数据库系统课程基础实践报告

篇章3：使用数据库持久层操作数据库篇

1. 报告基本信息

- 姓名：杨郭霆
- 班级：网工20
- 学号：200602314

2. 围绕指定案例编写系统

【任务目的】

1. 理解关系型数据库与面向对象程序之间的转换方法

学习关系型数据库操作时转换为面向对象程序之间的必要性和方法。

2. 掌握使用DAO技术构建朴素数据库持久层的方法

结合提供的学习资料，编写指定数据库的持久层框架。

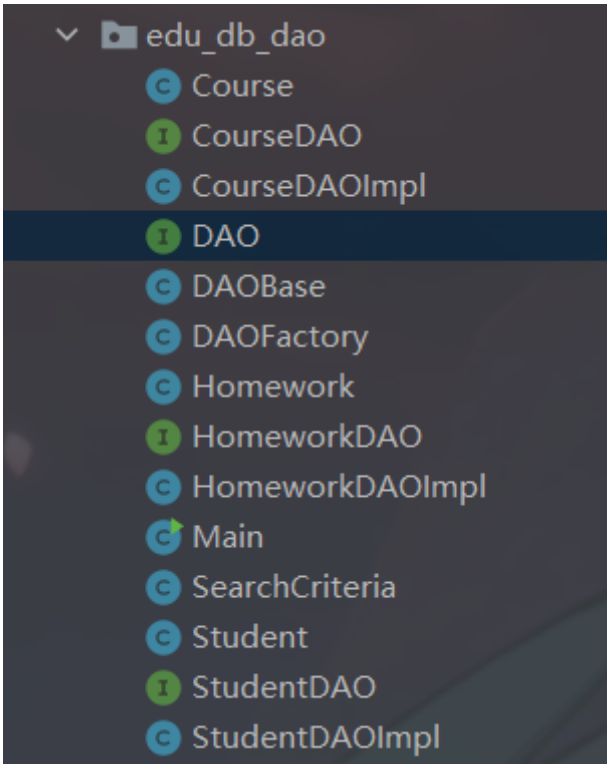
【任务环境】

- 硬件环境：Windows10
- 软件环境：SQL Server 2019/Java17

【任务步骤完成情况】

1. 根据已经建立的EDUCATION数据库，使用DAO过程建立数据持久层，编写访问数据的实体类、接口、DAO对象和工厂类。

源代码（包含关键代码的解释）：



如图为本项目包的全部代码文件

***Course*组**

```
//Course
package edu_db_dao;

public class Course {
    private String course_no;
    private String course_name;
    private String course_credit;
    private String course_time;
    private String course_teacher;

    public Course(String course_no, String course_name, String course_credit, String
course_time, String course_teacher) {
        this.course_no = course_no;
        this.course_name = course_name;
        this.course_credit = course_credit;
        this.course_time = course_time;
        this.course_teacher = course_teacher;
    }

    public Course() {

    }

    public String getCourse_no() {
        return course_no;
    }

    public void setCourse_no(String course_no) {
        this.course_no = course_no;
    }

    public String getCourse_name() {
        return course_name;
    }
```

```
    public void setCourse_name(String course_name) {
        this.course_name = course_name;
    }

    public String getCourse_credit() {
        return course_credit;
    }

    public void setCourse_credit(String course_credit) {
        this.course_credit = course_credit;
    }

    public String getCourse_time() {
        return course_time;
    }

    public void setCourse_time(String course_time) {
        this.course_time = course_time;
    }

    public String getCourse_teacher() {
        return course_teacher;
    }

    public void setCourse_teacher(String course_teacher) {
        this.course_teacher = course_teacher;
    }

    @Override
    public String toString() {
        return "Course{" +
                "course_no='" + course_no + '\'' +
                ", course_name='" + course_name + '\'' +
                '}';
    }
}
```

```
//CouresDAO
package edu_db_dao;

import java.util.List;

public interface CourseDAO {
    void addCourse(Course course);
    void updateCourse(Course course);
    void deleteCourse(String course_no);
    Course getCourse(String course_no);
    List<Course> findCourses(SearchCriteria searchCriteria);
}
```

```
//CourseDAOImpl
package edu_db_dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
```

```
import java.util.List;

public class CourseDAOImpl extends DAOBase implements CourseDAO{
    private static final String COURSE_INSERT_SQL = "INSERT INTO C(cno,cn,cs,ct,teacher)
VALUES(?,?,?,?,?) ";

    @Override
    public void addCourse(Course course) {
        Connection con = null;
        try{
            con = getConnection();
            PreparedStatement psmt = con.prepareStatement(COURSE_INSERT_SQL);
            psmt.setString(1, course.getCourse_no());
            psmt.setString(2, course.getCourse_name());
            psmt.setString(3, course.getCourse_credit());
            psmt.setString(4, course.getCourse_time());
            psmt.setString(5, course.getCourse_teacher());
            psmt.executeUpdate();
            psmt.close();
        }catch(Exception e){
            e.printStackTrace();
        }finally {
            try{
                con.close();
            }catch (Exception e){
                e.printStackTrace();
            }
        }
    }

    @Override
    public void updateCourse(Course course) {

    }

    @Override
    public void deleteCourse(String course_no) {

    }

    private static final String COURSE_SELECT_SQL = "SELECT * FROM C WHERE cno=?";

    @Override
    public Course getCourse(String course_no) {
        Connection con = null;
        Course course = new Course();
        try{
            con = getConnection();
            PreparedStatement psmt = con.prepareStatement(COURSE_SELECT_SQL);
            psmt.setString(1, course_no);
            ResultSet rs = psmt.executeQuery();
            while (rs.next()){
                course.setCourse_no(rs.getString("cno"));
                course.setCourse_name(rs.getString("cn"));
                course.setCourse_credit(rs.getString("cs"));
                course.setCourse_time(rs.getString("ct"));
                course.setCourse_teacher(rs.getString("teacher"));
            }
            psmt.close();

        }catch(Exception e){
```

```
            e.printStackTrace();
        }finally {
            try{
                con.close();
            }catch (Exception e){
                e.printStackTrace();
            }
        }
        return course;
    }

    @Override
    public List<Course> findCourses(SearchCriteria searchCriteria) {
        return null;
    }
}
```

