|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 《数据库原理》实验报告 | | | | |
| 题目：实验八  数据库综合实验 | 学号 | 姓名 | 班级 | 日期 |
| **2020302915** | **向振羽** | **10042001** | **2022.12.14** |

**目录**

[一. 实验概述 2](#_Toc13352)

[系统名称 2](#_Toc9460)

[系统简介 2](#_Toc26427)

[系统功能介绍 2](#_Toc8022)

[配置运行说明 2](#_Toc866)

[二. 实验内容 2](#_Toc22163)

[1. 需求分析 4](#_Toc12722)

[1) 用户需求分析 4](#_Toc27185)

[2) 系统数据流图 5](#_Toc19281)

[3) 系统数据字典 5](#_Toc3983)

[2. 概念结构设计 8](#_Toc13736)

[1) E-R图 8](#_Toc11729)

[3. 逻辑结构设计 10](#_Toc16914)

[1）关系模式：由概念结构向逻辑结构的转化 10](#_Toc2861)

[2）函数依赖集、关系模式的优化 10](#_Toc5717)

[3）表设计 10](#_Toc30631)

[4）数据库的完整性和安全性设计 11](#_Toc26823)

[4. 物理结构设计 11](#_Toc15828)

[5. 程序设计 12](#_Toc20050)

[三. 实验中出现的问题以及解决方案 15](#_Toc13214)

[四. 实验心得 16](#_Toc7101)

1. **实验概述**

**系统名称**

学生信息管理系统Student information management system

**系统简介**

该系统是一个由python语言编程形成的DBMS，操作系统为Windows 11，编译运行环境为pycharm。数据库的建立和表的创建使用MySQL workbench。

**系统功能介绍**

此系统功能分为面向学生和面向管理员两部分。面向管理员的部分可以实现学生信息的添加、删除、修改、查询、排序和浏览；面向学生的部分，学生可以注册自己的个人信息，对自己的信息进行浏览。管理员有整个数据库系统的管理权限。

**配置运行说明**

* 下载并安装MySQL

# 下载并安装 MySQL workbench

* 下载并安装Anaconda和Pycharm

# 打开安装好的MySQL workbench

* 在其中建立student2数据库，运行 “sql代码”文件夹中的“学生数据库.sql”文件
* 导入完成后，打开“运行代码”文件夹，点击.py文件，用Pycharm打开



* 接下来修改相关测试，如user名，password密码
* 请点击Pycharm上方的运行键，即可连接数据库并启动进入学生信息管理系统的登录页面，登陆后的其他操作请参照后文中的内容进行：



1. **实验内容**
2. **需求分析**
3. **用户需求分析**

面向管理员：

* 学生信息管理
* 查看全部用户详细信息
* 编辑用户详细信息
* 增加用户信息
* 删除用户信息
* 查询特定用户信息
* 账户信息管理
* 修改学生密码
* 修改自己密码
* 增加管理员密码

面向普通用户：

* 个人信息
* 查看我的个人的详细信息
* 个人密码的修改
* 联系管理员进行密码修改
* 修改自己密码
* 个人信息修改
* 联系管理员进行所在院系等个人信息的修改

1. **数据字典**

**数据流：**

1. 学生信息数据流

数据流名：学生信息

说　　明：根据学生信息建立或者编辑学生记录

来源去向：学生→学生管理

数据结构：学生信息表

1. 用户信息数据流

数据流名：用户信息

说　　明：根据学号和密码鉴定用户身份为管理员或者学生

来源去向：用户→安全管理

数据结构：人员信息表

1. 用户身份数据流

数据流名：用户身份

说　　明：不同用户身份进入的处理过程不同

来源去向：安全管理→管理系统/查询系统

数据结构：学号+密码，管理员信息表，人员信息表

1. 浏览请求数据流

数据流名：浏览请求

说　　明：通过学生学号查询个人信息和成绩信息

来源去向：学生→浏览个人信息和成绩信息

数据结构：学号，成绩信息表，个人信息表

1. 增加请求数据流

数据流名：增加请求

说　　明：根据学号和个人信息，进行增加处理，并创建一条相应的学生信息记录和账户信息记录

来源去向：学生→增加管理

数据结构：学号，学生信息表，学生账户信息表

1. 删除请求数据流

数据流名：删除请求

说　　明：根据学号，进行删除处理，并更新相应的学生信息记录和账户信息记录

来源去向：学生→删除管理

数据结构：学号，学生信息表，学生账户信息表

**数据存储：**

1. D1：学生记录

数据存储名：学生记录

说 明：记录每一个学生的信息，学号是主码，性别只能为“男”或“女”

