Working with Database

Comparison between ADO and ADO.NET:

ADO is a Microsoft technology. It stands for ActiveX Data Objects. It is a Microsoft Active□X component. ADO is automatically installed with Microsoft IIS. It is a programming interface to access data in a database

ADO.NET is a set of classes that expose data access services for .NET Framework programmers. ADO.NET provides a rich set of components for creating distributed, data- sharing applications. It is an integral part of the .NET Framework, providing access to relational, XML, and application data. ADO.NET supports a variety of development needs, including the creation of front-end database clients and middle-tier business objects used by applications, tools, languages, or Internet browsers.

ADO : ActiveX Data Objects and ADO.Net are two different ways to access database in Microsoft.

ADO	ADO.Net
ADO is base on **COM : Component Object Modelling based.**	ADO.Net is based on **CLR : Common Language Runtime based.**
ADO stores data in **binary** format.	ADO.Net stores data in **XML** format i.e. parsing of data.
ADO can't be integrated with XML because ADO have limited access of XML.	ADO.Net can be integrated with XML as having robust support of XML.
In ADO, data is provided by **RecordSet**.	In ADO.Net data is provided by **DataSet** or **DataAdapter**.
ADO is **connection oriented** means it requires continuous active connection.	ADO.Net is **disconnected**, does not need continuous connection.
ADO gives rows as single table view, it scans sequentially the rows using MoveNext method.	ADO.Net gives rows as collections so you can access any record and also can go through a table via loop.
In ADO, You can create only Client side cursor.	In ADO.Net, You can create both Client & Server side cursor.
Using a single connection instance, ADO can not handle multiple transactions.	Using a single connection instance, ADO.Net can handle multiple transactions.

Working with Connection, Command:

Working with Connection:

Process of creating connection:

For SQL Server [Note: Sql Server must be installed]

```
String conn_str;
SqlConnection connection;
conn_str = "Data Source=DESKTOP-EG4ORHN\SQLEXPRESS; Initial Catalog=billing; User ID=sa;Password=24518300";
connection = New SqlConnection(conn_str);
```

(Here, DESKTOP-EG4ORHN\SQLEXPRESS refers to a data source and billing refers to database name)

Working with Command:

```
SqlConnection connection;
SqlCommand command;
String conn_str="Data Source=Raazu\\SQLEXPRESS; Initial Catalog=billing; User ID=sa;Password=24518300";
connection = new SqlConnection(conn_str);
connection.Open();

String sql = "insert into tblCustomer(name,address) values(“Raaju”, “Birtamode”)";
command = new SqlCommand(sql, connection);
command.ExecuteNonQuery();
connection.Close();
```

For MS-Access [Note: Access Database Engine must be installed]

```
OleDbConnection conn;
OleDbCommand command;
string constr =
"Provider=Microsoft.ACE.OLEDB.12.0;DataSource=C:\\Users\\Raazu\\Documents\\Visual Studio 2012\\Projects\\DatabaseTest\\testdb.accdb";
conn = new OleDbConnection(constr);
conn.Open();
string sql = "INSERT INTO tblStudent(name,address) VALUES('Ram','Btm')";
```

```
command = new OleDbCommand(sql,conn);
command.ExecuteNonQuery();
Console.WriteLine("Data Inserted Successfully !");
```

DataReader, DataAdaper, Dataset and Datatable :

DataReader is used to read the data from database and it is a read and forward only connection oriented architecture during fetch the data from database. DataReader will fetch the data very fast when compared with dataset. Generally we will use ExecuteReader object to bind data to dataReader.

```
//Example SqlDataReader sdr = cmd.ExecuteReader();
```

DataReader

- Holds the connection open until you are finished (don't forget to close it!).
- Can typically only be iterated over once.
- Is not as useful for updating back to the database

DataSet is a disconnected orient architecture that means there is no need of active connections during work with datasets and it is a collection of DataTables and relations between tables. It is used to hold multiple tables with data. You can select data form tables, create views based on table and ask child rows over relations. Also DataSet provides you with rich features like saving data as XML and loading XML data.

```
//Example
DataSet ds = new DataSet();
da.Fill(ds);
```

