**景德镇陶瓷大学**

**题目：**毕业设计管理系统

**学 院：** 信息工程学院

**班 级：**  计算机二班

**专 业：**  计算机科学与技术

**指导教师：** 林卫中 **职称** 副教授

**小组成员：**熊濡、梁敦峰、胡阳、陈政、徐诚

目录

目录 I

第一章 绪论 1

1.1 开发背景与现状 1

第二章 开发环境介绍 3

2.1 开发工具IDEA 3

2.2 数据库MySQL 3

2.2.1 MySQL简述 3

2.2.2 MySQL的优点和不足 3

2.2.3 MySQL存储引擎 3

第三章 系统需求分析 4

3.1 功能需求 4

3.1.1 基本功能需求 4

3.1.2 功能模块 5

3.2 性能需求 5

第四章 系统设计 6

4.1 数据库设计 6

4.1.1 物理设计 6

4.1.2 E-R模型设计 7

4.1.3 表/字段及表间关系设计 12

4.2 模块设计 26

4.2.1 信息管理模块 26

4.2.2 课题管理模块 27

4.2.3 毕业设计管理模块 27

4.2.4 流程管理模块 29

4.2.5 消息通知模块 30

4.2.6 系统管理员模块 31

第五章 系统的编码与实现 32

5.1 系统的实现平台 32

5.1.1 系统环境 32

5.1.2 后台数据库选择及配置 32

5.2 子系统功能模块的实现 32

5.2.1 信息管理模块 32

5.2.2 课题管理模块 36

5.2.3 毕业设计管理模块 40

5.2.4 流程管理模块 47

5.2.5 消息通知模块 50

5.2.6 系统管理员模块 53

第一章 绪论

1.1 开发背景与现状

我国互联网技术和信息网络技术发展迅速，各行各业都在进行互联网信息化管理，教育行业也不例外，互联网为高校管理信息化提供了必备的物质基础[1]。随着教育改革的深入发展以及办学规模的不断扩大，各高校工作量倍增，特别是毕业设计过程管理工作更是成指数递增。面对大量并且繁琐的管理工作，我们原来的管理方式已经成为学校信息化必须改进的问题。因此，利用信息技术和网络技术实现管理信息化，对工作进行高效的管理，成为各大高校的建设目标[2]。

毕业设计是大学生本科生涯的最后一班岗，也是最重要的任务，能让学生在享受创新的过程中也付诸实践行动，也是对学生步入社会的一次综合训练[3]。在毕业设计的教学管理中，涉及到许多环节，如指导教师的选择、课题的制定、课题的考核、任务书、开题报告、过程检验、毕业设计分数、毕业答辩等。涉及大量人员，如专业教师、所有毕业生、专业领导等。因此，毕业设计的教学管理是一项系统而复杂的工作[4]。

在传统的毕业设计管理模式中，存在着管理不规范、工作复杂、效率低下等突出问题[5]。例如：毕业设计课题的选择、过程管理的监控、文档和数据的保留、最终结果的统计分析等。没有得到有效规范的管理，导致学校的教学管理落后于现代信息化的管理要求。针对毕业设计管理中的问题，旨在提高学生的毕业设计质量，设计开发了一套适合本学院的本科毕业设计过程管理系统，实现了本学院毕业设计管理的高效、信息化、标准化和便捷化，充分减轻了师生的负担。

1.2 开发设计的意义

毕业设计管理系统的开发不仅可以提高工作效率，减轻日常管理人员的负担，而且为学生和教师提供了更大的便利[6]。目前，信息部和计算机工程学院仍然采用传统的手工方式进行毕业设计工作，这需要大量的人力和物力。在开展工作的过程中，也体现了师生之间的交流，教师对学生的指导也不方便。根据学校的毕业设计管理制度，结合信息化和计算机工程学院学生毕业设计管理的特点，对毕业设计管理进行了分析和设计。该系统从项目申请、项目评审、学生选题、相关资料提交、论文评价、毕业答辩等方面管理毕业设计的全过程。毕业设计管理系统的开发不仅方便了师生之间的交流，方便了教师对学生的指导，而且使毕业设计过程更加规范，进一步提高了教师的工作效率，对提高人才培养质量起到了积极的作用[7]。因此，开发毕业设计管理系统具有很强的现实意义。

第二章 开发环境介绍

2.1 开发工具IDEA

IDEA，是目前比较流行的java开发工具[8]。也是被开发人员公认的最好用的开发工具之一，特别是它自身携带的一些插件，如智能代码助手，代码自动提示，各种版本工具等等。IntelliJ IDEA是一个相对较新的Java IDE。它是Java开发环境中最有用的。高度优化的智能算法使普通任务变得非常容易。该软件以其智能的实时分析和便捷的重构功能深受大家的喜爱。缺点是它很复杂，初学者很难理解[9]。

2.2 数据库MySQL

2.2.1 MySQL简述

MySQL是目前在互联网行业公认的最好的关系型数据库管理系统，在选择上，MySQL相对于其他的数据库而言是开源的，对于大多数创业型公司而言都是一个极好的选择，在使用上也较为容易的建立相应的网站系统[10]。

2.2.2 MySQL的优点和不足

1．MySQL具有天然的移植性，能够在任何平台上得到一致的使用。

2．是一个支持超大数据规模的数据库系统

3．是开放源代码的，相对版本更新较快

4．优化的SQL查询算法，有效地提高查询速度，性能很出色。

5．对于使用环境而言，MySQL可以不受限制的在各种应用程序中使用，是采用低耦合的方式进行架构的。

2.2.3 MySQL存储引擎

MySQL和其他数据库系统很明显的不同的地方在于，MySQL支持可插拔式的引擎机制 [11]。

MySQL的存储引擎主要有下面几个：

MyISAM。是MySQL以前版本中的默认数据库引擎，这个引擎是一个不支持事务的引擎，对于目前往往针对速度快的系统进行使用。

InnoDB。主要用于需要强事务的业务条件，能支持行级锁定，支持各种事务[12]

BDB ,Memory ,Merge ,Archive

第三章 系统需求分析

3.1 功能需求

目前大多数学校开展毕业设计的大部分工作都是采用人工方式，对于管理人员而言，这是一个极具压力的工作，也不利于对毕业设计过程的监控和指导[13]。

现如今，计算机技术得到高速的发展，信息化手段也变得多彩多样，传统的手工工作方式已经远远不能适应现如今高速的社会，无法很好的完成毕业设计过程的管理工作，提高毕业设计过程管理水平的主要途径是运用先进的信息技术，开发出高效综合的毕业设计过程管理平台，并从毕业设计工作的实际出发，解决工作中的关键任务结点问题，充分利用先进的计算机信息化技术，实现毕业设计过程管理工作的自动化，减少教师和学生的工作量,提高工作效率,摆脱传统手工操作模式带来的低效与误差[14]。

3.1.1 基本功能需求

本系统共涉及系统管理员、全体毕业生、指导教师、评阅教师和教研室主任、学院负责人六种角色。通过这六种角色操作的相互配合，共同完成了从创建课题、审核课题、双向选题、任务书、开题报告、过程检查、毕业论文等功能。在设计该系统时，应尽可能的以本科生毕业设计论文工作管理办法为系统基础，搭建便于师生进行操作。系统在实现上具有如下功能：

1．信息管理模块：该模块主要负责对基础用户信息进行管理，主要包括对教师信息、学生信息、教研室信息、专业信息等进行维护管理。

2．课题管理模块：该模块主要负责对课题进行管理，主要包括对课题的申报、审批以及师生之间的双向选题。

3．毕业设计管理模块：该模块主要是负责对从任务书、开题报告、过程检查、指导老师以及评阅老师进行评分到毕业答辩的过程内容进行管理。

4．流程管理模块：该模块主要负责对整个毕业设计流程的控制，保证能对每个学生的毕业设计过程进行监控。

5．消息通知模块：消息通知模块主要是嵌入在系统之中的功能，主要是对具体当前要进行的操作进行一个管理，确保信息的时效性。

6．系统管理员模块：该模块主要是由系统管理员对所新增的学院信息进行添加，保证系统的横向扩展性。

3.1.2 功能模块

毕业设计过程管理系统主要的功能有：信息管理、课题管理、毕业设计管理、流程管理以及消息管理、系统管理员管理等，如图1所示。

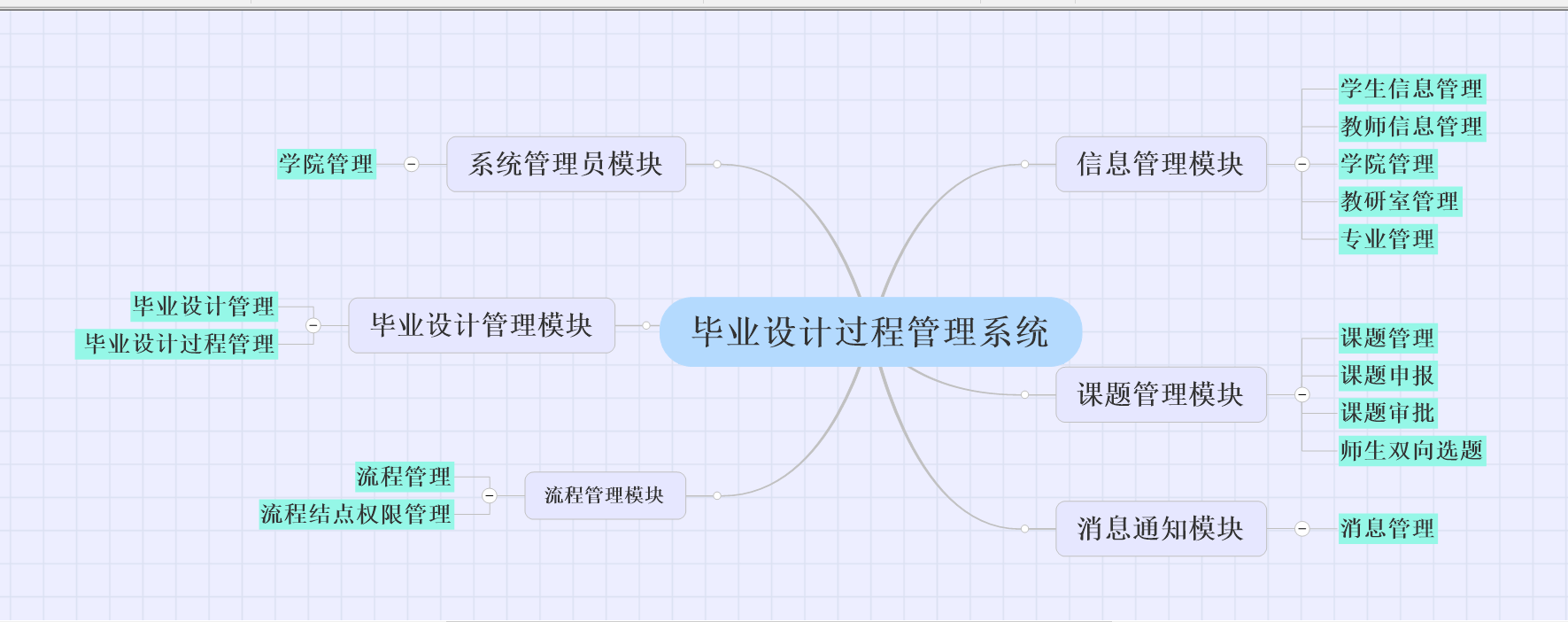


图1 毕业设计过程管理系统的主要功能模块

3.2 性能需求

除了系统的功能性需求，系统的稳定性、易用性、安全性、可扩展性等性能需求也是系统需求分析的重要组成部分，在该系统的设计与实现过程中，需要满足以下需求：

稳定性：系统要求能稳定运行，小规模并发量时，不易产生系统崩溃或响应时间变慢等情况。

易用性：在系统开发中，我们应该充分的考虑用户的使用便捷性，分类清晰性，设计出符合用户操作习惯的信息化系统平台。

可扩展性：系统采用模块化设计，在系统上线后依然能够根据用户的需求，对各大功能模块进行扩充，从而适应用户需求的变化。

正确性：系统应当正确的满足实际需求，保证能完成相应的工作。

第四章 系统设计

4.1 数据库设计

数据库设计(Database Design)是指根据用户需求，在特定的数据库管理系统上设计数据库结构并建立数据库的过程[15]。

数据库是决定系统的质量的关键因素，在系统进行开发前，我们应该充分考虑验证当前数据库的正确性，避免在开发中期因为数据库设计的问题而导致系统难以正确、稳定的运行。所以我们应该严格按照数据库设计的流程，并对用户需求进行建模，最后再进行验证保证得到正确、稳定、高效的数据库[16]。