输入数据流：学生信息

输出数据流：增加学生信息

数据描述 ：学号，姓名，性别，年龄

1. D2：账户记录

数据存储名：账户记录

说 明：记录每一个账户的信息，账户名/学号是主码

输入数据流： 账户名/学号，密码

输出数据流：账户信息

数据描述 ：账户名/学号，密码

1. D3：成绩记录

数据存储名：成绩记录

说 明：记录每位学生的信息，学号和课程名是主码

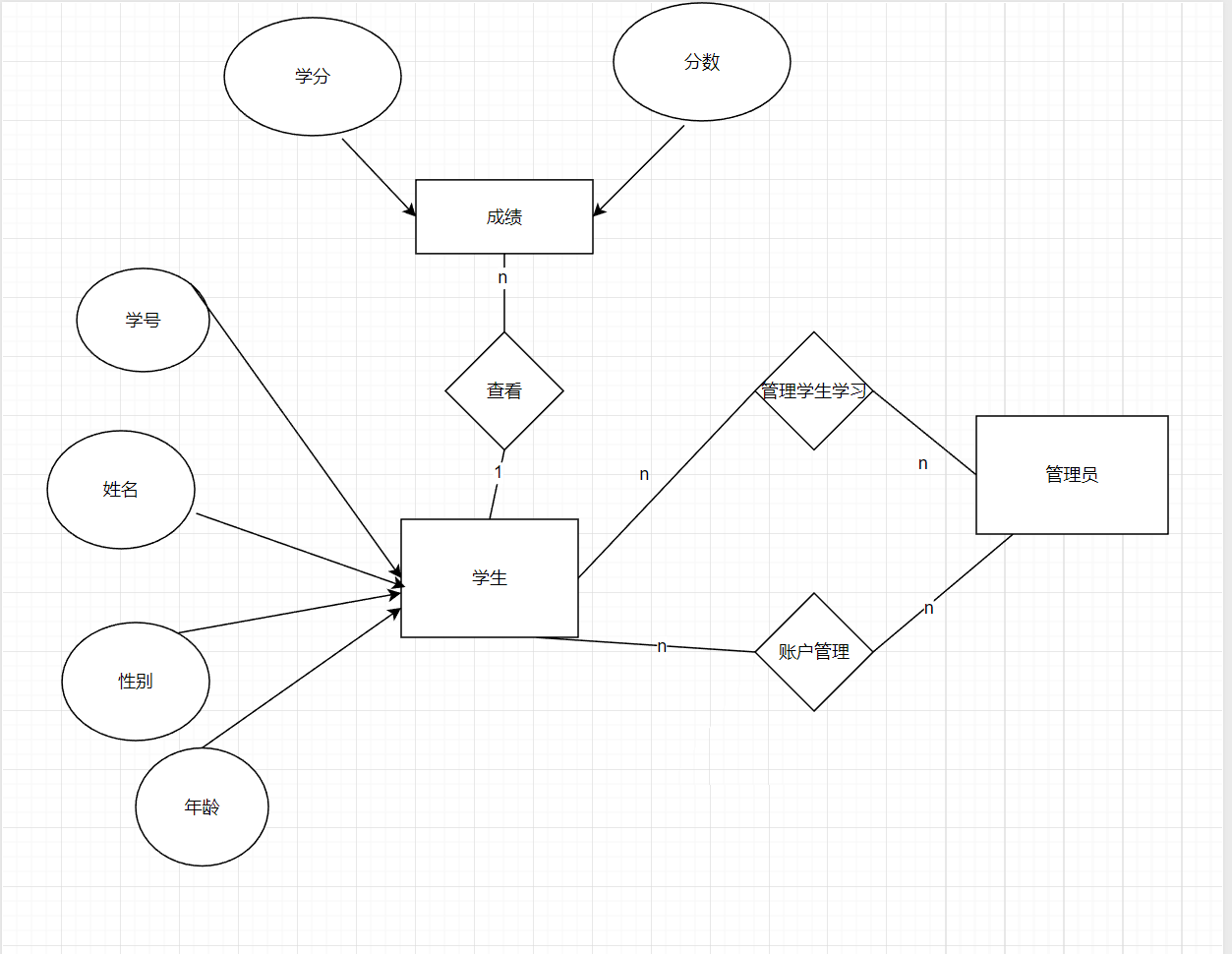
输入数据流：学号、课程名、学分、成绩

输出数据流：成绩记录

数据描述 ：学号、课程名、学分、成绩

1. **概念结构设计**

**E-R图：**



1. **逻辑结构设计**
2. **关系模式：由概念结构向逻辑结构的转化**

* 数据库中的3个实体的完整描述如下：

用户：用户名，用户密码（管理员/学生）

学生：学号，姓名，性别，年龄

课程：课名，学分

* 用户中学生和管理员的身份以两个不同的表作以区别
* 学生和课程的成绩联系由新建一张成绩表来实现
* 关系模型：（主键已用下划线标出）

admin\_id：（admin\_id，admin\_pass）

stu\_login\_k：（stu\_id，stu\_pass）

student\_k：（id，name，gender，age，dept）

course：（cno，credit）

t\_course：（SNO，COURSE，CREDIT，GRADE）

* 视图：（学生用户操作在视图上进行）

|  |
| --- |
| CREATE VIEW course\_view AS SELECT \* from t\_course;  CREATE VIEW stu\_view AS SELECT \* from student\_k; |

1. **函数依赖集、关系模式的优化**

= {学号→姓名，学号→性别，学工号→年龄，学工号→系名}

= {课名→学分}

= {（学号，课名）→成绩}

以上函数依赖集不存在非主属性和主属性对码的部分函数依赖和传递函数依赖，故该函数依赖集满足BCNF。

1. **数据库的完整性**

|  |
| --- |
| -- 给student\_k表的gender字段添加约束check1  alter table student\_k add constraint check1 check(gender in ('男','女')); |

1. **物理结构设计**

在student\_k表的name属性上建立索引，便于管理员查询时快速查找。

|  |
| --- |
| create index sel\_name on student\_k(name) |

1. **程序设计**

**1）代码架构设计**

1. **创建的类**

在程序中我设计了11个类，分别实现 “StartPage—主界面”、“AdminPage—管理员登陆页面”、“StudentPage—学生登陆页面”，“AdminManage—管理员操作界面”、“StudentView—学生查看信息界面”、“Selectgrade—成绩查看页面”、“Changekey—学生账号修改页面”、“AboutPage—账号系统”、“Adminlogin—管理员账号设置” 、“CreateAdminPage—创建管理员账号” 、“AdminChange—管理员密码修改”。

