November 1995

THE Quick Answer

The independent monthly guide to Q&A expertise

Ditto Your Data During Search/Updates

When you're adding new records to a database, you can press F5 in any field to "ditto" the field in the previous record to the new record. You didn't have this option in Search/Update mode until now.

By Jeff Nitka

DOS WIN

I seem to get all the clients who want Q&A's Ditto feature in Search/Update but can't have it because it works only in Add Data. I can't fault them, though, the option to ditto during Search/Update would make updating groups of related records much easier. As it is, when updating records, you have to manually type in any changes, even if the new value is the same in the record you just updated.

It's my experience that Q&A users often prefer to update records manually rather than run Mass Updates because they can see their updates as they enter them. This is where a ditto option in Search/Update would come in handy. I'll show you a technique that does for you during Search/Update what the F5 Ditto key does during Add Data—and provides a side benefit as well.

Laying the foundation

To start, you'll need to add three fields to your form. Make them invisible, if you like, and unreachable by placing them on a blank page and making them read-only. Name them Number, Ptr (for pointer), and Data. Format Number and Ptr for numbers, and Data for text. In the Navigation Spec, place a < *PgUp* command in each of them so control can't accidentally be passed to them. Finally, make Number a Speedy field at the Speed-up Spec.

To make the Search/Update ditto feature work, you'll create an on-record-exit program to execute whenever you leave a record other than by deleting it or pressing Esc. To make a program execute when exiting a record, press F8 in the Program Spec, then enter the appropriate field ID number on the On Record Exit ID line. The program you'll use depends on whether or not the @Number function is already being used in the database.

Continues on page 3

C O N T E N T S

MARBLE PUBLICATIONS

README.1ST

WE introduce two new features this month—*The Quick Answer*'s *Ad Data* Advertiser's supplement, and a **Jeff Nitka** column on Q&A programming techniques. First, about the advertiser's supplement.

During the September 15 Q&A 5.0 Master Seminar in Washington, D.C., more than one Q&A developer pinned your poor editor to the wall demanding to know why *The Quick Answer* doesn't offer an advertising service—a classified section, for example—for Q&A consultants, developers and trainers, and suppliers of specialty databases and compatible add-on products. Message received. The *Ad Data* advertiser's supplement will travel with *The Quick Answer* on a quarterly basis—the one in this issue tells you all about it. The deadline for the January supplement is December 15, so be sure to get your ads to us right away as space is limited and we expect a good number of comers.

This month's supplement plugs several of our own products. You can now order the *Video, Disk & Documentation Pack* from the spectacular Q&A 5.0 Master Seminar. You can also order the *Q&A Program Evaluator* (it helps you debug your programming), the *Q&A 5.0 QuickStart Upgrade Companion* utility, and the *QuickClick CalendarPlus* (a perpetual calendar database). The *Q&A Program Evaluator* is available for Q&A 4.0 and Q&A 5.0. The 5.0 edition, along with the *QuickClick Calendar Plus*, will ship as soon as Q&A 5.0 ships, which looks as though it might be this month.

Jeff's *Program Spec* column will be devoted to Q&A programming tips and traps. This month, he explains how Q&A can evaluate programming statements on a Yes/No basis, and offers ways you can take advantage of this logical decision-making power.

In a second article, Jeff shows you how to get the same Ditto (field copying) performance during Search/Update that you get by pressing F5 when adding a batch of new records. Jeff's ditto technique goes Q&A's one better—it works when you move backward in a stack of records as well as when you move forward.

Although a single XLookup statement can retrieve data from up to 23 external fields, what if you need to retrieve and display information from a range of records when there might be just one qualifying record, or there might be a dozen or none? I'll show you how to fetch and display date-sensitive reminder notes and invoice history information by record range. You can adapt the techniques to satisfy similar lookup requirements.

In a second article, I'll show you how to use a Lookup Table or external database to specify the selections that appear on your Q&A 5.0 @Userselect pick-lists. This way, you can make additions and deletions to these lists without having to reprogram the database.

Tom Marcellus
Editor

THE QuickAnswer

Volume 6, Issue 11

Editor Tom Marcellus
Publisher Michael Bell
Copy Editor Laurie Moloney

The Quick Answer (ISSN 1052-3820) is published monthly (12 times per year) by Marble Publications, Inc., 9717 Delamere Ct., Rockville, MD 20850. Cost of domestic subscriptions: 12 issues, $79; 24 issues, $142. Outside the U.S.: 12 issues, $99; 24 issues, $172. Single copy price: $10; outside the U.S., $12.50. All funds must be in U.S. currency. Back issues are available upon request, for the same price as a single copy.

Second-class postage pending at Rockville, MD. POSTMASTER: Send address changes to *The Quick Answer*, PO Box 9034, Gaithersburg, MD 20898-9034.

Copyright © 1995 by Marble Publications, Inc. All rights reserved. No part of this periodical may be used or reproduced in any fashion whatsoever (except in the case of brief quotations embodied in critical articles and reviews) without the prior written consent of Marble Publications, Inc.

Address editorial correspondence, @HELP questions, or requests for special permission to: Marble Publications, Inc., *The Quick Answer*, PO Box 9034, Gaithersburg, MD 20898-9034. Phone 800-780-5474 or 301-424-1658. Fax 301-424-1658. CompuServe 73370,1575. Prodigy NEPY97A.

For Q&A technical support, call Symantec: 503-465-8600.