4.1.1 物理设计

从需求分析可以看出，本系统存在以下几个实体：

1. 学生实体：用于显示学生的信息，包括学号、姓名、密码、年制、证件信息、联系方式、年龄、性别、地址，届别**、**专业信息、学生类型、学院等。

2. 教师实体：用于显示教师的信息，包括教师工号、姓名、性别、入校时间、任职状态、专业技术职称、密码、联系方式、地址等

3. 学院实体：用于显示学院基本信息，包括学院代码、学院名称等。

4. 教研室实体：包括教研室名称、教研室主任Id，所属学院Id等。

5. 专业实体：包括专业编号、专业代码、专业名称、所属教研室等。

6. 管理员实体：包括账号、密码。

7. 课题实体：包括课题编号、课题年份、课题中文名称、课题英文名称、课题实现要求、课题来源、课题性质、课题学生上限、课题备注、课题指导老师、学生集合、课题状态等。

8. 选题实体：包括所属学生、指导老师、评阅老师、所属课题。

9. 流程定义实体：包括流程定义名称、流程定义实例化角色。

10. 流程实例实体：包括流程实例名、所属流程定义ID、流程实例状态、流程实例化者ID。

11. 任务定义实体：包括任务名、所属流程定义ID、任务定义任务类型、任务定义执行角色、任务定义返回的任务定义ID、父任务定义ID。

12. 任务实例实体：任务实例所属流程实例ID、任务实例所属任务定义ID、任务实例执行角色ID、任务实例ID、任务定义返回的任务实例ID、任务实例状态。

13. 通知消息实体：包括通知发起者、通知所属、通知内容、通知状态。

4.1.2 E-R模型设计

根据实体设计，可以得到各个实体E-R图和部分子系统E-R图。

1. 学生实体E-R图，如图2所示。

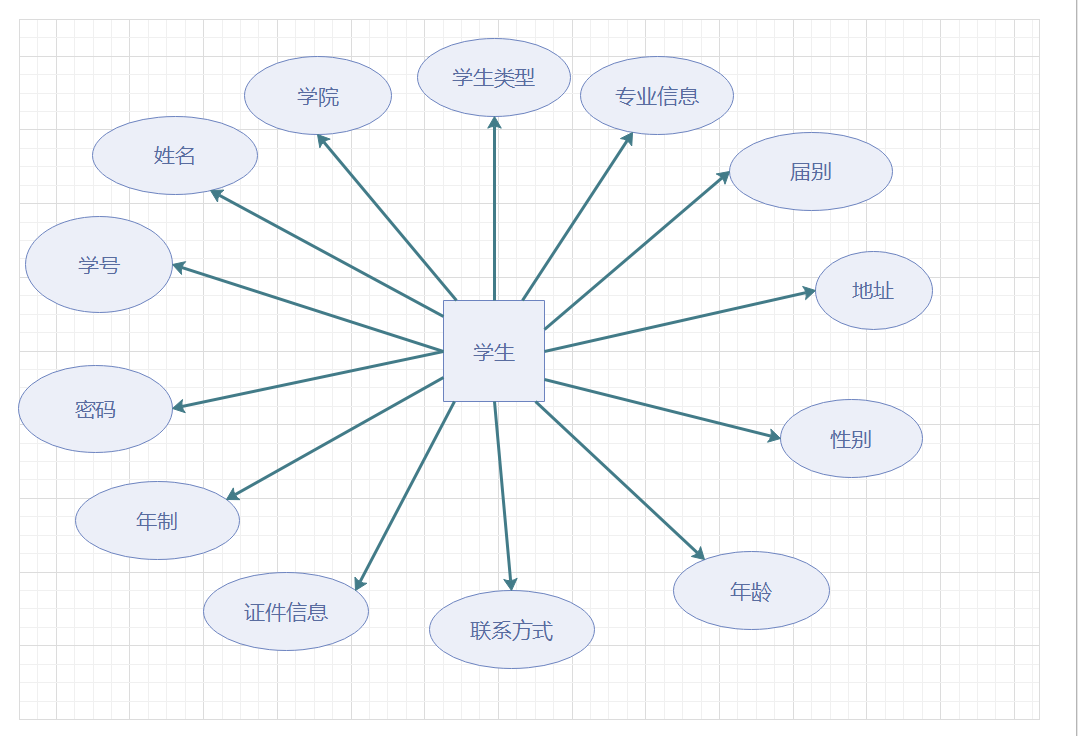


图2 学生实体E-R图

2. 教师实体E-R图，如图3所示。

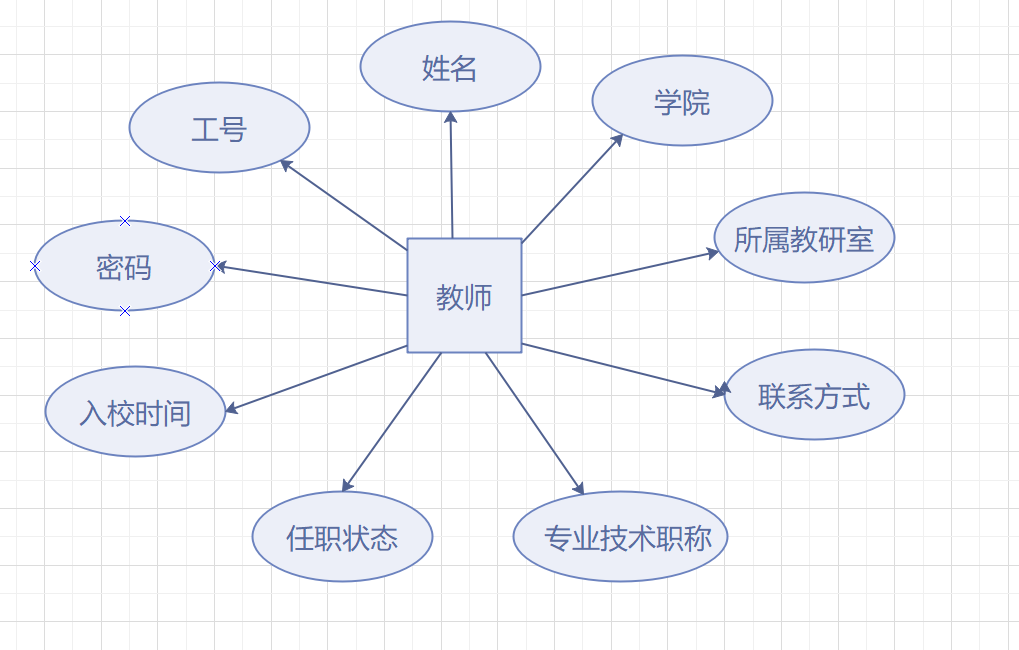


图3 教师实体E-R图

3. 学院实体E-R图，如图4所示。

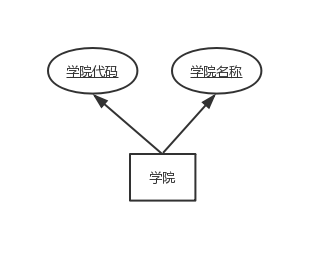


图4 学院实体E-R图

4. 教研室实体E-R图，如图5所示。

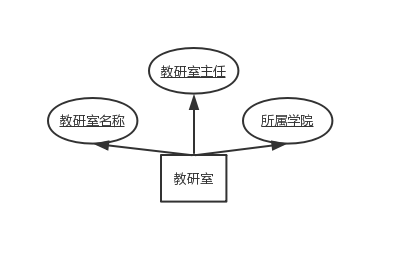


图5 教研室实体E-R图

5. 专业实体E-R图，如图6所示。

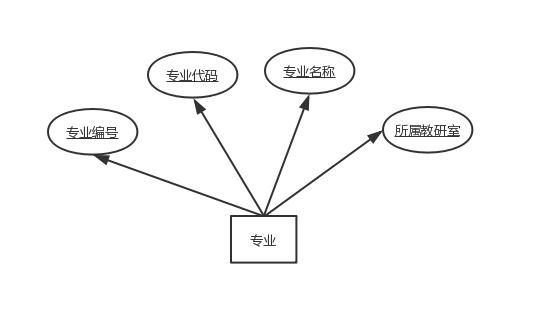


图6 专业实体E-R图

6. 管理员实体E-R图，如图7所示。

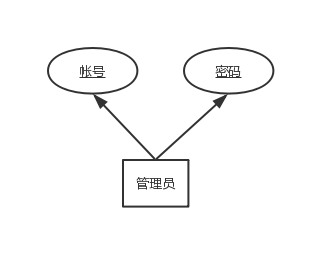


图7 管理员实体E-R图

7. 课题实体E-R图，如图8所示。

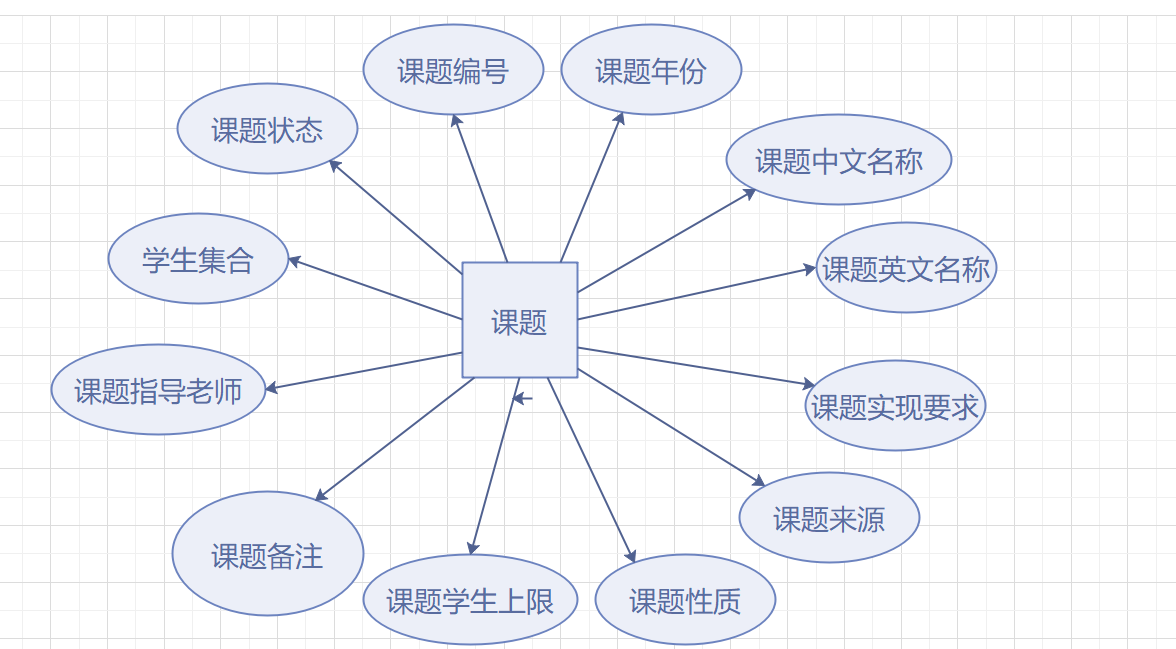


图8 课题实体E-R图

8. 选题实体E-R图，如图9所示。

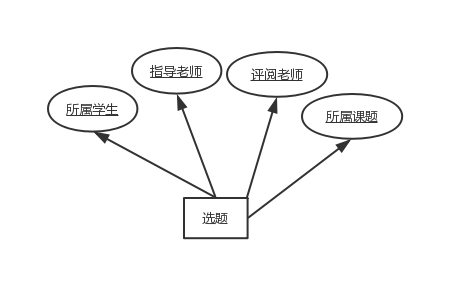


图9管理员实体E-R图

9. 流程定义实体E-R图，如图10所示。

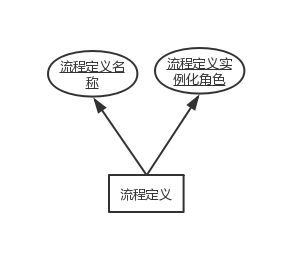


图10 流程定义实体E-R图

10. 流程实例实体E-R图，如图11所示。

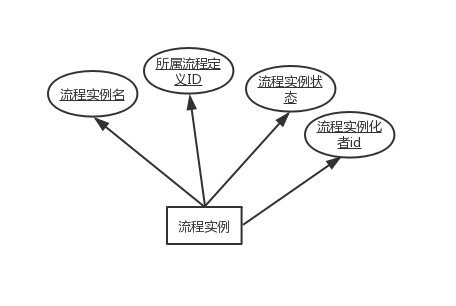


图11流程实例实体E-R图

11. 任务定义实体E-R图，如图12所示。

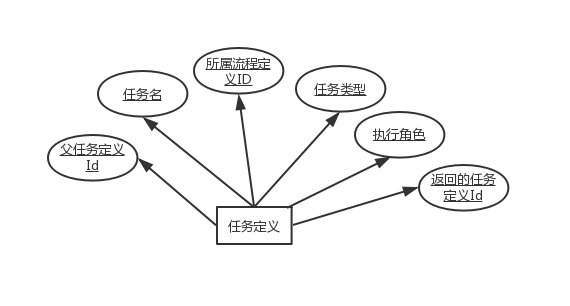


图12任务定义实体E-R图

12. 任务实例实体E-R图，如图13所示。

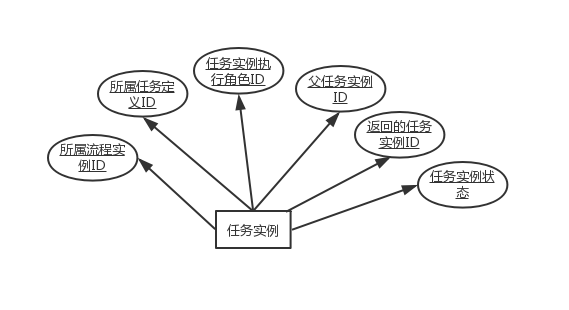


图13任务实例实体E-R图

13. 通知消息实体E-R图，如图14所示。

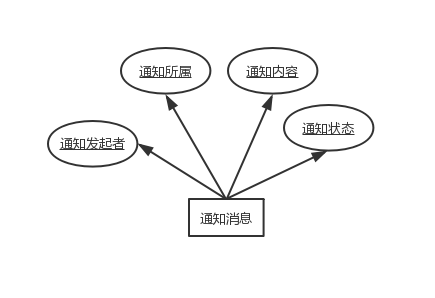


图14通知消息实体E-R图

4.1.3 表/字段及表间关系设计

本系统的数据库其中包括二十五个表，详情请见下面的数据库表

1．教师基础信息表

教师基础信息表存储了教师的非功能基本信息。如表1所示。

表1 教师基础信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| teacher\_basic\_id | char | 40 | N | P\_K |
| job\_number | varchar | 20 | N |  |
| name | varchar | 32 | N |  |
| sex | char | 1 | N |  |
| birthday | varchar | 20 | N |  |
| induction\_date | varchar | 40 |  |  |
| job\_statue | varchar | 2 | N |  |
| unit\_number | char | 20 | N |  |
| unit\_name | varchar | 40 | N |  |
| highest\_education | char | 20 | N |  |
| highest\_degree | char | 20 | N |  |
| learn\_edge\_structure | varchar | 20 | N |  |
| professional\_title | varchar | 20 | N |  |
| subject\_category | varchar | 10 | N |  |
| teaching\_type | varchar | 5 | N |  |
| teaching\_professio\_name | varchar | 20 | N |  |
| teaching\_profession\_no | char | 10 | N |  |
| profession\_teaching\_date | char | 20 |  |  |
| experimental\_technical | char | 2 | N |  |
| double\_teacher\_type | char | 2 | N |  |
| engineering\_background | char | 2 | N |  |
| industry\_background | char | 2 | N |  |
| graduate\_tutor\_type | varchar | 20 |  |  |
| number\_of\_doctor | varchar | 20 |  |  |
| number\_of\_master | varchar | 20 |  |  |
| teacher\_basic\_create | char | 40 | N |  |
| teacher\_basic\_modified | char | 40 | N |  |

2．教师登录表

教师登录表用于存放教师的功能性的基本信息，如表2所示。

表2 教师登录表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| user\_teacher\_id | varchar | 20 | N | P\_K |
| user\_teacher\_num | char | 20 | N |  |
| user\_teacher\_password | char | 15 | N |  |
| user\_teacher\_basic | varchar | 20 | N |  |
| user\_teacher\_section | varchar | 100 | N |  |
| user\_teacher\_guidance | char | 100 | N |  |
| user\_teacher\_guidance | char | 100 | N |  |
| user\_teacher\_create | char | 40 | N |  |
| user\_teacher\_modified | char | 40 | N |  |
| user\_teacher\_is\_recorder | char | 2 | N |  |
| user\_teacher\_college | char | 40 | N |  |
| user\_teacher\_is\_admin | char | 2 | N |  |
| user\_teacher\_is\_leader | char | 2 | N |  |

3. 学生基础信息表

存储了学生基础信息。如表3所示。

表3 学生基础信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| student\_basic\_id | int |  | N | P\_K |
| student\_basic\_num | varchar | 20 | N |  |
| student\_basic\_name | varchar | 10 | N |  |
| student\_basicd\_year | int |  | N |  |
| student\_basic\_grade | int |  | N |  |
| student\_basic\_idtype | char | 2 | N |  |
| student\_basic\_idcaard | varchar | 13 | N |  |
| student\_basic\_age | char | 4 |  |  |
| student\_basic\_sex | char | 2 |  |  |
| student\_basic\_nation | char | 2 |  |  |
| student\_basic\_pliticalvisage | char | 10 |  |  |
| student\_basicative\_place | varchar | 60 |  |  |
| student\_basi\_studenttype | char | 20 | N |  |
| student\_basi\_menttype | char | 20 |  |  |
