第 6 章 “数据库原理及应用”课程作业

姓名：袁昊男 学号：2018091618008

在一个成绩管理系统数据库 GradeDB 中，定义如下关系表：

STUDENT(SID, SName, Age, Sex)
GRADE(*SID*, *CID*, Score, Note)
COURSE(CID, CName, Teacher)

（1）编写 SQL 程序创建数据库关系表，并插入 10 个学生的 2 门课程（“数据库原理及应用”、“数据结构与算法”）成绩数据；

（2）编写触发器程序实现 STUDENT 表、COURSE 表与 GRADE 表之间的级联更新、级联删除；

（3）编写存储过程程序实现统计各课程不及格学生人数，并在屏幕输出；

（4）用 Mybatis+Servlet+jsp 编程实现查询课程学生成绩列表页面输出功能。

作业要求：在 PostgreSQL 数据库中，使用 SQL 程序创建 GradeDB 数据库及其数据库表，并插入样本数据。分别使用 PL/pgSQL 编程方法和 Java Web 编程方法，实现数据库后端编程和应用前端编程程序。给出每个问题解决的步骤、SQL 程序、PL/pgSQL 程序和 Java Web 程序，及其运行结果界面，并对结果进行说明。

1.1 数据表字段结构定义

表 1 学生表（STUDENT）

字段名称	字段编码	数据类型	字段大小	必填字段	备注
学号	SID	Char	13	是	主键
姓名	SName	Varchar	10	是	
年龄	Age	Char	2	是	
性别	Sex	Char	2	是	取值范围：男、女

表 2 成绩表（GRADE）

字段名称	字段编码	数据类型	字段大小	必填字段	备注
学号	SID	Char	13	是	主键，外键
课程号	CID	Char	8	是	主键，外键
成绩	Score	Char	3	是	取值范围：[0, 100]
备注	Note	Varchar	50	否	

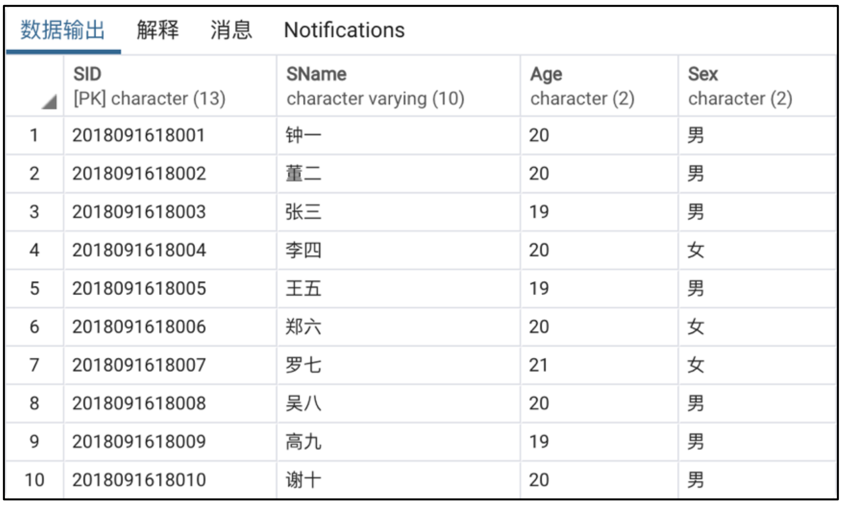
表 3 课程表（COURSE）

字段名称	字段编码	数据类型	字段大小	必填字段	备注
课程号	CID	Char	8	是	主键
课程名	CName	Varchar	20	是	
教师	Teacher	Varchar	10	是	

1.2　数据库及数据表创建 SQL 语句

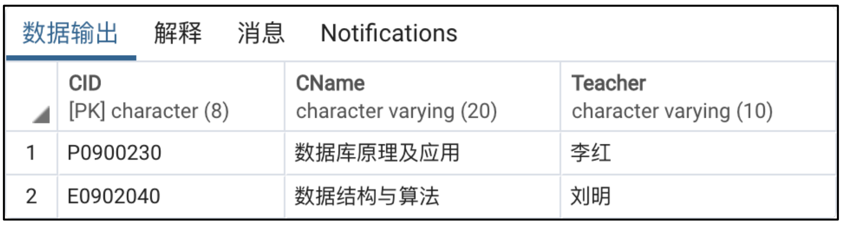
```sql
CREATE DATABASE "GradeDB"
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    CONNECTION LIMIT = -1;


CREATE TABLE public."STUDENT"
(
    "SID" character(13) NOT NULL,
    "SName" character varying(10) NOT NULL,
    "Age" character(2) NOT NULL,
    "Sex" character(2) NOT NULL,
    PRIMARY KEY ("SID"),
    CONSTRAINT "Sex" CHECK ("Sex" IN ('男', '女'))
)
WITH (
    OIDS = FALSE
);

ALTER TABLE public."STUDENT"
    OWNER to postgres;


CREATE TABLE public."GRADE"
(
    "SID" character(13) NOT NULL,
    "CID" character(8) NOT NULL,
    "Score" character(3) NOT NULL,
    "Note" character varying(50),
    PRIMARY KEY ("SID", "CID")
)
WITH (
    OIDS = FALSE
);

ALTER TABLE public."GRADE"
    OWNER to postgres;


CREATE TABLE public."COURSE"
(
    "CID" character(8) NOT NULL,
    "CName" character varying(20) NOT NULL,
    "Teacher" character varying(10) NOT NULL,
    PRIMARY KEY ("CID")
)
WITH (
    OIDS = FALSE
);

ALTER TABLE public."COURSE"
    OWNER to postgres;
```

