**学期大作业前瞻调研与简要分析**

1. **选题与调研**

这学期的学期大作业，经过调研选题最终被确定为校园社区二手商品交易平台的数据库部分。

针对目前校园内毕业季等二手商品流转的现状，我发现在南开大学并没有比较正式的交易平台，大多数同学选择利用校园集市等方式进行信息交流，但具体付款购买还需要交易者添加联系方式进行进一步交流，这种方式极其不便，同时也不利于保护个人隐私，因此学期大作业将针对这个任务进行一定的尝试与落实。下面是一些具体需求：

1. 核心功能需求：

* 用户管理：支持学生实名注册（需学号验证）、登录、个人信息维护（联系方式、收货地址等）
* 商品管理：支持多级分类（图书/电子产品/服饰等）、商品发布（标题/描述/图片/价格）、库存状态管理
* 交易流程：购物车功能、订单生成（含支付状态跟踪）、物流信息关联（校园自提点管理）
* 社交互动：商品收藏功能、买卖双方评价系统（信用评分）、论坛交流板块
* 通知系统：交易状态推送（短信/站内信）、违规商品下架通知

1. 特殊场景需求（可选）：

* 毕业季大件物品（如家具/自行车）交易专区
* 虚拟商品交易（如课程笔记/电子书资源）
* 校园跑腿服务（代取快递/代购）的供需对接
* 商品质量审核机制（管理员人工审核+AI图像识别）

