الفصل الثالث عشر

التعامل مع قواعد البيانات

Database Processing

Programming Concepts

في هذا الفصل

13

في هذا الفصل نبدأ الاستعمال الفعلي لـ VB.Net للتعامل مع قواعد البيانات وذلك من خلال النقاط التالية:

- الطرق المتاحة في VB.Net للتعامل مع قواعد البيانات
- التعامل مع قواعد البيانات من خلال Server Explore
- استخدام المعالجات DB Wizards
- استعمال أوامر الاتصال المباشر Direct Connection
- أدوات عرض البيانات DB Controls
- استعمال أوامر الاتصال غير المباشر Off Line DataBase
- علاقة قواعد البيانات بلغة XML

يتناول هذا الفصل الطرق المختلفة للتعامل مع قواعد البيانات من خلال لغات البرمجة وهى كما أشرنا من الموضوعات المهمة جدا فى عالم البرمجة فى الوقت الحاضر.

فى لغة VB.Net و C#

تعتبر طرق وقواعد واحدة مع وجدو الاختلاف الوحيد وهو طريقة كتابة الاوامر كما سبق وسوف نوضح ذلك أيضا.

يمكن التعامل مع قواعد البيانات من خلال Server Explore وفيها يتم التعامل مع قواعد البيانات من داخل بيئة التطوير كمطور وليس كمستخدم حيث تستطيع عن طريق هذه النافذة Server Explore فتح قواعد البيانات والتعامل مع عناصرها المختلفة.

استخدام المعالجات DB Wizards

يوفر VB.Net معالجات لانشاء النماذج التى تعتمد على قاعدة بيانات مع توفير العمليات الاساسية مثل عمليات الاضافة Insert والحذف Delete والتعديل Update

استعمال أوامر الاتصال المباشر Direct Connection

يوفر VB.Net مجموعة من الفصائل Classes التى توفر امكانية استعمال أوامر الاتصال المباشر Direct Connection بقاعدة البيانات حيث تستطيع فتح قاعدة البيانات واصدار الاوامر المختلفة لها مثل الفصيلة Connection class والفصيلة Command class وفصائل أخرى سوف نتناولها بالتفصيل.

أدوات عرض البيانات DB Controls

مجموعة من الأدوات التى يمكن ربطها بقاعدة البيانات لعرض البيانات فيها مثل الأداة الشهيرة Data Grid استعمال أوامر الاتصال غير المباشر Off Line DataBase.

يوفر VB.Net امكانية التعامل مع قاعدة البيانات بطريقة أوامر الاتصال غير المباشر Off Line DataBase وهذا يعنى عمل نسخة من البيانات فى ذاكرة الجهاز ثم القيام بالتعامل معها بالإضافة والحذف والتعديل فى انتظار أمر التحديث فيتم تحديث قاعدة البيانات الأصلية وهذا ما يسمى Off Line بحيث أننا لا نتعامل مع قاعدة البيانات مباشرة بل تعامل مع النسخة ، ويتم ذلك باستعمال فصيلة مشهورة فى VS.Net تسمى DataSet.

علاقة قواعد البيانات بلغة XML

لغة XML لغة حديثة تستخدم لتمثيل البيانات بما لها من ميزات عن لغة HTML ويوفر VS.Net تكامل كبير معها.

استعمال أدوات وفصائل وأوامر ADO.NET

وهي الطريقة الأخيرة والحديثة حتى لأن والموجودة في VB.NET بما توفره من تعديلات للفصيلة ADO وخصائص جديدة.

أدوات عرض البيانات DB Controls

يوجد مجموعة من الأدوات التي يمكن ربطها بقاعدة البيانات لعرض البيانات فيها مثل الأداة الشهيرة DataViewGrid وسوف نتناول ذلك فيما بعد .

استعمال أوامر الاتصال المباشر Direct Connection

يوفر VB.Net مجموعة من الفصائل Classes التي توفر امكانية استعمال أوامر الاتصال المباشر Direct Connection بقاعدة البيانات حيث تستطيع فتح قاعدة البيانات واصدار الاوامر المختلفة لها مثل الفصيلة Connection class والفصيلة Command class وفصائل أخرى سوف نتناولها بالتفصيل.

تعريفات

1ـ الفصيلة OleDbConnection

تستخدم هذه الفصيلة في تحديد مواصفات الاتصال Connection بقاعدة البيانات مثل اسم الجهاز وقاعدة البيانات وكلمة السر وغيره عن طريق الخاصية ConnectionString ثم فتح قاعدة البيانات باستعمال الدالة ()Open.

ويوجد من هذه الفصيلة شكلان:

- الأول هي OleDbConnection: وتستخدم عند الرغبة في لتعامل مع أي نوع من أنواع قواعد البيانات (Ms Access,Oracle,SQL Server)

وهي معرفة في NameSpace بالاسم OleDb داخل NameSpace بالاسم Data داخل

NameSpace بالاسم System لذلك للتعامل مع هذه الفصيلة لابد من كتابة السطر التالى فى أول البرنامج قبل تعريف الفصيلة.

فى VB.NET

```
Imports System.Data.OleDb
```

فى C#

```
using System.Data.OleDb
```

الفصيلة SqlConnection

وهى تستخدم عند الرغبة فى التعامل مع قاعدة بيانات من النوع SQL Server وقد أعدتها ميكروسوفت لتسهل التعامل مع قواعد البيانات الخاصة بها من النوع SQL Server حيث تعتبر هى النوع الافتراضى لديها.

ولاستعمال هذه الفصيلة لابد من كتابة السطر التالى فى أول البرنامج

```
Imports System.Data.SqlClient
```

فى لغة C#

```
using System.Data.SqlClient
```

2. الفصيلة OleDbCommand

وتستعمل فى تحديد جملة SQL التى تحدد البيانات التى ستتعامل معها مع تحديد متغير (هدف Object) فصيلة الاتصال بقاعدة البياناتConnection التى نتعامل معها وتأخذ صورتين مثل الفصيلة OleDbConnection.

الأول:**OleDbCommand**

وهى للتعامل مع أى نوع من أنواع قواعد البيانات.

والثانية **SqlCommand**

للتعامل مع قواعد البيانات من النوع SQl Server.

3. الفصيلة OleDbDataReader

تستخدم هذه الفصيلة لقراءة البيانات التى تم تحديدها باستخدام الفصيلة OleDbCommand

وذلك باستعمال الدالة ()ExecuteReader العضوة في الفصيلة OleDbCommand وضع النتيجة في هدف الفصيلة OleDbDataReader ثم التعامل مع هذه البيانات وتأخذ صورتين مثل الفصيلة OleDbConnection.

- **الأولى OleDbDataReader**

كما ذكرنا للتعامل مع أي نوع من قواعد البيانات.

- **والثانية SqlDataReader**

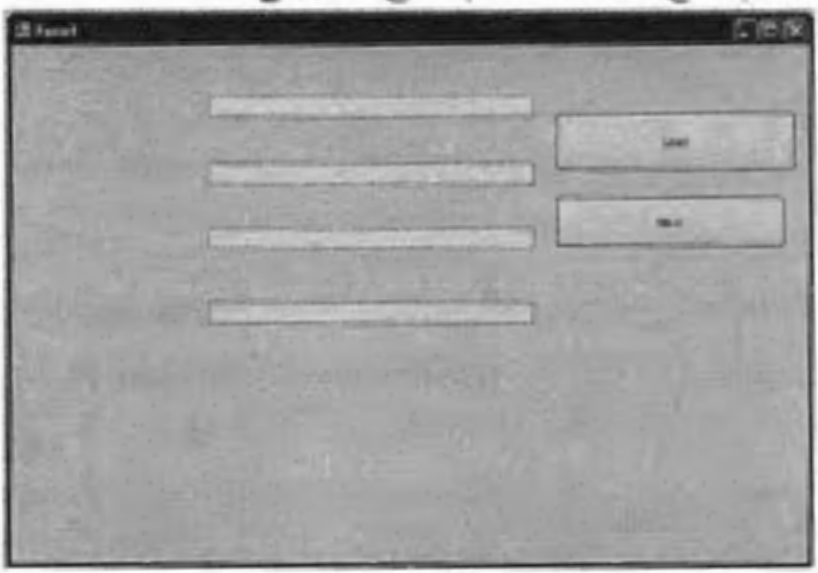
للتعامل مع قواعد البيانات من النوع SQl Server.

ويتم توضيح هذه النقاط بمثال كما في الخطوات التالية:

مثال

1- أبدأ تطبيق جديد من لنوع Windows Application كما سبق.

2- قم بتوقيع الأدوات وتصميم نموذج للتعامل مع البيانات كما في الشكل (1-13).



الشكل(1- 13)

3-اضغط بالماوس مرتين فوق النموذج Form ولكتب في أول الملف سطور فتح NameSpace للتعامل مع فصائل قواعد البيانات كما في الشكل (2-13).

فى VB.NET

```
1  Imports System.Data
2  Imports System.Data.OleDb
3
4
5  Public Class Form3
```

الشكل(2- 13)

فى لغة C#

```
1  using System;
2  using System.Drawing;
3  using System.Collections;
4  using System.ComponentModel;
5  using System.Windows.Forms;
6  using System.Data;
7  using System.Data.OleDb ;
8
9  namespace WindowsApplication14
```

- فى السطر رقم 2 تم استعمال الامر Imports لفتح الـ OleDb NameSpace كما أشرنا من قبل.

4- قم بكتابة سطور تعريف أهداف Objects من الفصيلة OleDbConnection للتعامل مع قاعدة البيانات والفصيلة OleDbDataReader لقراءة البيانات كما فى الشكل.

فى VB.NET

```
5  Public Class Form3
6      Inherits System.Windows.Forms.Form
7      Dim objConnection As New OleDbConnection()
8      Dim myReader As OleDbDataReader
9
```

فى لغة C#

```
14    public class Form1 : System.Windows.Forms.Form
15    {
16
17        OleDbConnection objConnection=new OleDbConnection();
18        OleDbDataReader myReader ;
19
```

5- اضغط زر الأمر المكتوب عليه Load Data والذى نستخدم فى فتح قاعدة البيانات وتحميل لبيانات أول مرة واكتب السطر التالى:

فى VB.NET

```
objConnection.ConnectionString = ("PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA
SOURCE=c:\Northwind.mdb")
```

فى لغة C#

```
objConnection.ConnectionString=
"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=c:\\Northwind.mdb";
```

- فى هذا السطر يتم تحديد مواصفات قاعدة البيانات التى نتعامل معها بتحديد قيمة المتغير ConnectionString لهدف الفصيلة OleDbConnection المعرف بالاسم ObjConnection وفيها يتم تحديد.

