Zusammenfassung der Diplomarbeit
Abstract of diploma thesis

Fachbereich Department	: :	**Electrical Engineering and Computer Science**
Studiengang University course	: :	**Information Technology**
Thema Subject	: :	**Programming of a barcode driven personal counter on the basis of an SQL database in MS Studio Visual CSharp**
Zusammenfassung Abstract	: :	

Attendance check is one thing that bothers professors. Traditional approaches such as roll call are often time-consuming (inefficient). This thesis therefore focuses on solving the problem by using barcode which is available on every student card.

The program "BarcodeScanner" based on the programming language CSharp has been developed for this purpose. It enables to check the attendance of each student by scanning their student cards and store the relating data to local database served by MySQL Server. Professors can additionally judge students' performance by their attendance.

Thesis also contains several theoretical studies. As a key to accessing database in applications, overview of ADO.NET interface is introduced. Then brief comparison of three databases, i.e. MySQL, MS SQL Server and PostgreSQL, is made (by discussing licensing schema, primary features and the advantages and disadvantages of them). Finally data control with SQL commands is illustrated with several detailed examples for beginners.

Verfasser Author	: :	**Yannan Shen**
Betreuender Professor/in Attending Professor	: :	**Prof. Dr.-Ing., Dipl.-Ing. Jörg Bayerlein**
WS / SS	:	**SS2010**

Content

Introduction

Motivation and design goal

A good performance of students is not only judged by their final score, but also the number of attendance. Professors have to take the responsibility of counting students' number before they start the lecture. That is a tough job, because in universities one lecture might be chosen by more than 50 students. And it seems to be impossible for professors to call the roll in each lecture. So what I have done is helping professors to count attendance of individual students accurately and quickly start their lecture. My design goal could be summarized into one sentence, easier usage and quicker performance but accurate counting.

Program called "BarcodeScanner" is based on Microsoft Visual C Sharp with connection to MySQL database. It could efficiently count attendance of individual students by scanning their student cards. Professors are allowed to import/export the student records from/into a text file. And with two modes, the security issue could be ensured. Moreover, professors can choose the minimum requirement of attendance, so that students' performance could be judged.

Thesis structure

To make my whole thesis more understandable for beginners, Chapter I presents some general knowledge about C Sharp, Database, and ODBC data source.

Then based on the general information Chapter 2 I illustrate concrete steps of installing necessary software including MySQL Server and its Connector/ODBC driver. To run my program properly, installing these software are essential.

Following Chapter 2, the next chapter, Chapter 3 focuses on programming mainly including its usage, detailed methods used to achieve program functions and the instruction of MySQL Workbench as an addition.

Since topic of this thesis is related to the communication between database and C Sharp application, Chapter 4 introduces an overview of ADO.NET interface, which is the most crucial part of building connection. It will be a supplement to Chapter 3.

Subsequently, Chapter 5 makes a comparison among MySQL, MS (Microsoft) SQL Server and PostgreSQL, three popular database servers/systems in today's PC world. The comparison includes their features and performance under different application environments.

To work with databases, SQL language is the most basic knowledge. It is a standard query language supported by almost every database services. Therefore, Chapter 6 presents a simple tutorial about how to write basic SQL statements including examples. It will be especially helpful to beginners.

Chapter 1 Basic knowledge

1.1 What is C#?

On June 26th 2000 the conception of C# (Pronounced C Sharp) war brought up by Microsoft, followed by version1.0 on Feb 13th 2002. C# came from Microsoft C and C++. It is the evolution of them. It contains all advantages that C++ contains, as well as, its own strong points. It is totally an object-oriented language.

"Microsoft C# is a new programming language designed for building a wide range of enterprise applications that run on the .NET Framework. C# is an elegant, simple, type-safe, object-oriented language that allows enterprise programmers to build a breadth of applications. "

----MSDN, Visual C# Language [1]

C# is using managed code, which means it requires the .NET Common Language Runtime (CLR) to execute.

"The CLR is managing memory, performing garbage collection, handling exceptions, and providing many more services that you, as a developer, don't have to write code for."

----C# station, what is C# [2]

C# is a strong language. Programmers can apply it to lots of different applications. For example, C# can be used to write a web page or a desktop application. And if programmers want to access data source, there is ADO.NET interface. My program is a desktop application with access to local data base.

1.2 Why use database?

Nowadays, over a million databases exist in the world. They are storing all kinds of information.

"A database consists of an organized collection of data for one or more multiple uses."

----Wiki, Database [3]

A database is divided into tables. And each table is further divided into rows and columns; these columns store the actual information. A database can handle very large numbers of records efficiently. Information stored as a database can be found easily and quickly, because databases support for such control functions like filtering or sorting.

Furthermore, the updating of data rapidly is very important. Data collected in a database could be much more convenient and quicker to update and manipulate. By using database, the

stored data can be applied into more other applications. And the same database can be accessed by different applications at the same time.

Moreover, Database has a better data protection and security system. For sure, Data collected in a database can be kept much longer than other type, for example, paper. [4]

My program retrieves all records such as personal information of students from databases. That is the easier way for me to apply them into a C# application and to manipulate them. These records can be safely kept in computer for a long time.

1.3 What is ODBC data source?

ODBC stands for Open Database Connectivity. It is a standard database access method, which was developed by the SQL Access group in 1992. [5]

ODBC adds a layer between databases server and applications, called driver. With this driver, programmers can access any data in their applications, whatever these data are handled by MySQL server, Microsoft SQL server or PostgreSQL server. ODBC driver is able to translate data queries from different applications into common commands that a database server can understand. Now almost all the database servers (DBMSs) provide their own ODBC drivers. For example, MySQL Server provides MySQL ODBC 5.1 driver. That driver is used in my program.

Chapter 2 Software Installation

This chapter introduces how to install the necessary software in order to run program “BarcodeScanner” properly. This program was designed based on Microsoft visual studio 2008 (CSharp). And database is served by MySql server. Microsoft visual studio 2008 is definitely too large and unnecessary for users to install. Since it is commercial software, it costs money to buy the license. But what users need to do is only installing MySQL [6], its ODBC driver and creating a system ODBC data source.

2.1 MySQL installation

At the MySQL Community Server download page [7], users could find almost all version of MySQL server matching their operation system. What I have installed is the essential version for windows (x86, 32-bit). The essential version is definitely enough.

Unzip the setup file and run the downloaded MSI file. After click the next button on the welcome page, users can see the following “Setup Type” page (Figure 2-1).

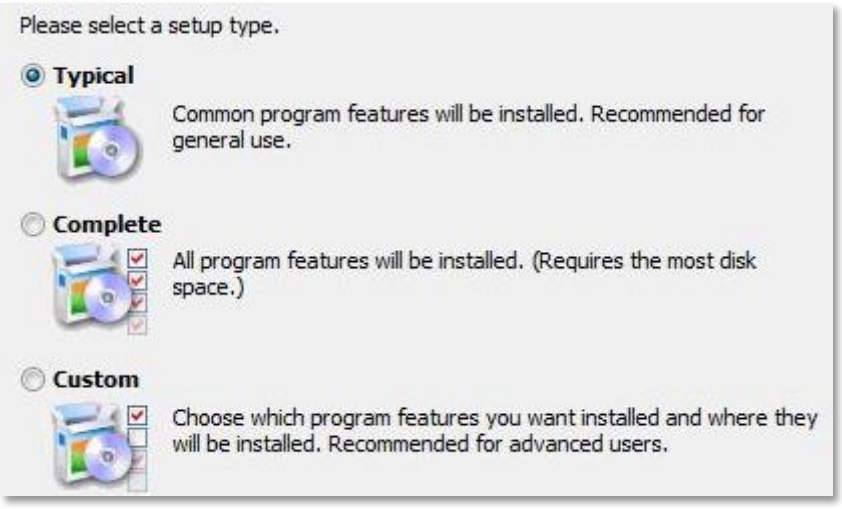

Figure 2-1 MySQL setup type selecting page

There are three different types of setup. With type “Typical”, MySQL server will be automatically installed into a path “C:\Program Files \MySQL \MySQL Server 5.1\” and its data files will be automatically stored into a path “C:\ProgramData\MySQL \MySQL Server 5.1\”. For beginner, this type will be better, because no path changing or settings are required. Someone who thinks the default installing path is not good can choose type “Custom”. With this type users can change installing path anywhere in their computers and as well as the related data files. “Complete” install is not necessary.

If users choose type “Typical”, they can just keep clicking button “Next” until finishing the whole installation. While if users choose type “Custom”, a features selection page will show on the screen. Users can freely choose which features to install or change installing path. But the main MySQL Server part should be remained.

The following figures show changing of installing path.

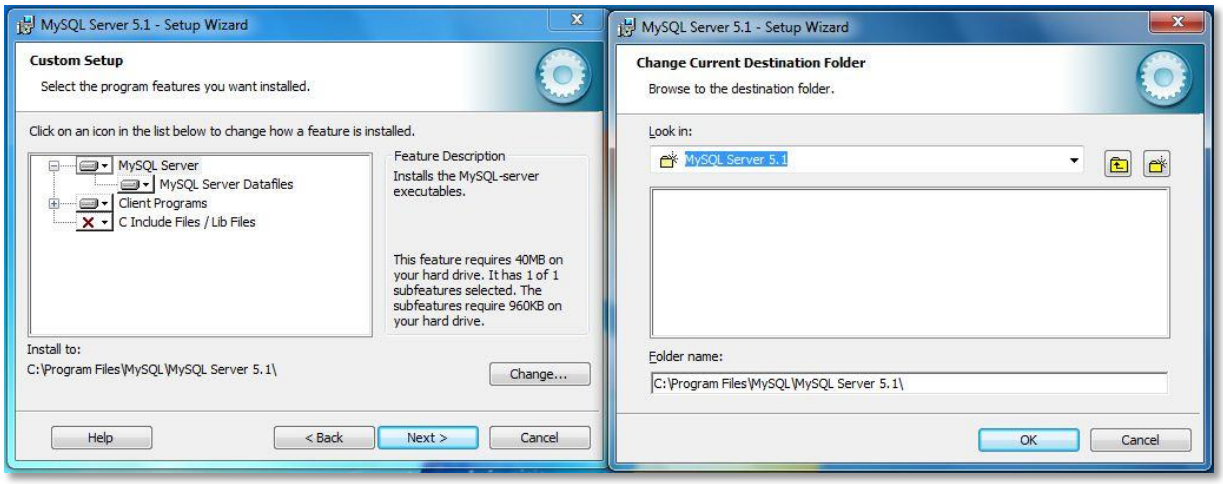

Figure 2-2 MySQL Custom setup changing server installing path

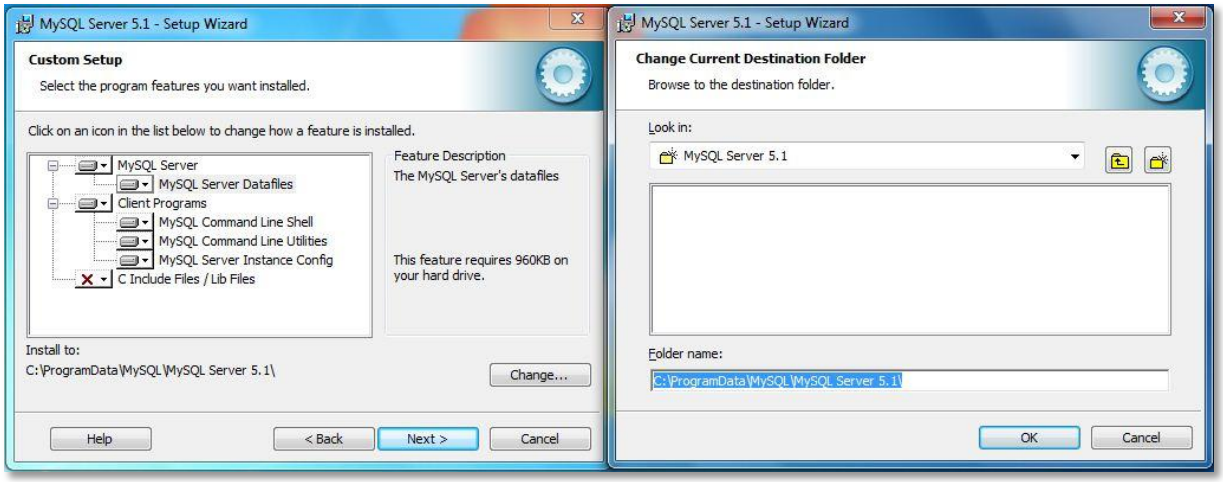

Figure 2-3 MySQL Custom setup changing data files installing path

After changing the path, users can also simply click button “Next” until the setup is finished.

2.2 MySQL configuration

The finish page of MySQL Server will give two check boxes which respectively are “Configuration” and “Registration” (Figure 2-4). MySQL Configuration is required, while registration depends on users’ own options. Check the “Configuration” box, and MySQL Server Instance Configuration Wizard will automatically run. Users can also find this wizard in MySQL installing fold.

Figure 2-4 MySQL Configuration / registration

During this process, users need to configure an instance of MySQL Server. Since program “BarcodeSanner” only require users to access local databases. To simplify this process, users can keep clicking button “Next” till they see security options (Figure 2-5).

In this page, Users are required to enter a password. This password is really important. It will be used to not only run server, but also build ODBC data source connection, as well as, login to program “BarcodeSanner”.

Figure 2-5 MySQL Configuration password setting

The following figure indicates process of configuration is successfully finished.

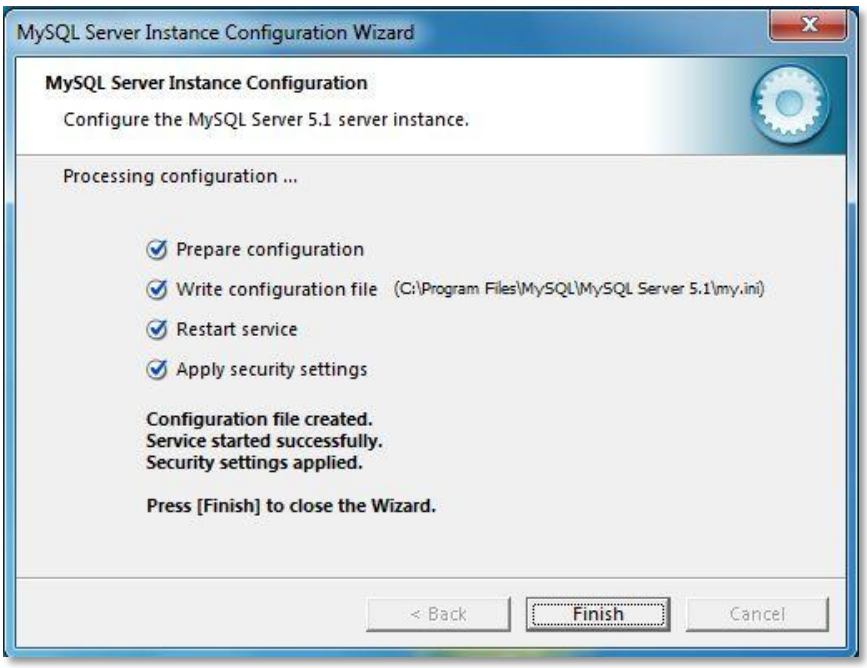

Figure 2-6 MySQL instance configuration succeed

After finishing the above two steps, MySQL Server should be able to run. For beginners, they can use MySQL Workbench, which is a GUI tool provided also by MySQL. The application of Workbench will be introduced in the following part.

But here I would like to introduce how to execute queries in command window. “MySQL command line client” can be found both in the installing file and Start Menu. Open this file and type in correct password. MySQL can run.