1. **实体简介、ER图与关系模式**
2. 实体定义
3. 用户（User）​​

主键：UserID

其它属性：学号、姓名、手机号

子类划分：

* 买家：仅支持商品浏览、下单、支付功能；
* 卖家：需提交身份证信息（补充属性：证件号、实名状态）；
* 管理员：商品审核、权限分配

1. 商品（Product）​​

主键：ProductID

外键：UserID（卖家）、CategoryID

其它属性：名称、描述、价格、上架时间、审核状态

1. 订单（Order）​​

主键：OrderID（时间戳+随机数）

外键：UserID、ProductID

其它属性：总金额、交易状态（待付款/已发货/已完成）

1. 评价（Review）​​

主键：ReviewID

外键：OrderID、UserID（用户）

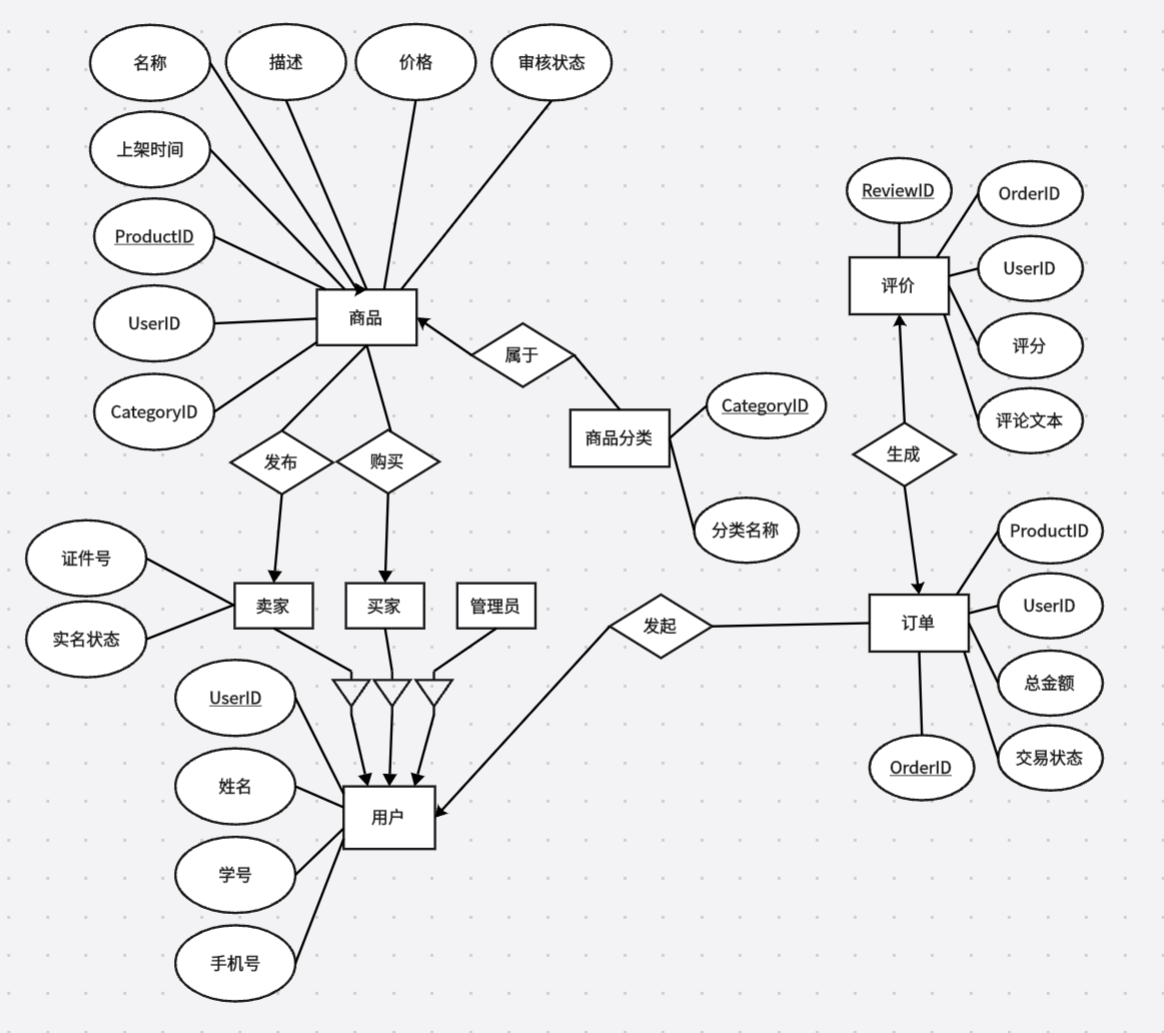
其它属性：评分（1-5星）、评论文本

1. 商品分类（Category）​​

主键：CategoryID（层次编码，如01.01表示图书-教材）

其它属性：分类名称

1. ER图



1. 关系模式

用户（UserID，学号，姓名，手机号）

买家（UserID）

卖家（UserID，证件号，实名状态）

管理者（UserID）

商品（ProductID，名称，描述，价格，上架时间，审核状态，UserID，CategoryID）

订单（OrderID，总金额，交易状态，UserID，ProductID）

评价（ReviewID，评分，评论文本，OrderID，UserID）

商品分类（CategoryID，分类名称）

1. **SQL语句设计与5种查询样例**
2. SQL语句设计

|  |
| --- |
| -- 用户基表（所有用户类型的公共属性）  CREATE TABLE User (  UserID VARCHAR(36) PRIMARY KEY,  学号 VARCHAR(20) UNIQUE NOT NULL,  姓名 VARCHAR(50) NOT NULL,  手机号 VARCHAR(11) UNIQUE NOT NULL,  )  -- 买家子表（继承自User）  CREATE TABLE 买家 (  UserID VARCHAR(36) PRIMARY KEY,  FOREIGN KEY (UserID) REFERENCES User(UserID) ON DELETE CASCADE  );  -- 卖家子表（继承自User）  CREATE TABLE 卖家 (  UserID VARCHAR(36) PRIMARY KEY,  证件号 VARCHAR(18) UNIQUE NOT NULL,  实名状态 ENUM('已认证','未认证') DEFAULT '未认证',  FOREIGN KEY (UserID) REFERENCES User(UserID) ON DELETE CASCADE  );  -- 管理员子表（继承自User）  CREATE TABLE 管理者 (  UserID VARCHAR(36) PRIMARY KEY,  FOREIGN KEY (UserID) REFERENCES User(UserID) ON DELETE CASCADE  );  -- 商品分类表（层次化编码）  CREATE TABLE 商品分类 (  CategoryID VARCHAR(10) PRIMARY KEY,  分类名称 VARCHAR(50) NOT NULL,  );  -- 商品表  CREATE TABLE 商品 (  ProductID VARCHAR(36) PRIMARY KEY,  名称 VARCHAR(100) NOT NULL,  描述 TEXT,  价格 DECIMAL(12,2) UNSIGNED NOT NULL,  上架时间 DATETIME DEFAULT CURRENT\_TIMESTAMP,  审核状态 ENUM('已审核','待审核','未通过') DEFAULT '待审核',  UserID VARCHAR(36) NOT NULL,  CategoryID VARCHAR(10) NOT NULL,  FOREIGN KEY (UserID) REFERENCES 卖家(UserID) ON DELETE CASCADE,  FOREIGN KEY (CategoryID) REFERENCES 商品分类(CategoryID),  );  -- 订单表  CREATE TABLE 订单 (  OrderID VARCHAR(32) PRIMARY KEY,  总金额 DECIMAL(12,2) UNSIGNED NOT NULL,  交易状态 ENUM('待付款','已发货','已完成') DEFAULT '待付款',  UserID VARCHAR(36) NOT NULL COMMENT '买家ID',  ProductID VARCHAR(36) NOT NULL,  FOREIGN KEY (UserID) REFERENCES 买家(UserID) ON DELETE CASCADE,  FOREIGN KEY (ProductID) REFERENCES 商品(ProductID),  );  -- 评价表  CREATE TABLE 评价 (  ReviewID INT AUTO\_INCREMENT PRIMARY KEY,  评分 TINYINT UNSIGNED CHECK (评分 BETWEEN 1 AND 5),  评论文本 TEXT,  OrderID VARCHAR(32) NOT NULL,  UserID VARCHAR(36) NOT NULL,  FOREIGN KEY (OrderID) REFERENCES 订单(OrderID),  FOREIGN KEY (UserID) REFERENCES 买家(UserID),  ); |

