Smart Search without the Hassles

Nirmala Sekhar

95 97 2000 2002 DOWNLOAD

No database application is complete without a comprehensive search facility. There are many methods of incorporating search routines in a form or a query. In this article, Nirmala Sekhar demonstrates a simple method of searching your data based on a given list of relevant fields. The example can be easily used in your database with very few changes.

AS I developed various applications in Access, I realized that almost all of my applications needed a search form. The earliest applications I developed had simple combo boxes in an unbound search form. Once the user had selected the values for the search, a command button would activate a query based on these combo boxes. The problem with this approach is that it involved a lot of hard-coding in the queries. And, of course, I had to make many amendments to the combo boxes as I moved from one database to the next.

As I spent some time thinking about this issue, I realized that although the fields and the tables varied from one application to another, the search process itself was more or less the same. The search process involved:

- A main search form
- The search to be based on a list of fields
- A subform that contained the resulting records
- The ability to choose the sort sequence for the resulting records
- An option to view all of the records (especially for smaller databases)

The ability to refine the search based on the current set of found records would be a nice feature to have (especially for databases with many thousands of records).

In **Figure 1**, you can see a sample search form that uses the Customers table in Northwind database.

Smart search

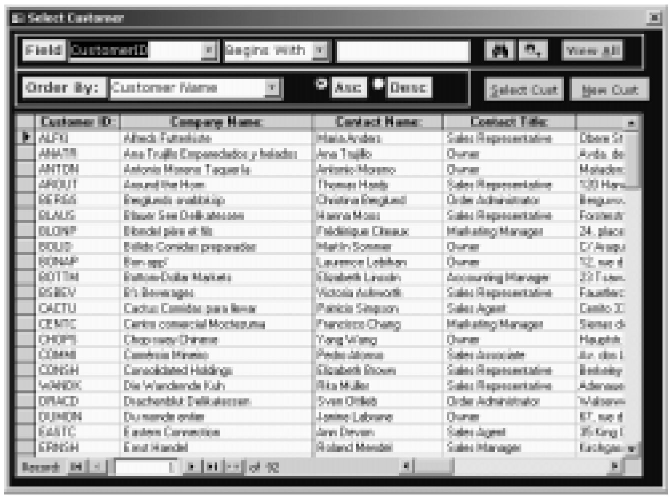
For my sample, I've used the Customers table from the Northwind database. The search form consisted of:

- A combo box that contains a list of suitable columns in the Customers table (see **Figure 2**)
- A combo box that defines where to look (that is, in the beginning of the field or anywhere in the field)
- A text box to input the search text

The code behind the search is very generic and can be used in any database just by changing the name of the table in the SQL command:

```
strSQL = "Select * from Customers Where [" _
         & Me![cboField] & "] "
strValue = Trim(Nz(Me![txtValue]))
Select Case Me![cboSearchType]
  Case 1 'Search in the beginning
         strSQL = strSQL & "Like '" & strValue & "*' "
  Case 2 'Search anywhere
         strSQL = strSQL & "Like '*" & strValue & "*' "
End Select
```

Once you have the SQL statement changed, all you have to do is change the recordsource of the



Customer ID:	Company Name:	Contact Name:	Contact Title:	
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere St
ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner	Avda. de
ANTON	Antonio Moreno Taquería	Antonio Moreno	Owner	Matadero
AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Han
BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator	Berguvs
BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative	Forsterstr
BLONP	Blondel père et fils	Frédérique Citeaux	Marketing Manager	24, place
BOLID	Bólido Comidas preparadas	Martín Sommer	Owner	C/ Araqu
BONAP	Bon app'	Laurence Lebihan	Owner	12, rue d
BOTTM	Bottom-Dollar Markets	Elizabeth Lincoln	Accounting Manager	23 Tsaw
BSBEV	B's Beverages	Victoria Ashworth	Sales Representative	Fauntlec
CACTU	Cactus Comidas para llevar	Patricio Simpson	Sales Agent	Cerrito 33
CENTC	Centro comercial Moctezuma	Francisco Chang	Marketing Manager	Sierras de
CHOPS	Chop-suey Chinese	Yang Wang	Owner	Hauptstr
COMMI	Comércio Mineiro	Pedro Afonso	Sales Associate	Av. dos L
CONSH	Consolidated Holdings	Elizabeth Brown	Sales Representative	Berkeley
WANDK	Die Wandernde Kuh	Rita Müller	Sales Representative	Adenaue
DRACD	Drachenblut Delikatessen	Sven Ottlieb	Order Administrator	Walserw
DUMON	Du monde entier	Janine Labrune	Owner	67, rue d
EASTC	Eastern Connection	Ann Devon	Sales Agent	35 King G
ERNSH	Ernst Handel	Roland Mendel	Sales Manager	Kirchgas

