Islamic University of Technology (IUT)

CSE 4308: Database Management Systems Lab

Lab Report # 1

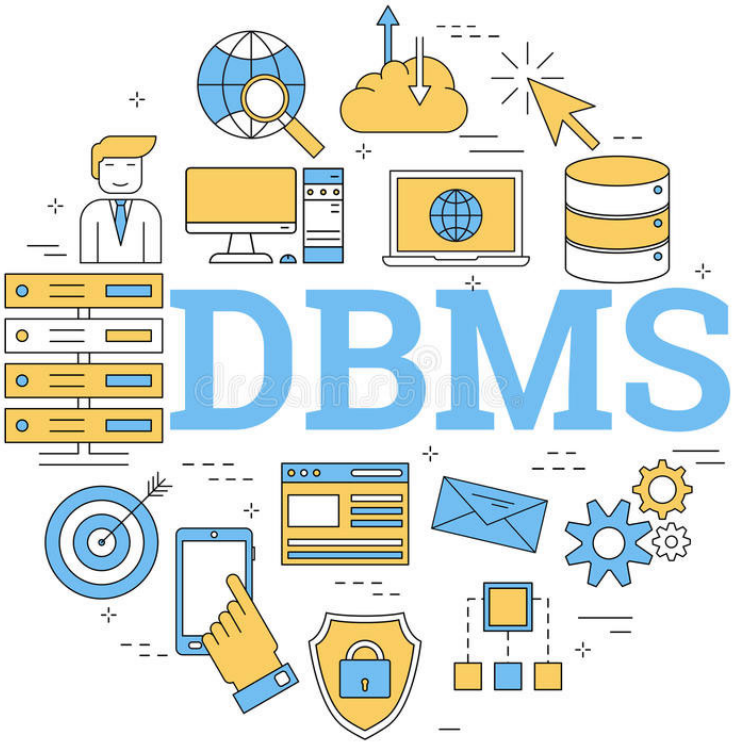
Submitted to:

Md. Bakhtiar Hasan,

Lecturer, CSE.

Submitted by:

M M Nazmul Hossain
ID 200042118
CSE (SWE)



Submission Date:

17.08.2022

Introduction

In our first database management systems lab, the goal was to complete a total of 3 (three) tasks.

The Lab provided us with two files, studentInfo.txt, and grades.txt. The file studentInfo.txt was a database of:

Student ID
Student Name
Batch
Department
Blood Group
Semester

The grades.txt was another database of:

Student ID
GPA
Semester

We had to do the following things:

1. Determine the Average GPA of all Students
2. Write GPA information of a student to the file following certain restrictions after taking Student ID, GPA, and Semester as input. If the input is invalid, discard it.
3. Take the student ID as input and display the name, the minimum GPA of that student, and the Semester of that GPA.

We could use any language we wanted to accomplish the given tasks.

Method

The Language I chose to tackle these tasks was C#, as I was already familiar with the concept of reading and writing in a file in C#. After opening the files and identifying the columns, I noted that the studentInfo file had 6 (six) columns and the grades file had 3 (three) columns. The columns were separated by semi-colons (;). The attributes of the columns were as mentioned in the introduction. I decided to compile all three tasks in the same file. With that in mind, I began coding.

The Code

GPA Class

```
using System;
using System.Collections.Generic;
using System.Text;

namespace DBMS_Lab_1
{
    class Gpa
    {
        public string Semester;
        public string GpA;
        public string studentID;
    }
}
```

studentInfo Class

```
using System;
using System.Collections.Generic;
using System.Text;

namespace DBMS_Lab_1
{
    class studentInfo
    {
        public string ID;
        public string Name;
        public string Batch;
        public string Department;
        public string BloodGroup;
        public string Semester;
        public List<Gpa> GPA = new List<Gpa>();
    }
}
```

