

2011

# Parameters

Door middel van parameters een database inlezen/aanpassen.

Door middel van parameters in VB.NET data in een MS Access database inserten, updaten en deleten alsook databindings leggen.



## Inhoud

Layout .....	3
Data inlezen .....	4
Tijdelijk overzicht: .....	7
Databindings .....	8
Tekstvakken ledigen .....	9
Insert .....	10
Update .....	12
Delete .....	13
Source code .....	14
Slot .....	17

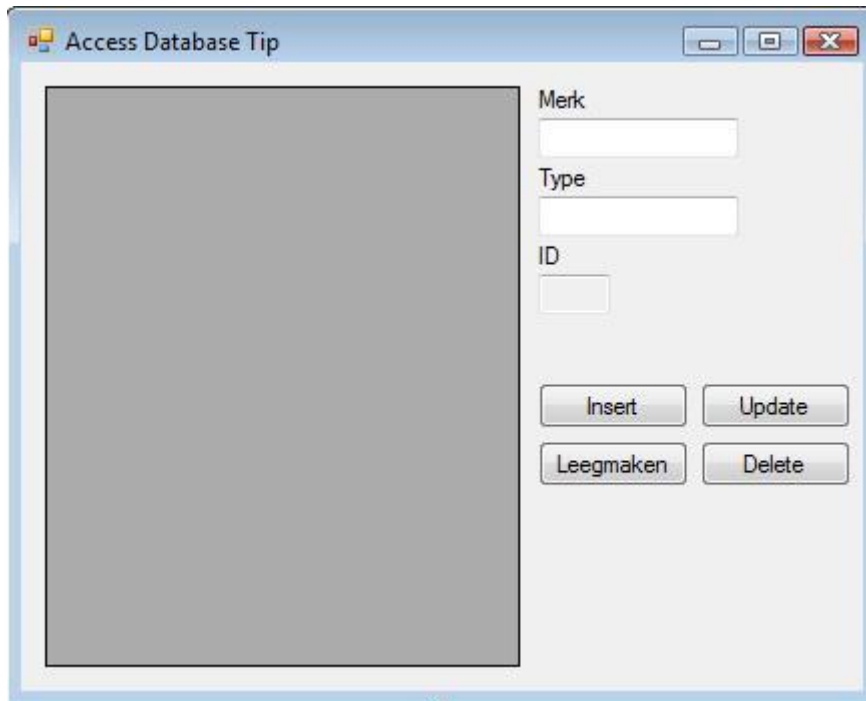
Hypernate

## Layout

We maken een nieuw 'Windows Form' project aan.

Plaats hierop volgende componenten:

- 1 DataGridView: dgvAutos
- 2 Labels: lblMerk, lblType, lblId
- 3 Textboxen: txtMerk, txtType, txtId(ReadOnly)
- 4 Buttons: btnLeegmaken, btnInsert, btnUpdate, btnDelete



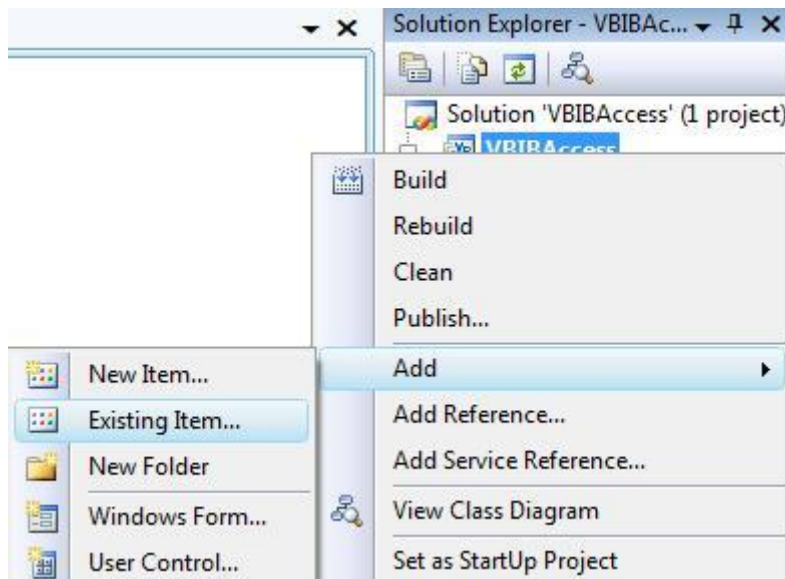
The screenshot shows a Windows Form titled "Access Database Tip". The form contains a large gray rectangular area on the left, which is a DataGridView. To the right of this area, there are three labels: "Merk", "Type", and "ID". Each label is followed by a text box. Below the text boxes, there are four buttons: "Insert", "Update", "Leegmaken", and "Delete". The form has a standard Windows window border with minimize, maximize, and close buttons in the top right corner.

