四 川 轻 化 工 大 学 实 验 报 告

院：**计算机科学与工程学院** 课程名称：**数据库系统原理** 日期：2022.5.11

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 姓名 | **李万余，胡鹏，罗荣** | 组号 | 5 | 学号 | **2022** | 实验室 | A6-329 |
| 专业 | **软件工程** | 班号 | **2020级4班** | 老师签名 |  |
| 实验  名称 | **实验一、数据库系统分析与设计** | | | | | 成绩评定 |  |
| 所用  仪器  材料 | **计算机，**  **Word 2019，**  **Visio 2016，**  **MySQL 8.0.28，**  **Navicat for MySQL 15** | | | | | | |
| 实  验  目  的  或  要  求 | **项目名称：物业管理系统**  **一、实验目的**   1. 掌握数据库设计的基本技术。 2. 熟悉数据库设计的每个步骤中的任务和实施方法,并加深对数据库系统概念和特点的理解。 3. 熟练掌握使用 SQL语言创建数据库、表、索引和修改表结构。 4. 熟练掌握使用 SQL语言向数据库输入数据、修改数据和删除数据的操作。 5. 掌握通过MySQL命令行和Navicat观察创建结果，并进行分析。掌握Java面向对象程序设计的基本思想。   **二、实验内容及要求**   1. 要求学生根据周围的实际情况，自选一个小型的数据库应用项目并深入到应用项目的现实世界中，进行系统分析和数据库设计。例如选择学籍管理系统、图书管理系统、材料管理系统或仓库管理系统等。 2. 用关系数据库理论对自己设计的数据模型进行评价，指出合理和不足之处，提出改进的方案。 3. 通过SQL语句，创建数据库和查看数据库属性。创建表、确定表的主码和约束条件，为某些字段建立索引。查看和修改表结构。 4. 通过SQL语句，在数据库的各个表中各输入8~30条记录。要求记录不仅满足数据约束要求，还要有表间关联的记录。 5. 通过MySQL命令行完成上述操作，并用Navicat观察结果。 6. **具体内容及要求：**   需求分析叙述合理、全面；在E－R图中注意实体间联系类型的表示；数据库设计时应避免数据冗余；数据库的表中,属性应是不可再分的最小项；逻辑设计应按照转化原则及完整性约束进行规范设计，各表的联系应通过外码实现。根据数据库逻辑设计结果，在数据库中进行每一个表的定义，并添加数据入库，编写每个表的SQL定义语句及数据添加语句，注意表格建立的顺序。 | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实  验  原  理 | **三、设计过程及结果：**   1. 简要需求分析   本系统由业主模块、管理员模块两部分组成。  业主模块包括由账户登录\注册、查看消息、投诉/建议、服务申请、缴费、查看房屋信息六部分组成。  管理员模块包括由查看账户、发布消息、评价管理、维修管理、服务管理五部分组成  图 1-1 系统功能模块图   1. 概念结构设计    1. 数据实体及相关的联系分析   根据需求分析，本系统主要包含实体如下：   1. 用户信息：用户ID，密码，姓名，手机号，邮箱，激活状态； 2. 投诉/建议：投诉/建议ID，投诉/建议时间，投诉/建议内容，被投诉/建议的对象； 3. 消息：消息ID，发布时间，消息内容，消息有效期,消息类型； 4. 费用：费用ID，开始时间，缴费时间，金额； 5. 费用类型：费用类型，费用类型ID； 6. 维修任务：维修ID，维修状态，维修内容； 7. 停车位：停车位ID，停车位位置，购买时间，停车位状态； 8. 管理员信息：管理员ID，密码，姓名，手机号，邮箱，激活状态； 9. 房屋：房屋ID，房屋位置，房屋状态；   本系统实体之间的联系如下：   1. 一个用户可以发送多条投诉/建议，一条投诉/建议只能由一个用户发出，即1：n关系； 2. 一个管理员可以响应多条投诉/建议，一条投诉/建议只能被一个管理员响应，即1：n关系； 3. 一个管理员可以发布多条消息，一条消息只能由一个管理员发布，即1：n关系； 4. 一个房屋可以产生多项费用，一项费用只能由一个房屋用户产生，即1：n关系； 5. 一个用户可以缴纳多项费用，一项费用只能由一个用户缴纳，即1：n关系； 6. 一个费用类型有多条费用，一条费用只属于一个费用类型，即1：n关系； 7. 一个用户可以发布多条维修任务，一个维修任务只能被一个用户发布，即1：n关系； 8. 一个管理员可以处理多条维修任务，一个维修任务可以被管理员多次处理，即n：m关系； 9. 一间房屋可以购买多个停车位，一个停车位只能被一个用户拥有，即1：n关系； 10. 一个用户可以拥有多个房屋，一个房屋可以属于多个用户，即n:m关系 11. 一个管理员可以更新多个房屋信息，一个房屋信息可以被一个管理员多次更新，即n：m关系； 12. 一个上级管理员可以有多个子管理员，一个下级管理员只能有一个上级管理员，即1：n关系；     1. E-R模型设计     图2-1 用户实体图 图2-2 管理员实体图    图2-3 房屋实体图 图2-4 投诉/建议实体图    图2-5 消息实体图 图2-6 费用实体图    图2-7 停车位实体图 图2-8 维修任务实体图    图2-8 系统总体E-R图   1. 逻辑结构设计    1. 关系模式建立   概念模型已经确定了实体以及属性联系等，逻辑结构设计主要是针对E-R模型转化为关系模式，根据转化规则，逻辑模型设计结果如下，其中划线的属性为主码。   