**Management Information System Design Document**

**18计科全英联合班 向天翼 201836020389**

**I.I Overview of tables**

To organize and manage all the record in this project, five tables (showed in Fig.1) are built in advance here, where the first four tables are built based on the given requirement and the fifth table ‘user\_data’ is used for store the user name and password for all the users in this project, including administrator, teachers and students.

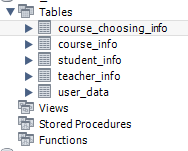


Fig.1 Overview of tables used in this project

Fig.2 gives the structure of user data table. The user name for administrator is ‘admin’ and there is only one administrator of this project. And user name for teachers is teacher ID and for student is student ID. The default password is ‘123456’ and all the password store in database has been encrypted using SHA1 algorithm. Based on the SHA1 function provided by SQL, this function can be implemented easily.

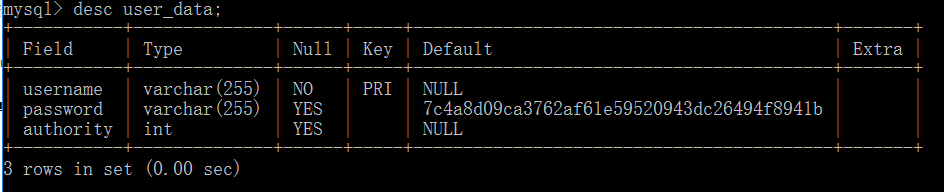


Fig.2 Structure of table ‘user\_data’

|  |  |
| --- | --- |
| Operation | SQL Command |
| Create course information table | CREATE TABLE course\_info (  id varchar(7) NOT NULL, name varchar(255) NOT NULL,  teacher\_id varchar(5) NOT NULL, credit double(255,1) NOT NULL,  grade\_limit int NOT NULL, canceled\_year int DEFAULT NULL,  PRIMARY KEY (id,teacher\_id), KEY id (id)  ); |
| Create student information table | CREATE TABLE student\_info (  id varchar(10) NOT NULL, name varchar(255),  sex varchar(10) NOT NULL, entrance\_age int NOT NULL,  entrance\_year int NOT NULL, class varchar(255),  PRIMARY KEY (id)  ); |
| Create teacher information table | CREATE TABLE teacher\_info (  id varchar(5) NOT NULL,  name varchar(255) NOT NULL,  courses text,  PRIMARY KEY (id), KEY id (id)  ); |
| Create course choosing information table | CREATE TABLE course\_choosing\_info (  chosen\_year int NOT NULL, score int DEFAULT NULL,  student\_id varchar(10) NOT NULL,  teacher\_id varchar(5) NOT NULL,  course\_id varchar(7) NOT NULL,  PRIMARY KEY (student\_id,teacher\_id,course\_id,chosen\_year),  FOREIGN KEY (student\_id) REFERENCES student\_info (id) ON DELETE CASCADE ON UPDATE CASCADE,  FOREIGN KEY (teacher\_id) REFERENCES teacher\_info (id) ON DELETE CASCADE ON UPDATE CASCADE,  FOREIGN KEY (course\_id) REFERENCES course\_info (id) ON DELETE CASCADE ON UPDATE CASCADE  ); |
| Create user account data table | CREATE TABLE user\_data (  username varchar(255) NOT NULL,  password varchar(255) DEFAULT '7c4a8d09ca3762af61e59520943dc26494f8941b',  authority int DEFAULT NULL,  PRIMARY KEY (username)  ); |

Table 1. SQL command for create initial tables

**I.II SQL Help class and Common class**

SQL Help class provide all the callable functions to manipulate database including querying function and modifying function. The parameters of the last two functions are the SQL command string, the connection object and the SQL command parameter list. All interactive operations on the database in this project are performed using these two functions. The SQL Help class also includes the connection string which is used to create connection with database.

