User to User BookStore
即用户既可以当买家也可以当卖家，
当卖家需要开店铺

balance/余额　registerDate/注册时间
address/收货地址
phoneNumber/手机号
email/邮箱
avatar/头像
password/密码
user_id(username)/用户ID（用户提供）

User/用户

由User表存储，需在Django中单独建立一个App为User提供注册/登录/注销 等服务

1　创建　1 or m

Store/商店

storeID(storeName)/商店ID（用户提供）
address/店铺地址
createDate/创建时间
info/店铺信息
sold/店铺总售出

由Store表存储

1

所属

count/书本数量

Cart/购物车

updateTime/更新时间

由Cart表存储，需在Django项目中单独建立一个App用于购物车页面，提供各种管理购物车的方式

购物车相当于一个缓存（本质是一种关系！！！也可以叫做收藏），实际订单提交时与购物车无关，只是提交关于各种书的组合，然后将此关系删除

1

售卖

由于实际应用的页面，Store和Book在Django项目中只需要一个App即可

m

所属

m

sumPrice/订单总金额
order_id/订单号（系统生成）
paid/是否已付款
success/交易是否成功
shippingAddress/发货地址
receivingAddress/收货地址
state/交易状态
各种时间

Order/订单

由Order表存储

一个订单对应一个用户和一个商店，其中的书可以是各种组合

1　组成　m

OrderBook/书本

m　部分　1

Book/书籍

bookID/书本ID（用户提供）
title/书名
ISBN
stock/库存
sold/售出数
其他参考P2 README

由Book表存储

组成时有条件，只有同一店铺的书本才能组成同一个订单

count/书本数量
bookPricce/书本总金额
部分冗余信息

由于实际应用，Order和OrderBook在Django项目中只需要一个App即可，用于维护订单状态，即订单 生成/提交/支付/完成