**OBJECTIVE:**

To make a desktop based Software for the management of a Library Database.

**MOTIVATIONS:**

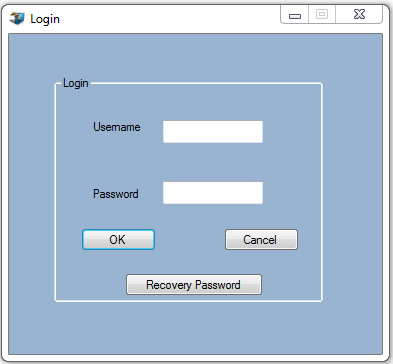
The motivation for doing this project was primarily an interest in undertaking a project in the interesting area of Database Management System. Library database management system is a popular topic for database management project and we wanted to make a desktop based project. From this point of view, we have choose this project. Our main target was to develop our skill in SQL queries and to learn a new language C#.

**DESCRIPTION:**

This System/Software allows a user (acting as an admin) to do certain things regarding a Library.

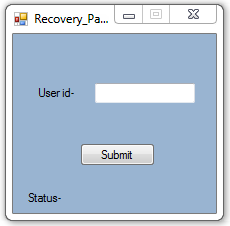
Here is how it works and what are its features:

At first Log In form/window will open.



Where Admin can log in by giving his username and password. After successful login Main\_Menu is accessible. But if Admin forgot his password then he can get his password by “**Recovery Password**” option.

Clicking **“Recovery Password”** option Admin get access in the form “**Recovery Password”**



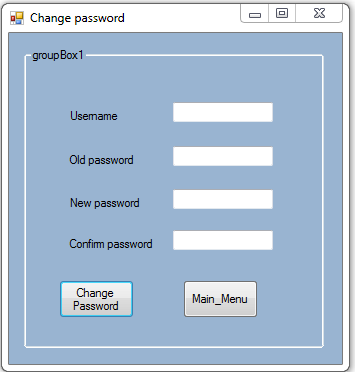
After entering user id and clicking submit button an email with password will be sent to his email address. Admin will get a message “Email sent successfully” if email is sent successfully else an error message will be shown.

From the Main-Menu form admin will have access to all the features of the system.

A brief description of the options in Main Menu are:

**Admin:**

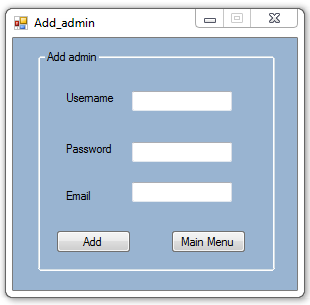
By clicking this toolstripmenu item admin will get another 4 options.

1. Clicking the option “**Change admin password**” will open the form 

Now admin can change password by giving his/her username, old password, new password and again confirming the new password .

If anything goes wrong corresponding message will be shown.

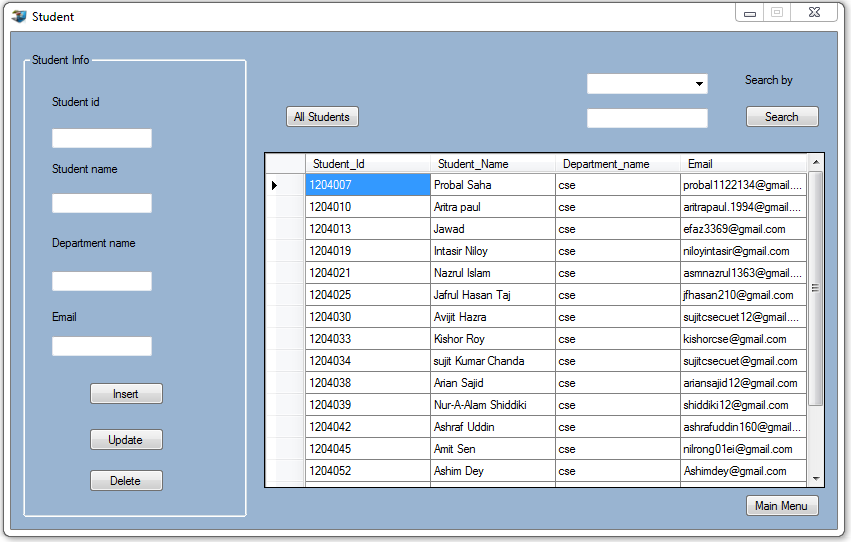
Then he can go back to main menu by clicking “Main\_Menu” button.

1. Clicking the option “**Add another admin**” will open the form 

Now one admin can **add** another admin by giving username name, password and email. As this page will not accessible without LogIn, so registering/adding new admin is secured.

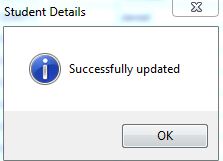
Using “Main Menu” button admin can go back to main menu.