***DAO*组**

```
//DAO
package edu_db_dao;
import java.sql.Connection;

public interface DAO {
    Connection getConnection();
}
```

```
//DAOBase
package edu_db_dao;

import java.sql.Connection;
import java.sql.DriverManager;

public class DAOBase implements DAO{
    @Override
    public Connection getConnection() {
        String URL="jdbc:sqlserver://localhost:1433; DatabaseName=education";
        Connection con = null;
        try{
            Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
            con = DriverManager.getConnection(URL,"stu","123");

        }catch (Exception ex){
            ex.printStackTrace();
        }
        return con;
    }
}
```

```
//DAOFactory
package edu_db_dao;

public class DAOFactory {
    private static DAOFactory daoFactory;
    static {
```

```
        daoFactory = new DAOFactory();
    }
    private DAOFactory(){

    }

    public static DAOFactory getInstance(){
        return daoFactory;
    }

    public static StudentDAO getStudentDAO(){
        return new StudentDAOImpl();
    }
}
```

***Homewo*组**

```
//Homework
package edu_db_dao;

public class Homework {
    private Student student;
    private Course course;
    private float score1;
    private float score2;
    private float score3;

}
```

```
//HomeworkDAO
package edu_db_dao;

import java.util.List;

public interface HomeworkDAO {
    void addHomework(Homework homework);
    void updateHomework(Homework homework);
    void deleteHomework(Student student, Course course);
    Homework getHomework(Student student, Course course);
    List<Homework> findHomeworks(SearchCriteria searchCriteria);
}
```

```
//HomeworkDAOImpl
package edu_db_dao;

import java.util.List;

public class HomeworkDAOImpl extends DAOBase implements HomeworkDAO{
    private static final String HOMEWORK_INSERT_SQL = "INSERT INTO SW(cno,sno,s1,s2,s3)
VALUES(?,?,?,?,?) ";

    @Override
    public void addHomework(Homework homework) {

    }
```

```
    @Override
    public void updateHomework(Homework homework) {

    }

    @Override
    public void deleteHomework(Student student, Course course) {

    }

    @Override
    public Homework getHomework(Student student, Course course) {
        return null;
    }

    @Override
    public List<Homework> findHomeworks(SearchCriteria searchCriteria) {
        return null;
    }
}
```