1. **需要的材料**

在picture文件夹中有界面设计所需的图片材料

****

**2）触发器设计**

**(1)** **insert\_s触发器**

在t\_course表上设计insert触发器insert\_s，该触发器的作用是在插入学生成绩信息时，科目及学分必须满足course表，否则无法插入。

|  |
| --- |
| After insert on t\_course  For each row  BEGIN  IF NEW.SNO NOT IN (SELECT DISTINCT(id) FROM student\_k)  THEN SIGNAL SQLSTATE '45000'  SET MESSAGE\_TEXT = '不能插入S表没有的数据';  END IF;  IF NEW.COURSE NOT IN (SELECT DISTINCT(cno) FROM course)  THEN SIGNAL SQLSTATE '45000'  SET MESSAGE\_TEXT = '不能插入C表没有的数据';  END IF;  END |

**(2)** **add\_s\_lock触发器**

每增加一位学生的信息，即在student\_k上插入一条信息，在其后为其创建账户，即在stu\_login\_k表增加对应的信息

|  |
| --- |
| After insert on student\_k  For each row  begin  set @id = new.id;  insert stu\_login\_k value(@id,'123456');  end |

**(3)** **del\_s\_lock触发器**

同理，每当管理员删除一个学生的信息时，其对于账户也随之被删除。

|  |
| --- |
| After delete on student\_k  For each row  begin  set @id = old.id;  delete from stu\_login\_k where stu\_id = @id;  end |

1. **存储过程设计**

改存储过程利用游标将学生所有已出成绩的科目学号相加，得出总学分。

|  |
| --- |
| CREATE DEFINER=`root`@`localhost` PROCEDURE `switch`(in id char(45))  BEGIN  DECLARE total FLOAT;  declare DONE INT DEFAULT 0;  DECLARE GREDIT FLOAT;  DECLARE C\_CREDIT CURSOR FOR  SELECT CREDIT  FROM t\_course  where SNO=id;  DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET DONE = 1;  OPEN C\_CREDIT;  FETCH C\_CREDIT INTO GREDIT;  set total = 0;  REPEAT  set total = total+GREDIT;  FETCH C\_CREDIT INTO GREDIT;  UNTIL DONE  END REPEAT;  SELECT total;  END |