1. Clicking the option “**Update student**” will open the form

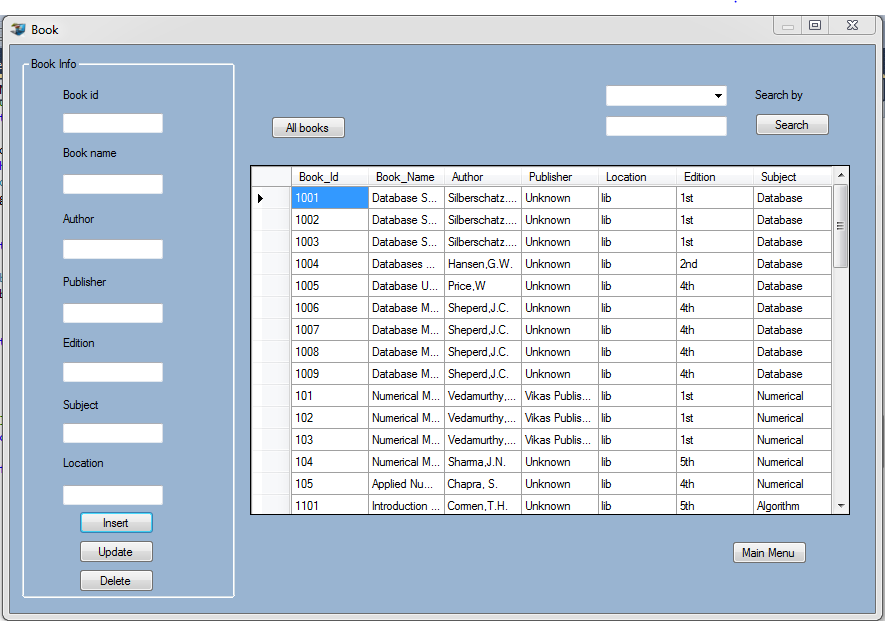


Admin can update student information by clicking respective Student\_id and changing or updating information of student. Finally clicking “Update” option the information will be added to the database.

For a successful update, this message will be shown…

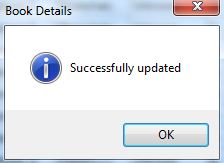


1. Clicking the option “**Update book**” will open the form



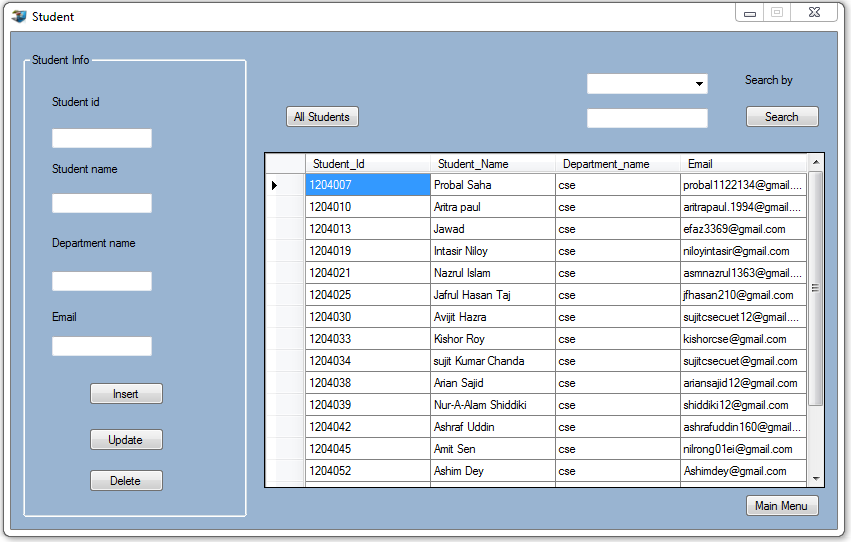
Admin can update book information by clicking respective Book\_id and changing or updating information of book and finally clicking “Update” option.

For a successful update, this message will be shown…



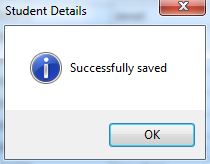
**Registration:** Clicking this option admin will get another 2 options

If admin clicks “**New student entry**” then the form will open

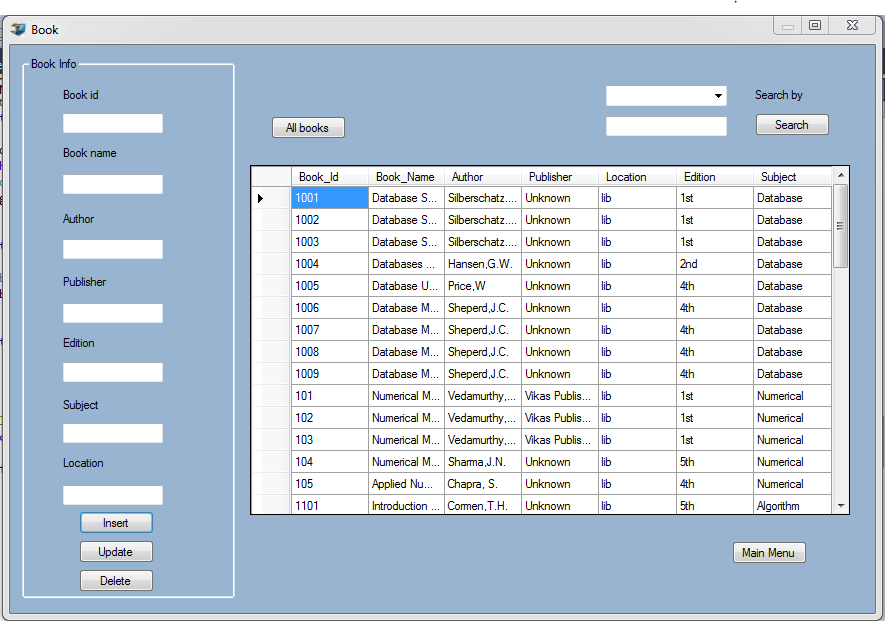


Admin can insert new students in database by this form. Admin must fill up the required inputs and click “**Insert**” option for inserting new students in the database.

