理员”和“客户”看到同一个 web 界面，但“客户”只需要看到自己的购买记录，对此同理对查询参数进行初始化，如果角色为“客户”则将筛选客户号为该客户账号的记录。

```
// 查询参数
queryParams: {
  pageNum: 1,
  pageSize: 10,
  bookId: null,
  consumerId: null,
  num: null,
  refund: '是',
  time: null,
  'book.name': null,
  'book.publish': null
},

created() {
  getUserProfile().then(response => {
    this.character = response.roleGroup;
    if(response.roleGroup === '客户'){
      this.queryParams.consumerId = response.data.userName;
    }
    this.getList();
  });
}
```

对于退货操作，实现原理比进货和购买相同但更简单，只需要将字段 refund 改为“是”即可，同样的思路，弹出窗口，点击确认调用了以下更新 purchase 表操作

```
/**确认退货按钮 */
submitrefund(){
  this.form.refund='是';
  this.$refs["form"].validate(valid => {
    if (valid) {
      if (this.form.id != null) {
        updatePurchase(this.form).then(response => {
          this.$modal.msgSuccess("退货成功");
          this.refundopen = false;
          this.getList();
        });
      }
    }
  });
}
```

对于库存量增加的实现同样使用到了触发器，检测 purchase 的 update，当有退货操作则将 stock.num 对应加上 purchase 的 num。

名	触发	插入	更新	删除
refund	AFTER	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

定义
<pre>1 BEGIN 2 if(old.refund = '否' AND new.refund = '是')THEN 3   update stock set stock.num = stock.num+new.num 4   where stock.book_id = new.book_id; 5 end if; 6 END</pre>

## 六. 总结

本课程设计中使用到了基本 mysql 的语法如建表过程的 create table。在生成代码中，包含了最基本的 select、insert、update、delete 操作；在多表关联查询中使用了 left join 拼接表；使用触发器解决库存量改变的问题等理论课的概念与知识。