数据库课程设计

题　　目　党的知识学习测试系统

专业班级　计算 2001

姓　　名　陈俊言

学　　号　2015040112

目录

一、 概述

随着高校学生中入党积极分子人数的不断增加，要求参加党校学习的人越来越多，党校的规模不断扩大，教学质量和教学效果难以保证。党校传统的教学方式已经无法满足新时期思想政治教育的需要。以因特网为代表的计算机网络技术已经在工作、学习和生活等各个领域普及应用，对人们的影响也日益加深，由此可见保证党课教学质量和教学效果，开发网上党校系统是大势所趋。

为了检测入党积极分子对党课的学习情况，并更好的统一对入党积极分子的管理与考核，笔者设计了党的知识学习测试系统用以优化积极分子管理。

设计过程主要是数据库的设计，包括需求分析、概念结构设计、逻辑结构设计、物理结构设计、相关表的建立以及数据库的实施。

该系统主要使用 JAVA、javascript、HTML 三种语言组成。前后端采用 vue + springboot 的框架进行设计，采用框架能有效增强代码的逻辑性、提升代码的可阅读性，能为后续维护提供极大的便利。

数据库使用 MariaDB 数据库，该数据库为 XAMPP 建站集成软件包中自带数据库，具有完美兼容 MySQL 数据库、并且开源的优点。

由于 MariaDB 数据库完美兼容 MySQL 数据库，因此笔者采用了 JDBC 接口连接数据库。

该系统主要由以下几个部分组成：

1. 登录模块：管理员、组织员、学员的区分与登录。

2. 超级管理员模块：学员管理功能：增删改查学员的信息、组织员管理功能：增删改查组织员的信息、题库管理功能：增删客观题与主观题题目、统计信息查看功能。

3. 组织员模块：学员管理功能、题库管理功能、统计信息查看功能、生成试卷，发布考核，主观题判卷，登记考核成绩。

4. 学员模块：自测功能、考核功能、查看考核情况。

党的知识学习测试系统能有效优化组织员与管理员对学员的考核与管理，为大规模积极分子的管理提供了思路。

二、 需求分析

（一） 数据字典

root（管理员）数据结构：

root=管理员账户+管理员密码+管理员类别

表 1-1 root

属性	数据类型	含义说明
root_acc	varchar(40)	管理员账户
root_pass	varchar(40)	管理员密码
type	int	账户类型
name	varchar(40)	管理员名

student（学员）数据结构：

student=学员 id+学员密码+学员名字+学员年级+学员学校+所属组织部

表 1-2 student

属性	数据类型	含义说明
stu_id	varchar(40)	学员 id
stu_pass	varchar(40)	学员密码
stu_name	varchar(40)	学员名字
stu_grade	varchar(40)	学员年级
stu_school	varchar(40)	学员学院
stu_orgamem	varchar(40)	所属组织部
type	int	账户类型

paper（试卷）数据结构：

paper=试卷编号+试卷名+创建人+创建时间+第 1 题编号+第 2 题编号+.....+第 25 题编号

表 1-3 paper

属性	数据类型	含义说明

id	int	试卷编号
name	varchar(40)	试卷名
creator	varchar(40)	创建人
createTime	varchar(40)	创建时间
one	int	第 n 题编号
two	int	第 n 题编号
three	int	第 n 题编号
four	int	第 n 题编号
five	int	第 n 题编号
six	int	第 n 题编号
seven	int	第 n 题编号
eight	int	第 n 题编号
nine	int	第 n 题编号
ten	int	第 n 题编号
eleven	int	第 n 题编号
twelve	int	第 n 题编号
thirteen	int	第 n 题编号
fourteen	int	第 n 题编号
fifteen	int	第 n 题编号
sixteen	int	第 n 题编号
seventeen	int	第 n 题编号
eighteen	int	第 n 题编号
nineteen	int	第 n 题编号
twenty	int	第 n 题编号
twentyone	int	第 n 题编号
twentytwo	int	第 n 题编号
twentythree	int	第 n 题编号

twentyfour	int	第 n 题编号
twentyfive	int	第 n 题编号

organizationer（组织员）数据结构：

organizationer=类别+用户 id+用户名

表 1-4 organizationer

属性	数据类型	含义说明
ormem_id	int	组织员 id
type	int	账户类型
ormem_acc	varchar(40)	组织员账号
ormem_pass	varchar(40)	组织员密码
ormem_name	varchar(40)	组织部名
school	varchar(40)	学院

school（学院）数据结构：

school=学院编号+学院名称

表 1-5 school

属性	数据类型	含义说明
school_id	int	学院编号
school_name	varchar(40)	学院名称

objective_problems（客观题）数据结构：

objective_problems=客观题编号+题干+选项 A+选项 B+选项 C+选项 D+答案

表 1-6 objective_problems

属性	数据类型	含义说明
objectpro_id	int	客观题编号
objectpro_com	varchar(200)	题干
objectpro_A	varchar(200)	选项 A

objectpro_B	varchar(200)	选项 B
objectpro_C	varchar(200)	选项 C
objectpro_D	varchar(200)	选项 D
objectpro_ans	varchar(200)	答案（只能是 ABCD）

subjective_problems（主观题）数据结构：

subjective_problems=类别+用户 id+用户名

表 1-7 subjective_problems

属性	数据类型	含义说明
subpro_id	int	主观题编号
subpro_com	varchar(400)	题干
subpro_ans	varchar(400)	参考答案

stupaper（学生试卷）数据结构：

stupaper=编号+学员编号+试卷编号

表 1-8 stupaper

属性	数据类型	含义说明
id	int	编号
stu	varchar(40)	学员编号
paper	int	试卷编号

stuanswer（学生答案）数据结构：

stuanswer=编号+学生编号+试卷编号+客观题分数+主观题答案+主观题分数+状态

表 1-9 stuanswer

属性	数据类型	含义说明
id	int	编号
stu_id	varchar(40)	学员编号
paper_id	int	试卷编号

objGrade	float	客观题分数
eryi	varchar(400)	二一题
erer	varchar(400)	二二题
ersan	varchar(400)	二三题
ersi	varchar(400)	二四题
erwu	varchar(400)	二五题
subGrade	float	主观题分数
status	int	状态

studentgrade（学生成绩）数据结构：

studentgrade=类别+用户 id+用户名

表 1-10 studentgrade

属性	数据类型	含义说明
id	int	编号
stu_id	varchar(40)	学员编号
CorrectedNum	int	正确题数
DisCorrectedNum	int	错误题数
questionMaster	int	谈话人编号
createTime	timestamp	创建时间

