实验 4 基于嵌入 SQL 的综合应用编程（4 学时）

一、实验目的

本次实验的主要目的是掌握嵌入 SQL 及主高级语言，学会使用嵌入 SQL 对数据库进行增、删、改、备份的方法。

二、实验要求

1．要求学生独立完成实验内容，画出E-R图及程序功能图；

2．按照实验步骤完成实验后，撰写报告内容，并对操作结果进行截图，写出主要关键程序代码。

三、实验内容、实验结果与主要程序代码

基于实验一的三个表采用嵌入式 SQL 语言及主语言编程实现数据库的录入、修改、删除和备份等管理功能，并能实现基于学号查询显示学生基本信息、课程名、成绩信息。

（一）画出 E-R 图及程序功能分析设计图

E-R 图见图 4-1，程序功能分析设计图见图 4-2。

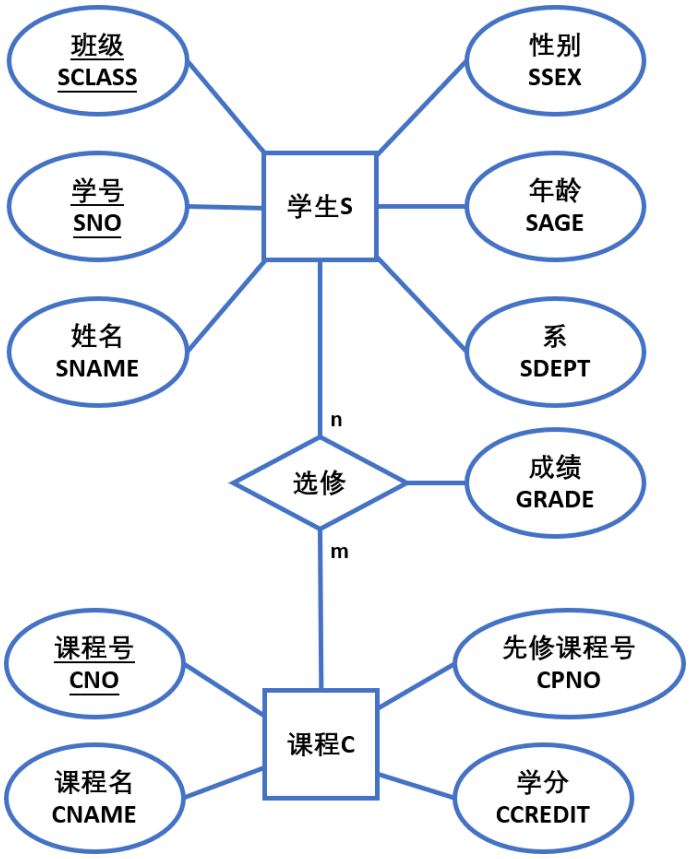


图 4-1 E-R 图

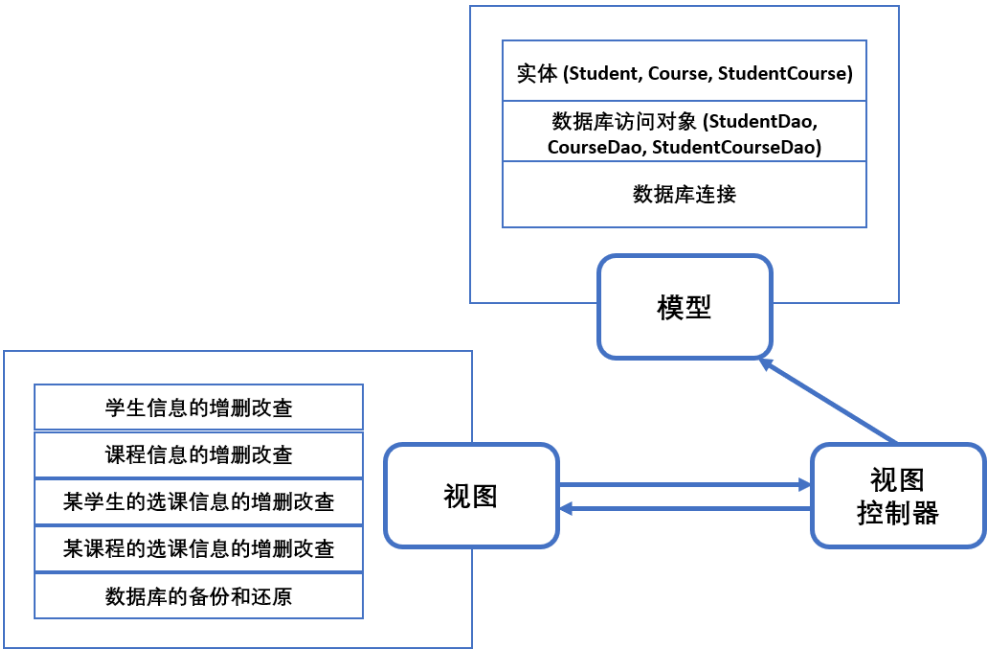
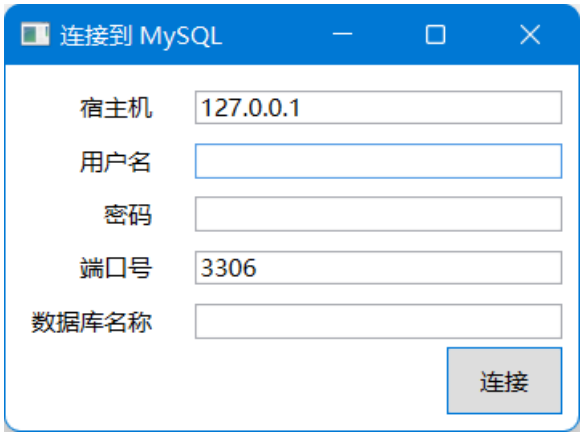


