This free back issue of
THE QUICK ANSWER is provided courtesy of…

The Monthly Newsletter for Sesame Database Manager
http://www.insidesesame.com
Read a Free Issue of Inside Sesame!

AND

LANTICA
SOFTWARE LLC.

Makers of Sesame Database Manager
Compatible with Symantec Q&A™
http://www.lantica.com

Sesame Seeds

Merge Print Database Records On-the-Fly

TOM MARCELLUS

Our FINAL Issue See page 2

A common need among database users is the ability to instantly merge-print the data from the records they're viewing. This includes not only printing out merge documents such as letters, invoices, sales orders, envelopes, mailing labels or the like, but also having options to send the document via fax or as an email attachment.

Microsoft Word and WordPerfect — the two most widely used word processors — support these capabilities.

We'll use Microsoft Word as an example. You can design your merge documents in Word, and design your Sesame application to generate the *data source* file for Word to merge with those documents. Your Sesame application can prompt you to specify the Word document to merge with, then start Word and open the merge document you want to print. In Word, you can print out the document on your printer, fax it, or send it by email.

In a nutshell, here's how you can make this work:

- You create your merge documents in Word. They can be as fancy as you like, and can include graphic elements such as your company logo and digital signature. They can have an attractive invoice-like form design with rows and columns of merged data inside Word tables. They can be documents that merge print envelopes or mailing labels.

- You can add a Command button named "Merge Print" to your Sesame database form. Programming attached to this button can perform the following actions:

  1. Display a menu of your Word merge documents, such as "Thank You" letter, "Follow-Up" letter, "Print Invoice", "5160 Avery Labels," and the like. You can have as many documents available on this menu as you need.

  2. Ask you if you want to include just the record currently on-screen or all the records you've retrieved.

  3. When you select the document to merge print, programming updates a special "data source" file for Word with the data from the selected record or records, starts Word, and automatically opens the merge document. You simply click the *Merge to Printer* button to print the documents, then exit Word to return to the Sesame database record you were on.

It's hard to imagine an easier way to print merge documents on demand.

Setting Up the Merge-Printing Application

Putting it together requires the following three steps:

1. Create a simple batch file.

2. Program Sesame to generate a *data source* file for Word, start Word, and pass it the name of the merge document.

3. Create your Word documents and link them to the *data source* file.

Step 1 — Create a Simple Batch File

The most reliable way to launch Word from Sesame and pass it the name of the document to open is to use a batch file. This is particularly true when you're working with the long paths and filenames on today's PCs.

Time to Say 'Goodbye...'
But Soon, 'Hello Again'

WHEN we launched THE QUICK ANSWER way back June 1990, it never occurred to us that we were in for this long a haul. Today, after 162 issues and nearly 14 years, the time has come to say goodbye to THE QUICK ANSWER and move on, as Q&A itself moves on, to the new Sesame database manager due out next month from Lantica Software.

Thank you for supporting THE QUICK ANSWER for so long. You've stayed with us these past few years (some since the early years) as we've taken you deeper and deeper inside Sesame. Because of that, you'll have a huge advantage — a real jump start on the coming Q&A replacement product.

Also deserving of thanks are those who routinely contributed their time and talents to making THE QUICK ANSWER not just a successful publication, but a unique, personalized software newsletter. Without devoted Q&Aers like Bill Halpern, Alec Mulvey, Erika Yoxall (all founding members of Lantica and intimately involved in Sesame's development) and T.J. Shufflin, THE QUICK ANSWER would never have made it this far.

We planned to continue publication of THE QUICK ANSWER concurrently (at least for a time) with INSIDE SESAME, our new Sesame newsletter, but that just will not be possible. Between now and the middle of next month, we will be heavily involved with the launch of Sesame, so there will be a one-month hiatus before the first issue of INSIDE SESAME appears in January.

As a subscriber, you have two options. If you don't instruct us to the contrary, we will simply "rollover" any remaining issues in your current QUICK ANSWER subscription to INSIDE SESAME — adding a month to your expiration date to make up for the December break.

If this isn't what you want, email us at *mailbox@quickanswer.com*. We will tell you how many issues you have remaining, what your pro-rata refund will be, and send you a refund check. If you'd prefer to have your account credited for technical support time with us at Marble Publications (for Q&A *or* Sesame) in lieu of a refund, we're amenable. Just let us know.

The subscription price for INSIDE SESAME will be $109.00 per year, $20.00 more than THE QUICK ANSWER. So if you plan to migrate to Sesame in the forseeable future, if not immediately, it might behoove you to "use up" your subscription by letting it rollover to INSIDE SESAME. You'll not only save a few bucks, you'll discover ways to get more out of Sesame, from basic starter articles and tips (including migrating from Q&A), to reader questions and answers in our Help Desk column, to the more advanced techniques and application solutions that were the hallmark of THE QUICK ANSWER.

Like THE QUICK ANSWER, INSIDE SESAME will be available exclusively online — except for one essential difference: you will be able to *read* and optionally *print* your INSIDE SESAME issues right from your browser while connected to *www.insidesesame.com*. You will still be able to download your issues if you like (as Acrobat PDFs), but no more self-extracting executables that have to be run after download just to get a look at your issue.