Figure 1. A sample search form.

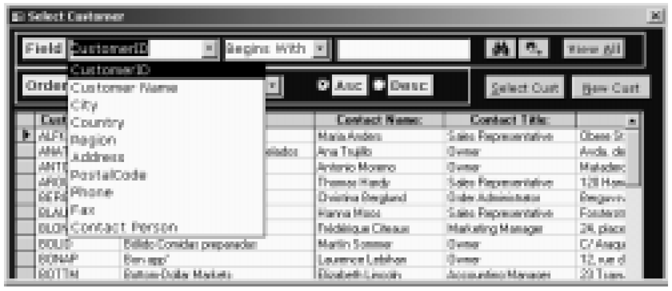


Figure 2. The combo box for selecting search columns.

subform. In this example, sub1 refers to the name of the subform control:

```
Me![sub1].Form.RecordSource = strSQL
Me![sub1].Requery
```

Of course, if the Form's View All option is turned on, then the SQL statement would be limited to this:

```
Select * from Customers
```

Smart sort

Very often, users need to search by one column and sort the results using another column. A separate combo box for the sort criteria with column names from the underlying table or query does the trick:

```
Public Function fnSort(strSQL As String)
Dim strTemp As String
 strTemp = strSQL & " ORDER By [" & Me![cboSort] & "]"
 Select Case Me![optsort]
    Case 1
        strTemp = strTemp & " ASC;"
    Case 2
        strTemp = strTemp & " DESC;"
 End Select
 Me![sub1].Form.RecordSource = strTemp
 Me![sub1].Requery
End Function
```

Progressive search

When dealing with a large number of records, users really enjoy the ability to refine their search. For instance, a user might be looking for a customer named John. The first run of the search returns a few hundred Johns. The user would like to narrow down the search to Johns living in England. This involves:

1. Identifying the current recordsource of the subform
2. Eliminating the Order clause from the preceding SQL statement
3. Adding another condition to the SQL statement's WHERE clause
4. Finally, restoring the ORDER clause

Here's the code:

```
Private Sub cmdProgSearch_Click()
Dim strSQL As String
Dim strValue As String
Dim varX
   strValue = Trim(Nz(Me![txtValue]))
   If strValue <> "" Then
      strSQL = Me![sub1].Form.RecordSource
      varX = InStr(1, strSQL, "ORDER")
      If Not IsNull(varX) And varX > 10 Then
         strSQL = Left(strSQL, varX - 1) & _
                  " And  [" & Me![cboField] & "] "
      End If

      Select Case Me![cboSearchType]
      Case 1   'Search in the beginning
        strSQL = strSQL & "Like '" & strValue & "*' "
      Case 2   'Search anywhere
        strSQL = strSQL & "Like '*" & strValue & "*' "
      End Select

      Call fnSort(strSQL)
   End If
End Sub
```

Selecting the customer

Once you have the customer that you're looking for, you need to be able to modify the data or view other details about the customer. I've used the Customers form from the Northwind database. The code for opening the Customers form is very simple:

```
Private Sub cmdSelect_Click()
Dim strA As String
If IsNull(sub1.Form!CustomerID) Or _
         sub1.Form.RecordsetClone.RecordCount = 0 Then
   MsgBox "Please highlight the Customer that " _
          & "you want to select.", vbExclamation
Else
   strA = "[CustomerID]='" & _
          Me!sub1.Form!CustomerID & "'"
   DoCmd.OpenForm "Customers", , , strA, _
                  acFormEdit, acDialog
End If
End Sub
```

