《数据库原理与应用》

报 告

开课学期： 年 学期

学 院： 学院

班 级： 班

组 别：

组长学号：

组长姓名：

分组成员名单及分工

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 学号 | 姓名 | 承担的主要工作 | 贡献值  （0——1） |
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

学生信息管理系统

# 1．前言

## 1.1 课题描述

编写一个“学生信息管理系统”软件，并实现以下功能：

学生个人信息录入功能

学生课程选修管理功能

学生成绩录入与查询功能

学生出勤记录管理功能

在管理学生信息的过程中，要求对学生表、课程表、成绩表进行实时维护。并提供基础数据的维护功能：如学生信息管理、课程信息管理、教师信息管理等。

## 1.2 课题背景及意义

随着教育信息化进程的推进，学生信息管理系统作为学校日常管理的重要工具，能够实现学生个人信息、课程选修、成绩录入与查询、出勤记录等方面的管理，减轻教师和学校管理人员的工作负担，提高信息管理效率。本系统不仅可以提供基础数据的维护功能，还能够帮助教师及时掌握学生的学习状态和发展趋势，为学校决策提供数据支持。

## 1.3 相关技术介绍

对所选用的DBMS进行简介：

MySQL：

MySQL是一个关系型数据库管理系统，由瑞典MySQL AB 公司开发，目前属于 Oracle 旗下产品。是最流行的关系型数据库管理系统之一，在 WEB 应用方面，它是最好的 RDBMS (Relational Database Management System，关系数据库管理系统) 应用软件之一。MySQL是一种关系数据库管理系统，关系数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。MySQL所使用的 SQL 语言是用于访问数据库的最常用标准化语言。MySQL 软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库。

# 2. 需求分析

## 2.1 组织结构分析

经调查，该学生管理系统组织结构如图2-1所示：

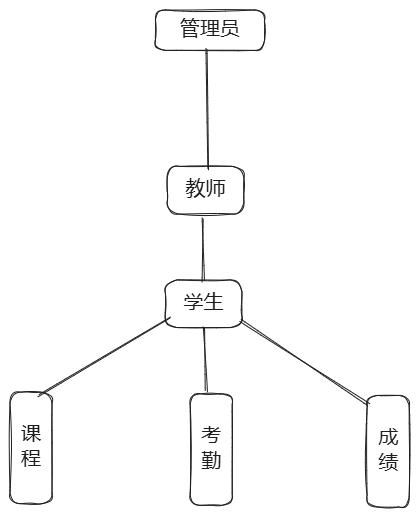


图2 \_1 系统组织结构图

## 2.2 系统功能分析

1. 学生信息管理： 学生信息首先由管理员录入，之后可以通过查询功能进行查看。学生信息可以随时更新或删除。
2. 课程选修管理： 学生可以在选修课程时进行选择，记录学生的选课信息。管理员也可以通过此模块进行课程的管理和查询。
3. 成绩管理： 教师或管理员输入学生的课程成绩，学生成绩可随时查询。
4. 出勤记录管理： 教师或管理员可以根据上课情况录入学生的出勤信息，学生的出勤情况可以查询。
5. 用户权限管理： 管理员可以配置系统的权限、账户和其他管理设置，确保系统的安全性与可控性。

## 2.3 系统性能分析

系统吞吐量：

* 预估在标准配置下（单服务器环境，处理能力一般），每秒能够处理 **50-100 TPS**。在高并发情况下，合理优化数据库、使用缓存、分布式架构等措施可以提高吞吐量至 **500 TPS** 甚至更高。

响应时间：

1. 对于单一查询（例如，查询单个学生的信息或成绩），响应时间应控制在 100-300 毫秒 之间。
2. 对于较复杂的查询（如成绩排名、学生出勤统计等），响应时间应保持在 1-2 秒 以内。
3. 在高并发情况下（例如，1000名学生同时查询成绩），响应时间可能会增加至 2-5 秒，但通过优化可以缩短。

