https://gitee.com/chen-xiny/tushu

1Git clone 若依框架

[Git 大全 - Gitee.com](https://gitee.com/all-about-git)

2准备redis 使用了本地的＋6389

发现本地的ip地址别人ping不通 不在同一局域网下

[跨局域网远程访问SQL数据库方法\_不在同一局域网连接sql-CSDN博客](https://blog.csdn.net/hhxgdxwb/article/details/103830484)

[不同网络环境下的Mysql数据库远程连接（局域网或外网）\_不在同一局域网连接sql-CSDN博客](https://blog.csdn.net/qq_65220135/article/details/123999217)

成功：主机：845719j0o5.goho.co 端口：57654 （映射到127.0.0.1 3306）

对于redis 的端口：59366 （映射到127.0.0.1 6389）



3试运行成功 先将细节改一改，推到gitee

若依的地方改成图书销售管理系统

User对应用户管理

Role对应角色管理

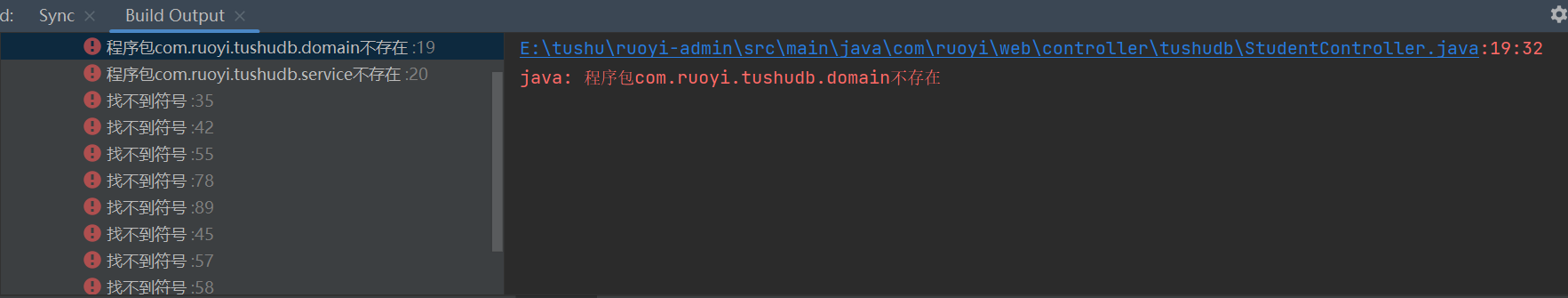
若依建表有个要求：表字段 和 表，都需要加注释，而且id自增，否则后面会报错。而且**注释就是生成页面的显示内容**

尝试添加目录：

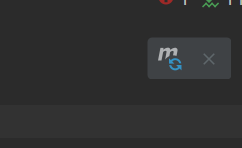
1. 删除main.Jar文件
2. 修改三个pom文件，不在创建文件夹的添加该tushudb依赖，本文件夹添加ruoyi-commom依赖
3. 菜单管理添加“图书销售管理”，路由地址tushudb
4. 生成代码填写该目录

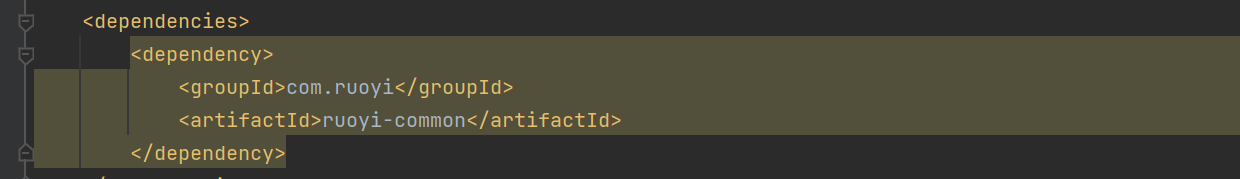


1. 将main移到tushudb/src，将vue/api移到ui/src/api，vue/view移到ui/src/view
2. 运行生成的sql文件
3. 分别重新运行前后端文件，发现学生表生成，但系统接口404异常
4. 尝试将control模块移到ruoyi-admin，在文件夹中移动，在编辑器可能有问题，果不其然，还是报错，感觉是添加依赖不起作用，所以control其实不需要移动



