

# 暨南大学本科实验报告专用纸

课程名称 数据库系统原理实验 成绩评定             
实验项目名称 暨南大学奖学金信息管理系统 指导教师             
实验项目编号 1 实验项目类型 综合性 实验地点 N116  
学生姓名 贝元琛 学号 2018053293  
学院 信息学院 系 计算机系 专业 计算机科学与技术  
实验时间 2020 年    月    日 ~    月    日

## 一、实验主题

采用指定的数据集《暨南大学 2017—2018 学年获国家励志奖学金的推荐人选名单》，根据数据集的管理需求开发一个 C/S 模式应用程序。根据本数据集的特点和管理需求，我设计开发了这个支持数据的增、删、改、查以及报表生成的暨南大学奖学金信息管理系统的应用程序。

## 二、实验开发环境

项目	名称	版本	版权	官网
数据库	MySQL	8.0.21	Oracle	<a href="https://www.mysql.com/">https://www.mysql.com/</a>
操作系统	Windows	Windows10	Microsoft	<a href="https://www.microsoft.com/">https://www.microsoft.com/</a>
数据库调试工具	SQLyog	13.1.6	Webyog	<a href="https://www.webyog.com/product/sqlyog">https://www.webyog.com/product/sqlyog</a>
编程语言	Java	1.8	Oracle	<a href="https://www.java.com/zh_CN/">https://www.java.com/zh_CN/</a>
开发工具	IntelliJ IDEA	Community 2019.1	jetbrains	<a href="https://www.jetbrains.com/idea/">https://www.jetbrains.com/idea/</a>
驱动接口	JDBC	8.0.21	Oracle	<a href="https://dev.mysql.com/downloads/connector/j/">https://dev.mysql.com/downloads/connector/j/</a>
界面设计	JavaFX	---	Oracle	---
报表工具	JasperReport +iReport	6.13	开源	<a href="https://community.jaspersoft.com/">https://community.jaspersoft.com/</a>

### 三、需求分析

#### 3.1 数据集分析

(1) 数据集名称:

《暨南大学 2017—2018 学年获国家励志奖学金的推荐人选名单》

(2) 数据集来源: 采用老师指定的数据集。

(3) 原始形式: Excel 格式文件, 共 709 条记录, 709 行 5 列

(各列属性名分别为: 序号、学生姓名、学院、专业、学号)

暨南大学2017—2018学年获国家励志奖学金的推荐人选名单				
序号	学生姓名	学院	专业	学号
1	林树宜	文学院	汉语言文学	2015051134
2	郭婉玉	文学院	汉语言文学	2015051159
3	辛腾旋	文学院	汉语言文学	2016050084

图 1: 数据集中前三条记录的情况

(4) 数据集的管理方: 暨南大学学生事务管理人员, 管理各年度各类各级学生奖学金的评定情况。

(5) 潜在用户分析: 各专业的辅导员、班级的班主任以及学生。其中辅导员、班主任会对数据集进行奖学金名单进行查询审核, 学生可以查看此名单了解自己各年度所获奖学金的情况。

#### 3.2 用户需求分析

对于开发的此应用程序, 要能够满足暨南大学学生事务管理人员对各类奖学金数据、获奖记录数据、学生数据的包括增删改查、报表生成等的管理需求, 要满足各专业的学生、辅导员以及班主任对奖学金数据、获奖记录数据的便捷准确的查询、模糊查询需求。

(1) 要准确的反馈获奖学生信息，即：获奖学生要准确定位区分，从获奖数据表单就必须确切的知道该学生是谁，故对于获奖学生的记录信息需要包括学号、姓名、年级、所在院系等信息。

(2) 要准确的反馈奖学金数据信息，即：对于各类奖学金能够清晰辨别，从数据表中就能清楚的知道奖学金编号、名称、颁发年度、颁发单位和奖学金等级等详细信息。

(3) 要能够满足各类用户的查询、添加、删除、修改各信息，导出奖学金报表的基本要求。

(4) 用户的使用界面要做到简洁、美观、操作简单方便。

### 3.3 系统功能需求分析

系统的各功能必须能够准确的反应包括奖学金、学生、学生获奖记录、学生所在的专业和学院这几个方面的信息。然后，系统功能要能够满足各用户的使用需求，包括数据记录的增删改查、报表生成等需求。开发的暨南大学奖学金信息管理系统的功能需求包括五个部分，即对奖学金信息、学生信息、学生获奖记录的增、删、查、改，以及指定获奖记录的报表生成和导出。

(1) 增加功能：此功能由用户输入所要增加的记录的详细信息，但是对于违反数据库数据规则的数据输入无法增加，并需要提示用户所输入的信息在哪里违反了数据库的数据规则，比如对于奖学金信息的增加功能，增加的奖学金编号不能重复，并且各输入的关键数据不能为空项。

(2) 删除功能：删除相应的奖学金或学生实体信息，并删除相应关

联的获奖记录关系，例如对于某奖学金信息的删除，该奖学金的获奖记录也应该相应的删除。

- (3) 查询功能：根据相应的实体的若干个关键字来查找相应的记录，关键字的数目可以有多个也可以只有一个，同时查询应该支持模糊查询，例如对于学生的奖学金获奖记录查询，学生姓名输入"吴"，应该能够输出所有满足条件的吴姓学生的获奖记录。
- (4) 修改功能：对相应的实体进行修改，但是必须遵循数据库对字段的约束条件，对不满足数据库的数据规则的修改，要提示用户输入的修改信息在哪里违反了数据库的数据规则。
- (5) 生成报表：对于有效、可公示的获奖记录表，管理人员应该能够生成及导出相应的报表，报表中要清晰的显示奖学金信息，各获奖人的信息，报表导出日期，页码信息等。

### **3.4 系统性能需求分析**

- (1) 系统要有较快的反应速度，对于用户在界面进行相应操作后，后端应该能及时进行用户指定的操作，并快速做出反馈。
- (2) 系统能有较好的容错能力，在连接数据库、操作数据库时候能及时反映当时状况，对错误信息能及时的进行处理。
- (3) 各模块对用户使用友好。例如：用户名登陆的友好提示和用户操作的准确指引。
- (4) 对使用系统的用户的信息安全性保护，例如：用户的密码在数据库不能以明文形式存储，应该加密后再存入数据库，保护用户信息安全。

## 四、实验设计

### 4.1 实体与属性设计方案

根据本数据集的特点，设计 6 个实体和其相应的属性：

**奖学金**（奖学金编号，奖学金名称，等级，颁发年份，颁发单位）

**学生**（学号，姓名，性别，出生年月，入学时间，专业号）

**学生获奖记录**（获奖记录号，学生学号，奖学金编号）

**专业**（专业号，专业名，所在系号）

**系**（系别号，系名，所属学院号）

**学院**（学院号，学院名，地址）

这些实体之间的联系如下：

- （1）一种奖学金可以有多个获奖学生，一个学生可以获得多种奖学金，因此学生和奖学金之间具有多对多的联系。
- （2）一个专业由多个学生组成，而每个学生只选择一个主修专业，因此专业和学生之间具有一对多的联系。
- （3）一个系可以有多个专业，每个专业隶属于一个系别，因此系别和专业之间具有一对多的联系。
- （4）一个学院可以有多个系别，每个系别隶属于一个学院，因此学院和系别之间具有一对多的联系。

