

毕业设计管理系统--

设计报告

姓名： 于豪杰

班级： 计科2203

学号： 2022040055

指导教师： 郭俊霞

**目录**

第一部分——开发前的设计和思考 .................................... 4

一、需求分析 .................................................. 4

1. 用户角色分析 ........................................... 4

2. 功能需求分析 ........................................... 5

3. 业务流程分析 ........................................... 6

二、概念结构设计 .............................................. 6

1. 实体分析 ............................................... 6

2. 关系分析 ............................................... 6

3. 实体关系图设计 ......................................... 7

三、逻辑结构设计 .............................................. 8

1. 主要关系模式设计 ....................................... 8

2. 范式分析过程 ........................................... 9

3. 完整性约束说明 .........................................10

四、物理结构设计 ..............................................11

1. 存储结构设计 ...........................................11

2. 文件组织方式 ...........................................12

第二部分——性能优化的思考 ........................................12

1. 索引设计 ..................................................12

1.数据量与性能关系.........................................12

2.查询时间改善.............................................12

3.行数减少效果.............................................13

4.综合评分.................................................13

5.性能分布.................................................14

二、视图设计 ..................................................14

1. 视图需求分析 ...........................................14

2. 视图实现方案 ...........................................16

三、触发器设计 ................................................16

1. 业务场景分析 ...........................................16

2. 触发器性能考虑 .........................................16

3. 触发器实现方案 .........................................16

第三部分、数据库运行与维护 ........................................19

一、转储与恢复 ................................................19

二、安全性与完整性控制 ........................................19

1. 用户权限管理方案 .......................................19

2. 数据加密方案 ...........................................19

3. 日志管理方案 ...........................................19

三、性能的监督、分析与改进 ....................................20

四、重组织与重构造 ............................................20

第四部分——数据库实施 ............................................20

一、技术栈选择 ................................................20

1. 技术选型依据 ...........................................20

2. 开发环境搭建 ...........................................21

二、前端页面设计 ..............................................21

1. 用户界面设计 ...........................................21

2. 页面布局设计 ...........................................23

3. 交互设计 ...............................................23

4. 响应式设计 .............................................24

三、后端对接数据库实现 ........................................25

1. 接口设计 ...............................................25

2. 数据访问层设计 .........................................26

3. 业务逻辑层设计 .........................................27

4. 异常处理机制 ...........................................28

第五部分——总结与反思 ............................................30

一、目前存在的问题 ............................................30

1. 系统功能方面 ...........................................30

2. 技术实现方面 ...........................................30

3. 安全性方面 .............................................30

二、未来改进的方向 ............................................30

1. 功能扩展 ...............................................30

2. 技术优化 ...............................................31

3. 安全加强 ...............................................31

三、项目经验总结 ..............................................31

1. 技术收获 ...............................................31

2. 项目管理经验 ...........................................31

3. 个人成长总结 ...........................................31

**第一部分——开发前的设计和思考**

**一、需求分析**

***题目要求：***

实践内容20 毕业设计管理系统  
建立一个毕业设计管理系统, 编写应用程序完成系统开发。  
1. 建立基本表：  
题目表（题目编号，题目名称，题目类型，课题来源，指导教师，内容简介，专业要求，学生要求）  
指导教师信息表（教师号，姓名，性别，出生日期，学历，职称，研究方向，电话，邮箱）  
学生信息表（学号，姓名，性别，专业，班级，电话，邮箱，备注）  
选题表（题目编号，题目名称，学号，姓名，性别，专业，审核）  
毕业设计进程表（任务书，文献综述，开题报告，原文翻译，中期检查，论文评审，导师意见，论文答辩）  
用户表（用户号，用户名，密码，用户类别）  
2. 要求实现以下功能：  
（1）登录功能  
系统有五类用户：教师、学生、管理员、系统管理员、访客  
（2）论文题目管理  
要求：论文题目申报，系主任审批，汇总发布  
（3）选题管理  
要求：学生选题，教师认可，系主任审批，选题结果发布  
（4）进程管理  
要求：任务书下达，文献综述提交，开题报告提交及确认，原文翻译提交，中期报告提交及中期检查，论文提交，导师评阅，同行评阅，论文答辩  
（5）用户管理  
要求：用户信息查询，用户注册，用户信息维护，用户权限修改  
（6） 系统维护