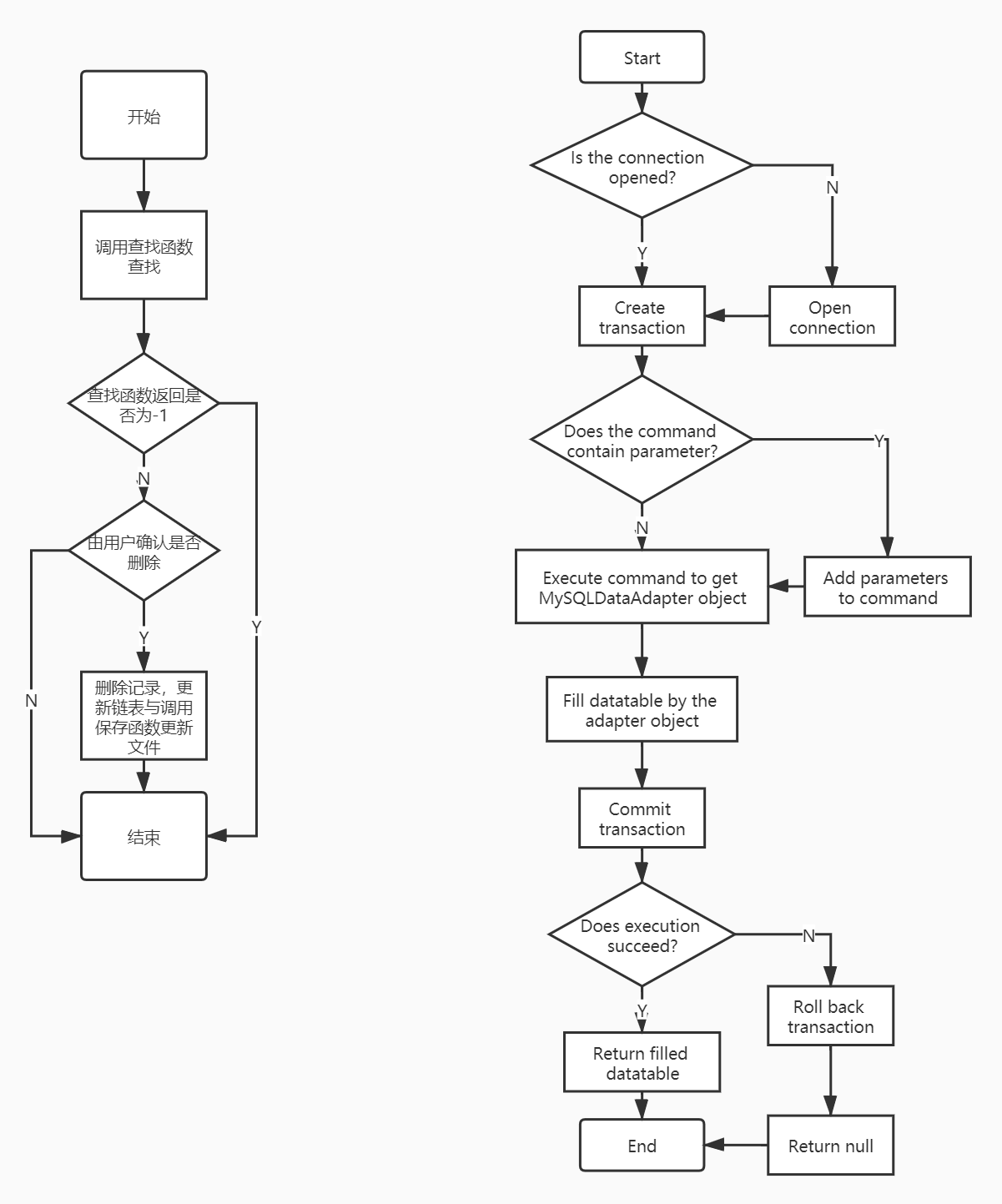
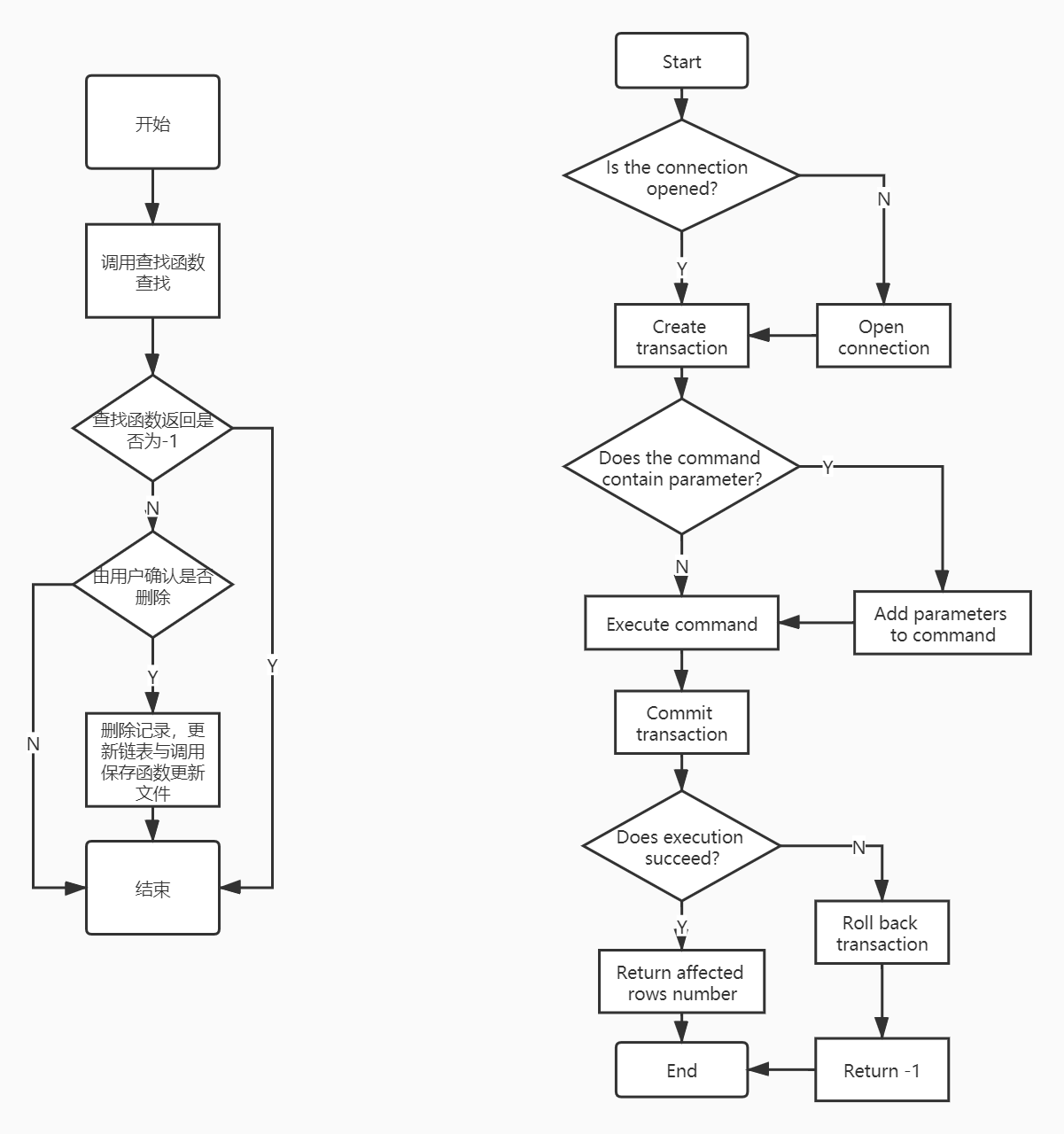


Fig.3 Logic flow of modifying function (left) and querying function (right)

The Common class is mainly used to store some common functions and objects. In order to distinguish students, teachers and administrators, an enumeration User Type is defined here. In addition, the error, prompt and choice messages are sent to the user, as well as the reading, writing and saving of CSV files. The functions related to them are all responsible for this class

**I.III Login Window**

According to the username and password from user, the program will query for authority information in user information table in the database. If the query result is empty, it will report the wrong username or password. Otherwise, it will record the query result and open the main window.