## Data inlezen

We gaan nu de database toevoegen aan ons project.

De database die ik gebruik kan je [hier](#) downloaden: (Unpakken met winRAR of iets dergelijks) Eerste en vooral is het sterk aangeraden om de database in je Bin\Debug map te plaatsen in mijn geval is dat: "C:\Users\pC\Documents\Visual Studio 2008\Projects\VBIBAccess\VBIBAccess\bin\Debug".

Dan rechtermuisknop op je projectnaam > Add > Existing Item in de 'Solution Explorer'.

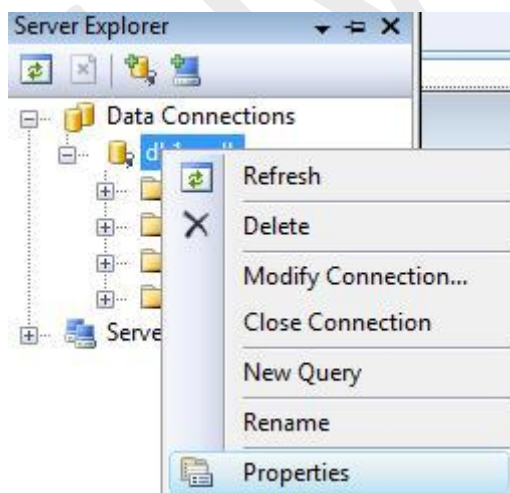


En dan zoek je je database die in je Bin/Debug staat. Wel even zien dat hij "All Files" of "Database Files" zoekt.

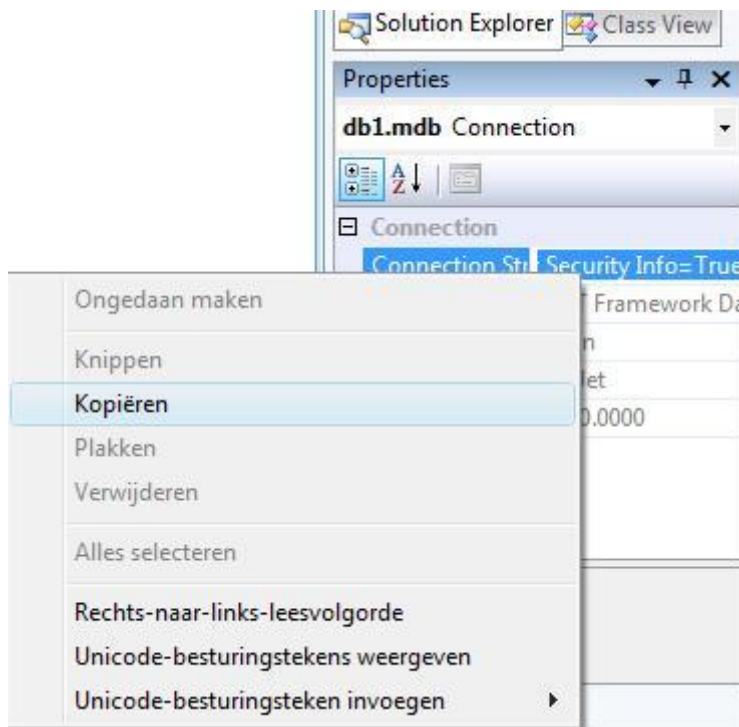
Dan vink je "Tables" aan en klik je op Finish. Hij maakt vanzelf een dataSet aan...Deze kan verwijderd worden.