***Student*组**

```
//Student
package edu_db_dao;

public class Student {
    private String student_no;
    private String student_name;
    private String student_sex;
    private String student_class;
    private String student_birthday;
    private String student_tel;

    public Student(String student_no, String student_name, String student_sex, String
student_class, String student_birthday, String student_tel) {
        this.student_no = student_no;
        this.student_name = student_name;
        this.student_sex = student_sex;
        this.student_class = student_class;
        this.student_birthday = student_birthday;
        this.student_tel = student_tel;
    }

    public Student() {

    }

//    public Student() {
//
//    }

    public String getStudent_no() {
        return student_no;
    }

    public void setStudent_no(String student_no) {
        this.student_no = student_no;
```

```
    }

    public String getStudent_name() {
        return student_name;
    }

    public void setStudent_name(String student_name) {
        this.student_name = student_name;
    }

    public String getStudent_sex() {
        return student_sex;
    }

    public void setStudent_sex(String student_sex) {
        this.student_sex = student_sex;
    }

    public String getStudent_class() {
        return student_class;
    }

    public void setStudent_class(String student_class) {
        this.student_class = student_class;
    }

    public String getStudent_birthday() {
        return student_birthday;
    }

    public void setStudent_birthday(String student_birthday) {
        this.student_birthday = student_birthday;
    }

    public String getStudent_tel() {
        return student_tel;
    }

    public void setStudent_tel(String student_tel) {
        this.student_tel = student_tel;
    }

    @Override
    public String toString() {
        return "Student{" +
                "student_no='" + student_no + '\'' +
                ", student_name='" + student_name + '\'' +
                '}';
    }
}
```

```java
//StudentDAO
package edu_db_dao;

import java.util.List;

public interface StudentDAO {
    void addStudent(Student student);
    void updateStudent(Student student);
    void deleteStudent(String student_no);
    Student getStudent(String student_no);
    List<Student> findStudents(SearchCriteria searchCriteria);
}
```

```java
//StudentDAOImpl
package edu_db_dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.List;

public class StudentDAOImpl extends DAOBase implements StudentDAO{
    private static final String STUDENT_INSERT_SQL = "INSERT INTO
S(sno,sn,sex,class,birthday,tel) VALUES(?,?,?,?,?,?) ";

    @Override
    public void addStudent(Student student) {
        Connection con = null;
        try{
            con = getConnection();
            PreparedStatement psmt = con.prepareStatement(STUDENT_INSERT_SQL);
            psmt.setString(1, student.getStudent_no());
            psmt.setString(2, student.getStudent_name());
            psmt.setString(3, student.getStudent_sex());
            psmt.setString(4, student.getStudent_class());
            psmt.setString(5, student.getStudent_birthday());
            psmt.setString(6, student.getStudent_tel());
            psmt.executeUpdate();
            psmt.close();
        }catch(Exception e){
            e.printStackTrace();
        }finally {
            try{
                con.close();
            }catch (Exception e){
                e.printStackTrace();
            }
        }
    }

    @Override
    public void updateStudent(Student student) {

    }

    @Override
    public void deleteStudent(String student_no) {

    }
```

```
    private static final String STUDENT_SELECT_SQL = "SELECT sno,sn,sex,class,birthday,tel FROM
S WHERE sno=?";

    @Override
    public Student getStudent(String student_no) {
        Connection con = null;
        Student student = new Student();
        try{
            con = getConnection();
            PreparedStatement psmt = con.prepareStatement(STUDENT_SELECT_SQL);
            psmt.setString(1, student_no);
            ResultSet rs = psmt.executeQuery();
            while (rs.next()){
                student.setStudent_no(rs.getString("sno"));
                student.setStudent_name(rs.getString("sn"));
                student.setStudent_sex(rs.getString("sex"));
                student.setStudent_class(rs.getString("class"));
                student.setStudent_birthday(rs.getString("birthday"));
                student.setStudent_tel(rs.getString("tel"));
            }
            psmt.close();

        }catch(Exception e){
            e.printStackTrace();
        }finally {
            try{
                con.close();
            }catch (Exception e){
                e.printStackTrace();
            }
        }
        return student;
    }

    @Override
    public List<Student> findStudents(SearchCriteria searchCriteria) {
        return null;
    }
}
```

2. 建立测试类，测试Student实体类和Homework实体类中的主要方法。

源代码（包含关键代码的解释）：

Main

```
package edu_db_dao;

public class Main {
    public static void main(String[] args){
        Student student = new Student(); //定义新的学生信息
        student.setStudent_no("S0010");
        student.setStudent_name("张三");
        student.setStudent_sex("男");
        student.setStudent_class("信息05");
        student.setStudent_birthday("2001-1-1");
        student.setStudent_tel("17777777777");
```

```
        DAOFactory.getInstance().getStudentDAO().addStudent(student);  //调用插入学生数据的方法

    }
}
```

【任务总结】

1. DAO模型中关键组成要素。
   - DAO接口：把对数据库的所有操作定义为抽象方法，可以提供多种实现
   - DAO实现类：针对不同数据库给出DAO接口定义方法的具体实现
   - 实体类：用于存放与传输对象数据
   - 数据库连接和关闭工具类：避免了数据库连接和关闭代码的重复使用，方便修改
2. DAO模型中的设计模式。
   - 数据库连接类：连接数据库
   - 实体类：封装数据
   - DAO接口：定义对数据库中表的原子操作（增、删、改、查）
   - DAO实现类：定义对数据库中表的原子操作（增、删、改、查）
   - DAO工厂类：生产各种DAO实例