To store data tables, databases should be created in advance. Users can create a new database used to store data tables related to program “BarcodeScanner” (Figure 2-7). But using an existing database is also all right. Note that Chapter 6 will introduce other basic SQL commands.

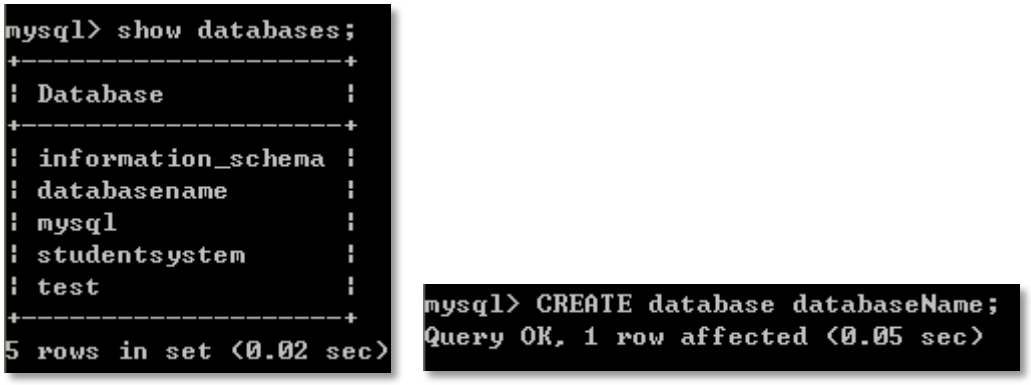

Figure 2-7 SQL command “SHOW databases;” and “CREATE database”

2.3 MySQL ODBC driver

To run program “BarcodeScanner”, an ODBC data source should be created. So this section presents how to create an ODBC data source using MySQL ODBC driver.

2.3.1 Installation

For users who have already installed old version of MySQL ODBC driver, for example MySQL ODBC 3.51 Driver, they can skip this step. Otherwise, users should follow the steps below to install an ODBC driver provided by MySQL. Here I recommend installing ODBC 5.1 driver. The process is similar to MySQL server setup wizard.

Download MySQL connector/ODBC 5.1 driver from the MySQL website [8] → Run the downloaded file →Choose “Typical” setup type → Click next → Finish

2.3.2 Option and Setting

After successfully installed the ODBC driver, the new ODBC data source can be created. [9]

Step 1 Go to control panel → Administrative tools → Find data sources (ODBC)

Step 2 Choose “system DSN” tag → Add new data source and choose MySQL ODBC 5.1 Driver (Figure 8) or maybe version 3.51 → Finish

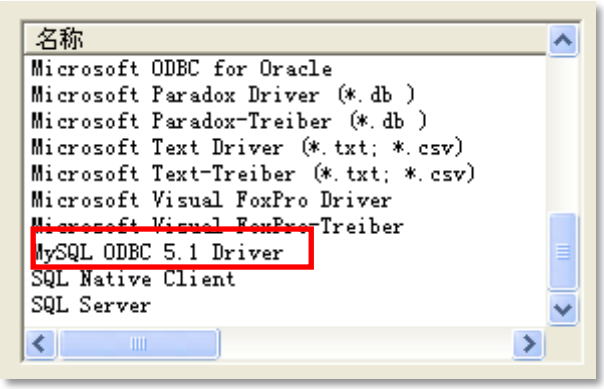

Figure 2-8 choose driver of ODBC data source

Step 3 Fill out **data source configuration** form

The data source configuration form is used to specify the new ODBC data source, and my program can access to the specific ODBC data source by those parameters/values.

The parameter Data Source Name will be the same as the user name of my program. Users can enter whatever they want. Parameter Description also does not matter. Since this program does not support accessing data served by other PC, so the parameter Server

should be “localhost”, Port should be the default value “3306”, and parameter User should be “root”. Parameter Password must be the same as the password used to run MySQL server. Otherwise database connection will be failed. Then choose a database and test the connection. If the database has been successfully connected, then the process of creating a new ODBC data source is finished and program “BarcodeScanner” can be work.

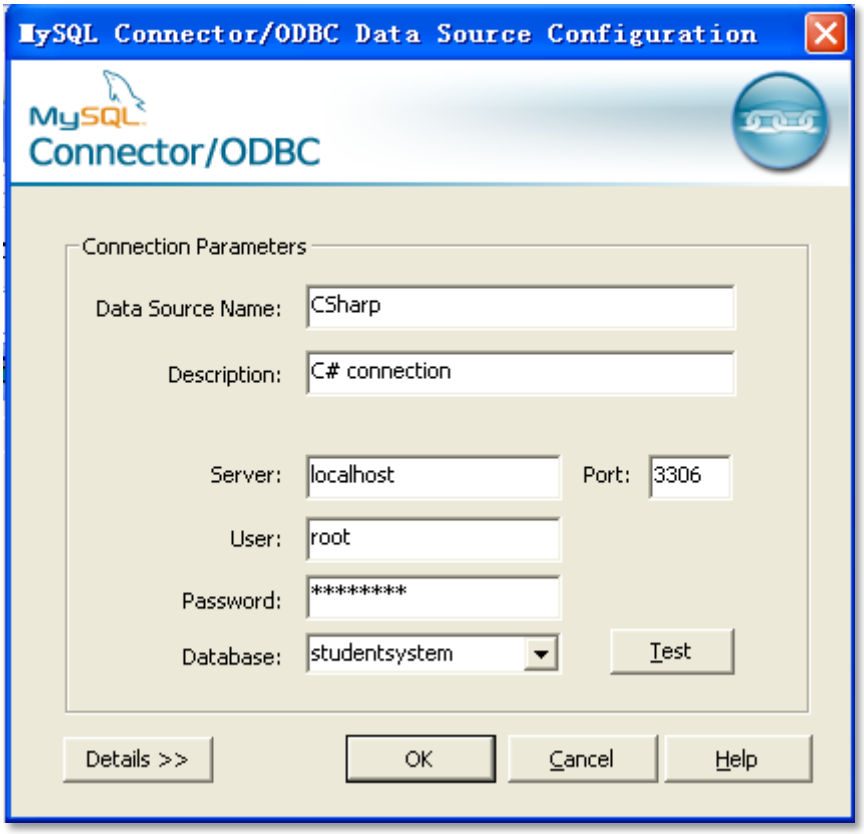

Figure 2-9 MySQL Connector/ODBC Data Source Configuration

In a short conclusion, this chapter gave an entire instruction of all necessary software used to support my program including MySQL Server, MySQL ODBC driver and creating a new ODBC data source. Development of my program also based on these software and settings. Therefore, they are the key to running my program.

Chapter 3 Programming

This chapter closely relates to program “BarcodeScanner”. It contains all parts of programming. The first section describes the usage of my program including brief description of two modes and basic operations. Then the next section explains some details, for example how to realize special functions in the program. And finally, a data management tool, “MySQL Workbench”, is introduced for beginners.

3.1 Usage

This section is about how to use my program properly including elementary introduction of appearance, components and functionalities of two modes.

3.1.1 Login

At the beginning, I would like to explain how to login. Because this program need to communicate with databases, a login window is necessary in order to get parameters from users to build a connection.

Step 1 fill out Login parameters

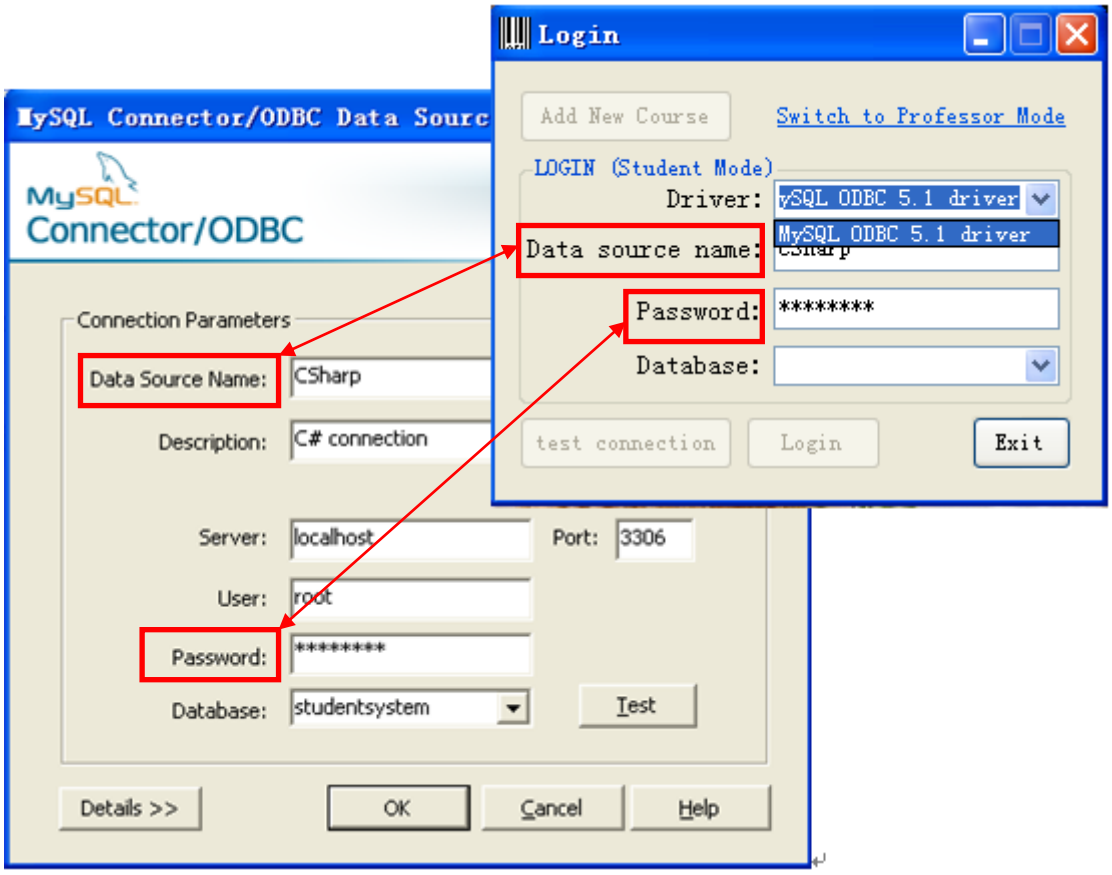

Figure 3-1 Login parameters sample

As I have noted, the login parameters “Data source name” and “Password” should be the same as connection parameters users set during their MySQL ODBC connector configuration. These parameters will still be used to access data source while saving changes. Users have the chance to select the version of driver. But now my program only supports for MySQL ODBC 5.1 and 3.51 drivers.

Step 2 test connection

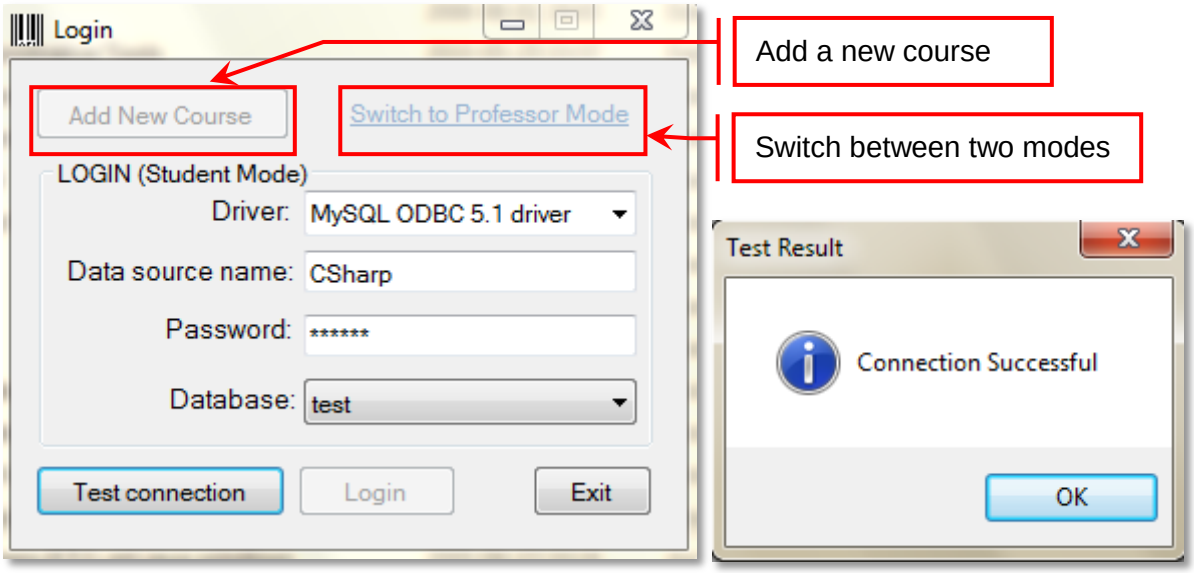

Figure 3-2 Test connection of program “BarcodeScanner”

After filling out all the required parameters, button “Test connection” will become enabled. Click it, and “Test Result” will be shown in a message box.

Step 3 Other operations

After successfully connecting, users can have two additional operations to do before login. But they are optional.

Firstly, they can add a new course. For users who are the first time running this program, adding some courses here is a good choice. But this is not the only entry to add new courses.

Secondly, they can choose a login mode. During programming I designed two login modes, the Student Mode and the Professor Mode. The purpose is to ensure data security. The student mode has no other functionalities except counting personal attendance. It was designed as the default mode. However, the professor mode contains all the extra functionalities, such as showing whole student list of the current course, updating students’ personal information or even adding/deleting new appointments to students.

After finishing the three steps above, we can login to the main program, “Attendance Counter” window.

3.1.2 Count attendance by using the Student Mode

As I have written in last page, there are two different modes. This part gives the introduction of Student Mode. This mode only contains the basic function, attendance counting.

The following figures show Attendance Counter window under Student Mode.

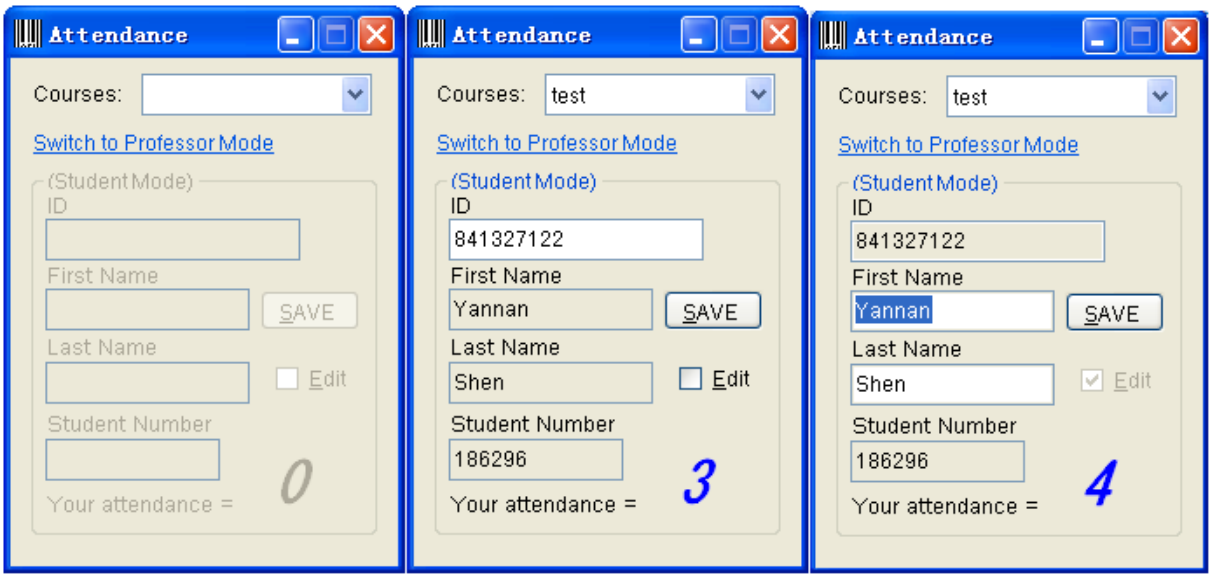

Figure 3-3 Attendance Counter (Student Mode)

After login, the first step for users to do is selecting one course from course list whichever mode they are using. Otherwise all other components or functions will not be usable. In other words, the counting function will be activated only after a course is selected.

