**《数据库课程设计报告》**

企业篇——人事管理系统

目录

[1 设计目的与要求 3](#_Toc139471062)

[1.1 设计目的 3](#_Toc139471063)

[1.2 设计要求 3](#_Toc139471064)

[2 设计内容 4](#_Toc139471065)

[3 需求分析 4](#_Toc139471066)

[3.1 用户需求 4](#_Toc139471067)

[3.2 功能模块分析 6](#_Toc139471068)

[3.3 功能模块划分 7](#_Toc139471069)

[3.4 数据流图 7](#_Toc139471070)

[4 概念结构设计 12](#_Toc139471071)

[4.1 涉及的实体 12](#_Toc139471072)

[4.2 实体之间的关系 12](#_Toc139471073)

[4.3 实体应该具有的属性 13](#_Toc139471074)

[4.4 ER图绘制 15](#_Toc139471075)

[5 逻辑结构设计与物理结构设计 16](#_Toc139471076)

[5.1 关系模型设计 16](#_Toc139471077)

[5.2 关系模式规范化 17](#_Toc139471078)

[5.3 外模式(视图)设计 17](#_Toc139471079)

[5.4 权限控制 18](#_Toc139471080)

[5.5 数据完整性的考虑 19](#_Toc139471081)

[6 代码编写与具体实现 20](#_Toc139471082)

[6.1 实现环境 20](#_Toc139471083)

[6.2 主要的业务流程 22](#_Toc139471084)

[6.3 安全性的考虑 22](#_Toc139471085)

[6.4 后端代码的性能考量 24](#_Toc139471086)

[6.5 前端页面使用的框架 26](#_Toc139471087)

[6.6 测试工作 27](#_Toc139471088)

[7 实际运行效果展示 28](#_Toc139471089)

[7.1 前端页面模块展示 28](#_Toc139471090)

[7.2 部分功能展示 33](#_Toc139471091)

[8 遇到的问题与解决办法 39](#_Toc139471092)

[8.1 问题一——MySQL DROP表卡死 39](#_Toc139471093)

[8.2 问题二——事务提交 40](#_Toc139471094)

[8.3 问题三——删除一周前的考勤记录问题 43](#_Toc139471095)

[总结 44](#_Toc139471096)

# 1 设计目的与要求

## 1.1 设计目的

此次课程设计主要从以下几个方面锻炼学生的实践动手能力：

1. 数据库设计与管理技能：通过课程设计，学生将学习数据库的基本理论、概念和设计原则，并掌握数据库管理系统的使用方法。这些技能对于未来从事与数据库相关的工作或研究具有重要意义。
2. 实际项目经验：在数据库课程设计中，学生通常需要完成一个真实的数据库项目，这将使他们有机会将理论知识应用于实际情况中，锻炼解决问题的能力和项目管理技能。
3. 团队合作与沟通能力：数据库课程设计通常是团队合作的过程，学生需要与同学一起协作完成项目。这将培养学生的团队合作和沟通能力，提高与他人协作的能力。
4. 问题解决与创新能力：在数据库课程设计中，学生可能会面临各种问题和挑战，需要通过分析和创新来解决这些问题。这将培养学生的问题解决能力和创新思维。
5. 软件工程实践：数据库课程设计通常需要学生按照软件工程的流程进行需求分析、设计、编码、测试和维护。这将使学生熟悉软件开发的实践流程，提高项目管理和软件开发能力。

## 1.2 设计要求

**总体要求**

分组：按课题组方式开发，每组3~4人，组长一名，角色划分、任务分配

数据库平台：Oracle、MS SQL Server、MySQL

开发平台：.NET、J2EE、PHP

软件架构：B/S、C/S

**具体要求**

课题组选题后，进入需求分析阶段。提过网上查阅、内部讨论等手段，明确课题的各种需求，包括：数据需求、功能需求及其他需求。需求分析由课题组共同完成！

在系统设计阶段，由课题组长进行任务分配。课题的所有成员均需完成一定的设计任务。

编程实现阶段，每一位成员均需完成一定的程序量。课题组中若存在没有参与编程的成员，除了该生得成绩为不及格外，组长的成绩也将降档处理。

# 2 设计内容

完成一个基本的人事管理系统，该管理系统需要涉及：员工基本信息、岗位系统、员工岗位变动、员工考勤信息、部门信息、学历信息、员工薪资及变动信息、员工培训信息等相关实体。同时从以下几个方面着重考虑：

1. 需求分析（课题组成员通力合作，力求需求分析的全面、有效！）
2. 数据库设计（数据涵盖系统的数据需求，关系模式均达到3NF，性能优化自己思考）
3. 数据完整性（分析关键表中的关键数据，制定自己的完整性约束规则！）
4. 安全性（必须具有基本的用户及其权限的控制手段！）
5. 视图（从操作的方便性、数据的安全性、数据的逻辑独立性等方面综合考虑！）
6. 触发器（根据对应用的理解，是否采用触发器？带来的好处？）
7. 存储过程（在充分理解系统业务处理的前提下，作出自己的判断，要运用得当！）
8. 索引（需结合数据量的估算、应用的特点等，做出合理的设计）
9. 系统功能（在需求分析的基础上尽可能全面一些！）
10. 用户界面友好性（多从用户的角度考虑！）

# 3 需求分析

## 3.1 用户需求

**（一）登陆管理**

1. 管理员工的登录——个人账号
2. 普通员工的登录——个人账号
3. 管理员对于新来员工的账号密码分配
4. 个人密码的修改

**（二）员工档案管理**

1. 管理员可以对员工的基本信息的增删改查，普通员工仅可查看；
2. 对公司里所有员工进行分配工号，并进行基本信息的录入；
3. 对新聘用的员工，将其信息加入到员工档案记录中；
4. 对于解聘的员工，将其信息从员工档案记录中删除。
5. 当员工信息发生变动时，修改员工档案记录中相应的属性。

**（三）员工岗位信息管理**

1. 对公司里所有员工的职务及岗位信息（岗位职责）进行记录；
2. 记录员工调动前后的具体职务，以及调动时间。

**（四）考勤管理**

1. 对员工上班刷卡的记录进行统一编号；登记员工上班时间（准时、迟到）。
2. 对员工下班刷卡的记录进行统一编号；登记员工下班时间（准时、早退）。
3. 当日考勤情况统计

**（五）部门信息管理**

1. 对于部门人数，部门主管，部门负责任务，部门成立时间等的管理

**（六）员工薪资及变动信息管理**

1. 各个部门的员工薪资情况
2. 员工薪资变动时间，原因，变动大小，增减

**（七）员工培训管理**

1. 对培训员工进行培训时间，培训地点，培训基本信息的管理

**（八）系统维护管理**

1. 对整个数据库进行整理、备份和授权管理。

## 3.2 功能模块分析

1. 用户登录模块
   1. 管理员登录
   2. 普通员工登录
   3. 个人密码修改
2. 员工档案管理模块
   1. 管理员对于个人档案的修改与查询
   2. 对于新聘用员工的档案录入
   3. 对于解聘员工的档案删除
   4. 员工对个人基本信息的查看
3. 员工岗位信息管理模块
   1. 对所有员工的岗位信息的增删改查
   2. 对员工岗位变动的记录
4. 考勤管理模块
   1. 员工签到签退记录
   2. 员工请假管理
   3. 员工加班管理
   4. 管理员查看当日考勤统计
5. 部门信息管理模块
   1. 部门人员情况的增删改查
6. 薪资管理模块
   1. 员工薪资情况的增删改查
   2. 员工薪资变动时间，原因，变动大小，增减
7. 员工培训管理模块
   1. 对培训基本信息的增删改查
8. 系统维护模块
   1. 管理员对系统维护信息进行管理与备份的操作。

