数据库应用课程设计

——田径运动会管理系统

个人信息： 121320107 金娟

121320207 雷前春

121320205 倪梦妃

指导教师： 丁祥武

时间：2015年4月19日

目录

[1 绪论 3](#_Toc417397097)

[1.1 系统设计背景 3](#_Toc417397098)

[1.2 系统设计的原则 3](#_Toc417397099)

[1.3 系统设计的目标 4](#_Toc417397100)

[2 相关技术介绍 5](#_Toc417397101)

[2.1 数据库 5](#_Toc417397102)

[2.2 JDBC 5](#_Toc417397103)

[2.3 Web技术 5](#_Toc417397104)

[2.4 JSP 5](#_Toc417397105)

[2.5 Ajax 5](#_Toc417397106)

[3 需求分析 6](#_Toc417397107)

[3.1 运动员 6](#_Toc417397108)

[3.1.1 基本信息管理 6](#_Toc417397109)

[3.1.2 报名 6](#_Toc417397110)

[3.1.3 查看比赛 6](#_Toc417397111)

[3.2 管理员 7](#_Toc417397112)

[3.2.1 赛程赛事管理 7](#_Toc417397113)

[3.2.2 各系管理 7](#_Toc417397114)

[3.2.3 运动会组委会管理 7](#_Toc417397115)

[3.2.4 信息需求 8](#_Toc417397116)

[4 概要设计 9](#_Toc417397117)

[4.1 系统的功能模块框图 9](#_Toc417397118)

[4.2 E-R图 9](#_Toc417397119)

[5 详细设计 10](#_Toc417397120)

[5.1 运动员 10](#_Toc417397121)

[5.1.1 功能模块详细设计 10](#_Toc417397122)

[5.2 管理员 11](#_Toc417397123)

[5.2.1 赛事赛程管理 11](#_Toc417397124)

[5.2.2 各系管理 14](#_Toc417397125)

[5.2.3 运动会组委会管理 15](#_Toc417397126)

[5.3 逻辑表结构的设计 16](#_Toc417397127)

[6 系统实现 17](#_Toc417397128)

[6.1 开发环境搭建 17](#_Toc417397129)

[6.2 数据库的实现 17](#_Toc417397130)

[6.2.1 关系表 17](#_Toc417397131)

[6.2.2 数据库实施代码 19](#_Toc417397132)

[6.3 数据库的连接 21](#_Toc417397133)

[6.4 主要功能模块的实现 23](#_Toc417397134)

[6.4.1 运动员 23](#_Toc417397135)

[6.4.2 管理员 26](#_Toc417397136)

[7 总结 35](#_Toc417397137)

[参考文献 35](#_Toc417397138)

# 绪论

## 系统设计背景

随着科技的日益发展，学校的信息管理数字化已经成为了体现学校综合实力的重要指标。田径运动会管理作为信息管理的一个组成部分，也是不可或缺的。而且目前学校的田径运动会管理工作在很大程度上还要依赖人工操作，如通过电子表格、文档等等办公软件来处理大量的数据。这种方式不仅给运动会信息管理人员带来了繁重的工作量，而且造成管理员和运动员之间沟通的缺乏，很难及时的反映学生的身体素质现状。在当前的形式下，通过IT技术等信息化手段，运动会管理信息化，智能化，以加强赛程数据管理，为决策层提供及时而有效的参考依据，从而达到提升学校信息管理水平的目的，是当今学校信息管理的大势所趋。

本次为学校举行的田径运动会设计运动会管理系统。要求对运动员（包括比赛项目和比赛成绩），比赛项目的设置等进行管理。具体来讲，田径运动会包括若干个比赛项目，由学校设置比赛项目和比赛规则，运动员根据比赛规则报名参赛，裁判对运动员进行分组分道，运动员参加比赛，由裁判评定和处理成绩，但须公平公正。

## 系统设计的原则

本系统从学校田径运动会的实际情况出发，坚持用户方便易用，运动会期间数据实时共享，从本质上改变以前繁琐、重复和低效率的手工操作的工作方式，提高了运动会管理数据的准确性和及时性，因而能更好地为学生和决策人服务。

为了该系统能更好的为运动会管理服务，本系统的建立应该遵循以下四个基本原则：

（1）规范化原则

先对学校田径运动会管理业务流程的综合分析，建立相关的规范的运动会管理制度，利用现有的信息技术实现管理的规范化和高效性。

（2）兼容性原则

本系统应能够与学校现有的管理系统在技术上兼容，在技术条件允许的前

提下，最大限度地利用现有的硬件、软件平台。使得在更大程度上提高管理部门的数据和信息资源的共享，为以后的领导决策提供必要的信息支持。

（3）易用性原则

根据学校田径运动会的业务需要，给各类用户提供高效，友好的用户界面，

交互更加的人性化和易操作。

（4）创新性原则

在系统建设过程中，将结合信息系统的特点，基于田径运动会流程管理思想

对现有的业务流程进行重新梳理，优化组合。在管理制度和业务实现等方面体现该系统的创新和开拓的思路。

## 系统设计的目标

为了使系统更好的为田径运动会管理服务，本系统的建立需要实现以下两个目标：

1. 系统具有较广的覆盖面

系统将根据整个田径运动会组织管理的具体情况量身定制，应该包括运动

会管理的各个方面，从比赛项目信息到运动员信息、从报名信息到比赛成绩信息等都应该包含到位。

1. 系统具有较高的安全性

我们除了视窗系统本身提供的安全措施外，系统还应该具有以下几项安全

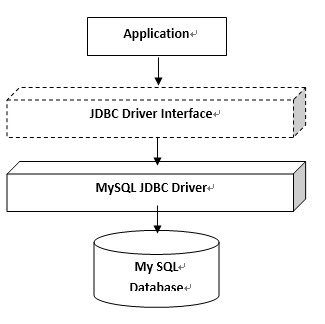
性能：数据库系统的身份核查；系统的登录和身份核查；非登录用户不允许直接进入工作页面。

# 相关技术介绍

## 数据库

本系统采用MYSQL数据库对系统数据进行存储,并且使用的MySql-font可视化工具对数据库进行可视化的管理，方便进行测试和运行。

## JDBC



## Web技术

本系统采用B/S架构，使用了java web技术，servlet技术和css技术等技术。

## JSP

本系统的web网页开发采用的JSP技术，以java语言为脚本语言，响应客户端的请求，动态生成所需网页。

## Ajax

本系统在多级联动下拉框以及根据输入下拉框或表单内容显示相关内容使用了ajax技术。

# 需求分析

## 运动员

### 基本信息管理

* 完善个人信息：对个人基本信息的完善，因为一开始的默认登录时数据库里只有学号和密码的信息，所以完善信息包括姓名、性别、班级和简介的完善。同时如果运动员已经完善过了，就会在相应位置出现对应的个人信息。

