西安建筑科技大学

课程设计（论文）

课程名称：信息系统综合设计

题　　目：员工人事管理系统

院　（系）：信控学院

专业班级：计算机 2003

姓　　名：刘一泽

学　　号：2209060331

指导教师：贺秦禄

2025 年 7　月　4 日

西安建筑科技大学课程设计（论文）任务书

专业班级：计算机 2203 学生姓名：刘一泽 指导教师（签名）：

一、课程设计（论文）题目：员工人事信息管理系统

二、本次课程设计（论文）应达到的目的

1. 使学生复习关系型数据库的相关理论知识，运用所学开展信息管理对象的“实体-关联”分析，设计信息存储的“关系数据库、数据表”。达到“根据需求能提出数据存储方案”的目的。

2. 使学生能在所设计的关系表上，规划相匹配的信息管理功能，对每个信息管理功能分析其详细处理流程。达到“根据需求和实际计算机工程的业务流程，提出信息管理技术方案”的目的。

3. 在上述工作基础上，使学生选择“数据库管理软件、程序开发软件、数据库接口 工具”，开展数据库实现、信息管理系统开发等方面的学习和锻炼。达到“用模块化程序设计思想进行程序系统设计、模块编程、测试”目的。积累数据库应用程序的综合开发能力，提高创新意识、创新能力。

三、本次课程设计（论文）任务的主要内容和要求（包括原始数据、技术参数、设计要求等）

1. 员工管理系统应设计的主要功能和数据要求如下：

(1) 用户管理：将系统用户分为“员工类、管理员类”，员工只能查询和修改自己的基本信息(包括年龄、性别、部门、邮箱、家庭住址、联系方式、家庭成员等)；管理员可以对员工指派工作部门、工作岗位、薪酬。

(2) 员工管理：设计员工对自己的基本信息进行查询和修改的功能。管理员可以按照身份证号和姓名添加、修改、注销员工，可以对指派员工的工作部门、工作岗位、薪酬；当然管理员需要维护公司的部门信息、岗位设置、薪酬标准等信息；对新员工、变动部门或岗位的员工，发出报到通知信息。相应的，需为员工设计报到后的“到岗确认”功能。

(3) 员工批量导入：设计管理员对多条员工基本信息的批量数据导入功能；假设用户名单格式为 Excel 文件。

(4) 查询模块：设计管理员按身份证号、姓名、电子邮件、所属部门等条件完成员工信息的精确查询和模糊查询功能。

(5) 统计模块：设计管理员对各部门员工新入职、离职情况以及分部门每月薪酬总额进行分析与统计的功能。

2. 课程设计要求：

(1) 学生以小组的形式进行合作设计、分工开发，每个小组选举一名组长，组织小组的日常设计；课设结束时，选出一位陈述总体设计、每人答辩自己分工设计部分。

(2) 应使用“可视化”界面方式设计员工管理系统。

(3) 数据库至少应设计 3 张以上关系表，表与表之间、表中属性与属性必须设计“主-从关系”，旨在应用“3NF 表、外键、触发器”。

(4) 不要设计员工管理系统的登录功能；系统应分成 2 个用户子系统设计，即“员工端”和“系统管理员端”。

(5) 课程设计的阶段性工作内容和要求，见下面的表 1.

(6) 课程设计应该提交“纸质件、电子件”资料 2 部分共 5 个单项，即：

- 课程设计任务书(每人 1 份)

- 课设小组与分工清单(每组1 份)
- 课程设计报告(每人1 份)
- 课程设计程序(每组1 份) ，结合“课设小组分工清单”划分每人的设计工作量
- 课程设计汇报电子演讲稿(每组1 份)”

其中，纸质件的装订顺序为：课程设计报告封面、设计任务书、课程设计小组及任务分工清单、设计报告正文。

(7) 课程设计的编程实现，建议 RDBMS 选用 MySQL，员工管理系统的程序开发平台建议选用 C#+ASP.NET。

四、应收集的资料及主要参考文献：

1. Parick O’Neil, Elizabeth O’Neil. 数据库:原理编程与性能(影印版)，高等教育出版社.
2. 王珊，萨师煊. 数据库系统概论(第5 版)，高等教育出版社.
3. (美)戴特，周成兴. SQL 与关系数据库理论，机械工业出版社
4. 周定康，许婕，李云洪，马明磊. 关系数据库理论及应用. 华中科技大学出版社.
5. 课程设计在线学习平台（超星）https://mooc1-1.chaoxing.com/course/99015745.html

五、审核批准意见

同意

教研室主任（签字）_____

表 1 课程设计的阶段性工作内容和要求

设计阶段	数据库设计要求	信息管理程序设计要求
需求分析	根据任务书要求，理清并陈述欲交由信息系统管理的人、事、物等“实体、实体属性”	
概要设计	实体、属性分析； 属性关联、实体关联分析； ER 图设计	2 个子系统的功能模块构成设计。 功能模块间的依存、层次关系分析。 陈述每个模块应具备“功能”。
详细设计	关系数据表转换、表结构设计； 关系表是否达到 2NF 或3NF 的检查、再分解； 最终关系表的主码确定； 各关系表之间的“主-从关系”、外码确定； 某些属性的取值约束； 某些属性之间的关联约束；	每个功能模块的： 模块内功能的初步分析 每个功能的处理流程分析
成员分工	应分工任务有：1)数据库、关系表实现 2)触发器设计 3)子系统各功能模块设计开发	
开发设计	选择某些属性的取值约束，或者某些属性之间的关联约束，或者具有“主-从关系”的表： 设计数据库端的触发器(关联对象-触发事件-处理逻辑)	每个功能模块的： 界面设计 模块内子功能分析与开发思路陈述 每个模块内子功能的处理流程分析
编程实现	编制 SQL 语句实现数据库、关系表的创建实现； 编制实现触发器	编程开发各功能模块的界面、子功能； 按 2 个子系统模块构成，集成子系统
测试	用 SQL 语句检查创建的数据库、表；用 SQL 向关系表中插入、修改数据；同时检查编	运行 2 个子系统，测试各项功能

成员分工

<table>
<tr><td>项目名称</td><td colspan="4">员工人事信息管理系统</td></tr>
<tr><td>组长</td><td colspan="2">梁桐</td><td>指导老师</td><td>贺秦禄</td></tr>
<tr><td>组员</td><td colspan="4">张轩、刘一泽</td></tr>
<tr><td>成员</td><td colspan="4">任务内容</td></tr>
<tr><td>全体成员</td><td colspan="4">数据库关系表的设计与建立</td></tr>
<tr><td rowspan="4">梁桐</td><td>触发器与前端设计</td><td colspan="3">触发器设计
对应模块系统管理员子系统页面设计</td></tr>
<tr><td rowspan="3">功能模块设计开发</td><td>调岗管理
模块设计</td><td rowspan="3" colspan="2">模块功能定义
模块内控件的触发事件、触发功能、
处理流程描述
控件触发事件的程序脚本
模块测试</td></tr>
<tr><td>薪资管理
模块设计</td></tr>
<tr><td>数据分析与统计模块
模块设计</td></tr>
<tr><td rowspan="4">张轩</td><td>前端页面设计</td><td colspan="3">员工子系统页面设计
对应模块系统管理员子系统页面设计</td></tr>
<tr><td rowspan="3">功能模块设计开发</td><td>信息查询
模块设计</td><td rowspan="3" colspan="2">模块功能定义
模块内控件的触发事件、触发功能、
处理流程描述
控件触发事件的程序脚本
模块测试</td></tr>
<tr><td>用户管理
模块设计</td></tr>
<tr><td>员工信息管理
模块设计</td></tr>
<tr><td rowspan="4">刘一泽</td><td>前端页面设计</td><td colspan="3">对应模块系统管理员子系统页面设计
系统管理员子系统初始页面设计</td></tr>
<tr><td rowspan="3">功能模块设计开发</td><td>部门管理
模块设计</td><td rowspan="3" colspan="2">模块功能定义
模块内控件的触发事件、触发功能、
处理流程描述
控件触发事件的程序脚本
模块测试</td></tr>
<tr><td>职位管理
模块设计</td></tr>
<tr><td>系统主页面</td></tr>
</table>

目录

1. 设计目的

本次课程设计旨在通过“员工人事信息管理系统”的构建与实现，使学生能够系统化地复习和应用关系型数据库的相关理论知识，并在实际需求背景下掌握信息管理系统的完整开发流程。通过本项目，学生需要基于“实体—关联”分析方法，完成对员工、管理员、部门、岗位、薪酬等业务实体的建模，并进一步设计符合第三范式的关系数据库和数据表，掌握数据主从关系、外键约束与触发器的运用，形成规范的数据存储方案。在此基础上，学生需结合不同用户角色（员工端与管理员端）的业务场景，分析并规划系统应具备的信息管理功能，如员工信息的增删改查、部门及岗位配置、薪酬发放、数据导入导出、统计分析等，进而梳理每个功能模块的内部处理逻辑与流程。项目的实施不仅要求学生在数据库层进行建模和约束定义，还需使用可视化开发平台（如 C#+ASP.NET）进行模块化程序的编写、调试与集成，从而实现系统完整的功能展示与用户交互。通过小组协作、任务分工、功能集成等过程，学生将在实际开发中提升综合编程能力，积累项目开发经验，增强创新意识和团队协同能力，并最终达到将数据库理论转化为可运行系统的目标，为今后从事信息系统开发或软件工程工作奠定坚实基础。

2. 需求分析

2.1 系统业务需求

员工人事管理系统将用户划分为“员工”和“管理员”两类，核心功能划分为三大模块：员工管理模块、查询模块和统计模块。

员工管理模块:

管理员功能：

维护公司部门信息、岗位设置及薪酬标准。

批量导入员工信息。

为新入职员工或发生部门/岗位变动的员工发送报到通知。

员工功能：

查询、修改个人基本信息。

接收报到通知后进行“到岗确认”。

查询模块:

管理员功能：支持根据身份证号、姓名、电子邮件、所属部门等条件，对员工信息进行精确查询和模糊查询。

统计模块:

管理员功能：实现各部门员工新入职与离职情况的分析统计，以及各部门每月薪酬总额的统计与分析。

2.2 核心业务实体识别与属性定义

2.2.1 部门实体：

属性：部门ID(did)、部门名称(dname)、部门电话(dtel)

公司通常由多个部门组成，每个部门负责不同的业务职能。部门实体记录了公司组织结构的基本信息，包括部门名称和联系方式。

部门ID作为主键唯一标识每个部门，部门名称和电话则记录部门的基本信息。

2.2.2 岗位实体：

属性：岗位ID(pid)、岗位名称(pname)

每个部门下设多个工作岗位，岗位实体定义了公司内各种职位的名称和归属部门。

岗位ID作为主键唯一标识每个岗位，岗位名称则描述岗位的具体职能。

2.2.3 薪酬实体：

属性：薪酬编号(mid)、薪酬等级(mlevel)、薪酬(mpay)

不同岗位有不同的薪酬标准，薪酬实体建立了岗位与薪资的对应关系。

薪酬编号作为主键唯一标识每个薪酬标准，薪酬等级和薪酬金额则定义了具体的薪资水平。

2.2.4 员工实体：

属性：员工编号(uid)、姓名(uname)、性别(usex)、入职时间(urzsj)、电话(utel)、就职状态(ustatus)、家庭住址(uadress)、身份证号(usfzh)、邮箱(umail)

员工是公司最重要的资源，员工实体记录了所有员工的基本信息和工作状态。

员工编号作为主键唯一标识每个员工，其他属性则全面记录了员工的个人信息和工作状态。

2.2.5 调岗记录实体：

属性：员工编号(NUid)、原本岗位(Npost)、去向岗位(Nposto)、调岗时间(Naddtime)、到岗时间(Ncontime)

员工在职业生涯中可能经历岗位变动，调岗记录实体跟踪这些变化。

记录编号作为主键唯一标识每条调岗记录，其他属性则详细记录了调岗的具体信息。

2.2.6 管理员实体：

属性：管理员编号(aid)、管理员姓名(aname)