## 3.3 功能模块划分

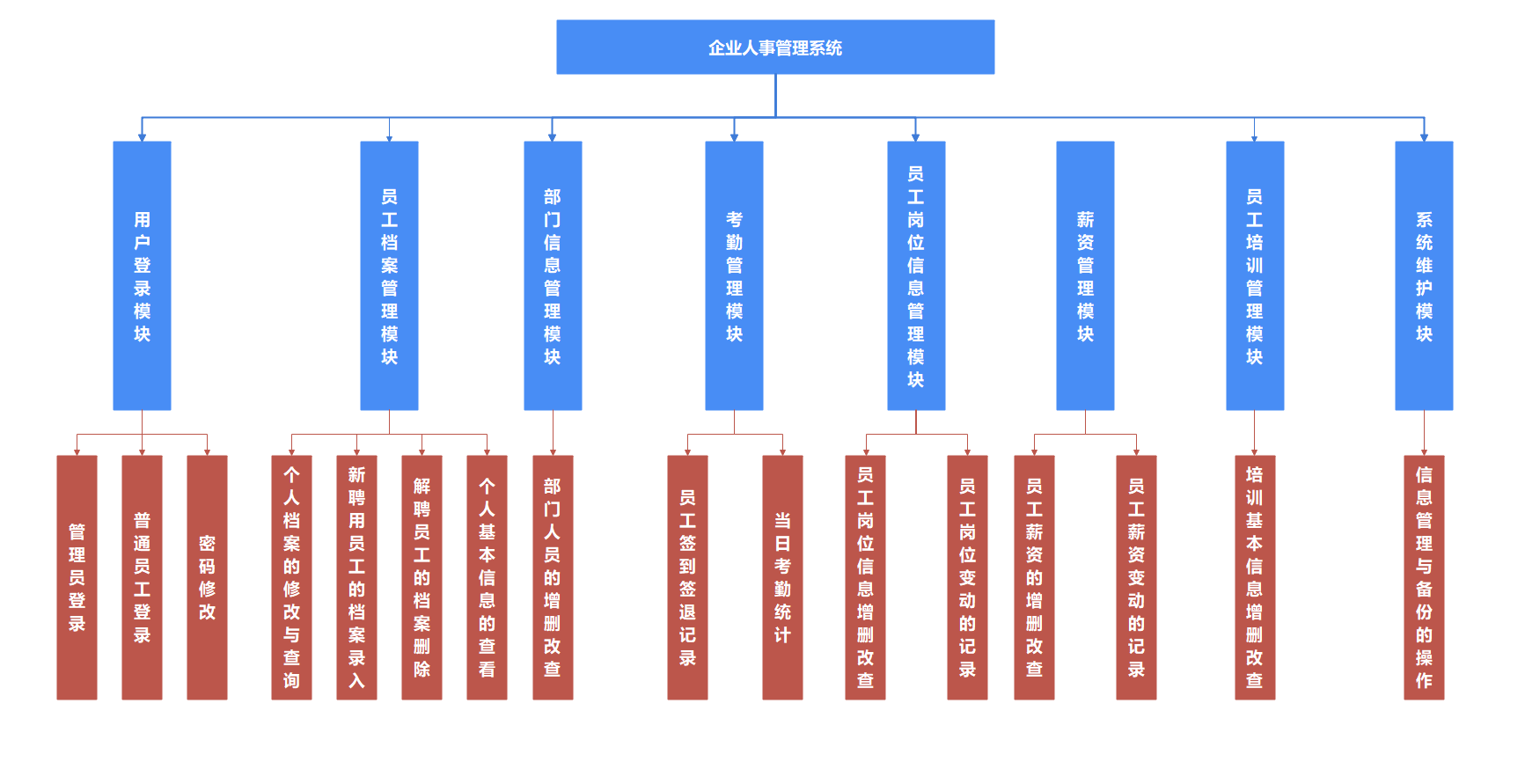


图 1 功能模块划分图

## 3.4 数据流图

顶层数据流图

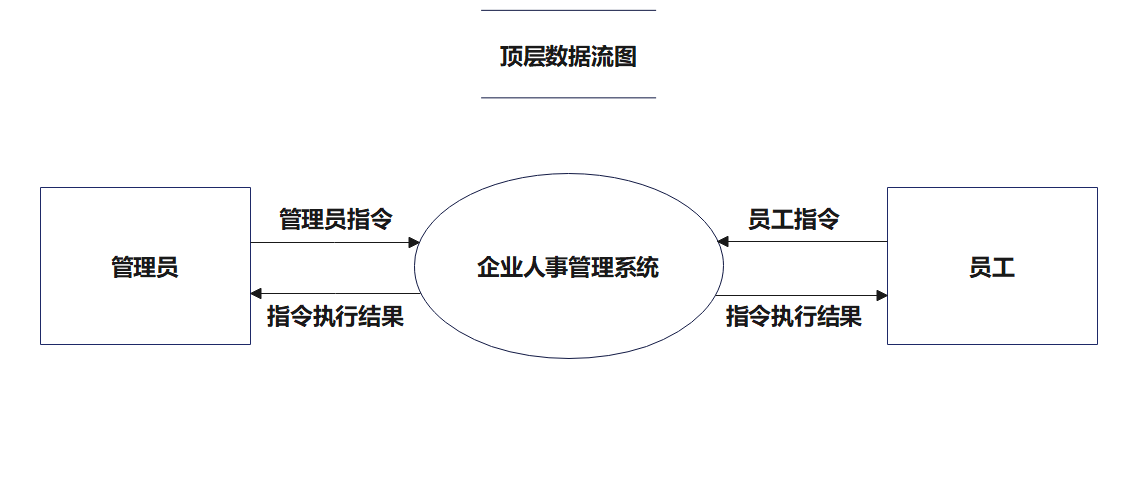


图 2 顶层数据流图

第一层数据流图

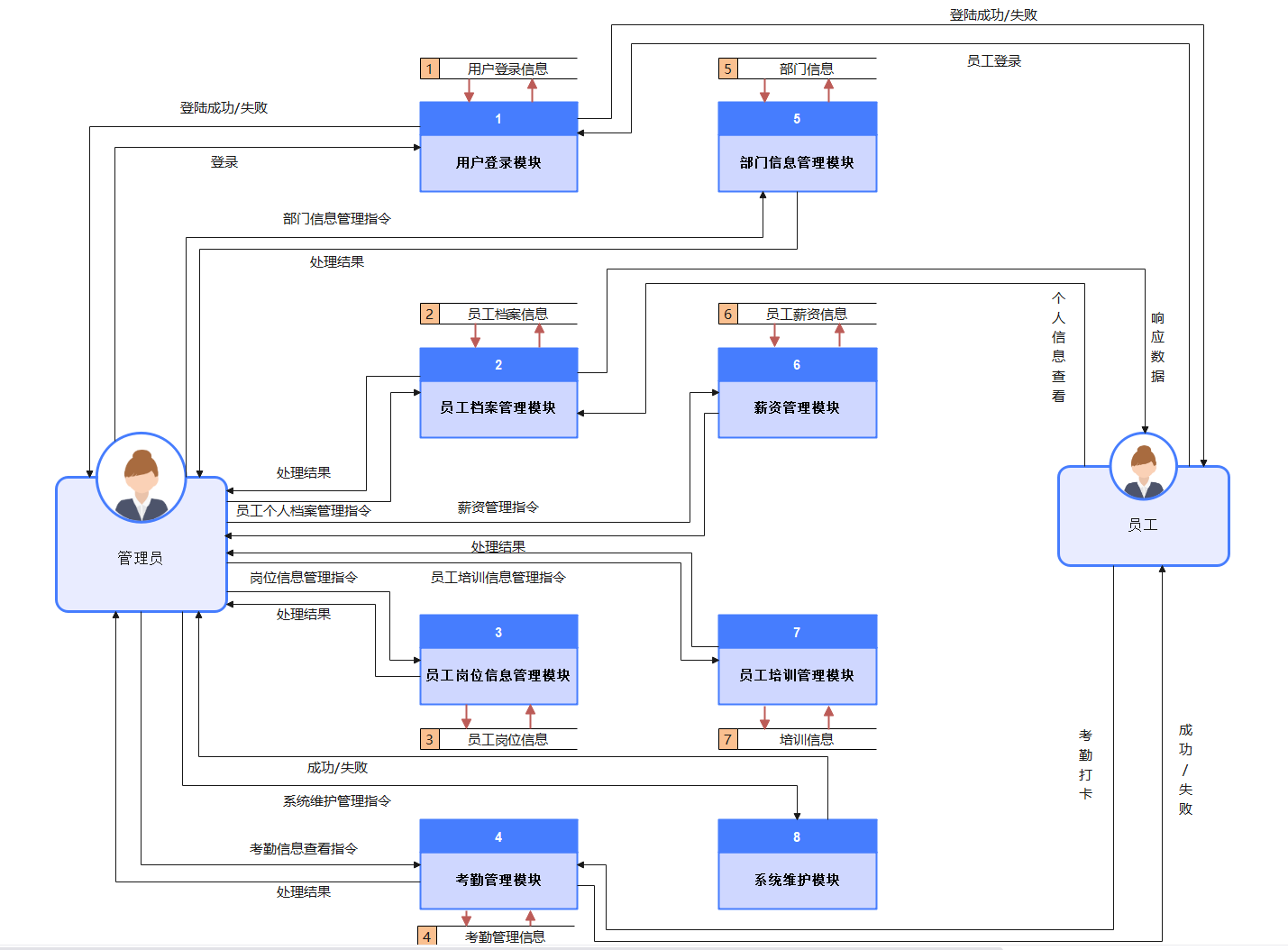


图 3第一层数据流图

第二层数据流图

用户登录管理

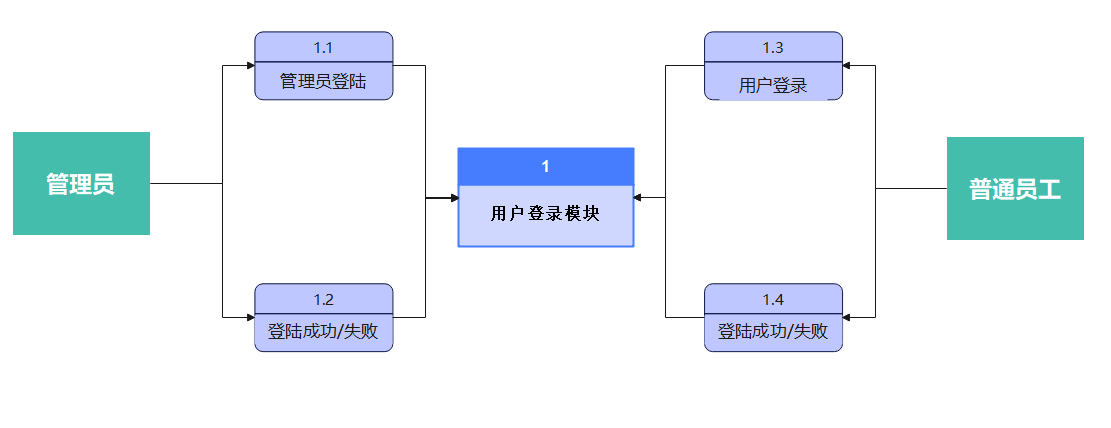


图 4 用户登录管理

员工档案管理

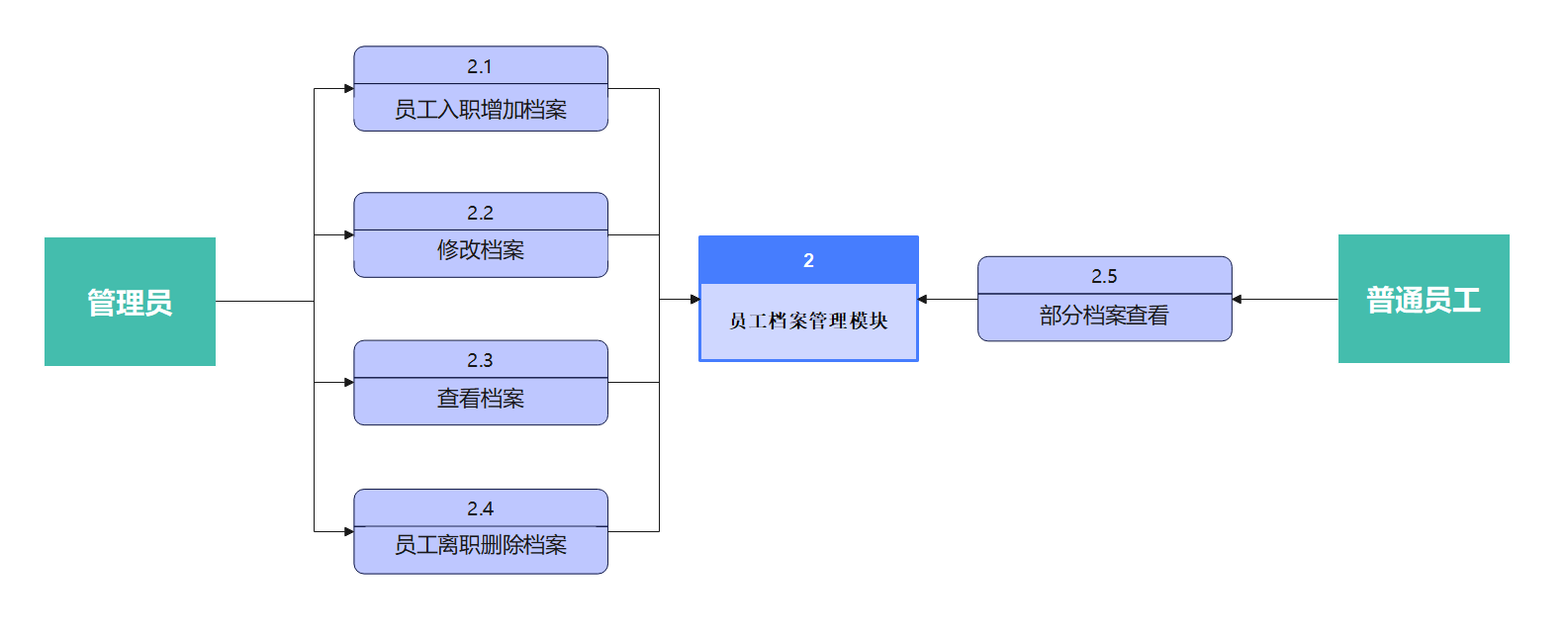


图 5 员工档案管理

岗位信息管理

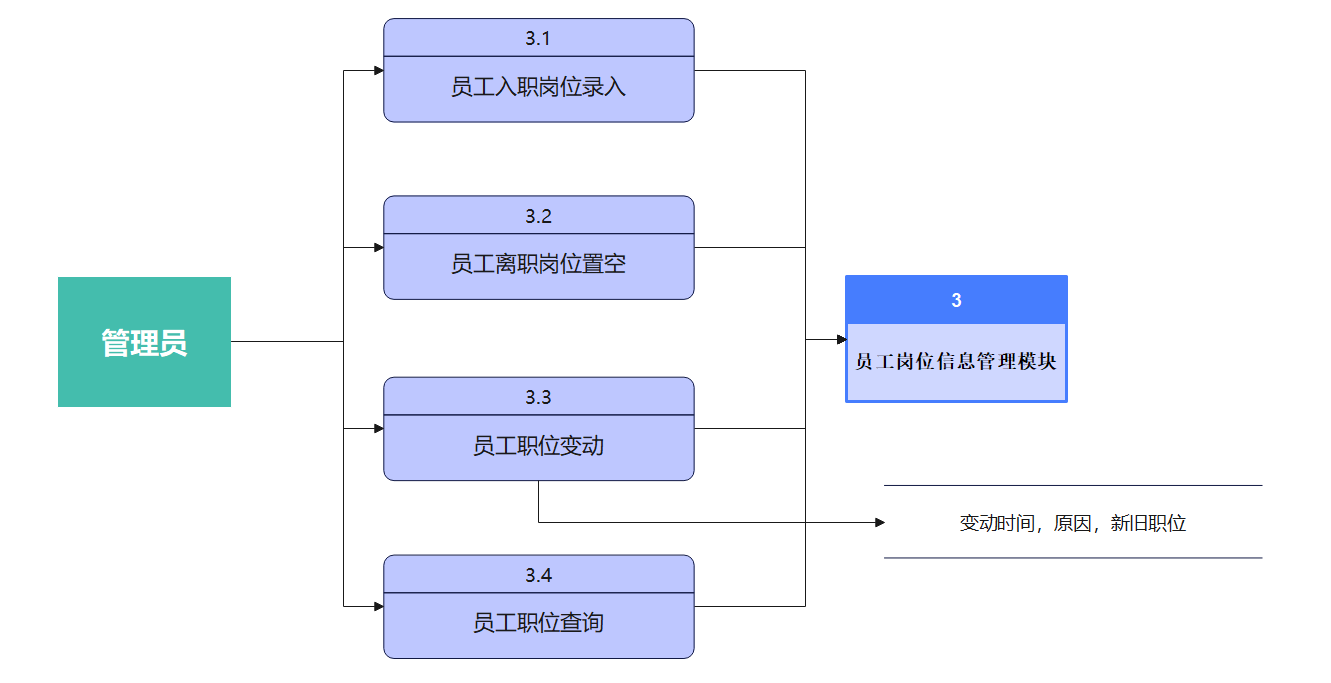


图 6 岗位信息管理

考勤状况管理

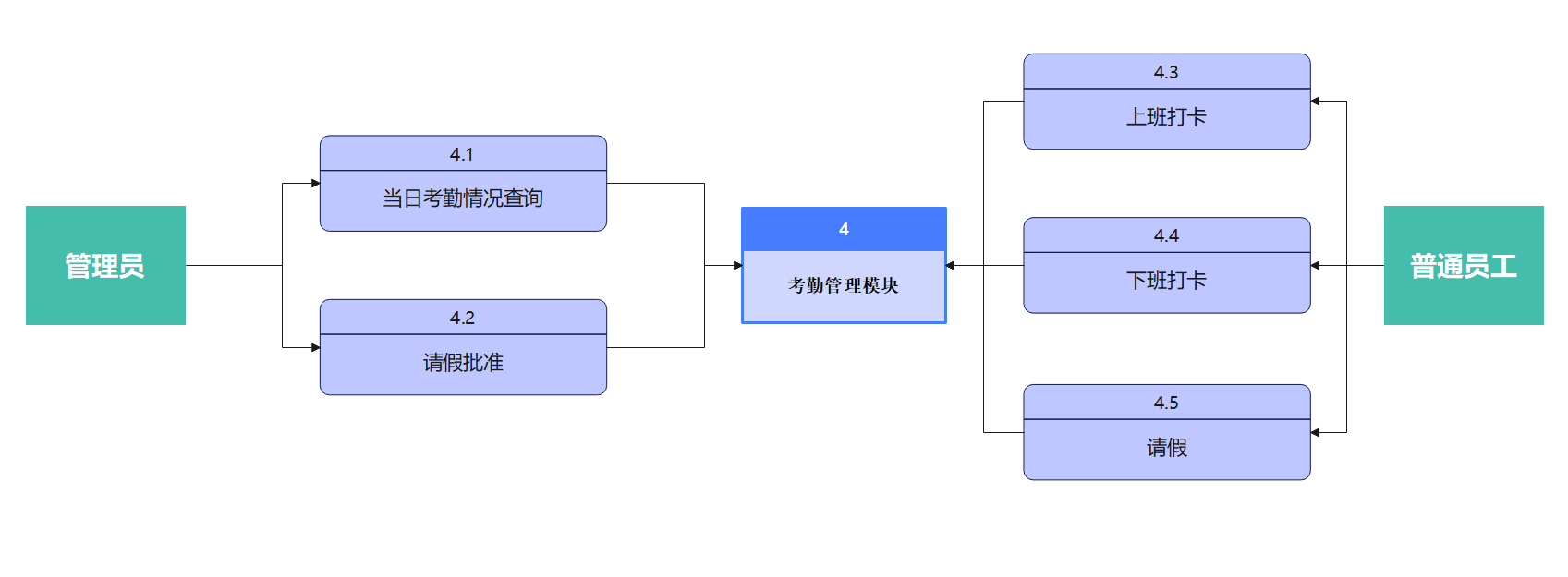


图 7 考勤状况管理

部门信息管理

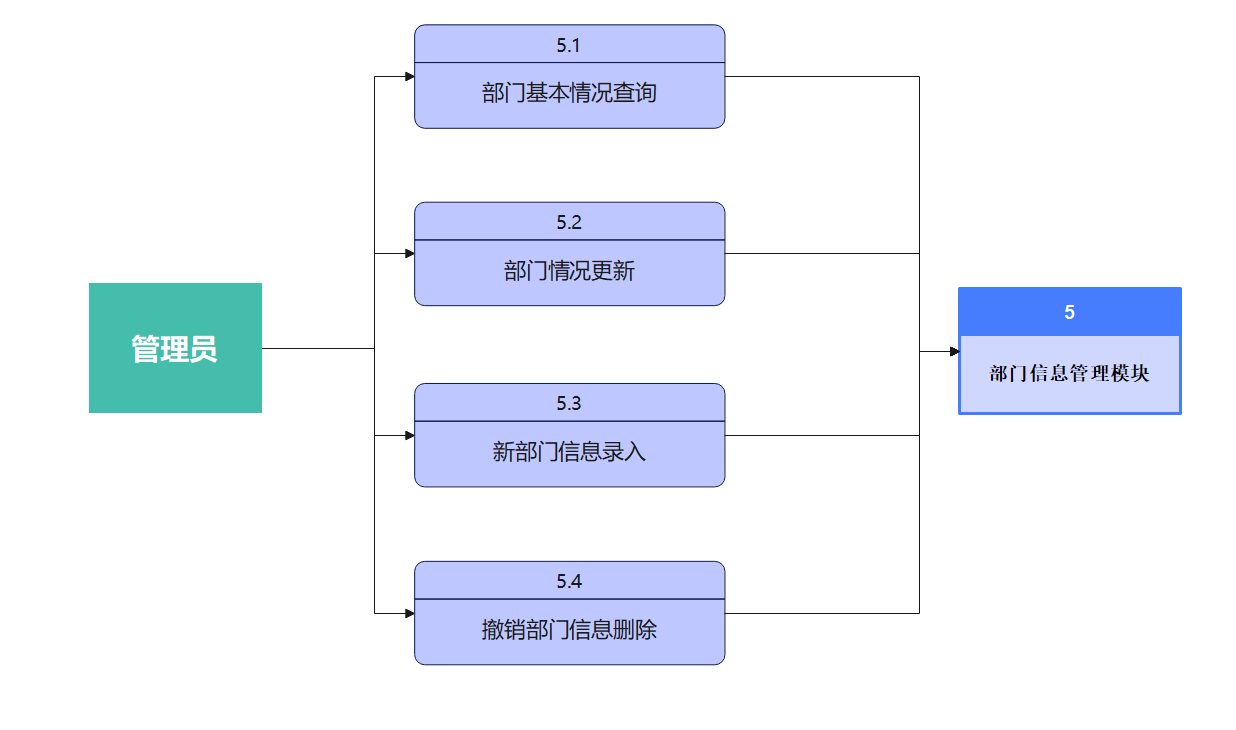


图 8 部门信息管理

薪资状况管理

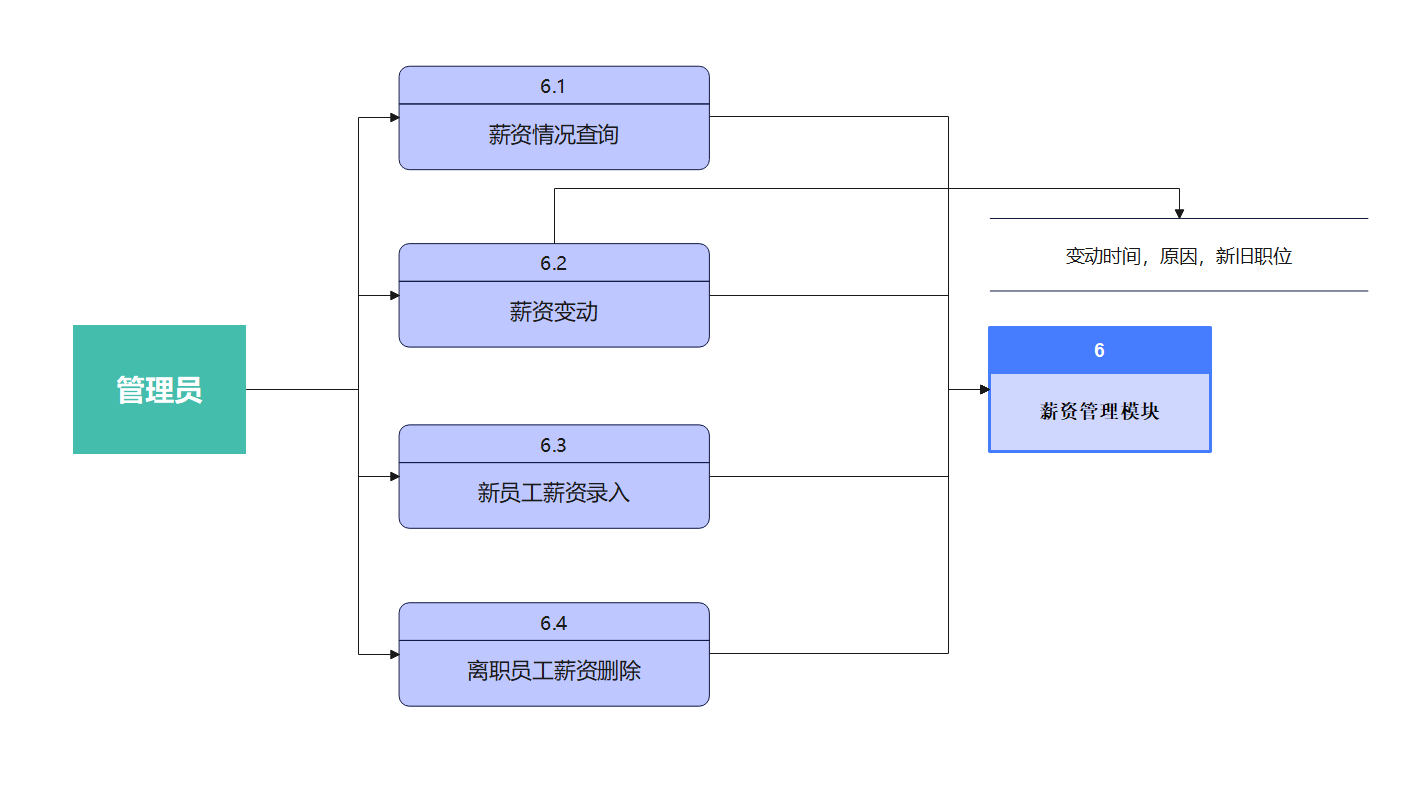


图 9 薪资状况管理

员工培训管理

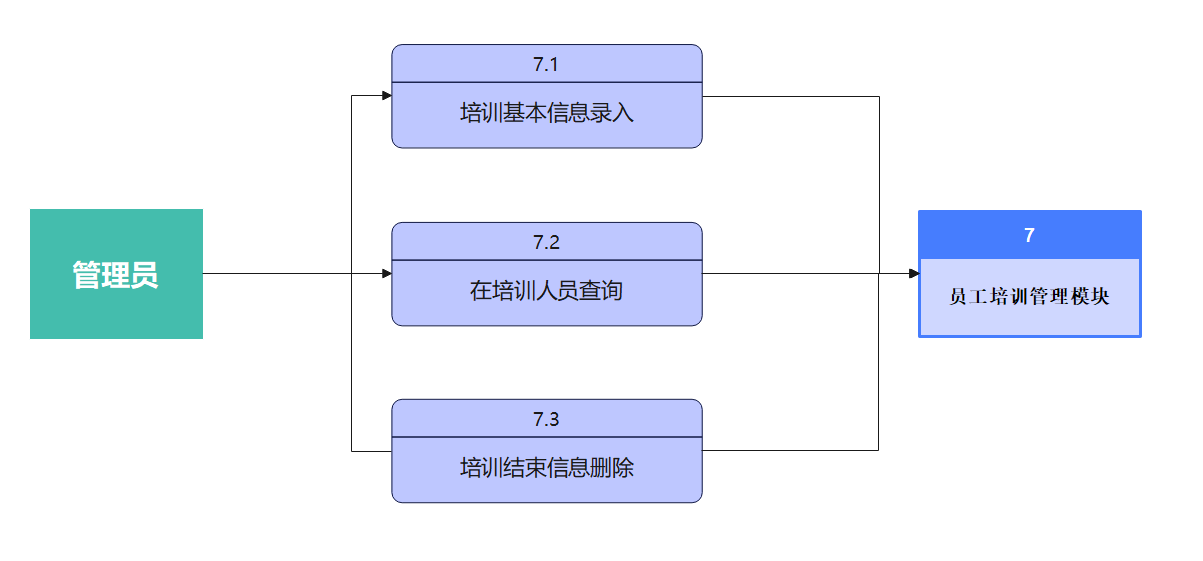


图 10 员工培训管理

系统维护

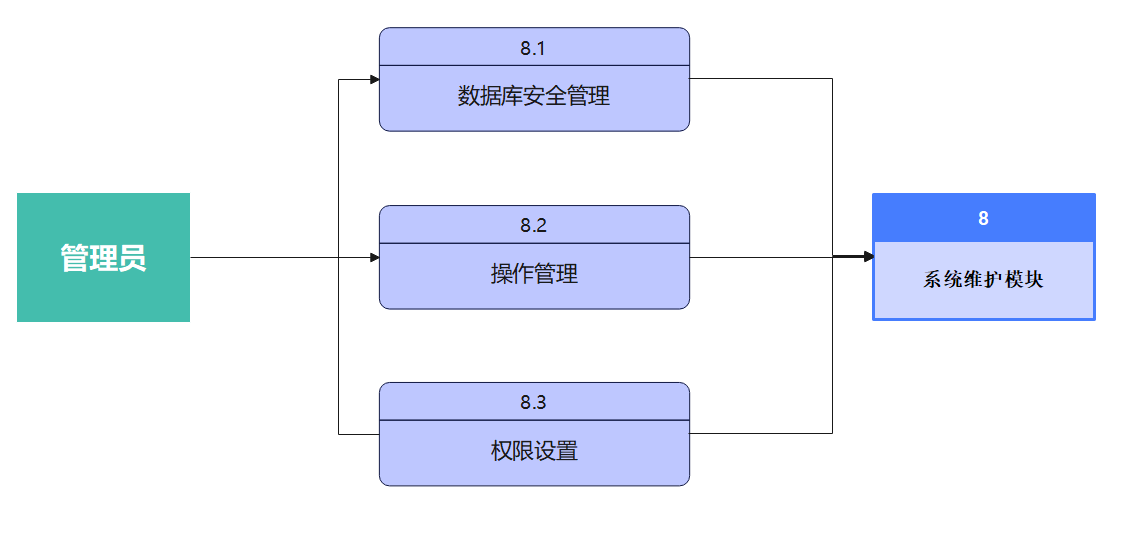


图 11系统维护

# 4 概念结构设计

## 4.1 涉及的实体

|  |
| --- |
| 用户登录信息实体 |
| 员工档案信息实体 |
| 岗位信息实体 |
| 考勤状况实体 |
| 部门信息实体 |
| 薪资状况实体 |
| 培训信息实体 |

表 1 涉及的实体表

## 4.2 实体之间的关系

1. 用户登录信息实体：假设为一个用户可以拥有一个登录信息，而一个登录信息只能关联一个用户。这是一个1对1的关系，其中一个用户对应一个登录信息，一个登录信息对应一个用户。
2. 员工档案信息实体：假设一个员工档案信息只能对应一个员工，而一个员工可以有一个档案信息。这是一个1对1的关系，其中一个员工对应一个档案信息，一个档案信息对应一个员工。
3. 岗位信息实体：假设一个岗位可以被多个员工所担任，但一个员工只能担任一个岗位。这是一个1对n的关系，其中一个岗位对应多个员工，一个员工对应一个岗位。
4. 考勤状况实体：假设一个员工只有1个考勤记录，而每个考勤记录只能对应一个员工。这是一个1对1的关系，其中一个员工对应1个考勤记录，一个考勤记录对应一个员工。
5. 部门信息实体：假设一个部门可以有多个员工，而一个员工只能属于一个部门。这是一个1对n的关系，其中一个部门对应多个员工，一个员工对应一个部门。
6. 薪资状况实体：假设一个员工可以有多个薪资记录，而每个薪资记录只能对应一个员工。这是一个1对n的关系，其中一个员工对应多个薪资记录，一个薪资记录对应一个员工。
7. 培训信息实体：假设一个员工可以参加多个培训活动，而一个培训活动可以有多个员工参加。这是一个多对多的关系，其中一个员工对应多个培训活动，一个培训活动对应多个员工。

## 4.3 实体应该具有的属性

1. 用户登录信息实体：
   * 用户ID
   * 用户名
   * 密码
   * 登录时间
   * 权限级别（管理员/普通员工）
2. 员工档案信息实体：
   * 员工ID
   * 姓名
   * 性别
   * 出生日期
   * 联系电话
   * 地址
   * 邮箱
   * 入职日期
3. 岗位信息实体：
   * 岗位ID
   * 岗位名称
   * 职责描述
4. 考勤状况实体：
   * 考勤记录ID
   * 日期
   * 当日上班时间
   * 当日下班时间
   * 当日是否缺勤
   * 当日缺勤类型（正常出勤/缺勤）
   * 以往缺勤总次数
5. 部门信息实体：
   * 部门ID
   * 部门名称
   * 部门负责人
   * 部门人数
6. 薪资状况实体：
   * 薪资记录ID
   * 薪资金额
   * 薪资类别（绩效/奖金/普通工资）
   * 发放日期
7. 培训信息实体：
   * 培训ID
   * 培训名称
   * 培训描述
   * 培训开始日期
   * 培训结束日期
   * 培训总人数