### 报名

* 查询赛程安排：比赛类型有两种：田赛和径赛，运动员可以根据自己的喜好来查询对应比赛类型有哪些比赛赛程。但是运动员在没有完善过个人信息时，因为不知性别，所以查询结果包括男女比赛。在完善过个人信息后，只会出现相应性别的比赛类型。
* 填报项目：运动员在要填报项目时必须已经完善好了个人信息，并且每个运动员只能最多报名三个项目，且不能重复报名，报名确认后不能修改退出。报名成功后会有相应选手编号
* 查询报名情况：只有在运动员报名项目后才可查看，显示运动员已经报名的项目的比赛赛程和运动员相应的选手编号

### 查看比赛

* 查看成绩：在运动员报名项目后才能查看，未比赛时一开始报名选手的成绩排名均为0，且会显示为比赛，只有在检录后，才会显示已比赛，通过管理员录入成绩后查询比赛成绩，只能知道自己的比赛成绩和排名。

## 管理员

### 赛程赛事管理

* 赛事管理：根据往年的比赛项目表，添加新的赛事；
* 查看赛事：查询、修改和删除已添加的赛事
* 赛程管理：根据已添加的比赛项目，对其添加新的赛程，可以实现不同赛程（初赛，复赛，决赛等）分别添加；
* 查看赛程：查询、修改和删除已添加的赛程

### 各系管理

* 运动员管理：通过选择学院信息和班级信息挑选并显示该学院或者该班级的运动员名单，并且显示其参加的项目等详细信息；
* 查询赛程安排：通过选择学院信息和班级信息挑选并显示该学院或者该班级中有运动员参与的项目，并且支持查看该项目详细赛程以及参与人员的具体信息；
* 查询排名信息：根据记分规则（第一名5分，第二名4分......）以及成绩的有效性，后台计算出班级得分，并且统计学院得分，并且对学院进行排名。通过选择学院，可以得到该学院中的各个班级的排名信息。

### 运动会组委会管理

* 检录：通过输入比赛编号显示该比赛运动员信息，根据比赛成绩为空并且有效标志为0判定该项目的运动员是否检录，支持单击按钮显示对某一运动员进行检录。
* 成绩录入：通过输入比赛编号显示该比赛已经检录的运动员信息，允许输入该运动员的比赛成绩以及其排名信息，并且判定该成绩是否有效，然后进行数据持久化保存。
* 成绩查看：通过输入比赛编号显示该比赛运动员的有效成绩，并且支持显示该项目的无效比赛数据。

### 信息需求

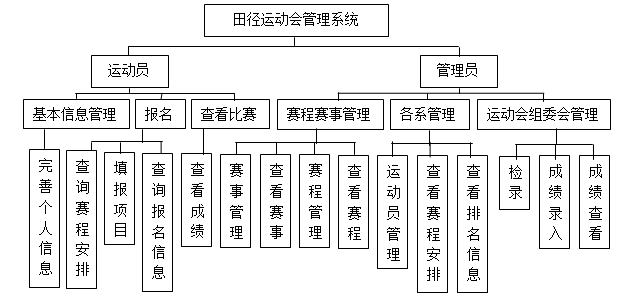
通过分析，我们可以得到本系统中主要管理的对象是项目、运动员、各学院、赛事、赛程、成绩，需存储的信息包括：

* 运动员：学号、运动员编号、姓名、性别、学院号、登录密码、简介
* 项目：项目类型、项目名；
* 学院：学院号、班级名称、学院名、学院得分；
* 管理员：账号、密码
* 赛事：赛事编号、赛事名、赛事参赛人数、赛事类型、赛事参赛者性别
* 成绩：运动员编号、赛事编号、成绩、排名、检录标识；
* 赛程：赛程编号、地址、开始时间、结束时间、赛事编号、赛程名；

# 概要设计

## 系统的功能模块框图

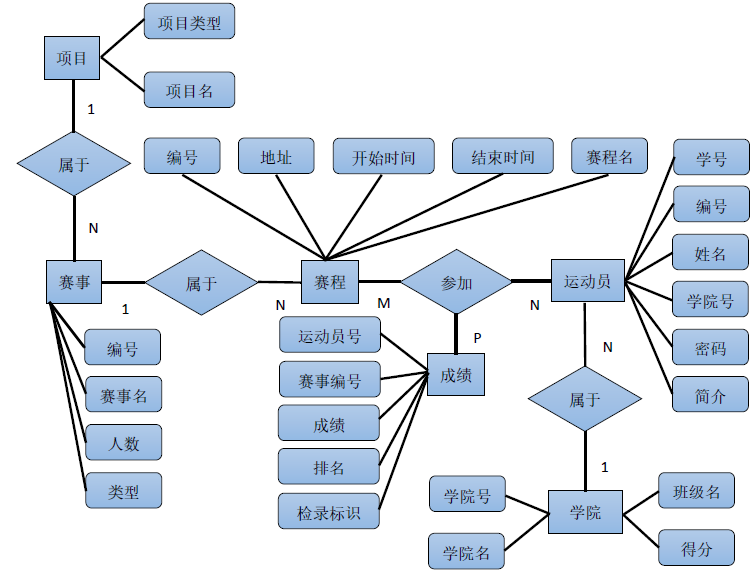
本系统的运动功能模块框图如图4-1。



**图4-1运动会功能模块框图**

## E-R图

本系统的实体-关系图（ER图）如图4-2。



**如图4-2运动会管理系统全局E-R图**

# 详细设计

## 运动员

### 功能模块详细设计

**用户个人信息完善**：

对数据库内容的提取，从数据库“学生”一张表中读取相应数据，为空则不显示否则在页面相应位置显示数据内容。并且运用AJXA技术显示联动的班级选项，选择相应专业弹出相应班级。最后运用表单的提交功能将修改后的个人信息提交到数据库中修改数据库中信息，确认提交前性别和班级不能为空，任何一个为空提示。提交成功提示“修改成功”并返回首页。具体文档安排及说明如表5-1。

**表5-1 完善个人信息相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | personal\_information.jsp | 修改完善个人信息主界面 |
| 2 | MyJsp.jsp | AJAX技术处理页面，异步刷新班级下拉框 |
| 3 | PerfectInformation.java | Servlet技术在表单提交时对表单内容进行读取和对比，最后修改数据库 |
| 4 | ClassList.java | javaBean技术获取班级列表 |
| 5 | StudentList.java | javaBean技术获取学生列表 |

**查询比赛赛程**：

点击下拉框异步刷新后续表格。显示比赛赛程的项目编号、项目名称、比赛地点、比赛开始时间和比赛结束时间。信息尚未完善时显示所有比赛赛程，信息完善后显示相应性别的比赛赛程。具体文档安排及说明如表5-2。

**表5-2 查询比赛赛程相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | compete\_schedule.jsp | 查询赛程安排主界面，下拉框选择比赛类型。 |
| 2 | scheduleTable.jsp | AJAX技术处理页面，异步刷新赛程安排表格 |