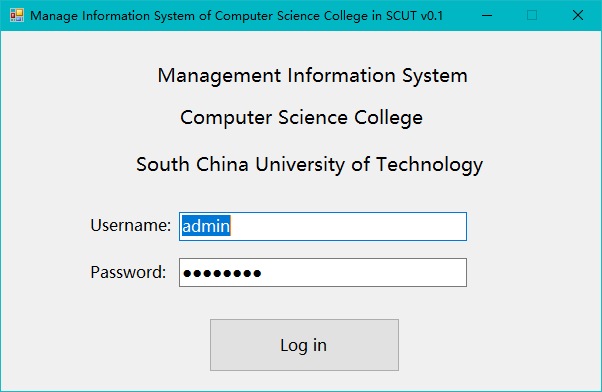


Fig.4 Login interface

|  |  |
| --- | --- |
| Operation | SQL Command |
| Query for authority | select authority from user\_data where username = @name and password = SHA1(@pwd); |

Table 2. SQL command for login

**I.IV Add record**

I.IV.I Add student record

There are two ways to add student information in this program. The first is to add the information of a single student by filling in the textboxes, the second is to fill in the information according to the generated CSV template file, and then add the student information in batches by reading the CSV file.

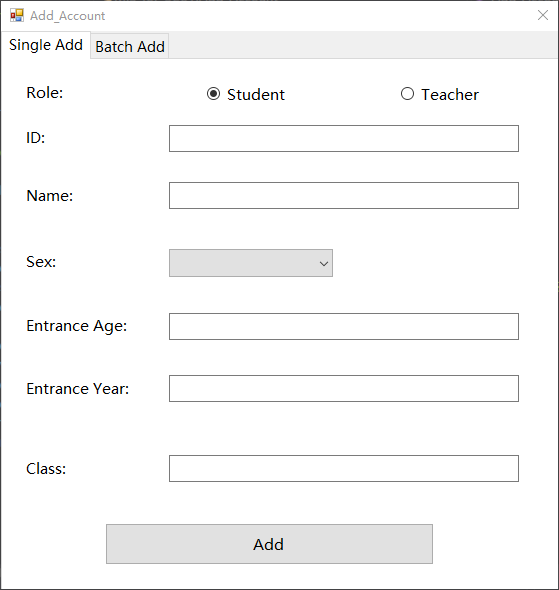
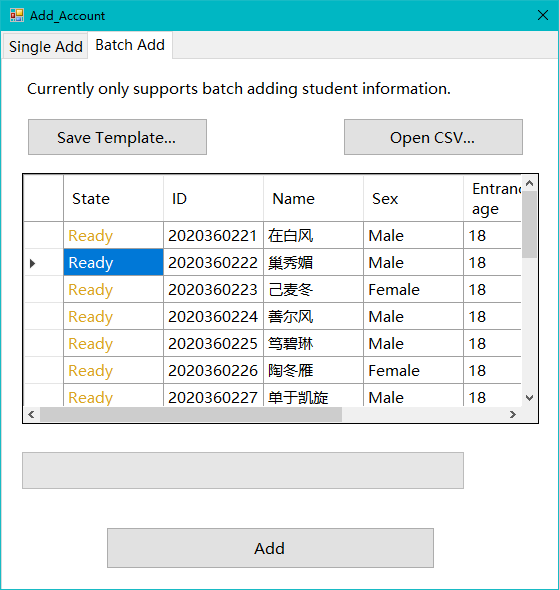
 

Fig.5 Two optional ways to add student information

I.IV.II Add teacher record

When adding teacher record, only teacher ID and teacher name are needed. The courses taught by the teacher will be automatically updated when adding courses.

I.IV.III Add course record

For adding new course, the ID, name, teacher ID, credit and grade limitation should be provided, and the canceled year is optional. In order to ensure the administrator can input correct teacher ID, a combo box is used here. When this form is loading, all the teacher information will be loaded into the item list of combo box. So, the administrator only needs to choose one teacher from the list.

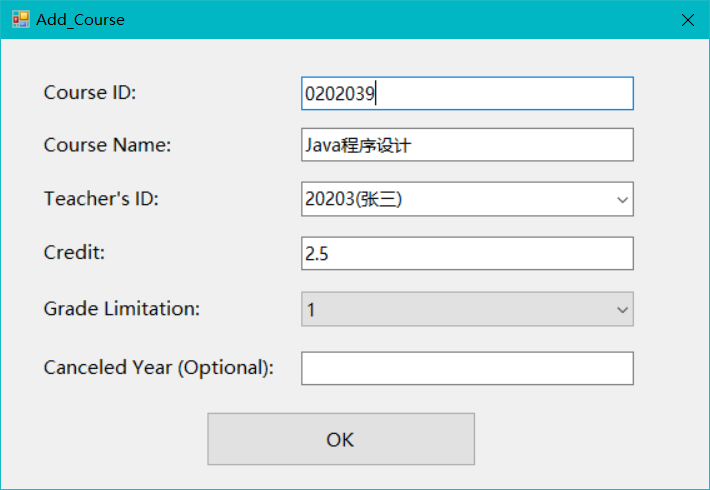


Fig.6 Add course information interface

I.IV.IV Add course choosing record

When this form is loading, current added course and corresponding teacher information, and current class are loaded. The administrator can add the course selection information of a single student, or add the course selection information of classmates at the same time, just add the student information to the list box named chosen students.

