

**计算机与信息学院**

**《数据库系统课程设计》报告**

|  |  |
| --- | --- |
| **选题题目：** | 11 企业篇之员工工资管理系统 |
| **学号姓名：** |  |
| **专业班级：** |  |
| **指导老师：** |  |

**2023年 7月 5 日**

目录

[一、 系统开发概述 3](#_Toc139642210)

[二、 系统规划 3](#_Toc139642211)

[2.1设计任务、要求 3](#_Toc139642212)

[2.2 所用软件环境或工具 4](#_Toc139642213)

[三、 需求分析 5](#_Toc139642214)

[3.1 用户需求说明 5](#_Toc139642215)

[3.1.1 数据需求 5](#_Toc139642216)

[3.1.2 事务需求 5](#_Toc139642217)

[3.2 系统需求说明 6](#_Toc139642218)

[四、 数据库逻辑设计 7](#_Toc139642219)

[4.1 E-R图 7](#_Toc139642220)

[4.2数据字典 8](#_Toc139642221)

[描述 8](#_Toc139642222)

[定义 8](#_Toc139642223)

[4.3 数据库的逻辑结构设计 9](#_Toc139642224)

[4.3.1 关系模式 9](#_Toc139642225)

[4.3.2 表的建立 10](#_Toc139642226)

[五、 数据库物理设计 11](#_Toc139642227)

[5.1 索引 11](#_Toc139642228)

[5.2 视图 11](#_Toc139642229)

[5.3 安全机制 12](#_Toc139642230)

[六、 应用程序设计 13](#_Toc139642231)

[6.1 登录模块 13](#_Toc139642232)

[6.2 主页面模块 14](#_Toc139642233)

[6.3 信息输入模块 21](#_Toc139642234)

[6.4 安全性验证模块 32](#_Toc139642234)

[七、 总结 37](#_Toc139642235)

1. 系统开发概述

在企业管理中，员工工资管理是一个重要的环节。传统的人工处理工资信息繁琐且容易出错，因此开发一个员工工资管理系统成为了必要且迫切的需求。这个系统可以提高工资管理的效率，减少人力资源的消耗，并确保工资信息的准确性和安全性。

本文将概述一个企业员工工资管理系统的开发过程。首先，根据问题需求设计E-R模型，然后根据E-R模型设计相应的关系模型，并判断其所对应的范式。对于合理的关系模型，可以设计相应的表。接下来，使用MySQL数据库作为系统的后端技术，并使用JSP作为前端技术，使用HTML和CSS进行页面设计。系统的架构采用B/S架构。

该系统涉及的实体包括部门信息、员工信息、工资信息、津贴、加班工资、代扣款项等。系统的基本功能包括增加、删除、修改和查询这些实体的信息。为了确保系统的安全性，需要实施权限控制，只有授权的用户才能访问特定的功能和数据。此外，系统还应具有视图功能，以保护后台数据的安全。为了防止SQL注入等安全威胁，在系统开发过程中采用了相应的安全措施。另外，为了确保数据的完整性和一致性，系统设计了强约束性，即通过设定合适的约束条件来限制数据的输入和修改。

1. 系统规划

2.1设计任务、要求

企业篇之员工工资管理系统

涉及：部门信息、员工信息、工资信息、津贴、加班工资、代扣款项等实体。

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。

课题验收的主要参考指标（含要求）：

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！）

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！）

5、触发器（根据对应用的理解，是否采用触发器？带来的好处？）

存储过程（在充分理解系统业务处理的前提下，做出自己的判断，要运用得当！）

6、索引（需结合数据量的估算、应用的特点等，做出合理的设计）

系统功能（在需求分析的基础上尽可能全面一些！）

用户界面友好性（多从用户的角度考虑！）

2.2 所用软件环境或工具

IDE：IDEA Ultimate-2023.1.1；

中间件：Tomcat-8.5.88

数据库：Mysql-8.0.33；

数据库连接驱动：Mysql-connector-java-8.0.25

操作系统：Windows10-家庭版-22H2

开发环境：JDK-1.8.0\_291；JRE- 1.8.0\_291

1. 需求分析

3.1 用户需求说明

3.1.1 数据需求

（1）数据库要求

|  |  |
| --- | --- |
| 类型 | 表名 |
| 用户信息表 | employee |
| 工资表 | salary |
| 津贴表 | allowance |
| 加班表 | overtimesalary |
| 代扣款项表 | deduction |
| 部门表 | department |