系统需要区分普通员工和管理员权限，管理员实体定义了具有管理权限的用户。

管理员编号作为主键唯一标识每个管理员，管理员姓名则记录管理员的基本信息。

2.3 用户具体需求获取

2.3.1 公司管理员需求获取

管理员用户需要以下功能：

部门信息管理：添加、删除、修改部门信息（部门名称、负责人、人数）。

岗位设置管理：添加、删除、修改岗位信息

（岗位名称、职责描述、薪酬标准）。

薪酬标准管理：维护薪酬标准信息（包含所属岗位、岗位职级、薪酬标准），支持编辑、删除和登记操作。

员工信息管理：对员工信息进行维护，包括：编辑、注销（离职）、登记（新增）、查看、以及批量导入。员工信息字段包括：工号、性别、年龄、薪酬、家庭成员、所属部门、联系电话、所属岗位、入职时间。

调岗管理：查看员工的到岗确认记录。登记、记录员工调岗信息（涉及员工、新部门、新岗位、薪资等级）。

员工信息查询：

精确查询：根据身份证号、姓名、电子邮件、所属部门等条件进行精确查找，获指定员工的详细信息。

模糊查询：根据部分关键词或条件进行灵活查询，以获取相关员工信息。

统计分析：

新入职情况分析：统计新入职员工情况（如新员工人数、所属部门分布），帮助了解员工流动。

离职情况分析：统计离职员工情况（如离职员工人数、原因、所属部门分布），帮助分析员工流失。

部门薪酬统计：按部门统计每月薪酬总额，支持分析部门薪酬支出，为财务预算提供参考。

2.3.2 公司员工需求获取

员工用户需要以下功能：

个人信息查询：使用员工编号或用户名查询个人基本信息（如姓名、部门、职位、联系方式）。

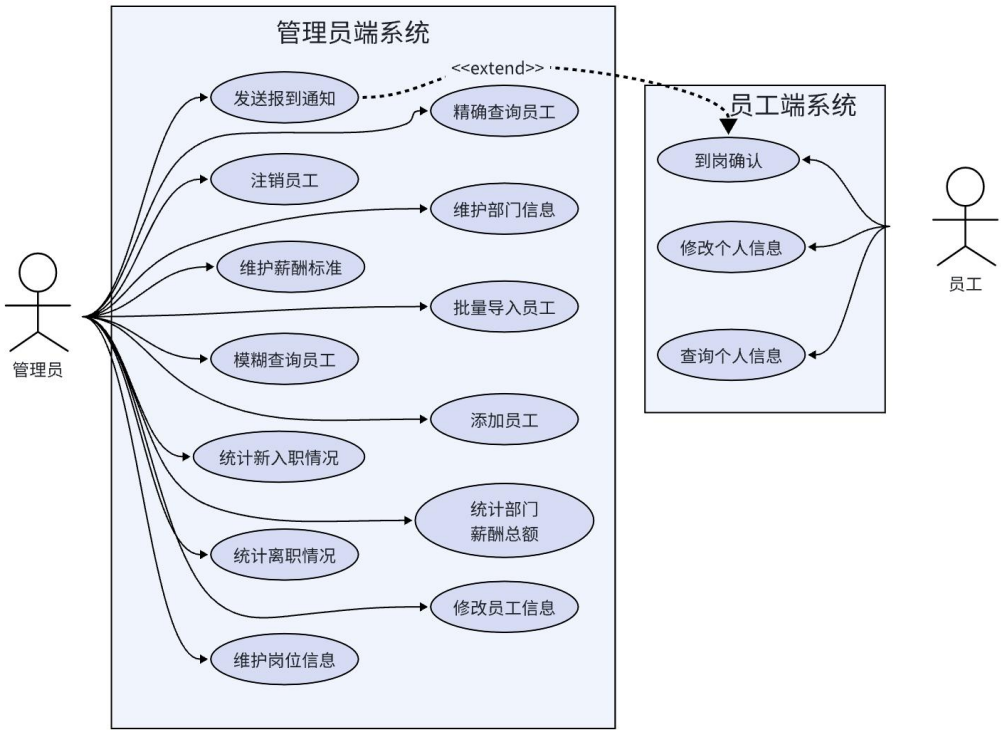


图2.4.1系统顶层用例图

个人信息修改：修改个人特定信息（如联系方式、家庭住址、紧急联系人）。

到岗确认：在每日设定的上班时间内，通过公司指定的手机应用或网页进行确认操作，表示已到达工作岗位。

2.4 系统顶层用例图

通过对系统业务需求的梳理，可明确系统应包含以下核心功能模块，并形成顶层用例图框架

核心功能总结：部门信息维护，岗位设置管理，薪酬标准管理，员工信息管理，调岗管理，查询功能，统计功能，到岗确认功能。并给出下面设计的顶层用例图。

2.4.1 员工端用例

参与者：员工（Employee）
主要用例：登录系统
查看个人信息
修改个人信息
查看部门信息
查看职位信息
查看薪资信息

查看通知公告
查询信息（如同事、部门、岗位等）
导入/导出个人数据（如有）
用例关系：“查看个人信息” 和 “修改个人信息” 是独立用例。
“查询信息” 可以包含“查询部门”、“查询职位”、“查询薪资”等子用例（包含关系）。
“查看通知公告” 可扩展为“确认已读通知”（扩展关系）。

员工登录系统后，可以查看和修改自己的个人信息，浏览所属部门和职位信息，查询自己的薪资记录，接收和查看系统发布的通知公告。员工还可以通过信息查询功能，检索相关的组织、岗位、同事等信息。部分系统可能支持员工导入或导出个人数据。

2.4.2 管理员端用例

参与者：管理员（Admin）
主要用例：登录系统
用户管理（增删改查用户/员工）
部门管理（增删改查部门）
职位管理（增删改查职位）
薪资管理（增删改查薪资）
通知公告管理（发布/编辑/删除通知）
管理员管理（如有多级管理员，增删改查管理员）
数据分析与统计
信息查询（全局查询）
导入/导出数据（如批量导入员工、导出报表）
用例关系：“用户管理”包含“添加用户”、“编辑用户”、“删除用户”、“查询用户”等子用例（包含关系）。
“部门管理”、“职位管理”、“薪资管理”等同理。
“通知公告管理”可扩展为“推送紧急通知”（扩展关系）。
“数据分析与统计”可包含“生成报表”、“导出统计结果”等子用例。
“信息查询”可包含“多条件查询”、“联合查询”等。

管理员登录系统后，拥有对用户、部门、职位、薪资、通知公告等核心数据的增删改查权限。管理员可以批量导入员工信息、发布和管理通知公告、进行全局信息查询、生成和导出各类统计报表。若系统支持多级管理员，还可对其他管理员账户进行管理。管理员的操作通常影响全局数据，权限较高。