For adding attendance, users have two ways. One way is type in ID number with computer keyboard, and press “Enter”. The other way is use a barcode scanner (Figure 3-4). The first method is slower and more inconvenient than the second one. However, for the second method, student card must be available. After doing either operation of them, student’s personal attendance will be displayed and be automatically plused by one. By the way, when using Student Mode, students can only see their own information.

And there is an additional function available in both modes. The name information of students is editable here. Check the edit (or Alt + E) and save changes after updating (or Alt + S). This is a quick entry for students to change their name information.

From the introduction above, we can find operations of Student Mode are very simple. Students are definitely able to take care of this application themselves, which means it is possible for users to do other things while using this program to count attendance. Therefore, it could be a timesaver.

Figure 3-4 Barcode Scanner

3.1.3 Count attendance by using Professor Mode

Now this part gives an introduction of Professor Mode. Actually the two modes were designed following the same design idea, although their appearances are presented in two different ways. The Professor Mode, on the basis of student mode, contains much more additional components and powerful functionalities.

For the Student Mode, I tried to keep the control as simple as possible. However, by contrast, for the Professor Mode, I tried to make it be powerful and able to be treated as a simplified database management tool. I added many extra functions, which allow professors to do as many operations as they want to control database and manipulate its data.

Before the introduction of Professor Mode, I would like firstly explain how to switch between two modes. To avoid students do some unexpected operations to my program, I designed a password checking function. What it means is from Student Mode to Professor Mode, program will ask for a password (Figure 3-5). Only with the right password, users could switch to the Professor Mode and use its functions. And to simplify the usage, the password string will be the same as login password (database connection password). But, on the contrary, from professor mode to student mode, no password checking is required.

Figure 3-5 switch from student mode to professor mode, password window

**Note: Program actually contains two entries of switching mode. One is on Login window, and another is on Attendance Counter window. However, password for switching mode will only be required when using link on attendance counter page.*

The following figure shows appearance of “Attendance Counter” under Professor Mode.

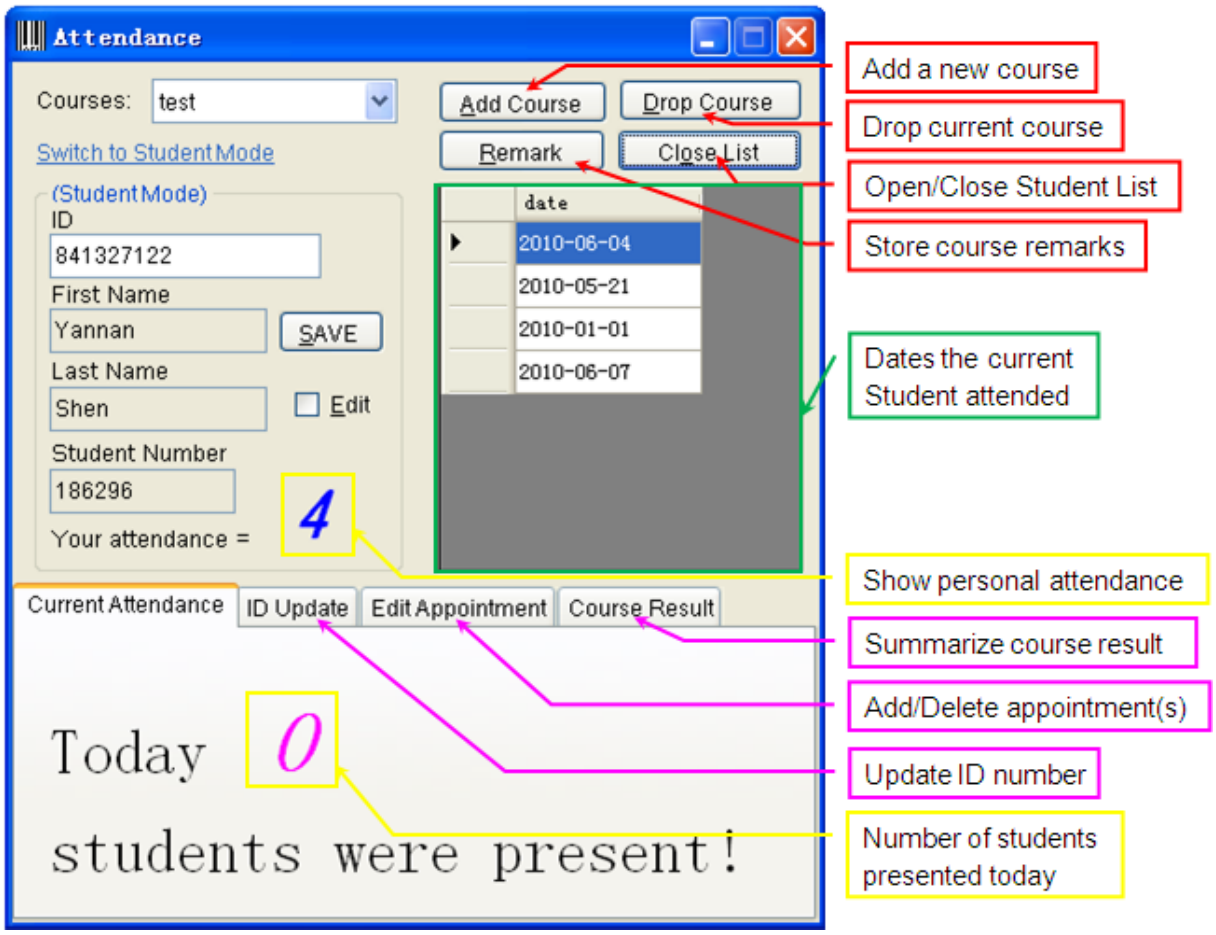

Figure 3-6 Attendance Counter (Professor Mode)

Functionalities list for Professor Mode:

1. Add New Courses (Alt + A): Users can add more courses into the local database.
2. Drop Courses (Alt + D): Users can freely drop some old courses.
3. Remarks (Alt + R): Allow users to add some remarks about this course.
4. Open/Close List (Alt + O): Allow users to go through personal information of all students.
5. Current Attendance: Show how many students presented this course today.
6. ID Update: Allow users to change students’ ID number.
7. Edit Appointment: Add a new appointment to the selected student or delete an old appointment from date table of the selected student.
8. Course Result: Judge students’ performance by their attendance status and showing result in a table.
9. Edit First Name / Last Name: The same function as in the Student Mode.

**Note: For ease of use, I added several hotkeys for these functions.*

The following several items are detailed explanation of functions shown in last page.

1. *New course adding*

My Program set two entries to active “AddNewCourse” window. One is on Login window, and another is on the “Attendance Counter” window of Professor Mode. Both entries connect to the same window (Figure 3-7). The “AddNewCourse” window only requires users to enter a course name they want to add. But for the course name, only letters and numbers are valid. Space or symbols cannot be identified. After users click button “CREATE”, program will do all the other operations regarding this new course. For example, check whether the course is already existed and create data tables for it.

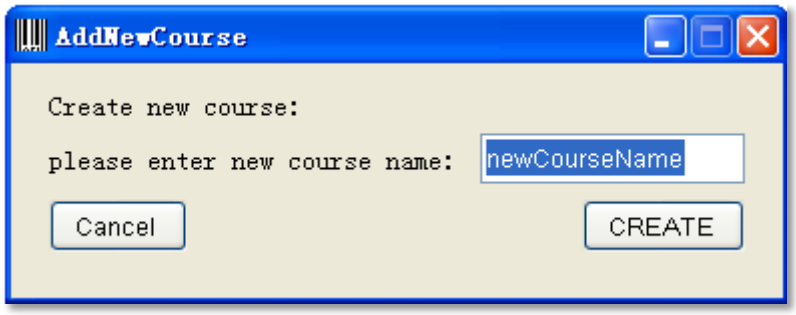

Figure 3-7 Window “AddNewCourse”

**Note: The course name is not case sensitive and only letters and numbers are allowed. (Structure of created tables will be introduce in the follow part)*

2. *Add and export remarks*

Concerning users might want to write down some remarks regarding the current course, for example, how many students visited today, I designed this function. For sure users can open other applications to write down, e.g. Text pad. But if program provides such a place for them to write these remarks, that will be more convenient. The button “Remark” relates to a window called “Remark”, which is shown below.

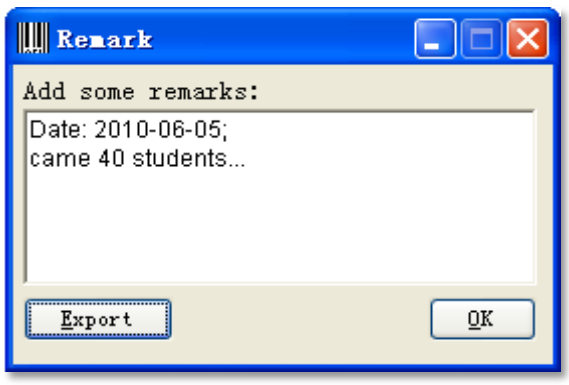

Figure 3-8 Window “Remark”

The window “Remark” contains a large textbox for uses to add information. Furthermore, it support for exporting these information. What this means is contents written in textbox could be exported into a text file.

**Note: After contents are exported, textbox will be automatically clear.*

3. Students list

The following figure shows window “StudentList”.

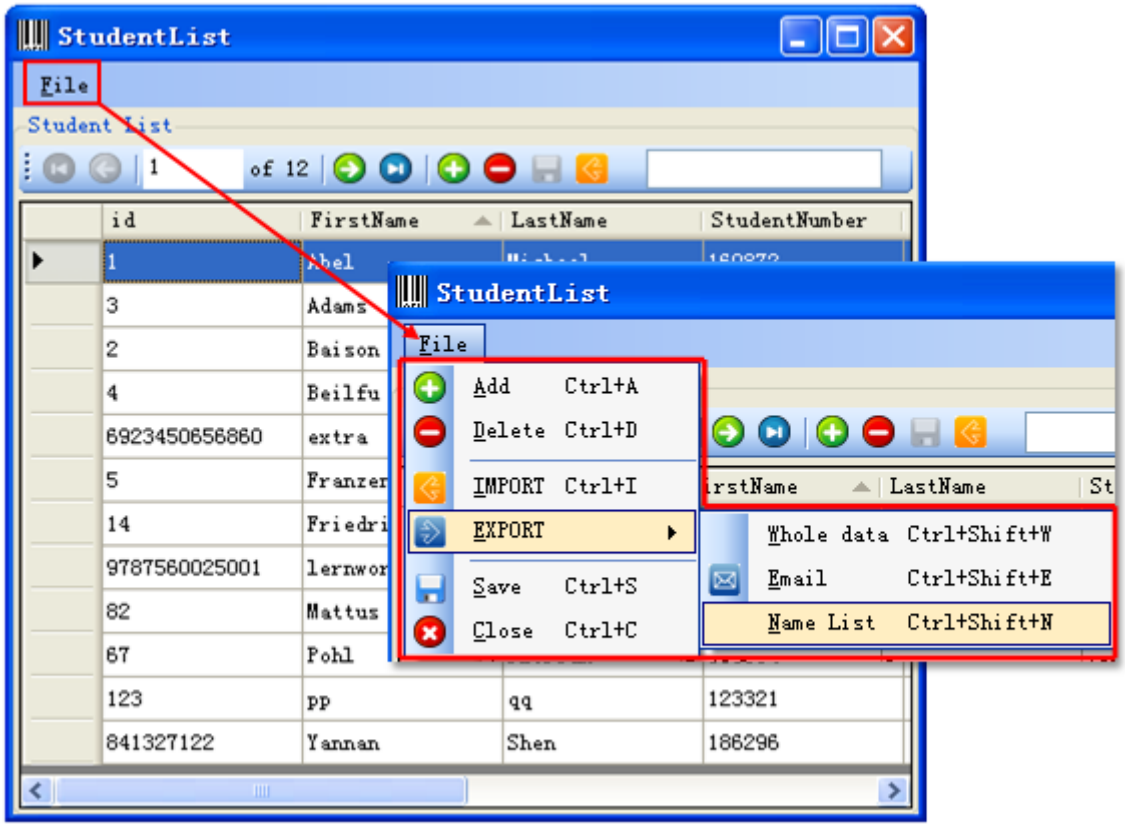

Figure 3-9 Window “StudentList”

Window “StudentList” displays all students’ personal information stored in local database related to the current selected course. It is closely related to window “Attendance Counter”, because information shown on window “Attendance Counter” will vary with the changing of selection in window “StudentList”, and vice versa.

In “StudentList”, all cells are editable, except column “ID”. And this table is able to automatically check whether contents of column “ID” and column “Student Number” are unique. Adding, deleting, updating, and saving records are all included in this window. The same as window “Attendance Counter”, components in “StudentList” also attached with hotkeys.

Furthermore, “StudentList” is accompanied by a navigator bar, which is used to control records. This navigator bar includes components of all functions. The operations from left to right are “Move to first record”, “Move to previous record”, “Position of selected record”, “Move to next record”, “Move to last record”, “Add new record”, “Delete selected record”, “Save changes”, “Import students” and “Search by first name”.

Moreover, “StudentList” contains a menu, which lists some of the above operations again. But it includes an extra function “Export”. This function allows users to export records into a text file. The contents could be the entire records, whole email addresses or only name list. This function enhances the flexibility of data control.

Here I briefly explain function “IMPORT”. Records here can be added one by one or imported in just a second from a text file. This text file should be prepared in advance and follow a fixed format. The following format simply shows the structure that a text file should have.

id; first name; last name; student number; email; contact phone; age; gender

**Note: Records should be saved after any changing. Otherwise the original data won’t be updated. And the structure of data tables and function “IMPORT” will be concretely explained again later.*

4. Update ID

ID column is a special column, because it is the primary key and is the only way to identify each student. That is the reason why I carefully treated all operations related to this column.

The Update ID tab (Figure 3-10) is mainly used to avoid a special situation that a student forgets to bring student card. If this student is a new student, users can at the first time give his/her a temporary ID when adding him/her to student list. And the next time when his/her card is available, users can change the temporary ID into the real one by scanning card.

The following figure shows tab “ID Update”. To update ID, users only need to scan the barcode of student’s card and press OK.

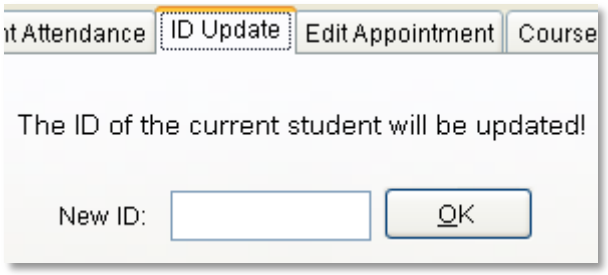

Figure 3-10 Tab “ID Update”

5. Edit appointment

The functions allow users to flexibly handle the attendance of each student. The tab “Edit Appointment” (Figure 3-11) has two functions: Add and Delete. What “ADD” means is any days can be added to attendance records of the current student. However, “DELETE” means a selected attendance record of this student could be canceled.

The following figure shows tab “Edit Appointment”.