Om de connectieString op te vragen doe je het volgende:

In de 'Server Explorer', rechtermuisknop op je database > Properties.



En dan rechtsonder in het vak 'Properties' dubbelklik je op de eigenlijke connectiestring zodat hij helemaal geselecteerd is en dan rechtermuisknop > Kopiëren.



Zo, Nu ga je de code kijken en plaats je deze regel onder Public Class Form1:  
(Je moet wel even de quotes herschikken en het laatste stukje wegdoen.)

Code:<sup>1</sup>

```
dim connectieString as string =  
Provider=Microsoft.Jet.OLEDB.4.0;Data  
Source=C:\Users\pC\Documents\Visual Studio  
2008\Projects\VBIBAccess\VBIBAccess\db1.mdb"
```

We gaan hier ook in één keer onze datatable<sup>2</sup> maken.

Code:

```
dim dtAutos as new DataTable
```

Ook de OleDbConnection gaan we hier aanmaken

Code:

```
dim connetie as new OleDb.OleDbConnection(connectieString)
```

Bij de 'connectie' heb ik de connectieString meegegeven die we twee regels hoger hadden gedeclareerd

<sup>1</sup> Je kan alle connectiestrings opvragen via [deze](#) website.

<sup>2</sup> Meer info over datatables vind je [hier](#).

We maken een nieuwe Sub procedure "dataLezen" en voegen onderstaande code toe.

Ik geef de SQL-opdracht mee (alles selecteren van tabel tblCars) dat moet uitgevoerd worden en de connectie.

De dataAdapter is eigenlijk een tussenstuk van je database en je programma. Je kan het zien als, indien je een iet wat abstract denkvermogen hebt, de stroomkabel tussen je PC en het stroomnet.

---

**Code:**

```
dim daAutos as new OleDb.OleDbDataAdapter("SELECT * FROM  
tblCars", connectie)
```

Nu zou de dataAdapter onze gegevens moeten bezitten.

Om deze door te geven naar onze dataTable gebruiken we de .Fill optie.

Voeg onderstaande code toe aan de sub "DataLezen".

De .clear gaat ervoor zorgen dat als we inlezen, inserten/updaten/verwijderen en terug inlezen, dat we de database geen 2 keer gaan laden. (Als je juist wil weten wat ik bedoel zet je deze zin even op commentaar voor je gaat testen bij de insert).

---

**Code:**

```
dtAutos.Clear()  
daAutos.Fill(dtAutos)
```

Nu bezit onze dataTable de gegevens.

Ga nu naar 'Designer Mode' en dubbel klik op het form, zodat je in de Visual Studio voor ons de 'Load' sub aanmaakt.

Hierin plaatsen we onze sub 'DataLezen()' en koppelen we de dataSource van onze DataGridView.

---

**Code:**

```
dataLezen()  
Me.dgvAutos.DataSource = dtAutos
```

Als we ons project opstarten, zie je dat onze DataGridView gevuld is met gegevens.

## Tijdelijk overzicht:

---

Code:

```
public class Form1

    dim connectieString as string =
    "Provider=Microsoft.Jet.OLEDB.4.0;Data
    Source=C:\Users\pC\Documents\Visual Studio
    2008\Projects\VBIBAccess\VBIBAccess\db1.mdb"
    dim dtAutos as new DataTable
    dim connetie as new OleDb.OleDbConnection(connectieString)
public sub dataLezen()
    dim daAutos as new OleDb.OleDbDataAdapter("SELECT * FROM
    tblCars", connetie)
    dtAutos.Clear()
    daAutos.Fill(dtAutos)
end sub
private sub Form1_Load(byval sender as System.object, byval e
as System.EventArgs) handles MyBase.Load
    dataLezen()
    Me.dgvAutos.DataSource = dtAutos
end sub

end class
```

## Datbindings

Datbindings zorgen ervoor dat in de textvakken, de waardes komen die er geselecteerd worden in een combobox of iets dergelijks. In ons geval de DataGridView.