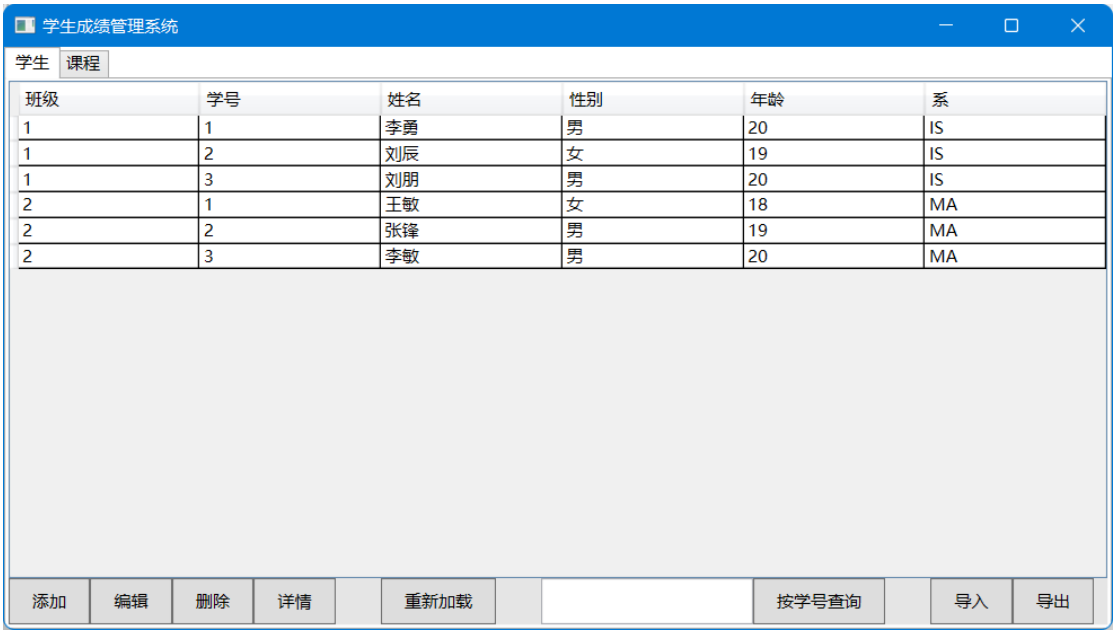
图 4-2 程序功能分析设计图

（二）功能实现界面图及主要程序代码（要有注释）

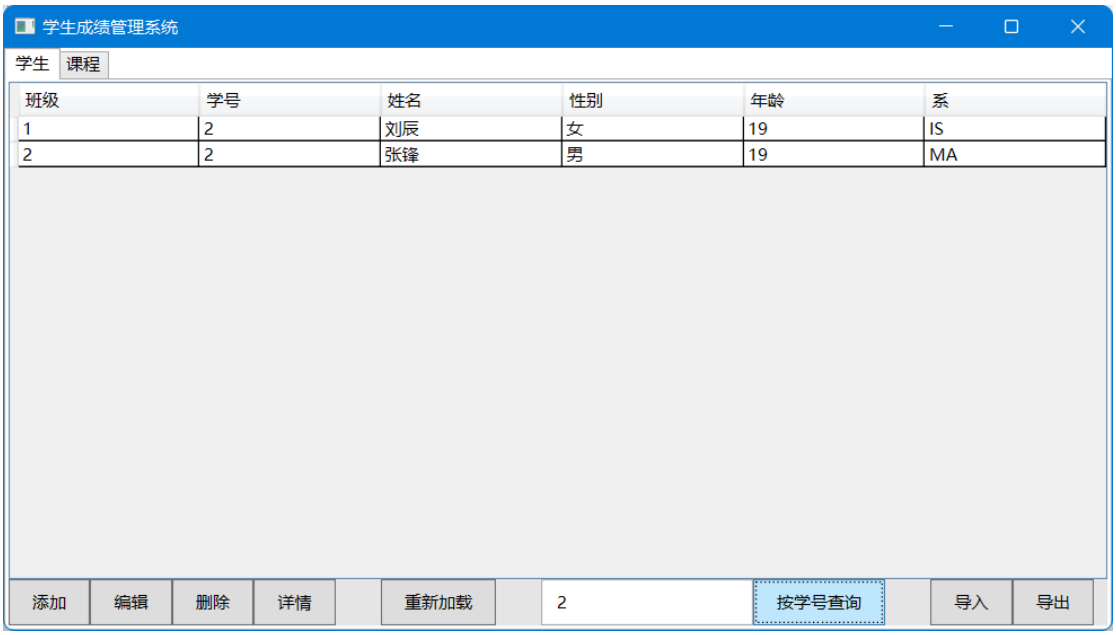
功能实现界面图见图 4-3。



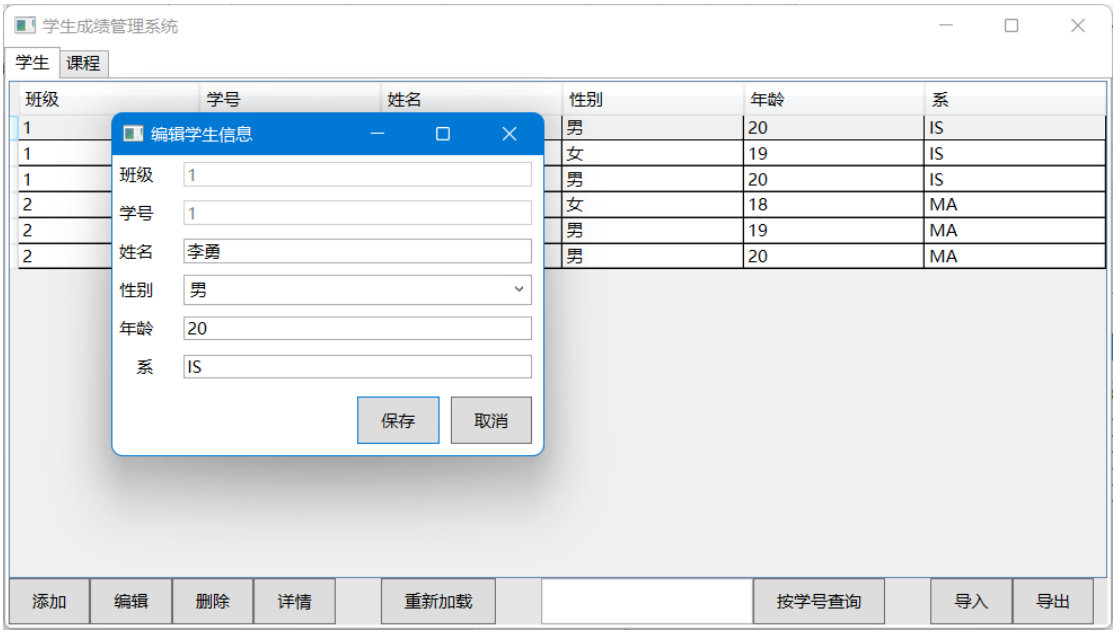
a) 连接界面



b） 查询全部学生

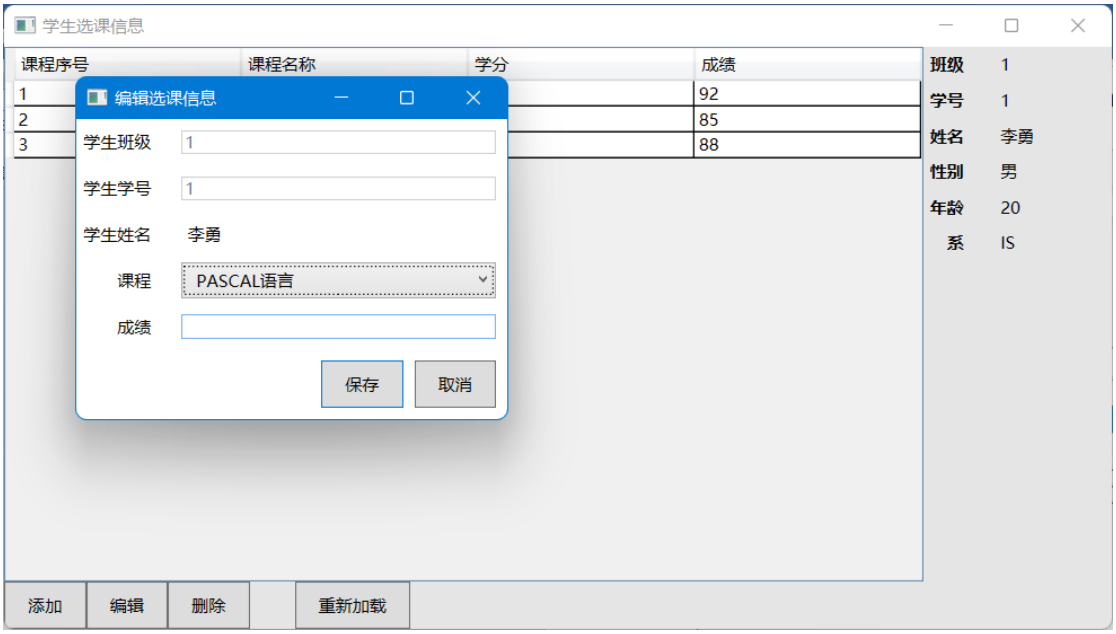


c） 按学号查询学生



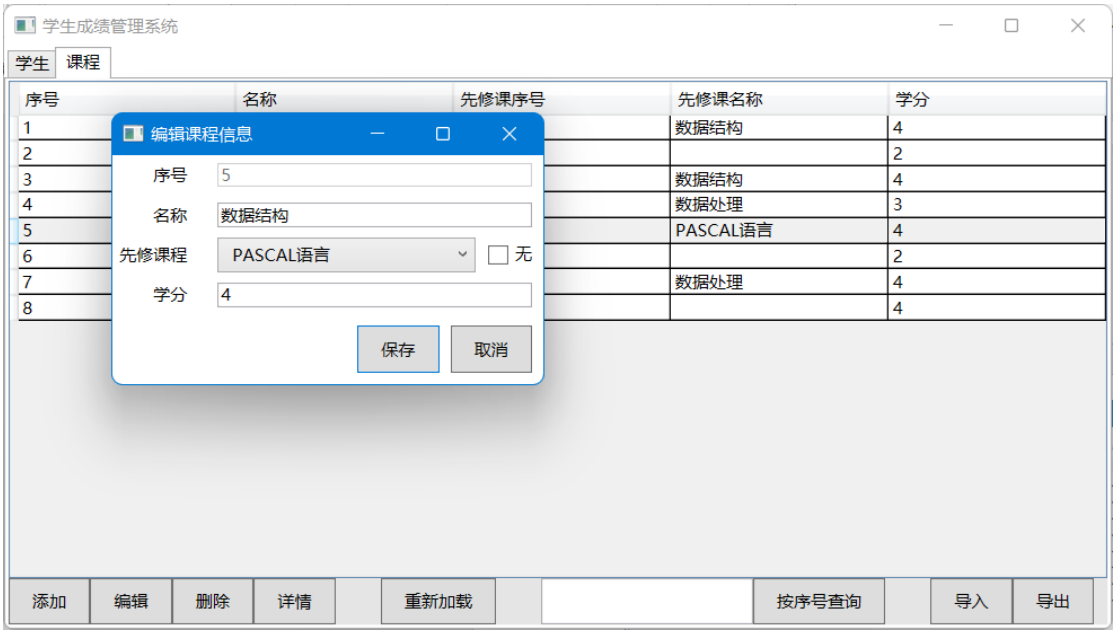
d） 编辑学生信息

e） 查询某学生选课信息

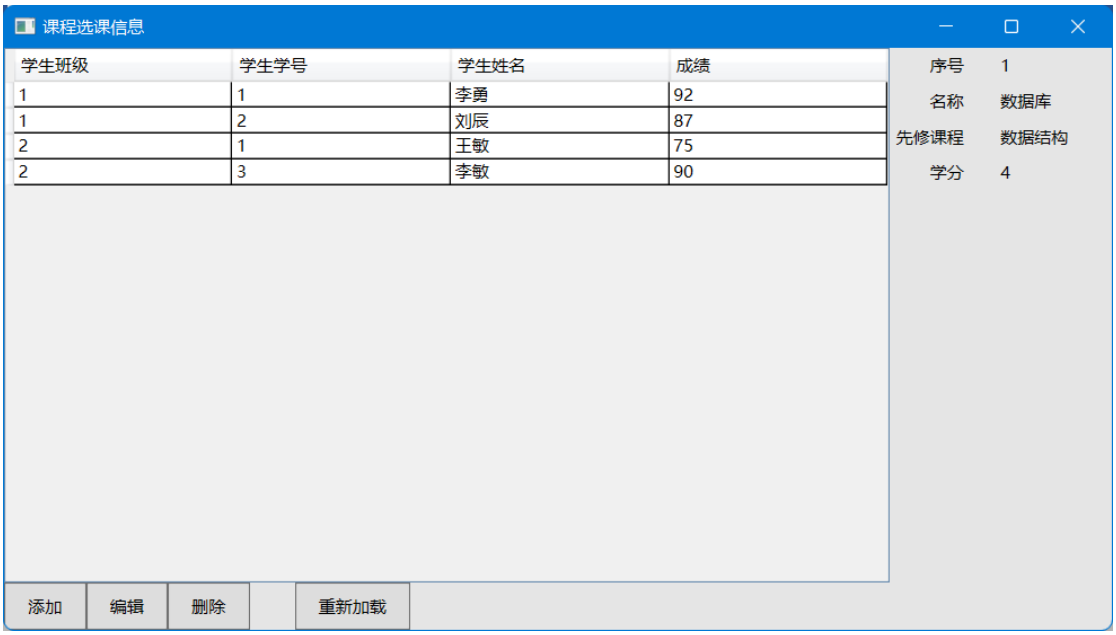


f） 添加学生选课信息

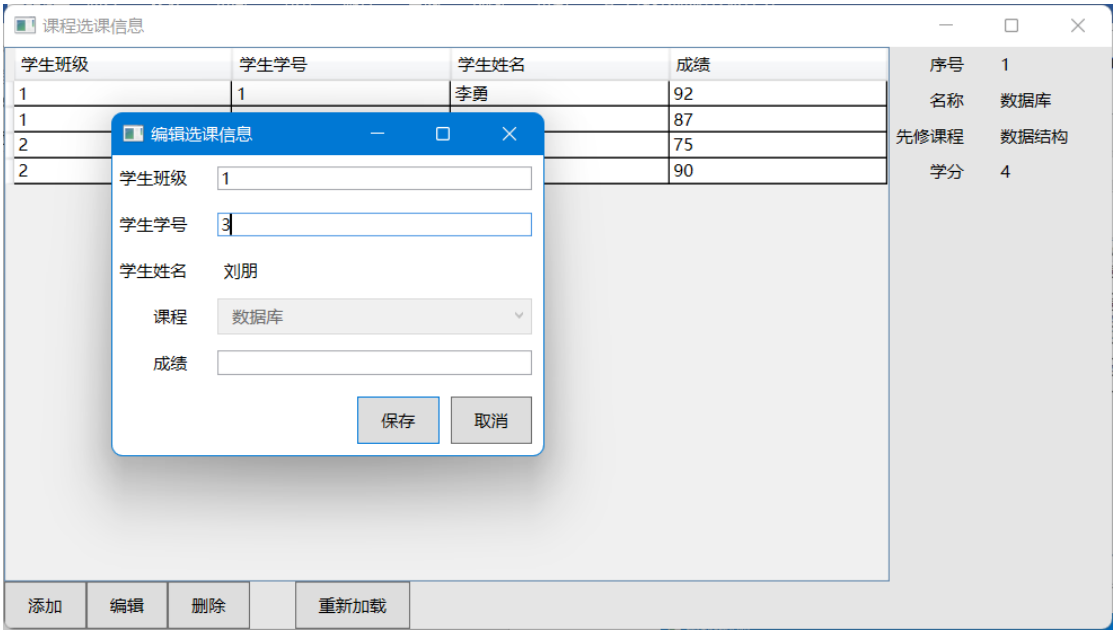
g） 查询所有课程



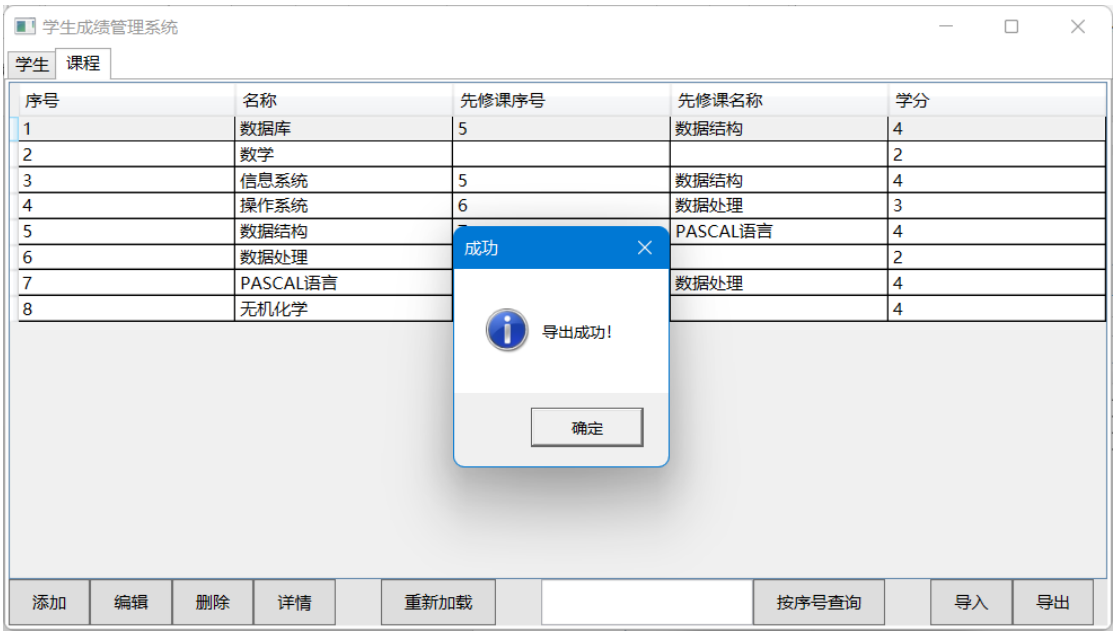
h） 编辑课程信息



i） 查询某课程选课信息



j） 添加课程选课信息



k） 数据库备份

图 4-3 功能实现界面图

主要程序代码：

```
namespace EveryDatabaseTeacherLovesStudentSystem.Model
{
  public class MyDatabase
  {
    internal MySqlConnection conn = null;
```

```
    public StudentDao StudentDao { get; private set; }
    public CourseDao CourseDao { get; private set; }
    public StudentCourseDao StudentCourseDao { get; private set; }

    private string server, user, passwd, database;
    private int port;

    // 打开数据库连接
    public async Task OpenConnectionAsync(string server, int port,
string user, string passwd, string database)
    {
      this.server = server;
      this.port = port;
      this.user = user;
      this.passwd = passwd;
      this.database = database;

      conn = new MySqlConnection
      {
        ConnectionString = string.Format("server={0}; port={1};
user={2}; password={3};", server, port, user, passwd, database)
      };
      await conn.OpenAsync();

      StudentDao = new StudentDao(conn);
      CourseDao = new CourseDao(conn);
      StudentCourseDao = new StudentCourseDao(conn);

      // 尝试创建数据库，若已存在则忽略
      try
      {
        string sql = "CREATE DATABASE " + database;
        MySqlCommand cmd = new MySqlCommand(sql, conn);
        await cmd.ExecuteNonQueryAsync();
      }
      catch (Exception)
      {
        // pass
      }

      await conn.ChangeDatabaseAsync(database);

      // 尝试创建表，若已存在则忽略
      await StudentDao.InitializeAsync();
```

```
        await CourseDao.InitializeAsync();
        await StudentCourseDao.InitializeAsync();
    }

    // 导入数据库
    public void Import(string fileName, string mySqlFileName =
"mysql")
    {
        using FileStream stream = new FileStream(fileName,
FileMode.Open, FileAccess.Read);
        using StreamReader reader = new StreamReader(stream);