## 4.2 E-R 图设计方案

### (1) 实体与属性图

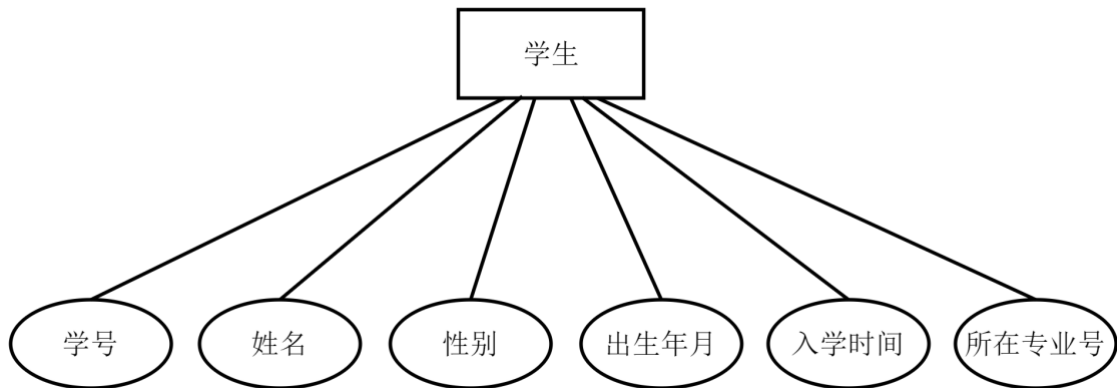


图 2：学生实体及属性

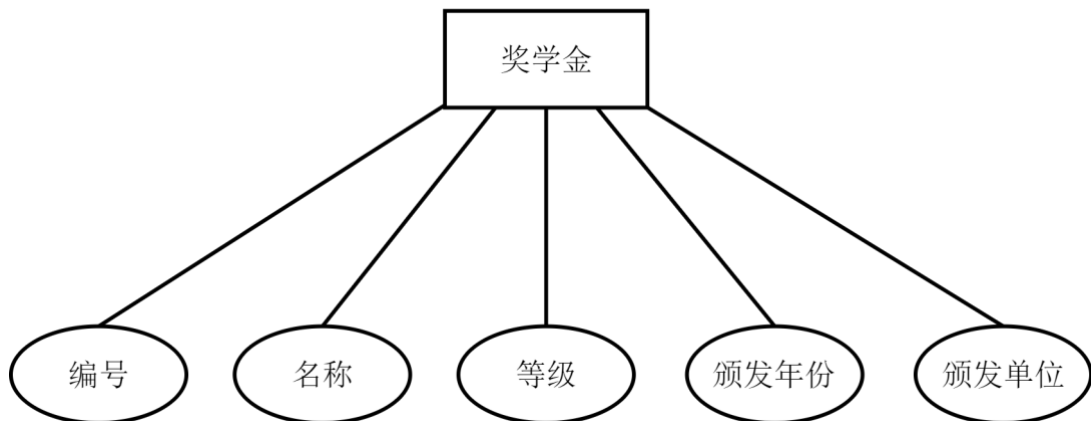


图 3：奖学金实体及属性

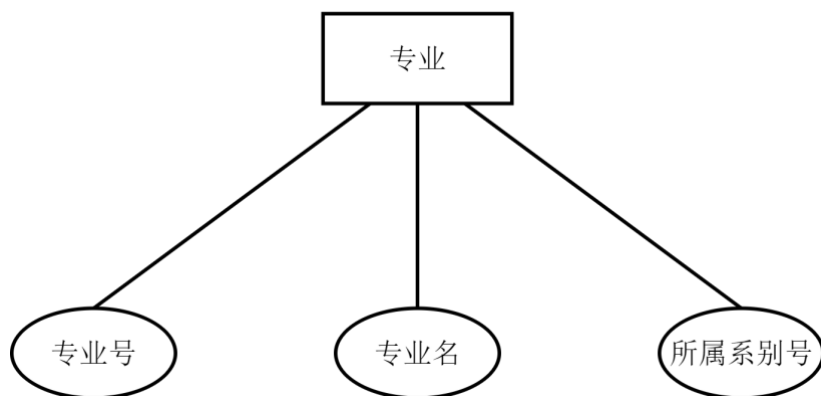


图 4：专业实体及属性

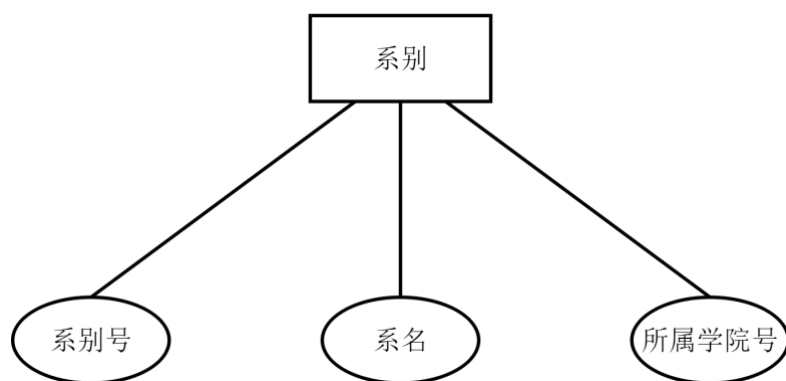


图 5: 系别实体及属性

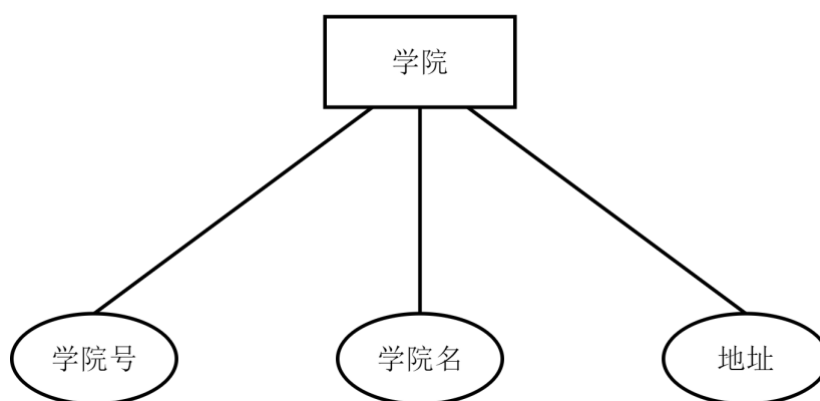


图 6: 学院实体及属性

## (2) 实体间关联图

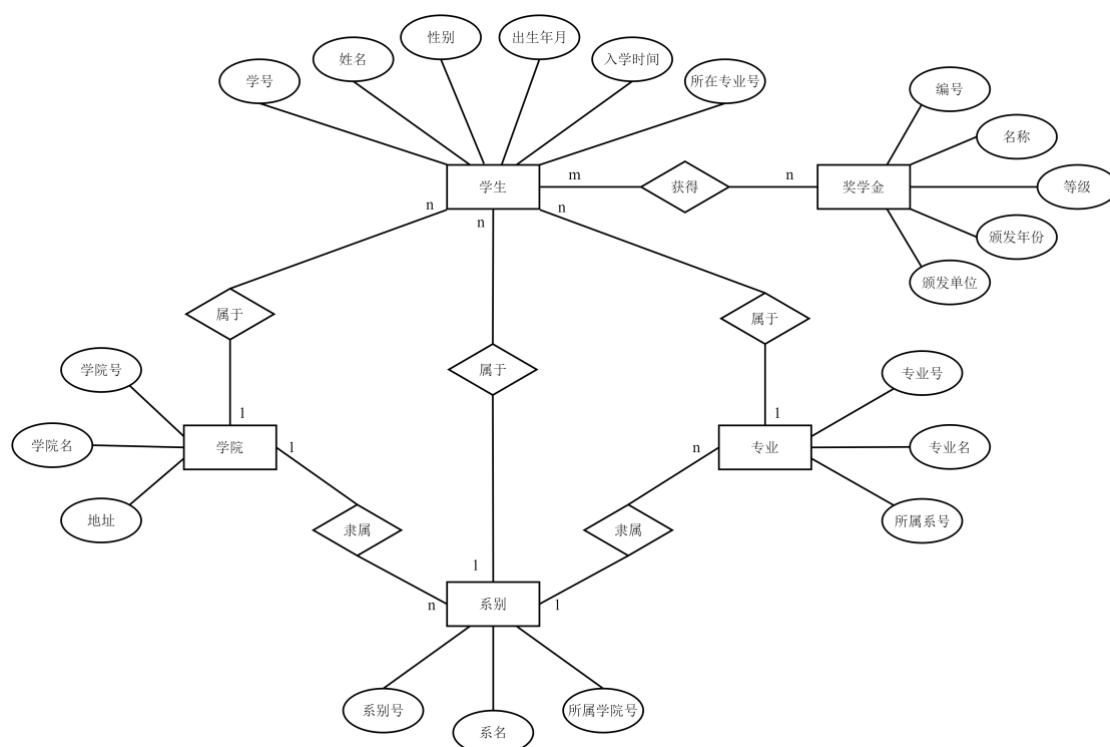


图 7: E-R 图

### 4.3 表单设计方案

本实验开发的暨南大学奖学金信息管理系统共设计有\_\_7\_\_个表单，其形式化表述为：

user (User\_id, User\_name, User\_password) ;

scholarship ( Scholar\_id, Scholar\_name, Scholar\_rank, Scholar\_year, Scholar\_issuer) ;

student (Stu\_id, Stu\_name, Stu\_sex, Stu\_birth, Stu\_grade, Major\_id) ;

stu\_scholar (id, Stu\_id, Scholar\_id) ;

major (Major\_id, Major\_name, Dept\_id) ;

department (Dept\_id, Dept\_name, College\_id) ;

college (College\_id, College\_name, College\_address) ;

表单详细设计如下：

#### (1) user

字段名	类型	长度(byte)	约束	字段说明
User_id	int	4	Primary key	系统的用户 id
User_name	varchar	30	Not null	系统登录的用户名
User_password	varchar	32	Not null	系统登录的用户 MD5 加密后密码

#### (2) scholarship

字段名	类型	长度(byte)	约束	字段说明
Scholar_id	int	4	Primary key	奖学金编号



Scholar_name	varchar	50	Not null	奖学金名称
Scholar_rank	varchar	10	Not null	奖学金等级
Scholar_year	int	4	Not null	奖学金颁发年份
Scholar_issuer	varchar	20	Not null	奖学金颁发单位

(3) student

字段名	类型	长度(byte)	约束	字段说明
Stu_id	char	10	Primary key	学生学号
Stu_name	varchar	15	Not null	学生姓名
Stu_sex	char	2	---	学生性别
Stu_birth	date	---	---	学生出生日期
Stu_grade	int	4	---	学生入学年份
Major_id	char	10	Not null	学生所属专业号

(4) stu\_scholar

字段名	类型	长度(byte)	约束	字段说明
id	int	4	Primary key	学生获奖记录号
Stu_id	char	10	Not null	获奖学生学号
Scholar_id	int	4	Not null	奖学金编号

(5) major

字段名	类型	长度(byte)	约束	字段说明
Major_id	char	10	Primary key	专业号
Major_name	varchar	15	Not null	专业名
Dept_id	char	10	Not null	所属系别号

#### (6) department

字段名	类型	长度(byte)	约束	字段说明
Dept_id	char	10	Primary key	系别号
Dept_name	varchar	15	Not null	系名
College_id	char	10	Not null	所属学院号

#### (7) college

字段名	类型	长度(byte)	约束	字段说明
College_id	char	10	Primary key	学院号
College_name	varchar	15	Not null	学院名
College_address	varchar	30	---	学院所在地址

### 4.4 系统功能模块设计

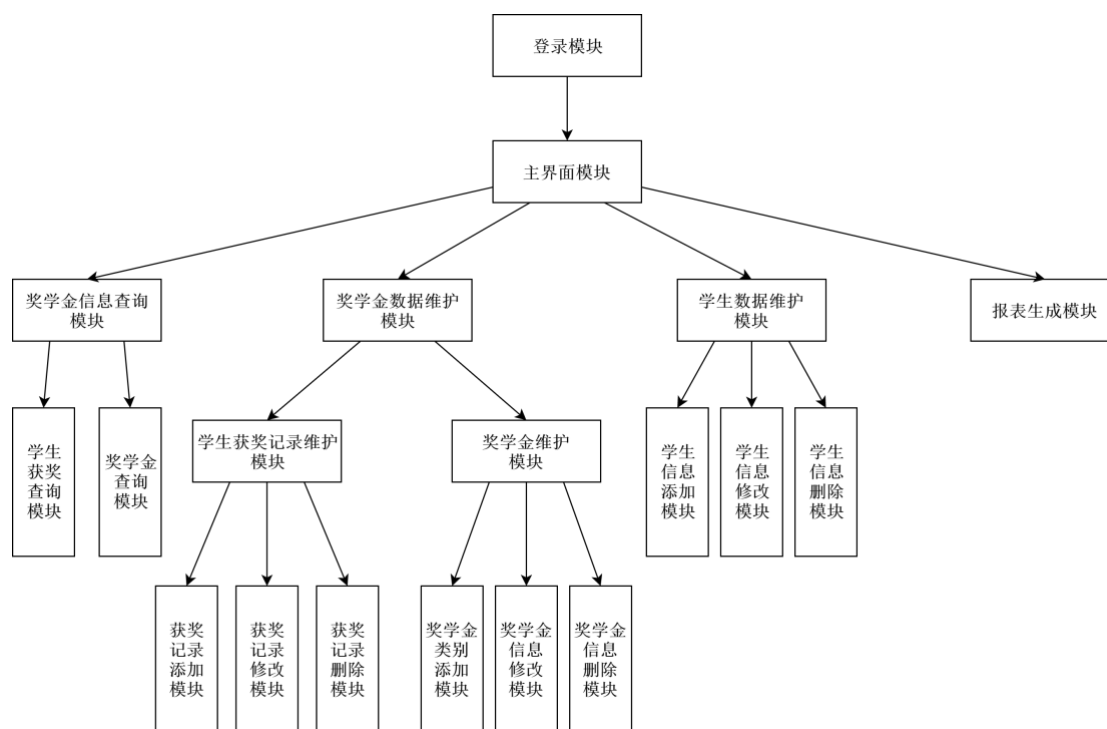


图 8：系统功能模块图

系统功能模块图中的各叶子节点是涉及数据库数据操作(增删改查、生成报表)的功能模块，其他非叶子节点是实现指引用户操作、对用户使用友好的必要功能模块。

## 4.5 数据录入、更新、删除和查询功能的设计

### 4.5.1 数据录入功能的设计

数据录入功能通过系统界面中用户在获奖记录添加模块、奖学金类别添加模块、学生信息添加模块输入的各属性值经过合法性判断后，若合法则再通过 SQL 语句写入数据库中；若不合法，则通过界面的提示窗口提示用户不合法的原因，引导用户输入合法的录入属性值。

各数据录入功能的核心代码：

```
/*
 * 学生获奖记录添加
 */
public int add(Connection con, String Stu_id, int Scholar_id) throws Exception {
    String sql="insert into stu_scholar values(?,?,?)";
    PreparedStatement pstmt=con.prepareStatement(sql);
    pstmt.setString(1,null);
    pstmt.setString(2,Stu_id);
    pstmt.setString(3,String.valueOf(Scholar_id));
    return pstmt.executeUpdate();
}
```

```
/*
 * 学生信息添加
 */
public int add(Connection con, Student student)throws Exception{
    String sql="insert into student values(?,?,?,?,?)";
    PreparedStatement pstmt=con.prepareStatement(sql);
    pstmt.setString(1,student.getId());
    pstmt.setString(2,student.getName());
    pstmt.setString(3,student.getSex());
    pstmt.setString(4,student.getBirth());
}
```

```

        pstmt.setString(5,String.valueOf(student.getGrade()));
        pstmt.setString(6,student.getM_id());
        return pstmt.executeUpdate();
    }

    /*
     * 奖学金类别添加
     */
    public int add(Connection con, Scholarship scholarship) throws Exception {
        String sql = "insert into scholarship values(?,?,?,?,?)";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1, String.valueOf(scholarship.getId()));
        pstmt.setString(2, scholarship.getScholarName());
        pstmt.setString(3, scholarship.getScholarRank());
        pstmt.setString(4, String.valueOf(scholarship.getScholarYear()));
        pstmt.setString(5, scholarship.getScholarIssuer());
        return pstmt.executeUpdate();
    }

```

#### 4.5.2 数据更新功能的设计

数据更新功能通过系统界面中用户在获奖记录修改模块、奖学金类别修改模块、学生信息修改模块输入修改的各属性值经过合法性判断后若合法则再通过 SQL 语句将修改数据写入数据库中；若不合法则通过界面的提示窗口提示用户不合法的原因，引导用户输入合法的修改属性值。

各数据更新功能的核心代码：

```

    /*
     * 获奖记录更新
     */
    public int update(Connection con,String stu_id,int scholar_id)throws Exception{
        String sql="update stu_scholar set Scholar_id=? where Stu_id=?";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,String.valueOf(scholar_id));
        pstmt.setString(2,stu_id);
        return pstmt.executeUpdate();
    }

```

```

/*
*学生更新
*/
public int update(Connection con, Student student)throws Exception{
    String sql="update student set
    Stu_name=?,Stu_sex=?,Stu_birth=?,Stu_grade=?,Major_id=? where Stu_id=?";
    PreparedStatement pstmt=con.prepareStatement(sql);
    pstmt.setString(1,student.getName());
    pstmt.setString(2,student.getSex());
    pstmt.setString(3,student.getBirth());
    pstmt.setString(4,String.valueOf(student.getGrade()));
    pstmt.setString(5,student.getM_id());
    pstmt.setString(6,student.getId());
    return pstmt.executeUpdate();
}

/*
*奖学金更新
*/
public int update(Connection con, Scholarship scholarship)throws Exception{
    String sql="update scholarship set
    Scholar_name=?,Scholar_rank=?,Scholar_year=?,Scholar_issuer=?
    where Scholar_id=?";

    PreparedStatement pstmt=con.prepareStatement(sql);
    pstmt.setString(1,scholarship.getScholarName());
    pstmt.setString(2,scholarship.getScholarRank());
    pstmt.setString(3,String.valueOf(scholarship.getScholarYear()));
    pstmt.setString(4,scholarship.getScholarIssuer());
    pstmt.setString(5,String.valueOf(scholarship.getId()));
    return pstmt.executeUpdate();
}

```

#### 4.5.3 数据删除功能的设计

数据删除功能通过系统界面中用户在获奖记录删除模块、奖学金类别删除模块、学生信息删除模块输入要删除的信息的关键字经过合法性判断后，若合法并且记录存在则再通过 SQL 语句在数据库中将相应的记录删除。

各数据删除功能的核心代码：

```
/*
 *学生获奖记录删除
 */
public int delete(Connection con, String stu_id, String scholar_id) throws Exception {
    //更新获奖信息表
    String sql = "delete from stu_scholar where Stu_id=? and Scholar_id=?";
    PreparedStatement pstmt = con.prepareStatement(sql);
    pstmt.setString(1, stu_id);
    pstmt.setString(2, scholar_id);
    return pstmt.executeUpdate();
}

/*
 *学生删除
 */
public int delete(Connection con, String id) throws Exception {
    //更新获奖信息表
    String sql0 = "delete from stu_scholar where Stu_id=?";
    PreparedStatement pstmt0 = con.prepareStatement(sql0);
    pstmt0.setString(1, id);
    pstmt0.executeUpdate();

    //更新学生表
    String sql = "delete from student where Stu_id=?";
    PreparedStatement pstmt = con.prepareStatement(sql);
    pstmt.setString(1, id);
    return pstmt.executeUpdate();
}

/*
 *奖学金删除
 */
public int delete(Connection con, int id) throws Exception {
    //更新获奖信息表
    String sql0 = "delete from stu_scholar where Scholar_id=?";
    PreparedStatement pstmt0 = con.prepareStatement(sql0);
    pstmt0.setString(1, String.valueOf(id));
    pstmt0.executeUpdate();

    //更新奖学金表
    String sql = "delete from scholarship where Scholar_id=?";
    PreparedStatement pstmt = con.prepareStatement(sql);
```

```

        pstmt.setString(1, String.valueOf(id));
        return pstmt.executeUpdate();
    }
}