Any sample databases or other files that come bundled with a particular issue of INSIDE SESAME will be separately downloadable using your Subscriber ID. We think this is a more sensible way to do it. Less fuss, more options. Even if your subscription expires, you'll *still* have access to all the issues and bundled freebies that were published and available when it was active. In other words, you'll always be able to get any issues and downloadables you missed or misplaced.

See you January 1st with INSIDE SESAME!

—Tom Marcellus

A typical installation of Microsoft Word will place the Word executable (winword.exe) in the following path:

```
c:\program  files\microsoft  office\
office\winword.exe
```

Check your PC's file system to make sure this is the case. Assuming it is, create a batch file with the following contents:

```
c:
cd\program  files\microsoft  office\office
winword.exe  %1
Exit
```

The *%1* is called a *replaceable parameter*. When your Sesame program runs this batch file, it will start Word and pass it the filename of the document to open. The *%1* will be replaced by that filename.

You can use NotePad, WordPad, or even Word to write the batch file. If you use WordPad or Word, be sure to save the file as a plain text file. Whatever program you use to create the file, name it *word.bat* and save it to:

```
c:\sesame\word.bat
```

Step 2 — Configure Your Sesame Application

Let's suppose you have an application for customers named *customers.db.* You want to set it up so you can merge print Word documents on demand as you browse your records. (This can be the sample customers.db database that is on the Sesame CD.) Follow these steps:

1. Open the application in Sesame.

2. On the Menu Tree, click open the *Application Utilities* menu.

3. Double-click *Redesign Current Application*.

4. Select a compatible Designer (.DSR) file from the files list, or type in a name for a new Designer file (such as *customer.dsr*). Sesame Designer will start.

5. On the Designer Menu Tree, open the *Forms* menu if it isn't already open.

6. Double-click on *Redesign/Customize a Form*.

7. Double-click on the form you want to work with. It should be one containing the layout elements (fields) you'll want to merge print. Designer will display the form in redesign mode.

8. Using the **Layout Element Adder**, click open the *Select Element Type* menu and choose *Command Button*.

9. In the Label field, type *Merge Print*.

10. Click the *Add Element to Form* button. Designer will place the button on the form.

11. Drag the new button to the bottom of the form, or to your preferred location.

12. In the **Commands** window, double-Click *Program Layout*. This opens the Program Editor.

13. In the Program Editor, click open the **Element** list and select your new *Merge Print* button.

14. Click open the **Event** list and select *On Element Entry*.

You can now type in your program. The program (next two pages) is designed to be adaptable to a variety of merge printing needs. It is apt to appear daunting at first. Just stick with it. All will be explained. Don't run this program until you've read this entire section. You have to create at least one merge document before the program will produce results.

Programming explanation

The program begins by declaring a bunch of variables and an array that will be required by the program as it runs.

Just below the comment line that says "Modifiable Parameters," there are three parameters you can modify as necessary.

• The *vDataPath* variable specifies the path to the *data source* file (*customer.txt*, in this case) that the program will create for your Word merge documents to use. If you don't have a *c:\sesame\docs* folder, we recommend that you create it now rather than changing the parameter in the program.

• The *vDocsPath* variable is the path to the folder where your Word merge documents are stored. The default is *c:\sesame\docs*.

• The *vWordPath* variable is the path to the word.bat batch file you created earlier: *c:\sesame\word.bat*.

Move down below the two subroutines to the "BEGIN MAIN PROGRAM" area. There, on the line below "Build documents menu," you'll see:

```
vSelectedDoc  =  @PopupMenu(
```

followed by the filenames of three documents — *5160*

```
var vFileHandle as Int              // For writing data file
var vHeader as String               // Data file header line
var vData as String                 // Data file data line
var n as Int = 1                     // General  purpose  counter
var vSelectedDoc as String          // Document selected from menu
var vOneOrAll as String              // Include record(s) option
var F as array[25] of String        // Mergeable fields array
var  vStartingRec   =  @ResultSetCurrentPosition()
var  vGoPrint  as String


// Next 3 lines user modifiable
var  vDataPath  as  String  =  "c:\sesame\docs\customer.txt"
var  vDocsPath  as  String  =  "c:\sesame\docs\"
var  vWordPath  as  String  =  "c:\sesame\word.bat"


SUBROUTINE    PrintCurrentRecordOnly()

If @Askuser("Merge  print  the  " + vSelectedDoc + "  for  "
+  vOneOrAll  +  "?","","")  Then
{

vGoPrint  =  "Yes"


n = 1
While  F[n]  <>  ""
{
    vHeader  =  vHeader + F[n]     + "^"
    vData    =  vData    + @(F[n]) + "^"
    n = n + 1
}

vFileHandle  =  fileOpen(vDataPath)
fileSeek(vFileHandle,   0)
fileWriteLn(vFileHandle,  vHeader  +  @NewLine()  +  vData)
fileClose(vFileHandle)

}

END   SUBROUTINE

SUBROUTINE    PrintAllRetrievedRecords()

var n as Int
var i as Int = 1

If  @Update  Then
{

If @Askuser("Merge  Print  " + vSelectedDoc + "  for  these  " + @ResultSetTotal() + "  retrieved
records?","( If not,  cancel  this  task,  press F7 for a  new  search,"," and  retrieve  the  records
you  DO  want  to  print. )")
Then
{

    vGoPrint  =  "Yes"

    n = 1
    While  F[n]  <>  ""
        {
        vHeader  =  vHeader + F[n]     + "^"
        n = n + 1
        }

    vFileHandle  =  fileOpen(vDataPath)
    fileSeek(vFileHandle,   0)
    fileWriteLn(vFileHandle,    vHeader)

    For  n  =  1 to @ResultSetTotal()

    ResultSetCurrentPosition(n)
```