        using var process = new Process();
        process.StartInfo.FileName = "cmd.exe";
        process.StartInfo.UseShellExecute = false;
        process.StartInfo.CreateNoWindow = true;
        process.StartInfo.RedirectStandardInput = true;
        process.Start();

        process.StandardInput.WriteLine(string.Format("\"{0}\" -u{1} -
p{2} {3} < \"{4}\"", mySqlFileName, user, passwd, database,
fileName));
        process.StandardInput.WriteLine("exit");

        process.WaitForExit();
        process.Close();
    }

    // 导出数据库
    public void Export(string fileName, string mySqlDumpFileName =
"mysqldump")
    {
        using FileStream stream = new FileStream(fileName,
FileMode.Create, FileAccess.Write);

        using var process = new Process();
        process.StartInfo.FileName = mySqlDumpFileName;
        process.StartInfo.Arguments = string.Format("-u{0} -p{1} {2}",
user, passwd, database);
        process.StartInfo.UseShellExecute = false;
        process.StartInfo.CreateNoWindow = true;
        process.StartInfo.RedirectStandardOutput = true;
        process.Start();
```

```
    process.StandardOutput.BaseStream.CopyTo(stream);

    process.WaitForExit();
    process.Close();
  }
}

// 数据库访问对象（Database Access Object）抽象基类
public abstract class Dao
{
  protected MySqlConnection conn;
  protected abstract string CreateTableSql { get; }

  public Dao(MySqlConnection conn)
  {
    this.conn = conn;
  }

  internal async Task InitializeAsync()
  {
    // 尝试创建表，若表已存在则忽略
    try
    {
      MySqlCommand cmd = new MySqlCommand(CreateTableSql, conn);
      await cmd.ExecuteNonQueryAsync();
    }
    catch (MySqlException)
    {
      // pass
    }
  }
}

// 用于访问学生表的 DAO
public class StudentDao : Dao
{
  // 建表语句
  protected override string CreateTableSql =>
    "CREATE TABLE S(" +
    "SCLASS INT, " +
    "SNO INT, " +
    "SNAME VARCHAR(10), " +
    "SSEX VARCHAR(2), " +
    "SAGE INT, " +
```

```
      "SDEPT CHAR(2), " +
      "PRIMARY KEY(SCLASS, SNO)" +
      ");";

    public StudentDao(MySqlConnection conn) : base(conn) { }

    // 查询所有行
    public async Task<IEnumerable<Student>> GetAllAsync()
    {
      string sql = "SELECT SCLASS, SNO, SNAME, SSEX, SAGE, SDEPT FROM
S;";