表1、数据库中包含的表

3.1.2 事务需求

1、用户登录；

2、权限验证；

3、查看和切换员工表、工资表、津贴表、加班表、代扣款项表、部门表；

4、在表格有多条信息时可以进行翻页；

5、新注册员工信息；新发放工资信息、津贴信息、加班工资信息；新增部门；新增代扣款项。

6、删除员工信息、工资信息、津贴信息、加班工资信息、部门信息、代扣款项信息；

7、编辑员工信息、工资信息、津贴信息、加班工资信息、部门信息、代扣款项信息；

8、按关键词搜寻信息；

9、退出登录；

10、用户成功操作时弹出相应提示；用户违规操作时拒绝操作并弹出相应提示。

3.2 系统需求说明

1、员工信息管理

主要实现普通员工对自己信息的查看以及管理员对所有员工的管理，普通员工只读信息，管理员可以新增员工并授予权限、编辑员工姓名、所属部门、入职日期等。

2、部门信息管理

主要实现管理员对部门信息的管理，管理员可以新增部门、编辑部门名称。

3、工资信息管理

主要实现管理员对工资信息的管理，管理员可以新发放一笔工资，并编辑收款员工、工资金额、发放日期等信息。

4、津贴信息管理

主要实现管理员对津贴信息的管理，管理员可以新发放一笔津贴，并编辑收款员工、津贴金额、发放日期等信息。

5、加班信息管理

主要实现管理员对加班信息的管理，管理员可以新发放一笔加班费，并编辑收款员工、加班费金额、加班时长、发放日期等信息。

6、代扣款项信息管理

主要实现管理员对代扣款项信息的管理，管理员可以新发放一笔代扣款项，并编辑扣款员工、代扣金额、扣款日期等信息。

1. 数据库逻辑设计

4.1 E-R图

|  |
| --- |
| 图1、E-R图 |

4.2数据字典

|  |  |  |
| --- | --- | --- |
| 名称 | 描述 | 定义 |
| 员工信息 | 企业的所有员工信息 | ID、密码、所属部门ID、权限、入职日期、姓名 |
| 部门信息 | 企业的所有部门信息 | ID、部门名称 |
| 工资信息 | 发放给员工的工资信息 | ID、基本工资、发放日期、收款员工ID |
| 津贴信息 | 发放给员工的津贴信息 | ID、津贴金额、发放日期、收款员工ID |
| 代扣款项信息 | 员工的代扣款项扣款信息 | ID、代扣金额、扣款日期、扣款员工ID |
| 加班费信息 | 发放给员工的加班费信息 | ID、加班费金额、发放日期、收款员工ID、加班时长 |

表2、数据字典

4.3 数据库的逻辑结构设计

4.3.1 关系模式

员工信息表（ID、密码、所属部门ID1、权限2、入职日期、姓名）

部门信息表（ID、部门名称）

工资信息表（ID、基本工资、发放日期、收款员工ID3）

津贴信息表（ID、津贴金额、发放日期、收款员工ID4）

代扣款项信息表（ID、代扣金额、扣款日期、扣款员工ID5）

加班费信息表（ID、加班费金额、发放日期、收款员工ID6、加班时长）

注：1、所属部门ID为部门信息表中ID的外码

2、权限有检查约束：只能为“管理员”或“普通员工”

3/4/5/6、收款员工ID为员工信息表中ID的外码

因为上述关系的所有非主属性都完全依赖于每个候选码，所有主属性都完全 函数依赖于每个不包含它的候选码，没有任何属性完全函数依赖于非码的任何一 组属性，所以上述关系属于 3NF，也属于 BCNF.