Program concludes on next page

```
    While F[i] <> “”
        {
        vData = vData + @(F[i]) + “^”
        i = i + 1
        }

    FileWriteLn(vFileHandle,    vData)

    @Msg(“Processed Record “ + n + “ of “ + @ResultSetTotal())
    vData = “”
    i = 1

    Next

    fileClose(vFileHandle)

    // Rewind to record where it all started
    ResultSetCurrentPosition(vStartingRec)

    }
}
Else
@Msgbox(“Printing retrieved records can only be done in Update mode.”,“”,“”)

END  SUBROUTINE

// BEGIN  MAIN  PROGRAM

// Build documents menu

vSelectedDoc  =  @PopupMenu(
“5160 Avery Labels.doc;” +
“Prospect Letter.doc;”    +
“New Client Letter.doc;” +
“Order Confirmation.doc;” , “SELECT MERGE DOCUMENT” )

// Build which record(s) menu

vOneOrAll = @PopupMenu(
“THIS Record Only;” +
“ALL Retrieved Records;” , “SELECT RECORD(S) TO MERGE PRINT” )

If vSelectedDoc <> “” and vOneOrAll <> “” Then
{

    // Delete existing data file. A new one will be generated.
    If FileExists(vDocsPath + vSelectedDoc) Then
    {
    fileDelete(vDataPath)

    // Fields available for merging
    F[1] = “First”;         F[2] = “Last”;      F[3] = “Company”
    F[4] = “Add1” ;         F[5] = “Add2”;      F[6] = “City”
    F[7] = “State”;         F[8] = “Zip” ;      F[9] = “Phone”
    F[10]= “Email”;             F[11]= “Position”; F[12]= “Live_Prospect”
    F[13]= “Date_Entered”; F[14]= “Hobbies”;   F[15]= “Currency”

    If vOneOrAll = “THIS Record Only” Then
        PrintCurrentRecordOnly()
    Else
        PrintAllRetrievedRecords()

    If vGoPrint = “Yes” Then
    {
        // Start Word via Word.bat and pass doc filename.
        n = @Shell(vWordPath + “ “ + @Chr(34) + vDocsPath + vselectedDoc + @Chr(34))
        If n = 1 Then
        @Msgbox(vDocsPath + vSelectedDoc,”word.bat not found or something else wrong.”,“”)
    }
    }
    Else
        @Msgbox(vDocsPath + vSelectedDoc + “ doesn’t exist.”,“”,”Aw shucks!”)
}
```

@Help

EDITED BY WILLIAM HALPERN

Converting Q&A Data to Other Programs

I have Q&A 3.0 for DOS. I would like to export my contacts and letters to Microsoft Excel. My original thought was to import my data into ACT! for Windows and then export from there to Excel. However, ACT! will import only Q&A 4.0 or higher databases. My question is this: Should I or can I go from Q&A 3.0 for DOS to Q&A 4.0 for Windows? (Will Q&A 4.0 for Windows convert my Q&A 3.0 databases?). If so, can I then do what I've described?

Michael

I am taking my time on this question because I have found that there are many people out there who just don't understand just what a database is or what it equates to in other software applications. There is always the confusion about a DOS-program database vs. a Windows-program database. Many people think that a spreadsheet is the same as a database, others totally misunderstand the concept of an integrated word processor in a database manager like Q&A. On the eve of the release of Sesame, I can't think of a better time to address these questions.

A database consists of individual records that store information about a specific topic (Sales, Stocks, Inventory, Customers, Friends, Music, Books, etc.). Each record has all of the information about one of the items in the topic (*Tom Sawyer* is a record in a Books database). A spreadsheet is similar to a database in some ways in that you can set it up so that the spreadsheet is the topic, each row is a record, and each column represents the fields in the records. So, Excel should be like Q&A or Sesame, right? No. The major difference is that a database is designed to let you easily retrieve individual records or a specific group of records, while a spreadsheet is designed to allow you to view many (all) "records" at a glance. However, the similarities in data structure of the two kinds of files usually do allow the data in databases to be easily converted to a spreadsheet format, and vice versa.

There is also a very special category of databases on the market that I like to call special purpose databases. ACT! is among these. It is designed to do only one thing — and to do it very well — manage sales- or customer-related *contacts*. However, it is not designed to store and track things like sales, stocks, inventory, books, or the like. Therefore, when you read that ACT! can import Q&A databases, it only makes sense if those databases involve *contact* (people) information. Any other type of database is not suitable for use in ACT!.

