using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

namespace AdoNetDemos

{

enum choice { Insert=1 , Update, Delete, GetRecords};

class InsertDemo

{

static SqlConnection connection=null;

static void Main()

{

int Choice;

string ch = "y";

while (ch == "y")

{

try

{

MainMenu();

Console.WriteLine("Enter Your Choice");

Choice = Byte.Parse(Console.ReadLine());

switch (Choice)

{

case (int)choice.Insert:

{

InsertRecord();

break;

}

case (int)choice.Update:

{

UpdateRecord();

break;

}

case (int)choice.Delete:

{

DeleteRecord();

break;

}

case (int)choice.GetRecords:

{

GetRecords();

break;

}

default:

{

Console.WriteLine("Invalid Choice");

break;

}

}

Console.WriteLine("Do you want to repeat the process");

ch = Console.ReadLine();

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

static void MainMenu()

{

Console.WriteLine("1. Insert Record");

Console.WriteLine("2. Update Record");

Console.WriteLine("3. Delete Record");

Console.WriteLine("4. Display All Records");

}

static SqlConnection GetConnnection()

{

connection = new SqlConnection("data source=LAPTOP-53S2KQS8;" +

"initial catalog=PracticeDb1;integrated security=true");

return connection;

}

static void InsertRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

string name = Console.ReadLine();

Console.WriteLine("Enter Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Salary");

int salary = int.Parse(Console.ReadLine());

SqlCommand command = new SqlCommand("Insert into Employee (id, name , address, salary) values(@id, @name,@address,@salary)", connection);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@name", name);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records inserted are "+ count);

connection.Close();

}

static void UpdateRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID whose Record you want to modisy");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter New Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Revised Salary");

int salary = int.Parse(Console.ReadLine());

SqlCommand command = new SqlCommand("Update Employee" +

"set address =@adress, salary = @salary where is=@id", connection);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records updated are " + count);

connection.Close();

}

static void DeleteRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID whose Record you want to delete");

int id = Byte.Parse(Console.ReadLine());

SqlCommand command = new SqlCommand("Delete Employee" +

"where id=@id", connection);

command.Parameters.AddWithValue("@id", id);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records deleted are " + count);

connection.Close();

}

static void GetRecords()

{

connection = GetConnnection();

SqlCommand command = new SqlCommand("Select \* from Employee",

connection);

connection.Open();

SqlDataReader reader = command.ExecuteReader();

if(reader.HasRows)

{

while(reader.Read())

{

Console.WriteLine(reader["id"].ToString()+ " " + reader["name"]);

}

}

reader.Close();

connection.Close();

