

**Java学生信息管理系统**

**二 级 学 院**

**专 业 班 级**

**学 生 姓 名**

**学 号**

**授 课 教 师**

**摘要：**

本课程设计旨在开发一个学生信息管理系统，用于有效地管理学生的基本信息。系统提供了学生信息的增删改查功能，以及查询和排序功能。此外，系统还支持学生信息的备份、导入和导出，以便于数据的存储和共享。

在需求分析阶段，我们明确了系统的功能需求和非功能需求。学生信息字段包括学号、姓名、年龄、性别、出生年月、地址、电话、E-mail和学院等。用户可以通过用户界面模块输入学生信息，并通过业务逻辑模块进行相应的操作。数据访问模块封装了与数据库的交互操作，通过SQL语句实现学生信息的读写和查询。

系统设计采用了分层架构，包括前端界面、后端逻辑和数据库。前端界面设计菜单与用户进行交互，提供友好的操作界面。后端逻辑处理用户请求、管理数据和执行各项操作的核心逻辑。数据库采用SQL Server，用于存储学生信息数据。

通过本次课程设计，我们成功开发了一个学生信息管理系统，能够满足学生信息的增删改查需求，并具备查询、排序、备份、导入和导出功能。系统提供了良好的用户界面和友好的交互体验，能够有效地管理学生的基本信息，并确保数据的安全性和可靠性。

通过设计和开发过程，我们掌握了Java编程语言和SQL Server数据库的应用，提高了系统设计和开发的能力。此外，我们还加深了对学生信息管理系统的理解和实际应用，培养了解决实际问题和需求分析的能力。

总之，本次课程设计为学生信息管理系统的设计和实现提供了一个可行的方案，达到了预期的目标和要求。

**关键字：**

**1. 学生信息管理系统：**

学生信息管理系统是一个软件系统，用于有效地管理和维护学生的基本信息。它可以提供学生信息的录入、修改、查询和浏览等功能，帮助学校或机构更好地管理学生数据。

**2. 增删改查：**

增删改查是指在学生信息管理系统中对学生信息进行的常见操作：

① 增加：向系统中添加新的学生信息记录。

② 删除：从系统中删除指定的学生信息记录。

③ 修改：对已有的学生信息进行更新和修改。

④ 查询：根据特定条件，通过系统进行学生信息的检索和查找。

**3. 备份、导入、导出：**

备份、导入和导出是学生信息管理系统中的重要功能，用于数据的保护和共享：

① 备份：将学生信息数据进行备份，以防止数据丢失或损坏。

② 导入：将外部数据源中的学生信息导入到系统中，以实现数据的批量导入和更新。

③ 导出：将系统中的学生信息导出到外部文件或数据库中，以便于数据的共享和使用。

**4. SQL Server：**

SQL Server是一种关系型数据库管理系统，由微软公司开发和维护。在学生信息管理系统中，SQL Server可以作为后端数据库，用于存储和管理学生的基本信息。它提供了强大的数据管理和查询功能，可以通过SQL语言进行数据操作和查询。

**1. 课程设计介绍**

本课程设计旨在开发一个学生信息管理系统，使用Java语言设计并与SQL Server数据库进行交互。系统提供了学生信息的增删改查功能，以及备份、导入和导出学生信息的操作。

在系统中，学生信息包括学号、姓名、年龄、性别、出生年月、地址、电话、E-mail和学院等字段。用户可以通过界面进行学生信息的录入、修改、删除和查询操作，以便方便地管理和维护学生的基本信息。

系统的查询功能支持按照姓名、学号、学院等条件进行精确查询，同时还可以对查询结果进行排序，方便用户根据自己的需求进行信息检索。此外，系统还提供了备份学生信息、导入外部数据和导出数据的功能，以便用户进行数据的备份和共享。

在课程设计过程中，我们将运用Java编程语言的基本概念和技术，包括类的定义、方法的编写和对象的使用等。同时，还将学习与SQL Server数据库的连接和操作，包括数据库的增删改查、备份和导入导出等功能的实现。

通过完成这个简单的课程设计，学生将掌握Java编程语言的应用和SQL Server数据库的操作，提升自己的编程能力和数据库管理能力。同时，还将锻炼问题解决和团队合作的能力，为进一步的学习和职业发展打下基础。

