广东东软学院

学生实验报告

**实验课程名称：数据库系统原理**

**实验项目名称：XXXX系统的数据库设计**

**实验类型：综合性**

**指导教师：**

**实验日期： 年 月 日**

|  |  |  |  |
| --- | --- | --- | --- |
| **学生姓名** | **此处填组长信息** | **学 号** | **此处填组长信息** |
| **班 级** |  | **专业名称** |  |
| **实验组**  **其他成员** | **此处填组员信息** | | |
| **实验地点** |  | | |
| **实验成绩**  **（教师签名）** |  | | |

|  |
| --- |
| **实验目的与要求**  1. 了解数据库设计的过程和主要任务;  2. 掌握数据库设计过程中的需求分析方法；  3. 理解概念模型，并掌握ER图的设计和画法;  4. 理解数据库规范理论，范式的要求，优化ER图，要至少满足3NF；  5. 掌握概念模型到逻辑模型的转换规则;  6. 掌握利用Power Disgn工具或者Microsoft Visio进行数据库设计的全过程  7. 熟练编写数据库设计说明书 |
| **实验原理与内容**  **1． 实验原理：**  1.数据库设计的过程  2.关系数据库规范理论  **2 . 实验内容：**  1. 分组并选择自选项目  2. 项目组可以采用角色扮演法，模拟需求调研的过程进行需求信息的收集，同时完成：选用项目的基本需求信息收集，并画出项目的系统功能图  3. 项目组根据项目的需求，将系统中的关键流程，用数据流图表达出来，要求每个项目组2-3个数据流图。  4. 根据需求和数据流图，描述系统中的重要数据存储和数据项  5. 根据以上需求，进行系统的概念设计，要求：  在数据流图中， 组织者、数据存储作为实体对象，数据字典中的数据组成作为实体的属性  设计唯一标识符、属性的数据类型、是否可以为空等，分析实体之间的关系，化解多对多的关系  6. 运用范式要求，规范化系统ER模型，要求，至少满足3NF，确定最终的项目ER图  7. 使用概念模型转换规则，将ER模型转换为逻辑结构表，结构表的模板请参见实验三报告内容。  8. 以上过程中的画图和转换，建议使用Power Disgner16.5或者Microsoft Visio完成。 |
| **实验设备与软件环境**  1.Windows XP Professional SP3/Windows7  2.SQL Server 2012、Power Disgner16.5或者Microsoft Visio |
| **实验过程与结果（**可以参考下面的结构来写**）**  1 需求分析  1.1 系统功能图    1.2 数据字典   * **宠物表 (Pet)**   + PetID：宠物ID（主键）   + Name：宠物名字   + Species：宠物种类（如猫、狗）   + Breed：品种   + Age：年龄   + OwnerID：主人ID（外键） * **主人表 (Owner)**   + OwnerID：主人ID（主键）   + Name：主人名字   + Phone：联系方式   + Address：地址 * **健康档案表 (HealthRecord)**   + RecordID：记录ID（主键）   + PetID：宠物ID（外键）   + Description：健康记录描述   + Date：记录日期 * **预约表 (Appointment)**   + AppointmentID：预约ID（主键）   + PetID：宠物ID（外键）   + ServiceType：服务类型（如洗澡、看病）   + AppointmentDate：预约日期   + Notes：备注   2 数据库概念设计  2.1 数据库E-R设计图    2.2 E-R设计图说明   1. 一个主人可以有多个宠物（Owner 和 Pet 的关系是一对多）。 2. 每个宠物可以有多个预约服务（Pet 和 Appointment 是一对多）。   3 数据库逻辑设计  3.1 数据库逻辑模式  **1. 关系模式**  **(1) 主人表 Owner**  关系模式：  Owner(OwnerID,Name,Phone,Address)Owner(\text{OwnerID}, \text{Name}, \text{Phone}, \text{Address})Owner(OwnerID,Name,Phone,Address)   * **主键 (Primary Key)**: OwnerID * **属性说明**:   + OwnerID: 主人ID，唯一标识每个主人。   + Name: 主人姓名。   + Phone: 主人联系方式。   + Address: 主人地址。   **(2) 宠物表 Pet**  关系模式：  Pet(PetID,Name,Species,Breed,Age,OwnerID)Pet(\text{PetID}, \text{Name}, \text{Species}, \text{Breed}, \text{Age}, \text{OwnerID})Pet(PetID,Name,Species,Breed,Age,OwnerID)   * **主键 (Primary Key)**: PetID * **外键 (Foreign Key)**: OwnerID 引用 Owner(OwnerID) * **属性说明**:   + PetID: 宠物ID，唯一标识每个宠物。   + Name: 宠物名字。   + Species: 宠物种类（如狗、猫）。   + Breed: 宠物品种（如金毛、布偶）。   + Age: 宠物年龄。   + OwnerID: 宠物的主人ID。   关系：   * 每个宠物属于一个主人 (OwnerID 是外键，指向 Owner 表的主键)。 * 一个主人可以拥有多个宠物。   **(3) 健康档案表 HealthRecord**  关系模式：  HealthRecord(RecordID,PetID,Description,Date)HealthRecord(\text{RecordID}, \text{PetID}, \text{Description}, \text{Date})HealthRecord(RecordID,PetID,Description,Date)   * **主键 (Primary Key)**: RecordID * **外键 (Foreign Key)**: PetID 引用 Pet(PetID) * **属性说明**:   + RecordID: 健康记录ID，唯一标识每条健康档案。   + PetID: 宠物ID，标识记录属于哪个宠物。   + Description: 健康记录的描述内容（如“接种疫苗”）。   + Date: 健康记录的日期。   关系：   * 每条健康记录对应一个宠物 (PetID 是外键，指向 Pet 表的主键)。 * 一个宠物可以有多条健康记录。   **(4) 预约表 Appointment**  关系模式：  Appointment(AppointmentID,PetID,ServiceType,AppointmentDate,Notes)Appointment(\text{AppointmentID}, \text{PetID}, \text{ServiceType}, \text{AppointmentDate}, \text{Notes})Appointment(AppointmentID,PetID,ServiceType,AppointmentDate,Notes)   * **主键 (Primary Key)**: AppointmentID * **外键 (Foreign Key)**: PetID 引用 Pet(PetID) * **属性说明**:   + AppointmentID: 预约记录ID，唯一标识每条预约。   + PetID: 宠物ID，标识预约是为哪个宠物服务。   + ServiceType: 服务类型（如洗澡、体检）。   + AppointmentDate: 预约日期。   + Notes: 预约的备注信息。   关系：   * 每条预约记录对应一个宠物 (PetID 是外键，指向 Pet 表的主键)。 * 一个宠物可以有多条预约记录。   **2. 关系间的连接**  以下是表之间的关系描述：   * **Owner 和 Pet**: OwnerID 是主键，Pet 表中的 OwnerID 是外键。   + 一个主人可以有多个宠物。   + 一个宠物只能属于一个主人。 * **Pet 和 HealthRecord**: PetID 是主键，HealthRecord 表中的 PetID 是外键。   + 一个宠物可以有多条健康记录。   + 每条健康记录属于一个宠物。 * **Pet 和 Appointment**: PetID 是主键，Appointment 表中的 PetID 是外键。   + 一个宠物可以有多条预约记录。   + 每条预约记录属于一个宠物。   