```

#### 4.5.4 数据查询功能的设计

数据查询功能通过系统界面中用户在学生获奖查询模块和奖学金查询模块中输入所要查询的信息的关键字，再通过 SQL 语句在数据库中按照所输入的关键字进行支持模糊查询的数据库查询操作，查询完成后将查询结果在界面中反馈给用户。

各数据查询功能的核心代码：

```

/*
 *学生获奖信息查询
 */
public ResultSet query(Connection con,Student student) throws Exception{
    StringBuffer sql=
        new StringBuffer(
            "SELECT student.Stu_id,student.Stu_name,major.Major_name,scholarship.Scholar_name
            FROM stu_scholar LEFT JOIN student ON student.Stu_id=stu_scholar.Stu_id LEFT JOIN
            scholarship ON scholarship.Scholar_id=stu_scholar.Scholar_id LEFT JOIN major ON
            major.Major_id=student.Major_id");
    if(!String_isEmpty.isEmpty(student.getId())){
        sql.append(" and student.Stu_id like '%" +student.getId()+"%'");
    }
    if(!String_isEmpty.isEmpty(student.getName())){
        sql.append(" and student.Stu_name like '%" +student.getName()+"%'");
    }
    PreparedStatement
    pstmt=con.prepareStatement(sql.toString().replaceFirst("and","where"));
    return pstmt.executeQuery();
}

/*
 *学生查询
 */
public ResultSet query(Connection con, String id) throws Exception {
    StringBuffer sql = new StringBuffer("select * from student");
    if (!String_isEmpty.isEmpty(id)) {
        sql.append(" and Stu_id like '%" + id + "%'");
    }
}

```

```

    }
    PreparedStatement pstmt = con.prepareStatement(sql.toString().replaceFirst("and",
"where"));
    return pstmt.executeQuery();
}

/*
*奖学金查询
*/
public ResultSet query(Connection con, Scholarship scholarship) throws Exception {
    StringBuffer sql = new StringBuffer("select * from scholarship");
    //由于 id 为 int 类型,id 为 0 标志其值为空
    if (scholarship.getId() != 0) {
        sql.append(" and Scholar_id=" + String.valueOf(scholarship.getId()));
    }
    if (String.isEmpty(scholarship.getScholarName())) {
        sql.append(" and Scholar_name like %" + scholarship.getScholarName() + "%");
    }
    if (String.isEmpty(scholarship.getScholarRank())) {
        sql.append(" and Scholar_rank=" + scholarship.getScholarRank() + "");
    }
    if (scholarship.getScholarYear() != 0) {
        sql.append(" and Scholar_year=" + String.valueOf(scholarship.getScholarYear()));
    }
    if (String.isEmpty(scholarship.getScholarIssuer())) {
        sql.append(" and Scholar_issuer like %" + scholarship.getScholarIssuer() + "%");
    }
    PreparedStatement pstmt = con.prepareStatement(sql.toString().replaceFirst("and",
"where"));
    return pstmt.executeQuery();
}

```

## 五、软件实现

### 5.1 用户登录

系统启动后，首先进入的就是用户登录的界面，界面为：

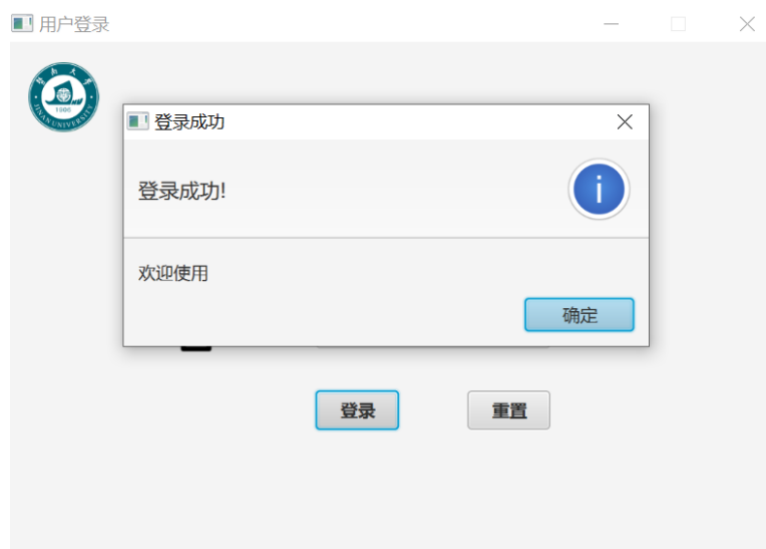




输入用户名和密码进行登录(测试用户名:admin, 密码:123456):



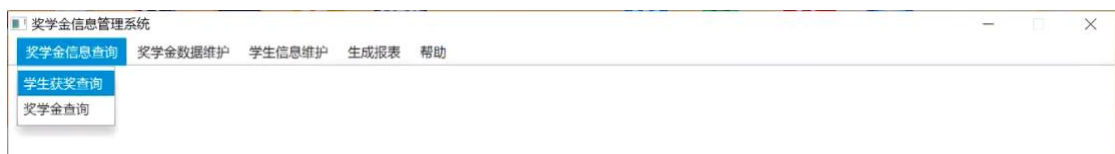
输入正确用户名和密码则正确登录，提示登录成功并跳转至主界面：







其中，主界面上方**奖学金信息查询**入口按钮包含**学生获奖查询**和**奖学金查询**两个数据库查询功能模块：



主界面上方**奖学金数据维护**入口按钮包含**学生获奖记录维护**和**奖学金维护**两个子模块，其中**学生获奖记录维护**子模块包含**添加**、**修改**、**删除**学生获奖记录三个数据库功能模块：



**奖学金维护**子模块包含**添加**、**删除**、**修改**奖学金信息三个数据库功能模块：



主界面上方**学生信息维护**入口按钮包含**添加、删除、修改**学生信息这三个数据库功能模块：

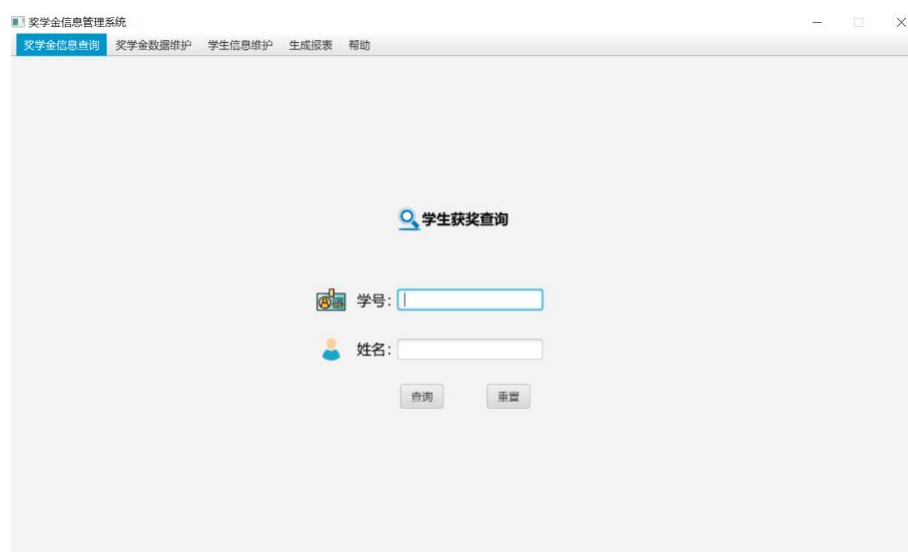


主界面上方**生成报表**入口按钮包含**生成/打印报表**功能模块：



### 5.3 学生获奖查询

学生获奖查询功能通过主界面上方的“奖学金信息查询”□“学生获奖查询”进入，界面为：



学生获奖查询功能展示：

输入测试学号：

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

学生获奖查询

学号: 2018053293

姓名:

查询 重置

点击查询按钮：

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

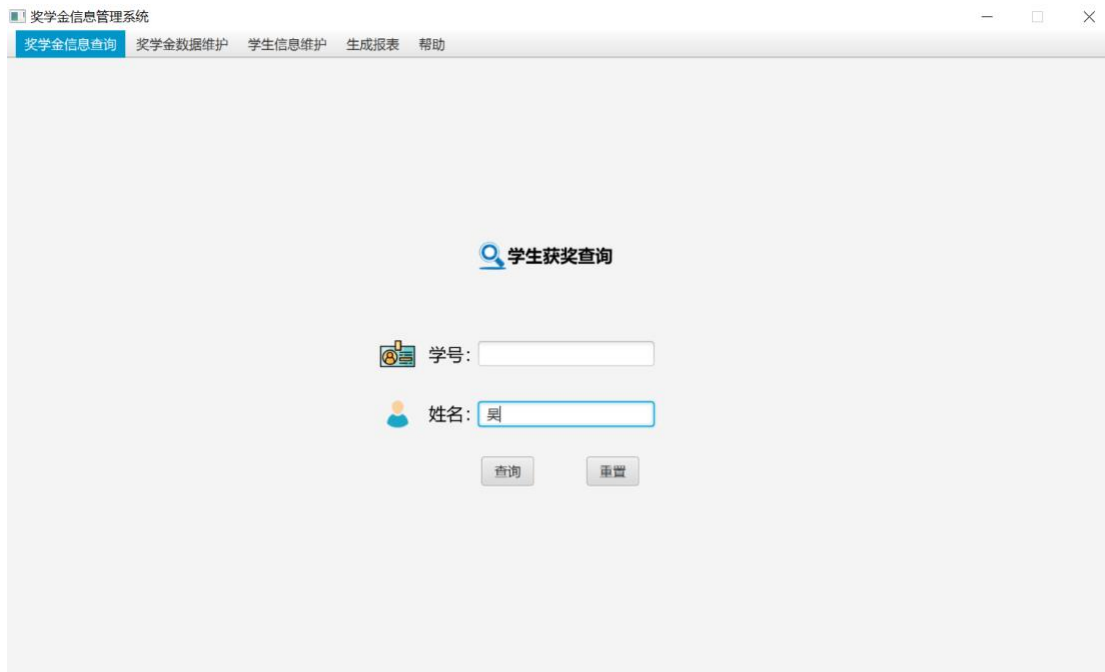
查询结果

学号	姓名	专业	所获奖学金名称
2018053293	贝元琛	计算机科学与技术	2017-2018学年国家奖学金

返回

同时，学生获奖查询功能支持模糊查询：

如：在姓名栏只输入“吴”：



点击查询按钮，可以看出所有吴姓学生的获奖信息都查询出：



## 5.4 奖学金查询

学生获奖查询功能通过主界面上方的“奖学金信息查询”□“奖学金查询”进入，界面为：

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

### 奖学金查询

奖学金编号:

奖学金名称:

奖学金等级: ☐ 国家级 ☐ 省级 ☐ 校级

颁发年份:

颁发单位:

奖学金查询功能支持各查询关键字单独或组合查询及模糊查询：

单关键字查询，如选择“国家级”查询所有国家级奖学金：

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

### 奖学金查询

奖学金编号:

奖学金名称:

奖学金等级: ☒ 国家级 ☐ 省级 ☐ 校级

颁发年份:

颁发单位:

点击查询按钮得到查询结果：



组合关键字查询：

如选择国家级及颁发年份为 2020：



点击查询得到查询结果：





模糊查询：

如查询 2018 年颁发单位含有“教育”关键字的奖学金：



点击查询得到查询结果：

可以看出所有颁发年份为 2018 年，颁发单位含有“教育”二字的奖

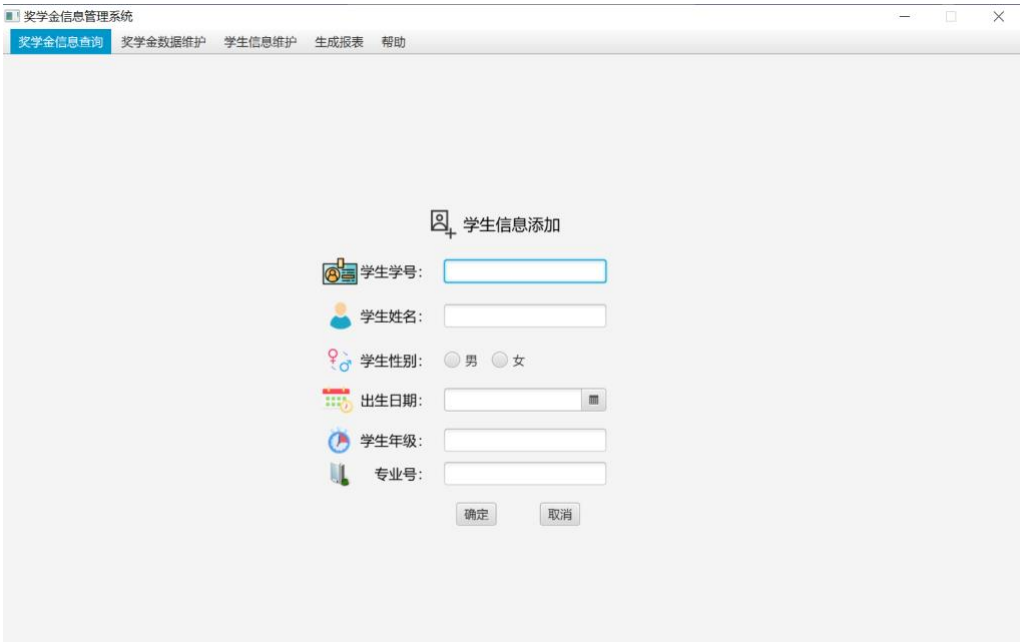
学金信息查询出：



## 5.5 学生信息维护

### 5.5.1 学生信息添加

学生信息添加功能通过主界面上方的“学生信息维护”□“添加学生信息”进入，界面为：



测试添加学生信息，其中出生日期可以通过日历控件直接进行选择：

奖学金信息管理系统

奖学金信息查询奖学金数据维护学生信息维护生成报表帮助

学生信息添加

学生学号：2020202020

学生姓名：测试学生

学生性别：☒男☐女

出生日期：

< 十二月 >

< 2001 >

星期日	星期一	星期二	星期三	星期四	星期五	星期六
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