发现pom.xml中有这个东西，好像按一下才会更新，确实，终于搞定了，测试完成，建表

如果有调用到common的话会变色



先写报告readme，以及测试外键能不能行得通，因为有外键情况下，主键不能递增

补充：主键唯一，主码是最小属性集，外键即sales、stock中的book\_id，book是主表，sales、stock是从表。

外键不一定是另一个表的主键，但必须是唯一性索引。

Book是另外两个表的存在依赖，另外两个表完全参与，使用双横线连接

Book和sales是一对多的关系，因此不是特化，

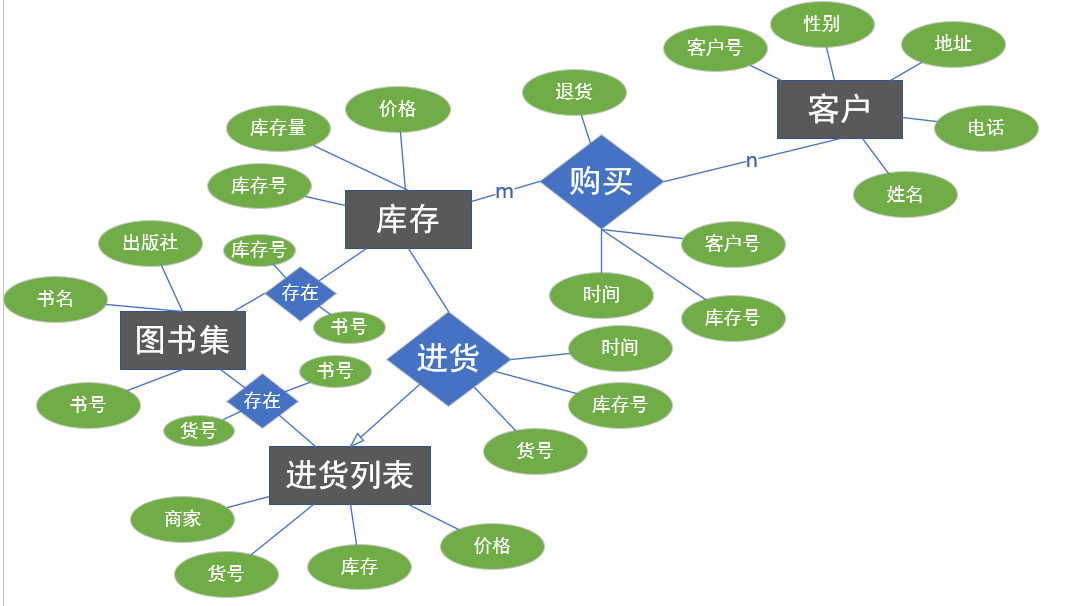
Sales一个实体只能对应book的一个，而book一个实体可以对应多个sales，一对多

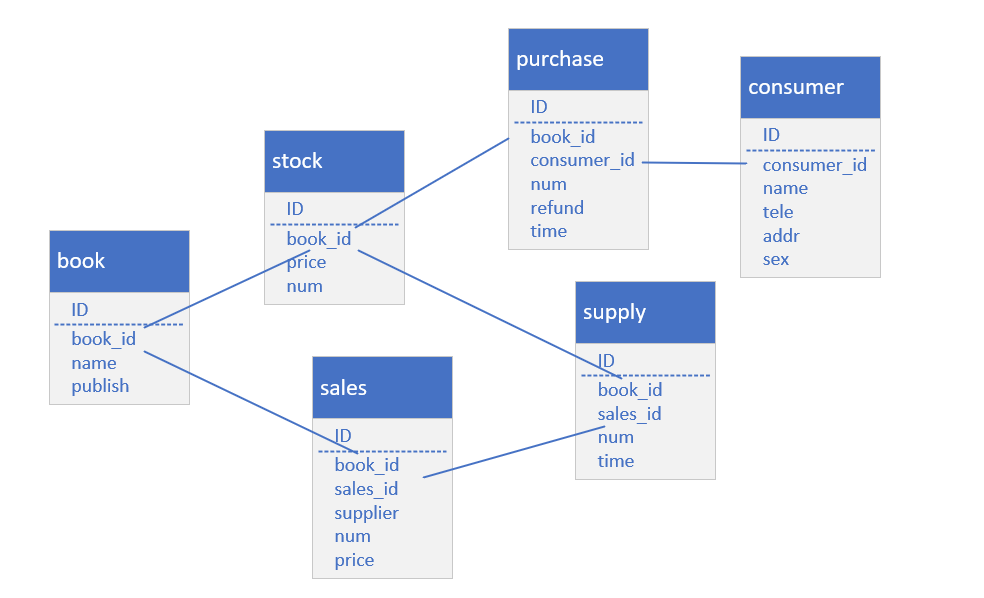
Supply一个实体只能对应sales的一个，sales的一个实体可以对应多个supply，一对多

因为若依中主子表关联只能选一个，所以尝试在从表中选择book，book选择单表

弱实体集：没有足够属性以形成主码。弱实体集的主码由该弱实体集的标识实体集的主码和该弱实体集的分辨符组成。分辨符用虚下划线。在ER图中stock是book的弱实体集，

由此得到





ID全部用于单调递增，实际使用中不需要stock\_id，可用book\_id代替

Book中的book\_id不重复

stock中的book\_id不重复的，sales中的book\_id可重复

purchase中保存销售记录，book\_\_id可不设外键

supply保存的是入货记录，不依赖于stock和sales，故不设外键

关系模式：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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 |