**1.用户角色分析**

系统共有五类用户角色，每个角色具有不同的权限和职责：

(1)教师

待审核学生选题申请:审核申请自己管理的题目的学生申请

论文题目管理：添加毕设题目以供学生选择，题目添加后需要等系主任审核才可被选择

查看学生进度：实时监督自己指导的学生目前位于哪个阶段

进度审核：实现进程管理中的前六个部分的审核功能，分别是任务书管理、文献综述评阅、开题报告评阅、外文翻译评阅、中期检查评阅、论文提交通过确认

论文评阅：包括自己指导的学生的论文评阅与被系主任分配的同行评阅

答辩评审：最终答辩中自己在内的答辩组委会中所需的答辩评审

（2）学生

选题申请：毕业设计的题目申请，在已通过的题目列表中进行选择和申请，等待后续教师和系主任审核

个人进度管理：在申请选题之后对自己的九个进程阶段的进度查看与文件提交入口，与教师进行交互的功能，包括前六个阶段的文件提交以及后续参加答辩

（3）管理员（系主任）

按我对本项目要求的理解，直接将系主任等同于管理员

选题列表审核：审核教师提交的题目，若满足要求即可通过题目申请

学生选题申请：审核教师通过的学生选题申请

用户管理：设定是管理员应当可对管理员身份以下的账号进行管理，主要是处理日常学院内教师和学生的账号问题

论文评阅分配：在学生提交论文并教师确认之后系主任对此学生的评阅人进行分配，本学院内挑选不同的评阅人实现导师评阅和同行评阅

答辩管理：系主任负责组建答辩委员会和答辩安排功能，与其他教师协商之后直接在平台上进行分配

（4）软件系统管理员

由于要求中没有提及太多，我自行设计了一些功能

系统日志查看：用户所有有关对数据库的写操作均被记录，其余的部分查询、登录等操作也被记录

系统消息：设计了系统消息查看以便维护系统

用户管理：系统管理员可对除了系统管理员以外的所有人的账号进行编辑

系统设置：对数据库进行备份、查看当前数据库状态，导出数据库中表的内容

（5）访客

由于是外部人员，只能查看选题列表和答辩安排信息，便于师生在忘记自身账号密码等紧急状况下查询必要公开信息，以及为非本校用户进行查看参考学习

**2.功能需求分析**

基础功能需求：

 用户登录与认证、多角色登录界面、身份认证、密码加密存储、登录状态维护、论文题目管理、教师题目申报、系主任题目审批、题目信息发布、题目信息修改、选题管理、学生选题申请、教师选题确认、系主任选题审批、选题结果公示、进度管理、任务书管理、文献综述管理、开题报告管理、原文翻译管理、中期检查管理、论文评审管理、答辩管理