4.3.2 表的建立

-- 部门表（Department）  
CREATE TABLE Department (  
 DepartmentID INT AUTO\_INCREMENT PRIMARY KEY,  
 DepartmentName VARCHAR(30) NOT NULL UNIQUE  
);  
  
-- 员工表（Employee）  
CREATE TABLE Employee (  
 EmployeeID INT AUTO\_INCREMENT PRIMARY KEY,  
 EmployeeName VARCHAR(30) NOT NULL,  
 Password VARCHAR(100) NOT NULL,  
 EmployeeLevel CHAR(10) NOT NULL CHECK (EmployeeLevel IN ('管理员', '普通员工')),  
 DepartmentID INT NOT NULL,  
 JoinDate DATE NOT NULL,  
 FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)  
);  
  
-- 工资表（Salary）  
CREATE TABLE Salary (  
 SalaryID INT AUTO\_INCREMENT PRIMARY KEY,  
 EmployeeID INT NOT NULL,  
 BaseSalary DECIMAL(15, 3) NOT NULL,  
 SalaryPaymentDate DATE NOT NULL,  
 FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)  
);  
  
-- 津贴表（Allowance）  
CREATE TABLE Allowance (  
 AllowanceID INT AUTO\_INCREMENT PRIMARY KEY,  
 EmployeeID INT NOT NULL,  
 AllowanceAmount DECIMAL(15, 3) NOT NULL,  
 AllowancePaymentDate DATE NOT NULL,  
 FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)  
);  
  
-- 加班工资表（OvertimeSalary）  
CREATE TABLE OvertimeSalary (  
 OvertimeSalaryID INT AUTO\_INCREMENT PRIMARY KEY,  
 EmployeeID INT NOT NULL,  
 OvertimeHours DECIMAL(10, 1) NOT NULL,  
 OvertimeSalaryAmount DECIMAL(15, 3) NOT NULL,  
 OvertimeSalaryPaymentDate DATE NOT NULL,  
 FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)  
);  
  
-- 代扣款项表（Deduction）  
CREATE TABLE Deduction (  
 DeductionID INT AUTO\_INCREMENT PRIMARY KEY,  
 EmployeeID INT NOT NULL,  
 DeductionAmount DECIMAL(15, 3) NOT NULL,  
 DeductionDate DATE NOT NULL,  
 FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)  
);

1. 数据库物理设计

5.1 索引

由于功能比较简单，因此只保留各表中的主键默认索引（主键值为自增）以及外键索引。

5.2 视图

创建各个表的完全视图，在进行数据查询和展示时通过视图进行操作，一定程度上保证了数据安全性。

CREATE VIEW DepartmentView AS  
SELECT \* FROM Department;  
  
CREATE VIEW EmployeeView AS  
SELECT \*  
FROM Employee;  
  
CREATE VIEW SalaryView AS  
SELECT \*  
FROM Salary;  
  
  
CREATE VIEW AllowanceView AS  
SELECT \*  
FROM Allowance;  
  
  
CREATE VIEW DeductionView AS  
SELECT \*  
FROM Deduction;  
  
CREATE VIEW OvertimeSalaryView AS  
SELECT \*  
FROM OvertimeSalary;

5.3 安全机制

1、**权限控制：**在新增员工时管理员可以编辑员工的权限，在登录时对登录员工权限进行判断后授予不同的操作权限。

2、**完整性约束：**数据库中包含实体完整性、参照完整性以及用户自定义完整性。在插入和编辑操作时，设计了数据字典候选项从而在一定程度上减少数据出错；在删除操作时会让用户进行二次确认，若操作违反完整性约束则系统拒绝操作。

3、**防SQL注入：**采用SQL预处理语句，只传递参数，在一定程度上可以防止SQL注入攻击。

4、**哈希加密：**登录密码使用SHA256验证和储存。

1. 应用程序设计（个人主要负责模块）

6.1 登录模块

在Session未保存或已过期的情况下，用户进入系统后，将弹出登录界面，用户可以填写用户名和密码登录系统，若通过数据库校验后存在该用户将成功登录，并弹出相应提示信息，同时部分用户信息将保存在Session中。

