目录

一、 课程设计题目

题目 3 人事管理系统
人事管理系统主要有以下几项功能要求：
1.新员工资料录入。自动分配员工号，记录员工基本信息并设置用户密码。
2.人事变动的详细记录，包括岗位和部门的调整。
3.员工信息的查询和修改，包括员工个人信息和密码等。
分为两部分，“管理员页面”和“员工页面”
管理员可查看并修改所有员工信息
员工只允许查询个人信息

二、 课程设计目的

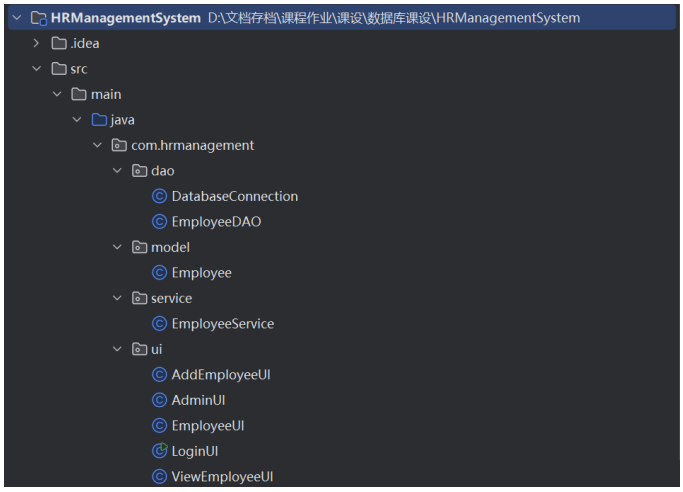
课设目的：
数据库系统课程设计是为了配合学习数据库系统原理及应用开发而设置的，是计算机科学与技术、 大数据、信息安全、物联网工程、软件工程、 智能制造等专业集中实践的教学环节，是将关系数据库理论知识转化为解决实际问题能力的重要环节。数据库系统课程设计目的在于加深对关系数据库理论知识的理解，通过使用具体的 DBMS，掌握一种实际的数据库管理系统并掌握其操作技术，熟练掌握使用数据库前端开发工具（如 VB、 C++、 Java、JSP、Delphi、 PowerBuilder 等），进一步提高运用数据库技术解决实际问题的能力。
实验目的：
1.熟悉 Java 面向对象编程及 Swing 图形界面的基本使用。
2.学习 MySQL 数据库的基础操作与 Java 的数据库交互（JDBC）。
3.实现一个简单的增删查改功能，用于模拟企业员工管理系统的基本功能。
4.掌握调试常见的数据库连接与操作问题的方法。

三、总体设计

1.背景介绍：
企业人事管理系统主要用于员工个人资料的录入、职务变动的记录和管理。使用人事管理系统，便于公司领导掌握人员的动向，及时调整人才的分配。
2.工具：
- 开发工具：（小皮面板 PHPStudy Pro 管理 ）MySQL、IntelliJ IDEA
- 运行环境：JDK 17 MySQL 5.7.26
- 数据库基础：使用 MySQL 来存储员工信息，通过 JDBC 实现 Java 与 MySQL 的交互。

- 图形界面：使用 Java Swing 构建用户友好的界面，模拟真实系统中的输入与显示流程。

3. 项目结构：




```
SRC/
    MAIN/
        JAVA/
            COM.HRMANAGEMENT/
                MODEL/ -- 实体类 (EMPLOYEE, DEPARTMENT, ETC.)
                DAO/ -- 数据库访问层 (CRUD 操作)
                SERVICE/ -- 业务逻辑层 (数据校验、变动处理)
                UI/ -- 用户界面 (SWING/JAVAFX 窗体)
            RESOURCES/
                SQL/ -- 初始化数据库的脚本
```