学生年级：专业号：

奖学金信息管理系统

奖学金信息查询奖学金数据维护学生信息维护生成报表帮助

学生信息添加

学生学号：2020202020

学生姓名：测试学生

学生性别：☒男☐女

出生日期：2001-12-25

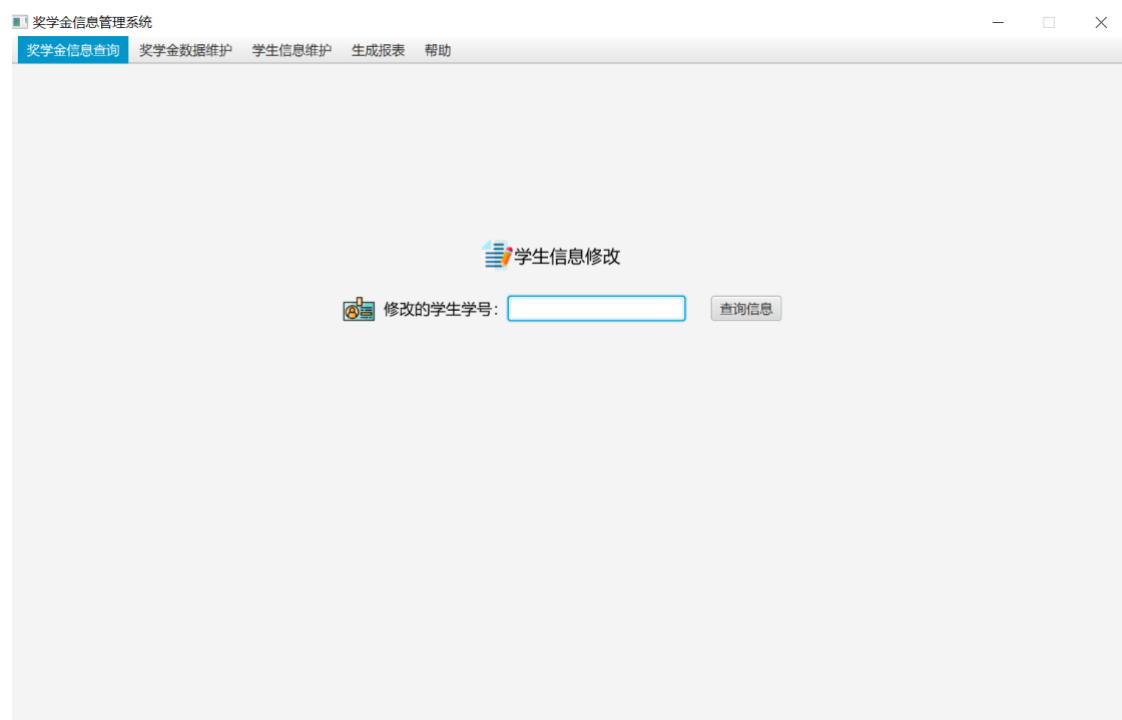
学生年级：2020

专业号：0001

确定取消

点击“确定”按钮进行添加：





修改上述测试学生的学生信息：



将专业号修改为 0002（软件工程）：

The screenshot shows a web application window titled "奖学金信息管理系统" (Scholarship Information Management System). The main menu includes "奖学金信息查询" (Scholarship Information Query), "奖学金数据维护" (Scholarship Data Maintenance), "学生信息维护" (Student Information Maintenance), "生成报表" (Generate Reports), and "帮助" (Help). The "学生信息维护" (Student Information Maintenance) tab is active, displaying the "学生信息修改" (Student Information Modification) form.

The form contains the following fields and controls:

- 修改的学生学号:** A text input field containing "2020202020" and a "查询信息" (Query Information) button.
- 学生信息:** A section header with a green message: "您可在下方对信息进行编辑修改" (You can edit the information below).
- 学生学号:** A text input field containing "2020202020".
- 学生姓名:** A text input field containing "测试学生" (Test Student).
- 学生性别:** A text input field containing "M".
- 出生日期:** A text input field containing "2001-12-25".
- 年级:** A text input field containing "2020".
- 专业号:** A text input field containing "0002", which is highlighted with a blue border.
- Buttons:** "确认修改" (Confirm Modification) and "取消修改" (Cancel Modification) buttons.

点击“确认修改”完成修改：

弹出提示框显示修改成功并显示修改结果窗口：

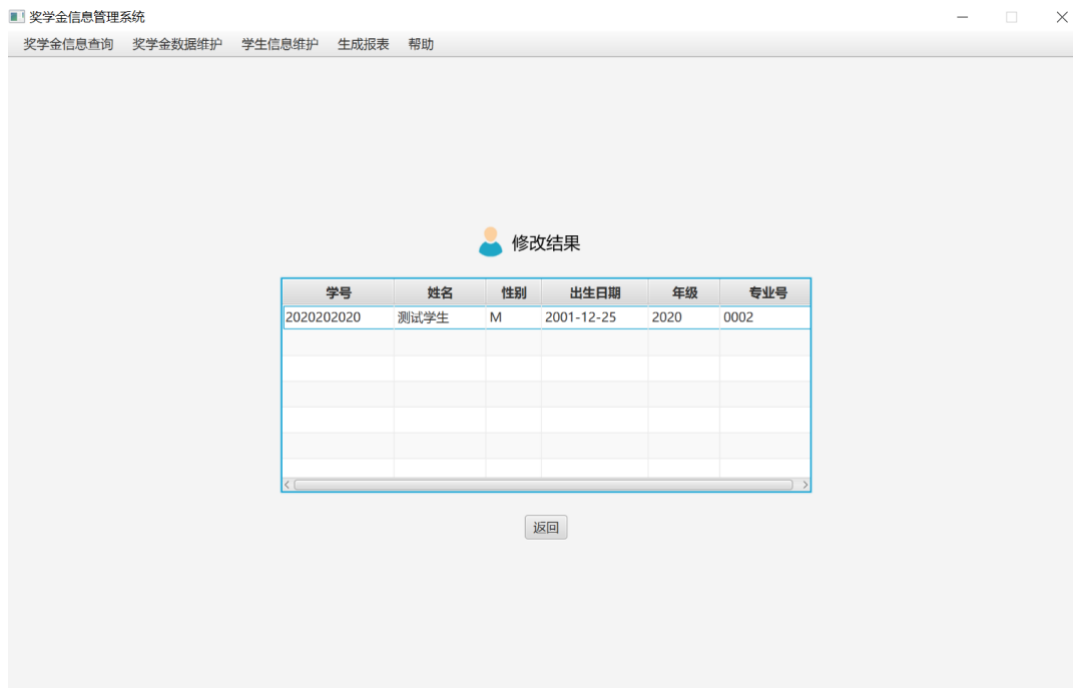
The screenshot shows the "学生信息修改成功!" (Student Information Modification Success!) dialog box. The dialog box contains the message "测试学生 同学的信息修改成功!" (Test Student, the student's information has been successfully modified!) and a "确定" (Confirm) button.

The background window, "学生信息维护" (Student Information Maintenance), displays a table with the following data:

学号	姓名	性别	出生日期	年级	专业号
2020202020	测试学生	M	2001-12-25	2020	0002

At the bottom of the window, there is a "返回" (Return) button.

从修改结果窗口中可以看出，测试学生的专业号已成功修改为 0002。



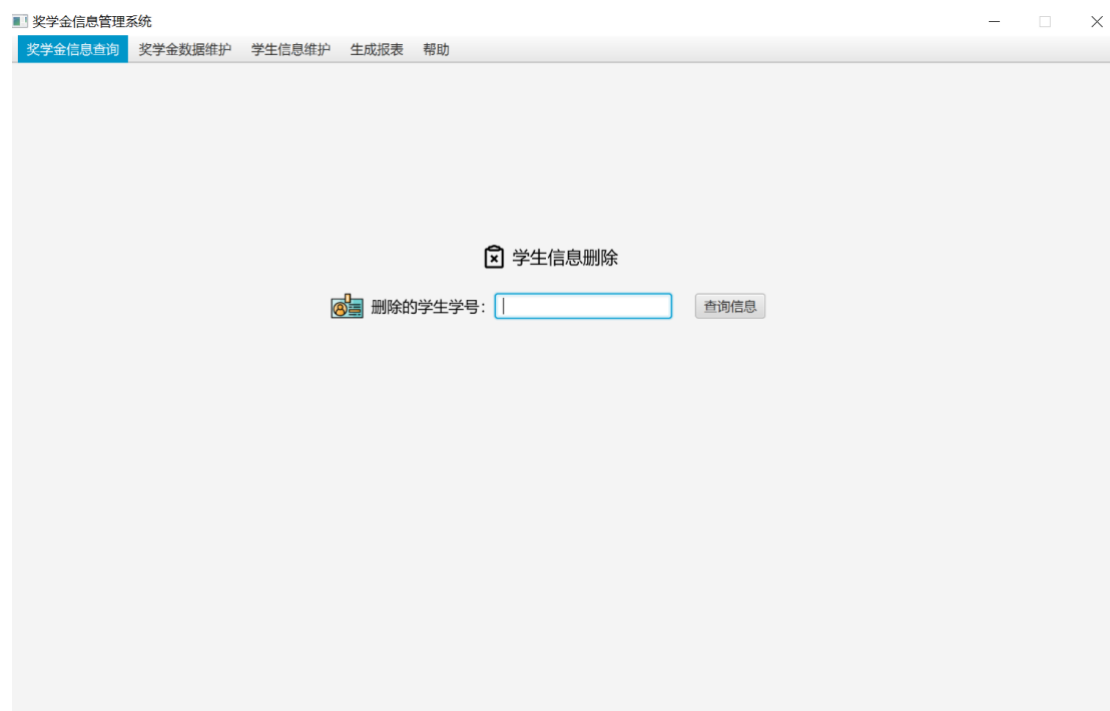
在数据库中也可以看出，测试学生的信息成功修改：

1 信息 2 表数据 3 信息						
<input type="checkbox"/>	Stu_id	Stu_name	Stu_sex	Stu_birth	Stu_grade	Major_id
<input type="checkbox"/>	2020000001	王俊凯	M	2000-02-02	2020	0004
<input type="checkbox"/>	2020000002	易烊千玺	M	2000-03-03	2020	0006
<input type="checkbox"/>	2020111111	黄晓明	M	2001-08-01	2020	0003
<input type="checkbox"/>	2020123456	刘瑞琦	F	2000-08-09	2020	0007
<input type="checkbox"/>	2020202020	测试学生	M	2001-12-25	2020	0002
<input type="checkbox"/>	2020222222	杨颖	F	2001-08-08	2020	0003
<input type="checkbox"/>	2020666666	程潇	F	2001-06-06	2020	0006
*	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)

数据库: db\_scholarship 表格: student

### 5.5.3 学生信息删除

学生信息删除功能通过主界面上方的“学生信息维护” □ “删除学生信息”进入，界面为：



查询并删除上述测试学生的信息：



点击“确认删除”完成删除：





为验证正确性，查询该学号学生：



可以看出，该测试学生信息已成功删除。

## 5.6 奖学金维护

### 5.6.1 奖学金种类添加

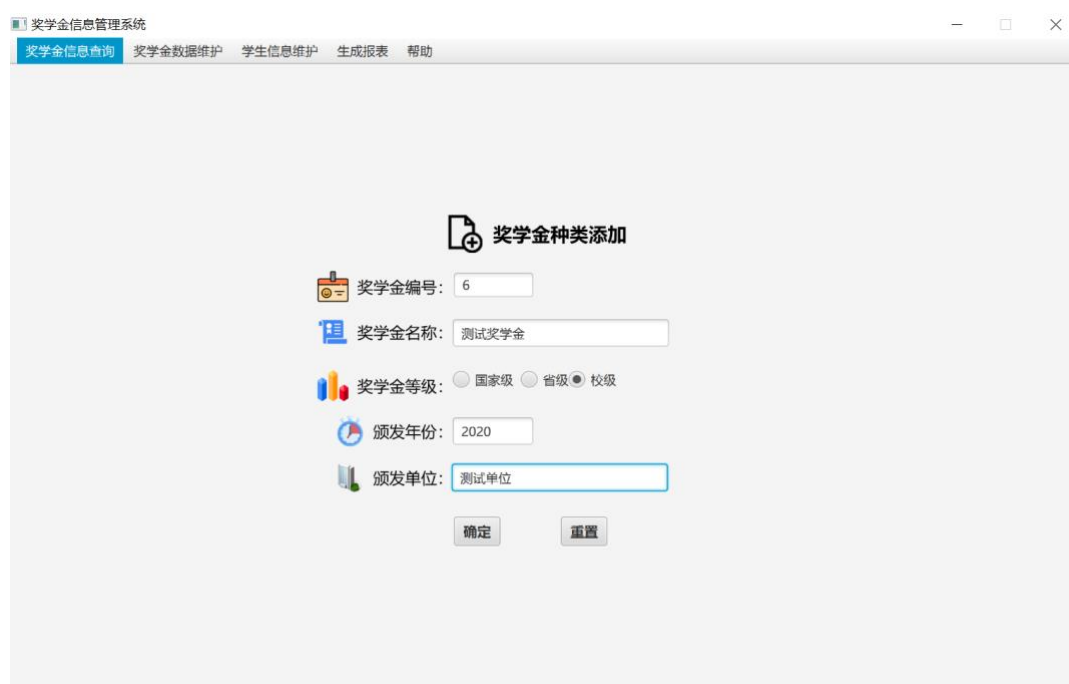
奖学金种类添加通过主界面上方的“奖学金数据维护”□“奖学金维护”□“添加新的奖学金”进入，界面为：



The screenshot shows a web application window titled "奖学金信息管理系统" (Scholarship Information Management System). The navigation bar includes "奖学金信息查询", "奖学金数据维护", "学生信息维护", "生成报表", and "帮助". The main content area is titled "奖学金种类添加" (Add Scholarship Type) with a plus icon. It contains the following fields and controls:

- 奖学金编号 (Scholarship Number): A text input field.
- 奖学金名称 (Scholarship Name): A text input field.
- 奖学金等级 (Scholarship Level): Three radio buttons labeled "国家级" (National), "省级" (Provincial), and "校级" (School). The "校级" option is selected.
- 颁发年份 (Issuance Year): A text input field.
- 颁发单位 (Issuance Unit): A text input field.
- Buttons: "确定" (Confirm) and "重置" (Reset).