存储容量：

基本存储需求：

* 学生基本信息（如姓名、性别、出生日期、联系方式等），每个学生约占 2-5 KB。
* 成绩记录、选课记录、出勤记录等表格数据，每个学生大约占用 5-10 KB。
* 假设一个学校有 10,000 名学生，每个学生的基本信息、成绩和出勤记录等数据总计大约为 100 MB - 200 MB（不包括备份数据）。

长期存储需求：

* 随着时间的推移，数据量的增加，存储需求会逐渐增加。例如，如果学校增加到 100,000 名学生，存储需求将接近 1 GB - 2 GB，此时还需要考虑长期数据的备份、日志记录等。

## 2.4 数据流程分析



## 2.5 数据字典

**1. 学生信息表 (students)**

| **字段名** | **数据类型** | **描述** | **备注** |
| --- | --- | --- | --- |
| student\_id | INT | 学生ID（主键） | 自增长 |
| name | VARCHAR(50) | 学生姓名 |  |
| gender | CHAR(1) | 性别 | M: 男, F: 女 |
| birthdate | DATE | 出生日期 |  |
| phone | VARCHAR(15) | 联系电话 | 可为空 |
| email | VARCHAR(50) | 邮箱 | 可为空 |
| address | VARCHAR(100) | 地址 | 可为空 |
| enrollment\_date | DATE | 入学日期 |  |
| major | VARCHAR(50) | 专业名称 |  |
| status | CHAR(1) | 学生状态 | A: 在读, G: 毕业, D: 休学 |

**2. 课程信息表 (courses)**

| **字段名** | **数据类型** | **描述** | **备注** |
| --- | --- | --- | --- |
| course\_id | INT | 课程ID（主键） | 自增长 |
| course\_name | VARCHAR(100) | 课程名称 |  |
| course\_code | VARCHAR(20) | 课程代码 | 唯一 |
| course\_credit | INT | 课程学分 |  |
| course\_teacher | VARCHAR(50) | 授课教师 |  |
| semester | VARCHAR(20) | 学期 |  |
| department | VARCHAR(50) | 所属学院 |  |

**3. 选课记录表 (course\_enrollment)**

| **字段名** | **数据类型** | **描述** | **备注** |
| --- | --- | --- | --- |
| enrollment\_id | INT | 选课记录ID（主键） | 自增长 |
| student\_id | INT | 学生ID（外键） | 关联students表 |
| course\_id | INT | 课程ID（外键） | 关联courses表 |
| enrollment\_date | DATE | 选课日期 |  |
| grade | DECIMAL(5,2) | 成绩 | 可为空（未评分） |
| status | CHAR(1) | 选课状态 | P: 已选, D: 已退 |

**4. 成绩表 (grades)**

| **字段名** | **数据类型** | **描述** | **备注** |
| --- | --- | --- | --- |
| grade\_id | INT | 成绩ID（主键） | 自增长 |
| student\_id | INT | 学生ID（外键） | 关联students表 |
| course\_id | INT | 课程ID（外键） | 关联courses表 |
| score | DECIMAL(5,2) | 课程成绩 |  |
| grade\_date | DATE | 成绩录入日期 |  |

**5. 出勤记录表 (attendance)**

| **字段名** | **数据类型** | **描述** | **备注** |
| --- | --- | --- | --- |
| attendance\_id | INT | 出勤记录ID（主键） | 自增长 |
| student\_id | INT | 学生ID（外键） | 关联students表 |
| course\_id | INT | 课程ID（外键） | 关联courses表 |
| date | DATE | 出勤日期 |  |
| status | CHAR(1) | 出勤状态 | P: 出席, A: 缺席, L: 迟到 |