（二） 数据流

表 1-11.数据流表

数据流名	数据流来源	数据流去向	组成
添加学员	管理员、组织员	学员数量增加	root,orgnizationer,student
删除学员	管理员、组织员	学员数量减少	root,orgnizationer,student
修改学员	管理员、组织员	学员信息被修改	root,orgnizationer,student
添加组织员	管理员	组织员数量增加	root,orgnizationer

删除组织员	管理员	组织员数量减少	root,orgnizationer
修改组织员	管理员	组织员被修改	root,orgnizationer
添加客观题	管理员、组织员	客观题数量增加	root,organizationer,objective_problems
删除客观题	管理员、组织员	客观题数量减少	root,organizationer,objective_problems
修改客观题	管理员、组织员	客观题被修改	root,organizationer,objective_problems
添加主观题	管理员、组织员	主观题数量增加	root,organizationer,subjective_problems
删除主观题	管理员、组织员	主观题数量减少	root,organizationer,subjective_problems
修改主观题	管理员、组织员	主观题被修改	root,organizationer,subjective_problems
生成试卷	组织员	试卷数量增加	organizationer,paper
删除试卷	组织员	试卷数量减少	organizationer,paper
分配试卷	组织员	学员试卷增加	organizationer,stupaper
参加考核	学员	学员回答增加、学生试卷减少	student,stuanswer,stupaper
录入成绩	组织员	学员回答被修改、学生成绩增加	organizationer,stuanswer,studentgrade

（三） 数据存储

表 2.数据存储表

数据存储名	输入的数据流	输出的数据流	组成
管理员登记表	管理员信息	管理员信息	root
组织员登记表	组织员信息	组织员信息	organizationer
学员登记表	学员信息	学员信息	student
试卷登记表	试卷信息	试卷信息	paper
客观题登记表	客观题信息	客观题信息	objective_problems
主观题登记表	主观题信息	主观题信息	subjective_problems
学院登记表	学院信息	学院信息	school
学员试卷登记表	学员试卷信息	学员试卷信息	stupaper
学员回答登记表	学员回答信息	学员回答信息	stuanswer