## 4.4 ER图绘制

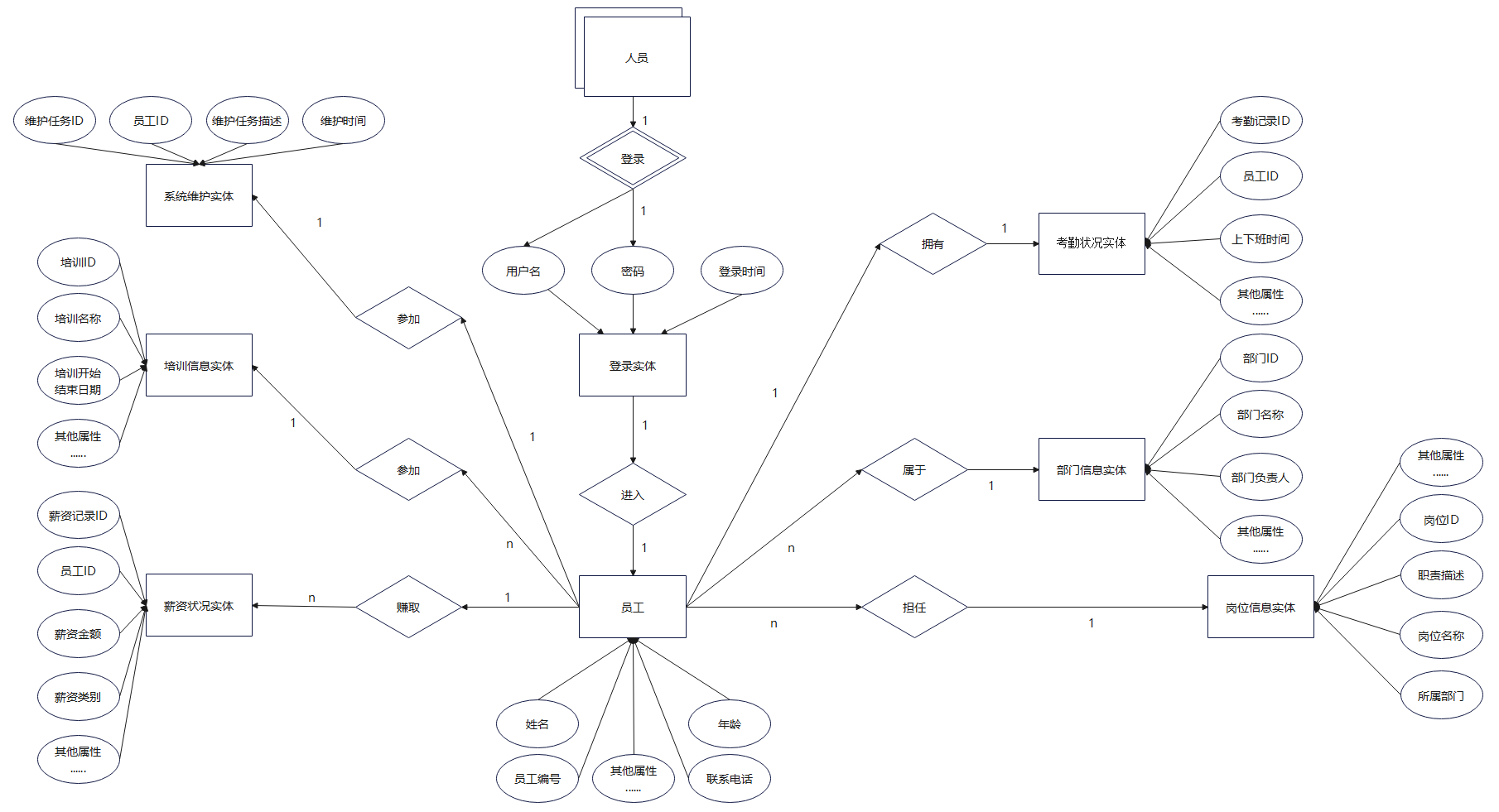


图 12 ER绘制

# 5 逻辑结构设计与物理结构设计

## 5.1 关系模型设计

**根据概念结构设计，同时依据以下三个原则：**

1. 一个m：n联系转换为一个关系模式。与该联系相连的各实体的码以及联系本身的属性均转换为关系的属性，而关系的码为各实体码的组合。
2. 一个1：n联系可以转换为一个独立的关系模式，也可以与n端对应的关系模式合并。如果转换为一个独立的关系模式，则与该联系相连的各实体的码以及联系本身的属性均转换为关系的属性，而关系的码为n端实体的码。
3. 一个1：1联系可以转换为一个独立的关系模式，也可以与任意一端对应的关系模式合并。

**所以对于ER图中1:1和1:n的关系模式，均采用合并的方式处理，对于多对多的关系模式，转化为一个新的关系模式。**

**因此开辟一个新的关系模式参加培训实体，对应员工参加培训的联系。据此可以设计关系模式如下所示：**

|  |  |
| --- | --- |
| **关系模式名** | **所含字段** |
| **用户登录信息(UserLoginInfo)** | **用户ID (UserID) [主键]**  **用户名 (Username)**  **密码 (Password)**  **上次登录时间 (LoginTime)**  **权限级别 (PermissionLevel)**  **员工ID (EmployeeID) [外键]** |
| **员工档案信息(EmployeeProfile)** | **员工ID (EmployeeID) [主键]**  **姓名 (Name)**  **性别 (Gender)**  **出生日期 (Birthdate)**  **联系电话 (ContactNumber)**  **地址 (Address)**  **邮箱 (Email)**  **入职日期 (HireDate)**  **所属岗位ID (PositionID) [外键]**  **所属部门ID (DepartmentID) [外键]** |
| **岗位信息(JobPosition)** | **岗位ID (PositionID) [主键]**  **岗位名称 (PositionName)**  **职责描述 (Responsibilities)** |
| **考勤状况(AttendanceStatus)** | **考勤记录ID (AttendanceID) [主键]**  **员工ID (EmployeeID) [外键]**  **当日上班时间 (StartTime)**  **当日下班时间 (EndTime)**  **当日是否缺勤 (Absent)**  **当日缺勤类型 (AbsentType)**  **以往缺勤总次数 (TotalAbsentCount)** |
| **部门信息(Department)** | **部门ID (DepartmentID) [主键]**  **部门名称 (DepartmentName)**  **部门负责人 (DepartmentManager)**  **部门人数 (NumberOfEmployees)** |
| **薪资状况(SalaryStatus)** | **薪资记录ID (SalaryID) [主键]**  **员工ID (EmployeeID) [外键]**  **薪资金额 (SalaryAmount)**  **薪资类别 (SalaryCategory)**  **发放日期 (PaymentDate)** |
| **培训信息(TrainingInfo)** | **培训ID (TrainingID) [主键]**  **培训名称 (TrainingName)**  **培训描述 (TrainingDescription)**  **培训开始日期 (StartDate)**  **培训结束日期 (EndDate)**  **培训总人数 (TotalParticipants)** |
| **参加培训关系(EmployeeTraining)** | **员工ID (EmployeeID) [主键]**  **培训ID (TrainingID) [主键]**  **是否完成培训(DoneTraining)** |

表 2 关系模式设计

## 5.2 关系模式规范化

对于以上关系模式而言，满足第一范式：表中每一列的属性都不可再分。同时满足第二范式：表中每一列的属性都不可再分，且非主属性完全依赖于主属性，同时满足第三范式：表中每一列的属性都不可再分，且非主属性完全依赖于主属性，且每个非主属性都不传递函数依赖于主属性。因此以上关系模式满足3NF。同时由于本关系模式中不存在多值依赖，所以可以可以基于上述关系模式建表进行后续的操作。

## 5.3 外模式(视图)设计

外模式设计（也称为用户界面设计或前端设计）是数据库设计中的一个重要环节，它涉及用户和数据库系统之间的交互界面。外模式设计可以提高数据安全：通过外模式设计，可以对用户进行权限控制和数据访问限制，确保只有经过授权的用户可以访问特定的数据。这有助于保护数据库中的敏感信息和防止未授权访问。同时也可以提升数据完整性：外模式设计可以在用户界面上实施数据验证和限制，确保用户输入的数据符合预期的格式和要求，从而提高数据的完整性和准确性。除此之外，外模式设计可以确保用户在进行数据操作时遵循一致的数据格式和规范，从而减少数据的冗余和不一致性，也可以针对用户的具体需求提供定制化的功能和界面，满足不同用户群体的需求。

员工视图：

|  |
| --- |
| 个人ID |
| 姓名 |
| 性别 |
| 出生日期 |
| 电话号码 |
| 住址 |
| 个人邮箱 |
| 雇佣日期 |
| 所属岗位 |
| 所属部门 |
| 薪资状况 |

表 3 员工视图所含字段

该视图为员工信息的视图，当员工登陆个人账号后，就可以查看自己的个人信息。

## 5.4 权限控制

为了区别员工与管理员，并对权限进行控制，我创建了两个用户来连接本数据库。在用户登陆后，我会根据用户的权限级别（在用户信息中）来判断，在进行数据库连接时应该采用何种权限级别的连接，从而就实现了不同用户的权限控制，保证了数据库的安全性。

|  |
| --- |
| -- 进行用户权限管理  CREATE USER 'client'@'localhost' IDENTIFIED BY '123456';  GRANT USAGE ON `personnel\_management`.\* TO 'client'@'localhost';  GRANT SELECT ON `personnel\_management`.`employee` TO 'client'@'localhost';  GRANT SELECT ON `personnel\_management`.`salary` TO 'client'@'localhost';  GRANT SELECT ON `personnel\_management`.`user\_info` TO 'client'@'localhost';  GRANT SELECT ON `personnel\_management`.`employee\_view` TO 'client'@'localhost';  GRANT INSERT ON `personnel\_management`.`attendance` TO 'client'@'localhost'; |

表 4 权限控制相关代码

## 5.5 数据完整性的考虑

数据完整性是指数据库中存储的数据必须满足预定义的规则和约束，确保数据的准确性、一致性和有效性。它是数据库设计和管理的一个重要方面，通过定义规则和约束来保证数据的正确性，防止数据损坏、冲突或错误。

数据完整性可以分为以下几个方面：

1. 实体完整性：确保每个表中的每一行都能够唯一地标识一个实体，通常通过定义主键来实现。
2. 参照完整性：确保表与表之间的关系是有效的，通常通过外键来实现。外键保证了关联表之间的数据一致性，确保引用表中的值在被关联表中存在。
3. 用户定义完整性：根据业务需求和应用规则，定义特定的数据完整性约束，保证特定的数据逻辑正确性。

对于系统管理员在进行数据的修改(增删改时)，我在前端页面对用户需要修改的数据进行了第一层的校验，如果用户需要修改的数据违反了定义的实体完整性，参照完整性，以及自行定义的完整性，就会对用户进行提示，并拒绝该操作。

对于越过前端页面，直接对后端服务器发送请求参数来进行内容修改的操作，我在后端也对参数进行了第二次验证，如果不满足完整性定义的要求，那么就拒绝该操作并返回提示信息。

而对于删除员工的操作，我在此系统中假定了如果要删除员工对应了该员工的离职，那么相应部门与相应岗位的内容以及员工的薪资等内容也应该进行相应的变化。因此为了保证数据的完整性，我定义了级联操作CASCADE，在对employee\_id具有外键约束的那些内容，如果删除该员工，那么对应的如该员工所在部门表的员工也会删除该记录。例如添加如下的约束：

|  |
| --- |
| ADD CONSTRAINT fk\_depart\_empl FOREIGN KEY (employee\_id) REFERENCES employee(employee\_id) ON DELETE CASCADE; |

而对于考勤信息，由于具有员工的上下班打卡功能，那么就需要标识员工是否上班迟到或者下班早退，因此通过创建触发器实现，如下:

|  |
| --- |
| CREATE TRIGGER set\_late\_type\_trigger BEFORE INSERT ON attendance  FOR EACH ROW  BEGIN  SET NEW.late\_type = CASE  WHEN NEW.log\_type = 0 AND TIME(NEW.log\_time) > TIME('09:00:00') THEN 1  WHEN NEW.log\_type = 1 AND TIME(NEW.log\_time) < TIME('18:00:00') THEN 2  ELSE 0  END;  END |

对于员工培训信息，由于需要标识该培训是否完成，因此定义了一个触发器实现根据该培训的截止时间来判断培训是否结束并自动修改表中对应字段，以及其它一些保证数据完整性的触发器及操作。

# 6 代码编写与具体实现

## 6.1 实现环境

系统环境：

windows10

图 13 系统环境

编程环境：



图 14 编程环境

Mysql版本：

MYSQL5.7.19

图 15 Mysql版本

使用的Web服务框架：

Tomcat 8.0.50

图 16 Web服务框架

## 6.2 主要的业务流程

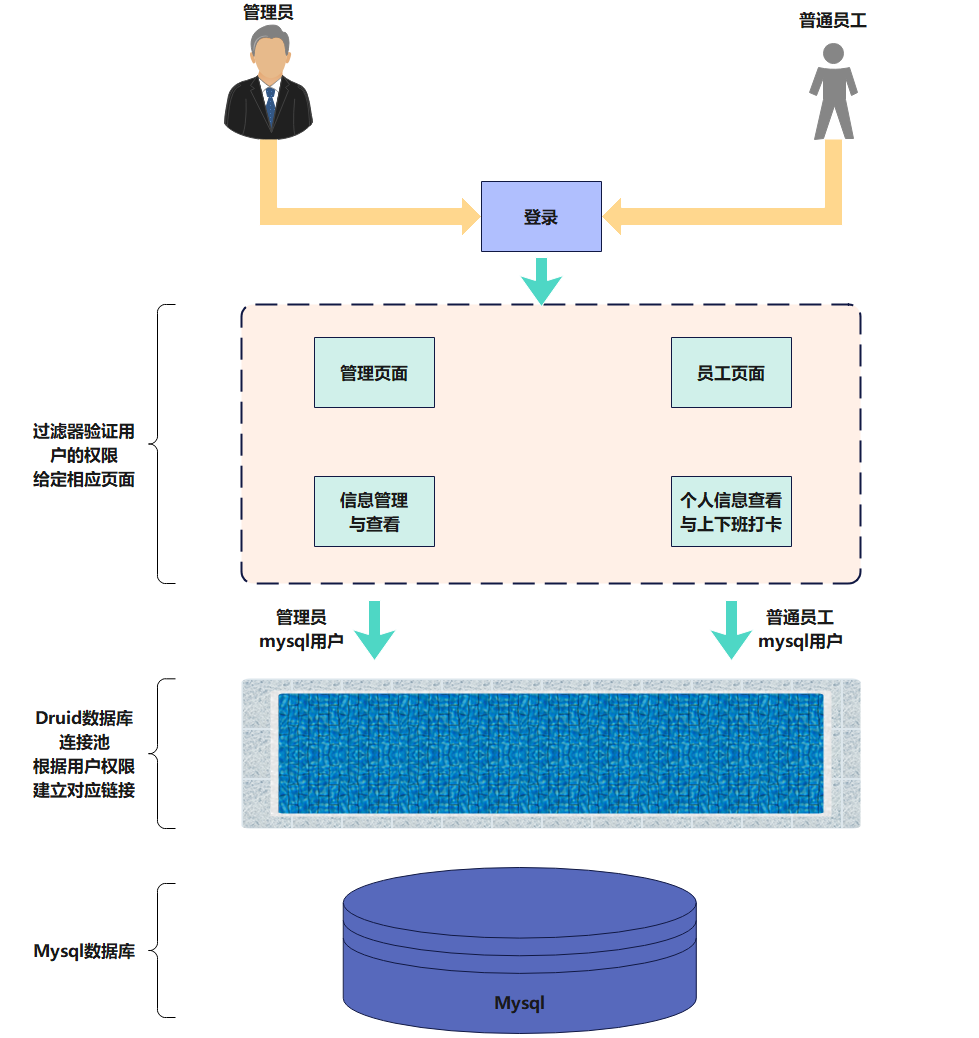


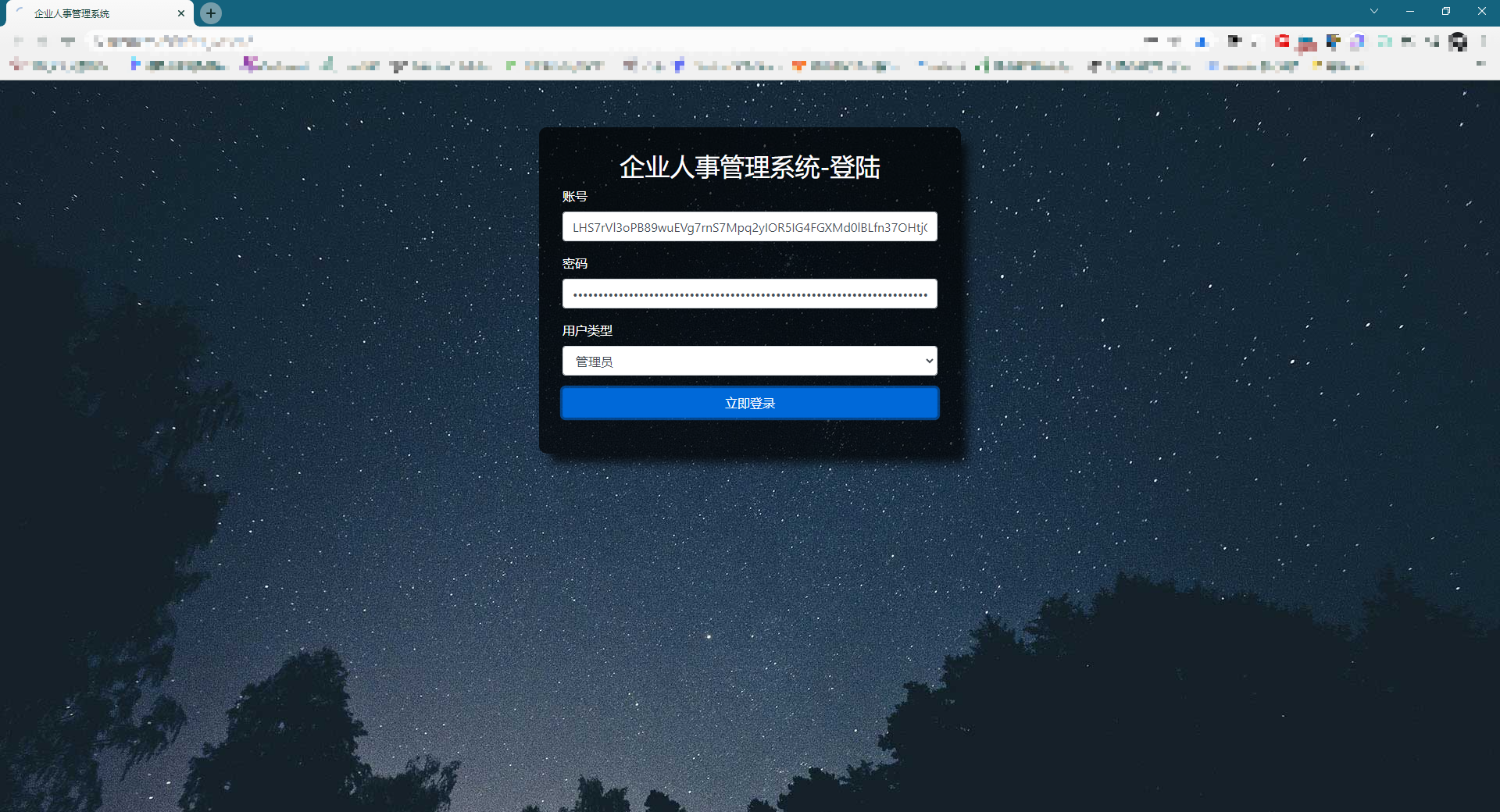
图 17 主要业务流程

如上图所示，在使用本系统之前，用户需要进行登录，本系统设置了两类用户：管理员与普通员工。用户在登陆后可以进入相应的界面，管理员可以进行所有的管理操作与查看操作，而普通员工的Mysql用户仅仅赋予了对部份表如考勤表的操作，以及查看个人信息的功能，采用视图查看。

## 6.3 安全性的考虑

考量到一定程度的安全性，本系统做了以下几点工作：

1. 在用户进行登陆时，对表单进行了加密处理，加密方式采用RSA非对称加密，通过服务器向客户机申请公钥，并将公钥保存，而后对表单内容进行加密处理，处理后再将表单的内容传给服务器，服务器再通过私钥进行解密操作，得到表单内容，进行验证。在进行登录时加密后的内容如下：



1. 为了防止用户绕过登录，以及普通员工访问到系统管理员才有权访问的页面，本系统添加了相应的过滤器进行处理，使得普通员工在登陆后也不能访问管理员需要处理的内容，同时所有用户也不能绕过登录。
2. 为了防止SQL注入的风险，由于SQL注入它允许攻击者通过在输入字段中插入恶意的SQL代码来操纵数据库，获取敏感信息或对数据库进行破坏。SQL注入攻击通常发生在未对用户输入进行充分验证和转义的情况下，因此再执行SQL语句时，本系统采用prepareStatement执行SQL语句。使用prepareStatement预编译，然后再根据通配符进行数据填值，可以提高执行效率的同时，而且保证排除SQL注入漏洞。
3. 同时为了防止数据库的误删除与其它不可抗因素导致的数据丢失，我对数据库进行了备份操作，以保证数据的安全性。



## 6.4 后端代码的性能考量

**MVC架构模式**

本次课程设计后端代码基于MVC设计模式，MVC作为一种软件架构模式，用于组织和分离应用程序的不同组成部分，包括数据、用户界面和业务逻辑。MVC代表Model-View-Controller，每个部分都有其特定的职责和功能：

1. Model（模型）：模型负责管理应用程序的数据和业务逻辑。它表示应用程序的核心部分，与数据的存取和处理相关。模型通常包含数据结构、数据库连接、数据操作等功能。

2. View（视图）：视图负责用户界面的展示和用户交互。它从模型中获取数据并将其显示给用户，同时监听用户的操作并将其反馈给控制器。视图是与平台无关的，可以有多种形式，如Web页面、图形界面、命令行界面等。

3. Controller（控制器）：控制器是模型和视图之间的中间人，负责处理用户的请求并根据请求更新模型和视图。它接收用户输入，调用模型处理数据，然后选择相应的视图来显示结果。控制器是应用程序的逻辑部分，协调模型和视图之间的交互。

MVC的优点包括：

1. 松耦合：MVC模式将应用程序分为不同的组件，使它们之间的依赖关系降低，从而提高了系统的可维护性和可扩展性。

2. 分工明确：MVC模式明确了每个组件的职责，使开发人员可以更容易地理解和修改代码。

3. 可复用性：MVC模式鼓励代码的重用，例如可以多个视图共享同一个模型。

4. 可测试性：由于MVC将应用程序分为不同的组件，可以更容易地编写单元测试，以验证每个组件的功能是否正确。

MVC模式广泛应用于各种软件开发项目中，特别是在Web开发中，使用该模式有助于构建结构清晰、可维护的应用程序。

**模板设计模式**

本系统开发大都遵循模板设计模式，该模式是一种行为设计模式，它定义了一个算法的框架，并允许子类在不改变算法结构的情况下重新定义算法中的某些步骤。

模板设计模式主要由两部分组成：

1. 抽象类（Abstract Class）：抽象类定义了算法的框架，其中包含一个或多个抽象方法，这些抽象方法是算法的关键步骤，由子类实现。抽象类还可以包含一些具体方法，这些方法在算法中是通用的，但也可以被子类覆盖。
2. 具体子类（Concrete Subclasses）：具体子类是继承抽象类并实现其中的抽象方法的类。每个具体子类可以根据需要重定义算法中的特定步骤，但不能改变算法的整体结构。