1.3 数据查询结果



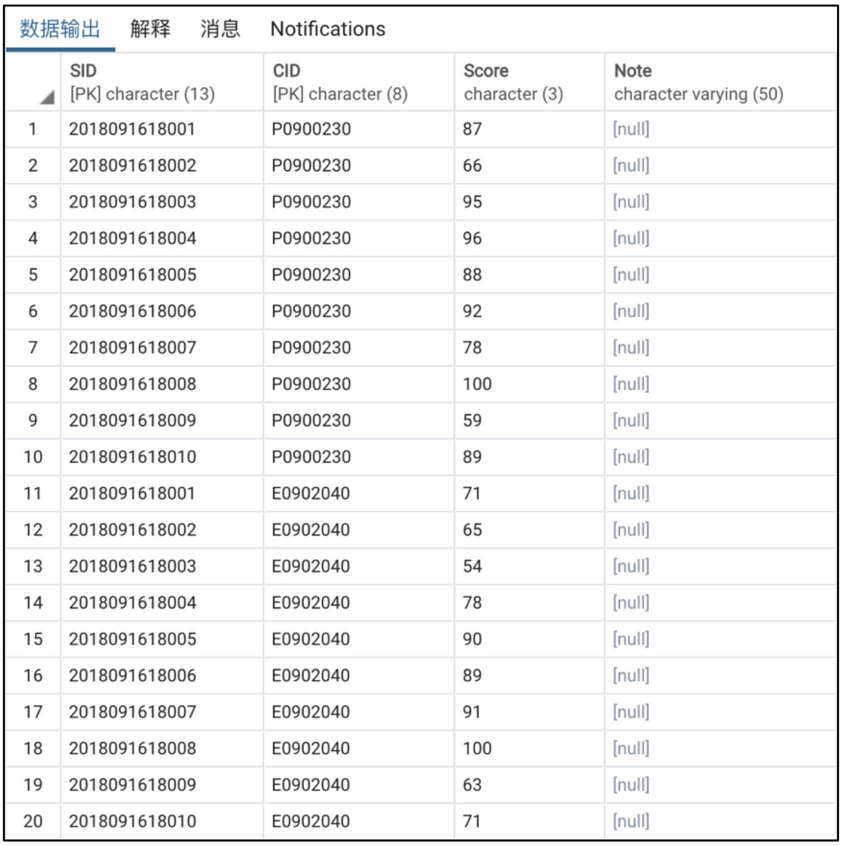
	SID [PK] character (13)	SName character varying (10)	Age character (2)	Sex character (2)
1	2018091618001	钟一	20	男
2	2018091618002	董二	20	男
3	2018091618003	张三	19	男
4	2018091618004	李四	20	女
5	2018091618005	王五	19	男
6	2018091618006	郑六	20	女
7	2018091618007	罗七	21	女
8	2018091618008	吴八	20	男
9	2018091618009	高九	19	男
10	2018091618010	谢十	20	男

图 1　学生表数据查询结果



	CID [PK] character (8)	CName character varying (20)	Teacher character varying (10)
1	P0900230	数据库原理及应用	李红
2	E0902040	数据结构与算法	刘明

图 2　课程表数据查询结果



	SID [PK] character (13)	CID [PK] character (8)	Score character (3)	Note character varying (50)
1	2018091618001	P0900230	87	[null]
2	2018091618002	P0900230	66	[null]
3	2018091618003	P0900230	95	[null]
4	2018091618004	P0900230	96	[null]
5	2018091618005	P0900230	88	[null]
6	2018091618006	P0900230	92	[null]
7	2018091618007	P0900230	78	[null]
8	2018091618008	P0900230	100	[null]
9	2018091618009	P0900230	59	[null]
10	2018091618010	P0900230	89	[null]
11	2018091618001	E0902040	71	[null]
12	2018091618002	E0902040	65	[null]
13	2018091618003	E0902040	54	[null]
14	2018091618004	E0902040	78	[null]
15	2018091618005	E0902040	90	[null]
16	2018091618006	E0902040	89	[null]
17	2018091618007	E0902040	91	[null]
18	2018091618008	E0902040	100	[null]
19	2018091618009	E0902040	63	[null]
20	2018091618010	E0902040	71	[null]