**6. 教师信息表 (teachers)**

| **字段名** | **数据类型** | **描述** | **备注** |
| --- | --- | --- | --- |
| teacher\_id | INT | 教师ID（主键） | 自增长 |
| name | VARCHAR(50) | 教师姓名 |  |
| gender | CHAR(1) | 性别 | M: 男, F: 女 |
| phone | VARCHAR(15) | 联系电话 | 可为空 |
| email | VARCHAR(50) | 邮箱 | 可为空 |
| department | VARCHAR(50) | 所属学院 |  |
| courses\_taught | TEXT | 所授课程 | 用逗号分隔课程ID |

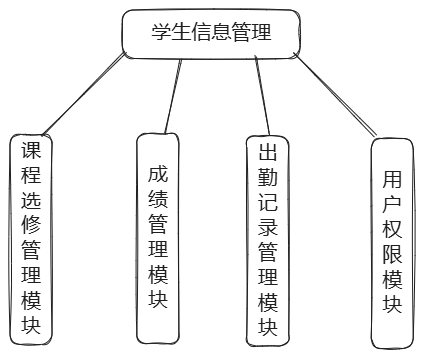
**7. 学生课程成绩汇总表 (student\_course\_summary)**

| **字段名** | **数据类型** | **描述** | **备注** |
| --- | --- | --- | --- |
| student\_id | INT | 学生ID（主键） | 关联students表 |
| total\_courses | INT | 总选修课程数 |  |
| total\_credits | INT | 总学分 |  |
| average\_grade | DECIMAL(5,2) | 平均成绩 |  |
| gpa | DECIMAL(5,2) | GPA |  |

# 3. 系统设计

## 3.1 功能模块设计

功能模块设计示例如图：



## 3.2 数据库设计

### 3.2.1 概念设计

ER图如下：



### 3.2.2 逻辑设计

students(student\_id, name, gender, birthdate, phone, email, address, enrollment\_date, major, status)

courses(course\_id, course\_name, course\_code, course\_credit, course\_teacher, semester, department)

course\_enrollment(enrollment\_id, student\_id, course\_id, enrollment\_date, grade, status)

grades(grade\_id, student\_id, course\_id, score, grade\_date)

attendance(attendance\_id, student\_id, course\_id, date, status)

teachers(teacher\_id, name, gender, phone, email, department, courses\_taught)

student\_course\_summary(student\_id, total\_courses, total\_credits, average\_grade, gpa)



### 3.2.3 物理设计

为了提高数据查询性能，我们需要为常用的查询字段设计索引。以下是几个常见的索引设计方案：

2.1. 学生表（students）

主键索引：student\_id 是学生表的主键，因此默认会创建主键索引。

辅助索引：可以为 email 和 phone 字段创建唯一索引，因为这两个字段有可能是查询、更新或检查唯一性的常用字段。

CREATE UNIQUE INDEX idx\_email ON students(email);

CREATE UNIQUE INDEX idx\_phone ON students(phone);

2.2. 课程表（courses）

主键索引：course\_id 是课程表的主键，因此默认会创建主键索引。

辅助索引：可以根据查询需求为 course\_code、course\_teacher 和 department 字段创建索引，尤其是 course\_code 常用于查询特定课程。

CREATE INDEX idx\_course\_code ON courses(course\_code);

CREATE INDEX idx\_course\_teacher ON courses(course\_teacher);

CREATE INDEX idx\_department ON courses(department);

2.3. 选课记录表（course\_enrollment）

主键索引：enrollment\_id 是选课记录表的主键，因此默认会创建主键索引。

外键索引：student\_id 和 course\_id 是外键字段，需要创建索引以优化联接查询和数据更新。

CREATE INDEX idx\_student\_id ON course\_enrollment(student\_id);

CREATE INDEX idx\_course\_id ON course\_enrollment(course\_id);

2.4. 成绩表（grades）

主键索引：grade\_id 是成绩表的主键，因此默认会创建主键索引。

外键索引：student\_id 和 course\_id 是外键字段，需要创建索引以优化联接查询和数据更新。

CREATE INDEX idx\_student\_id\_grade ON grades(student\_id);