Similar code can also be used in the DoubleClick event of the subform so that a simple double-click action will launch the Customers form. Once the customer details have been edited, the subform data is requeried in the Close event of the Customers form:

```
Private Sub Form_Close()
    Forms!frmSearch!sub1.Requery
End Sub
```

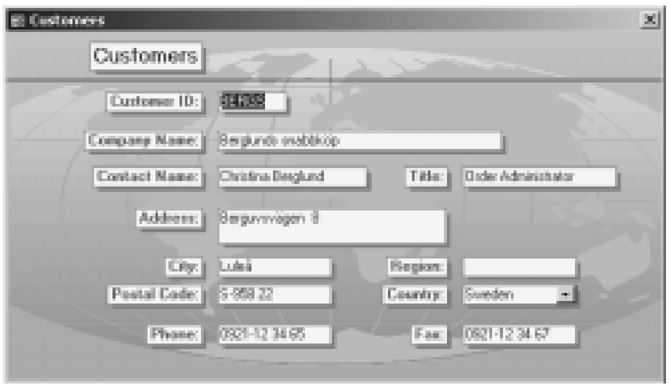
New customer

If the customer isn't present in the database, then a new customer can be created by using the same Customers form (see Figure 3). I've removed the navigation buttons of the Customers. Since the Customers form is opened in Dialog mode, this will prevent the creation of a new customer when the user is editing an existing customer:

```
Private Sub cmdNew_Click()
  DoCmd.OpenForm "Customers", , , ,acFormAdd, acDialog
End Sub
```

Reusing the form

The entire search functionality can be reused very easily. What you need to do is:



Customer ID:	BERGS
Company Name:	Berglunds snabbköp
Contact Name:	Christina Berglund
Title:	Order Administrator
Address:	Berguvsvägen 8
City:	Luleå
Region:	
Postal Code:	S-958 22
Country:	Sweden
Phone:	0921-12 34 65
Fax:	0921-12 34 67

Figure 3. The modified Customers form.

1. Change the list of column names in the two combo boxes.
2. Change the name of the Table / Form / Column names used in the code.
3. Create a simple form in datasheet view to display the relevant information. This form will be used as the Sourceobject for the subform in the main Search form.

Happy searching! ▲

DOWNLOAD **SEARCH.ZIP at www.smartaccessnewsletter.com**

Nirmala Sekhar runs her own software consultancy in Singapore. She's currently working on Access, ASP, and SQL Server 2000 platforms. nirmala@saicomsystems.com.

Access Subquery Techniques...

Continued from page 5

The path to SQL enlightenment

Subqueries are perhaps the simplest queries in Access that absolutely require you to write some SQL. Even if you use the QBE grid to construct your overall query (as I did for the first example in this article), you can't avoid writing an SQL statement for the subquery, either in a field definition or in a WHERE or HAVING clause. The requirement to write SQL makes many beginning Access developers shy away from using subqueries. That's a pity, because some problems (for example, the ranking query or the top per group query) are most easily solved by subqueries.

Rather than avoiding subqueries, I urge you to embrace them. Learning enough SQL to write subqueries will help you gain confidence in writing SQL statements, and ultimately you'll find that you can use this knowledge to write other types of queries directly in SQL. Access is practically unique as a product in letting you switch easily from a graphical view of a query to an SQL view. By learning how to make this switch on your own, you'll develop SQL skills that will serve you well in other less flexible products, as well as in writing VBA code that uses SQL statements. ▲

DOWNLOAD **SUBQUERY.ZIP at www.smartaccessnewsletter.com**

Mike Gunderloy is an independent consultant and author who lives and raises chickens in eastern Washington state. He's a co-author (with Paul Litwin and Ken Getz) of *Access 2002 Developer's Handbook* and a co-author (with Ken Getz) of *ADO Developer's Handbook*, both coming from Sybex. MikeG1@larkfarm.com.