DataAdapter will acts as a Bridge between DataSet and database. This dataadapter object is used to read the data from database and bind that data to dataset. Dataadapter is a disconnected oriented architecture.

```
//Example
SqlDataAdapter sda = new SqlDataAdapter(cmd);
DataSet ds = new DataSet();
da.Fill(ds);
```

DataAdapter

- Lets you close the connection as soon it's done loading data, and may even close it for you automatically.
- All of the results are available in memory.
- You can iterate over it as many times as you need, or even look up a specific record by index.
- Has some built-in faculties for updating back to the database.

DataTable represents a single table in the database. It has rows and columns. There is no much difference between dataset and datatable, dataset is simply the collection of datatables.
//Example
DataTable dt = new DataTable();
da.Fill(dt);

Difference between DataReader and DataAdapter:

1. A DataReader is an object returned from the ExecuteReader method of a DbCommand object. It is a forward-only cursor over the rows in the each result set. Using a DataReader, you can access each column of the result set, read all rows of the set, and advance to the next result set if there are more than one. A DataAdapter is an object that contains four DbCommand objects: one each for SELECT, INSERT, DELETE and UPDATE commands. It mediates between these commands and a DataSet though the Fill and Update methods.
2. DataReader is a faster way to retrieve the records from the DB. DataReader reads the column. DataReader demands live connection but DataAdapter needs disconnected approach.
3. Data reader is an object through which you can read a sequential stream of data. it's a forward only data wherein you cannot go back to read previous data. data set and data adapter object help us to work in disconnected mode. data set is an in cache memory representation of tables. the data is filled from the data source to the data set thro' the data adapter. once the table in the dataset is modified, the changes are broadcast to the database back throw; the data adapter.

Complete Example – CRUD operation (MS-Access):

```
using System;
using System.Data;
using System.Data.OleDb;
namespace DatabaseTest {
```

```
class Program {
        OleDbConnection conn;
        OleDbCommand command;
        void CreateConnection() {
        string constr = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=C:\\Users\\Raazu\\Documents\\Visual Studio
2012\\Projects\\DatabaseTest\\testdb.accdb";
        conn = new OleDbConnection(constr);
        conn.Open();
        }
        void InsertUpdateDelete(string sql) {
                command = new OleDbCommand(sql, conn);
                command.ExecuteNonQuery();
                Console.WriteLine("Operation Performed Successfully !");
        }
        void SelectRecords(string sql) {
                command = new OleDbCommand(sql, conn);
                OleDbDataAdapter adapter = new OleDbDataAdapter(command);
                DataTable dt = new DataTable();
                adapter.Fill(dt);
                if (dt.Rows.Count != 0) {
                        Console.WriteLine("Sid\t Name\t Address");
                        for (int i = 0; i < dt.Rows.Count;i++) {
                                string sid = dt.Rows[i]["sid"].ToString();
                                string name = dt.Rows[i]["name"].ToString();
                                string address = dt.Rows[i]["address"].ToString();
                                Console.WriteLine(sid+"\t"+name+"\t"+address);
                        }
                }
}
static void Main(string[] args) {
        Program obj = new Program();
        try { obj.CreateConnection();
        x: Console.WriteLine("1.Insert\t 2.Update\t 3.Delete\t 4.Select");
```

```
Console.WriteLine("Enter your choice: ");
int n = Convert.ToInt32(Console.ReadLine());
string sql="",nm = "", add = "";
int id=0;
switch (n) {
        case 1:
                Console.WriteLine("Ënter Name of Student: ");
                nm = Console.ReadLine();
                Console.WriteLine("Ënter Address of Student: ");
                add = Console.ReadLine();
                sql = "INSERT INTO tblStudent (name,address)
        VALUES('"+nm+"','"+add+"')";
                obj.InsertUpdateDelete(sql);
                break;
        case 2:
                Console.WriteLine("Enter id to be updated");
                id = Convert.ToInt32(Console.ReadLine());
                Console.WriteLine("Ënter Name of Student: ");
                nm = Console.ReadLine();
                Console.WriteLine("Ënter Address of Student: ");
                add = Console.ReadLine();
                sql = "UPDATE tblStudent SET name='"+nm+"',
        address='"+add+"' WHERE sid="+id; obj.InsertUpdateDelete(sql);
                break;
        case 3:
                Console.WriteLine("Enter id to be deleted");
                id = Convert.ToInt32(Console.ReadLine());
                sql = "DELETE FROM tblStudent WHERE sid="+id;
        obj.InsertUpdateDelete(sql);
                break;
        case 4:
                sql = "SELECT * FROM tblStudent";
                obj.SelectRecords(sql);
                break;
```

```
                default:
                        Console.WriteLine("Wrong Choice");
                        break;
                }
                goto x;
        }
        catch (Exception ex) {
                Console.WriteLine(ex);
                Console.WriteLine("Connection Failed !");
        }
        Console.ReadKey();
        }
}
```

