All change, please

Mark Whitehorn explains why changing properties needn't be as difficult as moving house.

The topic for this month is how to make multiple changes – first to data in the database and then to the database itself. I didn't plan it this way, it came about from a couple of emails that I received. At first glance they seemed to be unrelated, but as I wrote the column I discovered they went together like penguins and snow.

So let's start with the data. Redouane Doumer emailed with the following: 'I have been given a database. One of the tables contains suppliers' details. One of the fields includes the telephone numbers, but for some reason, the '0' is missing at the beginning. Because the table includes more than 3,000 records, I wonder if I could run a query that will add the missing '0', rather than doing it manually?'

An UPDATE query should do the trick. Given that the name of the field is, for example, TelNo, then the SQL expression to use in the update query will be something like:

```
UPDATE Table1 SET ↙
Table1.telno = '0'+[telno];
```

(*Key: ↙ code string continues*)

Clearly you will have to substitute the appropriate table and field names. You can build this in raw SQL or use the Access query builder (see the screenshot, above).

Remember, before you start, it's important to make sure that the TelNo field you have inherited is a text field and not a number field.

This is clearly a specific answer to Redouane's question, but it applies generally. You should never have to manually update lots of individual records. If you haven't come across update queries before, they are well worth investigating in detail because they can do so much work for you, enabling you to make changes to all (or some) of the records in a table with a single command. Is it a bird? Is it a plane? No, it's Update Query!

As a rather pathetically obvious plug, update queries are covered in more detail in the *Inside Relational Databases* book that can be purchased from *PCW*. However, before I get too smug, that noble tome totally fails to say anything at all about the next subject, so read on.

Databases contain more than just data, they also contain lots of other objects, such as forms, queries, etc. Objects such as forms contain other objects, such as text boxes, combo boxes, etc. All of these objects have properties such as colour, size, etc.

Suppose that you have hand-crafted an extensive database over a number of months. Then, one day, you realise that you need to change the property of every instance of a

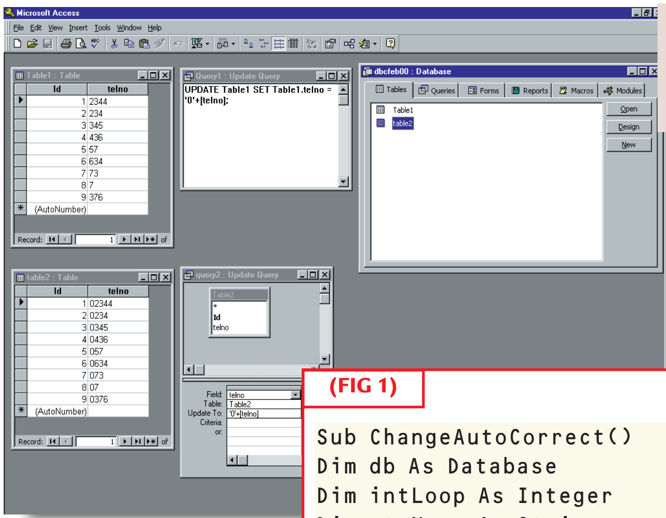


Table1 : Table

Id	telno
1	2344
2	234
3	345
4	436
5	57
6	634
7	73
8	7
9	376
(AutoNumber)	

Query1 : Update Query

```
UPDATE Table1 SET Table1.telno =
'0'+[telno];
```

table2 : Table

Id	telno
1	02344
2	0234
3	0345
4	0436
5	057
6	0634
7	073
8	07
9	0376
(AutoNumber)	

query2 : Update Query

Field: telno
Table: Table2
Update To: '0'+[telno]
Criteria:
or:

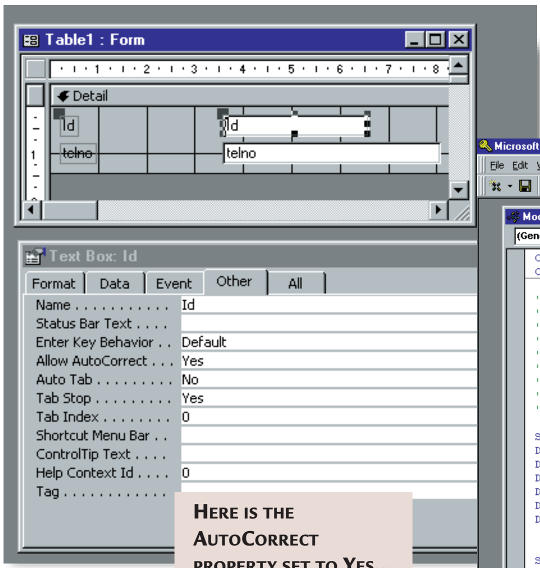
Microsoft Access
File Edit View Insert Tools Window Help
dbcfeb00 : Database
Tables Queries Forms Reports Macros Modules
Table1
table2
Open
Design
New
Record: 1 of

UPDATE QUERIES REMOVE THE NEED TO MANUALLY CHANGE SETTINGS

(FIG 1)

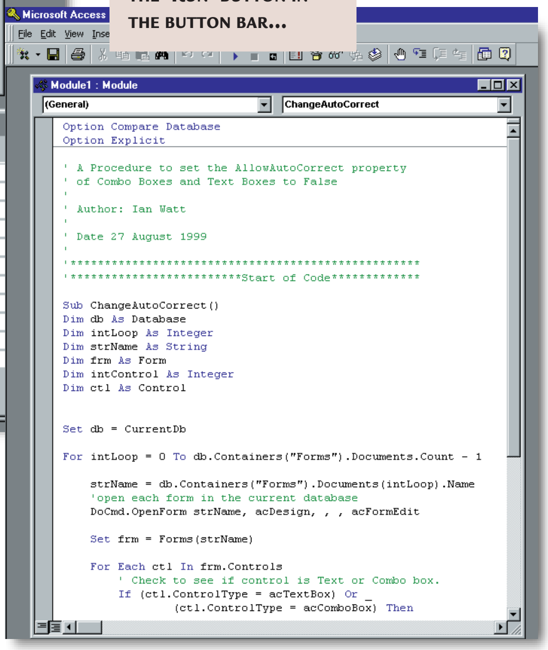
```
Sub ChangeAutoCorrect()
Dim db As Database
Dim intLoop As Integer
Dim strName As String
Dim frm As Form
Dim intControl As Integer
Dim ctl As Control

Set db = CurrentDb

For intLoop = 0 To db.Containers("Forms").↙
Documents.Count - 1

   strName = db.Containers("Forms").↙
Documents(intLoop).Name
    'open each form in the current database
    DoCmd.OpenForm strName, acDesign, , , ↙
acFormEdit

    Set frm = Forms(strName)

    For Each ctl In frm.Controls
      ' Check to see if control is Text or ↙
Combo box.
      If (ctl.ControlType = acTextBox) Or _
         (ctl.ControlType = acComboBox) Then
          ' Set control properties.
          With ctl
              .AllowAutoCorrect = False
          End With

        End If
    Next ctl

DoCmd.Close acForm, strName, acSaveYes
Next intLoop

End Sub
```

(*Key: ↙ code string continues*)



Name	Id
Status Bar Text	
Enter Key Behavior	Default
Allow AutoCorrect	Yes
Auto Tab	No
Tab Stop	Yes
Tab Index	0
Shortcut Menu Bar	
ControlTip Text	
Help Context Id	0
Tag	

HERE IS THE AUTOCORRECT PROPERTY SET TO YES...



```
Option Compare Database
Option Explicit

' A Procedure to set the AllowAutoCorrect property
' of Combo Boxes and Text Boxes to False
'
' Author: Ian Watt
'
' Date 27 August 1999
'
'****************************************************
'************************Start of Code*************

Sub ChangeAutoCorrect()
Dim db As Database
Dim intLoop As Integer
Dim strName As String
Dim frm As Form
Dim intControl As Integer
Dim ctl As Control


Set db = CurrentDb

For intLoop = 0 To db.Containers("Forms").Documents.Count - 1

    strName = db.Containers("Forms").Documents(intLoop).Name
    'open each form in the current database
    DoCmd.OpenForm strName, acDesign, , , acFormEdit

    Set frm = Forms(strName)

    For Each ctl In frm.Controls
        ' Check to see if control is Text or Combo box.
        If (ctl.ControlType = acTextBox) Or _
                (ctl.ControlType = acComboBox) Then
```

...BUT BY OPENING THE MODULE AND PRESSING THE 'RUN' BUTTON IN THE BUTTON BAR...

particular object in the database (say, the colour of every combo box). The thought of doing all of that by hand is horribly daunting and, despite the fact that update queries enable you to leap tall buildings with a single bound, sadly they are not much use here. But there's more than one way to skin a database...

