

**数据库课程设计**

**题 目: 二手房中介管理系统**

**2022 年 4月 29日**

目录

[一、 引言 4](#_Toc102142566)

[1.1 概述 4](#_Toc102142567)

[1.2 选题背景 4](#_Toc102142568)

[1.3 需求介绍 4](#_Toc102142569)

[二、 系统的功能模块划分 5](#_Toc102142570)

[2.1 二手房中介系统业务流程 5](#_Toc102142571)

[2.2 功能模块的划分 5](#_Toc102142572)

[2.3 数据流图 6](#_Toc102142573)

[2.3.1 环境图（顶层数据流图） 6](#_Toc102142574)

[2.3.2 一层数据流图 7](#_Toc102142575)

[2.3.3 二层数据流图 7](#_Toc102142576)

[2.4 数据字典 10](#_Toc102142577)

[2.4.1 数据流词条 10](#_Toc102142578)

[2.4.2 数据元素词条 11](#_Toc102142579)

[2.4.3 数据存储文件词条 12](#_Toc102142580)

[2.4.4 加工词条 14](#_Toc102142581)

[三、 数据库概念结构设计 15](#_Toc102142582)

[3.1 单个实体的E-R图 15](#_Toc102142583)

[3.1.1 用户实体的E-R图 15](#_Toc102142584)

[3.1.2 房屋实体的E-R图 15](#_Toc102142585)

[3.1.3 员工实体的E-R图 16](#_Toc102142586)

[3.2 数据表E-R图 16](#_Toc102142587)

[3.2.1 员工信息表E-R图 16](#_Toc102142588)

[3.2.2 客户信息表E-R图 17](#_Toc102142589)

[3.2.3 房源信息表E-R图 18](#_Toc102142590)

[3.2.4 求租意向表E-R图 18](#_Toc102142591)

[3.2.5 费用信息表E-R图 18](#_Toc102142592)

[3.2.6 朝向信息表E-R图 19](#_Toc102142593)

[3.2.7 房型信息表E-R图 20](#_Toc102142594)

[3.2.8 登录信息表E-R图 20](#_Toc102142595)

[3.2.9 楼层信息表E-R图 21](#_Toc102142596)

[3.2.10 用途信息表E-R图 21](#_Toc102142597)

[3.2.11 装修信息表E-R图 22](#_Toc102142598)

[3.3 整体E-R图 23](#_Toc102142599)

[四、 数据库逻辑结构设计 24](#_Toc102142600)

[4.1 调整后的E-R图 24](#_Toc102142601)

[4.2 关系模式及规范化处理 24](#_Toc102142602)

[4.3 数据库中的表 25](#_Toc102142603)

[4.3.1 员工信息表（employee） 25](#_Toc102142604)

[4.3.2 民族信息表(gov) 25](#_Toc102142605)

[4.3.3 学历信息表(study) 25](#_Toc102142606)

[4.3.4 客户信息表（users） 26](#_Toc102142607)

[4.3.5 房源信息表（house） 26](#_Toc102142608)

[4.3.6 房型信息表(type) 26](#_Toc102142609)

[4.3.7 装修信息表(fit) 27](#_Toc102142610)

[4.3.8 朝向信息表(favor) 27](#_Toc102142611)

[4.3.9 用途信息表(method) 27](#_Toc102142612)

[4.3.10 楼层信息表(floor) 27](#_Toc102142613)

[4.3.11 求租意向表(intent) 27](#_Toc102142614)

[4.3.12 收费信息表(money) 28](#_Toc102142615)

[4.3.13 登录信息表(login) 28](#_Toc102142616)

[五、 系统实现过程及完成效果 29](#_Toc102142617)

[5.1 建表语句 29](#_Toc102142618)

[5.2 完善相应的控制语句 32](#_Toc102142619)

[5.2.1 表的控制 32](#_Toc102142620)

[5.2.2 数据插入举例 33](#_Toc102142621)

[5.2.3 视图举例 34](#_Toc102142622)

[5.2.4 索引 35](#_Toc102142623)

[5.3 可编程性 35](#_Toc102142624)

[5.3.1 触发器 35](#_Toc102142625)

[5.3.2 存储过程(举例) 46](#_Toc102142626)

[5.3.3 函数 46](#_Toc102142627)

[六、 总结 47](#_Toc102142628)

[七、 参考资料 47](#_Toc102142629)

[八、 致谢 47](#_Toc102142630)

# 引言

## 概述

随着我国经济的发展，商品流通领域进入了一个空前发展的阶段，面对商界日益激烈的竞争，企业需要一个好的管理系统才能在激烈的竞争中发展。而随着房地产业的开发和发展，二手房中介行业也随之发展起来，由于房改政策的出台，购房、售房、租房的居民越来越多，这对二手房中介部门来说无疑也是一个好的发展契机。但是与国外相对成熟的二手房市场来说，我国的二手房中介交易市场发展比较落后，管理方式也相对于信息行业的快速发展而显得比较落后，导致数据的重复率和出错率升高，办公效率低下，给客户、管理者、出售者带来了不少的麻烦。在过去，二手交易的数据往往靠人工记录，如人工结算、人工登记卡片，人工填写资料等等，销售额的日结算和月盘点等都要耗费大量的人力物力，即使采用电子表格进行管理，但也只能进行一些简单的登记和记录。不仅不能把数据之间的关系形象化的表现出来，也不能进一步掌握市场的动态。而如果使用一个二手房中介管理系统，就能方便、灵活的对数据进行操作，比如查询、统计等。我们还能对数据库给予不同的权限，保障数据库的安全性。通过此中介系统可以实现对数据的长时存储，提供高质量的统计数据等。

## 选题背景

目前我国的中小企业数量较多、地区分布广泛、行业分布跨度大，随着全球经济一体化的发展，中小企业将面临外资企业和国外产品与服务的严峻挑战，比较而言，外资企业具有更为雄厚的资金实力、丰富的管理经验和先进的技术手段，因此，如果我国的中小企业不借助先进的管理思想转变经营观念、使用信息化提高企业的管理水平和工作效率，将很难在今后的国际竞争中取胜。然而企业管理在很多方面、很大程度上都必须借助信息化来完成，而我国中小企业的信息化水平还很低，与外资企业相比，还处于起步阶段。随着市场竞争的日益加剧，对于房产中介公司来说，软件平台的缺失与不平衡性称为制约其发展的阻碍之一。

对于房屋中介企业来说，二手房中介管理系统是一款非常实用的房屋中介软件。使用该软件，不仅可以详细的记录房源信息、用户信息等，同时本系统能够自动去查找和客户需求匹配的房源，可以方便客户同时提高使用者的工作质量与效率。同时能利用房屋中介管理系统的软件信息化技术重新优化现有的二手房交易模式，实现多层次全方位的沟通，实现复杂业务模型的建立和优化。通过对流程的分析，我们也可以发现现有流程的各种弊端和缺陷，了解和优化缺失的环节。因此，通过使用二手房交易系统不仅可以提升工作效率，降低运营风险，还能促进二手房中介产业的规范化、信息化、结构化，具有重要的现实意义。

## 需求介绍

随着时代的发展，房地产业蓬勃发展，而计算机科学也发展的日益成熟。我们可以运用计算机的强大功能，给房屋中介商和房屋交易双方设计一套合理有效的二手房房屋中介管理系统。

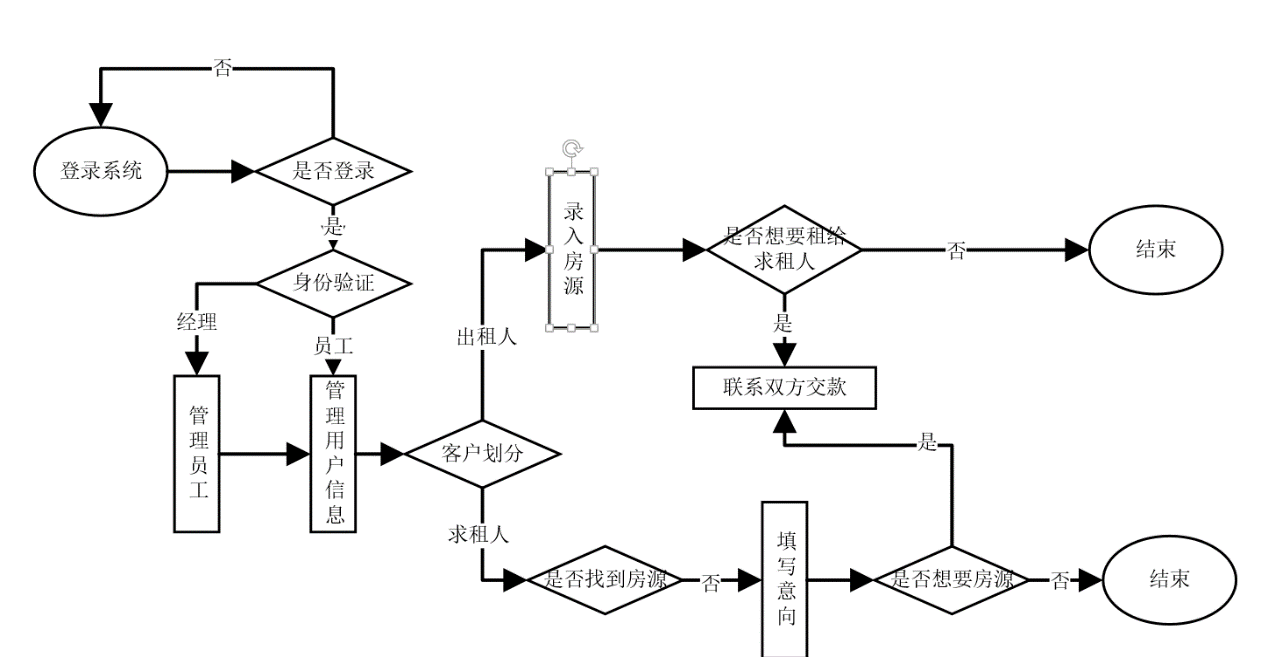
首先是用户的角度，用户分为两大类：求租人和出租人。不管是求租人还是出租人，在登录本系统时都需要先开始注册，填写个人信息保证发布信息的真实性。出租人注册完后必须紧跟出租信息，否则应该列为未注册成功。注册完成后也能修改二手房信息，包括户型、楼层、面积等。求租者方面应该设置偏好便于进行房源推荐。管理员可以根据求租者的意向匹配符合要求的房屋信息。

管理员分为经理和普通员工两类，经理可以进行员工信息的录入，更改，删除，赋予员工权限等。管理员可以进行自己信息的修改。管理员可以查看目前房屋的状态，例如租出、预定、未租出。同时系统在用户租到房屋时应该自动调整房屋的状态，将其从未租出自动改为租出。而当用户完成交易并归还时将房屋状态调整为未租出。当用户进行交易时，由管理员录入求租者、出租者、房屋、管理员等信息。