Connect C# to MySQL

- First make sure you have downloaded and installed the MySQL Connector/NET from the MySQL official website.
- Add reference MySql.Data in your project.

```
using MySql.Data.MySqlClient;
string constr = "SERVER=localhost; DATABASE=dbtest; UID=root; PASSWORD=;";
MySqlConnection conn = new MySqlConnection(constr);
```

Complete Program for CRUD operation

```
using MySql.Data.MySqlClient;
using System;
using System.Data;
namespace DatabaseTest {
        class Program {
                MySqlConnection conn;
                MySqlCommand command;
                void CreateConnection() {
                        string constr = "SERVER=localhost; DATABASE=dbtest; UID=root;
                PASSWORD=;";
                        conn = new MySqlConnection(constr);
```

```
            conn.Open();
        }
        void InsertUpdateDelete(string sql) {
            command = new MySqlCommand(sql, conn);
            command.ExecuteNonQuery();
            Console.WriteLine("Operation Performed Successfully !");
        }
        void SelectRecords(string sql) {
            command = new MySqlCommand(sql, conn);
            MySqlDataAdapter adapter = new MySqlDataAdapter(command);
             DataTable dt = new DataTable();
            adapter.Fill(dt);
            if (dt.Rows.Count != 0) {
                Console.WriteLine("Sid\t Name\t Address");
                for (int i = 0; i < dt.Rows.Count;i++) {
                    string sid = dt.Rows[i]["sid"].ToString();
                    string name = dt.Rows[i]["name"].ToString();
                    string address = dt.Rows[i]["address"].ToString();
                    Console.WriteLine(sid+"\t"+name+"\t"+address);
                }
            }
        }
        static void Main(string[] args) {
            Program obj = new Program();
            try {
                obj.CreateConnection();
                x: Console.WriteLine("1.Insert\t 2.Update\t 3.Delete\t 4.Select");
                Console.WriteLine("Enter your choice: ");
                int n = Convert.ToInt32(Console.ReadLine());
                string sql="",nm = "", add = "";
                int id=0;
                switch (n) {
                    case 1:
                        Console.WriteLine("Ënter Name of Student: ");
```

```
            nm = Console.ReadLine();
            Console.WriteLine("Ënter Address of Student: ");
            add = Console.ReadLine();
            sql = "INSERT INTO tblStudent (name,address)
VALUES('"+nm+"','"+add+"')";
            obj.InsertUpdateDelete(sql);
            break;
case 2:
            Console.WriteLine("Enter id to be updated"); id =
            Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Ënter Name of Student: ");
            nm = Console.ReadLine();
            Console.WriteLine("Ënter Address of Student: ");
            add = Console.ReadLine();
            sql = "UPDATE tblStudent SET name='"+nm+"',
address='"+add+"' WHERE sid="+id;
            obj.InsertUpdateDelete(sql);
            break;
case 3:
            Console.WriteLine("Enter id to be deleted");
            id = Convert.ToInt32(Console.ReadLine());
            sql = "DELETE FROM tblStudent WHERE
sid="+id;
            obj.InsertUpdateDelete(sql);
            break;
case 4:
            sql = "SELECT * FROM tblStudent";
            obj.SelectRecords(sql);
            break;
default:
            Console.WriteLine("Wrong Choice");
            break;
}
goto x;
```

```
                    }
                    catch (Exception ex) {
                        Console.WriteLine(ex);
                        Console.WriteLine("Connection Failed !");
                    }
                    Console.ReadKey();
                }
        }
}
```