扩展功能需求：

 消息通知系统、数据统计分析、文件在线预览、进度提醒、批量导入导出、评审意见反馈

**3.业务流程分析**

选题流程：

图示

低可信度描述已自动生成

进度管理流程：



答辩流程：



**二、概念结构设计**

**1.实体分析**

主要实体包括：

(1) 用户实体

基本属性：用户ID、用户名、密码、角色、姓名、性别等

扩展属性：部门、职称、专业、联系方式等

状态属性：登录次数、最后登录时间、账号状态等

(2) 论文选题实体

基本属性：题目ID、标题、类型、来源

描述属性：内容描述、专业要求、学生要求

状态属性：审核状态、创建时间、更新时间

(3) 进度管理实体

阶段定义：阶段ID、名称、类型、顺序、权重

进度记录：学生ID、阶段类型、状态、分数、评语

时间属性：提交时间、审核时间、截止时间

(4) 答辩管理实体

委员会：ID、名称、主席、秘书、委员

答辩安排：时间、地点、时长、状态

答辩成绩：各项评分、总分、评语、结果

**2.关系分析**

(1) 师生关系

导师指导学生（1:n）

体现在task\_assignments表中

包含选题分配和任务书下达

(2) 评审关系

教师评审论文（m:n）

通过thesis\_reviews\_assignment表实现

区分导师评阅和同行评阅

(3) 进度管理关系

学生-进度阶段（1:n）

通过progress\_records表记录

包含文件提交和教师审核

(4) 答辩关系

答辩委员会-学生（1:n）

通过defense\_arrangements表安排

包含时间地点和答辩成绩

**3.E-R图设计**

主要实体-属性图

图示

描述已自动生成

图示

描述已自动生成

E-R设计图

图示

描述已自动生成

**三、逻辑结构设计**

**1. 主要关系模式设计**

(1) 用户管理模块

Users(user\_id, username, password, role, staff\_id, student\_id, login\_count, name, gender, department, major, title, email, phone, introduction, status, last\_login\_time, last\_login\_ip, avatar\_url)

主码：user\_id

外码：无

（2）选题管理模块

Topics(topic\_id, title, type, source, teacher\_id, description, major\_requirements, student\_requirements, status)

主码：topic\_id

外码：teacher\_id 参照 Users(user\_id)

Topic\_Selections(selection\_id, student\_id, topic\_id, selection\_time, status, priority, reason)

主码：selection\_id

外码：student\_id 参照 Users(user\_id)

      topic\_id 参照 Topics(topic\_id)

唯一约束：(student\_id, topic\_id)

（3）进度管理模块

Progress\_Stages(id, name, type, sequence, required\_files, requires\_review, weight, description, status)

主码：id

外码：无

Progress\_Records(id, student\_id, stage\_type, status, file\_url, teacher\_comment, score, submit\_time, review\_time, deadline)

主码：id

外码：student\_id 参照 Students(student\_id)

唯一约束：(student\_id, stage\_type)

（4）论文评审模块

Thesis\_Submissions(id, student\_id, advisor\_id, version, file\_url, file\_size, title, keywords, abstract, status, advisor\_review\_status, advisor\_review\_time, advisor\_comments)

主码：id

外码：student\_id 参照 Students(student\_id)

advisor\_id 参照 Users(user\_id)

Thesis\_Reviews\_Assignment(id, student\_id, reviewer\_id, review\_type, status, assigned\_at, completed\_at)

主码：id

外码：student\_id 参照 Students(student\_id)

reviewer\_id 参照 Users(user\_id)

**2. 范式分析过程**

(1) 第一范式(1NF)检查

 所有表的属性都是原子的，不可再分

 例如：用户表中的联系方式分为email和phone两个字段

 所有表都满足1NF

(2) 第二范式(2NF)检查

 检查是否存在部分函数依赖

 例如：在Task\_Assignments表中，所有非主键属性都完全依赖于task\_id

 在Topic\_Selections表中，selection\_time、status、priority、reason都完全依赖于selection\_id

 所有表都满足2NF

(3) 第三范式(3NF)检查

 检查是否存在传递函数依赖

 Users表中将用户基本信息(Students)和用户认证信息分开存储，避免了传递依赖

 Defense\_Scores表中的总分(total\_score)依赖于各个分项分数，但这是计算所得，不构成传递依赖

 所有表都满足3NF

(4) BCNF(Boyce-Codd范式)检查

 检查是否所有决定因素都是候选键

 Users表分析：

 user\_id是主键

 staff\_id和student\_id是唯一键

 这些键都能唯一确定其他属性，满足BCNF

 Defense\_Committee表分析：

 chairman\_id、secretary\_id等都依赖于id(主键)

 不存在非主键属性对其他属性的决定，满足BCNF

 Topics表分析：

 topic\_id是主键

 teacher\_id只是外键，不能决定其他属性

 满足BCNF

 Task\_Assignments表分析：

 task\_id是主键

 (teacher\_id, student\_id, topic\_id)组合可以作为候选键

 其他属性都依赖于主键，满足BCNF

**3. 完整性约束说明**

(1) 实体完整性

 主键约束：所有表都定义了主键

 自动增长：使用AUTO\_INCREMENT确保ID唯一性

PRIMARY KEY (`user\_id`)

PRIMARY KEY (`topic\_id`)