最后需要统计函数根据时间或者操作者进行成交量的统计。在系统内部可以内置一些系统功能例如记事本、计算器等。同时能进行数据库的备份，密码等的修改。

# 系统的功能模块划分

## 2.1 二手房中介系统业务流程



## 2.2 功能模块的划分

用户信息管理模块：实现求租人员的信息设置，出租人员的信息设置，人员信息的更改等功能。

求租管理模块：实现房源的查询，房源状态的浏览，保存求租意向等功能。

员工信息管理模块：实现录入员工信息，修改员工信息，删除员工信息，更改员工权限等功能。

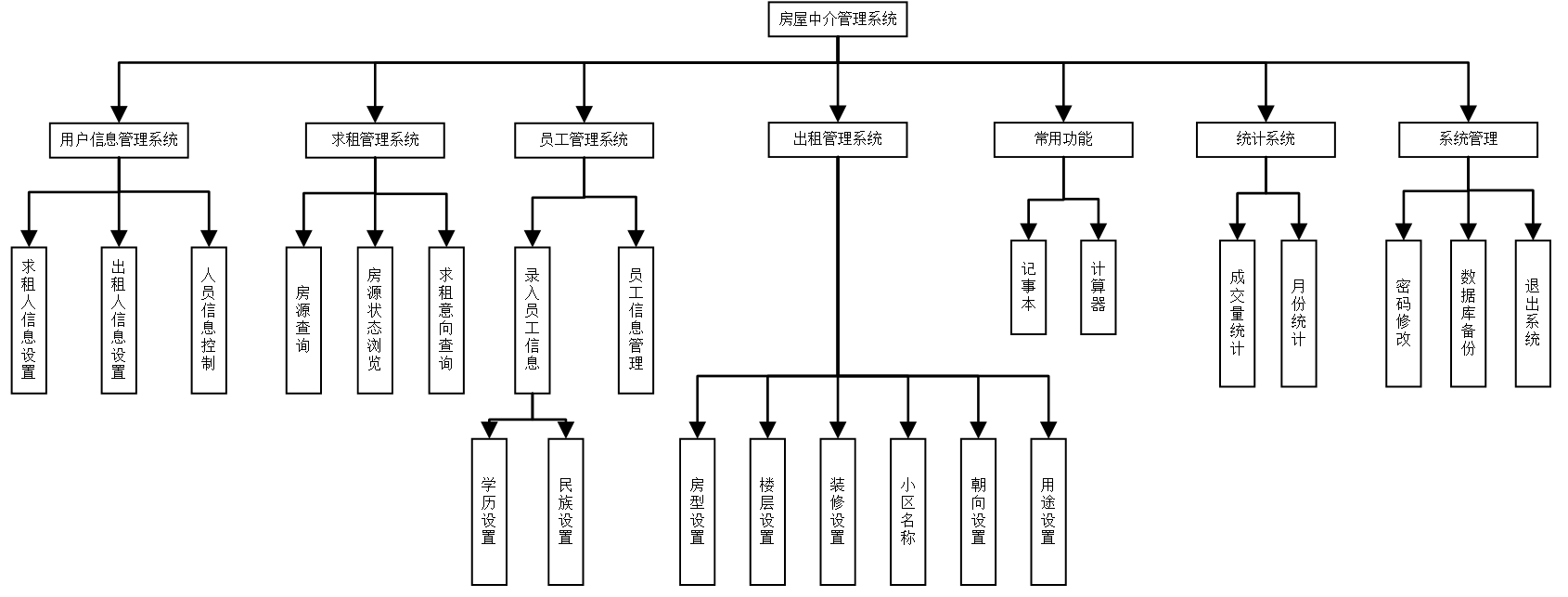
出租管理模块：实现房型设置、装修设置、朝向设置、用于等功能。

交费功能模块：实现收费设置，收费记录等功能。

业务统计模块：实现成交业务量统计等功能。

系统管理模块：实现密码更改、退出系统等功能。

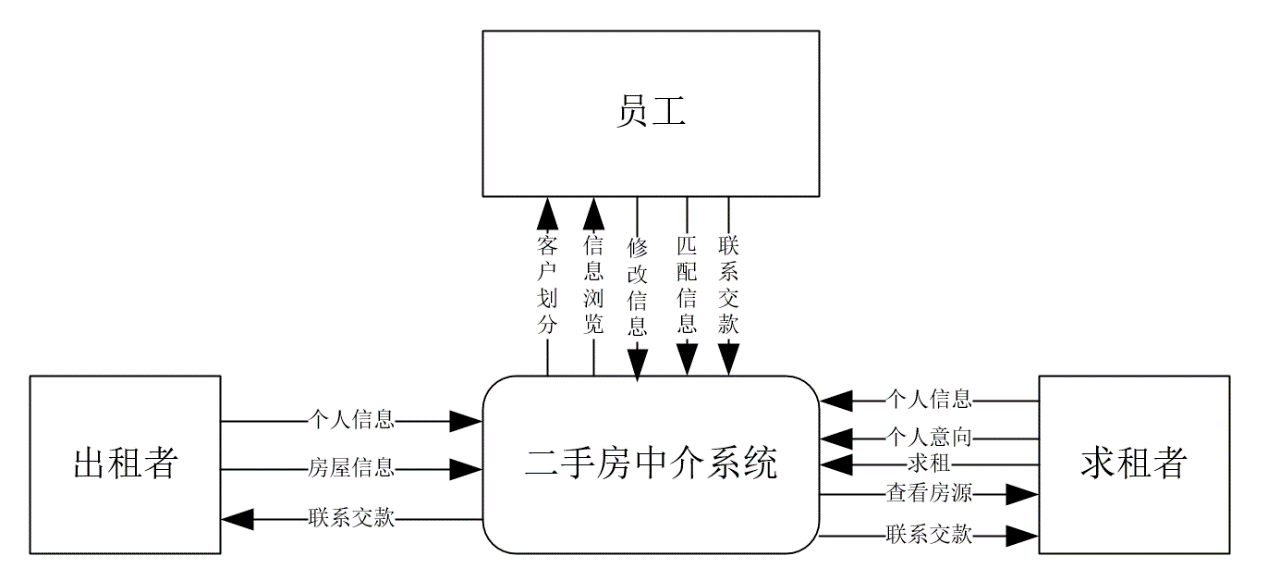
功能结构图如下：



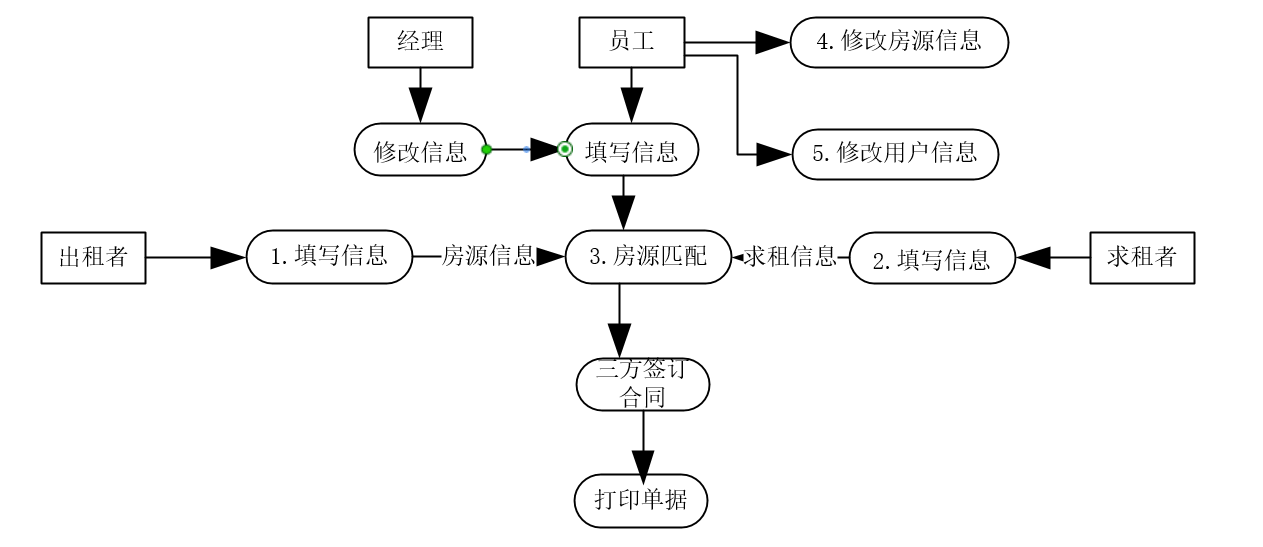
## 2.3 数据流图

### 2.3.1 环境图（顶层数据流图）

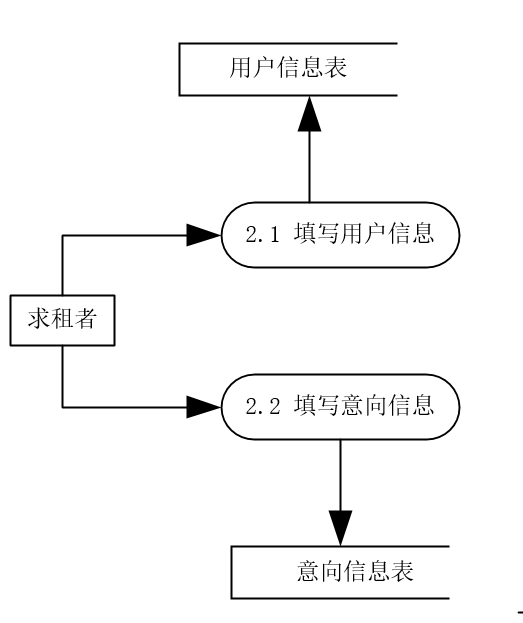
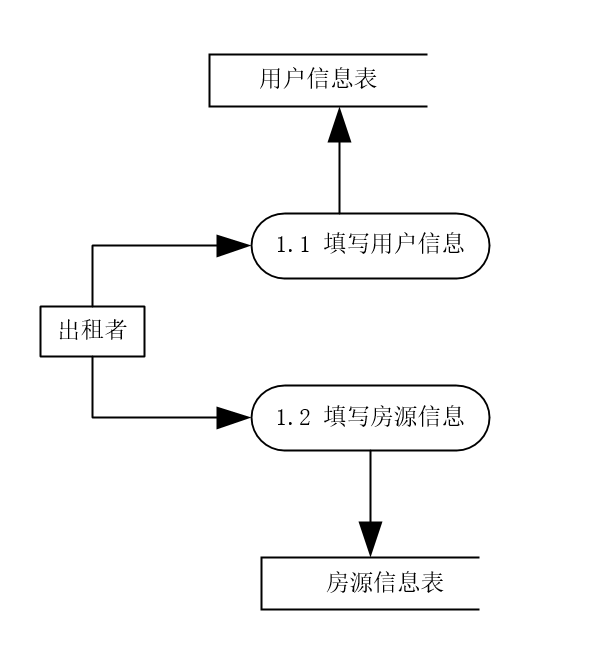
在顶层数据流图中，外部实体主要有出租者、求租者和后台管理员。系统内部存入用户信息、求租信息、出租信息，管理员可以对其修改并且匹配信息，联系卖家和买家签订合同并缴费。出租者可以传入房屋信息。求租者可以传入个人意愿，也可以查看房屋信息并进行求租。