If everything is ok then this message will be shown, else corresponding error will be reported.

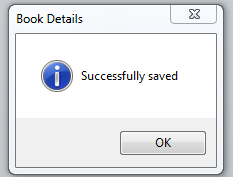


If admin clicks the “**New book entry**” then he will get the form

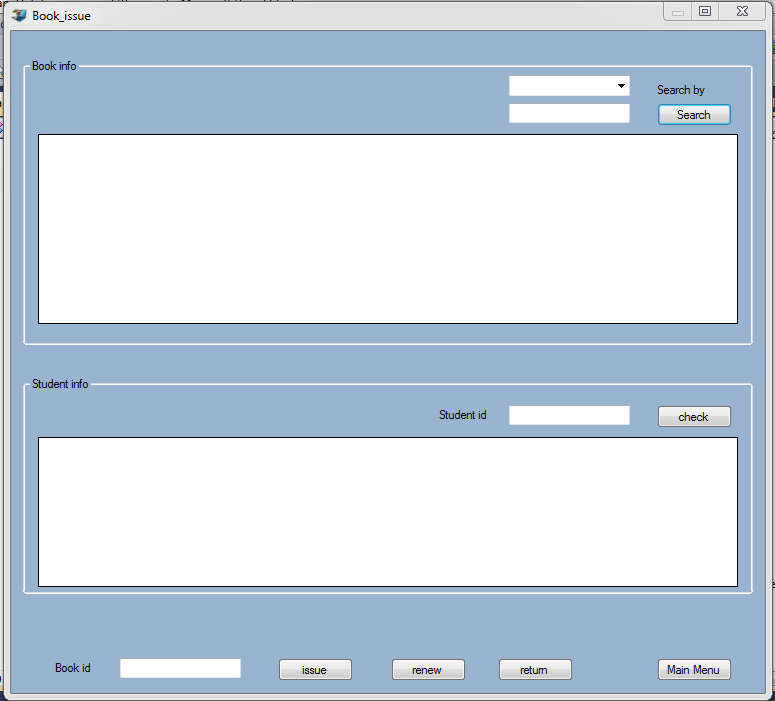


Admin can insert new books in database by this form. Admin must fill up the required inputs and click “Insert” option for inserting new books in the database.

If everything is ok then this a message will be shown to confirm Insertion, else corresponding error will be reported.



**Issue Book:** Clicking this option admin will get the form



For **Issue** a book , admin must follow the steps:

1. Admin search the book by ‘Book name’ or ‘Author’.
2. If the location of the book is ‘lib’ then the book is ready for issue else he can’t issue that book.
3. If book is ready for issuing then admin must check the student id to know how many books he have already borrowed from the library.
4. If student does not borrow more than 2 books then admin can issue the book.
5. For issuing book at first admin click the Book-Id then click “issue” Button.

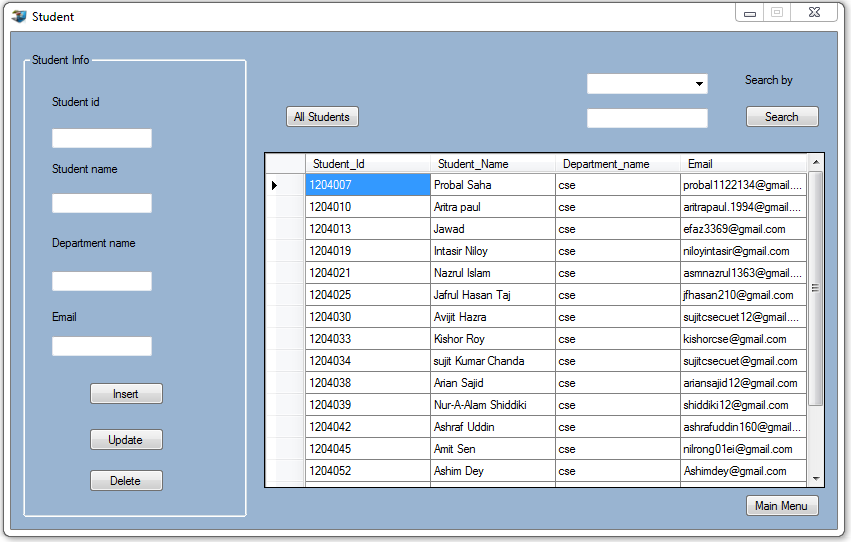
To **Renew** or **Return** a book, admin must follow the steps:

1. For renewing a book, admin must go to student id and click the book and click the “renew” option.
2. For returning a book, admin must go to student id and click the book and click the “return” option.