2.5 系统数据处理分析

2.5.1. 员工操作请求的数据处理流程

员工查询个人详细信息：

员工输入个人查询条件。

系统在员工信息数据库中进行搜索。

系统返回满足条件的相关员工信息。

员工修改个人资料：

员工编辑并提交更改后的个人信息。

系统接收更新请求。

系统将修改后的信息保存到员工信息数据库中，完成信息更新。

员工查看通知并执行确认：

员工查阅接收到的通知。

员工进行确认操作。

系统将该确认信息记录到通知记录数据库中。

系统相应地将该通知的状态更新为“已确认”。

2.5.2. 管理员操作请求的数据处理流程

管理员查询员工信息：

管理员输入员工标识（如编号或姓名）进行查询。

系统依据查询条件在员工信息数据库中进行检索。

系统返回符合要求的员工相关信息。

管理员批量导入员工信息：

管理员上传包含员工数据的 Excel 文件。

系统读取 Excel 文件中的数据内容。

系统将读取到的员工信息保存到员工信息数据库中。

管理员添加/删除员工信息：

管理员执行添加或删除员工操作。

系统在员工信息数据库中对数据进行相应的增加或移除操作，更新数据库。

管理员更新部门信息：

管理员修改部门的详细信息。

系统将更新后的数据写入部门信息数据库。

管理员维护岗位信息：

管理员执行添加、删除或修改岗位信息的操作。

系统依据操作更新岗位信息数据库中的内容。

管理员查看岗位信息：

管理员浏览岗位信息。

系统从岗位信息数据库中提取相应信息并展示给管理员。

管理员查看部门信息：

管理员浏览部门信息。

系统从部门信息数据库中提取相应信息并展示给管理员。

管理员更新薪酬标准：

管理员修改特定岗位等级的薪酬信息。

系统将变更应用于薪酬信息数据库中。

管理员获取统计信息：

管理员设定统计条件（如时间段、部门）。

系统根据条件在员工信息数据库、离职记录、入职记录、薪酬信息数据库等相关数据库中提取数据。

系统对数据进行汇总分析（如计算入职/离职人数、部门薪酬总额）。

系统将统计分析结果返回给管理员。

管理员发送通知给员工：

管理员创建并发送通知给指定员工。

系统将通知内容存储到通知记录数据库中，状态设为“待确认”。

系统向员工推送该通知。

待员工确认后，系统更新通知记录中的状态（见员工“到岗确认”流程）。

2.5.3. 数据流图介绍：

(1).员工子系统

员工子系统为员工用户提供三项核心功能：个人信息查询与修改、岗位变动

信息查询以及到岗确认信息提交，并据此构建其顶层数据流图。

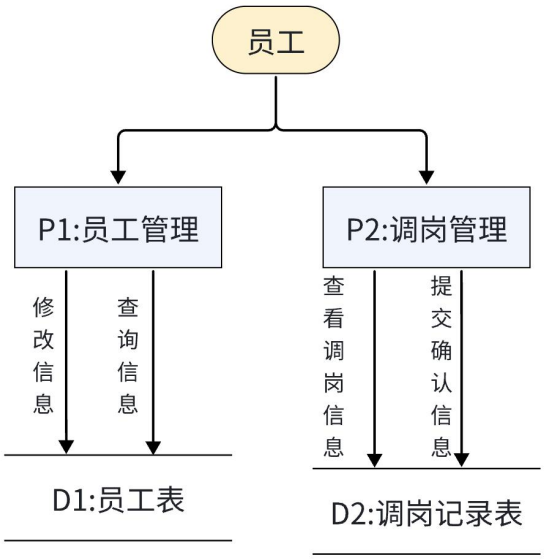


图2.5.1员工子系统顶层数据流图

(2).管理员子系统

管理员子系统为管理员提供七类核心操作：员工信息查询/批量导入/增删、部门信息维护、岗位信息维护、薪酬标准管理及统计数据分析，据此构建该子系统的顶层数据流图。

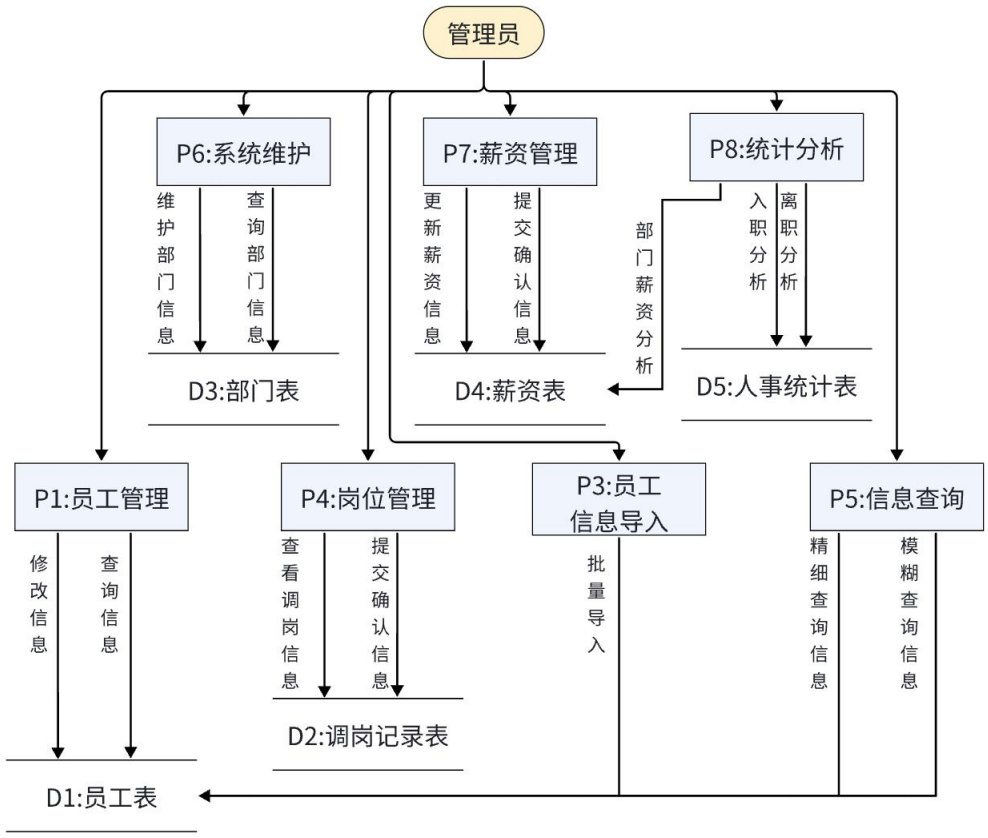


图2.5.2管理员子系统顶层数据流图

2.5.4 数据流处理描述

P1：员工管理：处理员工涉及自身信息请求

P2：调岗管理：处理涉及调岗的请求

P3：员工信息导入：处理涉及导入员工信息请求

P4：岗位管理：处理涉及岗位的请求

P5：信息查询：处理涉及查询请求

P6：系统维护：处理涉及部门的请求

P7：薪资管理：处理涉及薪酬的请求

P8：统计分析：处理涉及统计请求

对于员工子系统的数据流描述如下：

1.个人信息维护流程

查询操作

员工 → P1

输入：员工ID及查询请求

输出：当前个人信息详情

更新操作

员工 → P1

输入：员工ID、联系方式、地址等变更数据

输出：个人信息更新状态（成功/失败）

2. 调岗业务处理流程

调岗状态确认

员工 → P2

输入：员工ID及确认指令

输出：通知状态更新，返回确认操作结果

调岗记录查询

员工 → P2

输入：员工ID及调岗查询请求

输出：相关调岗通知单详情（部门/岗位/生效日期）

对于员工子系统数据流描述表如下表所示：

表2-1 员工子系统数据流描述表

序号	数据流名称	数据流位置	数据流位置
1	个人信息查询	员工→P1	员工ID+联系方式+性别+身份证号+年龄+姓名+部门+家庭成员
2	更新个人信息	员工→P1	员工ID+联系方式+性别+身份证号+年龄+姓名+部门+家庭成员
3	调岗信息确认	员工→P2	员工ID+联系方式+性别+身份证号+年龄+姓名+原先部门+原先岗位+调换岗位+调换部门
4	调岗信息查询	员工→P2	员工ID+联系方式+性别+身份证号+年龄+姓名+原先部门+原先岗位+调换岗位+调换部门

对于管理员子系统的数据流描述如下：

1.员工数据操作流程

员工信息管理

管理员 → P1

输入：身份证号/姓名 + 操作指令（增/删/改/查）

输出：操作结果状态（成功/失败）及关联数据

批量导入处理

管理员 → P4

输入：员工信息Excel文件

输出：导入执行状态 + 导入数据预览列表

员工信息查询

管理员 → P8

输入：复合查询条件组合

输出：精确匹配的查询结果数据集

2.组织架构维护流程

岗位管理

管理员 → P3

操作指令：

岗位配置（增/删/改）→ 输出岗位主数据列表

员工调岗指令 → 输出待确认状态凭证

部门维护

管理员 → P5

输入：部门操作指令（增/删/改）

输出：部门信息最新完整列表

3. 薪酬与统计流程

薪酬标准管理

管理员 → P6

输入：岗位职级 + 更新后薪酬值

输出：薪酬标准更新确认状态

数据统计分析

管理员 → P7

输入：分析维度指令（入职/离职/薪酬）

输出：结构化统计结果数据集

对于管理员子系统数据流描述表如图所示：

表2-2 管理员子系统数据流描述表

序号	数据流名称	数据流位置	数据流位置
1	员工信息管理	管理员→P1	身份证号+姓名
2	员工信息导入	管理员→P3	员工ID+联系方式+性别+身份证号+年龄+姓名+部门+家庭成员
3	调岗信息发出	管理员→P4	员工ID+联系方式+性别+身份证号+年龄+姓名+原先部门+原先岗位+调换岗位+调换部门
4	岗位管理	管理员→P4	部门+岗位+岗位薪资标准
5	精确条件查询	管理员→P5	身份证号码、姓名、电子邮件、部门
6	模糊条件查询	管理员→P5	身份证号码、姓名、电子邮件、部门（部分）
7	部门信息修改	管理员→P6	部门名称+部门ID+部门管理员
8	薪酬标准管理	管理员→P7	部门+岗位+岗位薪资标准
9	数据统计分析	管理员→P8	部门+部门月薪资+部门入职人数+部门离职人数

系统核心**数据存储**划分为五个逻辑单元：

D1员工表、D2调岗记录表、D3部门表、D4薪资表、D5人事统计表，

各存储实体的详细字段定义与数据结构如下表所示。

表2-3 数据存储表

编号	名称	输入	输出	结构	说明
D1	员工表	员工管理	员工信息	员工ID+联系方式+性别+身份证号+年龄+姓名+部门+岗位	员工ID唯一，非空数据合法约束
D2	调岗记录表	岗位管理	调岗信息	员工ID+性别+身份证号+姓名+原先部门+原先岗位+调换岗位+调换部门	员工ID唯一，非空数据合法约束
D3	部门表	系统维护	部门信息	部门名称+部门ID+部门管理员	部门ID唯一
D4	薪资表	薪资管理	薪资信息	部门+岗位ID+岗位薪资标准	岗位ID唯一，非空数据合法约束
D5	人事统计表	统计分析	分析信息	部门ID+部门月薪资+部门入职人数+部门离职人数	部门ID唯一

数据处理过程：

对于管理员子系统的数据处理过程描述如下：

员工子系统的数据处理过程，主要有2大块，P1用户管理下的个人信息查询(以下记为P1.1)、个人信息修改(以下记为P1.2)、P2调岗管理下的调岗信息查询(以下记为P2.1)、调岗信息确认，每块数据处理的具体名称、数据处理IPO(输入/处理/输出)结构如下表。

表2-4 员工子系统数据处理过程

过程编号	处理工程名称	输入	输出	处理说明
P1.1	查询信息	身份证号	员工信息	输入身份证号，点击查询，返回个人信息
P1.2	修改信息	员工要修改的个人信息	修改后的员工信息	进入信息修改界面，完成个人信息修改
P2.1	查看调岗信息	身份证号	调岗信息	在个人信息界面点击调岗信息查询，进入查询界面
P2.2	提交确认信息	确认信息	提交确认信息	在查询界面点击确认，并返回员工个人信息界面

管理员子系统的数据处理过程，主要有7大块，P1用户管理下的员工信息查询(以下记为P1),P3员工信息导入下的员工批量导入(以下记为P3),P4岗位管理下的增删改岗位(以下记为P4.1)、发出岗位信息（以下记为P4.2），P6系统维护下的部门信息修改(以下记为P6),P7薪资管理下的薪资管理(以下记为P7),P8统计分析下的对信息统计分析(以下记为P8),P5信息查询下的精细条件查询(以下记为P5.1) 模糊条件查询(以下记为P5.2)，每块数据处理的具体名称、数据处理 IPO(输入/处理/输出)结构如下表。

表2-5 管理员子系统数据处理过程

过程编号	处理工程名称	输入	输出	处理说明
P1	员工信息管理	员工ID	员工个人信息	输入身份证号，点击查询，返回个人信息
P3	员工信息导入	excel员工表格	员工信息导入	点击导入功能，选择导入excel表格
P4.1	岗位管理	修改岗位的信息	修改后的岗位信息	进入岗位信息修改界面，完成信息修改
P4.2	调岗信息发出	员工岗位变化	修改后员工个人信息	进入调岗界面，输入调岗信息，点击确认
P5.1	精确条件查询	完整员工ID	员工的员工ID+联系方式+性别+身份证号+年龄+姓名+部门+岗位	输入员工ID，身份证号、姓名、电子邮件、部门

P5.2	模糊条件查询	部分员工ID	员工的员工ID+联系方式+性别+身份证号+年龄+姓名+部门+岗位	输入员工ID，身份证号、姓名、电子邮件、部门
P6	部门信息修改	需要修改的部门信息	修改后的部门信息	进入部门修改界面，输入修改信息，点击确认
P7	薪资管理	输入部门+岗位+修改薪资	修改后的薪资标准	进入薪资修改界面，输入修改信息，点击确认
P8	对信息统计分析	无输入	部门薪资总额，入离职信息	点击分析，显示分析结果

2.5.5 细化数据流

（1）. 员工信息管理数据流图（Employee Information Management DFD）

该数据流图反映了员工和管理员对员工个人信息的查询和修改流程。员工信息管理模块与user、dept、posts等表交互，保证员工信息的准确性和实时更新。

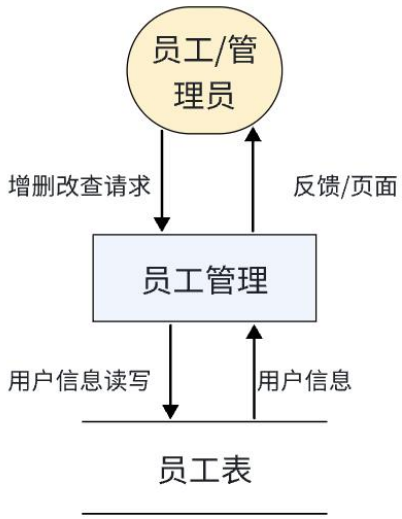


图2.5.3员工信息管理数据流图

（2）.部门管理数据流图（Department Management DFD）

介绍：该数据流图展示了管理员对部门信息的管理过程。管理员通过部门管理模块对部门信息进行增删改查操作，所有部门数据均存储在dept表中，实现了对组

织结构的有效维护和管理。

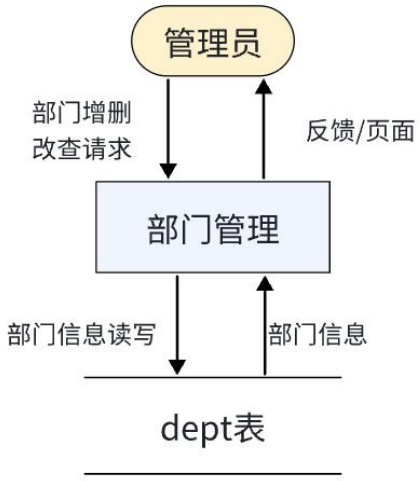


图2.5.4部门管理数据流图

（3）.职位管理数据流图（Post Management DFD）

介绍：该数据流图反映了管理员对职位信息的管理流程。管理员通过职位管理模块对职位进行增删改查，职位数据存储在posts表中，保证了职位体系的规范和岗位信息的准确。

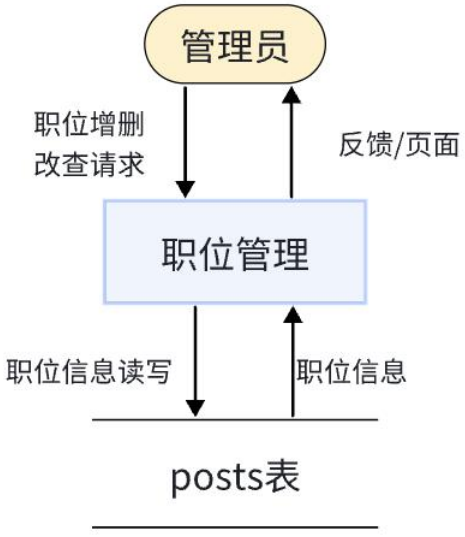


图2.5.5职位管理数据流图

（4）.管理员管理数据流图（Administrator Management DFD）

介绍：该数据流图描述了超级管理员对系统管理员账户的管理过程。通过管理员管理模块，超级管理员可以对管理员账户进行增删改查，所有管理员信息存储在administrators表中，确保系统权限的合理分配和安全管理。