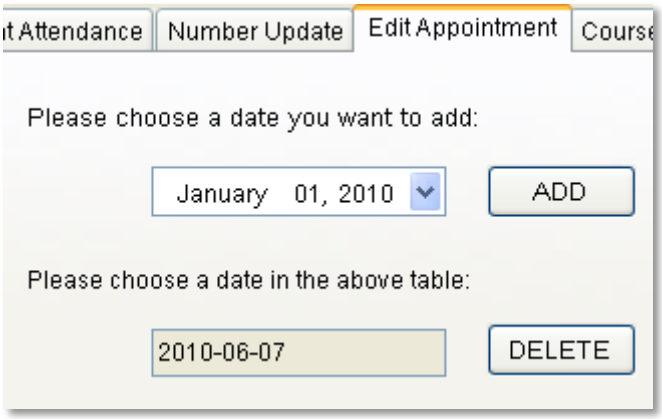

Figure 3-11 Tab “Edit Appointment”

Operations of these two functions are both simple. For adding, a date can be chosen from a calendar box (Figure 3-12). This can avoid a problem of different date format.

The following figure shows how to add an attendance.

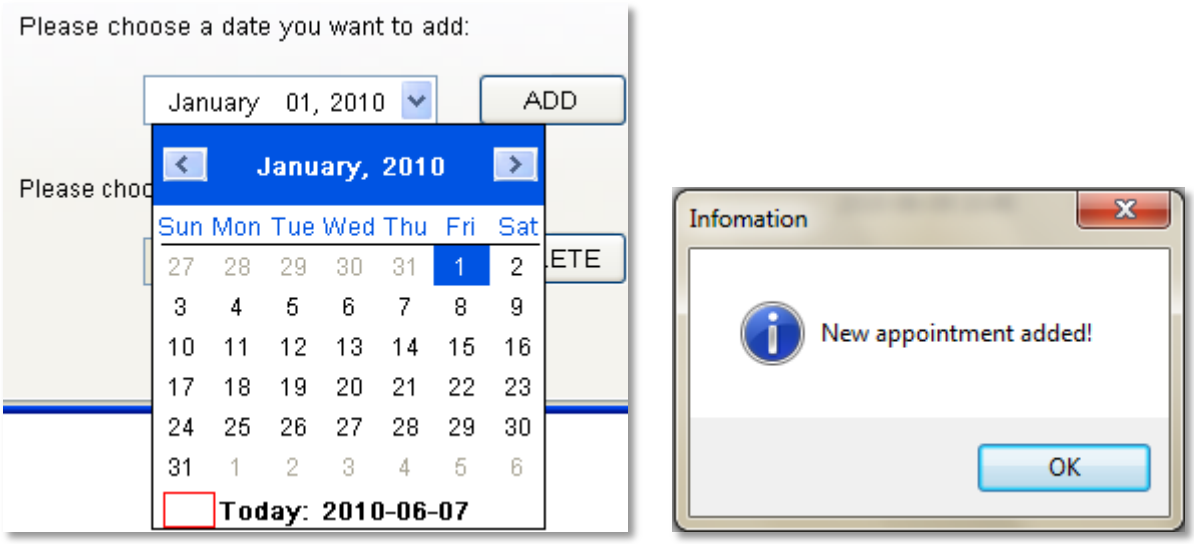

Figure 3-12 add an appointment _ Tab “Edit Appointment”

While for deleting, a date can be chosen from the date records shown right above the tab (Figure 3-13).

The following figure shows how to delete an attendance.

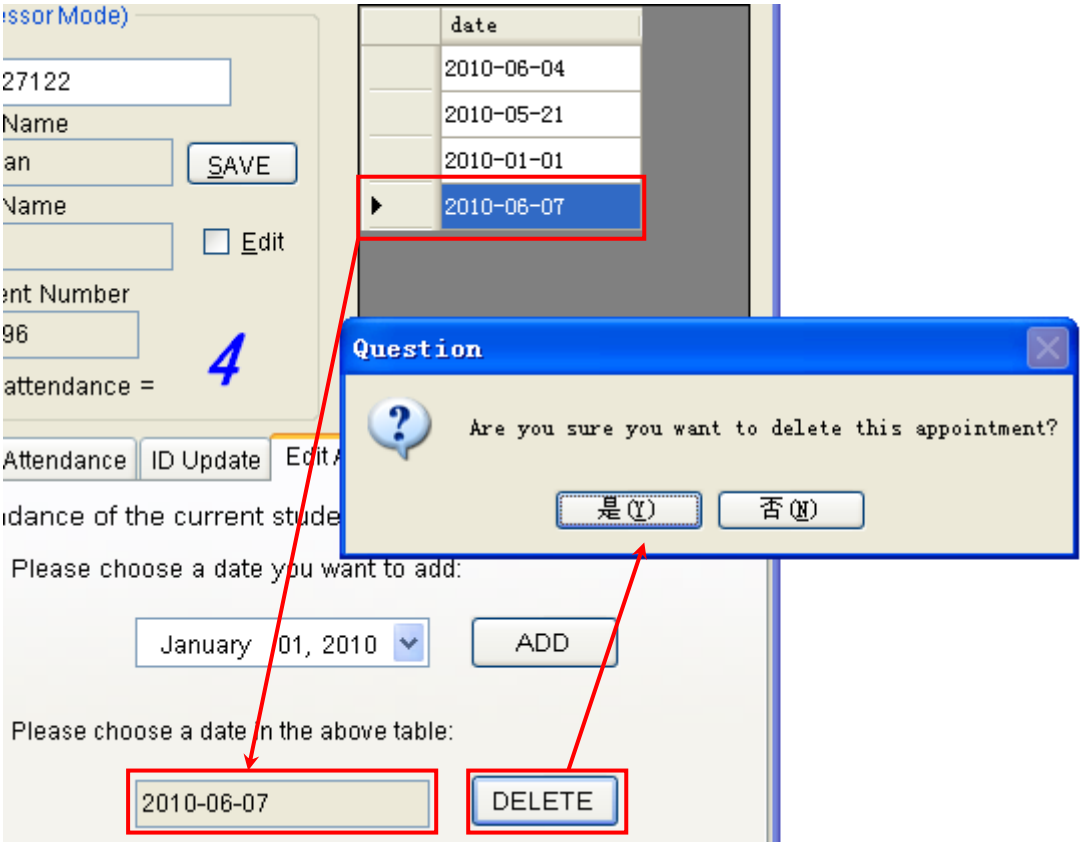

Figure 3-13 delete an appointment _ Tab “Edit Appointment”

6. Course result

The following figure shows tab “Course Result”.

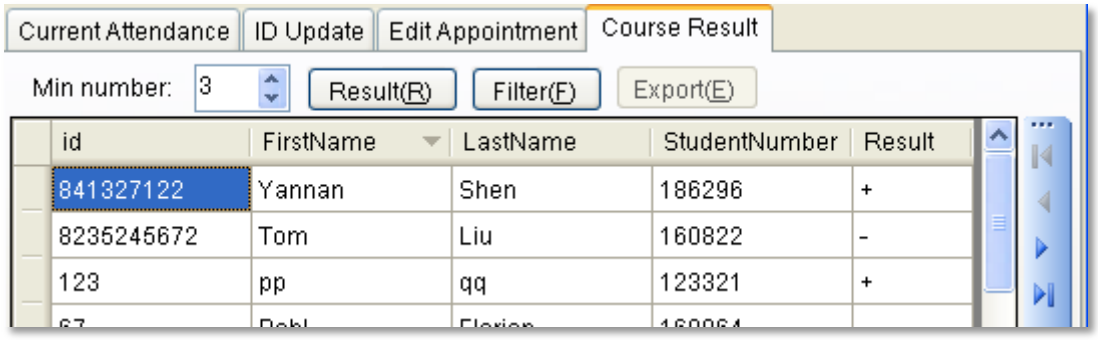

Figure 3-14 Tab “Course Result”

The purpose for this tab is to gather the attendance state of all students. I omitted the columns of unnecessary columns and only listed id, name, student number and result there. All contents here are not editable.

Column result is controlled by professors. They can mark pass or fail to each students by the number of their attendance. Result will be presented by two signs, “+” and “-”. “+” means attendance of this student achieves the minimum attendant requirement, while “-” means this student’s attendance is still not satisfied.

This tab contains an up/down list shows the minimum times students should be present. Professors could choose the number from 0 to 100.

Button “Result” is used to show final result, and this result depends on the number shown in up/down list. As I written, these results will be presented in column “Result” by “+” or “-“.

Meanwhile, button “Filter” is related to column “Result”. It allows user to filter out all records with “-”. But these records are only removed temporarily from this table. Press button “Result”, and all records will display again.

As an addition, button “Filter” could enable button “Export”. Button “Export” is extremely useful. It allows users to export all students who have a negative result into a text file. Users then can use this file to inform these students.

In a short conclusion, the above section explained all functions contained in my program step by step. By following this instruction, users must be able to clearly understand how the whole program works and how to properly handle it by themselves. So based on the usage, the next section includes detailed methods of my program for explaining how to achieve those special functions.

3.2 Detail explanation (with part of codes)

3.2.1 Structure of data tables

A database is able to contain lots of data tables. The same data table can be created in different databases. My program supports for two conditions. User could create all course tables in one database, or separately stored them in two or more databases.

As I designed each course is related to two data tables stored in a selected database. One is used to store the entire students' personal information records and another is used to store total students' attendance records. It means if one course, for example named "CSharpTutorial", has been created by user, meanwhile, two data table named "stu_CSharpTutorial" and "date_CSharpTutorial" will be created automatically in the database.

Structure of two data tables:

The following table shows structure of students' personal information table (stu_coursename).

Column name	Column type	Column details
ID	BIGINT	Primary key, Not null, Unique
FirstName	VARCHAR(45)	
LastName	VARCHAR(45)	
StudentNumber	INT	Not null, Unique
Email	VARCHAR(45)	
ContactPhone	VARCHAR(45)	
Age	INT(3)	
Gender	VARCHAR(45)	

Table 3-1 Structure of students' personal information table

The following table shows structure of students' attendance records table (date_coursename).

Column name	Column type	Column details
Serial	BIGINT	Not null, Unique, Automatically increase
Date	DATE	Not null
ID	BIGINT	Not null

Table 3-2 Structure of students' attendance records table

As I have written, I designed a function “IMPORT” to help users easier and quicker adding new records. This means the same records only need to be typed once, which can definitely save lots of time. But one drawback of this method is the imported file should be created manually by users and the following rules should be obeyed.

Rules of creating imported student records file:

1. The contents of imported file should based on the structure of students’ personal information table, which means each line of records should be exactly corresponding to the columns of the personal information data table.
2. Contents should be separated by a semicolon “**;**”, even the content is null.
3. Make sure the first content and fourth content (id and student number) are unique and not null.

Example record lines:

Complete record: (with all information)

*0841327122**;**Yannan**;**Shen**;**186296**;**sheny@stud.fh-luebeck.de**;**123456789**;**22**;**Female*

Incomplete record: (with some detail information)

*08235245672**;** Tom**;** Liu**;** 160822**;;;** 23*

*Or 08235245672**;** Tom**;** Liu**;** 160822**;** null**;** null**;** 23*

Basic record: (without detail information)

*0841325672**;** Abel**;** Michael**;** 160178**;***

The above three kinds of records are all able to validly applied to my program. If a text file of student records is created by following these rules, there will be no mistakes. Even an unexpected error occurs, program will show which line contains the error. Then users could easily correct it. By the way, a sample file “Student List_Sample.txt” can be found in my attached disc.

3.2.2 Data Binding

This part explains the method I used to bind data to a “DataGridView”, and at the same time bind this “DataGridView” to four textboxes and one “BindingNavigator”. The data I mentioned in this part is retrieved from data source and stored in a local data set. What this means is the data has no relationship with the original data.

The following method will achieve three functions. First, students’ records will be displayed in “DataGridView”. Secondly, this “DataGridView” could be control by a “BindingNavigator”. Finally, ID, FirstName, LastName, StudentNumber of the selected record in “DataGridView” will be shown separately in four textbox

```
private void BindingData ()
{
    //First a BindingSource object is required
    BindingSource tblBS = new BindingSource();
    //Then add some data as dataSource of this BindingSource
    // we assume dt is a data table which stored some data retrieved from data source
    tblBS.DataSource = dt;
    //Next step is to create all the components in the window form
    //Combine all these components to the same BindingSource object
    //Here the BindingSource object works like a bridge among these components
    //Add BindingSource to BindingNavigator
    bindingNavigator1.BindingSource = tblBS;
    //Add binding source to DataGridView
    dgView.DataSource = tblBS;
    //Add binding source to four textboxes
    textBoxID.DataBindings.Add(new Binding("Text", tblBS, "ID"));
    textBoxFN.DataBindings.Add(new Binding("Text", tblBS, "FirstName"));
    textBoxLN.DataBindings.Add(new Binding("Text", tblBS, "LastName"));
    textBoxSN.DataBindings.Add(new Binding("Text", tblBS, "StudentNumber"));
    //Now all components have refer to the same BindingSource object.
    //They will dynamically shown the same record which is selected.
}
```

The above codes are similar to my original codes, but not the same. I rewrite them in order to clearly explain this method. In fact, I added lots of other codes. For example, “*textBoxID.DataBindings.Clear();*” is used to clear the binding data of text boxes in advance.

3.2.3 Usage of data stream

My whole program contains one “IMPORT” and several “EXPORT” functions. To execute these functions, I used data steams to read and write data. And in order to use data stream, I imported a special name space, “System.IO”. Based on this namespace, streamReader and streamWriter classes could be available. Two methods StreamReader.ReadLine () and StreamWriter.WriteLine () were used to read or write one line of record. And to open and save file, I created an OpenFileDialog component and a SaveFileDialog component.

The following codes explain how to import data from a text file.

```
private void importRecords ()
{
    //create an OpenFileDialog component in order to choose a file.
    OpenFileDialog openStudentListFile = new OpenFileDialog ();
    //specify file type
    openStudentListFile. Filter = “Text files (*.txt)|*.txt”;
    string openFileName = openStudentListFile.FileName;
    if (openStudentListFile. ShowDialog () == DialogResult.OK)
    {
        //define a StreamReader object
        StreamReader sr = new StreamReader(openFileName);
        while (sr.Peek() >= 0) //End of the file
        {
            //read one line and split the string by “;”
            //and store sub strings into a string array
            string[] data = sr.ReadLine().Split(';');
            …
        }
        sr.Close(); //close the data stream
    }
}
```

The following codes explain how to export data into a text file.

```
private void exportRecords ()
{
    //create an SaveFileDialog component in order to save a file.
    SaveFileDialog saveStudentListFile = new SaveFileDialog ();
    //save file with type of text
    saveStudentListFile. Filter = "Text files (*.txt)|*.txt";
    //set a default name of saved file
    saveStudentListFile. FileName = "Student List";
    if (saveStudentList.ShowDialog() == DialogResult.OK &&
                                        saveStudentList.FileName.Length>0)
    {
        string saveFileName = saveStudentList.FileName;
        //define a stream writer object
        StreamWriter sw = new StreamWriter(saveFileName);
        …
        //write records to this file
        sw.WriteLine(recordLine);
        sw.Close(); //close the stream
    }
}
```

The above two methods are part of my original codes. It presented general methods of achieving data importing and exporting from and to a text file by using data streams. Note that Data Stream should immediately be closed after using it. Otherwise, the related file might not be used in other applications.

3.2.4 Get today's date

One important point to achieve counting attendance is get date of today. Many people may think it is a simple thing, because C# provides such function:

```
string today = DateTime.Today. ToString();
```

But actually programmers might face a problem that PCs can present date in different formats. Some people prefer the form like YYYY/MM/DD, while other people might prefer form like DD-MM-YYYY. The above code line can only get the date with the format of current PC. However, a column with type Date used in MySQL server must have the form YYYY-MM-DD, which means date only with this form can be inserted into data source provide by MySQL Server with no error.