1. 用户（用户ID，姓名，密码，激活状态，手机号，邮箱）   管理员（管理员ID，姓名，密码，激活状态，手机号，邮箱，父管理员ID）  房屋（房屋ID、位置、房屋状态、管理员ID）  投诉/建议（投诉/建议ID、被投诉/建议的对象、时间、内容、管理员ID, 用户ID）  消息（消息ID、内容、时间、有效期、管理员ID、消息类型）  费用（费用ID、开始时间、缴费时间、金额、房屋ID、用户ID、费用类型ID）  费用类型（费用类型ID、费用类型）  维修任务（维修ID、维修状态、内容、用户ID）  停车位（停车位ID、停车位位置、购买时间、停车位状态、房屋ID）  用户\_房屋关联（用户ID、房屋ID、购房时间、入住时间）  房屋\_管理员关联（更新ID、更新时间、更新内容、房屋ID、管理员ID）  管理员\_维修任务关联(维修记录ID、维修状态、维修任务分配时间、维修完成时间、维修ID、管理员ID)   * 1. 数据关系表结构   用户信息表(UserInfo)：用于记录用户的信息，手机号和邮箱必须一项非空且唯一，激活状态默认为未激活，主要数据结构如表3-1所示：  表3-1 用户信息表(UserInfo)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | UserID | int | 11 | 自增，主键 | 用户ID | | UserName | varchar | 12 | 非空 | 姓名 | | UserPassWord | varchar | 128 | 非空 | 密码 | | IsActive | tinyint | 1 | 非空，默认0 | 是否激活（0：未激活，1：已激活） | | UserPhone | varchar | 11 | 唯一 | 手机号 | | UserEmail | varchar | 20 | 唯一 | 邮箱 |   管理员信息表(AdminInfo)：用于记录管理员的信息，手机号和邮箱必须一项非空且唯一，激活状态默认为未激活，主要数据结构如表3-2所示：  表3-2 管理员信息表(AdminInfo)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | AdminID | int | 11 | 自增，主键 | 管理员ID | | AdminName | varchar | 12 | 非空 | 姓名 | | AdminPassWord | varchar | 128 | 非空 | 密码 | | IsActive | tinyint | 1 | 非空，默认0 | 是否激活（0：未激活，1：已激活） | | AdminPhone | varchar | 11 | 唯一 | 手机号 | | AdminEmail | varchar | 20 | 唯一 | 邮箱 | | PAdminID | int | 11 | 默认NULL | 父管理员ID |   房屋信息表(HouseInfo)：用于记录房屋的基本信息，房屋状态有无用户和有用户两种状态，默认为无人状态。主要数据结构如表3-3所示：  表3-3 房屋信息表(HouseInfo)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | HouseID | int | 11 | 自增，主键 | 房屋ID | | Location | varchar | 20 | 非空 | 位置 | | HouseState | tinyint | 1 | 非空，默认0 | 房屋状态（0：无用户，1：有用户） | | House\_AdminID | int | 11 | 非空，外键 | 管理员ID |   投诉/建议信息表(Complaint\_and\_AdviceInfo)：用于记录用户对小区各类服务的投诉/建议信息。主要数据结构如表3-4所示：  表3-4投诉/建议信息表(Complaint\_and\_AdviceInfo)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | C\_AID | int | 11 | 自增，主键 | 投诉/建议ID | | C\_ATime | Timestamp |  | 非空，默认CURRENT\_TIMESTAMP | 时间 | | C\_AObject | varchar | 20 |  | 对被投诉/建议的对象的描述 | | C\_AContent | Text |  | 非空 | 内容 | | C\_A\_UserID | int | 11 | 非空，外键 | 用户ID | | C\_A\_AdminID | int | 11 | 外键 | 管理员ID |   消息信息表(NewsInfo)：用于记录管理员发送的公告和信息，类型分为三类：单用户，多用户，全体用户。主要数据结构如表3-5所示：  表3-5 消息信息表(NewsInfo)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | NewsID | int | 11 | 自增，主键 | 消息ID | | NewsTime | Timestamp |  | 非空，默认CURRENT\_TIMESTAMP | 发送时间 | | NewsContent | Text |  | 非空 | 内容 | | News\_AdminID | int | 11 | 非空，外键 | 管理员ID | | NewsType | tinyint | 1 | 非空，默认0 | 消息类型（0：全体用户，1：多用户，2：单用户） | | NewsEndTime | Timestamp |  |  | 到期时间 |   费用类型信息表(CosttypeInfo)：用于记录各种费用类型信息。