学员成绩登记表	学员成绩信息	学员成绩信息	studengrade

（四） 处理过程

1. 添加学员

处理过程：添加学员

说明：管理员和组织员对学员进行添加

输入：学员 id、密码、姓名、年级、学院 id、组织员 id、专业

输出：是否添加成功

处理：管理员或组织员对学员进行添加，若学员已经存在，则无法添加。

2. 删除学员

处理过程：删除学员

说明：管理员和组织员对学员进行添加

输入：学员 id

输出：是否删除成功

处理：管理员或组织员对学员进行删除，若学员不存在，则无法删除。

3. 修改学员

处理过程：修改学员

说明：管理员和组织员对学员进行添加

输入：学员 id、密码、姓名、年级、学院 id、组织员 id、专业

输出：是否修改成功

处理：管理员或组织员对学员信息进行修改。

4. 添加组织员

处理过程：添加组织员

说明：管理员对组织员进行添加

输入：组织员 id、账号、密码、组织部名、学院 id

输出：是否添加成功

处理：管理员对组织员进行添加，若组织员已经存在，则无法添加。

5. 删除组织员

处理过程：删除组织员

说明：管理员对组织员进行添加

输入：组织员 id

输出：是否删除成功

处理：管理员对组织员进行删除，若组织员不存在，则无法删除。

6. 修改组织员

处理过程：修改组织员

说明：管理员对组织员进行添加

输入：组织员 id、账号、密码、组织部名、学院 id

输出：是否修改成功

处理：管理员对组织员信息进行修改。

7. 添加客观题

处理过程：添加客观题

说明：管理员或组织员对客观题进行添加

输入：客观题 id、题干、选项 A、选项 B、选项 C、选项 D、答案

输出：是否添加成功

处理：管理员或组织员对客观题进行添加，若客观题已经存在，则无法添加。

8. 删除客观题

处理过程：删除客观题

说明：管理员或组织员对客观题进行添加

输入：客观题 id

输出：是否删除成功

处理：管理员或组织员对客观题进行删除，若客观题不存在，则无法删除。

9. 修改客观题

处理过程：修改客观题

说明：管理员或组织员对客观题进行添加

输入：客观题 id、题干、选项 A、选项 B、选项 C、选项 D、答案

输出：是否修改成功

处理：管理员或组织员对客观题信息进行修改。

10.添加主观题

处理过程：添加主观题

说明：管理员或组织员对主观题进行添加

输入：主观题 id、题干、参考答案

输出：是否添加成功

处理：管理员或组织员对主观题进行添加，若主观题已经存在，则无法添加。

11.删除主观题

处理过程：删除主观题

说明：管理员或组织员对主观题进行添加

输入：主观题 id

输出：是否删除成功

处理：管理员或组织员对主观题进行删除，若主观题不存在，则无法删除。

12.修改主观题

处理过程：修改主观题

说明：管理员或组织员对主观题进行添加

输入：主观题 id、题干、参考答案

输出：是否修改成功

处理：管理员或组织员对主观题信息进行修改。

13.生成试卷

处理过程：生成试卷

说明：组织员对试卷的题目进行选择组合并生成试卷

输入：试卷 id、试卷名、生成者 id、客观第 n 题 id（1-20）、主观第 n 题 id（21-25）

输出：是否生成成功

处理：组织员对试卷进行生成。

14.删除试卷

处理过程：删除试卷

说明：组织员对试卷进行删除

输入：试卷 id

输出：是否删除成功

处理：组织员对试卷进行删除。

15.分配试卷

处理过程：分配试卷

说明：组织员对试卷进行分配

输入：试卷 id、学员 id

输出：学生试卷

处理：组织员通过试卷 id 与学员 id 将试卷分配给学员。

16.参加考核

处理过程：参加考核

说明：学员参加组织员的考核

输入：学员 id、试卷 id、客观题分数、主观第 n 题回答（21-25）

输出：学生试卷

处理:学员作答题目并提交试卷后前端自动判定客观题分数并提交考核信息、同时学生试卷中的对应行被删除。

17.录入成绩

处理过程：录入成绩

说明：组织员批改主观题并录入成绩

输入：学生回答 id

输出：学生总成绩与考核情况

处理：组织员批改主观题并录入成绩、同时在学生成绩表中生成学生总成绩与考核情况。

三、 概念结构设计

（一） E-R 图

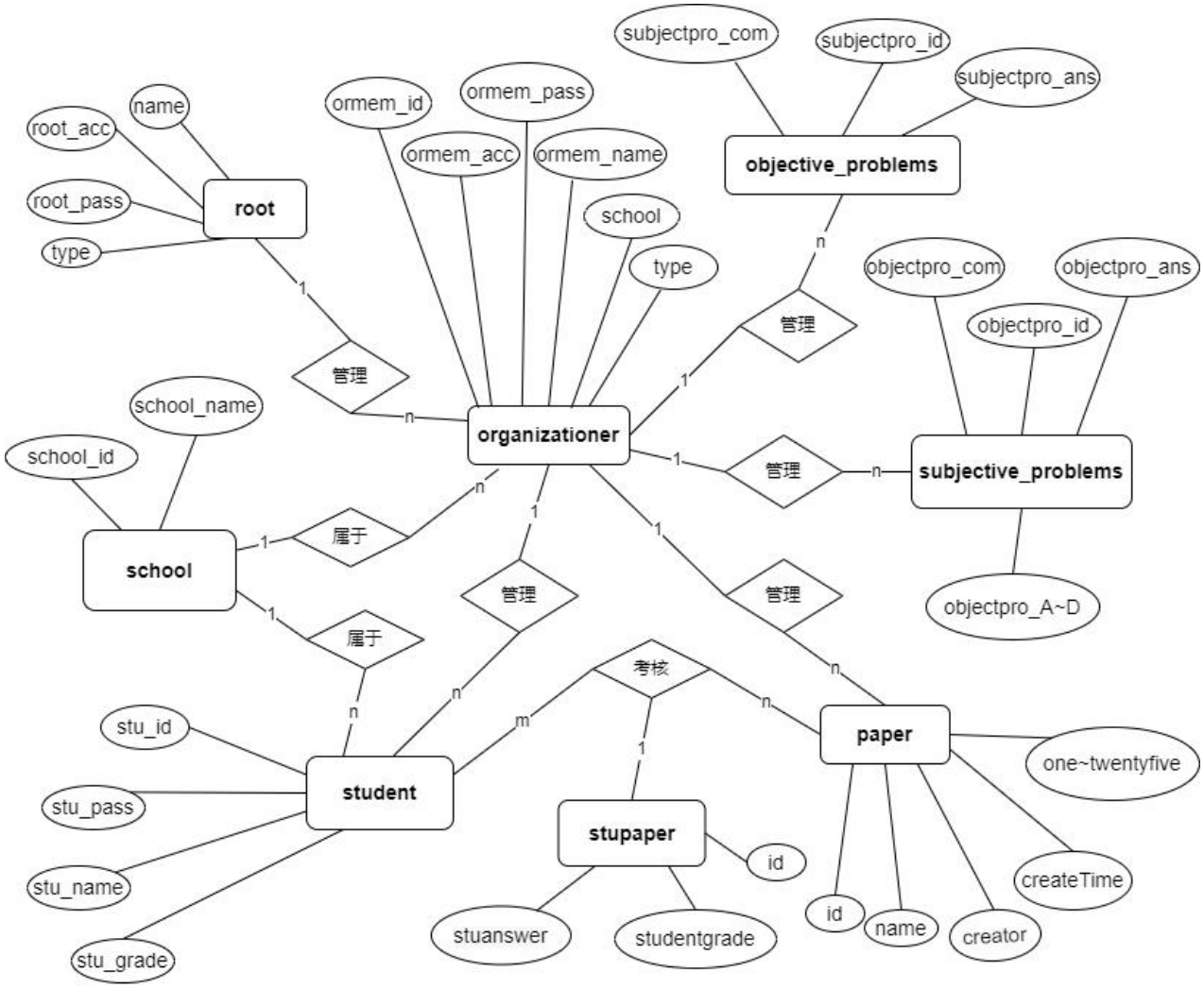

图 3-1. 系统的基本 E-R 图

（二） 系统说明书

1、系统要求

党的知识学习测试系统使用了 B/S 架构的系统，用户要求各功能使 用正常，系统响应比较快，运行稳定。本系统的用户有：普通教师， 可以查看修改个人信息，修改密码，提交项目申请，上传项目申请支 撑材料，修改项目信息；院管理员，可以设置本院教师申请和评审的 时间范围，可以查看所有本院教师申请的项目，给项目分配评审专家， 查看项目专家评审汇总，推荐项目到校级评审或者让对项目进行驳回 操作并给出驳回意见；校管理员，可以设置校内项目评

审的时间范围， 可以查看所有推荐到校的项目信息，为其分配专家，查看项目专家评 审汇总；DBA 管理员，设置每个人员的权限；专家，在规定时间内为 被分配的项目评审，下载支撑材料，填写评审意见。

系统主要功能如下：

1、题库维护

客观题，能够网上作答，并评判结果；主观题能够网上作答。

2、组织员考察

给考察对象出题，并记录答题的学生的名称、考核结果、志愿书编号。

3、管理员

管理员可以分配和查看帐号，修改用户密码，查看考核情况，查看各个学院登录及学习人数统计情况。

4、入党积极分子

入党积极分子可以登录这个学习系统，选择自动生成试卷进行自学自考。

5、功能包括：

１．试题库的维护；２．学员学习功能，可以自测；３．组织员给学员谈话并出考题，由学员现场网上答题，给出成绩．４．用户权限管理，一个组织部超级用户，１００多个组织员，很多学员（入党积极分子）；超级用户管理组织员，组织员管理学员．５．组织部要完成许多统计功能，比如：各单位学员的谈话考核情况，各单位学员的平时学习和自测情况，学习人数，学习效果．等等.