4. 系统层次结构设计

- Model 层

  定义了员工实体类 Employee，封装员工的基本属性，如 name、sex、department 等字段。

- DAO 层

  实现数据访问逻辑，封装与数据库的直接交互，包括员工数据的增删改查方法（如 addEmployee、getAllEmployees）。

- Service 层

  提供业务逻辑服务，调用 DAO 层的操作方法。

- UI 层

  使用 Java Swing 构建图形界面，包括员工信息的录入和查询显示。

5. 设计步骤

- 需求分析：确定系统的基本功能。
- 数据库设计：创建 hrmanagement 数据库，并设计管理系统的表与列。
- 分层架构设计：设计 Model、DAO、Service 和 UI 层，明确各层职责。

  数据访问（使用 JDBC）：

  EmployeeDAO 负责与数据库直接交互，提供以下方法：

addEmployee：新增员工信息。

getEmployeeId(int id)：通过 ID 查询员工。

getEmployeeByUsernameAndPassword(String username, String password)：用户登录验证。

getAllEmployees()：查询所有员工信息。

业务逻辑：

EmployeeService 封装与数据访问层的交互逻辑，为 UI 层提供服务。

用户界面层实现：

LoginUI

用户登录界面，实现管理员和普通员工的权限分流。

AdminUI

管理员主界面，提供查看所有员工和添加新员工功能入口。

AddEmployeeUI

添加新员工的界面，支持表单验证与数据提交。

ViewEmployeeUI

员工信息浏览界面，使用 JTable 显示员工列表，支持动态刷新。

EmployeeUI

普通员工信息查看界面。

- 功能实现：

设计数据存取（DAO）逻辑，确保数据库操作稳定性。

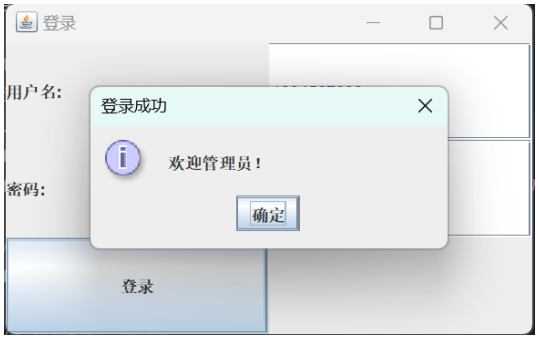
设计图形界面（Swing），模拟用户输入输出。

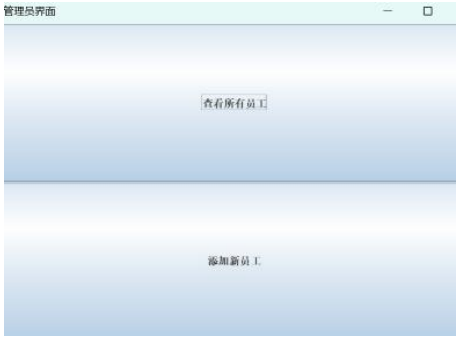
测试与调试：验证各功能模块正常运行，发现并解决问题。

6. 功能测试

通过模拟不同角色用户的登录与操作，对系统功能逐一验证。

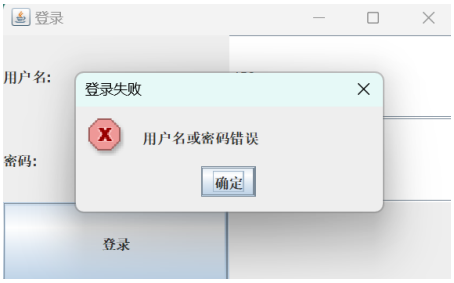
测试场景 1：管理员登录，成功查看员工列表，并新增一名员工。





测试场景 2：普通员工登录，仅能查看自身基本信息。

测试场景 3：输入无效的用户名或密码，提示登录失败。




```
---------------------
| 用户界面输入信息  |
---------------------
          |
---------------------
| 验证输入是否合法  |
---------------------
          |
     （合法继续）
---------------------
| 创建 Employee 对象 |
---------------------
          |
---------------------
|调用 Service.create |
---------------------
          |
---------------------
| DAO 操作数据库
---------------------
          |
---------------------
| 返回结果并提示用户 |
---------------------
```