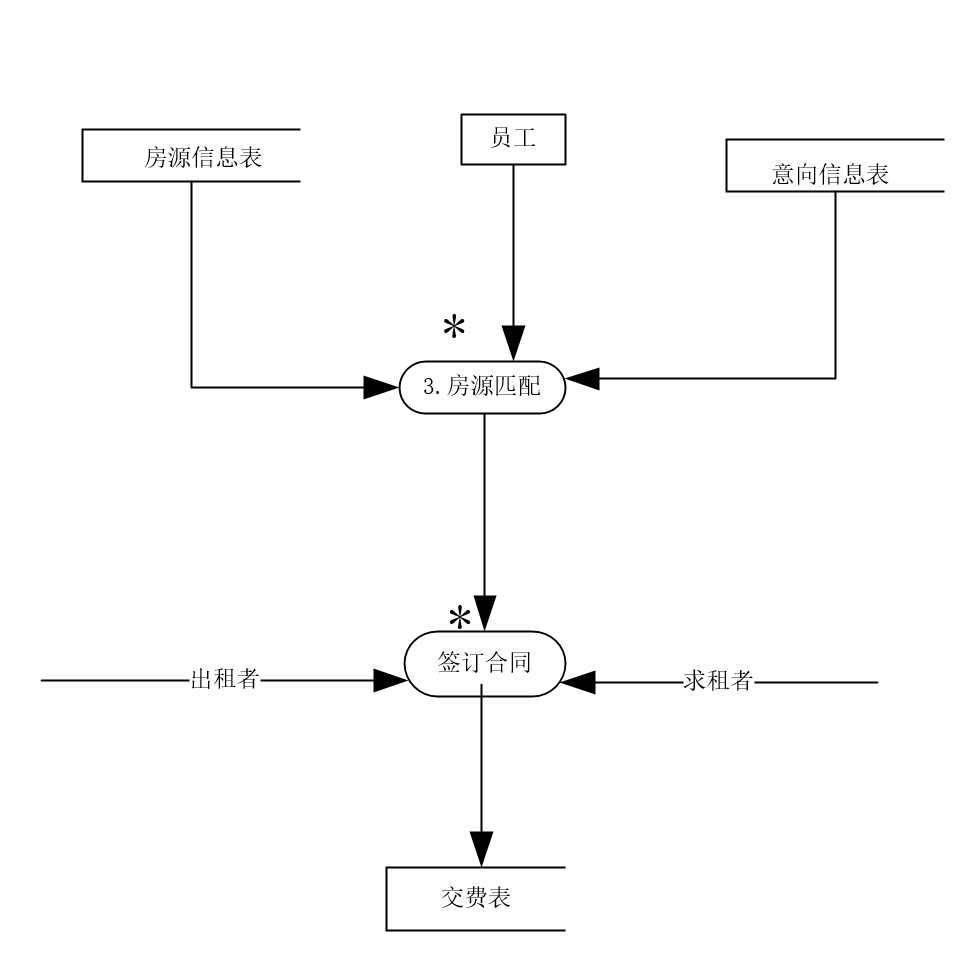


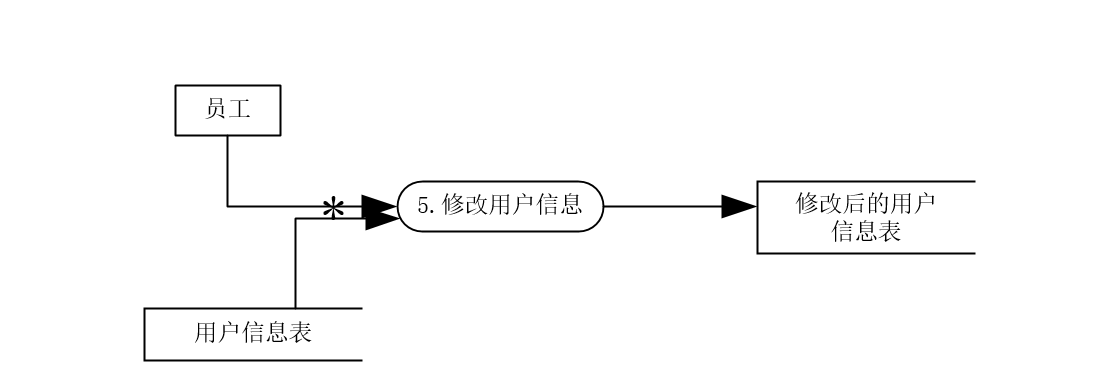
### 2.3.2 一层数据流图

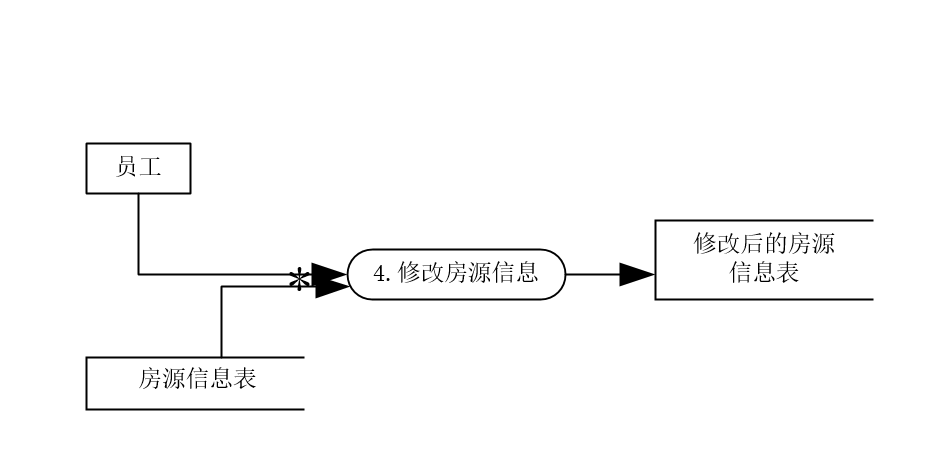


### 2.3.3 二层数据流图









## 2.4 数据字典

### 2.4.1 数据流词条

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据流名 | 简述 | 组成 | 来源 | 去向 |
| 用户注册信息 | 用户信息的记录，用于出租房屋、求租房屋以及通信联系 | 姓名、性别、联系方式、身份证、出生日期等 | 用户 | 用户信息管理 |
| 员工注册信息 | 员工信息的记录，方便进行房屋匹配和权限管理 | 姓名、性别、联系方式、出生日期、身份证号等 | 员工 | 员工信息管理 |
| 房源信息 | 出租房屋的信息 | 小区名称、房型、装修情况、朝向等 | 出租人 | 房屋信息管理 |
| 求租意向信息 | 求租者对租房的要求 | 房型、面积、装修情况、楼层、朝向等 | 求租人 | 房屋信息管理 |
| 修改用户信息 | 帮助用户修改信息 | 新的联系方式、出生日期、性别等 | 用户 | 用户管理系统 |
| 修改员工信息 | 帮助修改员工信息 | 新的出生日期、联系方式、性别、民族、等 | 员工 | 员工信息管理系统 |
| 修改房源信息 | 帮助修改房源信息 | 新的房型、小区名、占地面积、朝向、楼层、装修情况等 | 求租者 | 房源信息管理系统 |
| 修改密码 | 帮助设置或修改员工的登陆密码 | 新的密码 | 员工 | 员工信息管理系统 |

### 2.4.2 数据元素词条

|  |  |  |  |
| --- | --- | --- | --- |
| 数据元素名 | 类型 | 取值范围 | 相关数据元素及数据结构 |
| 员工编号 | varchar |  | 员工信息表，收费信息表，登录信息表 |
| 员工姓名 | varchar |  | 员工信息表 |
| 员工性别 | varchar | ‘男‘，’女‘ | 员工信息表 |
| 员工出生日期 | datetime |  | 员工信息表 |
| 员工身份证 | varchar |  | 员工信息表 |
| 员工联系方式 | varchar |  | 员工信息表 |
| 员工地址 | varchar |  | 员工信息表 |
| 员工民族 | varchar |  | 员工信息表，学历表 |
| 民族 | varchar |  | 学历表 |
| 员工学历 | varchar |  | 员工信息表，学历表 |
| 学历 | varchar |  | 学历表 |
| 员工工资 | int |  | 员工信息表 |
| 客户编号 | varchar |  | 客户信息表，收费信息表 |
| 客户姓名 | varchar |  | 客户信息表 |
| 客户性别 | varchar | ‘男‘，’女‘ | 客户信息表 |
| 客户出生日期 | datetime |  | 客户信息表 |
| 客户联系方式 | varchar |  | 客户信息表 |
| 客户身份证 | varchar |  | 客户信息表 |
| 客户类型 | varchar | ‘出租’, ’求租’ | 客户信息表 |
| 客户房屋编号 | varchar |  | 客户信息表，房源信息表 |
| 客户注册时间 | datetime |  | 客户信息表 |
| 房屋编号 | varchar |  | 房源信息表、求租意向表、收费信息表 |
| 物业名称 | varchar |  | 房源信息表 |
| 房型编号 | varchar |  | 房源信息表、求租意向表、房型信息表 |
| 房型 | varchar |  | 房源信息表、房型信息表 |
| 状态 | varchar | ‘未租’、‘已租‘、’预定‘ | 房源信息表 |
| 装修编号 | varchar |  | 房源信息表、求租意向表、装修信息表 |
| 装修情况 | varchar |  | 装修信息表 |
| 朝向情况 | varchar |  | 朝向信息表 |
| 朝向编号 | varchar |  | 房源信息表、求租意向表、朝向信息表 |
| 用途情况 | varchar |  | 用途信息表 |
| 用途编号 | varchar |  | 房源信息表、求租意向表、用途信息表 |
| 价格 | int |  | 房源信息表、求租意向表 |
| 楼层编号 | varchar |  | 房源信息表、求租意向表、楼层信息表 |
| 楼层 | varchar |  | 楼层信息表 |
| 建筑年限 | varchar |  | 房源信息表 |
| 建筑面积 | int |  | 房源信息表、求租意向表 |
| 费用金额 | int |  | 费用信息表 |
| 费用编号 | varchar |  | 费用信息表 |
| 付款日期 | datetime |  | 费用信息表 |
| 登录编号 | varchar |  | 登录信息表 |
| 登录名 | varchar |  | 登录信息表 |
| 密码 | varchar |  | 登录信息表 |
| 权限 | varchar | ‘1‘， ’0‘ | 登录信息表 |

### 2.4.3 数据存储文件词条

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 用户名 | 简述 | 组成 | 输入 | 输出 | 存取方式 |
| 学历信息表 | 员工的学历 | 学历、学历编号 | 员工添加 | 员工信息 | 关键码 |
| 民族信息表 | 员工的民族 | 民族、民族编号 | 员工添加 | 员工信息 | 关键码 |
| 员工信息表 | 员工的基本资料 | 姓名、性别、员工编号、出生日期、联系方式、地址、工资、学历、民族 | 员工添加 | 员工基本信息 | 关键码 |
| 客户信息表 | 客户的基本资料 | 姓名、性别、客户编号、出生日期、联系方式、注册日期、客户类别 | 客户添加 | 客户基本信息 | 关键码 |
| 房源信息表 | 房源的基本信息 | 房屋编号、小区名称、房型编号、状态、装修编号、朝向编号、用途编号、价格、楼层编号、建筑年限、建筑面积 | 客户添加 | 房源的基本信息 | 关键码 |
| 房型信息表 | 房型信息 | 房型编号、房型 | 客户添加 | 房型信息 | 关键码 |
| 楼层信息表 | 楼层信息 | 楼层、楼层编号 | 客户添加 | 楼层信息 | 关键码 |
| 朝向信息表 | 朝向信息 | 朝向、朝向编号 | 客户添加 | 朝向信息 | 关键码 |
| 登录信息表 | 登录信息 | 登录名、密码、权限、员工号 | 员工添加 | 登录信息 | 关键码 |
| 收费信息表 | 员工收费信息 | 求租人、出租人、员工、收费、房源编号、时间、金额 | 客户添加 | 收费信息 | 关键码 |
| 求租意向表 | 求租人的意向信息 | 求租人编号、房源房型、房源装修情况、房源楼层、房源用途、房源朝向、面积 | 客户添加 | 求租意向信息 | 关键码 |
| 装修情况表 | 装修信息 | 装修情况、装修编号 | 客户添加 | 装修信息 | 关键码 |
| 用途信息表 | 用途信息 | 用途、用途编号 | 客户添加 | 用途信息 | 关键码 |

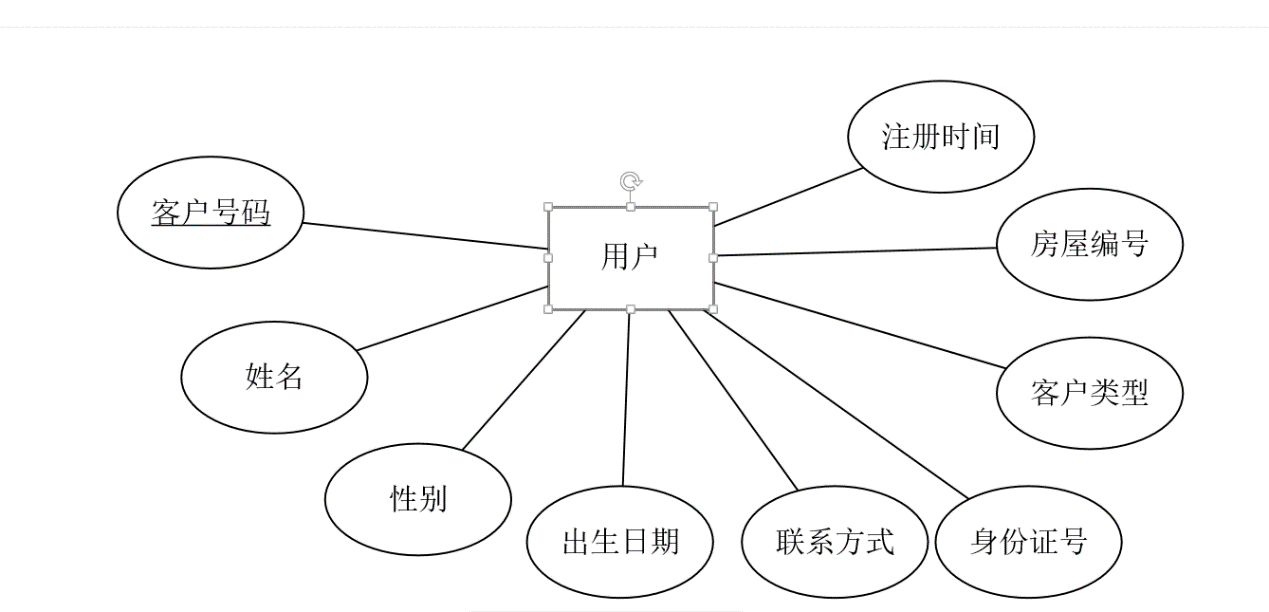
### 2.4.4 加工词条

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 加工名 | 编号 | 简述 | 输入 | 输出 | 加工逻辑 |
| 用户填写个人信息 | 1.1 | 用户注册后系统写入数据库 | 用户的基本信息，包括姓名、性别、身份证号、出生日期、联系方式等 | 用户信息表 | 接受用户的注册信息并将信息存入数据库 |
| 用户填写房源信息 | 1.2 | 用户填写房源信息 | 房源的基本信息，包括楼层、装修情况、小区名称等 | 房源信息表 | 接受用户的房源信息并将信息存入数据库 |
| 员工填写个人信息 | 2 | 员工填写个人信息 | 员工的个人信息，包括姓名、联系方式、出生日期、性别、工资等 | 员工信息表 | 接受员工的个人信息信息并将信息存入数据库 |
| 用户填写意向信息 | 2.3 | 用户填写意向信息 | 用户的意向信息，包括房子的房型、装修情况、楼层、朝向等 | 用户意向表 | 接受用户的意向信息并将信息存入数据库 |
| 用户员工签订合同 | 3 | 用户、员工填写个人信息并交费 | 出租者、求租者、员工、房源信息等 | 收费信息表 | 记录房子的租借情况，将出租者、求租者、员工的信息存入数据库 |
| 员工修改用户信息 | 5 | 员工修改用户的信息 | 修改后的性别、出生日期、身份证号等 | 用户信息表 | 修改用户的个人信息信息并在数据库中修改 |
| 员工修改房源信息 | 4 | 员工修改房源信息 | 修改后的房型、装修情况、小区名称、朝向等 | 房源信息表 | 修改用户的房源信息信息并在数据库中修改 |

