**集 美 大 学**

**数据库原理** 课程设计

**题目：**\_\_\_\_\_\_\_\_\_ 电子商城系统\_\_\_\_\_\_\_\_\_\_\_\_

**评语：**

|  |
| --- |
|  |

学 院 计算机工程 班 级 计算1814

姓 名 黄勋 学 号 201821121104

周圣博 201821121117

成 绩 指导老师 林颖贤

2020年 5月

**目 录**

[设计总说明 I](#_Toc31433)

[第一章 需求分析 1](#_Toc18611)

[1.1 需求调研 1](#_Toc22838)

[1.2 数据流图 2](#_Toc25946)

[1.3 数据字典 6](#_Toc17043)

[第二章 概念结构设计 11](#_Toc13805)

[2.1 系统模块设计图 11](#_Toc19197)

[2.2 概念数据模型（CDM） 12](#_Toc19941)

[第三章 逻辑结构设计 14](#_Toc4245)

[3.1 物理数据模型（PDM） 14](#_Toc28994)

[3.2 视图设计及数据库安全性设计 15](#_Toc29743)

[第四章 物理结构设计 19](#_Toc24311)

[4.1选择合适的DBMS 19](#_Toc23716)

[4.2定义数据库、表及字段的命名规范 19](#_Toc8063)

[4.3 反范式化设计 22](#_Toc378)

[第五章 系统实现与测试 24](#_Toc345)

[5.1 系统实现功能 24](#_Toc24571)

[第六章 优缺点自我评价 42](#_Toc31303)

[6.1系统设计的自我评价 42](#_Toc22014)

[6.1 体会和建议 42](#_Toc7029)

[参考文献 44](#_Toc30696)

# 设计总说明

1. **系统开发目的**

近几年，随着网络建设的发展，电子商城类似于现实世界当中的商店，差别是利用电子商务的各种手段，达成从买到卖的过程的[虚拟商店](https://baike.baidu.com/item/%E8%99%9A%E6%8B%9F%E5%95%86%E5%BA%97/2820292" \t "https://baike.baidu.com/item/_blank)，从而减少中间环节，消除运输成本和代理中间的差价，造就对普通消费和加大市场流通带来巨大的发展空间。因此，本系统的设计是为消费者与商家之间的交易模式提供一个平台进行线上交易，克服实体店交易的缺点，以达到经济、便捷、稳定、安全、可靠的交易。

1. **设计内容**

设计一个关于数据库系统原理的电子市场系统。

1. **设计要求**

要求本系统实现的基本功能如下：

1. 实现在线下单功能，用户可以根据自己需求检索已有的商品，并选择心仪的商品下单购买。
2. 实现二手转卖功能，商家可以通过自己的账号将二手商品上传到商城进行二手转卖。
3. 要求本系统附带兑换折扣券功能，用户可以通过消费获取的积分兑换折扣券，可以用户下一次消费时进行折扣。
4. 要求本系统提供限时打折系列商品，系统设置一部分商品在指定时间段的内打折。
5. 要求本系统附带查询订单物流的功能，用户在系统内可以查询本人订单的物流信息。
6. 实现拍卖功能，用户可以对商家上架的拍卖商品进行竞拍，最后价高者得。
7. 要求本系统附带查询商品评价的功能，用户可以在浏览各个商品的评价。同时用户也可以对自己购买商品进行评价。
8. **本系统功能简介**

本系统主要分用户注册与登录，找回密码，检索商品，查询折扣商品，创建与查询订单，兑换折扣券，上架二手商品，用户与商品管理，折扣券管理，订单管理10个功能，其各个功能介绍如下：

1. 用户注册与登录：为用户详细提供注册账号的功能，用户根据自己的填入的信息进行账号的注册与登录。
2. 找回密码：本系统提供了找回密码的功能，用户可以根据自己预留的手机号、邮箱等信息进行重置密码。
3. 检索商品：本系统提供了检索商品的功能，用户可以根据期望商品的关键字进行检索。
4. 查询折扣商品：本系统提供了查询折扣商品功能，供用户一键查询折扣的商品。
5. 创建与查询订单：用户可以根据自己的需求创建订单顺利的通过WEB在线的方式，直接生成购买订单,同时也可以对自己的订单记录进行查询。
6. 兑换积分商品：本系统提供了兑换积分商品功能，用户可以根据自己在平台获取的积分进行积分商品的兑换。
7. 上架二手商品：本系统提供了二手商品交易功能，用户可以上传二手商品进行销售。
8. 用户与商品管理：本系统提供了用户与商品管理功能，管理员可以管理用户的账号以及商品的信息。
9. 管理员管理：为便于本系统的管理，管理员有类型之分，超级管理员可以对管理员的信息进行管理。
10. 订单管理：本系统提供了订单管理功能，管理员可以进行订单管理，主要用于订单出错等问题出现时候的修改。
11. **设计平台**

本系统是在Windows 10环境下开发的，使用的开发软件采用Eclipse，数据库采用的是SQL Server 2012。

1. **开发情况**

本系统是由计算1814的黄勋和周圣博，共同开发，其工作分配情况如下：

数据库设计部分：

* 黄勋负责需求分析与调研，周圣博负责根据需求分析编写数据库建表代码，并根据代码完成CDM、PDM的设计。
* 黄勋负责数据库存储过程的编写，周圣博负责数据库触发器的编写，同时两人对自己编写的SQL代码进行自编数据测试。
* 数据库课程设计的补充要求方面如：考虑事务的并发处理、考虑数据库安全性与完整性的情况等则由两人经过讨论得出结果后对代码进行补充或修改。

JavaEE界面设计部分：

* 黄勋主要负责系统用户服务操作的界面功能设计，周圣博主要负责商家、管理员对数据库修改操作的界面功能设计。

# 需求分析

## 1.1 需求调研

1. 记录每个用户的信息。用户的信息包括用户账号、用户密码、用户昵称、用户地区、注册手机号、真实姓名、身份证号、用户邮箱、性别、出生日期、用户积分。用户账号不可以重复。其中用户密码为6-12位，注册手机号每个用户唯一。用户邮箱必须含有“@”，用户性别只能填入男或女,身份证号只能是18位。
2. 记录每个商品管理人员的信息。管理人员的信息包括管理人员账号、管理员密码、真实姓名、身份证号、邮箱、手机号，性别。其中密码为6-12位，邮箱必须含有“@”，性别只能填入男或女,身份证号只能是18位。
3. 记录所有商品的信息。商品信息包括商品号、商品名称、生产厂家、卖家、商品价格、商品介绍、是否进口、具体规格、适宜人群、注意事项、商品预览图、商品出售方式（包括正常、二手、兑换）、库存、商品分类名称、商品分类介绍。商品号不可以重复，一个商品只能属于一个商品分类，同一个商品名称可以有多种商品介绍（颜色、口味等等）。
4. 用户登录到系统后，可以通过检索或者浏览、选择商品，将商品加入到购物车，商品可以长期存放在虚拟的购物车里面，购物车里面有用户账号、商品号、商品数量等信息。用户也可以通过购物车确认购买商品，用户自选系统备好的第三方支付平台进行付款创建订单，对于正常的订单，信息包括订单号、用户账号、商品号、地址信息、快递单号、购买数量、拍卖号（当且仅当交易类型为拍卖时填写）、商品名、支付平台、订单建立时间、用户备注。然后商品线下配送。一个用户可以加入多个商品到购物车，一个商品也能被多个用户加入到购物车，但是一个用户只能加入同一个商品到购物车一次。一个用户可以购买多个商品，一个商品也能被多个用户购买，一个用户能购买多次同一个商品。
5. 提供二手商品专柜，用户可以通过二手商城将自己购买的商品创建二手商品进行转卖，二手商品信息与正常商品一起存储（利用商品出售方式区分）。
6. 用户可以浏览自己的购物车，历史订单以及订单的物流信息。每个用户只能查看自己的购物车和历史订单。
7. 提供积分兑换功能，用户可以通过自己的积分购买优惠券或者是限定的一些商品。兑换物品也同正常商品一起存储。
8. 提供限时打折系列商品，系统设置一部分商品在一定时间的内打折。设立一个限时打折专栏，包含：打折开始时间、打折商品号、打折后商品价格、打折结束时间。
9. 记录所有的地址信息。地址信息包括：序号、用户账号，地址信息，收件人电话，收件人姓名，默认地址标识。用户可以选择一个地址信息作为自己的默认地址信息，在购买商品的时候会自动填入该地址。
10. 记录所有商品的评价，当用户需要查询一个商品的评价时候，检索并反馈出表中所有该商品号的评价。评价表包含订单号、评价时间、评价内容、追加评价。
11. 提供拍卖功能，拍卖表记录拍卖号、商品号、拍卖开始时间、拍卖结束时间、当前最高价（初始值为当前最高价）、当前最高价用户账号（可为空表示无人出价）。
12. 记录所有商品管理信息，商品管理表包括序号、商品号、商品进货数量、进货时间、进货管理人员。
13. 记录所有订单的配送信息，信息包括快递单号、发货时间、发货地、签收时间、快递员、物流信息.