**课程设计小组成员构成：**

沈宇辰：

① 代码编写，测试，调试。

② 课程设计报告书写。

刘天宇：

王经旭：

**2. 系统设计**

1. 系统架构：

① 前端界面：菜单设计

② 后端逻辑：实现系统的核心业务逻辑，包括学生信息的处理、验证和操作。

③ 数据库：使用SQL Server，用于存储和管理学生信息数据。

2. 模块划分：

① 用户界面模块：负责与用户交互，接收用户输入并显示查询结果、错误提示等信息。

② 学生信息管理模块：处理学生信息的增删改查等操作，包括对学生信息的验证和数据访问。

③ 数据库访问模块：封装与数据库的交互操作，包括连接数据库、执行SQL语句和获取查询结果等功能。

3. 数据库设计：

① 创建学生信息表：设计合适的表结构，包括学号、姓名、年龄、性别、出生年月、地址、电话、E-mail、学院等字段。

② 设定主键和索引：为学生信息表设置适当的主键和索引，以提高查询和操作的性能。

4. 数据流：

① 用户界面模块接收用户输入并传递给学生信息管理模块进行处理。

② 学生信息管理模块调用数据库访问模块进行数据库操作，包括数据的读取、更新和删除。

③ 数据库访问模块与数据库进行交互，执行相应的SQL语句进行数据操作和查询。

5. 文件操作：

① 导入和导出功能通过文件操作实现，可以使用Java的文件读写操作来实现学生信息的导入和导出。

**2．1 需求分析**

**1. 功能需求：**

① 学生信息录入：系统应该允许用户输入学生的学号、姓名、年龄、性别、出生年月、地址、电话、E-mail、学院等信息，并将其保存到数据库中。

② 学生信息查询：系统应该提供按学号、姓名、学院等条件进行学生信息的查询，并能显示满足条件的学生信息列表。

③ 学生信息修改：系统应该允许用户对已有的学生信息进行修改，包括更新学生的个人信息。

④ 学生信息删除：系统应该提供删除学生信息的功能，让用户能够删除不需要的学生记录。

⑤ 学生信息浏览：系统应该支持浏览所有学生信息的功能，以便用户查看所有学生的详细信息。

**2. 非功能需求：**

① 用户友好性：系统应该具有易于使用和直观的用户界面，使用户能够轻松地进行操作和导航。

② 数据安全性：学生信息应该被安全地存储和保护，只有授权的用户才能访问和修改学生信息。

③ 性能和可扩展性：系统应该能够处理大量的学生信息，并具备良好的性能和可扩展性，以满足未来的需求变化和数据增长。

**3. 备份、导入、导出需求：**

① 备份：系统应该提供学生信息的备份功能，让用户能够定期备份学生信息数据，以防止数据丢失。

② 导入：系统应该支持从外部文件或数据库导入学生信息的功能，以便用户能够批量导入学生数据。

③ 导出：系统应该具备将学生信息导出到外部文件或数据库的功能，以便用户能够与其他系统或应用程序共享学生数据。

**2．2 总体设计**

**1. 用户界面层：**

① 用户界面层负责与用户进行交互，接收用户的输入请求和展示查询结果。

② 使用输出语句创建菜单，设计用户友好的界面，包括学生信息录入表单、查询界面和结果显示界面。

③ 用户界面层通过事件监听和处理，将用户操作传递给业务逻辑层进行处理。

**2. 业务逻辑层：**

① 业务逻辑层包含处理学生信息的核心业务逻辑，负责验证数据、执行操作和处理业务规则。

② 学生信息的增删改查功能将在业务逻辑层中实现。