其优点可以实现代码复用：算法的框架由抽象类实现，各个子类只需要实现自己特定的步骤，从而实现代码的复用。同时灵活性较好，通过子类重写抽象方法，可以灵活地定制算法的某些步骤，而不影响整个算法的结构。也能提高可维护性，将算法的公共部分封装在抽象类中，减少了重复代码，使得代码更易于维护和修改。

**数据库连接池技术**

在数据库连接时，本系统使用了数据库连接池技术。由数据库连接池负责分配、管理和释放数据库连接，它允许应用程序重复使用一个现有的数据库连接，而不是再重新建立一个。

数据库连接是一种关键的有限的昂贵的资源，这一点在多用户的网页应用程序中体现得尤为突出。一个数据库连接对象均对应一个物理数据库连接，每次操作都打开一个物理连接，使用完都关闭连接，这样造成系统的性能低下。数据库连接池的解决方案是在应用程序启动时建立足够的数据库连接，并讲这些连接组成一个连接池(简单说：在一个“池”里放了好多半成品的数据库联接对象)，由应用程序动态地对池中的连接进行申请、使用和释放。对于多于连接池中连接数的并发请求，应该在请求队列中排队等待。并且应用程序可以根据池中连接的使用率，动态增加或减少池中的连接数。连接池技术尽可能多地重用了消耗内存地资源，大大节省了内存，提高了服务器地服务效率，能够支持更多的客户服务。通过使用连接池，将大大提高程序运行效率，同时，我们可以通过其自身的管理机制来监视数据库连接的数量、使用情况等。

## 6.5 前端页面使用的框架

为了方便前端页面的开发，本系统基于Layui框架+Jquery+Jsp页面开发前端页面。Layui作为一款轻量级的前端UI框架，专注于提供易用、简洁的界面元素和交互体验。它基于HTML、CSS、JavaScript构建，广泛用于Web开发，特别是适用于后台管理系统、企业级应用等项目。

Layui的主要特点包括：

1. 简洁易用：Layui提供了一系列简洁、美观的UI组件，使得开发者可以轻松构建出漂亮的界面。
2. 模块化设计：Layui采用模块化的设计思想，使得每个功能都是独立的模块，可以根据需求选择性加载，减少了代码冗余。
3. 响应式布局：Layui支持响应式布局，可以适应不同尺寸的设备，使得页面在PC和移动端都有良好的显示效果。
4. 丰富的组件：Layui提供了丰富的UI组件，包括按钮、表单、表格、导航、弹窗等，满足了大部分常见页面的需求。
5. 灵活定制：Layui提供了丰富的配置项和定制方式，使得开发者可以根据自己的需求对框架进行定制。
6. 多种扩展插件：Layui还提供了许多扩展插件，如日期选择器、分页、上传等，方便开发者快速集成功能。

Layui的官方网站提供了详细的文档和示例，方便开发者学习和使用。由于其简洁易用的特点，Layui在国内得到了广泛的应用和认可，成为了一款备受欢迎的前端UI框架。

jQuery是一个快速、简洁的JavaScript库，主要用于简化HTML文档的遍历、操作、事件处理以及动画等操作。它是一个开源的跨平台JavaScript库，由John Resig在2006年发布，广泛用于Web开发。

jQuery的主要特点包括：

1. 简化DOM操作：jQuery提供了简洁的语法和API，可以更方便地选取和操作HTML文档中的元素，比原生JavaScript更加简单和灵活。
2. 事件处理：jQuery可以轻松处理元素的事件，如点击、鼠标移入、键盘按下等，同时支持事件委托，提高了事件处理的效率。
3. AJAX支持：jQuery提供了简单易用的AJAX方法，方便发送HTTP请求和处理响应，实现无刷新的数据交互。
4. 动画效果：jQuery内置了丰富的动画效果，可以轻松实现淡入淡出、滑动、缩放等动画效果，提升用户体验。
5. 跨浏览器兼容性：jQuery封装了许多常用的跨浏览器处理方法，解决了不同浏览器之间的兼容性问题。
6. 插件扩展：jQuery支持插件扩展，有大量的第三方插件可供选择，方便开发者快速添加功能。

由于jQuery的简洁易用和跨浏览器兼容性，它在Web开发中得到了广泛的应用，成为了最受欢迎的JavaScript库之一。

## 6.6 测试工作

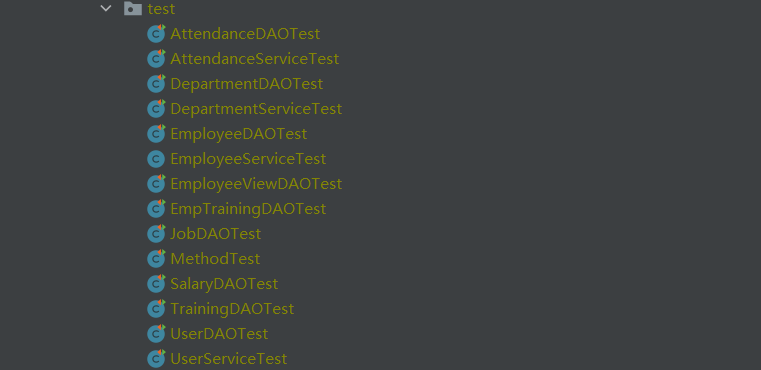
在此次开发过程中，为了提高开发的效率与质量，我进行了模块测试的工作，基于Junit框架完成。

Junit是一个用于Java编程语言的开源测试框架，用于编写和运行单元测试。它是测试驱动开发（Test-Driven Development，TDD）的关键工具之一，可以帮助开发人员编写自动化的单元测试，以确保代码的质量和正确性。

Junit的主要特点和作用包括：

1. 自动化测试：Junit允许开发人员编写自动化的测试用例，通过运行这些测试用例可以自动检测代码中的错误和缺陷。
2. 单元测试：Junit主要用于编写单元测试，即对代码中最小的可测试单元进行测试，通常是一个方法或函数。
3. 注解支持：Junit使用注解来标记测试方法，例如@Test用于标记测试方法，@Before和@After用于标记在测试方法之前和之后需要执行的方法。
4. 断言机制：Junit提供了一系列断言方法，用于验证测试结果是否符合预期，例如assertEquals()用于比较两个值是否相等，assertTrue()用于判断一个条件是否为真等。
5. 测试套件：Junit支持将多个测试类组织成测试套件，便于批量运行测试用例。
6. 异常测试：Junit可以很方便地测试代码中是否抛出了期望的异常。
7. 测试报告：Junit会生成测试报告，展示测试用例的运行结果和覆盖率等信息。

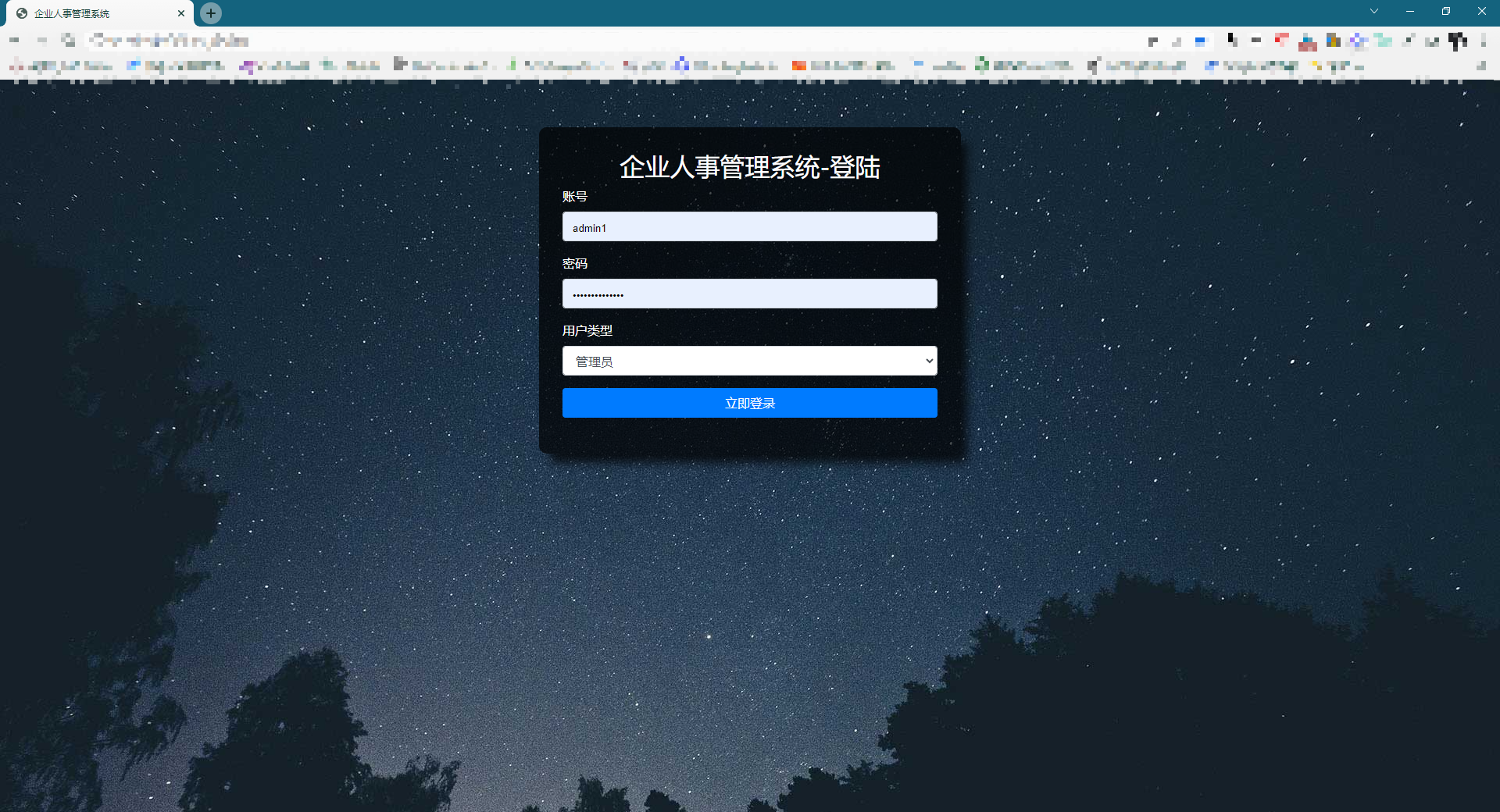
使用Junit进行单元测试可以大大提高代码的稳定性和可维护性，同时也可以帮助开发人员快速发现和修复代码中的问题。它是Java开发中非常常用的测试框架之一，被广泛应用于各种Java项目中。下图展示了此次课程设计的测试模块的部分类：