## 1.2 数据流图

如图1图2图3图4图5图6的数据流图所示，本系统中主要分为用户和管理员两大模块，用户模块通过系统进行账号的注册登录、订单评价、查询商品信息、上架商品、创建订单、查询订单的各个操作，管理员模块通过系统进行管理员信息、订单信息、商品信息、用户信息的管理。

图 1 顶层数据流图



图 2 0层数据流图



图 3 1层数据流图1



图 4 1层数据流图2



图 5 1层数据流图3



图 6 1层数据流图4

## 1.3 数据字典

表3-1　地址信息表（Address）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 地址序号 | Add\_no | Int | PK，NOT NULL | 主键，自增长，步长为1 |
| 用户账号 | Cus\_no | Char(10) | NOT NULL | 记录是哪一个用户的地址信息，插入时需要保证用户表中存在账号，修改时也会随着改变删除时修改为‘&&’ |
| 收件人姓名 | Add\_Nname | Varchar(10) | NOT NULL | 收件人的姓名 |
| 收件人电话 | Add\_tel | Char(16) | NOT NULL | 收件人电话号码 |
| 地址信息 | Add\_inf | Varchar(20) | NOT NULL | XX省XX市…… |
| 默认地址 | Add\_def | Char(1) |  | 为默认地址则填1，默认地址一个用户只能有一个为1 |

表3-2　商品管理员表（Admin）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 管理员账号 | Adm\_no | char(20) | PK,not null | 主键 |
| 管理员密码 | Adm\_pass | Varbinary(100) | NOT NULL | 用来保存加密后的密码，注册时密码长度必须介于6到12之间 |
| 姓名 | Adm\_Rname | Varchar(10) | NOT NULL | 管理员的真实姓名 |
| 身份证号 | Adm\_ID | char(18) | CHECK(LEN(Adm\_ID)=18),not NULL | 身份证号码必须为18位 |
| 用户邮箱 | Adm\_eml | varchar(20) | CHECK(Adm\_eml like '%@%') | 邮箱必须含有‘@’符号 |
| 手机号 | Adm\_tel | char(16) | UNIQUE,NOT NULL | 手机号每个管理员，都要不相同 |
| 性别 | Adm\_sex | Nvarchar(1) | CHECK(Adm\_sex in('男','女')),NOT NULL | 性别只能是男或者女 |

表3-3　拍卖表（Auction）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 拍卖号 | Au\_ID | Int | PK,not null | 主键,自增长，步长为1 |
| 商品号 | Cs\_no | char(30) | FK,NOT NULL | 外键，拍卖的商品 |
| 拍卖开始时间 | Au\_Btim | datetime | check(Datename(minute,Au\_Btim)=30 or Datename(minute,Au\_Btim)=0)，NOT NULL | 2000-01-01 00：00：00 |
| 拍卖结束时间 | Au\_Etim | datetime | check(Datename(minute,Au\_Btim)=30 or Datename(minute,Au\_Btim)=0)，NOT NULL | 2000-01-01 00：00：00 |
| 当前最高价格 | Au\_hmon | money |  | 当前的最高出价,初始值为底价 |
| 当前最高价用户账号 | Cus\_no | char(10) | FK | 当前的最高出价人，外键，初始为空 |

表3-4　商品管理信息表（Cmana）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 商品管理序号 | Cm\_ID | Int | PK,not null | 主键,自增长，步长为1 |
| 商品进货数量 | Cm\_num | Int |  | 商品的进货数量，插入后自动加入对应商品的库存中 |
| 进货时间 | Cm\_tim | datetime | default(getdate()) | 2000-01-01 00：00：00  若无输入填入系统时间 |
| 商品号 | Gs\_no | char(30) | FK,not NULL | 外键，商品表修改时这也会修改 |
| 进货管理人员 | Cus\_no | char(10) | FK,Not NULL | 外键，获取进货的商家 |

表3-5　用户表（Customer）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 用户账号 | Cus\_no | char(10) | PK,not null | 主键 |
| 用户密码 | Cus\_pass | Varbinary(100) | NOT NULL | 用来保存加密后的密码，注册时密码长度必须介于6到12之间 |
| 用户昵称 | Cus\_Nname | Nvarchar(12) | NOT NULL | 用户给他人显示的昵称 |
| 用户地区 | Cus\_reg | nvarchar(8) |  | 用户所在地 |
| 注册手机号 | Cus\_tel | char(16) | UNIQUE,NOT NULL | 注册手机号每个用户，都要不相同 |
| 真实姓名 | Cus\_Rname | varchar(10) |  | 用户的名字 |
| 身份证号 | Cus\_ID | char(18) | CHECK(LEN(Adm\_ID)=18),not NULL | 身份证号码必须为18位 |
| 用户邮箱 | Cus\_eml | varchar(20) | CHECK(Adm\_eml like '%@%') | 邮箱必须含有‘@’符号 |
| 性别 | Cus\_sex | Nvarchar(1) | CHECK(Cus\_sex in('男','女')),NOT NULL | 性别只能是男或者女 |
| 出生日期 | Cus\_bdate | date |  | 2000-01-01 |
| 用户积分 | Cus\_fra | int |  | 初始默认为0 |
| 用户类型 | Cus\_Gar | Nvarchar(5) |  | 注册默认为普通用户 |

表3-6　配送信息表（Delivery）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 快递单号 | De\_ID | char(30) | PK,not null | 主键 |
| 发货时间 | De\_Btim | datetime |  | 2000-01-01 00：00：00 |
| 发货地 | De\_adr | Varchar(20) |  | 商品出发的地方 |
| 收货时间 | De\_Etim | datetime |  | 2000-01-01 00：00：00 |
| 快递员 | De\_peo | char(10) |  | 负责本单的快递员 |
| 物流信息 | De\_vel | Varchar(50) |  | 物流的及时信息 |

表3-7　打折商品专柜表（Discount）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 打折开始时间 | Di\_Btim | datetime | PK,not null | 主键,2000-01-01 00：00：00 |
| 商品号 | Gs\_no | char(10) | PK,FK,not null | 和开始时间为联合主键 |
| 打折结束时间 | Di\_Etim | datetime | Not null | 2000-01-01 00：00：00 |
| 打折后商品价格 | Di\_mon | money |  | 单件打折商品的价格 |

表3-8　商品评价表（Evaluate）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 中文 | 列名 | | 数据类型 | | 约束 | 描述 |
| 订单号 | Or\_ID | | char(30) | | PK,FK,not null | 主键和外键，只有当订单完成并签收才能评价 |
| 评价时间 | ev\_tim | | datetime | | default(getdate()) | 2000-01-01 00：00：00，若无输入填入系统时间 |
| 评价内容 | ev\_con | | Varchar(60) | |  | 对这个商品的评价 |
| 追加评价 | ev\_mcon | | varchar(60) | |  | 对商品的追加评价 |
|  | |  | |

表3-9　商品表（Goods）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 商品号 | Gs\_no | char(30) | PK,not null | 主键，由时间等信息生成 |
| 卖家 | Gs\_seller | char(10) | FK | 外键，该商品对应的商家 |
| 商品出售方式 | Gs\_type | Nvarchar(4) | check([Gs\_type]='兑换' OR [Gs\_type]='二手' OR [Gs\_type]='正常')，NOT NULL | 判断商品是什么商品，兑换商品还是二手商品还是普通的商品 |
| 商品名称 | Gs\_name | nvarchar(20) | NOT NULL | 货物的名字 |
| 生产厂家 | Gs\_fac | varchar(12) |  | 商品的生产厂家 |
| 商品价格 | Gs\_mon | money |  | 商品的单价 |  |  |  |  |
| 商品介绍 | Gs\_inf | varchar(30) |  | 商品的基本介绍 |
| 是否为进口 | Gs\_imp | Char(1) |  | 为1时表示为进口商品 |
| 具体规格 | Gs\_sta | varchar(20) |  | 商品大小容量重量等信息 |
| 适用人群 | Gs\_sui | varchar(20) |  | 商品对应的人群 |
| 注意事项 | Gs\_note | varchar(20) |  | 使用时需要注意的地方 |
| 商品预览图 | Gs\_jps | image |  | 图片 |
| 库存 | Gs\_num | int |  | 商品的可出售数量 |
| 商品分类名称 | Gc\_name | nvarchar(10) | Not NULL | 商品的类别 |
| 商品分类介绍 | Gc\_inf | varchar(30) |  | 该类别的共性 |