In onze "Load" sub maken we de DataBindings aan:

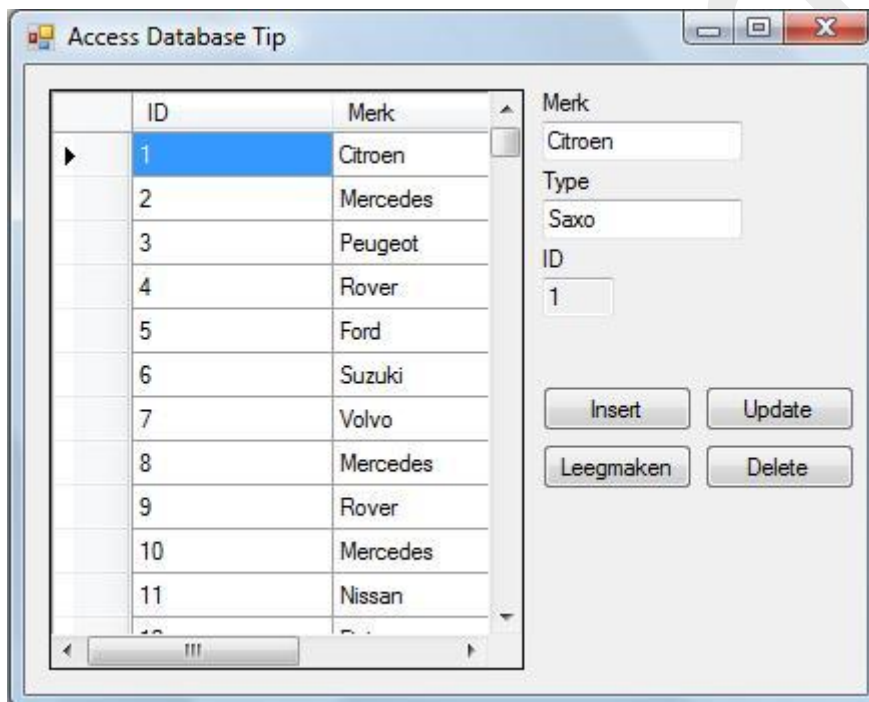
Code:

```
Me.txtId.DataBindings.Add("Text", dtAutos, "ID", false,
DataSourceUpdateMode.Never)
Me.txtMerk.DataBindings.Add("Text", dtAutos, "Merk", false,
DataSourceUpdateMode.Never)
Me.txtType.DataBindings.Add("Text", dtAutos, "Model", false,
DataSourceUpdateMode.Never)
```

De uitleg voor de 1<sup>e</sup> coderegel, de rest is papegaaienwerk: Het type is Text vanuit de dataTable dtAutos, en de kolom die je eruit wil is "ID" (de zelfde naam als in de database dus).

Als je je project nu start, kan je zien dat als je wat selecteert uit je database de TextBox'en de overeenkomstige waardes bevatten!

Tot zover het Inlezen van gegevens en de datbindings hiervan.



## Tekstvakken ledigen

Gemakshalve gaan we terug 'Designer Mode' en dubbelklikken we op de Button "Leegmaken".  
In de click sub plaatsen we:

Code:

```
for each ctrl as Control In me.Controls
  if TypeOf ctrl is TextBox then
    ctrl.Text = string.empty
  end if
next
```

Dit geheel terzijde, maar maakt het net iets makkelijker :).