下一步工作：

尝试添加book表，出错：

1. ID没有显示
2. 修改和删除键用不了，只能添加

猜测：id为自增，书号为其他表的外键，导致涉及这两个操作时只能增加，无法修改和删除，甚至id没法显示。（错误的猜测）

看到controlmapping那些文件都没被调用到



书号应该改成char（10）

重建了张book表就可以了，估计难道是外键的问题

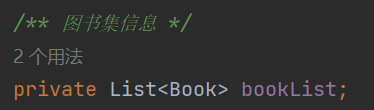
Stock表尝试主子表，结果如下，可参考读取到图书集的思路

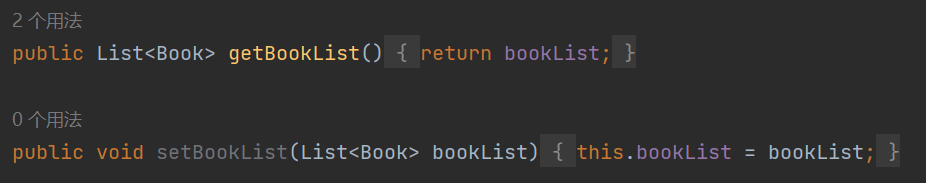


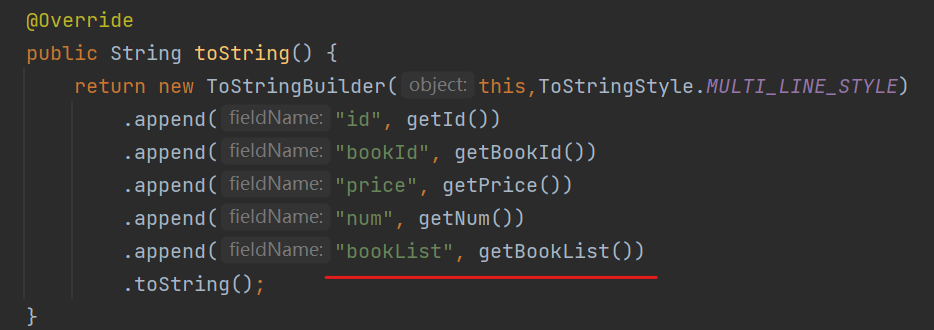
一共6个文件，以及vue文件

Cotroller无需修改的样子

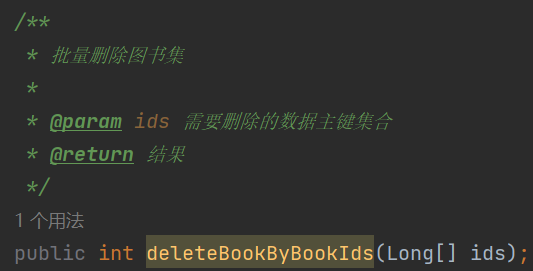
Stock文件包括

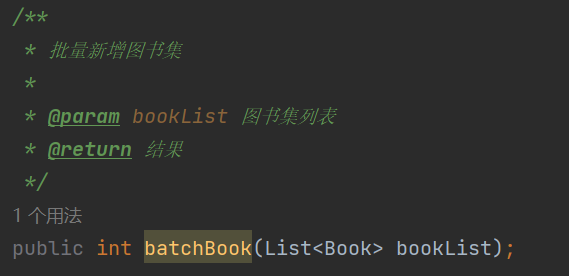


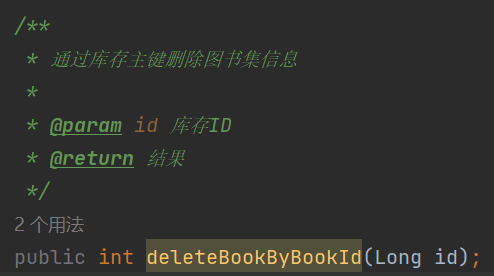