表3-10　订单表（Order）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 订单号 | Or\_ID | Char(30) | PK,not null | 主键,生成订单与时间有关 |
| 用户账号 | Cus\_no | char(10) | NOT NULL | 购买的用户，订单生成时需要参照用户，当用户修改时也会随着改，删除时保留 |
| 商品号 | Gs\_no | char(30) | NOT NULL | 购买的商品，订单生成时需要参照商品，当商品修改时也会随着改，删除时保留 |
| 购买数量 | Or\_num | int |  |  |
| 支付平台 | Or\_pla | nvarchar(6) |  | 可用于判断表格 |
| 用户备注 | Or\_inf | varchar(30) |  | 订单中的备注 |
| 配送地址信息 | Add\_no | Int | not NULL | 获取配送地址信息,订单生成时需要参照地址，当地址修改时也会随着改，删除时保留 |
| 快递单号 | De\_ID | char(30) | FK | 外键，获取配送单号 |
| 拍卖号 | Au\_ID | Int | FK | 外键，判断是否是拍卖商品 |
| 下单时间 | Or\_tim | Datetime | default(getdate()) | 2000-01-01 00:00:00，为空时获取系统时间 |
| 商品名 | Gs\_name | Nvarchar(20) |  | 适当的冗余获取商品名称 |

表3-11　购物车表（Shopping Cart）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文 | 列名 | 数据类型 | 约束 | 描述 |
| 用户账号 | Cus\_no | Char(10) | PK,FK,not null | 主键 |
| 商品号 | Gs\_no | char(10) | PK,FK,not null | 和用户账号为联合主键 |
| 商品数量 | SC\_num | int |  | 购买商品的数量 |

# 第二章 概念结构设计

## 系统模块设计图

如图7功能模块图所示，本系统主要分系统用户功能模块和管理员功能模块，其各个模块功能如下：

* 管理员模块

1. 用户管理：管理员通过账号进行用户信息的增删改查。
2. 商品管理：管理员通过账号进行用户部分信息的增删改查。
3. 订单管理：管理员通过账号进行订单部分信息的增删改查。

* 系统用户模块
* 商家模块

1. 商品上下架与库存修改：商家可以根据情况进行商品的上下架、以及商品的进货修改库存。
2. 选择商品进行折扣销售：商家可以根据情况选择部分商品进行折扣销售。
3. 选择商品进行折扣销售：商家可以根据情况选择部分商品进行拍卖销售。

* 普通用户模块

1. 用户注册：用户可以输入信息创建系统的账号。
2. 用户登录：用户根据注册的账号登录，使用系统。
3. 检索商品：用户可以根据自己的需求，模糊搜索商品名来找到心仪的商品。
4. 竞拍商品：用户可以自由竞拍已上架的竞拍商品，竞拍时间结束出价最高者得到商品。
5. 查询订单：用户可以查询自己的订单记录，并观看订单的细节信息以及配送物流信息等。
6. 兑换积分商品：用户在商场消费可以获得积分，用户可以通过获取的积分进行积分商品的兑换。
7. 查询折扣商品：用户可以进行当前时间段打折商品的查询，从中挑选商品能更加便宜地进行消费。
8. 创建订单购买商品：用户确定好购买商品后，可以进行创建订单，系统生成订单，用户进行支付，支付成功则购买成功。