CREATE INDEX idx\_course\_id\_grade ON grades(course\_id);

2.5. 出勤记录表（attendance）

主键索引：attendance\_id 是出勤记录表的主键，因此默认会创建主键索引。

外键索引：student\_id 和 course\_id 是外键字段，需要创建索引以优化联接查询和数据更新。

CREATE INDEX idx\_student\_id\_attendance ON attendance(student\_id);

CREATE INDEX idx\_course\_id\_attendance ON attendance(course\_id);

2.6. 教师表（teachers）

主键索引：teacher\_id 是教师表的主键，因此默认会创建主键索引。

辅助索引：email 和 phone 字段可以创建唯一索引，因为教师的邮箱和电话是唯一的。

CREATE UNIQUE INDEX idx\_teacher\_email ON teachers(email);

CREATE UNIQUE INDEX idx\_teacher\_phone ON teachers(phone);

2.7. 学生课程成绩汇总表（student\_course\_summary）

主键索引：student\_id 是学生课程成绩汇总表的主键，因此默认会创建主键索引。

外键索引：student\_id 是外键字段，需要创建索引以优化数据更新。

CREATE INDEX idx\_student\_id\_summary ON student\_course\_summary(student\_id);

# 4. 系统实施

## 4.1 设备与软件选型方案

1. 设备选型方案
2. 设备选型定义：设备选型指根据拟建项目的生产能力和技术方案，来确定设备的型号与规格。它需要有一次性、不可更改、兼容性、扩展性、功能性、性能性、易用性等参数阀值来参照
3. 设备选型原则：厂商的选择，扩展性考虑，可靠性，可管理性，安全性，QoS控制能力，标准性和开放性。
4. 设备选型的参考参数：
5. CPU：聚合运算、并行度、硬件RAM结构、内存匹配程度，是选择CPU的重要指标。
6. 内存：结果集响应时间、部署的软件负载程度、操作系统、数据交互、内存总线，是选择内存的重要指标。
7. 存储：根据业务数据应用场景，单位时间的吞吐量，I/O效率，数据总量，读写是否分离，顺序读写，随机读写，数据安全级别，HA，备份与恢复时间等等考虑。
8. 预估数据总量：一般以一年为一个时间单位，根据接口规范和需求说明书，把所有表中所有字段的长度累加起来，并计算出总的记录条数，这样就可以大概算出总的数据量是多少，在根据超配原理，放大2~3倍来得出结论。
9. 网络设备：网络吞吐量、最大并发会话数、速率、是否数据加密、带宽、支持多少光纤口和电口，电路保护、安全保护模式、支持几个磁盘通道（一个HBA+一个磁盘子系统=一个磁盘通道）光口or电口。
10. 根据I/O效率可分：PCI-E卡->SSD->SAS->SCSI->SATA
11. 操作系统：收费版、免费版、32bit、64bit

## 4.2 程序设计

**1. 创建数据库**

CREATE DATABASE student\_management\_system;

USE student\_management\_system;

**2. 创建学生信息表（students）**

CREATE TABLE students (

student\_id INT AUTO\_INCREMENT PRIMARY KEY, -- 学生ID，主键

name VARCHAR(50) NOT NULL, -- 学生姓名

gender CHAR(1) NOT NULL, -- 性别 M: 男, F: 女

birthdate DATE NOT NULL, -- 出生日期

phone VARCHAR(15), -- 联系电话

email VARCHAR(50), -- 邮箱

address VARCHAR(100), -- 地址

enrollment\_date DATE NOT NULL, -- 入学日期

major VARCHAR(50) NOT NULL, -- 专业

status CHAR(1) NOT NULL -- 学生状态 A: 在读, G: 毕业, D: 休学

);

INSERT INTO students (name, gender, birthdate, phone, email, address, enrollment\_date, major, status)

VALUES

('张三', 'M', '2005-01-15', '13800138000', 'zhangsan@example.com', '成都市武侯区', '2023-09-01', '计算机科学', 'A'),