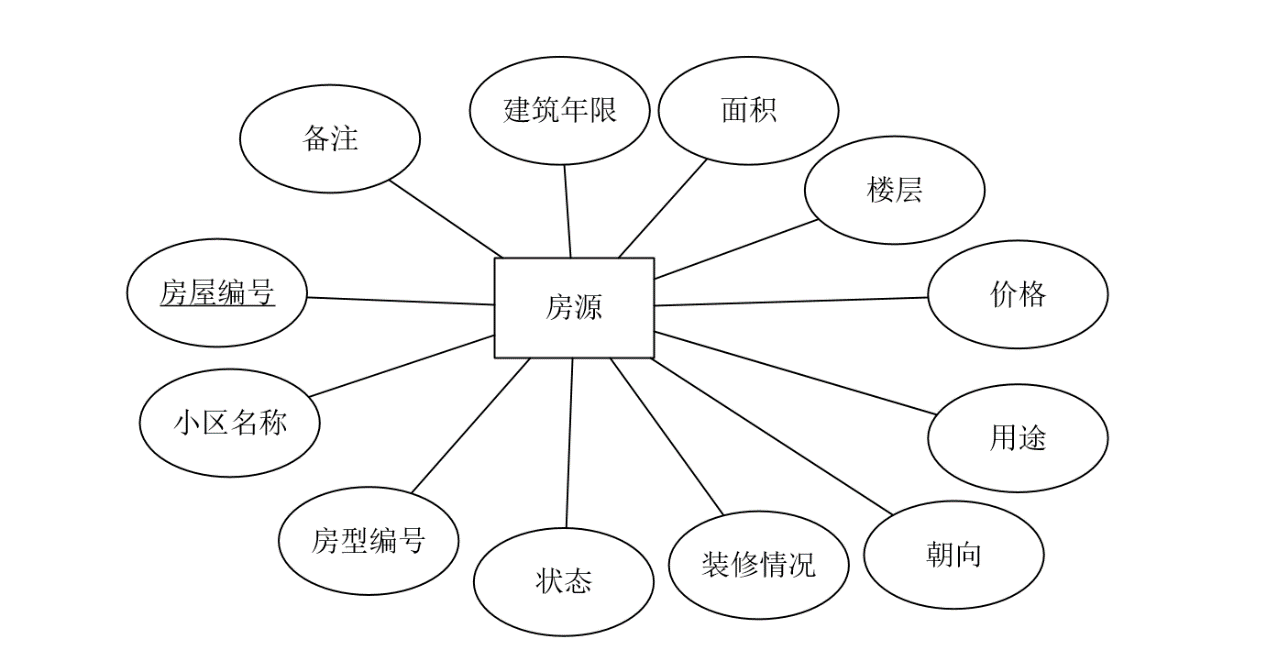
# 数据库概念结构设计

## 3.1 单个实体的E-R图

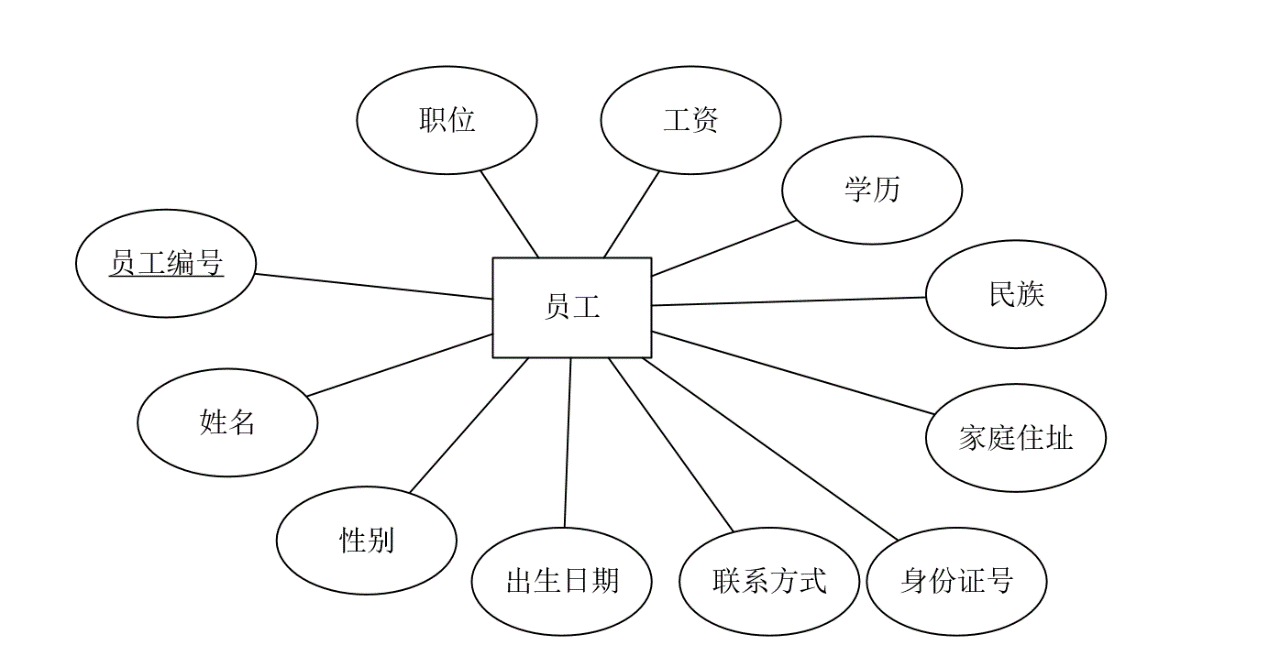
### 3.1.1 用户实体的E-R图



### 3.1.2 房屋实体的E-R图

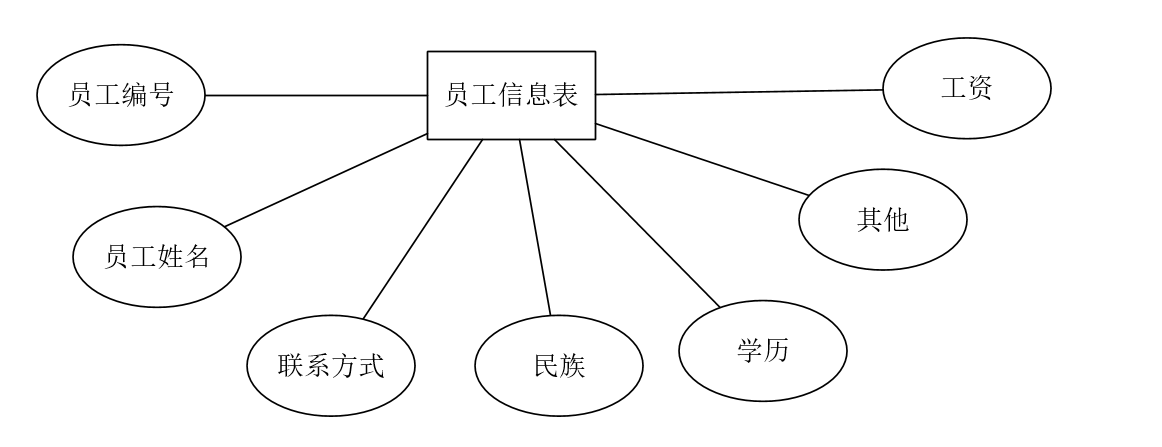


### 3.1.3 员工实体的E-R图

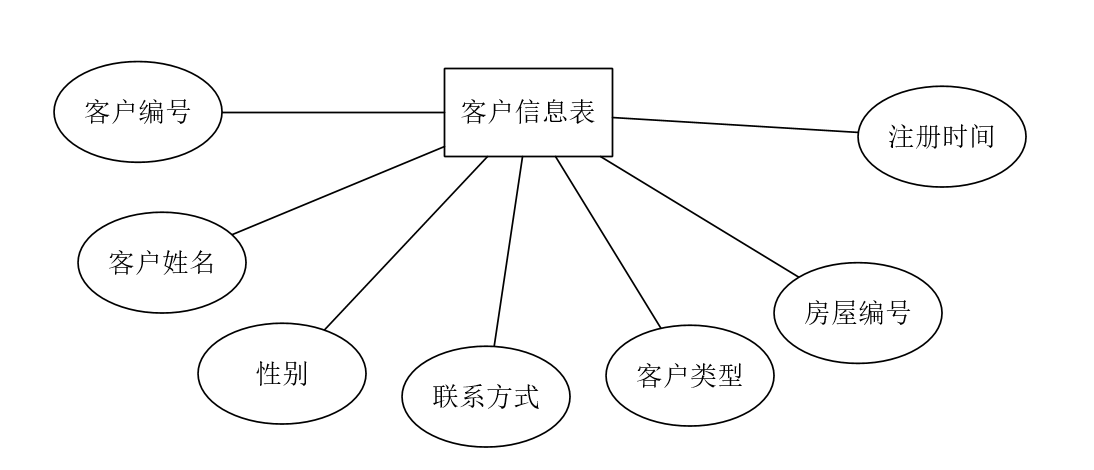


## 3.2 数据表E-R图

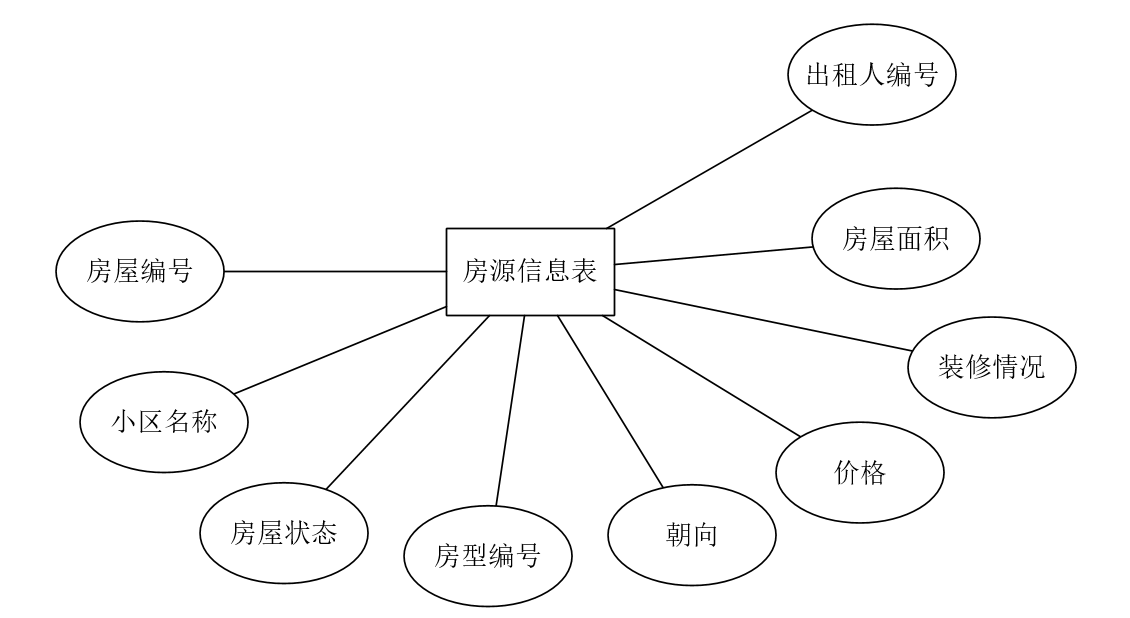
### 3.2.1 员工信息表E-R图



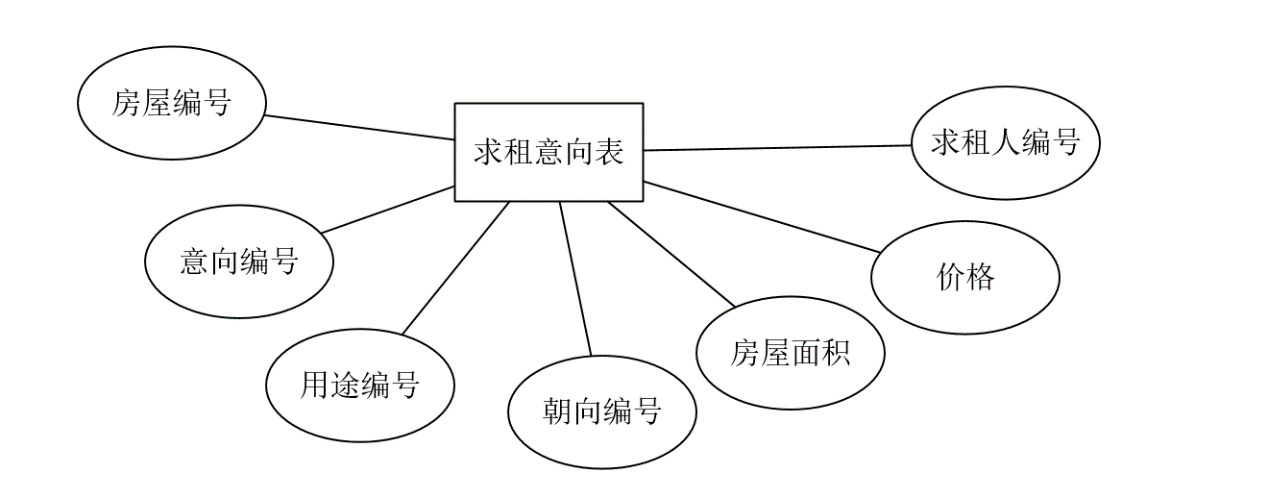
### 3.2.2 客户信息表E-R图



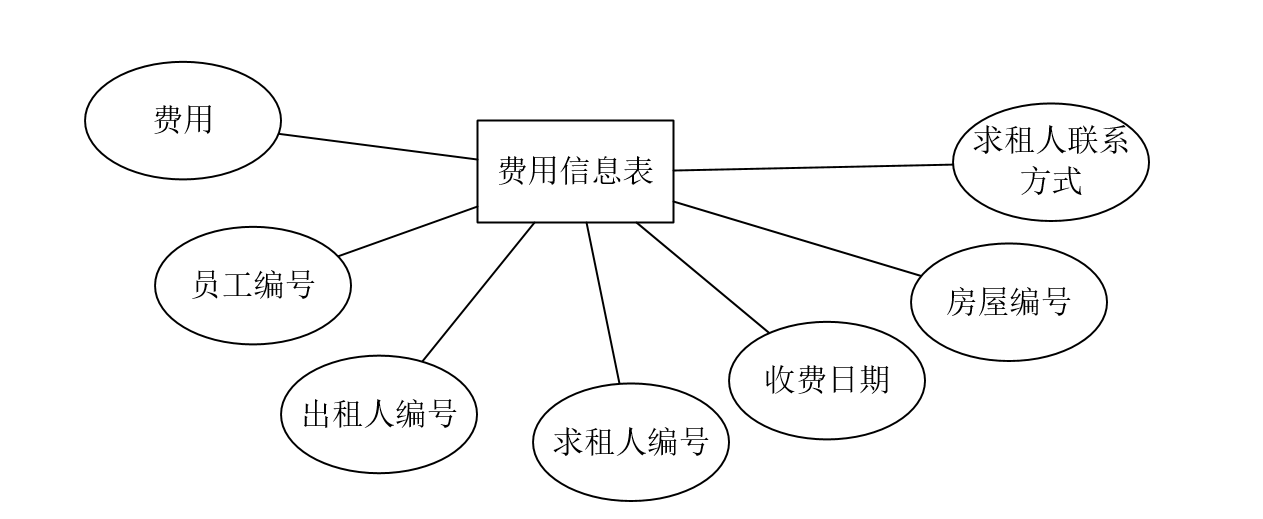
### 3.2.3 房源信息表E-R图



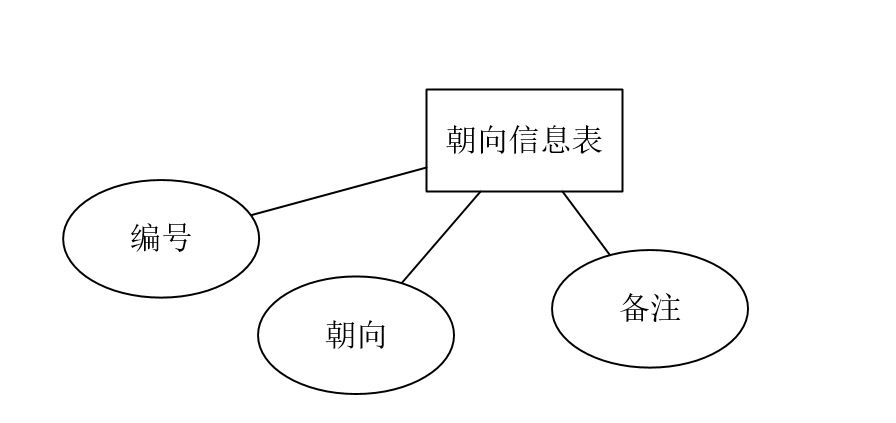
### 3.2.4 求租意向表E-R图



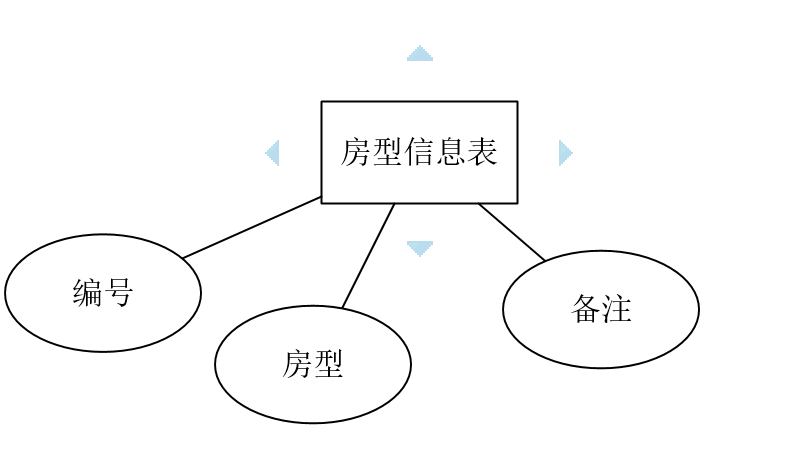
### 3.2.5 费用信息表E-R图



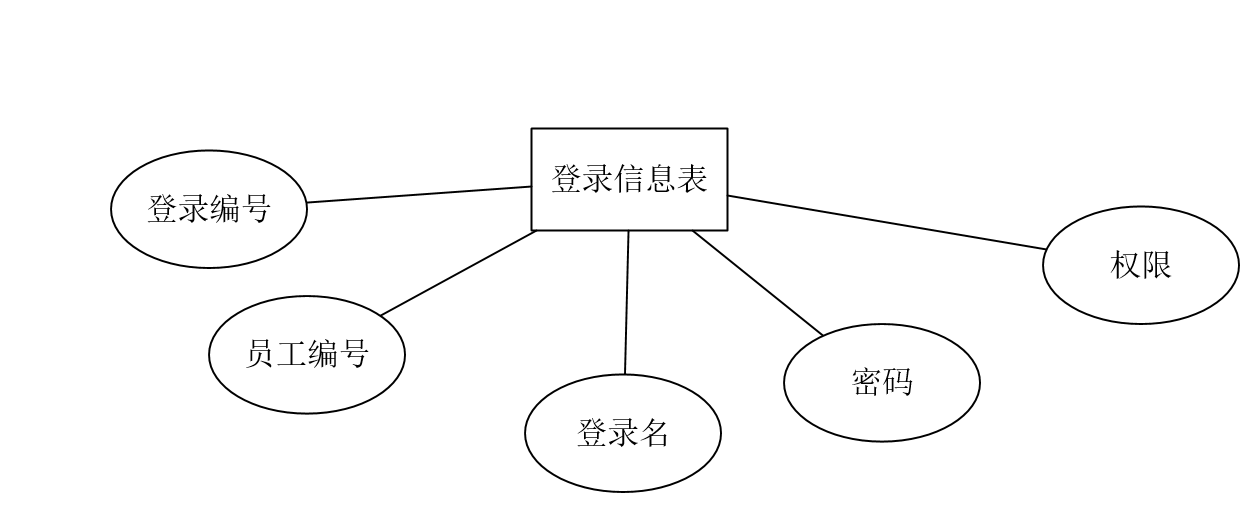
### 3.2.6 朝向信息表E-R图



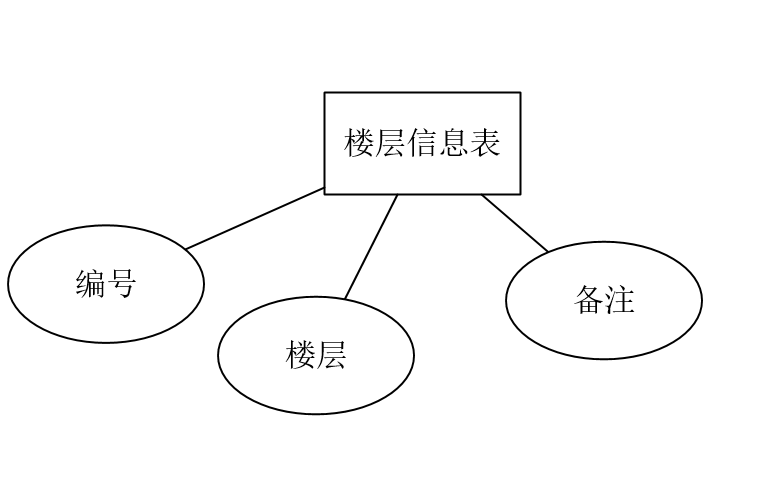
### 3.2.7 房型信息表E-R图



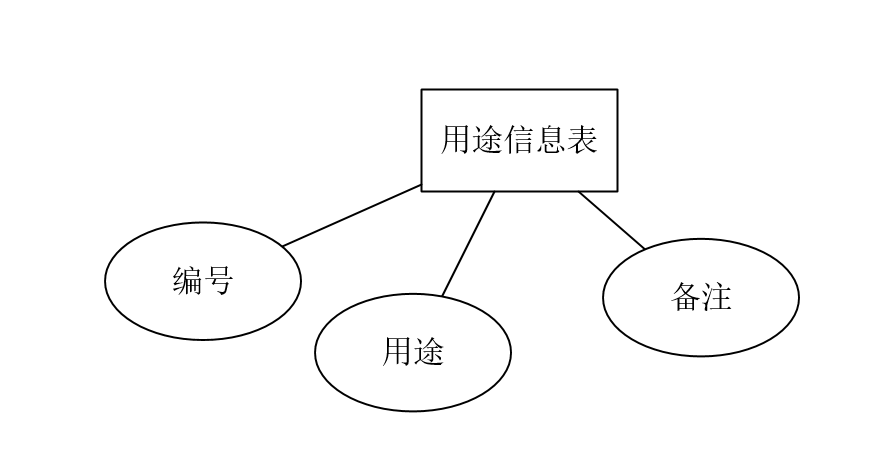
### 3.2.8 登录信息表E-R图



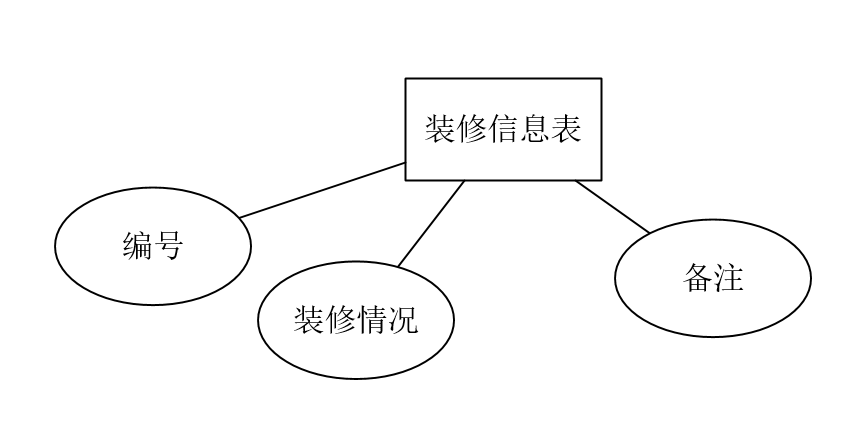
### 3.2.9 楼层信息表E-R图



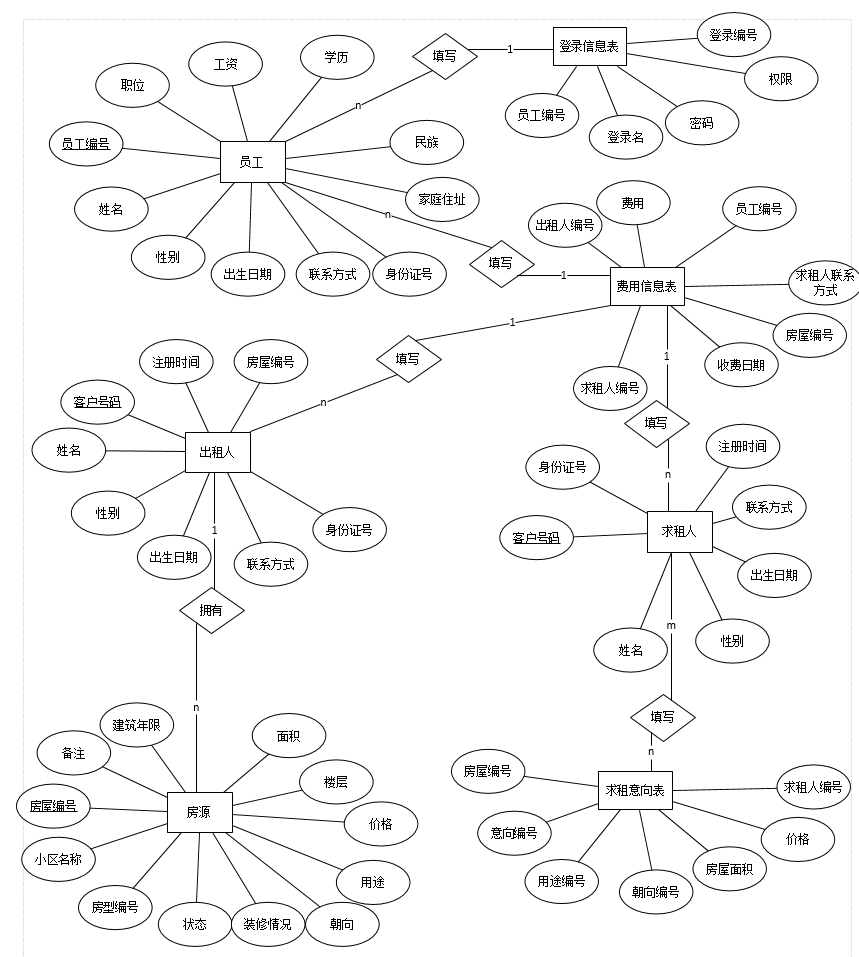
### 3.2.10 用途信息表E-R图



### 3.2.11 装修信息表E-R图

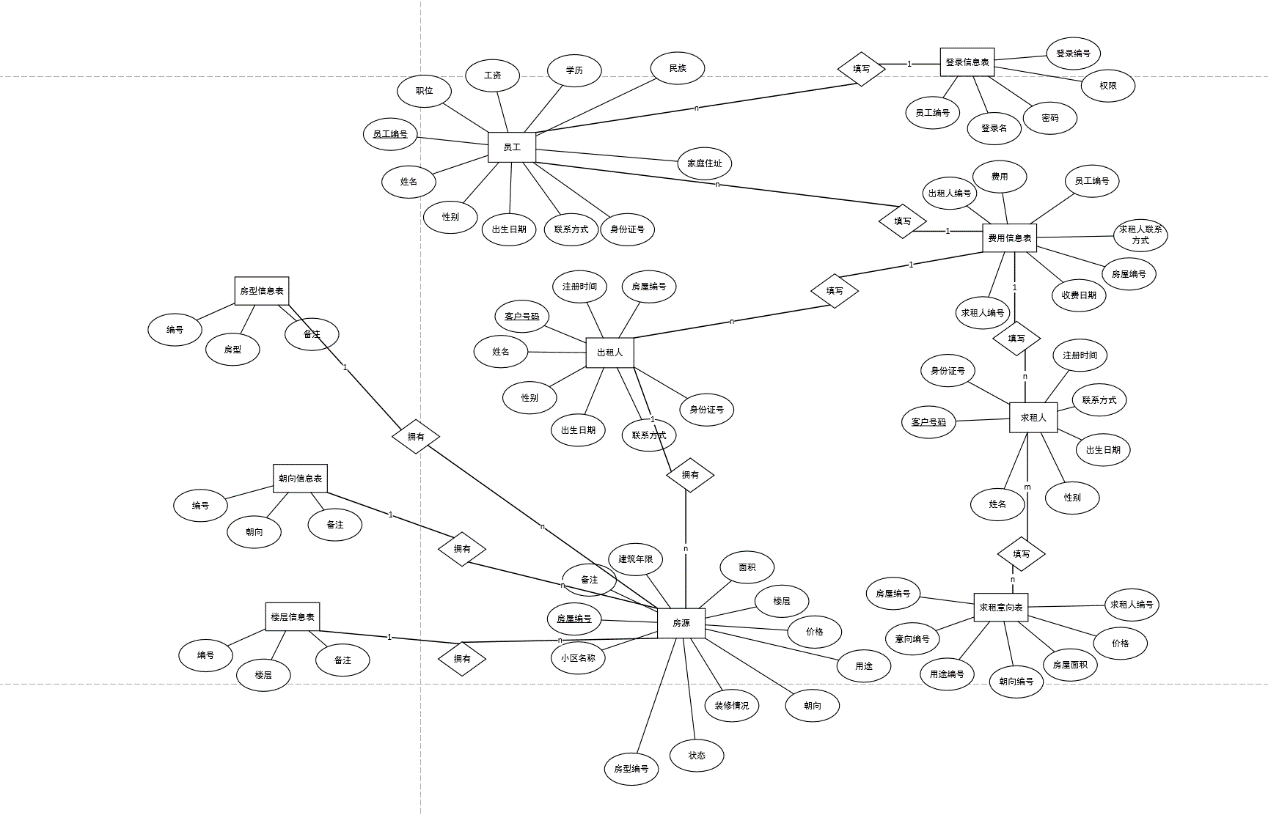


## 3.3 整体E-R图



# 数据库逻辑结构设计

## 4.1 调整后的E-R图



## 4.2 关系模式及规范化处理

1. 员工信息表（员工编号，姓名，性别，出生日期，联系方式，身份证号，地址，民族编号，学历编号，工资）
2. 民族信息表（民族编号，民族名，备注）
3. 学历信息表（学历编号，学历名，备注）
4. 客户信息表（客户编号，姓名，性别，出生日期，联系方式，身份证号，客户类型，房屋编号，注册日期）
5. 房源信息表（房源编号，小区名称，房型编号，状态，装修编号，朝向编号，用途编号，价格，楼层编号，建筑年限，面积，用户编号，备注）
6. 房型信息表（房型编号，房型名称，备注）
7. 装修信息表（装修编号，装修情况，备注）
8. 朝向信息表（朝向编号，朝向情况，备注）
9. 用途信息表（用途编号，用途情况，备注）
10. 楼层信息表（楼层编号，楼层信息，备注）
11. 求租意向表（意向编号，用户编号，房屋类型，装修编号 ，楼层编号，朝向编号，用途编号，价格，面积）
12. 收费信息表（费用编号，员工编号，员工姓名，房屋编号，出租人编号，出租人姓名，出租人联系方式，求租人编号，求租人姓名，求租人联系方式，金额，付款日期，备注）
13. 登录信息表（登录编号，员工编号，登录名，密码，权限）

## 4.3 数据库中的表

### 4.3.1 员工信息表（employee）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 描述 | 约束 | 主键否 |
| emp\_ID | varchar | 10 | 员工编号 | not null | 主键 |
| emp\_name | varchar | 20 | 姓名 |  | 否 |
| emp\_sex | varchar | 10 | 性别 | ‘男’，‘女‘ | 否 |