At first I was thinking changing the date form of PC, it did work. But for users, this solution is too troublesome. After a more deeply study of C# functions, I finally found a good solution. So changing of date form of PC is no more required.

The following method shows how to get today's date with form of YYYY-MM-DD.

```
private String getTodayDate()
{
    string year = DateTime.Today.Year.ToString(); //get Year
    string month = DateTime.Today.Month.ToString(); //get Month
    string day = DateTime.Today.Day.ToString(); // get Day
    //return today's date in form YYYY-MM-DD
    return year + "-" + month + "-" + day;
}
```

The whole section introduced data table structures and summarized several special methods I used during the process of programming. These methods I shown can be helpful to other programmers.

In next section, a database management tool, MySQL Workbench, is illustrated especially for beginners.

3.3 GUI (Graphical User Interface) tools ---- MySQL Workbench

Although my program can also be treated as a simplified data management tool, it is definitely not that flexible and powerful as the official GUI tool provided by MySQL. For beginners, MySQL Workbench is an ideal data management tool provided with MySQL Server. On one hand, users can use Workbench to deeply understand the structure of data tables related to my program. On the other hand, they can easily control other data tables which are not involved in my program.

"MySQL Workbench is a visual database design tool that integrates SQL development, administration, database design, creation and maintenance into a single, seamless environment for the MySQL database system."

----Wiki, MySQL Workbench [10]

Firstly, software should be installed properly. The sample version I use here is MySQL Workbench 5.2, which is the latest development release. Download the software from MySQL download page [11]. Users should make sure to download the right version matching the operating system, because MySQL Workbench is available on Windows, Linux, Mac OS and maybe other systems. The installation is nothing special, and users with no experiences will also be able to handle it.

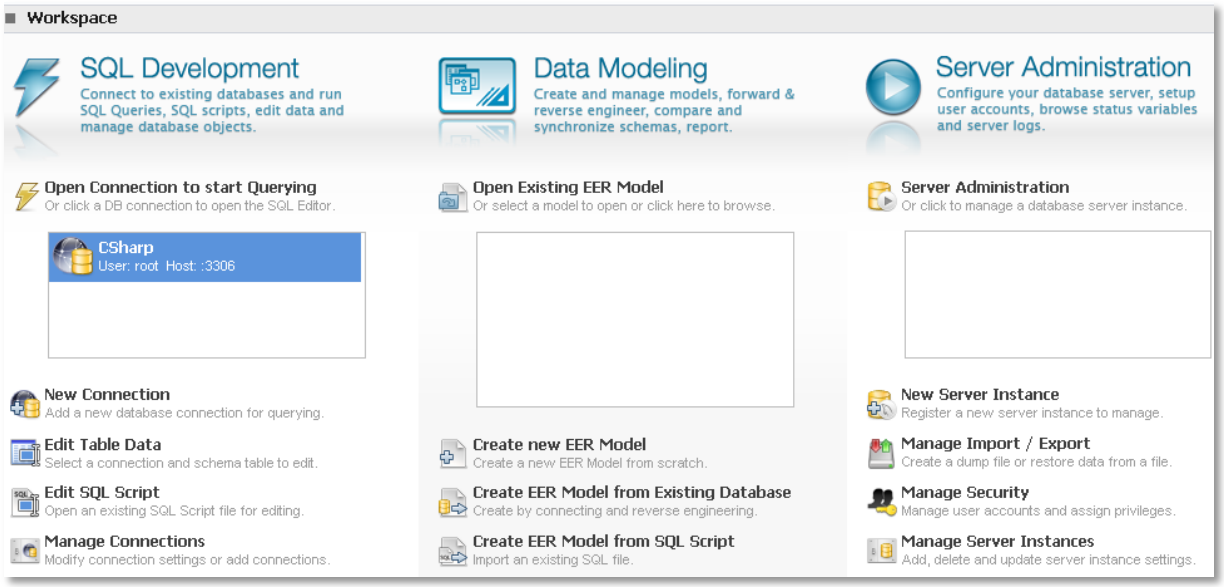

Figure 3-15 MySQL workbench homepage

The above figure shows the home page of Workbench 5.2. Users can find there are three main parts, SQL Development, Data Modelling and Server administration. In this thesis, I only focus on the first part, SQL development. I would like first introduce how to open a connection and then the basic control of databases.

*Note: A connection served by MySQL Server should have been created in advance.
See example of how to create a connection in ODBC data source, refer to Chapter 2.*

The following figure shows how to open a connection.

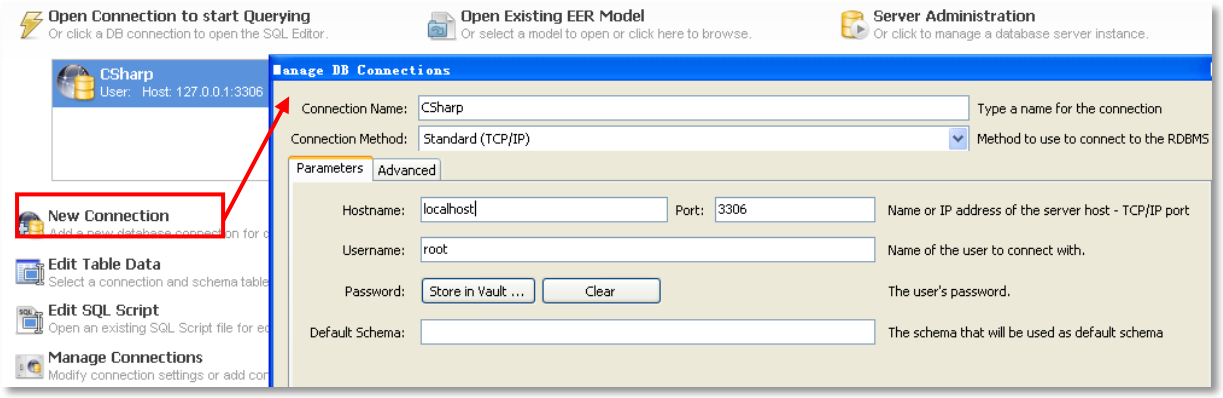

Figure 3-16 Open Connections _ MySQL Workbench

First, click new connection and fill out all the parameters depend on different connections. And then test connection (on the bottom of the window). If all the parameters are right, this connection will be executed and shown in a little screen. By double clicking, it will open and users can start querying.

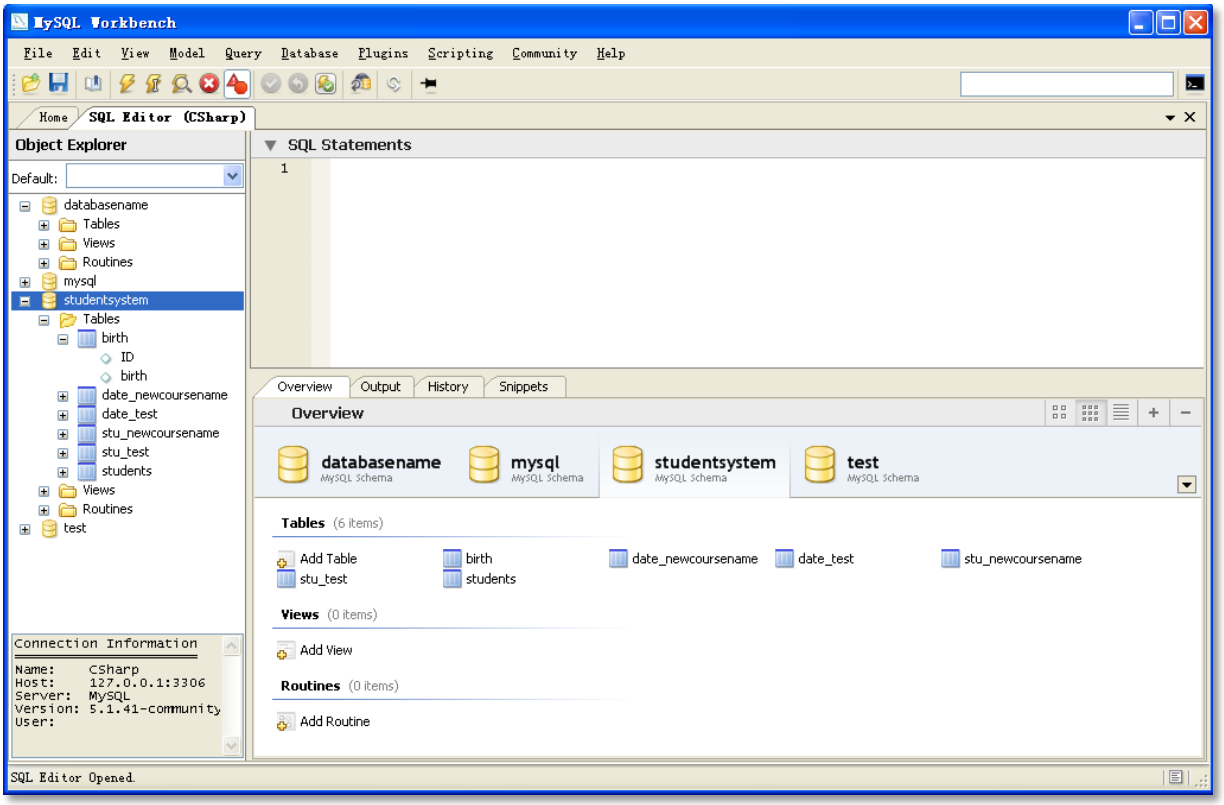

Figure 3-17 Main interface of MySQL Workbench

The main interface of MySQL Workbench 5.2 includes two parts (See Figure 3-17 in the last page). The left part is an Object Explorer, which contains total objects' catalog related to the current connection. Its contents will be narrowed down to each column of tables. And it is easy for users to go through the whole structure.

The right part is further divided into two parts. The above space area is used to query SQL statements. Users can enter a single line of SQL statement, a paragraph of SQL statements into there or even import a file with extension ".sql", and execute them (Hotkey is Ctrl + enter). Almost all SQL statements here are the same as in a command window, but much simpler.

And the part below contains several tabs, they are "Overview", "Output", "History", "Snippets", and sometimes "Result" will show there too. "Overview" is, the same as "Object Explorer", used to show objects of connections, but this form more focuses on child objects of a MySQL schema, such as data tables or views. "Output" shows the output of querying SQL statement(s), either succeeded or failed. "History" stores users' recent querying records. And "Snippets" allows users to store a piece of query statement for ease of use next time.

The brief explanation of main interface above brings us to the basic operations of MySQL Workbench.

a. Schemas (databases) manipulation. Users have two entries to manipulate schemas.

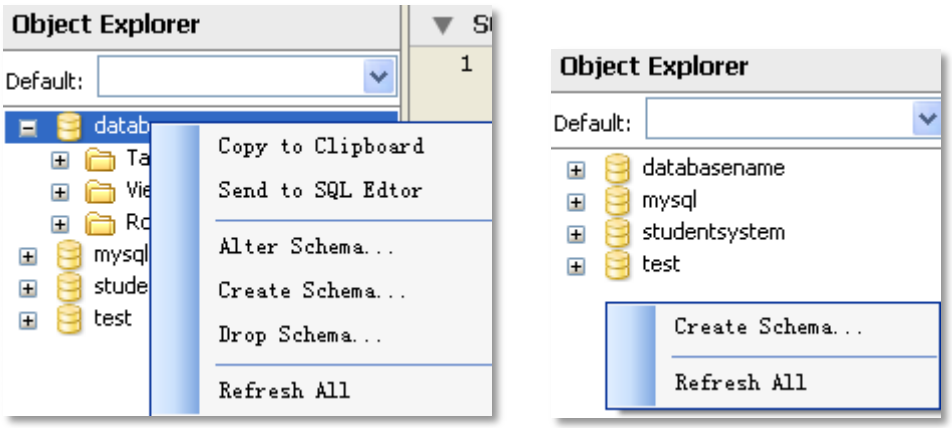

Figure 3-18 Control Schema (Object Explorer) _ MySQL Workbench

One entry is included in "Object Explorer". Right click an existing schema "Object Explorer", and users can see several operations towards this schema, such as alter, create or drop schema. Or even by right click the write space of "Object Explorer" can link to the operation of creating schemas.

Another entry is even simpler than the first one. On the right side of tab "Overview", users can definitely see symbols "+" and "-". Press "+" can add a new schema, and press "-" can drop the selected schema.

b. Data tables control. The same as the entries for schemas manipulation, users have two ways for manipulating tables. Still, using “Object Explorer” is one way, and using tab “Overview” is another way.

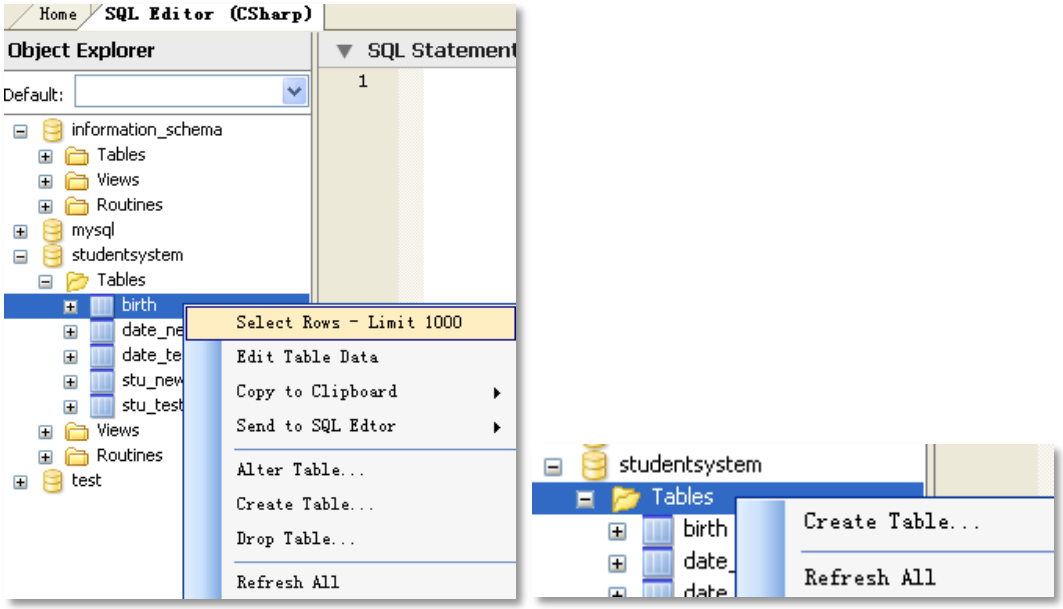

Figure 3-19 Control Table (Object Explorer) _ MySQL Workbench

Right click an existing table in “Object Explorer”, almost all available operations towards this table are listed there, such as alter, create or drop table. For creating a table, users can also right click the fold “Tables” or click “Add Table” shown on tab “Overview” (Figure 3-20).

