All buttoned up

Mark Whitehorn presents another slant on last month's buttons and captions query, which is far from sewn up. Plus, the complexities of data entry in Access, and CUSTOMER care.

In the February issue I published a letter from Glen Rowe. He wanted multiple buttons on a form, all with different captions. Whenever a button was pressed, a query would run which returned values based upon the caption on the button. For example, if you press the button labelled "Penguins" you see the records which relate to Penguins: pressing the "Fish" button yields information about Fish. I produced a solution by building a query which snatched the caption from the button which had just been pressed.

I also wrote the following: "The obvious solution at first is to try to pass the button's caption to the query as a parameter. As far as I know (and I stand to be corrected) this can't be done."

James Talbut replied: "Passing a parameter to a parameter query in Access can be done, but it's not very pleasant and means the you need to use DAO (Data Access Objects), which can be good or bad depending on the circumstances."

The solution that James then provides does, as he suggests, pass a parameter to a parameter query. What it doesn't do is pass the *caption* of the button as a parameter (which is still, as far as I know, impossible). In turn, this means his solution is somewhat more awkward than the one shown in the February issue because every button you add to the form must have its caption *and* its OnClick property altered. With the February solution, all you had to do was clone the button and alter the caption. However, James's solution to the initial problem is still well worth studying since it illustrates a very different way of solving the conundrum.

He continues: "Before I include all the bumf to show you how it's done, I'll just mention the implications.

"To actually access the resultant data you need to use DAO to step through a recordset. If you have a large amount of data this can be inefficient (OK, it's always inefficient, but it's more noticeable with a large dataset), particularly if you are intending to perform some secondary operation/selection on the data. However, the query itself is still run by the JET engine so the main data manipulation routines are still run as quickly as Access can do them.

"To control the parameters you need to use the Parameters collection of the QueryDef object for the query you are interested in. The most simple example I could come up with looks like this:

Table: "People"

ID	Name	Value
1	Fred	1
2	Jack	4
3	Fred	9
4	Harry	16
5	Anastasia	25
6	Bob	36
7	Martin	49
8	Stephen	64
9	Harry	81

(See Fig 1.)

```
Fig 1 Query, "Person":
SELECT DISTINCTROW People.ID, People.Name, People.Value
FROM People
WHERE (((People.Name)=[Person]));
```

```
Fig 2 Form Code, behind "People"
Private Function Button_Click(sName As String)
Dim qdfPeople As QueryDef
Dim rsPerson As Recordset
Dim sRowSource As String

Set qdfPeople = CurrentDb.QueryDefs("Person")

 qdfPeople.PARAMETERS("Person") = sName
 Set rsPerson = qdfPeople.
OpenRecordset

 While Not rsPerson.EOF
   If Len(sRowSource) > 0 Then sRowSource = sRowSource & ";"
   sRowSource = sRowSource & rsPerson.Fields("Name") & " - " &
rsPerson.Fields("Value")
rsPerson.MoveNext
 Wend
 List10.RowSourceType = "Value List"
 List10.RowSource = sRowSource
End Function
```

p286 ➢

Form, “People”:
Six command buttons, each with their OnClick property set to:

```
=Button_Click(“Fred”)
```

with Fred replaced by the caption on that particular button. The buttons are labelled Fred, Bob, Harry, Martin, Anastasia and Stephen. At the bottom is an empty list box called List10.
(See Fig 2, page 285.)

“And that’s it. Hope it’s useful.”

Very interesting, James. I have constructed an example file (in Access 7.0) based on this example called TALBUT.MDB which is on the CD.

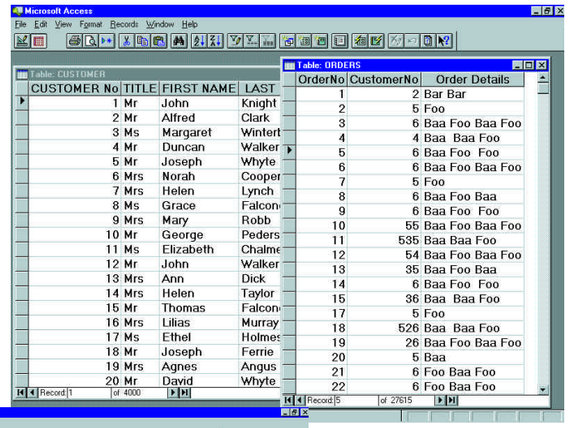
Next an email from Norway: “I have a problem that I’ve worked on for several months now, without finding any easy solution,” *writes Tore Saetre, of Bergen*. “I’ve developed and am maintaining a Microsoft Access database that keeps track of customers. The database contains two main tables: Customer and Sales. The problem is that I need to have a field in the Customers-table that shows how many orders the customer has in the Sales-table.