1. 5种查询样例

* 单表查询

查询未审核商品信息

|  |
| --- |
| SELECT 名称, 价格, 上架时间  FROM 商品  WHERE 审核状态 = '待审核'; |

* 多表连接查询

带用户信息的订单明细

|  |
| --- |
| SELECT 订单.OrderlD, 总金额, 用户.姓名, 用户.手机号  FROM 订单  NATURAL JOIN 用户  NATURAL JOIN 商品; |

* 多表嵌套查询

查询从未下单用户

|  |
| --- |
| SELECT 学号, 姓名  FROM 用户  WHERE UserID NOT IN (  SELECT DISTINCT UserID  FROM 订单  ); |

* EXISTS查询

查询有在售商品的卖家

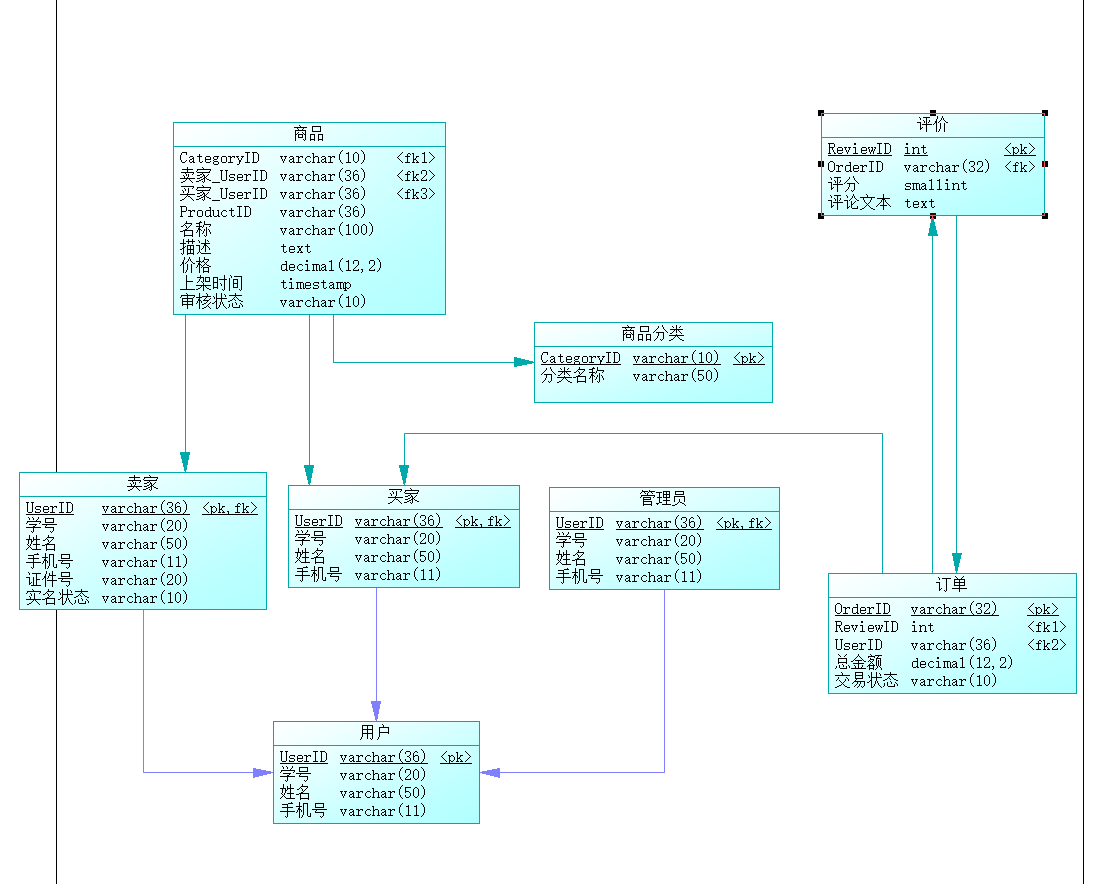
|  |
| --- |
| SELECT 姓名, 证件号  FROM 用户  NATURAL JOIN 卖家  WHERE EXISTS (  SELECT 1  FROM 商品  WHERE UserID = 卖家.UserID  AND 审核状态 = '已审核'  ); |