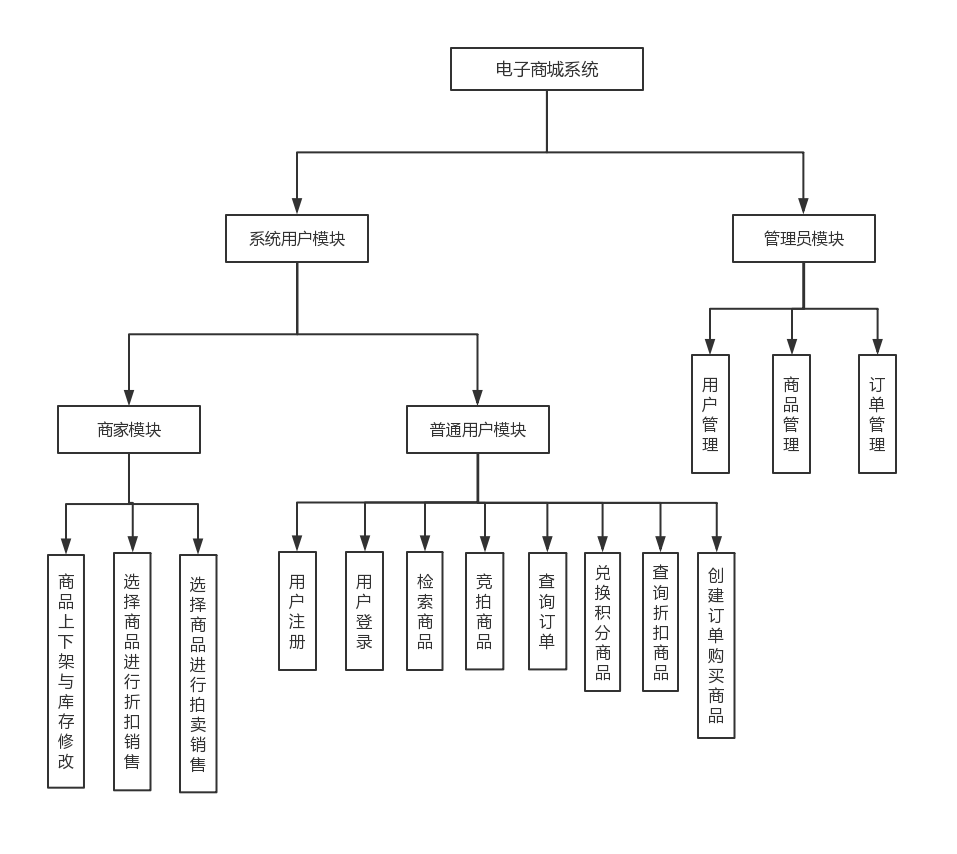


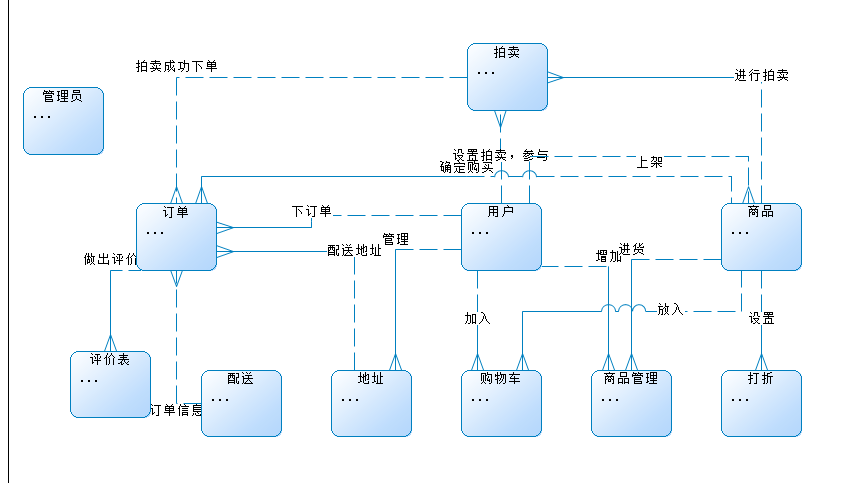
图 7 功能模块图

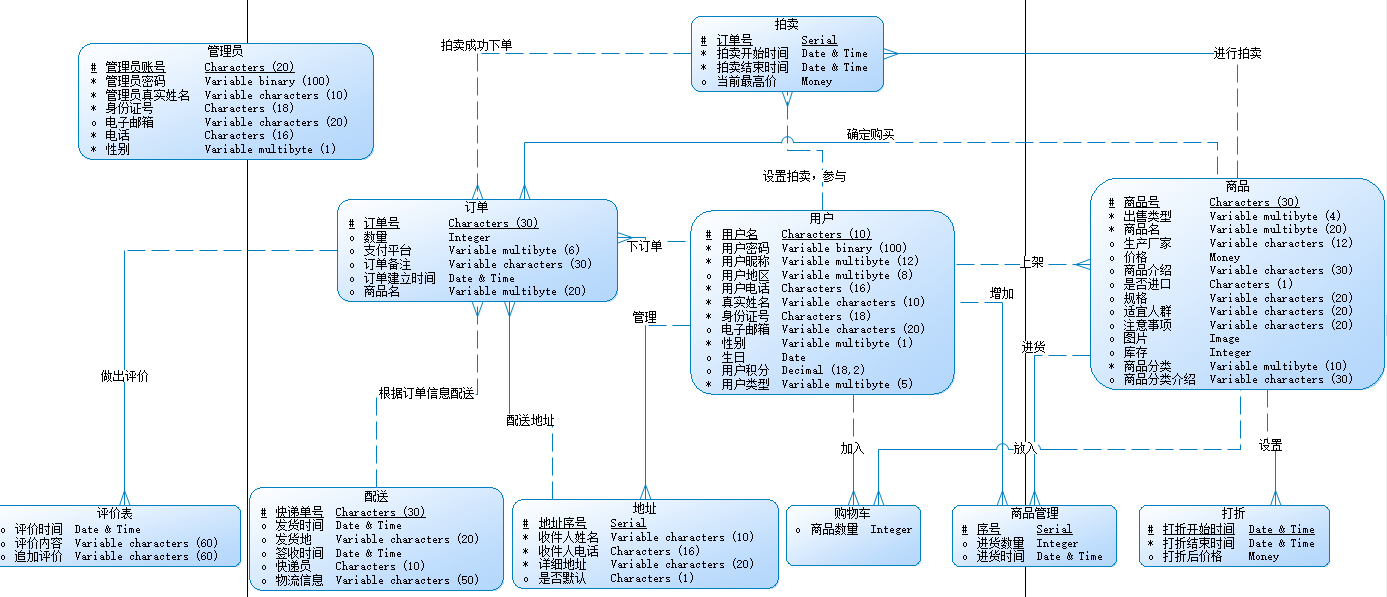
## 概念数据模型（CDM）

文件：



图片：





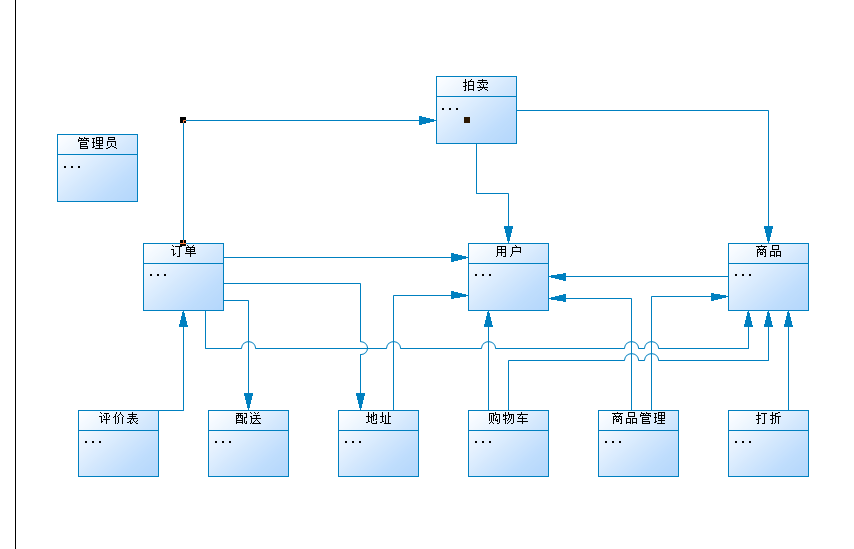
# 第三章 逻辑结构设计

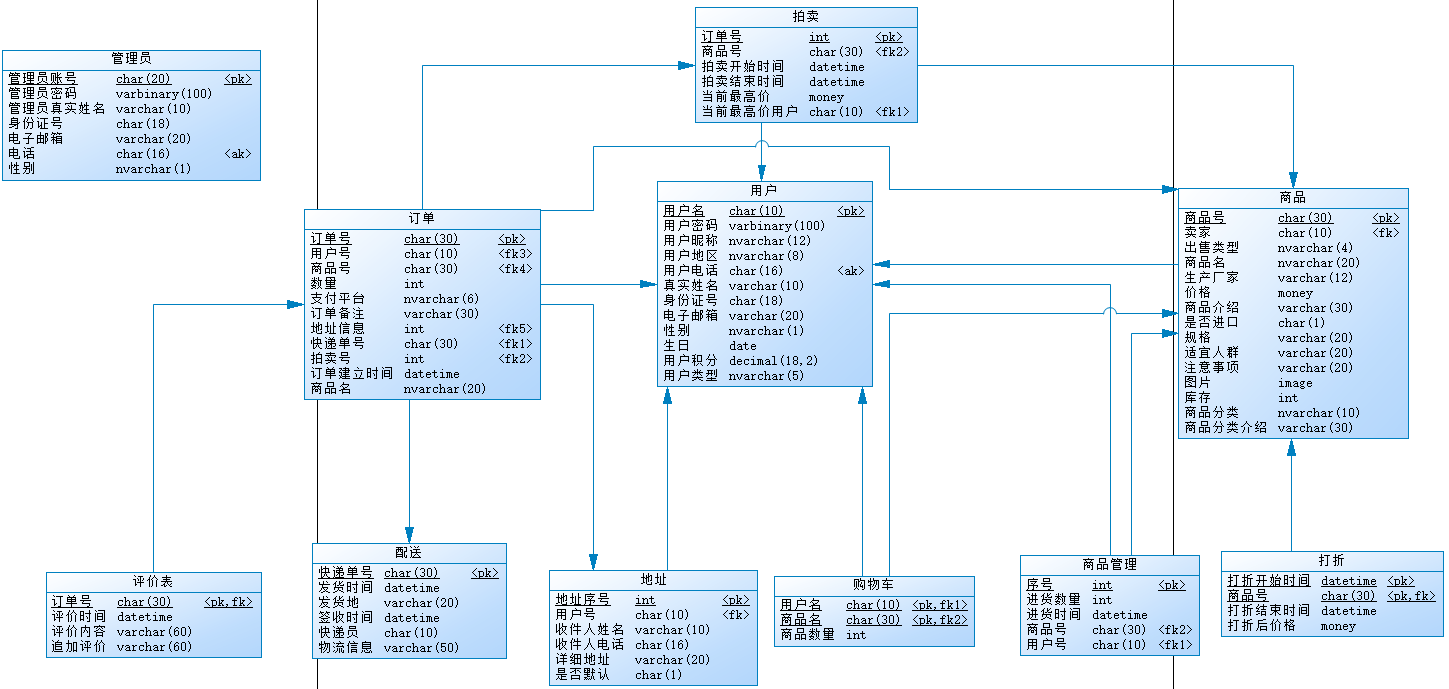
## 物理数据模型（PDM）

文件：



图片：





## 视图设计及数据库安全性设计

**视图设计：**

为每一个创建账号的系统用户创建以其账号命名的视图，具体包括：属于用户购买的订单表视图，属于用户购物车的购物车表视图，属于用户评价的评价表视图，属于用户地址的地址表视图，属于用户配送信息的配送信息表视图，属于商家出售的订单表视图，属于商家管理的商品管理表视图，属于商家添加的折扣商品表视图，属于商家添加的拍卖商品表视图。

视图创建代码：

select @user=Cus\_no,@Gar=Cus\_Gar from inserted

if @user is not null

begin

if @Gar='普通用户'

begin

exec('grant select on Goods to '+@user)

exec('grant select on Discount to '+@user)

exec('grant select on Evaluate to '+@user)

exec('grant select,insert on Auction to '+@user)

--可以查看并修改自己的购物车

set @sqlview='V1\_'+@user

exec('create view '+@sqlview+'

as

select \* from [Shopping Cart]

where Cus\_no='''+@user+'''')

exec('grant select,update on '+@sqlview+' to '+@user)

--可以查看自己的订单

set @sqlview='V2\_'+@user

exec('create view '+@sqlview+'

as

select \* from [Order]

where Cus\_no='''+@user+'''')

exec('grant select on '+@sqlview+' to '+@user)

--可以查看自己的评价，并修改

set @sqlview='V3\_'+@user

exec('create view '+@sqlview+'

as

select \* from Evaluate

where Cus\_no='''+@user+'''')

exec('grant select,update on '+@sqlview+' to '+@user)

--可以查看自己的地址，并修改

set @sqlview='V4\_'+@user

exec('create view '+@sqlview+'

as

select \* from Address

where Cus\_no='''+@user+'''')

exec('grant select,update on '+@sqlview+' to '+@user)

--可以查看自己的配送信息

set @sqlview='V5\_'+@user

exec('create view '+@sqlview+'

as

select \* from Delivery

where De\_ID in ( select De\_ID

from [Order]

where Cus\_no='''+@user+''')')

exec('grant select on '+@sqlview+' to '+@user)

end

else if @Gar='店家'

begin

--可以查看自己的订单

set @sqlview='V1\_'+@user

exec('create view '+@sqlview+'

as

select \* from [Order]

where Gs\_no in ( select Gs\_no

from Goods

where Cus\_no='''+@user+''')')

exec('grant select on '+@sqlview+' to '+@user)

exec('grant select,update on '+@sqlview+' to '+@user)