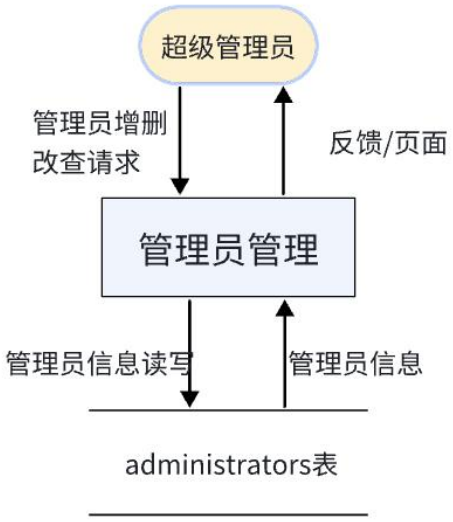


图2.5.6管理员管理数据流图

（5）.薪资管理数据流图（Salary/Payment Management DFD）

介绍：该数据流图展示了管理员或财务人员对员工薪资信息的管理流程。通过薪资管理模块，薪资数据被录入、修改或查询，所有薪资信息存储在mpays表中，实现了薪酬管理的自动化和规范化。

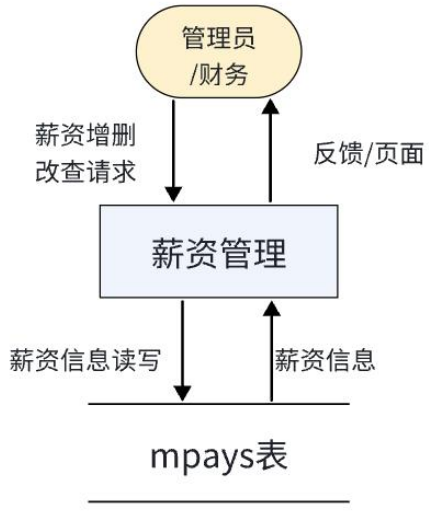


图2.5.7薪资管理数据流图

（6）.信息查询数据流图（Information Query DFD）

介绍：该数据流图展示了管理员或员工通过信息查询模块，对多张表（如user、dept、posts、mpays、dnotice）进行联合查询的过程。该模块为用户提供了灵活的信息检索和数据整合能力。

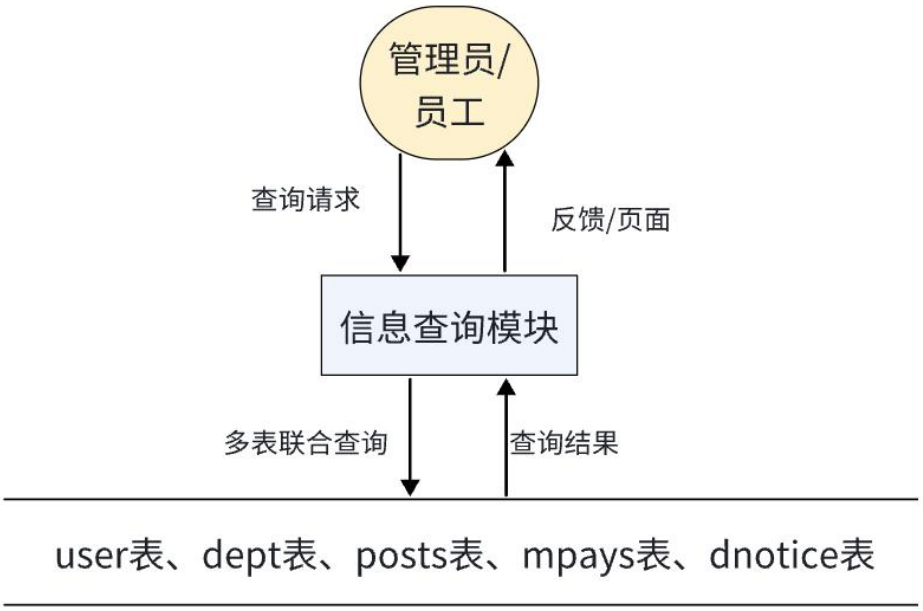


图2.5.8信息查询数据流图

（7）.数据分析与统计数据流图（Analysis & Statistics DFD）

介绍：该数据流图描述了管理员或领导通过数据分析与统计模块，对系统内各类数据（如user、mpays、dept等）进行统计分析的流程。分析结果以报表或图表形式反馈给用户，辅助管理决策和业务优化。

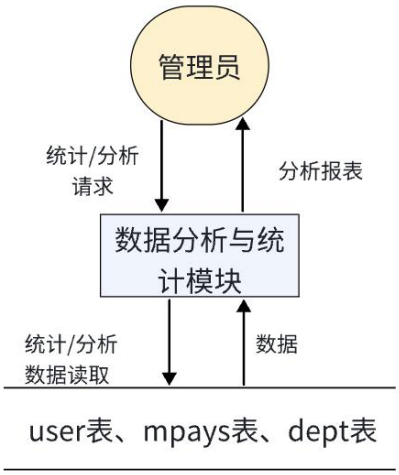


图2.5.9数据分析与统计数据流图

3. 系统设计

3.1系统开发平台选择

为满足企业管理系统的高效开发需求，本研究采用现代化技术栈构建标准化开发环境，具体配置如下：

集成开发环境 (IDE):

Visual Studio 2022: 作为核心开发工具，基于其强大的项目管理、智能编码提示（IntelliSense）及深度调试功能，显著提升开发效率。项目采用 ASP.NET Core Web 应用 (模型-视图-控制器)模板初始化，该模板内置成熟的架构规范，有助于快速搭建系统骨架，减少基础代码的重复编写。此环境为高效实现基于 ASP.NET Core 的三层架构（模型 Model、视图 View、控制器 Controller）提供了坚实的基础支撑。

数据库管理与可视化工具:

MySQL:选作关系型数据库管理系统 (RDBMS)。基于其稳定性、广泛社区支持及开发团队对该技术的熟悉度，确保了数据层开发的效率与可靠性。项目核心数据存储与结构化查询均在此平台上完成。

前端调试与设计辅助工具:

Google Chrome:作为主要的前端运行与调试环境，充分利用其内建的开

发者工具 (DevTools)。特别是 F12 功能提供的实时 HTML/CSS 审查、JavaScript 调试及网络监控能力，对于界面设计的精准把控、样式调整、性能优化及快速排错至关重要，显著加速了前端开发迭代周期。

开发语言与技术体系:

为构建高性能、可维护性强的应用系统，并契合所选框架，本项目采用以下分层技术栈：

后端: 采用 C#作为主要编程语言。C#结合.NET 平台的高性能特性、强类型安全以及成熟的生态系统（如 Entity Framework Core），是实现业务逻辑、数据处理与 API 服务的理想选择，有力保障了后端开发的效率与健壮性。关键引入的 NuGet 程序包包括：

Microsoft.AspNetCore.App (包含核心 Web 框架)

Microsoft.EntityFrameworkCore (对象关系映射 ORM，简化数据库交互)

Microsoft.NETCore.App (.NET Core 基础库)

Microsoft.AspNetCore.Mvc.Rendering (辅助视图渲染)

System.Threading.Tasks (支持异步编程)

System.Collections.Generic (常用集合类)

前端: 采用 HTML、CSS、JavaScript 三大核心 Web 技术构建用户界面(UI)与实现交互逻辑。为提升界面开发的一致性与效率，项目集成以下前端库：

Bootstrap:提供响应式布局框架与丰富的 UI 组件库，快速搭建美观、一致性强的界面。

jQuery:简化 DOM 操作、事件处理及 AJAX 通信，加速客户端脚本开发。

数据库交互:核心操作采用结构化查询语言 (SQL)编写。结合 Entity Framework Core 的 ORM 能力，在保证数据操作灵活性与可控性的同时，也通过高阶抽象减少了直接手写 SQL 的复杂性和潜在错误,优化了数据访问层的开发流程。

架构模式:

项目采用模型-视图-控制器 (Model-View-Controller, MVC) 架构模式进行设计，该模式天然支持关注点分离(Separation of Concerns)，分工明确：

模型 (Model):定义数据结构和业务规则，主要封装数据库实体（表/视图）及

项目核心业务逻辑类，部分通过 ORM 工具自动生成，部分根据特定需求手动定义。

视图 (View):基于 Razor 语法（嵌入 C#代码的 HTML 模板，.cshtml 文件）构建用户界面。视图负责数据呈现与用户交互逻辑，本项目通过分区视图等技术实现了员工界面与管理员界面的分离与视觉统一（应用公共布局文件）。

控制器 (Controller):作为核心业务调度单元，接收用户请求，协调模型处理数据，并选择渲染相应的视图返回响应。控制器在 MVC 中扮演视图与模型之间交互枢纽的角色，其清晰的路由与动作方法设计是实现高效请求处理的关键。

3.2 系统功能组成设计

3.2.1系统整体功能

本员工人事管理系统的整体功能架构主要划分为两个核心子系统，即员工模块与管理员模块。系统以企业人事信息管理需求为导向，综合考虑员工日常操作与管理员业务处理的双重需求，构建了分工明确、功能完整的业务处理框架。

一、员工模块功能

员工模块主要面向公司普通职员用户，涵盖以下功能子集：

基本信息管理：员工可登录系统后对自身的基础资料进行查看与维护，确保人事档案的准确性与及时性；

报到确认功能：新入职员工可通过系统完成入职报到操作，实现数字化到岗确认流程；

信息查询功能：员工可对个人档案信息进行快速查询，以支持其个性化的数据核查与管理。

二、管理员模块功能

管理员模块作为系统的管理中枢，面向公司人力资源管理人员和系统管理员，功能涵盖人事信息的全面处理与数据决策支持，具体包括以下七大功能类别：

员工管理：

员工信息查询：提供员工档案的精确查询与模糊查询功能，支持管理员灵活检索目标数据；

员工信息修改：管理员可对员工资料进行实时维护与更新；

调岗管理：支持管理员对员工的岗位变更进行配置与管理，保障人力资源配置的合理性；

薪资管理：对薪酬标准、薪资结构等进行集中管理，并支持历史记录追溯；

岗位管理：管理员可对企业岗位体系进行维护与优化，配置岗位职责与权限；

信息导入功能：系统支持批量导入员工信息及部门、岗位等基础数据，提高数据录入效率；

维护功能：包括部门信息、岗位设置、报到通知等静态信息的增删改查操作，确保系统数据的完整性与一致性；

查询与统计分析功能：

查询模块：包括精细查询（依据精确字段查询员工信息）与模糊查询（依据关键字进行泛化检索）；

数据分析模块：支持对各部门员工的入职趋势分析、离职数据统计以及每月部门薪酬总额的自动统计，辅助企业人事决策。

员工模块侧重于满足员工对个人信息的查看与报到需求，而管理员模块则围绕“数据采集—信息管理—分析决策”流程展开，形成了以员工信息为核心的人事管理闭环系统。各模块间协同联动，共同构成一个结构清晰、功能完备、操作高效的企业级人事管理信息系统。