}

}

}

**Best Place to store connectionString is config file**

Config file contains configuration settings for entire application

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />

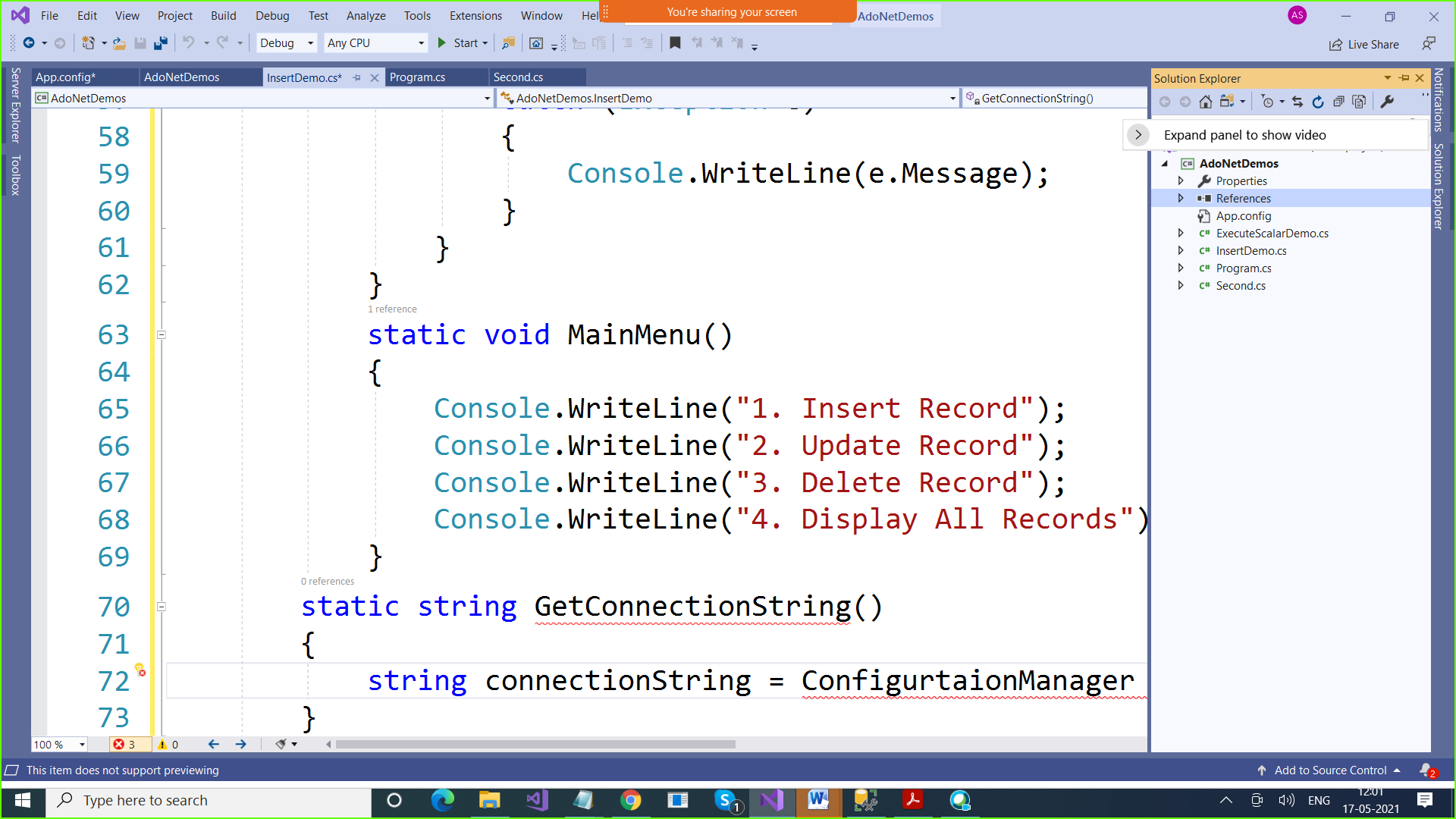
</startup>

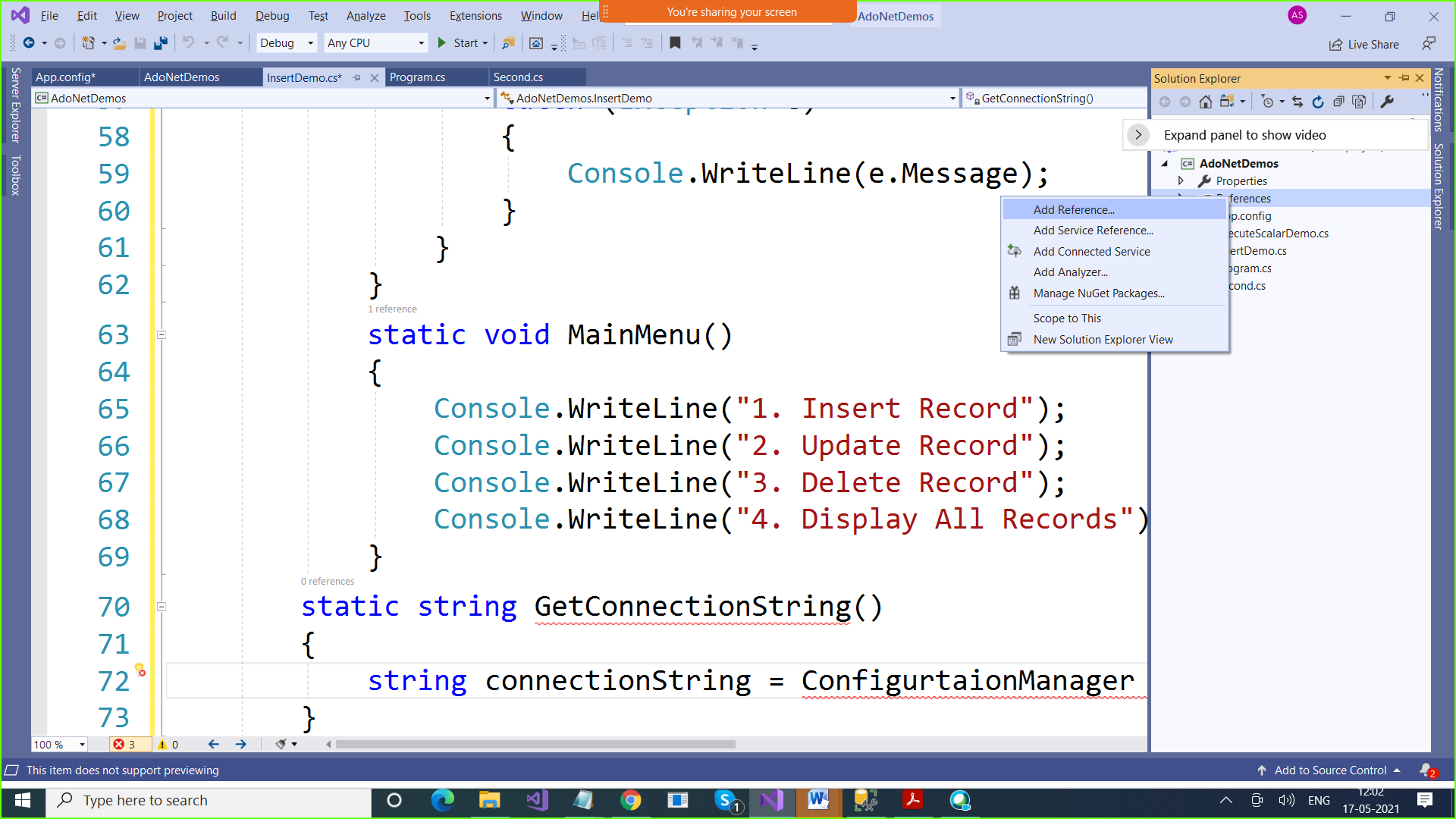
**<connectionStrings>**

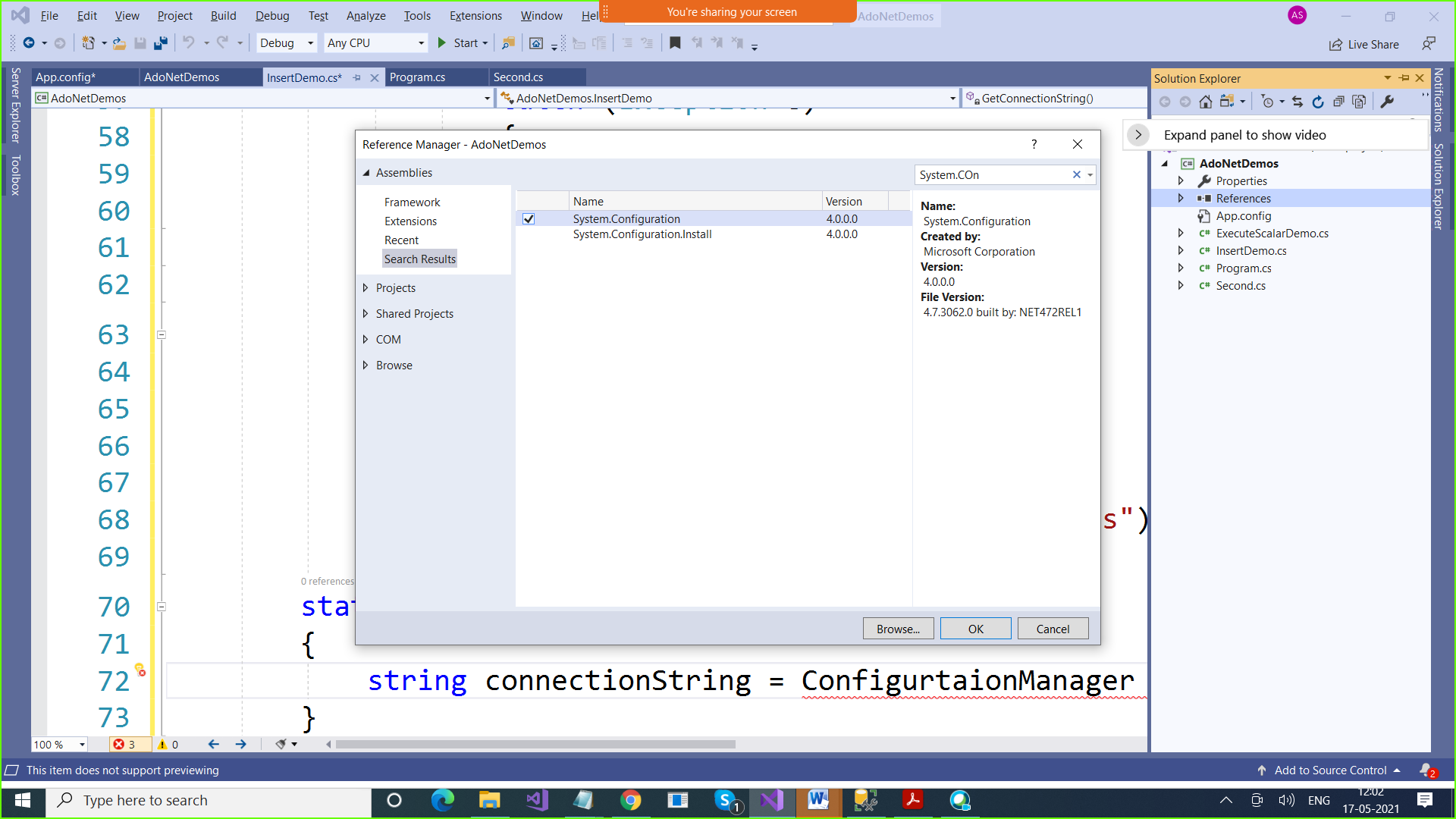
**<add name="MyConnection" connectionString="data source=LAPTOP-53S2KQS8;initial catalog=PracticeDb1;integrated security=true"/>**

**</connectionStrings>**

</configuration>







**We need using System.Configuration namespace**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

**using System.Configuration;**

namespace AdoNetDemos

{

enum choice { Insert=1 , Update, Delete, GetRecords};

