西安建筑科技大学

课程设计（论文）

课程名称：数据库课程设计

题　　目：员工考勤管理系统

院　（系）：信息与控制工程学院

专业班级：计算机 1701

姓　　名：蒋坤

学　　号：1706020115

指导教师：孔月萍

2020 年 6 月 19 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1701 学生姓名：蒋坤 指导教师（签名）：

一、课程设计（论文）题目：员工考勤信息管理系统

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.员工考勤信息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个人基本信息，但员工工号、所属部门、岗位、职务、薪资信息不能修改。管理员用户可以添加新入职员工的基本信息，维护在岗员工的工号、所属部门、岗位、职务、薪资信息，如果有员工离职，管理员可以注销员工信息，但不得删除员工数据。

(2) 员工考勤信息管理：设计员工查询自己的出勤记录、出差信息、请假信息、加班信息等功能。设计管理员查询员工信息、录入和修改员工的出勤记录、请假信息、加班信息功能。

(3) 员工批量导入：设计管理员对员工基本信息的多条数据一次性导入功能；假设员工基本信息名录格式为 Excel 文件。

(4) 查询模块：设计管理员对员工基本信息、员工各种考勤信息的多角度综合查询功能，要有多种精确条件查询和部分模糊条件查询。

(5) 统计模块：设计管理员生成“每月分部门考勤表”对“每月各部门员工考勤分布情况”进行统计与分析的功能。

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”界面方式设计员工考勤信息管理系统。

(3) 数据库至少应设计 3 张关系表，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) 不要设计员工考勤信息管理系统的登录功能;所以系统程序应该分为成 2 个用户子系统设计，即“员工考勤管理系统用户端”和“员工考勤管理系统管理员端”。

(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)
- 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，员工考勤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

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；同时检查编制的触发器是否工作正常	运行2个子系统，测试各项功能

组员分工

	模块	负责人	备注
数据库实现	关系表创建	陈旭坤	
	月统计视图创建语句	蒋坤	
	触发器设计+实现	李凯	
UI+功能实现	项目整体结构	陈旭坤	
	员工-个人信息模块（2个）	李凯	员工的修改是有限的 eg.姓名、电话等可以，薪资、部门等不行
	员工-考勤信息模块（4个功能）	蒋坤	
	管理员-导入模块（1个功能）		
	管理员-月统计模块（4个功能）		可按部门筛选
	管理员-个人信息模块（5个功能）	李凯	
	管理员-出勤信息模块（5个功能）		
	管理员-加班信息模块（5个功能）	蒋坤	
	管理员-请假信息模块（5个功能）		
	管理员-出差信息模块（5个功能）	陈旭坤	
	管理员-部门信息模块（5个功能）		

概述

员工考勤管理系统的实现目标，主要对员工的个人信息以及员工考勤信息进行管理。

管理员可通过增、删、改、查等操作，实现对员工的信息的维护。员工可以查询自己的员工个人信息和考勤信息，并且修改自己的员工个人信息。

根据任务书所述，为了进一步理清用户需求，我们小组进行了需求分析。其中包含了针对信息管理系统类系统的数据处理分析、系统数据字典等方面的分析。

由于本系统的设计目标，是实现一个基于 web 端的信息管理系统，因此在设计中，会涉及到数据库设计、系统框架设计、系统模块设计、模块功能设计、界面设计等部分设计。

目录

一、设计目的

信息系统综合设计是软件开发系列理论课程的实践性辅助教学过程，是在学习 C 语言程序设计、面向对象程序设计、数据结构和数据库系统等课程后开设的。旨在学习基于数据库的综合信息管理系统的开发途径、方法和工具，并结合关系型数据库管理系统(Relational Database Management System, 以下简称 RDBMS)平台和 Java (或其它 Web .应用开发)平台下开展实际的数据管理操作和基于数据库的小型信息系统设计。使学生进一步理解和掌握数据库系统的应用框架、开发原理和开发技术，利用现有的数据建模工具、数据库管理系统软件、程序开发平台规范、科学地完成一个微型数据库应用系统的设计与实现。培养其信息系统的综合设计能力，训练基于 Web 的微型信息管理系统的开发技术，锻炼设计思路表达、设计文档撰写的能力;为后续课程的学习、毕业设计环节以及将来的实际工作打好坚实的基础。

信息系统综合设计要求学生在教师的指导下，利用特定的数据库设计环境，针对具体的问题，完成从需求分析、数据库概念设计、数据库逻辑设计到数据库管理功能的设计,提出能反映实际计算机工程问题的设计目标与要求、业务流程、技术方案，并经过编码、调试、改进等全部开发过程，最终形成--一个能反映应用需求的 Web 环境下数据库信息管理系统。

二、需求分析

1. 系统业务分析

（1）角色划分及业务总分析

通过对任务书进行初步分析，员工考勤管理信息系统主要是通过员工对自己的信息进行管理，以及管理员对员工的信息进行操作，从而实现对员工信息的考勤管理。

对系统进行初步分析后，从使用者的角度来说，该系统可以由员工部分和管理员部分构成。对于他们各自的业务，我们有如下的分析：

a）员工：

对于员工，主要有两个部分的业务。
第一个是用户管理，主要是员工对修改个人基本信息进行修改和查询。
第二个是考勤管理，主要是查询个人考勤信息。

b）管理员：

对于管理员，主要由四个部分的业务。
第一个是用户管理，主要是对员工的基本信息进行增删改查。
第二个是考勤管理，但是这个部分由 4 个子部分组成。

1.出勤管理，主要是对员工的出勤信息进行增删改查。
2.加班管理，主要是对员工的加班信息进行增删改查。
3.请假管理，主要是对员工的请假信息进行增删改查。
4.出差管理，对员工的出差信息进行增删改查。

第三个是导入模块：主要是可以一次性导入员工基本信息的多条数据

第四个是统计模块：主要是生成“每月分部门考勤表”。

（2）个人主要任务

1.员工考勤管理模块：员工可以查询自己的考勤统计信息。

2.管理员考勤管理模块——加班管理：对全体员工的加班信息进行增删改查。

3.管理员考勤管理模块——请假管理：对全体员工的加班信息进行增删改查。

4.导入模块：通过 excel 文件将员工信息进行导入。

5.统计模块：对于全体员工的考勤信息进行统计并展示

2. 系统数据处理分析

（1）员工子系统

提供员工个人信息的查询，及部分员工个人信息的修改。提供员工个人考勤信息查询。

考勤信息展示：展示员工考勤信息。包括出勤、出差、请假、加班四种考勤记录。

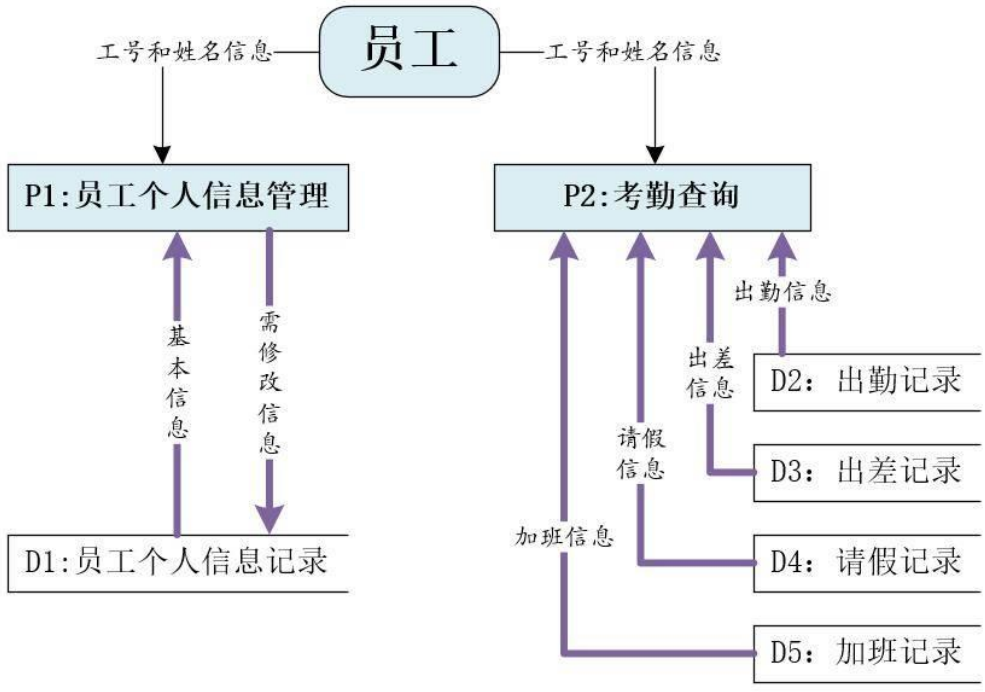


图 2-1 员工子系统数据流图

（2）管理员子系统

对员工基本信息和考勤信息的进行操作（增删改查），可以通过 excel 表录入员工信息，可以对员工考勤信息进行统计。

考勤信息操作：

对于员工的四种考勤信息进行操作，包括增加、删除、修改、查询其中查询包括按照工号模糊查询和对照姓名精确查询。

员工信息批量导入：

输入 excel 表的地址，可以将员工个人信息的多条数据一次性导入。

信息统计：

对全部员工考勤信息按月进行统计并展示。

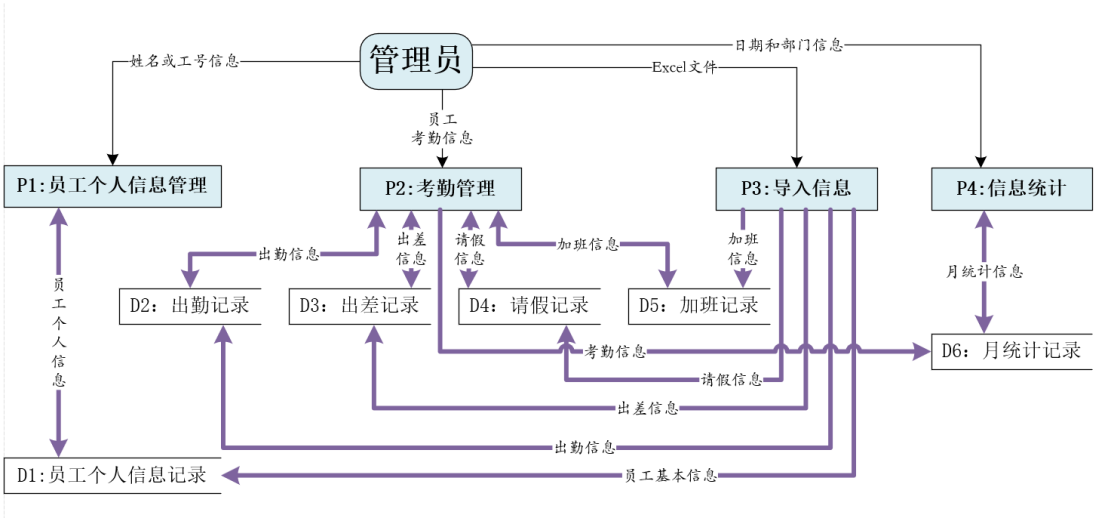


图 2-2 管理员数据流图

3. 系统数据字典

（1）描述数据流

a）员工子系统

基本信息校验：

员工 → P1，填写员工姓名和工号，检查是否匹配。

考勤信息查询：

员工 → P2，考勤信息浏览，分为出勤信息、出差信息、请假信息、加班信息组成见图 3-1。

表 3-1 员工子系统的数据流定义

序号	数据流名称	数据流位置	结构定义
1	员工校验/员工个人信息	员工 → P1	工号+姓名
2	考勤信息	员工 → P2	出勤信息+出差信息+请假信息+加班信息

b）管理员子系统

员工考勤管理：

管理员→ P1，查询员工的考勤，在考勤记录出错或变动时进行修改，信息组成见表 3-2。

员工信息导入：

管理员→ P2，可以将员工个人信息、员工各种考勤信息的多条数据一次性导入，记录信息组成见表 3-2。

统计信息：

管理员→ P3，通过条件查询对每月分部门员工出勤信息的统计，记录信息组成见表 3-2。

表 3-2 管理员子系统的数据流定义

序号	数据流名称	数据流位置	结构定义
1	员工考勤信息	管理员 → P1	工号+姓名+（出勤信息/出差信息/请假信息/加班信息）
2	导入信息	管理员→ P2	工号+姓名+性别+年龄+联系电话+所属部门+岗位+职务+薪资信息+在职情况
3	统计信息	管理员→ P3	工号+姓名+（本月出勤统计信息/本月出差天数/本月加班时长/本月请假天数）