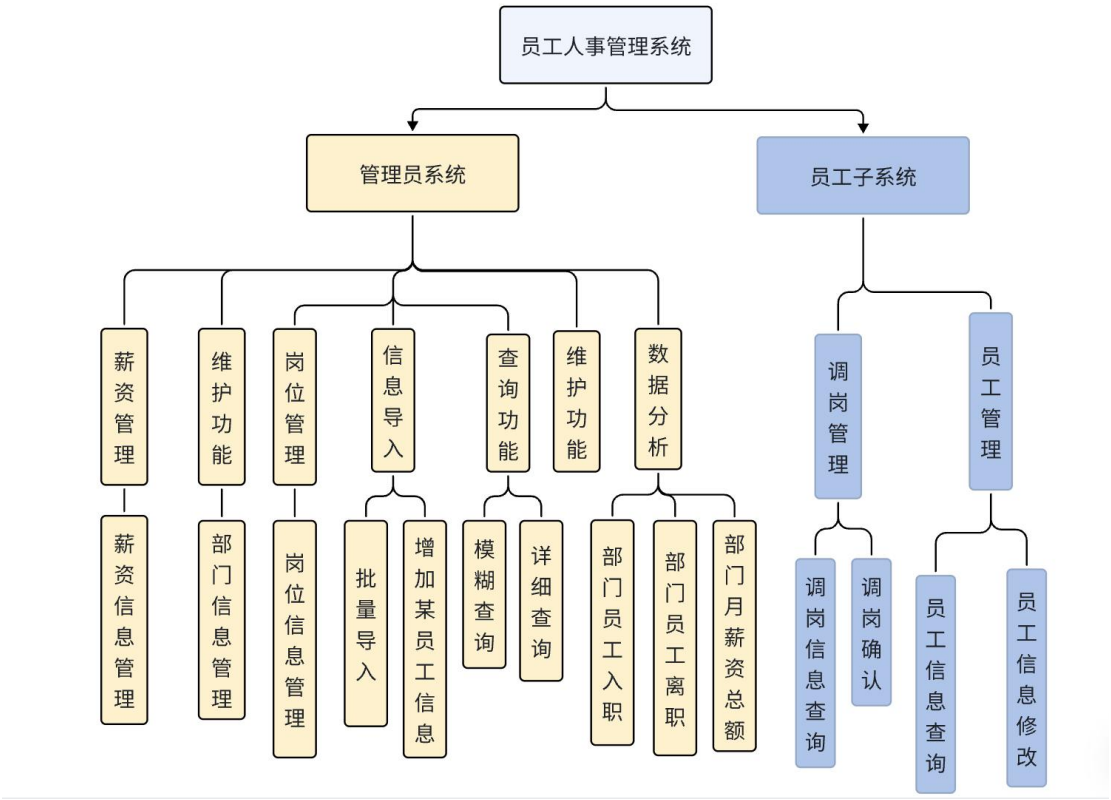


图3.2.1 系统功能模块图

3.2.2实体与属性

在本系统的业务建模过程中，部门、岗位、员工及管理员之间存在复杂的多对多关系，形成了高度灵活的人力资源组织结构。具体而言，一个部门可设置多个不同的岗位，每一岗位依据企业实际运营需求可划分为若干薪资等级，从而支持多层次、多阶段的职级管理体系。

每个岗位下可对应多个员工，且员工数量并无固定上限。此外，管理员作为系统的维护主体，其职责之一即为对员工进行管理，每位管理员可同时负责多个员工的日常信息维护、调岗审批及薪资管理等事务。

由于员工在实际工作过程中可能存在多次晋升或岗位调整，因此系统需对每一次调岗操作进行记录，以构建完整的员工职业轨迹档案。由此，一个员工实体将关联多个调岗记录实体，形成“一对多”的实体关系。

在整体 E-R（实体-关系）模型中，系统设计涵盖了关键业务实体及其属性。以图 3.2.2 为例，展示了“部门”实体及其核心属性。该实体包括三个基本属性

字段，分别为“部门ID”（用于唯一标识部门）、“部门名称”（用于反映部门职能或归属）以及“部门电话”（用于业务联络与内部通信）。

图 3.2.2 部门实体属性

图 3.2.3 表示了岗位实体以及岗位实体属性，其中岗位包括 2 个属性（岗位 ID、岗位名称）。

图 3.2.3 岗位实体属性

图 3.2.4 表示了薪酬实体以及薪酬实体属性，其中薪酬包括 3 个属性（薪酬等级、薪酬，薪酬编号）

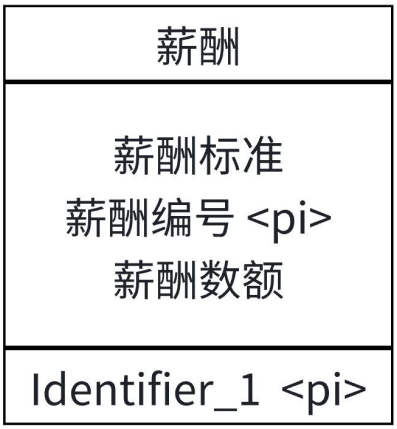


图 3.2.4 薪酬实体属性

图 3.2.5 表示了管理员实体以及其实体属性，其中管理员包括 2 个属性（管理员编号，管理员姓名）。

图 3.2.5 管理员实体属性

图 3.2.6 表示了员工实体以及其实体属性，其中员工包括 10 个属性（姓名、编号、部门、岗位、性别、就职状态、家庭住址、薪资编号、电话、入职时间）。

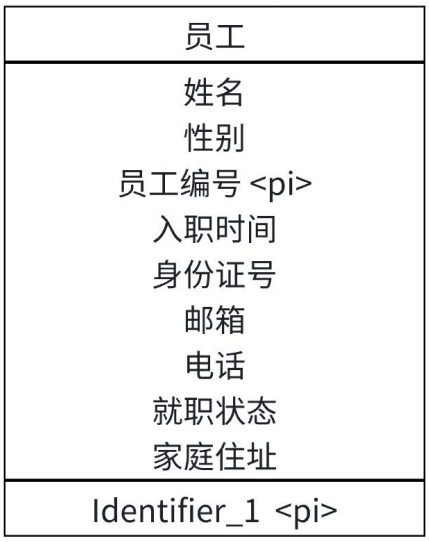


图 3.2.6 员工实体属性

图 3.2.7 表示了调岗记录实体以及其实体属性，其中调岗记录包括 6 个属性

（调岗时间、去向岗位、原本岗位、记录编号、到岗时间、员工编号）。

图 3.2.7 调岗记录实体属性

3.2.3 实体与属性关联分析

（一）管理员

在人员管理系统中，“管理员”类用户的职责范围涉及多个业务实体与操作流程，主要管理对象包括两类人员：员工与管理员自身。管理员负责的管理事务

涵盖：岗位信息维护、调岗记录管理、员工信息批量导入、员工信息查询、员工信息修改、部门信息维护及薪酬标准设置等内容。

在各项业务操作中，事务之间构成了密切的逻辑关联关系，具体包括：

管理员负责维护部门基本信息；

管理员对薪酬标准进行设定与调整；

管理员完成员工信息的批量导入操作；

管理员执行员工信息的查询功能，包括精细查询与模糊查询；

管理员处理员工调岗相关事务并调度岗位变动；

管理员维护岗位的基本设置与岗位结构；

员工可查询个人调岗信息，并通过系统进行到岗确认；

员工可对自身信息进行查询与修改操作。

通过上述管理行为，管理员在系统中承担核心的组织架构维护与人事流程调度角色，确保数据的准确性与系统功能的可用性。

（二）员工

在人员管理系统中，员工作为系统的基础用户角色，其操作权限和数据交互主要围绕自身人事信息的管理展开。涉及的对象仍包括员工与管理员两个基本实体，员工的业务操作包括：个人信息的查询与修改、调岗信息的查看及调岗结果的确认等。

事务之间的主要联系体现在以下几个方面：

员工查看管理员导入的员工基础信息；

员工可对个人信息进行自主修改操作；

员工查看管理员所录入的调岗信息内容；

员工可对调岗信息进行在线确认；

管理员对调岗记录进行添加和管理；

管理员对员工个人信息执行录入与维护操作。

上述操作流程体现了员工与管理员之间在信息流和业务逻辑上的双向交互，支持了人事信息的全面性和动态管理的可实施性。

3.2.4 ER 图设计

员工人事管理系统中涉及的对象与实体结构具有层次分明、关联紧密的特点。系统的主要参与对象包括管理员与普通员工两个角色。二者之间通过岗位、部门、调岗记录等中介实体建立起完整的业务关系模型。

系统中各主要实体间的关联关系可概括如下：

每个员工对应一个岗位实体，岗位实体从属于具体的部门实体；

每名员工可能关联多个家庭成员实体与调岗记录实体，以记录其完整的人事信息与职业变动过程；

每个岗位对应特定的薪资等级，系统将岗位与薪酬标准分离建模，独立形成“薪资等级”实体，以支持岗位层级化管理。

根据以上业务关系与数据流逻辑，设计完成了员工人事管理系统的整体 E-R 图。该图清晰地展示了系统中各业务实体之间的逻辑关系与数据依赖，为后续的数据库结构设计与系统实现提供了理论依据与建模支撑。

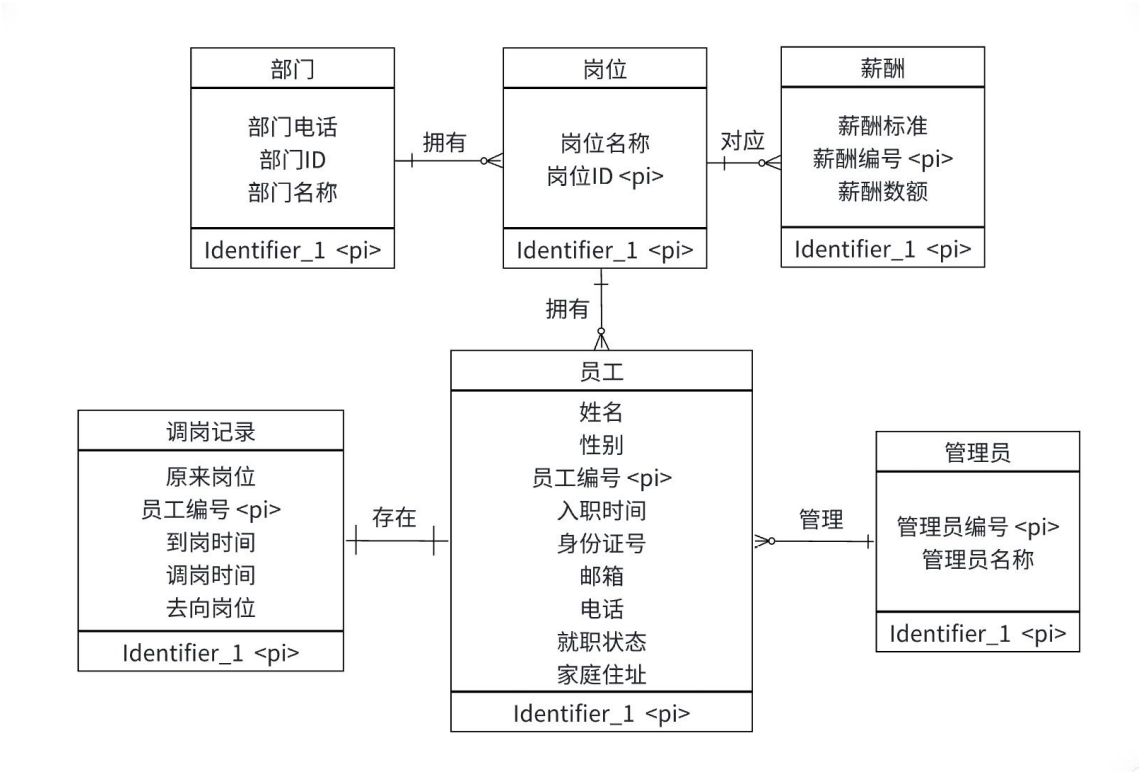


图 3.2.8 E-R图

3.3 数据库结构设计

3.3.1 概念模型转换到关系逻辑模型

① 根据实体与关系之间转换的一般规则，一个实体型可以转换为一个关系模式。结合本系统的业务模型，实体“部门、岗位、薪酬、家庭成员、员工、调岗记录、管理员”可直接转换为六个关系表。各关系表的结构说明如下：

1) 部门表

部门实体对应的关系表为 dept(did, dname, dtel)。其中 did 为主键，用于唯一标识每个部门。其余属性 dname 和 dtel 完全依赖于主键，具备函数依赖关系：dept：{did→dname, dtel}。

2) 岗位表

岗位实体转换为 posts(pid, pname)。其中 pid 为岗位编号主键，pname 表示岗位名称，依赖关系为：posts：{pid→pname}。

3) 薪酬表

薪酬实体映射为 mpays(mid, mlevel, mpay)。其中 mid 为薪酬编号主键，mlevel 表示薪酬等级，mpay 表示具体薪资。依赖关系为：mpays：{mid→mlevel, mpay}。

4) 员工表

员工实体映射为 user(uid, uname, usex, urzsj, utel, ustatus, uadress, usfzh, umail)。其中 uid 为员工编号主键，其余属性完全依赖于主键，依赖关系为：user：{uid→uname, usex, urzsj, utel, ustatus, uadress, usfzh, umail}。