**4）窗体设计**

**(1)开始界面**

开始界面四个按钮分别如下图所示。



**(2)管理员/学生登录界面**



**(3)管理员身份登陆后看到的主页面**

**学生信息页面**



点击表中任意一行的内容，即可看到该同学的个人详细信息，并且点击学号、姓名、性别、年龄可以进行正序逆序排列，如下图所示：



在信息页面，可以插入、更改、删除和查询四项功能，其中新建和删除会对应建立或删除学生账户。

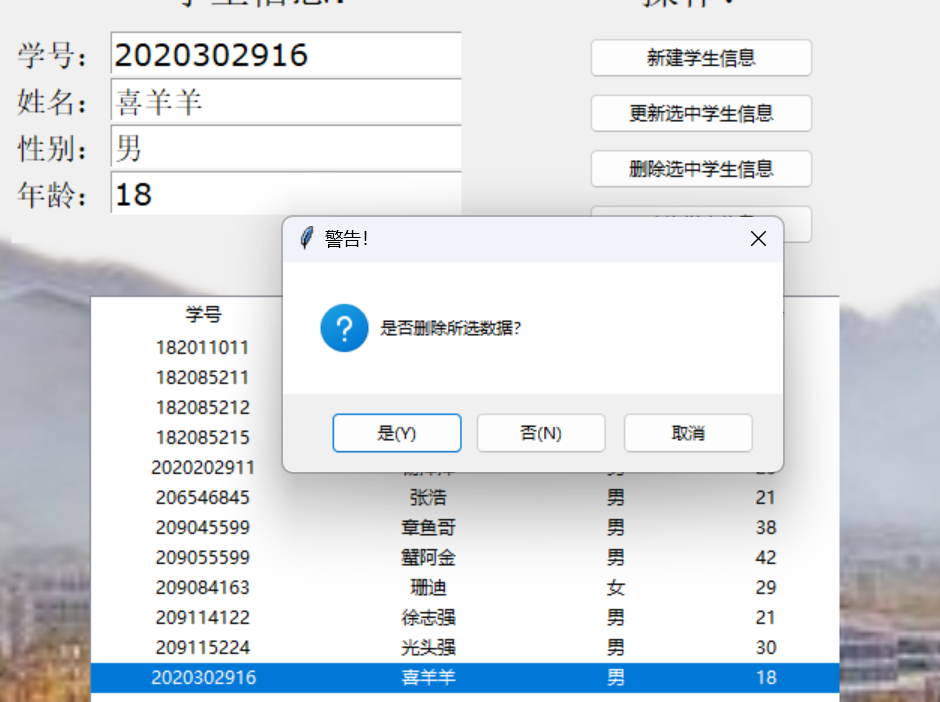
**新建操作：**

****

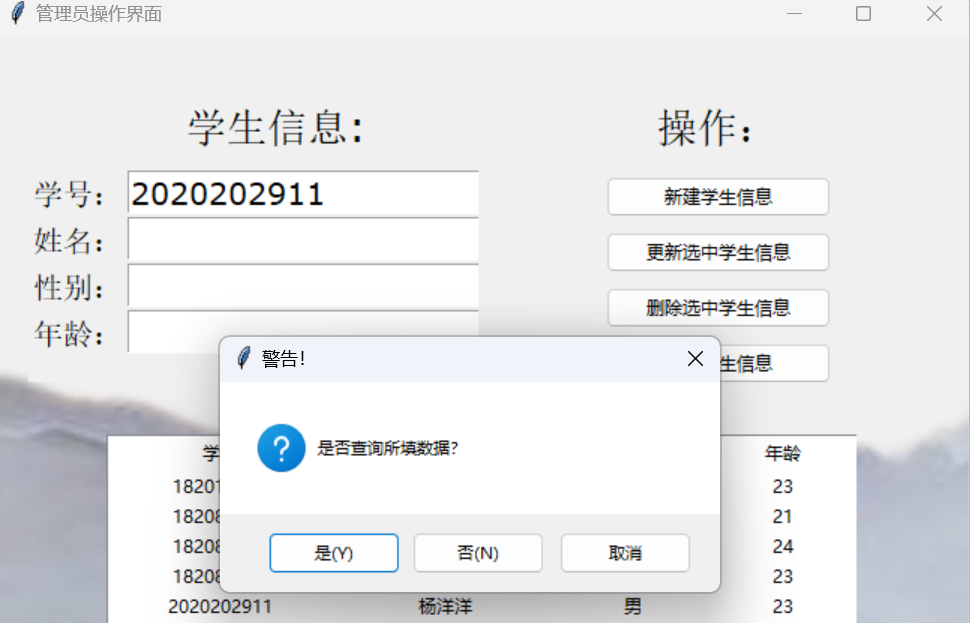
**更新操作：**



**删除操作：**



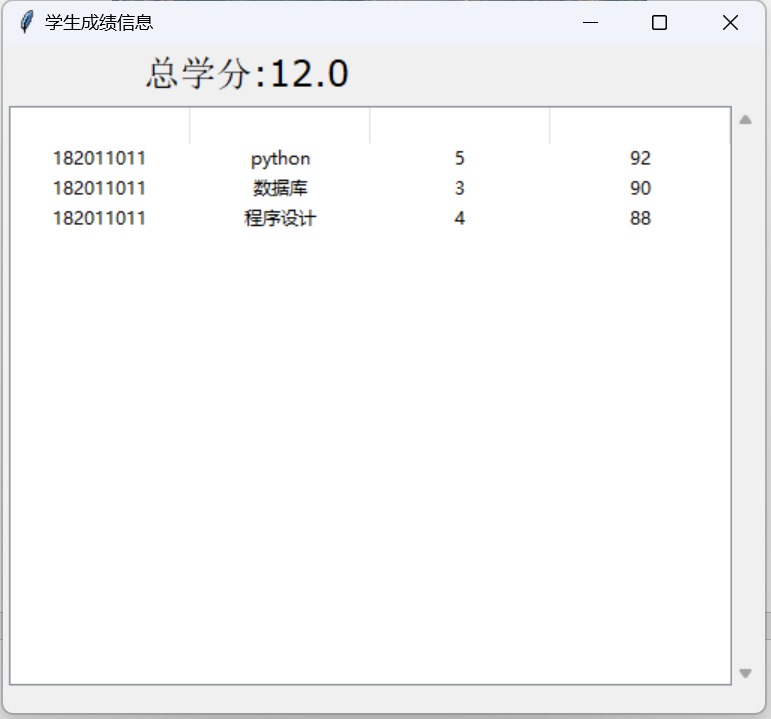
**查询操作：**

****

**(3)学生用户身份登陆后看到的主页面**



查看成绩：



修改密码：



**(4)账户系统**

可以选择自己修改密码，如果密码忘记课通知管理员修改密码

****

1. **实验中出现的问题以及解决方案**

其实整个实验中遇到的问题还是很多的，在选题的时候就出现了困难，虽然在实验指导书里已经有很多例子了，但是我还是不知道怎么选择，鉴于咱们之前就已经多次使用的student库，就想到做一个学生信息管理系统，毕竟很多东西可以直接使用，所以着手做了这个实验内容。

在此前，我其实从来都没有接触过tkinter窗体设计相关的一些知识，这次的大作业可以说是从头学起，所以过程还是比较艰辛的，比如刚开始想要将下载的图片设为窗口背景但总是不如意。