添加测试奖学金，详细测试信息如图所示：



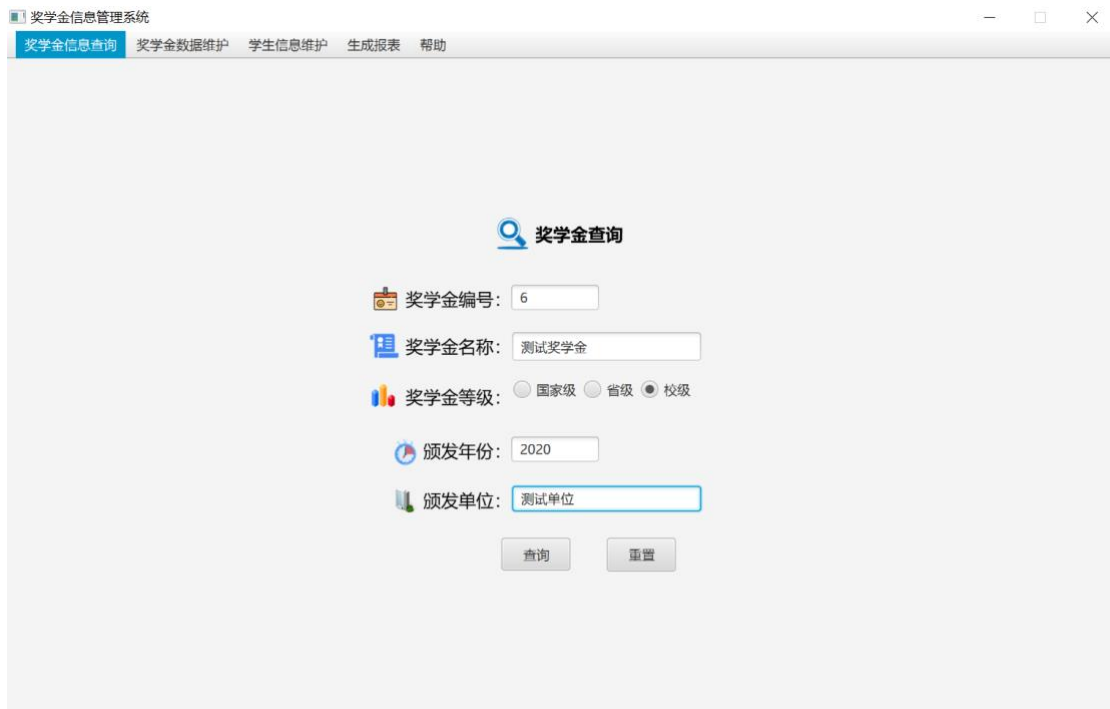
This screenshot shows the same "奖学金种类添加" form, but with test data entered into the fields:

- 奖学金编号: 6
- 奖学金名称: 测试奖学金
- 奖学金等级: 校级 (selected)
- 颁发年份: 2020
- 颁发单位: 测试单位
- Buttons: "确定" and "重置"