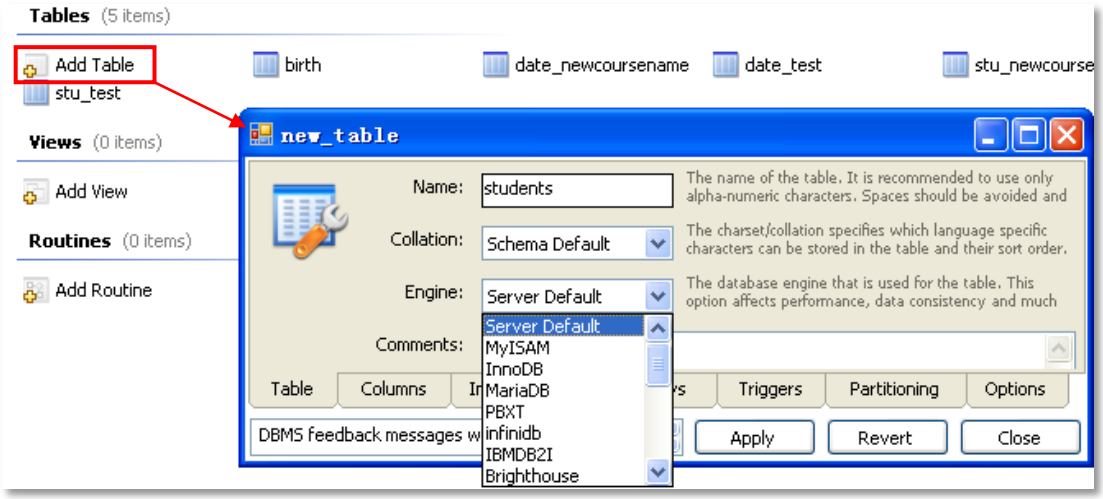

Figure 3-20 Control Table (tab “Overview”) _ MySQL Workbench

The following part is an example of adding a new table named “students”.

After users choosing “Add Table”, a new window will show. This window contains several tabs. The first tab “Table” requires users to enter table name, choose collation and storage engine.

MySQL server supports for many data storage engines. Depending on different performance requirements, users can choose the most suitable one. More detailed information about MySQL data storage engines is involved in Chapter 5.

The second tab “Columns” contains the column information of a table. The following figure shows a sample column structure of table “students” I defined.

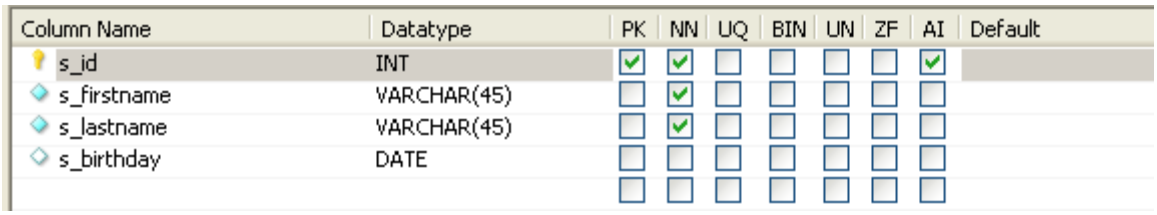

Figure 3-21 Column information of tables _ MySQL Workbench

Users can enter any column name they want, choose column types, and select the basic properties of columns. PK represents “primary key”; NN represents “not null”; BIN represents “is binary column”; UN represents “unsigned data type”; ZF represents “if value is numeric fill in 0”; AI represents “automatic increase”. In this way, columns’ properties can be easily and quickly added. Of course such properties can be added in other ways. For example, a primery key can be added also in tab “Indexes”. By the way, in the old version of MySQL Workbench there are only two of them, NN and AI.

And the other tabs are respectively “Indexes”, “Foreign keys”, “Triggers”, “Partitioning”, and “Options”. These functions are related to advanced manipulation of data tables. In order not to be aside from my topic. I will not concretely discuess these advanced functions.

After adding all columns and other controls, users can apply them. This time the querying SQL scripts will show. If it contains something that is not exactly what users expected, querying could be delayed. After checking these SQL scripts, operations will be finally applied to the database.

c. Data records (data rows) management. Users can right click a table no matter in the “Object explorer” or tab “Overview”, and then choose Edit Table Data (Refer to Figure 3-19) or they can double click a table shown in tab “Overview”. The same result will show below.

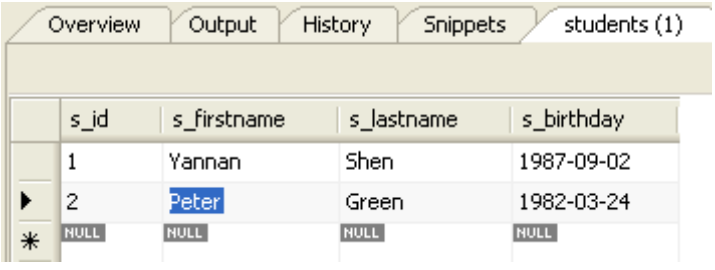

Figure 3-22 Edit records of table _ MySQL Workbench

After updating all records, press the right mark to apply those changes.

The above contents give a short overview of how to use MySQL Workbench to control a database including simple applications. I tried to clearly explain the most basic functions. In a short conclusion, using MySQL Workbench is really a wise choice for beginners as well as professionals to control their database properly.

3.4 Conclusion

This chapter went through the entire parts of program from outside, the usage, to inside, code/detailed explanation.

The usage part gives all details of how to use program properly, which help users to know program well. The detail part explains structure of database and 3 basic methods applied to program. Finally, as an addition, a database management tool MySQL Workbench was illustrated with detailed examples and screen shots. This tool is an extra support for my program. But my program, for sure, is able to run well without installing MySQL Workbench. Therefore, installing it or not is just depending on each individual user.

There is still one important thing missed in this chapter. That is how to retrieve data from the original data source. To achieve this, a crucial interface should be introduced, the ADO.NET interface. It is the fundamental of connecting databases and applications, therefore, next chapter will present an overview of ADO.NET as well as its basic applications.

Chapter 4 Overview of ADO.NET

In general, ADO.NET interface is a bridge connecting data sources and applications. Because of a strong set of elements and software components provided by ADO.NET, programmers can easily retrieve data stored in local or remote data sources. ADO.NET is definitely a crucial part of Microsoft .NET framework. This chapter aims to introduce structure and components of ADO.NET as well as the related applications.

**Note:* In order to use databases in C Shrap, a code line “using System.Data;” should be added into programs. (“System.Data” is a namespace)

4.1 ADO.NET components

ADO.NET is formed by two main components.

- .NET Framework data providers
- ADO.NET DataSet object

4.1.1 .NET Framework data providers

The .NET Framework data providers are designed for faster data accessing and easier command execution. It minimizes the layers used to get connection to data sources. There are four data providers included in .NET Framework.

The following table lists the four .NET Framework data providers. [13]

.NET Framework Data Provider	Description
For SQL Server	Provides data connection to Microsoft SQL Server version 7.0 or later. Uses the *System.Data.SqlClient* namespace.
For OLE DB	For data sources exposed using OLE DB. Uses the *System.Data.OleDb* namespace.
For ODBC	For data sources exposed using ODBC. Uses the *System.Data.Odbc* namespace.
For Oracle	For Oracle data sources. Uses the *System.Data.OracleClient* namespace.

Table 4-1 .NET Framework data providers

The structure of .NET Framework data provider for SQL server is different from others. It uses a different protocol to communicate with SQL Server. It will perform much better because it access SQL Server directly instead of adding the extra layers, for example, in OLE DB or ODBC providers. The following figure shows the different structure of data providers for SQL Server and for OLE DB, which might be helpful to understand. [14]

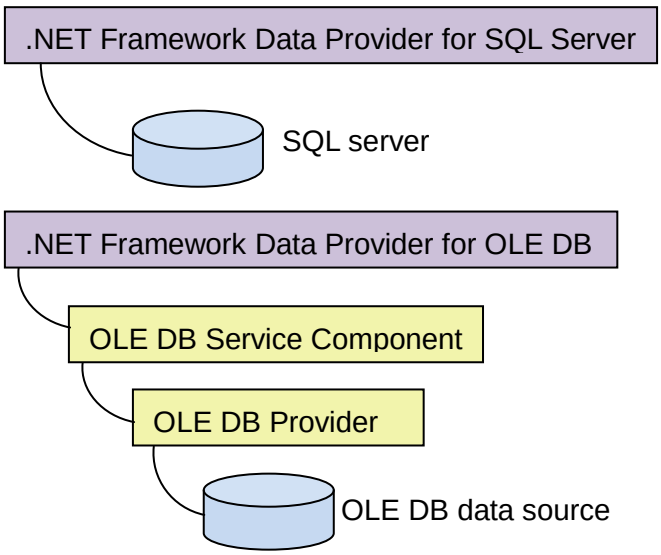

Figure 4-1 different structure of data providers for SQL Server and for OLE DB

**Note: The .NET Framework Data Provider for ODBC has a similar structure to the .NET Framework Data Provider for OLE DB; For Instance, OLE DB Service Component will be ODBC Service Component.*

Each .NET Framework data provider is composed by four core objects including Connection, Command, DataReader and DataAdapter. According to four types of data provider, there are four types of Connection, Command, DataReader and DataAdapter. They will be introduced separately subsequently.

The following table lists these four core objects and their purposes.

Object	**Purpose**
Connection	Builds a specific connection between applications and database servers.
Command	Executes a command at a data source, and exposes Parameters.
DataReader	Reads a forward-only, read-only stream of data from a data source.
DataAdapter	Communicate between the DataSet objects and data source. Uses Command objects to execute SQL commands at data source.

Table 4-2 core objects of a .NET Framework data provider

Connection Objects:

Four connection objects are listed below:

- **SqlConnection**: to connect to MS SQL Server version 7.0 or later (Provider for SQL)
- **OleDbConnection**: to connect to OLE DB data source (Provider for OLE DB)
- **OdbcConnection**: to connect to ODBC data source (Provider for ODBC)
- **OracleConnection**: to connection to Oracle data source (Provider for Oracle)

The OleDbConnection is the special one. It could also be used to connect to MS SQL Server.

"To connect to MS SQL Server version 6.X or earlier using the OLE DB Provider for SQL Server (SQLOLEDB), use the OleDbConnection object of the .NET Framework Data Provider for OLE DB."

----MSDN, Connecting to a data source using ADO.NET [15]

Actually, to connect to MS SQL Server version 7.0 or later by using the .NET Framework Data Provider for OLE DB is also feasible, but connection should pass an extra layer in .NET Framework Data Provider for OLE DB. It will decrease the speed and lower the performance of data source access. See the different structure of data providers for SQL Server and for OLE DB (Figure 4-1).

Talking about the connection objects, there is a primary property related to them, which is **ConnectionString**. Programmers can access to a specific database, because different connection strings can point to different databases. Moreover, there are two primary methods of connection object, which are **Open ()** and **Close ()**. The method Open () used to open a connection based on property ConnectionString. And the method Close (), on the contrary, used to close an opened connection. A connection should be open right before it is required and should be close as soon as programmers have finished with it.

If data adapters or data commands have been used in the application, then calling Open () and Close () methods programmatically can be optional. (The objects adapter and command will be introduced in the following part.) Because when a method such as **Fill ()** of data adapter is called, this method will check whether connection is already open. If not, the adapter opens the connection, and closes it after executing the method. But if the connection is open, the adapter will execute its method and do not close the connection. Programmers can flexibly use the connection objects.

DataAdapter Objects:

The DataAdapter objects build a bridge between data source and DataSet objects. They are used to exchange data. Generally, an adapter specified what data to fill into or get out of a dataset. But it needs an open connection to read and write data.

Four types of DataAdapter objects are listed below.

- **SqlDataAdapter:** specific to SQL Server. As I written above, it does not have to go through an OLE DB layer, so it is faster. But only used with SQL Server 7.0 or later.
- **OleDbDataAdapter**: used with any data sources exposed by an OLE DB provider.
- **OdbcDataAdapter**: access ODBC data sources
- **OracleDataAdapter**: access Oracle databases.

DataReader Objects:

A DataReader Object is used to get a read-only and forward-only stream of data from a database connection. It is not as widely used as other objects. It is only useful in Web Forms. In other words, it cannot be used in a Windows Form application.

Command Objects:

Command Objects is supported by adapters. And an adapter supports four main properties.

- **SelectCommand:** to get records from the data collections
- **InsertCommand:** to insert records into the data collections
- **UpdateCommand:** modify records in the data collections
- **DeleteCommand:** delete records from the data collections

The above four properties are instances of four command classes including **SqlCommand**, **OleDbCommand**, **OdbcCommand** and **OracleCommand**. These four classes of Command objects should be used with the same type of DataAdapter objects and Connections objects. What this means is if a SqlConnection object is created, programmers should apply SqlDataAdapter and execute SqlCommands.

Furthermore, Command objects support a property CommandText, which contains a reference to an SQL statement. Basic SQL commands will be illustrated in Chapter 6.

4.1.2 The ADO.NET DataSet

This part shows an overview of another main component of ADO.NET, the DataSet objects. If we say a .NET data provider works for connecting to a database, executing commands and retrieving data as results, then the DataSet object acts as a manager to handle those results locally.

"It can be used with multiple and differing data sources, with XML data, or to manage data local to the application."

----MSDN, ADO.NET DataSet [17]

Keeping a connection to data source occupies valuable system resources; therefore, it is not a wise choice to keep a connection all the time. A DataSet object allows programmers to manipulate data regardless of data source, which means data can be varied locally. But this does not mean disconnection from data source. Connection is just automatically been filled into a connection pool supported by each data provider. After the local data being updated, programmers can bring the connection back and save changes to data source. Applications of DataSet will be introduced in the following part.

The following figure shows the simplified model of DataSet.

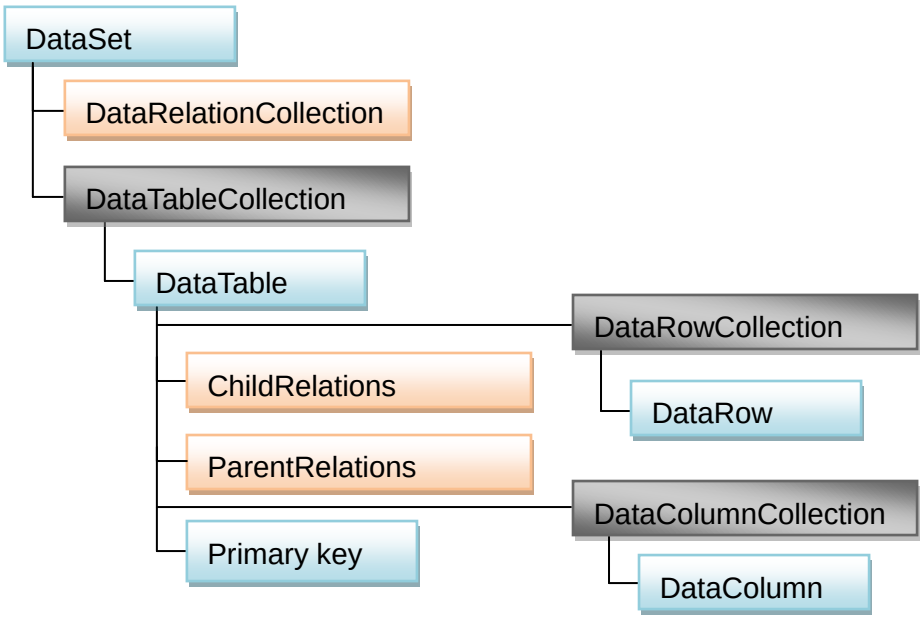

Figure 4-2 Simplified model of ADO.NET DataSet Object

An ADO.NET DataSet object contains two sub-objects, the **DataTableCollection** object and the **DataRelationCollection** object.

The **DatatableCollection** object collects null or multiple DataTable objects; Each DataTable object contains **DataRow** objects and **DataColumn** objects, which respectively collected in **DataRowCollection** and **DataColumnCollection**.

The **DataRelationCollection** object is used to collect relationships; while relationships are represented by **DataRelation** object. They build the parent and/or child relations between two DataTable objects. In other words, the relationships make it possible to connect several data tables within one DataSet.

4.2 ADO.NET applications

To more deeply understand ADO.NET interface, we should go through some applications. This section will introduce several examples including usage of data providers and usage of DataSet objects.