密码的比对方面是采用将密码转换为SHA256哈希值后与数据库中所存值进行比较。

// 接收参数  
String employeeId = request.getParameter("id"); //用户名  
String password = EncryptSha256Util.*getSha256Str*(request.getParameter("password")); //用户密码,转换为sha256后提交到数据库中进行验证  
String employeeLevel=null;

|  |
| --- |
| 图2、用户登录界面 |
| 图3、用户登录成功提示信息 |

6.2 主页面模块

登录成功后将进入主页面，在这里可以在几张表之间切换，同时对表中数据进行查看或者关键词搜索。若是普通用户登录，将只能看到与自己有关的数据，并且没有数据操作的权限；若管理员登入，将能对各表单进行新增、编辑、删除数据等操作。

|  |
| --- |
| 图4、员工表界面 |
| 图5、部门表界面 |
| 图6、工资表界面 |
| 图7、津贴表界面 |
| 图8、加班费表界面 |
| 图9、代扣款项表界面 |
| 图10、通过关键词搜索模块 |

6.3 信息输入模块

当在每个表的界面点击上方的蓝色加号按钮，将进入新增数据功能页面，当点击数据右侧对应的编辑按钮时，将进入到重新编辑该数据的页面。两个页面实际通过一个页面实现，在加载页面时通过判断用户是点击的新增数据还是编辑数据来执行接下来的代码。

例如在更新工资的页面中，若判断操作为新增数据则转到INSERT相关的SQL语句；反之则转到UPDATE相关的SQL语句，同时将原有的数据传到前端进行展示。

// 接收参数  
int id = 0;  
String isUpdate = request.getParameter("id");  
String isUpdateString = null;  
double BaseSalary = 0;  
String EmployeeName = null;  
Date SalaryPaymentDate = null;  
conn = new DBClass().get\_con();  
if(isUpdate != null) {//检测操作是新增数据还是编辑数据,null则为新增数据  
 String sql = "SELECT salary.\* , employee.EmployeeName " +  
 "FROM salary JOIN employee " +  
 "on salary.EmployeeID = employee.EmployeeID WHERE SalaryID=?";  
 id = Integer.*parseInt*(request.getParameter("id"));  
 pstmt = conn.prepareStatement(sql);  
 pstmt.setInt(1, id);  
 rs = pstmt.executeQuery();  
 isUpdateString = "编辑工资信息";  
 if (rs.next()) {  
 // 从数据库中取出内容  
 SalaryPaymentDate = rs.getDate("SalaryPaymentDate");  
 BaseSalary = rs.getDouble("BaseSalary");  
 EmployeeName = rs.getString("EmployeeName");  
 }  
}else{  
 isUpdateString = "新增工资信息";  
 BaseSalary = 1500;  
}  
String sql = "SELECT EmployeeName, EmployeeID FROM employee " ;//用来展示可以发放工资的员工  
pstmt = conn.prepareStatement(sql);  
rs = pstmt.executeQuery();

|  |
| --- |
| 图11、新增工资信息，默认工资为1500，同时发放日期是空值 |
| 图12、编辑工资信息，原来的信息将保留展示在表单中 |

以下是不同表更新数据页面的表单展示：

|  |
| --- |
| 图13、编辑员工信息界面 |
| 图14、编辑部门信息界面 |
| 图15、编辑津贴信息界面 |
| 图16、编辑加班费信息界面 |
| 图17、编辑代收款项信息界面 |

此外，还通过数据字典和日期组件的方式提高了用户填入数据时的体验，同时限制可以两者的输入框不能手动输入数据。一定程度减少了数据的格式出错。

<div class="form-group">  
 收款员工:  
 <select name="EmployeeID">  
 <% while (rs.next()) { %>  
 <option value="<%= rs.getString("EmployeeID") %>" <%= rs.getString("EmployeeName").equals(EmployeeName) ? "selected='selected'" : "" %>>  
 <%= rs.getString("EmployeeName") %>  
 </option>  
 <% } %>  
 </select>  
</div>  
<div class="form-group">  
 发放日期:  
 <div><input type="text" name="SalaryPaymentDate" value="<%=SalaryPaymentDate%>"  
 id="date" readonly style="resize:none;overflow-y:visible;font-size: 17px;color: darkslategray"></div>  
 <script>  
 laydate.render({  
 elem: '#date',  
 trigger: 'click'  
 });  
 </script>  
</div>