点击确定按钮添加，提示添加成功：



验证正确性，在奖学金查询模块中查询该添加的测试奖学金：



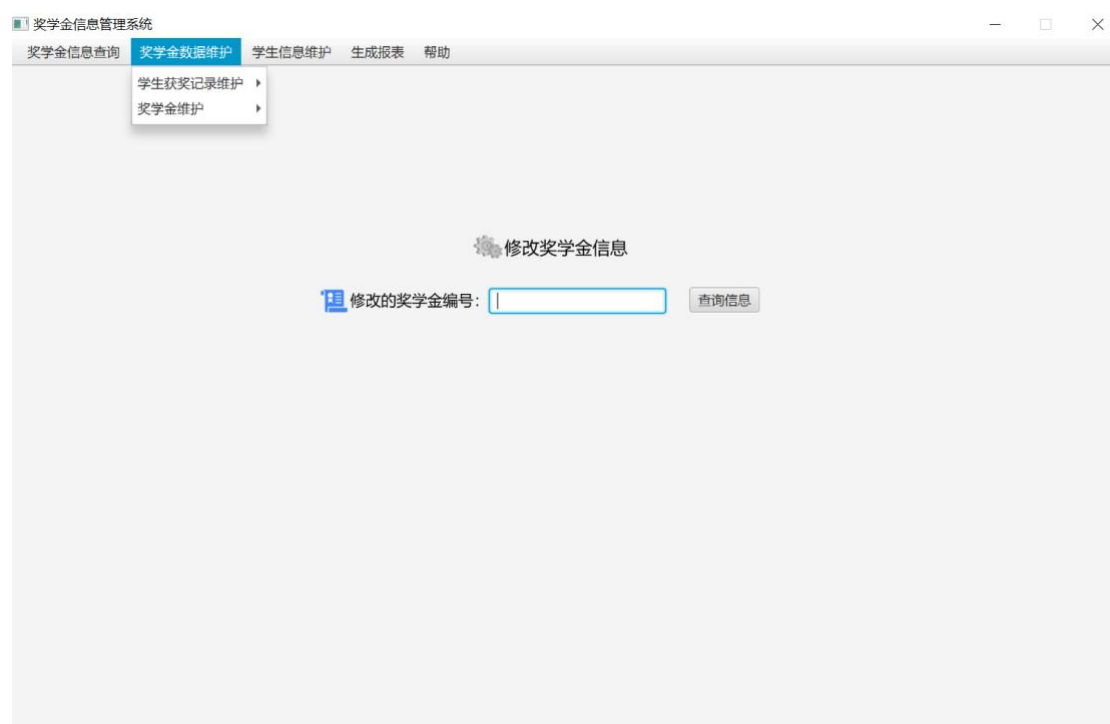
点击查询查找是否在数据库中已成功添加该奖学金：



从查询结果可以看出，该测试奖学金成功添加入数据库。

## 5.6.2 奖学金信息修改

奖学金信息修改通过主界面上方的“奖学金数据维护”□“奖学金维护”□“修改奖学金信息”进入，界面为：



修改上述添加的测试奖学金信息（编号为 6）：

奖学金信息管理系统

奖学金信息查询奖学金数据维护学生信息维护生成报表帮助

修改奖学金信息

修改的奖学金编号：

奖学金信息：您可在下方对信息进行编辑修改

奖学金编号：

奖学金名称：

奖学金等级：

颁发年份：

颁发单位：

如修改颁发年份为 2019，颁发单位改为“测试颁发单位”：

奖学金信息管理系统

奖学金信息查询奖学金数据维护学生信息维护生成报表帮助

修改奖学金信息

修改的奖学金编号：

奖学金信息：您可在下方对信息进行编辑修改

奖学金编号：

奖学金名称：

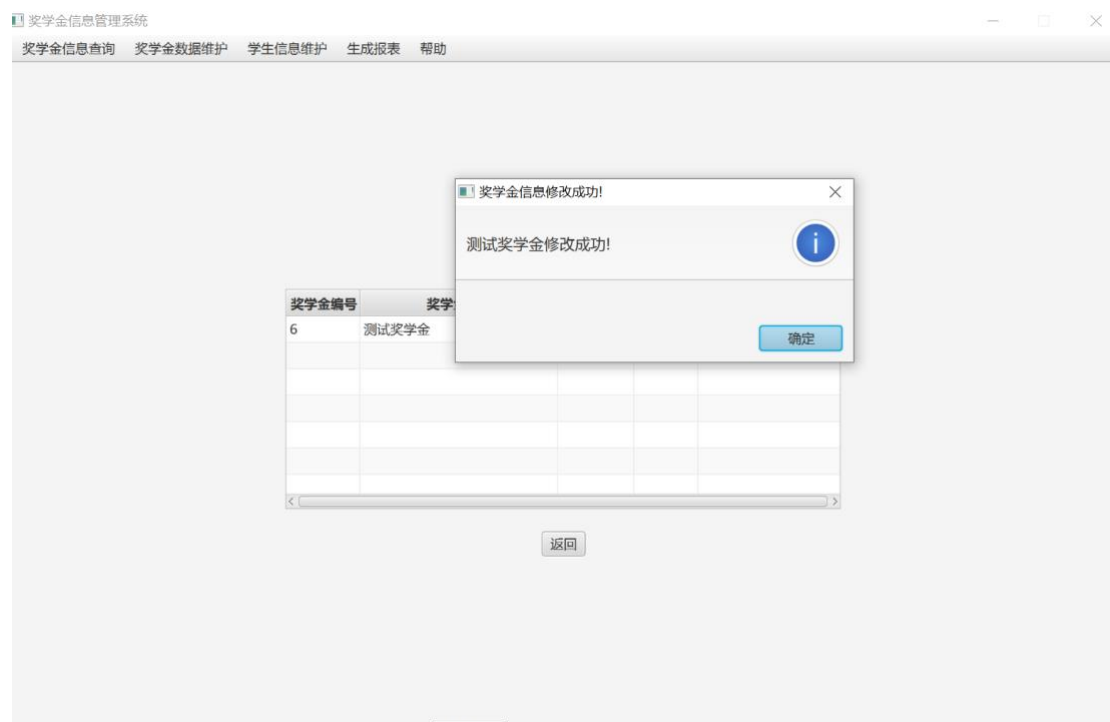
奖学金等级：

颁发年份：

颁发单位：

点击确认修改：

弹出修改成功提示框并显示修改结果界面：

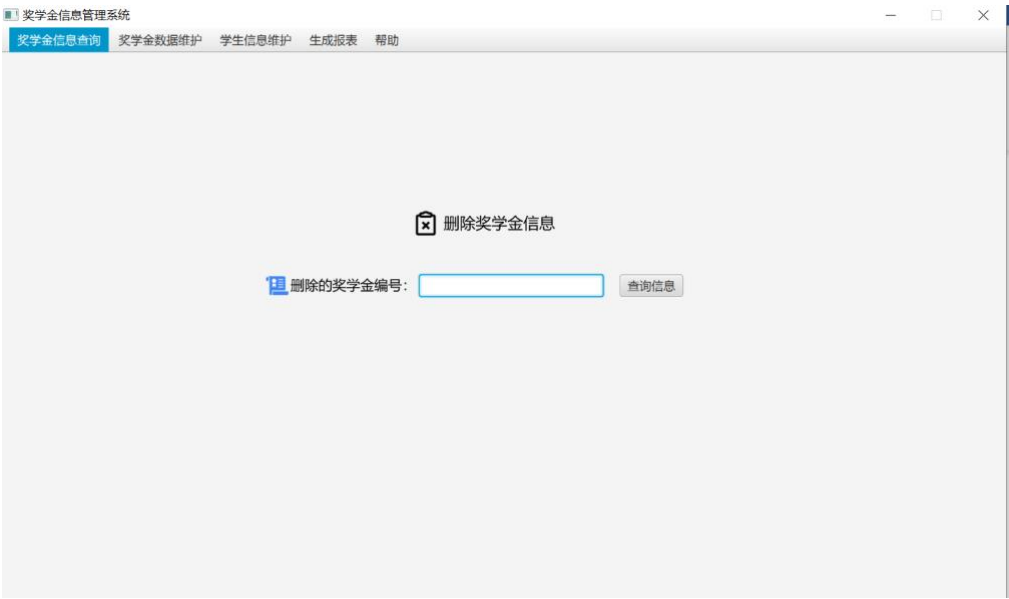


可以看出，测试奖学金的颁发年份，颁发单位已相应的分别修改至2019年和“测试颁发单位”。

### 5.6.3 奖学金信息删除

奖学金信息删除通过主界面上方的“奖学金数据维护” □ “奖学金维

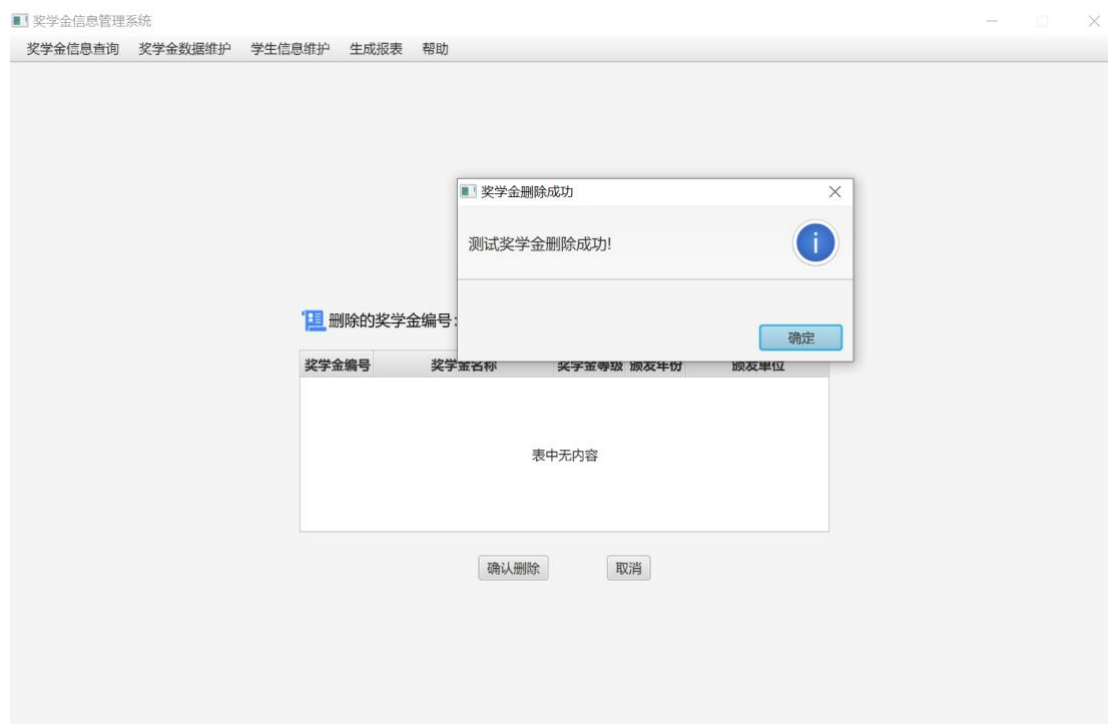
护” □ “删除奖学金信息” 进入，界面为：



通过删除上述测试奖学金进行功能测试：



点击确认删除按钮：



为验证正确性，在奖学金查询功能模块中查询该奖学金，判断是否已成功删除：





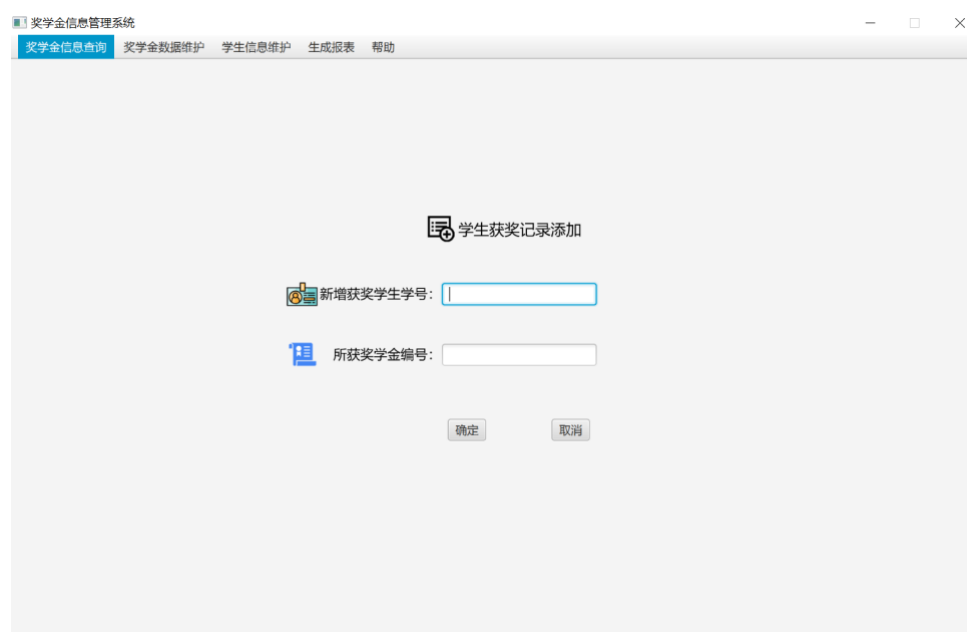


可以看出，该测试奖学金已经成功删除。

## 5.7 学生获奖记录维护

### 5.7.1 学生获奖记录添加

学生获奖记录添加功能通过主界面上方的“奖学金数据维护” □ “学生获奖记录维护” □ “添加学生获奖记录” 进入，界面为：



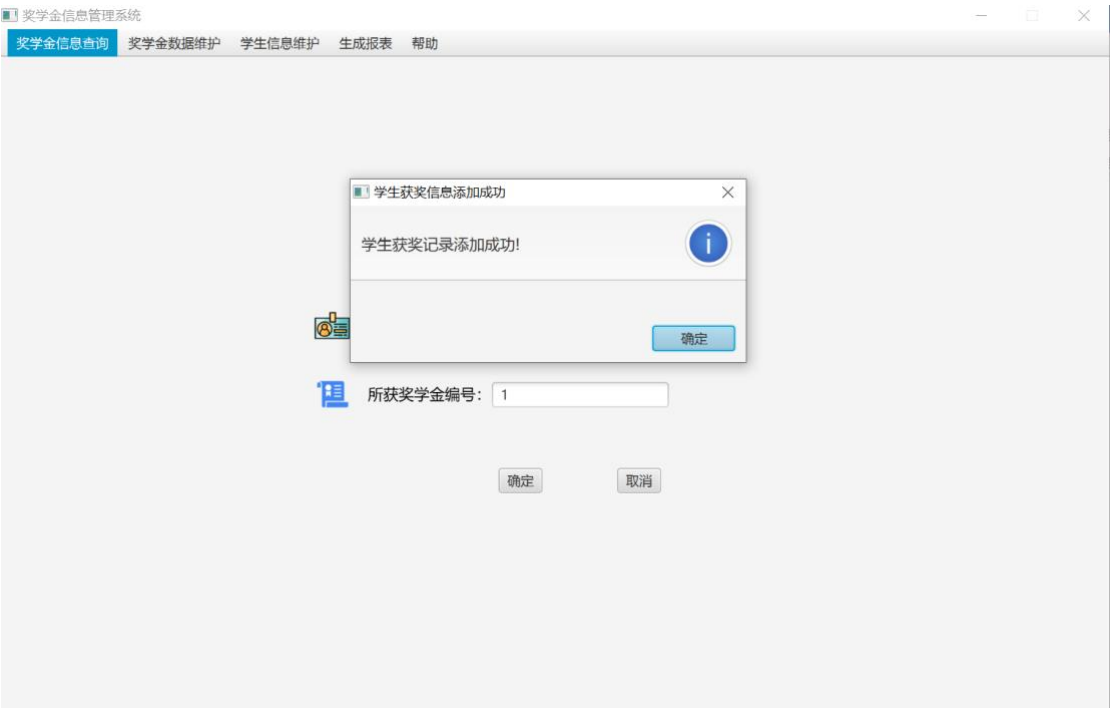
测试添加学生获奖记录：

测试的学生学号：2018112233（姓名为小明）；

测试的奖学金编号：1（名称为 2017-2018 学年国家奖学金）；



点击“确定”按钮添加：



为验证正确性，在学生获奖查询功能中查询所添加的学生学号：

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

学生获奖查询

学号: 2018112233

姓名:

查询 重置

点击“确定”查询：

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

查询结果

学号	姓名	专业	所获奖学金名称
2018112233	小明	计算机科学与技术	2017-2018学年国家奖学金

返回

可以看出，添加的学生获奖记录成功记录在数据库中。

### 5.7.2 学生获奖记录修改

学生获奖记录添加功能通过主界面上方的“奖学金数据维护”□“学

生获奖记录维护” □ “修改学生获奖记录” 进入，界面为：

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

学生获奖记录修改

修改的获奖学生学号:

所获奖学金编号:

查询输入的信息

测试修改上述添加的学生获奖记录：

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

学生获奖记录修改

修改的获奖学生学号:

2018112233

所获奖学金编号:

1

查询输入的信息

学生获奖信息

您可在下方对信息进行修改

获奖学生学号:

2018112233

奖学金编号:

1

确认修改 取消

将所获奖学金编号修改为 3（2018 年度广东省优秀大学生奖学金）：

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

### 学生获奖记录修改

修改的获奖学生学号: 2018112233

所获奖学金编号: 1

查询输入的信息

学生获奖信息 您可在下方对信息进行修改

获奖学生学号: 2018112233

奖学金编号: 3

确认修改 取消

点击“确认修改”完成修改:

奖学金信息管理系统

奖学金信息查询 奖学金数据维护 学生信息维护 生成报表 帮助

学生信息修改成功!

学号为:2018112233的同学的获奖信息修改成功!

确定

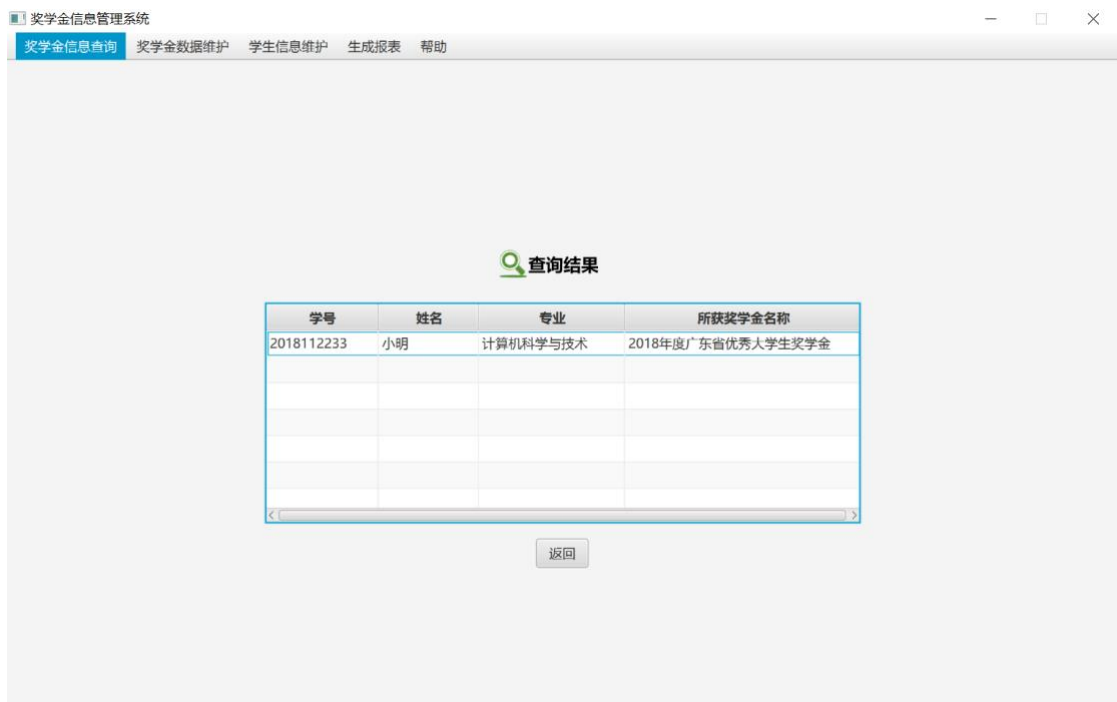
学生获奖信息 您可在下方对信息进行修改

获奖学生学号: 2018112233

奖学金编号: 3

确认修改 取消

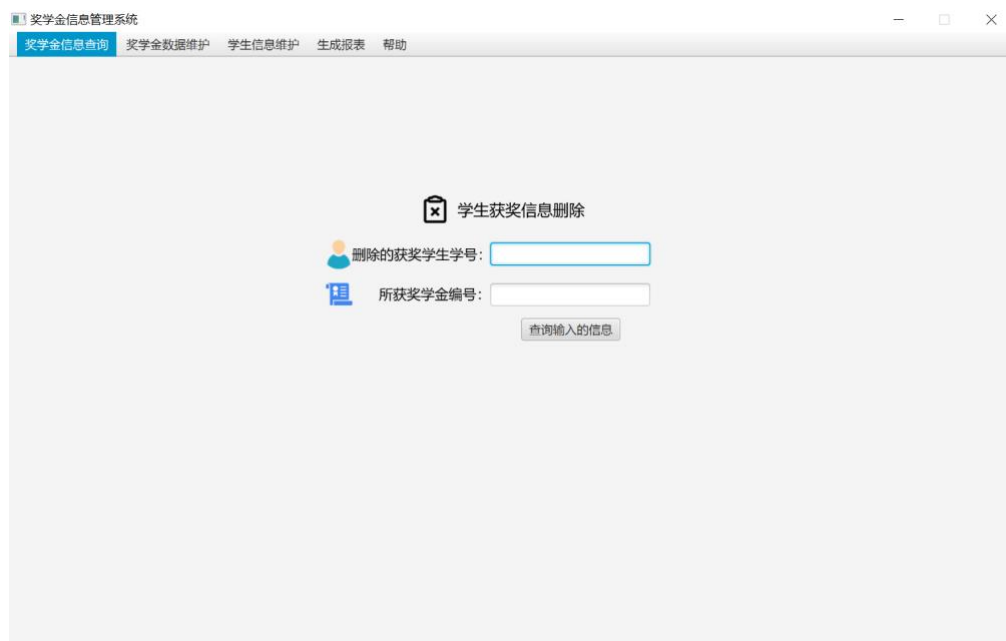
为验证正确性，在学生获奖查询功能中查询该学生获奖信息是否有进行变更:



可以看出，该学生所获奖学金已成功修改为编号为 3 的奖学金（2018 年度广东省优秀大学生奖学金）。

### 5.7.3 学生获奖记录删除

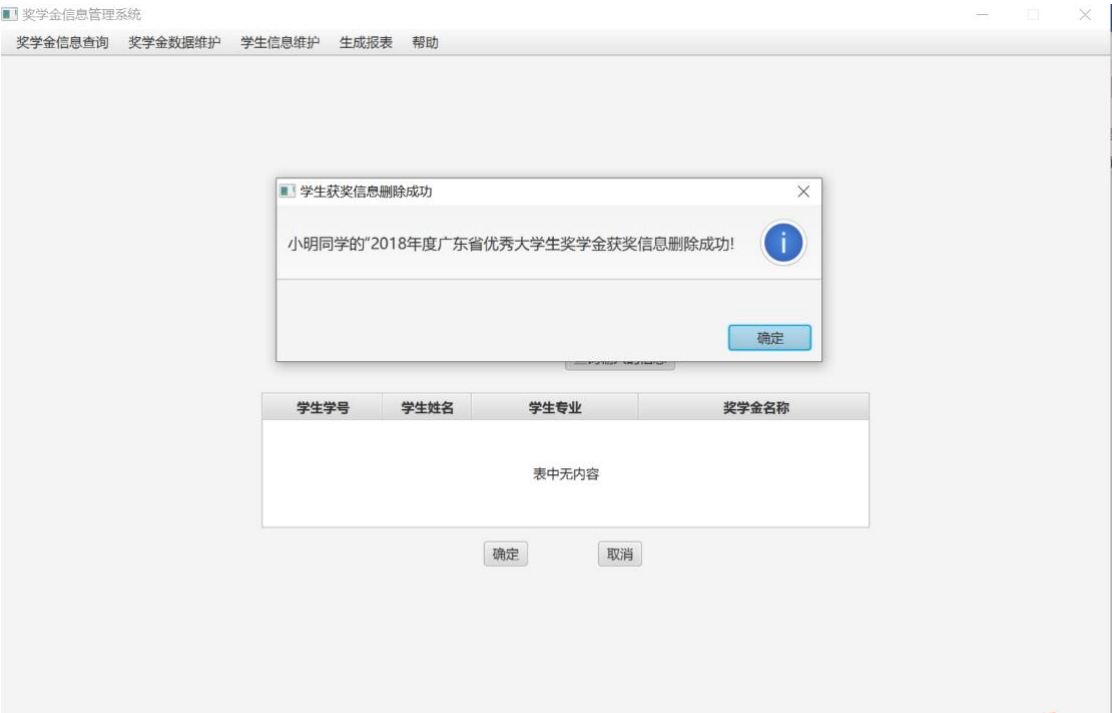
学生获奖记录删除功能通过主界面上方的“奖学金数据维护”□“学生获奖记录维护”□“删除学生获奖记录”进入，界面为：



删除上述测试的学生获奖记录：



点击“确定”按钮完成记录的删除：



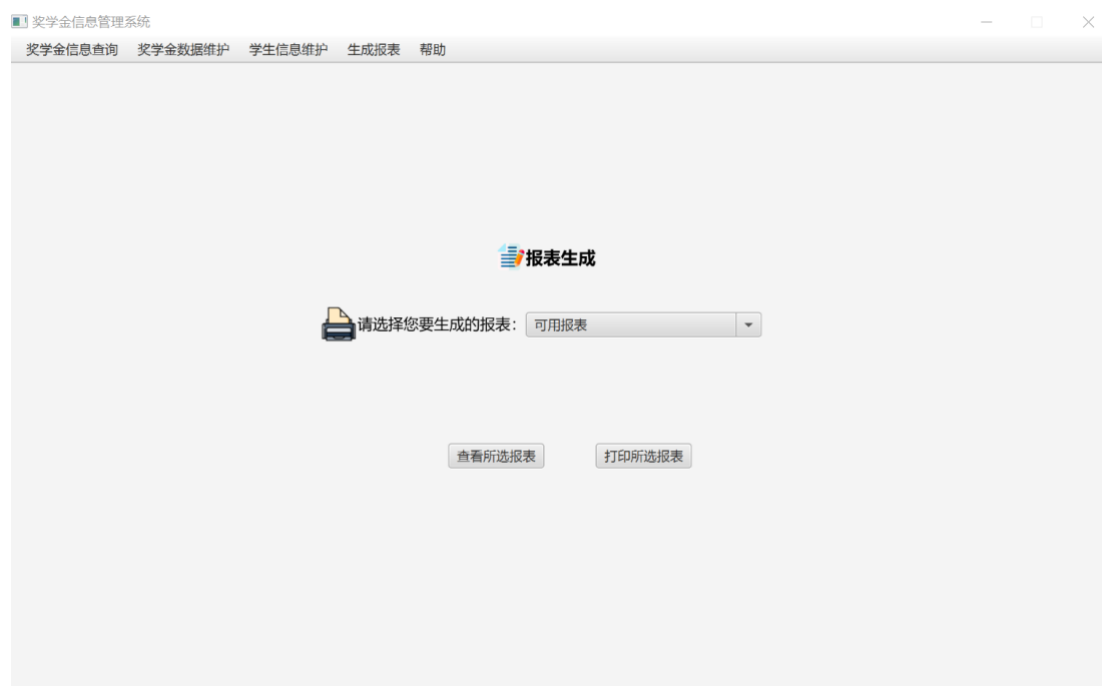
为验证正确性，在学生获奖查询功能中查询该学生获奖信息是否已经删除：



可以看出，该条学生获奖记录已经成功删除。

## 5.8 生成/打印报表

生成/打印报表功能通过主界面上方的“生成报表”□“生成/打印报表”进入，界面为：





可以通过“可用报表”栏目按钮选择要生成/打印的报表：



选择“2019-2020 学年国家励志奖学金”，再点击“查看所选报表”按钮查看生成的报表：

暨南大学2019—2020学年获国家励志奖学金的推荐人选名单			
名单导出日期：2020/12/25			
姓名	专业	学院	学号
贾玲	播音主持	新闻与传播学院	2017333333
吴宣仪	视觉媒体	新闻与传播学院	2018000000
沈腾	计算机科学与技术	信息科学技术学院	2018222222
李娜	汉语言文学	文学院	2019222222
王源	软件工程	信息科学技术学院	2020000000
王俊凯	电子信息工程	信息科学技术学院	2020000001
易烊千玺	视觉媒体	新闻与传播学院	2020000002
黄晓明	应用数学	信息科学技术学院	2020111111
王一博	计算机科学与技术	信息科学技术学院	2017222222
吴亦凡	软件工程	信息科学技术学院	2018233333
关晓彤	视觉媒体	新闻与传播学院	2019010203
鹿晗	视觉媒体	新闻与传播学院	2019020304
孟美岐	播音主持	新闻与传播学院	2019111111
杨颖	应用数学	信息科学技术学院	2020222222
程潇	视觉媒体	新闻与传播学院	2020666666

当前页码：1 / 共 2 页

名单导出日期: 2020/12/25

姓名	专业	学院	学号
张天爱	汉语言文学	文学院	2018111111
邓紫棋	汉语言文学	文学院	2018666666
华晨宇	通信工程	信息科学技术学院	2019233333
张勇	计算机科学与技术	信息科学技术学院	2018123456
刘瑞琦	播音主持	新闻与传播学院	2020123456

当前页码: 2 / 共 2 页

生成的报表如上图所示, 指明了对应的奖学金名称, 获奖学生的信息, 名单导出日期和页码等详细信息。

点击“打印所选报表”按钮, 跳转到打印界面:

Print

Printer: 联创打印管理系统

Properties

Advanced

Help ?

Copies: 1

☐ Print in grayscale (black and white)

☐ Save ink/toner ⓘ

Pages to Print

☒ All ☐ Current ☐ Pages 1 - 2

More Options

Page Sizing & Handling ⓘ

SizePosterMultipleBooklet

☐ Fit ☐ Actual size  
☒ Shrink oversized pages ☐ Custom Scale: 100 %

☐ Choose paper source by PDF page size

☐ Print on both sides of paper

Orientation:  
☒ Auto ☐ Portrait ☐ Landscape

Comments & Forms

Document and Markups

Summarize Comments

Scale: 99%

8.26 x 11.69 Inches

暨南大学2019-2020学年国家励志奖学金获得者入选名单

名单导出日期: 2020/12/25

姓名	专业	学院	学号
张天爱	汉语言文学	文学院	2018111111
邓紫棋	汉语言文学	文学院	2018666666
华晨宇	通信工程	信息科学技术学院	2019233333
张勇	计算机科学与技术	信息科学技术学院	2018123456
刘瑞琦	播音主持	新闻与传播学院	2020123456

当前页码: 1 / 共 2 页

Page 1 of 2

Print

Cancel

## 六、实验总结

本次实验我完成了这个暨南大学奖学金信息管理系统，这是我完成的第一个从数据库数据表单到系统功能、系统界面、系统实现全部个人独立完成的项目，在此之前我没有做过这么完整的一个项目。

在实验中也遇到了一些问题，如：

(1) 我选择的界面实现框架 `javafx` 是一个网络上学习资源比较少的框架，当时选择这个框架是因为这个框架相较于 `swing` 更具美观性。在界面实现过程中，开始时由于这个框架需要 XML, CSS 等的基础知识，我先自学了这些需要用到的基础知识，然后在开始实现的时候发现中文资料比较不全，有一些可以用到的资料是英文的，之前一直习惯了看中文学习文档，这时就有些不习惯，但在有些问题实在找不到中文资料的时候，就开始去学习谷歌上的一些 `javafx` 问题英文解答，开始时速度很慢，后来就慢慢习惯并能从一些 `javafx` 英语解答中获得自己想要的答案并编程实现。

(2) 涉及的模块较多，开始时感觉到有些混乱。我是使用 `java` 语言完成整个系统的实现，由于我将每个功能模块封装成一个类，而每个功能界面又是一个类，每种数据库操作也是一个类，在界面事件和相应的功能模块和相应的数据库操作之间的联系比较多，开始感到很乱，后面我是去重新梳理了整个系统的模块之间的结构和关系，再学习了 `java` 项目的分层结构，对各模块根据属性分层处理，对各模块实现之后根据整个系统的结构彼此之间连接起来。

通过此次实验我熟悉了一个数据库系统的完整开发流程，对数据库的增删改查等各操作更加清晰。同时，经过这次实验之后，对理论课中包括主码、外码等的知识点通过实践之后有了一个更加深刻直观的理解。并且，此次实验的系统通过 java 编程，经过这次实验开发系统之后，可以感受到我的编程能力也得到了很大程度的提升。

# 暨南大学本科实验报告专用纸(附页)

---

附录:

主要功能的源代码清单:

## 1、登录界面:

```
package pers.YuanchenBei.dbwork.view;

import javafx.application.Platform;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.stage.Stage;
import pers.YuanchenBei.dbwork.dao.UserDao;
import pers.YuanchenBei.dbwork.model.User;
import pers.YuanchenBei.dbwork.utils.Db_utils;
import pers.YuanchenBei.dbwork.utils.String_isEmpty;
import pers.YuanchenBei.dbwork.utils.MD5_Encryption;

import java.io.IOException;
import java.sql.Connection;
import java.util.Optional;

public class LoginController {

    @FXML
    private TextField userName_txt;

    @FXML
    private PasswordField password_txt;

    @FXML
    private Button login_btn;

    @FXML
    private Button reset_btn;

    private Db_utils dbUtils=new Db_utils();
    private UserDao userDao=new UserDao();
```

```

/*
 *登录按钮事件处理
 */
@FXML
void login_event(ActionEvent event) {
    String userName = this.userName_txt.getText();
    String password = this.password_txt.getText();

    if(String_isEmpty.isEmpty(userName)){
        Alert alert = new Alert(Alert.AlertType.WARNING); // 创建一个消
        息对话框
        alert.setTitle("好像出了点问题...");
        alert.setHeaderText("用户名不能为空!"); // 设置对话框的头部文本
        alert.setContentText("请重新输入用户名和密码");
        alert.setResizable(false);
        // 设置对话框的内容文本
        alert.show(); // 显示对话框
        this.userName_txt.setText("");
        this.password_txt.setText("");
    }else if(String_isEmpty.isEmpty(password)){
        Alert alert=new Alert(Alert.AlertType.WARNING);
        alert.setTitle("好像出了点问题...");
        alert.setHeaderText("密码不能为空!"); // 设置对话框的头部文本
        alert.setContentText("请重新输入用户名和密码");
        alert.setResizable(false);
        // 设置对话框的内容文本
        alert.show(); // 显示对话框
        this.userName_txt.setText("");
        this.password_txt.setText("");
    }
    else{
        password = MD5_Encryption.md5(password); //对密码进行 MD5 加
        密

        User user=new User(userName,password);
        Connection con=null;
        try {
            con=dbUtils.getCon();
            User current_user=userDao.login(con,user);
            if(current_user!=null){
                Alert alert=new Alert(Alert.AlertType.INFORMATION);
                alert.setTitle("登录成功");
                alert.setHeaderText("登录成功!"); // 设置对话框的头部文
                本
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

```

        alert.setContentText("欢迎使用");
        alert.setResizable(false);
        // 设置对话框的内容文本
        alert.show(); // 显示对话框
        Thread.sleep(1500);
        alert.close();
        Platform.runLater() -> {
            //创建主界面窗口
            try {
                new MainFrameStart().start(new Stage());
            } catch (IOException e) {
                e.printStackTrace();
            }
            //关闭登陆窗口
            Stage now=(Stage)login_btn.getScene().getWindow();
            now.hide();
        });
    } else {
        Alert alert=new Alert(Alert.AlertType.ERROR);
        alert.setTitle("好像出了点问题...");
        alert.setHeaderText("用户名或密码错误!"); // 设置对话框
        的头部文本
        alert.setContentText("请确认后重新输入用户名和密码");
        alert.setResizable(false);
        // 设置对话框的内容文本
        alert.show(); // 显示对话框
        this.userName_txt.setText("");
        this.password_txt.setText("");
    }
    } catch (Exception e) {
        e.printStackTrace();
    }
}