      MySqlCommand cmd = new MySqlCommand(sql, conn);
      using DbDataReader reader = await cmd.ExecuteReaderAsync();

      LinkedList<Student> result = new LinkedList<Student>();
      while (await reader.ReadAsync())
      {
        result.AddLast(new Student
        (
          reader.GetInt32(0),
          reader.GetInt32(1),
          reader.GetString(2),
          reader.GetString(3),
          reader.GetInt32(4),
          reader.GetString(5)
        ));
      }
      return result;
    }

    // 按学号（sno）查询
    public async Task<IEnumerable<Student>> GetByNumberAsync(int
number)
    {
      string sql = "SELECT SCLASS, SNO, SNAME, SSEX, SAGE, SDEPT FROM
S WHERE SNO = @number;";
      MySqlCommand cmd = new MySqlCommand(sql, conn);
      cmd.Parameters.AddWithValue("number", number);
      using DbDataReader reader = await cmd.ExecuteReaderAsync();

      LinkedList<Student> result = new LinkedList<Student>();
      while (await reader.ReadAsync())
      {
        result.AddLast(new Student
```

```
        (
          reader.GetInt32(0),
          reader.GetInt32(1),
          reader.GetString(2),
          reader.GetString(3),
          reader.GetInt32(4),
          reader.GetString(5)
        ));
      }
      return result;
    }

    // 按班级（sclass）与学号（sno）查询
    public async Task<Student> GetOneByClsAndNumberAsync(int cls, int
number)
    {
      string sql = "SELECT SCLASS, SNO, SNAME, SSEX, SAGE, SDEPT FROM
S WHERE SCLASS = @cls AND SNO = @number;";
      MySqlCommand cmd = new MySqlCommand(sql, conn);
      cmd.Parameters.AddWithValue("cls", cls);
      cmd.Parameters.AddWithValue("number", number);
      using DbDataReader reader = await cmd.ExecuteReaderAsync();

      if (await reader.ReadAsync())
      {
        return new Student
        (
          reader.GetInt32(0),
          reader.GetInt32(1),
          reader.GetString(2),
          reader.GetString(3),
          reader.GetInt32(4),
          reader.GetString(5)
        );
      }
      else
      {
        return null;
      }
    }

    // 插入一行
    public async Task InsertOneAsync(Student stu)
    {
```

```
      string sql = "INSERT INTO S(SCLASS, SNO, SNAME, SSEX, SAGE,
SDEPT) VALUES (@cls, @number, @name, @sex, @age, @dept);";