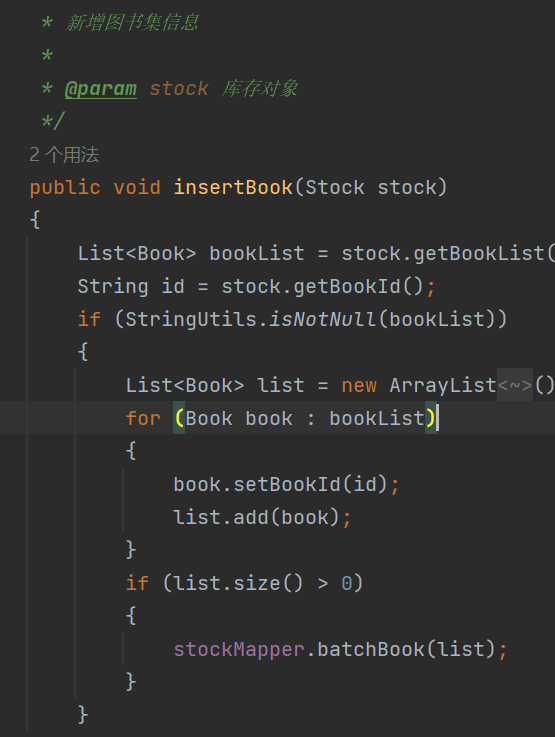
Mapper文件





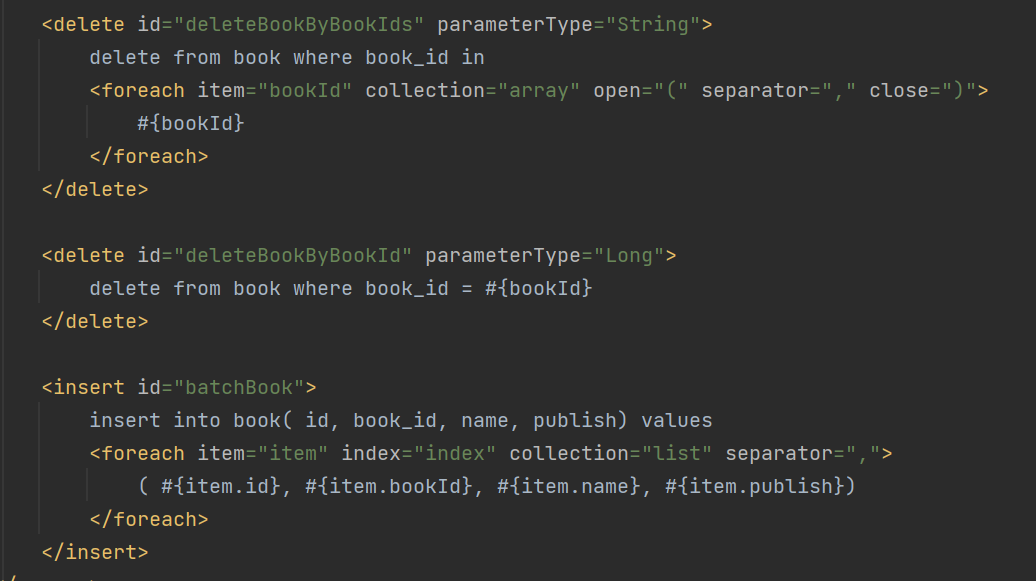


Serviceimpl文件



Service也没有

Mapper.xml



可以推出主要传递路径为mapper.xml到mapper到service Serviceimpl

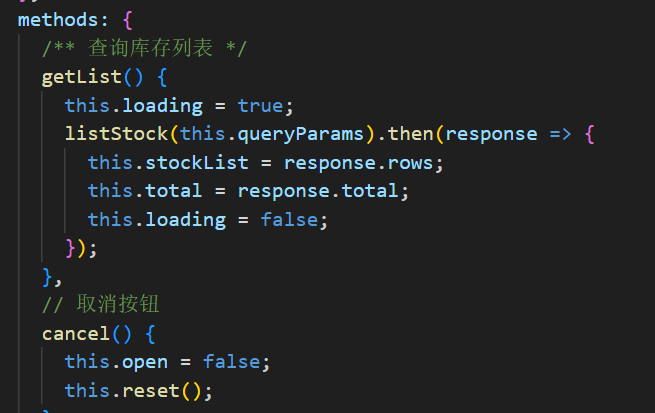
Controler设置权限，允许stockService执行

(1)Stock维护了stock这个类，使其能够调用get set函数比如getBookList，加成员函数

(2)Mapper中维护了stockMapper这个类，调用sql语言的函数（定义新sql语句）

(3)Service维护了IstockService这个类，有service基本的函数，供controler设置权限

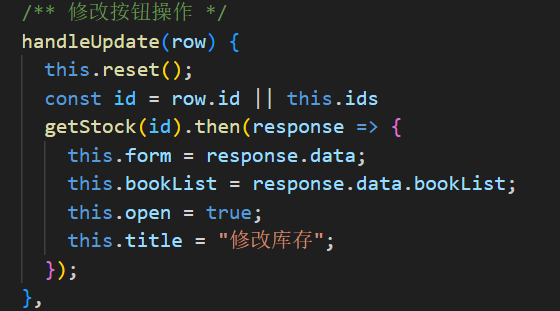
Serviceimpl维护了StockServiceImpl可以调用stock和stockMapper类的函数，写逻辑