**Database:**

Clicking this option admin will get 2 options

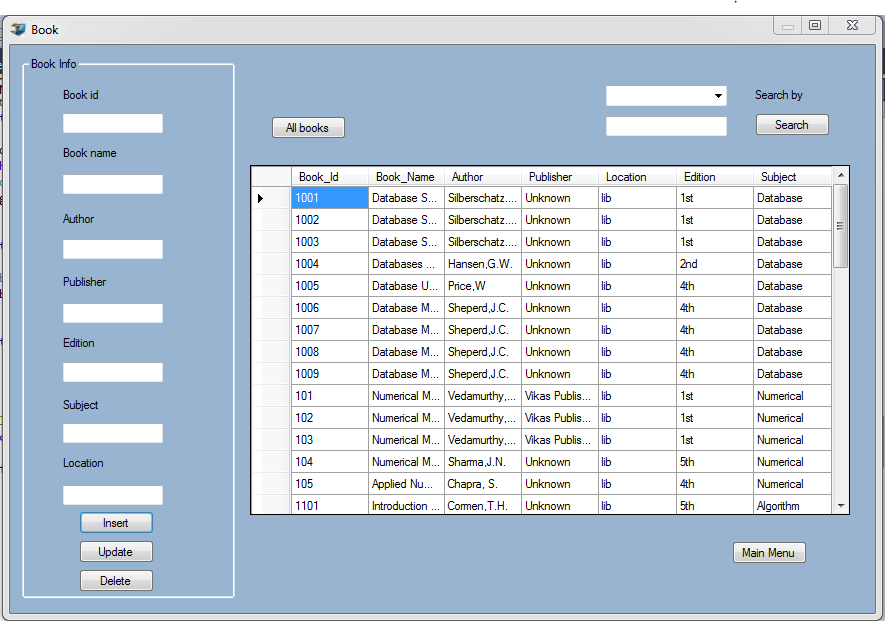
If he/she clicks “**Student**” option then admin will get the form



In this form admin can insert student, update student information, delete any student from database, search specific student by student id, student name, department name.

This form/window also allows an admin to control over the Student database.

Clicking “**Book**” option admin will get the form

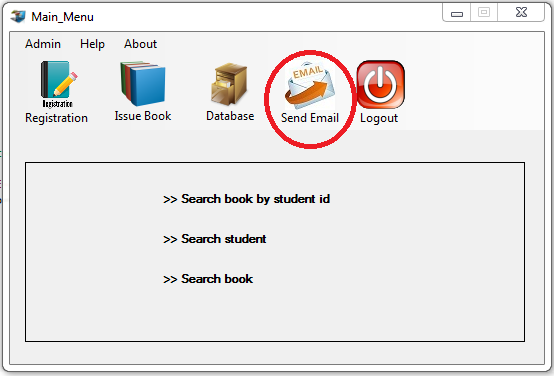


In this form admin can insert book, update book information, delete any book from database, search specific book by book id, book name, author, subject.

This form/window also allows an admin to control over the Book database.

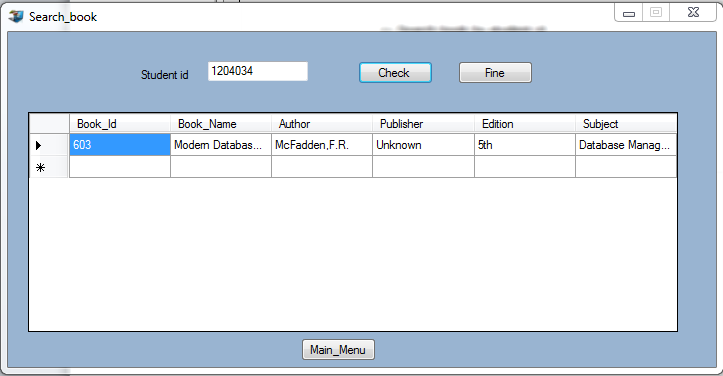
**Send Email:**

By clicking this option admin can send email to students. Email will be sent to a student if he/she has at least a book that have passed 30 days after its last renewal or issue and has not returned yet.



**Search book by student id:**

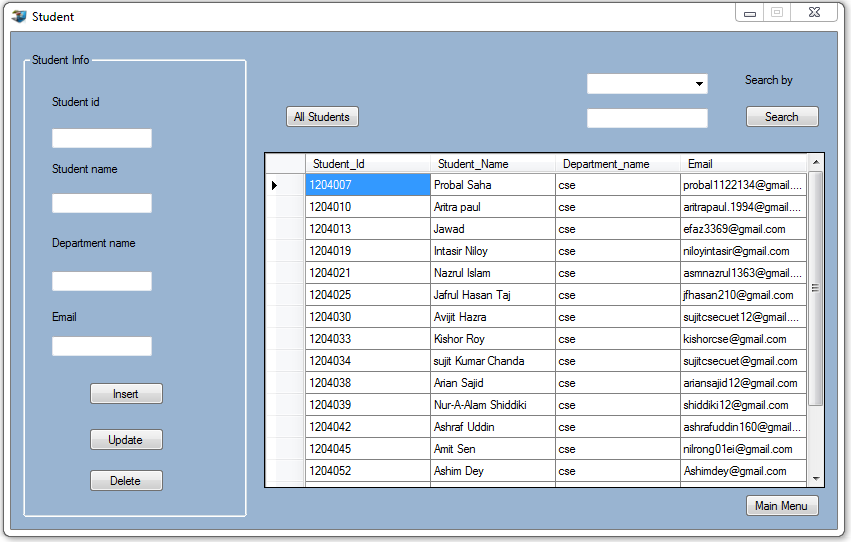
This option will open the form



Now admin can see the books by student id and find the fine of a student in this form. Give the student id as input and click “**Check**” to see books that the student borrowed or click “**Fine**” to get the fine of that student(a student will be fined 1 taka per book per day after 30 days from issue date).