* 聚合操作查询

分类商品统计

|  |
| --- |
| SELECT 类名称, COUNT(ProductID) AS 商品数量  FROM 商品分类  NATURAL JOIN 商品  GROUP BY CategoryID; |

1. **Powerdesigner实现**
2. 用powerdesigner绘制ER图



1. 生成的MySQL代码

|  |
| --- |
| /\*外键声明在外\*/  /\*==============================================================\*/  /\* DBMS name: MySQL 5.0 \*/  /\* Created on: 2025/4/10 20:05:08 \*/  /\*==============================================================\*/  drop table if exists 买家;  drop table if exists 卖家;  drop table if exists 商品;  drop table if exists 商品分类;  drop table if exists 用户;  drop table if exists 管理员;  drop table if exists 订单;  drop table if exists 评价;  /\*==============================================================\*/  /\* Table: 买家 \*/  /\*==============================================================\*/  create table 买家  (  UserID varchar(36) not null,  学号 varchar(20) not null,  姓名 varchar(50) not null,  手机号 varchar(11) not null,  primary key (UserID)  );  /\*==============================================================\*/  /\* Table: 卖家 \*/  /\*==============================================================\*/  create table 卖家  (  UserID varchar(36) not null,  学号 varchar(20) not null,  姓名 varchar(50) not null,  手机号 varchar(11) not null,  证件号 varchar(20) not null,  实名状态 varchar(10) not null default '未实名',  primary key (UserID)  );  /\*==============================================================\*/  /\* Table: 商品 \*/  /\*==============================================================\*/  create table 商品  (  CategoryID varchar(10),  卖家\_UserID varchar(36),  买家\_UserID varchar(36),  ProductID varchar(36) not null,  名称 varchar(100) not null,  描述 text,  价格 decimal(12,2) not null,  上架时间 timestamp not null,  审核状态 varchar(10) not null default '待审核'  );  /\*==============================================================\*/  /\* Table: 商品分类 \*/  /\*==============================================================\*/  create table 商品分类  (  CategoryID varchar(10) not null,  分类名称 varchar(50) not null,  primary key (CategoryID)  );  /\*==============================================================\*/  /\* Table: 用户 \*/  /\*==============================================================\*/  create table 用户  (  UserID varchar(36) not null,  学号 varchar(20) not null,  姓名 varchar(50) not null,  手机号 varchar(11) not null,  primary key (UserID)  );  /\*==============================================================\*/  /\* Table: 管理员 \*/  /\*==============================================================\*/  create table 管理员  (  UserID varchar(36) not null,  学号 varchar(20) not null,  姓名 varchar(50) not null,  手机号 varchar(11) not null,  primary key (UserID)  );  /\*==============================================================\*/  /\* Table: 订单 \*/  /\*==============================================================\*/  create table 订单  (  OrderID varchar(32) not null,  ReviewID int,  UserID varchar(36),  总金额 decimal(12,2) not null,  交易状态 varchar(10) not null default '待付款',  primary key (OrderID)  );  /\*==============================================================\*/  /\* Table: 评价 \*/  /\*==============================================================\*/  create table 评价  (  ReviewID int not null,  OrderID varchar(32),  评分 smallint not null,  评论文本 text,  primary key (ReviewID)  );  alter table 买家 add constraint FK\_Inheritance\_1 foreign key (UserID)  references 用户 (UserID);  alter table 卖家 add constraint FK\_Inheritance\_2 foreign key (UserID)  references 用户 (UserID);  alter table 商品 add constraint FK\_发布 foreign key (卖家\_UserID)  references 卖家 (UserID);  alter table 商品 add constraint FK\_属于 foreign key (CategoryID)  references 商品分类 (CategoryID);  alter table 商品 add constraint FK\_购买 foreign key (买家\_UserID)  references 买家 (UserID);  alter table 管理员 add constraint FK\_Inheritance\_3 foreign key (UserID)  references 用户 (UserID);  alter table 订单 add constraint FK\_发起 foreign key (UserID)  references 买家 (UserID);  alter table 订单 add constraint FK\_生成 foreign key (ReviewID)  references 评价 (ReviewID);  alter table 评价 add constraint FK\_生成2 foreign key (OrderID)  references 订单 (OrderID); |