5-1 نوع قاعدة البيانات بالعبارة ;ROVIDER=Microsoft.Jet.OLEDB.4.0 حيث تعبر عن MsAccess ويتم تغييرها عند التعامل مع قاعدة بيانات أخرى.

5-2 تحديد قاعدة البانات التى نرغب فى التعامل معها وذلك بكتابة العبارة DATA SOURCE=c:\Northwind.mdb وفيها يتم استعمال الجملة Data source ثم تحديد اسم ومسار قاعدة البيانات وفى حالتنا هى c:\Northwind.mdb وهى قاعدة البانات التى تضيفها ميكروسوت كمثال داخل برنامج قواعد البيانات Ms Access.

ملحوظة

للتعامل مع قاعدة بيانات من النوع SQL Server يتم استبدال مواصفات فتح قاعدة البيانات فقط وجميع العمليات كما هي وتأخذ الصفات فتح قاعدة البيانات الصورة التالية:

```
Dim cn As New SqlClient.SqlConnection()
cn.ConnectionString = "data source=server;initial catalog=EMPLOYEE;user ID=sa"
```

- في هذه السطور تم تعريف المتغير cn من نوع SqlConnection.
- ثم تحديد مواصفات الاتصال بتحديد قيمة الخاصية ConectionString وفيها.
- Data Source تساوى القيمة Server وهو اسم الجهاز الخادم Server الموجود عليه قاعدة البيانات.
- Initial catalog تحدد اسم قاعدة البيانات وفى حالتنا هى EMPLOYEE.
- user ID اسم المستخدم المعرف فى SQl Server والمستخدم للدخول على قاعدة البيانات EMPLOYEE وهو فى حالتنا sa وهو المستخدم المعرف فى SQl Server وبالتالى تصبح سطور البرنامج هكذا:

```
Sub Main()

Dim conn1 As New SqlConnection()
conn1.ConnectionString = "Data Source=Azab;Initial Catalog=CSO;Persist
Security Info=True;User ID=sa;Password=azabazab"
Dim cmd As New SqlCommand
cmd.CommandType = CommandType.Text
cmd.CommandText = "SELECT CustomerID, CompanyName FROM Customers"

cmd.Connection = conn1
' Create a SqlParameter for each parameter in the stored procedure.

Dim reader As SqlDataReader
Dim previousConnectionState As ConnectionState = conn1.State
Try
If conn1.State = ConnectionState.Closed Then
conn1.Open()
End If
reader = cmd.ExecuteReader()
Using reader
While reader.Read
' Process SprocResults datareader here.
Console.Write(reader.GetValue(0))
```

```
Console.WriteLine("----" + reader.GetValue(1))

End While
End Using
Finally
If previousConnectionState = ConnectionState.Closed Then
conn1.Close()
End If
End Try

Console.ReadKey()
End Sub
```

6-اكتب سطور فتح قاعدة البيانات.

```
conn.ConnectionString = ("PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA
SOURCE=c:\Northwind.mdb")
conn.Open()
MsgBox("Opened")

Dim strsql As String
strsql = "select * from Employees"

cmd.CommandType = CommandType.Text
cmd.CommandText = "SELECT * FROM Customers"

cmd.Connection = conn
' Create a SqlParameter for each parameter in the stored procedure.

Dim previousConnectionState As ConnectionState = conn.State
Try
If conn.State = ConnectionState.Closed Then
conn.Open()
End If
reader = cmd.ExecuteReader()
reader.Read()

TextBox1.Text = reader.GetValue(0)
TextBox2.Text = reader.GetValue(1)
TextBox3.Text = reader.GetValue(2)
TextBox4.Text = reader.GetValue(3)
Finally
If previousConnectionState = ConnectionState.Closed Then
conn.Close()
End If
End Try
```

فى لغة C#

```
158        private void button1_Click(object sender, System.EventArgs e)
159        {
160            objConnection.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;I
161            objConnection.Open ();
162            string strSQL;
163            strSQL="SELECT EmployeeID,LastName,FirstName,Title FROM Employees OR
164            OleDbCommand myCommand = new OleDbCommand(strSQL,objConnection);
165            myReader = myCommand.ExecuteReader();
166            myReader.Read();
167
168            textBox1.Text=myReader.GetInt32   (0)+"";
169            textBox2.Text =myReader.GetString (1);
170            textBox3.Text =myReader.GetString (2);
171            textBox4.Text =myReader.GetString (3);
172
```

فى هذه السطور

- فى السطر 112 تم تعريف مواصفات قاعدة البيانات كما سبق وشرحنا.
- فى السطر رقم 113 تم فتح قاعدة البيانات باستدعاء الدالة ()Open مع متغير وصلة قاعدة البيانات objConnection.
- فى السطر 115 تم الاعلان عن متغير strSQL من نوع string.
- فى السطر رقم 116 تم تحديد جملة الـ SQL المطلوب تنفيذها وذلك بوضعها فى المتغير strSQL.
- فى السطر 118 تم تعريف المتغير objCommand من نوع الفصيلة OleDbCommand التى تتعامل مع جملة الـ SQL وتنفذها مع ارسال جملة الـ SQL التى حددناها كمعامل وبالتالى أصبح المتغير objCommand يشير الى جملة الـ SQL المحددة.
- فى السطر رقم 119 تم استدعاء الدالة ExecuteReader مع المتغير objCommand وبالتالى تنفيذ جملة الـ SQL المحددة مسبقا ووضع النتيجة فى المتغير myReader

المعرف فى أول البرنامج من النوع OleDbReader والذى يقوم بقراءة بيانات من قاعدة بيانات.

- فى السطر رقم 120 يتم استدعاء الدالة Read() مع المتغير myReader والتى تقوم بقراءة سجل Record.
- فى السطور 122و123و124و125 يتم وضع قيم الحقول Fields المحددة (EmployeeID,FirstName,LastName,Title) فى مربعات النصوص TextBox1,TextBox2,TextBox3,TextBox4 وبالتالى تظهر بيانات أول سجل فى الشاشة.

7-اكتب سطر زر الامر الثانى الذى يعرض السجل التالى:

```
reader.Read()

TextBox1.Text = reader.GetValue(0)
TextBox2.Text = reader.GetValue(1)
TextBox3.Text = reader.GetValue(2)
TextBox4.Text = reader.GetValue(3)
```

فى لغة C#

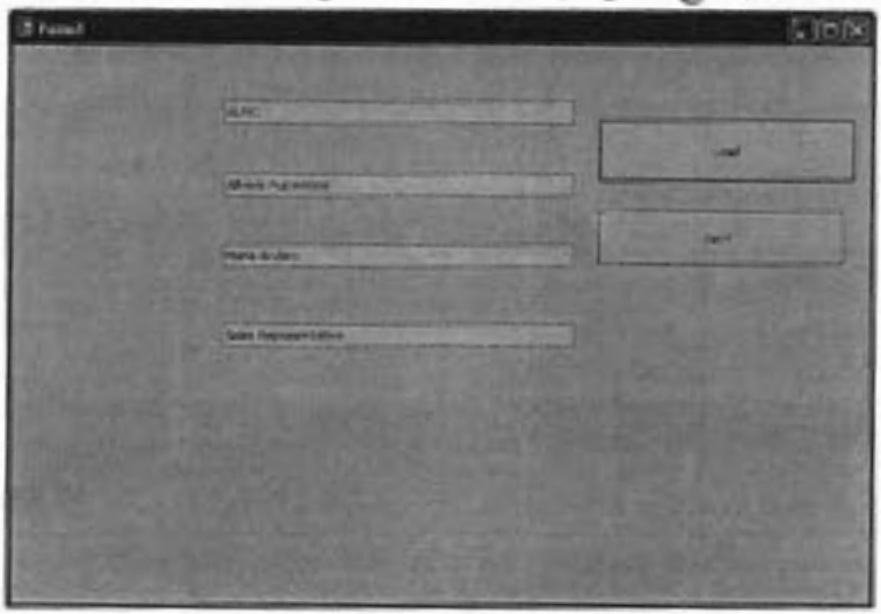
```
175        private void button2_Click(object sender, System.EventArgs e)
176        {
177            myReader.Read();
178
179            textBox1.Text=myReader.GetInt32   (0)+"";
180            textBox2.Text =myReader.GetString (1);
181            textBox3.Text =myReader.GetString (2);
182            textBox4.Text =myReader.GetString (3);
183
184        }
```

فى هذه السطور:

- فى السطر رقم 130 يتم استدعاء الدالة Read() مع المتغير myReader والتى تقوم بقراءة سجل Record.

- فى السطور 131و132و133و134 يتم وضع قيم الحقول Fields المحددة(EmployeeID,FirstName,LastName,Title) فى مربعات النصوص TextBox1,TextBox2,TextBox3,TextBox4 وبالتالى تظهر بيانات السجل التالى فى الشاشة كلما تم الضغط على هذا الزر.

8- نفذ البرنامج لتحصل على النتيجة التالية الشكل (3-13):



الشكل(3- 13)

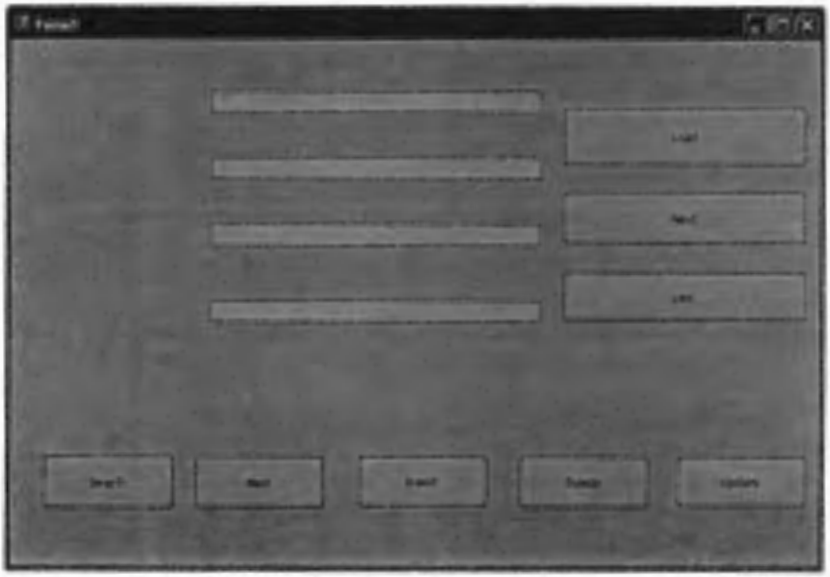
تحقيق عمليات قواعد البيانات

من الضرورى تحقيق العمليات الأساسية لقواعد البيانات وهى:

- الاضافة Insert والحذف Delete والتعديل Update والبحث ويتم ذلك بسهولة بتغيير جملة الـ SQL التى تمثل العملية المطلوبة.