| student\_basic\_gmethods | char | 20 |  |  |
| student\_basic\_nalcode | char | 20 | N |  |
| student\_basic\_major | char | 20 | N |  |
| student\_basic\_majrname | char | 20 |  |  |
| student\_basic\_is\_major | char | 20 |  |  |
| student\_basic\_is\_ability | char | 2 |  |  |
| student\_basic\_house | char | 2 |  |  |
| student\_basic\_ctiontypes | char | 2 |  |  |
| student\_basic\_ation | varchar | 20 |  |  |
| student\_basic\_mentmode | varchar | 20 |  |  |
| student\_basic\_reason | varchar | 20 |  |  |
| student\_basic\_phone | char | 20 |  |  |
| student\_basic\_college | varchar | 20 |  |  |
| student\_basic\_level | char | 4 |  |  |

4. 学生登录表

存储了学生的功能性的基本信息。如表4所示

表4 学生登录表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| user\_student\_id | char | 40 | N | P\_K |
| user\_student\_num | varchar | 20 | N |  |
| user\_student\_password | char | 11 | N |  |
| user\_student\_basic | char | 40 | N |  |
| user\_student\_is\_operate | int | 0 | N |  |
| user\_student\_is\_topic | int | 0 |  |  |
| user\_student\_be\_college | char | 40 | N |  |
| user\_student\_be\_major | char | 40 | N |  |
| user\_student\_create | char | 40 |  |  |
| user\_student\_modified | char | 40 |  |  |

5. 学生选题表

存储了学生选择课题的基本信息。如表5所示

表5 学生选题表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| topic\_select\_id | char | 40 | N | P\_K |
| topic\_select\_student | char | 40 | N |  |
| topic\_select\_teach\_tutor | char | 40 | N |  |
| topic\_select\_teach\_riew | char | 40 |  |  |
| topic\_select\_topic | char | 40 | N |  |
| topic\_select\_create | char | 20 | N |  |
| topic\_select\_modified | char | 20 | N |  |

6. 管理员信息表

存储了系统管理员的基本信息。如表6所示

表6 管理员信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| admin\_id | int |  | N | P\_K |
| admin\_account | varchar | 20 | N | F\_K |
| admin\_password | char | 11 | N |  |
| admin\_gmt\_create | char | 20 |  |  |
| admin\_gmt\_modified | char | 20 |  |  |

7. 教研室表

存储了教研室的基本信息。如表7所示

表7 教研室表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| section\_id | char | 40 | N | P\_K |
| section\_name | varchar | 20 | N | F\_K |
| section\_college\_id | char | 40 | N |  |
| section\_leader | char | 40 | N |  |
| section\_gmt\_create | char | 20 |  |  |
| section\_gmt\_modified | char | 20 |  |  |

8. 学院表

存储了学院表的基本信息。如表8所示

表8 学院表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| college\_id | char | 40 | N | P\_K |
| college\_code | char | 40 | N |  |
| college\_name | varchar | 11 | N |  |
| college\_gmt\_create | char | 20 |  |  |
| college\_gmt\_modified | char | 20 |  |  |

9. 专业表

存储了专业有关的基本信息。

表9 专业表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| major\_id | int |  | N | P\_K |
| major\_professionalcode | varchar | 20 | N | F\_K |
| major\_name | char | 11 | N |  |
| major\_belong\_section | char | 40 |  |  |
| major\_gmt\_create | char | 20 |  |  |
| major\_gmt\_modified | char | 20 |  |  |

10. 课题表

存储了课题的基本信息。

表10 课题表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| topic\_id | char | 40 | N | P\_K |
| topic\_name\_chinese | varchar | 20 | N | F\_K |
| topic\_name\_english | vhar | 11 | N |  |
| topic\_requirement | text | 0 | N |  |
| topic\_source | char | 10 | N |  |
| topic\_type | char | 10 | N |  |
| topic\_student\_max | int | 11 | N |  |
| topic\_remark | varchar | 512 |  |  |
| topic\_teacher | char | 40 | N |  |
| topic\_student | varchar | 255 |  |  |
| topic\_examine\_state | char | 4 | N |  |
| topic\_student\_num | int | 11 |  |  |
| topic\_gmt\_create | char | 20 |  |  |
| topic\_gmt\_modified | char | 20 |  |  |

11. 任务书表

存储了任务书的基本信息。

表11 任务书表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| taskbook\_id | char | 40 | N | P\_K |
| bysjglxt\_task\_student | varchar | 20 | N | F\_K |
| taskbook\_acon\_required | text |  |  |  |
| taskbook\_reference | text |  |  |  |
| taskbook\_plan | text |  |  |  |
| taskbook\_opinion | text |  |  |  |
| taskbook\_gmt\_create | char | 20 |  |  |
| taskbook\_gmt\_modified | char | 20 |  |  |

12 . 开题报告表

存储了开题报告信息的基本信息。

表12 开题报告信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| report\_opening\_id | char | 40 | N | P\_K |
| report\_opening\_student | varchar | 20 | N | F\_K |
| report\_opening\_survey | text | 0 | N |  |
| report\_opening\_main | text |  |  |  |
| report\_opening\_detail | text |  |  |  |
| report\_opening\_rence | text |  |  |  |
| report\_opening\_plan | text |  |  |  |
| report\_opening\_create | char | 40 |  |  |
| report\_opening\_modified | char | 40 |  |  |