('李四', 'F', '2004-11-10', '13800138001', 'lisi@example.com', '成都市青羊区', '2022-09-01', '软件工程', 'A'),

('王五', 'M', '2005-02-20', '13800138002', 'wangwu@example.com', '成都市锦江区', '2023-09-01', '电子工程', 'A'),

('赵六', 'F', '2004-05-30', '13800138003', 'zhaoliu@example.com', '成都市高新区', '2021-09-01', '网络工程', 'A'),

('孙七', 'M', '2005-07-25', '13800138004', 'sunqi@example.com', '成都市成华区', '2023-09-01', '人工智能', 'A');

**3. 创建课程信息表（courses）**

CREATE TABLE courses (

course\_id INT AUTO\_INCREMENT PRIMARY KEY, -- 课程ID，主键

course\_name VARCHAR(100) NOT NULL, -- 课程名称

course\_code VARCHAR(20) UNIQUE NOT NULL, -- 课程代码，唯一

course\_credit INT NOT NULL, -- 课程学分

course\_teacher VARCHAR(50), -- 授课教师

semester VARCHAR(20), -- 学期（如：2024-2025 秋季学期）

department VARCHAR(50) -- 所属学院

);

INSERT INTO courses (course\_name, course\_code, course\_credit, course\_teacher, semester, department)

VALUES

('数据结构', 'CS101', 3, '李教授', '2024-2025 秋季学期', '计算机科学与技术'),

('操作系统', 'CS102', 4, '王教授', '2024-2025 秋季学期', '计算机科学与技术'),

('电路原理', 'EE101', 3, '张教授', '2024-2025 秋季学期', '电子工程'),

('数据库原理', 'CS103', 3, '赵教授', '2024-2025 春季学期', '计算机科学与技术'),

('人工智能基础', 'AI101', 4, '孙教授', '2024-2025 秋季学期', '人工智能');

**4. 创建选课记录表（course\_enrollment）**

CREATE TABLE course\_enrollment (

enrollment\_id INT AUTO\_INCREMENT PRIMARY KEY, -- 选课记录ID，主键

student\_id INT NOT NULL, -- 学生ID，外键

course\_id INT NOT NULL, -- 课程ID，外键

enrollment\_date DATE NOT NULL, -- 选课日期

grade DECIMAL(5, 2), -- 成绩（可为空）

status CHAR(1) NOT NULL, -- 选课状态 P: 已选, D: 已退

FOREIGN KEY (student\_id) REFERENCES students(student\_id),

FOREIGN KEY (course\_id) REFERENCES courses(course\_id)

);

INSERT INTO course\_enrollment (student\_id, course\_id, enrollment\_date, grade, status)

VALUES

(1, 1, '2024-09-01', NULL, 'P'),

(1, 2, '2024-09-01', NULL, 'P'),

(2, 1, '2023-09-01', NULL, 'P'),

(3, 3, '2024-09-01', NULL, 'P'),

(4, 2, '2024-09-01', NULL, 'P');

**5. 创建成绩表（grades）**

CREATE TABLE grades (

grade\_id INT AUTO\_INCREMENT PRIMARY KEY, -- 成绩ID，主键

student\_id INT NOT NULL, -- 学生ID，外键

course\_id INT NOT NULL, -- 课程ID，外键

score DECIMAL(5, 2) NOT NULL, -- 成绩

grade\_date DATE NOT NULL, -- 成绩录入日期

FOREIGN KEY (student\_id) REFERENCES students(student\_id),

FOREIGN KEY (course\_id) REFERENCES courses(course\_id)

);

INSERT INTO grades (student\_id, course\_id, score, grade\_date)

VALUES

(1, 1, 90.5, '2024-12-01'),

(1, 2, 85.0, '2024-12-01'),

(2, 1, 88.0, '2023-12-01'),

(3, 3, 92.0, '2024-12-01'),

(4, 2, 78.5, '2024-12-01');