وقد تم شرح جمل SQL المسؤلة عن هذه العمليات ولتوضيح كيفية تحقيق ذلك تابع معنا الخطوات التالية:

1-عد الى البرنامج السابق وقم باعادة تصميم نموذج البيانات الذى صممناه ليصبح كما فى الشكل (4-13).



الشكل(4- 13)

ف هذا الشكل نلاحظ اننا أضفنا أزرار أوامر Buttons للعمليات المطلوبة ويتبقى كتابة الأوامر التى تحققها كما يلى:

2- اكتب أوامر العمليات التالية.

زر البحث Search

يسمح هذا الزر للمستخدم بكتابة كود الموظف EmployeeID المطلوب البحث عنه ف مربع النص الأول TextBox1 ثم الضغط على الزر Search فتظهر بيانات هذا الموظف وتظهر سطور أوامر هذه العملية كالتالى:

```
Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click

conn.ConnectionString = ("PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA
SOURCE=c:\Northwind.mdb")
cmd.CommandType = CommandType.Text
Dim custIdTxtx As String
custIdTxtx = TextBox1.Text + ""

cmd.CommandText = "SELECT * FROM Customers where CustomerID='" +
custIdTxtx
cmd.Connection = conn
```

```
' Create a SqlParameter for each parameter in the stored procedure.
Dim previousConnectionState As ConnectionState = conn.State
Try
If Not conn.State = ConnectionState.Closed Then

conn.Close()

End If

If conn.State = ConnectionState.Closed Then
conn.Open()
End If
reader = cmd.ExecuteReader()
reader.Read()

TextBox1.Text = reader.GetValue(0)
TextBox2.Text = reader.GetValue(1)
TextBox3.Text = reader.GetValue(2)
TextBox4.Text = reader.GetValue(3)


Finally
If previousConnectionState = ConnectionState.Closed Then
conn.Close()
End If
End Try
End Sub
```

فى لغة C#

```
string strSQL;
strSQL="SELECT EmployeeID,LastName,FirstName,Title FROM
Employees WHERE EmployeeID="+textBox1.Text;
OleDbCommand myCommand=new
OleDbCommand(strSQL,objConnection);
myReader = myCommand.ExecuteReader();
myReader.Read();
textBox1.Text=myReader.GetInt32   (0)+"";
textBox2.Text =myReader.GetString (1);
textBox3.Text =myReader.GetString (2);
textBox4.Text =myReader.GetString (3);
myReader.Close ();
```

فى هذه السطور:

هذه هى نفس سطور أوامر زر أمر عرض السجل التالى ولكن الجديد هو:

1- وضع محتوى مربع النص TextBox1 وهو الرقم المطلوب البحث به فى المتغير srchtxt وذلك فى السطر رقم 190.

2-تعديل جملة الـ SQL باضافة جزء الشرط Where الذى يحدد السجلات المطلوب عرضها كما فى السطر 191.

مع جمع محتوى المتغير srchtxtعلى جملة السSQL وبالتالى يتم الحصول على السجلات التى فيها قيمة الحقل EmployeeID تساوى القيمة المطلوب البحث عنها والموجودة فى المتغير srchtxt والمأخوزة من مربع النص TextBox1 وباقى السطور كما سبق.

زر سجل جديد New

هذا الزر لا يحقق أحد عمليات قواعد البيانات فقط يؤدى الى مسح محتويات مربعات النصوص TextBox لتسمح للمستخدم بكتابة بيانات جديدة أو لكتابة كود موظف للبحث عنه ولذلك فأن سطوره بسيطة وهى كالتالى:

```
Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
    TextBox1.Text = ""
    TextBox2.Text = ""
    TextBox3.Text = ""
    TextBox4.Text = ""
End Sub
```

```
260         private void button4_Click(object sender, System.EventArgs e)
261         {
262             textBox1.Text="";
263             textBox2.Text ="";
264             textBox3.Text ="";
265             textBox4.Text ="";
266         }
```

زر الاضافة Insert

يسمح هذا الزر للمستخدم بكتابة بيانات جديدة وحفظها بقاعدة البيانات كما في السطور التالية:

```
conn.ConnectionString = ("PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA
SOURCE=c:\Northwind.mdb")
cmd.CommandType = CommandType.Text

cmd.CommandText = "Insert Into Customers
(CustomerID,CompanyName,ContactName,ContactTitle) values
('code1','CSO','Omr','Mis')"
cmd.Connection = conn
' Create a SqlParameter for each parameter in the stored procedure.
Dim previousConnectionState As ConnectionState = conn.State
If Not previousConnectionState = ConnectionState.Closed Then
conn.Close()
End If
conn.Open()
cmd.ExecuteNonQuery()

conn.Close()
```

```
269        private void button5_Click(object sender, System.EventArgs e)
270        {
271            string str0GA;
272    str0GA="INSERT INTO Employees(EmployeeID,LastName,FirstName,Title) VALUES(99,'Omr','sss','dddd')
273            OleDbCommand myCommand = new OleDbCommand(str0GA,objConnection);
274            myCommand.ExecuteNonQuery ();
275
276
277        }
```

في هذه السطور:

- تم تغيير جملة الـ SQL كما في السطر 227 الى جملة Insert Into التي شرحناها من قبل والمسؤلة عن إضافة سجل جديد مع تثبيت القيم المطلوب إضافتها وعدم أخذها من مربعات النصوص من الشاشة وذلك للتجربة.

- فى السطر 229 تم استدعاء الدالة ExecuateNonQuery() التى تقوم بتنفيذ جمل SQL غير الاستعلامية أى التى لا تحتوى على الأمر Select مثل أوامر الاضافة Insert وأوامر الحذف Delete والتحديث Update وغيره.
- وبالتالى يقوم هذا الزر باضافة سجل جديد وحفظ القيم المحددة فى جملة Insert به ولكن المطلوب حفظ القيم التى يكتبها المستخدم فى مربعات النصوص TextBox لذلك قم بتعديل سطور أوامر الزر Insert كما يلى:

زر الاضافة بعد التعديل Insert

فى الفقرة السابقة قمنا بإضافة قيم محددة وذلك لتجربة عملية الإضافة فقط ولكن الطبيعى هو استقبال القيم المطلوب إضافتها من المستخدم ثم إضافتها لذلك قم بتعديل سطور عملية الاضافة Insert لتصبح كما فى السطور التالية:

```
Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
        conn.ConnectionString = ("PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA
SOURCE=c:\Northwind.mdb")
        cmd.CommandType = CommandType.Text
        Dim txt1, txt2, txt3, txt4, vals As String

        txt1 = "('" + TextBox1.Text + "'"
        txt2 = ",'" + TextBox2.Text + "'"
        txt3 = ",'" + TextBox3.Text + "'"
        txt4 = ",'" + TextBox4.Text + "')"

        vals = txt1 + txt2 + txt3 + txt4

        Dim sql1 As String
        sql1 = "Insert Into Customers
(CustomerID,CompanyName,ContactName,ContactTitle) values "
        Dim lastSQL As String
        lastSQL = sql1 + vals
        MsgBox(lastSQL)
        cmd.CommandText = lastSQL
        cmd.Connection = conn
        ' Create a SqlParameter for each parameter in the stored procedure.
        Dim previousConnectionState As ConnectionState = conn.State
        If Not previousConnectionState = ConnectionState.Closed Then
            conn.Close()
```

```
    End If
    conn.Open()
    cmd.ExecuteNonQuery()
    MsgBox("Record Inserted ..")

    conn.Close()
End Sub
```

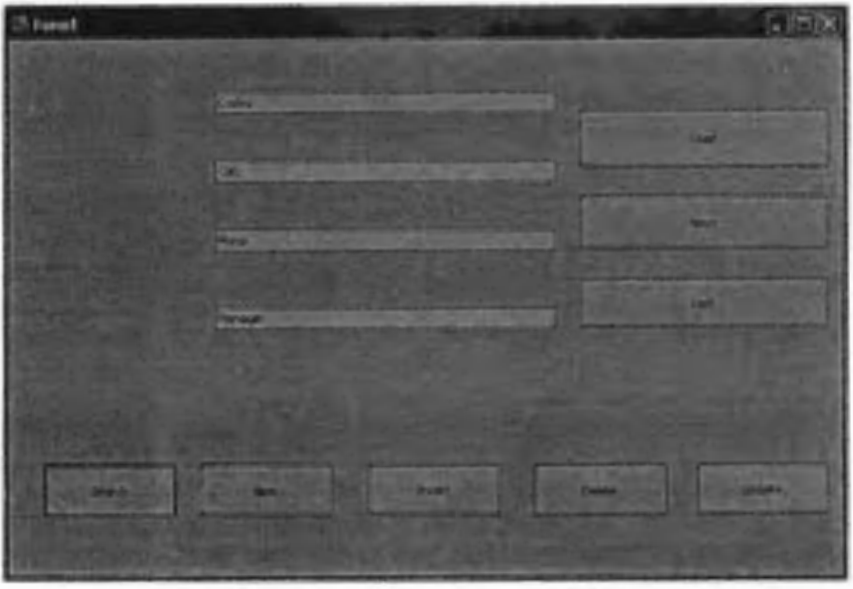
```
226         private void button5_Click(object sender, System.EventArgs e)
227         {
228 string str2,s2;
229 int EmployeeID_v;
230 string LastName_v;
231 string FirstName_v;
232 string Title_v;
233
234 EmployeeID_v=Int32.Parse (textBox1.Text) ;
235 LastName_v=textBox2.Text ;
236 FirstName_v=textBox3.Text ;
237 Title_v=textBox4.Text ;
238
239 str2="INSERT INTO Employees(EmployeeID,LastName,FirstName,Title)";
240 s2="VALUES("+EmployeeID_v+","+"'"+LastName_v+"'" +","+"'"+FirstName_v+"'"+","+"'"+Title_v+"'"+")";
241 str2+=s2;
242     OleDbCommand myCommand = new OleDbCommand(str2,objConnection);
243     myCommand.ExecuteNonQuery ();
244
245
246
247         }
```

في هذه السطور:

- في السطر رقم 226 تم تعريف اربعة متغيرات من النوع string.
- في السطور 227و228و229و230 تم وضع قيم مربعات النصوص TextBox1,TextBox2,textBox3,TextBox4 وهي القيم التي كتبها المستخدم ليضيفها الى قاعدة البيانات في المتغيرات المعرفة.
- في السطر رقم 232 يتم كتابة جملة الـ SQL المطلوب تنفيذها وهي جملة Insert السابقة ولكن بتعديلها وجمع عليها قيم المتغيرات بدلاً من وضع قيم ثابتة.
- في السطر 234 يتم استعمال الدالة MsgBox() لعرض محتوى متغير جملة الـ SQL لمراجعتها ومعرفة نتيجة جمع المتغيرات على جملة الـ SQL.
- في السطر رقم 236 يتم استدعاء الدالة ExecuateNonQuery() لتنفيذ جملة الـ SQL

وإضافة السجل الجديد.