--可以查看自己的商品管理记录

set @sqlview='V2\_'+@user

exec('create view '+@sqlview+'

as

select \* from Cmana

where Cus\_no='''+@user+'''')

exec('grant select,insert on '+@sqlview+' to '+@user)

--可以查看,修改，增加自己的打折商品

exec('create view '+@sqlview+'

as

select \* from Discount

where Gs\_no in ( select Gs\_no

from Goods

where Gs\_seller='''+@user+''')')

exec('grant select,update,insert on '+@sqlview+' to '+@user)

--可以查看,修改，增加自己的拍卖商品

exec('create view '+@sqlview+'

as

select \* from Evaluate

where Gs\_no in ( select Gs\_no

from Goods

where Gs\_seller='''+@user+''')')

exec('grant select,update,insert on '+@sqlview+' to '+@user)

end

**数据库安全性设计：**

* 对于每一个普通用户，GRANT属于其购买的订单表视图的select权限，订单表的insert权限，属于其购物车的购物车表视图的select、update权限，订单表的insert权限，属于其评价的评价表视图的select、update权限，评价表的insert权限，属于其地址的地址表视图select、update权限，地址信息表的insert权限，属于其配送信息的配送信息表视图的select权限。
* 对于每一个商家，赋予其普通用户相同权限的基础上，GRANT属于其出售的订单表视图的select、update权限，订单表的insert权限，属于其管理的商品管理表视图的select权限，商品管理表的insert权限，属于其添加的折扣商品表视图的select、update权限，折扣商品表的insert权限，属于其添加的拍卖商品表视图的select、update权限，拍卖商品表的insert权限。

SQL代码：

select @user=Cus\_no,@Gar=Cus\_Gar from inserted

if @user is not null

begin

if @Gar='普通用户'

begin

exec('grant select on Goods to '+@user)

exec('grant select on Discount to '+@user)

exec('grant select on Evaluate to '+@user)

exec('grant select,insert on Auction to '+@user)

--可以查看并修改自己的购物车

exec('grant select,update on '+@sqlview+' to '+@user)

--可以查看自己的订单

exec('grant select on '+@sqlview+' to '+@user)

--可以查看自己的评价，并修改

exec('grant select,update on '+@sqlview+' to '+@user)

--可以查看自己的地址，并修改

exec('grant select,update on '+@sqlview+' to '+@user)

--可以查看自己的配送信息

exec('grant select on '+@sqlview+' to '+@user)

end

else if @Gar='店家'

begin

--可以查看自己的订单

exec('grant select,update on '+@sqlview+' to '+@user)

--可以查看自己的商品管理记录

exec('grant select,insert on '+@sqlview+' to '+@user)

--可以查看,修改，增加自己的打折商品

exec('grant select,update,insert on '+@sqlview+' to '+@user)

--可以查看,修改，增加自己的拍卖商品

exec('grant select,update,insert on '+@sqlview+' to '+@user)

end

* 对于每一个管理员，将其加入GRANT了所有表的select、update、insert权限的数据库角色——’Role\_Admin’。

SQL代码：

exec sp\_addrole 'Role\_Admin'

grant select,update,delete,insert on Customer to Role\_Admin

grant select,update,delete,insert on [Address] to Role\_Admin

grant select,update,delete,insert on Goods to Role\_Admin

grant select on [Admin] to Role\_Admin

grant select,update,delete,insert on [Shopping Cart] to Role\_Admin

grant select,update,delete,insert on [Order] to Role\_Admin

grant select,update,delete,insert on Discount to Role\_Admin

grant select,update,delete,insert on Evaluate to Role\_Admin

grant select,update,delete,insert on Auction to Role\_Admin

grant select,update,delete,insert on Cmana to Role\_Admin

grant select,update,delete,insert on Delivery to Role\_Admin

# 第四章 物理结构设计

## 4.1选择合适的DBMS

我们使用的DBMS是RDBMS，即关系数据库，使用的软件是微软公司的SQL server。使用它的原因是因为它有很多优点：  
 1、真正的客户服务器体系结构。  
 2、图形化用户界面，更加直观、简单。   
 3、丰富的编程接口工具，为用户进行程序设计提供更多选择余地。  
 4、SQLserver和Windows NT完成集成，可以利用NT的愈多功能。  
 5、对web技术的支持，使用户能够容易的将数据库中的数据发布到web上。方便连接前端与数据库。  
 6、我们的电脑使用的就是windows操作系统，软件的获取和使用可以直接在自己的电脑上进行。  
 7、平时的教学过程中对于SQL Server的使用也是最多的，对于它的建立数据库以及对数据库的操作较为熟悉。

## 4.2定义数据库、表及字段的命名规范

所选的DBMS系统选择合适的字段类型详情见表3-1到表3-11。

SQL代码：

--用户表

CREATE TABLE Customer (

Cus\_no char(10) NOT NULL, --用户账号

Cus\_pass varbinary(100) NOT NULL , --CHECK(LEN(Cus\_pass)>=6 and LEN(Cus\_pass)<=12),(在存储过程里面实现）

Cus\_Nname nvarchar(12) NOT NULL, --用户昵称

Cus\_reg nvarchar(8) NULL, --用户地区

Cus\_tel char(16) UNIQUE NOT NULL, --注册手机号

Cus\_Rname varchar(10) NOT NULL, --真实姓名

Cus\_ID char(18) CHECK(LEN(Cus\_ID)=18) not NULL,--身份证号

Cus\_eml varchar(20) CHECK(Cus\_eml like '%@%'),--用户邮箱

Cus\_sex nvarchar(1) CHECK(Cus\_sex in('男','女')) Not NULL,--性别

Cus\_bdate date ,--出生日期

Cus\_fra decimal(18, 2), --用户积分

Cus\_Gar nvarchar(5) CHECK(Cus\_Gar='普通用户' OR Cus\_Gar='店家') Not NUlL,--用户类型

CONSTRAINT PK\_Customer PRIMARY KEY(Cus\_no)

);

--地址信息

CREATE TABLE [Address] (

Add\_no int identity(1,1) NOT NULL, --序号

Cus\_no char(10) NOT NULL, --用户账号

Add\_Nname varchar(10) NOT NULL, --收件人名字

Add\_tel char(16) NOT NULL, --收件人电话

Add\_inf varchar(20) NOT NULL, --地址信息

Add\_def char(1) NULL,--是否为默认地址，是则填1

CONSTRAINT PK\_Add PRIMARY KEY(Add\_no)

);

--商品管理员

CREATE TABLE [Admin] (

Adm\_no char(20) primary key not null, --管理员帐号，主键

Adm\_pass varbinary(100) NOT NULL, --CHECK(LEN(Adm\_pass)>=6 and LEN(Adm\_pass)<=12)存储过程里面实现 --管理员密码

Adm\_Rname varchar(10) NOT NULL, --姓名

Adm\_ID char(18) CHECK(LEN(Adm\_ID)=18) not NULL,--身份证号

Adm\_eml varchar(20) CHECK(Adm\_eml like '%@%'),--用户邮箱

Adm\_tel char(16) UNIQUE NOT NULL, --手机号

Adm\_sex nvarchar(1) CHECK(Adm\_sex in('男','女')) Not NULL,--性别

) ;

--商品表

CREATE TABLE Goods (

Gs\_no char(30) primary key not NULL, --商品号

Gs\_seller char(10), --卖家

Gs\_type nvarchar(4) check([Gs\_type]='兑换' OR [Gs\_type]='二手' OR [Gs\_type]='正常') not null,--商品出售方式

Gs\_name nvarchar(20) not null, --商品名称

Gs\_fac varchar(12), --生产厂家

Gs\_mon money , --商品价格

Gs\_inf varchar(30), --商品介绍

Gs\_imp char(1) NULL, --是否为进口，是则填1

Gs\_sta varchar(20), --具体规格

Gs\_sui varchar(20), --适宜人群

Gs\_note varchar(20), --注意事项

Gs\_jps image, --商品预览图

Gs\_num int, --库存

Gc\_name nvarchar(10) not NULL,--商品分类名称

Gc\_inf varchar(30) --商品分类介绍

foreign key(Gs\_seller) references Customer(Cus\_no)ON UPDATE CASCADE, --外键

);

--购物车表

CREATE TABLE [Shopping Cart] (

Cus\_no char(10) NOT NULL, --用户账号

Gs\_no char(30) not NULL, --商品号

SC\_num int, --商品数量

CONSTRAINT PK\_SC PRIMARY KEY (Cus\_no, Gs\_no), --主键

foreign key(Cus\_no) references Customer(Cus\_no), --外键

foreign key(Gs\_no) references Goods(Gs\_no) --外键

);