2、数据流图

(1)管理员模块数据流图

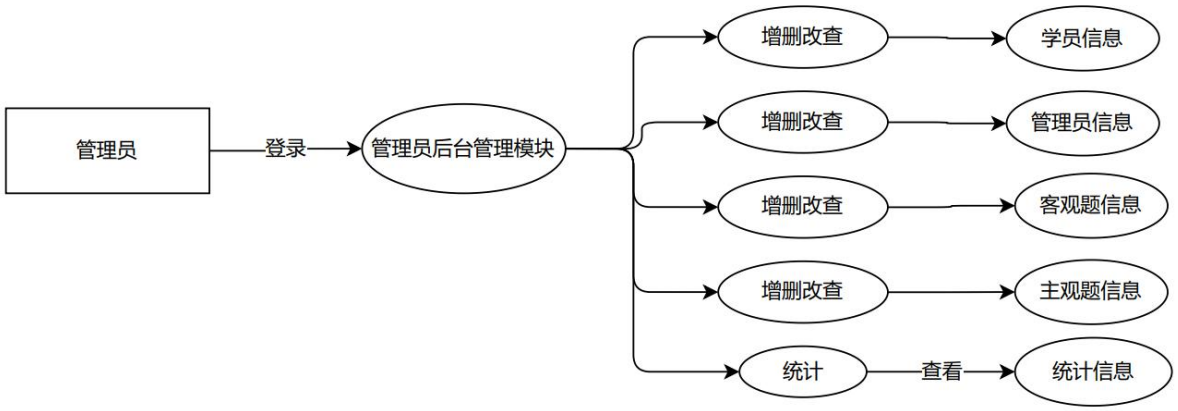

图 3-2.管理员模块数据流图

(2)组织员模块数据流图

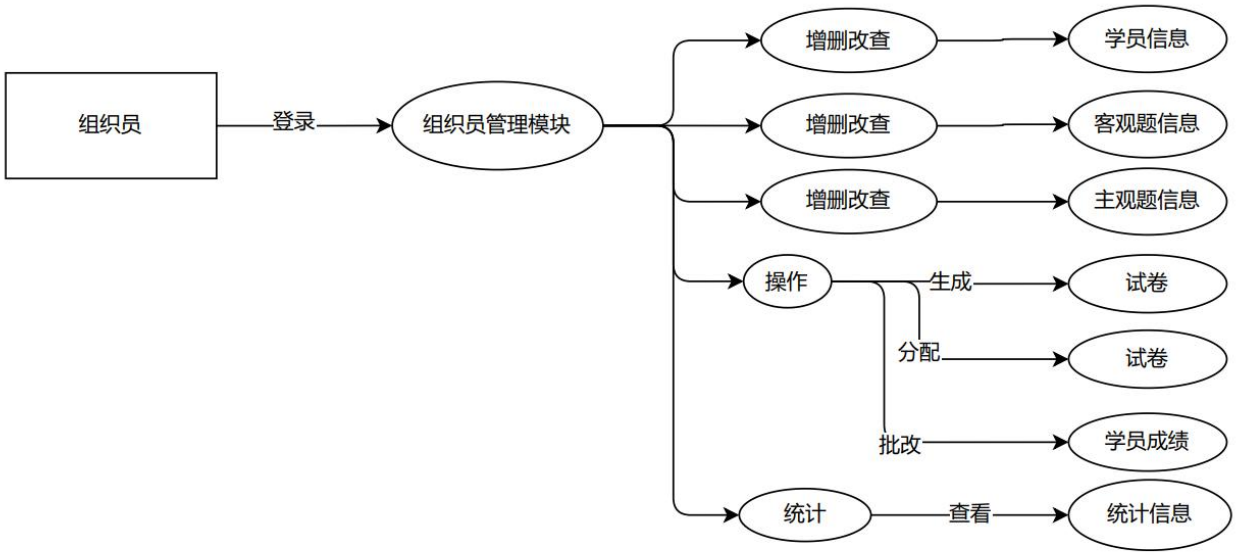

图 3-3.组织员模块数据流图

(3)学员模块数据流图

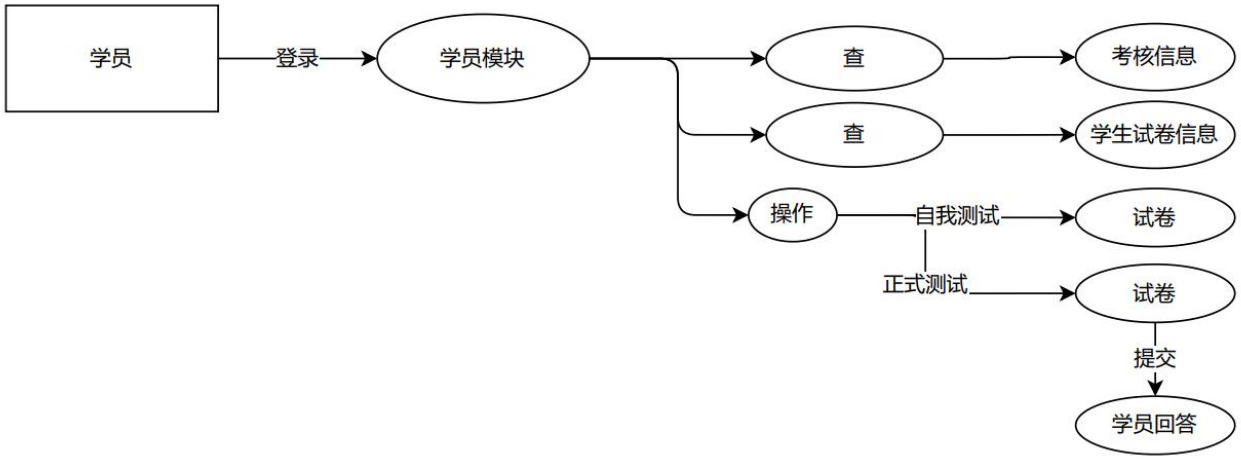

图 3-4. 学员模块数据流图

四、 逻辑结构设计

（一） 关系模式

根据 E-R 图向关系模型的转换原则，党的知识学习测试系统的 E-R 图可以转换为下列关系模式：

1. 管理员（账号，姓名，类型，名字）

2. 学员（学员 id，学员密码，学员名字，学员年级，学员学院，所属组织部，账户类型）

3. 试卷（试卷编号，试卷名，创建人，创建时间，第 n 题编号）

4. 组织员（组织员 id，账户类型，组织员账号，组织员密码，组织部名，学院 id）

5. 学院（学院编号，学院名称）

6. 客观题（客观题编号，题干，选项 A，选项 B，选项 C，选项 D，答案）

7. 主观题（主观题编号，题干，参考答案）

8. 学生试卷（编号，学员编号，试卷编号）

9. 学生答案（编号，学员编号，试卷编号，客观题分数，二一题，二二题，二三题，二四题，二五题，主观题分数，状态）

10. 学生成绩（编号，学员编号，正确题数，错误题数，谈话人编号，创建时间）

（二） 数据表的详细结构信息

root（管理员）表：

root=管理员账户+管理员密码+管理员类别

表 4-1 root

属性	数据类型	属性中文名	完整性约束
root_acc	varchar(40)	管理员账户	主码
root_pass	varchar(40)	管理员密码	
type	int	账户类型	默认 1
name	varchar(40)	管理员名	