| emp\_birth | datetime |  | 出生日期 |  | 否 |
| emp\_phone | varchar | 20 | 联系方式 |  | 否 |
| emp\_cardID | varchar | 20 | 身份证号 |  | 否 |
| emp\_address | varchar | 50 | 地址 |  | 否 |
| emp\_govID | varchar | 10 | 民族编号 |  | 否 |
| emp\_studyID | varchar | 10 | 学历编号 |  | 否 |
| emp\_pay | numeric | numeric(19,4) | 工资 | >0 | 否 |

### 4.3.2 民族信息表(gov)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | | 约束 | | 主键否 |
| gov\_ID | varchar | | 10 | | 民族编号 | | not null | | 主键 | |
| gov\_name | varchar | | 20 | | 民族名 | |  | | 否 | |
| gov\_remark | varchar | | 50 | | 备注 | |  | | 否 | |

### 4.3.3 学历信息表(study)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | | 约束 | | 主键否 |
| study\_ID | varchar | | 10 | | 学历编号 | | not null | | 主键 | |
| study\_name | char | | 20 | | 学历 | |  | | 否 | |
| study\_remark | varchar | | 50 | | 备注 | |  | | 否 | |
|  |  | |  | |  | |  | |  | |

### 4.3.4 客户信息表（users）

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | | 约束 | | 主键否 |
| user\_ID | varchar | | 10 | | 客户编号 | | not null | | 主键 | |
| user\_name | varchar | | 20 | | 姓名 | |  | | 否 | |
| user\_sex | varchar | | 4 | | 性别 | | ‘男‘，’女‘ | | 否 | |
| user\_birth | datetime | |  | | 出生日期 | |  | | 否 | |
| user\_phone | varchar | | 20 | | 联系方式 | |  | | 否 | |
| user\_cardID | varchar | | 20 | | 身份证号 | |  | | 否 | |
| user\_type | varchar | | 10 | | 客户类型 | | ‘lend’,’want’ | | 否 | |
| house\_ID | varchar | | 10 | | 房屋编号 | |  | | 否 | |
| user\_record | datetime | |  | | 注册日期 | |  | | 否 | |

### 4.3.5 房源信息表（house）

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | | 约束 | | 主键否 |
| house\_ID | varchar | | 10 | | 房源编号 | | not null | | 主键 | |
| house\_company | varchar | | 50 | | 小区名称 | |  | | 否 | |
| house\_typeID | varchar | | 10 | | 房型编号 | |  | | 否 | |
| house\_state | varchar | | 10 | | 状态 | | ‘reserve’,’free’,’lent’ | | 否 | |