用户输入信息 --> 验证输入 --> 封装为 Employee 对象 --> 调用 Service 层 --> Service 调用 DAO 层 --> 数据库执行插入语句 --> 返回结果

四、详细设计

1.数据库部分

- 创建数据库 hrmanagement，在其中创建五个表

department	存放部门信息
edu_level	存放学历信息
job	存放职位信息
person	存放人员信息

personnel　　存放人员
personnel_change　　存放变化信息

- E-R 图

参考课程设计指导书中给出的已有结构和 E-R 图，做出细微的修改

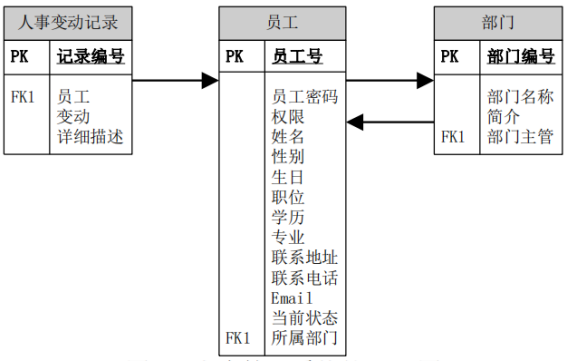


图 3、人事管理系统的 E-R 图

实体名称	属性	说明
员工 (Employee)	- EmployeeID (主键)	员工的唯一标识
	- Password	登录密码
	- Name	员工姓名
	- Gender	性别
	- Birthdate	出生日期
	- DepartmentID (外键)	所属部门，引用 Department 表中的 DepartmentID
	- Job	职位
	- EduLevel	受教育程度
	- Address	家庭住址
	- Tel	联系电话
	- Email	邮箱地址
	- State	当前状态 (T: 员工，F: 非员工)

实体名称	属性	说明
部门 (Department)	- DepartmentID (主键)	部门的唯一标识
	- Name	部门名称
	- Manager	部门经理
	- Description	部门简介

实体名称	属性	说明
人事变动 (PersonnelChange)	- ChangeID (主键)	变动记录的唯一标识
	- EmployeeID (外键)	关联的员工 ID，引用 Employee 表中的 EmployeeID
	- ChangeType	变动类型 (如：新增、调整职位、辞退等)
	- Description	变动详细说明
	- ChangeDate	变动日期

员工与部门：
一个部门可以包含多个员工（1:N）。
Employee 表通过 DepartmentID 外键与 Department 表关联。
员工与人事变动：
一个员工可以有多次人事变动记录（1:N）。
PersonnelChange 表通过 EmployeeID 外键与 Employee 表关联。

- person 表

数据库表设计以 person 表为核心，用于存储员工的基本信息和状态信息。
ID：确保每位员工的唯一性，用作表中的主键。设置为自增字段，数据库会自动为分配 ID。
PASSWD：员工登录系统的密码。
AUTHORITY：控制员工在系统中的权限级别。0 为普通员工，1 为管理员。
NAME、SEX、BIRTHDAY、DEPARTMENT、JOB、EDU_LEVEL、TEL、EMAIL：用于存储员工的基本身份信息，所属部门、职务和联系方式。与部门表和职位表建立外键关联，实现动态管理。
STATE：用于区分员工的工作状态（在职或离职）。
REMARK：为员工提供额外描述空间，如工作表现记录或其他重要信息。使用 text 类型以支持存储大量内容。

```
mysql> describe person;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| ID         | int(11)      | NO   | PRI | NULL    | auto_increment |
| PASSWD     | varchar(255) | NO   |     | NULL    |                |
| AUTHORITY  | int(11)      | YES  |     | 0       |                |
| NAME       | varchar(100) | NO   |     | NULL    |                |
| SEX        | char(1)      | YES  |     | NULL    |                |
| BIRTHDAY   | date         | YES  |     | NULL    |                |
| DEPARTMENT | int(11)      | YES  |     | NULL    |                |
| JOB        | int(11)      | YES  |     | NULL    |                |
| EDU_LEVEL  | int(11)      | YES  |     | NULL    |                |
| SPCIALTY   | varchar(100) | YES  |     | NULL    |                |
| ADDRESS    | varchar(255) | YES  |     | NULL    |                |
| TEL        | varchar(20)  | YES  |     | NULL    |                |
| EMAIL      | varchar(100) | YES  |     | NULL    |                |
| STATE      | char(1)      | YES  |     | T       |                |
| REMARK     | text         | YES  |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
```