      MySqlCommand cmd = new MySqlCommand(sql, conn);
      cmd.Parameters.AddWithValue("cls", stu.Cls);
      cmd.Parameters.AddWithValue("number", stu.Number);
      cmd.Parameters.AddWithValue("name", stu.Name);
      cmd.Parameters.AddWithValue("sex", stu.Sex);
      cmd.Parameters.AddWithValue("age", stu.Age);
      cmd.Parameters.AddWithValue("dept", stu.Dept);

      await cmd.ExecuteNonQueryAsync();
    }

    // 更新一行
    public async Task UpdateOneAsync(Student stu)
    {
      string sql = "UPDATE S SET SNAME = @name, SSEX = @sex, SAGE =
@age, SDEPT = @dept WHERE SCLASS = @cls AND SNO = @number;";
      MySqlCommand cmd = new MySqlCommand(sql, conn);
      cmd.Parameters.AddWithValue("cls", stu.Cls);
      cmd.Parameters.AddWithValue("number", stu.Number);
      cmd.Parameters.AddWithValue("name", stu.Name);
      cmd.Parameters.AddWithValue("sex", stu.Sex);
      cmd.Parameters.AddWithValue("age", stu.Age);
      cmd.Parameters.AddWithValue("dept", stu.Dept);

      await cmd.ExecuteNonQueryAsync();
    }

    // 删除一行
    public async Task DeleteOneAsync(Student stu)
    {
      string sql = "DELETE FROM S WHERE S.SCLASS = @stuCls AND S.SNO
= @stuNumber;";
      MySqlCommand cmd = new MySqlCommand(sql, conn);
      cmd.Parameters.AddWithValue("stuCls", stu.Cls);
      cmd.Parameters.AddWithValue("stuNumber", stu.Number);

      await cmd.ExecuteNonQueryAsync();
    }
  }

  // 用于访问课程表的 DAO
  public class CourseDao : Dao
```

```
  {
    // 建表语句
    protected override string CreateTableSql =>
      "CREATE TABLE C(" +
      "CNO INT PRIMARY KEY, " +
      "CNAME VARCHAR(20), " +
      "CPNO INT, " +
      "CCREDIT INT" +
      ");";

    public CourseDao(MySqlConnection conn) : base(conn) { }

    // 查询所有行
    public async Task<IEnumerable<Course>> GetAllAsync()
    {
      string sql = "SELECT C.CNO, C.CNAME, C.CPNO, C.CCREDIT,
C2.CNAME FROM C " +
        "LEFT JOIN C C2 ON C.CPNO = C2.CNO;";
      MySqlCommand cmd = new MySqlCommand(sql, conn);
      using DbDataReader reader = await cmd.ExecuteReaderAsync();

      LinkedList<Course> result = new LinkedList<Course>();
      while (await reader.ReadAsync())
      {
        result.AddLast(new Course
        (
          reader.GetInt32(0),
          reader.GetString(1),