13. 进展情况记录表

存储了进展情况记录的基本信息。

表13 进展情况记录表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| record\_progress\_id | char | 40 | N | P\_K |
| record\_progress\_student | char | 40 | N |  |
| record\_progress\_start | char | 20 | N |  |
| record\_progress\_stop | char | 20 |  |  |
| record\_progress\_record | text |  |  |  |
| record\_progress\_opinion | text |  |  |  |
| record\_progress\_stage | char | 20 | N |  |
| record\_progress\_create | char | 40 |  |  |
| record\_progress\_modified | char | 40 |  |  |

14. 个人学习工作总结表

存储了个人学习工作总结信息的基本信息。

表14 个人学习工作总结表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| summary\_id | char | 40 | N | P\_K |
| summary\_student | char | 40 | N |  |
| summary\_gmt\_start | char | 20 | N |  |
| summary\_gmt\_stop | char | 20 | N |  |
| summary\_summary | text |  |  |  |
| summary\_opinion | text |  |  |  |
| summary\_gmt\_create | char | 40 |  |  |
| summary\_gmt\_modified | char | 40 |  |  |

15. 毕业论文表

存储了毕业论文信息的基本信息。

表15 毕业论文表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| dissertation\_id | char | 40 | N | P\_K |
| dissertation\_student | char | 40 | N |  |
| dissertation\_file | text | 0 | N |  |
| dissertation\_create | char | 40 |  |  |
| dissertation\_modified | char | 40 |  |  |

16. 形式审查表

存储了形式审查表的基本信息。

表16 形式审查表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| examination\_formal\_id | char | 40 | N | P\_K |
| formal\_student | char | 40 | N |  |
| formal\_is\_cover | int | 11 | N |  |
| formal\_is\_a4 | int | 11 | N |  |
| formal\_is\_format | int | 11 | N |  |
| formal\_abstract\_chinese | int | 11 | N |  |
| is\_chinese\_keyword | int | 11 | N |  |
| formal\_is\_catalog | int | 11 | N |  |
| is\_abstract\_foreign | int | 11 | N |  |
| formal\_is\_headline | int | 11 | N |  |
| formal\_is\_punctuation | int | 11 | N |  |
| formal\_is\_typo | int | 11 | N |  |
| formal\_is\_reference\_ten | int | 11 | N |  |
| formal\_reference\_foreign | int | 11 | N |  |
| formal\_is\_reference\_new | int | 11 | N |  |
| formal\_is\_reference\_num | int | 11 | N |  |
| formal\_reference\_format | int | 11 | N |  |
| formal\_is\_metaphase | int | 11 | N |  |
| formal\_is\_summary | int | 11 | N |  |
| formal\_rogress\_actual | int | 11 | N |  |
| formal\_gress\_complete | int | 11 | N |  |
| formal\_is\_progress\_logic | int | 11 | N |  |
| formal\_is\_chart | int | 11 | N |  |
| formal\_is\_enclosure | int | 11 | N |  |
| formal\_teacher\_opinion | int | 11 | N |  |
| formal\_is\_leader\_opinion | char | 20 |  |  |
| formal\_gmt\_create | char | 20 |  |  |

17. 指导教师评价表

存储了指导教师评价表的基本信息。

表17 指导教师评价表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| evaluate\_tutor\_id | char | 40 | N | P\_K |
| evaluate\_tutor\_student | char | 40 | N |  |
| teacher\_comment | text | 0 | N |  |
| grade\_training\_objective | int | 11 |  |  |
| grade\_difficulty | int | 11 |  |  |
| grade\_workload | int | 11 |  |  |
| grade\_bind | int | 11 |  |  |
| grade\_comprehensive | int | 11 |  |  |
| grade\_reference | int | 11 |  |  |
| grade\_mental\_design | int | 11 |  |  |
| grade\_computing | int | 11 |  |  |
| grade\_foreign\_language | int | 11 |  |  |
| grade\_computer | int | 11 |  |  |
| grade\_innovate | int | 11 |  |  |
| grade\_analysis | int | 11 |  |  |
| grade\_chart | int | 11 |  |  |
| grade\_instructions | int | 11 |  |  |
| grade\_practicability | int | 11 |  |  |
| grade\_normalization | int | 11 |  |  |
| grade\_total | int | 11 |  |  |
| is\_teacher\_opinion | int | 11 |  |  |
| gmt\_create | char | 20 |  |  |
| gmt\_modified | char | 20 |  |  |

18. 评阅老师评价表

存储了评阅老师的基本信息。

表18 评阅老师评价表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| evaluate\_review\_id | char | 40 | N | P\_K |
| evaluate\_review\_student | char | 40 | N |  |
| teacher\_comment | text | 0 | N |  |
| grade\_training\_objective | char | 40 |  |  |
| grade\_difficulty | char | 40 |  |  |
| grade\_workload | int | 11 |  |  |
| grade\_bind | int | 11 |  |  |
| grade\_comprehensive | int | 11 |  |  |
| grade\_reference | int | 11 |  |  |
| grade\_experimental\_design | int | 11 |  |  |
| grade\_computing | int | 11 |  |  |
| grade\_foreign\_language | int | 11 |  |  |
| grade\_computer | int | 11 |  |  |
| grade\_innovate | int | 11 |  |  |
| grade\_analysis | int | 11 |  |  |
| grade\_chart | int | 11 |  |  |
| grade\_instructions | int | 11 |  |  |
| grade\_practicability | int | 11 |  |  |
| grade\_normalization | int | 11 |  |  |
| grade\_total | int | 11 |  |  |
| is\_teacher\_opinion | int | 11 |  |  |
| gmt\_create | int | 11 |  |  |

19. 答辩评分及成绩评定表

存储了答辩评分及成绩评定信息。

表19 答辩评分及成绩评定表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| defence\_id | char | 40 | N | P\_K |
| defence\_student | char | 40 | N |  |
| defence\_record | text | 0 | N |  |
| defence\_leade\_comment | char | 40 |  |  |
| defence\_grade\_writing | char | 40 |  |  |
| defence\_normalization | text | 0 |  |  |
| defence\_grade\_complete | int | 11 |  |  |
| defence\_technology | int | 11 |  |  |
| defence\_practicability | int | 11 |  |  |
| defence\_appearance | int | 11 |  |  |
| defence\_grad\_statement | int | 11 |  |  |
| defence\_grade\_answer | int | 11 |  |  |
| defence\_grade\_defence | int | 11 |  |  |
| defence\_evaluate\_tutor | int | 11 |  |  |
| defence\_evaluate\_review | int | 11 |  |  |
| defence\_total | double | 11 |  |  |
| defence\_finally | double | 11 |  |  |
| defence\_gmt\_create | int | 11 |  |  |
| defence\_gmt\_modified | char | 4 |  |  |

20. 流程定义表

存储了流程定义信息的基本信息。

表20 流程定义表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| process\_definition\_id | char | 40 | N | P\_K |
| process\_definition\_name | varchar | 100 | N |  |
| process\_def\_ance\_role | int | 11 | N |  |
| process\_def\_gmt\_create | char | 40 |  |  |
| process\_def\_modified | char | 40 |  |  |

21. 任务定义表

存储了任务定义信息的基本信息。

表21 任务定义表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| process\_instance\_id | char | 40 | N | P\_K |
| process\_instance\_name | varchar | 100 | N |  |
| process\_cess\_definition | int | 11 | N |  |
| process\_instance\_state | char | 40 |  |  |
| process\_instance\_man | char | 40 |  |  |
| process\_instance\_create | int | 11 | N |  |
| process\_instance\_modified | int | 11 | N |  |
| process\_instance\_id | char | 40 |  |  |
| process\_instance\_name | char | 40 |  |  |

22. 任务实例表

存储了任务实例信息的基本信息。

表22 任务实例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| task\_definition\_id | char | 40 | N | P\_K |
| task\_definition\_name | varchar | 100 | N |  |
| process\_definition | int | 11 | N |  |
| task\_definition\_type | char | 40 |  |  |
| task\_definition\_role | char | 40 |  |  |
| task\_definition\_return | char | 40 | N |  |
| task\_definition\_father | char | 40 | N |  |
| task\_definition\_create | int | 11 |  |  |
| task\_definition\_modified | varchar | 100 |  |  |

23. 流程实例表

存储了流程实例信息的基本信息。

表23 流程实例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| task\_instance\_id | char | 40 | N | P\_K |
| process\_instance | varchar | 100 | N |  |
| task\_definition | int | 11 | N |  |
| task\_instance\_role | varchar | 255 |  |  |
| task\_instance\_father | char | 40 |  |  |
| task\_instance\_return | char | 40 |  |  |
| task\_instance\_state | char | 40 |  |  |

24. 通知表

存储了通知的基本信息。

表24 通知表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| notice\_id | char | 40 | N | P\_K |
| notice\_launch | varchar | 100 | N |  |
| notice\_belong | int | 11 | N |  |
| notice\_content | varchar | 255 |  |  |
| notice\_state | int | 11 | N |  |
| notice\_gmt\_create | char | 40 |  |  |
| notice\_gmt\_modified | char | 40 |  |  |

25. 评语表

存储了评语的基本信息。

表25 评语表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 数据长度 | 是否为空 | Key |
| comment\_id | char | 40 | N | P\_K |
| comment\_grade | varchar | 100 | N |  |
| comment\_category | int | 11 | N |  |
| comment\_content | varchar | Text |  |  |
| comment\_college | char | 40 |  |  |
| comment\_gmt\_create | char | 40 |  |  |
| comment\_gmt\_modified | char | 40 |  |  |

4.2 模块设计

4.2.1 信息管理模块

该模块的主要操作角色为学院管理员，学院管理员在登录后能对教师的信息、学生的信息、教研室信息以及专业信息进行维护及管理。是系统的数据基础，采用excel表格数据进行导入并且进行修改的方式，保证学生、教师、教研室和专业的数据完善和时效。如图15，图16，图17所示。