- نفذ البرامج عند هذه الخطوة واضغط الزر New ثم اكتب بيانات سجل جديد ثم اضغط بالماوس الزر Insert يتم اضافة السجل كما فى الشكل (5-13) .



الشكل(5- 13)

زر الحذف Delete

يؤدى الضغط على زر الأمر Delete إلى حذف بيانات الموظف الذى كتب كوده فى مربع النص الأول TextBox1 ويتم تحقيق ذلك بالأوامر التالية :

```
Private Sub Button6_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button6.Click
        conn.ConnectionString = ("PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA
SOURCE=c:\Northwind.mdb")
        cmd.CommandType = CommandType.Text
        Dim txt1 As String

        txt1 = "" + TextBox1.Text + ""

        Dim sql1 As String
        sql1 = "Delete from  Customers where CustomerID=" + txt1

        MsgBox(sql1)
```

```
    cmd.CommandText = sql1
    cmd.Connection = conn
    ' Create a SqlParameter for each parameter in the stored procedure.
    Dim previousConnectionState As ConnectionState = conn.State
    If Not previousConnectionState = ConnectionState.Closed Then
       conn.Close()
    End If
    conn.Open()
    cmd.ExecuteNonQuery()
    MsgBox("Record Deleted ..")

    conn.Close()
    TextBox1.Text = ""
    TextBox2.Text = ""
    TextBox3.Text = ""
    TextBox4.Text = ""
End Sub
```

فى لغة C#

```
private void button6_Click(object sender, System.EventArgs e)
{
    string strSQL;
    string EmployeeID_v;

    EmployeeID_v=textBox1.Text;
    strSQL="DELETE FROM  Employees WHERE EmployeeID="+EmployeeID_v;
    OleDbCommand myCommand = new OleDbCommand(strSQL,objConnection)
    myCommand.ExecuteNonQuery ();

    textBox1.Text="";
    textBox2.Text ="";
    textBox3.Text ="";
    textBox4.Text ="";

}
```

فى هذه السطور

يتم تغيير جملة الـ SQL من جملة Insert إلى جملة Delete وذلك كما فى السطر رقم 248 ثم تنفيذ هذه الجملة وبالتالى حذف السجل المكتوب رقمه فى مربع النص TextBox1.

زر تحديث البيانات Update

ويتيح لك هذا الزر إمكانية تعديل قيم بيانات موجودة وفيه يتم تغيير جملة الـ SQL إلى جملة Update كما فى السطور.

```
Private Sub Button7_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button7.Click
        conn.ConnectionString = ("PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA
SOURCE=c:\Northwind.mdb")
        cmd.CommandType = CommandType.Text
        Dim txt1, txt2, txt3, txt4, vals As String

        txt1 = "" + TextBox1.Text + ""
        txt2 = "" + TextBox2.Text + ""
        txt3 = "" + TextBox3.Text + ""
        txt4 = "" + TextBox4.Text + ""

        vals = txt1 + txt2 + txt3 + txt4

        Dim sql1 As String
        sql1 = "update Customers set  CompanyName=" + txt2 +
",ContactName=" + txt3 + ",ContactTitle=" + txt4 + " where CustomerID=" +
txt1
        MsgBox(sql1)
        cmd.CommandText = sql1
        cmd.Connection = conn
        ' Create a SqlParameter for each parameter in the stored procedure.
        Dim previousConnectionState As ConnectionState = conn.State
        If Not previousConnectionState = ConnectionState.Closed Then
            conn.Close()
        End If
        conn.Open()
        cmd.ExecuteNonQuery()
        MsgBox("Record Updated ..")

        conn.Close()

    End Sub
```

فى لغة C#

```
string strSQL,s2;
int EmployeeID_v;
string LastName_v;
string FirstName_v;
string Title_v;

EmployeeID_v=Int32.Parse (textBox1.Text) ;
LastName_v=textBox2.Text ;
FirstName_v=textBox3.Text ;
Title_v=textBox4.Text ;

strSQL="UPDATE  Employees SET LastName='"+LastName_v+"',";
s2="FirstName='"+FirstName_v+"',Title='"+Title_v+"' WHERE EmployeeID="+EmployeeID_v;
strSQL+=s2;
OleDbCommand myCommand = new OleDbCommand(strSQL,objConnection);
myCommand.ExecuteNonQuery ();
```

كما ذكرنا يمكن تنفيذ هذا المثال مع قاعدة بيانات من النوع SQl Server وذلك بتغيير قيمة الخاصية ConnectionString كما أشرنا فى الملحوظة السابقة.

زر الأمر Last

اكتب السطور التالية :

```
Private Sub Button8_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button8.Click
    While (reader.Read)
        TextBox1.Text = reader.GetValue(0)
        TextBox2.Text = reader.GetValue(1)
        TextBox3.Text = reader.GetValue(2)
        TextBox4.Text = reader.GetValue(3)
    End While
End Sub
```

استعمال أوامر الاتصال غير المباشر Off Line DataBase

يوفر VB.Net امكانية التعامل مع قاعدة البيانات بطريقة أوامر الاتصال غير المباشر Off Line DataBase وهذا يعنى عمل نسخة من البيانات فى ذاكرة الجهاز ثم القيام بالتعامل معها بالاضافة والحذف والتعديل فى انتظار أمر التحديث فيتم تحديث قاعدة البيانات الأصلية وهذا ما يسمى Off Line بحيث اننا لا نتعامل مع قاعدة البيانات مباشرة بل تعامل مع النسخة ، ويتم ذلك باستعمال فصيلة مشهورة فى VS.Net تسمى DataSet وسوف نتعرض لهذه النقطة بالتفصيل فى فصل تالى.

علاقة قواعد البيانات بلغة XML

لغة XML لغة حديثة تستخدم لتمثيل البيانات بما لها من ميزات عن لغة HTML ويوفر VS.Net تكامل كبير معها فتستطيع فتح بيانات فى الصورة xml ووضعها فى قاعدة البيانات وكذلك حفظ بيانات من قاعدة البيانات فى الصورة xml وسوف نتعرض لهذه النقطة بالتفصيل فيما بعد.

فى لغة Java

ربط قواعد البيانات والجافا Java Data Base Connectivity (JDBC)

فى لغة الجافا Java يتم التعامل مع قواعد البيانات من خلال مجموعة من الفصائل Classes والـ interfaces وهذة الفصائل تسمى JDBC وهى اختصار لـ Java Data Base JDBC Connectivity.

وهذه الفصائل تستخدم لغة SQL للقيام بالاستعلامات فى برامج قواعد البيانات المختلفة.

وتشتمل مجموعة فصائل JDBC على الفصائل التى تمكننا من القيام بالمهام الشائعة عند التعامل مع قواعد البيانات مثل:

1. القيام بالربط بين الجافا وقاعدة البيانات.
2. إنشاء الجمل باستخدام SQL.

3. تنفيذ تلك الجمل داخل قاعدة البيانات (الاستعلامات).
4. رؤية نتيجة تنفيذ الاستعلامات فى صورة سجلات.

وتوجد فصائل JDBC داخل الحزمة java.sql

وكما قلنا قبل ذلك فلكى نقوم بإنشاء تطبيقات قواعد البيانات مع لغات البرمجة فلابد من وجود واجهة Interface بين قاعدة البيانات ولغة البرمجة وتستخدم لغة الجافا الواجهة ODBC ولكن من خلال فصائل JDBC واصطلح على تسمية ذلك JDBC-ODBC BRIDGE .

الربط بين فصائل JDBC وواجهة قواعد البيانات ODBC

The JDBC-ODBC BRIDGE

ويقوم JDB-ODBC Bridge بالسماح لمشغلات JDBC drivers لكى تستخدم كأنها ODBC عن طريق تحويل استدعاء دوال JDBC إلى استدعاء وظائف ODBC.

ODBC

وODBC هى واجهة قد قامت ببنائها شركة مايكروسوفت لكى تسهل عملية بناء برامج تتصل ببرامج قواعد البيانات وفق معايير محددة وثابتة. وكل شركة منتجة لبرنامج من برامج قواعد البيانات تقوم بتوفير مشغل Driver ويتم دمجة فى واجهة ODBC حتى يستطيع المطورون استخدام برامج قواعد البيانات وفق معايير موحدة.

وتوجد واجهة ODBC فى لوحة التحكم Control Panel فى ويندوز 95.98 وموجودة المجلد ضمن Administrative Tools الموجودة فى لوحة التحكم ويندوز 2000 و ODBC بها مشغلات Drivers لقواعد البيانات الموجودة على جهازك فكل شركة من شركات قواعد البيانات تقوم بتوفير مشغل Driver خاص ببرنامج قاعدة بياناتها وذلك كما يتضح من الشكل (6-13).