主要数据结构如表3-6所示：  表3-6 费用类型信息表(CosttypeInfo)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | CosttypeID | int | 11 | 自增，主键 | 费用类型ID | | CostType | varchar | 20 | 非空 | 费用类型 |   费用信息表(CostInfo)：用于记录各种费用信息。主要数据结构如表3-7所示：  表3-7费用信息表(CostInfo)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | CostID | int | 11 | 自增，主键 | 费用ID | | CostStartTime | Timestamp |  | 默认CURRENT\_TIMESTAMP | 开始时间 | | PayTime | Timestamp |  |  | 缴费时间 | | Money | double | 8 | 非空 | 金额 | | HouseID | int | 11 | 非空，外键 | 房屋ID | | UserID | int | 11 | 非空，外键 | 用户ID |   维修任务信息表(TaskInfo)：用于记录维修信息，维修状态分为（未开始，已开始，已完成）三种，默认为未开始。主要数据结构如表3-8所示：  表3-8维修任务信息表(TaskInfo)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | TaskID | int | 11 | 自增，主键 | 维修ID | | TaskState | tinyint | 1 | 非空，默认0 | 维修状态（0：未开始，1：已开始，2：已完成） | | TaskContent | Text |  | 非空 | 维修内容 | | Task\_UserID | int | 11 | 非空，外键 | 用户ID |   停车位信息表(PCarInfo) ：用于记录停车位信息，状态分为三种（已购买 ，未购买）默认为空闲，只有当状态为已预定时，用户ID为非空。主要数据结构如表3-9所示：  表3-9停车位信息表(PCarInfo)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | PCarID | int | 11 | 自增，主键 | 停车ID | | PCarState | tinyint | 1 | 非空，默认0 | 停车位状态（1：已购买 0：未购买） | | PCarLocation | varchar | 20 | 非空 | 位置 | | BuyTime | Timestamp |  |  | 购买时间 | | HouseID | int | 11 | 外键 | 房屋ID |   用户\_房屋关联表(User\_HouseRelation)：用于记录房屋的归属用户，购买信息。主要数据结构如表3-10所示：  表3-10 用户\_房屋关联表(User\_HouseRelation)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | UH\_Relation\_UserID | int | 11 | 外键，主键 | 用户ID | | UH\_Relation\_HouseID | int | 11 | 外键，主键 | 房屋ID | | GetHouseTime | Timestamp |  | 非空 | 购房时间 | | MoveTime | Timestamp |  | 非空 | 入住时间 |   房屋\_管理员关联(UpdataRelation)：用于记录管理员更新房屋信息（一个管理员可能多次更新同一房屋信息）。主要数据结构如表3-11所示：  表3-11 房屋\_管理员关联(UpdataRelation)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | UpdataID | int | 11 | 自增，主键 | 更新ID | | UpdataContent | Text |  | 非空 | 更新内容 | | UpdataTime | Timestamp |  | 非空，默认CURRENT\_TIMESTAMP | 更新时间 | | Updata\_HouseID | int | 11 | 非空，外键 | 房屋ID | | Updata\_AdminID | int | 11 | 非空，外键 | 管理员ID |   管理员\_维修任务关联(Admin\_TaskRelation)：用于记录管理员分配维修任务信息，维修状态分为正在维修，和维修完成，维修失败三种，默认正在维修，当维修完成时，维修完成时间为非空。