③ 业务逻辑层可以根据需要调用数据访问层进行数据库操作，如插入、更新、删除和查询等。

**3. 数据访问层：**

① 数据访问层封装对数据库的访问和操作，负责与数据库进行交互。

② 使用Java JDBC技术与SQL Server数据库建立连接，并执行SQL语句进行数据操作。

③ 数据访问层可以提供基本的CRUD操作，以及备份、导入和导出功能的实现。

**2．3 功能设计**

**1. 学生信息的增删改查：**

① 增加学生信息：提供学生信息录入表单，用户可以输入学生的学号、姓名、年龄、性别、出生年月、地址、电话、E-mail、学院等信息，并将其保存到数据库中。

② 删除学生信息：根据学号或其他条件，用户可以选择删除指定的学生信息记录，从数据库中移除该学生的信息。

③ 修改学生信息：用户可以对已有的学生信息进行修改，包括更新学生的个人信息，例如修改地址、电话或学院等字段。

④ 查询学生信息：提供按学号、姓名、学院等条件进行学生信息的查询功能，用户可以输入相关条件，系统将显示满足条件的学生信息列表。

**2. 备份、导入和导出：**

① 学生信息的备份：提供备份功能，让用户能够定期备份学生信息数据，以防止数据丢失。用户可以选择备份的路径和文件名，系统将生成备份文件保存。

② 学生信息的导入：支持从外部文件或数据库导入学生信息的功能，以便用户能够批量导入学生数据。用户可以选择导入的文件或数据源，并指定导入的规则和映射关系。

③ 学生信息的导出：具备将学生信息导出到外部文件或数据库的功能，以便用户能够与其他系统或应用程序共享学生数据。用户可以选择导出的目标文件或数据库，并指定导出的字段和格式。

**3. 系统各功能模块设计**

**1. 用户界面模块：**

① 设计菜单作为用户界面，包括学生信息录入表单、查询界面和结果显示界面。

② 提供输入验证功能，确保用户输入的数据符合规定的格式和要求。

③ 实现与用户的交互逻辑，接收用户的操作请求，并将请求传递给业务逻辑模块进行处理。

④ 显示查询结果、错误提示和操作成功消息等信息给用户。

**2. 学生信息管理模块：**

① 提供学生信息的增加、删除、修改和查询功能。

② 对用户输入的学生信息进行验证，确保数据的完整性和准确性。

③ 调用数据访问模块进行数据库操作，如插入、更新、删除和查询等。

④ 处理业务规则，如禁止重复的学生学号或其他约束条件。

3. 数据访问模块：

① 封装与数据库的交互操作，负责数据库连接、执行SQL语句和获取查询结果等功能。

② 建立与SQL Server数据库的连接，并确保连接的安全性和稳定性。

③ 提供对学生信息表的CRUD操作，包括插入、更新、删除和查询等。

④ 实现备份、导入和导出功能，通过文件操作或数据库操作实现数据的备份和迁移。

**3．1 界面设计**

**1. 主界面：**

使用输出语句设计一个清晰、简洁的主界面，展示系统的功能和操作入口。

**2. 学生信息录入界面：**

使用输出语句提示用户输入相关信息。

**3. 学生信息查询界面：**

使用输出语句提示用户输入相关信息后输出相应学生信息。

**4. 学生信息展示界面：**

直接输出数据库内所有学生信息。

**3．2 功能实现**

**1. 学生信息的增删改查功能实现：**

① 增加学生信息：通过界面的提示信息，用户输入学生的各个字段信息，并将其保存到数据库中。在后端，接收表单数据并进行数据验证，然后调用数据库操作的方法将数据插入到学生信息表中。