الشكل (13-6)

استخدام JDBC-ODBC Bridge

لكى استخدم JDBC-ODBC Bridge يستلزم أن احدد ثلاثة أشياء :

1- مشغل JDBC-ODBC Bridge الموجود فى لغة جافا وهو : sun.jdbc.odbc.JdbcOdbcDriver

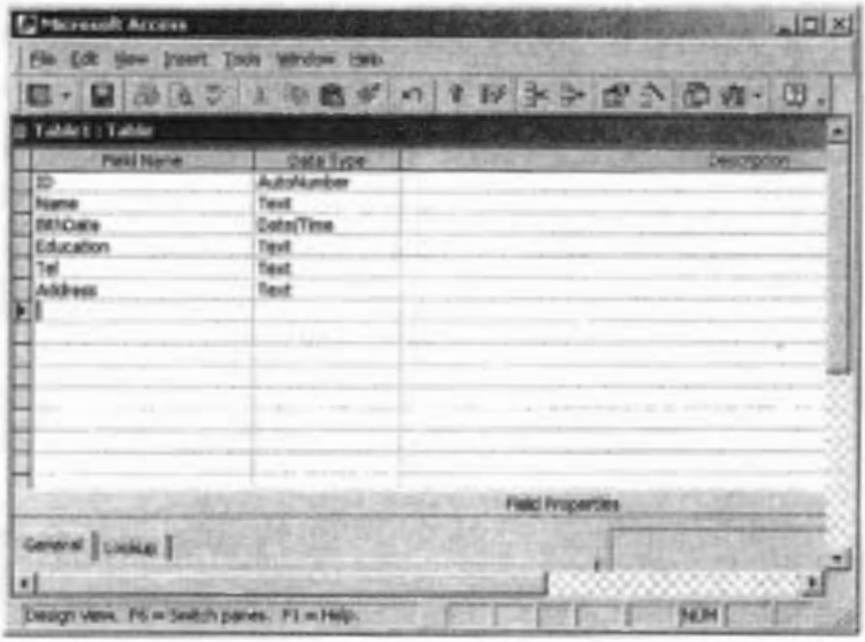
2- مشغل ODBC An ODBC Driver

3- قاعدة البيانات المرتبطة بالمشغل ODBC

وبعد أن أوضحنا بشيء من التفصيل تعال معى نرى من خلال مثال عملى كيفية تطبيق ذلك.

1- البرامج المطلوبة لتطبيق هذا المثال هو برنامج Access لإنشاء قاعدة البيانات. قم بتشغيل برنامج Access وقم بإنشاء قاعدة البيانات وأعطها اسم وليكن Javadb1 .

2- قم بإنشاء جدول Table جديد بالاسم Basicdata وقم بإنشاء أسماء الحقول Field Name وأنواعها DataTypes وذلك كما فى الشكل (7-13).

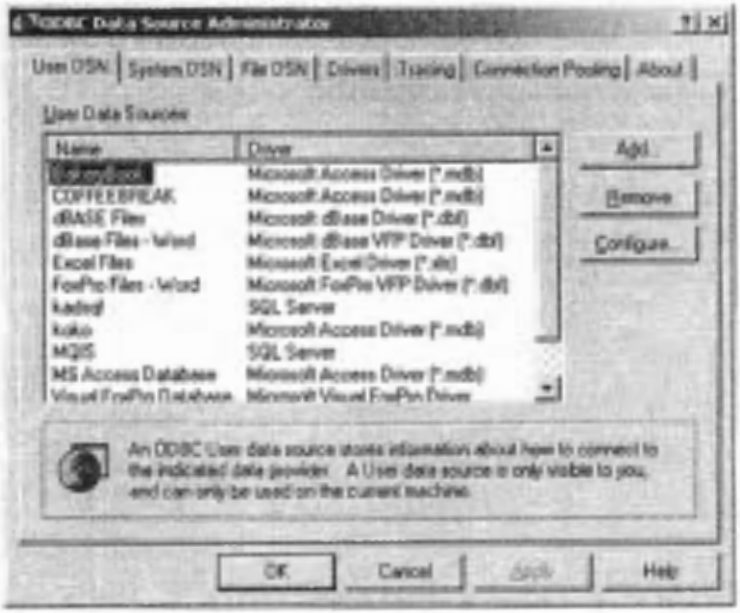


الشكل (13-7)

3- قم بفتح لوحة التحكم Control Panel واضغط مرتين على ODBC كما في الشكل (13-8).

الشكل (13-8)

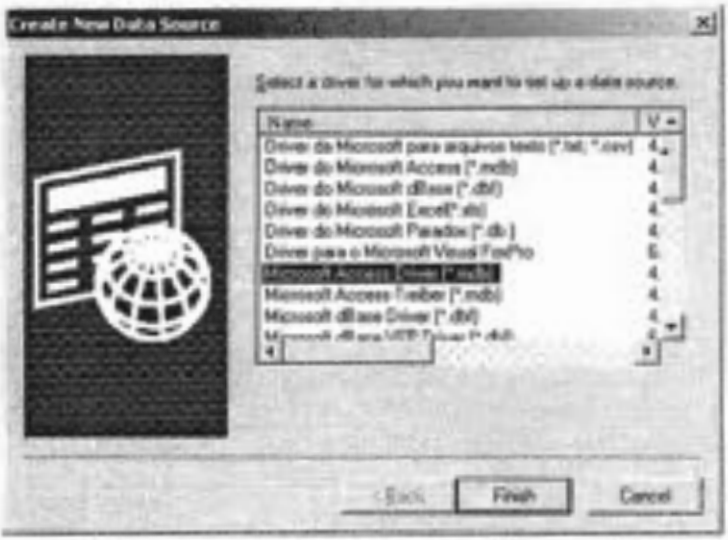
4- فيظهر الشكل (9-13).



الشكل (9-13)

5- اختار صفحة (تبويب) User DSN وهو أول تبويب على أقصى الشمال كما في الشكل.

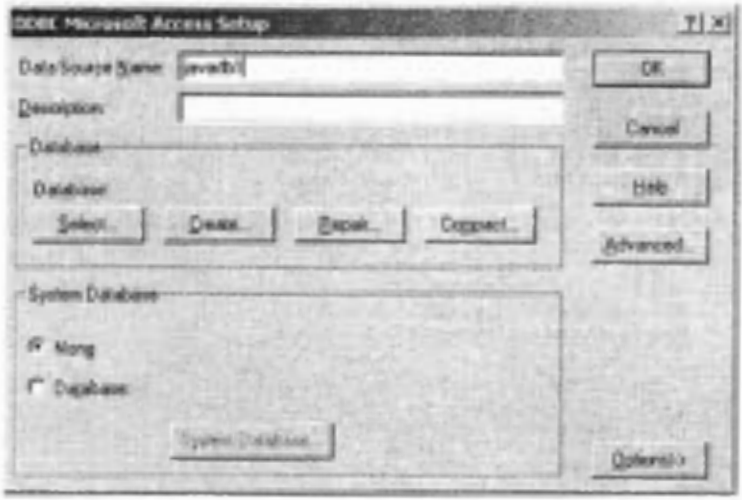
6- أنقر زر Add فيظهر الشكل (10-13).



الشكل (10-13)

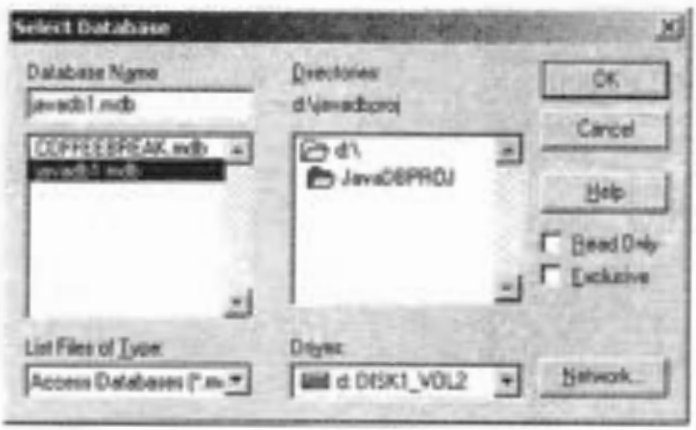
7- اختر Microsoft Access Driver (mdb) كما فى الشكل ثم اضغط Finish.

8- يظهر لك الشكل (11-13).



الشكل (11-13)

يطلب منك فى الحقل الأول أن تكتب اسم الوصلة وليكن Javadb1 وليس بالضرورة أن يكون نفس اسم قاعدة البيانات التى أنشأناها ولكننا هنا كتبنا نفس الاسم لكى للسهولة ثم اضغط على زر Select لكى نختار قاعدة البيانات التى أنشأناها فيظهر الشكل (12-13).



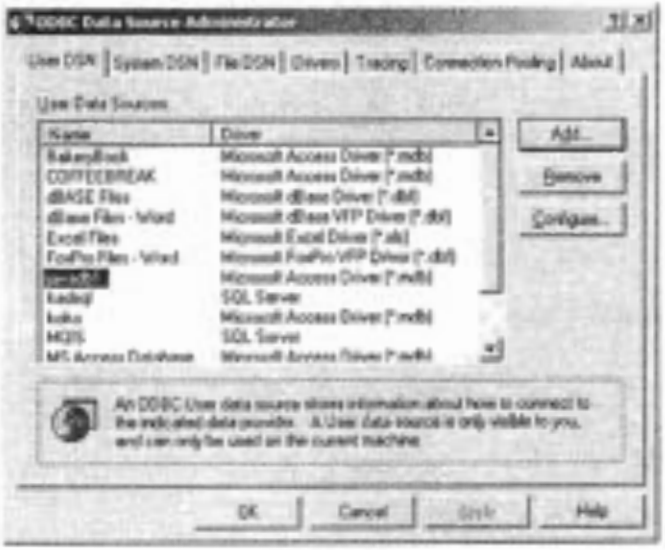
الشكل (21-13)

فقم بالذهاب إلى المسارpath الذى حفظت فيه قاعدة البيانات Javadb1 فتجد مثلا أننى قمت بحفظ قاعدة البيانات فى المسار التالى:

D:>JavaDBProJ \Javadb1.Mdb

9- بعد ذلك اضغط على الزر OK فيظهر مربع الحوار الأخير فاضغط على الزر OK ايضاً.

10- فتجد الشكل (13-13)



الشكل (13-13)

وقد ظهر به اسم Javadb1 الذى أنشأناه بعد ذلك اضغط على الزر OK.

لاحظ أن فى الخطوات من 3 إلى 7 قمنا بتنفيذ خطوتين هما:

1- تحديد ODBC Driver وفى هذا المثال اخترنا مشغل برنامج قاعدة البيانات Access.

2- تحديد قاعدة البيانات المرتبطة بالمشغل الذى اخترناه والتى هى Javadb1.

بذلك انتهى جزء إنشاء قاعدة البيانات ثم إضافة وصلة باسمها فى ODBC يقى لنا جزء كود الجافا java وهى أسهل من الجزء السابق فقط هناك ثلاثة خطوات رئيسية أقوم بها.