Methods下写函数

ListStock到stock.js文件下，调用get





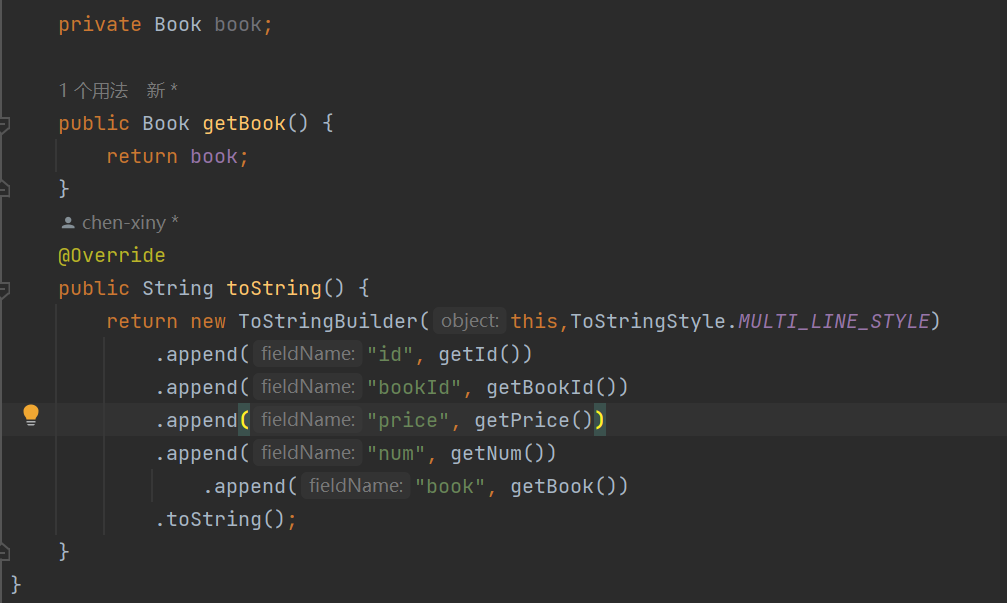
使用代码生成功能将所有表的代码生成





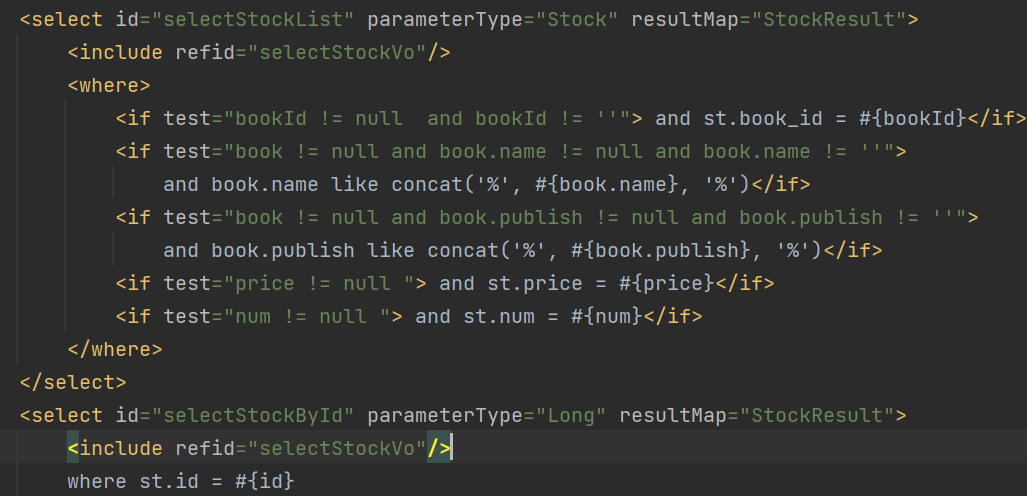


下一步：实现第一个功能：库存展示图书信息



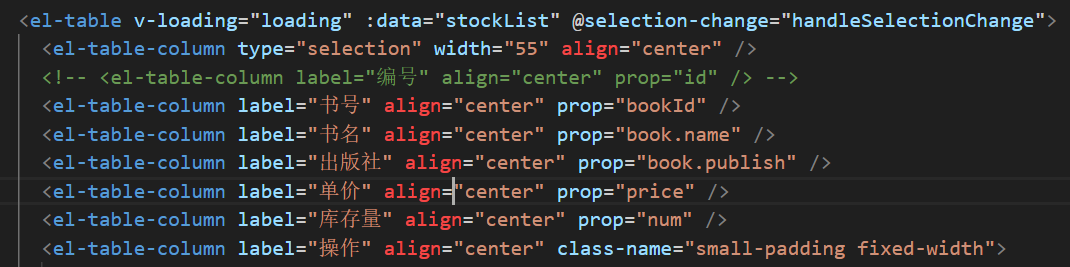
添加其他属性用于export，这里不需要





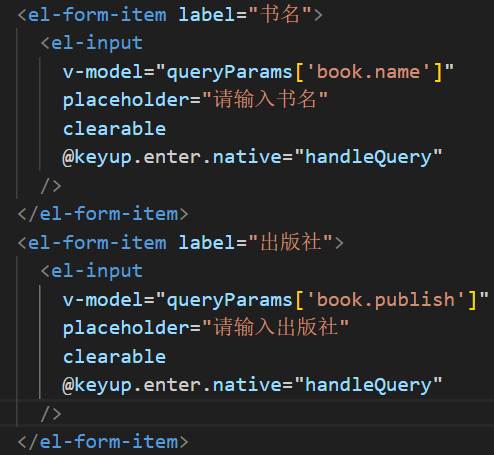
比较左边为带\_的，右边为property

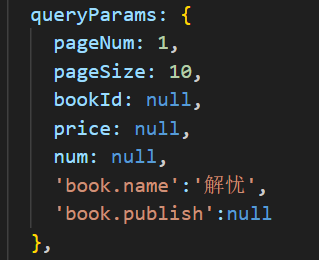
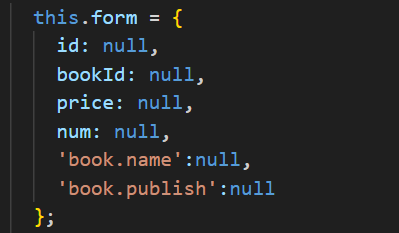
注意：下面selectbyid中id以及原来的比较中要加入st.