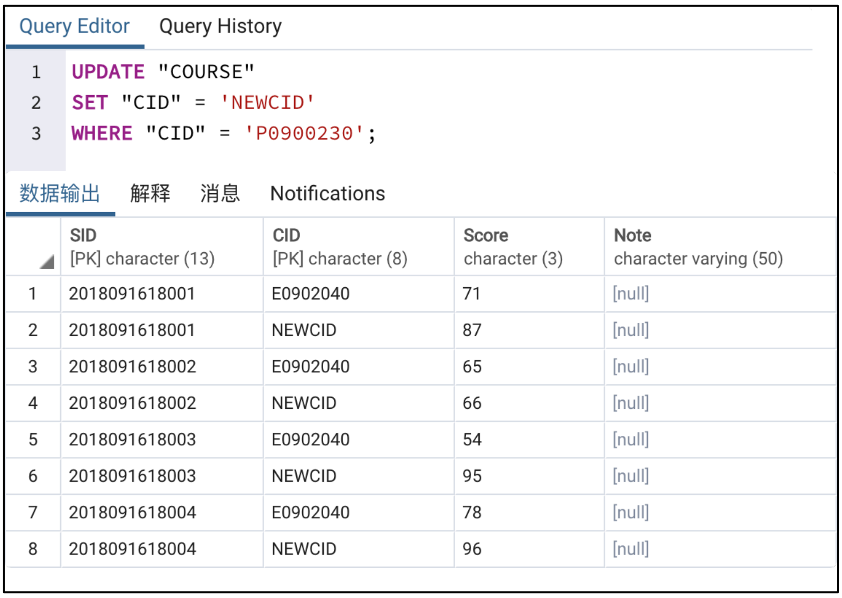
图 3　成绩表数据查询结果

2.1 触发器创建 SQL 语句

```sql
CREATE FUNCTION delete_sid ()
RETURNS TRIGGER AS $delete$
BEGIN
    DELETE FROM "GRADE"
    WHERE "GRADE"."SID" = OLD."SID";
RETURN OLD;
END;
$delete$ LANGUAGE plpgsql;


CREATE TRIGGER delete_trigger_sid
AFTER DELETE ON "STUDENT"
FOR EACH ROW EXECUTE PROCEDURE delete_sid();


CREATE FUNCTION delete_cid ()
RETURNS TRIGGER AS $delete$
BEGIN
    DELETE FROM "GRADE"
    WHERE "GRADE"."CID" = OLD."CID";
RETURN OLD;
END;
$delete$ LANGUAGE PLPGSQL;


CREATE TRIGGER delete_trigger_cid
AFTER DELETE ON "COURSE"
FOR EACH ROW EXECUTE PROCEDURE delete_cid();


CREATE OR REPLACE FUNCTION update_sid ()
RETURNS TRIGGER AS $update$
BEGIN
    UPDATE "GRADE"
    SET "GRADE"."SID" = NEW."SID"
    WHERE "GRADE"."SID" = OLD."SID";
RETURN NEW;
END;
$update$ LANGUAGE plpgsql;


CREATE TRIGGER update_trigger_sid
AFTER UPDATE ON "STUDENT"
FOR EACH ROW EXECUTE PROCEDURE update_sid();


CREATE OR REPLACE FUNCTION update_cid ()
RETURNS TRIGGER AS $update$
BEGIN
    UPDATE "GRADE"
    SET "GRADE"."CID" = NEW."CID"
    WHERE "GRADE"."CID" = OLD."CID";
RETURN NEW;
END;
$update$ LANGUAGE plpgsql;

```

```sql
CREATE TRIGGER update_trigger_cid
AFTER UPDATE ON "COURSE"
FOR EACH ROW EXECUTE PROCEDURE update_cid();
```

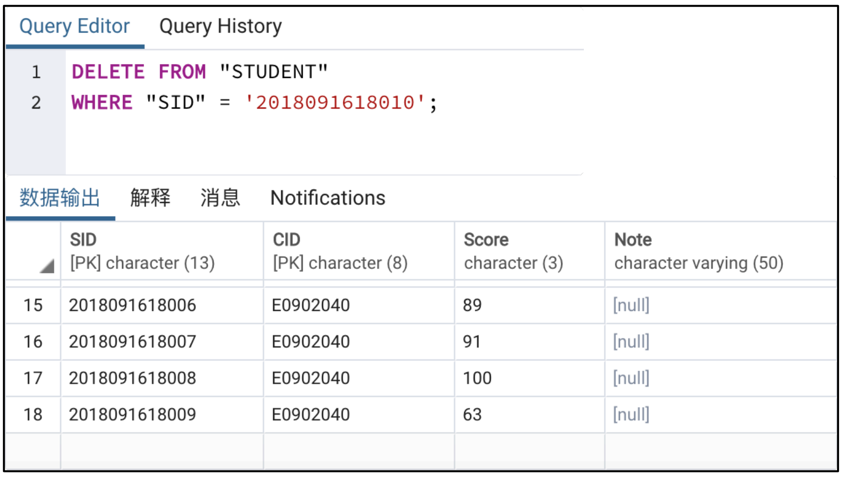
2.2 级联更新、删除操作结果



```sql
UPDATE "COURSE"
SET "CID" = 'NEWCID'
WHERE "CID" = 'P0900230';
```

	SID [PK] character (13)	CID [PK] character (8)	Score character (3)	Note character varying (50)
1	2018091618001	E0902040	71	[null]
2	2018091618001	NEWCID	87	[null]
3	2018091618002	E0902040	65	[null]
4	2018091618002	NEWCID	66	[null]
5	2018091618003	E0902040	54	[null]
6	2018091618003	NEWCID	95	[null]
7	2018091618004	E0902040	78	[null]
8	2018091618004	NEWCID	96	[null]

图 4　级联更新后成绩表查询结果



```sql
DELETE FROM "STUDENT"
WHERE "SID" = '2018091618010';
```

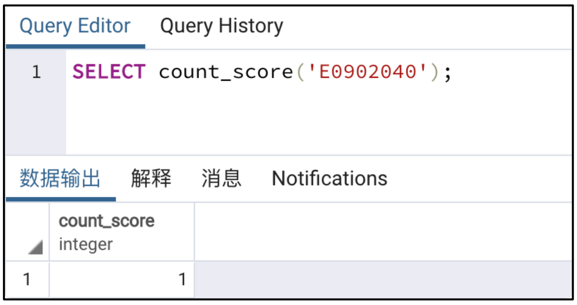
	SID [PK] character (13)	CID [PK] character (8)	Score character (3)	Note character varying (50)
15	2018091618006	E0902040	89	[null]
16	2018091618007	E0902040	91	[null]
17	2018091618008	E0902040	100	[null]
18	2018091618009	E0902040	63	[null]

图 5　级联删除后成绩表查询结果

3.1 存储过程创建 SQL 语句

```sql
CREATE OR REPLACE FUNCTION count_score (courseid text)
RETURNS integer AS $count$
DECLARE
    count_fail integer;
BEGIN
    SELECT SUM(CASE WHEN CAST("Score" AS integer) < 60
        THEN 1 ELSE 0 END)
    FROM "GRADE" WHERE "CID" = courseid
    INTO count_fail;
    RETURN count_fail;
END;
$count$ LANGUAGE plpgsql
```

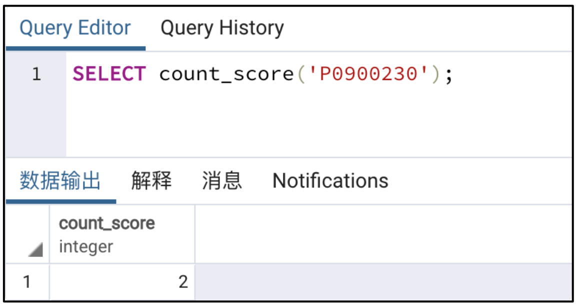
3.2 不及格学生人数统计结果



```
SELECT count_score('E0902040');
```

```
count_score
integer
1        1
```

图 6 “数据结构与算法”不及格人数查询结果



```
SELECT count_score('P0900230');
```

```
count_score
integer
1        2
```