class InsertDemo

{

static SqlConnection connection=null;

static void Main()

{

int Choice;

string ch = "y";

while (ch == "y")

{

try

{

MainMenu();

Console.WriteLine("Enter Your Choice");

Choice = Byte.Parse(Console.ReadLine());

switch (Choice)

{

case (int)choice.Insert:

{

InsertRecord();

break;

}

case (int)choice.Update:

{

UpdateRecord();

break;

}

case (int)choice.Delete:

{

DeleteRecord();

break;

}

case (int)choice.GetRecords:

{

GetRecords();

break;

}

default:

{

Console.WriteLine("Invalid Choice");

break;

}

}

Console.WriteLine("Do you want to repeat the process");

ch = Console.ReadLine();

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

static void MainMenu()

{

Console.WriteLine("1. Insert Record");

Console.WriteLine("2. Update Record");

Console.WriteLine("3. Delete Record");

Console.WriteLine("4. Display All Records");

}

**static string GetConnectionString()**

**{**

**string connectionString = ConfigurationManager.AppSettings**

**["MyConnection"].ToString();**

**return connectionString;**

**}**

static SqlConnection GetConnnection()

{

**connection = new SqlConnection(GetConnectionString());**

// connection = new SqlConnection("data source=LAPTOP-53S2KQS8;" +

// "initial catalog=PracticeDb1;integrated security=true");

return connection;

}

static void InsertRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

string name = Console.ReadLine();