To prove that this isn't an abstract problem, this solution came about as an answer to a real problem. Ken Sheridan's email in the November column pointed out that Microsoft's AutoCorrect feature could have 'interesting' consequences for database users. If the AutoCorrect property for a control on a form is set to True then it can change correctly-entered data into something that might be desirable elsewhere, but is totally inappropriate in the current context. He pointed out that setting the control's AutoCorrect property to False cures this problem.

This clearly struck a chord with several readers. In particular, the same problem had already caused Ian Watt some serious grief and he emailed me with a generic cure.

Ian has a group of users who share a database, with copies of the front end on the PCs and the data tables on a server. Values can be entered into a combo box, which then auto-completes the nearest match, taking values from a table on the server. If no matches are found, the user is offered the chance to add the data to the table.

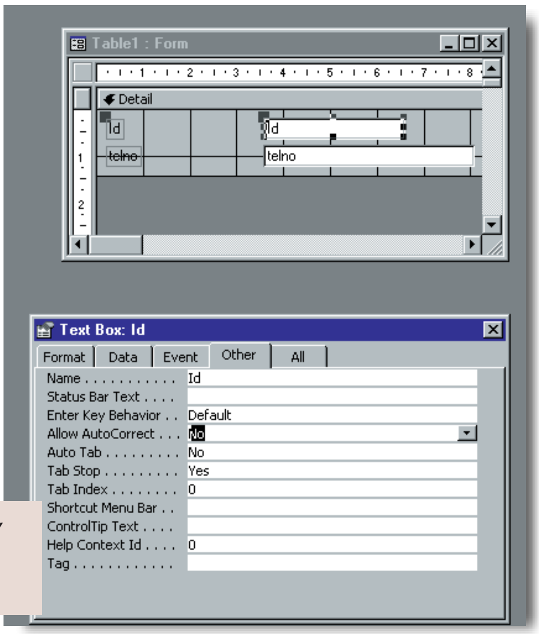
One user found that, when she entered a value (eg James) it picked up a value for James Stick stored in the table. When she tabbed to the next control, the value in the combo box was changed to James Stock and it offered her the chance to add James Stock to the table. She changed it back to James Stick and when she tabbed it again it changed once more to James Stock and she was offered the chance... loop ad infinitum.

This happened on no other PC. The data tables are shared and even when the front end was copied from a PC on which it worked to her PC, the problem persisted.

After much work it was discovered that this user had set AutoCorrect in Microsoft Word to change 'stick' to 'stock' and Access had inherited this command.

Ian found the same solution as Ken (namely, that setting the control's AutoCorrect property to False cured the problem) but realised that the same problem potentially lurked in all the controls in all his existing databases, some of which had numerous forms with many Combo and Text Boxes. His solution is an elegant piece of code (see Fig 1 on opposite page).

An Access database that demonstrates this is available on this month's cover disc. I have copied the code to this sample database and if you run it, it will change the properties of the objects in the forms.



Name	Id
Status Bar Text	
Enter Key Behavior	Default
Allow AutoCorrect	No
Auto Tab	No
Tab Stop	Yes
Tab Index	0
Shortcut Menu Bar	
ControlTip Text	
Help Context Id	0
Tag	

...IT MAGICALLY CHANGES THE PROPERTIES

How can you use this code more generally? You simply add references to the properties that you want to change. For example:

```
' Set control properties.
 With ctl
  .AllowAutoCorrect = False
   .BackColor = .BackColor + 1
```

has the bizarre effect of altering the background colour of the controls every time the code is run.

As always, this code is provided with no guarantees; be sure to back up your database before trying it. But it worked when I tried it.

PCW CONTACTS

Mark Whitehorn welcomes your feedback on the Databases column. Contact him via the PCW editorial office, or email: database@pcw.co.uk