**6. 创建出勤记录表（attendance）**

CREATE TABLE attendance (

attendance\_id INT AUTO\_INCREMENT PRIMARY KEY, -- 出勤记录ID，主键

student\_id INT NOT NULL, -- 学生ID，外键

course\_id INT NOT NULL, -- 课程ID，外键

date DATE NOT NULL, -- 出勤日期

status CHAR(1) NOT NULL, -- 出勤状态 P: 出席, A: 缺席, L: 迟到

FOREIGN KEY (student\_id) REFERENCES students(student\_id),

FOREIGN KEY (course\_id) REFERENCES courses(course\_id)

);

INSERT INTO attendance (student\_id, course\_id, date, status)

VALUES

(1, 1, '2024-09-10', 'P'),

(1, 2, '2024-09-10', 'P'),

(2, 1, '2023-09-10', 'P'),

(3, 3, '2024-09-10', 'A'),

(4, 2, '2024-09-10', 'P');

**7. 创建教师信息表（teachers）**

CREATE TABLE teachers (

teacher\_id INT AUTO\_INCREMENT PRIMARY KEY, -- 教师ID，主键

name VARCHAR(50) NOT NULL, -- 教师姓名

gender CHAR(1) NOT NULL, -- 性别 M: 男, F: 女

phone VARCHAR(15), -- 联系电话

email VARCHAR(50), -- 邮箱

department VARCHAR(50), -- 所属学院

courses\_taught TEXT -- 所授课程（课程ID列表）

);

INSERT INTO teachers (name, gender, phone, email, department, courses\_taught)

VALUES

('李教授', 'M', '13900139000', 'li@example.com', '计算机科学与技术', 'CS101, CS102'),

('王教授', 'M', '13900139001', 'wang@example.com', '计算机科学与技术', 'CS102'),

('张教授', 'M', '13900139002', 'zhang@example.com', '电子工程', 'EE101'),

('赵教授', 'F', '13900139003', 'zhao@example.com', '计算机科学与技术', 'CS103'),

('孙教授', 'M', '13900139004', 'sun@example.com', '人工智能', 'AI101');

**8. 创建学生课程成绩汇总表（student\_course\_summary）**

CREATE TABLE student\_course\_summary (

student\_id INT PRIMARY KEY, -- 学生ID，外键

total\_courses INT NOT NULL, -- 总选修课程数

total\_credits INT NOT NULL, -- 总学分

average\_grade DECIMAL(5, 2), -- 平均成绩

gpa DECIMAL(5, 2), -- GPA

FOREIGN KEY (student\_id) REFERENCES students(student\_id)

);

INSERT INTO student\_course\_summary (student\_id, total\_courses, total\_credits, average\_grade, gpa)

VALUES

(1, 2, 7, 87.75, 3.5),

(2, 1, 3, 88.0, 3.6),

(3, 1, 3, 92.0, 3.8),

(4, 2, 6, 83.25, 3.2),

(5, 2, 7, 85.0, 3.4);

## 4.3 系统测试

**查询指定课程的学生信息**

-- 查询选修课程“数据结构”的学生信息

SELECT students.student\_id, students.name

FROM students

JOIN course\_enrollment ON students.student\_id = course\_enrollment.student\_id

JOIN courses ON course\_enrollment.course\_id = courses.course\_id

WHERE courses.course\_name = '数据结构';



-- 更新学生“张三”在“数据结构”课程的成绩

UPDATE grades

SET score = 95.0

WHERE student\_id = 1 AND course\_id = 1;