**填报项目**：

在未完善个人信息前，点击报名会提示“请先完善个人信息”并返回首页。在个人信息完善之后，主界面是一个表单，一共有三组下拉框，表示运动员最多可以报名三个项目运动。填报项目按田赛径赛区分。提交时项目不能全为空，不能有两项相同，否则提示相应错位。确认提交后，弹出提示框“是否确定报名，报名成功后不能修改”，点击“确定”确定报名，“取消”取消提交。报名成功后弹出“选手编号”并返回主界面。具体文档安排及说明如表5-3。

**表5-3 填报项目相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | signup.jsp | 运动员报名主界面. |
| 2 | MyJsp.jsp | AJAX技术处理页面，异步刷新比赛类型相应比赛 |
| 3 | SignUp.java | Servlet技术获取表单提交的内容，将其结果修改到数据库 |

**查询报名信息**：

在运动员未报名前点击会弹出提示框“请先报名”并返回首页，在运动员已经报名后，会显示相应报名信息，选手编号、比赛赛程的项目编号、项目名称、比赛地点、比赛开始时间和比赛结束时间。具体文档安排及说明如表5-4。

**表5-4 填报项目相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | searchSignup.jsp | 报名结果信息主界面 |

**查看成绩**：

在运动员未报名前点击会弹出提示框“请先报名”并返回首页，在运动员已经报名后，会显示相应报名信息及选手比赛成绩、比赛与否。具体文档安排及说明如表5-5。

**表5-5 填报项目相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | record.jsp | 比赛成绩主界面，显示报名项目的比赛成绩排名和比赛状况 |

## 管理员

### 赛事赛程管理

**赛事管理：**

根据往年的比赛项目表，添加新的赛事**。**选择对应项，运用ajax刷新“名称”，点击“提交”按钮提交表单信息，根据选择的性别，赛事类型和比赛的名称对应的编号生成赛事编号，以保证赛事编号的唯一性，此编号为赛事表的主键，调用servlet将表单信息插入到数据库的project表中；若是本项比赛已经添加过，则提示不能重复添加赛事，跳回本页面的未输入状态。具体文档安排及说明如表5-6。

**表5-6 赛事管理相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | manager\_saishi.jsp | 赛事管理的主界面，显示下拉框并提交数据 |
| 2 | manager\_project\_name.jsp | 异步刷新项目名称信息，以下拉框的方式显示 |
| 3 | servlet/ProjectServlet.java | 用于后台查询所需信息 |
| 4 | servlet/SaishiServlet.java | 用于提交表单数据到后台，将赛事信息插入到数据库 |

**查看赛事：**

①查询赛事：运用ajax局部刷新赛事表，以仔细查看赛事信息，在每一项赛事的右方分别添加两个功能按钮“修改”和“删除”。

②修改赛事：无论是一次ajax局部刷新还是两次ajax局部刷新后显示的赛事信息表，都可以选择右方的“修改”按钮，查询数据库后运用ajax技术再次刷新赛事表，只显示当前赛事的修改情况；

③删除赛事：无论是一次ajax局部刷新还是两次ajax局部刷新后显示的赛事信息表，都可以选择右方的“删除”按钮，查询数据库后运用ajax技术再次刷新赛事表，只显示当前赛事的情况。具体文档安排及说明如表5-7。

**表5-7 查看赛事相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | manager\_chakansaishi.jsp | 查看赛事安排的主界面，显示选择下拉框以及添加过的赛事信息 |
| 2 | select\_saishi.jsp | 异步刷新赛事信息表，显示该表格 |
| 3 | Select\_categoryyy.jsp | 用于异步刷新显示比赛类型“田赛/径赛”下拉框 |
| 4 | select\_saishix.jsp | 第二次异步刷新赛事信息表，显示该表格 |
| 5 | select\_updatex.jsp | 点击“修改”按钮后，异步刷新要修改的赛事信息 |
| 6 | select\_deletex.jsp | 点击“删除”按钮后，异步刷新要删除的赛事信息，点击返回 |
| 7 | servlet/ ProjectServlet.java | 用于后台查询所需信息并且删除赛事 |
| 8 | servlet/ChakansaishiServlet.java | 用于提交表单数据（修改后的数据）到后台，将数据库的赛事信息更新 |

**赛程管理：**

根据已添加的比赛项目，对其添加新的赛程，可以实现不同进程（初赛，复赛，决赛等）的添加。选择对应项，多次运用ajax局部刷新各个下拉框，点击“提交”按钮提交表单信息，调用servlet将表单信息插入到数据库的schedule表中；若是本项赛程已经添加过，则提示不能对同一赛程的同一进程重复添加，跳回本页面的未输入状态。具体文档安排及说明如表5-8。

**表5-8 赛程管理相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | manager\_saicheng.jsp | 赛事管理的主界面，显示下拉框并提交数据 |
| 2 | select\_name.jsp | 异步刷新比赛名称下拉框，显示查询到的比赛名称 |
| 3 | select\_category.jsp | 用于异步刷新显示查询到的比赛类型“田赛/径赛”下拉框 |
| 4 | select\_name.jsp | 第二次异步刷新比赛名称下拉框，显示查询到的比赛名称 |
| 5 | manager\_project\_name.jsp | 异步刷新项目名称信息，以下拉框的方式显示 |
| 6 | servlet/ProjectServlet.java | 用于后台查询所需信息 |
| 7 | servlet/SaichengServlet.java | 用于提交表单数据到后台，将赛程信息插入到数据库 |

**查看赛程：**

①查询赛程：运用ajax局部刷新赛程表，以仔细查看赛程信息，在每一项赛程的右方分别添加两个功能按钮“修改”和“删除”。

②修改赛程：无论是一次ajax局部刷新还是两次ajax局部刷新后显示的赛程信息表，都可以选择右方的“修改”按钮，查询数据库后运用ajax技术再次刷新赛程表，只显示当前赛程的修改情况；

③删除赛程：无论是一次ajax局部刷新还是两次ajax局部刷新后显示的赛程信息表，都可以选择右方的“删除”按钮，查询数据库后运用ajax技术再次刷新赛程表，只显示当前赛程的情况。具体文档安排及说明如表5-9。

**表5-9 查看赛程相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | manager\_chakansaicheng.jsp | 查看赛程安排的主界面，显示选择下拉框以及添加过的赛程信息，提交修改了的赛程 |
| 2 | select\_schedule.jsp | 异步刷新赛程信息表，显示该表格 |
| 3 | select\_categoryy.jsp | 用于异步刷新显示比赛类型“田赛/径赛”下拉框 |
| 4 | select\_ schedulex.jsp | 选择了比赛类型后，第二次异步刷新赛程信息表，显示该表格 |
| 5 | select\_update.jsp | 点击“修改”按钮后，异步刷新要修改的赛程信息 |
| 6 | select\_delete.jsp | 点击“删除”按钮后，异步刷新要删除的赛程信息，点击返回 |
| 7 | servlet/ ProjectServlet.java | 用于后台查询所需信息并且删除赛程 |
| 8 | servlet/ChakansaichengServlet.java | 用于提交表单数据（修改后的数据）到后台，将数据库的赛程信息更新 |