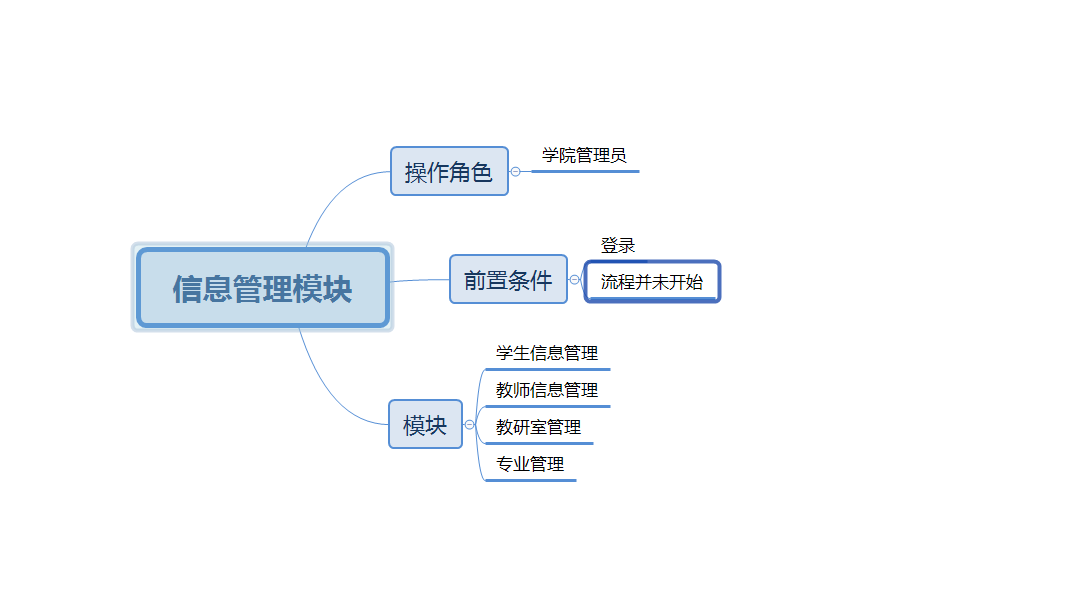


图15 信息管理模块

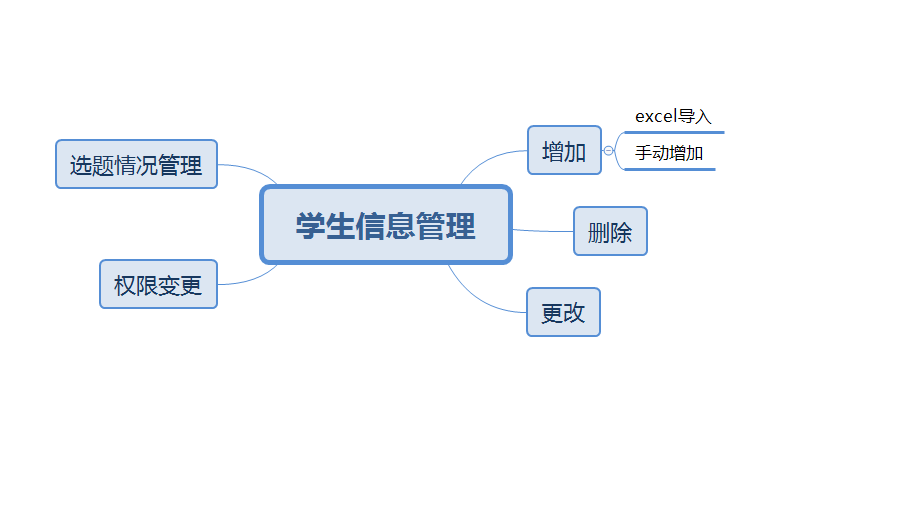


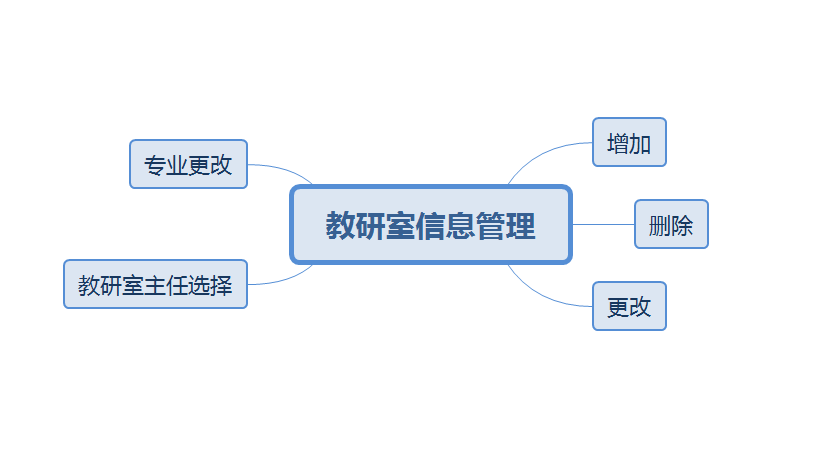
图16 学生信息管理

图17 教研室信息管理

4.2.2 课题管理模块

该模块主要负责对课题进行管理，主要包括对课题的申报、审批以及师生之间的双向选题，核心是要完成教师创建课题、学院管理员审批课题、学生选题的功能流程。具体的流程如图18所示。

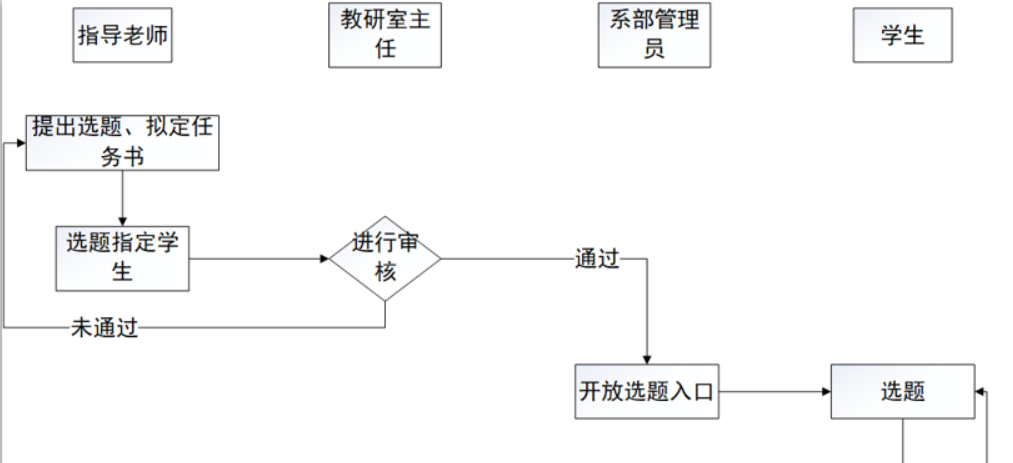


图18 选题流程

4.2.3 毕业设计管理模块

该模块主要是负责对从任务书、开题报告、过程检查、指导老师以及评阅老师进行评分到毕业答辩的过程内容进行管理。流程图如19所示。

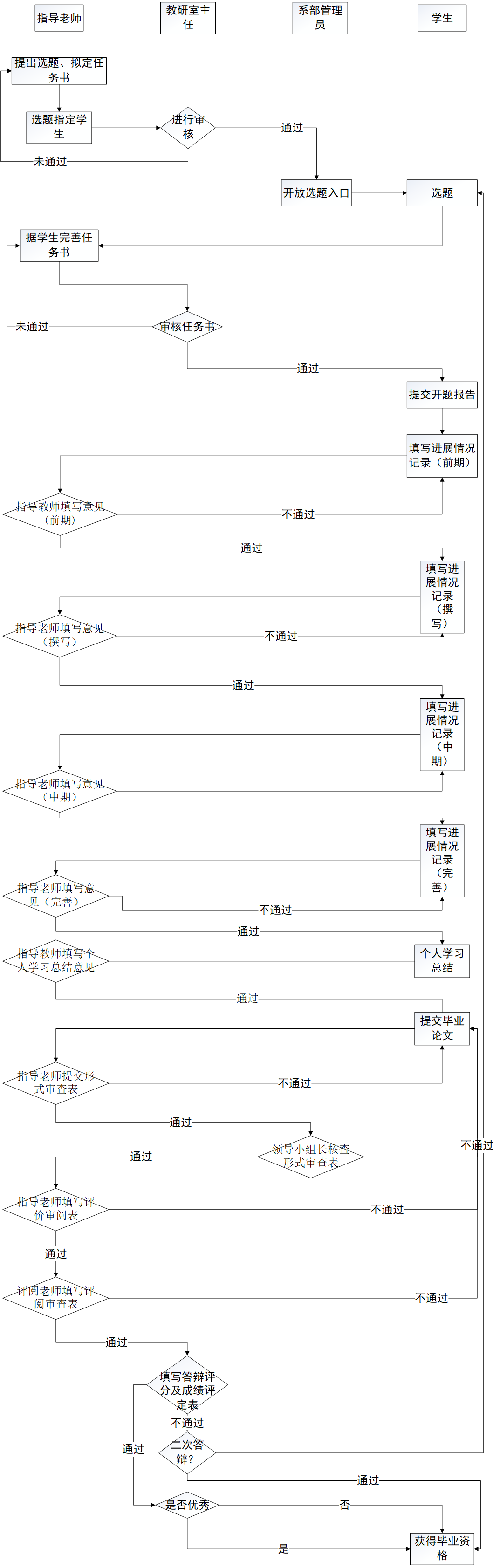


图19 毕业设计流程

4.2.4 流程管理模块

该模块主要是对整个毕业设计过程中的流程节点进行控制，保证能对每个学生的毕业设计过程中进行监控以及管理。如图20，图21，图22所示。

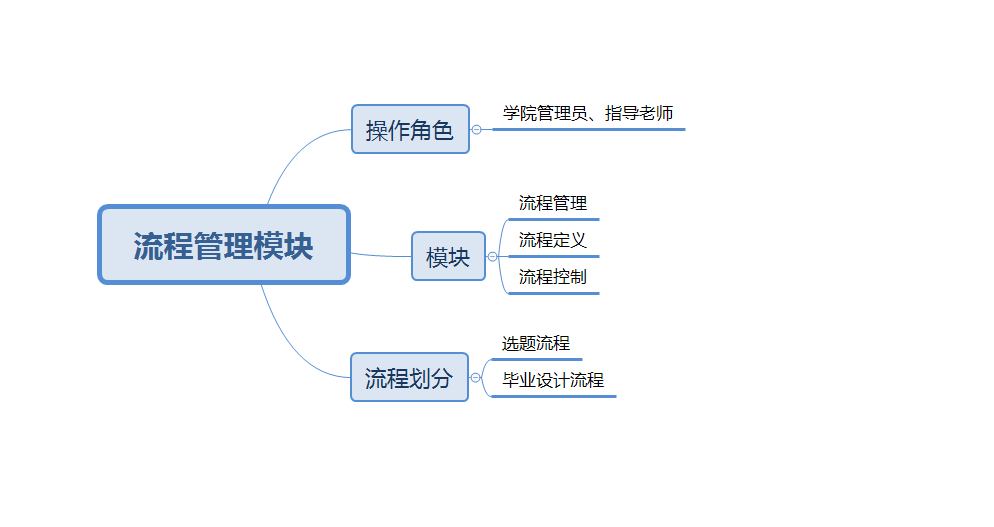


图20 流程管理

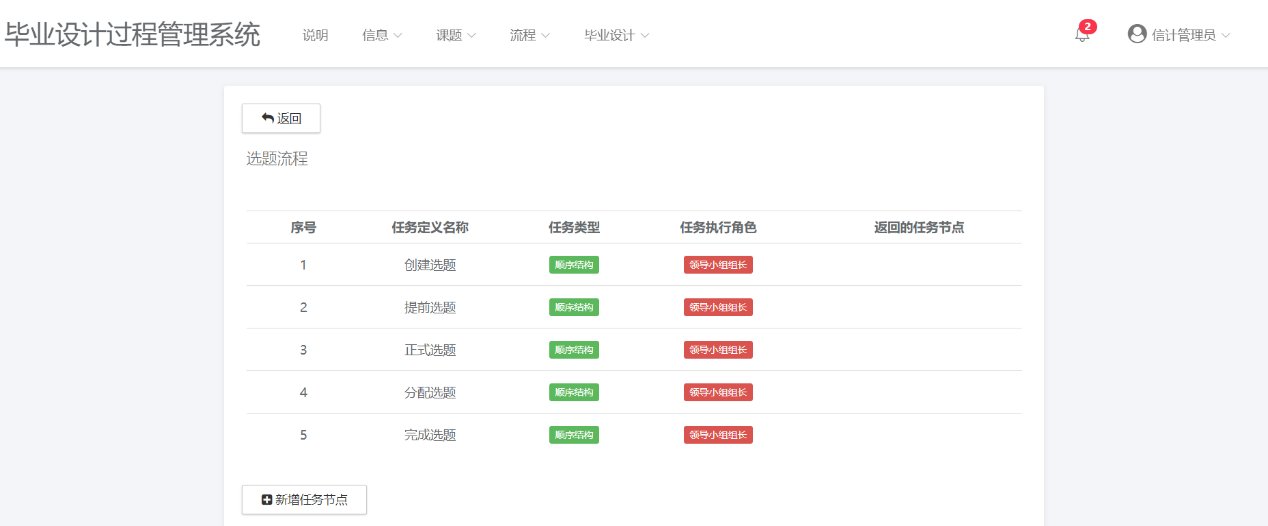


图21 选题流程管理页面



图22毕业设计流程管理页面

4.2.5 消息通知模块

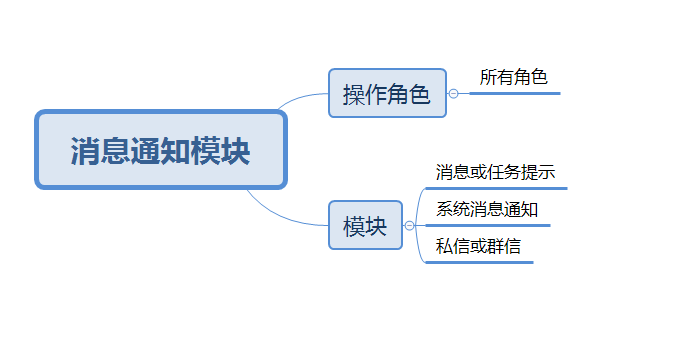
消息通知模块主要是嵌入在系统之中的功能，主要是对具体当前要进行的操作进行一个管理，确保信息的时效性，同时能针对性的进行web端的非即时通讯。如图23，图24所示。