（2）描述数据存储

综合“员工子系统”与“管理员子系统”的数据存储，主要有 6 大块：D1 基本信息记录，D2 出勤记录，D3 出差记录，D4 请假记录，D5 加班记录，D6 出勤月统计记录,D7 出差月统计记录,D8 加班月统计记录,D9 请假月统计记录,。每块数据存储的具体数据名称，数据结构如表 3-3。

表 3-3 员工考勤管理系统的数据存储定义

编号	名称	输入	输出	结构	说明
D1	员工个人	管理员增	录入，查询，	姓名，性别，电话，工号，所	工号唯一非空,员工不

	信息记录	加信息/导入信息	修改，注销处理	属部门、岗位、职务、薪资信息、在职情况	能对所属部门、岗位、职务、薪资信息、在职情况信息进行修改
D2	出勤记录	管理员增加信息	录入，查询，修改处理	编号+工号+起始时间+结束时间+出勤记录（正常、迟到、早退、旷工）	
D3	出差记录	管理员增加信息	录入，查询，修改处理	编号+工号+起始时间+结束时间+出差时长	
D4	请假记录	管理员增加信息	录入，查询，修改处理	编号+工号+起始时间+结束时间+请假时长	
D5	加班记录	管理员增加信息	录入，查询，修改处理	编号+工号+起始时间+结束时间+加班时长	
D6	出勤月统计记录	出勤信息信息统计	查询处理	工号+姓名+年份+月份+本月正常出勤天数+本月迟到天数+本月早退天数+本月旷工天数	
D7	出差月统计记录	出差信息信息统计	查询处理	工号+姓名+年份+月份+本月正常出差天数	
D8	请假月统计记录	请假信息信息统计	查询处理	工号+姓名+年份+月份+本月请假天数	
D9	加班月统计记录	加班信息信息统计	查询处理	工号+姓名+年份+月份+本月加班时长	

（3）描述数据处理过程

a）员工子系统的数据处理过程

员工子系统的数据处理过程，主要有 5 大块，P1 员工管理下的员工个人信息管理、P2 考勤管理下的出勤查询（以下记为 P2.1）、P2 考勤管理下的出差查询（以下记为 P2.2）、P2 考勤管理下的请假查询（以下记为 P2.3）、P2 考勤管理下的加班查询（以下记为 P2.4），每块数据处理的具体名称、数据处理 IPO(输入/处理/输出)结构如表 3-4。

表 3-4 员工子系统的处理过程定义

过程编号	过程处理名	输入	输出	处理说明
P1	员工个人信息	员工姓名+工号	员工个人信息	若员工姓名与工号匹配，则可以输入有权限修改的信息，将其存储于基本信息记录表中，修改后的信息可以返回给员工
P2.1	出勤查询	时间段	员工出勤信息	根据输入的时间段，可以看到自己的出勤情况

P2.2	出差查询	时间段	员工出差信息	根据输入的时间段，可以看到自己的出差情况
P2.3	请假查询	时间段	员工请假信息	根据输入的时间段，可以看到自己的请假情况
P2.4	加班查询	时间段	员工加班信息	根据输入的时间段，可以看到自己的加班情况

b）管理员子系统的数据处理过程

管理员子系统的数据处理过程，主要有 7 大块，P1 管理员管理下的员工个人信息管理，P2 考勤管理下的出勤查询（以下记为 P2.1）、P2 考勤管理下的出差查询（以下记为 P2.2）、P2 考勤管理下的请假查询（以下记为 P2.3）、P2 考勤管理下的加班查询（以下记为 P2.4），P3 导入信息，P4 信息统计，每块数据处理的具体名称、数据处理 IPO(输入/处理/输出)结构如表 3-5。

表 3-5 管理员子系统的处理过程定义

过程编号	过程处理名	输入	输出	处理说明
P1	员工个人信息管理	员工查询条件	员工个人信息	支持精确查询和模糊查询，输入要修改的信息，将其存储于基本信息记录表中，修改后的信息可以返回基本信息管理
P2.1	出勤信息更新	要更新的信息	员工出勤信息	支持精确查询和模糊查询，输入要录入或修改的出差出勤信息，将其存储于考勤信息记录表中，录入或修改后的信息可以返回考勤管理
P2.2	出差信息更新	要更新的信息	员工出差信息	支持精确查询和模糊查询，输入要录入或修改的出差信息，将其存储于考勤信息记录表中，录入或修改后的信息可以返回考勤管理
P2.3	请假信息更新	要更新的信息	员工请假信息	支持精确查询和模糊查询，输入要录入或修改的请假信息，将其存储于考勤信息记录表中，录入或修改后的信息可以返回考勤管理
P2.4	加班信息更新	要更新的信息	员工加班信息	支持精确查询和模糊查询，输入要录入或修改的加班信息，将其存储于考勤信息记录表中，录入或修改后的信息可以返回考勤管理
P3	导入信息	excel 文件	员工信息	将 excel 文件中员工基本信息和考勤信息的多条数据一次性导入到相应的记录表中
P4	信息统计	已统计信息	月统计信息	根据统计条件对各部门的每月考勤进行统计，将统计的信息返回信息统计中

三、系统设计

1）系统开发框架

（1）软件系统架构设计

图 4-1 员工考勤管理系统框架结构图

员工考勤管理系统使用三层架构进行 web 端应用系统的设计。三层架构由表现层（UI），业务逻辑层（BLL）和数据访问层（DAL）组成。
UI 层主要是在客户机上，接收用户输入的数据和显示处理后用户需要的数据。由于考勤管理系统有多个用户，且各用户所使用的终端会有不同，因此需要 UI 可在多客户机上运行。
BLL 层是 UI 层和 DAL 层之间的桥梁，用于实现业务逻辑。BLL 层在这里主要是通过服务器上的程序，在接受到 UI 层的用户指令后，连接访问 DAL 层，对数据库传递增、删、改、查的指令操作，并将指令执行后的结果返回到 UI 层。
DAL 层主要是实现对数据库中的数据的增、删、改、查，并将存储在数据库中的数据提交给 BLL 层，同时将 BLL 层处理的数据保存到数据库。

（2）服务器工作要求

OS： windows 10 (或 Windows 7)
DBMS： MYSQL
DBAS 开发平台：C#
DB 接口：ODBC

（3）客户机工作要求

OS： windows 10
浏览器： IE 10 、Edge、Firefox、Chrome 等主流浏览器

（4）数据库与应用系统的接口关系

考勤管理系统主要由 ODBC，连接数据库应用系统和 MySQL。

由于本程序是基于 ODBC 实现的，因此有如下的基于 ODBC 的各层间的连接关系图。

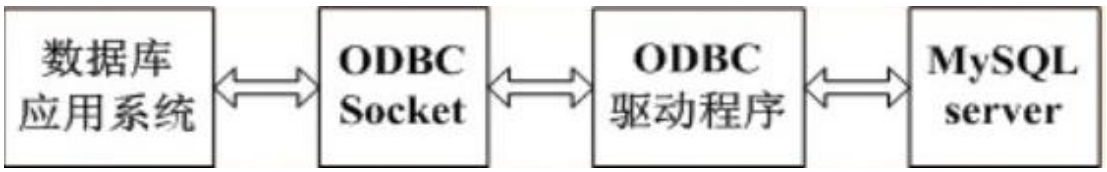


图 4-2 数据库与应用系统的接口关系图

（5）三层交互手段

数据层----业务层

EntityFramework 插件： 微软官方提供的 ORM 工具，ORM 让开发人员节省数据库访问的代码时间，将更多的时间放到业务逻辑层代码上。EF 提供变更跟踪、唯一性约束、惰性加载、查询事物等。开发人员使用 Linq 语言，对数据库操作如同操作 Object 对象一样省事。EF 有三种使用场景，1. 从数据库生成 Class，2.由实体类生成数据库表结构，3. 通过数据库可视化设计器设计数据库，同时生成实体类。

业务层----用户层

主要用 viewdata 向用户层传值，用 Ajax 技术向业务层传值

ViewData：ViewData 是一个字典集合，通过 key 值读取对应的 value 使用方法。

Ajax 技术：Ajax 即“Asynchronous Javascript And XML”（异步 JavaScript 和 XML），是指一种创建交互式、快速动态网页应用的网页开发技术，无需重新加载整个网页的情况下，能够更新部分网页的技术。

通过在后台与服务器进行少量数据交换，Ajax 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

```
function search() {
    var xmlhttp;
    var id = document.getElementById("s_id" ).value;
    var name = document.getElementById("s_name").value;

        if (window.XMLHttpRequest) {// code for IE7+, Firefox, Chrome, Opera, Safari
            xmlhttp = new XMLHttpRequest();
        }
        else {// code for IE6, IE5
            xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
        }

        xmlhttp.onreadystatechange = function () {
            if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                alert("查询成功");
                document.getElementById("table").innerHTML = xmlhttp.responseText;
                document.getElementById("mydiv").style.display = "";
                document.getElementById("div").style.display = "none"
            }
        }

        xmlhttp.open("Post", "/manager/over_search", true);
        xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");
        xmlhttp.send("&id="+id+"&name="+name);
}
```

图 4.3Ajax 样例

open 设定请求的参数，send 发送函数的参数。 onreadystatechange 是回调函数，负责做响应处理。

2. 系统功能组成设计

员工考勤管理系统主要由 2 个子系统构成，分别是员工子系统和管理员子系统。

员工子系统主要由 2 个模块组成，分别是用户管理模块和考勤管理模块。

管理员子系统主要由 4 个模块组成，分别是用户管理模块，考勤管理模块，导入模块和统计模块。其中，考勤管理模块是由 4 个子模块组成，分别是出勤子模块，加班子模块，请假子模块和出差子模块组成组成。