图 7 “数据库原理及应用”不及格人数查询结果

4.1 GradeDB 中创建 gradeview 视图

```sql
create view GradeView as
select "STUDENT"."sid", "STUDENT"."sname", "STUDENT"."sex",
"COURSE"."cid", "COURSE"."cname", "COURSE"."teacher",
"GRADE"."score", "GRADE"."note"
from "GRADE" join  "STUDENT" on "GRADE"."sid" = "STUDENT"."sid"
join "COURSE" on "GRADE"."cid" = "COURSE"."cid";
```

4.2 创建 com.stu.entity 包与 Student 类

```java
package com.stu.entity;


public class Student {
    private String sid;
    private String sname;
    private String sex;
    private String cid;
    private String cname;
    private String teacher;
    private String score;
    private String note;

    public Student() { }

    public String getsid(){ return sid; }
    public String getsname(){ return sname; }
    public String getsex(){ return sex; }
    public String getcid(){ return cid; }
    public String getcname(){ return cname; }
    public String getteacher(){ return teacher; }
    public String getscore(){ return score; }
    public String getnote(){ return note; }

    public void setsid(String sid) { this.sid = sid; }
    public void setsname(String sname) { this.sname = sname; }
    public void setsex(String sex) { this.sex = sex; }
    public void setcid(String cid) { this.cid = cid; }
    public void setcname(String cname) { this.cname = cname; }
    public void setteacher(String teacher) { this.teacher = teacher; }
    public void setscore(String score) { this.score = score; }
    public void setnote(String note) { this.note = note; }

    public Student(String sid, String sname, String sex, String cid, String cname, String teacher, String score, String note) {
        this.sid=sid;
        this.sname=sname;
        this.sex=sex;
        this.cid=cid;
        this.cname=cname;
        this.teacher=teacher;
        this.score=score;
        this.note=note;
    }
    @Override
    public String toString() {
        return "Student{" +
                "sid=" + sid + ", sname=" + sname + ", sex=" + sex + ", cid=" + cid +
                ", cname=" + cname + ", teacher=" + teacher + ", score=" + score + note + ", note=" + note +
                '}';
    }
}
```

4.3 配置全局映射文件 mybatis.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
        PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
        "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
    <environments default="default">
        <environment id="default">
            <transactionManager type="JDBC"></transactionManager>
            <dataSource type="POOLED">
                <property name="driver" value="org.postgresql.Driver"/>
                <property name="url" value="jdbc:postgresql://localhost:5432/GradeDB"/>
                <property name="username" value="postgres"/>
                <property name="password" value="********"/>
            </dataSource>
        </environment>
    </environments>
    <mappers>
        <mapper resource="com/stu/mapper/StudentMapper.xml"/>
    </mappers>
</configuration>
```

4.4　配置局域映射文件 StudentMapper.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
        PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
        "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.mapper.StudentMapper">
    <select id="selAll" resultType="com.stu.entity.Student">
        select * from gradeview where sid = #{sid}
    </select>
</mapper>
```

4.5　创建 com.stu.service 包与 StudentService 接口

```java
package com.stu.service;
import com.stu.entity.Student;

import java.io.IOException;
import java.util.List;

public interface PeopleService {
    List<People> show() throws IOException;
}
```

4.6　创建 StudentServiceImpl 类

```java
package com.stu.service.impl;

import com.stu.entity.Student;
import com.stu.service.StudentService;
import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;

import java.io.IOException;
import java.io.InputStream;
import java.util.List;

public class PeopleServiceImpl implements PeopleService {
    @Override
    public List<People> show() throws IOException {
        InputStream is = Resources.getResourceAsStream("mybatis.xml");
        SqlSessionFactory factory = new SqlSessionFactoryBuilder().build(is);
        SqlSession session = factory.openSession();
        List<People> list = session.selectList("com.stu.mapper.StudentMapper.selAll");
        session.close();
        return list;
    }
}
```

4.7　创建 com.stu.servlet 包与 ShowServlet 类

```java
package com.stu.servlet;


```

```java
import com.stu.entity.Student;
import com.stu.service.StudentService;
import com.stu.service.impl.StudentServiceImpl;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.util.List;

@WebServlet("/show")
public class ShowServlet extends HttpServlet{
    private StudentService studentService = new StudentServiceImpl();
    @Override
    protected void service(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        List<Student> list = studentService.show();
        req.setAttribute("list", list);
        req.getRequestDispatcher("/index.jsp").forward(req, resp);
    }
}
```