在进行数据库中各表在窗体上展示的过程中，我不知道由表的连接得到的字段应该如何显示在窗体中，经过查阅资料，我得知，Python tkinter 树形列表控件可以很容易展示。

还有就是按键设置，按键少了一个参数导致无法使用还有就是跳转失败导致整个窗体界面突然崩溃，所幸经过长时间修改后，窗体正常了，虽然还有一些按键的函数错位，但是我最后都重新调整了回来。

1. **实验心得**

在整个大作业开始之前，我一直是抱着比较忐忑的心，因为确实这对我来说有难度，在完成期间我学习了python中应用tkinter库设计界面。

可以说这次的大作业对我来说是个不小的挑战，是对我这个学期以来对数据库理论和实验掌握程度的考察，所幸我最后通过了考察。大作业是对理论课本第七章第八章的很好的实践学习，让我亲身经历了一遍一个数据库管理系统从无到有的过程。这是一次于我而言很好的锻炼机会，有很多内容是靠自学完成的，虽然最后的成品可能并不是很完美，但是里面真的倾注了我很多的心血，当我在日后随着我的能力提升，我可能还会将这个系统继续美化和完善。

数据库实验到这里也就告一段落了，不论是SQL语言的增删改查，MySQL图形化界面的操作和使用，数据的完整性和安全性约束，视图和索引的创建，还是ODBC/JDBC的数据库连接，存储过程和触发器和编写，事务与并发控制，到最后独立完成一个数据库管理系统的设计和实现，都让我收获了很多也学到了很多，我会带着这些收获，继续后面所有科目的学习。

**批阅者：**

**批阅日期：**

**实验成绩：**

**批注：**