主要数据结构如表3-12所示：  表3-12 管理员\_维修任务关联(Admin\_TaskRelation)表结构   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 字段名 | 类型 | 长度 | 约束条件 | 备 注 | | Admin\_TaskID | int | 11 | 自增，主键 | 维修信息ID | | ATaskState | tinyint | 1 | 非空，默认0 | 维修状态（0：正在维修，1：维修完成，2：维修失败） | | TaskStartTime | Timestamp |  | 非空，默认CURRENT\_TIMESTAMP | 维修任务分配时间 | | TaskEndTime | Timestamp |  |  | 维修完成时间 | | Admin\_Task\_TaskID | int | 11 | 非空，外键 | 维修ID | | Admin\_Task\_AdminID | int | 11 | 非空，外键 | 管理员ID |  1. 数据库及关系表的建立 2. 创建数据库（property\_management）   创建数据库SQL语句  Create database property\_management;  用户信息表（userinfo）  （1）创建表SQL语句  CREATE TABLE userinfo(  userid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '用户ID',  username VARCHAR(12) NOT NULL COMMENT '用户名',  userpassword VARCHAR(255) NOT NULL COMMENT '密码',  isactive TINYINT NOT NULL COMMENT '是否激活(0:未激活, 1:已激活)' DEFAULT(0),  userphone VARCHAR(11) UNIQUE COMMENT '手机号码',  useremail VARCHAR(50) UNIQUE COMMENT '邮箱'  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO userinfo (`userid`, `username`, `userpassword`, `isactive`, `userphone`, `useremail`) VALUES (1, '磨暮', 'jjfN8m$J(nJ9rX@auPwe', 1, '15362264107', 'wwmgbp@qq.com');  （4）插入10条语句后的数据表    管理员信息表(AdminInfo)  （1）创建表SQL语句  CREATE TABLE admininfo(  adminid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '管理员ID',  adminname VARCHAR(12) NOT NULL COMMENT '管理员名',  adminpassword VARCHAR(255) NOT NULL COMMENT '密码',  isactive TINYINT NOT NULL COMMENT '是否激活(0:未激活, 1:已激活)' DEFAULT(0),  adminphone VARCHAR(11) UNIQUE COMMENT '手机号码',  adminemail VARCHAR(50) UNIQUE COMMENT '邮箱',  padminid INT DEFAULT(NULL) COMMENT '父管理员ID',  FOREIGN KEY (padminid) REFERENCES admininfo(adminid)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO admininfo ('adminid', 'adminname', 'adminpassword', 'isactive', 'adminphone', 'adminemail', 'padminid') VALUES (21, '班代天', 'AZHR1p00XcXjy&k@$9\r\n', 1, '18414137037', 'thlrq@263.net', NULL), ('adminid', 'adminname', 'adminpassword', 'isactive', 'adminphone', 'adminemail', 'padminid') VALUES (24, '都凡之', 'pumusnDX(viM^sL', 1, '13891263138', 'citwbdp@265.com', 21), ('adminid', 'adminname', 'adminpassword', 'isactive', 'adminphone', 'adminemail', 'padminid') VALUES (2, '蒙舒方', '!ZBfutlizhT6m~TcB8le', 1, '15734882709', 'hdala@56.com', 21), ('adminid', 'adminname', 'adminpassword', 'isactive', 'adminphone', 'adminemail', 'padminid') VALUES (7, '戢齐敏', 'Wb&Hp!r%s%XhgsS&fDM', 1, '18029828973', 'nibao@163.com', 21), ('adminid', 'adminname', 'adminpassword', 'isactive', 'adminphone', 'adminemail', 'padminid') VALUES (15, '曾合美', '^g9&&ReWu4o', 1, '17377237638', 'wbu@163.net', 21);  （4）插入10条语句后的数据表     1. 