student（学员）表：

student=学员 id+学员密码+学员名字+学员年级+学员学校+所属组织部

表 4-2 student

属性	数据类型	属性中文名	完整性约束
stu_id	varchar(40)	学员 id	主码
stu_pass	varchar(40)	学员密码	
stu_name	varchar(40)	学员名字	
stu_grade	varchar(40)	学员年级	
stu_school	varchar(40)	学员学院	外码
stu_orgamem	varchar(40)	所属组织部	外码
type	int	账户类型	默认 3

paper（试卷）表：

paper=试卷编号+试卷名+创建人+创建时间+第 1 题编号+第 2 题编号+.....+第 25 题编号

表 4-3 paper

属性	数据类型	属性中文名	完整性约束
id	int	试卷编号	主码

name	varchar(40)	试卷名	
creator	varchar(40)	创建人	外码
createTime	varchar(40)	创建时间	默认当前时间
one	int	第 n 题编号	外码
two	int	第 n 题编号	外码
three	int	第 n 题编号	外码
four	int	第 n 题编号	外码
five	int	第 n 题编号	外码
six	int	第 n 题编号	外码
seven	int	第 n 题编号	外码
eight	int	第 n 题编号	外码
nine	int	第 n 题编号	外码
ten	int	第 n 题编号	外码
eleven	int	第 n 题编号	外码
twelve	int	第 n 题编号	外码
thirteen	int	第 n 题编号	外码
fourteen	int	第 n 题编号	外码
fifteen	int	第 n 题编号	外码
sixteen	int	第 n 题编号	外码
seventeen	int	第 n 题编号	外码
eighteen	int	第 n 题编号	外码
nineteen	int	第 n 题编号	外码
twenty	int	第 n 题编号	外码
twentyone	int	第 n 题编号	外码
twentytwo	int	第 n 题编号	外码
twentythree	int	第 n 题编号	外码
twentyfour	int	第 n 题编号	外码

twentyfive	int	第 n 题编号	外码

organizationer（组织员）表：

organizationer=类别+用户 id+用户名

表 4-4 organizationer

属性	数据类型	属性中文名	完整性约束
ormem_id	int	组织员 id	主码
type	int	账户类型	默认 2
ormem_acc	varchar(40)	组织员账号	
ormem_pass	varchar(40)	组织员密码	
ormem_name	varchar(40)	组织部名	
school	varchar(40)	学院 id	外码

school（学院）表：

school=学院编号+学院名称

表 4-5 school

属性	数据类型	属性中文名	完整性约束
school_id	int	学院编号	主码
school_name	varchar(40)	学院名称	

objective_problems（客观题）表：

objective_problems=客观题编号+题干+选项 A+选项 B+选项 C+选项 D+答案

表 4-6 objective_problems

属性	数据类型	属性中文名	完整性约束
objectpro_id	int	客观题编号	主码
objectpro_com	varchar(200)	题干	
objectpro_A	varchar(200)	选项 A	
objectpro_B	varchar(200)	选项 B	

objectpro_C	varchar(200)	选项 C	
objectpro_D	varchar(200)	选项 D	
objectpro_ans	varchar(200)	答案	‘A’ ‘B’ ‘C’ ‘D’

subjective_problems（主观题）表：

subjective_problems=类别+用户 id+用户名

表 4-7 subjective_problems

属性	数据类型	属性中文名	完整性约束
subpro_id	int	主观题编号	主码
subpro_com	varchar(400)	题干	
subpro_ans	varchar(400)	参考答案	

stupaper（学生试卷）表：

stupaper=编号+学员编号+试卷编号

表 4-8 stupaper

属性	数据类型	属性中文名	完整性约束
id	int	编号	主码
stu	varchar(40)	学员编号	外码
paper	int	试卷编号	外码

stuanswer（学生答案）表：

stuanswer=编号+学生编号+试卷编号+客观题分数+主观题答案+主观题分数+状态

表 4-9 stuanswer

属性	数据类型	属性中文名	完整性约束
id	int	编号	主码
stu_id	varchar(40)	学员编号	外码
paper_id	int	试卷编号	外码
objGrade	float	客观题分数	

eryi	varchar(400)	二一题	
erer	varchar(400)	二二题	
ersan	varchar(400)	二三题	
ersi	varchar(400)	二四题	
erwu	varchar(400)	二五题	
subGrade	float	主观题分数	
status	int	状态	0、1

studentgrade（学生成绩）表：

studentgrade=类别+用户 id+用户名

表 4-10 studentgrade

属性	数据类型	属性中文名	完整性约束
id	int	编号	主码
stu_id	varchar(40)	学员编号	外码
CorrectedNum	int	正确题数	
DisCorrectedNum	int	错误题数	
questionMaster	int	谈话人编号	外码
createTime	timestamp	创建时间	默认当前时间