You can export your database data to a Standard ACSII file with a .txt filename extension. Such a file can easily be imported into Excel (or ACT!). Then you will have all of your data in a single table view that you can sort. Your programming and reports will be gone, as will your ability to find and isolate a specific set of records or even a single record with multiple retrieval parameters.

Unless your current database is so simple that you don't need such capabilities, converting to a spreadsheet seems to me to be an exercise in future futility. Your data will be "in Windows," but what will you be able to do with it? Your Q&A for DOS data can be exported/imported to just about any Windows program, including all of the Microsoft Office products. The O/S for which the program was designed is irrelevant when it comes to the raw data. Data from a DOS program is no different from that of a Windows program.

If you need to manipulate and easily find specific information in your data, you need to be in a database, be it Q&A or Sesame or even one of those others that are out there. Use the correct tools for the job!

You cannot export your Q&A "letters" to Excel under any conditions. Excel is a spreadsheet. Your letters can be exported to "Document ASCII" and then be read in by Microsoft Word. If they were designed in Q&A as merge documents and contain merge fields, they will be essentially useless in any program outside of Q&A and will need to be recreated. Symantec added a word processor to Q&A for one reason only — to give you the ability to easily create simple documents that would "merge" the information from multiple records in your database. This means that you only have to type a letter once and can then send it, filled with the information from individual database records, to many

---

people. These merge letters are not exportable to any other program as merge documents.

Reports From Multiple Databases

I've been trying to accomplish something lately that the manual doesn't help with. It just seems too complicated or something. I have two databases, one a client database and another an invoicing database. I sometimes need a list or report showing how much we've invoiced certain clients. Not a problem, I did the programming shown below in the client database, and the report I designed using the resulting information works great!

```
>#401:#401=@XLOOKUP("AIA-INV.DTF",#402,"FILE
NO.","F0003")
```

The problem is that if there is more than one invoice for that client, the computer doesn't pick it up. I know that it can be done and I've found info on that in the manual. But I can't seem to make it work. Most of the fields needed in the report are from the client database. *F0003* is the field in the invoicing database that holds the amount of that one invoice. *#401* is the field in the client database that I want to hold the data from the invoicing database. *#402* is the field in the client database that I want it to match on (the client file number). The field for the file number is named *File No.* in the invoicing database and *#402* in the client database. Can you give me any help with this? It would be greatly appreciated!

Connie

This is not a case of "the computer doesn't pick it up." You cannot do an XLookup and expect it to find more than one record. XLookups search for a unique value and stop at the first record they find. However, there are a few ways to do what you want.

1. Prior to running that report, you could run a "Post" from the Invoice database to *add* ALL of the invoice values for a customer to a single field in the customer record. The only issue here is that you would either have to clear the customer record field first (so you get a good total) or post only records later than the last date you posted. Either way would work. With Q&A 5.0's XPost, you could even do the posting in real time each time you enter an invoice.

2. Another way would be to do a report from the invoice database where you get the customer data by using derived columns and looking into the customer database. This is doable but is pretty intensive to design. It may also be time consuming to run depending on the size and location of the databases.

3. You could also create an interim database, combining customer and invoice data by utilizing mass update, export, and import routines. This way, the report

comes from a single database. Again, much more difficult to initially set up.

While methods 2 and 3 would also give you the ability to run aging reports and to print invoice totals by date, I would vote for #1 to get your simple report. Post your invoice totals to the customer record and then do the report. (Forget about the XLookup in the field — that will not work at all).

Is Lesspace Working?

I have a question regarding a new windows XP professional PC and Q&A 5.0. The new PC's file system is NTFS. How do you configure Q&A? I don't think either the 2gbfix or Lesspace will work. When loading either 2gbfix or Lesspace from a command line prompt and running a DIR command on the C: drive of the windows XP box, the full hard drive size is shown. In an attempt to determine if Lesspace is actually working, I loaded Lesspace from the command line and it appears that by doing a DIR command that it still recognizes the entire hard drive, not a "smaller" drive. We run Q&A from the C: drive (Windows XP box), with the data residing on a Novell Server (F: drive). We can limit the volume size within Novell but do we need to limit the size of the C: drive for Q&A to operate correctly?

Andy

Let me try and set you straight on this. Lesspace is not a utility that will work on internal Windows commands. It only works with resident DOS programs like Q&A. If you want to see if it is working, press Ctrl-F3 at the Q&A main menu screen and look at the disk space being reported there (line 10). As for the "2gbfix" you mention, I don't know what that is.

NTFS has nothing to do with it, neither does a Novell formatted drive. Lesspace works on *all* drives that Q&A sees, be they local and/or network.

One word of advice, if your server is Novell 4 or later, be very careful with your Q&A databases. We have seen many crashes of Q&A databases when stored on newer Novell drives. We have found it best to host the databases on a drive that is *not* part of the server (non-Novell volume). It has to do with the write-behind caching and can be fixed by a real expert in Novell (not me) with the required know-how. My advice, in any case, is to keep your Q&A data away from Novell drives and never, ever run Recover on a Q&A database across a network to a Novell drive.