Console.WriteLine("Enter Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Salary");

int salary = int.Parse(Console.ReadLine());

SqlCommand command = new SqlCommand("Insert into Employee (id, name , address, salary) values(@id, @name,@address,@salary)", connection);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@name", name);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records inserted are "+ count);

connection.Close();

}

static void UpdateRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID whose Record you want to modisy");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter New Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Revised Salary");

int salary = int.Parse(Console.ReadLine());

SqlCommand command = new SqlCommand("Update Employee" +

"set address =@adress, salary = @salary where is=@id", connection);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records updated are " + count);

connection.Close();

}

static void DeleteRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID whose Record you want to delete");

int id = Byte.Parse(Console.ReadLine());

SqlCommand command = new SqlCommand("Delete Employee" +

"where id=@id", connection);

command.Parameters.AddWithValue("@id", id);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records deleted are " + count);

connection.Close();

}

static void GetRecords()

{

connection = GetConnnection();

SqlCommand command = new SqlCommand("Select \* from Employee",

connection);

connection.Open();

SqlDataReader reader = command.ExecuteReader();

if(reader.HasRows)

{

while(reader.Read())

{

Console.WriteLine(reader["id"].ToString()+ " " + reader["name"]);

}

}

reader.Close();

connection.Close();

}

}

}

Connection.Close() > It will close the connection

But the connection object is still in memory

**command.Dispose();**

**connection.Dispose();**

**These two statements will destroy object. They will be removed from memory**

**Garbage Collection > Removing unused objects from memory**

**CLR does Garbage Collection**

**CLR will do garbage collection for only managed code**

**By Connections & Command objects are unmanaged**

**Which means their life cycle is not in control of CLR**

**So we have to explicitly destroy these objects**

**HOW DO WE EXPLICTLY DESTROY**

1. **Call Dispose Method**
2. **Make use of “Using” Block**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Configuration;

namespace AdoNetDemos

{

enum choice { Insert=1 , Update, Delete, GetRecords};

class InsertDemo

{

static SqlConnection connection=null;

static void Main()

{

int Choice;

string ch = "y";

while (ch == "y")

{

try

{

MainMenu();

Console.WriteLine("Enter Your Choice");

Choice = Byte.Parse(Console.ReadLine());

switch (Choice)

{

case (int)choice.Insert:

{

InsertRecord();

break;

}

case (int)choice.Update:

{

UpdateRecord();

break;

}

case (int)choice.Delete:

{

DeleteRecord();

break;

}

case (int)choice.GetRecords:

{

GetRecords();

break;

}

default:

{

Console.WriteLine("Invalid Choice");

break;

}

}

Console.WriteLine("Do you want to repeat the process");

ch = Console.ReadLine();

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

static void MainMenu()

{

Console.WriteLine("1. Insert Record");

Console.WriteLine("2. Update Record");

Console.WriteLine("3. Delete Record");

Console.WriteLine("4. Display All Records");

}

static string GetConnectionString()

{

string connectionString = ConfigurationManager.AppSettings

["MyConnection"].ToString();

return connectionString;

}

static SqlConnection GetConnnection()

{

connection = new SqlConnection(GetConnectionString());

// connection = new SqlConnection("data source=LAPTOP-53S2KQS8;" +

// "initial catalog=PracticeDb1;integrated security=true");

return connection;

}

static void InsertRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

string name = Console.ReadLine();

Console.WriteLine("Enter Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Salary");

int salary = int.Parse(Console.ReadLine());

SqlCommand command = new SqlCommand("Insert into Employee (id, name , address, salary) values(@id, @name,@address,@salary)", connection);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@name", name);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records inserted are "+ count);

connection.Close();

**command.Dispose();**

**connection.Dispose();**

}

static void UpdateRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID whose Record you want to modisy");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter New Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Revised Salary");

int salary = int.Parse(Console.ReadLine());

SqlCommand command = new SqlCommand("Update Employee" +

"set address =@adress, salary = @salary where is=@id", connection);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records updated are " + count);

connection.Close();