**Search student:**

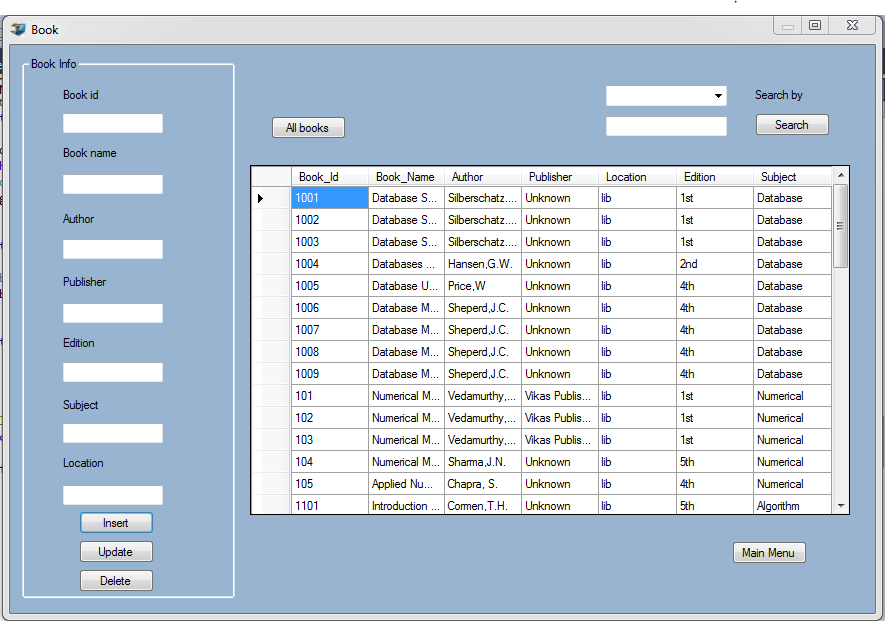
Clicking this option admin will get the form

****

Now admin can search student by student id, student name, department name.

**Search book:**

Clicking this option admin will get the form



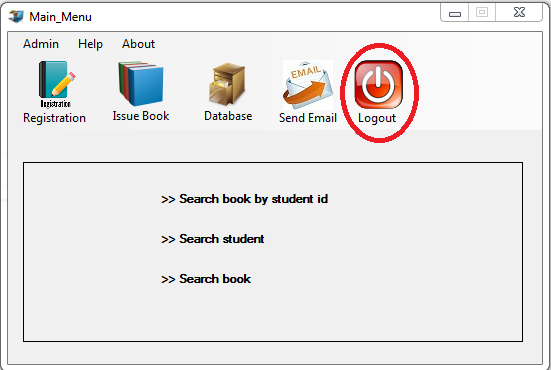
Now admin can search book by Book id, Book name, author, subject.

**Help & About:**

In “Help” option admin can know about how to use this System/Software.

In “About” option admin will know about the System/Software.

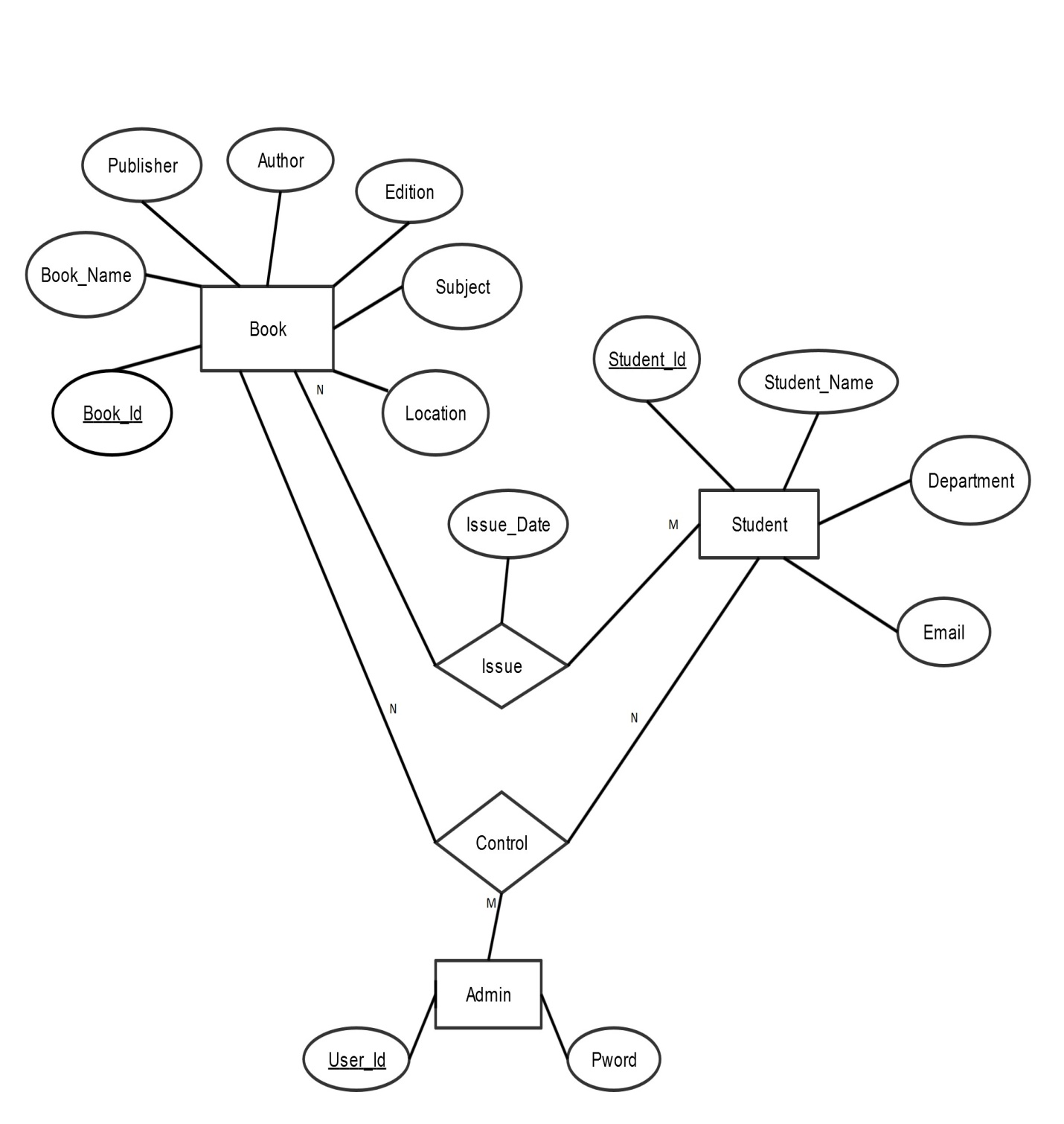
**Logout:** By clicking this option admin can log out from the system.