1- إنشاء متغير من نوع String وذلك لكى احتفظ فيه بعنوان الوصلة التى أنشأتها فى ODBC كالتالى:

```
String URL = " jdbc:odbc:Javadb1";
```

2-أقوم بتحميل المشغل الموجود فى الجافا لكى أقوم بالربط بقاعدة البيانات كالأتى:

```
Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
```

3- بعد تحميل المشغل نستطيع إنشاء وصلة Connection بقاعدة البيانات التى أنشأناها كالأتى:

```
Connection Con= DriverManager.getConnection, (URL, " ", " ")
```

حيث نجد أنها تأخذ ثلاثة معاملات.

1. عنوان وصلة قاعدة البيانات التى أنشأناها واعطيناه URL نظر لأن URL يحتوى على العنوان كما فى الخطوة 1.
2. المعامل الثانى يحتوى على اسم من مستخدم لقاعدة البيانات وحيث أننا لم نحدد ذلك تركناه فارغ " ".
3. المعامل الثالث كلمة السر لمستخدم قاعدة البيانات وأيضا تركناه فارغ " ".

بعد هذه الخطوات الثلاثة نستطيع أن نفعل ما نريد فى قاعدة البيانات فتعال معى نرى ذلك تفصيلياً كما فى نص البرنامج التالى:

```
1. import java.sql.*;
2. class Javadb1 {
3.  public static void main (String arg[ ] )
4. {
5.       String URL = " Jdbc: odbc:javadb1";
6.       try{
7.       Class.forName = (" sun.jdbc.odbc.JdbcOdbcDriver");
8. Connection Con= DriverManager.getConnection (URL, " ", " ");
9. Statement stmt = Con. CreatStatement();
10. String qry1 ="select * from Basicdata";
11. ResultSet  rcord= stmt.executeQuery (qry1);
12. While (rcord. next( ) ){
13. System.out.println(rcord.getString     ("ID")     +     ";"     +     rcord.
```

```
    getString("Name")+","+rcord.getString("Birthdata")+","                +
    rcord.getString("Education")+","  +  rcord.getString  ("Tel")+","  +
    rcord.getString ("Address"));
14. }
15. stmt.close();
16. }catch (Exception e){
17. System.out.println("Error " + e. to string());
18.    }
19. }
20. }
```

فى هذا المثال تم الآتى:

- فى السطر رقم 1 أوضحنا أننا نريد استخدام الحزمة java.sql لأنها تحتوى على الفصائل والـInterfaces مثل ....,RecordSet,Connection.
- فى السطر رقم 5 تم تخزين عنوان الوصلة Connection فى المتغير URL.
- فى السطر رقم 6 قمنا بإنشاء معالج استثناء Exception Handling لأن الدوال التى ستستخدمها تلقى استثناءات.
- فى السطر رقم 7 قمنا بتحميل Load المشغل Driver الموجود فى لغة جافا.
- فى السطر رقم 8 قمنا بإنشاء وصلة وذلك باستخدام Connection Interface وأعطيناها المعاملات (المتغير URL)الذى سبق وأنشأناه فى السطر 5 والذى يحتوى على عنوان الوصلة.
- قمنا بإنشاء هدف Instance من Statement Interface وذلك لفتح قناة مع قاعدة البيانات واستخدامها فى تنفيذ أوامر SQL وذلك من كما فى السطر 10.
- قمنا بكتابة أمر استعلام Query وذلك باستخدام الأمر ()stmt.executeQuery والذى يقوم بإرجاع سجلات فكان لابد من استخدام هدف Object (rcord)من نوع ResultSet لاستقبال هذه السجلات.
- فى السطر 12 أردت ان ادخل الهدف rcord فى Loop ولكن ما معنى ()rcord.next ؟
- عندما نقوم بإرسال استعلام إلى قاعدة البيانات يكون هناك مؤشر cursor يشير إلى ما قبل السجل الأول فتقوم الدالة next بإعلامه أن يتحرك إلى السجل الأول.

- بعد ذلك يقوم السطر 13 بعرض نتائج الاستعلام وذلك بأن يجعل الهدف rcord يقوم باستدعاء الدالة getString والتى تأخذ معامل واحد وهو أما اسم الحقل أو رقم الحقل فمثلا الحقل "ID" هو الحقل رقم 1 فبدلا من اكتب "ID" يمكن اكتب 1 وهكذا يتضح من السطر أننى أريد أن أعرض سجل مكون من حقول ID , Tel , Address Education, Birth Data, Name.
- فى السطر 16 أقوم بإغلاق قناة التعامل مع قاعدة البيانات أى بشكل آخر أوضح أننى انتهيت ولم أعد أريد تنفيذ أوامر أخرى مع قاعدة البيانات.
- فى السطر 17-19 أقوم بمعالجة الاستثناء لو حدث خطأ نتج عن عدم وجود قاعدة البيانات أو أن الربط لم يتم أو لأى استثناء قد يحدث.

هذا المثال يوضح كيفية عرض نتائج استعلام فى بيئة سطر الأوامر Command Prompt وسنتعرض لاحقا لمثال من خلال البيئة الرسومية GUI ولكن قبل ذلك يهمنا التعرض بشئ من التفصيل لـ ResultSet Interface.

الدوال المعرفة فى ResultSet Interface

يوجد العديد من الدوال المعرفة فى ResultSet Interface للتعامل مع البيانات الموجودة فى قاعدة البيانات فمثلا عند التعامل مع حقل يحوى التاريخ فى قاعدة البيانات فلابد من وجود دالة للتعامل مع مثل هذا النوع من البيانات واليك بعض هذة الدوال.

```
getDate(String)
getDouble(String)
getFloat(String)
getInt(String)
getLong(String)
getString(String)
```

ويتوقف استخدامك للدالة على نوع البيانات الموجودة فى حقول قاعدة البيانات.

كما أن ResultSet Interface يحتوى على الدوال التى تساعدك على التحرك Navigate فى قاعدة البيانات وهى.

afterLast() التحرك الى ما بعد السجل الأخير فى قاعدة البيانات.

beforeFirst() التحرك الى ما قبل السجل الأول فى قاعدة البيانات.

()first التحرك إلى السجل الأول فى قاعدة البيانات.

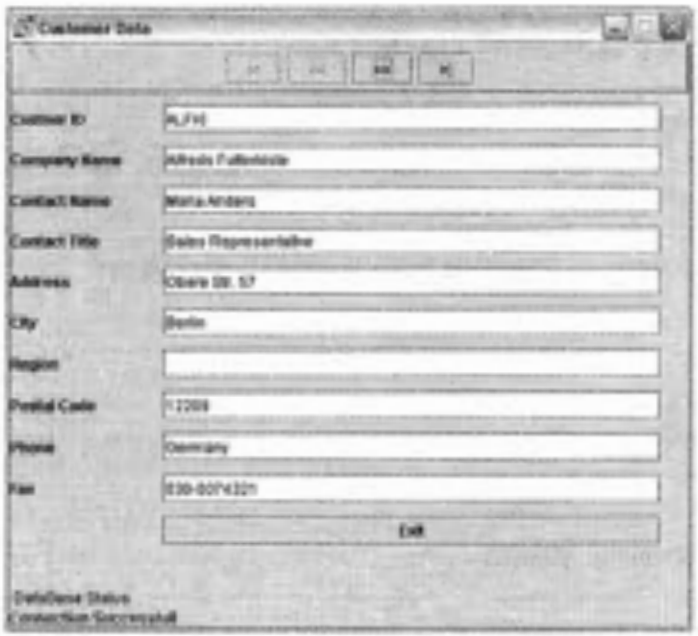
()Last التحرك إلى السجل الأخير فى قاعدة البيانات.

()previous التحرك إلى ما قبل السجل الحالى فى قاعدة البيانات.

()next التحرك إلى السجل التالى فى قاعدة البيانات.

والآن جاء موعدنا مع مثال يستخدم واجهة المستخدم الرسومية GUI وفى هذا المثال سوف نستخدم قاعدة البيانات NorthWind المدمجة فى برنامج قاعدة البيانات Access وستجدها على القرص المدمج مع الكتاب لذلك قم بعمل الخطوات من 3-10 السابق ذكرها لعمل وصلة فى ODBC وذلك بفتح لوحة التحكم ثم الوصول الى ايقونة ODBC ثم قم بعمل وصلة واعطها اسم وليكن NorthWind و اربطها مع قاعدة البيانات NorthWind التى غالبا ستجدها فى الدليل Folder الذى يحتوى ملفات Ms Office.

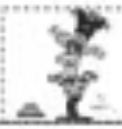
وسوف نتعامل مع جدول واحد من قاعدة البيانات NorthWind وهو جدول العملاء Custmers وهذا الجدول يحتوى على عشرة حقول تمثل بيانات العملاء والشكل التالى يبين البرنامج بعد بناء الواجهة الرسومية.



والشكل كما هو واضح مكون من 10 JTextField و12 JLabel و5 Jbutton و 4 Jpanel

واليك الكود.