Q&A Slow Only Over Network

We have run Q&A 4.0 for DOS on a single machine for many years with no problems. We just bought and installed a Network pack to allow three users to access

Avery Labels.doc, *Prospect Letter.doc*, *New Client Letter.doc* and *Order Confirmation.doc* — and ending with:

"SELECT MERGE DOCUMENT")

Follow this simple structure to add and remove the documents you want to be available on your popup menu. These must be the actual filenames of your Word merge documents.

Important: To start with, you'll need to add at least one *existing* Word document to the list. Any old document will do. Start Word, type a line of text in a new document and save it to *c:\sesame\docs* with the filename *just for testing.doc*.

Next, you'll find another @PopupMenu() menu that includes these two selections:

THIS Record Only
ALL Retrieved Records

Everything that follows in the program depends on a Word document having been selected and a choice having been made as to which records to include:

If vSelectedDoc <> "" and vOneOrAll <> "" Then ...

If you click outside either menu or press Esc, the menu will disappear and the program will end.

Otherwise, the program will know which document you've selected along with which records you want to merge print. In this case, it will do two things:

• Delete any existing *c:\sesame\docs\customer.txt* file (the *data source* file for Word). A *new* data source file will be generated by the subroutines to follow.

• Populate the *F* (for *Fields*) array. The table-like structure for populating (filling) the *F* array gives you an easy way to specify which form elements (fields) whose data you want to make available as merge fields in your Word merge documents.

In the sample program, 15 elements are specified, from **First** (first name) as *F[1]* to **Currency** (a money field) as *F[15]*. Replace these element names with your own form's element names. If you have fewer than 15 elements, delete the remaining ones. If you have more than 25 elements to specify, be sure to increase the *F* array's declaration to however many you have, and number them accordingly as you populate the array. Declare the array for the exact number of fields you intend to include in your *data source* file for Word, and populate the array elements with that exact same number of values.

Note: You can pass as many data values to Word as you need. Actually, you're better off passing too many than too few. Why? Well, first of all because you don't have to insert merge fields for all of them in any of your Word documents. Secondly, if you decide later to add more values to the *data source* file, you will have to update your Word documents to reflect the structural change in that file, which is added work.

There are two subroutines in the program — one that handles printing just the current record (*PrintCurrentRecordOnly*), and one that handles printing all the retrieved records (*PrintAllRetrievedRecords*). The program branches off to one or the other depending your earlier menu selection now stored in the *vOneOrAll* variable.

PrintCurrentRecordOnly begins by prompting you to confirm that you want to merge print just the current record with the selected document.

It then loops through the *F* array, creating a delimited string value for the *data source* file's header line (*vHeader*) and a matching delimited string for its data line (*vData*). The header line in the *data source* file provides Word with the *field names*. The data line specifies the *data values* (merge values) that correspond to those field names.

Having a header line in the *data source* file enables you to insert your merge fields in the Word document by field name. Without it, Word won't be able to provide you with a sensible *Insert Merge Field* list without your having to do extra work.

Note: The *vData* variable is filled using the *@(indirect field reference)* function (@ is the abbreviation for the *@Field* function). As the loop processes the field names in the *F* array, when it comes to F[3], for example, the value of F[3] is "Company". Accordingly, *@(F[3])* retrieves the value from the **Company** field. Details on variables, arrays, functions and other programming commands can be found in the *Sesame Programming Guide*.

The program continues by writing the header and data lines to the *customer.txt* data source file.

Now that the data source file has been updated for the current record, the program continues by launching the word.bat batch file and passing it the selected document filename to merge the data with.

The *PrintAllRetrievedRecords* subroutine works similarly, but has more to do. It runs only in *Update* mode, when you've retrieved one or more records. It prompts you to confirm that you want to print the selected merge document for *x* number of records. The *x* is courtesy of the @ResultSetTotal() function — it returns

the number of records in the result or answer set. This final prompt helps ensure that you don't accidentally try to merge print all the records in the database when you intended to print just one or a selected group.

After you've confirmed that you want to include all the records in the answer set, the subroutine prints the header line to the customer.txt file, then moves to the first record in the answer set and adds its data to the same file. It then continues, looping through each subsequent record in the answer set, adding its data to the file. This loop continues while the loop counter n is less than or equal to @ResultSetTotal().

While the subroutine is doing its work, the status or message line at the bottom of your Sesame window shows a running count of the number of records processed along with the total number to be processed.

When the subroutine is finished, before the batch file that launches Word is run, it re-displays the record you were on when you started the process. Assuming that everything has been set up correctly, you'll find yourself in Word, ready to merge print.

In Word, if you're merge printing multiple records, clicking *File / Print* will print only the first one. To merge print all the records in the *data source* file, click the small *Merge to Printer* button on the Merge toolbar. (See Figure 1 below.)

It's time to create your Word merge documents, but before doing so, you need to run your new Sesame program at least once to generate the *data source* file that Word will need to merge with your selected document.

Before leaving the Program Editor, click the **Test** button to test the program. If you receive any error messages, fix the offending code now. When the program tests error-free, do the following:

1. Click *File / Save All to File*. Type in a path and filename to save your programming to, such as *c:\sesame\customer.pgm*.

2. Click *File / Close* to close the Program Editor.

3. Back in Designer, double click *Save Layout Design* on the **Commands** menu.

4. Double-Click *Close Design Tab*.

5. Still in Designer, open the **Application** menu and click

Preview Application.