This Software don’t allow maximization though it can be minimized. The “Close Button” will not close this system immediately. It will ask for the confirmation.

But the “**Cancel**” button at Login form/window will close the system immediately.

**ER DIAGRAM:**



**RELATIONAL MODEL:**

Admin

|  |  |
| --- | --- |
| User\_Id | Pword |

Admincontrol

|  |  |  |
| --- | --- | --- |
| User\_Id | Book\_Id | Student\_Id |

Book

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Book\_Id | Book\_Name | Author | Publisher | Edition | Subject | Location |

Student

|  |  |  |  |
| --- | --- | --- | --- |
| Student\_Id | Student\_Name | Department | Email |

Studentbook

|  |  |  |
| --- | --- | --- |
| Student\_Id | Book\_Id | Issue\_Date |

**NORMALIZATION:**

First normal form: A relation is in first normal form (1NF) if it contains no multivalued attributes.

Second normal form: Any partial functional dependencies have been removed.

Third normal form: A relation is in third normal form (3NF) if it is in second normal form and has no transitive dependencies.

A transitive dependency in a relation is a functional dependency between two (or more) non key attributes.

Now according to definition normalize the table Book

Book

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Book\_Id | Book\_Name | Author | Publisher | Edition | Subject | Location |

Here, the table Book has no partial dependency. So we have to convert it in 3rd normal form.

The table Book is in 2nd normal form and it has no transitive dependencies so table is in 3rd normal form.

Now according to definition normalize the table Student

Student

|  |  |  |  |
| --- | --- | --- | --- |
| Student\_Id | Student\_Name | Department | Email |

Here, the table Student has no partial dependency. So we have to convert it in 3rd normal form.

The table Student is in 2nd normal form and it has no transitive dependencies so table is in 3rd normal form.

Now according to definition normalize the table Admin

Admin

|  |  |
| --- | --- |
| User\_Id | Pword |

Here, the table Admin has no partial dependency. So we have to convert it in 3rd normal form.

The table Admin is in 2nd normal form and it has no transitive dependencies so table is in 3rd normal form.

Now according to definition normalize the table Admincontrol

Admincontrol

|  |  |  |
| --- | --- | --- |
| User\_Id | Book\_Id | Student\_Id |

Here, the table Admincontrol has no partial dependency. So we have to convert it in 3rd normal form.

The table Admincontrol is in 2nd normal form and it has no transitive dependencies so table is in 3rd normal form.

Now according to definition normalize the table Studentbook

Studentbook

|  |  |  |
| --- | --- | --- |
| Student\_Id | Book\_Id | Issue\_Date |

Here, the table Studentbook has no partial dependency. So we have to convert it in 3rd normal form.

The table Studentbook is in 2nd normal form and it has no transitive dependencies so table is in 3rd normal form.

**QUERY:**

We used different types of queries in this project such as insert, update, delete, search, nested sub queries, aggregate functions.

Details about the queries:

**Insert query:** This query is used for insert any data in the database. Example- An insert query that is used in this project is

"INSERT INTO Student(Student\_Id, Student\_Name, Department\_Name, Email) VALUES (@d1,@d2,@d3,@d4)";