组织结构如下图示：

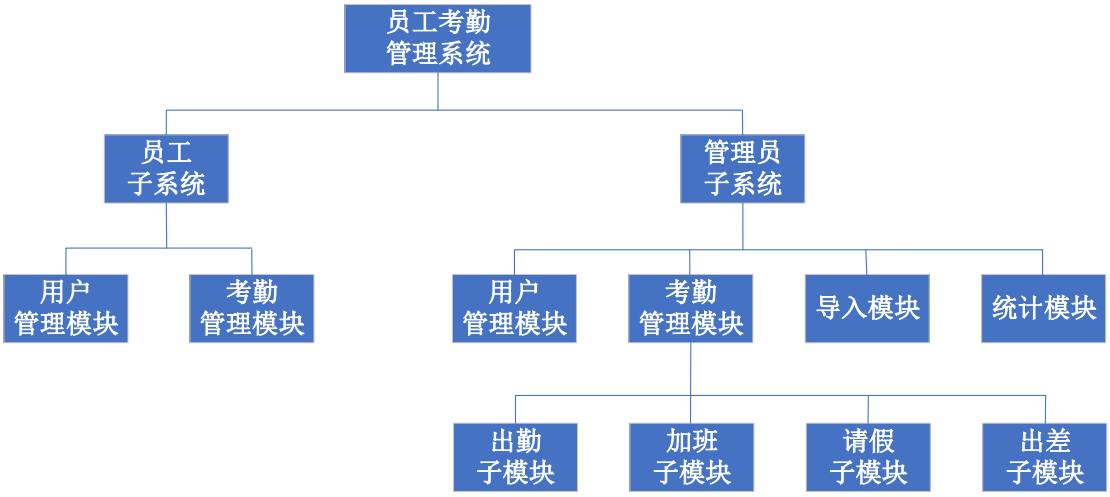


图5-1系统模块结构图

3. 数据库结构设计

（1）系统管理的对象/实体

员工考勤管理系统涉及的人有 2 类：员工和管理员，管理的事物有员工个人信息、出勤记录、出差信息、请假信息、加班信息，它们之间的主要事务联系有：

管理员和员工的个人信息都存于“员工个人信息表”里面，区别在于岗位不同。管理员是管理岗，员工是普通岗。

管理员和员工均会出勤、出差、请假和加班，从而会产生相应记录。

员工能修改自己的个人信息，能查询自己的考勤记录。

管理员能对员工个人信息、出勤记录、出差信息、请假信息、加班信息和部门信息进行增删改查

由以上分析，可得到以下概念模型的 E-R 图：

局部 E-R 图；

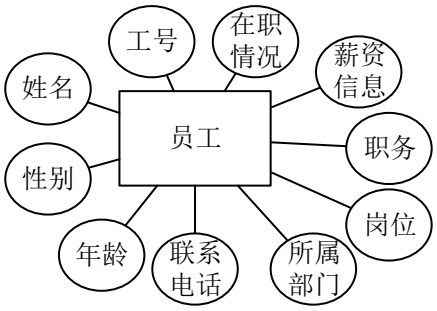


图 6-1 员工 E-R 图

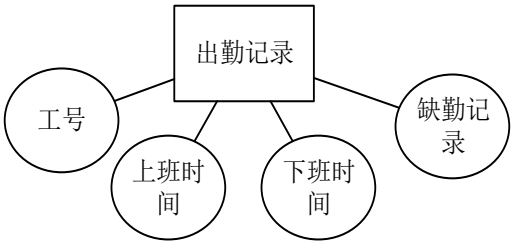


图 6-2 出勤 E-R 图

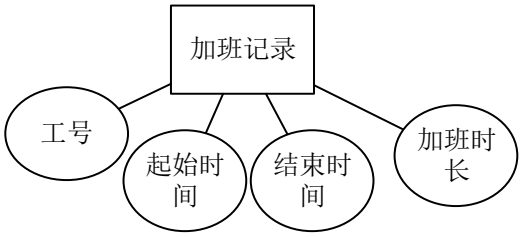


图 6-3 加班 E-R 图

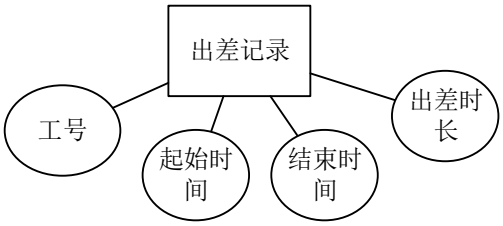


图 6-4 出差 E-R 图

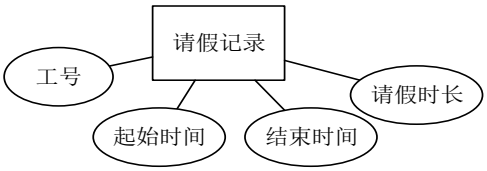


图 6-5 请假 E-R 图

图 6-6 部门 E-R 图

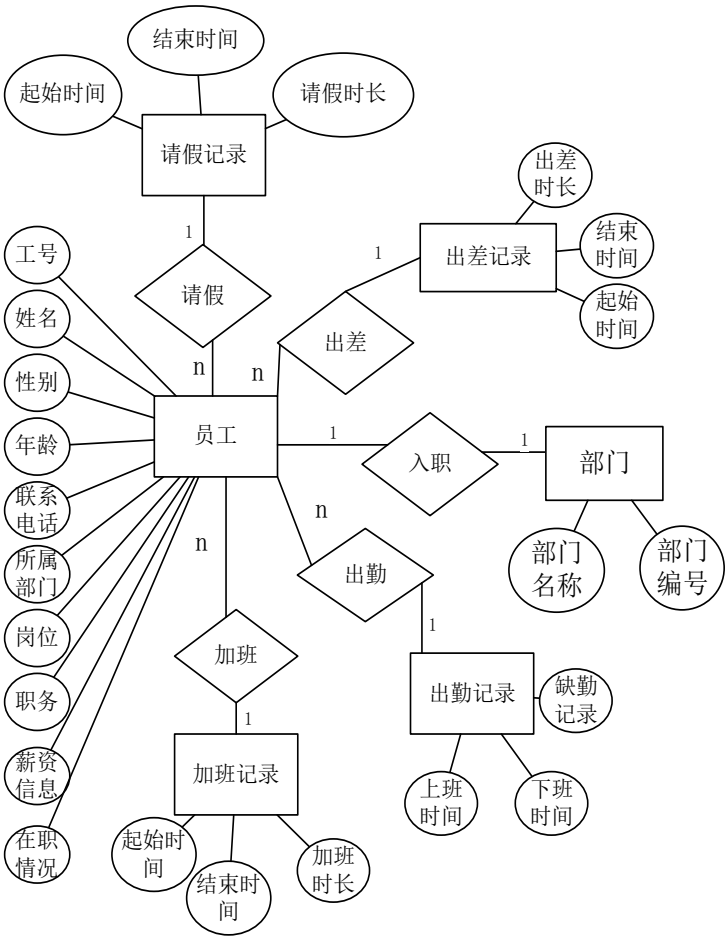


图 6-7 整体 E-R 图

（2）概念模型转换到关系逻辑模型

由以上关系模型，和 4 条转换规则，可得到关系逻辑模型：

员工个人信息表（工号,姓名,性别,年龄,联系电话,所属部门,岗位,职务,薪资信息,在职情况）
Worker(w_id,w_name,w_sex,w_age,w_telephone,w_department,w_post,w_duty,w_wage,w_on_job) PK:w_id FK：w_department

加班记录（工号，加班编号，起始时间，结束时间，加班时长）
Overtime(w_id,over_num,over_start_time,over_end_time,over_duration)
FK:w_id PK:over_num

请假记录（工号，请假编号，起始时间，结束时间，请假时长）
OffWork(w_id,off_num,off_start_time,off_end_time,off_duration)
FK:w_id PK:off_num

出差记录（工号，出差记录，起始时间，结束时间，出差时长）
Business(w_id,bus_num,bus_start_time,bus_end_time,bus_duration)
FK:w_id PK：bus_num

出勤记录（工号，出勤记录，上班时间，下班时间，缺勤记录）
Attendance(w_id,att_num,att_start_time,att_end_time,att_absence)
FK:w_id PK:att_num

部门信息（部门编号，部门名称）
department(d_id,d_name)
PK:d_id

（3）定义关系模型

根据得到的关系逻辑模型定义，整理出“员工考勤管理系统”的全部标识符、约束等信息如表 6-8

表 6-8 数据库信息

表名	属性	属性名	数据类型	长度	可空	属性约束	表级约束
员工个人信息表 Worker	工号	w_id	int				主键为 w_id
	姓名	w_name	VARCHAR	10	yes		
	性别	w_sex	VARCHAR	1	yes	男/女	外键为 w_department
	年龄	w_age	INT		yes		

	联系电话	w_telephone	VARCHAR	15	yes		
	部门编号	w_department	VARCHAR	10	yes		
	岗位	w_post	VARCHAR	3	yes	管理岗/普通岗	
	职务	w_duty	VARCHAR	10	yes		
	薪资	w_wage	INT		yes		
	在职情况	w_on_job	VARCHAR	2	yes	在职/离职	
加班记录 Overtime	加班编号	over_num	int				主键为 over_num 外键为 w_id
	工号	w_id	int				
	起始时间	over_start_time	DATETIME		no		
	结束时间	over_end_time	DATETIME		no		
	加班时长	over_duration	FLOAT		yes		
请假记录 OffWork	请假编号	off_num	int				主键为 off_num 外键为 w_id
	工号	w_id	int				
	起始时间	off_start_time	DATETIME		no		
	结束时间	off_end_time	DATETIME		no		
	请假时长	off_duration	FLOAT		yes		
出差记录 Business	出差编号	bus_num	int				主键为 bus_num 外键为 w_id
	工号	w_id	int				
	起始时间	bus_start_time	DATETIME		no		
	结束时间	bus_end_time	DATETIME		no		
	出差时长	bus_duration	FLOAT		yes		
出勤记录 Attendance	出勤编号	att_num	int				主键为 att_num 外键为 w_id
	工号	w_id	int				
	上班时间	att_start_time	DATETIME		yes		
	下班时间	att_end_time	DATETIME		yes		
	缺勤记录	att_absence	VARCHAR	2	yes	迟到/早退/旷工/正常	
部门信息 department	部门编号	d_id	VARCHAR	4	no		主键为 d_id
	部门名称	d_name	VARCHAR	10	yes		

四、数据库实施与数据准备

我在这次课设中的数据库，主要负责视图的构建和统计模块的测试数据。
以下是我们的数据库实体列表：

1. 视图

（1）视图的意义：

在做设计模块的时候，因为我们所学的业务层和数据库之间的交互手段比较单一，所以就采取了，通过视图来统计数据，然后通过 EF 控件建立连接，达到数据统计的效果。
对于四种考勤信息，主要采取按月合计的手段，将每月的请假天数、加班时长、出差天数做一合计。特别的，对于出勤的统计，是将当月每天的出勤情况进行总结，既正常次数、迟到次数、早退次数、旷工次数。