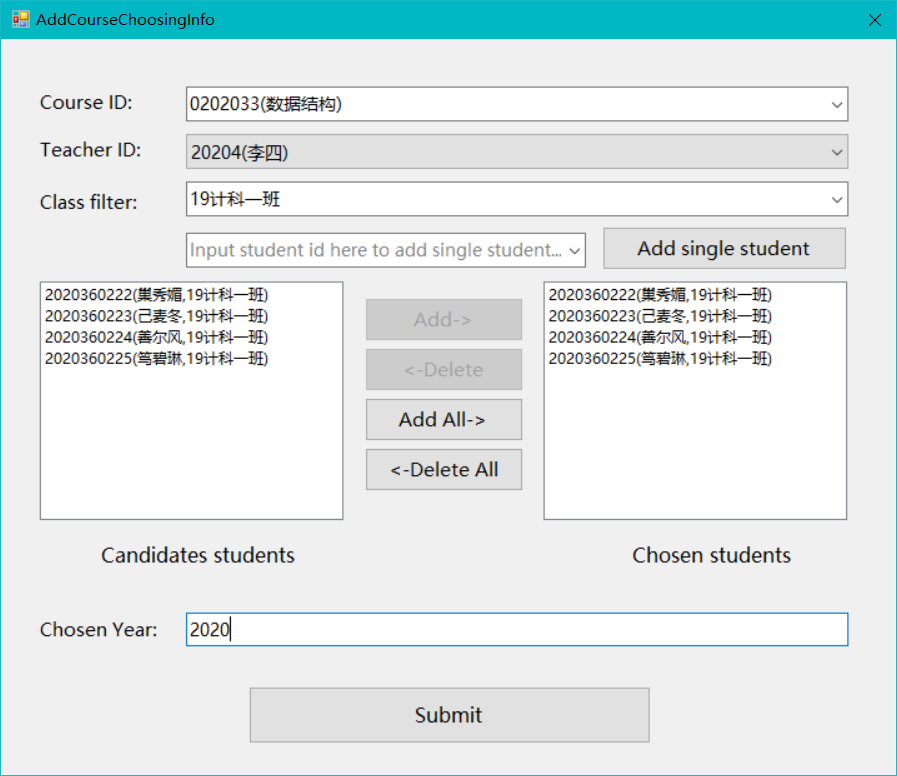


Fig.7 Add course choosing information interface

|  |  |
| --- | --- |
| Operation | SQL Command |
| Add student record | insert into user\_data (username,authority) values(@student\_id,0);  insert into student\_info values(@student\_id,@student\_name,@sex,@age,@year,@class); |
| Add teacher record | insert into user\_data (username,authority) values(@name,1);  insert into teacher\_info (id,name) values(@id,@name); |
| Add course record | insert into course\_info values (@id,@name,@teacher,@credit,@grade,@year);  update teacher\_info set courses = @new\_course where id = @teacher; |
| Add course choosing record | insert into course\_choosing\_info (student\_id,teacher\_id,course\_id,chosen\_year) values(@student\_id,@teacher\_id,@course\_id,@year); |

Table 3. SQL command for adding record

**I.V Modify and delete record**

All the modify and delete function except modifying student score is only accessible for administrator account.

I.V.I Student information record

To modify a single record for a student, his or her student ID should be provided to query current information about this student. In default, all the new information is same as current information, the administrator can make changes on new information and then apply changes. Or delete this student record. Before do the delete operation on database, a confirm dialog will be showed. If the administrator confirms this operation, all the record in user data table, student table and course choosing table about this student will be deleted in the same time.

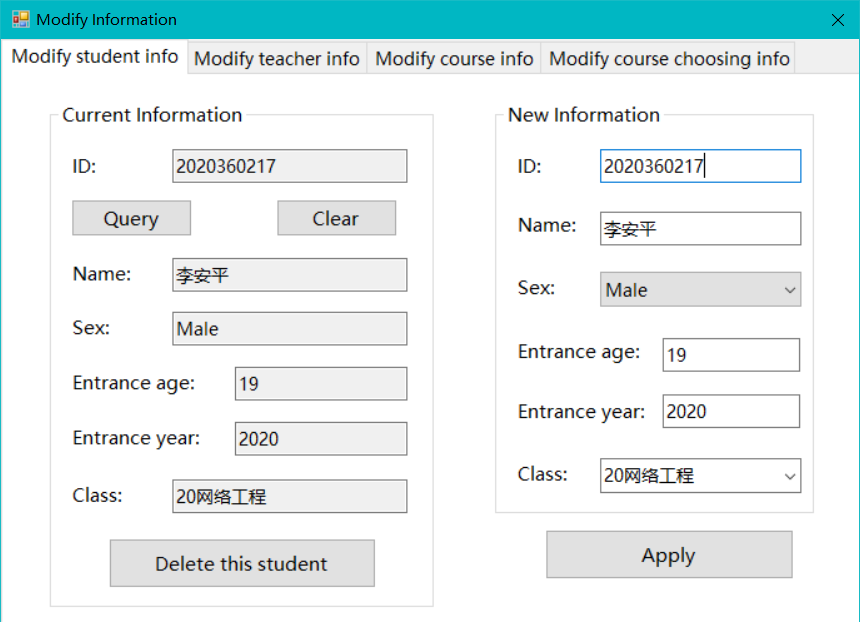


Fig.8 Modifying or deleting student information interface

I.V.II Teacher information record

Modifying or deleting teacher record is similar as doing these operations on student record. The teacher ID should be provided to query current information for a teacher, then the administrator can choose to delete this record or apply changes on this record.

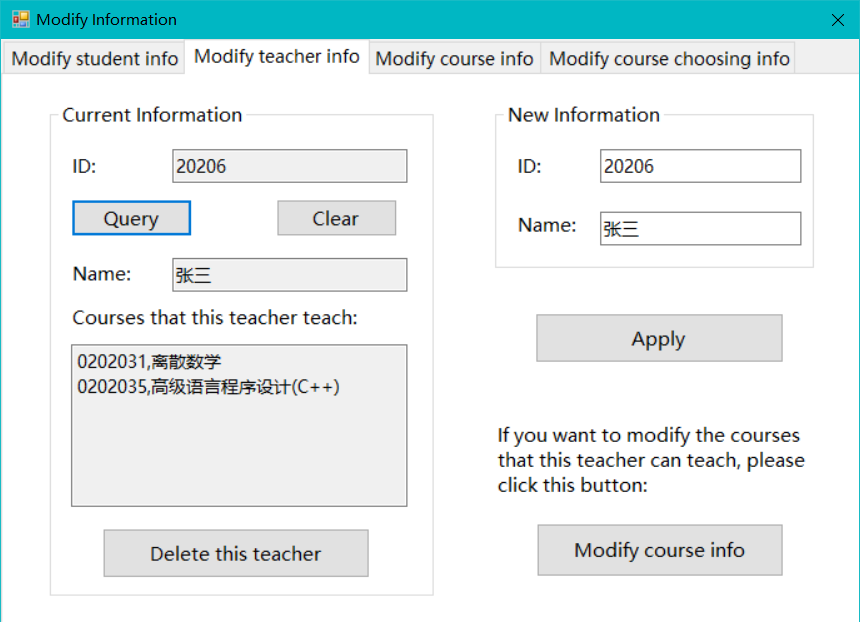


Fig.9 Modifying or deleting teacher information interface

I.V.III Course information record

Modifying or deleting course record is similar as doing these operations on student record. But since a single course may be taught by multiple teachers, the query result will be multiple records with course ID as the query condition. So, when there are multiple records, the administrator should choose correct teacher and make change on it or delete it. If the administrator want to delete a whole course, he/she can click ‘Delete course’ button, then all the records related to this course ID will be deleted from database.