4.2.1 Usage of data provider

The following codes show how to get a connection to MySQL server with ODBC driver.

```
using System.Data.Odbc;
OdbcConnection conn;
private void ConnectToData()
{
    conn = new OdbcConnection(); //create a new OdbcConnection object
    //set connection string to point to a data source
    conn.ConnectionString =
        "Driver = {MySQL ODBC 3.51/5.1 Driver }; Dsn = Data source name; " +
        "UID = UserName; PWD = myPassword; Database = myDatabase" ;
    conn.Open(); //open the connection
    MessageBox.Show(conn.State.ToString()); //check whether connection built or not
    conn.Clost(); //close the connection
}
```

Property **ConnectionString** consists of a string with attribute/value pairs of information required to connect to a database. It could point to a specific database. The contents of ConnectionString can be verified, but the usage of four connection objects is similar. The above example builds an **OdbcConnection** object, and users can also build other connection objects. For more information of connection strings, please refer to Reference [18]. Since we used OdbcConection here, "using System.Data.Odbc;" must be added at the beginning of program. If other connections are used, the corresponding name space must be imported.

Log on to the specific database is the base of retrieving its data. As long as the connection is built successfully, programmers can retrieve data to local data sets and do some corresponding operations. The parts of code in next page introduces how to handle **Command** objects. I choose command SELECT and DELETE as instances.

The following codes show general methods to execute SELECT command.

```
private void SelectDataFromDataServer ()
{
    ConnectToData();//Log on to specified database
    try{
//method 1: Create an OdbcCommand with SELECT command to the specified database
        OdbcCommand com = new OdbcCommand("SELECT * FROM someTable", conn);
        //create an OdbcDataAdapter to execute this command object.
        OdbcDataAdapter da = new OdbcDataAdapter(com);
// OR method 2: Use property SelectCommand of OdbcDataAdapter to execute the same
SELECT command
        OdbcDataAdapter da = new OdbcDataAdapter();
        da. SelectCommand = new OdbcCommand("SELECT * FROM someTable", conn);

        //define an DataTable or DataSet to store the executed results.
        DataTable dt = new DataTable();
        DataSet ds = new DataSet();
        //Fill results into DataTable or DataSet.
        da.Fill(dt);
        //Or fill results into a table of ds named “List”
        da.Fill(ds,"List");
    }
    catch (Exception ex) //catch an error
    {
        MessageBox.Show(ex.Message);
        if (conn.State == ConnectionState.Open) //Finally close the connection.
        {
            conn.Close();
        }
    }
}
```

Because we opened an **OdbcConnection**, we have to use **OdbcCommand** and **OdbcDataAdapter**. As I noted, there is no need to call Open () and Close () methods of connection programmatically when methods like Fill () are used. The reason is, such methods are able to check state of connection automatically before executing.

Though some references show such methods will close the connection after execution, to advoid error occurs, I still recommand programmers to make sure closing connection finally.

The following codes show methods to execute commands DELETE.

```
private void DeleteDataInDataServer ()
{
    ConnectToData(); // Log on to specified database
    try
    {
//method 1: Create an OdbcCommand with DELETE command to the specified database
        OdbcCommand com = new OdbcCommand
                        ("DELETE FROM someTable WHERE someCondition", conn);
        //create an OdbcDataAdapter to execute this command.
        OdbcDataAdapter da = new OdbcDataAdapter(com);
//OR method 2: Directly use property DeleteCommand of OdbcDataAdapter.
        OdbcDataAdapter da = new OdbcDataAdapter();
        da. DeleteCommand = new OdbcCommand
                        ("DELETE FROM someTable WHERE someCondition", conn);

        conn.Open(); //open connection
         com.ExecuteNonQuery(); //execute command
        conn.Close(); //close connection
    }
    catch (Exception ex) //catch an error
    {
        MessageBox.Show(ex.Message);
        if (conn.State == ConnectionState.Open) //Finally close the connection.
        {
            conn.Close();
        }
    }
}
```

Different from command SELECT, we did not use methods such as Fill (), so open and close connection should be manually written. And command “UPDATE” or “INSERT INTO” can be applied in the similar form.

All the above examples are related to the original data source, including how to build a connection with data source, how to retrieve data from data source and how to upload updates into data source. Then the following examples concentrate on manipulation of data locally with DataSet object.

4.2.2 Usage of DataSet

A DataSet object is independent to database server. It provides lots of useful methods and properties for programmers to deal with data locally.

The following four examples are the basic operations of DataSet objects.

Example 1: Get name of a DataSet object by using property DataSetName.

```
DataSet ds= new DataSet ("StudentList");
MessageBox. Show (ds. DataSetName);
```

Example 2: Get name of DataTables collected in a DataSet object.

```
private void ShowTableName(DataSet ds)
{
    foreach (DataTable t in ds.Tables)
    {
        string str = t.TableName;
        MessageBox. Show(str);
    }
}
```

Example 3: Add a new DataTable with three columns and one record into a DataSet object.

```
DataSet ds = new DataSet ("StudentSystem"); //create a new DataSet named “StudentSystem”
DataTable dt = ds.Tables.Add("Name"); //Create a new DataTable named “Name”
DataColumn sID = dt.Columns.Add("id", typeof (int)); //add a DataColumn to Table “Name”
dt.Columns.Add("FirstName", typeof (string));
dt.Columns.Add("LastName", typeof (string));
dt.PrimaryKey = new DataColumn[] { sID }; //make column “id” the primary key column
dt.Columns["id"].ReadOnly = true; //column “id” is read only
dt.Columns["id"].Unique = true; //records of column “id” is unique
ds.Tables.Add(dt); //add the new DataTable to the DataSet
DataRow row = dt.NewRow();
row ["id"] = 1;
row ["FirstName "] = "Yannan ";
row ["LastName "] = "Shen ";
dt.Rows.Add(row);
```

Data is retrieved from data source and stored into DataSet objects. Then programmers could do operations like insert, delete or update to edit them.

Example 4: Add a DataRelation to a DataSet object.

```
DataSet ds = new DataSet();
DataRelation relation = ds. Relations. Add
        ("People", ds. Tables["People"].Columns["ID"], ds.Tables["Class"].Columns["ID"]);
```

4.3 Conclusion

The following figure illustrates the structure of ADO.NET platform. [16][20]

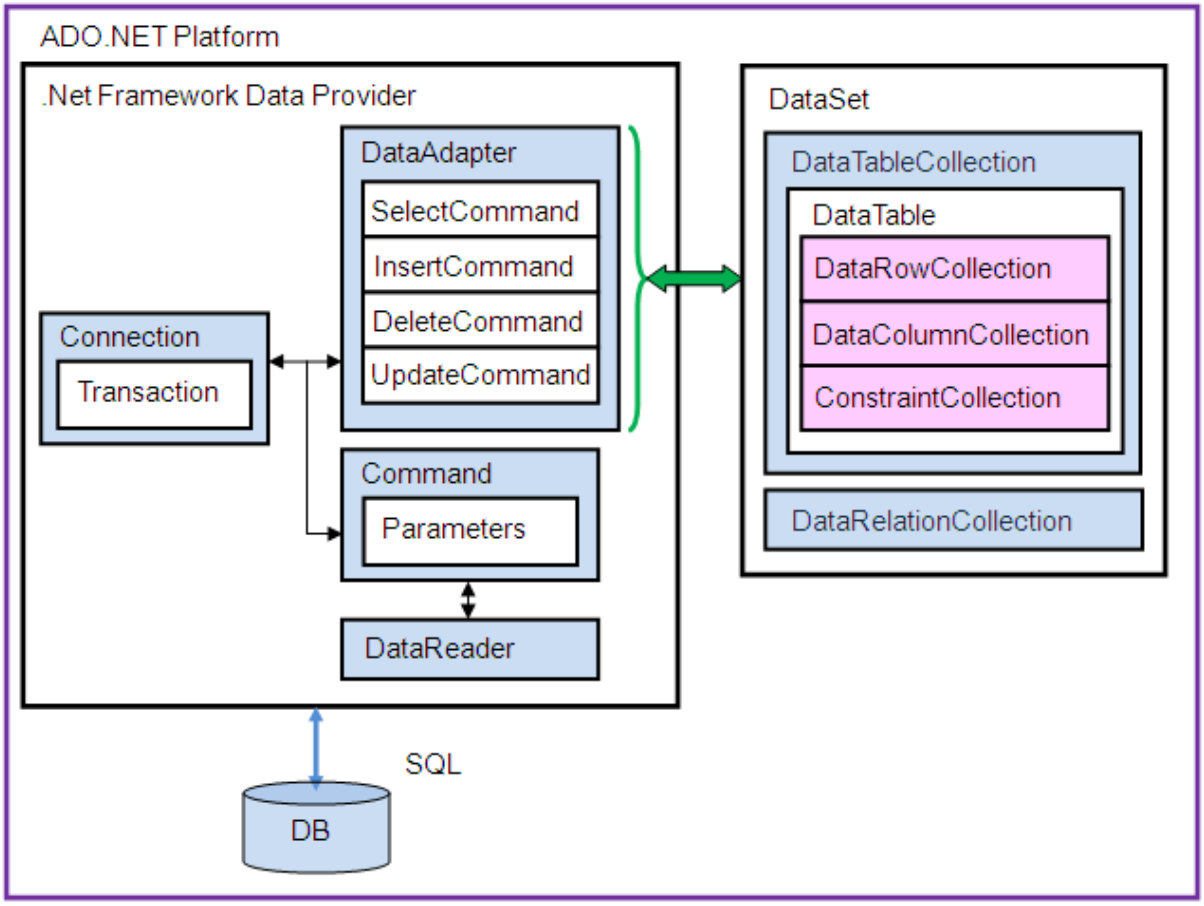

Figure 4-3 ADO.NET architecture

From the figure above, we can clearly find two components of ADO.NET and the simplified architecture of them.

All in all, this chapter introduced the general knowledge of ADO.NET interface mainly including the structure and components. Besides, it was supplemented by several examples.

ADO.NET is the key to binding C# programming and database connection together. It could support many kinds of database servers. Then this brings us to next chapter, comparison of different database servers.

Chapter 5 Comparison of Database Servers

What is a database server?

"A database server is a computer program that provides database services to other computer programs or computers"

----Wiki, database server [21]

In today's PC world, several DBMSs (Database management system) exist with their own logic and structures such as MySQL, Microsoft SQL Server, Access, Oracle, DB2 and PostgreSQL.

Due to the above phenomenon programmers always need to face a question, which database should they choose and which one is the best? This question is still a hot issue. In my opinion, everything exists in this world must have its own reasons. It is hard and unfair for us to say which one is better or the best. All the advantages should depend on different using environments and areas. And the existing database servers have their own drawbacks and advantages using in different applications.

In this chapter I try to summarize a relatively fair comparison of MySQL, MS (Microsoft) SQL server and PostgreSQL.

5.1 Open source VS. Commercial

Talking about the comparison of MySQL, MS (Microsoft) SQL server and PostgreSQL, the most obvious difference among them is the different license structure. So this section will show the difference concentrated on their license modes. [22]

As we all know, MySQL and PostgreSQL both come from the open source community, while SQL server is commercial software. The licensing schemes and costs of them are different.

MySQL is so called commercial open source software. The same as SQL Server, MySQL has a two-tiered licensing scheme.

The first licensing schemes of them both are free. But the restriction is different. SQL Server offers a free license for only development, which means the database system cannot be applied or related to any commercial area. However, MySQL offers a GNU General Public License (GPL) for GPL projects. What this means is as long as the developed project follows GPL rules, MySQL database can be used freely in any environment. But this GPL license requires total public distribution of application including original source code.

Then this brings us to the second licensing scheme. They both provide a commercial license for the commercial applications. For applying SQL database into commercial applications,

purchasing a SQL server commercial license is definitely required. As I noted, for using GPL license, the application must be total public including distributing its source code. So for someone who wants to keep his/her application source code secret has to purchase MySQL commercial license. This license is provided by MySQL AB, the company created MySQL software.

Although both second licenses cost money, the price of them is totally different. Purchasing a license of using SQL Server is much more expansive than MySQL commercial license. Since there are lots of versions, I cannot tell the accurate price of both licenses. But as far as I know the price of buying a standard edition SQL Server license is up to 1000 dollars.

On the contrary, PostgreSQL, as full open source software, only has one layer license. It is released under the Berkeley License (BSD) scheme. What this means is as long as a copy of the BSD license is included, PostgreSQL could be applied in whatever applications and even without distributing any source code.

During the past decade years, open source communities tried to improve quality of their software and made them more commercial. But whether the features and performance of such open source software are able to catch up to commercial software? Let us go to the next section and see some details.

5.2 Features and performance

To make the comparison more noticeable, I try to present it in a table including primary features and related performance of MySQL, MS SQL Server and PostgreSQL. [23]

Features	MySQL	MS SQL Server	PostgreSQL
operation system	Windows, Linux, Unix, Mac, FreeBSD, Solaris	Only windows	Earlier versions used mainly in Linux. The latest versions can run also on Windows 2000+, Mac, FreeBSD, Solaris
Storage engine	Have multiple independent open source storage engines, such as MyISAM, InnoDB [1], Cluster or BerkleyDB.	One powerful storage system	Only use Postgre Storage System
Install/Maintenance process	Easiest	Relatively hard	Medium
Speed	Fastest	Relatively slow	Medium
Support for Triggers [2]	YES, but only execute SQL Statements	YES	YES, and also able to execute user-defined function
Support for Indexes [3]	Most limited types of Indexes supported	Not support as many types of Indexes as PostgreSQL does, but for some types of Indexes, it has other techniques to achieve the similar result	Support the most types of Indexes, e.g. Partial Indexes, Functional Indexes
Support for Stored procedure	YES, but not in the older version	YES	Use stored function, which is similar to stored procedure
Support for Foreign key	YES, but only within InnoDB	YES	YES
Graphical Views	NO	YES	NO
Multi rows insert	YES	YES, but only with SQL Server 2008+	YES

Updatable Views	Able to automatically update single table view. But no support for updating more complex views.	Able to automatically update even 2 table views. Have triggers to support updating of more complex views	Not automatic. Rules for updating views should be manually written. But very complex views can be updateable.
Support for Replication	YES, and including master - master (build-in) replication	YES for all types	YES, but it seems only PostgreSQL 8.4+ will support the build-in replication
Support for Partition	MySQL 5.1+ can support for all kinds of partitions (Range, Hash, Range + Hash, List Partitions)	Only Range Partition	Support for all kinds of partitions (Range, Hash, Range + Hash, List Partitions)
Support for FULL JOIN	Only MySQL 5.1+ support for FULL JOIN command	YES	YES
Security	Relatively weak	Robust and powerful	Medium

Table 5-1 Cross comparison of MySQL, SQL Server and PostgreSQL features

---

[1] *MyISAM is the original default storage engine for the MySQL, which provides a high speed of reading data, but it was gradually token place by InnoDB, a more advanced storage engine. InnoDB enhances the support for transaction, foreign key and data integrity.*

[2] *Index: a system to make selecting information easier.*

[3] *Trigger: automatically execute code in response to a certain event on table or view in a database.*

5.3 Conclusion

To cross compare three databases is not an easy job, because they all have their own advantages and drawbacks. Now to make a conclusion, I list the primary advantages and disadvantages below.

DBMSs	Advantage	Disadvantage
MySQL	● Ease of use ● Quicker performance ● Suitable for small/middle projects. ● Low / free licensing fee.	● As commercial open source software, MySQL still not achieve commercial requirements in some areas. ● Not powerful to support advanced functions and objects. ● Relatively weaker security.
SQL Server	● Suitable for larger projects. ● Well performance on advanced functions. ● Has a strong backup and data protection mechanism. ● Strong authentication and security system.	● A little bit complicated to use for beginners. ● High licensing fee. ● Only used in Windows. ● Speed of performance is relatively low.
PostgreSQL	● As full open source software, new version updating is the quickest. ● Stable, powerful functionality and high reliability. ● Performance of functions is closer to commercial software. ● No additional licensing costs.	● Have less low-budget hosting sites and open sources support it than MySQL. ● Slower than MySQL. ● Old version does not run well on Windows system.