6. Select *Search/Update*, then select the form you just redesigned.

7. Press F10 to display a filled out record.

8. Click your new *Merge Print* button.

9. Select the *just for testing.doc* you created earlier.

Word should start, the selected document should open, and your *data source* file (*customer.txt*) should have been created.

Now would be a good time to confirm that *customer.txt* is indeed in the *c:\sesame\docs* folder. Open the file and verify that the field names in the header line correspond to the data values on the data line. The file should be structured like this:

```
First^Last^Company^Add1^Add2^City^State^Zip ...
Bill^Smith^Jiffy Products^123 Main St.^^Orange^CA^92455 ...
```

Exit Word and return to Sesame. Take the necessary steps (as covered in the *Sesame Users Guide*) to save your work before exiting. If you simply exit Designer (after saving your layout) without Reconciling your changes with the application (customer.db in this case), you'll still have the Designer (.DSR) file you can return to later.

Step 3 — Design Your Word Merge Documents

Now that you have a *data source* file (customer.txt), you can design your Word documents to use it, then insert your merge fields.

> **Note:** Give yourself a head start. If you have merge documents created in Q&A for DOS, you can open them in Q&A's word processor and save them to ASCII format (Ctrl-F8) to a different filename. (Or you can export them to *Document ASCII*.) This strips out all assigned fonts, but leaves your text, paragraphing and merge field tags intact. You can open the resulting plain text files in Word and take it from there. You won't have to retype all your text, and you'll see right where your merge fields need to go.

With one of your Word documents open, follow

Figure 1. In Word, Click the *Merge to Printer* button to merge print multiple records.

these steps. They might differ somewhat depending on which version of Word you're using:

1. On the menubar, select *Tools / Mail Merge* to open the Mail Merge Helper.

2. Under **1. Main Document** select *Create / Form Letters / Active Window* in that order.

3. Under **2. Data Source**, select *Get Data / Open Data Source*.

4. In the **Data Source** dialog, open the *Files of Type* dropdown and select *Text Files [.txt]*.

5. Navigate to where *customer.txt* is stored (the *c:\sesame\docs* folder) and open it. Word will ask you to confirm the data source as a text file. Click OK.

   At this point, Word may prompt you to specify the *field delimiter* (select the "^" character from the dropdown) and the *record delimiter*, which should correctly be {Enter} by default. Word might also want to know if the first row of the data file contains *field names*. It does.

6. If everything is as it should be, Word will inform you that it "...found no merge fields in your main document..." (Amazing!) Rub your hands together, agree that you want to add some, and an **Insert Merge Fields** menu (containing all your field names) will appear at the top left of the window on the now-visible **Merge** toolbar.

7. Add your merge fields by first placing the cursor where you want to insert a field, then selecting the field from the **Insert Merge Fields** menu.

The procedure for designing a *mailing label* template is a bit different, but no more difficult. Using the Mail Merge Helper, you simply select *Tools / Mail Merge / Main Document / Create / Mailing Labels*. Then, after you've specified your data source, Word will display a *Label Options* dialog where you select the type or brand of label sheet to use. It then provides a small *Create Labels* window with an *Insert Merge Field* button. You insert your merge fields just as you would in a regular merge document.

The procedure for designing a merge *envelope* template is similar to designing a label template.

If you've inserted any fields that contain numbers, money or date values, you may need to apply special formatting to them.

Special data format options
When you have a Word merge document open, you'll see a button on the toolbar that has a little "ABC" on it

(*View Merged Data*). If you look hard, you can just see it in Figure 1. You click on this little button to toggle between *view field names* mode and *view data* mode. In *view data* mode, you can see the actual merged values.

> **Note:** When you save a merge document for actual use, always save it with the *data displayed* (rather than with the merge field names displayed) because this is the view you want when you open the document later for printing. Seeing the actual data when opening the document is more reassuring than seeing the field names.

Merged dates will likely appear in Sesame's internal *YYYY/MM/DD* format, and number and money values may need some special formatting.

Word supports *formatting switches* you can use to apply special formats to any merge field that needs them. In Word Help, search on *Format Merged Data* and you'll get information on how to do this.

To format a merge field, press Alt-F9 to expose the hidden merge field codes. If you have a date value merge field such as *Order_Date*, add a date-formatting switch to the merge field code as shown below. (The switch itself is shown in boldface.):

{MERGEFIELD Order_Date **TIME \a "MMMM d, yyyy"**}

This will display the merged date in the **January 25, 2004** format instead of the unformatted **2004/01/25**.

For a money field named *Amount*, you can add the following formatting switch (shown in boldface) to give you a dollar sign, a comma between the thousands, and two decimal places):

{MERGEFIELD Amount **\#$#,#0.00**}

You can use variations on these switches to achieve any format you might need. There are even formatting switches for all uppercase or lowercase. Don't get all mired down trying to format your Sesame data to look "just right" in your merge documents. Word itself has all the tools you need for this.

When you've completed and saved your new merge document (be sure to save it with the *merge data displayed*), you can then copy it by saving it to a different filename. This saves time if you're creating a series of merge documents that are similar in content, have mostly the same merge fields, and use the same *data source* file.