（三） 系统结构图

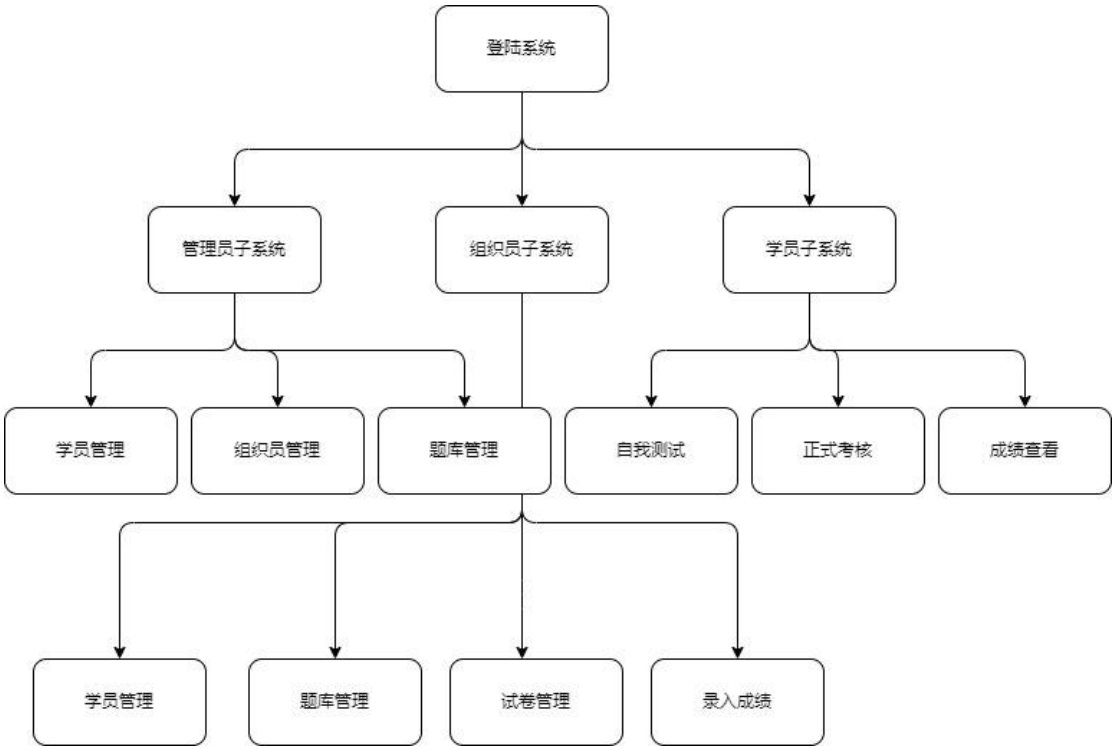

图 4-1. 系统结构图

五、 物理设计

（一） 存储安排

确定数据库物理结构的内容要确定数据的存放位置和存储结构（关系、索引、聚簇、日志、备份）和确定系统配置。影响数据存放位置和存储结构的因素既包括硬件环境也包括应用需求（存取时间、 存储空间利用率、维护代价）。由于此系统规模较小，所以可以将此系统放于同一磁盘上进行使用。

1、存取方法选择

DBMS 常用存取方法有，索引方法，目前主要是 B+树索引方法、聚簇（Cluster）方法和 HASH 方法。本系统对数据库物理设计的要求不高，采用 MariaDB 默认分配的物理空间即可满足实践要求，本系统对物理设计并未采取特别设计。

2、存取路径建立

数据库数据备份、日志文件备份等由于只在故障恢复时才使用，而且数据量很大，可以考虑与数据库对象（表、索引等）放在不同的磁盘以改进系统的性能。

（二） 模块设计（IPO 表）

IPO 表：提交考核
输入：学员 id、试卷 id、客观题分数、主观第 n 题回答（21-25）
输入：学生试卷
处理：学员作答题目并提交试卷后前端自动判定客观题分数并提交考核信息、同时学生试卷中的对应行被删除。

IPO 表：录入成绩
输入：学生回答,id
输入：学生总成绩与考核情况
处理：组织员批改主观题并录入成绩、同时在学生成绩表中生成学生总成绩与考核情况。

IPO 表：分配试卷
输入：试卷 id、学员 id
输入：学生试卷
处理：组织员对试卷进行分配，组织员通过试卷 id 与学员 id 将试卷分配给学员。

六、 数据库实施

（一） 创建关系模式

root（管理员）表：

root=管理员账户+管理员密码+管理员类别

```
create table root
(
    root_acc  varchar(40)   not null
        primary key,
    root_pass varchar(40)   null,
    type      int default 1 null,
    name      varchar(40)   null
);
```

student（学员）表：

student=学员 id+学员密码+学员名字+学员年级+学员学校+所属组织部

```
create table student
(
    stu_id      varchar(40)   not null
        primary key,
    stu_pass    varchar(40)   null,
    stu_name    varchar(40)   null,
    stu_grade   varchar(40)   null,
    stu_school  int           null,
    stu_orgamem int           null,
    major       varchar(40)   null,
    type        int default 3 null,
    constraint student_school_school_id_fk
        foreign key (stu_school) references school (school_id)
);
```

paper（试卷）表：

paper=试卷编号+试卷名+创建人+创建时间+第 1 题编号+第 2 题编号+.....+第 25 题编号

```
create table paper
(
    id          int auto_increment
        primary key,
    name        varchar(40)                          null,
    creator     int                                  null,
    createTime  datetime default current_timestamp() null,
    one         int                                  null,
    two         int                                  null,
    three       int                                  null,
    four        int                                  null,
    five        int                                  null,
    six         int                                  null,
    seven       int                                  null,
```

```
eight       int                                 null,
nine        int                                 null,
ten         int                                 null,
eleven      int                                 null,
twelve      int                                 null,
thirteen    int                                 null,
fourteen    int                                 null,
fifteen     int                                 null,
sixteen     int                                 null,
seventeen   int                                 null,
eighteen    int                                 null,
nineteen    int                                 null,
twenty      int                                 null,
twentyone   int                                 null,
twentytwo   int                                 null,
twentythree int                                 null,
twentyfour  int                                 null,
twentyfive  int                                 null,
constraint paper_objectproblem_objectpro_id_fk
    foreign key (one) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_10
    foreign key (ten) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_11
    foreign key (eleven) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_12
    foreign key (twelve) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_13
    foreign key (thirteen) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_14
    foreign key (fourteen) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_15
    foreign key (fifteen) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_16
    foreign key (sixteen) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_17
    foreign key (seventeen) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_18
    foreign key (eighteen) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_19
    foreign key (nineteen) references objective_problems (objectpro_id),
constraint paper_objectproblem_objectpro_id_fk_2
    foreign key (two) references objective_problems (objectpro_id),
```

```
    constraint paper_objectproblem_objectpro_id_fk_20
        foreign key (twenty) references objective_problems (objectpro_id),
    constraint paper_objectproblem_objectpro_id_fk_3
        foreign key (three) references objective_problems (objectpro_id),
    constraint paper_objectproblem_objectpro_id_fk_4
        foreign key (four) references objective_problems (objectpro_id),
    constraint paper_objectproblem_objectpro_id_fk_5
        foreign key (five) references objective_problems (objectpro_id),
    constraint paper_objectproblem_objectpro_id_fk_6
        foreign key (six) references objective_problems (objectpro_id),
    constraint paper_objectproblem_objectpro_id_fk_7
        foreign key (seven) references objective_problems (objectpro_id),
    constraint paper_objectproblem_objectpro_id_fk_8
        foreign key (eight) references objective_problems (objectpro_id),
    constraint paper_objectproblem_objectpro_id_fk_9
        foreign key (nine) references objective_problems (objectpro_id),
    constraint paper_subjectproblem_subjectpro_id_fk_21
        foreign key (twentyone) references subjective_problems (subpro_id),
    constraint paper_subjectproblem_subjectpro_id_fk_22
        foreign key (twentytwo) references subjective_problems (subpro_id),
    constraint paper_subjectproblem_subjectpro_id_fk_23
        foreign key (twentythree) references subjective_problems (subpro_id),
    constraint paper_subjectproblem_subjectpro_id_fk_24
        foreign key (twentyfour) references subjective_problems (subpro_id),
    constraint paper_subjectproblem_subjectpro_id_fk_25
        foreign key (twentyfive) references subjective_problems (subpro_id)
);
```