### 各系管理

**运动员管理：**

设置一个二级联动下拉框选择学院及其班级信息，异步刷新运动员基本信息，通过按钮显示运动员详细信息以及填报项目信息，具体文档安排及说明如表5-10。

**表5-10 运动员管理相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | competerInClass\_manager.jsp | 运动员管理的主界面，显示选择下拉框以及运动员信息 |
| 2 | competerTable\_manager.jsp | 异步刷新运动员信息，显示该表格 |
| 3 | departmentselect.jsp | 用于异步刷新显示班级下拉框 |
| 4 | competer\_project.jsp | 用于显示根据competer\_id查找该运动员的基本信息以及其参与的项目 |
| 5 | juan\_servlet/ClassServlet.java | 用于后台查询所需信息 |
| 6 | juan\_servlet/StudentServlet.java | 用于后台查询运动员基本信息 |

**查看赛程安排：**

设置一个二级联动下拉框选择学院及其班级信息，异步刷新该学院有人参与的项目信息以及后台计算出参与每个项目的人数，通过按钮显示项目赛程详细信息以及选择该项目的运动员名单，具体文档安排及说明如表5-11。

**表5-11 查看赛程安排相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | scheduleInClass\_manager.jsp | 查看赛程安排的主界面，显示选择下拉框以及赛程信息 |
| 2 | scheduleTable\_manager.jsp | 异步刷新赛程信息，显示该表格 |
| 3 | departmentselect.jsp | 用于异步刷新显示班级下拉框 |
| 4 | project\_schedule.jsp | 用于显示根据project\_id查找该项目的基本信息以及其赛程信息 |
| 5 | project\_competerlist.jsp | 用于显示根据project\_id和class\_name查询该项目中该学院参与的人员信息 |
| 6 | juan\_servlet/ClassServlet.java | 用于后台查询所需的系别信息 |
| 7 | juan\_servlet/StudentServlet.java | 用于后台查询所需信息 |
| 8 | juan\_servlet/Project\_Schedule.java | 用于后台查询项目信息 |
| 9 | juan\_servlet/Project\_Schedule.java | 用于后台查询赛程信息 |

**查看排名信息：**

根据积分规则计算每个班级的总得分情况，然后统计每个学院的总分以及班级运动员人数并对其进行排名显示，通过一个下拉框选择学院信息，异步刷新表格显示学院中的班级排名信息，具体文档安排及说明如表5-12。

**表5-12 查看排名相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | rankingInClass\_manager.jsp | 查看排名信息的主界面，显示选择下拉框以及排名信息 |
| 2 | rankingTable\_manager.jsp | 异步刷新排名信息，显示该表格 |
| 3 | juan\_servlet/RecordsServlet.java | 用于计算班级得分，并且进行后台查询所需信息 |

### 运动会组委会管理

**检录：**

设置两个输入框分别输入项目编号和运动员编号，通过读取这两个输入框的内容查找其相应的数据，并且异步显示，支持用按钮对其检录状态进行修改，具体文档安排及说明如下表5-13：

**表5-13 检录相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | Checkin.jsp | 检录管理的主界面，显示输入框及查询按钮，并且显示查询到的信息 |
| 2 | CheckinTable.jsp | 异步刷新检录信息，显示该表格 |
| 3 | CheckinOperation.jsp | 用于检录信息的持久化保存，不在可视化界面显示信息 |
| 4 | juan\_servlet/RecordsServlet.java | 用于获取成绩信息及其检录情况，持久化保存检录信息 |

**成绩录入：**

设置两个输入框分别输入项目编号和运动员编号，通过读取这两个输入框的内容查找其相应的数据并显示，在每条记录后面设置两个输入框，用其运动员编号动态生成其id，通过表单提交到后台进行处理，具体文档安排及说明如表5-13。

**表5-14 成绩录入相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | recordsInput.jsp | 成绩录入的主界面，显示输入框及查询按钮，并且显示查询到的信息 |
| 2 | recordsInput Table.jsp | 异步刷新成绩信息，显示该表格 |
| 3 | juan\_servlet/RecordsServlet.java | 用于后台持久化保存成绩信息 |

**成绩查看**

设置两个输入框分别输入项目编号和运动员编号，通过读取这两个输入框的内容查找其相应的数据，并且异步显示，支持按钮查看该项目的无效数据，具体文档安排及说明如表5-15。

**表5-15 成绩查看相关文档说明**

|  |  |  |
| --- | --- | --- |
| 文件编号 | 文件名 | 文件说明 |
| 1 | recordsForProject.jsp | 成绩查看的主界面，显示输入框及查询按钮，并且显示查询到的信息 |
| 2 | recordsTable.jsp | 异步刷新成绩信息，显示该表格 |
| 3 | defaultRecordsTable.jsp | 无效数据表格 |
| 4 | juan\_servlet/RecordsServlet.java | 用于后台查询成绩信息 |

## 逻辑表结构的设计

* 运动员（学生）信息（学号、运动员编号、姓名、性别、学院编号、学生密码、简介）
* 管理员（管理员账号，管理员密码）
* 学院（学院编号、班级、学院名称、学院得分）
* 项目（项目编号，项目类型、项目名）
* 赛事（赛事编号、赛事名、赛事参赛人数、赛事类型、赛事参赛者性别）
* 赛程（赛程编号、地址、开始时间、结束时间、赛事编号、赛程名）
* 成绩（运动员编号、赛事编号、成绩、排名、检录标识）

# 系统实现

## 开发环境搭建

本系统数据库采用MySQL数据库和myEclipse开发系统，开发软件前电脑应配备好MySQL和myEclipse软件。

## 数据库的实现

### 关系表

将前面得到的关系模式转换为MySQL5.5支持的具体关系表如下：

1、student运动员信息表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段宽度** | **是否允许空** | **说明** |
| **Student\_number** | **INT** | **10** | **NOT NULL** | **学号** |
| **Competer\_id** | **INT** | **10** | **NOT NULL** | **主键，参赛者编号** |
| **Student\_name** | **CHAR** | **16** | **NULL** | **姓名** |
| **Student\_sex** | **CHAR** | **4** | **NULL** | **性别** |
| **Class\_id** | **INT** | **10** | **NULL** | **外键，学院编号** |
| **Student\_password** | **CHAR** | **20** | **NULL** | **学生密码** |
| **Student\_introduction** | **VARCHAR** | **1024** | **NULL** | **简介** |

2、manager管理员表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段宽度** | **是否允许空** | **说明** |
| **Manager\_id** | **INT** | **10** | **NOT NULL** | **主键，管理员账号** |
| **Manager\_password** | **CHAR** | **20** | **NULL** | **管理员密码** |