          reader.IsDBNull(2) ? (int?)null : reader.GetInt32(2),
          reader.GetInt32(3),
          reader.IsDBNull(4) ? "" : reader.GetString(4)
        ));
      }
      return result;
    }

    // 按课程号（cno）查询
    public async Task<IEnumerable<Course>> GetByNumberAsync(int
number)
    {
      string sql = "SELECT C.CNO, C.CNAME, C.CPNO, C.CCREDIT,
C2.CNAME FROM C " +
        "LEFT JOIN C C2 ON C.CPNO = C2.CNO " +
        "WHERE C.CNO = @number;";
```

```
        MySqlCommand cmd = new MySqlCommand(sql, conn);
        cmd.Parameters.AddWithValue("number", number);
        using DbDataReader reader = await cmd.ExecuteReaderAsync();

        LinkedList<Course> result = new LinkedList<Course>();
        while (await reader.ReadAsync())
        {
          result.AddLast(new Course
          (
            reader.GetInt32(0),
            reader.GetString(1),
            reader.IsDBNull(2) ? (int?)null : reader.GetInt32(2),
            reader.GetInt32(3),
            reader.IsDBNull(4) ? "" : reader.GetString(4)
          ));
        }
        return result;
    }

    // 插入一行
    public async Task InsertOneAsync(Course course)
    {
        string sql = "INSERT INTO C(CNO, CNAME, CPNO, CCREDIT) VALUES
(@number, @name, @prevCourseNumber, @credit);";
        MySqlCommand cmd = new MySqlCommand(sql, conn);
        cmd.Parameters.AddWithValue("number", course.Number);
        cmd.Parameters.AddWithValue("name", course.Name);
        cmd.Parameters.AddWithValue("prevCourseNumber",
course.PrevCourseNumber);
        cmd.Parameters.AddWithValue("credit", course.Credit);

        await cmd.ExecuteNonQueryAsync();
    }

    // 更新一行
    public async Task UpdateOneAsync(Course course)
    {
        string sql = "UPDATE C SET CNAME = @name, CPNO =
@prevCourseNumber, CCREDIT = @credit WHERE CNO = @number;";
        MySqlCommand cmd = new MySqlCommand(sql, conn);
        cmd.Parameters.AddWithValue("number", course.Number);
        cmd.Parameters.AddWithValue("name", course.Name);
        cmd.Parameters.AddWithValue("prevCourseNumber",
course.PrevCourseNumber);
```

```
        cmd.Parameters.AddWithValue("credit", course.Credit);

        await cmd.ExecuteNonQueryAsync();
    }

    // 删除一行
    public async Task DeleteOneAsync(Course course)