（2）视图的数量：

对于视图本来选择只建一张，但是因为 MySQL 不支持外连接，一个人在某月可能没有四种考勤信息中的某一种，所以只有一张视图是建立不起来的。

（3）创建视图的 MySQL 代码：

依次为创建出勤记录统计视图、出差记录统计视图、请假记录统计视图、加班记录统计视图

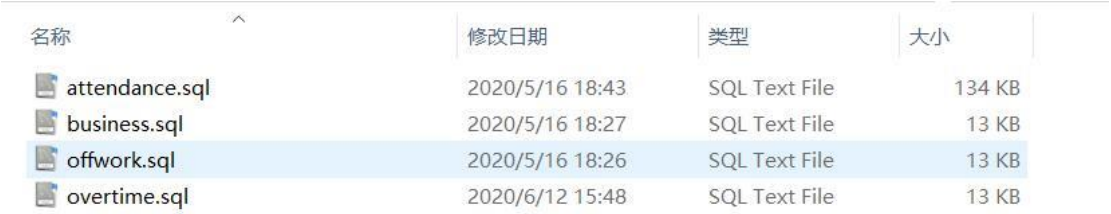
```
1.  CREATE
2.  VIEW `attendance_statistic` AS
3.      SELECT
4.          `worker`.`w_id` AS `id`,
5.          `worker`.`w_name` AS `name`,
6.          YEAR(`attendance`.`att_start_time`) AS `year`,
7.          MONTH(`attendance`.`att_start_time`) AS `month`,
8.          SUM((`attendance`.`att_absence` = '迟到')) AS `late_times`,
9.          SUM((`attendance`.`att_absence` = '早退')) AS `leave_early_times`,
10.         SUM((`attendance`.`att_absence` = '旷工')) AS `absenteeism_times`,
11.         SUM((`attendance`.`att_absence` = '正常')) AS `normal_times`
12.     FROM
13.         (`worker`
14.         JOIN `attendance` ON ((`worker`.`w_id` = `attendance`.`w_id`)))
15.     GROUP BY `worker`.`w_id` , YEAR(`attendance`.`att_start_time`) , MONTH(`
    attendance`.`att_start_time`)
16.     ORDER BY `worker`.`w_id` , YEAR(`attendance`.`att_start_time`) , MONTH(`
    attendance`.`att_start_time`)
17.
18. CREATE
19. VIEW `business_statistic` AS
20.     SELECT
21.         `worker`.`w_id` AS `id`,
22.         `worker`.`w_name` AS `name`,
23.         YEAR(`business`.`bus_start_time`) AS `year`,
24.         MONTH(`business`.`bus_start_time`) AS `month`,
```

```
        SUM(`business`.`bus_duration`) AS `total_bus_duration`
    FROM
        (`worker`
        JOIN `business` ON ((`worker`.`w_id` = `business`.`w_id`)))
    GROUP BY `worker`.`w_id` , YEAR(`business`.`bus_start_time`) , MONTH(`business`.`bus_start_time`)
    ORDER BY `worker`.`w_id` , YEAR(`business`.`bus_start_time`) , MONTH(`business`.`bus_start_time`)

CREATE
VIEW `offwork_statistic` AS
    SELECT
        `worker`.`w_id` AS `id`,
        `worker`.`w_name` AS `name`,
        YEAR(`offwork`.`off_start_time`) AS `year`,
        MONTH(`offwork`.`off_start_time`) AS `month`,
        SUM(`offwork`.`off_duration`) AS `total_off_duration`
    FROM
        (`worker`
        JOIN `offwork` ON ((`worker`.`w_id` = `offwork`.`w_id`)))
    GROUP BY `worker`.`w_id` , YEAR(`offwork`.`off_start_time`) , MONTH(`offwork`.`off_start_time`)
    ORDER BY `worker`.`w_id` , YEAR(`offwork`.`off_start_time`) , MONTH(`offwork`.`off_start_time`)

CREATE
VIEW `overtime_statistic` AS
    SELECT
        `worker`.`w_id` AS `id`,
        `worker`.`w_name` AS `name`,
        YEAR(`overtime`.`over_start_time`) AS `year`,
        MONTH(`overtime`.`over_start_time`) AS `month`,
        SUM(`overtime`.`over_duration`) AS `total_over_duration`
    FROM
        (`worker`
        JOIN `overtime` ON ((`worker`.`w_id` = `overtime`.`w_id`)))
    GROUP BY `worker`.`w_id` , YEAR(`overtime`.`over_start_time`) , MONTH(`overtime`.`over_start_time`)
    ORDER BY YEAR(`overtime`.`over_start_time`) , MONTH(`overtime`.`over_start_time`)
```

2. 统计模块测试数据

因为负责统计模块的测试数据，所以只是创建了四种考勤记录的数据。在统计时需要按照年月分组，所以要保证每种记录每个人每个月至少有一条，工作量比较大。因此，使用了 mockaroo 脚本数据生成网站，生成出勤记录 500 条，出差、请假、加班记录各 100 条。因为是随机生成的数据，所以，在某些地方可能不太合适，例如，某人某月出差 20 天，请假 20 天这样的记录，但是不影响统计功能。

以下为数据脚本列表，及部分代码展示。



名称	修改日期	类型	大小
attendance.sql	2020/5/16 18:43	SQL Text File	134 KB
business.sql	2020/5/16 18:27	SQL Text File	13 KB
offwork.sql	2020/5/16 18:26	SQL Text File	13 KB
overtime.sql	2020/6/12 15:48	SQL Text File	13 KB

（1）出勤：

```
1. insert into attendance (att_num, w_id, att_start_time, att_end_time, att_abs
   ence) values (1, 1, '2019-07-09', '2020-02-17', '旷工');
2. insert into attendance (att_num, w_id, att_start_time, att_end_time, att_abs
   ence) values (2, 1, '2019-08-13', '2019-08-03', '迟到');
3. insert into attendance (att_num, w_id, att_start_time, att_end_time, att_abs
   ence) values (3, 3, '2019-11-03', '2020-02-27', '旷工');
4. insert into attendance (att_num, w_id, att_start_time, att_end_time, att_abs
   ence) values (4, 2, '2020-01-06', '2019-07-31', '旷工');
5. insert into attendance (att_num, w_id, att_start_time, att_end_time, att_abs
   ence) values (5, 3, '2019-09-17', '2020-02-03', '正常');
```

（2）出差：

```
1. insert into business (bus_num, w_id, bus_start_time, bus_duration, bus_end_t
   ime) values (1, 1, '2020-03-08', 10, '2020-03-18');
2. insert into business (bus_num, w_id, bus_start_time, bus_duration, bus_end_t
   ime) values (2, 1, '2019-10-07', 20, '2019-10-27');
```

（3）请假：

```
1. insert into offwork (off_num, w_id, off_start_time, off_duration, off_end_ti
   me) values (1, 1, '2019-11-09', 13, '2019-11-22');
2. insert into offwork (off_num, w_id, off_start_time, off_duration, off_end_ti
   me) values (2, 3, '2020-03-16', 3, '2020-03-19');
```

（4）加班：

```
1. insert into overtime (over_num, w_id, over_start_time, over_duration, over_e
   nd_time) values (1, 1, '2019-06-11', 2, '2019-06-13');
2. insert into overtime (over_num, w_id, over_start_time, over_duration, over_e
   nd_time) values (2, 2, '2020-03-15', 12, '2020-03-27');
```

五、功能模块设计与开发

1. 员工子系统

（1）考勤管理模块：

输入自己的工号与姓名，可以选择展示出勤信息、加班信息、出差信息、请假信息中任何一种信息。如果姓名与工号不匹配则提示错误信息，并返回。个人考勤模块初始界面如下：

图 5.1 个人考勤管理界面

个人考勤管理按钮处理业务流程，可见下图。

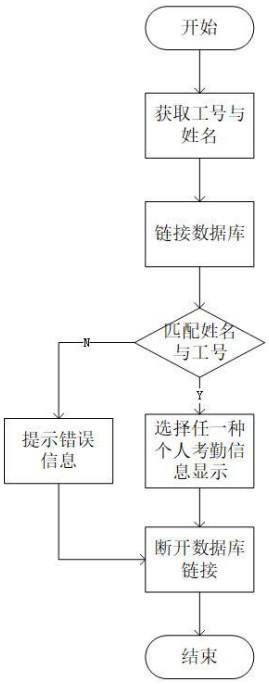


图 5.2 个人考勤管理按钮处理流程图（四合一）

2. 管理员子系统

（1）管理员考勤管理模块

管理员选择管理员信息管理模块后，可以选择任一考勤信息进行展示。系统将所有员工的个人考勤信息返回界面，管理员可以增加、修改、查询或删除考勤信息，其中查询包括精确查询（工号）和模糊查询（姓名）。