图23 消息通知模块

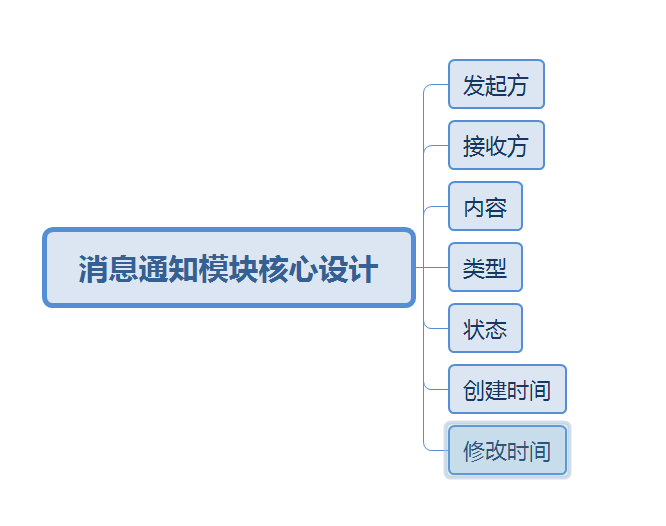


图24 消息通知模块核心设计

4.2.6 系统管理员模块

系统管理员主要是对学院进行管理，保证系统的横向扩展性，对于系统管理员来说，主要的操作就是添加新学院，并针对学院的负责人进行修改。如图25所示。

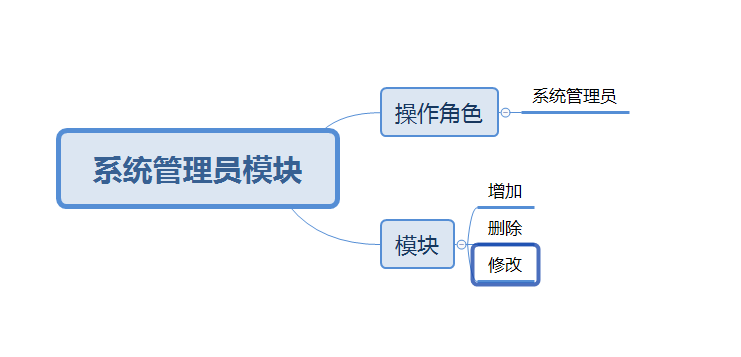


图25 系统管理员模块

第五章 系统的编码与实现

5.1 系统的实现平台

5.1.1 系统环境

1．系统软件环境

（1）服务端

操作系统：Windows Server 2008

数据库系统：MySQL5.5

（2）客户端

操作系统：适用于各种平台；

开发工具：IDEA，Navicat8.0，Hbuilder；

开发语言：Java，Javascript，HTML，CSS。

5.1.2 后台数据库选择及配置

本系统在数据方面决定采用MySQL数据库，原因有几下几点：

1. MySQL简单实用，而且在日常的学习中有更多的资料以及老师能进行辅导。

2. MySQL的性能非常的好，并且非常的稳定，还很少出现异常问题或宕机问题。

3. MySQL是开放源代码的，所以没有版权制约，在选择上，使用成本低

4. MySQL软件的体积非常小，并且安装使用特别简单，容易维护，安装，维护成本低。

5. MySQL用的人比较多，比如像阿里用的就是用的MySQL。

6. MySQL对Java语言有较好的支持，而我们所采用的就是Java语言，能较好的支持，相对而言，就能很好的使用我们的系统了。

5.2 子系统功能模块的实现

5.2.1 信息管理模块

信息管理模块可以分为：学生信息管理模块和教师信息管理模块，信息管理模块主页面菜单如图26所示。



图26信息管理模块菜单

1. 学生信息管理

学生信息管理主页面如图27所示。

学生信息管理主要是对学生的基础信息进行维护，主要保证学生信息在平台上是正确有效的，同时也希望能对学生的基本权限进行控制，对学生的选题结果进行控制。

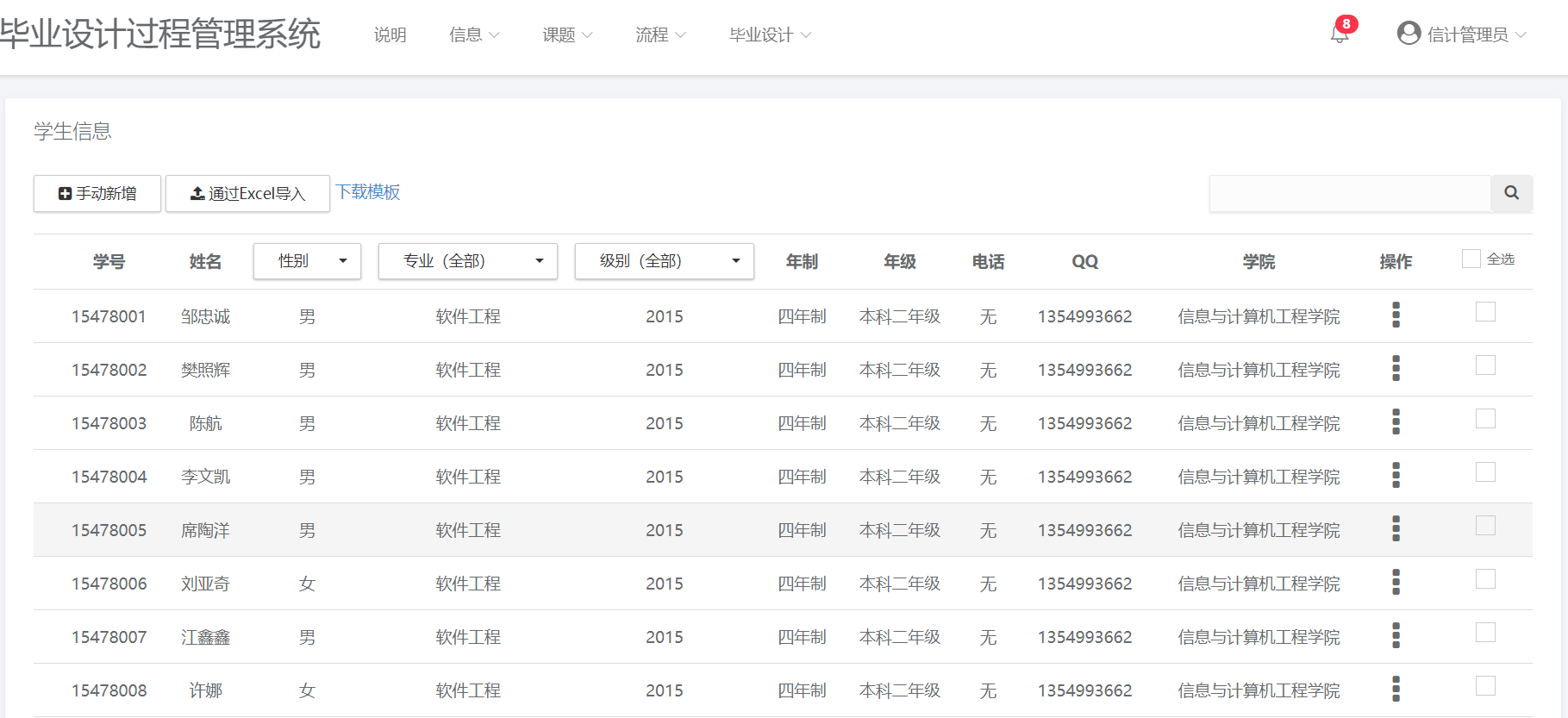


图27学生信息管理主页面

2. 教师信息管理

教师信息管理主页面如图28所示。

教师信息管理主要是提供教师基础信息，为学生选择指导老师提供一定的帮助。同时，也间接的对教师的基础信息进行系统内部的维护。



图28教师信息管理主页面

在进行教师和学生信息管理的时候，我们通常需要通过Excel将信息进行导入，减少人工对基础信息的重复维护，导入信息的具体实现如下：

/\*\*

\* 利用反射将List<Map<String,Object>> 结构生成相应的List<T>数据

\*/

public static <T> List<T> toObjectPerproList(List<Map<String, Object>> list, Class<T> clazz) throws Exception {

T t = null;

List<T> returnList = new LinkedList<T>();

for (int i = 0; i < list.size(); i++) {

// 利用反射创建Class对应的对象

t = clazz.newInstance();

// 遍历Map对象，用反射填充对象的属性

for (Map.Entry<String, Object> ent : list.get(i).entrySet()) {

String name = ent.getKey();

Object value = ent.getValue();

// 用反射赋值

setFieldValue(t, name, value);

}

returnList.add(t);

}

return returnList;

}

/\*\*

\* 2003<将excel表中的数据放入到List<Map<String, Object>>中

\*

\* @param workbook

\* @return

\*/

public static <T> List<Map<String, Object>> parseUpdateExcel(Workbook workbook, String user) {

userPropertiesBundle = ResourceBundle.getBundle(user);

List<Map<String, Object>> result = new LinkedList<Map<String, Object>>();

XSSFRow row\_0 = (XSSFRow) workbook.getSheetAt(0).getRow(0);

int excleRowLength = row\_0.getPhysicalNumberOfCells();

// 相应的javabean类的属性名称数组

String[] columnName = new String[excleRowLength]; try {

// 从资源文件中获取

for (int i = 0; i < columnName.length; i++) {

// 获取第0行i列的数据

String str = row\_0.getCell(i).getStringCellValue().trim();

str = new String(str.getBytes("UTF-8"), "ISO-8859-1").trim();

if (userPropertiesBundle.containsKey(str)) {

columnName[i] = userPropertiesBundle.getString(str);

}

}

} catch (UnsupportedEncodingException e) {

e.printStackTrace();

}

for (int sheetIndex = 0; sheetIndex < workbook.getNumberOfSheets(); sheetIndex++) {

// sheet

XSSFSheet sheet = (XSSFSheet) workbook.getSheetAt(sheetIndex);

// row

for (int rowIndex = 1; rowIndex < sheet.getLastRowNum()+1; rowIndex++) {

XSSFRow row = sheet.getRow(rowIndex);

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

for (int cellIndex = 0; cellIndex < row.getLastCellNum(); cellIndex++) {

// cell

XSSFCell cell = row.getCell(cellIndex);

if (columnName[cellIndex] != null && columnName[cellIndex].trim().length() > 0) {

// 该列值在对应的java对象中有值

// 取出当前cell的值和对应Javabean类的属性放入到map中

map.put(columnName[cellIndex].trim(), getCellValue(cell));

}

}

result.add(map);

}

}

return result;

}

5.2.2 课题管理模块

课题管理模块主要是对本年课题以及历年课题进行管理，同时在内部完成教师创建选题、学院管理员的审核选题、以及师生之间的互选[17]。

本年课题主页面如图29所示。



图29本年课题主页面

历年课题主页面如图30所示。



图30历年课题主页面

学院管理员对课题进行审核页面如图31所示。



图31审核课题

在对于师生进行互选的过程中，我们主要采用的是教师创建好课题后可以指定学生提前进行选题，一个学生可以被指定为多个课题。当然，被指定的学生也可以提前进行选题等待正式选题流程开始一起进行选题，以此来达到师生互选的功能。