(2) 参照完整性

 外键约束：使用FOREIGN KEY确保关联数据有效性

 级联操作：某些关系设置了ON DELETE CASCADE

*CONSTRAINT* `thesis\_submissions\_ibfk\_1`

*FOREIGN KEY* (`student\_id`)

*REFERENCES* `students` (`student\_id`)

*ON DELETE CASCADE*

(3) 用户定义的完整性

 非空约束：NOT NULL

 默认值：DEFAULT

 检查约束：enum类型

 唯一性约束：UNIQUE

`status` enum('active','inactive','locked')

*DEFAULT* 'active'

UNIQUE KEY `idx\_staff\_id` (`staff\_id`)

 进度阶段顺序控制

 评分范围限制

 状态转换规则

`score` decimal(5,2) CHECK (score >= 0 AND score <= 100)

`sequence` int CHECK (sequence > 0)

**四、物理结构设计**

**1. 存储结构设计**

(1) 表空间设计

*使用InnoDB存储引擎，支持事务和外键*

ENGINE=InnoDB

*字符集使用utf8mb4，支持完整的Unicode字符*

*DEFAULT* CHARSET=utf8mb4

*COLLATE*=utf8mb4\_0900\_ai\_ci

(2) 主要表的存储结构

用户表：频繁访问，需要高效索引

Users表：

- 主键索引：user\_id (聚集索引)

- 唯一索引：idx\_staff\_id, idx\_student\_id

- 组合索引：idx\_username\_status (username,status)

选题表：需要支持复杂查询

Topics表：

- 主键索引：topic\_id (聚集索引)

- 组合索引：idx\_teacher\_status (teacher\_id,status)

- 普通索引：idx\_teacher\_id (teacher\_id)

进度记录表：高频写入

Progress\_Records表：

- 主键索引：id (聚集索引)

- 唯一索引：unique\_student\_stage (student\_id,stage\_type)

**2. 文件组织方式**

(1) 索引组织

*主键索引采用B+树结构*

*PRIMARY KEY* (`id`)

*二级索引也使用B+树，存储主键值*

KEY `idx\_student\_status` (`student\_id`,`status`)