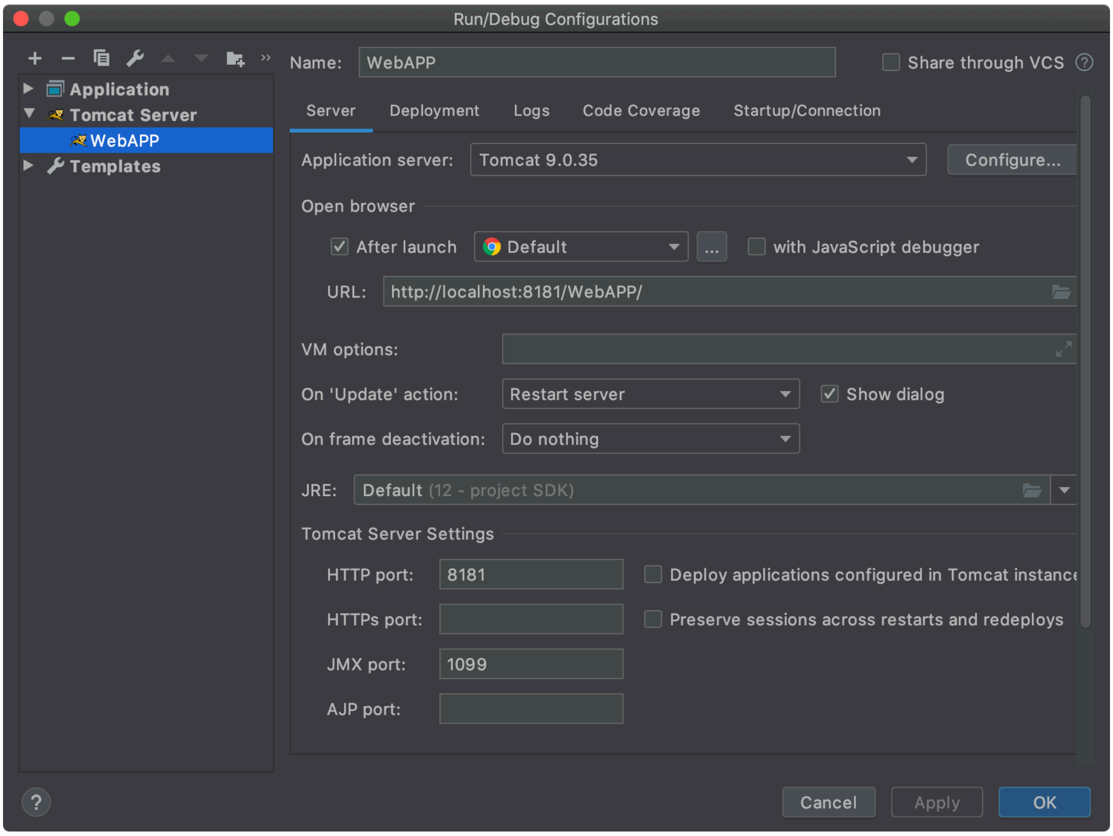
4.8 创建 com.stu.test 包与 Test 类

```java
package com.stu.servlet;

import com.stu.entity.Student;
import com.stu.service.StudentService;
import com.stu.service.impl.StudentServiceImpl;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.util.List;

@WebServlet("/show")
public class ShowServlet extends HttpServlet{
    private StudentService studentService = new StudentServiceImpl();

@Override
    protected void service(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        List<Student> list = studentService.show();

        req.setAttribute("list", list);
        req.getRequestDispatcher("/index.jsp").forward(req, resp);
    }
}
```

4.9 创建 JSP 页面

```jsp
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page import="com.test1.db.StudentDB" %>
<%@ page import="java.util.List" %>
<%@ page import="com.test1.entity.Student" %>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>学生成绩查询系统</title>
    <style>
        #footer {
            height: 40px;
            line-height: 40px;
            position: fixed;
            bottom: 0;
            width: 100%;
            text-align: center;
            background: #333;
            color: #fff;
            font-family: Arial;
            font-size: 12px;
            letter-spacing: 1px;
        }
    </style>
</head>

<div align="center">
    <h1>学生成绩查询系统</h1>
</div>

<div align="center">
    <div style="display: inline-block;margin-right: 20px">
        <form action="student.jsp">
        <input id="sid" name="sid" placeholder="按学号查询" style="width: 120px">
        <input type="submit" value="查询" style="color: white;background-color: black">
        </form>
    </div>

    <div style="display: inline-block;margin-right: 20px;margin-left: 20px">
        <form action="student.jsp">
            <input id="sname" name="sname" placeholder="按姓名查询" style="width: 80px">
            <input type="submit" value="查询" style="color: white;background-color: black">
        </form>
    </div>

    <div style="display: inline-block;margin-right: 20px;margin-left: 20px">
        <form action="student.jsp">
            <input id="cid" name="cid" placeholder="按课程号查询" style="width: 100px">
        <input type="submit" value="查询" style="color: white;background-color: black">
        </form>
    </div>

    <div style="display: inline-block;margin-left: 20px">
        <form action="student.jsp">
            <input id="cname" name="cname" placeholder="按课程名查询" style="width: 130px">
            <input type="submit" value="查询" style="color: white;background-color: black">
        </form>
    </div>
</div>

<div align="center">
<table border="1" cellspacing="0">
    <tr>
        <td bgcolor="#d3d3d3" align="center" style="width: 145px;height: auto">学号</td>
        <td bgcolor="#d3d3d3" align="center" style="width: 50px;height: auto">姓名</td>
        <td bgcolor="#d3d3d3" align="center" style="width: 50px;height: auto">性别</td>
        <td bgcolor="#d3d3d3" align="center" style="width: 95px;height: auto">课程号</td>
        <td bgcolor="#d3d3d3" align="center" style="width: 170px;height: auto">课程名</td>
        <td bgcolor="#d3d3d3" align="center" style="width: 70px;height: auto">任课教师</td>
        <td bgcolor="#d3d3d3" align="center" style="width: 50px;height: auto">成绩</td>
        <td bgcolor="#d3d3d3" align="center" style="width: 80px;height: auto">备注</td>
    </tr>

    <%
        StudentDB sdb = new StudentDB();
        String sname = request.getParameter("sname");
        String sid = request.getParameter("sid");
        String cname = request.getParameter("cname");
        String cid = request.getParameter("cid");
        String sql = null;
        if (sid != null) {
            sql = "select * from gradeview where sid like '%" + sid + "%' order by cid,sid";
        }
        else if(sname != null) {
            sql = "select * from gradeview where sname like '%" + sname + "%' order by cid,sid";
        }
        else if (cid != null) {
            sql = "select * from gradeview where cid like '%" + cid + "%' order by sid,cid";
        }
        else if (cname != null) {
            sql = "select * from gradeview where cname like '%" + cname + "%' order by sid,cid";
        }
        else {
            sql = "select * from gradeview order by cid, sid";
        }

        List<Student> lst = sdb.getStudent(sql);
        request.setAttribute("stuLst", lst);
    %>
    <c:forEach items="${stuLst}" var="stu">
        <tr id="${stu.sid}">
            <td align="center">${stu.sid}</td>
            <td align="center">${stu.sname}</td>
            <td align="center">${stu.sex}</td>
            <td align="center">${stu.cid}</td>
            <td align="center">${stu.cname}</td>
            <td align="center">${stu.teacher}</td>
            <td align="center">${stu.score}</td>
            <td align="center" style="color: red">${stu.note}</td>
        </tr>
    </c:forEach>
</table>
</div>
<div align="center">
    <a href="student.jsp">主查询页面</a>
</div>

<div id="footer">CopyRight © 2020 汤泡饭</div>
</body>
</html>
```