结果成功显示



加入了搜索功能



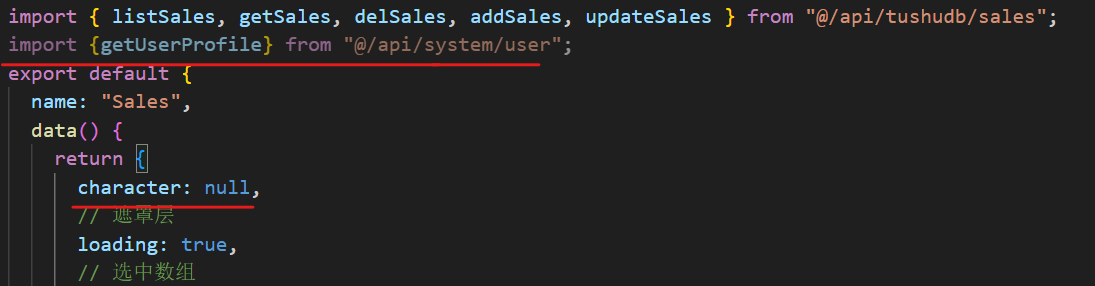
第三天：将进货同样引入图书信息，更新成功

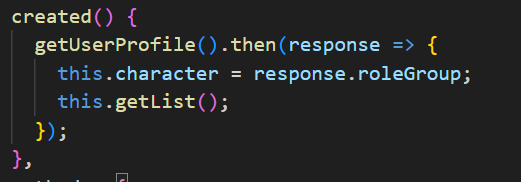
插一个建表注意事项：

1. Id小写，注释编号，id自动递增
2. 检查是否由索引、外键
3. 记得整个表的注释

实现进货功能

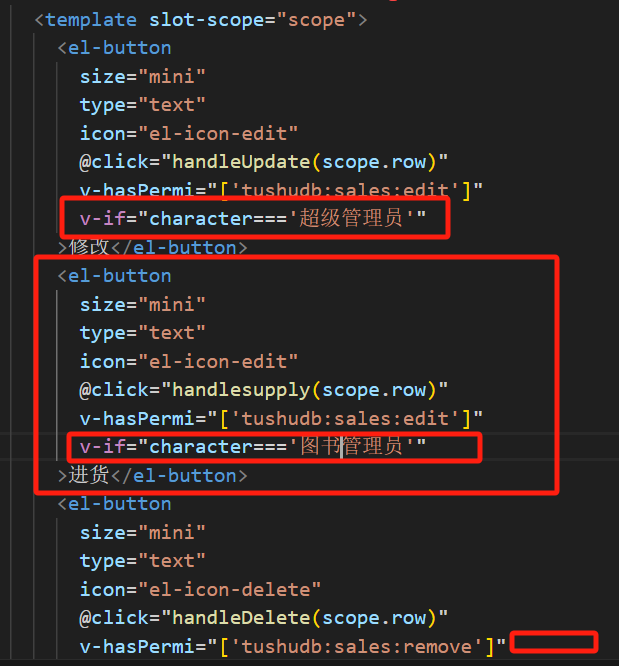
先引入character，前端得到当前用户身份

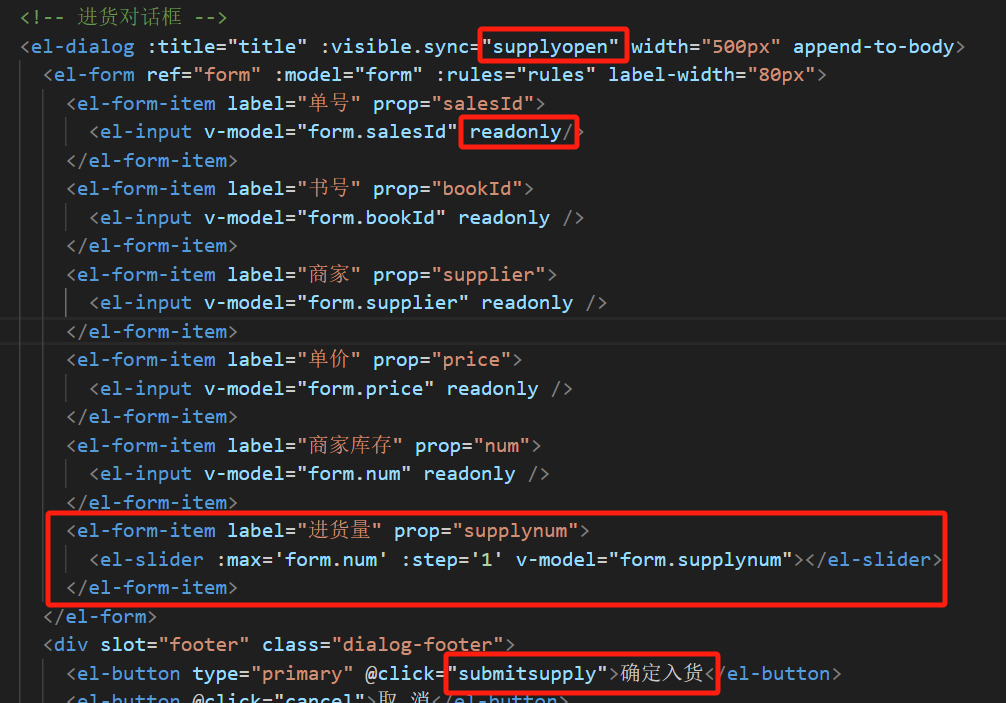


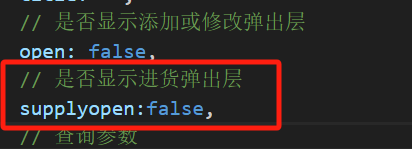
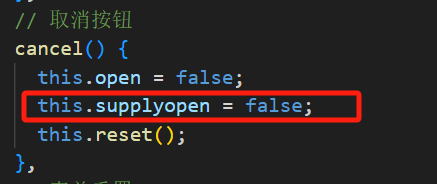


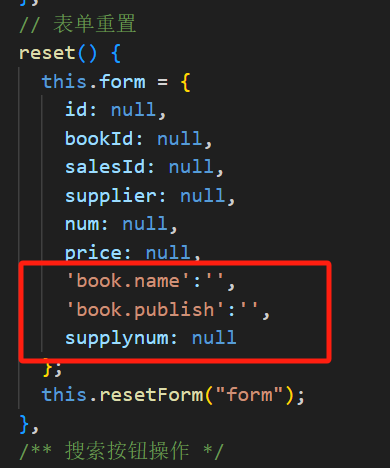