3、class学院信息表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段宽度** | **是否允许空** | **说明** |
| **Class\_id** | **INT** | **10** | **NOT NULL** | **学院编号,主键，外键** |
| **Class\_depertment** | **VARCHAR** | **255** | **NOT NULL** | **班级** |
| **Class\_name** | **VARCHAR** | **50** | **NULL** | **学院名称** |
| **Class\_score** | **INT** | **10** | **NULL** | **学院得分** |

4、category\_name项目信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段宽度** | **是否允许空** | **说明** |
| **Id** | **INT** | **11** | **NOT NULL** | **主键，项目编号** |
| **category** | **CHAR** | **20** | **NULL** | **项目类型** |
| **name** | **CHAR** | **20** | **NULL** | **项目名** |

5、schedule赛程表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段宽度** | **是否允许空** | **说明** |
| Schedule**\_id** | **INT** | **4** | **NOT NULL** | **赛程编号，主键** |
| Schedule**\_address** | **CHAR** | **24** | **NULL** | **地点** |
| Schedule**\_starttime** | **DATETIME** |  | **NULL** | **开始时间** |
| Schedule**\_endtime** | **DATETIME** |  | **NULL** | **结束时间** |
| **Project\_id** | **INT** | **10** | **NOT NULL** | **项目编号，外键** |
| Schedule**\_name** | **VARCHAR** | **50** | **NULL** | **赛程名** |

6、project比赛项目表

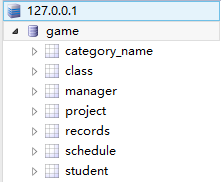
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段宽度** | **是否允许空** | **说明** |
| **Project\_id** | **INT** | **10** | **NOT NULL** | **比赛项目编号，主键** |
| **Project\_name** | **CHAR** | **20** | **NULL** | **项目名称** |
| **Project\_number** | **INT** | **8** | **NULL** | **人数** |
| **Project\_category** | **CHAR** | **16** | **NULL** | **类型** |
| **Project\_sex** | **VARCHAR** | **4** | **NULL** | **项目性别** |

7、recordes成绩表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **字段类型** | **字段宽度** | **是否允许空** | **说明** |
| **Competer\_id** | **INT** | **10** | **NOT NULL** | **参赛者编号，联合主键** |
| **Project\_id** | **INT** | **10** | **NOT NULL** | **项目编号，联合主键** |
| **Grades\_grades** | **CHAR** | **20** | **NULL** | **成绩** |
| **Grades\_ranking** | **INT** | **8** | **NULL** | **名次** |
| **Grades\_flag** | **INT** | **4** | **NULL** | **检录标识** |

### 数据库实施代码

**1、创建数据库“game”：**



**2、创建关系表**

1)创建学生信息表：

DROP TABLE IF EXISTS `student`;

CREATE TABLE `student` (

`Student\_number` int(10) NOT NULL DEFAULT '0',

`Competer\_id` int(10) NOT NULL DEFAULT '0',

`Student\_name` char(16) DEFAULT NULL,

`Student\_sex` char(4) DEFAULT NULL,

`Class\_id` int(10) DEFAULT NULL,

`Student\_password` char(20) DEFAULT NULL,

`Student\_introduction` varchar(1024) DEFAULT NULL,

PRIMARY KEY (`Student\_number`),

KEY `fk\_class\_id` (`Class\_id`),

KEY `Competer\_id` (`Competer\_id`),

CONSTRAINT `fk\_class\_id` FOREIGN KEY (`Class\_id`) REFERENCES `class` (`Class\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

2)创建管理员信息表：

DROP TABLE IF EXISTS `manager`;

CREATE TABLE `manager` (

`Manager\_id` int(10) NOT NULL,

`Manager\_password` char(20) DEFAULT NULL,

PRIMARY KEY (`Manager\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

3)创建系信息表：

DROP TABLE IF EXISTS `class`;

CREATE TABLE `class` (

`Class\_id` int(10) NOT NULL DEFAULT '0',

`Class\_department` varchar(255) NOT NULL DEFAULT 'null',

`Class\_name` varchar(50) DEFAULT NULL,

`Class\_score` int(10) DEFAULT '0',

PRIMARY KEY (`Class\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

4)创建项目表：

DROP TABLE IF EXISTS `category\_name`;

CREATE TABLE `category\_name` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`category` char(20) DEFAULT NULL,

`name` char(20) DEFAULT NULL,

PRIMARY KEY (`Id`)

) ENGINE=InnoDB AUTO\_INCREMENT=25 DEFAULT CHARSET=utf8;

5)创建赛事信息表：

DROP TABLE IF EXISTS `project`;

CREATE TABLE `project` (

`Project\_id` int(10) NOT NULL,

`Project\_name` char(20) DEFAULT NULL,

`Project\_number` int(8) DEFAULT NULL,

`Project\_category` char(16) DEFAULT NULL,

`Project\_sex` varchar(4) DEFAULT NULL,

PRIMARY KEY (`Project\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

6)创建赛程信息表：

DROP TABLE IF EXISTS `schedule`;

CREATE TABLE `schedule` (

`Schedule\_id` int(10) NOT NULL AUTO\_INCREMENT,

`Schedule\_address` char(24) DEFAULT NULL,

`Schedule\_starttime` datetime DEFAULT NULL,

`Schedule\_endtime` datetime DEFAULT NULL,

`Project\_id` int(10) NOT NULL,

`Schedule\_name` varchar(50) DEFAULT NULL,

PRIMARY KEY (`Schedule\_id`),

KEY `Project\_id` (`Project\_id`),

CONSTRAINT `schedule\_ibfk\_1` FOREIGN KEY (`Project\_id`) REFERENCES `project` (`Project\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=7 DEFAULT CHARSET=utf8;

7)创建成绩信息表：

DROP TABLE IF EXISTS `records`;

CREATE TABLE `records` (

`Competer\_id` int(10) NOT NULL,

`Project\_id` int(10) NOT NULL,

`Grades\_grades` char(20) DEFAULT NULL,

`Grades\_ranking` int(8) DEFAULT '0',

`Grades\_flag` int(1) DEFAULT '0',

PRIMARY KEY (`Competer\_id`,`Project\_id`),

KEY `Project\_id` (`Project\_id`),

KEY `Competer\_id` (`Competer\_id`),

CONSTRAINT `records\_ibfk\_1` FOREIGN KEY (`Competer\_id`) REFERENCES `student` (`Competer\_id`),

CONSTRAINT `records\_ibfk\_2` FOREIGN KEY (`Project\_id`) REFERENCES `project` (`Project\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

**3、数据初始化**

1)将项目信息输入到项目表里：

INSERT INTO `category\_name`

VALUES (1,'田赛','跳高'),(2,'田赛','撑杆跳高'),(3,'田赛','跳远'),(4,'田赛','三级跳远'),

(5,'田赛','铅球'),(6,'田赛','铁饼'),(7,'田赛','链球'),(8,'田赛','标枪'),

(9,'田赛','十项全能'),(10,'径赛','100米'),(11,'径赛','200米'),(12,'径赛','400米'),

(13,'径赛','800米'),(14,'径赛','1500米'),(15,'径赛','5000米'),(16,'径赛','10000米'),

(17,'径赛','马拉松'),(18,'径赛','3000米障碍跑'),(19,'径赛','110米跨栏'),

(20,'径赛','400米跨栏'),(21,'径赛','20公里竞走'),(22,'径赛','50公里竞走'),

(23,'径赛','4X100接力跑'),(24,'径赛','4X400接力跑');