1. **分析**

1.两种设计的主要差异

（1） 继承关系实现方式不同

* PowerDesigner生成

采用"基表+子表"的物理继承模式，但存在逻辑矛盾：

用户表是独立基表，买家/卖家/管理员通过UserID外键关联基表，但未使用级联约束。

实际形成了"用户-买家"和"用户-卖家"的弱关联关系，而非真正的继承，可能导致数据孤岛（如删除基表用户记录后，子表记录残留）。

* 自己生成

明确使用逻辑继承 + 物理外键约束：

基表User存储公共属性，子表（买家/卖家/管理员）通过UserID外键引用基表，并添加ON DELETE CASCADE级联删除。

更符合面向对象设计，保证数据一致性（删除用户时自动清理子表记录）。

（2）商品与订单关联逻辑不同

* powerdesigner生成

商品表直接关联买家\_UserID字段，存在逻辑错误：商品应属于卖家，买家通过订单购买商品，而非直接绑定商品。

导致数据冗余（一个商品只能被一个买家购买）和业务逻辑混乱。

* 自己生成

商品表仅关联卖家，订单表通过ProductID和UserID（买家）建立购买关系。

符合实际业务场景（一个商品可被多个买家下单，订单记录购买行为）。

（3）状态字段设计差异

* powerdesigner生成

使用varchar存储状态（如审核状态 varchar(10)），存在风险：允许无效状态值（如输入"未知状态"），需依赖应用层校验。

* 自己生成

使用ENUM类型（如审核状态 ENUM('已审核','待审核','未通过')）：数据库层约束状态值，避免脏数据。

2.PowerDesigner生成的SQL特点

（1）外键后置声明

* 生成代码特点：

所有外键约束通过ALTER TABLE语句后置添加，而非在表定义时直接声明。

|  |
| --- |
| -- 示例：先创建表，后添加外键  CREATE TABLE 商品 (...);  ALTER TABLE 商品 ADD CONSTRAINT FK\_发布 FOREIGN KEY (卖家\_UserID) REFERENCES 卖家 (UserID); |

* 原因：

避免表创建顺序的依赖问题。若在商品表定义时直接引用卖家表，而卖家表尚未创建，会导致SQL执行失败。后置声明确保所有表结构就绪后再建立关系。

（2）缺少高级约束

生成代码未使用ENUM、CHECK等约束，而是通过varchar和注释描述业务逻辑（如审核状态 varchar(10) not null default '待审核'）。

影响：依赖应用层校验，数据库层无法保证数据完整性。

（3）无级联删除

生成的外键约束未指定ON DELETE CASCADE，删除父表记录时可能导致子表残留数据（需手动处理）。

3.对后期实现的影响

（1）数据一致性风险

powerdesigner生成的弱外键约束和错误关联（如商品绑定买家）可能导致数据不一致，需额外开发补偿逻辑（如手动清理孤儿记录）。

（2）开发复杂度

powerdesigner生成的状态字段（varchar）需在应用层实现校验逻辑，增加开发成本；自己生成通过ENUM和CHECK约束减少代码量。

（3）可维护性

自己生成的继承关系清晰，表结构符合范式，更易扩展（如新增用户子类型只需添加表，无需修改基表）。