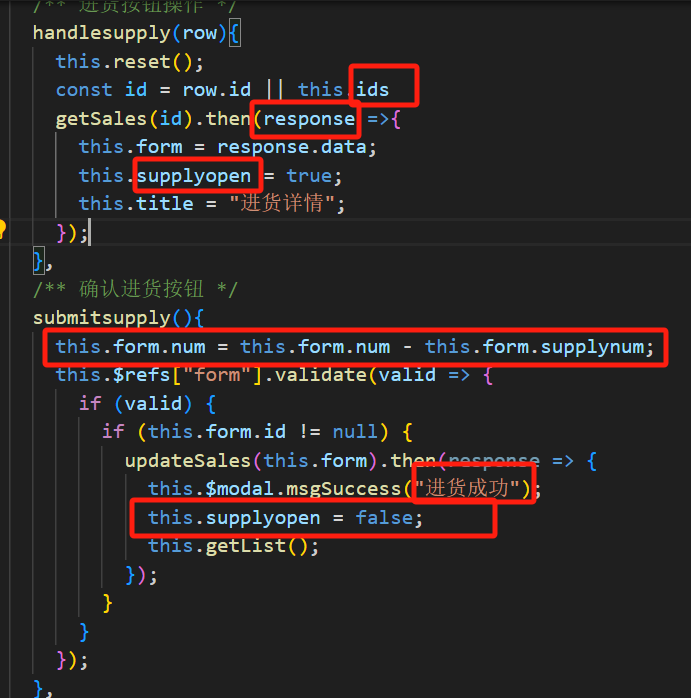
根据用户身份不同，如“超级管理员”看到“修改”键，而“图书管理员”看到“进货”键。

图书管理员：





入货触发supply表插入

参考

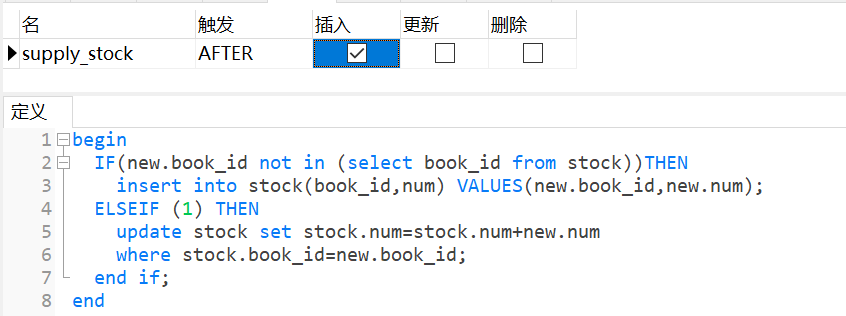
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Supply | | ID | | Book\_id | | Sales\_id | | Num | | time | |
|  | | Int | | Char(5) | | Char(5) | | Int | | datetime | |
| 解释 | |  | | 书号 | | 单号 | | 入货数量 | | 入货时间 | |
| Sales | ID | | Book\_id | | 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 |