organizationer（组织员）表：

organizationer=类别+用户 id+用户名

```
create table organizationer
(
    ormem_id   int auto_increment
        primary key,
    type       int default 2 null,
    ormem_acc  varchar(40)   null,
    ormem_pass varchar(40)   null,
    ormem_name varchar(40)   null,
    school     int           null,
    constraint organizationer_ormem_acc_uindex
        unique (ormem_acc),
    constraint organizationer_school_school_id_fk
```

```
        foreign key (ormem_id) references school (school_id),
    constraint organizationer_school_school_id_fk_2
        foreign key (school) references school (school_id)
);
```

school（学院）表：

school=学院编号+学院名称

```
create table school
(
    school_id   int auto_increment
        primary key,
    school_name varchar(40) null
)
    comment '学院';
```

objective_problems（客观题）表：

objective_problems=客观题编号+题干+选项 A+选项 B+选项 C+选项 D+答案

```
create table objective_problems
(
    objectpro_id  int auto_increment
        primary key,
    objectpro_com varchar(200) null,
    objectpro_A   varchar(200) null,
    objectpro_B   varchar(200) null,
    objectpro_C   varchar(200) null,
    objectpro_D   varchar(200) null,
    objectpro_ans varchar(200) null,
    constraint ans
        check (`objectpro_ans` = 'A' or `objectpro_ans` = 'B' or `objectpro_ans` = 'C' or
`objectpro_ans` = 'D')
);
```

subjective_problems（主观题）表：

subjective_problems=类别+用户 id+用户名

```
create table subjective_problems
(
    subpro_id  int auto_increment
        primary key,
    subpro_com varchar(400) null,
    subpro_ans varchar(400) null
);
```

stupaper（学生试卷）表：

stupaper=编号+学员编号+试卷编号

```
create table stupaper
(
    id    int auto_increment
        primary key,
    stu   varchar(40) null,
    paper int         null,
    constraint stuPaper_student_stu_id_fk
        foreign key (stu) references student (stu_id),
    constraint stupaper_paper_id_fk
        foreign key (paper) references paper (id)
);
```

stuanswer（学生答案）表：

stuanswer=编号+学生编号+试卷编号+客观题分数+主观题答案+主观题分数+状态

```
create table stuanswer
(
    id       int auto_increment
        primary key,
    stu_id   varchar(40)   null,
    paper_id int           null,
    objGrade float         null,
    eryi     varchar(400)  null,
    erer     varchar(400)  null,
    ersan    varchar(400)  null,
    ersi     varchar(400)  null,
    erwu     varchar(400)  null,
    subGrade float         null,
    status   int default 0 null,
    constraint stuanswer_paper_id_fk
        foreign key (paper_id) references paper (id),
    constraint stuanswer_student_stu_id_fk
        foreign key (stu_id) references student (stu_id),
    constraint type
        check (`status` = 0 or `status` = 1)
);
```

studentgrade（学生成绩）表：

studentgrade=类别+用户 id+用户名

```
create table studentgrade
(
    id              int auto_increment
        primary key,
    stu_id          varchar(40)                          null,
    CorrectedNum    int                                  null,
    DisCorrectedNum int                                  null,
    questionMaster  int                                  null,
    createTime      timestamp default current_timestamp() null,
    constraint grade_student_stu_id_fk
        foreign key (stu_id) references student (stu_id),
    constraint studentgrade_organizationer_ormem_id_fk
        foreign key (questionMaster) references organizationer (ormem_id)
);
```

（二） 程序代码

（1）统计已经考核的人数（通过 count 统计与 group by 分类在数据库中选取已经考核过的学生人数统计后将数据规范化输出）

```
@CrossOrigin
@RequestMapping(value = "api/getKaoheRenshu", method = RequestMethod.GET)
public Object getKaoheRenshu() {
    List<ResultData> list = studentServer.getKaoheRenshu();
    JSONArray jsonArray = new JSONArray();
    JSONObject jsonObject = new JSONObject();
    class Datatype {
        String name;
        int value;
        public String getName() {
            return name;
        }
        public int getValue() {
            return value;
        }
        public void setName(String name) {
            this.name = name;
        }
        public void setValue(int data) {
            this.value = data;
        }
    }
    for (ResultData resultData : list) {
```

```
            String s = schoolMapper.getNameById(resultData.getType());
            Datatype data = new Datatype();
            data.setName(s);
            data.setValue(resultData.getData());
            jsonArray.add(data);
        }
        jsonObject.put("data", jsonArray);
        return jsonObject;
    }
```

```
<resultMap id="getMeigeSchoolRenshuMapper" type="com.cjy.party.domain.ResultData">
    <id property="type" column="type"/>
    <result property="data" column="data"/>
</resultMap>
<select id="getKaoheRenshu" resultMap="getMeigeSchoolRenshuMapper">
    select stu_school as type, COUNT(*) as data
    from student,studentgrade
    where student.stu_id = studentgrade.stu_id
    group by stu_school;
</select>
```