2)将管理员信息输入到管理员信息表里：

INSERT INTO `manager` VALUES (1,'1');

3)将学生信息输入到学生信息表里：

INSERT INTO `student`

VALUES

(121320107, ,' ','', ,'121320107',''),

(121320205, ,'','', ,'121320205',''),( 121320207, ,'','f, ,'121320207','');

## 数据库的连接

本系统采用的数据库是mysql数据库，利用数据库连接语句将系统与数据库相连以进行操作。

JDBC(Java Database Connectivity）即数据库连接，它提供了用于执行SQL语句标准的Java API。

JDBC为Java应用程序提供了一系列由Java语言编写的类与接口，使其能够快速高效地访问数据库；可以方便实现多种关系数据库的统一操作；

JDBC程序访问数据库步骤:

1.使用Class类的forName方法，将驱动程序类加载到JVM（Java虚拟机）中.

2.成功加载驱动后，进一步获得连接对象；

3.先使用java.sql包中的Connection类声明一个对象

4.再使用类DriverManager调用它的静态方法getConnection创建这个连接对象

5.建立连接时应捕获SQLException异常 ：

具体流程如图7-1所示

导入java.sql包

开 始

附加驱动

建立数据源

使用ResultSet对象

执行SQL语句

创建Statement对象

创建Connect对象

加载并注册相应驱动程序

结 束

关闭Connection对象

图7-1 JDBC程序访问数据库步骤

具体实现语句：

**private** **static** Connection *conn*;//连接对象，用于与数据库取得连接

PreparedStatement pstmt;// 预编译语句对象，用于执行预编译的SQL语句，执行效率比Statement高

**public** **static** Connection getConn()

{

**try** {

String url = "jdbc:mysql://localhost:3306/game";

Class.*forName*("com.mysql.jdbc.Driver");

*conn* = DriverManager.*getConnection*(url, "root", "1123581321");//驱动程序管理类，用于加载和卸载各种驱动程序，并建立与数据库的连接

**return** *conn*;

} **catch** (Exception e) {

System.*err*.println("数据库连接异常:" + e.getMessage());

**return** **null**;

}

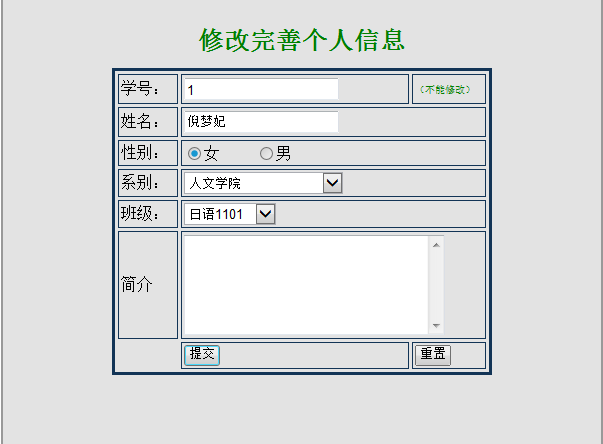
}

## 主要功能模块的实现

### 运动员

**修改完善个人信息**

方法介绍：采用的是ajax技术实现二级菜单的级联，联动生成系别的班级下拉列表，通过提交表单，将运动员信息添加到数据库中。（如图6-1）



**图6-1**

**查询赛程安排（如图6-2、6-3）**



**图6-2**

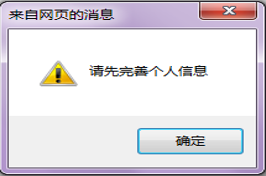


**图6-3**

**填报项目**

方法介绍：采用的是ajax技术实现多个二级菜单的级联，联动生成比赛类型下拉列表，通过提交表单，将报名信息添加到数据库中，生成运动员编号。（如图6-4,6-5）

个人信息未完善（如图6-4）：



**图6-4**

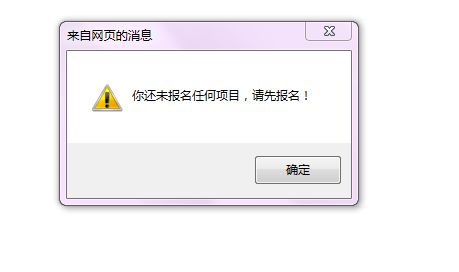
个人信息已完善（如图6-5）：



**图6-5**

**查询报名情况**

未报名时（如图6-6）：



**图6-6**

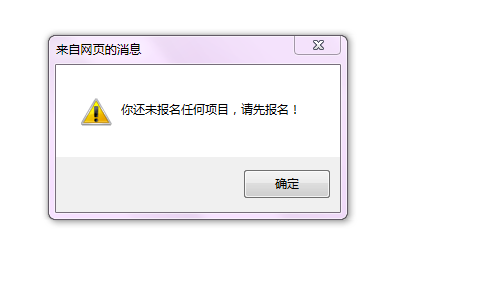
已报名（如图6-7）：



**图6-7**

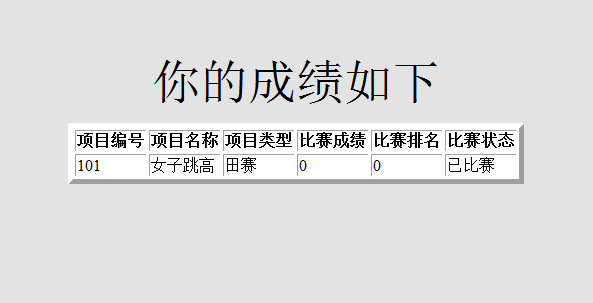
**查看成绩**

未报名时（如图6-8）：



**图6-8**

已报名：

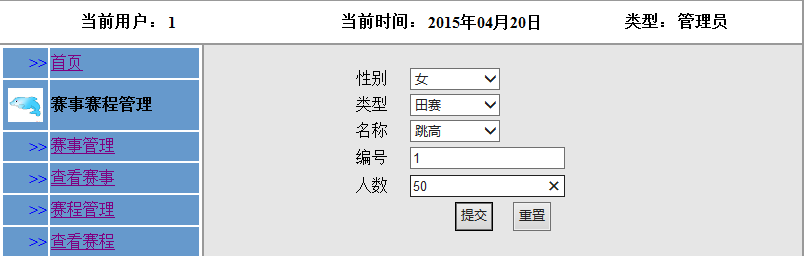


**图6-9**

### 管理员

**赛事管理**

方法介绍：多次采用的是ajax技术实现二级菜单的级联，以及信息表格的变化，通过按钮绑定onClick事件，并且通过两个jsp界面传参数，在第二个页面读取后，读取数据，查询出名称的详细信息的jsp页面，通过提交表单，将新赛事添加到数据库中。（如图6-10）