同用一个num，需要中途改num=supplynum, id =null





实现入货后库存变动，supply加入触发器



下一步

客户界面：

在添加用户时将客户信息也添加上

判断是否是“客户”，

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 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 |

比较麻烦，还是分别在用户信息和客户信息分别导入吧

实现购买操作

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 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 |

同样的方法，改进：使用了let构造form ，以及consumer\_id通过username

实现成功

下一步

根据用户看到仅属于自己的购买记录：  
同样引入import character 初始化get character 进行判定 初始化查询参数getlist

页面改好

增加退货功能

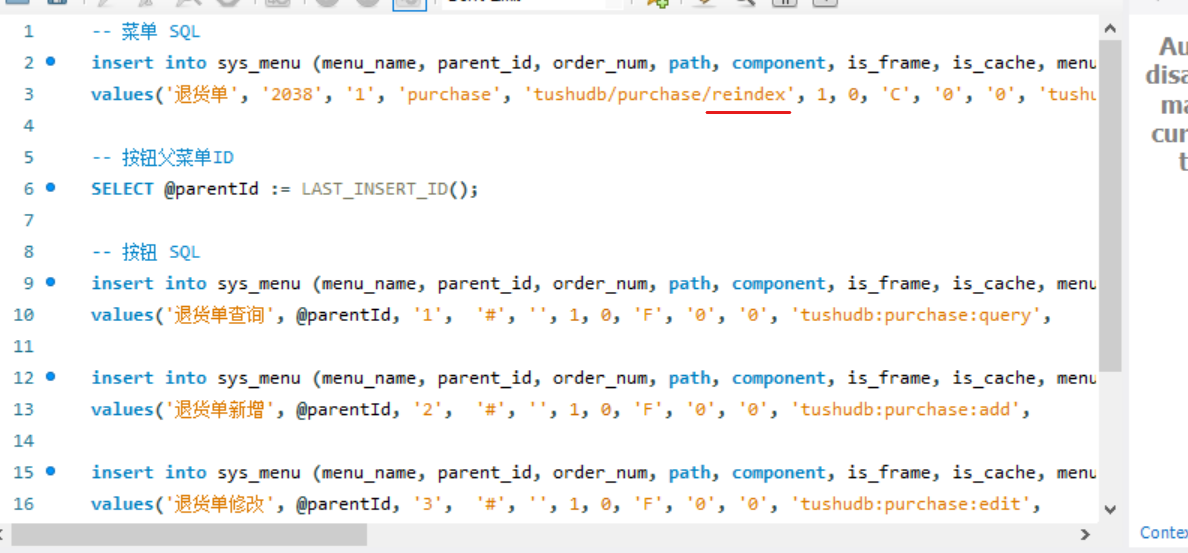
同样的方法：将自己purchase表更新即可

需要加一个触发器，当退货时库存量+

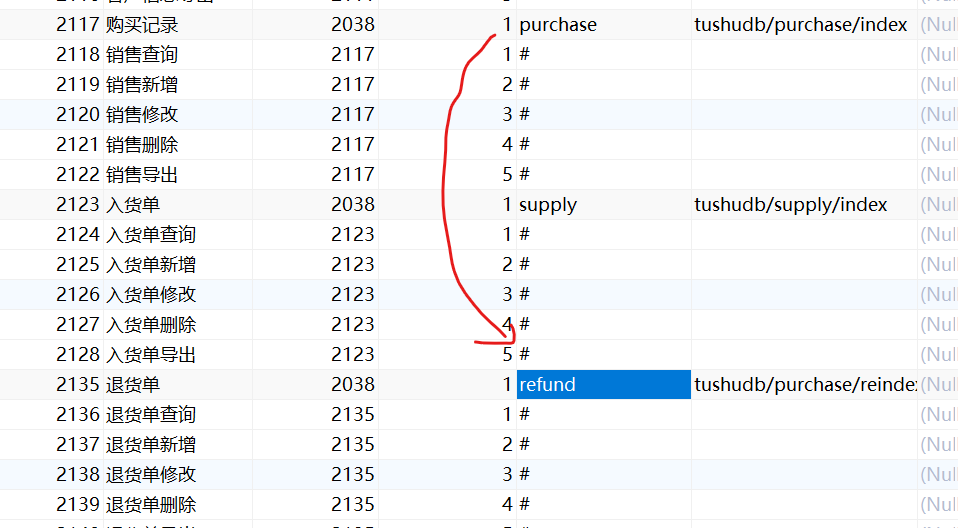


生成退货单

运行



改了path，可以在前面改，忘了而已



改改页面，记得测试用户看到的

购买记录purchase 入货单supply 退货单purchase的页面加入图书信息