Hypenate

## Insert

Om een insert te doen met parameters, is het niet nodig om een nieuwe subprocedure te maken. Ga naar 'Designer Mode' en dubbelklik op de Button "Insert" en voeg onderstaande code toe.

Code:

```
daAutos.InsertCommand.Connection = connective
daAutos.InsertCommand = New OleDb.OleDbCommand("INSERT INTO
tblCars (Merk, Model) VALUES (@Merk, @Model)")

daAutos.InsertCommand.Parameters.Add("@Merk",
OleDb.OleDbType.VarChar).Value = Me.txtMerk.Text
daAutos.InsertCommand.Parameters.Add("@Model",
OleDb.OleDbType.VarChar).Value = Me.txtType.Text

connectie.Open()
daAutos.InsertCommand.ExecuteNonQuery()
connectie.Close()

dataLezen()
```

Zoals je ziet geef ik de 2 kolommen mee waar hij bepaalde waardes (values) moet insteken.

Zoals je kan staan er voor die Values een @-teken, dit wordt gedaan als mijn gebruik gaat maken van zogenaamde parameters.

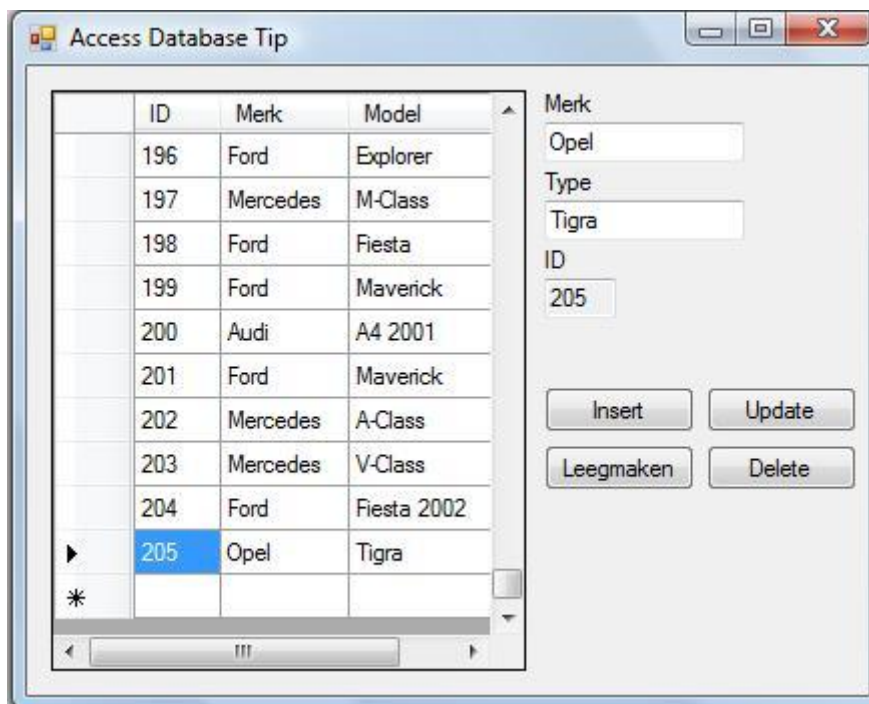
Zoals je ziet vul ik de parameters in, en "vul" ik de placeholders (@Merk en @Model) in die we hadden geplaatst bij onze InsertCommand met respectievelijk txtMerk.Text en txtType.Text. Ik geef tevens ook het "type" van @Merk en @Model mee, hiermee bedoel ik of het String (VarChar) of Integer (Numeriek) is. Deze moet overeenkomen met de veld-types hebt gekozen in de MS Access database.

Daarna wordt de connectie geopend, ik voer een ExecuteNonQuery() uit, dit is het moment waarop hij de eigenlijke opdracht gaat uitvoeren en de waardes in onze database gaat plaatsen.

Achteraf sluit ik de connectie terug en ga ik de gegevens opnieuw inlezen zodat onze dataGridView direct wordt geüpdatet.