指定学生提前选题页面如图32所示。



图32指定学生提前选题

学生选题主页面如图33所示。



图33学生提前选题

指定学生提前选题的模块主要是通过将学生的信息以数组的格式保存在数据库中，然后再在前端页面对数据进行读取和判断，以保证前端页面的正确展示。具体实现如下。

// 指定学生选题

@Override

public boolean distributionTopicStudent(String topicID, String studentID) {

boolean flag = false;

bysjglxt\_notice bysjglxt\_notice = null;

bysjglxt\_topic bysjglxt\_topic = new bysjglxt\_topic();

bysjglxt\_notice = new bysjglxt\_notice();

bysjglxt\_topic = topicInformationManagementDao.getBysjglxtTopicById(topicID);

if (bysjglxt\_topic == null)

return false;

// 判断指定学生中是否存在某学生

if (bysjglxt\_topic.getTopic\_student() != null && bysjglxt\_topic.getTopic\_student().trim().length() > 0) {

if ((bysjglxt\_topic.getTopic\_student().trim()).indexOf(studentID) != -1) {

// 如果包含 的话，也就是说是调用了取消指定的方法

bysjglxt\_topic.setTopic\_student(bysjglxt\_topic.getTopic\_student().replaceAll(studentID + "#&#", "")); bysjglxt\_notice.setNotice\_content(((String)(properties.get("removeZaoSelectTopic"))).replaceAll("topic\_num", bysjglxt\_topic.getTopic\_num()).replaceAll("topic\_chinese", bysjglxt\_topic.getTopic\_name\_chinese()));

} else {

// 如果不包含,则是调用的指定的方法

bysjglxt\_topic.setTopic\_student(bysjglxt\_topic.getTopic\_student() + studentID + "#&#");

bysjglxt\_notice.setNotice\_content(((String)(properties.get("addZaoSelectTopic"))).replaceAll("topic\_num", bysjglxt\_topic.getTopic\_num()).replaceAll("topic\_chinese", bysjglxt\_topic.getTopic\_name\_chinese()));

}

} else {

bysjglxt\_topic.setTopic\_student(studentID + "#&#");

bysjglxt\_notice.setNotice\_content(((String)(properties.get("addZaoSelectTopic"))).replaceAll("topic\_num", bysjglxt\_topic.getTopic\_num()).replaceAll("topic\_chinese", bysjglxt\_topic.getTopic\_name\_chinese()));

}

// 通知学生

bysjglxt\_notice.setNotice\_id(TeamUtil.getUuid());

bysjglxt\_notice.setNotice\_launch(bysjglxt\_topic.getTopic\_teacher());

bysjglxt\_notice.setNotice\_belong(studentID);

bysjglxt\_notice.setNotice\_gmt\_create(TeamUtil.getStringSecond());

bysjglxt\_notice.setNotice\_leixing(4);

bysjglxt\_notice.setNotice\_state(2); bysjglxt\_notice.setNotice\_gmt\_modified(bysjglxt\_notice.getNotice\_gmt\_create());

flag = topicInformationManagementDao.saveObj(bysjglxt\_notice);

// 课题表

bysjglxt\_topic.setTopic\_gmt\_modified(TeamUtil.getStringSecond());

flag = topicInformationManagementDao.CreateTopic(bysjglxt\_topic);

return flag;

}

5.2.3 毕业设计管理模块

毕业设计管理模块主要是针对毕业设计过程管理手册的内容及流程来进行设计的，按照角色来划分的话，主要分成下面几块：

毕业设计管理

我的毕业设计

我指导的学生

我评阅的学生

毕业设计管理主界面如图34所示。

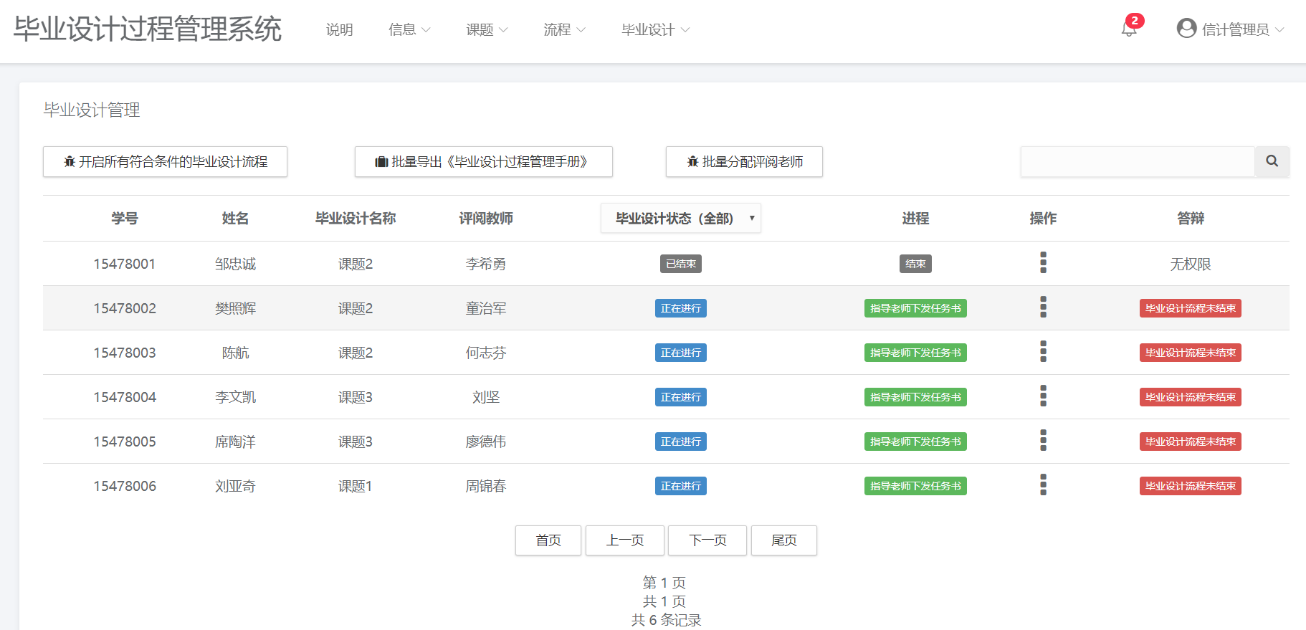


图34毕业设计管理主页面

在该页面中，主要能查看到学生的毕业设计进度情况、对学生的流程中的结点进行控制、以及进入学生的毕业设计过程管理手册页面。

流程结点控制主页面如图35所示。



图35流程结点控制

学生的毕业设计主页面展示如图36所示。



图36学生的毕业设计主页面

我指导的学生主页面展示如图37所示。

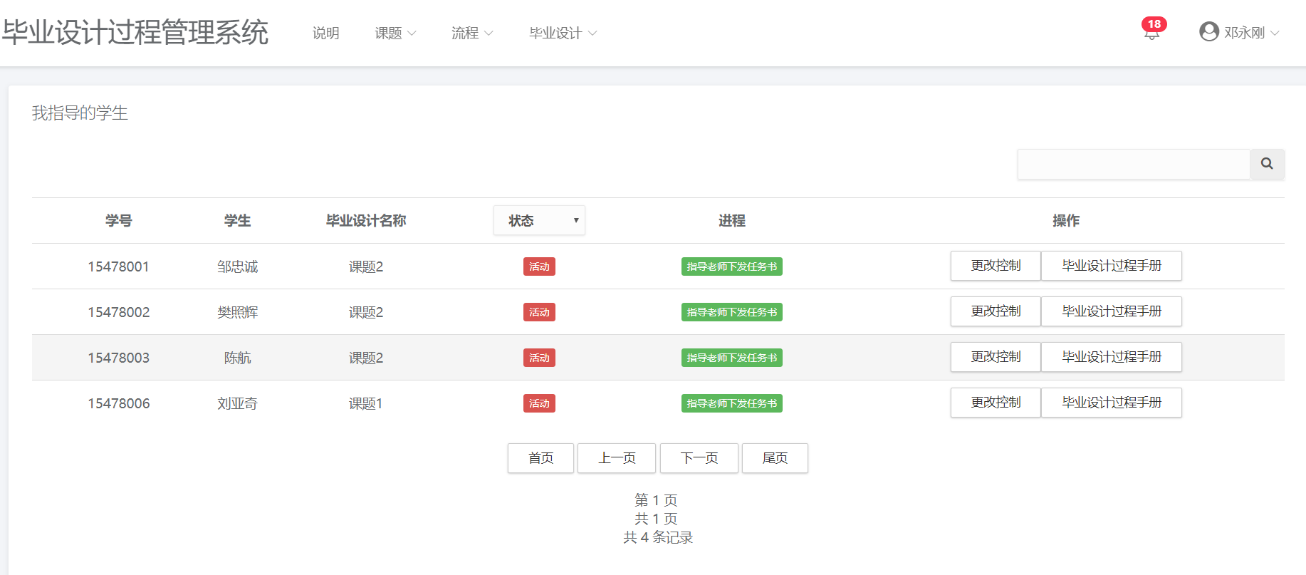


图37我指导的学生

在进行毕业设计过程的时候，我们往往需要师生之间的文件往来，保证相互之间的沟通和过程文档的正确性，所以我们采用一个文件交互的模式，师生之间可以通过文件来进行交互并且利用我们的消息平台对双方之间进行消息的互通，确保不会出现问题。具体页面展示如图38所示。



图38文件交互实现

/\*\*

\* 保存毕业论文

\*

\* @throws @IOException

\*/

@Override

public int saveDissertation(File file, String oldFileName, String userId, String newFileName) throws IOException {

bysjglxt\_student\_user studentUser = new bysjglxt\_student\_user();

studentUser = graduationProjectManagementDao.getStudentUserByUserId(userId);

if(studentUser==null) {

return 0;

}

//获取学院信息

bysjglxt\_college college = new bysjglxt\_college();

college = graduationProjectManagementDao.getCollegeById(studentUser.getUser\_student\_belong\_college());

//获取课题最大的year

String year = graduationProjectManagementDao.getMaxTopicYear();

/\*

\* 获取路径

\*/

String lj = "";

try {

Properties props = new Properties();

props.load(this.getClass().getClassLoader().getResourceAsStream("file.properties"));

lj = props.getProperty("lj");

} catch (Exception e) {

e.printStackTrace();

}

boolean flag = false;

String path = "";

bysjglxt\_dissertation bysjglxt\_dissertation = new bysjglxt\_dissertation();

// 如果新文件为空

if (file == null) {

path = lj + "bysjglxt/"+college.getCollege\_code()+"/"+year+"/毕业论文/";

// 判断旧文件是否处于空的状态

if ("".equals(oldFileName) || !(oldFileName.trim().length() > 0)) {

// 如果是空

// 判断：在数据库中属于这个学生的毕业论文是否存在

bysjglxt\_dissertation = graduationProjectManagementDao.getThesisByStudent(userId);

if (bysjglxt\_dissertation != null && bysjglxt\_dissertation.getDissertation\_student() != null

&& bysjglxt\_dissertation.getDissertation\_student().trim().length() > 0) {

// 如果存在,则将原有毕业论文删除

// 先进行删除

// 删除学生上传的文件

path = path + bysjglxt\_dissertation.getDissertation\_file();

File deleteFile = new File(path);

deleteFile.delete();

flag = graduationProjectManagementDao.deleteThesisByUserId(userId);

return 1;

} else {

// 如果不存在,不进行任何操作

return 1;

}

} else {

// 如果不是空

// 不进行操作

return 1;

}

} else {

path = lj + "bysjglxt/"+college.getCollege\_code()+"/"+year+"/毕业论文/";

// 如果新文件存在

bysjglxt\_dissertation = graduationProjectManagementDao.getThesisByStudent(userId);

// 1.判断是否有属于这个学生的毕业论文存在

if (bysjglxt\_dissertation != null && bysjglxt\_dissertation.getDissertation\_student() != null