② 删除学生信息：根据学号或其他条件，用户选择要删除的学生信息记录，并确认删除操作。在后端，接收删除请求并调用数据库操作的方法执行删除操作。

③ 修改学生信息：用户输入提示的信息，选择要修改的学生信息记录，并在界面上更新相应字段的值。在后端，接收更新请求并调用数据库操作的方法执行更新操作。

④ 查询学生信息：用户在查询界面输入查询条件，系统接收查询请求并调用数据库操作的方法进行查询。将满足条件的学生信息从数据库中检索出来，并在界面上展示查询结果。

**2. 备份、导入和导出功能实现：**

① 学生信息的备份：用户选择备份功能，系统调用数据库操作的方法将学生信息表中的数据导出为备份文件，保存到项目根目录。

② 学生信息的导入：用户选择导入功能，并选择项目根目录内的文件。系统读取导入文件中获取学生信息数据，并调用数据库操作的方法将数据导入学生信息表中。

③ 学生信息的导出：用户选择导出功能，系统调用数据库操作的方法将学生信息表中的数据导出为文件保存到项目根目录。

**4. 课程设计总结和体会**

在完成这个简单的课程设计过程中，我深入学习和应用了Java编程语言以及与SQL Server数据库的交互。以下是我的总结和体会：

**1. 学到的知识和技能：**

① 熟悉了Java编程语言的基本语法和面向对象编程的概念，掌握了类的定义、方法的编写和对象的使用等。

② 学习了与SQL Server数据库的连接和操作，包括数据库的增删改查、备份和导入导出等功能的实现。

③ 掌握了编写简单的用户界面（GUI）的基本方法，包括表单设计、按钮点击事件的处理和界面跳转等。

**2. 体会到的挑战和解决方案：**

① 挑战：对数据库操作的理解和实现，包括连接数据库、编写SQL语句以及处理数据库异常等方面存在一定的难度。

② 解决方案：通过学习数据库基础知识和参考网上相关文档，逐步理解和掌握数据库操作的方法和技巧，并进行测试和调试。

**3. 收获和展望：**

- 通过这个简单的课程设计，我对Java编程和SQL Server数据库的应用有了更深入的了解和实践，提升了自己的技能和能力。

- 了解了系统开发的基本流程和方法，包括需求分析、系统设计、功能实现和测试等，为未来的开发工作打下了基础。

- 在未来的学习和实践中，我希望能够进一步拓展和完善这个学生信息管理系统，增加更多功能和扩展性，提高系统的性能和稳定性。

总体而言，这个简单的课程设计给了我宝贵的实践机会，让我能够将所学的理论知识应用到实际项目中。通过解决问题和克服困难，我不仅提升了技术能力，还培养了团队合作、问题解决和自我学习的能力。这将对我未来的学习和职业发展产生积极的影响。

**5．致谢**

**在完成这个课程设计的过程中，我要表达我的衷心感谢和致谢：**

**首先，我要感谢我的老师，感谢您对我的悉心指导和支持。在整个课程设计过程中，按照平时跟您学习的知识，帮助我解决了许多技术和设计上的问题。您的教学对我的学习和成长起到了重要的推动作用。**

**其次，我要感谢我的同学和朋友们，感谢你们在我遇到困难和挑战时给予的鼓励和支持。你们的合作和讨论使得这个课程设计更加丰富和完善。我们一起面对问题、解决问题，共同努力取得了这个成果。**

**最后，我还要感谢学校提供的学习资源和学习环境，为我提供了良好的学习平台。感谢各位教师的教育和培养，让我具备了相应的知识和技能去完成这个课程设计。**

**再次向以上所有给予帮助和支持的人表示由衷的感谢！你们的支持和鼓励是我不断前行的动力，我将继续努力学习和成长，回报你们的关怀和期望。谢谢大家！**

**参考文献**（四号粗）

[1]Java程序设计基础（第七版）.清华大学出版社,2021.7

**附录**

1. StudentManagementSystem

（1）数据库连接

public class StudentManagementSystem {  
 private static final String *DB\_URL* = "jdbc:sqlserver://localhost:1433;databaseName=StudentDB;encrypt=false";  
 private static final String *DB\_USERNAME* = "sa";  
 private static final String *DB\_PASSWORD* = "341122";

（2）添加学生信息

public void addStudent(Student student) {  
 try (Connection conn = DriverManager.*getConnection*(*DB\_URL*, *DB\_USERNAME*, *DB\_PASSWORD*)) {  
 String sql = "INSERT INTO students (student\_id, name, age, gender, birthdate, address, phone, email, college) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)";  
 PreparedStatement statement = conn.prepareStatement(sql);  
 statement.setInt(1, student.getStudentId());  
 statement.setString(2, student.getName());  
 statement.setInt(3, student.getAge());  
 statement.setString(4, student.getGender());  
 statement.setDate(5, new java.sql.Date(student.getBirthdate().getTime()));  
 statement.setString(6, student.getAddress());  
 statement.setString(7, student.getPhone());  
 statement.setString(8, student.getEmail());  
 statement.setString(9, student.getCollege());  
 statement.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
}

（3）删除学生信息

public boolean deleteStudent(int studentId) {  
 try (Connection conn = DriverManager.*getConnection*(*DB\_URL*, *DB\_USERNAME*, *DB\_PASSWORD*)) {  
 String sql = "DELETE FROM students WHERE student\_id = ?";  
 PreparedStatement statement = conn.prepareStatement(sql);  
 statement.setInt(1, studentId);  
 statement.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return false;  
}