--配送信息

CREATE TABLE Delivery (

De\_ID char(30) PRIMARY KEY NOT NULL, --快递单号

De\_Btim datetime , --发货时间

De\_adr varchar(20), --发货地

De\_Etim datetime , --收货时间

De\_peo char(10), --快递员

De\_vel varchar(50), --物流信息

);

--拍卖表

CREATE TABLE Auction (

Au\_ID int identity(1,1) PRIMARY KEY NOT NULL, --拍卖号

Gs\_no char(30) not null, --商品号

Au\_Btim datetime not null check(Datename(minute,Au\_Btim)=30 or Datename(minute,Au\_Btim)=0), --拍卖开始时间

Au\_Etim datetime not null check(Datename(minute,Au\_Etim)=30 or Datename(minute,Au\_Etim)=0), --拍卖结束时间

Au\_hmon money, --当前最高价格

Cus\_no char(10) , --当前最高价用户账号

foreign key(Gs\_no) references Goods(Gs\_no), --外键

foreign key(Cus\_no) references Customer(Cus\_no), --外键

);

--订单表

CREATE TABLE [Order] (

Or\_ID char(30) PRIMARY KEY NOT NULL, --订单号

Cus\_no char(10) not null, --用户账号

Gs\_no char(30) not null, --商品号

Or\_num int, --购买数量

Or\_pla nvarchar(6), --支付平台

Or\_inf varchar(30), --用户备注

Add\_no int NOT NULL, --配送地址序号

De\_ID char(30) foreign key references Delivery(De\_ID), --快递单号

Au\_ID int foreign key references Auction(Au\_ID), --拍卖号

Or\_tim datetime default(getdate()),--建立订单时获取系统时间

Gs\_name nvarchar(20) , --商品名称

);

--打折商品专柜

CREATE TABLE Discount (

Di\_Btim datetime NOT NULL, --打折开始时间

Gs\_no char(30) not NULL, --商品号

Di\_Etim datetime NOT NULL, --打折结束时间

Di\_mon money, --打折后商品价格

CONSTRAINT PK\_Di PRIMARY KEY (Di\_Btim, Gs\_no), --主键

foreign key(Gs\_no) references Goods(Gs\_no)ON UPDATE CASCADE, --外键

);

--商品评价表

CREATE TABLE Evaluate (

Or\_ID char(30) not NULL, --订单号

ev\_tim datetime default(getdate()), --评价时间

ev\_con varchar(60), --评价内容

ev\_mcon varchar(60), --追加评价

CONSTRAINT PK\_EV PRIMARY KEY (Or\_ID), --主键

foreign key(Or\_ID) references [Order](Or\_ID), --外键

);

exec proc\_ordercreate 'xxx3','2020xxx10618182428393000',1,1,'支付宝'

select \* from Evaluate where Or\_ID in (select Or\_ID from [Order] where Gs\_no='2020xxx10618182344700000 ')

--商品管理信息

CREATE TABLE Cmana (

Cm\_ID int identity(1,1) PRIMARY KEY NOT NULL, --序号

Cm\_num int, --商品进货数量

Cm\_tim datetime default(getdate()), --进货时间

Gs\_no char(30) not null, --商品号

Cus\_no char(10) not null, --用户帐号

foreign key(Gs\_no) references Goods(Gs\_no)ON UPDATE CASCADE, --外键

foreign key(Cus\_no) references Customer(Cus\_no), --外键

);

## 4.3 反范式化设计

1.商品表Goods关系模式所能达到的最高范式为2NF,因为存在商品号->商品分类名称，商品分类名称->商品分类介绍的传递依赖。产生了数据冗余，商品分类介绍字段放在商品表能够使得每一次显示商品信息的时候不需要额外产生连接其他表来显示分类信息的开销。

2.订单表[Order]关系模式所能达到的最高范式为2NF，因为存在订单号->拍卖号，拍卖号->商品号，拍卖号->用户账号的传递依赖以及订单号->商品号，商品号->商品名称的传递依赖。产生了数据冗余，但是好处有如下两点：一、每一次显示订单信息的时候不需要将订单表与商品表两张大表进行连接，降低了查询的开销；二、可以通过拍卖号字段是否为空来进行该商品的交易是拍卖的方式还是正常的方式。

# 第五章 系统实现与测试

## 5.1 系统实现功能

用户注册

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用存储过程proc\_reguser创建用户。  *sf*=**new** Configuration().configure("myhiber.cfg.xml").buildSessionFactory();  Session session=*sf*.openSession();  String procName="";  procName="exec proc\_reguser '"+st.getCus\_no()+"','"+st.getCus\_pass()+"','"+st.getCus\_Nname()+"','"+st.getCus\_reg()+"','"+st.getCus\_tel()+"','"+st.getCus\_Rname()+"','"+st.getCus\_ID()+"','"+st.getCus\_eml()+"','"+st.getCus\_sex()+"','"+st.getCus\_bdate()+"',"+st.getCus\_fra()+",'"+st.getCus\_Gar()+"'";  ~~SQLQuery~~ sqlquery=session.createSQLQuery(procName);  List list =sqlquery.~~list~~(); |
| 1. 存储过程proc\_reguser密码的加密，用户的插入，账号的创建。   select '创建成功！'  exec('exec sp\_addlogin '+@Cus\_no+','''+@Cus\_pass+'''')  exec('exec sp\_adduser '+@Cus\_no+','+@Cus\_no)  OPEN SYMMETRIC KEY IDKEY DECRYPTION BY CERTIFICATE Cert1; --打开密钥  declare @pass varbinary(100)  set @pass=ENCRYPTBYKEY(Key\_GUID('IDKEY'), cast(@Cus\_pass as varbinary(100)))  insert into Customer(Cus\_no,Cus\_pass,Cus\_Nname,Cus\_reg,Cus\_tel,Cus\_Rname,Cus\_ID,Cus\_eml,Cus\_sex ,Cus\_bdate,Cus\_fra,Cus\_Gar)  values(@Cus\_no,@pass,@Cus\_Nname,@Cus\_reg,@Cus\_tel,@Cus\_Rname,@Cus\_ID,@Cus\_eml,@Cus\_sex ,@Cus\_bdate,@Cus\_fra,@Cus\_Gar)  CLOSE SYMMETRIC KEY IDKEY;  4.触发器trig\_Cu创建对应的视图以及GRANT权限 |
| 5.用户创建成功。 |

* 用户登录

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用存储过程proc\_enpycus获得用户的密码。  *sf*=**new** Configuration().configure("myhiber.cfg.xml").buildSessionFactory();  Session session=*sf*.openSession();  String procName="";  **if**(type.equals("admin")){  procName="exec proc\_enpyadm '"+id+"'";  }  **else**{  procName="exec proc\_enpycus '"+id+"'";  }  System.***out***.println(procName);  ~~SQLQuery~~ sqlquery=session.createSQLQuery(procName);  List list =sqlquery.~~list~~();  session.close();  **return** (String) list.get(0); |
| 3、存储过程proc\_enpycus获得用户的密码。  OPEN SYMMETRIC KEY IDKEY DECRYPTION BY CERTIFICATE Cert1; --打开密钥  declare @pass varbinary(100)  set @pass=ENCRYPTBYKEY(Key\_GUID('IDKEY'), cast(@Cus\_pass as varbinary(100)))  select @pass  CLOSE SYMMETRIC KEY IDKEY; |
| 1. web判断密码是否相同，相同则登录到主页面：     不同则显示：并返回登陆界面。 |

* 商品检索

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate通过hql语句获取对应结果集并通过处理返回。  String hql;  **if** (id == "all")  hql = "from bean.Goods where Gs\_type!='兑换'";  **else**  hql = "from bean.Goods where Gs\_name like '%" + id + "%' and Gs\_type!='兑换'";  List<Goods> r = (List<Goods>) Hiber\_DButil.*executeHQL*(hql); |
| 3、存储过程proc\_usersear获得相同的结果。  --用户关键字查询商品  create proc proc\_usersear(@sear nvarchar(20))--利用通配符模糊查找  as  exec('select \* from Goods  where Goods.Gs\_name like ''%'+@sear+'%''');  go |
| 4、处理过的结果显示：  发现界面中没有库存的商品背景为红色且无加入购物车以及购买的选项。  QQ截图20200623142528 |

* 用户将商品加入购物车

|  |
| --- |
| 1、通过界面用户进行操作。 |
| 2、web前端通过hibernate通过调用存储过程proc\_addcar。  String hql = "exec proc\_addcar '" + Customerid + "','" + id + "'";  Hiber\_DButil.*execproc*(hql); |
| 3、存储过程proc\_usersear将商品加入当前用户的购物车。  create proc proc\_addcar(@Cus\_no char(10),@Gs\_no char(30))  as  if((select count(\*) from [Shopping Cart] where @Cus\_no=Cus\_no and @Gs\_no=Gs\_no)=1)  update [Shopping Cart]  set SC\_num=SC\_num+1;  else  insert into [Shopping Cart]  values(@Cus\_no,@Gs\_no,1)  select '加入成功！'  GO |
| 4、页面显示表示加入购物车成功  购物车中的信息增加了 |