|  |
| --- |
| 图18、填入表单时的数据字典候选项 |
| 图19、填入日期时的日期组件 |

6.4 安全性验证模块

当用户插入的数据不符合数据库的完整性约束时，系统将拒绝操作，并弹出相应提示。例如员工姓名和日期不应该是空值。

if(EmployeeName == null || EmployeeName.isEmpty()||EmployeeName.equals("null")){  
 flag = false;  
 errInfo = "员工姓名不能是空值！";  
}

Date JoinDate = null;  
try {  
 JoinDate = Date.*valueOf*(request.getParameter("JoinDate"));  
}catch (java.lang.IllegalArgumentException e){  
 flag = false;  
 errInfo = "入职日期不能是空值！";  
}

|  |
| --- |
| 图20、员工姓名为空值时拒绝操作 |
| 图21、入职日期为空值时拒绝操作 |

当用户准备删除数据时，这将是一个相对危险的操作，系统将二次提醒用户是否确认删除；若确认删除后系统检测到删除的数据存在违反完整性约束的情况，将拒绝删除并弹出提示。（本系统没有设计级联删除功能等，只会拒绝操作）例如在删除部门时，若部门里存在员工，则无法删除该部门。

String sql = "DELETE FROM " + table + " WHERE " + keyName + "=?";  
 boolean flag = true;  
 conn=new DBClass().get\_con();  
 pstmt = conn.prepareStatement(sql);  
 pstmt.setInt(1, id);// 设置删除条件  
 try {  
 pstmt.executeUpdate();  
 } catch (SQLException e) {  
 flag = false;  
 String constraintName = e.getSQLState();  
 if (constraintName.equals("23000")) {//检测到违反外键约束  
%>  
 <div class="container">  
 <div class="infoBox">  
 <div>数据删除失败，存在有与该数据相关联的其他数据！<br><br><br>  
 <div><a href="list\_employee.jsp">取消删除</a></div><br>  
 </div>  
 </div>  
 </div>  
<%  
 }//end 检测违反外键约束  
 } finally {  
 pstmt.close();  
 conn.close();  
 }  
%>

|  |
| --- |
| 图22、二次确认是否删除数据 |
| 图23、当存在相关联的数据时拒绝操作 |
| 图24、删除成功的提示语 |

1. 总结

通过本次数据库课程设计，我和团队的成员成功完成了一个基于B/S架构的员工管理系统。这个项目不仅锻炼了我的编程能力，还培养了我从需求分析到数据库设计、完善数据库，再到最终的调试与部署的能力。在这个过程中，我学到了许多有关数据库的重要知识和技能。

首先，我学会了如何进行数据库设计。通过需求分析，我们确定了系统中的实体和关系，并使用这些信息设计了数据库的表和字段。我学会了选择合适的主键和外键，以确保数据的完整性和一致性。

在数据库操作方面，我学会了使用SQL语言执行各种操作。我掌握了SQL语句的基本语法，并学会了创建表、插入数据、查询数据、更新数据和删除数据、模糊查找等操作。我还学会了使用预处理语句来预防SQL注入攻击，确保系统的安全性。

为了提高数据库的安全性，我还学会了使用视图来保护数据库的安全性。通过创建视图，我可以隐藏敏感数据并限制用户的访问权限，确保数据的安全性。

在项目的开发过程中，我深刻体会到了前端与后端的配合的重要性。我学会了将前端界面与后端数据库进行有效的交互。通过JSP技术，我可以在前端页面中嵌入Java代码，并与数据库进行数据交互。这使得系统具备了良好的用户界面和强大的数据处理能力。

通过这个项目，我不仅加深了对数据库的理解，还提高了实际操作的能力。我相信这些学到的知识和技能将对我未来在数据库开发和管理领域的工作产生积极的影响。我期待将来能够运用这些知识，设计和开发出更加稳定、高效的数据库系统。