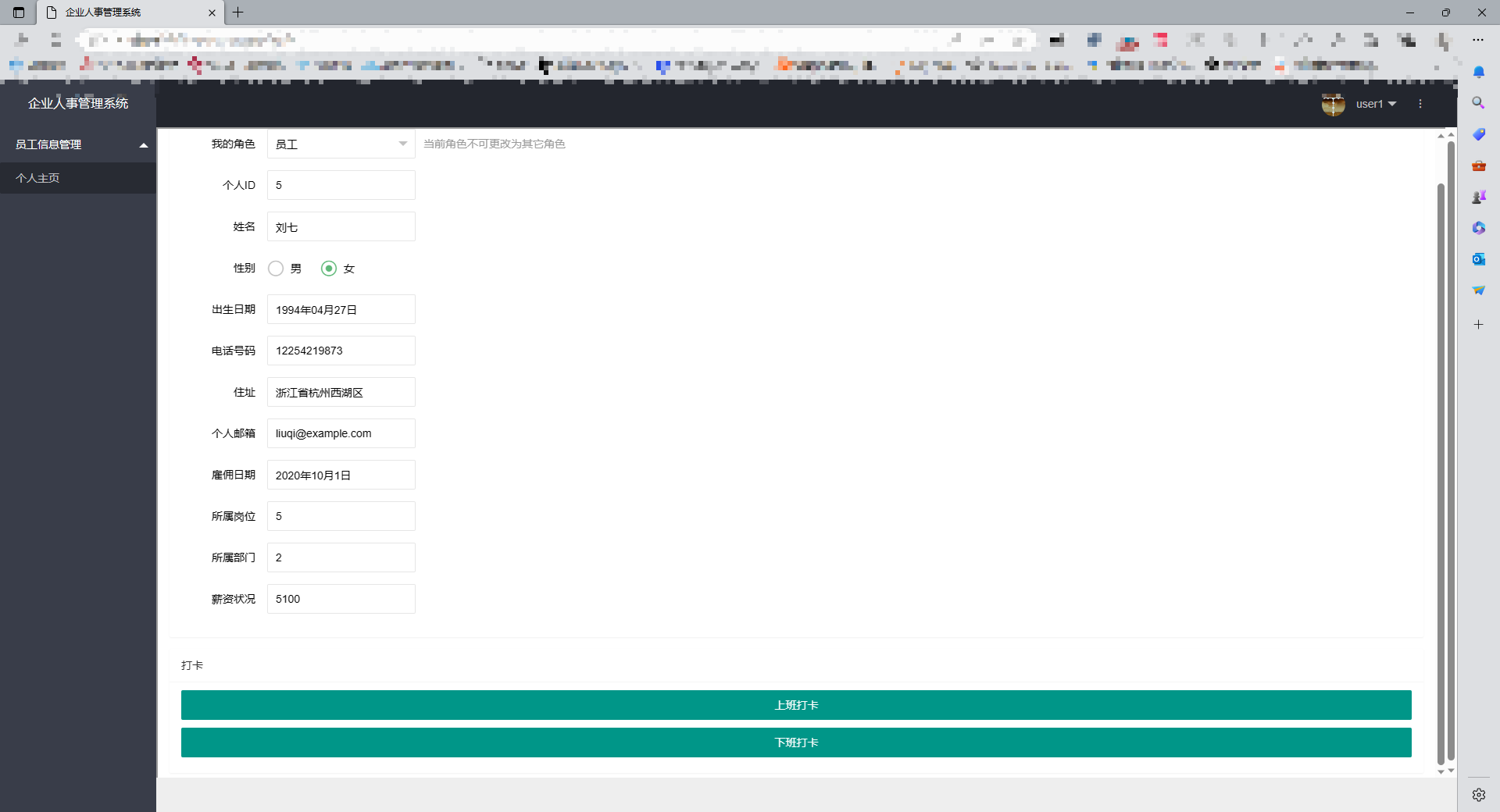
# 7 实际运行效果展示

## 7.1 前端页面模块展示

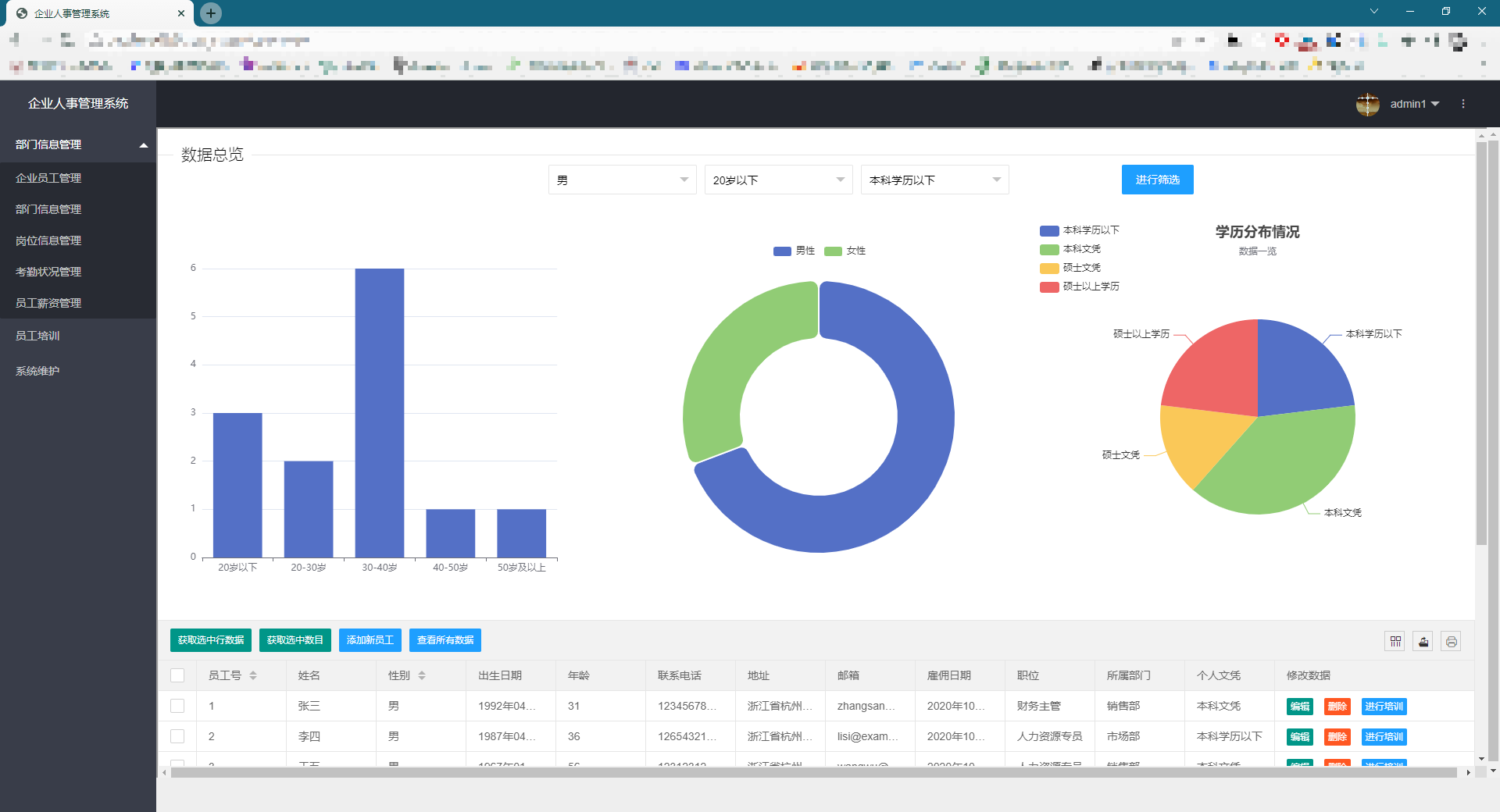
登录页面(可以选择是管理员或员工登录)：



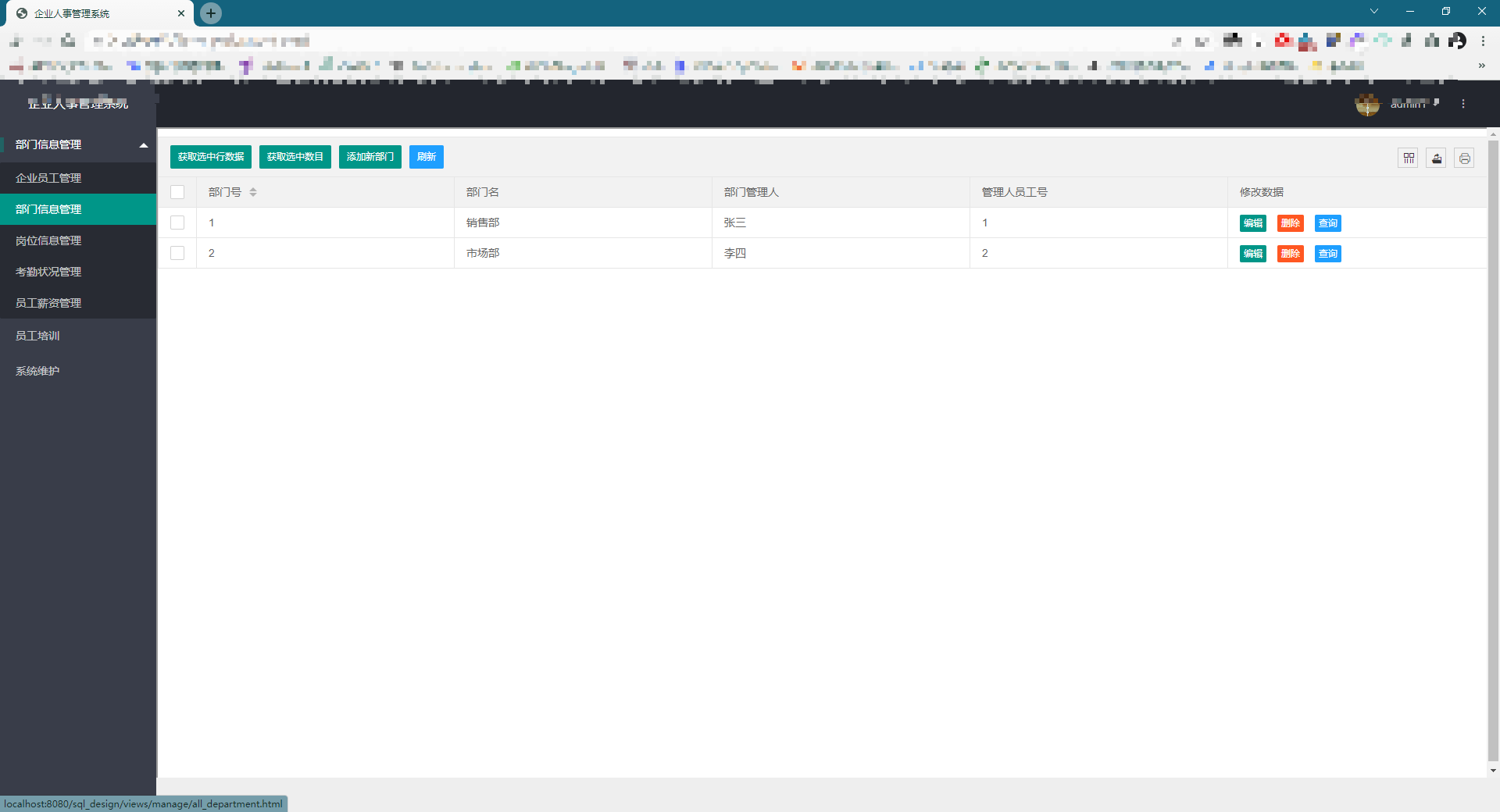
普通员工登录成功后的页面：



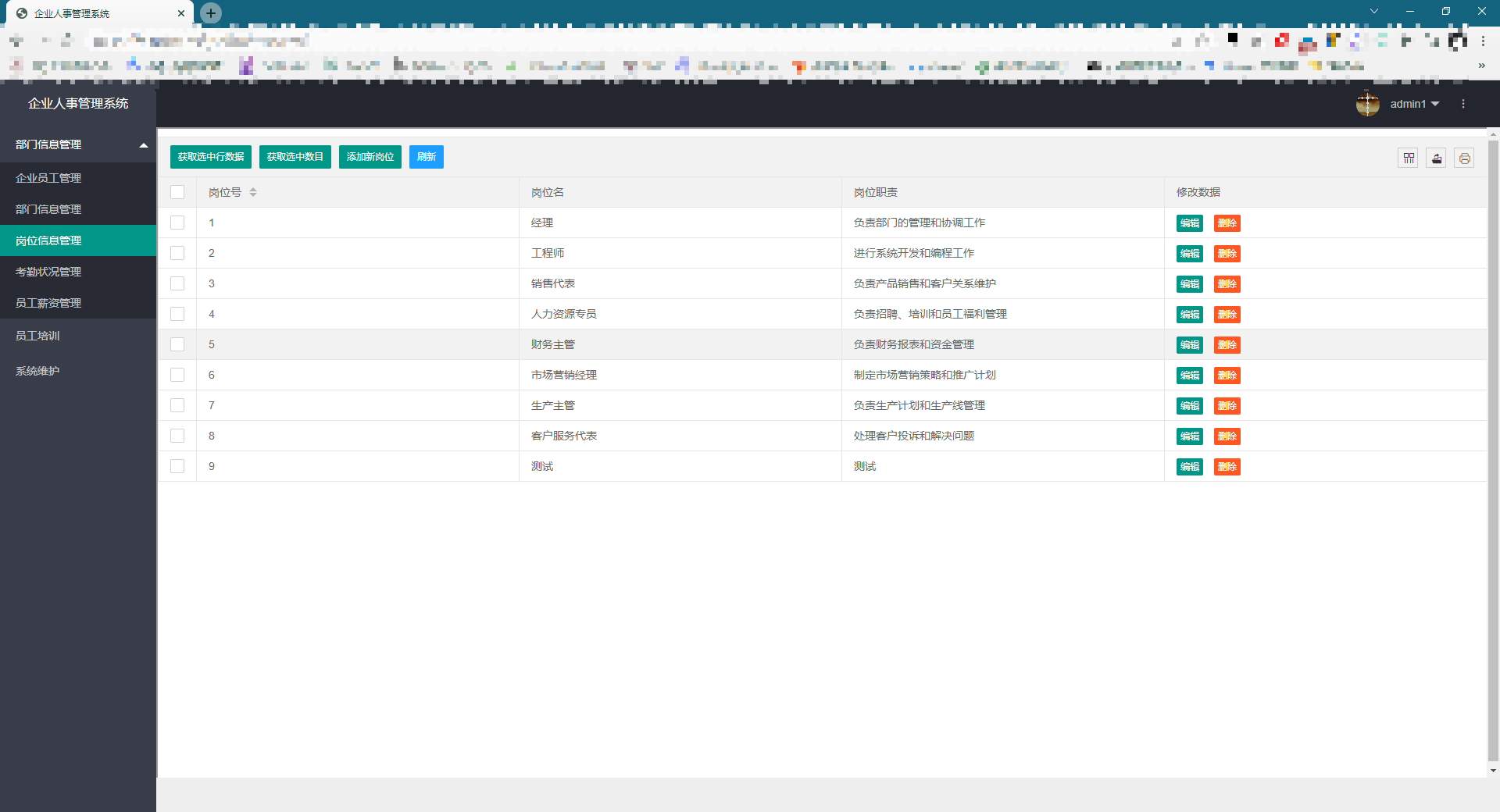
管理员登录成功后的页面：



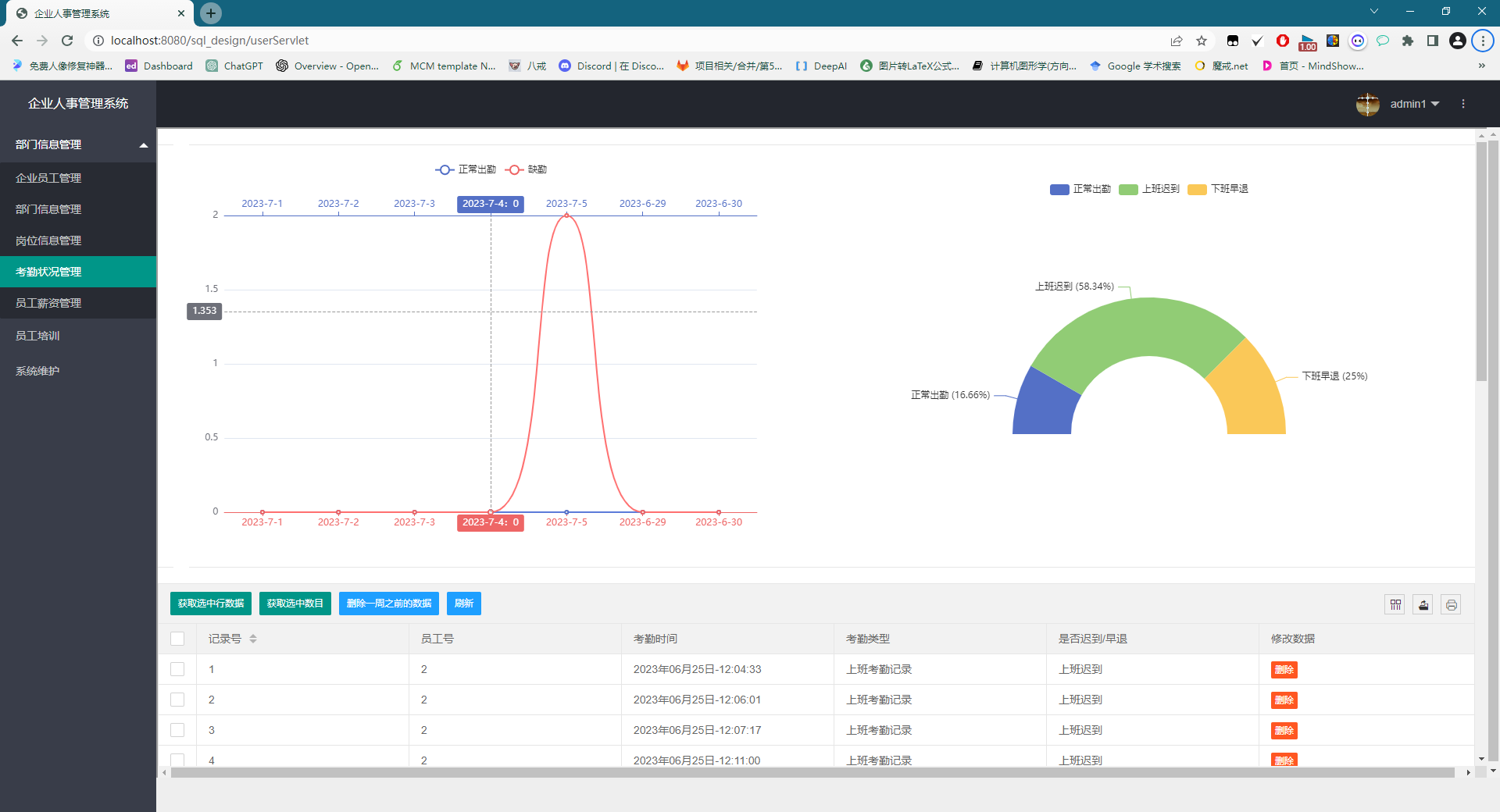
部门信息管理页面：



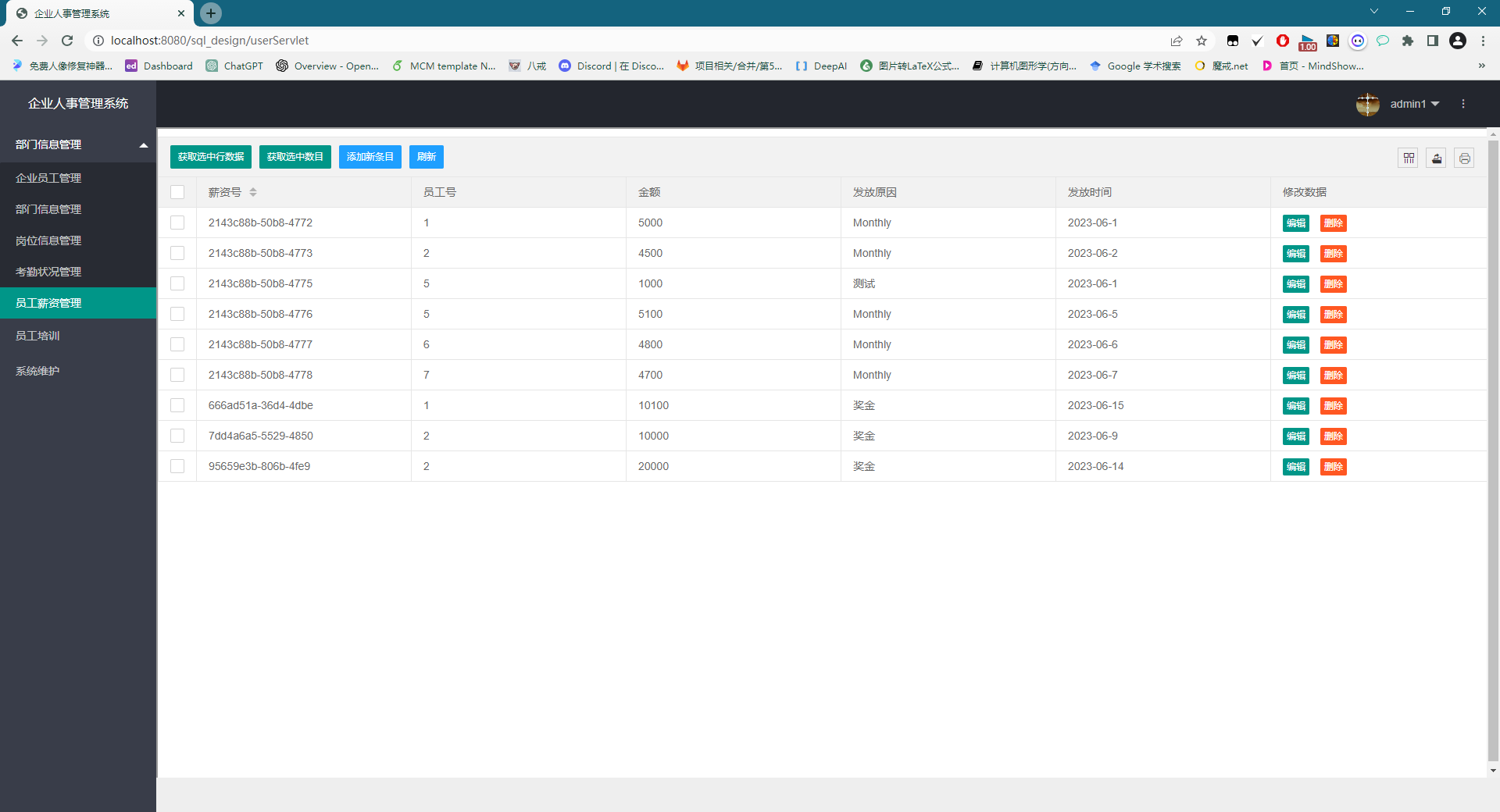
岗位信息管理页面：



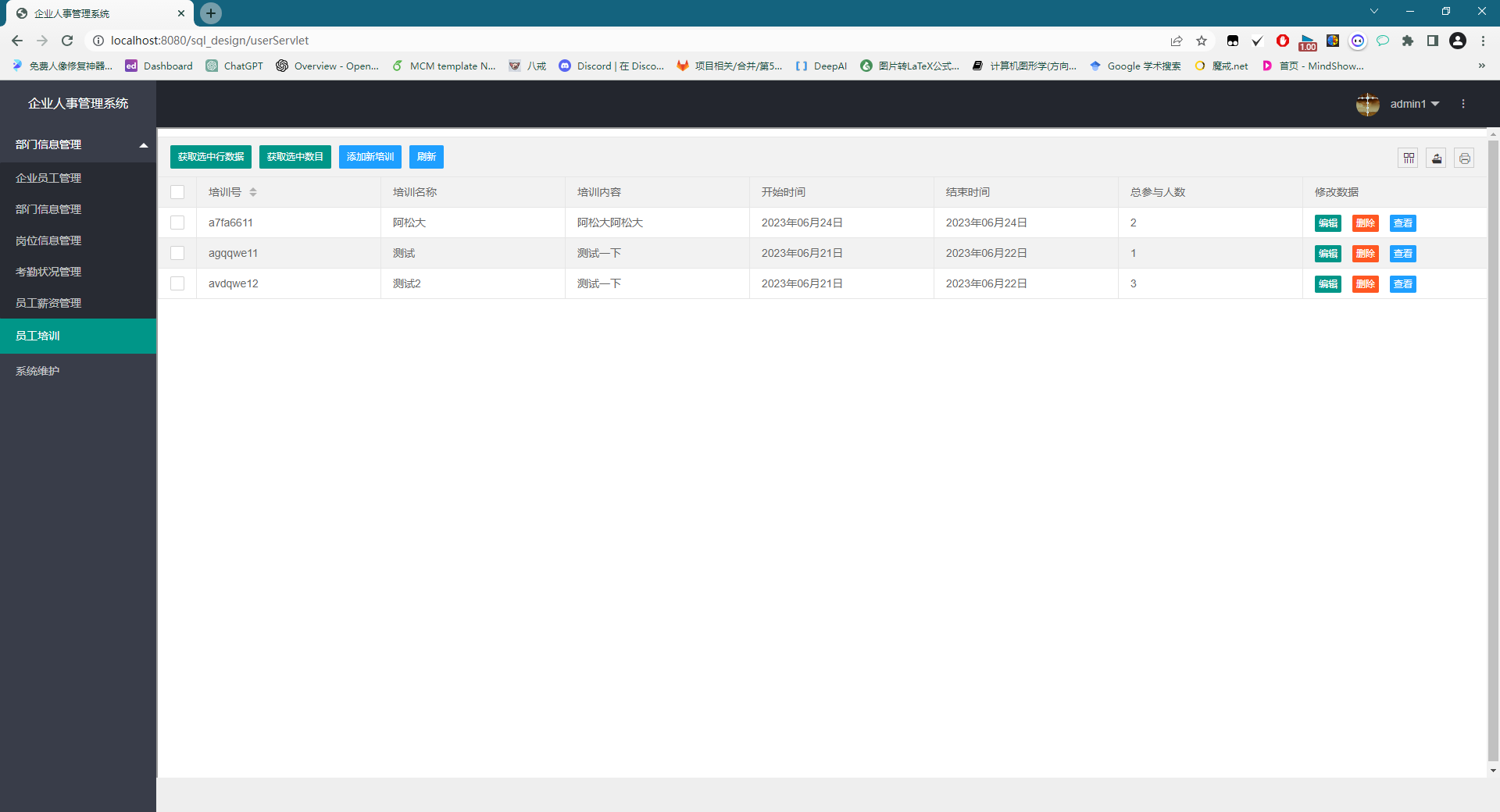
考勤状况管理页面：



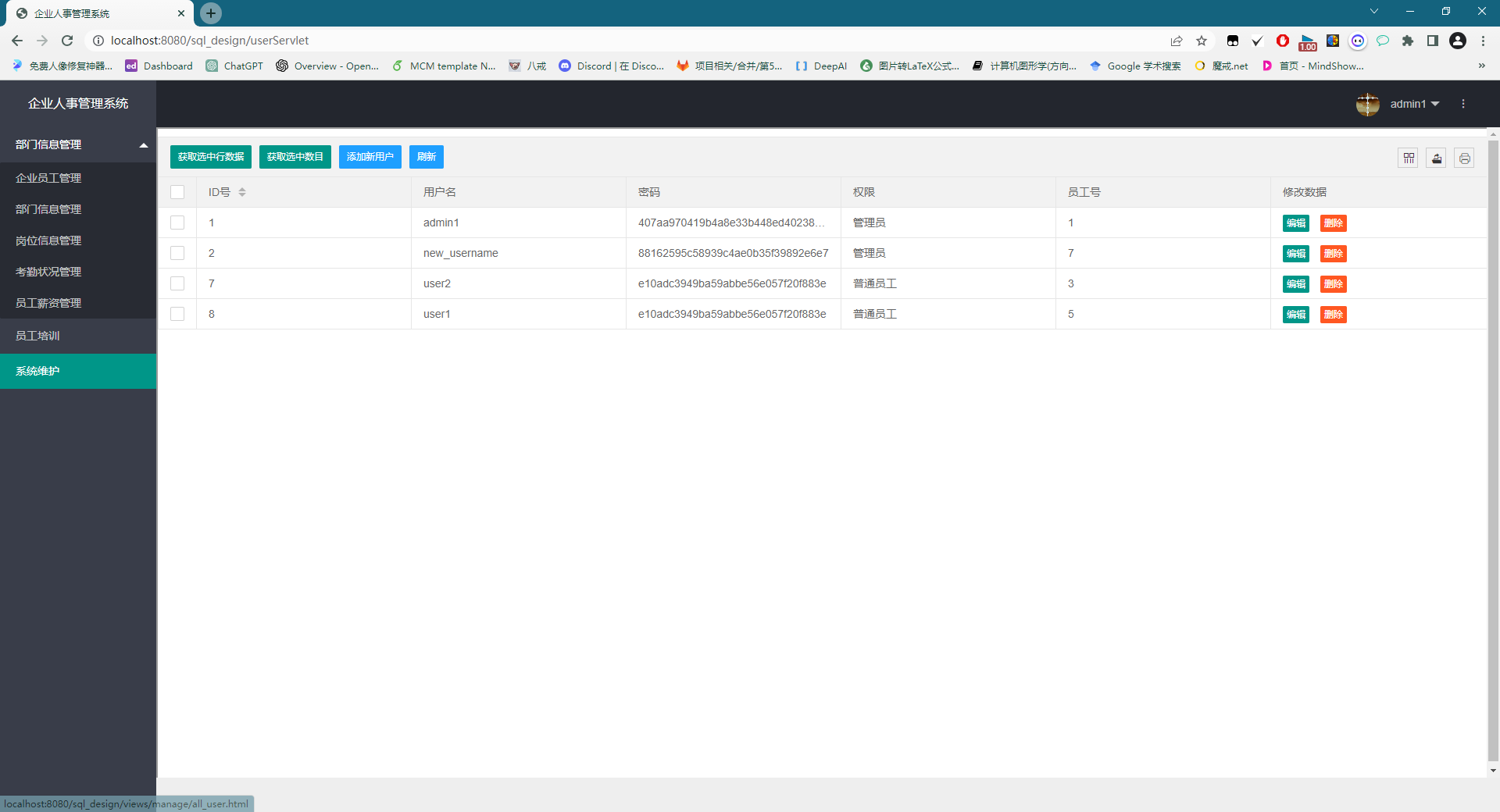
员工薪资管理界面：



员工培训页面：

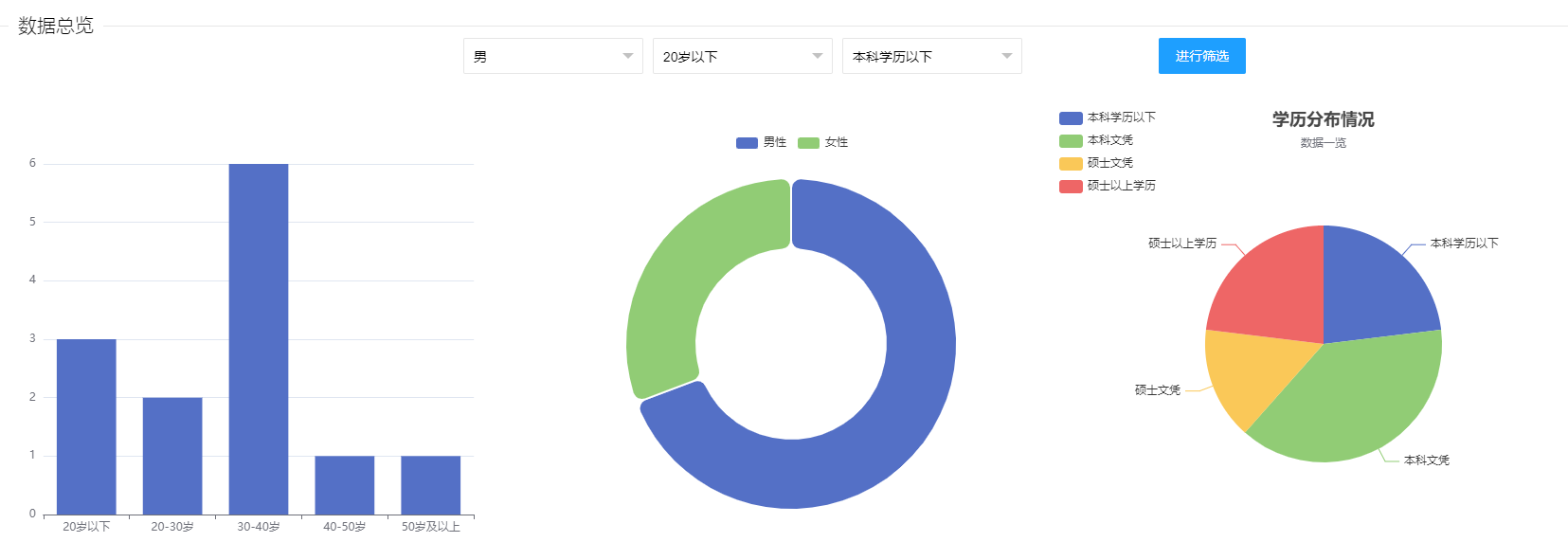


系统维护页面：