يجب ترجمة وتنفيذ هذا المثال باستخدام JDK1.4



```
1. ●/
2. 2 ● TestDB.java
3. ●
4. Created on January 25, 2001, 6:17 PM
5. /●
6. ●● /
7. ●
8. /●
9. public class TestDB extends javax.swing.JFrame {
10. ●● /Creates new form TestDB /●
11. public TestDB (){
12. initComponents();
13. setupPosition();    / /Method to put Form in screen center
14. setupIcon();   / /Method to cahnge Form Icon
15. setSize(550,500); //to set Form Size
16. startDB();    / /to satrt DB upon Intializing
17. }
18. / /Implementing setupIcon
```

```
19. private void setupPosition()
20. {
21. java.awt.Dimension
      screen=java.awt.Toolkit.getDefaultToolkit().getScreenSize();
22. java.awt.Dimension frm=this.getPreferredSize();
23. if(frm.height>screen.height)
24. frm.height=screen.height;
25. if(frm.width>screen.width)
26. frm.width=screen.width;
27. setLocation((screen.width-550)/2,(screen.height-500)/2);
28. setTitle("Customer Data");
29. }
30. private void setupIcon()
31. {
32. java.awt.Image
      img=java.awt.Toolkit.getDefaultToolkit().getImage("e\\:icn.jpg");
33. setIconImage(img);
34. }
35. private String url="jdbc:odbc:NorthWind";
36. //Statement stmt=null;
37. java.sql.Statement stmt=null;
38. java.sql.Connection con=null;
39. java.sql.ResultSet rcord=null;
40. //Implementing of the DataBase
41. private void startDB()
42. {
43. try
44. {
45. Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
46. con=java.sql.DriverManager.getConnection(url,"kadry","18111970");
47. stmt=con.createStatement( java.sql.ResultSet.TYPE_SCROLL_INSENSITIVE,
48. java.sql.ResultSet.CONCUR_UPDATABLE);
49. rcord=stmt.executeQuery("SELECT * FROM Customers");
```

```
50. if(rcord.next())
51. {
52. if(rcord.isFirst())
53. {
54. jButton1.setEnabled(false);
55. jButton2.setEnabled(false);
56. }
57. jTextField1.setText(rcord.getString(10));
58. jTextField2.setText(rcord.getString(2));
59. jTextField3.setText(rcord.getString(3));
60. jTextField4.setText(rcord.getString(4));
61. jTextField5.setText(rcord.getString(5));
62. jTextField6.setText(rcord.getString(6));
63. jTextField7.setText(rcord.getString(7));
64. jTextField8.setText(rcord.getString(8));
65. jTextField9.setText(rcord.getString(9));
66. jTextField10.setText(rcord.getString(10));
67. }
68. jLabel6.setText("Connection Successful");
69. {catch(Exception e){
70. jLabel6.setText("Connection Not Successful"+e.toString());
71. }
72. }
73. /** This method is called from within the constructor to
74. initialize the form.
75. WARNING: Do NOT modify this code. The content of this method is
76. always regenerated by the Form Editor.
77. */
78. private void initComponents(){
79. jPanel1 = new javax.swing.JPanel();
80. jButton1 = new javax.swing.Jbutton();
81. jButton2 = new javax.swing.Jbutton();
82. jButton3 = new javax.swing.Jbutton();
83. jButton4 = new javax.swing.Jbutton();
```

```
84. jPanel3 = new javax.swing.Jpanel();
85. jLabel1 = new javax.swing.Jlabel();
86. jLabel2 = new javax.swing.Jlabel();
87. jLabel4 = new javax.swing.Jlabel();
88. jLabel3 = new javax.swing.Jlabel();
89. jLabel7 = new javax.swing.Jlabel();
90. jLabel71 = new javax.swing.Jlabel();
91. jLabel72 = new javax.swing.Jlabe();
92. jLabel73 = new javax.swing.Jlabel();
93. jLabel74 = new javax.swing.Jlabel();
94. jLabel75 = new javax.swing.Jlabel();
95. jPanel2 = new javax.swing.Jpanel();
96. jLabel5 = new javax.swing.Jlabel();
97. jLabel6 = new javax.swing.Jlabel();
98. jPanel4 = new javax.swing.Jpanel();
99. jTextField1 = new javax.swing.JtextField();
100.    jTextField2 = new javax.swing.JtextField();
101.    jTextField3 = new javax.swing.JtextField();
102.    jTextField4 = new javax.swing.JtextField();
103.    jTextField5 = new javax.swing.JtextField();
104.    jTextField6 = new javax.swing.JtextField();
105.    jTextField7 = new javax.swing.JtextField();
106.    jTextField8 = new javax.swing.JtextField();
107.    jTextField9 = new javax.swing.JtextField();
108.    jTextField10 = new javax.swing.JtextField();
109.    jButton5 = new javax.swing.Jbutton();
110.    setResizable(false);
111.    addWindowListener(new java.awt.event.WindowAdapter(){
112.    public void windowClosing(java.awt.event.WindowEvent evt(){
113.    exitForm(evt);
114.    }
115.    });
116.    jPanel1.setBorder(new
    javax.swing.border.BevelBorder(javax.swing.border.BevelBorder.RAISED));
117.    jPanel1.setPreferredSize(new java.awt.Dimension(40, 40));
118.    jButton1.setText;(">|")
119.    jButton1.addActionListener(new java.awt.event.ActionListener(){
120.    public void actionPerformed(java.awt.event.ActionEvent evt){
121.    jButton1ActionPerformed(evt);
122.    }
123.    });
124.    jPanel1.add(jButton1);
```

```
125.    jButton2.setText(">>");

126.    jButton2.addActionListener(new java.awt.event.ActionListener (){
127.    public void actionPerformed(java.awt.event.ActionEvent evt ){
128.    jButton2ActionPerformed(evt);
129.    }
130.    });
131.    jPanel1.add(jButton2);
132.    jButton3.setText("<<");

133.    jButton3.addActionListener(new java.awt.event.ActionListene (){
134.    public void actionPerformed(java.awt.event.ActionEvent evt(){
135.    jButton3ActionPerformed(ev);
136.    }
137.    });
138.    jPanel1.add(jButton3);
139.    jButton4.setText("|<");

140.    jButton4.addActionListener(new java.awt.event.ActionListene(){
141.    public void actionPerformed(java.awt.event.ActionEvent evt} (

142.    jButton4ActionPerformed(evt);
143.    }
144.    });
145.    jPanel1.add(jButton4);
146.    getContentPane().add(jPanel1, java.awt.BorderLayout.NORTH);
147.    jPanel3.setLayout(new
    java.awt.FlowLayout(java.awt.FlowLayout.CENTER, 5, 10));

148.    jPanel3.setPreferredSize(new java.awt.Dimension(100, 100));
149.    jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.LEF);
150.    jLabel1.setText("Custner ID");
151.    jLabel1.setPreferredSize(new java.awt.Dimension(100, 23));
152.    jPanel3.add(jLabel1);
153.    jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
154.    jLabel2.setText("Company Name");
155.    jLabel2.setPreferredSize(new java.awt.Dimension(100, 23));
156.    jPanel3.add(jLabel2);
157.    jLabel4.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
158.    jLabel4.setText("Contact Name");
159.    jLabel4.setPreferredSize(new java.awt.Dimension(100, 23));
160.    jPanel3.add(jLabel4);
161.    jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
```

```
jLabel3.setText("Contact Title");
jLabel3.setPreferredSize(new java.awt.Dimension(100, 23));
jPanel3.add(jLabel3);
jLabel7.setText("Address");
jLabel7.setPreferredSize(new java.awt.Dimension(100, 23));
jPanel3.add(jLabel7);
jLabel71.setText("City");
jLabel71.setPreferredSize(new java.awt.Dimension(100, 23));
jPanel3.add(jLabel71);
jLabel72.setText("Region");
jLabel72.setPreferredSize(new java.awt.Dimension(100, 23));
jPanel3.add(jLabel72);
jLabel73.setText("Postal Code");
jLabel73.setPreferredSize(new java.awt.Dimension(100, 23));
jPanel3.add(jLabel73);
jLabel74.setText("Phone");
jLabel74.setPreferredSize(new java.awt.Dimension(100, 23));
jPanel3.add(jLabel74);
jLabel75.setText("Fax");
jLabel75.setPreferredSize(new java.awt.Dimension(100, 23));
jPanel3.add(jLabel75);
getContentPane().add(jPanel3, java.awt.BorderLayout.WEST);
jPanel2.setLayout(new java.awt.BorderLayout());
jLabel5.setFont(new java.awt.Font("Dialog", 0, 12));
jLabel5.setText(" DataBase Status");
jPanel2.add(jLabel5, java.awt.BorderLayout.CENTER);
jLabel6.setForeground(new java.awt.Color(255, 0, 0));
jLabel6.setText("jLabel6");
jPanel2.add(jLabel6, java.awt.BorderLayout.SOUTH);
getContentPane().add(jPanel2, java.awt.BorderLayout.SOUTH);
jPanel4.setLayout(new
    java.awt.FlowLayout(java.awt.FlowLayout.CENTER, 5, 10));
jTextField1.setPreferredSize(new java.awt.Dimension(400, 23));
jTextField1.addActionListener(new java.awt.event.ActionListener (){

public void actionPerformed(java.awt.event.ActionEvent evt){
jTextField1ActionPerformed(evt);
}
});
jPanel4.add(jTextField1);
jTextField2.setPreferredSize(new java.awt.Dimension(400, 23));
jPanel4.add(jTextField2);
jTextField3.setPreferredSize(new java.awt.Dimension(400, 23));
```

```
203.    jPanel4.add(jTextField3);
204.    jTextField4.setPreferredSize(new java.awt.Dimension(400, 23));
205.    jPanel4.add(jTextField4);
206.    jTextField5.setPreferredSize(new java.awt.Dimension(400, 23));
207.    jPanel4.add(jTextField5);
208.    jTextField6.setPreferredSize(new java.awt.Dimension(400, 23));
209.    jPanel4.add(jTextField6);
210.    jTextField7.setPreferredSize(new java.awt.Dimension(400, 23));
211.    jPanel4.add(jTextField7);
212.    jTextField8.setPreferredSize(new java.awt.Dimension(400, 23));
213.    jPanel4.add(jTextField8);
214.    jTextField9.setPreferredSize(new java.awt.Dimension(400, 23));
215.    jPanel4.add(jTextField9);
216.    jTextField10.setPreferredSize(new java.awt.Dimension(400, 23));
217.    jPanel4.add(jTextField10);
218.    jButton5.setText("Exit");
219.    jButton5.setPreferredSize(new java.awt.Dimension(400, 23));
220.    jButton5.addActionListener(new java.awt.event.ActionListener (){
221.    public void actionPerformed(java.awt.event.ActionEvent evt){
222.    jButton5ActionPerformed(evt);
223.    }
224.    });
225.    jPanel4.add(jButton5);
226.    getContentPane().add(jPanel4, java.awt.BorderLayout.CENTER);
227.    pack();
228.    }
229.    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt){
230.     // //Add your handling code here:
231.    System.exit(0);
232.    {
233.    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt){
234.     // //Add your handling code here:
235.    try{
236.    if(rcord.last())
237.    {
238.    jButton1.setEnabled(true);
239.    jButton4.setEnabled(false);
240.    if(rcord.isLast())
241.    {
242.    jButton4.setEnabled(false);
```

```
243.    jButton3.setEnabled(false);
244.    jButton2.setEnabled(true);
245.    jButton1.setEnabled(true);
246.    }
247.    jTextField1.setText(rcord.getString(1));
248.    jTextField2.setText(rcord.getString(2));
249.    jTextField3.setText(rcord.getString(3));
250.    jTextField4.setText(rcord.getString(4));
251.    jTextField5.setText(rcord.getString(5));
252.    jTextField6.setText(rcord.getString(6));
253.    jTextField7.setText(rcord.getString(7));
254.    jTextField8.setText(rcord.getString(8));
255.    jTextField9.setText(rcord.getString(9));
256.    jTextField10.setText(rcord.getString(10));
257.    {
258.    //stmt.close;()
259.    {catch(Exception e){jLabel6.setText("Failed to move Last
    "+e.toString());
260.    }
261.    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt){
262.    //Add your handling code here:
263.    try{
264.    if(rcord.previous())
265.    {
266.    jButton1.setEnabled(true);
267.    jButton4.setEnabled(true);
268.    if(rcord.isFirst())
269.    {
270.    jButton1.setEnabled(false);
271.    jButton2.setEnabled(false);
272.    }
273.    jTextField1.setText(rcord.getString(1));
274.    jTextField2.setText(rcord.getString(2));
275.    jTextField3.setText(rcord.getString(3));
276.    jTextField4.setText(rcord.getString(4));
277.    jTextField5.setText(rcord.getString(5));
278.    jTextField6.setText(rcord.getString(6));
279.    jTextField7.setText(rcord.getString(7));
280.    jTextField8.setText(rcord.getString(8));
281.    jTextField9.setText(rcord.getString(9));
282.    jTextField10.setText(rcord.getString(10));
```

```
283.    {
284.     //stmt.close;()
285.    {catch(Exception e){(Label6.setText("Failed to move Previous
    "+e.toString());
286.    }
287.    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt){
288.     //Add your handling code here:
289.    try{
290.    if(rcord.next())
291.    {
292.    jButton1.setEnabled(true);
293.    jButton4.setEnabled(true);
294.    if(rcord.isLast())
295.    {
296.    jButton4.setEnabled(false);
297.    jButton3.setEnabled(false);
298.    jButton1.setEnabled(true);
299.    jButton2.setEnabled(true);
300.    {
301.    jTextField1.setText(rcord.getString(1));
302.    jTextField2.setText(rcord.getString(2));
303.    jTextField3.setText(rcord.getString(3));
304.    jTextField4.setText(rcord.getString(4));
305.    jTextField5.setText(rcord.getString(5));
306.    jTextField6.setText(rcord.getString(6));
307.    jTextField7.setText(rcord.getString(7));
308.    jTextField8.setText(rcord.getString(8));
309.    jTextField9.setText(rcord.getString(9));
310.    jTextField10.setText(rcord.getString(10));
311.    jButton2.setEnabled(true);
312.     //jTexField5.setText(rcord.getString(4));
313.    {
314.     //stmt.close;()
315.    }catch(Exception e){jLabel6.setText("Failed to move Next
    "+e.toString());
316.    }
317.    private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt)
```

```
318.      / /Add your handling code here:
319.     {
320.     private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
321.      / /Add your handling code here:
322.     try{
323.     if(rcord.first())
324.     {
325.     if(rcord.isFirst())
326.     {
327.     jButton1.setEnabled(false);
328.     jButton2.setEnabled(false);
329.     jButton3.setEnabled(true);
330.     jButton4.setEnabled(true);
331.     }
332.     jTextField1.setText(rcord.getString(1));
333.     jTextField2.setText(rcord.getString(2));
334.     jTextField3.setText(rcord.getString(3));
335.     jTextField4.setText(rcord.getString(4));
336.     jTextField5.setText(rcord.getString(5));
337.     jTextField6.setText(rcord.getString(6));
338.     jTextField7.setText(rcord.getString(7));
339.     jTextField8.setText(rcord.getString(8));
340.     jTextField9.setText(rcord.getString(9));
341.     jTextField10.setText(rcord.getString(10));
342.     {
343.      / /stmt.close;()
344.     }catch(Exception e){jTextField1.setText("Failed to move First
    "+e.toString());}
345.     }
346.      ** /Exit the Application /*
347.     private void exitForm(java.awt.event.WindowEvent evt){
348.     System.exit(0);
349.     {
350.     ** /
351.     @param args the command line arguments
```

```
352.    /**
353.    public static void main(String arg[]){
354.    new TestDB().show();
355.    }
356.     // /Variables declaration - do not modify
357.    private javax.swing.JPanel jPanel4;
358.    private javax.swing.JButton jButton5;
359.    private javax.swing.JPanel jPanel3;
360.    private javax.swing.JPanel jPanel2;
361.    private javax.swing.JButton jButton4;
362.    private javax.swing.JButton jButton3;
363.    private javax.swing.JPanel jPanel1;
364.    private javax.swing.JButton jButton2;
365.    private javax.swing.JButton jButton1;
366.    private javax.swing.JTextField jTextField10;
367.    private javax.swing.JTextField jTextField9;
368.    private javax.swing.JTextField jTextField8;
369.    private javax.swing.JTextField jTextField7;
370.    private javax.swing.JTextField jTextField6;
371.    private javax.swing.JTextField jTextField5;
372.    private javax.swing.JTextField jTextField4;
373.    private javax.swing.JTextField jTextField3;
374.    private javax.swing.JTextField jTextField2;
375.    private javax.swing.JTextField jTextField1;
376.    private javax.swing.JLabel jLabel7;
377.    private javax.swing.JLabel jLabel6;
378.    private javax.swing.JLabel jLabel5;
379.    private javax.swing.JLabel jLabel4;
380.    private javax.swing.JLabel jLabel3;
381.    private javax.swing.JLabel jLabel75;
382.    private javax.swing.JLabel jLabel2;
383.    private javax.swing.JLabel jLabel74;
384.    private javax.swing.JLabel jLabel1;
385.    private javax.swing.JLabel jLabel73;
386.    private javax.swing.JLabel jLabel72;
387.    private javax.swing.JLabel jLabel71;
388.     // /End of variables declaration
389.    }
```