房屋信息表(HouseInfo)   （1）创建表SQL语句  CREATE TABLE houseinfo(  houseid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '房屋ID',  location VARCHAR(20) NOT NULL COMMENT '位置',  housestate TINYINT NOT NULL COMMENT '房屋状态(0:无用户, 1:有用户)' DEFAULT 0,  house\_adminid INT NOT NULL COMMENT '管理员ID',  FOREIGN KEY (house\_adminid) REFERENCES admininfo(adminID)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO houseinfo (`houseid`, `location`, `housestate`, `house\_adminid`) VALUES (1, '清溪小区4 栋8楼20号', 1, 8),(2, '清溪小区5 栋2楼12号', 1, 16);  （4）插入10条语句后的数据表     1. 投诉/建议信息表(Complaint\_and\_AdviceInfo)   （1）创建表SQL语句  CREATE TABLE complaint\_and\_acviceinfo(  c\_aid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '投诉/建议ID',  c\_atime TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '时间',  c\_aobject VARCHAR(20) COMMENT '对被投诉/建议的对象的描述',  c\_acontent TEXT NOT NULL COMMENT '内容',  c\_a\_userid INT NOT NULL COMMENT '用户ID',  c\_a\_adminid INT COMMENT '管理员ID',  FOREIGN KEY (c\_a\_userid) REFERENCES userinfo(userid),  FOREIGN KEY (c\_a\_adminid) REFERENCES admininfo(adminID)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO complaint\_and\_acviceinfo (`c\_aid`, `c\_atime`, `c\_aobject`, `c\_acontent`, `c\_a\_userid`, `c\_a\_adminid`) VALUES (1, '2018-03-21 06:25:59', 'xxxxxx', 'xxxxxxx', 14, 20),(2, '2017-11-07 09:23:03', 'xxxxxx', 'xxxxxxx', 55, 25);  （4）插入10条语句后的数据表     1. 消息信息表(NewsInfo)   （1）创建表SQL语句  CREATE TABLE newsinfo(  newsid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '消息ID',  newstime TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '发送时间',  newscontent TEXT NOT NULL COMMENT '内容',  newstype TINYINT NOT NULL DEFAULT 0 COMMENT '房屋状态(0:全体用户, 1:多用户, 2:单用户)',  newsendtime TIMESTAMP COMMENT '到期时间',  news\_adminid INT NOT NULL COMMENT '管理员ID',  FOREIGN KEY (news\_adminid) REFERENCES admininfo(adminid)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO newsinfo (`newsid`, `newstime`, `newscontent`, `news\_adminid`, `newstype`, `newsendtime`) VALUES (1, '2020-12-16 08:37:21', 'xxxx', 20, 1, '2020-04-18 07:03:44'),(2, '2019-05-19 11:59:08', 'xxxx', 25, 2, NULL);  （4）插入10条语句后的数据表     1. 费用类型信息表(CosttypeInfo)   （1）创建表SQL语句  CREATE TABLE costtypeinfo(  costtypeid int AUTO\_INCREMENT PRIMARY KEY COMMENT '费用类型ID',  costtype VARCHAR(20) NOT NULL COMMENT '费用类型'  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO costtypeinfo (`costtypeid`, `costtype`) VALUES (1, '水费'),(2, '电费');  （4）插入10条语句后的数据表     1. 