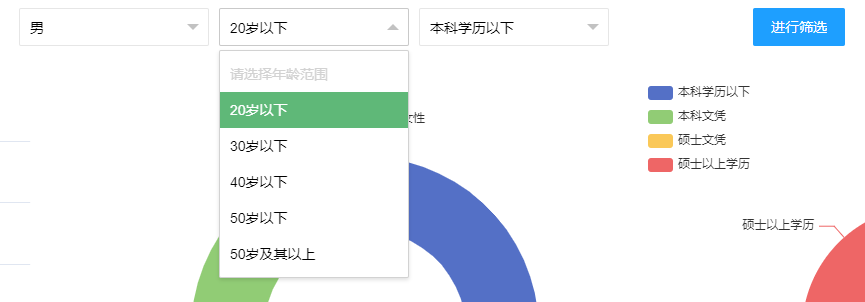
## 7.2 部分功能展示

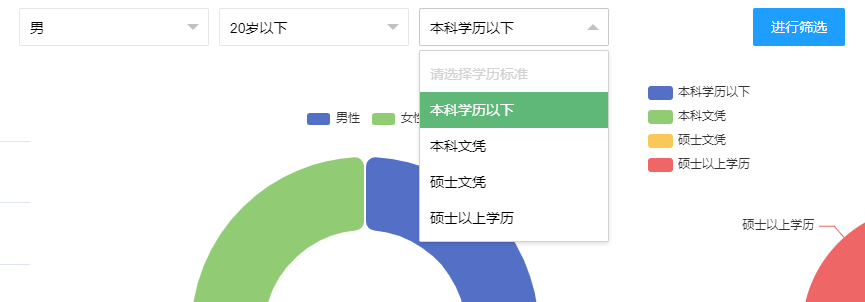
员工信息布局一览图：



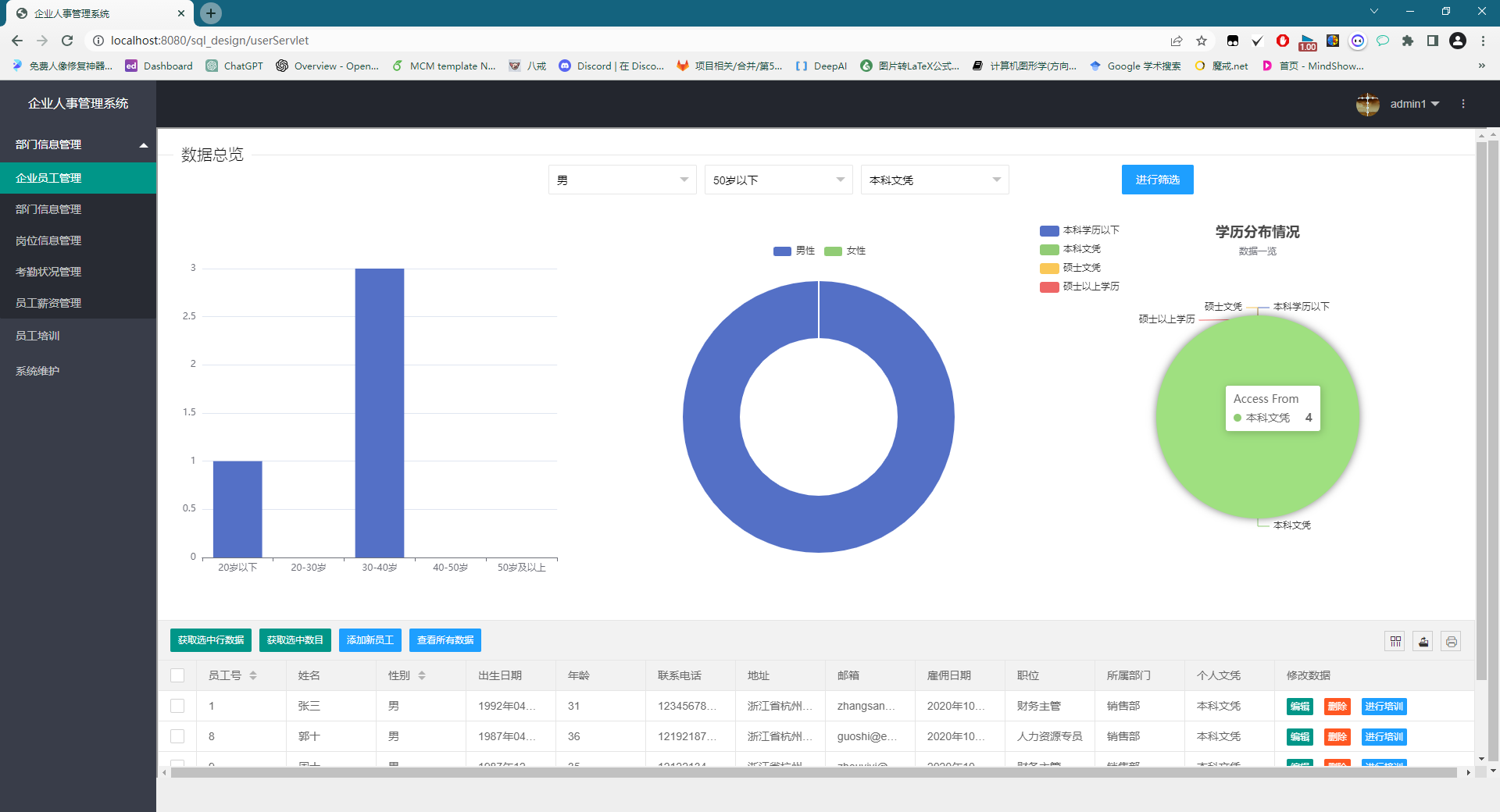
支持的筛选条件：



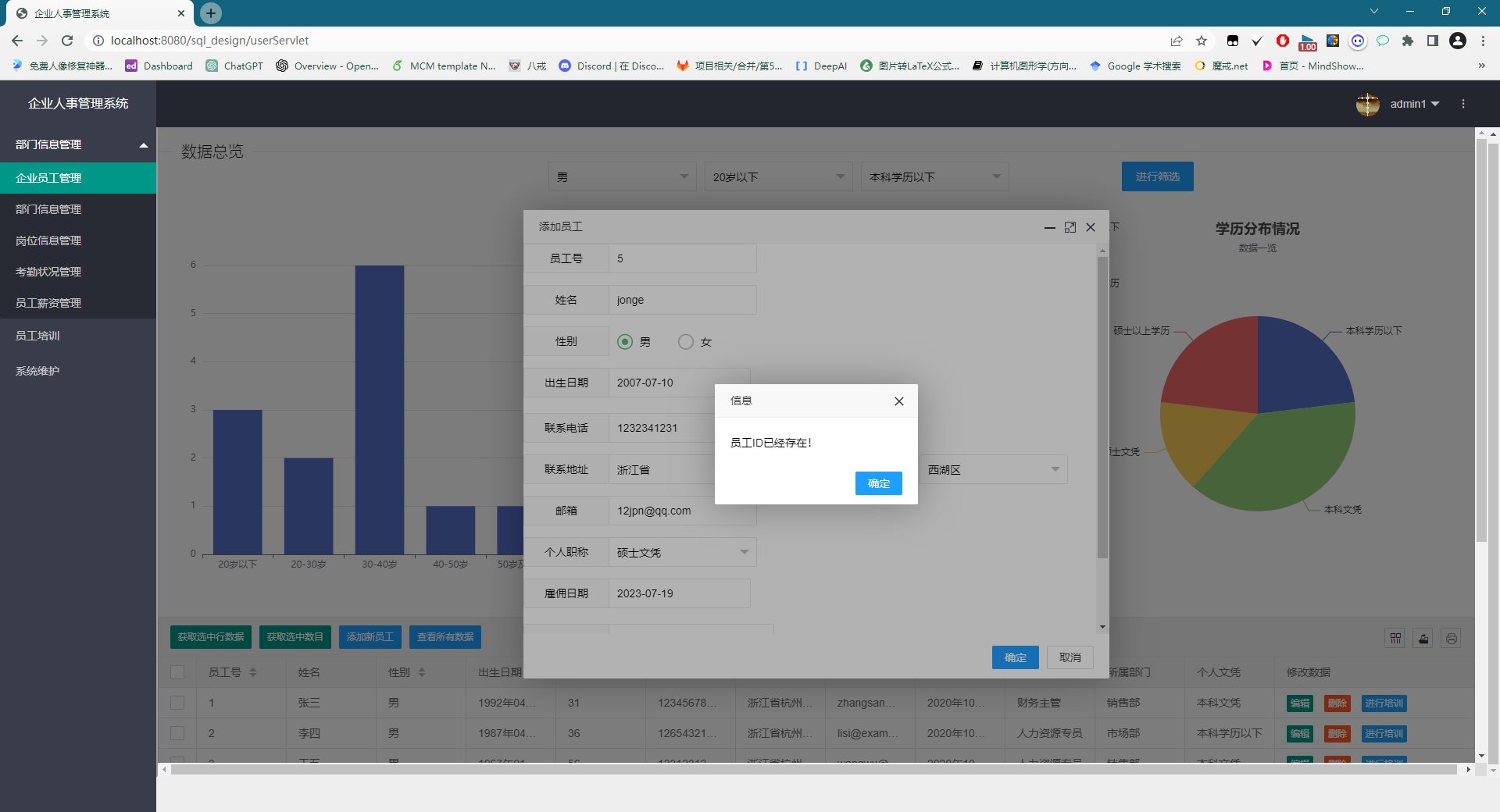




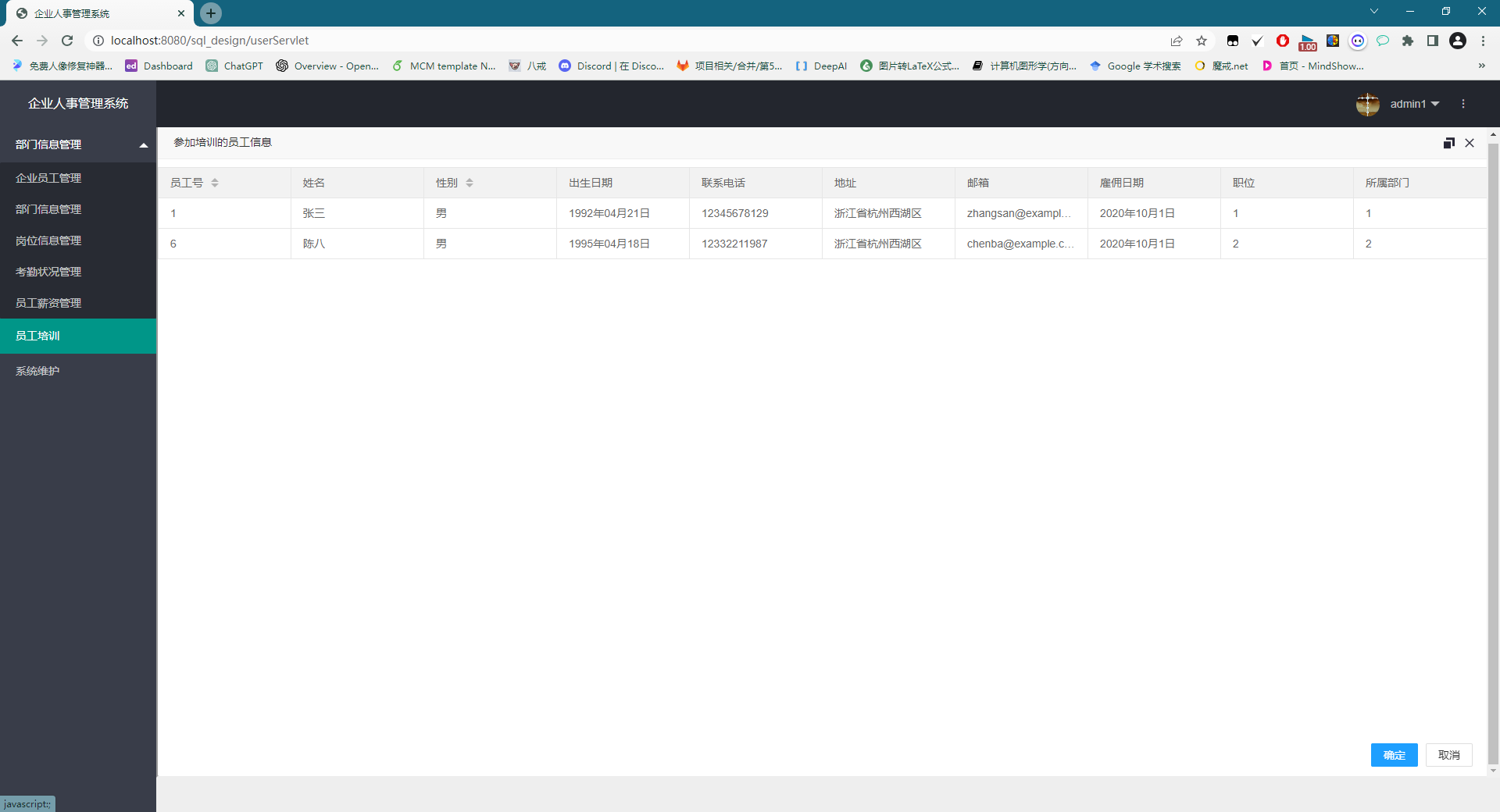
在企业员工管理首页面，可以进行员工信息的数据总览，并且可以根据需求从中筛选自己需要的类型，如选择男，50岁以下，本科学历：



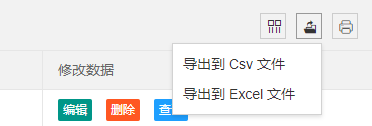
可以清晰地看出其年龄，性别，学历状况的分布。除此以外还可以对员工个人信息进行增删与修改，如果进行的操作违反了数据的完整性，或者格式出现错误就会进行相应的提示：



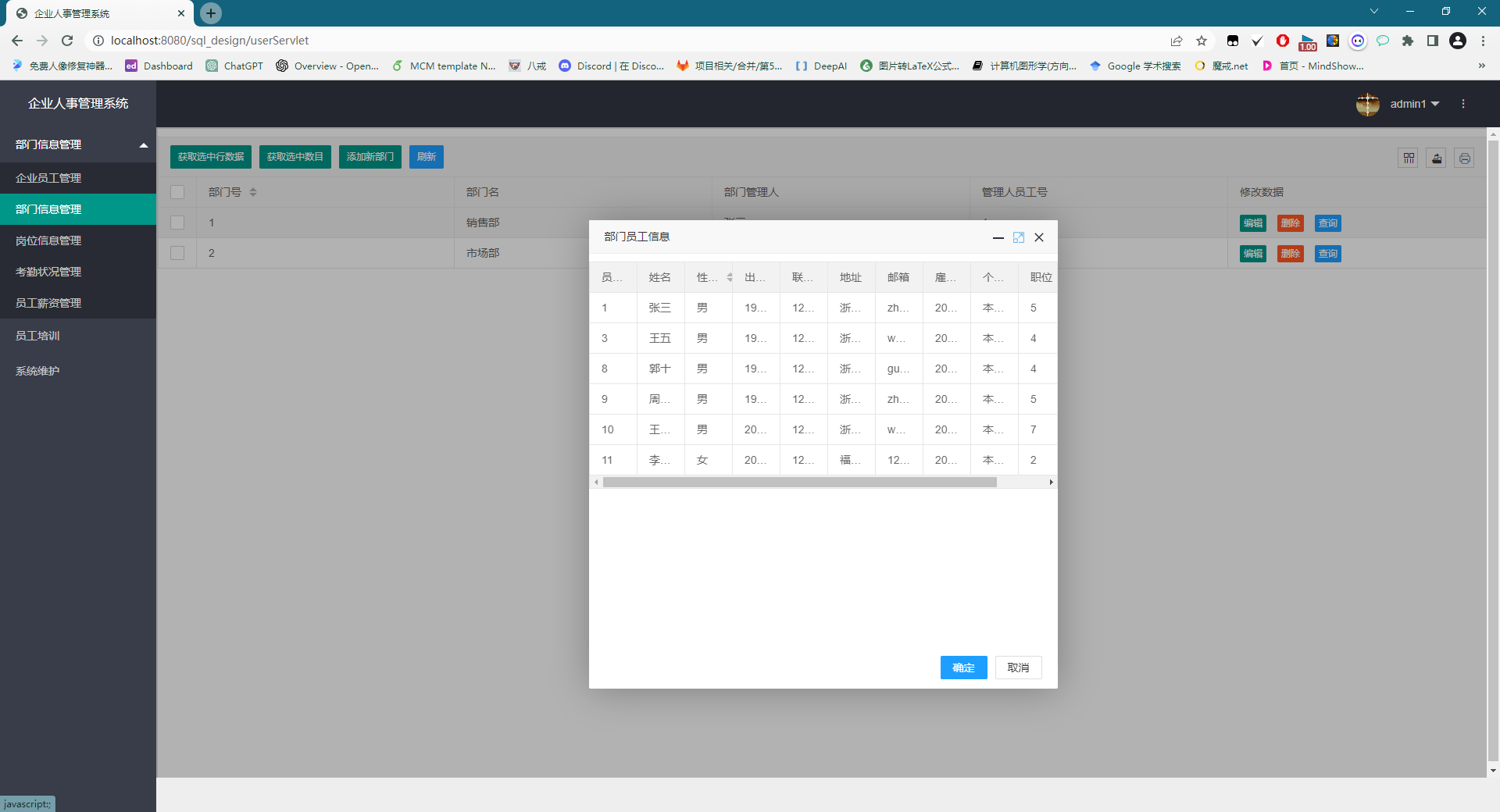
可以对相应的员工进行培训，在输入相应的培训ID后，就可以在培训信息中看到参与该培训的人员。