“My first idea was to make a query that counts orders in Sales for each customer. The query lists each customer and how many orders the customers has. I tried to move that value to the Customer-table with an update query that updates a NoOrders-field in Customer through a link to the first query. The problem is that the first query can’t be updated and the second query won’t update my Customers-table.”

I receive many questions like this one, questions which hinge upon the desire to store redundant data in tables. My initial reply to Tore was that, in general, storing derivable data is a bad idea because if the data changes (as you make more sales), the data in the CUSTOMER table goes out of date. It is normally preferable to use a query to calculate the information you need: whenever you want to see this data, you can run the query which shows you the customer details and the number of sales.

Tore was only partially convinced and replied: “But data in the sales table is only changed and added to once a month. The rest of the month the user browses through CUSTOMER’s 4,000 records and needs to see as much data on each customer as possible. Running a query for each customer is too time-consuming. I see why it can’t be done in a ordinary customer/sales database, but I still hope to find a solution to this problem.

Fig 3 (right) Small samples from both the CUSTOMER and ORDERS tables in the database “Tore”



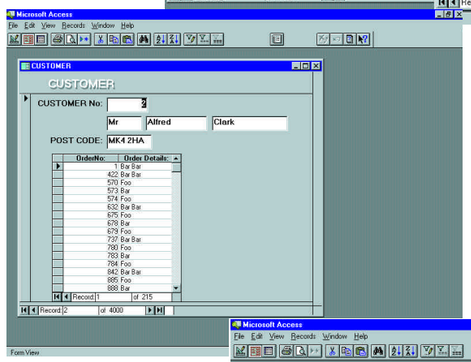
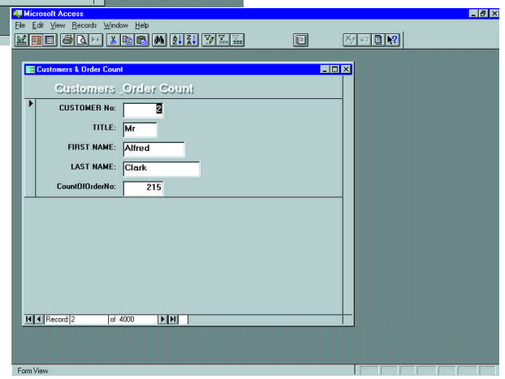


Fig 4 (left) The form called “Customers & Orders” shows customer details and order details

Fig 5 (below) The form called “Customers & Orders Count” shows one record for each customer



“For an experiment, do you have an example of how I can run a query to see customer details based on the customer currently displayed in a form? Do I need to run a code, macro or function and how do I get the query to filter everything but the record on-screen? Thanks for your help.”

Okay, we’ll solve this one both ways. There is a sample database on the cover CD (in Access 2.0) called TORE.MDB. This has two tables ,CUSTOMER and ORDERS. CUSTOMER contains simple details of 4,000 mythical people, ORDERS contains about 27,500 orders. Each customer has at least three orders to their name (or, in this case, to their CustomerNo) and some have considerably more. Fig 3 shows small samples from both tables.

The form called “Customers & Orders” (Fig 4) shows customer details and order details. The main form is based on CUSTOMER, the sub-form (which is called Sub1) is based on a query which simply performs a join on the two base tables. This form displays one record for each of the 4,000 customers.

The form called “Customers & Orders Count” (Fig 5) again shows one record for each customer. However, instead of listing the details about each order, it simply shows how many orders each customer has placed. This form is based upon the query called “Customer & Order Count”.

These two forms are variants which answer Tore’s request for an “example of how I can run a query to see customer details based on the customer currently displayed in a form”.