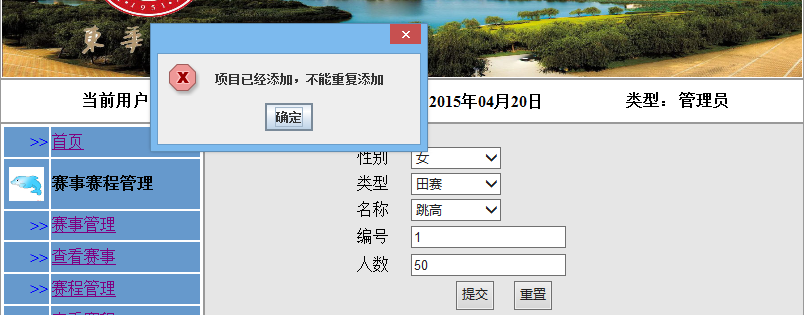


**图6-10**

不可以重复添加同一比赛，否则显示错误提示（如图6-11、图6-12）：



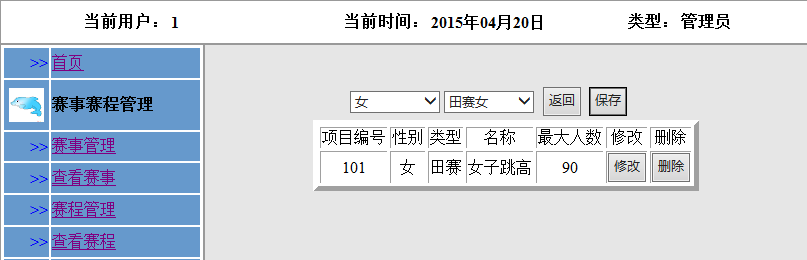
**图6-11**



**图6-12**

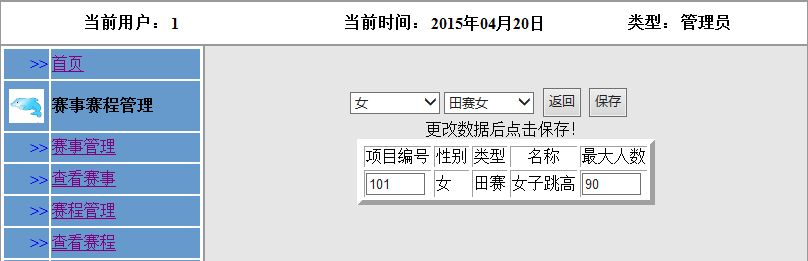
**查看赛事**

方法介绍：多次采用的是ajax技术实现二级菜单的级联，以及信息表格的变化（同上），通过按钮绑定onClick事件，并且通过两个jsp界面传参数，在第二个页面读取后，读取数据，查询出比赛类型和赛事的详细信息的jsp页面，可以通过点击按钮“修改”和“删除”赛事信息。（如图6-13）



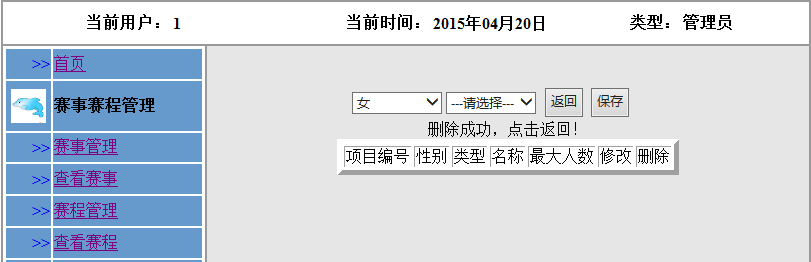
**图6-13**

修改赛事（如图6-14）：



**图6-14**

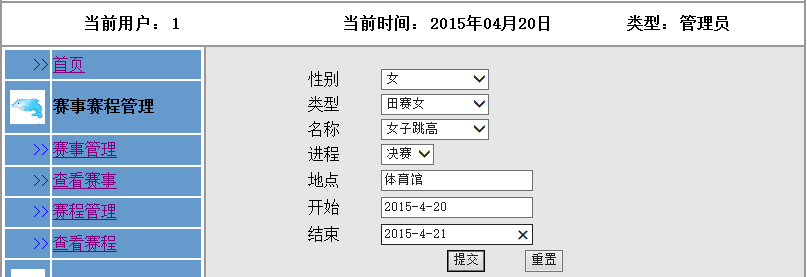
删除赛事（如图6-15）：



**图6-15**

**赛程管理**

方法介绍：同上（如图6-16））



**图6-16**

一项比赛可以有多个进程（初赛，复赛，决赛），但是不能对同一比赛的同一进程重复添加赛程，否则显示错误提示（如图6-17、6-18）：



**图6-17**



**图6-18**

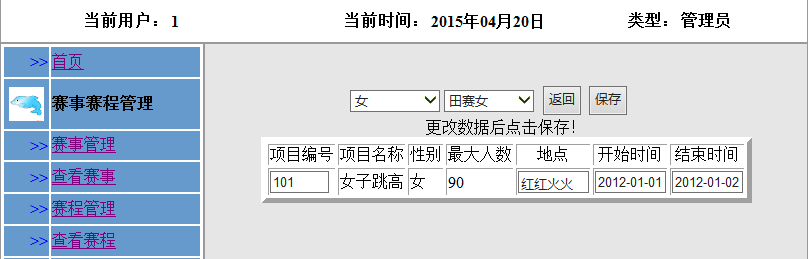
**查看赛程**

方法介绍：同上（如图6-19）



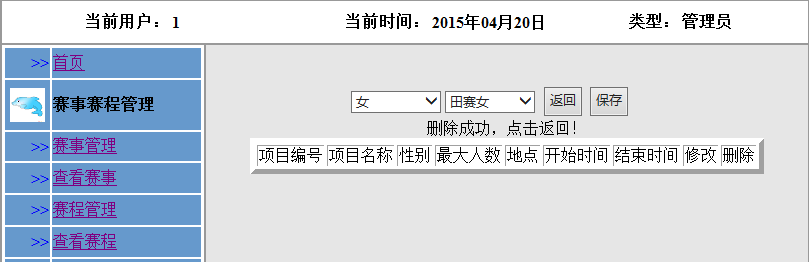
**图6-19**

修改赛程（如图6-20）：



**图6-20**

删除赛程（如图6-21）：



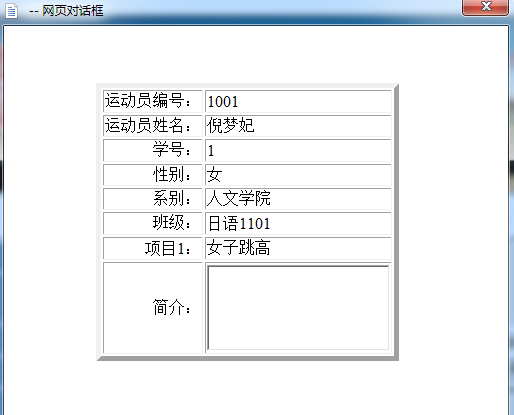
**图6-21**

**运动员管理**

方法介绍：多次采用的是ajax技术实现二级菜单的级联，以及信息表格的变化（如图），并且通过按钮绑定onClick事件，并且通过两个jsp界面传参数，在第二个页面读取后，读取数据，利用窗口显示运动员详细信息的jsp页面（如图6-22、图6-23）。