where,

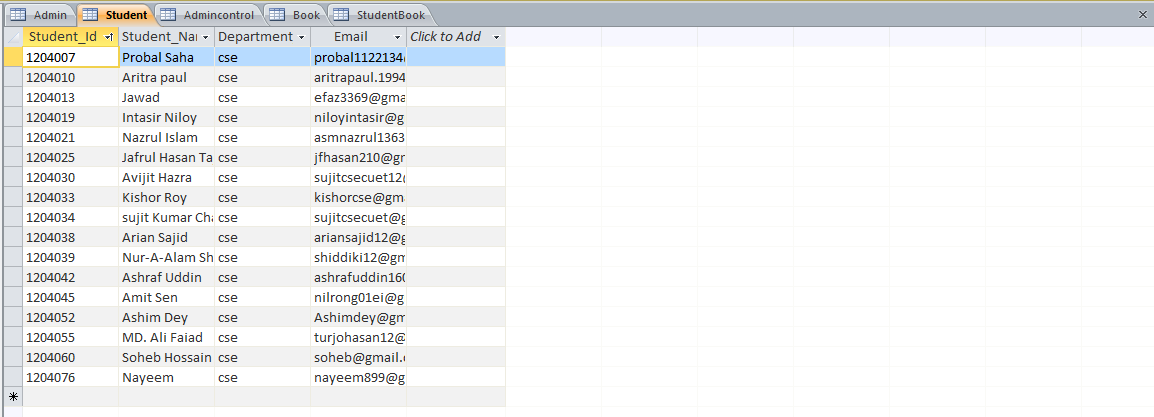
d1=textstudentid.Text

d2=textstudentname.Text

d3=textdepartmentname.Text

d4=textemail.Text

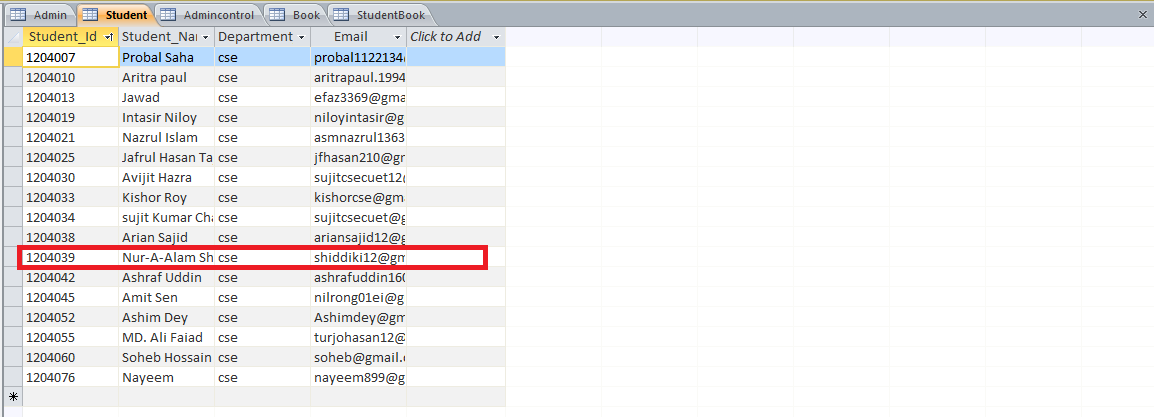
Before using the query the student table was empty that means there was no data but after using the query the table is like



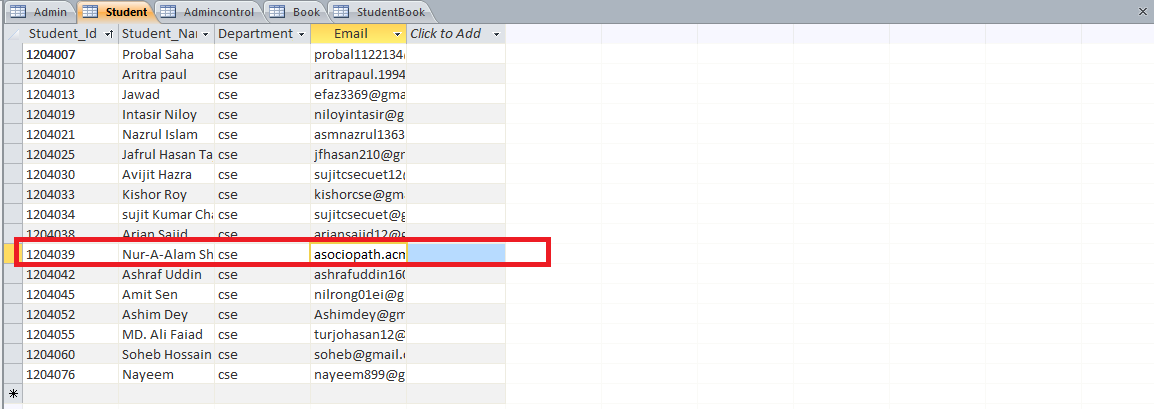
**Update query:** This query is used for update any data in the database. Example- An update query that is used in this project is

"UPDATE Student SET [Student\_Name]= '" + textstudentname.Text + "',[Department\_Name]= '" + textdepartmentname.Text + "',[Email]='" +textemail.Text+ "'where [Student\_Id]= '"+textstudentid.Text+"'";

Before using this query,table is like



After using this query, table is like

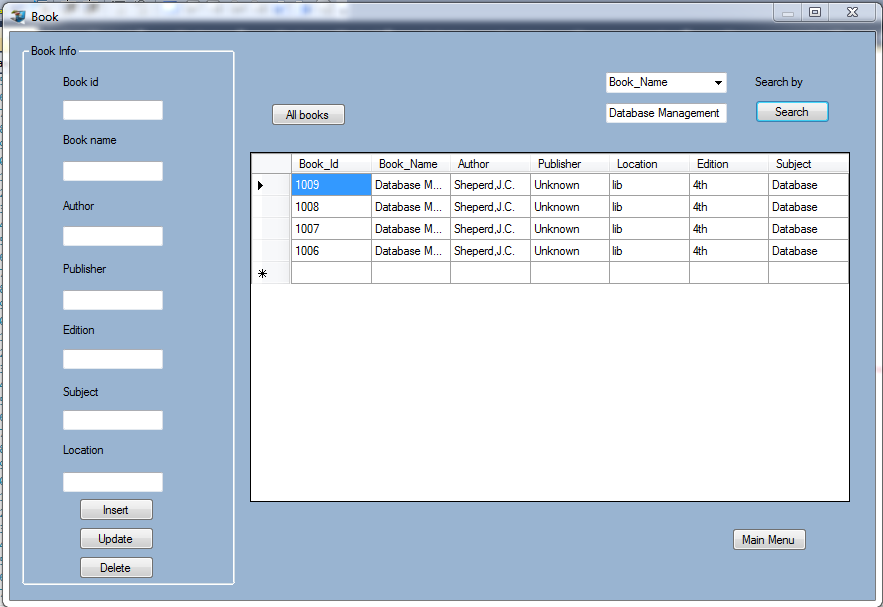


Before using the query in the red marked point there was an email id(shiddiki12@gmail.com). But after using the query in the red marked point there is an email id(asociopath.acm@gmail.com). Here email id is the updated information.