/*
 *重置按钮事件处理
 */
@FXML
void reset_event(ActionEvent event) {
    this.userName_txt.setText("");
    this.password_txt.setText("");
}

```

```
}
```

## 2、学生信息增加、删除、修改、查询功能：

```
package pers.YuanchenBei.dbwork.dao;
```

```
import pers.YuanchenBei.dbwork.model.Scholarship;  
import pers.YuanchenBei.dbwork.model.Student;  
import pers.YuanchenBei.dbwork.utils.String_isEmpty;
```

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

```
public class StudentDao {
```

```
    /*  
     * 学生信息添加  
     */
```

```
    public int add(Connection con, Student student)throws Exception {  
        String sql="insert into student values(?,?,?,?,?,?)";  
        PreparedStatement pstmt=con.prepareStatement(sql);  
        pstmt.setString(1,student.getId());  
        pstmt.setString(2,student.getName());  
        pstmt.setString(3,student.getSex());  
        pstmt.setString(4,student.getBirth());  
        pstmt.setString(5,String.valueOf(student.getGrade()));  
        pstmt.setString(6,student.getM_id());  
        return pstmt.executeUpdate();  
    }
```

```
    /*  
     *学生查询  
     */
```

```
    public ResultSet query(Connection con, String id) throws Exception {  
        StringBuffer sql = new StringBuffer("select * from student");  
        if (String_isEmpty.isEmpty(id)) {  
            sql.append(" and Stu_id like %" + id + "%");  
        }  
        PreparedStatement pstmt  
con.prepareStatement(sql.toString().replaceFirst("and", "where"));  
        return pstmt.executeQuery();  
    }
```

```
    /*
```



```

    *学生删除
    */
    public int delete(Connection con, String id) throws Exception {
        //更新获奖信息表
        String sql0 = "delete from stu_scholar where Stu_id=?";
        PreparedStatement pstmt0 = con.prepareStatement(sql0);
        pstmt0.setString(1, id);
        pstmt0.executeUpdate();

        //更新学生表
        String sql = "delete from student where Stu_id=?";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1, id);
        return pstmt.executeUpdate();
    }

    /*
    *学生更新
    */
    public int update(Connection con, Student student) throws Exception {
        String sql="update student set
        Stu_name=?,Stu_sex=?,Stu_birth=?,Stu_grade=?,Major_id=? where Stu_id=?";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,student.getName());
        pstmt.setString(2,student.getSex());
        pstmt.setString(3,student.getBirth());
        pstmt.setString(4,String.valueOf(student.getGrade()));
        pstmt.setString(5,student.getM_id());
        pstmt.setString(6,student.getId());
        return pstmt.executeUpdate();
    }
}

```

### 3、奖学金信息增加、删除、修改、查询功能：

```

package pers.YuanchenBei.dbwork.dao;

import javafx.scene.chart.ScatterChart;
import pers.YuanchenBei.dbwork.model.Scholarship;
import pers.YuanchenBei.dbwork.utils.String_isEmpty;

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

```

```

import java.util.stream.Stream;

public class ScholarshipDao {
    /*
     * 奖学金类别添加
     */
    public int add(Connection con, Scholarship scholarship) throws Exception {
        String sql = "insert into scholarship values(?,?,?,?)";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1, String.valueOf(scholarship.getId()));
        pstmt.setString(2, scholarship.getScholarName());
        pstmt.setString(3, scholarship.getScholarRank());
        pstmt.setString(4, String.valueOf(scholarship.getScholarYear()));
        pstmt.setString(5, scholarship.getScholarIssuer());
        return pstmt.executeUpdate();
    }

    /*
     * 奖学金查询
     */
    public ResultSet query(Connection con, Scholarship scholarship) throws
    Exception {
        StringBuffer sql = new StringBuffer("select * from scholarship");
        //由于 id 为 int 类型,id 为 0 标志其值为空
        if (scholarship.getId() != 0) {
            sql.append(" and Scholar_id=" + String.valueOf(scholarship.getId()));
        }
        if (String.isEmpty(scholarship.getScholarName())) {
            sql.append(" and Scholar_name like '%" +
scholarship.getScholarName() + "%");
        }
        if (String.isEmpty(scholarship.getScholarRank())) {
            sql.append(" and Scholar_rank=" + scholarship.getScholarRank() +
""");
        }
        if (scholarship.getScholarYear() != 0) {
            sql.append(" and Scholar_year=" +
String.valueOf(scholarship.getScholarYear()));
        }
        if (String.isEmpty(scholarship.getScholarIssuer())) {
            sql.append(" and Scholar_issuer like '%" +
scholarship.getScholarIssuer() + "%");
        }
        PreparedStatement pstmt =

```

```

con.prepareStatement(sql.toString().replaceFirst("and", "where"));
    return pstmt.executeQuery();
}

/*
 *奖学金删除
 */
public int delete(Connection con, int id) throws Exception {
    //更新获奖信息表
    String sql0 = "delete from stu_scholar where Scholar_id=?";
    PreparedStatement pstmt0 = con.prepareStatement(sql0);
    pstmt0.setString(1, String.valueOf(id));
    pstmt0.executeUpdate();

    //更新奖学金表
    String sql = "delete from scholarship where Scholar_id=?";
    PreparedStatement pstmt = con.prepareStatement(sql);
    pstmt.setString(1, String.valueOf(id));
    return pstmt.executeUpdate();
}

/*
 *奖学金更新
 */
public int update(Connection con, Scholarship scholarship) throws Exception {
    String sql="update scholarship set
    Scholar_name=?,Scholar_rank=?,Scholar_year=?,Scholar_issuer=?
    where
    Scholar_id=?";
    PreparedStatement pstmt=con.prepareStatement(sql);
    pstmt.setString(1,scholarship.getScholarName());
    pstmt.setString(2,scholarship.getScholarRank());
    pstmt.setString(3,String.valueOf(scholarship.getScholarYear()));
    pstmt.setString(4,scholarship.getScholarIssuer());
    pstmt.setString(5,String.valueOf(scholarship.getId()));
    return pstmt.executeUpdate();
}
}

```

#### 4、学生获奖记录增加、删除、修改、查询功能：

```
package pers.YuanchenBei.dbwork.dao;
```

```
import pers.YuanchenBei.dbwork.model.Scholarship;
```

```
import pers.YuanchenBei.dbwork.model.Student;
```

```

import pers.YuanchenBei.dbwork.utils.String_isEmpty;

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

public class StuScholarDao {
    /*
     *查询学生获奖信息
     */
    public ResultSet query(Connection con,Student student) throws Exception{
        StringBuffer sql=
            new StringBuffer("SELECT
student.Stu_id,student.Stu_name,major.Major_name,scholarship.Scholar_name
FROM stu_scholar LEFT JOIN student ON student.Stu_id=stu_scholar.Stu_id LEFT
JOIN scholarship ON scholarship.Scholar_id=stu_scholar.Scholar_id LEFT JOIN
major ON major.Major_id=student.Major_id");
        if(String_isEmpty.isEmpty(student.getId())){
            sql.append(" and student.Stu_id like '%" +student.getId()+"%'");
        }
        if(String_isEmpty.isEmpty(student.getName())){
            sql.append(" and student.Stu_name like '%" +student.getName()+"%'");
        }
        PreparedStatement
pstmt=con.prepareStatement(sql.toString().replaceFirst("and","where"));
        return pstmt.executeQuery();
    }

    /*
     *添加学生获奖信息
     */
    public int add(Connection con, String Stu_id, int Scholar_id) throws Exception{
        String sql="insert into stu_scholar values(?,?,?)";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,null);
        pstmt.setString(2,Stu_id);
        pstmt.setString(3,String.valueOf(Scholar_id));
        return pstmt.executeUpdate();
    }

    /*
     *查询获奖表信息(上面那个查询是详细查出每一个,这个查询是仅在获奖表
中查)
     */

```

```

        public ResultSet query_table(Connection con, String stu_id, String
scholar_id)throws Exception{
            StringBuffer sql = new StringBuffer("SELECT Stu_id,Scholar_id FROM
stu_scholar");
            if(String_isEmpty.isEmpty(stu_id)){
                sql.append(" and Stu_id like \"%"+stu_id+"%");
            }
            if(String_isEmpty.isEmpty(scholar_id)){
                sql.append(" and Scholar_id like \"%"+scholar_id+"%");
            }
            PreparedStatement
pstmt=con.prepareStatement(sql.toString().replaceFirst("and","where"));
            return pstmt.executeQuery();
        }

```

```

/*
 *修改获奖信息表
 */

```

```

public int update(Connection con,String stu_id,int scholar_id)throws Exception{
    String sql="update stu_scholar set Scholar_id=? where Stu_id=?";
    PreparedStatement pstmt=con.prepareStatement(sql);
    pstmt.setString(1,String.valueOf(scholar_id));
    pstmt.setString(2,stu_id);
    return pstmt.executeUpdate();
}

```

```

/*
 *删除部分的查询
 */

```

```

        public ResultSet query_del(Connection con,String stu_id,String scholar_id)
throws Exception{
            StringBuffer sql=
                new StringBuffer("SELECT
student.Stu_id,student.Stu_name,major.Major_name,scholarship.Scholar_name
FROM stu_scholar LEFT JOIN student ON student.Stu_id=stu_scholar.Stu_id LEFT
JOIN scholarship ON scholarship.Scholar_id=stu_scholar.Scholar_id LEFT JOIN
major ON major.Major_id=student.Major_id");
            if(String_isEmpty.isEmpty(stu_id)){
                sql.append(" and student.Stu_id like \"%"+stu_id+"%");
            }
            if(String_isEmpty.isEmpty(scholar_id)){
                sql.append(" and scholarship.Scholar_id="+scholar_id);
            }
            PreparedStatement

```

```

pstmt=con.prepareStatement(sql.toString().replaceFirst("and","where"));
    return pstmt.executeQuery();
}

/*
 *学生获奖信息删除
 */
public int delete(Connection con, String stu_id, String scholar_id) throws
Exception {
    //更新获奖信息表
    String sql = "delete from stu_scholar where Stu_id=? and Scholar_id=?";
    PreparedStatement pstmt = con.prepareStatement(sql);
    pstmt.setString(1, stu_id);
    pstmt.setString(2,scholar_id);
    return pstmt.executeUpdate();
}
}

```

## 5、报表功能：

```
package pers.YuanchenBei.dbwork.utils;
```

```

import net.sf.jasperreports.engine.JRException;
import net.sf.jasperreports.engine.JasperExportManager;
import net.sf.jasperreports.engine.JasperFillManager;
import net.sf.jasperreports.engine.JasperPrint;

```

```

import java.sql.Connection;
import java.util.HashMap;
import java.util.Map;

```

```
public class ReportExport {
```

```
    public static void export() throws JRException {
```

```

        Db_utils dbUtils=new Db_utils();
        Connection con=null;
        try {
            con=dbUtils.getCon();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

```

```
    Map<String, Object> rpt = new HashMap<String, Object>();
```

```

        JasperPrint                jasperPrint                =
JasperFillManager.fillReport("E:/report/report1.jasper", rpt,con);
        JasperExportManager.exportReportToPdfFile(jasperPrint,
"E:/report/report1.pdf");
    }
}

package pers.YuanchenBei.dbwork.view;

import javafx.application.HostServices;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.control.Alert;
import javafx.scene.control.SplitMenuButton;
import pers.YuanchenBei.dbwork.utils.ReportExport;

import java.io.File;
import java.io.IOException;

public class ReportExportController{
    private String filepath;
    private boolean selected=false;
    @FXML
    private SplitMenuButton top_button;

    @FXML
    void print_event(ActionEvent event) {
        try{
            if(selected){
                ReportExport.export();
                Runtime.getRuntime().exec("cmd.exe /C start AcroRd32 /P
"+filepath);
            }else{
                Alert alert = new Alert(Alert.AlertType.WARNING); // 创建一个消息对话框
                alert.setTitle("请先选择所要生成的报表!");
                alert.setHeaderText("您还没有选择所要生成的报表"); // 设置对话框的头部文本
                alert.setContentText("请您先选择所要生成的报表");
                alert.setResizable(false);
                alert.show();
            }
        }
    }
}

```

```

        }catch (Exception e){
            e.printStackTrace();
        }
    }

    @FXML
    void view_event(ActionEvent event) {
        try{
            if(selected){
                ReportExport.export();
                Runtime.getRuntime().exec("cmd.exe /C start AcroRd32
"+filepath);
            }else{
                Alert alert = new Alert(Alert.AlertType.WARNING); // 创建一个消息对话框
                alert.setTitle("请先选择所要生成的报表!");
                alert.setHeaderText("您还没有选择所要生成的报表"); // 设置对话框的头部文本
                alert.setContentText("请您先选择所要生成的报表");
                alert.setResizable(false);
                alert.show();
            }
        }catch (Exception e){
            e.printStackTrace();
        }
    }

    @FXML
    void scholar_id5(ActionEvent event) {
        top_button.setText("2019-2020 学年国家励志奖学金");
        filepath="E:/report/report1.pdf";
        selected=true;
    }

    Parent createNode() throws IOException {
        return FXMLLoader.load(getClass().getResource("report_export.fxml"));
    }
}

```