2.前端设计

- 系统功能划分：前端主要通过 GUI 界面为用户提供可视化的操作平台，分为以下几个主要功能模块：

- 登录模块：
  输入用户 ID 和密码。
  验证登录信息，登录成功后跳转至主界面，登录失败显示错误信息。
- 主界面模块：
  提供功能入口，包括添加员工、查看员工信息、修改员工信息等。
  使用菜单或按钮布局，方便用户选择功能。
- 添加员工模块：
  提供输入表单，用户输入员工姓名、性别、职位等信息。
  点击“保存”按钮后，将信息写入数据库。
- 查看员工信息模块：
  查询数据库中的员工数据并以表格形式展示。
  支持按姓名或职位进行搜索，快速定位目标员工。

- 前端界面实现

（1）登录界面设计

- 功能描述：提供用户登录验证入口。
- 设计：
  创建登录界面（LoginUI）：要求用户输入用户名和密码，然后根据登录的角色（管理员或普通员工）来判断允许用户进行的操作。
  修改 EmployeeDAO，添加按用户名和密码查询员工的方法。

用户验证和角色区分：在数据库中， person 表中的 AUTHORITY 字段标识用户的权限

创建管理员界面（AdminUI），调整 ViewEmployeeUI 以允许管理员查看所有员工，员工只能查看自己的信息

根据角色控制权限

关键代码

```
public class LoginUI {
    public static void main(String[] args) {
        JFrame frame = new JFrame("登录");
        frame.setSize(400, 250);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new GridLayout(3, 2));
        // 添加组件：用户名
        frame.add(new JLabel("用户名:"));
        JTextField usernameField = new JTextField();
        frame.add(usernameField);
        frame.add(new JLabel("密码:"));
        JPasswordField passwordField = new JPasswordField();
        frame.add(passwordField);
        JButton loginButton = new JButton("登录");
        frame.add(loginButton);
```

- 界面结构：

  用户 ID 和密码输入框各 1 个。

  “登录”按钮 1 个。

  错误提示信息区域。

（2）主界面设计

- 功能描述：展示系统功能菜单，供用户选择具体操作。
- 界面结构：

  顶部菜单栏，包含 “添加员工”、“查看员工信息”等功能项。

  信息区域：显示系统欢迎信息或功能简介。

底部状态栏：显示当前时间或系统状态。

（3）添加员工界面设计

- 功能描述：允许用户通过表单输入新员工信息，并保存到数据库。
- 设计：

  界面设计：

  创建一个窗口，包含 JLabel（标签）显示字段名称，以及 JTextField（文本框）用于输入员工信息。

  一个 JButton（按钮）用于提交数据。

  事件监听：

  通过添加 ActionListener，捕获用户点击“保存”按钮的动作。

  输入：

  在提交数据前检查用户是否输入了所有必填项。

  若输入无效，弹出错误提示（JOptionPane.showMessageDialog）。

  数据库操作：

  使用 JDBC 提供的 PreparedStatement，构造 INSERT SQL 语句。

关键代码：

```java
public class AddEmployeeUI extends JFrame {

    public AddEmployeeUI() {
        setTitle("添加新员工");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE); // 确保关闭窗口时只关闭当前窗口
        setLayout(new GridLayout(6, 2));
        add(new JLabel("姓名:"));
        JTextField nameField = new JTextField();
        add(nameField);
        add(new JLabel("性别:"));
        String[] genders = {"男", "女"};
        JComboBox<String> sexComboBox = new JComboBox<>(genders);
        add(sexComboBox);
        add(new JLabel("部门:"));
        JTextField departmentField = new JTextField();
        add(departmentField);
        add(new JLabel("职务:"));
        JTextField jobField = new JTextField();
        add(jobField);
        add(new JLabel("联系方式:"));
        JTextField telField = new JTextField();
```