| house\_fitID | varchar | | 10 | | 装修编号 | |  | | 否 | |
| house\_favorID | varchar | | 10 | | 朝向编号 | |  | | 否 | |
| house\_mothedID | varchar | | 10 | | 用途编号 | |  | | 否 | |
| house\_price | numeric | | numeric(18,0) | | 价格 | | >0 | | 否 | |
| house\_floorID | varchar | | 10 | | 楼层编号 | |  | | 否 | |
| house\_buildYear | varchar | | 10 | | 建筑年限 | |  | | 否 | |
| house\_area | varchar | | 20 | | 面积 | | >0 | | 否 | |
| house\_remark | varchar | | 50 | | 备注 | |  | | 否 | |
| user\_ID | varchar | | 10 | | 用户编号 | |  | | 否 | |

### 4.3.6 房型信息表(type)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | | 约束 | | 主键否 |
| type\_ID | varchar | | 10 | | 房型编号 | | not null | | 主键 | |
| type\_name | varchar | | 20 | | 房型名称 | |  | | 否 | |
| type\_remark | varchar | | 50 | | 备注 | |  | | 否 | |

### 4.3.7 装修信息表(fit)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | 约束 | | | 主键否 |
| fit\_ID | varchar | | 10 | | 装修编号 | | | not null | 主键 | |
| fit\_name | varchar | | 20 | | 装修情况 | | |  | 否 | |
| fit\_remark | varchar | | 50 | | 备注 | | |  | 否 | |

### 4.3.8 朝向信息表(favor)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | 约束 | | | 主键否 |
| favor\_ID | varchar | | 10 | | 朝向编号 | | | not null | 主键 | |
| favor\_name | varchar | | 20 | | 朝向 | | |  | 否 | |
| favor\_remark | varchar | | 50 | | 备注 | | |  | 否 | |

### 4.3.9 用途信息表(method)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | 约束 | | | 主键否 |
| method\_ID | varchar | | 10 | | 用途编号 | | | not null | 主键 | |
| method\_name | varchar | | 20 | | 用途 | | |  | 否 | |
| method\_remark | varchar | | 50 | | 备注 | | |  | 否 | |

### 4.3.10 楼层信息表(floor)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | 约束 | | | 主键否 |
| floor\_ID | varchar | | 10 | | 楼层编号 | | | not null | 主键 | |
| floor\_name | varchar | | 20 | | 楼层信息 | | |  | 否 | |
| floor\_remark | varchar | | 50 | | 备注 | | |  | 否 | |