## 4.4 运行维护

**1.1. 数据库性能监控**

为了确保数据库系统的稳定性和高效性，我们需要定期监控系统的各项性能指标，包括查询性能、系统负载、响应时间等。

常见的监控指标：

* **查询响应时间**：检查长时间运行的查询，优化查询语句或添加索引。
* **数据库连接数**：监控数据库连接数，防止连接池溢出。
* **磁盘使用情况**：监控磁盘空间，防止磁盘满载。
* **CPU和内存使用情况**：监控数据库服务器的CPU和内存使用率，确保系统资源不会被过度消耗。
* **锁等待情况**：监控数据库的锁等待情况，避免死锁或长时间的资源竞争。

**SQL查询示例**：监控慢查询

-- 查看执行时间较长的查询

SHOW FULL PROCESSLIST;

**1.2. 优化查询性能**

通过以下方法来优化数据库性能：

* **添加索引**：针对查询频繁的字段添加索引，尤其是 JOIN 和 WHERE 子句中的字段。
* **查询重写**：优化 SQL 查询语句，避免使用低效的查询方式，如避免在 WHERE 子句中使用函数。

**1.3. 定期备份**

为了防止数据丢失，定期对数据库进行备份是至关重要的。备份策略一般包括全备份、增量备份和差异备份。

* **全量备份**：定期备份整个数据库。
* **增量备份**：只备份自上次备份以来发生变化的数据。
* **差异备份**：备份自上次全备份以来所有变更的数据。

**备份命令：**

-- 全备

mysqldump -u root -p --all-databases > full\_backup.sql

-- 增量备份

mysqldump -u root -p --all-databases --where="last\_update > '2024-01-01'" > incremental\_ba

# 5、本组各成员在团队中承担的工作说明及课程设计体会

## 5.2 成员1学号姓名：

工作说明：开展项目的整体技术设计，背景需求分析

收获和体会：通过整体技术设计的工作，我对项目的全局框架有了更加清晰的了解。需求分析不仅帮助我深入理解了客户的实际需求，还让我认识到在设计阶段提前规划系统的可扩展性和易维护性的重要性。在与团队成员沟通时，我发现了自己对技术细节的认知不足，提升了自己在沟通和协调方面的能力。

## 5.2 成员2学号姓名：

工作说明：分析系统流程，给出了系统的流程分析图

收获和体会：在分析系统流程的过程中，我学到了如何从用户的需求出发，梳理出流程的各个环节。通过制作流程图，我能够清晰地呈现出系统的运作方式和信息流转的路径，提升了自己对复杂问题的分析能力。也意识到流程设计不仅要考虑功能的实现，还要顾及系统的稳定性和用户体验。

## 5.2 成员3学号姓名：

工作说明：指出所需的数据字典。分析出功能模块。

收获和体会：数据字典的编写让我更深刻地理解了数据库与应用程序之间的关系，也锻炼了我整理信息、定义标准化内容的能力。在分析功能模块时，我学会了如何根据系统的需求划分模块，并合理分配开发工作，确保每个模块能够顺利运行并相互协作

## 5.2 成员4学号姓名：

工作说明： 分析概念设计，创建出ER图，逻辑结构，物理结构

收获和体会：通过创建ER图和逻辑结构，我学到了如何将需求转化为实际的数据库模型，并在此基础上进行物理结构的设计。在这个过程中，我加强了对数据库设计的理解，特别是在如何优化数据库结构，提高查询效率方面，获得了宝贵的经验。

## 5.2 成员5学号姓名：

工作说明：数据库实施，编写sql 代码。

收获和体会：在数据库实施的过程中，我深入学习了SQL编程，特别是在优化查询语句和设计高效数据库表结构方面积累了经验。虽然遇到了一些性能瓶颈问题，但通过团队的讨论和不断试验，我成功解决了这些问题，提升了自己解决实际问题的能力。

## 5.2 成员6学号姓名：

工作说明：进行系统测试，并提出运行维护方案

收获和体会：系统测试让我更加了解了如何保证软件的质量，发现并修复潜在的bug是一个复杂但又充满挑战的过程。在编写维护方案时，我思考了如何确保系统在长期运行中的稳定性和可扩展性，这让我对软件生命周期的管理有了更深刻的理解。