> **Note:** When you've finished designing each document, save it to the *c:\sesame\docs* folder using a compact but descriptive filename such as *Sales Order.doc*, *New Client Letter.doc*, *Follow Up Letter 1.doc* or the like. These are the

filenames you will add to your program for use on the document selection menu in your Sesame form, so you want them to make sense. Just don't get carried away with exceedingly lengthy file names.

Finish up your other merge documents in the same fashion.

Test Your Work

Back in Sesame, test your new document printing application. With a filled out record displayed, click your **Merge Print** button. Select a document or label sheet from the menu, then tell Sesame whether you want to merge print just the displayed record or all the records you've retrieved. Word should start and the document should open with the merged data displayed. Click the *Merge to Printer* button to print out the document(s).

If your computer supports these options, you can optionally, for each merge record, click *File / Send To* and chose either *EMail Recipient (as Attachment)* or *Fax Recipient*. Consult your Microsoft Word documentation for information on sending *multiple* emails or faxes.

Note: Before merge printing multiple records, it's always a good idea to sort them in Sesame by some meaningful criteria. This way, if something goes wrong with the printer in the middle of the print job, you can always re-run the job later and start with the first record that didn't print earlier.

To return to your Sesame record, simply click *File / Exit*.

Conclusion

So there you have it, a way to use Word to print merge documents, labels or envelopes on demand from any database record, in a way that doesn't require any expert know-how on the part of the user doing the printing.

If you wanted to, you could optionally add a few lines of code to the program to stamp the record, say, with the date and the name of the merge document that was printed, giving you a kind of audit trail of who got what and when.

Tom Marcellus is editor of *The Quick Answer.*

@Help

our databases at the same time. Now, just saving a record on one of the additional machines can take up to two minutes while a "working" message displays at the bottom of the screen. We have a peer-to-peer network with the server being a Windows 95 PC that used to be the single station. The other machines are Windows 98. We are running Lesspace and have set the Sharing Mode on the databases to *Allow*. **When we set them back to** *Automatic*, **the databases won't support multiple accesses but they are again fast. How can we fix this problem?**

Dominic

You had me at a loss, so I asked you to send me the databases so that I could analyze the problem. I was able to duplicate everything you complained about. And, finding what I did, I was not a bit surprised. You had broken almost every rule in your database designs.

Your database had 19,000 records. No problem. It had *14* speedy fields. Big problem. Most of the speedy fields had no data at all in them. Bigger problem. The database had the Intelligent Assistant taught. Along with the other issues, another big problem.

The basics. Keep your speedy fields to a minimum. *Never* exceed five per database. Make a field speedy only when absolutely necessary — as a key to an XLookup or a field on which you frequently search. But *never* use a speedy field where more than sixty percent of the data in that field will not be unique values. This includes empty fields. To Q&A, blank fields contain duplicate values. Also, do not teach the Intelligent Assistant (IA) about a database if you are not using the IA. It substantially increases the size and load on the index file.

We removed twelve speedy fields from your database. This involved clearing the Speed Up Spec, resaving it, then replacing the two needed speedies. We removed the IA by copying the design to a new database (without the IA), then copying all of the records to it. Finally, we set the new database to *Allow* sharing. We put this on your peer-to-peer network and it was instantaneous in saving records from all stations.

Bill Halpern is co-owner of Professional Computer Technology Associates (PCTA) in Newtown, Pennsylvania and is President and CEO of Lantica Software, LLC. 215-598-8440. bill@pcta-usa.com, www.pcta-usa.com.

Need a Function that Sesame Doesn't Have? Design It Yourself!

In addition to the host of built-in programming functions and commands in Sesame, Sesame supports what are called *User-Defined-Procedures*. These include *subroutines* and *user-defined functions*. The difference between subroutines and functions is that a function always returns a value which can be used, for example, in expressions. Here's an example of a user-defined function:

```
Function calcVAT( amount as double ) as double
Return amount * .16
End  Function

var a as double = 100, b as double
b = a + calcVAT( a )
WriteLn( "Amount: ", a, " VAT:", b )
```

The function declaration is a bit different from a subroutine declaration. It uses the word "function" and "end function" instead of "subroutine" and "end subroutine" in its declaration. In addition to that it has a type of its own (in this case, "double"). That type is the type of the return value.

In its statement part, a function uses the word "return" to signal that the value followed by this word is the return value of the function — what gets returned to the program that called it. The function shown above accepts a double and returns a double. In the program, the function is called in line 6. The formal parameter "amount" gets assigned the value of the actual parameter ("a"), the function returns a value (16.0) and this value is used in the expression in line 7 as the value of the function call. B gets assigned the value of 116.

User-defined functions (called *UDFs*) can be useful for converting between data types. Take date values. Sesame stores them internally in a YYYY/MM/DD format (just like Q&A). Regardless of how a date value might be *formatted to display* on a form, if you want to use it in a string expression and need to make it look like a normal date, you have to deal with the YYYY/MM/DD format. There's no built-in function to convert date values to a sensible-looking string format.