**图6-22**



**图6-23**

**查看赛程安排**

方法介绍：多次采用的是ajax技术实现二级菜单的级联，以及信息表格的变化（同上），并且通过按钮绑定onClick事件，并且通过两个jsp界面传参数，在第二个页面读取后，读取数据，利用窗口显示赛程详细信息的jsp页面以及该学院或班级填报该项目的名单（如图6-24）。



**图6-24**

**查看排名信息**

方法介绍：首先通过对后台有效数据进行处理，计算出每个班级的得分情况，然后根据班级的隶属情况算出每个学院的总分进行排名，初始状态显示学院排名信息（如图6-25）。选择学院的下拉框用ajax技术刷新表格，显示该学院中各个班级的排名情况（如图6-26）。通过绑定按钮的onClick事件回到初始状态，及学院总分排名。

**计算分数代码：**

**public** **void** calculateTheScore() **throws** SQLException{

DbConn dc = **new** DbConn();

dc.update("update class set class\_score = 0");

ResultSet rs = dc.select("select competer\_id, Grades\_ranking from records where Grades\_flag=1");//有效成绩

**while**(rs.next()){

**int** cid = rs.getInt(1);

**int** score = 6-rs.getInt(2);

**if**(rs.getInt(2)==0) **continue**;

**int** class\_id = getClassidByCid(cid);

**int** classscore = getScoreByClassid(class\_id)+score;

dc.update("update class set class\_score="+classscore+" where class\_id = "+class\_id);

}}

**int** getClassidByCid(**int** cid){

**int** classid = 0;

DbConn dc = **new** DbConn();

ResultSet rs = dc.select("select class\_id from student where competer\_id="+cid);

**try** {

rs.next();

classid = rs.getInt(1);

} **catch** (SQLException e) {

e.printStackTrace();} **return** classid;}

**int** getScoreByClassid(**int** classid){

**int** score = 0;

DbConn dc = **new** DbConn();

ResultSet rs = dc.select("select class\_score from class where class\_id="+classid);

**try** {

rs.next();

score = rs.getInt(1);

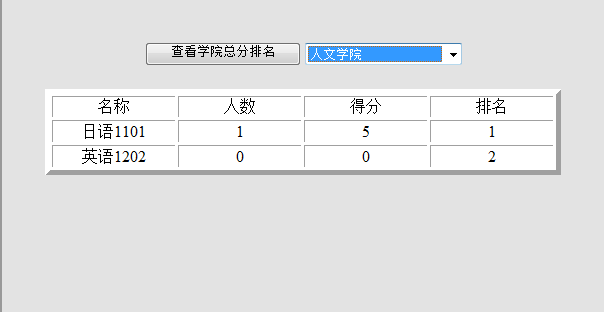
} **catch** (SQLException e) {

e.printStackTrace();} **return** score;

}



**图6-25**



**图6-26**

**检录**

方法介绍：通过在表单的输入，获取项目编号以及运动员编号，通过ajax技术刷新表格信息，利用按钮的onClick事件实现检录与取消检录并且为每个按钮生成动态ID，实现按钮的动态读取（如图6-27）



**图6-27**

**成绩录入**

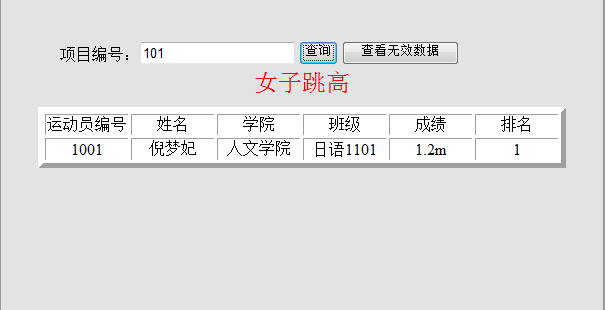
方法介绍：通过表单输入获取项目编号和运动员编号，传到后台获取该项目的有效比赛记录，通过ajax技术刷新表格。通过多选框选项判别该比赛数据是否有效。为每个输入框以及多选选框动态生成ID，便于后台读取。通过按钮调用后台doPost方法，实现数据的持久化保存（如图6-28）。



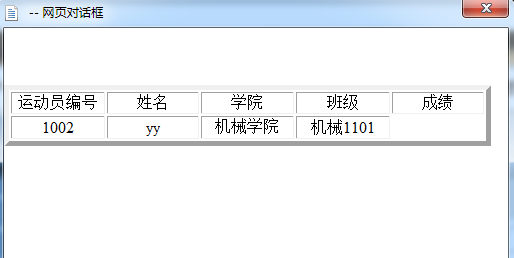
**图6-28**

**成绩查看**

方法介绍：输入项目编号，通过表单的方式递交到后台，查询该项目的有效比赛记录（如图6-29）。通过给按钮绑定onClick事件，并且将参数传到另一个jsp界面，显示该项目的无效比赛数据（如图6-30）。



**图6-29**



**图6-30**

# 总结

通过这几周本小组在一起设计的合作和交流。我们大家在一起查阅了许多的资料，在这个过程中接触了很多新的东西，获取了很多新的信息。由于要独立地进行自己分工系统的设计，需要自己去分析架构和设计，提高了自己分析问题和解决问题的能力。同样在此过程中，我们初步的熟悉了田径运动会管理系统的基本流程和运动会举行过程中各个阶段的运作情况。在动手实践的过程中，我们遇到问题或是自己想办法解决或是大家一起讨论。培养了自己的团队协作精神和个人动手能力。并且更加深刻的认识了数据库管理系统的本质和内涵。

实践是对理论学习的一个很好的巩固，并且加深了对理论知识的认识，学到了平时很多书上所不能得到的实践经验，由于实践中碰到许多以前从未碰到过的问题，需要很多新的知识，并且促进对理论知识的学习。通过这次实践课，让我们学会了很多平时学不到的东西，对我们影响颇大！

# 参考文献

1. 《ASP.NET Web 应用开发技术》 喻钧 白小军 清华大学出版社
2. 《数据库系统教程》 施伯乐 丁宝康 汪卫 高等教育出版社
3. 《软件工程导论》 张海藩 清华大学出版社
4. 《轻量级Java EE企业应用实战》 李刚 电子工业出版社出版社
5. 《数据结构》严蔚敏 吴伟明 清华大学出版社