算法描述：

将用户输入的信息封装为参数。

打开数据库连接。

构建 INSERT INTO 语句，将参数值写入表中。

插入成功后，向用户显示提示，并清空表单。

- 界面结构：

输入框：员工姓名、性别、职位等信息输入框。

操作按钮：包括“保存”和“取消”按钮。

（4）查看员工信息界面设计

查询部分支持灵活的条件构建。若用户未输入条件，则查询所有记录。

使用循环遍历数据库结果集，按行将数据填充到表格中。

- 功能描述：提供一个表格展示数据库中员工信息，并支持按条件查询。
- 设计：

界面设计：

在窗口中放置 JTable 用于显示数据。

提供搜索条件的输入框和查询按钮。

数据获取：

使用 JDBC 连接到数据库。

根据用户是否提供查询条件，动态构建 SQL 查询语句（带 WHERE 条件或无条件）。

数据绑定：

将查询结果绑定到 DefaultTableModel。

利用 ResultSet 遍历查询结果，将每条记录添加到表格模型中。

界面更新：

将构建好的表格模型设置到 JTable，刷新界面显示数据。

```java
public ViewEmployeeUI() {
    setTitle("查看所有员工");
    setSize(600, 400);
    setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    setLayout(new BorderLayout());
    // 创建表格组件
    table = new JTable();
    model = new DefaultTableModel();
    table.setModel(model);
    model.addColumn("ID");
    model.addColumn("姓名");
    model.addColumn("部门");
    model.addColumn("职位");
    model.addColumn("状态");
    // 初始化数据
    loadEmployeeData();
    // 将表格放入滚动面板中
    JScrollPane scrollPane = new JScrollPane(table);
    add(scrollPane, BorderLayout.CENTER);
    // 添加工具栏或按钮区域
    JPanel buttonPanel = new JPanel();
    JButton refreshButton = new JButton("刷新列表");
    buttonPanel.add(refreshButton);
    add(buttonPanel, BorderLayout.SOUTH);
    // 刷新按钮点击事件
    refreshButton.addActionListener(e -> loadEmployeeData());
    setVisible(true);
}
```

- 算法描述：
  打开数据库连接。
  动态构建查询 SQL，考虑用户是否输入筛选条件。
  执行查询，获取 ResultSet。
  遍历 ResultSet 并填充表格模型。
  更新界面显示。
- 界面结构：

顶部搜索栏，包含搜索框和查询按钮。
主区域为员工信息表格。

五、 结果与分析

遇到的问题：

1. 数据库连接报错：Access denied for user 'root'@'localhost' (using password: YES)
   以为是用户名和密码的问题，但多次尝试并修改后无果，查找资料又寻找 Mysql 版本和权限问题，但都一一排除，最后发现是本地端口的设置的问题