I ran this database on a 486/100 with 16Mb of RAM using Access 2.0. “Customers & Orders” opens almost instantaneously, and you can scroll through the records at the rate of about 400 per minute. “Customers & Orders Count” takes about 25 seconds to open, but once open you can scroll through the records at the rate of about 1,600 per minute.

To put this into simplistic terms, the first one does the calculations for each record as you ask to see it (which is adding about 0.1 seconds to the scroll time between each pair of records). The second one does all of the calculations for all of the records before showing any of them to you.

These forms are simply based on queries: no code or macros are required. As you can see, using the first form, it is possible to see customer and order details with essentially no speed hit at all for these numbers of records.

The query called “Customers & Orders Count” (the one which takes 25 seconds to run) can also be used, if required, to create the base table that Tore requested. I have included a Make-Table version of the same query in TORE.MDB. If you run this, it will create a base table called CUST which is just like CUSTOMERS except that it includes a field which shows how many orders each customer has placed.

In Tore’s case this could be run once a month, and then CUSTOMERS replaced by CUST; in that case, the 25-second speed hit disappears. The downside is that we are now reliant upon redundant data in the database. If anyone forgets to renew the tables at the end of the month, then all of the users of the database start to work with inaccurate data. You pays your money, you takes your choice — speed or security.

Quickies

■ Here’s a quick one from Gareth Wade: “When running a simple Report, I need to add all the time lengths and give a running

Fig 6

```
Private Sub LastName_AfterUpdate()

End Sub
```

Simply add a line so that it now reads;

```
Private Sub LastName_AfterUpdate()

    Me!LastName = StrConv(Me!
LastName, 3)

End Sub
```

Fig 7

```
Me!LastName = StrConv(Me!LastName, 3)
```

Fig 8

```
Private Sub LastName_AfterUpdate()
Dim Length As Integer

Me!LastName = StrConv(Me!LastName, 3)
If Left(Me!LastName, 3) = “Mac” Then
   Length = Len(Me!LastName)Length - 3)
   Me!LastName = StrConv(Me!LastName, 3)
   Me!LastName = “Mac”  Me!LastName
End If

If Left(Me!LastName, 2) = “Mc” Then
   Length = Len(Me!LastName)
   Me!LastName = Right(Me!LastName, Length - 2)
   Me!LastName = StrConv(Me!LastName, 3)
   Me!LastName = “Mc” + Me!LastNameEnd If

End Sub
```

p288 ➢

total. Easy enough; but when Access gets to 23:59, it reverts back to 00:00. What format do I use to carry on past that magic 24-hour mark ? I have tried all the manuals and help files but can't find the answer."

I don't know the answer to this one. Anyone else care to solve it?

■ And this one from Malcolm Rowley: "I find I am put off by the complexity required in Access to solve simple problems, e.g. data entry character formatting.

In Alpha I simply have to specify the word/character format I require, e.g. Upper, Lower, Word (first character upper case, the rest of the data left as is) in the field rules, and this is applied to all data entry that takes place and can be re-applied to all existing data. This is ideal for data entry of names and addresses; the only difficulty being the McCartneys (etc.) of this world. To have any simple form of formatting that stores the data in an access table in a particular format, do I really have to resort to code? Is there a simple way of accepting data entry in Access and storing it as first character upper case, the rest left as is?

For example: macdonalds becomes Macdonalds / macIntyre becomes MacIntyre / 27 park avenue becomes 27 Park Avenue. A simple solution would be appreciated."

The answer is that you have to use code, but only one line so it may just meet your requirement for simplicity! Suppose you have a form called Capital which displays a field called LastName in a text box called LastName. Switch to design, double-click on the textbox in question, select the Event properties, click on the one called "After Update", click on the ellipsis button which appears (three dots) and choose Code Builder. Between the two lines which appear (Fig 6) "Me" means the current form, "LastName" is the name of the textbox, and "StrConv" is a function (only available in Access 7.0 and above) which converts strings. The "three" tells the string conversion function to do the type of conversion that you asked for (capitalising the first letter of each word).

So, Fig 7 means "make the contents of the textbox called LastName equal to the same as it is now, but with all of the first letters of the words capitalised". This will convert: penguin penguinsson to Penguin Penguinsson / 23 the larches to 23 The Larches / mcdonald to Mcdonald and so on.