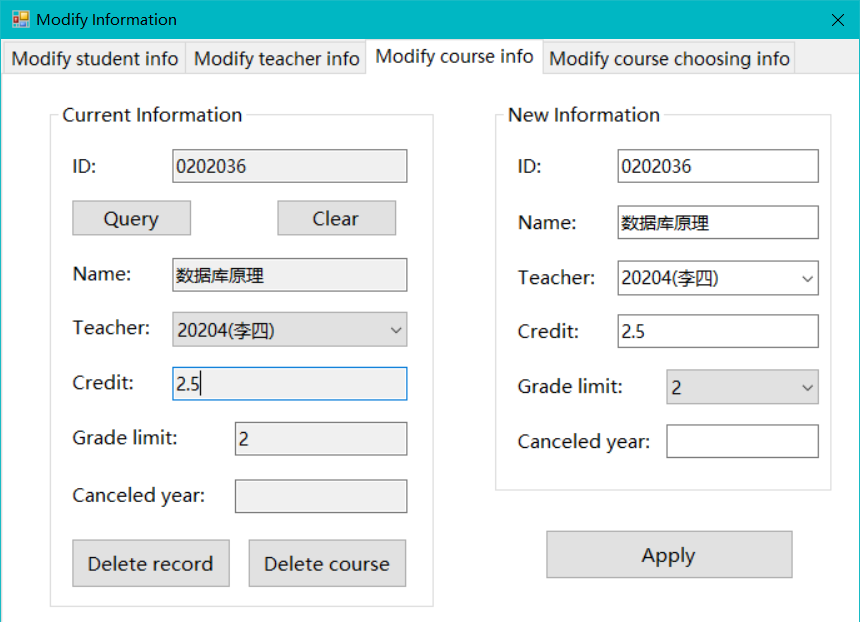


Fig.10 Modifying or deleting course information interface

I.V.IV Course choosing information record

To modify or deleting course choosing record, a student ID should be the query condition, then all the course choosing record of this student will be listed. Then the administrator can modify or delete target record by clicking corresponding button of the record, which is showed in Fig.10.

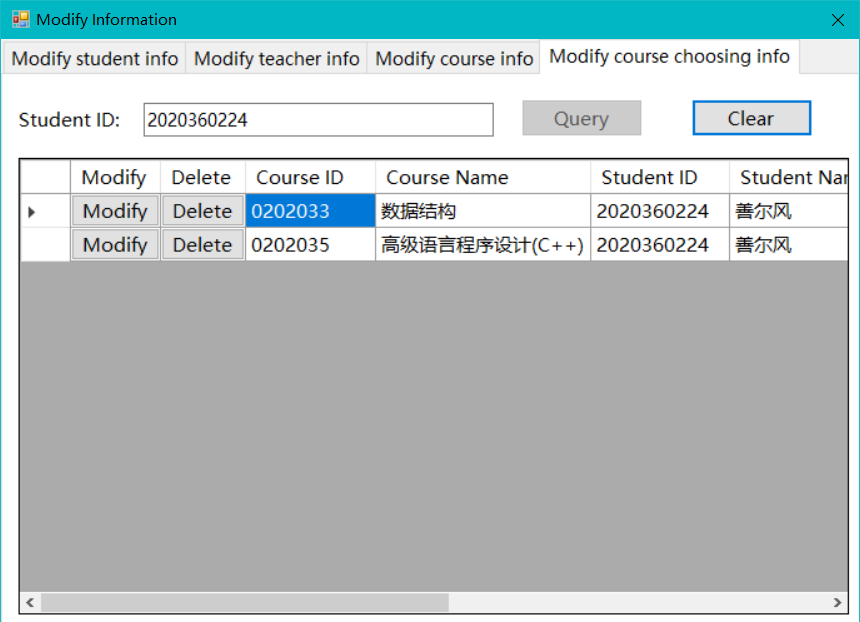


Fig.11 Modifying or deleting course choosing information interface

If the administrator wants to delete this record, just click ‘Delete’ and confirm this operation. If he/she wants to modify this record, a new dialog will be showed, which contain all the information about this record, which is showed in Fig.11. According to authority control principle, the administrator can modify all the attributes of this record except score, as we can see the score text box in ‘New Information’ group box is set to read-only. When the administrator has finished modifying, he/she can click ‘Apply’ button to apply modifying.

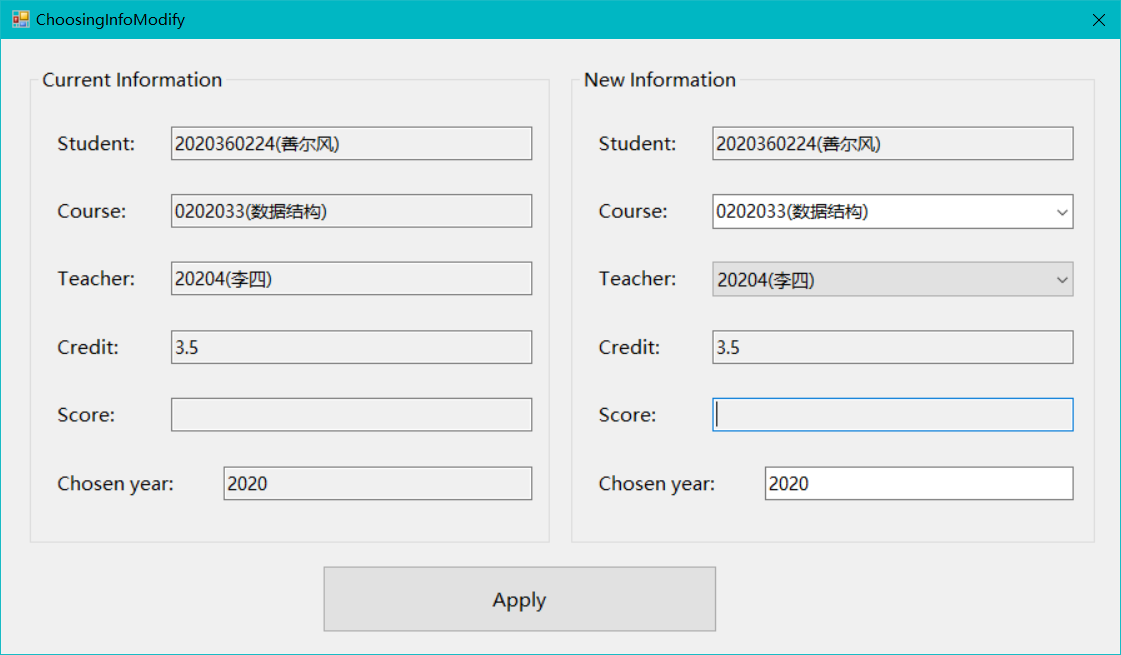


Fig.12 Modifying course choosing information (for administrator)

I.V.V Modify score information

This function is only available for teacher user and each teacher can only modify the grade information of the courses that he/she teaches according to authority control. The teacher can choose a course from the dropdown list and choose correct chosen year, then all the course choosing record related to current condition will be showed like Fig.12.