If you allow the raw internal date format to come through in a string expression like this, where **OrderDate** is a date type form element (field):

```
"The Order Date was " + OrderDate + "."
```

You'll get this:

```
The Order Date was 2004/02/12.
```

You can "hand-parse" the date value to use it in a string expression:

```
"The Order Date was " + @Month$(OrderDate)
+ " " + @DOM(OrderDate) + ", "
+ @Year(OrderDate) + "."
```

Which will get you the desired format:

```
The Order Date was March 15, 2004.
```

But if you need your program to perform a number of such conversions, you might find it worthwhile to create a UDF that converts date values *for* you.

The following sample program (next page) shows how a user-defined function named *cDate* (for *Convert Date*) does all the work. The contents of two date elements — **OrderDate** and **ShipDate** — are passed to *cDate* as date values, and emerge from it as string values, so they can be used as sensible-looking dates elsewhere in the program.

The first @Msgbox, which doesn't use *cDate*, produces this:
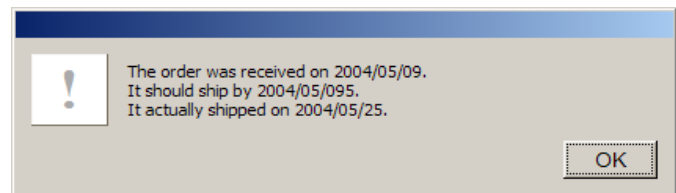
Using *cDate*, the second @Msgbox produces this (below):
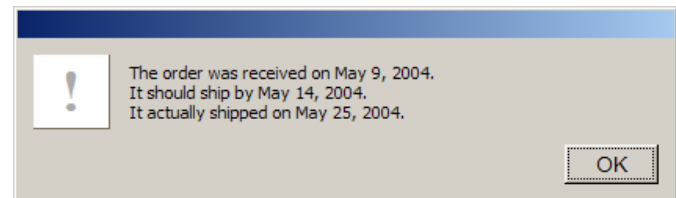
The program is shown on the next page.

As you can see, the *return* value of the function replaces each *call* to the function.

Instead of adding UDFs like this to every program that requires them, you can place them in the *Global Code* section of your form where they'll be available to all your programs executing in all your records.

SBasic requires user-defined subroutines and functions to occur in the program *ahead* of the statements that call them. So if you use any UDFs or subroutines in your programming, it's a good practice to put them at or near the top of your program.

Concludes next page

Quick Tip Using Dual Monitors

Windows is much more exciting when you have two monitors, so that when you start a new program, it doesn't obscure the program you may still need to use.

My computer was originally supplied with a Matrox G450 dual head graphics card, but I have only lately realised what it was for.

For instance, you can have Sesame running on one monitor and Sesame Designer on the other one. Or you can pop up Q&A in a window on one screen to check an address while you're working in Word on another. I like to leave Lotus Organiser on the right hand monitor while using the left hand screen as my main one.

Large hi-resolution screens are nice, but separate monitors allow something else to happen without disturbing what you're doing. It's like putting activities into boxes, and these activities can be "trained" to pop up on the "other" monitor, out of the way but easily accessible.

If I were starting out, I would get an ATI Radeon 9200 dual head card because with its accompanying software you can save different Windows configurations — that is, preconfigured arrangements of opened programmes — for different applications or users. You can also, with a mouse click, stretch a spreadsheet across both screens to get a cinerama-like view with twice the columns.

I now have identical 17" TFT screens. For a low cost taster, you can start with just adding a new single graphics card, which Windows will see and install as additional hardware, along with a second-hand 15" CRT. All the settings will be there if you right-click on the empty desktop and go to Properties > Settings.

The photo shows how my two monitors sit on my desk and take up very little real estate. Q&A 5.0 is running on the left, with Lotus Organiser on the right. This is luxury.

Roger Skidmore, Isle of Wight, rogerskid@supanet.com

UDFs cont'd from page 12

```
/*  cDate  function  converts  date  values
    in  fields  or  variables  from  the  internal
    'YYYY/MM/DD'  format  to  easily  readable
    'Month  Day,  Year'  string  format */

FUNCTION  cDate(theDate  as  Date)  as  String

var  StringDate = @Month$(theDate) + "  "
+ @DOM(theDate) + ",  " + @Year(theDate)

return  StringDate

END  FUNCTION
```

```
If  OrderDate <> ""  and  ShipDate <> ""  Then
{
// Without  Date  conversion  function

@Msgbox("The  order  was  received  on  " +
OrderDate + ".",
"It  should  ship  by  " + OrderDate + 5 + ".",
"It  actually  shipped  on  " + ShipDate + ".")

// With  Date  conversion  function

@Msgbox("The  order  was  received  on  " +
cDate(OrderDate)  +  ".",
"It  should  ship  by  " + cDate(OrderDate + 5) +
".",
"It  actually  shipped  on  " + cDate(ShipDate) +
".")
}
```

The Quick Answer online

Visit Our Web Site at
www.quickanswer.com

- Complete Back Issue Index 1990 to date
- Comprehensive Topic Index 1990 to date
- Info on Online Subscription Benefits

- Info on the Q&A successor product
- Q&A Tips and Ideas
- Searchable Solutions Database
- Dozens of Free Files to Download
- Full Database Applications
- Q&A Consultants Directory
- Links to Useful Web Sites & More!