费用信息表(CostInfo)   （1）创建表SQL语句  CREATE TABLE costinfo(  costid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '费用ID',  coststarttime TIMESTAMP DEFAULT CURRENT\_TIMESTAMP COMMENT '开始时间',  paytime TIMESTAMP COMMENT '缴费时间',  costmoney DOUBLE NOT NULL COMMENT '金额',  cost\_houseid INT NOT NULL COMMENT '房屋ID',  cost\_userid INT NOT NULL COMMENT '用户ID',  cost\_costtypeid INT NOT NULL COMMENT '费用类型ID',  FOREIGN KEY (cost\_houseid) REFERENCES houseinfo(houseid),  FOREIGN KEY (cost\_userid) REFERENCES userinfo(userid),  FOREIGN KEY (cost\_costtypeid) REFERENCES costtypeinfo(costtypeid)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO costinfo (`costid`, `coststarttime`, `paytime`, `costmoney`, `cost\_userid`, `cost\_houseid`, `cost\_costtypeid`) VALUES (1, '2021-06-08 12:33:55', '2021-06-13 07:43:55', 170.083920017742514, 7, 20, 2),(2, '2020-09-15 08:09:39', '2020-09-16 10:03:03', 636.686875427754899, 36, 7, 3);  （4）插入10条语句后的数据表     1. 维修任务信息表(TaskInfo)   （1）创建表SQL语句  CREATE TABLE taskinfo(  taskid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '维修ID',  taskstate TINYINT NOT NULL DEFAULT 0 COMMENT '维修状态(0:未开始, 1:已开始, 2:已完成)',  taskcontent TEXT NOT NULL COMMENT '维修内容',  task\_userid INT NOT NULL COMMENT '用户ID',  FOREIGN KEY (task\_userid) REFERENCES userinfo(userid)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO taskinfo (`taskid`, `taskstate`, `taskcontent`, `task\_userid`) VALUES (1, 2, '暖气故障', 70),(2, 2, '修门', 77);  （4）插入10条语句后的数据表     1. 停车位信息表(PCarInfo)   （1）创建表SQL语句  CREATE TABLE pcarinfo(  pcarid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '停车位ID',  pcarstate TINYINT NOT NULL DEFAULT 0 COMMENT '停车位状态(0:空闲, 1:已停车, 2:已预定)',  pcarloction VARCHAR(20) NOT NULL COMMENT '位置',  buytime TIMESTAMP COMMENT '购买时间',  pcar\_houseid INT COMMENT '房屋ID',  FOREIGN KEY (pcar\_houseid) REFERENCES houseinfo(houseid)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO pcarinfo (`pcarid`, `pcarstate`, `pcarloction`, `buytime`, `pcar\_houseid`) VALUES (1, 1, '1', '2017-03-26 17:31:07', 26),(2, 1, '2', '2021-04-19 05:34:36', 22);  （4）插入10条语句后的数据表     1. 