我负责加班信息管理和请假信息管理，两者的初始界面大体相同。

加班初始界面：

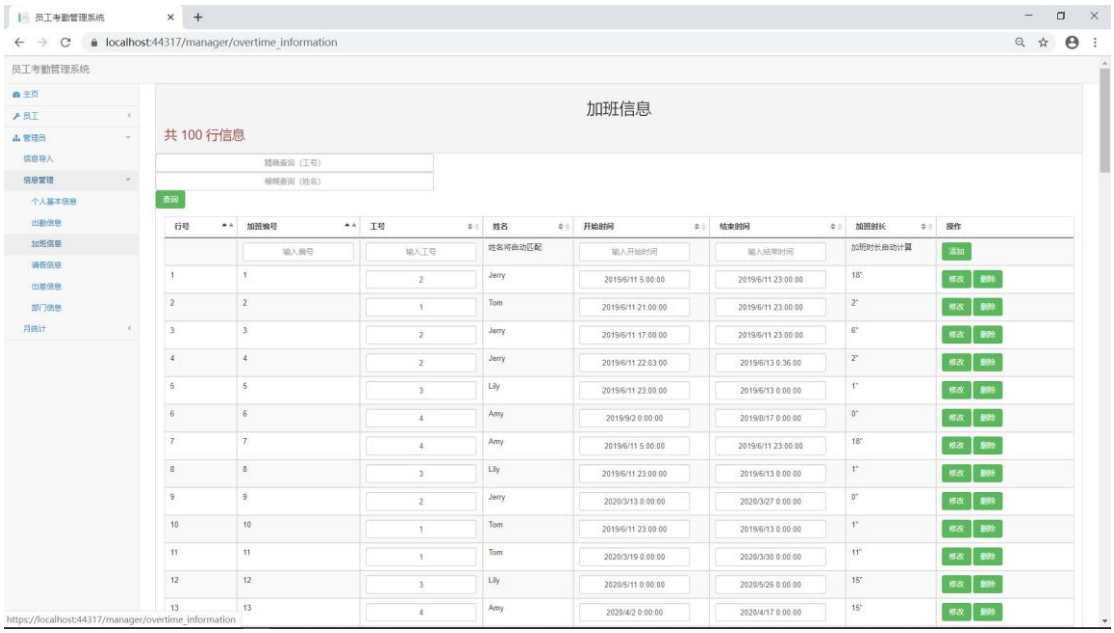


图 5.3 加班初始界面

请假初始界面：

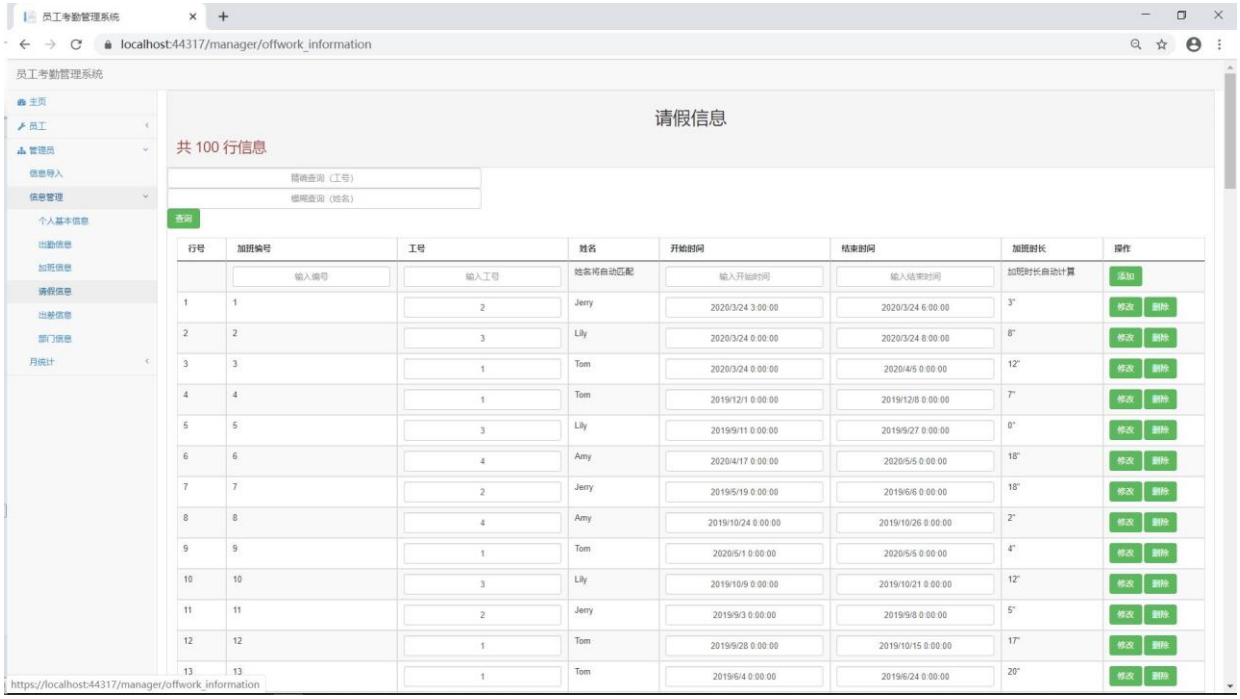


图 5.4 请假初始界面

增加按钮处理流程图：

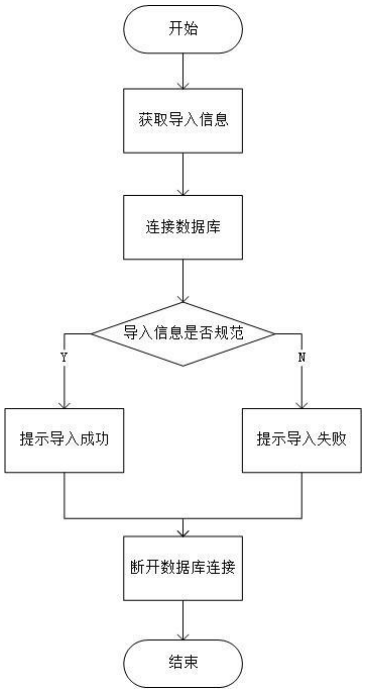


图 5.5

信息修改按钮处理流程图：

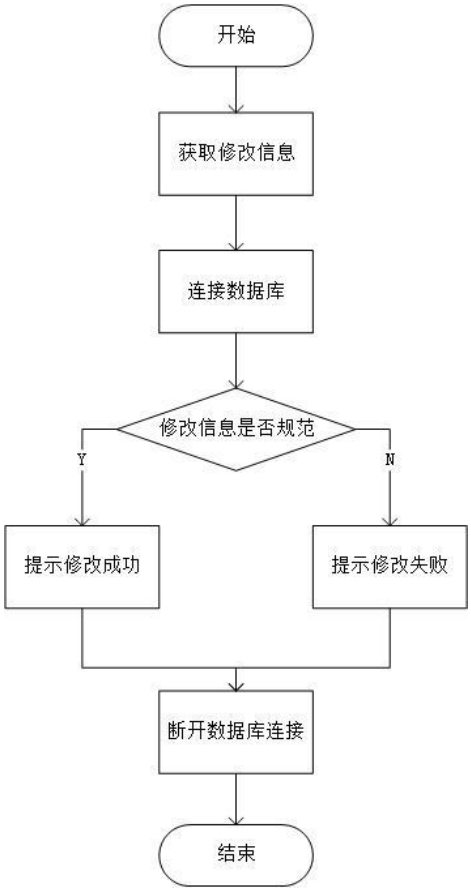


图 5.6

信息删除按钮处理流程图：

图 5.7

查询按钮处理流程图：

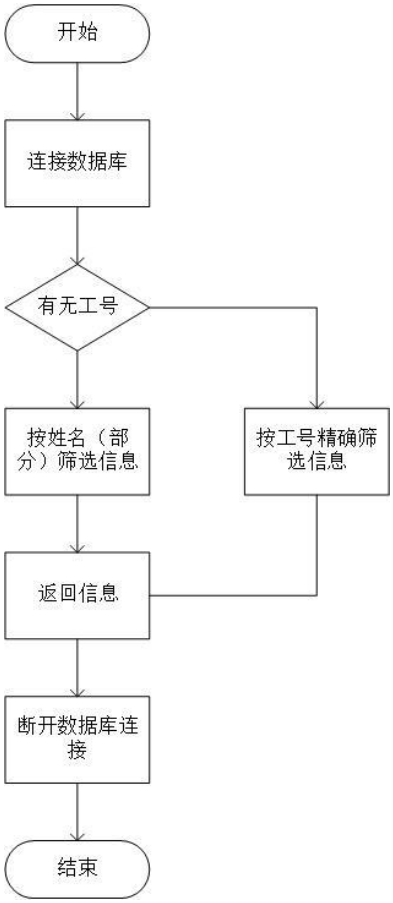


图 5.8

（2）导入模块：

管理员选择管理员导入模块后，提供 excel 文件所在位置批量导入，录入有误则提示后返回。

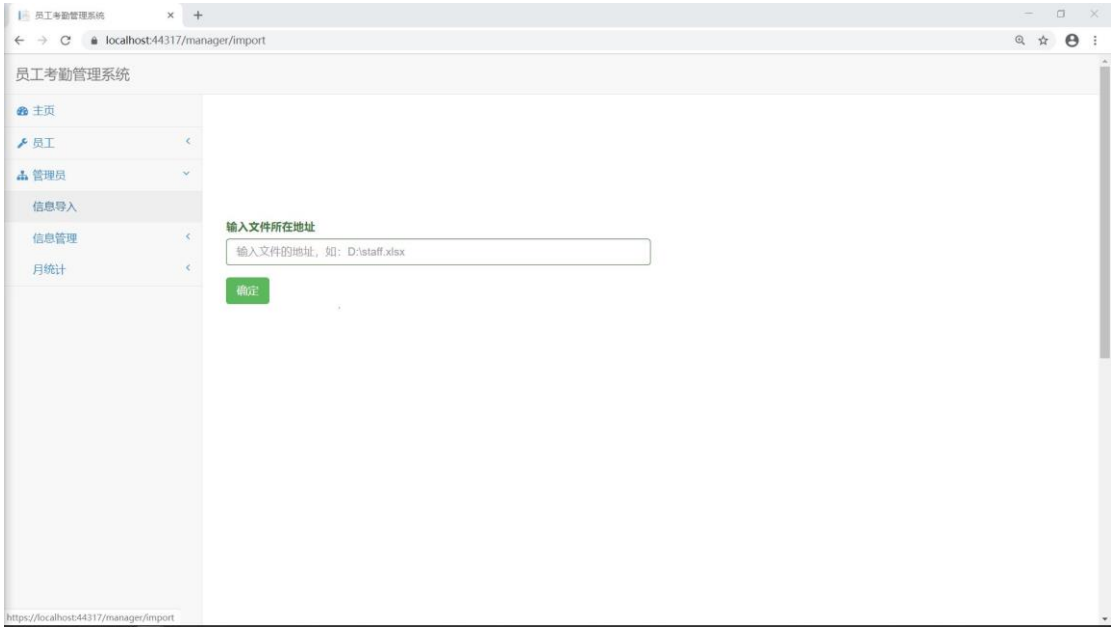


图 5.9 信息导入

确认（excel 批量导入）按钮处理流程图：

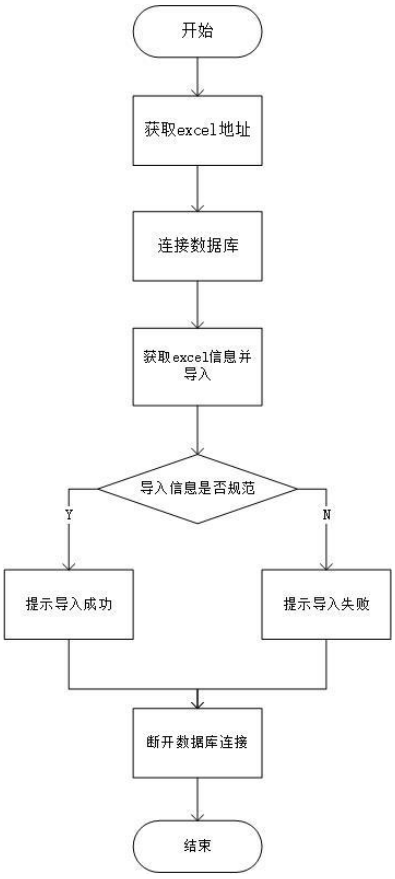


图 5.10

（3）统计模块：

管理员选择统计模块后，可以选择展示出勤信息、加班信息、出差信息、请假信息中任何一种统计信息，系统显示所有人的月总结考勤信息，管理员还可以选择某个部门来筛选信息。

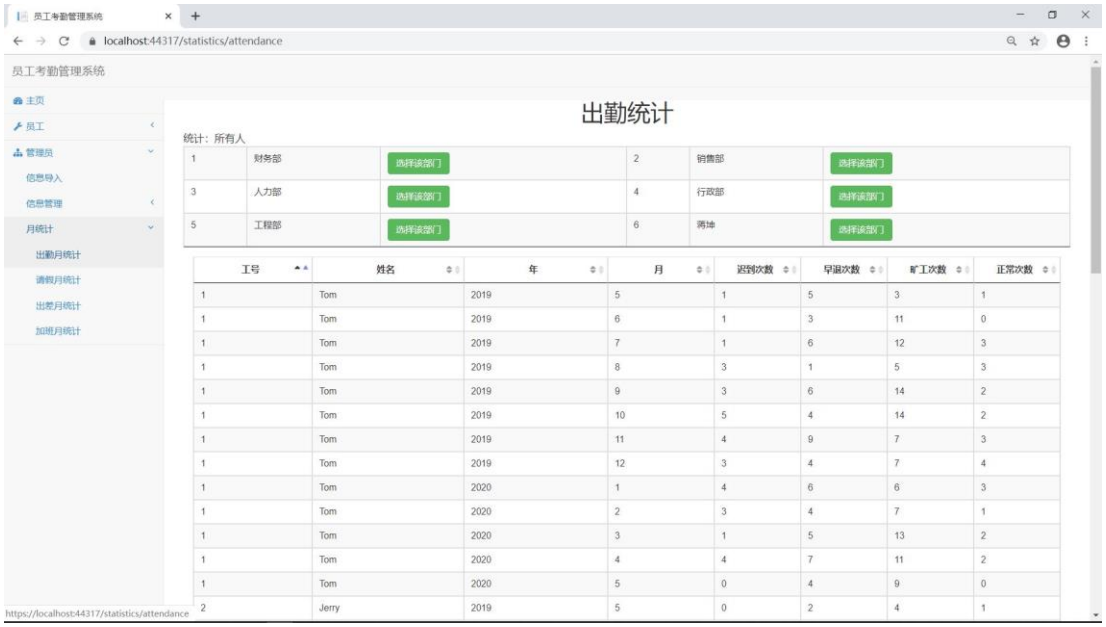


图 5.11 统计模块（出勤信息统计为例）

部门选择按钮处理流程图：

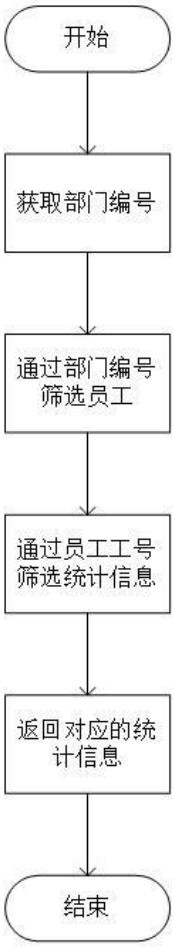


图 5.12

六、系统测试与分析

1. 员工考勤管理模块

（1）匹配失败：

正确的匹配是工号为 1，姓名为 Tom

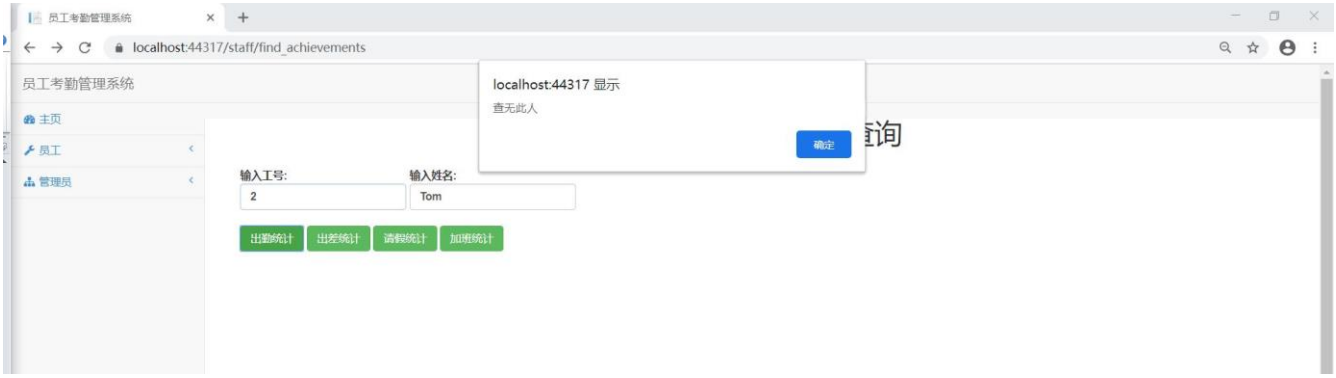


图 6.1 姓名工号不匹配

（2）出勤：

工号	姓名	年	月	迟到次数	早退次数	旷工次数	正常次数
1	Tom	2019	5	1	5	3	1
1	Tom	2019	6	1	3	11	0
1	Tom	2019	7	1	6	12	3
1	Tom	2019	8	3	1	5	3
1	Tom	2019	9	3	6	14	2
1	Tom	2019	10	5	4	14	2
1	Tom	2019	11	4	9	7	3
1	Tom	2019	12	3	4	7	4
1	Tom	2020	1	4	6	6	3
1	Tom	2020	2	3	4	7	1
1	Tom	2020	3	1	5	13	2
1	Tom	2020	4	4	7	11	2
1	Tom	2020	5	0	4	9	0

图 6.2 显示该员工的出勤信息

（3）出差：

工号	姓名	年	月	出差统计
1	Tom	2019	6	24
1	Tom	2019	7	27
1	Tom	2019	8	7
1	Tom	2019	9	33
1	Tom	2019	10	90
1	Tom	2019	11	47
1	Tom	2019	12	21
1	Tom	2020	1	1
1	Tom	2020	3	29
1	Tom	2020	4	11

图 6.3 显示该员工的出差信息

（4）请假：

工号	姓名	年	月	请假统计
1	Tom	2019	5	19
1	Tom	2019	6	30
1	Tom	2019	7	12