5) 调岗记录表

调岗记录实体映射为 dnotice(nuid, npost, nposto, naddtime, ncontime)。其中 nuid 为调岗记录编号主键，npost 为原岗位，nposto 为调入岗位，naddtime 为调岗时间，ncontime 为到岗时间。依赖关系为：dnotice：{nuid→npost, nposto, naddtime, ncontime}。

6) 管理员表

管理员实体转换为 Administrator(aid, aname)。其中 aid 为管理员编号主键，aname 为管理员姓名。依赖关系为：Administrator：{aid→aname}。

② 部门与岗位之间的关系为一对多关系，即一个部门可以拥有多个岗位，而每个岗位只能隶属于一个部门。根据该关系结构，应将部门信息以外键形式引入岗位表中。因此岗位表扩展为：posts(pid, pname, did)。

③ 薪酬与岗位之间同样为一对多关系，一个薪酬标准可对应多个岗位，但每个岗位只能隶属于一个薪酬等级。因此可将岗位信息引入薪酬表中，形成新的结构：mpays(mid, mlevel, mpay, pid)。

④ 员工与岗位、部门及薪酬标准之间也为一对多关系。即一个部门可包含多名员工，一名员工只属于一个部门；一个岗位可有多个员工，一名员工仅任职于一个岗位；一个薪酬标准适用于多个员工，一名员工对应唯一的薪酬编号。故应将岗位 ID、部门 ID 及薪酬编号引入员工表中。更新后的员工表结构为：user(uid, uname, usex, urzsj, utel, ustatus, uadress, usfzh, umail, pid, did, mid)。

3.3.2 定义关系模型及表结构

通过上述实体与关系的映射与结构扩展，最终可得出本系统中数据库的完整表结构设计如下：

表3-1员工信息表（user）

属性名	字段类型	可空	属性约束	表级约束
Uid（员工编号）	int	否	PRIMARY KEY	
Uname（员工姓名）	varchar(50)	否	NOT NULL	
Usex（性别）	varchar(10)	否	NULL	主键：Uid 外键：Pid 外键：Did 外键：Mid
Urzsj（入职时间）	varchar(20)	否	NULL	
Utel（联系电话）	varchar(20)	否	NULL	
Ustatus（员工状态）	varchar(20)	否	NULL	
Uadress（家庭地址）	varchar(200)	是	NULL	

属性名	字段类型	可空	属性约束	表级约束
Pid（岗位ID）	int	否	FOREIGN KEY	
Did（部门ID）	int	否	FOREIGN KEY	
Mid（薪资ID）	int	否	FOREIGN KEY	
Usfzh（身份证号）	varchar(18)	否	NULL	
Umail（电子邮箱）	varchar(100)	否	NULL	

表3-2部门信息表（dept）

属性名	字段类型	可空	属性约束	表级约束
Did（部门编号）	int	否	PRIMARY KEY	
Dname（部门名称）	varchar(50)	否	NOT NULL	主键：Did
Dtel（部门电话）	varchar(20)	是	NULL	

表 3-3 岗位信息表（posts）

属性名	字段类型	可空	属性约束	表级约束
Pid（岗位编号）	int	否	PRIMARY KEY	
Pname（岗位名称）	varchar(50)	否	NOT NULL	主键：Pid 外键：Did
Did（部门ID）	int	否	FOREIGN KEY	

表 3-4　薪酬信息表（mpays）

属性名	字段类型	可空	属性约束	表级约束
Mid（薪酬编号）	int	否	PRIMARY KEY	
Mlevel（薪酬等级）	varchar(20)	否	NOT NULL	主键：Mid 外键：Pid
Mpay1（薪酬金额）	int	否	NOT NULL	
Pid（岗位ID）	int	否	FOREIGN KEY	

表3-5调岗通知信息表

属性名	字段类型	可空	属性约束	表级约束
Nuid（员工编号）	int	否	PRIMARY KEY	主键：Nuid 外键：Npost

属性名	字段类型	可空	属性约束	表级约束
Npost（原岗位ID）	int	否	FOREIGN KEY	外键：Nposto
Nposto（目标岗位ID）	int	否	FOREIGN KEY	
Naddtime（调岗时间）	varchar(20)	否	NOT NULL	
Ncontime（到岗时间）	varchar(20)	否	NOT NULL	

4. 数据库实施与数据准备

4.1 数据库的物理模型

根据管理员模块与员工模块的功能需求，系统共需建立六张基础数据表。数据库的物理模型采用全编码形式完成，即通过完整 SQL 语句实现数据库结构设计与创建，以增强对数据库语言及其操作机制的理解。

系统开发环境为 MySQL，结合 SQLYog Community 可视化客户端完成数据库连接与操作。在 SQLYog 中连接本地 MySQL 实例后，创建本系统数据库并逐一执行建表语句。以下为各个表的结构及 SQL 语句说明。

① 创建部门信息表（dept）

该表用于存储部门基本信息，供管理员进行组织结构维护与查询操作，SQL 创建语句如下：

```
CREATE TABLE `dept`
( `did` int NOT NULL,
  `dname` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `dtel` varchar(50) COLLATE utf8mb4_general_ci DEFAULT NULL,
  PRIMARY KEY (`did`)
)
ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci
```

本表用于存储部门编号、部门名称及联系电话，其中 did 作为主键。字符集统一采用 utf8mb4，支持多语言及特殊字符，存储引擎选用 InnoDB，以支持事务及外键约束操作。

② 创建岗位信息表（posts）

该表用于记录各类岗位信息，并与部门信息建立外键关联，SQL 创建语句如下：

```
CREATE TABLE `posts`
( `pid` int NOT NULL,
  `pname` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `did` int DEFAULT NULL,
  PRIMARY KEY (`pid`),
  KEY `did` (`did`),
  CONSTRAINT `posts_ibfk_1` FOREIGN KEY (`did`) REFERENCES `dept` (`did`)
)
ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci
```

该表中通过外键约束将岗位信息与部门信息关联，实现部门与岗位之间的一对多结构。

③ 创建调岗记录信息表（dnotice）

该表用于记录员工调岗过程中的历史信息，与员工表和岗位表建立多重外键关联，SQL 创建语句如下：

```
CREATE TABLE `dnotice`
(  `NUid` int NOT NULL,
  `Npost` int NOT NULL,
  `Nposto` int NOT NULL,
  `Naddtime` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `Ncontime` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  PRIMARY KEY (`NUid`),
  KEY `Npost` (`Npost`),
  KEY `Nposto` (`Nposto`),
  CONSTRAINT `dnotice_ibfk_1` FOREIGN KEY (`NUid`) REFERENCES `user` (`u
id`),
  CONSTRAINT `dnotice_ibfk_2` FOREIGN KEY (`Npost`) REFERENCES `posts` (`
```

```
pid`),
  CONSTRAINT `dnotice_ibfk_3` FOREIGN KEY (`Nposto`) REFERENCES `posts`
(`pid`)
)
ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci
```

该表定义了三个外键约束，确保调岗记录所引用的用户与岗位信息在相应表中存在，有效保障数据一致性。

④ 创建员工信息表（user）

该表是系统的核心表之一，用于存储员工的基础信息，并与岗位、部门、薪酬三张表通过外键建立联系，SQL 创建语句如下：

```
CREATE TABLE `user`
(  `uid` int NOT NULL,
  `uname` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `usex` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `urzsj` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `utel` varchar(50) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `ustatus` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `uadress` varchar(50) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `pid` int DEFAULT NULL,
  `did` int DEFAULT NULL,
  `mid` int DEFAULT NULL,
  `usfzh` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `umail` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci DE
FAULT NULL,
  PRIMARY KEY (`uid`),
  KEY `pid` (`pid`),
  KEY `did` (`did`),
  KEY `mid` (`mid`),
  CONSTRAINT `user_ibfk_1` FOREIGN KEY (`pid`) REFERENCES `posts` (`pid`),
  CONSTRAINT `user_ibfk_2` FOREIGN KEY (`did`) REFERENCES `dept` (`did`),
  CONSTRAINT `user_ibfk_3` FOREIGN KEY (`mid`) REFERENCES `mpays` (`mid
`)
```

```
)
ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci
```

该表字段覆盖员工姓名、性别、联系方式、职位、部门、薪资编号等信息。通过外键约束，保证员工数据与岗位、部门和薪酬标准的一致性。

⑤ 创建薪酬信息表（mpays）

该表用于存储薪资等级与对应岗位的薪酬信息，并与岗位表建立外键关联，SQL 创建语句如下：

```
CREATE TABLE `mpays`
(
  `mid` int NOT NULL,
  `mlevel` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `mpay` int NOT NULL,
  `pid` int DEFAULT NULL,
  PRIMARY KEY (`mid`),
  KEY `pid` (`pid`),
    CONSTRAINT `mpays_ibfk_1` FOREIGN KEY (`pid`) REFERENCES `posts` (`pid`)
)
ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci
```

mpays 表主要用于定义各薪资等级的标准金额，并与具体岗位进行对应，用于后续薪酬计算与统计分析模块。

至此，系统中全部核心功能所依赖的数据表已完成设计与创建。各表之间通过外键建立逻辑约束与业务联动关系，支撑系统整体数据一致性与完整性。但在实现过程中，仍存在部分需求逻辑未能通过结构约束直接实现，因此本系统后续通过补充触发器机制完成个别操作的自动化处理，进一步提升系统的数据可靠性。

5. 功能模块设计与开发

经过系统功能需求分析，本项目确定了以下九个核心功能模块：用户管理模

块、信息查询模块、员工信息管理模块、部门管理模块、职位管理模块、薪资管理模块、调岗管理模块、统计分析模块和系统主页面。本段将对以上功能模块进行详细的业务流程分析。

功能模块设计原则有以下几点：

1. 业务完整性原则

系统功能模块的设计严格遵循企业人事管理的完整业务流程，从员工入职、在职管理到离职调岗，涵盖人事管理的全生命周期。每个模块都对应企业人事管理中的关键业务环节，确保系统能够满足实际工作需求。

2. 用户角色分离原则

基于不同用户角色的权限和职责差异，将系统功能划分为管理员端和员工端两大类。管理员拥有完整的数据管理权限，员工只能进行有限的信息查询和个人信息维护，确保数据安全和操作规范。

3. 模块化设计原则

采用高内聚、低耦合的模块化设计思想，每个功能模块相对独立，便于开发、测试、维护和后期功能扩展。模块间通过标准化的数据接口进行交互，保证系统的稳定性和可扩展性。

4. 数据驱动原则

以企业组织架构和人事数据为核心，围绕部门、岗位、员工、薪酬等核心实体设计功能模块。通过合理的数据关联和业务逻辑，实现数据的一致性和完整性管理。

编号	模块名称	主要功能	用户	业务价值	技术实现
1	用户管理模块	员工档案的操作、Excel 批量导入、员工状态管理	管理员	实现员工信息的集中化管理	UsersController+ Excel
2	信息查询模块	多条件组合查询、模糊匹配搜索、查询结果展示	管理员	快速定位员工信息，支持复杂查询需求	InfoQueryController
3	员工信	个人信息查	员工	员工自助服务，减	EmployeeController

编号	模块名称	主要功能	用户	业务价值	技术实现
	息管理模块	询、基本资料修改、调岗信息查看		少管理员工作量	
4	部门管理模块	部门信息维护、组织架构管理、部门层级关系	管理员	建立清晰的组织架构，支持业务流程	DeptsController
5	职位管理模块	岗位信息管理、职位层级、部门关联关系	管理员	规范岗位设置，明确职责分工	PostsController
6	薪资管理模块	薪酬标准设定、等级管理、岗位薪酬关联	管理员	建立公平的薪酬体系，规范薪酬管理	MpaysController
7	调岗管理模块	调岗通知发布、员工确认、状态跟踪	管理员/员工	规范调岗流程，提高沟通效率	DnoticesController
8	数据统计分析模块	入职离职统计、薪酬分析、报表	管理员	为决策提供数据支持，优化人力资源配置	AnalysisController
9	系统主页面	系统主页展示、导航菜单管理、	全部用户	作为系统的统一入口，提供一致的用户体验和操作界面	HomeController