**command.Dispose();**

**connection.Dispose();**

}

static void DeleteRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID whose Record you want to delete");

int id = Byte.Parse(Console.ReadLine());

SqlCommand command = new SqlCommand("Delete Employee" +

"where id=@id", connection);

command.Parameters.AddWithValue("@id", id);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records deleted are " + count);

connection.Close();

**command.Dispose();**

**connection.Dispose();**

}

static void GetRecords()

{

connection = GetConnnection();

SqlCommand command = new SqlCommand("Select \* from Employee",

connection);

connection.Open();

SqlDataReader reader = command.ExecuteReader();

if(reader.HasRows)

{

while(reader.Read())

{

Console.WriteLine(reader["id"].ToString()+ " " + reader["name"]);

}

}

reader.Close();

connection.Close();

**command.Dispose();**

**connection.Dispose();**

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Configuration;

namespace AdoNetDemos

{

enum choice { Insert=1 , Update, Delete, GetRecords};

class InsertDemo

{

static SqlConnection connection=null;

static SqlCommand command = null;

static void Main()

{

int Choice;

string ch = "y";

while (ch == "y")

{

try

{

MainMenu();

Console.WriteLine("Enter Your Choice");

Choice = Byte.Parse(Console.ReadLine());

switch (Choice)

{

case (int)choice.Insert:

{

InsertRecord();

break;

}

case (int)choice.Update:

{

UpdateRecord();

break;

}

case (int)choice.Delete:

{

DeleteRecord();

break;

}

case (int)choice.GetRecords:

{

GetRecords();

break;

}

default:

{

Console.WriteLine("Invalid Choice");

break;

}

}

DestroyObjects();

Console.WriteLine("Do you want to repeat the process");

ch = Console.ReadLine();

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

static void MainMenu()

{

Console.WriteLine("1. Insert Record");

Console.WriteLine("2. Update Record");

Console.WriteLine("3. Delete Record");

Console.WriteLine("4. Display All Records");

}

static void DestroyObjects()

{

connection.Dispose();

command.Dispose();

}

static string GetConnectionString()

{

string connectionString = ConfigurationManager.AppSettings

["MyConnection"].ToString();

return connectionString;

}

static SqlConnection GetConnnection()

{

connection = new SqlConnection(GetConnectionString());

// connection = new SqlConnection("data source=LAPTOP-53S2KQS8;" +

// "initial catalog=PracticeDb1;integrated security=true");

return connection;

}

static void InsertRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

string name = Console.ReadLine();

Console.WriteLine("Enter Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Salary");

int salary = int.Parse(Console.ReadLine());

command = new SqlCommand("Insert into Employee (id, name , address, salary) values(@id, @name,@address,@salary)", connection);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@name", name);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records inserted are "+ count);

connection.Close();

}

static void UpdateRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID whose Record you want to modisy");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter New Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Revised Salary");

int salary = int.Parse(Console.ReadLine());

command = new SqlCommand("Update Employee" +

"set address =@adress, salary = @salary where is=@id", connection);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records updated are " + count);

connection.Close();

}

static void DeleteRecord()

{

connection = GetConnnection();

Console.WriteLine("Enter ID whose Record you want to delete");

int id = Byte.Parse(Console.ReadLine());

command = new SqlCommand("Delete Employee" +

"where id=@id", connection);

command.Parameters.AddWithValue("@id", id);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records deleted are " + count);

connection.Close();

}

static void GetRecords()

{

connection = GetConnnection();

command = new SqlCommand("Select \* from Employee",

connection);

connection.Open();

SqlDataReader reader = command.ExecuteReader();

if(reader.HasRows)

{

while(reader.Read())

{

Console.WriteLine(reader["id"].ToString()+ " " + reader["name"]);

}

}

reader.Close();

connection.Close();

}

}

}

**static void InsertRecord()**

**{**

**using (SqlConnection connection = GetConnnection())**

**{**

**Console.WriteLine("Enter ID");**

**int id = Byte.Parse(Console.ReadLine());**

**Console.WriteLine("Enter Name");**

**string name = Console.ReadLine();**

**Console.WriteLine("Enter Address");**

**string address = Console.ReadLine();**

**Console.WriteLine("Enter Salary");**

**int salary = int.Parse(Console.ReadLine());**

**using (SqlCommand command = new SqlCommand("Insert into Employee (id, name , address, salary) values(@id, @name,@address,@salary)", connection))**