**Search query:** This query is used for search any data from database. Example- A search query that is used in this project is

"SELECT (Book\_Id)as [Book\_Id],(Book\_Name) as [Book\_Name],(Author) as [Author],(Publisher)as[Publisher],(Location) as [Location],(Edition) as [Edition],(Subject) as [Subject] from Book where Book\_Name like '" + textsearch.Text + "%'order by Book\_Name";

After using this query we get



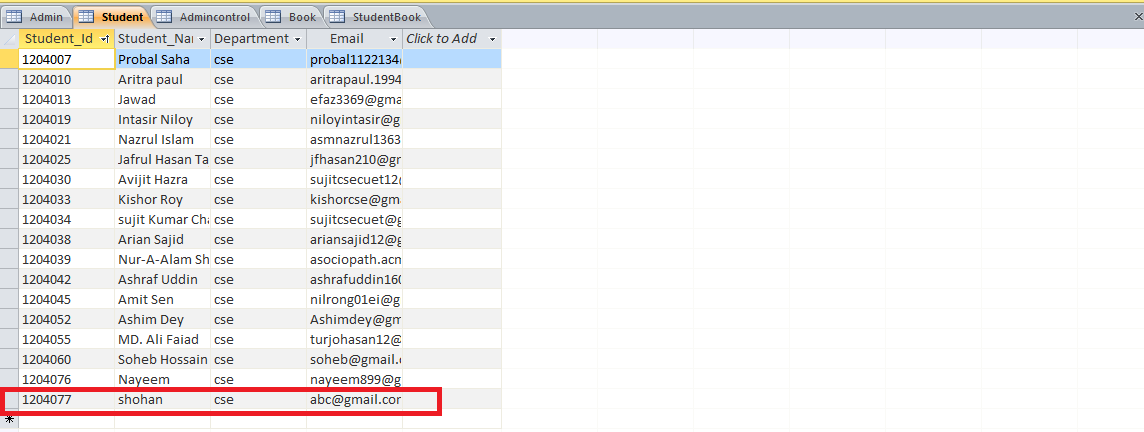
Here we search the books named Database Management and get this result.

**Delete Query:** This query is used for delete any data from database. Example- A delete query that is used in this project is

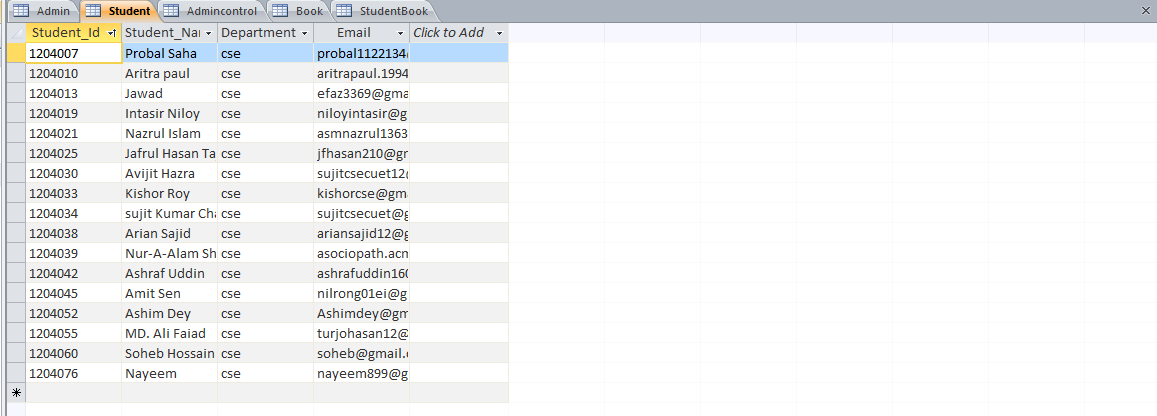
"delete from Student where Student\_Id=@DELETE1;";

command.Parameters["@DELETE1"].Value = textstudentid.Text;

Before using this query, table is like



After using this query, table is like



Before using delete query red marked information were present in the student table but after using the delete query that information are removed from table.

**Sub query:** We used this query for getting information from two different tables. The sub query that we used is

"Select Email From Student where Student\_Id in(select Student\_Id FROM Studentbook where DateDiff('d',[Issue\_Date], NOW())=31) ";

using this query we get the email id of students from student table where the students are those whose book issue date and system date difference is 31 in studentbook table.

**Aggregate function:** There are various aggregate functions such as avg, min, max, sum, count. Here we used “sum” function.

"select sum(Fine) from Studentbook where Student\_Id='" + textsearch.Text + "'";

By this query we get the total fine of a student.

**LIMITATIONS:**

* 1. This system can send email only in gmail address. So admin must be careful inserting email address.
  2. It is preferable to use mouse than keyboard.