下文我将详细介绍我所负责的三个模块

5.1 系统主界面以及管理员初始选择功能界面

5.1.1 员工/管理员身份进入选择

标题：显示“欢迎访问——员工管理系统”。

布局：居中显示的登录框，适配不同屏幕尺寸。

功能：用户可以选择身份（员工或管理员）。

点击“进入系统”按钮后，跳转到对应的后台管理页面。

图5-1员工/管理员身份选择进入界面

5.1.2 管理员界面

设计整体布局：

顶部为标题"员工管理系统-管理员端"

中部实现管理模块的链接：

部门管理，岗位管理，薪酬标准，员工管理，调岗管理，信息查询，统计分析

图5-2管理员初始选择功能界面

特殊交互功能 ：

button:hover	鼠标悬停时按钮颜色变深（视觉反馈）

在 CSS 中，button:hover 是一个 伪类选择器（Pseudo-class），用于定义当用户将鼠标指针悬停在按钮上时的样式变化。

视觉反馈通过样式变化（如颜色、阴影、边框等）提示用户该元素可交互。

提升用户体验：让用户直观感知到按钮的“可点击性”。

5.2 部门管理模块

部门管理模块是管理员维护公司组织架构的基础工具，可对部门信息进行增删改查，管理部门与岗位、员工的关联关系。

5.2.1 添加新部门

点击 “新建部门” 按钮，进入表单页面；输入部门名称、部门电话

系统自动生成唯一部门 ID，确认后保存至数据库。

5-3新建部门界面

5.2.2 修改部门信息

在部门列表中点击 “编辑”，可修改部门名称或电话，修改后实时同步到数据库，同时影响关联的岗位和员工。

5.2.3 删除部门

点击 “删除” 前，系统自动检查是否有岗位或员工关联该部门：

若有岗位属于该部门，提示 “请先移除该部门下的所有岗位”；若有员工属于该部门，提示 “请先将员工调至其他部门”；无关联时，直接删除部门并更新列表。

岗位必须属于某个部门，若删除部门，系统会提示 “请先删除该部门下的所有岗位”。

员工表中记录部门 ID，若修改部门名称，所有该部门员工的部门信息会同步更新；若删除部门，员工调岗时需选择其他部门。

员工调岗时需选择目标部门 ，调岗记录中会记录原部门和目标部门的名称。


员工管理系统 - 管理员端
部门管理
岗位管理
薪酬标准
员工管理
调岗管理
信息查询
统计分析
删除部门
警告：您确定要删除这个部门吗？此操作不可撤销。
部门编号
1
部门名称
技术部
部门电话
010-12345678
取消
确认删除

图 5-5 删除部门界面

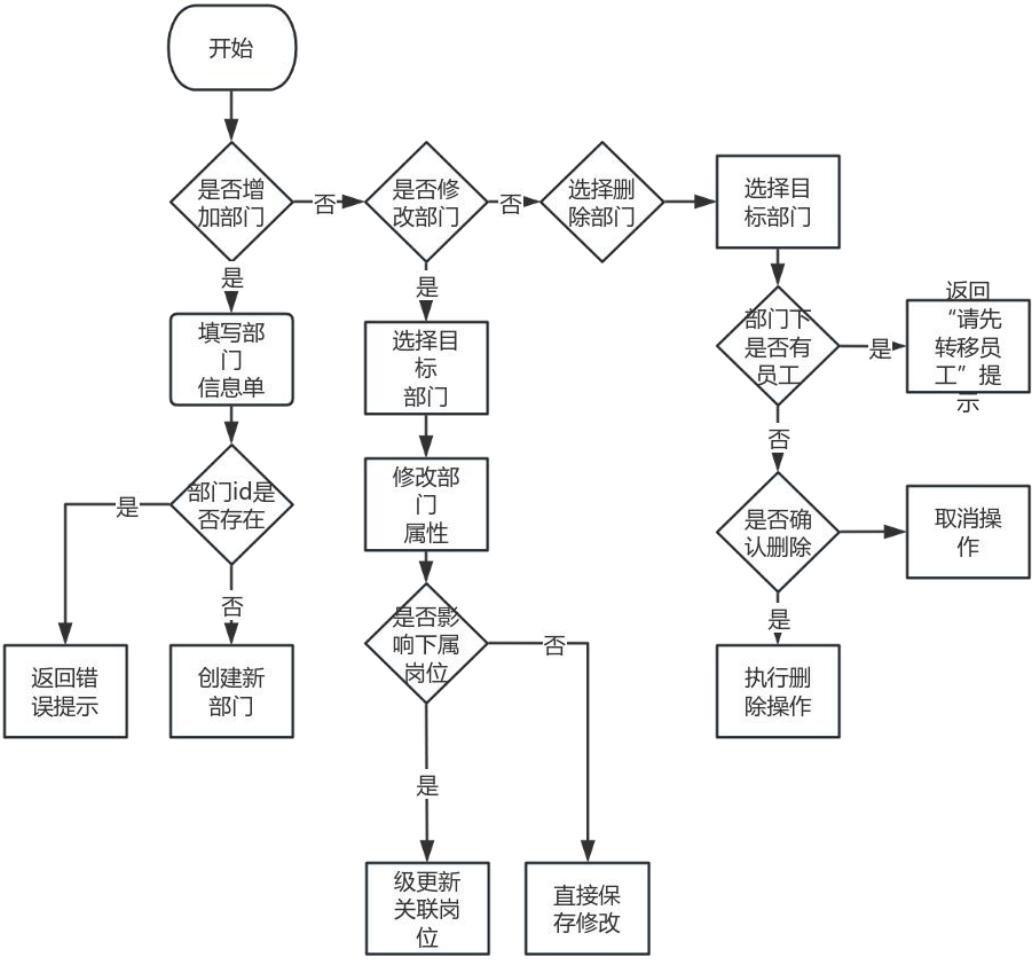

开始
是否增加部门
否
是否修改部门
否
选择删除部门
选择目标部门
是
是
是
填写部门信息单
选择目标部门
部门下是否有员工
是
返回“请先转移员工”提示
否
部门id是否存在
是
否
修改部门属性
是否确认删除
取消操作
返回错误提示
创建新部门
是否影响下属岗位
否
是
否
执行删除操作
级更新关联岗位
直接保存修改

图5-6部门管理流程图

5.3 职位管理模块

岗位管理是公司组织架构的核心子系统，管理员通过此模块实现岗位体系的动态维护，确保岗位与部门、员工的关联关系始终一致。其核心能力包括岗位基础信息管理、跨部门岗位调整、员工岗位归属维护。

5.3.1 添加新岗位

点击管理员端“新建岗位”按钮，进入岗位创建表单。

输入必填信息：

岗位名称（唯一约束，不可重复）

所属部门（从部门列表下拉选择）

系统自动生成唯一 岗位 ID，点击“确认”保存至 posts 表。

通过外键约束（did）确保所选部门存在于 dept 表

岗位名称重复时触发错误提示：”岗位名称已存在”。

图5-7新建岗位界面

5.3.2 修改岗位信息

在岗位列表点击目标岗位的“编辑”按钮。

允许修改字段：

岗位名称（需重新校验唯一性）

所属部门（切换至其他有效部门）

保存后实时同步数据库，并级联更新关联数据：

该岗位下所有员工的 user 表 pid 字段自动更新。

调岗记录表 dnotice 中原岗位名称同步变更（通过外键约束）。

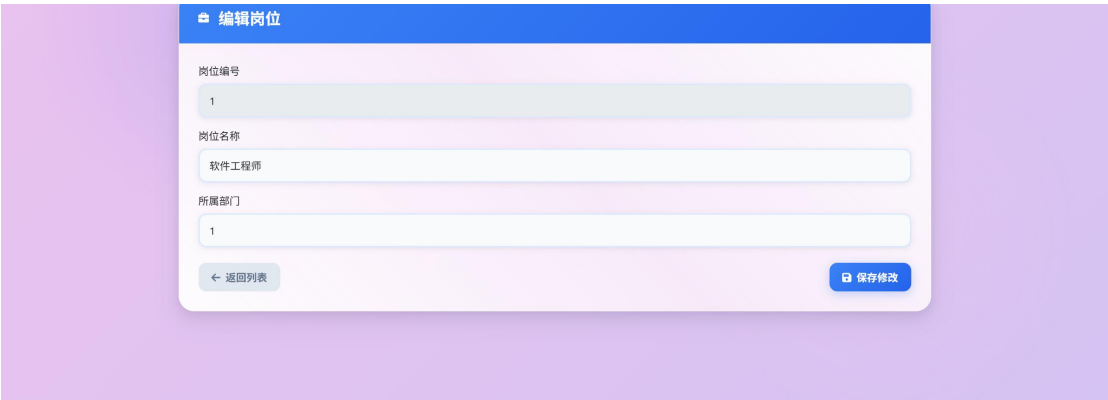


图5-8修改岗位界面

5.3.3 删除岗位

点击岗位列表中的“删除”按钮。

系统自动执行关联性检查：

存在关联员工：查询 user 表，若该岗位下仍有员工（pid=目标岗位ID），提示 "请先将该岗位员工调至其他岗位"，禁止删除。

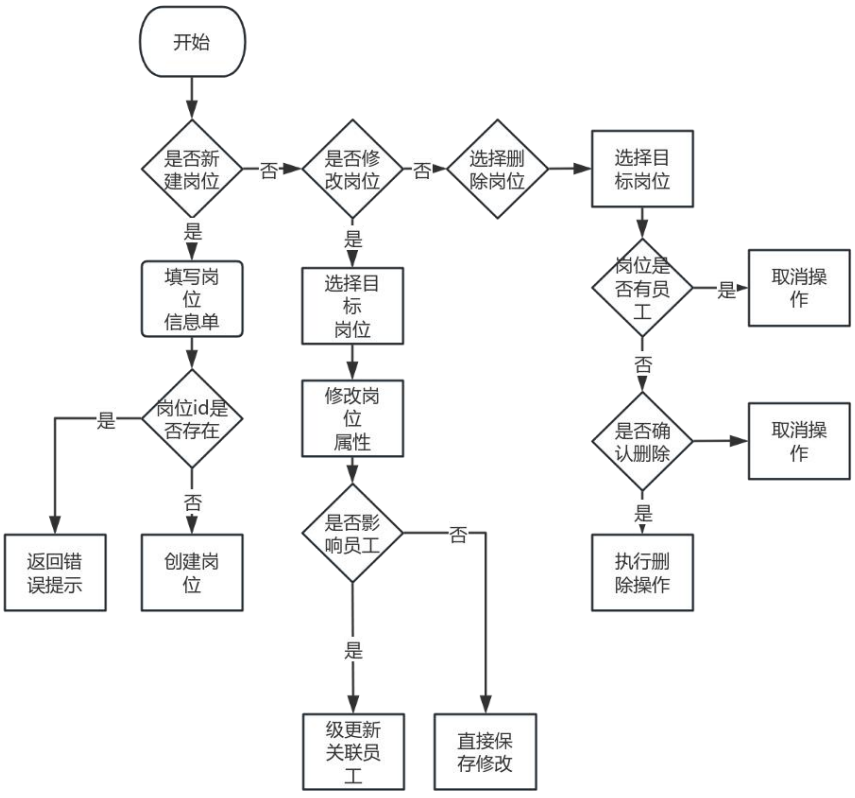
存在调岗记录：检查 dnotice 表中原岗位（Npost）或目标岗位（Nposto）关联记录，提示"请先清理相关调岗记录" 。

无关联数据时，直接删除岗位并刷新列表。

删除操作触发事务处理，确保数据一致性

岗位必须归属有效部门（posts.did → dept.did），删除部门前需先移除其下所有岗位。

图5-9删除岗位界面



5-10职位管理总流程图

5.4 功能模块开发实现

角色选择表单

```
<select class="form-control custom-select" id="roleSelect" name="role">
    <option value="/Employee/Index">员工</option>
    <option value="/Users/Index">管理员</option>
</select>
```

跳转逻辑

```
document.getElementById('roleSelectionForm').addEventListener('submit', function (event) {
    event.preventDefault(); // 阻止默认提交
    window.location.href = document.getElementById('roleSelect').value; // 跳转
});
```