### 4.3.11 求租意向表(intent)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | 约束 | | | 主键否 |
| intent\_ID | varchar | | 10 | | 意向编号 | | | not null | 主键 | |
| user\_ID | varchar | | 10 | | 用户编号 | | |  | 否 | |
| intent\_typeID | varchar | | 10 | | 房屋类型 | | |  | 否 | |
| intent\_fitID | varchar | | 10 | | 装修编号 | | |  | 否 | |
| intent\_floorID | varchar | | 10 | | 楼层编号 | | |  | 否 | |
| intent\_favorID | varchar | | 10 | | 朝向编号 | | |  | 否 | |
| intent\_methodID | varchar | | 10 | | 用途编号 | | |  | 否 | |
| intent\_price | numeric | | numeric(19,4) | | 价格 | | | >0 | 否 | |
| intent\_area | varchar | | 20 | | 面积 | | | >0 | 否 | |

### 4.3.12 收费信息表(money)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | | 约束 | | 主键否 |
| money\_ID | varchar | | 10 | | 费用编号 | | not null | | 主键 | |
| money\_pay | numeric | | numeric(18,0) | | 金额 | | >0 | | 否 | |
| emp\_ID | varchar | | 10 | | 员工编号 | |  | | 否 | |
| emp\_name | varchar | | 20 | | 员工姓名 | |  | | 否 | |
| house\_ID | varchar | | 10 | | 房屋编号 | |  | | 否 | |
| money\_time | varchar | | 50 | | 付款日期 | |  | | 否 | |
| money\_remark | varchar | | 100 | | 备注 | |  | | 否 | |
| lend\_ID | varchar | | 10 | | 出租人编号 | |  | | 否 | |
| lend\_name | varchar | | 20 | | 出租人姓名 | |  | | 否 | |
| lend\_phone | varchar | | 20 | | 出租人联系方式 | |  | | 否 | |
| want\_ ID | varchar | | 10 | | 求租人编号 | |  | | 否 | |
| want\_name | varchar | | 20 | | 求租人姓名 | |  | | 否 | |
| want\_ phone | varchar | | 20 | | 求租人联系方式 | |  | | 否 | |

### 4.3.13 登录信息表(login)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 列名 | | 数据类型 | | 长度 | | 描述 | | 约束 | | 主键否 |
| login\_ID | varchar | | 10 | | 登录编号 | | not null | | 主键 | |
| emp\_ID | varchar | | 10 | | 员工编号 | |  | | 否 | |
| login\_name | varchar | | 20 | | 登录名 | |  | | 否 | |
| login\_pwd | varchar | | 15 | | 密码 | |  | | 否 | |
| login\_power | varchar | | 10 | | 权限 | |  | | 否 | |

# 系统实现过程及完成效果

## 5.1 建表语句

--数据库建表

--（1）员工信息表

create table employee(

emp\_ID varchar (10) not null,

emp\_name varchar (20),

emp\_sex varchar (10),

emp\_birth datetime,

emp\_phone varchar (20),

emp\_cardID varchar (20),

emp\_address varchar (50),

emp\_govID varchar (10),

emp\_studyID varchar (10),

emp\_pay numeric(19,4)

)

--（2）民族信息表

create table gov(

gov\_ID varchar (10) not null,

gov\_name varchar (20),

gov\_remark varchar (50)

)

--（3）学历信息表

create table study(

study\_ID varchar (10),

study\_name char (20),

study\_remark varchar (50)

)

--（4）客户信息表

create table users(

user\_ID varchar (10) not null,

user\_name varchar (20),

user\_sex varchar (4),

user\_birth datetime,

user\_phone varchar (20),

user\_cardID varchar (20),

user\_type varchar (10),

house\_ID varchar (10),

user\_record datetime

)

--（5）房源信息表

create table house (

house\_ID varchar (10) ,

house\_company varchar (50),

house\_typeID varchar (10),

house\_state varchar (10),

house\_fitID varchar (10),

house\_favorID varchar (10),

house\_mothedID varchar (10),

house\_price numeric(18,0),

house\_floorID varchar (10),

house\_buildYear varchar (10),

house\_area varchar (20),

house\_remark varchar (50),

user\_ID varchar (10)

)

--（6）房型信息表

create table type(

type\_ID varchar (10),

type\_name varchar (20),

type\_remark varchar (50)

)

--（7）装修信息表

create table fit(

fit\_ID varchar (10) not null,

fit\_name varchar (20),

fit\_remark varchar (50)

)

--（8）朝向信息表

create table favor(

favor\_ID varchar (10),

favor\_name varchar (20),

favor\_remark varchar (50)

)

--(9)用途信息表

create table method(

method\_ID varchar (10),

method\_name varchar (20),

method\_remark varchar (50)

)

--（10）楼层信息表

create table floor(

floor\_ID varchar (10)not null,

floor\_name varchar (20),

floor\_remark varchar (50)

)

--（11）求租意向表

create table intent(

intent\_ID varchar (10) not null,

user\_ID varchar (10),

intent\_typeID varchar (10),

intent\_fitID varchar (10),

intent\_floorID varchar (10),

intent\_favorID varchar (10),

intent\_methodID varchar (10),

intent\_price numeric(19,4),

intent\_area varchar (20)

)

--（12）收费信息表

create table money(

money\_ID varchar (10) not null,

money\_pay numeric(18,0),

emp\_ID varchar (10),

emp\_name varchar (20),

house\_ID varchar (10),

money\_time varchar (50),

money\_remark varchar (100),

lend\_ID varchar (10),

lend\_name varchar (20),

lend\_phone varchar (20),

want\_ID varchar (10),

want\_name varchar (20),

want\_phone varchar (20)

)

--（13）登录信息表

create table login(

login\_ID varchar (10) not null,

emp\_ID varchar (10),

login\_name varchar (20),

login\_pwd varchar (15),

login\_power varchar (10)

)

## 5.2 完善相应的控制语句

### 5.2.1 表的控制

--（1）员工信息表

alter table employee add constraint e1 check (emp\_sex in('男','女'))

--（3）学历信息表

alter table study add constraint s1 check (study\_ID is not null)

--（4）客户信息表

alter table users add constraint u1 check (user\_sex in('男','女'))

alter table users add constraint u2 check (user\_type in('lend','want'))

--（5）房源信息表

alter table house add constraint h1 check (house\_ID is not null)

alter table house add constraint h2 check (house\_state in ('reserve','free','lent'))

--（6）房型信息表

alter table type add constraint t1 check (type\_ID is not null)

--（8）朝向信息表

alter table favor add constraint f1 check (favor\_ID is not null)

--（9）用途信息表

alter table method add constraint m1 check (method\_ID is not null)

### 5.2.2 数据插入举例

--插入语句

--（1）员工信息表

insert into employee(

emp\_name,

emp\_sex,

emp\_birth,

emp\_phone,

emp\_cardID,

emp\_address,

emp\_govID,

emp\_studyID ,

emp\_pay

)

values(

'mr','男', '2001-11-15 00:00:00.000', '13652412345',

'101010222000101014','汤臣一品','gov1','stu1' ,2000.0000

)

--（2）民族信息

insert into gov(gov\_name,gov\_remark) values('汉族','人数最多')

insert into gov(gov\_name,gov\_remark) values('回族','不吃猪肉')

--（4）客户信息表

insert into users

(user\_name,

user\_sex,

user\_birth,

user\_phone,

user\_cardID,

user\_type,

house\_ID

)

values

('阿松大','男','1991-11-15 00:00:00.000','13255525552','152452633996874152','lend','hou1')

insert into users

(user\_name,

user\_sex,

user\_birth,

user\_phone,

user\_cardID,

user\_type

)

values

('大王','男','1998-8-15 20:00:00.000','1254478965','325145215478965415','want')

### 5.2.3 视图举例

（1）员工视图

go

create view employee\_view

as

select

e1.emp\_ID 员工编号,

e1.emp\_name 姓名,

e1.emp\_birth 出生日期,

e1.emp\_phone 联系方式,

e1.emp\_pay 工资,

o1.login\_power 权限,

o1.login\_pwd 密码

from employee e1,login o1

where e1.emp\_ID=o1.emp\_ID

（2）房屋视图

go

create view house\_view

as

select

h1.house\_ID 房屋编号,

h1.house\_company 小区名称,

h1.house\_area 面积,

h1.house\_price 价格,

f1.favor\_name 朝向,

f2.fit\_name 装修情况,

f3.floor\_name 楼层,

m1.method\_name 用途,

t1.type\_name 类型,

h1.house\_state 状态,

h1.user\_ID 用户编号

from

house h1,

favor f1,

fit f2,

floor f3,

method m1,

type t1

where

h1.house\_favorID=f1.favor\_ID and

h1.house\_fitID=f2.fit\_ID and

h1.house\_floorID = f3.floor\_ID and

h1.house\_mothedID =m1.method\_ID and

h1.house\_typeID =t1.type\_ID

### 5.2.4 索引

--索引

create unique index q1 on employee(emp\_ID asc)

create unique index q2 on favor(favor\_ID asc)

create unique index q3 on fit(fit\_ID asc)

create unique index q4 on floor(floor\_ID asc)

create unique index q5 on gov(gov\_ID asc)

create unique index q6 on house(house\_ID asc)

create unique index q7 on intent(intent\_ID asc)

create unique index q8 on login(login\_ID asc)

create unique index q9 on method(method\_ID asc)

create unique index q10 on money(money\_ID asc)

create unique index q11 on study(study\_ID asc)

create unique index q12 on type(type\_ID asc)

create unique index q13 on users(user\_ID asc)

## 5.3 可编程性

### 5.3.1 触发器

--触发器

--（1）员工信息表

go

create trigger tr1 on employee instead of insert

as

begin

declare

@emp\_ID varchar (10) ,

@emp\_name varchar (20),

@emp\_sex varchar (10),

@emp\_birth datetime,

@emp\_phone varchar (20),

@emp\_cardID varchar (20),

@emp\_address varchar (50),

@emp\_govID varchar (10),

@emp\_studyID varchar (10),

@emp\_pay numeric(19,4)

select @emp\_ID=Max(emp\_ID) from employee

select @emp\_name=emp\_name from inserted

select @emp\_sex=emp\_sex from inserted

select @emp\_birth=emp\_birth from inserted

select @emp\_phone=emp\_phone from inserted

select @emp\_cardID=emp\_cardID from inserted

select @emp\_address=emp\_address from inserted

select @emp\_govID =emp\_govID from inserted

select @emp\_studyID=emp\_studyID from inserted

select @emp\_pay=emp\_pay from inserted

if(@emp\_ID is null)

set @emp\_ID='emp1'

else

set @emp\_ID='emp'+cast(cast(substring(@emp\_ID,4,4) as int)+1 as varchar(20))

insert into employee values(@emp\_ID,@emp\_name,@emp\_sex,

@emp\_birth,@emp\_phone,@emp\_cardID,

@emp\_address,@emp\_govID,@emp\_studyID,@emp\_pay)

end

--（2）民族信息表

go

create trigger tr2 on gov instead of insert

as

begin

declare @gov\_ID varchar(10),@gov\_name varchar(20),@gov\_remark varchar(50)

select @gov\_ID=Max(gov\_ID) from gov

select @gov\_name=gov\_name from inserted

select @gov\_remark =gov\_remark from inserted

if(@gov\_ID is null)

set @gov\_ID='gov1'

else

set @gov\_ID='gov'+cast((cast(substring(@gov\_ID,4,4) as int)+1) as varchar(10))

if exists(select gov\_name from gov where gov\_name=@gov\_name)

rollback

else

begin

insert into gov values(@gov\_ID,@gov\_name,@gov\_remark)

end

end

drop trigger tr2

--（3）学历信息表

go

create trigger tr3 on study instead of insert

as

begin

declare @study\_ID varchar(10),@study\_name varchar(20),@study\_remark varchar(50)

select @study\_ID=Max(study\_ID) from study

select @study\_name=study\_name from inserted

select @study\_remark =study\_remark from inserted

if(@study\_ID is null)

set @study\_ID='study1'

else

set @study\_ID='study'+cast((cast(substring(@study\_ID,4,4) as int)+1) as varchar(10))

if exists(select study\_name from study where study\_name=@study\_name)

rollback

else

begin

insert into study values(@study\_ID,@study\_name,@study\_remark)

end

end

--（4）客户信息表

go

create trigger tr10 on users instead of insert

as

begin

declare

@user\_ID varchar (10) ,

@user\_name varchar (20),

@user\_sex varchar (10),

@user\_birth datetime,

@user\_phone varchar (20),

@user\_cardID varchar (20),

@user\_type varchar (50),

@house\_ID varchar (10),

@user\_record datetime

select @user\_ID=Max(user\_ID) from users

select @user\_name=user\_name from inserted

select @user\_sex=user\_sex from inserted

select @user\_birth=user\_birth from inserted

select @user\_phone=user\_phone from inserted

select @user\_cardID=user\_cardID from inserted

select @user\_type = user\_type from inserted

select @house\_ID =house\_ID from inserted

select @user\_record =getdate()