（4）更新学生信息

public void updateStudent(Student student) {  
 try (Connection conn = DriverManager.*getConnection*(*DB\_URL*, *DB\_USERNAME*, *DB\_PASSWORD*)) {  
 String sql = "UPDATE students SET name = ?, age = ?, gender = ?, birthdate = ?, address = ?, phone = ?, email = ?, college = ? WHERE student\_id = ?";  
 PreparedStatement statement = conn.prepareStatement(sql);  
 statement.setString(1, student.getName());  
 statement.setInt(2, student.getAge());  
 statement.setString(3, student.getGender());  
 statement.setDate(4, new java.sql.Date(student.getBirthdate().getTime()));  
 statement.setString(5, student.getAddress());  
 statement.setString(6, student.getPhone());  
 statement.setString(7, student.getEmail());  
 statement.setString(8, student.getCollege());  
 statement.setInt(9, student.getStudentId());  
 statement.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
}

（5）根据学号查询

public Student getStudentById(int studentId) {  
 try (Connection conn = DriverManager.*getConnection*(*DB\_URL*, *DB\_USERNAME*, *DB\_PASSWORD*)) {  
 String sql = "SELECT \* FROM students WHERE student\_id = ?";  
 PreparedStatement statement = conn.prepareStatement(sql);  
 statement.setInt(1, studentId);  
 ResultSet resultSet = statement.executeQuery();  
 if (resultSet.next()) {  
 Student student = new Student();  
 student.setStudentId(resultSet.getInt("student\_id"));  
 student.setName(resultSet.getString("name"));  
 student.setAge(resultSet.getInt("age"));  
 student.setGender(resultSet.getString("gender"));  
 student.setBirthdate(resultSet.getDate("birthdate"));  
 student.setAddress(resultSet.getString("address"));  
 student.setPhone(resultSet.getString("phone"));  
 student.setEmail(resultSet.getString("email"));  
 student.setCollege(resultSet.getString("college"));  
 return student;  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return null;  
}

（6）查询所有学生信息

public List<Student> getAllStudents() {  
 List<Student> students = new ArrayList<>();  
 try (Connection conn = DriverManager.*getConnection*(*DB\_URL*, *DB\_USERNAME*, *DB\_PASSWORD*)) {  
 String sql = "SELECT \* FROM students";  
 PreparedStatement statement = conn.prepareStatement(sql);  
 ResultSet resultSet = statement.executeQuery();  
 while (resultSet.next()) {  
 Student student = new Student();  
 student.setStudentId(resultSet.getInt("student\_id"));  
 student.setName(resultSet.getString("name"));  
 student.setAge(resultSet.getInt("age"));  
 student.setGender(resultSet.getString("gender"));  
 student.setBirthdate(resultSet.getDate("birthdate"));  
 student.setAddress(resultSet.getString("address"));  
 student.setPhone(resultSet.getString("phone"));  
 student.setEmail(resultSet.getString("email"));  
 student.setCollege(resultSet.getString("college"));  
 students.add(student);  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return students;  
}

（7）根据姓名、学院查询学生信息

public List<Student> searchStudents(String keyword) {  
 List<Student> students = new ArrayList<>();  
 try (Connection conn = DriverManager.*getConnection*(*DB\_URL*, *DB\_USERNAME*, *DB\_PASSWORD*)) {  
 String sql = "SELECT \* FROM students WHERE name LIKE ? OR address LIKE ? OR college LIKE ?";  
 PreparedStatement statement = conn.prepareStatement(sql);  
 statement.setString(1, "%" + keyword + "%");  
 statement.setString(2, "%" + keyword + "%");  
 statement.setString(3, "%" + keyword + "%");  
 ResultSet resultSet = statement.executeQuery();  
 while (resultSet.next()) {  
 Student student = new Student();  
 student.setStudentId(resultSet.getInt("student\_id"));  
 student.setName(resultSet.getString("name"));  
 student.setAge(resultSet.getInt("age"));  
 student.setGender(resultSet.getString("gender"));  
 student.setBirthdate(resultSet.getDate("birthdate"));  
 student.setAddress(resultSet.getString("address"));  
 student.setPhone(resultSet.getString("phone"));  
 student.setEmail(resultSet.getString("email"));  
 student.setCollege(resultSet.getString("college"));  
 students.add(student);  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return students;  
}