1	Tom	2019	8	13
1	Tom	2019	9	53
1	Tom	2019	10	15
1	Tom	2019	11	58
1	Tom	2019	12	52
1	Tom	2020	1	18
1	Tom	2020	2	15
1	Tom	2020	3	18
1	Tom	2020	4	17
1	Tom	2020	5	4

图 6.4 显示该员工的请假信息

（5）加班：

工号	姓名	年	月	加班总计
1	Tom	2019	6	22
1	Tom	2019	7	27
1	Tom	2019	8	38
1	Tom	2019	9	18
1	Tom	2019	10	27
1	Tom	2019	12	61
1	Tom	2020	1	47
1	Tom	2020	2	7
1	Tom	2020	3	40
1	Tom	2020	4	15
1	Tom	2020	5	33

图 6.5 显示该员工的加班信息

2. 管理员考勤管理模块

这个模块我只负责加班信息管理和请假信息管理，这两者类似，这里以加班信息为例。

（1）初始状态：

行号	加班编号	工号	姓名	开始时间	结束时间	加班时长	操作
	输入编号	输入工号	姓名将自动匹配	输入开始时间	输入结束时间	加班时长自动计算	添加
1	2	1	Tom	2019/6/11 21:00:00	2019/6/11 23:00:00	2"	修改 删除
2	3	2	Jerry	2019/6/11 17:00:00	2019/6/11 23:00:00	6"	修改 删除
3	4	2	Jerry	2019/6/11 22:03:00	2019/6/13 0:36:00	2"	修改

图 6.6

（2）增加：

增加信息如下：

编号：1　工号：3　开始时间：2020/6/18 17:00:00

结束时间：2020/6/18 22:00:00

结果如下：

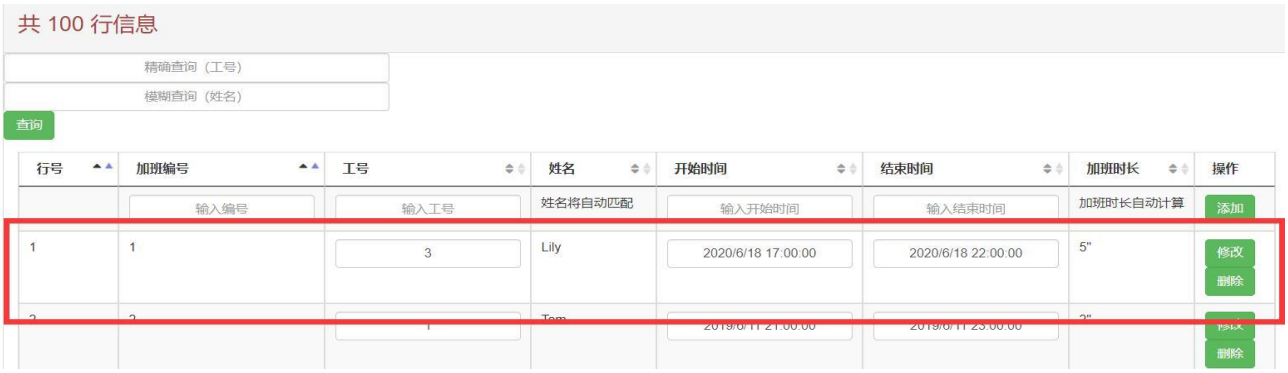


图 6.7 添加成功

（3）删除：

删除编号为 2 的记录，结果如下：

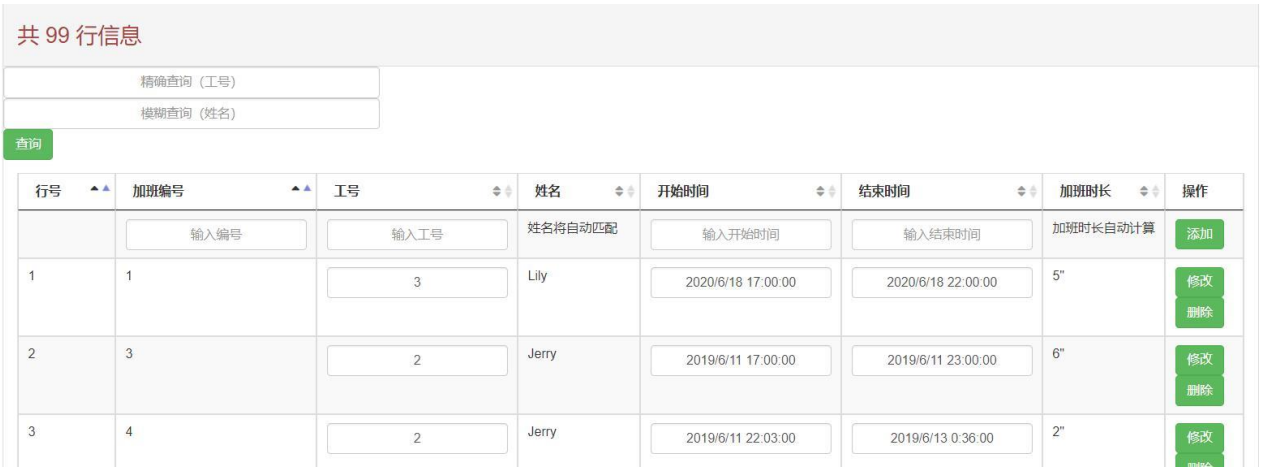


图 6.8 删除成功

（4）修改：

将新添加的记录的开始时间改为 2020/6/18 19:00:00，结果如下：

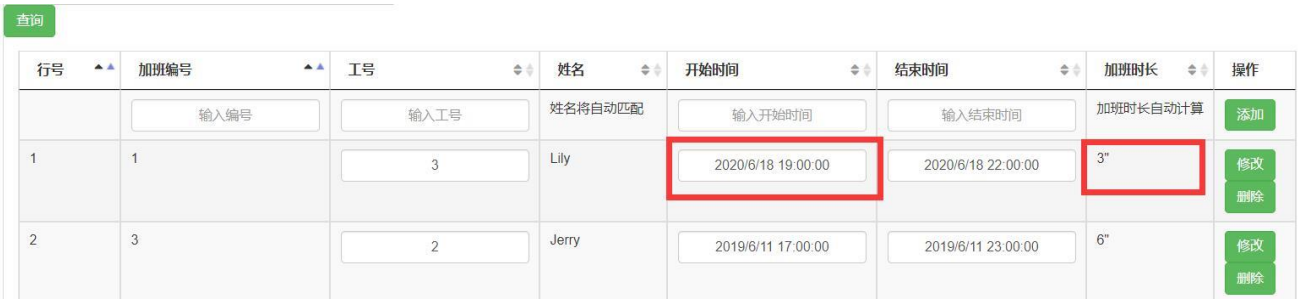


图 6.9 修改成功

（5）精确查询（工号）：

查询工号为 2 的员工的加班信息，结果如下：

行号	加班编号	工号	姓名	开始时间	结束时间	加班时长
1	3	2	Jerry	2019/6/11 17:00:00	2019/6/11 23:00:00	6
2	4	2	Jerry	2019/6/11 22:03:00	2019/6/13 0:36:00	2
3	9	2	Jerry	2020/3/13 0:00:00	2020/3/27 0:00:00	0
4	14	2	Jerry	2019/9/2 0:00:00	2019/9/11 0:00:00	9
5	24	2	Jerry	2019/9/3 0:00:00	2019/9/10 0:00:00	7
6	25	2	Jerry	2019/9/21 0:00:00	2019/10/7 0:00:00	16
7	26	2	Jerry	2020/5/10 0:00:00	2020/5/21 0:00:00	11
8	29	2	Jerry	2020/1/2 0:00:00	2020/1/6 0:00:00	4