(2) 数据存储

 行格式：DYNAMIC（默认）

 页大小：16KB（InnoDB默认）

 BLOB/TEXT类型单独存储

**第二部分——性能优化的思考**

**一、索引设计**

**1.数据量与性能关系：**

大数据量表（>1000行）的索引平均性能提升更显著

progress\_records（7212行）student\_stage\_scores（5978行）表现最好

小数据量表的性能提升不稳定

图片包含 文本

描述已自动生成

*图1：性能提升与数据量的散点图*

**2.查询时间改善：**

 平均查询时间减少了约30%

 users表的索引改善最明显

 有些索引反而增加了查询时间，需要优化

图表, 条形图

描述已自动生成

图2：*查询时间对比图（前15个表）*

**3.行数减少效果：**

 大多数索引（>80%）实现了99%以上的行数减少

 只有少数索引效果不理想（如progress\_stages.sequence）

背景图案

描述已自动生成

图3：*行数减少百分比图*

**4.综合评分：**

计算公式如下：



高分索引主要集中在大表上

progress\_records和student\_stage\_scores的索引表现最好

小表的索引综合评分普遍较低

图表, 条形图

描述已自动生成

图4：*综合评分排名（Top 15）*

**5.性能分布：**

 约40%的索引性能表现良好（>1.5倍提升）

 约20%的索引需要优化（<1倍提升）

 中等性能（1-1.5倍提升）占比最大

图表, 条形图

描述已自动生成

图5：*性能分布统计*

**二、视图设计**

**1. 视图需求分析**

根据毕业设计管理系统的业务需求，主要有以下几类视图需求：

(1) 选题管理需求

 需要实时掌握选题情况统计

 需要查看教师指导学生情况

 需要了解选题审批状态

(2) 进度管理需求

 需要监控学生各阶段进度

 需要统计各阶段完成情况

(3) 论文评审需求

 需要查看评审分配情况

 需要汇总评审意见和分数

 需要追踪评审进度

(4) 答辩管理需求

 需要查看答辩安排详情

 需要统计答辩成绩

 需要汇总答辩委员会信息

(5) 综合统计需求

 需要统计学生全过程成绩

 需要评估教师指导效果

 需要分析学生整体表现

**2. 视图实现方案**

(1) 选题相关视图

v\_topic\_selection\_status：选题情况统计视图

功能：统计每个选题的选择人数和批准人数

实现：通过LEFT JOIN连接topics和topic\_selections表

v\_teacher\_supervision\_stats：教师指导情况统计视图

功能：统计教师指导的学生数量和平均成绩

实现：通过多表JOIN实现教师-学生-成绩关联

(2) 进度管理视图

v\_student\_progress\_overview：学生进度总览视图

功能：显示学生各阶段进度状态和时间情况

实现：通过JOIN连接progress\_records和progress\_stages表

(3) 论文评审视图

v\_thesis\_review\_assignments：论文评审分配视图

功能：显示论文评审的分配和完成情况

实现：通过多表JOIN关联评审信息

v\_thesis\_review\_summary：评审意见汇总视图

功能：汇总评审分数和意见

实现：通过GROUP BY和聚合函数实现

(4) 答辩相关视图

v\_defense\_arrangement\_details：答辩安排详情视图

功能：显示答辩安排的完整信息

实现：通过多表JOIN关联答辩委员会信息

v\_defense\_score\_summary：答辩成绩汇总视图

功能：统计答辩成绩的平均分、最高分等

实现：通过GROUP BY和聚合函数实现

(5) 综合统计视图

v\_student\_comprehensive\_scores：学生综合成绩视图

功能：统计学生全过程的各项成绩

实现：通过复杂的CASE WHEN和权重计算实现

**三、触发器设计**

**1. 业务场景分析**

(1) 选题管理场景

 选题审批后自动创建任务书阶段

 任务书更新时同步更新进度记录等

(2) 进度管理场景

 阶段完成后自动创建下一阶段

 成绩录入时自动计算加权分数

 进度状态变更时自动更新时间戳

(3) 论文评审场景

 评审完成时自动更新论文状态

 需要修改时自动增加版本号

 同行评议完成时更新论文状态

(4) 答辩管理场景

 答辩评分时自动计算总分

 所有评委评分后自动更新答辩状态

**2. 触发器性能考虑**

已采用和执行的优化策略如下：

 减少触发器中的复杂查询

 避免触发器链式反应

 避免触发器循环更新

 集中在更新少但触发率高的表中

**3. 触发器实现方案**

文本

描述已自动生成(1) 选题相关触发器

(2) 进度管理触发器

文本

描述已自动生成  
文本

描述已自动生成

文本

描述已自动生成

(3) 论文评审触发器

文本

描述已自动生成  
(4) 答辩管理触发器

文本

描述已自动生成

**第三部分、数据库运行与维护**

**一、转储与恢复**

由于是课程设计的项目，并不是实际应用，此部分考虑不多，只在超管界面下添加了数据库手动备份与恢复。

通过mysqldump将当前数据库内容导出为本地.sql文件，并记录保存时间与版本，之后若要恢复到某个版本可以通过对应版本文件来恢复。

**二、安全性与完整性控制**

**1. 用户权限管理方案**

账号登录均存放在设计的users表中，并不是真实的数据库用户，因此权限控制通过前后端页面的身份验证和操作权限限制来严格保证不会越级操作。

比如，在用户管理界面中对用户的新增、编辑、删除功能添加了身份验证，系统管理员开放对除了系统管理员以下身份的权限，管理员（系主任）只能对管理员以下身份账户信息进行操作。

**2. 数据加密方案**

在users表存储的时候对密码进行加密，采用typescript的标准hash函数进行加密

文本

描述已自动生成

**3. 日志管理方案**

建立了system\_logs表来记录所有用户对数据库更新、修改、删除操作，部分关键查询操作也记录在内

文本

描述已自动生成

并采用统一的记录格式来区分不同操作

电脑屏幕截图

描述已自动生成

**三、性能的监督、分析与改进**

如第二部分所示，进行了索引、视图、触发器的实施

**四、重组织与重构造**

无

**第四部分——数据库实施**

**一、技术栈选择**

**1. 技术选型依据**

选择 MySQL 8.0 的主要原因:

 完善的事务支持,满足学生选题等并发操作

 JSON数据类型支持,适合存储灵活的日志信息

 窗口函数支持,便于分析进度数据

 活跃的社区支持和丰富的文档资源

 与 Node.js 良好的集成性

选择Node.js+Express 的原因:

 异步非阻塞I/O,适合高并发场景

 TypeScript 支持,提供类型安全

 丰富的npm生态系统

 前后端统一的JavaScript 技术栈

选择Vue3+TypeScript 的原因:

 组合式API提供更好的代码组织

 TypeScript带来类型安全

 响应式系统适合实时更新的场景

 完善的开发工具支持

**2. 开发环境搭建**

后端环境搭建：

文本

描述已自动生成

前端环境搭建：

文本

描述已自动生成

**二、前端页面设计**

**1. 用户界面设计**

**1.1 视觉层次**

电脑屏幕的照片上有文字

描述已自动生成文本

描述已自动生成

图形用户界面, 应用程序, Teams

描述已自动生成

**1.2 交互反馈**

文本

描述已自动生成

电脑屏幕截图

描述已自动生成

**2. 页面布局设计**

**2.1 整体布局**

文本

描述已自动生成

**3. 交互设计**

**3.1 导航交互**

文本

描述已自动生成

**3.2 表单交互**

电脑屏幕的照片上有文字

描述已自动生成

**4. 响应式设计**

**4.1 断点设计**

文本

描述已自动生成

**4.2 响应式布局**

文本

描述已自动生成

**三、后端对接数据库实现**

**1. 接口设计**

**1.1 RESTful API 规范**

文本

描述已自动生成

**1.2 统一响应格式**

文本

描述已自动生成

**2.数据访问层设计**

**2.1 数据库连接池优化**

文本

描述已自动生成

**2.2 DAO层实现**

文本

描述已自动生成

**3. 业务逻辑层设计**

**3.1 Service层实现**

文本

描述已自动生成

**4. 异常处理机制**

**4.1 自定义错误类型**

文本

描述已自动生成

**4.2 全局错误处理中间件**

文本

描述已自动生成

一些项目页面图片示例：  
图形用户界面, 应用程序, Word

描述已自动生成

图形用户界面, 应用程序

描述已自动生成

**第五部分——总结与反思**

**一、目前存在的问题**

**1. 系统功能方面**

 缺乏数据导出功能，不支持批量数据处理

 文件上传功能较为简单，缺乏预览和在线编辑功能

 消息通知系统不够完善，缺乏实时推送功能

 缺少移动端适配，用户体验有待提升

**2. 技术实现方面**

 部分代码重复度较高，需要进一步抽象和复用

 错误处理机制不够完善，日志记录不够详细

 缺乏完整的单元测试和集成测试

 数据库查询优化空间较大，部分查询效率较低

**3. 安全性方面**

 密码策略相对简单，缺乏复杂度要求

 敏感数据加密方式需要加强

 缺乏完整的操作审计日志

 XSS和SQL注入防护需要加强

**二、未来改进的方向**

**1. 功能扩展**

 增加数据分析和可视化功能

 开发移动端应用或响应式界面

 完善文件管理系统，支持在线预览和编辑

 引入实时通知系统，支持即时消息推送

**2. 技术优化**

 重构代码，提高代码复用率

 优化数据库设计和查询性能

 完善测试体系，提高代码覆盖率

 引入缓存机制，提升系统响应速度

**3. 安全加强**

 实现更严格的密码策略

 加强数据加密和传输安全

 完善权限控制和访问审计

 增加防攻击措施

**三、项目经验总结**

**1. 技术收获**

 掌握了前后端分离开发模式

 深入理解了Node.js和React的开发特点

 提升了数据库设计和优化能力

 积累了项目架构和性能优化经验

**2. 项目管理经验**

 学会了合理规划开发周期

 掌握了版本控制和代码管理

 理解了模块化设计的重要性

 提高了问题分析和解决能力

**3. 个人成长总结**

 提高了独立解决问题的能力

 增强了技术文档阅读和编写能力

 培养了持续学习的习惯

 积累了实际项目经验