Run nu het programma, klik op de Button "Leegmaken", en voeg dan de de vakken "Merk en Type" in. Klik dan op de Button "Insert".

Als je nu terug naar beneden scrolt zal je zien dat je merk en type toegevoegd zijn.



## Update

Ga naar 'Designer Mode' en dubbelklik op de Button "Update".

Bij de 'Click' sub voegen we een gelijkaardige code toe als bij Insert.

Code:

```
daAutos.UpdateCommand.Connection = connective
daAutos.UpdateCommand = new OleDb.OleDbCommand("UPDATE tblCars
SET Merk = @Merk, Model = @Model WHERE ID = @ID")

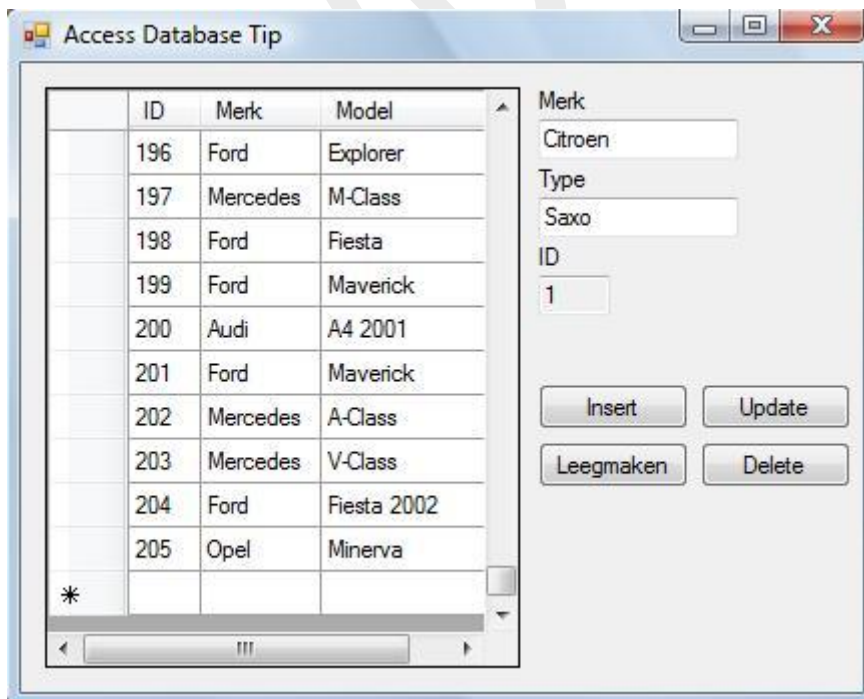
daAutos.UpdateCommand.Parameters.Add("@Merk",
OleDb.OleDbType.VarChar).Value = Me.txtMerk.Text
daAutos.UpdateCommand.Parameters.Add("@Model",
OleDb.OleDbType.VarChar).Value = Me.txtType.Text
daAutos.UpdateCommand.Parameters.Add("@ID",
OleDb.OleDbType.Integer).Value = cint(Me.txtId.Text)

connectie.open()
daAutos.UpdateCommand.ExecuteNonQuery()
connectie.close()

dataLezen()
```

Het woord "InsertCommand" is nu "UpdateCommand" geworden en het SQL-command is veranderd. En dan de placeholder @ID die word ingevuld. Let op dat deze van het type Integer is omdat onze kolom in de database het veld-type 'AutoNummering' heeft.

Als je het programma nu runt, een rij selecteerd, de gegevens aanpast (en dus niet op "Leegmaken" klikt, want dan zal het programma crashen. Ik hou het allemaal zo simpel mogelijk) en dan op "Update" klikt. Word je aanpassing doorgevoerd.



## Delete

Ga naar 'Designer Mode' en dubbelklik op de Button "Delete".