同时所有表格中的数据均支持筛选列，导出CSV，Excel文件或者打印，如下：



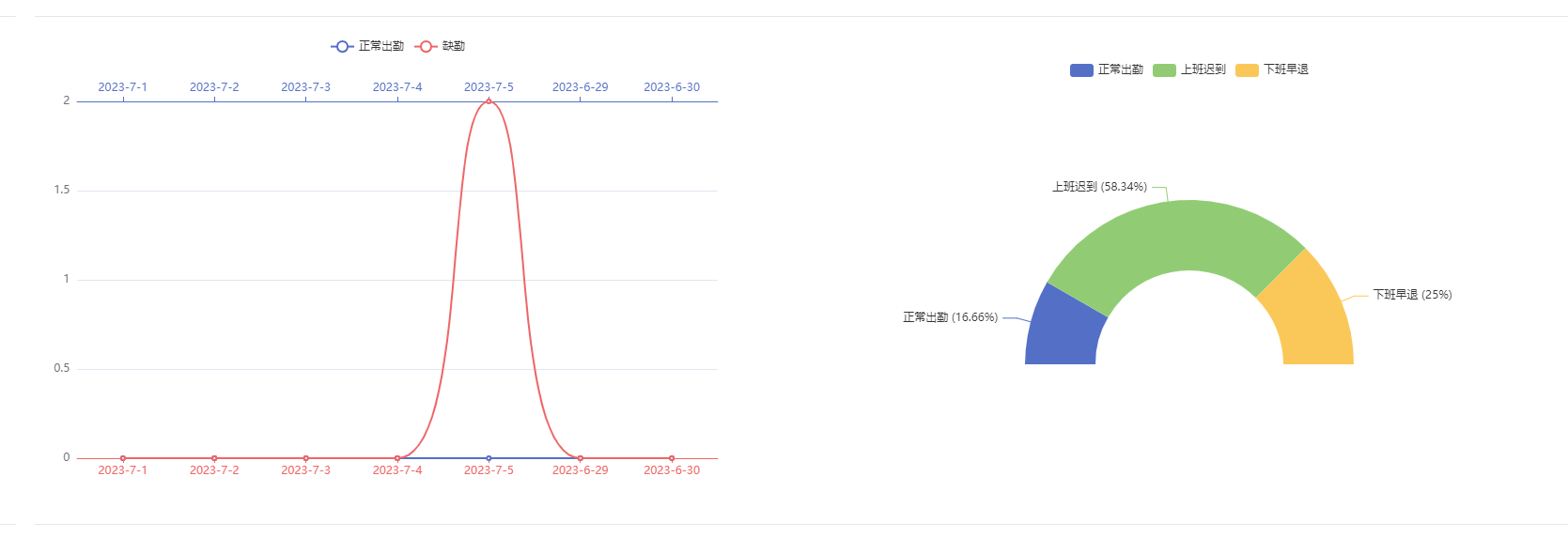
在部门信息管理界面，可以查看所有隶属于该部门的人员的信息：

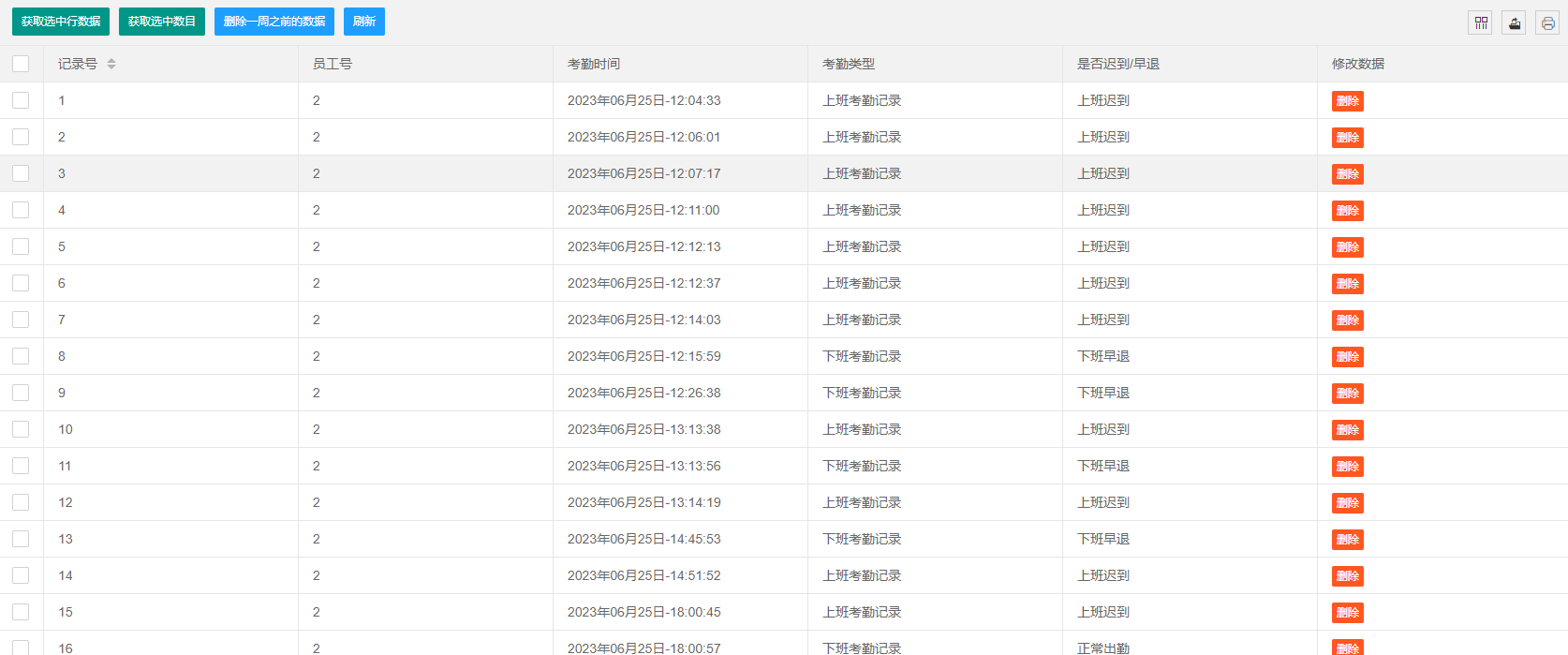


如果删除掉了某一个员工，则其相应的部门信息也会更新，以及该员工对应的薪资记录也会清空。

岗位信息管理页面，同样可以进行岗位信息的增加修改删除。

考勤信息管理页面，进行了近期的考勤情况的可视化：

在页面下方也展示了对应地考勤记录：

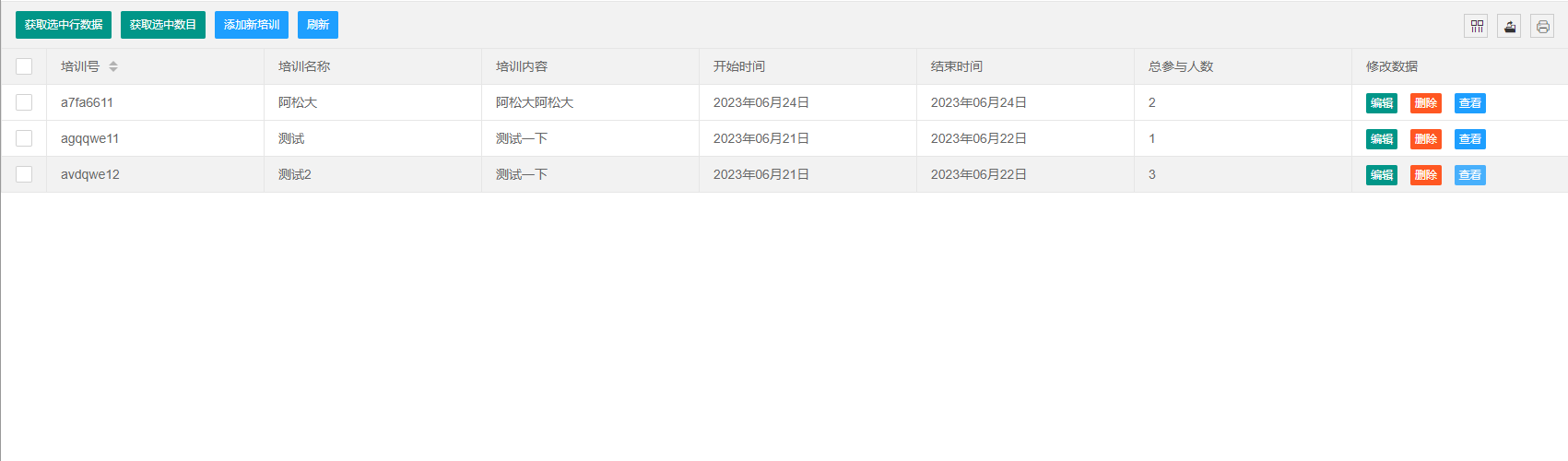


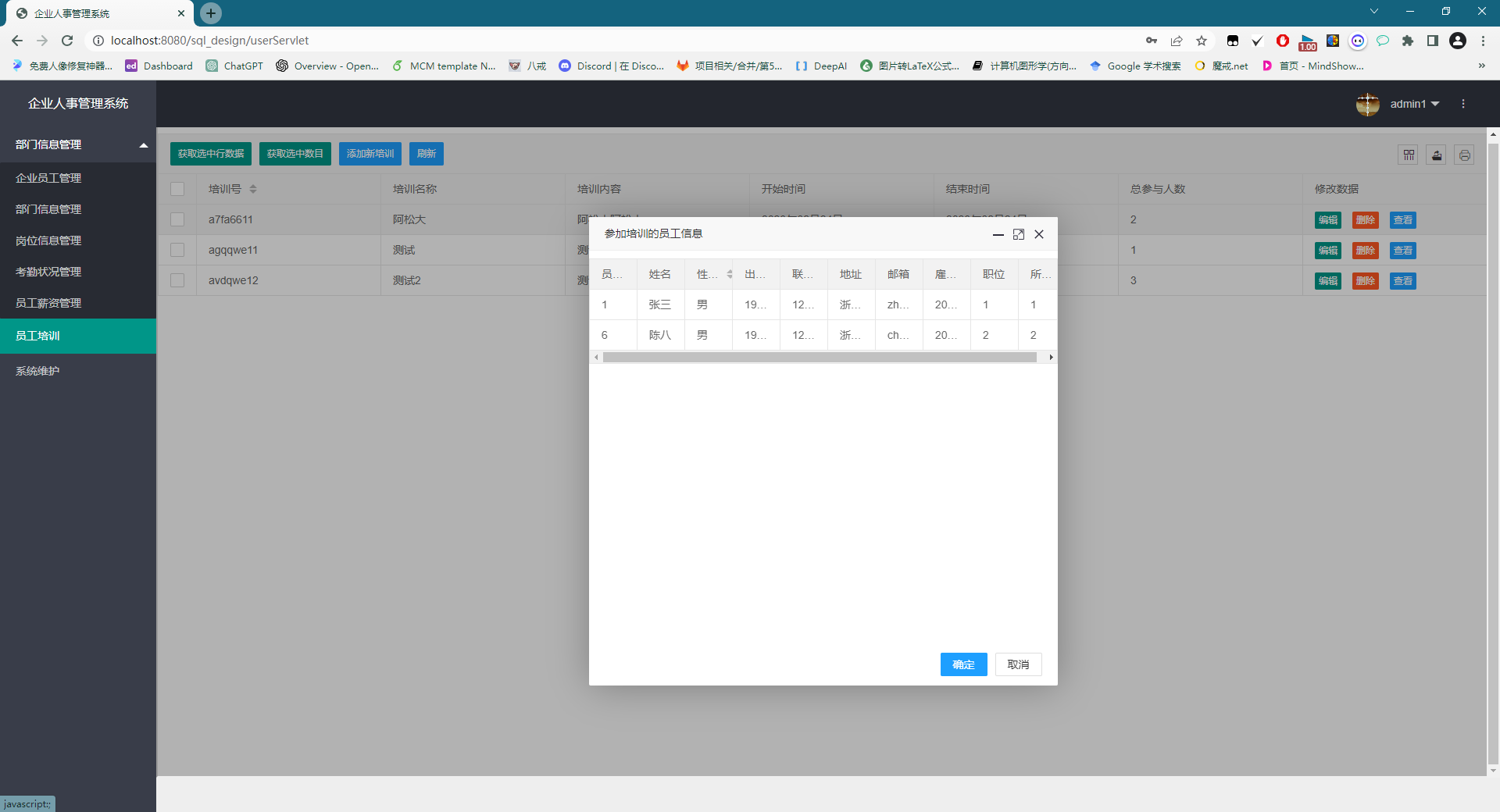
可以支持删除之前一周地数据以及普通删除功能，该表并不能被修改。

员工薪资管理页面通过JAVA中地UUID生成地薪资号，展示了薪资金额，发放原因，发放金额与时间。

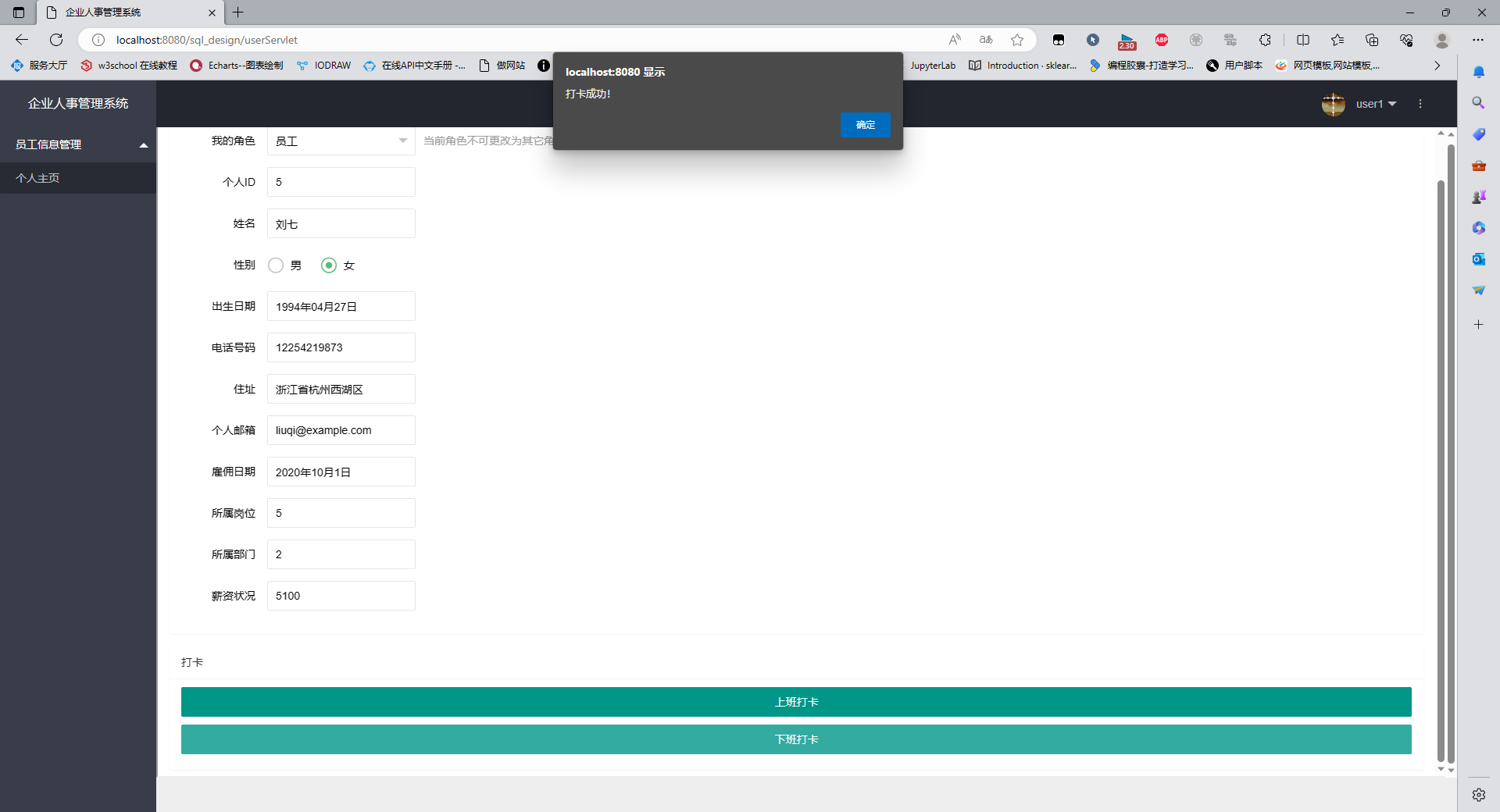


员工培训界面展示了所有的培训以及培训内容，开始结束时间，总参与人数等信息，并且可以查看所有参加该培训人员的信息。

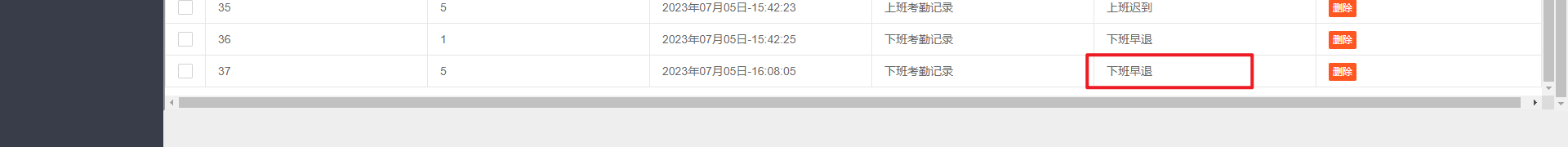




在员工个人登陆后的页面，支持员工查看自己的个人信息，但并不允许员工对个人信息进行修改，员工仅具有对其的查看权限。同时员工还能进行上下班的打卡，如果上班打卡时间晚于上午9：00或者下班打卡时间小于下午18：00，则相应的触发器会认定该员工未能正常出勤。



当我在下午16:00点击下班打卡后，系统会提示打卡成功，之后就可以在考勤信息管理界面看到该条打卡记录：



可以看到该次打卡被认定为下班早退。

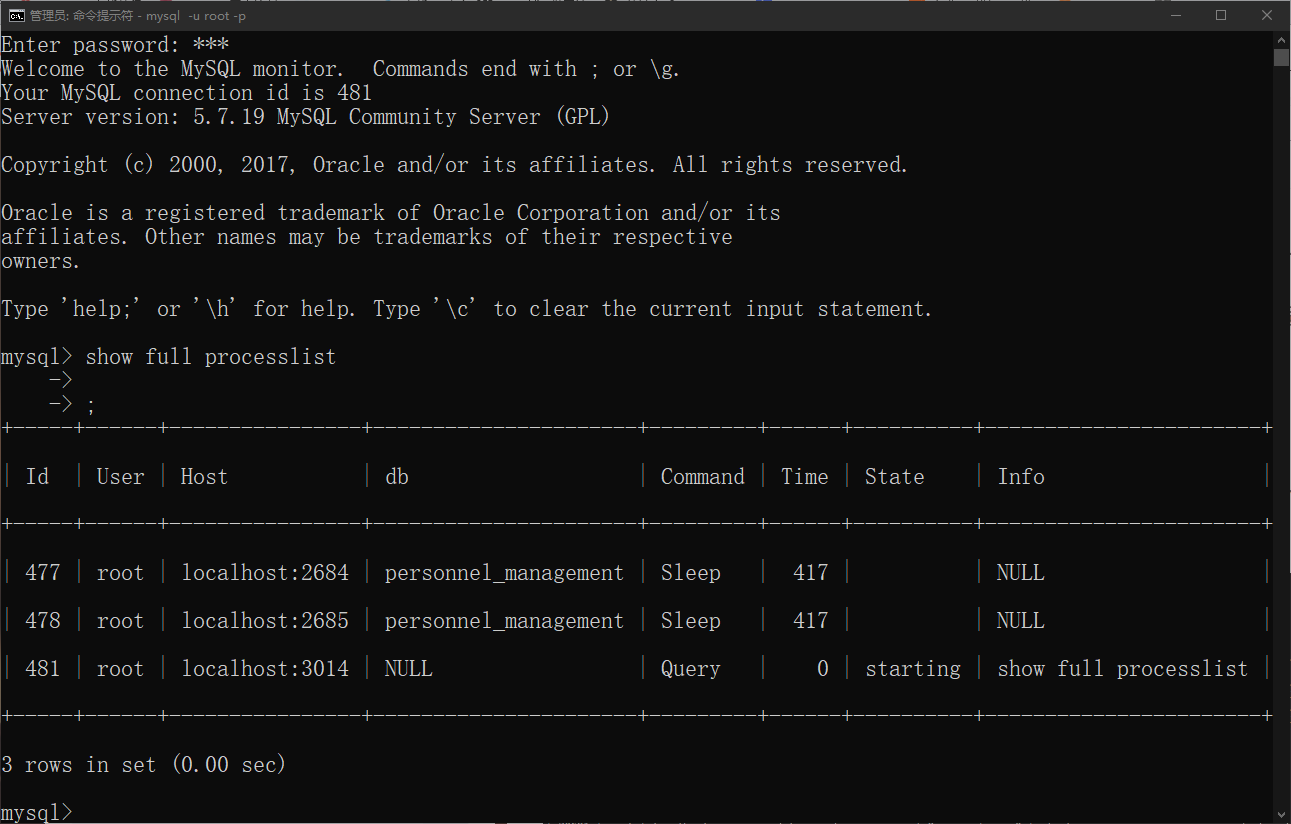
# 8 遇到的问题与解决办法

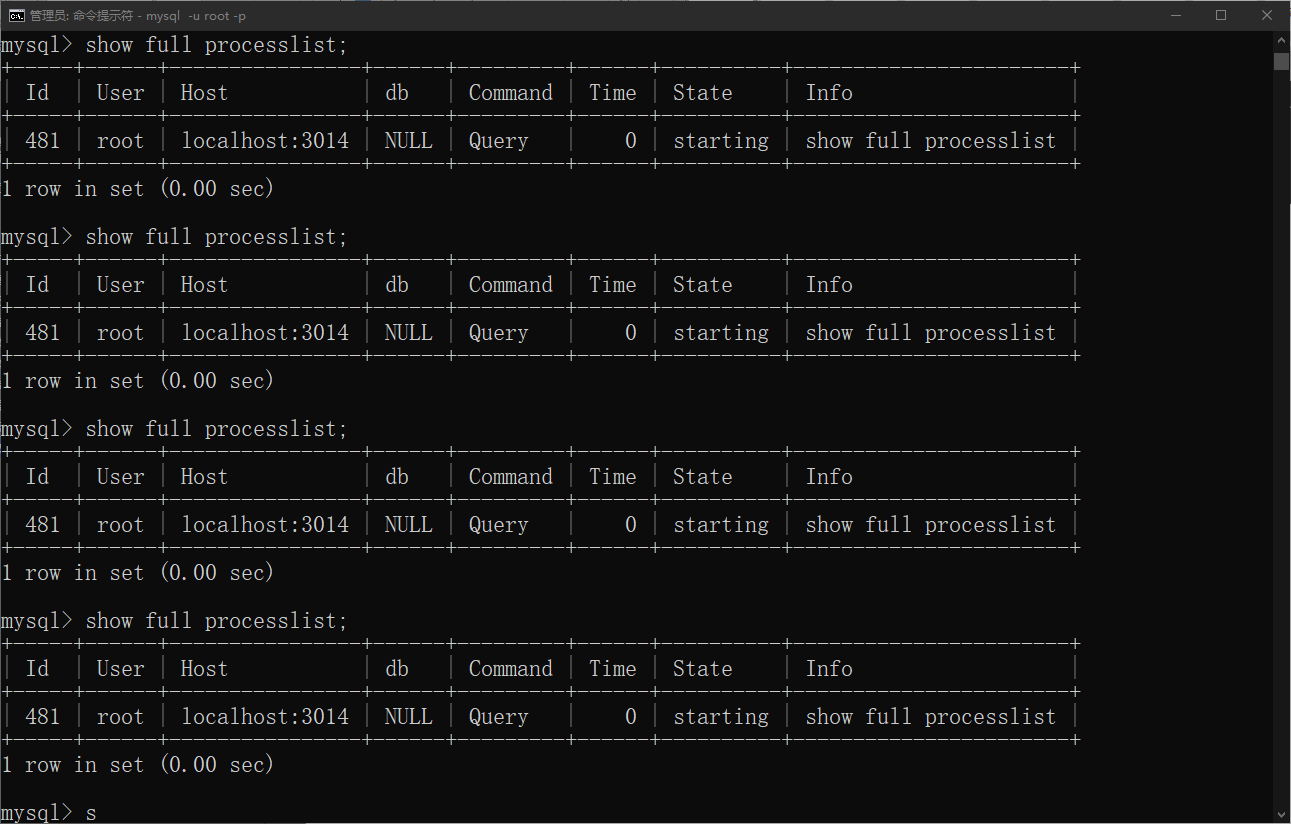
## 8.1 问题一——MySQL DROP表卡死

由于我当时创建employee表的过程中有一点内容写错了，我需要更改，所以我就想着把employee表先删除，然后重新创建并插入数据，但是在实际运行

Drop table employee时发现并不能正常执行，一直卡死，在网上搜了一些方法，最后发现是因为我在应用程序端还在使用该表，所以造成了该表无法删除。但是当我停止tomcat服务后，继续执行上述操作，发现还是不能执行，所以我继续查了相关的说明。通过搜查资料，我了解到可能是频繁的修改一个表的数据，那么这么表会被锁死，造成假死现象的原因导致的。