■ On a slightly different tack, I haven't used Alpha for well over a year, but I agree that

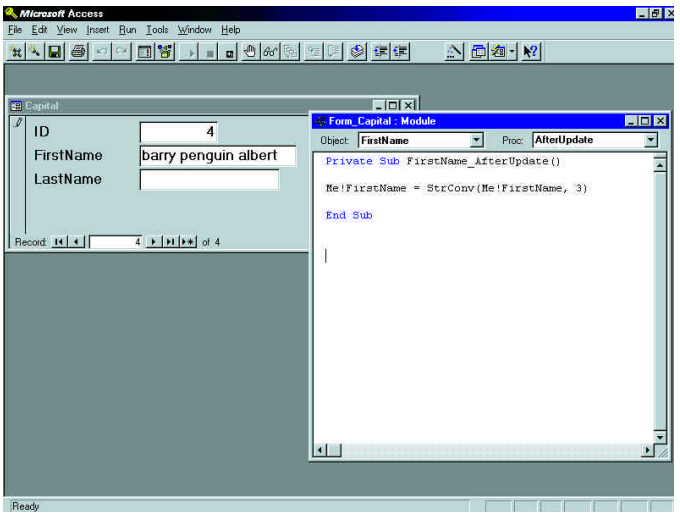
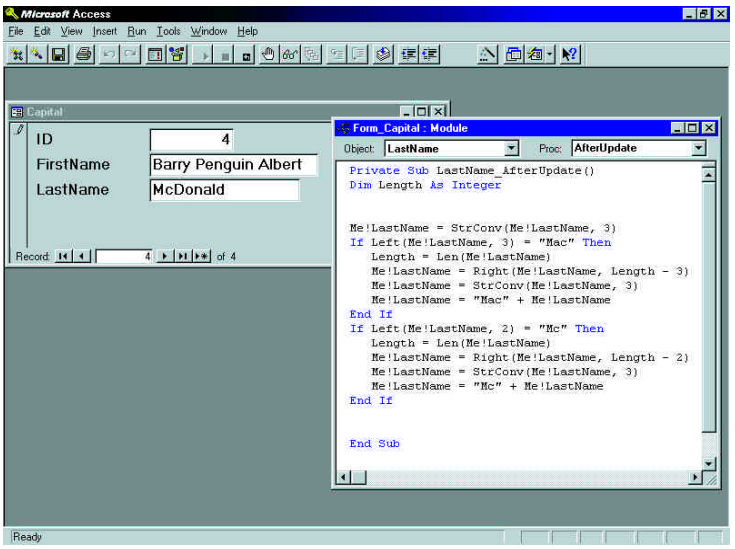


Fig 9 (left) This single line of code will capitalise all distinct words placed in the FirstName field *(see Fig 10)*

Fig 10 (below) This more complex code capitalises two of the more complex surname types in the LastName field



processes like this one are easier (and certainly more intuitive) in Alpha than in Access. In fact, to quote from a review I wrote at the time of its release: "…of these, the Field Rules are the most impressive, if only because Alpha copes with them better than any other RDBMS I have seen."

There is clearly a trade-off in design terms, which these two products exemplify. In both products, the designers identified a host of commonly needed functions and made them easily available from the interface. This approach has pros and cons. The good news is that commonly performed processes (like designing tables and building queries) are easy: the bad news is that if you happen to want a function that the designers didn't provide, you suddenly have to go to a more complex area, such as coding.

One difference between the two products is where the line was drawn. In this case, the Alpha designers decided to include capitalisation in the "easy" set and the Access designers didn't.

I have always found that, whatever product I use, I eventually reach a stage where I have to use code simply because it is impossible for the designers to put everything into the "easy" set. On a happier note, once you do start to use code, your horizons expand considerably. You can, for example, begin to deal with unusually capitalised names like those you mention. If you expand the code to that shown in Figs 9 and 10, this will change macdonalds to MacDonalds, and mctavish to McTavish. (Also see the Access 7.0 sample database called ROWLEY.MDB.)

I am not suggesting that this is perfect code. As usual in the sample code I give, there is no error trapping. In addition, there are some people who prefer the letter after Mac or Mc to be left in lower case. This code is simply provided as an example of what can be done.

PCW Contact

Mark Whitehorn welcomes readers' correspondence and ideas for the Databases column at database@pcw.vnu.co.uk