（8）备份学生信息

public void backupData(String filename) {  
 try (Connection conn = DriverManager.*getConnection*(*DB\_URL*, *DB\_USERNAME*, *DB\_PASSWORD*);  
 FileWriter writer = new FileWriter(filename)) {  
 String sql = "SELECT \* FROM students";  
 PreparedStatement statement = conn.prepareStatement(sql);  
 ResultSet resultSet = statement.executeQuery();  
 while (resultSet.next()) {  
 int studentId = resultSet.getInt("student\_id");  
 String name = resultSet.getString("name");  
 int age = resultSet.getInt("age");  
 String gender = resultSet.getString("gender");  
 Date birthdate = resultSet.getDate("birthdate");  
 String address = resultSet.getString("address");  
 String phone = resultSet.getString("phone");  
 String email = resultSet.getString("email");  
 String college = resultSet.getString("college");  
  
 String studentInfo = String.*format*("%d,%s,%d,%s,%s,%s,%s,%s,%s",  
 studentId, name, age, gender, birthdate, address, phone, email, college);  
  
 writer.write(studentInfo);  
 writer.write("\n");  
 }  
 } catch (SQLException | IOException e) {  
 e.printStackTrace();  
 }  
}

（9）导入学生信息

public void importData(String filename) {  
 try (Connection conn = DriverManager.*getConnection*(*DB\_URL*, *DB\_USERNAME*, *DB\_PASSWORD*);  
 BufferedReader reader = new BufferedReader(new FileReader(filename))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 String[] studentData = line.split(",");  
 if (studentData.length == 9) {  
 int studentId = Integer.*parseInt*(studentData[0]);  
 String name = studentData[1];  
 int age = Integer.*parseInt*(studentData[2]);  
 String gender = studentData[3];  
 DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");  
 java.util.Date birthdate = dateFormat.parse(studentData[4]);  
 String address = studentData[5];  
 String phone = studentData[6];  
 String email = studentData[7];  
 String college = studentData[8];  
  
 Student student = new Student(studentId, name, age, gender, birthdate, address, phone, email, college);  
 addStudent(student);  
 }  
 }  
 } catch (SQLException | IOException | ParseException e) {  
 e.printStackTrace();  
 }  
}

（10）导出学生信息

public void exportData(String filename) {  
 try (Connection conn = DriverManager.*getConnection*(*DB\_URL*, *DB\_USERNAME*, *DB\_PASSWORD*);  
 FileWriter writer = new FileWriter(filename)) {  
 List<Student> students = getAllStudents();  
 for (Student student : students) {  
 String studentInfo = String.*format*("%d,%s,%d,%s,%s,%s,%s,%s,%s",  
 student.getStudentId(), student.getName(), student.getAge(),  
 student.getGender(), student.getBirthdate(), student.getAddress(),  
 student.getPhone(), student.getEmail(), student.getCollege());  
  
 writer.write(studentInfo);  
 writer.write("\n");  
 }  
 } catch (SQLException | IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

2. Student

（1）学生信息管理getter/setter方法  
public class Student {  
 private int studentId;  
 private String name;  
 private int age;  
 private String gender;  
 private Date birthdate;  
 private String address;  
 private String phone;  
 private String email;  
 private String college;  
  
 public Student() {  
 }  
  
 public Student(int studentId, String name, int age, String gender, Date birthdate, String address, String phone, String email, String college) {  
 this.studentId = studentId;  
 this.name = name;  
 this.age = age;  
 this.gender = gender;  
 this.birthdate = birthdate;  
 this.address = address;  
 this.phone = phone;  
 this.email = email;  
 this.college = college;  
 }  
  
 public int getStudentId() {  
 return studentId;  
 }  
  
 public void setStudentId(int studentId) {  
 this.studentId = studentId;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public int getAge() {  
 return age;  
 }  
  