4 数据库表设计  4.1 数据库表  涉及的内容：定义数据库、表 及字段命名规范；为表中字段选择合适的数据类型；建立数据库结构（DML/DDL）  （要求有建表语句，以及插入、删除、修改、更新语句）  -- 创建数据库  CREATE DATABASE cw;  USE cw;  -- 创建主人表  CREATE TABLE Owner (  OwnerID INT PRIMARY KEY IDENTITY(1,1), -- 主键，自动递增  Name NVARCHAR(50) NOT NULL, -- 主人名字  Phone NVARCHAR(15) NOT NULL, -- 联系方式  Address NVARCHAR(100) -- 地址  );  -- 创建宠物表  CREATE TABLE Pet (  PetID INT PRIMARY KEY IDENTITY(1,1), -- 主键，自动递增  Name NVARCHAR(50) NOT NULL, -- 宠物名字  Species NVARCHAR(50) NOT NULL, -- 宠物种类  Breed NVARCHAR(50), -- 品种  Age INT, -- 年龄  OwnerID INT NOT NULL, -- 主人ID（外键）  FOREIGN KEY (OwnerID) REFERENCES Owner(OwnerID) ON DELETE CASCADE  );  -- 创建健康档案表  CREATE TABLE HealthRecord (  RecordID INT PRIMARY KEY IDENTITY(1,1), -- 主键，自动递增  PetID INT NOT NULL, -- 宠物ID（外键）  Description NVARCHAR(255), -- 健康记录描述  Date DATE NOT NULL, -- 记录日期  FOREIGN KEY (PetID) REFERENCES Pet(PetID) ON DELETE CASCADE  );  -- 创建预约表  CREATE TABLE Appointment (  AppointmentID INT PRIMARY KEY IDENTITY(1,1), -- 主键，自动递增  PetID INT NOT NULL, -- 宠物ID（外键）  ServiceType NVARCHAR(50) NOT NULL, -- 服务类型  AppointmentDate DATE NOT NULL, -- 预约日期  Notes NVARCHAR(255), -- 备注  FOREIGN KEY (PetID) REFERENCES Pet(PetID) ON DELETE CASCADE  );  -- 插入主人数据  INSERT INTO Owner (Name, Phone, Address)  VALUES  ('张三', '12345678901', '北京市朝阳区'),  ('李四', '98765432101', '上海市浦东新区');  -- 插入宠物数据  INSERT INTO Pet (Name, Species, Breed, Age, OwnerID)  VALUES  ('小花', '狗', '金毛', 3, 1),  ('小黑', '猫', '布偶', 2, 1),  ('旺财', '狗', '柴犬', 4, 2);  -- 插入健康档案数据  INSERT INTO HealthRecord (PetID, Description, Date)  VALUES  (1, '接种疫苗', '2024-01-01'),  (2, '感冒治疗', '2024-02-15');  -- 插入预约数据  INSERT INTO Appointment (PetID, ServiceType, AppointmentDate, Notes)  VALUES  (1, '洗澡', '2024-03-01', '预约上午时间'),  (3, '体检', '2024-03-05', '检查牙齿健康'); |
| **操作异常问题与解决方案**  **1. 外键约束引发的异常**  **问题描述**  当插入或删除数据时，外键约束可能导致以下异常：   * 尝试插入一条 Pet 数据，但关联的 OwnerID 在 Owner 表中不存在。 * 尝试删除一条 Owner 数据时，关联的 Pet 数据存在外键依赖而无法删除。   **原因分析**   * 外键约束强制要求父表中必须存在对应的主键数据，或者在删除父表数据时需要处理子表的依赖。   **解决方案**   1. **插入数据前检查依赖表：**    * 在插入 Pet 数据之前，确保 OwnerID 已存在于 Owner 表中。 |
| **实验总结**  在本次实验中，我们成功设计并实现了一个宠物管理系统数据库。从需求分析到E-R图设计，再到数据库表的创建和数据操作，每一步都致力于模拟实际业务需求。我们遇到了一些典型问题，例如外键约束导致的数据插入失败、日期格式错误及性能瓶颈。这些问题通过调整数据插入顺序、严格控制数据格式以及引入适当的索引得到解决。此外，实验过程中深入了解了外键约束在维护数据库完整性中的作用以及SQL查询优化的重要性。总的来说，本次实验不仅提高了我们对数据库设计和操作的理解，也增强了处理实际数据库问题的能力。 |