- هذا الكود به الكثير من الدوال والفصائل التي قد تعرضنا لها خلال الفصول السابقة ولكن مايهمنا هنا هو الكود المظلل حيث أنه يحتوى على الكود الخاص بانشاء والتعامل مع قاعدة البيانات وهذا ماسنشرحة الآن.
- فى السطور من 39-43 قمنا بإنشاء متغيرات للتعامل مع قاعدة البيانات وهذه المتغيرات هى.
- URL وهو متغير من نوع String لتحفظ فية عنوان الوصلة.
- Stmt وهو هدف من نوع Statement.
- con وهو هدف من نوع Connection.
- rcord وهو هدف من نوع ResultSet.
- وقد اعلنا عن هذة المتغيرات والاهداف خارج نطاق اى دالة حتى نستطيع استخدامهم من خلال اى دالة او منشأ Block فى الفصيلة.
- فى السطور من 45-79 نبنى الدالة ()startDB وقد استدعيناها فى دالة البناء الـ Constructor فى السطر 18 حتى يتم تنفيذها حال انشاء الهدف من الفصيلة TestDB.
- فى هذة الدالة(()startDB) قمنا بشحن الاهداف السابق اعلانها فى السطور من 39-43 فقمنا بتحميل المشغل كما فى السطر 49.
- فى السطر 50 شحن الهدف con بعنوان وصلة قاعدة البيانات url,username,password.
- فى السطر 51-52 شحن الهدف stmt ولكن مايهمنا هذة المرة هو وجود المعاملين.

ResultSet.TYPE_SCROLL_INSENSITIVE,

ResultSet.CONCUR_UPDATABLE

فهذان المعاملان هما اللذان يمنحانا القدرة على التحرك فى قاعدة البيانات Navigating أى أننا بدونهما نستطيع التحرك فى السجلات إلى الأمام فقط.

حاول حذفهما ثم اعد ترجمة البرنامج وتنفيذة وحاول التحرك للخلف او اول او اخر سجل فى قاعدة البيانات وانظر ماذا ترى!

- فى السطر 54 نقوم بتنفيذ استعلام فى الجدول Customers وسوف يحتفظ الهدف rcord بنتيجة هذا الاستعلام.

- فى السطر 55 أقوم بفحص ماذا كان مؤشر قاعدة البيانات يشير إلى السجل الأول.
- فى السطور من 57 -61 اقوم باختبار ما إذا كان المؤشر يشير فعلا للسجل الأول فإذا كان فعلا يشير إليه فيقوم بجعل زرى التحرك (أول سجل >| و سجل سابق للحالى <<)غير فعالين.
- فى السطور من 62-71 نقوم بسحب pull البيانات من قاعدة البيانات وعرضهم من خلال أدوات الكتابة jTextField .
- فى السطر 73 نقوم بعرض رسالة تفيد نجاح الاتصال بقاعدة البيانات من خلال أداة العنوان jLabel6 الموجودة أسفل الفورم وتعرض الرسائل باللون الأحمر.
- فى السطور 75-78 نقوم بمعالجة أى استثناء قد يحدث ونقوم بجعل jLabel6 يعرض رسالة تحمل الاستثناء الحادث.

لاحظ :

اننا جعلنا jLabel6 يعمل كشريط حالة Status Bar

- فى السطر 79 قوس إنهاء الدالة.
- وبذلك عند انشاء هدف من الفصيلة كما فى السطر413 يتم استدعاء الدالة startDB() .
- فى السطور من 284-311 كود زر التحرك إلى أخر سجل وفى السطور 293-297 نقوم فى حالة إذا كان المؤشر يشير إلى السجل الأخير بجعل زرى التحرك (لأخر سجل |< والسجل التالى <<)غير فعالان.
- فى السطور من 313-383 كود زر التحرك إلى السجل السابق <<
- فى السطور 320-324 نقوم فى حالة اذا كان المؤشر يشير الى السجل الاول بجعل زرى التحرك(لاول سجل >| والسجل السابق <<)غير فعالان.
- فى السطور من 340-370 كود زر التحرك إلى السجل التالى << وفى السطور 353-374 نقوم فى حالة إذا كان المؤشر يشير إلى السجل الاخير بجعل زرى التحرك (لأخر سجل |< والسجل التالى <<)غير فعالان.
- فى السطور من 376-402 كود زر التحرك إلى السجل الأول >|

- فى السطور 387-381 نقوم فى حالة إذا كان المؤشر يشير إلى السجل الأول بجعل زرى التحرك(الأول سجل <| والسجل السابق <<)غير فعالان.
- لاحظ أننا وضعنا الكود الخاص بكل زر بين try-catch لمعالجة أى استثناء قد يحدث وإذا حدث استثناء يتم عرضة من خلال إداة العنوان lLabel6.

وبهذا نكون قد انتهينا من فصل قواعد البيانات وقد تعرضنا لقدر بسيط فى التعامل مع قواعد البيانات ولنا جولات أخرى فى التعامل مع قواعد البيانات فى الجزء الثانى من الكتاب.

فى Oracle

يتم استعمال أوامر SQL التى شرحناها مباشرة لتحقيق عمليات قواعد البيانات ولا تحتاج إلى المقدمات والأوامر التى استعملناها فى اللغات الأخرى.
لذلك هى من أسهل اللغات لإعداد تطبيقات قواعد البيانات.