初始界面优化实现

```
    button {
        background-color: #007bff;
        color: white;
        border: none;
        cursor: pointer;
        border-radius: 5px;
    }

        button:hover {
            background-color: #0056b3;
        }
```

部门的增删改实现

增加部门

```
// GET: 显示添加表单
public IActionResult Create()
{
    return View(); // 返回空表单
}

// POST: 处理添加请求
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Create([Bind("Did,Dname,Dtel")] Dept dept)
{
    if (ModelState.IsValid)
    {
        _context.Add(dept);                          // 添加到数据库上下文
        await _context.SaveChangesAsync();     // 保存到数据库
        return RedirectToAction(nameof(Index)); // 跳转到部门列表
    }
    return View(dept); // 验证失败时返回表单
}
```

部门的修改

```
// GET: 显示编辑表单
public async Task<IActionResult> Edit(int? id)
{
    if (id == null) return NotFound();
    var post = await _context.Posts.FindAsync(id); // 根据 ID 查询岗位
    if (post == null) return NotFound();
    ViewData["Did"] = new SelectList(_context.Depts, "Did", "Did", post.Did);
// 部门下拉框（当前值选中）
    return View(post);
}

// POST: 处理编辑请求
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Edit(int id, [Bind("Pid,Pname,Did")] Post
post)
{
```

```
16.         if (id != post.Pid) return NotFound();
17.         if (ModelState.IsValid)
18.         {
19.             try
20.             {
21.                 // 检查是否有员工关联该岗位
22.                 var employeesExist = await _context.Users.AnyAsync(u => u.Pid
    == id);
23.                 _context.Update(post);              // 更新数据库上下文
24.                 await _context.SaveChangesAsync(); // 保存到数据库
25.             }
26.             catch (DbUpdateConcurrencyException) // 处理并发冲突
27.             {
28.                 if (!PostExists(post.Pid)) return NotFound();
29.                 else throw;
30.             }
31.             return RedirectToAction(nameof(Index));
32.         }
33.         ViewData["Did"] = new SelectList(_context.Depts, "Did", "Did", post.Did);
34.         return View(post);
35. }
```

部门的删除

```
1.  // GET: 显示删除确认页
2.  public async Task<IActionResult> Delete(int? id)
3.  {
4.      if (id == null) return NotFound();
5.      var post = await _context.Posts
6.          .Include(p => p.DidNavigation) // 加载关联部门信息
7.          .FirstOrDefaultAsync(m => m.Pid == id);
8.      if (post == null) return NotFound();
9.      return View(post);
10. }
11.
12. // POST: 处理删除请求
13. [HttpPost, ActionName("Delete")]
14. [ValidateAntiForgeryToken]
15. public async Task<IActionResult> DeleteConfirmed(int id)
```

```
{
    var post = await _context.Posts.FindAsync(id);
    if (post == null) return NotFound();

    // 检查是否有员工关联该岗位
    var employeesExist = await _context.Users.AnyAsync(u => u.Pid == id);
    if (employeesExist)
    {
        ViewBag.ErrorMessage = "无法删除该职位，因为存在关联的员工。";
        return View("Delete", post); // 阻止删除并提示
    }

    _context.Posts.Remove(post);        // 从数据库上下文移除
    await _context.SaveChangesAsync(); // 保存到数据库
    return RedirectToAction(nameof(Index));
}
```

岗位的增删改实现

增加职位

```
// GET: 显示创建表单
public IActionResult Create()
{
    ViewData["Did"] = new SelectList(_context.Depts, "Did", "Did"); // 部门下拉框
    return View();
}

// POST: 处理创建请求
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Create([Bind("Pid,Pname,Did")] Post post)
{
    if (ModelState.IsValid) // 验证数据
    {
        _context.Add(post);                // 添加到数据库上下文
        await _context.SaveChangesAsync(); // 保存到数据库
        return RedirectToAction(nameof(Index)); // 跳转到列表页
```

```
18.         }
19.         ViewData["Did"] = new SelectList(_context.Depts, "Did", "Did", post.Did); // 重新加
    载下拉框（验证失败时）
20.         return View(post); // 返回表单并显示错误
21.     }
```

修改职位

```
1.  public async Task<IActionResult> Edit(int? id)
2.  {
3.      if (id == null) return NotFound();
4.      var post = await _context.Posts.FindAsync(id); // 根据 ID 查询岗位
5.      if (post == null) return NotFound();
6.      ViewData["Did"] = new SelectList(_context.Depts, "Did", "Did", post.Did); // 部门下
    拉框（当前值选中）
7.      return View(post);
8.  }
9.
10. // POST: 处理编辑请求
11. [HttpPost]
12. [ValidateAntiForgeryToken]
13. public async Task<IActionResult> Edit(int id, [Bind("Pid,Pname,Did")] Post post)
14. {
15.     if (id != post.Pid) return NotFound();
16.     if (ModelState.IsValid)
17.     {
18.         try
19.         {
20.             // 检查是否有员工关联该岗位
21.             var employeesExist = await _context.Users.AnyAsync(u => u.Pid == id);
22.             if (employeesExist)
23.             {
24.                 ViewBag.ErrorMessage = "无法编辑该职位，因为存在关联的员工。";
25.                 return View(post); // 阻止编辑并提示
26.             }
27.             _context.Update(post);              // 更新数据库上下文
28.             await _context.SaveChangesAsync(); // 保存到数据库
29.         }
30.         catch (DbUpdateConcurrencyException) // 处理并发冲突
```

```
            {
                if (!PostExists(post.Pid)) return NotFound();
                else throw;
            }
            return RedirectToAction(nameof(Index));
        }
        ViewData["Did"] = new SelectList(_context.Depts, "Did", "Did", post.Did);
        return View(post);
    }
```

岗位的删除

```
    // GET: 显示删除确认页
    public async Task<IActionResult> Delete(int? id)
    {
        if (id == null) return NotFound();
        var post = await _context.Posts
            .Include(p => p.DidNavigation) // 加载关联部门信息
            .FirstOrDefaultAsync(m => m.Pid == id);
        if (post == null) return NotFound();
        return View(post);
    }
    // POST: 处理删除请求
    [HttpPost, ActionName("Delete")]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> DeleteConfirmed(int id)
    {
        var post = await _context.Posts.FindAsync(id);
        if (post == null) return NotFound();

        // 检查是否有员工关联该岗位
        var employeesExist = await _context.Users.AnyAsync(u => u.Pid == id);
        if (employeesExist)
        {
            ViewBag.ErrorMessage = "无法删除该职位，因为存在关联的员工。";
            return View("Delete", post); // 阻止删除并提示
        }

        _context.Posts.Remove(post);        // 从数据库上下文移除
```

```
28.         await _context.SaveChangesAsync(); // 保存到数据库
29.         return RedirectToAction(nameof(Index));
30.   }
```

6. 课程设计技术经验总结

在项目开发过程中，我们团队遭遇了一系列技术难题，通过深入讨论与实践探索，最终成功攻克。这些问题涉及页面路由与布局、控制器业务逻辑、数据库设计等多个关键领域，其解决过程为后续开发积累了宝贵经验。

问题 1：数据库设计的迭代与重构

项目初期的数据库设计存在扩展性不足的问题。岗位表缺失部门 ID 字段，使得“某部门下所有岗位”的查询需进行复杂的多表关联操作，严重影响查询效率；针对部门表名称与简称的数据冗余问题，采用计算属性实现两者的自动同步更新，降低数据维护成本。

问题 2：跨浏览器兼容性问题

在 Chrome 里开发得好好的界面，到 Firefox 和 Edge 里就“变形”了，CSS 动画效果不一样，表单样式也变丑了，响应式布局断点直接失效。我们在网上查找资料，加浏览器前缀，用 Autoprefixer 自动处理，还在 BrowserStack 上一个个浏览器测试，总算是让界面在不同浏览器里成功运行。

问题 3：数据验证与用户体验的平衡策略

在业务规则验证方面，纯前端验证有安全风险，纯后端验证影响用户体验。我们在前端利用 HTML5 属性与 JavaScript 实现即时验证；后端进行模型验证；在服务层对关键业务逻辑进行再次验证，实现安全性与用户体验的平衡。

参考文献

[1]李为为,王玉琼,宋丽萍.基于 ASP.NET 的网站系统研究——以教务管理系统为例[J].太原师范学院学报(自然科学版),2018,17(01):41-46.

[2] 周晓俊. ASP.NET 中 Excel 数据处理的技术实现[J]. 信息与电脑(理论版),2018(06):113-115.

[3] 胡强.MySQL 数据库常见问题分析与研究[J].电脑编程技巧与维护,2019(12):91-92.

附件 5　　信控学院课程设计成绩评定标准及成绩评定表

所属学院： 信息与控制工程学院　学生姓名： 刘一泽　学号：2209060331

专业：计算机科学与技术　　班级： 计算机 2203 班　成绩评定：

项目		分值	优秀 (100≥x≥90)	良好 (90>x≥80)	中等 (80>x≥70)	及格 (70>x≥60)	不及格 (x<60)	评分
平时成绩 30%	学习态度	目标 1 10%	学习态度认真，科学作风严谨，严格保证设计时间并按任务书规定的进度开展各项工作	学习态度比较认真，科学作风良好，能按期圆满完成任务书规定的任务	学习态度尚好，遵守组织纪律，基本保证设计时间，按期完成各项工作	学习态度尚可，能遵守组织纪律，能按期完成主要任务	学习态度马虎涣散，工作不严谨，不能保证设计时间和进度	
	需求分析概要设计	目标 2 10% 目标 4 10%	需求分析详细到位、目标明确，方案设计合理，能综合多种约束条件权衡和评价方案的可行性	需求分析详细到位、目标明确，方案设计合理，能结合部分约束条件权衡方案的可行性	需求分析较全面、目标明确，方案设计基本合理。	需求分析尚可，目标基本明确，方案设计基本合理	需求分析不明确，目标模糊，方案设计不够合理	
设计成果与报告成绩 40%	详细设计系统开发	目标 1 10% 目标 2 5% 目标 3 5%	文献查阅能力强，调查调研非常合理、可信，系统详细设计思路合理，分析逻辑正确，有很强的知识运用、系统开发、集成、验证能力	文献查阅能力强，调查调研可信，系统详细设计思路合理，分析逻辑正确，具有知识运用、系统开发、集成、验证能力	具有文献查阅能力，调查调研能力，系统详细设计基本合理，分析逻辑大部分正确，具有知识运用、系统开发、验证能力	有文献查阅能力，调研基本合理，完成了系统详细设计，分析逻辑部分正确，有知识运用、系统开发能力	有文献查阅能力，调研不够清楚，系统详细设计不合理，逻辑有错误，知识运用、系统开发能力不足	
	报告撰写	目标 1 5% 目标 2 10% 目标 3 5%	内容完整，结构合理，设计思路清楚、逻辑性强，报告表达准确、流畅，设计图表符合规范要求	内容完整，结构合理，设计思路清楚、逻辑合理，报告表达比较准确，设计图表符合规范要求	结构合理，内容尚完整，设计思想陈述基本通顺，设计图表比较规范	结构与内容陈述大部分合理、完整，文字、设计图表勉强达到规范化要求	内容空泛，结构混乱，文字表达不清，设计图表达不到规范化要求	
答辩成绩 30%		目标 1 5% 目标 2 5% 目标 3 5% 目标 4 15%	答辩时，设计框架陈述完整，设计内容及开发思路清楚，模块处理逻辑清晰，研发成果完整，程序运行正常。回答问题完全正确，有自己较独特的见解	设计框架陈述完整，设计内容及开发思路比较清楚，模块处理逻辑较清晰，研发成果比较完整，程序运行基本正常。回答问题正确,有自己的见解	答辩时，设计框架陈述基本完整，设计内容及开发思路比较清楚，模块处理逻辑、研发成果基本完整，程序运行部分正常。回答问题基本正确	答辩时，设计内容及开发思路尚清楚，模块处理逻辑、研发成果部分完整，程序可部分运行。能回答主要问题,且基本正确	答辩时，设计内容及开发思路陈述不清，模块处理逻辑不明，程序研发不完整。主要问题回答不正确	
百分制小计								

指导教师签名：　　2025 年 7 月 4 日