4.10　配置 Tomcat 服务器



Name: WebAPP

□ Share through VCS

Application server: Tomcat 9.0.35

Open browser

☑ After launch　Default　□ with JavaScript debugger

URL: http://localhost:8181/WebAPP/

VM options:

On 'Update' action: Restart server　☑ Show dialog

On frame deactivation: Do nothing

JRE: Default (12 - project SDK)

Tomcat Server Settings

HTTP port: 8181　□ Deploy applications configured in Tomcat instance

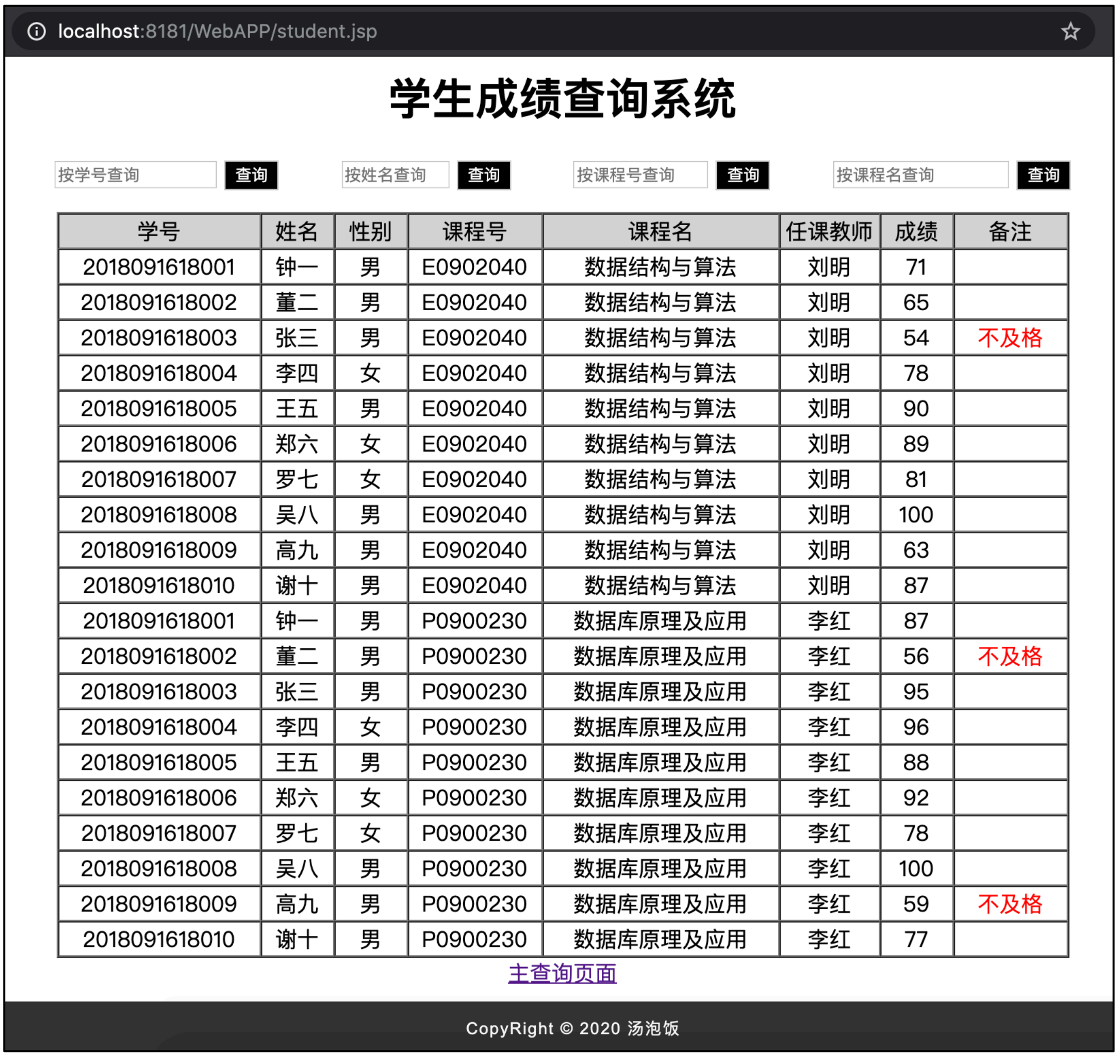
HTTPs port:　□ Preserve sessions across restarts and redeploys

JMX port: 1099

AJP port:

图 8　IDEA 中配置 Tomcat 服务器

4.11 运行页面



localhost:8181/WebAPP/student.jsp

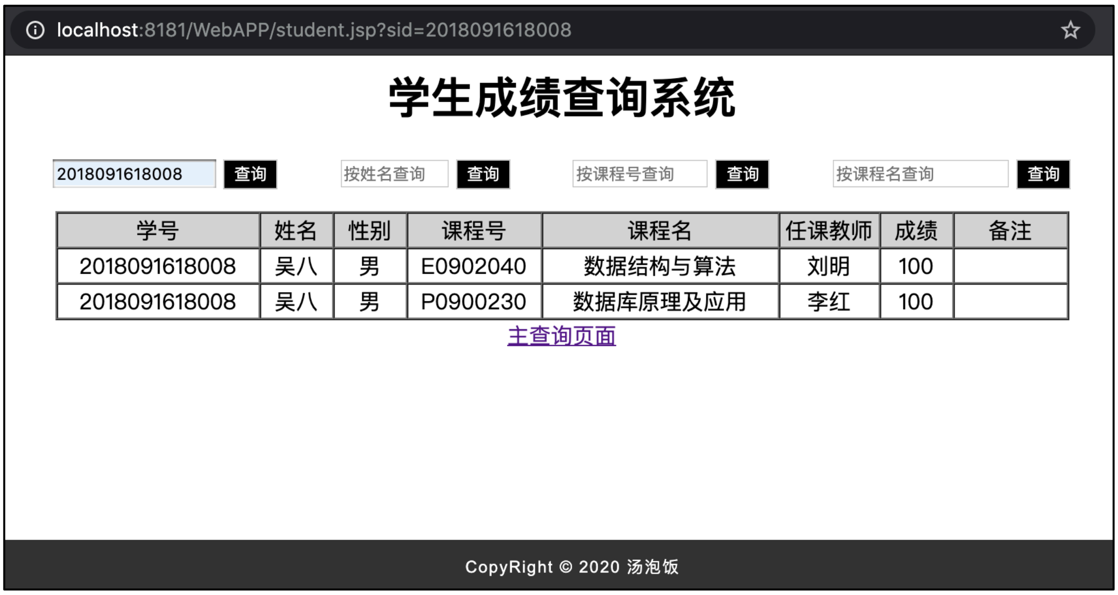
学生成绩查询系统

学号	姓名	性别	课程号	课程名	任课教师	成绩	备注
2018091618001	钟一	男	E0902040	数据结构与算法	刘明	71	
2018091618002	董二	男	E0902040	数据结构与算法	刘明	65	
2018091618003	张三	男	E0902040	数据结构与算法	刘明	54	不及格
2018091618004	李四	女	E0902040	数据结构与算法	刘明	78	
2018091618005	王五	男	E0902040	数据结构与算法	刘明	90	
2018091618006	郑六	女	E0902040	数据结构与算法	刘明	89	
2018091618007	罗七	女	E0902040	数据结构与算法	刘明	81	
2018091618008	吴八	男	E0902040	数据结构与算法	刘明	100	
2018091618009	高九	男	E0902040	数据结构与算法	刘明	63	
2018091618010	谢十	男	E0902040	数据结构与算法	刘明	87	
2018091618001	钟一	男	P0900230	数据库原理及应用	李红	87	
2018091618002	董二	男	P0900230	数据库原理及应用	李红	56	不及格
2018091618003	张三	男	P0900230	数据库原理及应用	李红	95	
2018091618004	李四	女	P0900230	数据库原理及应用	李红	96	
2018091618005	王五	男	P0900230	数据库原理及应用	李红	88	
2018091618006	郑六	女	P0900230	数据库原理及应用	李红	92	
2018091618007	罗七	女	P0900230	数据库原理及应用	李红	78	
2018091618008	吴八	男	P0900230	数据库原理及应用	李红	100	
2018091618009	高九	男	P0900230	数据库原理及应用	李红	59	不及格
2018091618010	谢十	男	P0900230	数据库原理及应用	李红	77	

主查询页面

CopyRight © 2020 汤泡饭

图 9 主查询页面



localhost:8181/WebAPP/student.jsp?sid=2018091618008

学生成绩查询系统

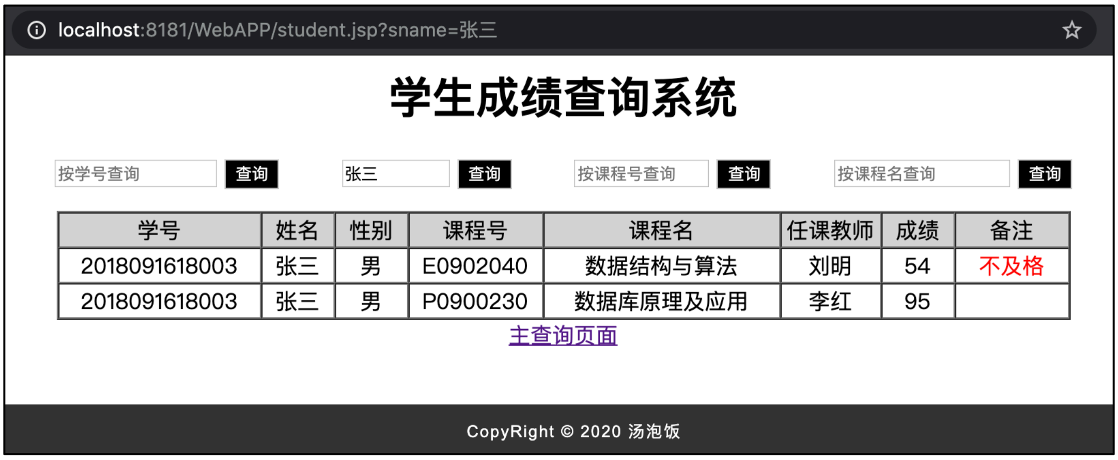
2018091618008

学号	姓名	性别	课程号	课程名	任课教师	成绩	备注
2018091618008	吴八	男	E0902040	数据结构与算法	刘明	100	
2018091618008	吴八	男	P0900230	数据库原理及应用	李红	100	