* 用户创建订单

|  |
| --- |
| 1、通过界面用户进行操作。 |
| 2、web前端通过hibernate通过调用存储过程proc\_ordercreate。  String hql;  **if** (pos > 0) {  hql = "exec proc\_ordercreate '" + userid + "','" + id + "'," + num + "," + pos + ",'" + pla + "'";  } **else** {  hql = "exec proc\_ordercreate '" + userid + "','" + id + "'," + num + ",null,'" + pla + "'";  }  String s = Hiber\_DButil.execproc(hql); |
| 3、存储过程proc\_ordercreate通过传入的参数，利用事务创建订单。  create proc [dbo].[proc\_ordercreate](@Cus\_no char(10),@Gs\_no char(30),@num int,@Add\_no char(10),@pla nvarchar(6))  as  set transaction isolation level serializable  BEGIN TRANSACTION  declare @id char(30),@mon money;  set @mon=0  exec proc\_id @Cus\_no,@id output  if(@pla='官方')    if(((select Gs\_mon from Goods where Gs\_no=@Gs\_no)\*@num)<=(select Cus\_fra from Customer where @Cus\_no=Cus\_no))--判断积分够不够  begin  update Customer  set Cus\_fra=Cus\_fra-((select Gs\_mon from Goods where Gs\_no=@Gs\_no)\*@num)  where Cus\_no=@Cus\_no  end  else  begin  select ('积分不够,兑换失败！');  return ;  end  insert into [Order](Or\_ID, Cus\_no,Gs\_no,Or\_num,Add\_no,Gs\_name,Or\_pla)  values(@id,@Cus\_no,@Gs\_no,@num,isnull(@Add\_no,(select Add\_no from [Address] where Cus\_no=@Cus\_no and Add\_def='1')),(select Gs\_name from Goods where @Gs\_no=Gs\_no),@pla)    COMMIT TRANSACTION  if((select Au\_ID from [Order] where [Order].Or\_ID=@id) is not null) --是拍卖  set @mon=(select Au\_hmon from Auction where Au\_ID in (select Au\_ID from [Order] where [Order].Or\_ID=@id))  else --不是拍卖  begin  set @Gs\_no=(select Gs\_no from [Order] where Or\_ID=@id)  if(@Gs\_no in (select Gs\_no from Discount where getdate()>=Di\_Btim and getdate()<=Di\_Etim and Discount.Gs\_no=@Gs\_no))--有折扣  begin  declare @mon1 int  set @mon1=(select Discount.Di\_mon from Discount where getdate()>=Di\_Btim and getdate()<=Di\_Etim and Discount.Gs\_no=@Gs\_no)  set @mon= @mon1\*@num  end  else  set @mon= @num\*(select Gs\_mon from Goods where Gs\_no=@Gs\_no)  end  if(@pla is not null and @pla !='官方')--添加积分  begin  update Customer  set Cus\_fra=Cus\_fra+@mon\*0.1  where Cus\_no=@Cus\_no  end  select '购买成功！' |
| 4、页面显示表示购买成功。  订单表中的信息增加了    商品的库存减少了    用户积分增加了 |

* 用户查询折扣商品

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用hql语句连接数据库进行查询获得结果集。  String hql;  hql = "from bean.Goods where Gs\_no in " + " ( " + " select Gs\_no " + " from Discount "  + " where getdate()>=Di\_Btim and getdate()<=Di\_Etim " + " )";  List<Goods> r = (List<Goods>) Hiber\_DButil.*executeHQL*(hql); |
| 1. 处理后的结果显示：   QQ截图20200623143543 |

* 用户查询订单

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用hql语句连接数据库进行查询获得结果集。  String hql;  hql = "from bean.Order where Cus\_no= '" + userid + "'";  List<Order> r = (List<Order>) Hiber\_DButil.*executeHQL*(hql); |
| 1. 处理后的结果显示：   QQ截图20200623143949 |

* 用户评价商品

|  |
| --- |
| 1、通过界面用户进行操作。 |
| 2、web前端通过hibernate通过调用存储过程proc\_Evaluatenew 。  String hql = "exec proc\_Evaluatenew '" + userid + "','" + id + "','" + eva + "'";  Hiber\_DButil.*execproc*(hql); |
| 3、存储过程proc\_Evaluatenew将根据传入的参数进行商品评价表内容的修改。  --用户评价商品（包括追加）  create proc proc\_Evaluatenew(@Cus\_no char(30),@Or\_ID char(30),@evalu nvarchar(60))  as  if((select count(\*) from Evaluate where @Cus\_no=Cus\_no and @Or\_ID=Or\_ID)=0)--表示没有评价过  insert into Evaluate(Cus\_no,Or\_ID,ev\_con)  values(@Cus\_no,@Or\_ID,@evalu)--创建评价  else --表示评价过  update Evaluate  set ev\_mcon=@evalu --修改追评  where @Or\_ID=Or\_ID  select '评价成功！'  go |
| 4、页面显示  表示评价成功  商品评价表中的信息也变化了 |

* 用户查询订单物流信息

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用hql语句连接数据库进行查询获得结果集。  String hql;  hql = "from bean.Delivery where De\_ID= '" + idStr + "'";  List<Delivery> r = (List<Delivery>) Hiber\_DButil.*executeHQL*(hql); |
| 3、处理后的结果显示：  QQ截图20200623144500 |

* 用户查看购物车

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用hql语句连接数据库进行查询获得结果集。  String hql;  hql = "from bean.Shopping\_Cart where Cus\_no= '" + userid + "'";  List<Shopping\_Cart> r = (List<Shopping\_Cart>) Hiber\_DButil.*executeHQL*(hql); |
| 3、处理后的结果显示：  QQ截图20200623144723 |

* 用户竞拍商品

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用存储过程proc\_Aucprice。  String hql = "exec proc\_Aucprice '" + userid + "'," + id + "," + price;  System.***out***.println(hql);  Hiber\_DButil.*execproc*(hql); |
| 1. 存储过程proc\_Aucprice通过事务处理的方式根据传入的参数进行拍卖商品表的更新：   --用户竞拍价格（事务处理）  create proc proc\_Aucprice(@Cus\_no char(10),@id int,@mon money)--更新竞拍信息  as  set transaction isolation level serializable  BEGIN TRANSACTION  update Auction  set Au\_hmon=@mon,Cus\_no=@Cus\_no  where Au\_ID=@id  COMMIT TRANSACTION  select '竞拍成功!'  go |
| 1. 结果显示：   表示竞拍成功！  竞拍商品表中的对应数据也进行了更改。 |

* 用户兑换商品

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用存储过程proc\_ordercreate。（与之前订单创建相同） |
| 3、存储过程proc\_ordercreate通过事务处理的方式根据传入的参数创建订单：（与之前订单创建相同） |
| 4、结果显示：  表示兑换成功！  订单表中的对应数据也进行了更改。  用户的积分也减少了。 |

* 用户添加地址

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 1. web前端通过hibernate调用存储过程proc\_ordercreate。   String hql = "exec proc\_addcar '" + Customerid + "','" + id + "'";  Hiber\_DButil.*execproc*(hql); |
| 1. 存储过程proc\_ordercreate根据传入参数在地址信息表中添加地址信息。   --用户加入商品进购物车  create proc proc\_addcar(@Cus\_no char(10),@Gs\_no char(30))  as  if((select count(\*) from [Shopping Cart] where @Cus\_no=Cus\_no and @Gs\_no=Gs\_no)=1)  update [Shopping Cart]  set SC\_num=SC\_num+1;  else  insert into [Shopping Cart]  values(@Cus\_no,@Gs\_no,1)  select '加入成功！' |
| 4、结果显示：  表示添加成功！  地址表中的信息增加了。 |

* 商家进货

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用存储过程proc\_Cma。  String hql = "exec proc\_Cma '" + u.getGs\_no() + "'," + u.getCm\_num() + ",'" + u.getCus\_no() + "'";  Hiber\_DButil.*execproc*(hql); |
| 3、存储过程proc\_Cma根据传入的参数创建订单：（与之前订单创建相同）  --商家进货  CREATE proc proc\_Cma(@Gs\_no char(30),@Cm\_num int,@Cus\_no char(10))  as  if exists(select \* from Goods where @Gs\_no=Gs\_no)  begin  set transaction isolation level serializable  BEGIN TRANSACTION  insert into Cmana(Cm\_num,Gs\_no,Cus\_no)  values(@Cm\_num,@Gs\_no,@Cus\_no)  COMMIT TRANSACTION  select '建立成功!'  end  else  begin  select '没有该商品号，请先建立商品'  end |