图 6.9 查询成功

（6）模糊查询（姓名）：

查询名字中带 i 的员工的加班信息，结果如下：

行号	加班编号	工号	姓名	开始时间	结束时间	加班时长
1	1	3	Lily	2020/6/18 19:00:00	2020/6/18 22:00:00	3
2	5	3	Lily	2019/6/11 23:00:00	2019/6/13 0:00:00	1
3	8	3	Lily	2019/6/11 23:00:00	2019/6/13 0:00:00	1
4	12	3	Lily	2020/5/11 0:00:00	2020/5/26 0:00:00	15
5	18	3	Lily	2020/1/24 0:00:00	2020/1/31 0:00:00	7
6	23	3	Lily	2019/8/18 0:00:00	2019/8/27 0:00:00	9
7	31	3	Lily	2019/12/18 0:00:00	2020/1/1 0:00:00	14

图 6.10 查询成功（Lily）

3. 导入模块

导入前只有四个人：Tom　Jerry　Lily　Amy

图 6.11 导入前

C9

	A	B	C	D	E	F	G	H	I	J
1	5	jk	男	18	55555	1	普通岗	测试员	2000	在职
2	6	cxk	男	19	5151656	2	管理岗	项目经理	6000	在职
3	7	lk	女	20	1561618	3	普通岗	开发工程师	7000	在职
4										

图 6.12excel 文件

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

w_id	w_name	w_sex	w_age	w_telephone	w_department	w_post	w_duty	w_wage	w_on_job
1	Tom	男	27	123456789	1	管理岗	财务管理员	10000	在职
2	Jerry	男	45	789456123	2	普通岗	销售员	6000	在职
3	Lily	女	32	456312789	3	普通岗	招聘员	6000	在职
4	Amy	女	36	789132456	4	普通岗	程序员	6000	离职
5	jk	男	18	55555	1	普通岗	测试员	6000	在职
6	cxk	男	19	5151656	2	管理岗	项目经理	10000	在职
7	lk	女	20	1561618	3	普通岗	开发工程师	7000	在职
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

图 6.13 导入成功（MySQL workbench）

4. 统计模块

选择部门：

下图是数据库中已存数据，员工对应部门编号与自己工号的部门。

w_id	w_name	w_sex	w_age	w_telephone	w_department	w_post	w_duty	w_wage	w_on_job
1	Tom	男	27	123456789	1	管理岗	财务管理员	10000	在职
2	Jerry	男	45	789456123	2	普通岗	销售员	6000	在职
3	Lily	女	32	456312789	3	普通岗	招聘员	6000	在职
4	Amy	女	36	789132456	4	普通岗	程序员	6000	离职
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

图 6.14 员工信息

我们选择部门编号为 1 的财务部，结果如下图，符合预期。

出勤统计

统计：所有人

1	财务部	选择该部门	2	销售部	选择该部门
3	人力部	选择该部门	4	行政部	选择该部门
5	工程部	选择该部门			

工号	姓名	年	月	迟到次数	早退次数	旷工次数	正常次数
1	Tom	2019	5	1	5	3	1
1	Tom	2019	6	1	3	11	0
1	Tom	2019	7	1	6	12	3
1	Tom	2019	8	3	1	5	3
1	Tom	2019	9	3	6	14	2
1	Tom	2019	10	5	4	14	2
1	Tom	2019	11	4	9	7	3
1	Tom	2019	12	3	4	7	4
1	Tom	2020	1	4	6	6	3
1	Tom	2020	2	3	4	7	1
1	Tom	2020	3	1	5	13	2
1	Tom	2020	4	4	7	11	2
1	Tom	2020	5	0	4	9	0

图 6.15

我们选择部门编号为 5 的财务部，结果如下图，符合预期。

图 6.16

七、课程设计技术经验总结

这次课设中，我学会了许多关于 ASP.net 体系的知识，也让我对于数据库的运用更上一层。在我看来，数据库课程设计主要的目标是利用课程中学到的数据库知识和技术较好的开发设计出数据库应用系统，去解决各行各业信息化处理的要求。通过这次的课程设计，可以巩固我们对数据库基本原理和基础理论的理解，掌握数据库应用系统设计开发的基本方法，进一步提高我们综合运用所学知识的能力。

通过这次的课程设计，加深了我对数据库技术相关理论的理解，增强动手能力，掌握使用现行较为流行的数据库理论和数据库开发工具进行数据库管理操作（如建库、建立完整性约束，建立视图，对数据库中数据进行查询，更改等操作）的方法。

具体应用方面，使我对 SQL 语言要有较深入的了解和掌握，对数据库的管理要有一定程度的了解，并让我将之前学习过的程序设计知识、前端构架知识和数据库管理知识相结合，能够建立一个具有较完整结构的 web 应用。这次课设中，我也体会到了在一个小型项目团队中如何合作，互相帮助，共同完成项目。

最主要的是我对于应用开发中，数据层、业务层、表示层的理解加深，也学到了其进行数据交互的一些手段。

八、附录

以下为我负责模块的部分代码

1. 员工考勤管理模块

（1）业务层（C#）：

以下是员工考勤管理模块的控制器。

```
1.  public class staffController : Controller
2.  {
3.      // GET: /<controller>/
4.
```

```
5.      public IActionResult find_achievements()
6.      {
7.          using (var context = new staff_attendanceContext())
8.          {
9.
10.         }
11.             return View();
12.     }
13.     public bool check_staff(int id = -1,string name=" ")
14.     {
15.         using (var context = new staff_attendanceContext())
16.         {
17.             var worker = context.Worker.ToList();
18.             if (worker.Find(o => o.WId == id && o.WName == name) != null)
19.                 return true;
20.         }
21.         return false;
22.     }
23.
24.     public IActionResult attendance([FromQuery]int id)
25.     {
26.         using (var context = new staff_attendanceContext())
27.         {
28.             this.ViewBag.name = "所有人";
29.             var att = context.AttendanceStatistic.ToList();
30.             if (id != 0)
31.                 att.RemoveAll(o => o.Id != id);
32.             this.ViewData["att"] = att;
33.             if (id > 0) this.ViewBag.name = att[0].Name;
34.         }
35.         return View();
36.     }
37.     public IActionResult bussiness([FromQuery]int id)
38.     {
39.         using (var context = new staff_attendanceContext())
40.         {
41.             this.ViewBag.name = "所有人";
42.             var bus = context.BusinessStatistic.ToList();
43.             if (id != 0)
44.                 bus.RemoveAll(o => o.Id != id);
45.             this.ViewData["bus"] = bus;
46.             if (id > 0) this.ViewBag.name = bus[0].Name;
47.         }
48.         return View();
```

```
    }
    public IActionResult offwork([FromQuery]int id)
    {
        using (var context = new staff_attendanceContext())
        {
            this.ViewBag.name = "所有人";
            var off = context.OffworkStatistic.ToList();
            if (id != 0)
                off.RemoveAll(o => o.Id != id);
            this.ViewData["off"] = off;
            if (id > 0) this.ViewBag.name = off[0].Name;
        }
        return View();
    }
    public IActionResult overtime([FromQuery]int id)
    {
        using (var context = new staff_attendanceContext())
        {
            this.ViewBag.name = "所有人";
            var over = context.OvertimeStatistic.ToList();
            if (id != 0)
                over.RemoveAll(o => o.Id != id);
            this.ViewData["over"] = over;
            if (id > 0) this.ViewBag.name = over[0].Name;
        }
        return View();
    }

}
```

（2）数据交互（js）：

以选择出勤记录为例。

```
function att()
{
    var wid = document.getElementById("nid").value;
    var wname = document.getElementById("name").value;


    var xmlhttp;
    if (window.XMLHttpRequest)
    {// code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp = new XMLHttpRequest();
    }
```

```
    else
    {// code for IE6, IE5
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
        {
            var f = xmlhttp.responseText;
            if (xmlhttp.responseText == "true")
                window.location.href = "@Url.Action("attendance","staff")?id
=" + wid;
            else
                alert("查无此人");
        }
    }

    xmlhttp.open("POST", "/staff/check_staff", true);
    xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlenco
ded");
    xmlhttp.send("&id=" + wid + "&name=" + wname);
}