主查询页面

CopyRight © 2020 汤泡饭

图 10 按学号精确查询结果



localhost:8181/WebAPP/student.jsp?sname=张三

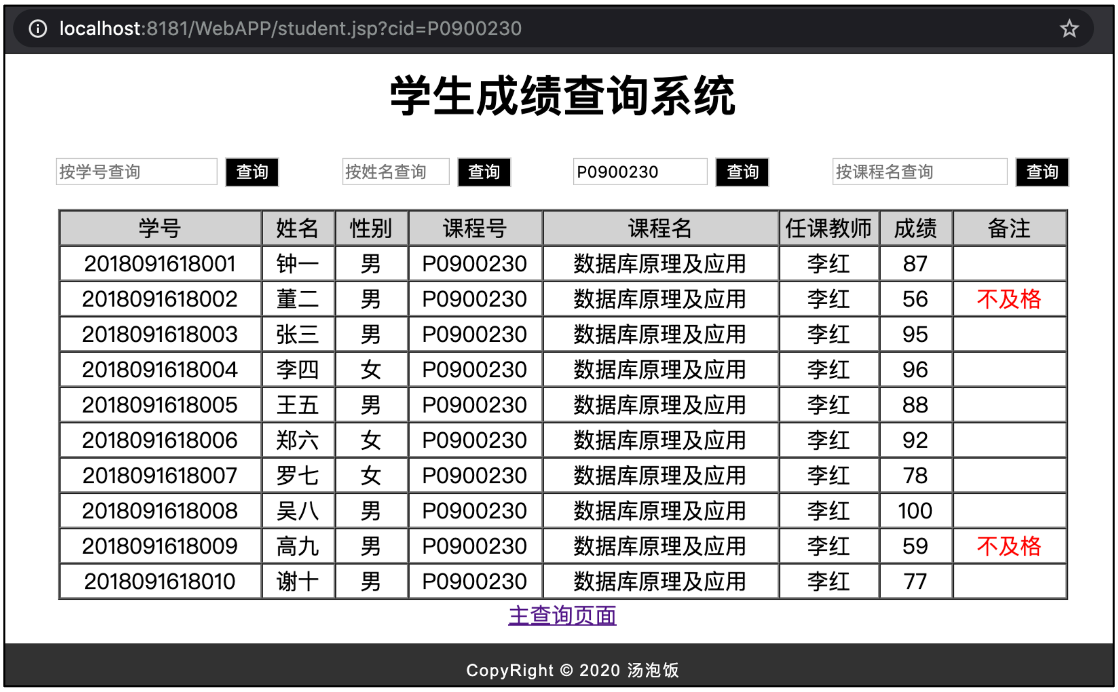
学生成绩查询系统

张三

学号	姓名	性别	课程号	课程名	任课教师	成绩	备注
2018091618003	张三	男	E0902040	数据结构与算法	刘明	54	不及格
2018091618003	张三	男	P0900230	数据库原理及应用	李红	95	

CopyRight © 2020 汤泡饭

图 11　按姓名精确查询结果



localhost:8181/WebAPP/student.jsp?cid=P0900230

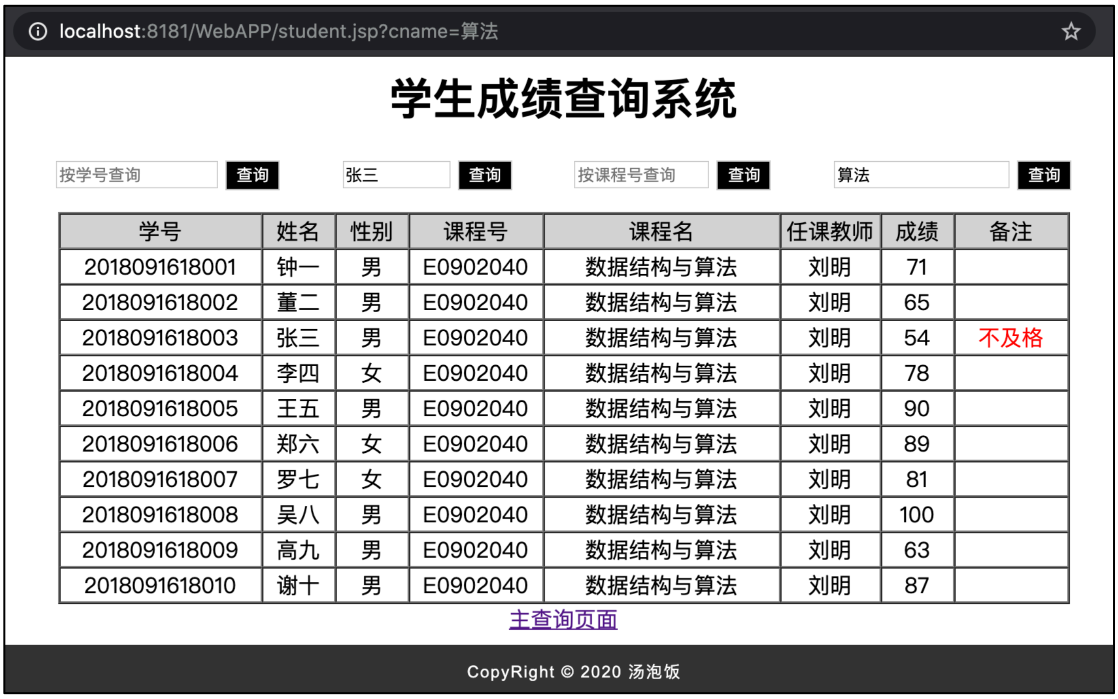
学生成绩查询系统

P0900230

学号	姓名	性别	课程号	课程名	任课教师	成绩	备注
2018091618001	钟一	男	P0900230	数据库原理及应用	李红	87	
2018091618002	董二	男	P0900230	数据库原理及应用	李红	56	不及格
2018091618003	张三	男	P0900230	数据库原理及应用	李红	95	
2018091618004	李四	女	P0900230	数据库原理及应用	李红	96	
2018091618005	王五	男	P0900230	数据库原理及应用	李红	88	
2018091618006	郑六	女	P0900230	数据库原理及应用	李红	92	
2018091618007	罗七	女	P0900230	数据库原理及应用	李红	78	
2018091618008	吴八	男	P0900230	数据库原理及应用	李红	100	
2018091618009	高九	男	P0900230	数据库原理及应用	李红	59	不及格
2018091618010	谢十	男	P0900230	数据库原理及应用	李红	77	

CopyRight © 2020 汤泡饭

图 12　按课程号精确查询结果



localhost:8181/WebAPP/student.jsp?cname=算法

学生成绩查询系统

张三

算法

学号	姓名	性别	课程号	课程名	任课教师	成绩	备注
2018091618001	钟一	男	E0902040	数据结构与算法	刘明	71	
2018091618002	董二	男	E0902040	数据结构与算法	刘明	65	
2018091618003	张三	男	E0902040	数据结构与算法	刘明	54	不及格
2018091618004	李四	女	E0902040	数据结构与算法	刘明	78	
2018091618005	王五	男	E0902040	数据结构与算法	刘明	90	
2018091618006	郑六	女	E0902040	数据结构与算法	刘明	89	
2018091618007	罗七	女	E0902040	数据结构与算法	刘明	81	
2018091618008	吴八	男	E0902040	数据结构与算法	刘明	100	
2018091618009	高九	男	E0902040	数据结构与算法	刘明	63	
2018091618010	谢十	男	E0902040	数据结构与算法	刘明	87	

CopyRight © 2020 汤泡饭

图 13　按课程名模糊查询结果