main Function

```
using System;
using System.Collections.Generic;
using System.IO;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Globalization;

namespace DBMS_Lab_1
{
    class Program
    {
        static void Main(string[] args)
```

```
        {
            string stdInfo = "F:\\DBMS Lab\\Lab 1\\studentInfo.txt";
            string grades = "F:\\DBMS Lab\\Lab 1\\grades.txt";
            List<studentInfo> stdntInfo = new List<studentInfo>();
            double studentCount = 0;
            using (var reader = new StreamReader(@"F:\\DBMS Lab\\Lab
1\\studentInfo.txt"))
            {
                while (!reader.EndOfStream)
                {
                    var line = reader.ReadLine();
                    var values = line.Split(';');
                    studentInfo student = new studentInfo();
                    student.ID = values[0];
                    student.Name = values[1];
                    student.Semester = values[2];
                    student.BloodGroup = values[3];
                    student.Department = values[4];
                    stdntInfo.Add(student);

                }
            }
            NumberFormatInfo setPrecision = new NumberFormatInfo();
            setPrecision.NumberDecimalDigits = 2;

            double GpaSum = 0;
            using (var read = new StreamReader(@"F:\\DBMS Lab\\Lab
1\\grades.txt"))
            {
                while(!read.EndOfStream)
                {
                    var line = read.ReadLine();
                    var values = line.Split(';');
                    foreach(studentInfo std in stdntInfo)
                    {
                        if(std.ID==values[0])
                        {
                            Gpa gpa = new Gpa();
                            gpa.GpA = values[1];
                            gpa.Semester = values[2];
                            gpa.studentID = values[0];
                            std.GPA.Add(gpa);
                            GpaSum = GpaSum + Convert.ToDouble(gpa.GpA);
                            studentCount++;
                        }
```

```
                    }
                }
            }
            double averageGpa = GpaSum / studentCount;
            Console.WriteLine("The Average GPA is : " +
averageGpa.ToString("N",setPrecision)+"."+"\n");

            string studentID1;
            string GPA1;
            string Semester1;
            Console.WriteLine("This is Task 2. Input Student GPA.");
            Console.WriteLine("Input Student ID: ");
            studentID1 = Console.ReadLine();
            Console.WriteLine("Input Student GPA: ");
            GPA1 = Console.ReadLine();
            Console.WriteLine("Input Student Semester: ");
            Semester1 = Console.ReadLine();
            bool flag = false;
            int count = 0;
            foreach (studentInfo std in stdntInfo)
            {
                if(std.ID == studentID1)
                    count++;
            }
            if(count==0)
            {
                flag = true;
                Console.WriteLine("This student Id doesn't exist in the
Database");
            }
            foreach (studentInfo std in stdntInfo)
            {
                if (std.ID == studentID1)
                {
                    foreach (Gpa gpa in std.GPA)
                    {
                        if (Semester1 == gpa.Semester)
                        {
                            flag = true;
                        }
                    }
                    if (flag == true)
                    {
                        Console.WriteLine("The GPA Information for this semester
Already Exists");
```

```
                        break;
                    }
                }
                if (Convert.ToDouble(GPA1) > 4.00 || Convert.ToDouble(GPA1) <=
2.50)
                {
                    flag = true;
                    Console.WriteLine("This GPA is not Legal.");
                    break;
                }
            }

            if (flag == false)
            {
                using (StreamWriter sr = File.AppendText(grades))
                {
                    string s = studentID1 + ";" + GPA1 + ";" + Semester1;
                    sr.WriteLine(s);
                }
                foreach (studentInfo std in stdntInfo){
                   if(std.ID == studentID1)
                    {
                        Gpa gp = new Gpa();
                        gp.studentID = studentID1;
                        gp.GpA = GPA1;
                        gp.Semester = Semester1;
                        std.GPA.Add(gp);
                    }
                }
                Console.WriteLine("GPA Added Successfully. Moving on to Task
3.");
            }
            else
            {
                Console.WriteLine("Could Not Add GPA. Moving on to Task 3");
            }
            Console.WriteLine("This is Task 3. Input a Student ID: ");
            string StudentID2=Console.ReadLine();
            bool task3flag = false;
            foreach(studentInfo std in stdntInfo)
            {
                if(std.ID == StudentID2)
                {
                    Console.WriteLine(std.Name);
                    task3flag = true;
```

```
                    double min = 5.00;
                    string minSem = "99";
                    foreach(Gpa gpa in std.GPA)
                    {
                        if(min>Convert.ToDouble(gpa.GpA))
                        {
                            min = Convert.ToDouble(gpa.GpA);
                            minSem = gpa.Semester;
                        }
                    }
                    Console.WriteLine("The Minimum GPA was " +
Convert.ToString(min) + " achieved in the " + minSem + " Semester.");
                }
            }
            if(task3flag==false)
            {
                Console.WriteLine("That Student ID is not in the Database");
            }

            Console.ReadLine();

        }
    }
}
```