Complete CRUD operation for SQL Server

```
using System;
using System.Data;
using System.Data.SqlClient;
namespace DatabaseTest {
        class Program {
                SqlConnection conn;
                SqlCommand command;
                void CreateConnection() {
                        string constr="Data Source=Raazu\\SQLEXPRESS; Initial
                Catalog=dbtest; User ID=sa;Password=24518300";
                        conn = new SqlConnection(constr);
                        conn.Open();
                }
                void InsertUpdateDelete(string sql) {
                        command = new SqlCommand(sql, conn);
                        command.ExecuteNonQuery();
                        Console.WriteLine("Operation Performed Successfully !");
                }
                void SelectRecords(string sql) {
                        command = new SqlCommand(sql, conn);
                        SqlDataAdapter adapter = new SqlDataAdapter(command);
                        DataTable dt = new DataTable();
                        adapter.Fill(dt);
```

```
            if (dt.Rows.Count != 0) {
                Console.WriteLine("Sid\t Name\t Address");
                for (int i = 0; i < dt.Rows.Count;i++) {
                    string sid = dt.Rows[i]["sid"].ToString();
                    string name = dt.Rows[i]["name"].ToString();
                    string address = dt.Rows[i]["address"].ToString();
                    Console.WriteLine(sid+"\t"+name+"\t"+address);
                }
            }
        }
        static void Main(string[] args) {
            Program obj = new Program();
            try {
                obj.CreateConnection();
                x: Console.WriteLine("1.Insert\t 2.Update\t 3.Delete\t 4.Select");
                Console.WriteLine("Enter your choice: ");
                int n = Convert.ToInt32(Console.ReadLine());
                string sql="",nm = "", add = "";
                int id=0;
                switch (n) {
                    case 1:
                        Console.WriteLine("Ënter Name of Student: ");
                        nm = Console.ReadLine();
                        Console.WriteLine("Ënter Address of Student: ");
                        add = Console.ReadLine();
                        sql = "INSERT INTO tblStudent (name,address)
                    VALUES('"+nm+"','"+add+"')";
                        obj.InsertUpdateDelete(sql);
                        break;
                    case 2:
                        Console.WriteLine("Enter id to be updated");
                        id = Convert.ToInt32(Console.ReadLine());
                        Console.WriteLine("Ënter Name of Student: ");
                        nm = Console.ReadLine();
```

```
                    Console.WriteLine("Ënter Address of Student: ");
                    add = Console.ReadLine();
                    sql = "UPDATE tblStudent SET name='"+nm+"',
                address='"+add+"' WHERE sid="+id;
                    obj.InsertUpdateDelete(sql);
                    break;
                case 3:
                    Console.WriteLine("Enter id to be deleted");
                    id = Convert.ToInt32(Console.ReadLine());
                    sql = "DELETE FROM tblStudent WHERE
                sid="+id;
                    obj.InsertUpdateDelete(sql);
                    break;
                case 4:
                    sql = "SELECT * FROM tblStudent";
                obj.SelectRecords(sql);
                    break;
                default:
                    Console.WriteLine("Wrong Choice");
                    break;
                    }
                goto x;
            }
            catch (Exception ex) {
                Console.WriteLine(ex); Console.WriteLine("Connection Failed !");
            }
            Console.ReadKey();
        }
    }
}
```