 public void setAge(int age) {  
 this.age = age;  
 }  
  
 public String getGender() {  
 return gender;  
 }  
  
 public void setGender(String gender) {  
 this.gender = gender;  
 }  
  
 public Date getBirthdate() {  
 return birthdate;  
 }  
  
 public void setBirthdate(Date birthdate) {  
 this.birthdate = birthdate;  
 }  
  
 public String getAddress() {  
 return address;  
 }  
  
 public void setAddress(String address) {  
 this.address = address;  
 }  
  
 public String getPhone() {  
 return phone;  
 }  
  
 public void setPhone(String phone) {  
 this.phone = phone;  
 }  
  
 public String getEmail() {  
 return email;  
 }  
  
 public void setEmail(String email) {  
 this.email = email;  
 }  
  
 public String getCollege() {  
 return college;  
 }  
  
 public void setCollege(String college) {  
 this.college = college;  
 }  
  
 @Override  
 public String toString() {  
 return "Student{" +  
 "studentId=" + studentId +  
 ", name='" + name + '\'' +  
 ", age=" + age +  
 ", gender='" + gender + '\'' +  
 ", birthdate=" + birthdate +  
 ", address='" + address + '\'' +  
 ", phone='" + phone + '\'' +  
 ", email='" + email + '\'' +  
 ", college='" + college + '\'' +  
 '}';  
 }  
}

3.Main

（1）菜单实现

public class Main {  
 public static void main(String[] args) {  
 StudentManagementSystem system = new StudentManagementSystem();  
 Scanner scanner = new Scanner(System.*in*);  
  
 while (true) {  
 System.*out*.println("请选择要执行的操作:");  
 System.*out*.println("1. 添加学生信息");  
 System.*out*.println("2. 查询学生信息");  
 System.*out*.println("3. 更新学生信息");  
 System.*out*.println("4. 查询所有学生信息");  
 System.*out*.println("5. 根据关键字查询学生信息");  
 System.*out*.println("6. 删除学生信息");  
 System.*out*.println("7. 备份学生信息");  
 System.*out*.println("8. 导入学生信息");  
 System.*out*.println("9. 导出学生信息");  
 System.*out*.println("0. 退出程序");  
  
 int choice = scanner.nextInt();  
 scanner.nextLine();  
  
 switch (choice) {  
 case 1:  
 *// 添加学生信息* System.*out*.println("请输入学生信息：");  
 System.*out*.print("学号: ");  
 int studentId = scanner.nextInt();  
 scanner.nextLine();  
 System.*out*.print("姓名: ");  
 String name = scanner.nextLine();  
 System.*out*.print("年龄: ");  
 int age = scanner.nextInt();  
 scanner.nextLine();  
 System.*out*.print("性别: ");  
 String gender = scanner.nextLine();  
 System.*out*.print("出生日期 (yyyy-MM-dd): ");  
 String birthdateStr = scanner.nextLine();  
 Date birthdate = *parseDate*(birthdateStr);  
 System.*out*.print("地址: ");  
 String address = scanner.nextLine();  
 System.*out*.print("电话: ");  
 String phone = scanner.nextLine();  
 System.*out*.print("电子邮件: ");  
 String email = scanner.nextLine();  
 System.*out*.print("学院: ");  
 String college = scanner.nextLine();  
  
 Student student = new Student(studentId, name, age, gender, birthdate, address, phone, email, college);  
 system.addStudent(student);  
 System.*out*.println("学生信息添加成功。");  
 break;  
 case 2:  
 *// 查询学生信息* System.*out*.print("请输入要查询的学生的学号: ");  
 int searchId = scanner.nextInt();  
 Student retrievedStudent = system.getStudentById(searchId);  
 if (retrievedStudent != null) {  
 System.*out*.println("学生信息:");  
 System.*out*.println("学号: " + retrievedStudent.getStudentId());  
 System.*out*.println("姓名: " + retrievedStudent.getName());  
 System.*out*.println("年龄: " + retrievedStudent.getAge());  
 System.*out*.println("性别: " + retrievedStudent.getGender());  
 System.*out*.println("出生日期: " + retrievedStudent.getBirthdate());  
 System.*out*.println("地址: " + retrievedStudent.getAddress());  
 System.*out*.println("电话: " + retrievedStudent.getPhone());  
 System.*out*.println("电子邮件: " + retrievedStudent.getEmail());  
 System.*out*.println("学院: " + retrievedStudent.getCollege());  
 } else {  
 System.*out*.println("未找到学生信息。");  
 }  
 break;  
 case 3:  
 *// 更新学生信息* System.*out*.print("请输入要更新的学生的学号: ");  
 int updateId = scanner.nextInt();  
 scanner.nextLine();  
 Student studentToUpdate = system.getStudentById(updateId);  
 if (studentToUpdate != null) {  
 System.*out*.println("请输入更新后的学生信息：");  
 System.*out*.print("姓名: ");  
 String updatedName = scanner.nextLine();  
 System.*out*.print("年龄: ");  
 int updatedAge = scanner.nextInt();  
 scanner.nextLine();  
 System.*out*.print("性别: ");  
 String updatedGender = scanner.nextLine();  
 System.*out*.print("出生日期 (yyyy-MM-dd): ");  
 String updatedBirthdateStr = scanner.nextLine();  
 Date updatedBirthdate = *parseDate*(updatedBirthdateStr);  
 System.*out*.print("地址: ");  
 String updatedAddress = scanner.nextLine();  
 System.*out*.print("电话: ");  
 String updatedPhone = scanner.nextLine();  
 System.*out*.print("电子邮件: ");  
 String updatedEmail = scanner.nextLine();  
 System.*out*.print("学院: ");  
 String updatedCollege = scanner.nextLine();  
  