Bij de 'Click' sub voegen we een gelijkaardige code toe als bij Insert en Update:

Code:

```
daAutos.DeleteCommand.Connection = connective
daAutos.DeleteCommand = new OleDb.OleDbCommand("DELETE * FROM
tblCars WHERE ID = @ID")
daAutos.DeleteCommand.Parameters.Add("@ID",
OleDb.OleDbType.VarChar).Value = cint(Me.txtId.Text)

connectie.open()
daAutos.DeleteCommand.ExecuteNonQuery()
connectie.close()

dataLezen()
```

## Source code

---

Code:

```
Public Class Form1

Dim connectieString As String =
"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=C:\Users\pC\Documents\Visual Studio
2008\Projects\VBIBAccess\VBIBAccess\db1.mdb"

Dim dtAutos As New DataTable
Dim connectie As New OleDb.OleDbConnection(connectieString)
Dim daAutos As New OleDb.OleDbDataAdapter("SELECT * FROM
tblCars", connectie)

Private Sub Form1_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load

dataLezen()
dgvAutos.DataSource = dtAutos

Me.txtId.DataBindings.Add("Text", dtAutos, "ID", False,
DataSourceUpdateMode.Never)
Me.txtMerk.DataBindings.Add("Text", dtAutos, "Merk", False,
DataSourceUpdateMode.Never)
Me.txtType.DataBindings.Add("Text", dtAutos, "Model", False,
DataSourceUpdateMode.Never)

End Sub

Private Sub btnLeegmaken_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnLeegmaken.Click

For Each ctrl As Control In Me.Controls
If TypeOf ctrl Is TextBox Then
ctrl.Text = String.Empty
End If
Next

End Sub

Public Sub dataLezen()

dtAutos.Clear()
daAutos.Fill(dtAutos)

End Sub
```

---

---

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

```
daAutos.InsertCommand.Connection = connectie  
daAutos.InsertCommand = New OleDb.OleDbCommand("INSERT INTO  
tblCars (Merk, Model) VALUES (@Merk, @Model)")
```

```
daAutos.InsertCommand.Parameters.Add("@Merk",  
OleDb.OleDbType.VarChar).Value = Me.txtMerk.Text  
daAutos.InsertCommand.Parameters.Add("@Model",  
OleDb.OleDbType.VarChar).Value = Me.txtType.Text
```

```
connectie.Open()  
daAutos.InsertCommand.ExecuteNonQuery()  
connectie.Close()
```

```
dataLezen()
```

```
End Sub
```

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

```
daAutos.UpdateCommand.Connection = connectie  
daAutos.UpdateCommand = New OleDb.OleDbCommand("UPDATE tblCars  
SET Merk = @Merk, Model = @Model WHERE ID = @ID")
```

```
daAutos.UpdateCommand.Parameters.Add("@Merk",  
OleDb.OleDbType.VarChar).Value = Me.txtMerk.Text  
daAutos.UpdateCommand.Parameters.Add("@Model",  
OleDb.OleDbType.VarChar).Value = Me.txtType.Text  
daAutos.UpdateCommand.Parameters.Add("@ID",  
OleDb.OleDbType.Integer).Value = CInt(Me.txtId.Text)
```

```
connectie.Open()  
daAutos.UpdateCommand.ExecuteNonQuery()  
connectie.Close()
```

```
dataLezen()
```

```
End Sub
```

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

```
daAutos.DeleteCommand.Connection = connectie  
daAutos.DeleteCommand = New OleDb.OleDbCommand("DELETE * FROM  
tblCars WHERE ID = @ID")
```

```
daAutos.DeleteCommand.Parameters.Add("@ID",  
OleDb.OleDbType.VarChar).Value = CInt(Me.txtId.Text)
```

```
connectie.Open()  
daAutos.DeleteCommand.ExecuteNonQuery()  
connectie.Close()
```

```
dataLezen()
```

```
End Sub
```

```
End Class
```

Hypenate

## Slot

Het hele programma is [hier](#) te downloaden.

Hypernate