* 商家添加折扣商品

|  |
| --- |
| 1、通过界面用户输入信息。  IMG_256IMG_257 |
| 2、web前端通过hibernate调用存储过程proc\_newDis。  String hql = "exec proc\_newDis '" + id + "','" + Cus\_no + "','" + btime + "','" + etime + "'," + mon + "";  System.***out***.println(hql);  String s = Hiber\_DButil.*execproc*(hql); |
| 3、存储过程proc\_newDis根据传入参数在地址信息表中添加地址信息。  create proc proc\_newDis(@Cus\_no char(10),@Gs\_no char(30),@btime datetime,@etime datetime,@mon money)  as  if(@Gs\_no in (select Gs\_no from Goods where @Cus\_no=Gs\_seller))--判断商品卖家是不是他  begin  insert into Discount  values(@btime,@Gs\_no,@etime,@mon)  select '添加成功！'  end  else  select '添加失败！您没用管理该商品权限'  go |
| 4、结果显示：  表示添加成功！  地址表中的信息增加了。 |

* 商家添加竞拍商品

|  |
| --- |
| 1、通过界面用户输入信息。 |
| 2、web前端通过hibernate调用存储过程proc\_newAuc。  String hql = "exec proc\_newAuc '" + id + "','" + Cus\_no + "','" + btime + "','" + etime + "'," + mon + "";  String s = Hiber\_DButil.*execproc*(hql); |
| 3、存储过程proc\_newAuc根据传入参数在地址信息表中添加地址信息。  create proc proc\_newAuc(@Cus\_no char(10),@Gs\_no char(30),@btime datetime,@etime datetime,@mon money)  as  if(@Gs\_no in (select Gs\_no from Goods where @Cus\_no=Gs\_seller))--判断商品卖家是不是他  begin  if((select Gs\_num from Goods where Gs\_no=@Gs\_no)<1)  select '添加失败，库存不足'  else  begin  insert into Auction(Gs\_no,Au\_Btim,Au\_Etim,Au\_hmon,Cus\_no)  values(@Gs\_no,@btime,@etime,@mon,null)  select '添加成功！'  update Goods  set Gs\_num=Gs\_num-1 --库存减一  end  end  else  select '添加失败！您没用管理该商品权限的'  go |
| 4、结果显示：  表示添加成功！  拍卖表中也添加了一条新记录 |

* 管理员管理商品（与商家大致相同，为了防止内容过于臃肿，就不详细演示）

|  |
| --- |
|  |
| QQ截图20200623152324 |
| QQ截图20200623152343 |

* 管理员管理订单

|  |
| --- |
| 1、通过界面用户进行操作。 |
| 2、web前端通过hibernate通过调用存储过程proc\_Ord 。  String hql = "exec proc\_Ord '" + u.getOr\_ID() + "'," + u.getOr\_num() + "," + u.getAdd\_no() + ",'"  + u.getDe\_ID() + "'";  Hiber\_DButil.*execproc*(hql); |
| 3、存储过程proc\_Ord根据传入的参数进行订单对应信息的更新  create proc proc\_Ord(@Or\_ID char(30),@Or\_num int,@Add\_no int,@De\_ID char(30))  as  if exists(select \* from [Order] where @Or\_ID=Or\_ID)  begin  update [Order] set Or\_num=@Or\_num,Add\_no=@Add\_no,De\_ID=@De\_ID where Or\_ID=@Or\_ID  end  select '修改成功!'  go |

* 管理员管理用户

|  |
| --- |
| 1、通过界面用户进行操作。 |
| 2、web前端通过hibernate通过调用存储过程proc\_deleteuser 。  String hql = "exec proc\_deleteuser '" + id + "'";  Hiber\_DButil.*execproc*(hql);  **return** 1; |
| 3、存储过程proc\_deleteuser根据传入的参数进行用户表的信息更新  create proc proc\_deleteuser(@Cus\_no char(10))  as  delete from Cmana  where Cus\_no=@Cus\_no  delete from Goods  where Gs\_seller=@Cus\_no  delete from [Shopping Cart]  where Cus\_no=@Cus\_no  delete from Customer  where Cus\_no=@Cus\_no  select '删除成功!'  go |

# 第六章 优缺点自我评价

## 6.1系统设计的自我评价

**黄勋：**整个系统设计的功能总体上还是比较完整的，是一个功能比较完善的在线购物系统，能够在管理员-顾客-商家三者之间通过各自在界面上的操作进行正确的交易，在最基础的电子商城系统上添加了商品的折扣销售、竞拍销售两种新的销售方式，在数据库的设计方面考虑到了图片存储、加密和解密、角色权限设置等多个补充要求。然而，对于这个系统本身还是有很多的不足，例如：没有对购物车进行设计出批量进行购买的功能，没有对数据库的性能优化进行测试，图片上传方式较麻烦，无法真正实现付款的流程而是仅仅填一个支付平台，由于前端技术不精湛无法设计出用户体验感更好、更美观的界面，以及最终并没有尝试着去使用redis来缓存常用表信息来提高系统的查询效率，这些不足都是以后的设计值得再次注意并完善的。

**周圣博：**本次实验我主要负责的是数据库建表，建立触发器和前端管理员加上商家的部分操作。本次实验早早开始准备。这个实验到目前为止的表中的元素都是变化了若干次的，到了后来甚至我觉得我们的表不应该那么写了，但是由于系统的大致框架已经构建好，不好做太大的修改了，所以我们只能退而求其次，只对表做一些小层次上的修改。但也因此我觉得我们的表的设计没有很完美了。也许是java做前端设计时还是比较复杂的，所以到后面有些数据库去迎合java的感觉了，导致各表之间的联系被弱化了，但主要还是围绕着数据库而设计的，java中调用了大量的数据库中的存储过程就是一个体现。如果我们的表能够保持各表之间联系的完整性，又能较好的实现前端，我觉得才是一个完美的设计，这一次的设计还是有很多不足的。

## 6.1 体会和建议

**黄勋：**通过完成本次数据库的课程设计，我对数据库的所学知识进行了一次比较笼统的复习，温故而知新，上到关系模式的建立下到范式的考虑都在与组员不断地讨论、修错的过程中得到了新的理解。从一开始的自信满满，到后来在实现数据库的过程中不断发现自身数据库的漏洞再修正，逐渐明白了一个设计好一个优秀的数据库的难度，也在这个自我纠正的过程对设计好一个数据库得以管中窥豹——“要设计好一数据库，需求分析是最不可或缺的根基”。在设计数据库的过程中，对“考虑事务和并发控制问题”和“数据库如何存储图片”这两个问题的思考也进一步拓展了我在数据库的知识面，让我学习到了一些动手实践才能明白的新知。同时，虽然本次课程设计是数据库方面，但是编写前端界面还增强了我JavaEE的代码能力。对于数据库方面的建议，一方面我觉得可以增加一些题目：例如知乎、CSDN这样的交流平台的信息管理系统、酷狗、网易云这样的音乐平台信息管理系统等，另一方面我觉得可以为每个课题增加一些固定的检索功能作为存储过程来让设计者编写，让设计者在完成这些存储过程的过程中增强自身的T-SQL代码能力。

**周圣博：**这学期的实验是我们第一次动用两个不同的方式来组成一个程序。用Java和数据库，两个有着很大不同的方式来组成一个管理系统。云计算、大数据和物联网是互联网新的发展方向，当今被看好的人工智能目前也是基于这三者之上而发展起来的。大数据更被称之为金矿。而数据库就是用来保存这些大数据的。但想要运用大数据就不能把大数据给孤立开来，就必须要和其他东西联系起来，不然就只能是无人识得的千里马了。这学期学习的java就是对数据库的一个初步调用。通过java实现数据库中有的不同的功能，比如事务、游标等等。本次实验中我也暴露出来了不少的问题，对部分java和SQL的内容还是有些生疏，所以一开始我的功能实现进度还是比较慢的，也体现出我对这些知识的实践还不够多。希望未来的我可以不断学习不断实践突破自己。  
通过实验的过程中，我也发现了数据库追求安全性的必要。数据一旦被窃取可能就是个人的隐私、社会的财产的流失。所以加密是数据库中十分重要一步。本次课设中对加密的运用我们只局限在了密码中，主要是整个数据库都进行加密会太复杂。所以我们只采取了部分加密的方式。对于数据库这门课我的建议是可以对适当得介绍一下当前工业体系中对数据库的使用。这样也有助于拓宽同学的思维往实践的方向发展。其他方面我觉得老师做得都很完善。

# 参考文献