Table 5-1 summary of advantages and disadvantages

Generally speaking, for applications which require high performance and have a large and complex structure, SQL Server might be a wiser choice. However, for low budget or relatively small applications, MySQL and PostgreSQL might be better. But I personally think though PostgreSQL is not supported by as many open source software/hosts as MySQL nowadays, its future could be much brighter. Because of the updating speed of PostgreSQL version, its overall performance can come to the commercial level quickly. Now the performance of its latest version is really close to a commercial database. And with the benefit of no costs, lots of companies have fixed their eyes on PostgreSQL.

Chapter 6 Basic SQL Commands

"SQL stands for Structured Query Language. It is a standard language for accessing and manipulating databases."

----W3School, SQL tutorial [23]

As noted in the Introduction part, SQL language is supported by just about every database products on the market including MySQL, SQL server, Access, Oracle, DB2, etc. SQL could be the most basic knowledge for people wants to work with databases. It is a powerful and flexible standard language. People can easily manipulate data by querying SQL statements. SQL Syntax statements, words or phrases are always in lower case, while keywords are in uppercase. But not all versions are case sensitive!

This chapter presents basic SQL commands with concrete examples for beginners. But I would like first explain two concepts.

- SQL statements / commands: perform most of the actions on a database
- Database tables: contain several records (rows) with some contents (columns)

The following table shows a database table named "Students". It contains three records (one for each student) and five columns (ID, Firstname, Lastname, Studentnumber and Age). It will also be applied to explain examples in the following sections.

ID	Firstname	Lastname	Studentnumber	Age
1	Oven	White	01234	21
2	Peter	Green	01235	28
3	Sara	Doe	01236	20

Table 6-1 data table "Students"

6.1 SELECT Command

The "SELECT" command is one of the most common commands of SQL. It is used to select data from a database table. A SQL select commands can be followed by restrictions or not.

SQL select commands without restrictions can be presented in two forms.

```
SELECT column_name(s)
FROM table_name
```

This form selects the specified column(s) from the table

OR

```
SELECT * FROM table_name
```

This form selects all contents in the table

The select commands with restrictions might be followed by a keyword “WHERE”. This keyword is usually followed by one or more conditions pointed to some specific records.

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value AND/OR column_name operator value
```

EXAMPLE: Select all the students whose age is 20 years old from “Students” (Table 6-1).
SOLUTION:

```
SELECT * FROM Students
WHERE age = ’20’
```

RESULT (Table 6-2):

ID	Firstname	Lastname	Studentnumber	Age
3	Sara	Doe	01236	20

Table 6-2 SQL example: execute SELECT command

6.2 INSERT INTO Command

The “INSERT INTO” command is used to insert a new record into a database table. It also can be written in two forms.

```
INSERT INTO table_name
VALUES (value1, value2, value3 ...)
```

OR

```
INSERT INTO table_name (column1, column2, column3 ...)
VALUES (value1, value2, value3 …)
```

The first form inserts an entire record into the table. What this means is the values should be exactly corresponding to each columns of this table. The second form inserts values to some specific columns. And what this means is value 1 will be insert into column 1, value 2 will be insert into column 2, and so on.

EXAMPLE: Insert the following record into table “Student” (Table 6-1).

4	Ann	Moeller	01237	N/A

There are two forms of solution:

```
INSERT INTO Students
VALUES (‘4’, ’Ann’, ’Moeller’, ’01237’, NULL)
```

OR

```
INSERT INTO Students (ID, Firstname, Lastname, Studentnumber)
VALUES (‘4’, ’Ann’, ’Moeller’, ’01237’)
```

RESULT (Table 6-3):

ID	Firstname	Lastname	Studentnumber	Age
1	Oven	White	01234	21
2	Peter	Green	01235	28
3	Sara	Doe	01236	20
4	***Ann***	***Moeller***	***01237***	

Table 6-3 SQL example: execute INSERT command

6.3 UPDATE Command

SQL “UPDATE” command used to update existing record in the database table. It is, the same as “SELECT” command, can also be written with or without conditions. The general form is as follow:

```
UPDATE table_name
SET column1=value, column2=value2 ...
WHERE column_name operator value
```

EXAMPLE: Change Sara Doe’s age to 30.

SOLUTION:

```
UPDATE Student
SET Age=’30’
WHERE Firstname = ’Sara’ AND Lastname = ’Doe’
```

RESULT (Table 6-4):

ID	Firstname	Lastname	Studentnumber	Age
1	Oven	White	01234	21
2	Peter	Green	01235	28
3	Sara	Doe	01236	***30***
4	Ann	Moeller	01237	

Table 6-4 SQL example: execute UPDATE command

6.4 DELETE Command

SQL “DELETE FROM” command is used to remove a record (row) from a database table. This command has to contain one or more condition(s).

```
DELETE FROM table_name
WHERE column_name operator value
```

EXAMPLE: Delete the student whose student number is ‘01234’.

SOLUTION:

```
DELETE FROM Student
WHERE Studentnumber = ‘01234’
```

RESULT: Oven White’s record is gone (Table 6-5).

ID	Firstname	Lastname	Studentnumber	Age
2	Peter	Green	01235	28
3	Sara	Doe	01236	30
4	Ann	Moeller	01237	

Table 6-5 SQL example: execute DELETE command

6.5 CREATE DATABASE command

“CREATE DATABASE” command is used to create one new database.

```
CREATE DATABASE database_name
```

EXAMPLE: Create a new database called “StudentsInformation”.

SOLUTION:

```
CREATE DATABASE StudentsInformation
```

People can also show all databases by querying command “SHOW databases”. And the result will return a table with all databases’ name.

6.6 CREATE TABLE command

"CREATE TABLE" command is used to create a new table in a specified database. Before querying this command, one database should be chosen by querying "USE database_name".

"CREATE TABLE" command is similar to "CREATE DATABASE" command, but it is more complicated, because people should define each column of that table. Also some special indexes of columns can be added, such as primary key. The general form is shown below.

```
CREATE TABLE table_name
(
    column_name1 data_type,
    column_name2 data_type,
    column_name3 data_type,
    ....
)
```

EXAMPLE: Create a table called "Birth" with columns (ID, Name, Birthday) into database "StudentsInformation". Column ID has primary key and its data type is integer. Column Name is composed by characters. And column Birthday is a date. All columns except Birthday cannot be null.

SOLUTION:

```
USE StudentsInformation
CREATE TABLE Birth
(
    ID int NOT NULL,
    Name varchar(50) NOT NULL,
    Birthday DATE NULL,
    PRIMARY KEY (ID)
)
```

**Note:* Refer to reference [24] to learn more data types.

6.7 DROP command

The "DROP" command includes "DROP DATABASE" and "DROP TABLE" commands. "DROP DATABASE" command will drop the current database including all the following data tables. In the same way, "DROP TABLE" command will drop the current table as well as all its data columns. "DROP" commands should be carefully treated.

The following commands are the general form of “DROP DATABASE” and “DROP TABLE”.

```
DROP DATABASE database_name
```

```
DROP TABLE table_name
```

6.8 ALTER TABLE command

“ALTER TABLE” command can be used in three ways. First is adding a column, second one is droping a column, and the last one is updating data type of a column.

To add a column in a table:

```
ALTER TABLE table_name
ADD column_name datatype
```

To remove a column from a table:

```
ALTER TABLE table_name
DROP COLUMN column_name
```

To alter data type of a column in a table:

```
ALTER TABLE table_name
ALTER COLUMN column_name datatype
```

EXAMPLE:

First, add a column Birthday with data type DATE into table “Student” (Table 6-1).

SOLUTION:

```
ALTER TABLE Student
ADD Birthday DATE
```

RESULT (Table 6-6):

ID	Firstname	Lastname	Studentnumber	Age	Birthday
1	Oven	White	01234	21	***NULL***
2	Peter	Green	01235	28	***NULL***
3	Sara	Doe	01236	20	***NULL***

Table 6-6 SQL example: execute adding column to a table

Secondly, remove the column Age from table “Student”.

SOLUTION:

```
ALTER TABLE Birth
DROP COLUMN Birthday
```

RESULT (Table 6-7):

ID	Firstname	Lastname	Studentnumber	Birthday
1	Oven	White	01234	NULL
2	Peter	Green	01235	NULL
3	Sara	Doe	01236	NULL

Table 6-7 SQL example: execute dropping column from a table

Last, update the data type of column Studentnumber of table “Student” to character.

SOLUTION:

```
ALTER TABLE Student
ALTER COLUMN Studentnumber varchar (50)
```

6.9 Conclusion

This chapter summarized the most basic knowledge of SQL commands. Besides all the above operations, SQL still offers lots of advanced statements, such as JOIN or UNION. For beginners, such statements might not necessary.

SQL can be divided into two parts: The Data Manipulation Language (DML) and the Data Definition Language (DDL). My introduction included the basic DML such as SELECT, INSERT, UPDATE and DELETE, as well as some DDL such as CREATE/DROP TABLE and DATABASE commands.

Nowadays, many programmers do not want to type each SQL command by themselves. On one hand it is hard to remember all formats of SQL statements. On the other hand it is too slow to control data. They prefer to use, as I mentioned, the database management tools. Actually, almost all database servers will be provided with a management tool. For example, MySQL Workbench, what I introduced in Chapter 3, or SQL Server management studio express for MS SQL Server. But the importance of SQL commands still cannot be neglected.

Chapter 7 Summary and outlook

Thesis:

Topic of this thesis is about programming with MS Visual Studio C Sharp. And it requires connecting to a local data source served by MySQL Server.

This thesis firstly introduces the installation of required software including MySQL Server and its driver. They are the base of running my program. Then it illustrates the usage of my program followed by detailed explanation of functions. It includes structure of data tables, data binding, usage of data streams, and method to handle today's date.

ADO.NET interface is the key to accessing a database in C#. Therefore, this thesis contains a whole chapter explaining the components and structure of ADO.NET with detailed examples.

Since how to make a wise choice from the most database servers is nowadays a hot issue, I try to make a brief but fair comparison among MySQL, SQL Server and PostgreSQL.

For beginners, I conclude the basic knowledge of writing SQL statements and try to introduce how to use a GUI tool, MySQL Workbench, in this thesis.

Programming:

Towards programming, it was a tough but meaningful process. To me, MySQL Server is relatively familiar. I have done some other applications based on it. And I know the basic SQL Commands. Working with MySQL Server for me is not a tough job. But what really hard at the beginning was applying SQL statements to C#. C# was a total new programming language to me, and I did not have any experience about it before. But fortunately, with the help of my knowledge of other programming language and MSDN, I quickly got the key to writing a C# win form application.

Personally, I feel satisfied about the result. It achieves the original design goal. It could handle the task, counting attendance, quickly and accurately. It is easy to use and has sufficient functions. However, there is no perfect program in the world. I still think my program could be improved in the following two points.

1. Access to data sources served by other database systems, for example PostgreSQL. Users could flexibly choose other database systems and build connection to them.

2. Import personal information not only from a text file, but other types, for example a excel file. It must be easier for users to add new records. The same to "EXPORT" function.

Appendix 1 Lists

Figure List

Table List

Contents of Disc

- File *"BarcodeScann.exe"*
- Source code of program "BarcodeScanner"
- Thesis (PDF file): *"diplomThesis_YannanShen.pdf"*
- Sample file of student list: *"Student List_Sample.txt"*
- MySQL Database Server:
  "mysql-essential-5.1.48-win32.msi" & "mysql-essential-5.1.48-winx64.msi"
- MySQL Connector/ODBC 5.1 driver:
  "mysql-connector-odbc-5.1.6-win32.msi" & "mysql-connector-odbc-5.1.6-winx64.msi"
- MySQL Workbench 5.2:
  "mysql-workbench-ce-5.2.24-rc-win32.msi"

Appendix 2 Reference

[1] Visual C# Language, MSDN,
URL:http://msdn.microsoft.com/en-us/library/aa287558%28v=VS.71%29.aspx

[2] what is C#, C# station, **URL**:http://www.csharp-station.com/

[3] Database, Wikipedia, last modified on May 2010,
URL:http://en.wikipedia.org/wiki/Database

[4] Database and data capture, Page 4,
URL:http://www.bbc.co.uk/schools/gcsebitesize/ict/databases/2databasesrev4.shtml

[5] ODBC, last modified on April 10, 2007, **URL**:http://www.webopedia.com/term/o/odbc.html

[6] MySQL Server Installation on Windows, Revision 2.1, Nov 2007,
URL:http://www.netikus.net/

[7] MySQL Server Download page, **URL**:http://dev.mysql.com/downloads/mysql/

[8] MySQL Connector/ODBC driver download page,
URL:http://dev.mysql.com/downloads/connector/odbc/

[9] How to use MySQL Database with Visual C# 2008 Express Edition, David Bolton,
URL:http://cplus.about.com/od/howtodothingsinc/ss/mysqlnet.htm

[10] MySQL Workbench, Wikipedia, last modified on May 2010,
URL:http://en.wikipedia.org/wiki/MySQL_Workbench

[11] MySQL Workbench 5.2 download page,
URL:http://dev.mysql.com/downloads/workbench/5.2.html

[12] Developing .NET Data Provider Application,
URL:http://docs.openlinksw.com/st/dnetdevel.html

[13] Introduction to ADO.NET adapter, MSDN,
URL:http://msdn.microsoft.com/en-us/library/acb32th4%28v=VS.71%29.aspx

[14] .NET Framework Data Providers, MSDN,
URL:http://msdn.microsoft.com/en-us/library/a6cd7c08%28v=VS.80%29.aspx

[15] Connecting to a data source using ADO.NET, .NET Framework Developer's Guide,
MSDN, **URL**:http://msdn.microsoft.com/en-us/library/32c5dh3b%28v=VS.71%29.aspx

[16] Using ADO.NET for beginners, Huseyin Altindag, Last updated 12 Sep 2005,
URL:http://www.codeproject.com/KB/database/DatabaseAcessWithAdoNet1.aspx

[17] ADO.NET DataSet, .NET Framework Developer's Guide, MSDN,
URL:http://msdn.microsoft.com/en-us/library/zb0sdh0b%28v=VS.71%29.aspx

[18] Database ConnectionStrings, **URL**:http://www.dofactory.com/Connect/Connect.aspx

[19] Introduction to ADO.NET Connection Design Tools, Visual Basic and Visual C# Concepts,
MSDN, **URL**:http://msdn.microsoft.com/en-us/library/wxt2cwcc%28v=VS.71%29.aspx

[20] ADO.NET Architecture, .NET Framework Developer's Guide, MSDN,
ULR:http://msdn.microsoft.com/en-us/library/27y4ybxw%28v=VS.71%29.aspx

[21] Database Server, Wiki, Last modified on June 2010,
URL:http://en.wikipedia.org/wiki/Database_server

[22] PostgreSQL vs. MySQL vs. Commercial Databases, Tim Conrad, April 2004,
URL:http://www.devx.com/dbzone/Article/20743/0/page/1

[23] Comparison of Microsoft SQL Server 2005, MySQL 5.x and PostgreSQL 8.3,Leo Hsu &
Regina Obe, May 2008,
URL:http://www.postgresonline.com/journal/index.php?/archives/51-Cross-Compare-of-SQL-Server,-MySQL,-and-PostgreSQL.html

[24] SQL tutorial, W3School, **URL**:http://www.w3schools.com/sql/default.asp

[25] SQL data types, W3School, **URL**:http://www.w3schools.com/sql/sql_datatypes.asp