    {
        string sql = "DELETE FROM C WHERE C.CNO = @number;";
        MySqlCommand cmd = new MySqlCommand(sql, conn);
        cmd.Parameters.AddWithValue("number", course.Number);

        await cmd.ExecuteNonQueryAsync();
    }
}

// 用于访问选课表的 DAO
public class StudentCourseDao : Dao
{
    // 建表语句
    protected override string CreateTableSql =>
        "CREATE TABLE SC(" +
        "SCLASS INT, " +
        "SNO INT, " +
        "CNO INT, " +
        "GRADE INT, " +
        "PRIMARY KEY(SCLASS, SNO, CNO)" +
        ");";

    public StudentCourseDao(MySqlConnection conn) : base(conn) { }

    // 查询指定学生的所有选课信息
    public async Task<IEnumerable<StudentCourse>>
GetByStudentAsync(Student stu)
    {
        string sql = "SELECT SC.SCLASS, SC.SNO, SC.CNO, SC.GRADE,
C.CNAME, C.CCREDIT FROM SC, C " +
          "WHERE SC.CNO = C.CNO AND SC.SCLASS = @stuCls AND SC.SNO =
@stuNumber;";
        MySqlCommand cmd = new MySqlCommand(sql, conn);
        cmd.Parameters.AddWithValue("stuCls", stu.Cls);
        cmd.Parameters.AddWithValue("stuNumber", stu.Number);
        using DbDataReader reader = await cmd.ExecuteReaderAsync();
```

```
      LinkedList<StudentCourse> result = new
LinkedList<StudentCourse>();
      while (await reader.ReadAsync())
      {
        result.AddLast(new StudentCourse
        (
          reader.GetInt32(0),
          reader.GetInt32(1),
          reader.GetInt32(2),
          reader.GetInt32(3),
          stu.Name,
          reader.GetString(4),
          reader.GetInt32(5)
        ));
      }
      return result;
    }

    // 查询指定课程的所有选课信息
    public async Task<IEnumerable<StudentCourse>>
GetByCourseAsync(Course course)
    {
      string sql = "SELECT SC.SCLASS, SC.SNO, SC.CNO, SC.GRADE,
S.SNAME FROM SC, S " +
        "WHERE SC.SCLASS = S.SCLASS AND SC.SNO = S.SNO AND SC.CNO =
@courseNumber;";
      MySqlCommand cmd = new MySqlCommand(sql, conn);
      cmd.Parameters.AddWithValue("courseNumber", course.Number);
      using DbDataReader reader = await cmd.ExecuteReaderAsync();

      LinkedList<StudentCourse> result = new
LinkedList<StudentCourse>();
      while (await reader.ReadAsync())
      {
        result.AddLast(new StudentCourse
        (
          reader.GetInt32(0),
          reader.GetInt32(1),