   由于本地数据库搭建完成是在一年前，当时设置成了 3307 端口，但是在做课设的时候一直以为是 3306，一直没有意识到这个问题

2. 报错：’java.sql.PreparedStatement’ 中的 ’setInt(int, int)’ 无法应用于 ’(int, java.lang.String)’’com.hrmanagement.model.Employee’ 中的 ’setAuthority(java.lang.String)’ 无法应用于 ’(int)’
   解决：AddEmploy 方法的参数与其他类中不同。在 AUTHORITY 字段上传参用了“setInt”，但传递给 String 类型，在 Employee 类中同样是类型错误的问题，修改 AddEmploy 方法和 Employ 类后解决
3. 在上一个问题中延伸出来的另一个问题是用户输入字符的类型
   addemployeeui 报错：java: 不兼容的类型: java.lang.String 无法转换为 int
   重新检查并修改代码中的类型后解决
4. 问题：
   EmployeeService.createEmployee 方法没有完全初始化数据
   AddEmployeeUI 和 EmployeeDAO.addEmployee 数据不匹配
   解决：扩展 AddEmployeeUI 界面支持这些字段的输入

```
add(new JLabel("联系方式:")); add(telField);
add(new JLabel("邮箱:")); add(emailField);
add(new JLabel("备注:")); add(remarkField);
```

   在 AddEmployeeUI 中添加这些信息的设置框，让用户来设置

```
                emp.getTel(),
                emp.getEmail(),
                emp.getState()
```

5. 问题：运行登录页面后正常，但登录后在图形窗口添加的员工信息没有在数据库中显示，也不能在 ViewEmployeeUI 中查询到
   解决：发现是 createEmployee()和数据库中设置的字段不完全匹配，根据数据库中表与列的设置重新修改函数内容后解决
6. 报错：无法解析 ’EmployeeService’ 中的方法 ’getAllEmployees’
   解决：根据 IDEA 的报错自动修复提示发现是 EmployeeService 方法依赖 EmployeeDAO 的 getAllEmployees 的问题
   修改 getAllEmployees 方法，并返回 List<Employee> 类型，之前的返回类型错误导致调用失败。
7. 在页面窗口设计以及美化中参考了许多博客内容
   https://ost.51cto.com/posts/4139
   https://blog.csdn.net/qq_37413883/article/details/80038384
   https://www.tusij.com/color/BADCAD
8. 进行异常处理，使用 try catch,catch 用于抛出异常信息

六、小结与心得体会

在本次实验中，我设计一个简单的员工管理系统，学习并实践了数据库操作、前后端交互、和异常处理等。实验中，我不仅对 Java 编程、Swing 图形界面开发、JDBC 数据库操作等技术有了更深入的了解，还通过实际编写代码和调试程序，解决了事务提交、数据验证等问题，增强了我在实际开发中的问题解决能力。前端界面与后端数据库之间的互动是本次实验的核心部分。通过 Swing 提供的图形界面组件，实现了一个简单易用的用户界面。通过 JDBC 的 Statement 和 PreparedStatement 对象，将数据传递到数据库并进行操作。此前对数据库和前端交互了解不足，只是停留两门课程的分别学习中。通过实践，我才真正理解了数据从前端到后端再到数据库的流动过程。

在系统设计中，遇到了不少问题，从一开始的数据库连接，到后来的代码程序异常，尤其是数据无法正确提交到数据库的问题。经过多次调试，才意识到事务提交的细节，而最开始并没有注意到。

让我觉得困难的部分不止对于系统框架的编写，还有关于图形界面的美化，在一开始对内容的不断调整导致窗口大小、排版出现错乱，在这部分进行了多次修改，参考了很多博客文章。对于代码出现的异常，在我排查不出问题时也请教了同学以及借助 ChatGPT 帮我检查。

同时，在写报告总结的过程中我也发现，遇到的很多报错和问题都是类似的，甚至好几次的报错是由于同一个错误，但在我的每一次修改中都遗留下了一些细枝末节没有被完善，才导致同一个错误不断出现。

对于这个系统的优化还可以从这些方面改进：

增强界面交互，由于对图形化界面设计的学习并不是很多，大多数是参考网上资料，做出了一个较为简单基础的界面，虽在设计过程中已做出优化（如窗口大小，以及部分提交内容使用下拉框选择），但还有很多可以改善的用户友好功能或窗口的美化。

数据库优化，考虑数据库的优化策略，如分库分表、数据备份、索引优化等，提高系统的性能。

功能扩展：未来可以增加更多的功能模块，如员工考勤、薪资管理、绩效评估等，以构建更为完整的管理系统。