用户\_房屋关联表(User\_HouseRelation)   （1）创建表SQL语句  CREATE TABLE user\_houserelation(  uh\_relation\_userid INT COMMENT '用户ID',  uh\_relation\_houseid INT COMMENT '房屋ID',  gethousetime TIMESTAMP COMMENT '购房时间',  movetime TIMESTAMP COMMENT '入住时间',  PRIMARY KEY (uh\_relation\_userid, uh\_relation\_houseid),  FOREIGN KEY (uh\_relation\_userid) REFERENCES userinfo(userid),  FOREIGN KEY (uh\_relation\_houseid) REFERENCES houseinfo(houseid)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO user\_houserelation (`uh\_relation\_userid`, `uh\_relation\_houseid`, `gethousetime`, `movetime`) VALUES (30, 40, '2018-07-25 08:54:34', '2018-10-13 03:20:08'),(64, 40, '2017-12-13 01:51:14', '2018-11-25 11:01:23');  （4）插入10条语句后的数据表     1. 房屋\_管理员关联(UpdataRelation)   （1）创建表SQL语句  CREATE TABLE updatarelation(  updataid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '更新ID',  updatacontent TEXT NOT NULL COMMENT '更新内容',  updatatime TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '更新时间',  updata\_houseid INT NOT NULL COMMENT '房屋ID',  updata\_adminid INT NOT NULL COMMENT '管理员ID',  FOREIGN KEY (updata\_houseid) REFERENCES houseinfo(houseid) ,  FOREIGN KEY (updata\_adminid) REFERENCES admininfo(adminid)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO updatarelation (`updataid`, `updatacontent`, `updatatime`, `updata\_houseid`, `updata\_adminid`) VALUES (1, 'xxxx', '2018-01-17 06:07:09', 39, 8),(2, 'xxxx', '2017-06-19 12:07:37', 13, 8);  （4）插入10条语句后的数据表     1. 管理员\_维修任务关联(Admin\_TaskRelation)   （1）创建表SQL语句  CREATE TABLE admin\_taskrelation(  admin\_taskid INT AUTO\_INCREMENT PRIMARY KEY COMMENT '维修信息',  ataskstate TINYINT NOT NULL DEFAULT 0 COMMENT '维修状态(0:正在维修, 1:维修完成, 2:维修失败)',  taskstarttime TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '维修任务分配时间',  taskendtime TIMESTAMP COMMENT '维修完成时间',  admin\_task\_taskid INT NOT NULL COMMENT '维修ID',  admin\_task\_adminid INT NOT NULL COMMENT '管理员ID',  FOREIGN KEY (admin\_task\_taskid) REFERENCES taskinfo(taskid),  FOREIGN KEY (admin\_task\_adminid) REFERENCES admininfo(adminid)  );  （2）产生表格式    （3）插入数据SQL语句  INSERT INTO admin\_taskrelation (`admin\_taskid`, `ataskstate`, `taskstarttime`, `taskendtime`, `admin\_task\_taskid`, `admin\_task\_adminid`) VALUES (67, 1, '2020-02-23 21:48:25', '2017-12-11 18:03:04', 61, 13),(68, 1, '2018-04-02 03:22:27', '2021-02-21 13:53:33', 62, 13);  （4）插入10条语句后的数据表 |
| 附  原  始  实  验  记  录 | **四、实验总结**  （1）用关系数据库理论对自己设计的数据模型进行评价，指出合理和不足之处，提出改进的方案。  数据库的基本结构分三个层次，即物理数据层、概念数据层、逻辑数据层。  在概念数据层和逻辑数据层：及构建E—R模型和建立数据库时，经过小组讨论和老师修正已具有较强的合理性，在现在的模式下大大减小了数据的冗余度。  不知道能不能很好的契合后续的开发  改进方案：在以后的开发过程中对数据库进行合理的修改  （2）在关系定义和数据装载过程中遇到了哪些问题，是怎么解决的。  1.在创建管理员信息表时外键问题：  解决方法：在创建表时，先添加父管理员，在添加子管理员。  2.：导入数据时因为外键导致导入失败  解决方法：忽略外键。 |