 studentToUpdate.setName(updatedName);  
 studentToUpdate.setAge(updatedAge);  
 studentToUpdate.setGender(updatedGender);  
 studentToUpdate.setBirthdate(updatedBirthdate);  
 studentToUpdate.setAddress(updatedAddress);  
 studentToUpdate.setPhone(updatedPhone);  
 studentToUpdate.setEmail(updatedEmail);  
 studentToUpdate.setCollege(updatedCollege);  
  
 system.updateStudent(studentToUpdate);  
 System.*out*.println("学生信息更新成功。");  
 } else {  
 System.*out*.println("未找到要更新的学生信息。");  
 }  
 break;  
 case 4:  
 *// 查询所有学生信息* List<Student> allStudents = system.getAllStudents();  
 System.*out*.println("所有学生信息:");  
 for (Student s : allStudents) {  
 System.*out*.println("学号: " + s.getStudentId() + ", 姓名: " + s.getName());  
 }  
 break;  
 case 5:  
 *// 根据关键字查询学生信息* System.*out*.print("请输入要查询的关键字: ");  
 String keyword = scanner.nextLine();  
 List<Student> searchResults = system.searchStudents(keyword);  
 System.*out*.println("查询结果:");  
 for (Student s : searchResults) {  
 System.*out*.println("学号: " + s.getStudentId() + ", 姓名: " + s.getName());  
 }  
 break;  
 case 6:  
 *// 删除学生信息* System.*out*.print("请输入要删除的学生的学号: ");  
 int deleteId = scanner.nextInt();  
 scanner.nextLine();  
 boolean deleteSuccess = system.deleteStudent(deleteId);  
 if (!deleteSuccess) {  
 System.*out*.println("未找到要删除的学生信息。");  
 } else {  
 System.*out*.println("学生信息删除成功。");  
 }  
 break;  
 case 7:  
 *// 备份学生信息* System.*out*.print("请输入备份文件名: ");  
 String backupFilename = scanner.nextLine();  
 system.backupData(backupFilename);  
 System.*out*.println("学生信息备份成功。");  
 break;  
 case 8:  
 *// 导入学生信息* System.*out*.print("请输入要导入的文件名: ");  
 String importFilename = scanner.nextLine();  
 system.importData(importFilename);  
 System.*out*.println("学生信息导入成功。");  
 break;  
 case 9:  
 *// 导出学生信息* System.*out*.print("请输入导出文件名: ");  
 String exportFilename = scanner.nextLine();  
 system.exportData(exportFilename);  
 System.*out*.println("学生信息导出成功。");  
 break;  
 case 0:  
 *// 退出程序* System.*out*.println("程序已退出。");  
 System.*exit*(0);  
 break;  
 default:  
 System.*out*.println("无效的选项，请重新选择。");  
 break;  
 }  
  
 System.*out*.println(); *// 打印空行，以提升可读性* }  
 }  
  
 *// 解析日期字符串为Date对象* private static Date parseDate(String dateStr) {  
 try {  
 SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");  
 return dateFormat.parse(dateStr);  
 } catch (ParseException e) {  
 e.printStackTrace();  
 return null;  
 }  
 }  
}