Explanation

First, I created a GPA class that would store id, result, and semester. Then I created a studentInfo class which contained attributes for student ID student Name, Batch, Department, Blood Group, and Semester. There is also a list of GPAs that will store the GPA of that student in different semesters. And finally, I created a list of students where all the student information is saved. I did this to bring all the information in the two files to the same place to attain a level of freedom while solving the tasks.

I stored the path of the files in a string and used StreamReader to read the files alongside a foreach loop. Using semi-colon (;) as the split value, the studentInfo file generates five (5) values, and the grades file into three (3) values. I then store them accordingly in different instances of studentInfo and GPA accordingly. I matched the student ID with previously saved student information to save all of their GPA in the different semesters in the list allocated to their instance. This process will continue until the EndOfStream of each file. Thus, I created a working database of students with their results in C#, and the hard part is already over.

Task 1

Task 1 asked us to calculate the average GPA of the students. So, while reading the grades file using the foreach loop, I used a gradeSUM double variable to add all the GPAs in the file. I also used a count variable to count the number of entries to the file. After that, I just divided the gradeSUM variable with the count variable in an answer variable. But here I encountered a problem. The answer variable, being a double, was displaying up to six (6) decimal points. To solve that, I used the built-in Number Format Info class to set the precision of the answer variable to 2 decimal places. After that, I just had to display the answer variable using Console.WriteLine and Task 1 were complete.

Task 2

I divided Task 2 into different parts. First, I took Student ID, GPA, and Semester as input and stored them in variables. First, I wanted to check if the student ID existed in the database. I failed to check for this edge case during the Lab. So, I ran a foreach loop simply to go through all the students and see if the student ID matches with any. If there are no matches, the count variable does not increase throughout the loop, which sets an error flag as "true." After that, I used the foreach loop to traverse through the studentInfo list and saw if the input Semester already has a GPA saved for that particular student. Then I checked if the GPA was within the legal range of 2.50 and 4.00. If none of these conditions are met, the flag for an error becomes true, and I wrote a custom error message for each edge case. If the error flag is "true," the executable file would then show a prompt that the GPA could not be added to the file and move on to the next

Task. If the flag is not "true", we Append a new line of text to the grades folder, which includes the input student ID, GPA and Semester separated by semi-colons (;). Then I added the newly added information to the grades file inside the studentInfo list. After that Task 2 is complete.

Task 3

For Task 3, I had to take Student Id as input and then Display the name of that student, the minimum GPA, and the Semester of that GPA for that student. To accomplish this task, I took the student ID as input. I ran a foreach loop and checked if the student ID exists in the database. If it doesn't, an error message would show the student ID doesn't exist in the database. If a matching student ID is found, then the name of the student matching the student ID would be printed. I ran another foreach loop which would go through the list of GPAs stored in that instance of studentInfo. In that foreach loop, I would check all the GPAs and save the minimum GPA and the Semester of that GPA in two separate variables. After that, I would print the minimum GPA and semester of that GPA. An edge case I didn't consider, was what-if he had achieved his minimum GPA in multiple semesters. I assume I can do that using an array or list to store the semesters instead of a variable. And with that, all three tasks are completed and the program closes.

Conclusion

In order to save time and not have to create the database, again and again, I chose to complete all the tasks in the same file. If needed, I could create three separate solution files which would carry out all three (3) tasks separately, or I could run an infinite loop so that all three (3) tasks are carried out continuously until the user decides to close the program. All in all, this was a fun Lab that helped me recollect what I had learned in the previous Labs.