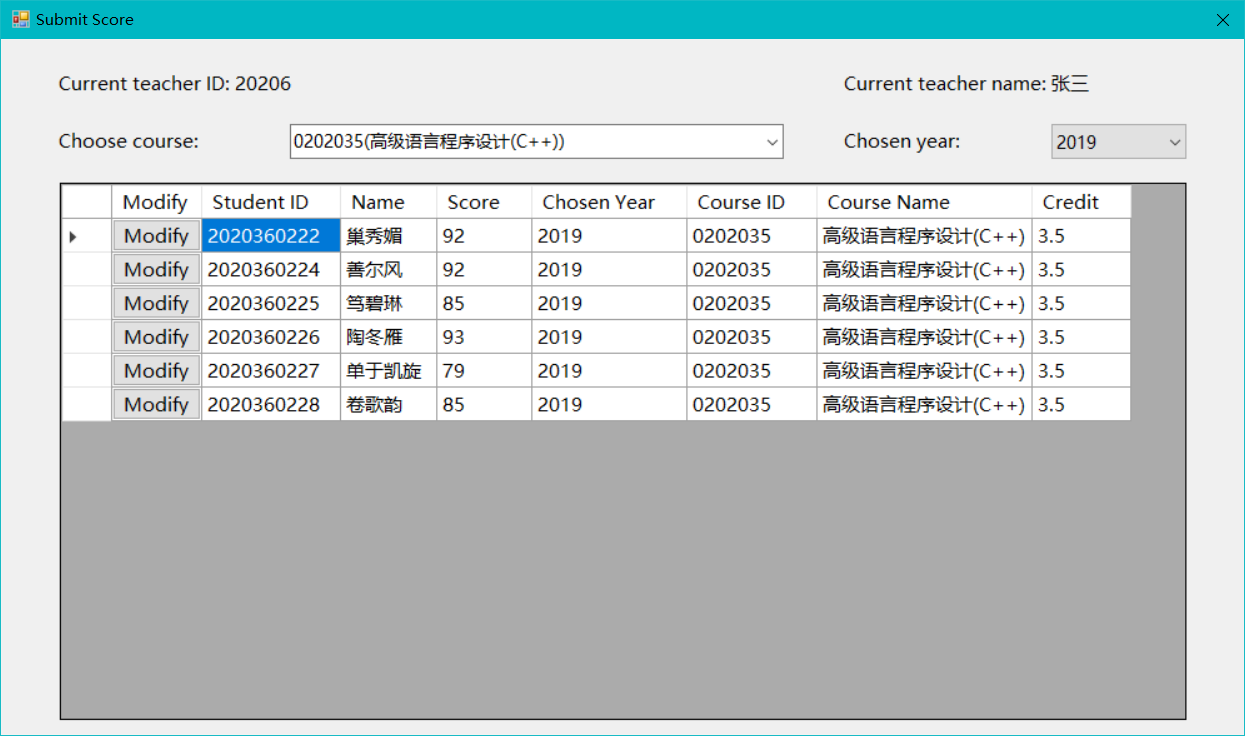


Fig.13 Submit score interface

Then the teacher can modify the score information in the new dialog by clicking ‘Modify’ button of corresponding record, which is showed in Fig.13. As we can see the score text box is changeable so that the teacher can modify score here.

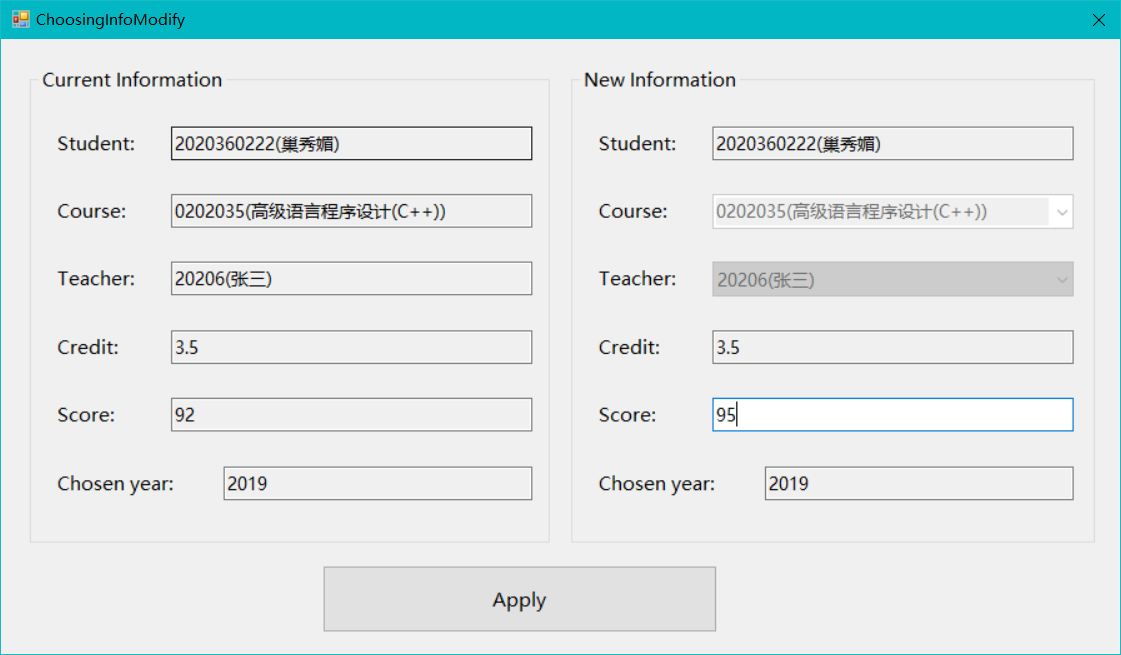


Fig.14 Modifying course score information (for teacher)