&& bysjglxt\_dissertation.getDissertation\_student().trim().length() > 0) {

// 如果存在

// 先进行删除

// 删除学生上传的文件

path = path + bysjglxt\_dissertation.getDissertation\_id() + bysjglxt\_dissertation.getDissertation\_file();

File deleteFile = new File(path);

deleteFile.delete();

// 删除学生毕业论文记录

flag = graduationProjectManagementDao.deleteThesisByUserId(userId);

if (!flag) {

return -1;

}

}

path = lj + "bysjglxt/"+college.getCollege\_code()+"/"+year+"/毕业论文/";

// 保存毕业论文

// 1.上传毕业论文

bysjglxt\_dissertation = new bysjglxt\_dissertation();

bysjglxt\_dissertation = graduationProjectManagementDao.getThesisByStudentId(userId);

if (bysjglxt\_dissertation != null) {

String fileNameP = studentUser.getUser\_student\_num()+"BYLW."+newFileName.substring(newFileName.lastIndexOf(".")+1);

path = path + fileNameP;

File newFile = new File(path);

FileUtils.copyFile(file, newFile);

// 存储数据到数据库

bysjglxt\_dissertation.setDissertation\_file(fileNameP);

bysjglxt\_dissertation.setDissertation\_gmt\_modified(bysjglxt\_dissertation.getDissertation\_gmt\_create());

flag = graduationProjectManagementDao.fillEmptyThesisRecord(bysjglxt\_dissertation);

if (!flag)

return -2;

} else {

return -2;

}

}

return 1;

}

5.2.4 流程管理模块

流程管理模块主要是对整个系统的流程进行拆分，我们将毕业设计过程管理工作划分为两个流程：

选题流程

毕业设计流程

这两个流程分别主要是对课题、毕业设计过程管理手册进行管理，完成对流程的控制。

针对流程的实现，在实现上，并没有引入别的框架而是借助框架的思想来自己进行简单的实现，保证对系统进行无依赖的松耦合[18]。

具体的实现过程主要是使用了一种符合系统需要的数据库设计，主要核心字在于如图39所示。



图39流程数据库实现

在上文中，说到过流程控制，对于流程控制，在实现上主要是针对性的添加对应的是否可编辑的状态，然后再在前台对页面进行控制，就能达到相应的功能了。

/\*\*

\* 开启毕业设计流程

\*

\*/

@Override

public int openGraduProcess(String teacherUserId) {

// 获取系部管理员学院

String college = collegeJudge(1, teacherUserId);

// 根据学生userId获取学生姓名

List<String> listGraduProcess = new ArrayList<>();

String studentName;

// 根据流程定义名获取流程定义ID

bysjglxt\_process\_definition processDefinition = new bysjglxt\_process\_definition();

// 根据流程定义

processDefinition = processManagementDao.getProcessDefinitionByName("毕业设计流程");

if (processDefinition == null)

return 2;

// 获取需要开启流程的学生列表

listGraduProcess = listOpenGraduationProjectProcessStudentId(processDefinition.getProcess\_definition\_id(),

college);

// 遍历学生依次开启毕业设计流程

for (String stringId : listGraduProcess) {

studentName = null;

int i = graduationProjectManagementService.startGraduationProjectProcess(stringId);

if (i != 1) {

return -3;

}

studentName = processManagementDao.getStudentNameByUserId(stringId);

if (studentName == null || studentName.trim().length() <= 0) {

continue;

}

openProcess(processDefinition.getProcess\_definition\_name() + "——" + studentName,

processDefinition.getProcess\_definition\_id(), stringId, 2);

if (i != 1) {

return -3;

}

}

if(processDefinition!=null && "毕业设计流程".equals(processDefinition.getProcess\_definition\_name())) {

bysjglxt\_teacher\_user teacherUser = processManagementDao.getTeacherByCollege(college);

String launch = teacherUser.getUser\_teacher\_id();

List<TeacherInformationDTO> listTeacherUser = processManagementDao.getTeacherUserByCollegeId(college);

List<StudentInformationDTO> listStudentUser = processManagementDao.getStudentUserByCollegeId(college);

//通知该系部所有拥有教研室老师

for (TeacherInformationDTO teacherInformationDTO : listTeacherUser) {

if(teacherInformationDTO.getBysjglxtTeacherUser().getUser\_teacher\_id().equals(launch)) {

createTeacherNotice(teacherInformationDTO.getBysjglxtTeacherUser().getUser\_teacher\_id(),launch,"cretaeAdminGraduation",2);

continue;

}

createTeacherNotice(teacherInformationDTO.getBysjglxtTeacherUser().getUser\_teacher\_id(),launch,"bootGraduation",4);

}

for (StudentInformationDTO studentInformationDTO : listStudentUser) {

createTeacherNotice(studentInformationDTO.getBysjglxtStudentUser().getUser\_student\_id(),launch,"bootGraduation",4);

}

bysjglxt\_notice notice = processManagementDao.getNoticeByBelongContentAndLeiXing(launch, ((String)(properties.get("createGraduationProcess"))),2);

notice.setNotice\_leixing(1);

notice.setNotice\_state(1);

notice.setNotice\_gmt\_modified(TeamUtil.getStringSecond());

processManagementDao.saveObj(notice);

}

return 1;

}

**5.2.5 消息通知模块**

消息通知模块主要是嵌入在系统中的，在整个系统启用的过程中，起到了非常重要的作用。主要是通过线上的群体或个体性的消息交互来达到消息通知的效果，进一步加强系统的功能粘合性[19]。

在消息通知模块中，主要涉及到三个方面：

1. 消息或任务提示

2. 系统消息通知

3. 私信或群信

消息或任务提示主要是在系统的首页进行展示的，主要是方便使用者在进行系统后能非常方便直观的观察到目前需要完成哪些功能或者已经完成了哪些功能、待完成的功能。页面展示如图40所示。



图40消息或任务提示

系统消息通知主要是通过气泡来显示的全局展示有哪些问题或消息还未读的，此时我们可以点击气泡进入消息页面，点击详情即可将消息的状态变成已读的状态。



图41系统消息

私信或群信则是相互之间的系统内部信息交互通道，主要是通过信息发送来实现通信的效果，已经具体到可以针对性的发送消息。具体实现页面如图42所示。



图42通信实现页面

// 获得所有未读状态的消息

@Override

public List<NoticeDTO> listNotice(String userId) {

List<NoticeDTO> listNoticeDTO = new ArrayList<NoticeDTO>();

NoticeDTO noticeDTO = null;

List<bysjglxt\_notice> listNotice = new ArrayList<bysjglxt\_notice>();

// 获取所有未读状态的消息

listNotice = noticeManagementDao.getListNoticeByBelongAndState(userId, 2);

// 遍历所有listNotice

for (bysjglxt\_notice bysjglxt\_notice : listNotice) {

noticeDTO = new NoticeDTO();

noticeDTO.setBysjglxt\_notice(bysjglxt\_notice);

noticeDTO.setLaunchName(getUserNameByUserId(bysjglxt\_notice.getNotice\_launch())); noticeDTO.setBelongName(getUserNameByUserId(bysjglxt\_notice.getNotice\_belong()));

listNoticeDTO.add(noticeDTO);

}

return listNoticeDTO;

}

@Override

public int getNoticeStateCount(String userId) {

List<bysjglxt\_notice> listNotice = new ArrayList<bysjglxt\_notice>();

listNotice = noticeManagementDao.getListNoticeCount(userId, 2);

return listNotice.size();

}

**5.2.6 系统管理员模块**

系统管理员模块主要是针对学院的管理控制，系统管理员可以在该页面对学院进行管理。如图43所示。

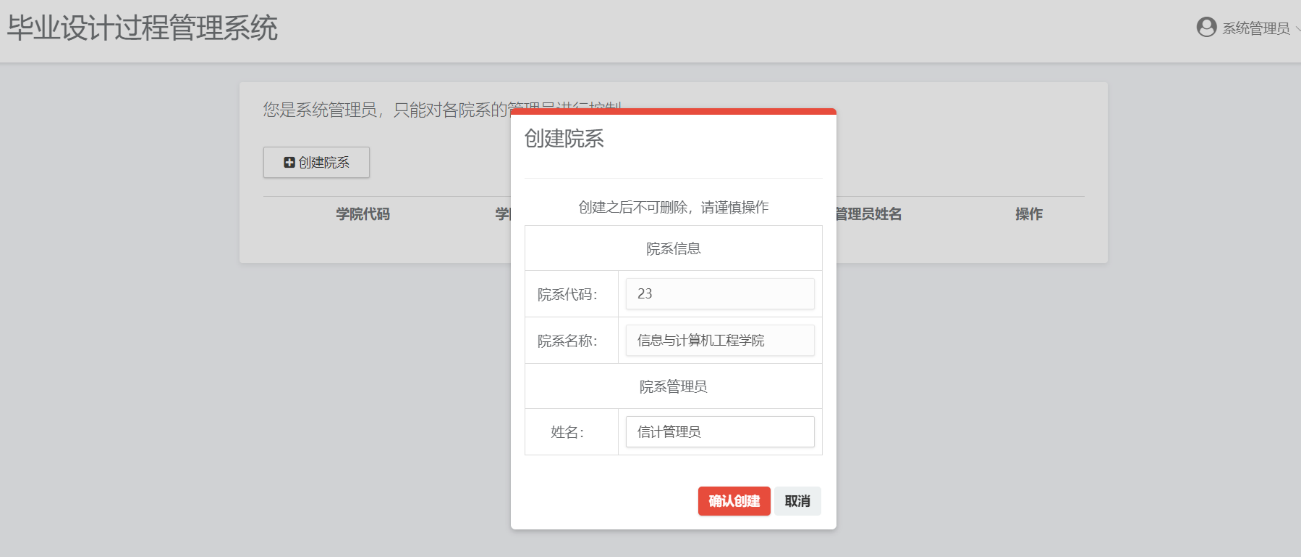


图43系统管理员首页

/\*\*

\* 根据学院编号获取学院管理员信息

\* @param college\_code

\* @return

\*/

@Override

public CollegeInformationDTO getCollegeAdminInfoByCollegeCode(String college\_code) {

bysjglxt\_college college = new bysjglxt\_college();

CollegeInformationDTO collegeInformationDTO = new CollegeInformationDTO();

TeacherInformationDTO teacherInformationDTO = new TeacherInformationDTO();

if(college\_code==null || "".equals(college\_code)) {

return null;

}

college = collegeManagementDao.getCollegeByCode(college\_code);

if(college == null) {

return null;

}

//根据该学院的学院信息获取学院管理员信息

bysjglxt\_teacher\_user teacherUser = new bysjglxt\_teacher\_user();

teacherUser = collegeManagementDao.getTeacherUserByCollegeId(college.getCollege\_id());

if(teacherUser !=null && teacherUser.getUser\_teacher\_basic()!=null && !"".equals(teacherUser.getUser\_teacher\_basic())) {

teacherInformationDTO.setBysjglxtTeacherUser(teacherUser);

bysjglxt\_teacher\_basic teacherBasic = new bysjglxt\_teacher\_basic();

teacherBasic = collegeManagementDao.getTeacherBasicById(teacherUser.getUser\_teacher\_basic());

if(teacherBasic!=null) {

teacherInformationDTO.setBysjglxtTeacherBasic(teacherBasic);

}

if(teacherUser.getUser\_teacher\_section()!=null && !"".equals(teacherUser.getUser\_teacher\_section().trim())) {

bysjglxt\_section section = new bysjglxt\_section();

section = collegeManagementDao.getSectionById(teacherUser.getUser\_teacher\_section().trim());

if(section!=null) {

teacherInformationDTO.setBysjglxtSection(section);

}

}

}

collegeInformationDTO.setCollege(college);

collegeInformationDTO.setTeacherInformationDTO(teacherInformationDTO);

return collegeInformationDTO;

}