```

2. 管理员考勤管理模块（以加班信息为例）

（1）业务层（C#）：

包括视图调用函数、增、删、改、查信息函数。

```
public IActionResult overtime_information()
{
    using (var context = new staff_attendanceContext())
    {
        var over = context.Overtime.ToList();
        var staff = context.Worker.ToList();
        this.ViewData["staff"] = staff;
        this.ViewData["over"] = over;
    }
        return View();
}
public bool detele_over(int num = -1)
{
    using (var context=new staff_attendanceContext())
    {
```

```
16.         var over = context.Overtime.ToList();
17.         if (num != -1)
18.         {
19.             context.Overtime.Remove(over[num]);
20.             context.SaveChanges();
21.             return true;
22.         }
23.
24.     }
25.     return false;
26. }
27. public bool change_over(int num = -1,int id=-1, string starttime=null, strin
    g endtime=null)
28. {
29.     using (var context = new staff_attendanceContext())
30.     {
31.         var over = context.Overtime.ToList();
32.         if (num != -1)
33.         {
34.             var x = over[num];
35.             context.Overtime.Remove(over[num]);
36.             context.SaveChanges();
37.             x.WId = id;
38.             x.OverStartTime = DateTime.Parse(starttime);
39.             x.OverEndTime = DateTime.Parse(endtime);
40.             TimeSpan dur = (TimeSpan)(x.OverEndTime - x.OverStartTime);
41.             x.OverDuration = dur.Hours;
42.             context.Overtime.Add(x);
43.             context.SaveChanges();
44.             return true;
45.         }
46.     }
47.     return false;
48. }
49. public bool add_over(int num = -1, int id = -1, string starttime = null, str
    ing endtime = null)
50. {
51.
52.     using (var context = new staff_attendanceContext())
53.     {
54.         Overtime x = new Overtime();
55.         x.WId = id;
56.         x.OverNum = num;
57.         x.OverStartTime = DateTime.Parse(starttime);
```

```
58.         x.OverEndTime = DateTime.Parse(endtime);
59.         TimeSpan dur = (TimeSpan)(x.OverEndTime - x.OverStartTime);
60.         x.OverDuration = dur.Hours;
61.         context.Overtime.Add(x);
62.         context.SaveChanges();
63.     }
64.     return false;
65. }
66. public string over_search(int id=-1,string name = null)
67. {
68.     string ans = "";
69.     using (var context = new staff_attendanceContext())
70.     {
71.         if (id != -1)
72.         {
73.             //得到全部出差记录
74.             var over = context.Overtime.Where(x => x.WId == id).ToList();
75.             //得到员工信息
76.             var workers = context.Worker.ToList();
77.             for(int i = 0; i < over.Count; i++)
78.             {
79.                 ans = ans + "<tr><td>" + (i + 1) + "</td>";
80.                 ans = ans + "<td>" + over[i].OverNum + "</td>";
81.                 ans = ans + "<td>" + over[i].WId + "</td>";
82.                 ans = ans + "<td>" + workers[id-1].WName + "</td>";
83.                 ans = ans + "<td>" + over[i].OverStartTime + "</td>";
84.                 ans = ans + "<td>" + over[i].OverEndTime + "</td>";
85.                 ans = ans + "<td>" + over[i].OverDuration + "</td></tr>";
86.             }
87.             return ans;
88.         }
89.
90.         //如果给的条件是姓名，模糊查询
91.         else if (name != null)
92.         {
93.             //把员工信息用姓名模糊查询进行筛选
94.             var workers = context.Worker.Where(x => x.WName.Contains(name)).
    ToList();
95.             for (int i = 0; i < workers.Count; i++)
96.             {
97.                 var over = context.Overtime.Where(x => x.WId == workers[i].W
    Id).ToList();
98.                 for(int j = 0; j < over.Count; j++)
99.                 {
```

```
                    ans = ans + "<tr><td>" + (j + 1) + "</td>";
                    ans = ans + "<td>" + over[j].OverNum + "</td>";
                    ans = ans + "<td>" + over[j].WId + "</td>";
                    ans = ans + "<td>" + workers[i].WName + "</td>";
                    ans = ans + "<td>" + over[j].OverStartTime + "</td>";
                    ans = ans + "<td>" + over[j].OverEndTime + "</td>";
                    ans = ans + "<td>" + over[j].OverDuration + "</td><tr>"
    ;
                }
            }
            return ans;
        }

    }
    return null;
}
```

（2）数据交互（js）：

过于增删改查的交互函数。

```
function detele(i)
{
    var xmlhttp;
    if (window.XMLHttpRequest)
    {// code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp = new XMLHttpRequest();
    }
    else
    {// code for IE6, IE5
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
        {
            window.location.href = "@Url.Action("overtime_information","mana
    ger")";
        }
    }

    xmlhttp.open("Post", "/manager/detele_over", true);
    xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlenco
    ded");
    xmlhttp.send("num=" + i);
```

```
}
function ch(i)
{
    var id = document.getElementById("wid_" + i).value;
    var start = document.getElementById("starttime_" + i).value;
    var end = document.getElementById("endtime_" + i).value;
    var xmlhttp;
    if (window.XMLHttpRequest)
    {// code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp = new XMLHttpRequest();
    }
    else
    {// code for IE6, IE5
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
        {
            window.location.href = "@Url.Action("overtime_information","manager")";
        }
    }

    xmlhttp.open("Post", "/manager/change_over", true);
    xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    xmlhttp.send("&num=" + i + "&id=" + id + "&starttime=" + start + "&endtime=" + end);
}
function add()
{
    var num = document.getElementById("num").value;
    var id = document.getElementById("id_a").value;
    var start = document.getElementById("start_a").value;
    var end = document.getElementById("end_a").value;

    var xmlhttp;
    if (window.XMLHttpRequest)
    {// code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp = new XMLHttpRequest();
    }
    else
    {// code for IE6, IE5
```

```
64.         xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
65.     }
66.
67.     xmlhttp.onreadystatechange = function() {
68.         if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
69.         {
70.             window.location.href = "@Url.Action("overtime_information","mana
    ger")";
71.         }
72.     }
73.
74.     xmlhttp.open("Post", "/manager/add_over", true);
75.     xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlenco
    ded");
76.     xmlhttp.send("&num=" + num + "&id=" + id + "&starttime=" + start + "&end
    time=" + end);
77. }
78. function search()
79. {
80.     var xmlhttp;
81.     var id = document.getElementById("s_id").value;
82.     var name = document.getElementById("s_name").value;
83.
84.     if (window.XMLHttpRequest)
85.     {// code for IE7+, Firefox, Chrome, Opera, Safari
86.         xmlhttp = new XMLHttpRequest();
87.     }
88.     else
89.     {// code for IE6, IE5
90.         xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
91.     }
92.
93.     xmlhttp.onreadystatechange = function() {
94.         if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
95.         {
96.             alert("查询成功");
97.             document.getElementById("table").innerHTML = xmlhttp.responseTex
    t;
98.             document.getElementById("mydiv").style.display = "";
99.             document.getElementById("div").style.display = "none"
             }
100.     }
101.
102.     xmlhttp.open("Post", "/manager/over_search", true);
```

```
    xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    xmlhttp.send("&id=" + id + "&name=" + name);
}
```

3. 导入模块

（1）业务层（C#）：

同时依赖了这三个 NuGet 包：ExcelDataReader,ExcelDataReader .DataSet,System.IO.FileSystem

```
public IActionResult import( string address=null)
{
     using (var context = new staff_attendanceContext())
    {
        if (address != null)//如果有地址
        {
            Worker n = new Worker();
            System.Text.Encoding.RegisterProvider(System.Text.CodePagesEncodingProvider.Instance);
            using (FileStream x = new FileStream(address, FileMode.Open, FileAccess.Read))
            {
                IExcelDataReader excelreader = ExcelReaderFactory.CreateOpenXmlReader(x);
                DataSet result = excelreader.AsDataSet();
                DataTable sheet = result.Tables[0];
                for(int i = 0; i < sheet.Rows.Count; i++)
                {
                    n.WId = int.Parse(sheet.Rows[i][0].ToString());
                    n.WName = sheet.Rows[i][1].ToString();
                    n.WSex = sheet.Rows[i][2].ToString();
                    n.WAge = int.Parse(sheet.Rows[i][3].ToString());
                    n.WTelephone = sheet.Rows[i][4].ToString();
                    n.WDepartment = int.Parse(sheet.Rows[i][5].ToString());

                    n.WPost = sheet.Rows[i][6].ToString();
                    n.WDuty = sheet.Rows[i][7].ToString();
                    n.WWage = int.Parse(sheet.Rows[i][8].ToString());
                    n.WOnJob = sheet.Rows[i][9].ToString();

                    context.Worker.Add(n);
                    context.SaveChanges();
                }
            }
```

```
        }

        var staff = context.Worker.ToList();
        this.ViewData["staff"] = staff;
    }
    return View();
}
```

（2）数据交互（js）：

```
function im()
{
    var x = document.getElementById("input").value;

    var xmlhttp;
    if (window.XMLHttpRequest)
    {// code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp = new XMLHttpRequest();
    }
    else
    {// code for IE6, IE5
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
        {
            window.location.href = "@Url.Action("import","manager")";
            alert("导入成功");
        }
    }

    xmlhttp.open("Post", "/manager/import", true);
    xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
    xmlhttp.send("&address=" + x);
}
```

4. 统计模块

（1）业务层（c#）：

以出勤统计为例：

```
public IActionResult attendance([FromQuery]int id)
{
    using (var context = new staff_attendanceContext())
    {
        var att = context.AttendanceStatistic.ToList();
        var work = context.Worker.ToList();
        if (id != 0)
        {
            work.RemoveAll(o => o.WDepartment == id);
            List<AttendanceStatistic>[] a = new List<AttendanceStatistic>[work.Count];
            for(int i = 0; i < work.Count; i++)//该部门每个员工的出勤统计信息分开存储
            {
                a[i] = att.FindAll(o => o.Id == work[i].WId);
            }
            att = a[0];
            for (int i = 0; i < work.Count-1; i++)//list 合并
            {
                att.AddRange(a[i + 1]);
            }
        }
        this.ViewData["att"] = att;
        if (id > 0) this.ViewBag.name = att[0].Name;
        var department = context.Department.ToList();
        this.ViewData["dep"] = department;
    }
    return View();
}
```

（2）数据交互（js）：

```
function search(did)
{
    var xmlhttp;
    var id = document.getElementById("s_id").value;

    if (window.XMLHttpRequest)
    {// code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp = new XMLHttpRequest();
    }
    else
    {// code for IE6, IE5
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
```

```
    }

    xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
        {
            alert("查询成功");
            document.getElementById("table").innerHTML = xmlhttp.responseTex
t;
            document.getElementById("mydiv").style.display = "";
            document.getElementById("div").style.display = "none"
         }
    }

    xmlhttp.open("Post", "/statistics/attendance", true);
    xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlenco
ded");
    xmlhttp.send("&id=" + did);
}
```

信控学院课程设计指导教师成绩评定标准及成绩评定表

学生姓名：蒋坤　　学号：1706020115　　专业班级：计算机 1701

所属学院：信控　　所在专业：计算机　　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	备注
学习态度	15	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中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，能按期完成任务	学习马虎，纪律涣散，工作作风不严谨，不能保证设计时间和进度		
技术水平与实际能力	25	设计合理、理论分析与计算正确，实验数据准确，有很强的实际动手能力、经济分析能力和计算机应用能力，文献查阅能力强、引用合理、调查调研非常合理、可信	设计合理、理论分析与计算正确，实验数据比较准确，有较强的实际动手能力、经济分析能力和计算机应用能力，文献引用、调查调研比较合理、可信	设计合理，理论分析与计算基本正确，实验数据比较准确，有一定的实际动手能力，主要文献引用、调查调研比较可信	设计基本合理，理论分析与计算无大错，实验数据无大错	设计不合理，理论分析与计算有原则错误，实验数据不可靠，实际动手能力差，文献引用、调查调研有较大的问题		
论文(计算书、图纸)撰写质量	60	结构严谨，逻辑性强，层次清晰，语言准确，文字流畅，完全符合规范化要求，书写工整或用计算机打印成文；图纸非常工整、清晰	结构合理，符合逻辑，文章层次分明，语言准确，文字流畅，符合规范化要求，书写工整或用计算机打印成文；图纸工整、清晰	结构合理，层次较为分明，文理通顺，基本达到规范化要求，书写比较工整；图纸比较工整、清晰	结构基本合理，逻辑基本清楚，文字尚通顺，勉强达到规范化要求；图纸比较工整	内容空泛，结构混乱，文字表达不清，错别字较多，达不到规范化要求；图纸不工整或不清晰		

指导教师签名：

年　　月　　日