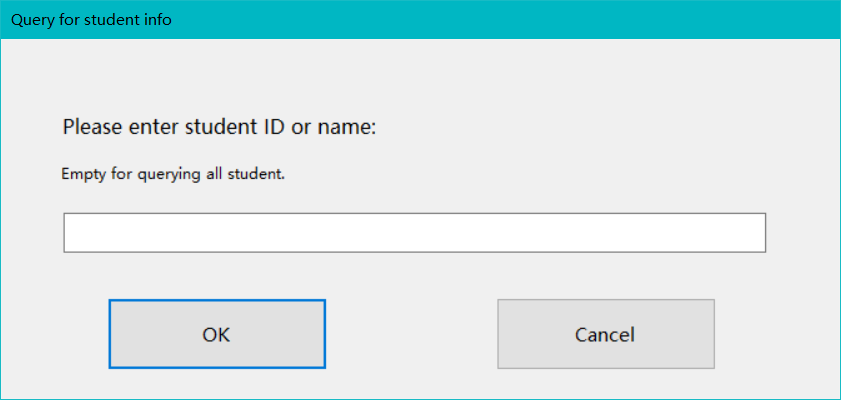
|  |  |
| --- | --- |
| Operation | SQL Command |
| Delete student record | delete from student\_info where id=@id;  delete from user\_data where username=@id; |
| Modify student record | update student\_info set id=@new\_id,name=@new\_name,sex=@new\_sex,entrance\_age=@new\_age,entrance\_year=@new\_year,class=@new\_class where id=@old\_id;update user\_data set username=@new\_id where username=@old\_id; |
| Delete teacher record | delete from teacher\_info where id=@id;delete from course\_info where teacher\_id=@id;delete from user\_data where username=@id; |
| Modify teacher record | update user\_data set username=@new\_id where username=@id;update course\_info set teacher\_id=@new\_id where teacher\_id=@id; |
| Delete a single course record | delete from course\_info where (id=@id and teacher\_id=@teacher\_id); |
| Delete whole course record | delete from course\_info where id=@id; |
| Modify course record | update course\_info set id=@id,name=@name,credit=@credit,grade\_limit=@grade,canceled\_year=@year where id=@old\_id; update course\_info set teacher\_id=@teacher\_id where (id=@id and teacher\_id=@old\_teacher\_id); |
| Delete choosing record | delete from course\_choosing\_info where (course\_id=@course\_id and teacher\_id=@teacher\_id and student\_id=@student\_id and chosen\_year=@year); |
| Modify choosing record (score information) | update course\_choosing\_info set teacher\_id=@new\_teacher\_id,course\_id=@new\_course\_id,chosen\_year=@new\_year,score=@score where (student\_id=@student\_id and teacher\_id=@old\_teacher\_id and course\_id=@old\_course\_id and chosen\_year=@old\_year); |

Table 4. SQL commands for modifying and deleting record

**I.VI Information Query**

I.VI.I Query student/teacher basic information

The student can only query the basic information about himself/herself. And the administrator and teachers can query all student or teacher information by input ID or name, the program will search for all records whose ID or name match the input. If the input is empty, all the records will be listed.



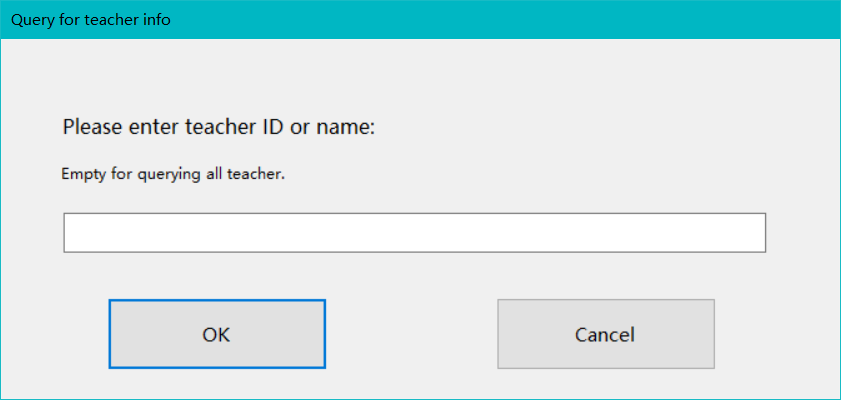


Fig.15 Query for teacher/student basic information interface

I.VI.II Query student score information/student course choosing information

This function can not only query score information but query course choosing information. The student can only query this information about himself/herself. And the administrator and teachers can query all student score information by input student ID or name and course ID or name, which is similar to query student/teacher basic information function.

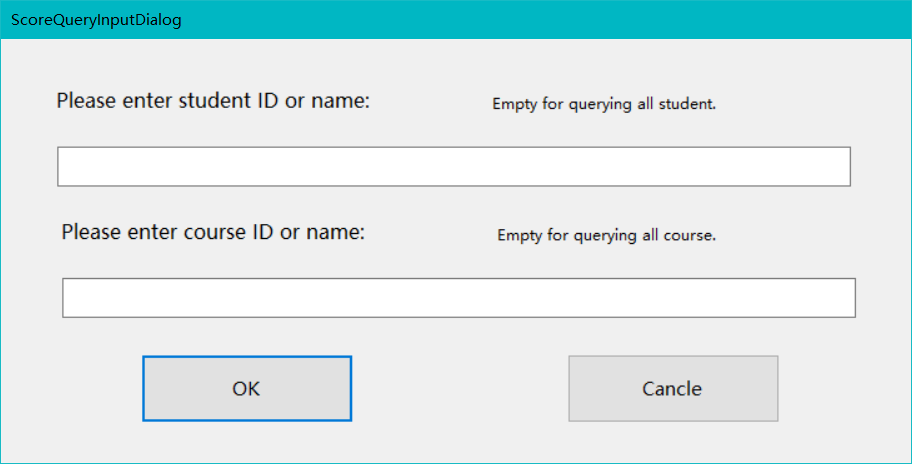


Fig.16 Query for student score information interface

I.VI.III Query course information

There are two ways to query course information. The administrator can use course ID or name to query all the record about this course or use teacher ID or name to query all course record that this teacher teaches.