if(@user\_type = 'lend')

begin

if(@user\_ID is null)

set @user\_ID='lend1'

else

set @user\_ID='lend'+cast(cast(substring(@user\_ID,4,4) as int)+1 as varchar(20))

end

if(@user\_type = 'want')

begin

if(@user\_ID is null)

set @user\_ID='want1'

else

set @user\_ID='want'+cast(cast(substring(@user\_ID,4,4) as int)+1 as varchar(20))

end

insert into users values(@user\_ID,@user\_name,@user\_sex,

@user\_birth,@user\_phone,@user\_cardID,

@user\_type,@house\_ID,@user\_record)

end

--（5）房源信息表

--drop trigger t11

go

create trigger t11 on house instead of insert

as

begin

declare

@house\_ID varchar (10) ,

@house\_company varchar (50),

@house\_typeID varchar (10),

@house\_state varchar (10),

@house\_fitID varchar (10),

@house\_favorID varchar (10),

@house\_methodID varchar (10),

@house\_price numeric(18,0),

@house\_floorID varchar (10),

@house\_buildYear varchar (10),

@house\_area varchar (20),

@house\_remark varchar (50),

@user\_ID varchar (10)

select @house\_ID=(select Max(house\_ID) from house)

select @house\_company= house\_company from inserted

select @house\_typeID= house\_typeID from inserted

select @house\_state= house\_state from inserted

select @house\_fitID= house\_fitID from inserted

select @house\_favorID= house\_favorID from inserted

select @house\_methodID= house\_mothedID from inserted

select @house\_price= house\_price from inserted

select @house\_floorID= house\_floorID from inserted

select @house\_buildYear= house\_buildYear from inserted

select @house\_area= house\_area from inserted

select @house\_remark= house\_remark from inserted

select @user\_ID = user\_ID from inserted

if(@house\_ID is null)

set @house\_ID='hou1'

else

set @house\_ID='hou'+cast(substring(@house\_ID,4,4)+1 as varchar(10))

insert into house values

(@house\_ID,

@house\_company,

@house\_typeID,

@house\_state,

@house\_fitID,

@house\_favorID,

@house\_methodID,

@house\_price,

@house\_floorID,

@house\_buildYear,

@house\_area,

@house\_remark,

@user\_ID )

end

--（6）房型信息表

go

create trigger tr4 on type instead of insert

as

begin

declare @type\_ID varchar(10),@type\_name varchar(20),@type\_remark varchar(50)

select @type\_ID=Max(type\_ID) from type

select @type\_name=type\_name from inserted

select @type\_remark =type\_remark from inserted

if(@type\_ID is null)

set @type\_ID='type1'

else

set @type\_ID='type'+cast((cast(substring(@type\_ID,4,4) as int)+1) as varchar(10))

if exists(select type\_name from type where type\_name=@type\_name)

rollback

else

begin

insert into type values(@type\_ID,@type\_name,@type\_remark)

end

end

--（7）装修信息表

go

create trigger tr5 on fit instead of insert

as

begin

declare @fit\_ID varchar(10),@fit\_name varchar(20),@fit\_remark varchar(50)

select @fit\_ID=Max(fit\_ID) from fit

select @fit\_name=fit\_name from inserted

select @fit\_remark =fit\_remark from inserted

if(@fit\_ID is null)

set @fit\_ID='fit1'

else

set @fit\_ID='fit'+cast((cast(substring(@fit\_ID,4,4) as int)+1) as varchar(10))

if exists(select fit\_name from fit where fit\_name=@fit\_name)

rollback

else

begin

insert into fit values(@fit\_ID,@fit\_name,@fit\_remark)

end

end

--（8）朝向信息表

go

create trigger tr6 on favor instead of insert

as

begin

declare @favor\_ID varchar(10),@favor\_name varchar(20),@favor\_remark varchar(50)

select @favor\_ID=Max(favor\_ID) from favor

select @favor\_name=favor\_name from inserted

select @favor\_remark =favor\_remark from inserted

if(@favor\_ID is null)

set @favor\_ID='favor1'

else

set @favor\_ID='favor'+cast((cast(substring(@favor\_ID,4,4) as int)+1) as varchar(10))

if exists(select favor\_name from favor where favor\_name=@favor\_name)

rollback

else

begin

insert into favor values(@favor\_ID,@favor\_name,@favor\_remark)

end

end

--(9)用途信息表

go

create trigger tr7 on method instead of insert

as

begin

declare @method\_ID varchar(10),@method\_name varchar(20),@method\_remark varchar(50)

select @method\_ID=Max(method\_ID) from method

select @method\_name=method\_name from inserted

select @method\_remark =method\_remark from inserted

if(@method\_ID is null)

set @method\_ID='method1'

else

set @method\_ID='method'+cast((cast(substring(@method\_ID,4,4) as int)+1) as varchar(10))

if exists(select method\_name from method where method\_name=@method\_name)

rollback

else

begin

insert into method values(@method\_ID,@method\_name,@method\_remark)

end

end

--（10）楼层信息表

go

create trigger tr7 on floor instead of insert

as

begin

declare @floor\_ID varchar(10),@floor\_name varchar(20),@floor\_remark varchar(50)

select @floor\_ID=Max(floor\_ID) from floor

select @floor\_name=floor\_name from inserted

select @floor\_remark =floor\_remark from inserted

if(@floor\_ID is null)

set @floor\_ID='floor1'

else

set @floor\_ID='floor'+cast((cast(substring(@floor\_ID,4,4) as int)+1) as varchar(10))

if exists(select floor\_name from floor where floor\_name=@floor\_name)

rollback

else

begin

insert into floor values(@floor\_ID,@floor\_name,@floor\_remark)

end

end

--（11）求租意向表

go

create trigger t12 on intent instead of insert

as

begin

declare

@intent\_ID varchar (10) ,

@user\_ID varchar (50),

@intent\_typeID varchar (10),

@intent\_fitID varchar (10),

@intent\_favorID varchar (10),

@intent\_floorID varchar (10),

@intent\_methodID varchar (10),

@intent\_price numeric(18,0),

@intent\_area varchar (20)

select @intent\_ID=(select Max(intent\_ID) from intent)

select @user\_ID = user\_ID from inserted

select @intent\_typeID= intent\_typeID from inserted

select @intent\_fitID= intent\_fitID from inserted

select @intent\_favorID= intent\_favorID from inserted

select @intent\_floorID= intent\_floorID from inserted

select @intent\_methodID = intent\_methodID from inserted

select @intent\_price= intent\_price from inserted

select @intent\_area= intent\_area from inserted

if(@intent\_ID is null)

set @intent\_ID='int1'

else

set @intent\_ID='int'+cast(substring(@intent\_ID,4,4)+1 as varchar(10))

insert into intent values

(@intent\_ID,

@user\_ID,

@intent\_typeID,

@intent\_fitID,

@intent\_favorID,

@intent\_floorID,

@intent\_methodID,

@intent\_price,

@intent\_area

)

end

--（12）收费信息表

go

create trigger t13 on money instead of insert

as

begin

declare

@money\_ID varchar (10) ,

@money\_pay numeric(18,0),

@emp\_ID varchar (10),

@emp\_name varchar (20),

@house\_ID varchar (10),

@money\_time varchar (50),

@money\_remark varchar (100),

@lend\_ID varchar (10),

@lend\_name varchar (20),

@lend\_phone varchar (20),

@want\_ID varchar (10),

@want\_name varchar (20),

@want\_phone varchar (20)

select @money\_ID=(select Max(money\_ID) from money)

select @money\_pay = money\_pay from inserted

select @emp\_ID = emp\_ID from inserted

select @emp\_name = emp\_name from inserted

select @house\_ID = house\_ID from inserted

select @money\_time = money\_time from inserted

select @money\_remark = money\_remark from inserted

select @lend\_ID = lend\_ID from inserted

select @lend\_name = lend\_name from inserted

select @lend\_phone = lend\_phone from inserted

select @want\_ID = want\_ID from inserted

select @want\_name = want\_name from inserted

select @want\_phone = want\_phone from inserted

if(@money\_ID is null)

set @money\_ID='money1'

else

set @money\_ID='money'+cast(substring(@money\_ID,6,4)+1 as varchar(10))

insert into money values(

@money\_ID ,

@money\_pay ,

@emp\_ID ,

@emp\_name ,

@house\_ID ,

@money\_time ,

@money\_remark ,

@lend\_ID ,

@lend\_name ,

@lend\_phone ,

@want\_ID ,

@want\_name ,

@want\_phone

)

end

--（13）登录信息表

go

create trigger t14 on login instead of insert

as

begin

declare

@login\_ID varchar(10),

@emp\_ID varchar (10),

@login\_name varchar (20),

@login\_pwd varchar (15),

@login\_power varchar (10)

select @login\_ID=Max(login\_ID) from login

select @emp\_ID = emp\_ID from inserted

select @login\_name=login\_name from inserted

select @login\_pwd = login\_pwd from inserted

select @login\_power = login\_power from inserted

if(@login\_ID is null)

set @login\_ID='log1001'

else

set @login\_ID='log'+cast(cast(substring(@login\_ID,4,4) as int)+1 as varchar(20))

if exists(select login\_name from login where login\_name=@login\_name)

rollback

else

begin

insert into login values(@login\_ID, @emp\_ID ,@login\_name,@login\_pwd, @login\_power)

end

end

### 5.3.2 存储过程(举例)

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER proc [dbo].[floor]

@floor\_ID varchar(10)=null

as

begin

delete from floor where floor\_ID=@floor\_ID

end

### 5.3.3 函数

--函数

--补0

go

create function zh(@p varchar(4))

returns varchar(2)

as

begin

if len(@p)<4

begin

set @p='0'+@p

end

return @p

end

# 总结

本次数据库课程设计的结果与目的系统基本相符合，有一定的灵活性，员工可以迅速上手掌握本二手房中介管理系统。同时，本数据库系统可以自动生成编号，每个编号唯一不重复，作为主码出现，由触发器自动生成，避免了人工输入的繁琐，降低了人工输入出错的可能性。数据库中的表分的较为细致，可以更加清晰。同时，建立视图来方便进行查找、修改、删除等操作。

数据库也存在一定的问题，比如触发器的代码过于繁琐重复，每个表都需要写一个触发器来自动生成ID，函数和存储过程建立较少等。

通过这次数据库课程设计，我掌握了SQL server的运用，也进一步理解了数据库的意义，我们可以将我们所学的数据库知识和现实中的问题联系起来，使问题变得简单化，更容易操作并且找出解决措施。二手房中介管理系统中需要建的表很多，必须要注意表与表之间的联系，否则可能牵一发而动全身，造成整个数据库的失败。

我也发现这其中需要的很多知识我们都没有接触过，上网查找资料的时候发现以前所学到的仅仅是皮毛，还有很多需要掌握的东西。同时也发现有很多已经学过的东西都没有理解到位，不能灵活运用于实际，不能很好的用来解决问题，这就需要我们自己不断的大量的实践，通过不断的自学，不断地发现问题，思考问题，进而解决问题。在这个过程中我们也可以深刻理解所学知识，同时可以学到不少很实用的东西，积累自己的经验。

通过本次数据库课程课程设计，我觉得不管做什么，始终都要相信自己。不要畏惧，不要怕遇到困难，要勇于去尝试。我们以后如果从事这方面的工作也需要不断地去学习去实践，通过这次二手房中介系统的设计完成，使我对自己所学知识进一步的了解，同时通过实践发现自己的不足，对知识的掌握及运用存在诸多的不完善。所以在今后的学习中我会继续努力，争取以后做出的系统更加完善。

# 参考资料

[1] 王珊，萨师煊. 数据库系统概论（第五版）

[2] 郑人杰，马素霞. 软件工程概论（第三版）

[3] 马骏. C#程序设计及应用教程(第四版)

# 致谢

数据库课程设计到此已经基本完成。

在数据库课程设计的这段时间中，我遇到了无数的难题和障碍，都在老师和同学的帮助下圆满度过。在此，我深深的感谢我的老师和同学们。首先要感谢我的指导老师姜秀娥老师，教会了我数据库的基本知识。感谢王文玉老师教给了我软件工程设计的基本要求和文档的书写。

感谢我的同学和朋友，在我设计数据库时给我提供帮助，给我提供测试数据。感谢我的室友，谢谢他们在生活上对我的关心，在学习上给予我帮助。

感谢母校，为我提供学习场所，让我可以在图书馆心无旁骛的完成数据库的设计和实现。

感谢我的家人和朋友，在这段时间给予我精神上和物质上的支持，在我遇到困难时给予我精神上的指导。