**解决办法：**

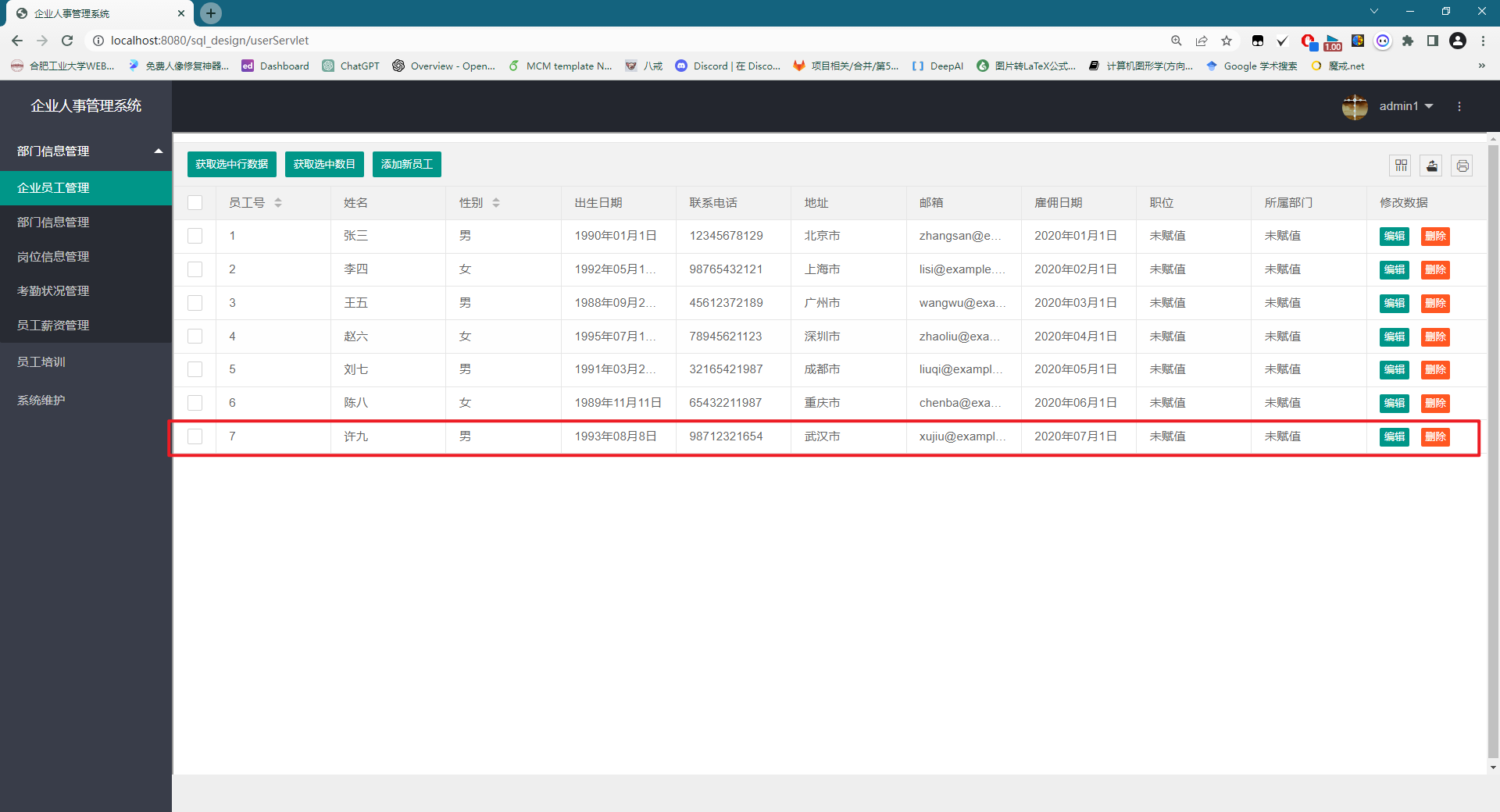
通过操作可以看到已经把与该数据库相关的进程都删除了。

接下来继续执行删除表的操作。

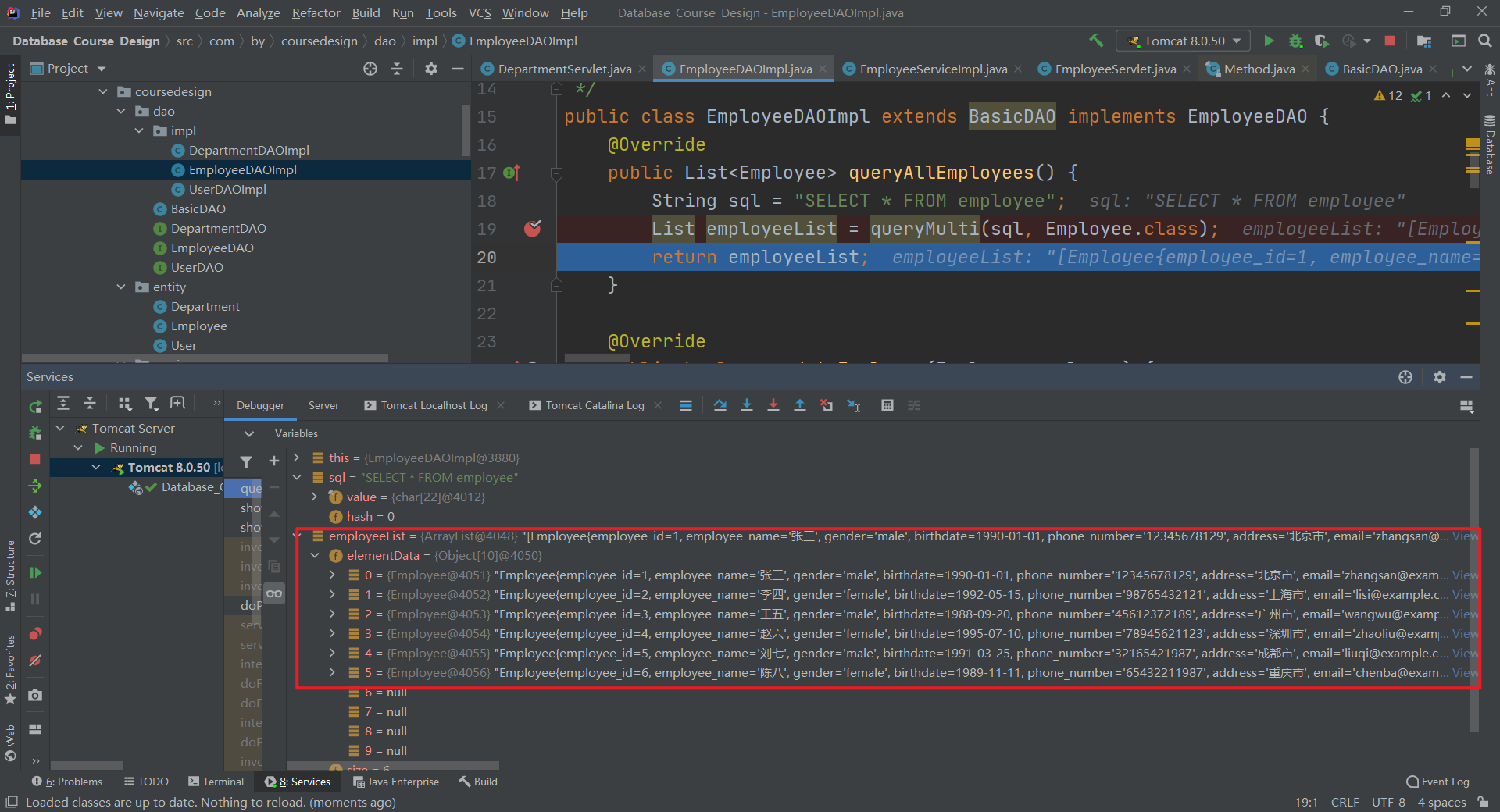
## 8.2 问题二——事务提交

在完成对部门信息管理的过程中，我还发现了一个十分怪异的问题，就是当我在前端删除某一条数据之后，在数据库中查看数据已经能够正常删除了。但当我多次刷新当前页面，会发现某一次这个被删除的数据突然又冒了出来。这让我感到十分的不解，因为毕竟我是从数据库查询的数据，每一次刷新都会向数据库执行一条sql语句，怎么会在没有更改表的情况下查询的结果不一样。然后我通过debug，多次刷新当前页面，突然发现确实在删除某一条数据之后，我多次刷新当前页面，这时执行到的sql语句会查询到明明已经删除的数据。

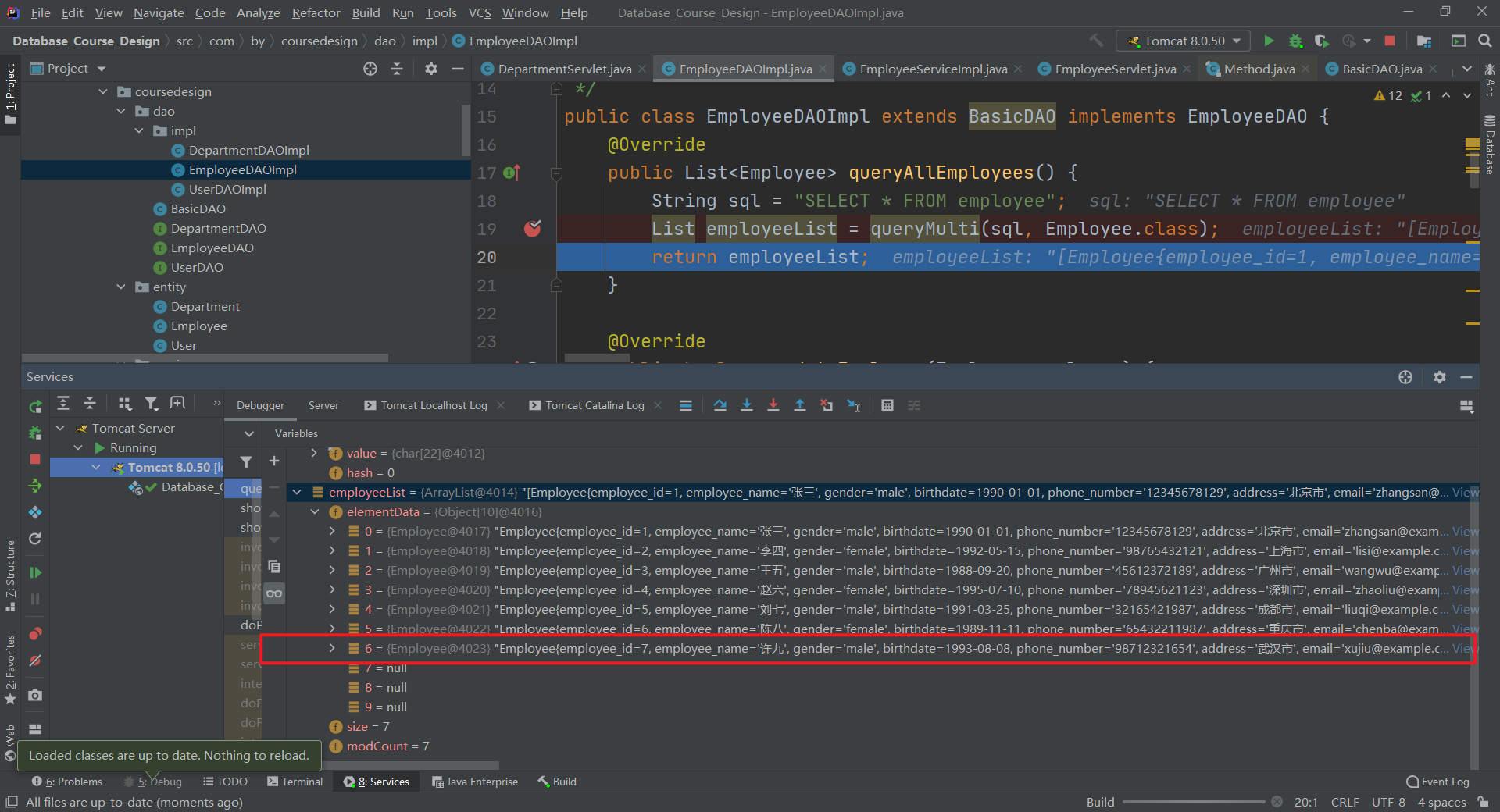
在开始的时候，我删除了陈九这条数据，如下所示：



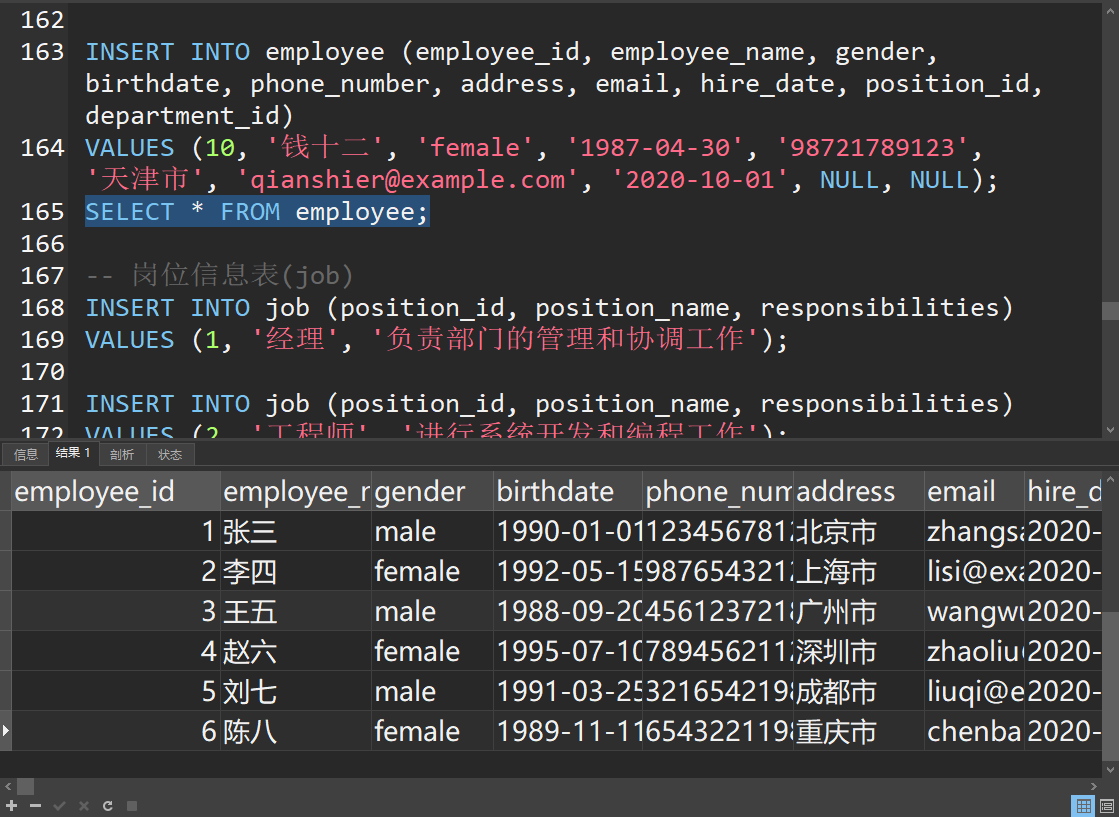
然后刚开始刷新能够正常显示剩余的6条数据：



但是当我频繁点击刷新按钮后，奇怪的事情发生了，某一次的debug，发现执行完sql语句后那条被删除的数据又回来了：



此时查看数据库根本没有该条数据：



**解决办法：**

绕了一大圈，查了很多资料，中遇找到了问题的原因所在，因为我刚开始是在BasicDAO中通过德鲁伊数据库连接池获取连接，并且由于在JDBCUtilsByDruid考虑了本系统可能出现多表操作，设置了事务手动提交，造成了此现象。也就是说在设置手动提交[SQL语句](https://so.csdn.net/so/search?q=SQL%E8%AF%AD%E5%8F%A5&spm=1001.2101.3001.7020)后，使用查询语句，这个语句并没有发送给MySQL进行查询，而是查的缓存。所以改了MySQL数据库里面的数据，并不会被Java程序查询到。而解决办法也十分的简单，就是在执行响应的sql操作后手动提交事务就可以解决此问题。

## 8.3 问题三——删除一周前的考勤记录问题

当我尝试写出删除一周之前的考勤记录的sql语句时，我尝试了下述的代码：

|  |
| --- |
| DELETE FROM attendance  WHERE attendance\_id IN (  SELECT attendance\_id  FROM attendance  WHERE log\_time < (  SELECT DATE\_SUB((SELECT MAX(log\_time) FROM attendance), INTERVAL 7 DAY) AS one\_week\_before  )  ); |

虽然经过验证SELECT能够正确筛选出需要删除的那些数据的attendance\_id号， 但是在delete时报出错误：

|  |
| --- |
| 执行结束提示：> 1093 - Unknown error 1093 |

**解决办法：**

通过查阅资料，我了解到产生这个错误的原因是SQL查询中包含了不能在子查询中引用的更新目标表。也就是说在同一个DELETE语句中，不能在子查询中引用正在被修改的目标表。为了解决这个问题，可以使用一个临时表或者子查询将需要删除的attendance\_id收集起来，然后再使用DELETE语句执行删除操作。如下所示：

|  |
| --- |
| DELETE FROM attendance  WHERE attendance\_id IN (  SELECT a.attendance\_id  FROM (  SELECT attendance\_id  FROM attendance  WHERE log\_time < (  SELECT DATE\_SUB((SELECT MAX(log\_time) FROM attendance), INTERVAL 7 DAY) AS one\_week\_before  )  ) a  ); |

这样就解决了问题。

# 总结

数据库课程设计作为学习数据库理论和应用的重要环节，通过实际设计和开发数据库应用系统，我深入理解了数据库的概念、原理和操作技术，并掌握数据库设计和管理的实践能力。以下是我完成此次数据库课程设计后的一些总结与个人感悟。

在本次数据库课程设计中，我深刻体会到了数据库在现代信息系统中的重要性和广泛应用。数据库是数据管理的核心，几乎涉及到每一个软件系统的数据存储和查询操作。通过这次设计，我学会了如何从实际需求出发，构建适合应用场景的数据库结构，设计合理的表和字段，并建立表之间的关联关系。

在数据库设计的过程中，我也遇到了一些问题。例如，如何合理地规划数据库的主键和外键，以及如何处理复杂的数据关系。通过与查阅相关的资料以及与同学的交流，我逐渐掌握了解决这些问题的方法和技巧，提高了自己的解决问题的能力。

在数据库的实现过程中，我不仅学会了使用SQL语言进行数据操作，包括插入、更新、删除和查询等操作。同时，我也了解了数据库索引的作用，以及如何优化查询语句，提高数据库查询性能。同时我还使用了触发器，视图，创建SQL用户进行权限控制等技术，让我更深刻地在实践中学会了数据库编程。

同时在此次课程设计完成过程中，我还学会了使用数据库连接池来更高效的连接，以及除了数据库本身的技术知识，我还学会了使用一些数据库管理工具，如Navicat等，这些工具可以简化数据库的管理和维护过程，提高工作效率。

此外，本次课程设计还锻炼了我的团队协作能力。在团队中，每个人分工负责不同的任务，我们需要相互配合，及时沟通，共同解决问题，同时在此次课程设计我还尝试了使用Git进行版本控制，也让我体会到了进行版本控制带来的诸多优点以及对团队开发带来的便捷。通过团队合作，我不仅学会了团队协作的技巧，还提高了沟通和协调能力。

最重要的是在完成此次课程设计之后，我认为自己对于软件开发的整体流程有了一个更深层次的切身体会。毕竟之前开发软件时常是规模较小的，并不经常会涉及到软件开发的全流程。而在此次课程设计中，我不仅仅完成了从需求分析，概念结构设计，还完成了逻辑结构设计编码与实现，还进行了软件的模块测试，让我深刻地从头到尾体验了一次软件开发的流程，我认为这给我的未来职业发展带来了很大的优势。

总结而言，本次数据库课程设计是我大学学习中的一次重要实践经验。通过这次设计，我不仅掌握了数据库的理论知识和实际操作技巧，还培养了解决问题和团队合作的能力，为我今后的学习和职业发展打下了坚实的基础。我相信在未来的工作中，这些在数据库课程设计中所学到的知识和经验将会派上用场，并让我在职业生涯中更加出色。