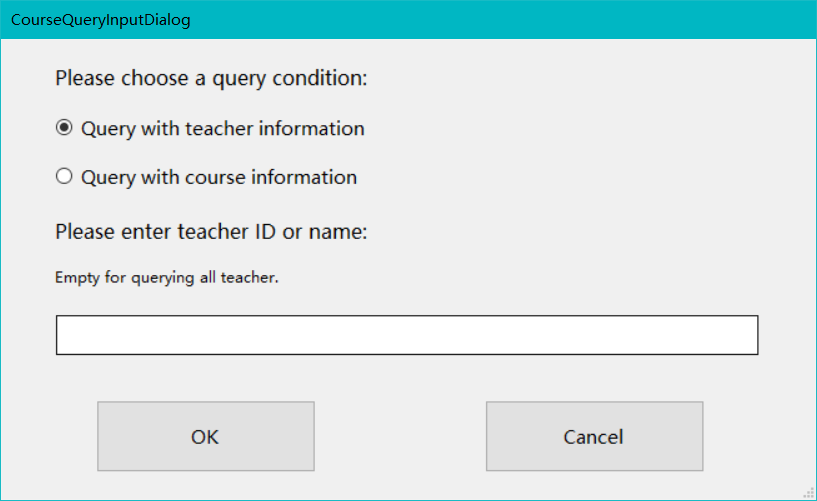


Fig.17 Query course information interface

I.VI.IV Query average score

According to the requirement, average score of a single student, all students, students in the same class or student in the same course can be queried. And all the average score is weighted average score based on course credit.

|  |  |
| --- | --- |
| Operation | SQL Command |
| Query student information | select id as 'Student ID',name as 'Student Name',sex as 'Sex',entrance\_age as 'Entrance Age',entrance\_year as 'Entrance Year',class as 'Class' from student\_info where where(id=@condition or name=@condition); |
| Query student score/course choosing information | select distinct course\_choosing\_info.student\_id as 'Student ID',student\_info.name as 'Student Name',student\_info.class as 'Class',course\_choosing\_info.teacher\_id as 'Teacher ID',teacher\_info.name as 'Teacher name',course\_choosing\_info.course\_id as 'Course ID',course\_info.name as 'Course Name',course\_info.credit as 'Credit',course\_choosing\_info.chosen\_year as 'Chosen Year' from course\_choosing\_info join course\_info join student\_info join teacher\_info on (course\_choosing\_info.course\_id=course\_info.id and course\_choosing\_info.teacher\_id=teacher\_info.id and course\_choosing\_info.student\_id=student\_info.id) where (course\_info.id=@condition or course\_info.name like @name\_condition); |
| Query course information (based on course) | select course\_info.id as 'Course ID',course\_info.name as 'Course Name',course\_info.teacher\_id as 'Teacher ID',teacher\_info.name as 'Teacher Name',course\_info.credit as 'Credit',course\_info.grade\_limit as 'Grade Limit',course\_info.canceled\_year as 'Canceled Year'  from course\_info join teacher\_info on (course\_info.teacher\_id=teacher\_info.id) where (course\_info.id=@condition or course\_info.name like @name\_condition); |
| Query course information (based on teacher) | select course\_info.id as 'Course ID',course\_info.name as 'Course Name',course\_info.teacher\_id as 'Teacher ID',teacher\_info.name as 'Teacher Name',course\_info.credit as 'Credit',course\_info.grade\_limit as 'Grade Limit',course\_info.canceled\_year as 'Canceled Year'  from course\_info join teacher\_info on (course\_info.teacher\_id=teacher\_info.id) where (teacher\_info.id=@condition or teacher\_info.name=@condition); |
| Query teacher information | select id as 'Teacher ID',name as 'Teacher Name' from teacher\_info where(id=@condition or name=@condition); |
| Query average score of a single student | "select student\_id as 'Student ID',name as 'Student Name',round(sum(cs)/sum(credit),2) as 'Weighted average score' from (select distinct student\_id,credit\*score as cs,credit,course\_id,chosen\_year,course\_choosing\_info.teacher\_id from course\_choosing\_info join course\_info on (course\_choosing\_info.course\_id=course\_info.id) where course\_choosing\_info.score is not null) as a join student\_info on (a.student\_id=student\_info.id) where(student\_id=@condition or name=@condition) group by student\_id; |
| Query average score of all student | select round(avg(was),2) as \"Weighted Average Score for All Students\" from (select student\_id as 'Student ID',name as 'Student Name',round(sum(cs)/sum(credit),2) as was from (select distinct student\_id,credit\*score as cs,credit,course\_id,chosen\_year,course\_choosing\_info.teacher\_id from course\_choosing\_info join course\_info on (course\_choosing\_info.course\_id=course\_info.id) where course\_choosing\_info.score is not null) as a join student\_info on (a.student\_id=student\_info.id) group by student\_id) as b; |
| Query average score of students in same class | select class as 'Class',round(avg(was),2) as \"Class Weighted Average Score\" from (select student\_id as 'Student ID',name as 'Student Name',round(sum(cs)/sum(credit),2) as was,class from (select distinct student\_id,credit\*score as cs,credit,course\_id,chosen\_year,course\_choosing\_info.teacher\_id from course\_choosing\_info join course\_info on (course\_choosing\_info.course\_id=course\_info.id) where course\_choosing\_info.score is not null) as a join student\_info on (a.student\_id=student\_info.id) where student\_id in (select student\_info.id from student\_info where class=@class) group by student\_id) as b group by class; |
| Query average score of students in same course | "select course\_id as 'Course ID',course\_info.name as 'Course Name',course\_choosing\_info.teacher\_id as 'Teacher ID',teacher\_info.name as 'Teacher Name',chosen\_year as 'Chosen year',round(avg(score),2) as 'Course Average Score' from course\_choosing\_info join teacher\_info join course\_info on (course\_choosing\_info.course\_id=course\_info.id and course\_choosing\_info.teacher\_id=teacher\_info.id) where course\_id=@cid and course\_choosing\_info.teacher\_id=@tid and chosen\_year=@year and score is not null; |

Table 5. SQL commands for querying record