          reader.GetInt32(2),
          reader.GetInt32(3),
          reader.GetString(4),
          course.Name,
          course.Credit
        ));
```

```
        }
        return result;
    }

    // 插入一行
    public async Task InsertOneAsync(StudentCourse stuCourse)
    {
        string sql = "INSERT INTO SC(SCLASS, SNO, CNO, GRADE) VALUES
(@stuCls, @stuNumber, @courseNumber, @grade);";
        MySqlCommand cmd = new MySqlCommand(sql, conn);
        cmd.Parameters.AddWithValue("stuCls", stuCourse.StudentCls);
        cmd.Parameters.AddWithValue("stuNumber",
stuCourse.StudentNumber);
        cmd.Parameters.AddWithValue("courseNumber",
stuCourse.CourseNumber);
        cmd.Parameters.AddWithValue("grade", stuCourse.Grade);

        await cmd.ExecuteNonQueryAsync();
    }

    // 更新一行
    public async Task UpdateOneAsync(StudentCourse stuCourse)
    {
        string sql = "UPDATE SC SET GRADE = @grade WHERE SC.SCLASS =
@stuCls AND SC.SNO = @stuNumber AND SC.CNO = @courseNumber;";
        MySqlCommand cmd = new MySqlCommand(sql, conn);
        cmd.Parameters.AddWithValue("stuCls", stuCourse.StudentCls);
        cmd.Parameters.AddWithValue("stuNumber",
stuCourse.StudentNumber);
        cmd.Parameters.AddWithValue("courseNumber",
stuCourse.CourseNumber);
        cmd.Parameters.AddWithValue("grade", stuCourse.Grade);

        await cmd.ExecuteNonQueryAsync();
    }

    // 删除一行
    public async Task DeleteOneAsync(StudentCourse stuCourse)
    {
        string sql = "DELETE FROM SC WHERE SC.SCLASS = @stuCls AND
SC.SNO = @stuNumber AND SC.CNO = @courseNumber;";
        MySqlCommand cmd = new MySqlCommand(sql, conn);
        cmd.Parameters.AddWithValue("stuCls", stuCourse.StudentCls);
        cmd.Parameters.AddWithValue("stuNumber",
```

```
stuCourse.StudentNumber);
        cmd.Parameters.AddWithValue("courseNumber",
stuCourse.CourseNumber);

        await cmd.ExecuteNonQueryAsync();
    }
  }
}
```

四、实验总结

在实验中有哪些重要问题或者事件？你如何处理的？你的收获是什么？

我在网上学习了程序中视图与数据分离的开发模式。这个程序将软件开发使用的 MVC 架构与数据库结合，对数据库的访问封装在了模型（Model）层。使用面向对象的特性将数据库的访问工作分到了多个类中，包括负责与数据库进行连接的 MyDatabase 类，负责访问 S/C/SC 表的数据库访问对象（Data Access Object）。用户在视图中输入，视图传递到控制器中做出判断，控制器调用 DAO 对数据进行增删改查，DAO 运行 SQL 语句。通过这种方式将视图和数据解耦，不仅让程序代码拥有更高的可读性、可维护性，还可以方便地为将来更换不同地数据库产品做准备。