（2）获取考核成绩（通过 session 获取学员 id，然后在数据库中查询 studentgrade 表并进行成绩计算等处理）

```
@CrossOrigin
@RequestMapping(value = "api/getHistory", method = RequestMethod.GET)
public Object getHistory(HttpServletRequest request) {
    JSONObject jsonObject = new JSONObject();
    JSONArray jsonArray = new JSONArray();
    String s = request.getParameter("id");
    Student student = studentServer.getStuByStuID(s);
    @JsonIgnoreProperties(value = {"hibernateLazyInitializer"})
    class Data {
        String name;
        int id;
        int CorrectedNum;
        int DisCorrectedNum;
        int grade;
        String questionMaster;
        String createTime;
        public Data(String name, int id, int correctedNum, int disCorrectedNum, int grade,
String createTime, String questionMaster) {
            this.name = name;
```

```
        this.id = id;
        this.CorrectedNum = correctedNum;
        this.DisCorrectedNum = disCorrectedNum;
        this.grade = grade;
        this.createTime = createTime;
        this.questionMaster = questionMaster;
    }
    public String getQuestionMaster() {
        return questionMaster;
    }
    public String getName() {
        return name;
    }
    public int getId() {
        return id;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getCorrectedNum() {
        return CorrectedNum;
    }
    public void setCorrectedNum(int correctedNum) {
        CorrectedNum = correctedNum;
    }
    public void setId(int id) {
        this.id = id;
    }
    public int getDisCorrectedNum() {
        return DisCorrectedNum;
    }
    public void setDisCorrectedNum(int disCorrectedNum) {
        DisCorrectedNum = disCorrectedNum;
    }
    public int getGrade() {
        return grade;
    }
    public void setGrade(int grade) {
        this.grade = grade;
    }
    public String getCreateTime() {
        return createTime;
```

```
        }
        public void setCreateTime(String createTime) {
            this.createTime = createTime;
        }
    }
    for (int i = 0; i < student.getGrades().size(); i++) {
        Data data = new Data(student.getStu_name(),
                student.getGrades().get(i).getId(),
student.getGrades().get(i).getCorrectedNum(),
                student.getGrades().get(i).getDisCorrectedNum(),
                student.getGrades().get(i).getCorrectedNum() * 4,
                student.getGrades().get(i).getCreateTime().toString(),
                student.getGrades().get(i).getQuestionMaster());
        jsonArray.add(data);
    }
    jsonObject.put("history", jsonArray);
    return jsonObject;
}
```

（三） 测试

（1）登录界面

图 6-1. 登录界面测试图

（2）学员管理界面

图 6-2. 学员管理界面测试图

（3）数据统计界面

图 6-3. 数据统计界面测试图

（4）考核界面

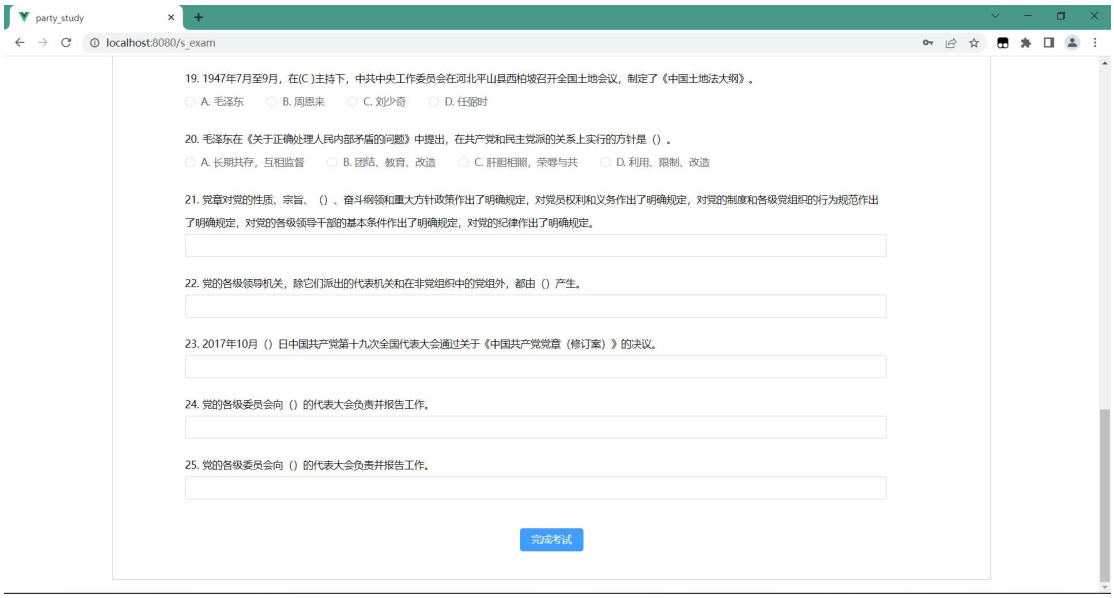

图 6-4. 考核界面测试图

七、 系统设计相关代码

（一） 存储过程的代码

（1）添加成绩的存储过程

```
create procedure addgrade(IN id varchar(40), IN Correct int, IN DisCorrect int, IN Master
int)
begin
    insert into studentGrade(stu_id, CorrectedNum, DisCorrectedNum, questionMaster)
        values (id, Correct, DisCorrect, Master);
end;
```

（2）计算成绩的储存过程

```
create
    definer = root@localhost procedure calgrade(IN obj int, IN sub int, OUT correct int,
OUT discorrect int)
begin
    set correct = (obj + sub) / 4;
    set discorrect = 25 - correct;
end;
```

（二） 触发器的代码

（1）录入成绩触发器（在管理员批改了试卷之后触发器将成绩录入进 studentgrade 表中，调用了 calgrade、findmaster、addgrade 三个储存过程来

计算成绩、查找谈话员、录入成绩）

```
DELIMITER $$
create trigger addgradeToStu
    after update
    on stuanswer
    for each row
    if NEW.status = 1 and OLD.status = 0 then
        call calgrade( obj: NEW.objGrade, sub: NEW.subGrade, correct: @correct, discorrect: @discorrect);
        call findmaster( id: NEW.stu_id, stu_orgamem: @id);
        call addgrade( a: NEW.stu_id, b: @correct, c: @discorrect, d: @id);
    end if;
$$
DELIMITER ;
```

（2）删除考核触发器（学生提交考核后，stupaper 表对应的已经分配的试卷应当被删除）

```
create trigger deleteStuPaper
    after insert on stuanswer
    for each row
    delete from stupaper where stu=NEW.stu_id and paper=NEW.paper_id;
```

八、 感想及心得体会

经过这一次数据库课程设计以后，我深刻的体会到设计和制作一套完备的系统是多么艰难。

这次的课程设计将我原本不太融汇贯通的理论知识重新整合起来，使我从更多角度体会到了数据库中每一个概念的意义。书本上的知识终究过于浅薄，本以为自己理论知识已经达到要求，可当我实战起来才发现自己举步维艰，只能在网上不停的搜索教程、网课等等，一步一步跟着去学，去体会每一行代码蕴含的逻辑，最终自己一步一个脚印地完成整个系统的开发。

由于我的对框架并不熟悉以及经验上的欠缺，在前期的概念设计中还是有许多不足的地方需要改进，这就需要我更多地尝试设计一个完备的系统，才能考虑得更加全面。

总之，我大体完成了系统要求的所有功能，并尝试了一些创新，这次的、课程设计的经历对我而言十分难得。相信在我以后的学习中，我也会秉持这种坚持学习的态度去面对更多的更难的问题。