**{**

**command.Parameters.AddWithValue("@id", id);**

**command.Parameters.AddWithValue("@name", name);**

**command.Parameters.AddWithValue("@address", address);**

**command.Parameters.AddWithValue("@salary", salary);**

**connection.Open();**

**int count = command.ExecuteNonQuery();**

**Console.WriteLine("No of Records inserted are " + count);**

**connection.Close();**

**}**

**}**

**}**

**Destroy SqlCollection & SqlCommand objects by “Using” Block**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Configuration;

namespace AdoNetDemos

{

enum choice { Insert=1 , Update, Delete, GetRecords};

class InsertDemo

{

static void Main()

{

int Choice;

string ch = "y";

while (ch == "y")

{

try

{

MainMenu();

Console.WriteLine("Enter Your Choice");

Choice = Byte.Parse(Console.ReadLine());

switch (Choice)

{

case (int)choice.Insert:

{

InsertRecord();

break;

}

case (int)choice.Update:

{

UpdateRecord();

break;

}

case (int)choice.Delete:

{

DeleteRecord();

break;

}

case (int)choice.GetRecords:

{

GetRecords();

break;

}

default:

{

Console.WriteLine("Invalid Choice");

break;

}

}

Console.WriteLine("Do you want to repeat the process");

ch = Console.ReadLine();

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

static void MainMenu()

{

Console.WriteLine("1. Insert Record");

Console.WriteLine("2. Update Record");

Console.WriteLine("3. Delete Record");

Console.WriteLine("4. Display All Records");

}

static string GetConnectionString()

{

string connectionString = ConfigurationManager.AppSettings

["MyConnection"].ToString();

return connectionString;

}

static SqlConnection GetConnnection()

{

SqlConnection connection = new SqlConnection(GetConnectionString());

// connection = new SqlConnection("data source=LAPTOP-53S2KQS8;" +

// "initial catalog=PracticeDb1;integrated security=true");

return connection;

}

static void InsertRecord()

{

using (SqlConnection connection = GetConnnection())

{

Console.WriteLine("Enter ID");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

string name = Console.ReadLine();

Console.WriteLine("Enter Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Salary");

int salary = int.Parse(Console.ReadLine());

using (SqlCommand command = new SqlCommand("Insert into Employee (id, name , address, salary) values(@id, @name,@address,@salary)", connection))

{

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@name", name);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records inserted are " + count);

connection.Close();

}

}

}

static void UpdateRecord()

{

using (SqlConnection connection = GetConnnection())

{

Console.WriteLine("Enter ID whose Record you want to modisy");

int id = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter New Address");

string address = Console.ReadLine();

Console.WriteLine("Enter Revised Salary");

int salary = int.Parse(Console.ReadLine());

using (SqlCommand command = new SqlCommand("Update Employee" +

"set address =@adress, salary = @salary where is=@id", connection))

{

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@address", address);

command.Parameters.AddWithValue("@salary", salary);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records updated are " + count);

connection.Close();

}

}

}

static void DeleteRecord()

{

using (SqlConnection connection = GetConnnection())

{

Console.WriteLine("Enter ID whose Record you want to delete");

int id = Byte.Parse(Console.ReadLine());

using (SqlCommand command = new SqlCommand("Delete Employee" +

"where id=@id", connection))

{

command.Parameters.AddWithValue("@id", id);

connection.Open();

int count = command.ExecuteNonQuery();

Console.WriteLine("No of Records deleted are " + count);

connection.Close();

}

}

}

static void GetRecords()

{

using (SqlConnection connection = GetConnnection())

{

using (SqlCommand command = new SqlCommand("Select \* from Employee",

connection))

{

connection.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Console.WriteLine(reader["id"].ToString() + " " + reader["name"]);

}

}

reader.Close();

connection.Close();

}

} }

}

}