Accessing databases using C# code

ADO Connected Lab

Objective

In this lab you will use the ADO Connected methods to extract customer records from the Northwind database and display the data using a Window’s form.

Please make sure the Northwind database is installed.   
Double click on the provided Northwind.sql file if you've not installed Northwind.  
Alternatively you can choose another database which you have installed in Microsoft SQL server

In this lab you will display all the **CustomerID, ContactName**s from the **Customers** table. When the user types a CustomerID, your application will display the customer’s details.

## Part 1 – Write code to read data from a table

Back in the Labs Console application, create a method called Lab12() and then call it from within the Main() method.   
A graphical Forms app would be better for this type of app but we want to keep it simple and focus on the coding part.

Type the following code the Lab12() method (extracted from the slide code)  
Please study the code to see what it does.  
  
  
private static void Lab12()  
{  
 SqlConnection cn = new SqlConnection(

@"Data Source=.\sqlexpress;Initial Catalog=Northwind;Integrated Security=True");  
 SqlCommand com = new SqlCommand("SELECT TOP 10 CustomerID, ContactName FROM customers", cn);

cn.Open();

SqlDataReader dr = com.ExecuteReader();

while (dr.Read())  
 {  
 Console.WriteLine(dr[0] + " - " + dr[1]);  
 }

cn.Close();

}  
  
There are a lot of compilation errors here because you’ve not yet imported the necessary libraries.  
Press **Ctrl-.** (control dot) on say the **SqlConnection** word and install package ‘**System.Data.SqlClient’**  
(see below for more help)

Graphical user interface, text, application, email

Description automatically generated

## Part 2 – Show the selected customer’s detail

In the Lab12() method after calling the cn.Close(), prompt the user to enter a customer’s id.

CustomerID in this table is held as a string (like *ANTON*)

After getting an ID from the user, display 4/5 columns’ data from the customers table.

We’ve provided the code for this part at the end of this file but try to do this without looking at the sample code.

## Part 3 – Make changes (if you have time)

Create a new method to insert a new customer into the **Customers** table.   
Note: CustomerID is alphabetic and should be 5 characters. It must also be unique (is a Primary Key)

Use the code in Lab12() to display the new customer you just inserted into the table

**Delete the new customer**

See if you can delete the new customer you just added.

Note: You will not be able to delete any other customer because they all have related records in the Orders table and the Referential Integrity constraint stops deletion of such a record.

**\*\* End \*\***

See the completed code sample below

private static void Lab12()

{

SqlConnection cn = new SqlConnection(

@"Data Source=.\sqlexpress;Initial Catalog=Northwind;Integrated Security=True");

SqlCommand com = new SqlCommand("SELECT TOP 10 CustomerID, ContactName FROM customers", cn);

cn.Open();

SqlDataReader dr = com.ExecuteReader();

while (dr.Read())  
 {   
 Console.WriteLine(dr[0] + " - " + dr[1]);  
 }

dr.Close(); // connection is left open and only the data reader is closed

Console.Write("Type a customer ID: ");

string id = Console.ReadLine();

com = new SqlCommand($"SELECT \* FROM customers WHERE CustomerID='{id}'", cn);  
  
 try {

dr = com.ExecuteReader();

dr.Read();

Console.WriteLine($"{dr[0]}\n{dr[1]}\n{dr[2]}\n{dr[3]}\n{dr[4]}\n{dr[5]}\n");

}

catch (Exception ex)   
 {  
 Console.WriteLine($"No customer found with the id of {id}");  
 Console.WriteLine(ex.Message);

}

finally  
 {  
 cn.Close();  
 }

}

Feel free to refactor the above code by separating out the connection string and the SQL statements into variables to make the code easier to read or by creating new methods to perform these tasks.