Q&A is a trademark of Symantec Corp. Other brand and product names are trademarks or registered trademarks of their respective holders.

This publication is intended as a general guide. It covers a highly technical and complex subject and should not be used for making decisions concerning specific products or applications. This publication is sold as is, without warranty of any kind, either express or implied, respecting the contents of this publication, including but not limited to implied warranties for the publication, quality, performance, merchantability, or fitness for any particular purpose. Marble Publications, Inc., shall not be liable to the purchaser or any other person or entity with respect to any liability, loss, or damage caused or alleged to be caused directly or indirectly by this publication. Articles published in *The Quick Answer* do not necessarily reflect the viewpoint of Marble Publications, Inc.

MARBLE PUBLICATIONS

Ditto Your Data

Continued from page 1

If the @Number function isn't already spoken for, use this program:

```
#10: Number = @Number
```

If the @Number function is already being used in the database, use this program instead:

```
#10: Number = @XLR(@Fn, 99999, "Number", "Number") + 1
```

You can assign a field ID other than 10, as long as it matches the field ID in the On Record Exit line. Either program will assign the highest number in the database to the Number field in the last viewed record.

Programming the new fields

When the cursor enters a field, your program will assign that field's ID number to Ptr. Here are examples for field #1 and field #5:

```
< #1: Ptr = 1

< #5: Ptr = 5
```

Add statements like these to any field you might want to ditto. Because Ptr will always contain the ID of the current field, your program will be able to determine the field in the previously viewed record to copy to the same field in the current record. The feat is accomplished with a program like this:

```
Data = @XLR(@Fn, 99999, "Number", "x#" + @Str(Ptr))
```

This @XLR command finds the record with the highest Number value—the previously viewed record—and returns to the Data field in the current record the value in the field referenced by Ptr. The last parameter in the @XLR command is a string concatenation of "x#" and the numeric value in Ptr, so Q&A will return the value from the field in the previous record that matches the current field in the current record.

Because the ditto feature is an option, you'll have to decide which of two techniques to use. If you don't already press F8 to execute Calc statements, you can use the first technique. (If the Calc Mode for the database is set to Manual, Calc statements execute when you press F8. If the Calc Mode is set to Automatic, they're executed automatically whenever any field is updated. To make the Calc Mode Automatic, press Shift-F8 when viewing a record, and choose Automatic). If you're not using F8 to execute Calc statements, then the first variation lets you use F8 as the ditto key for the calculation statement shown in Listing 1.

Listing 1. Use this program if the F8 key isn't already reserved for Calc statements.

```
#20:

If @Update then { Data =
@XLR(@Fn,99999,"Number","x#" + @Str(Ptr));

If Data <> "" then { @Msg("Ditto performed: " +
@Chr(34) + @(Ptr) + @Chr(34) + " replaced with " +
@Chr(34) + Data + @Chr(34)); @(Ptr) = Data }

Else @Msg("No value in previously observed record."
          + " No Ditto performed." ); }
```

The @Msg message recaps the ditto. This way, you know what was changed, so you can reenter the original value if the dittoed value proves undesirable.

If your F8 key is already being used to execute Calc statements, you'll need to modify your program. Listing 2 shows the modified program.

Listing 2. Use this program if the F8 key is already reserved for Calc statements.

```
#20:

If @Update then if @Lt(@(Ptr),2) = "—" then
{ Data = @XLR(@Fn, 99999, "Number", "x#" + @Str(Ptr));

If Data <> "" then { @Msg("Ditto performed: " + @Chr(34)
+ @XLu(@Fn, Number, "Number", "x#" + @Str(Ptr)) +
@Chr(34)
+ " replaced with " + @Chr(34) + Data + @Chr(34) );
@(Ptr) = Data; }

Else { @(Ptr) =
@XLu(@Fn, Number, "Number", "x#" + @Str(Ptr));
@Msg("No value in previously observed " +
     "record. No Ditto performed."); } }
```

With this second approach, you perform the ditto by typing two dashes ("—") in the pertinent field. If the Calc Mode is Automatic, the program will update the field with the value from the same field in the previously viewed record. If the Calc Mode is Manual, pressing F8 will get the same result. If the same field in the previous record is blank, the Else portion of the program restores the field's original value and informs you that it has done so.

The two approaches in use

Following are a few examples of the two techniques in action. First, let's look at the F8 Calc technique (the one without the dashes).

Suppose the following record is the first one that comes up during a Search/Update. The Ptr field contains "1" because you're in the first field, and the Number field contains the number assigned to this record when it was last viewed:

```
  Name: John Jones
Weight: 165 lbs
Height: 6' 3"

Number: 620
  Data:
   Ptr: 1
```

When you press F10 to display the next record, the Number field in the earlier John Jones record is made the highest Number in the database, courtesy of your on-record-exit program. Suppose the next record that comes up is this one:

```
  Name: Terry Smith
Weight: 115 lbs
Height: 5' 3"

Number: 535
  Data:
   Ptr: 1
```

When the cursor enters the first field of this Terry Smith record, Ptr is set to 1. The Number field contains the number assigned when the record was last viewed. If you press F8 to perform the ditto, "Terry Smith" is replaced by "John Jones," and a message appears to confirm it. Here's what the Terry Smith record now looks like:

```
  Name: John Jones
Weight: 115 lbs
Height: 5' 3"

Number: 535
  Data: John Jones
   Ptr: 1

Ditto performed: "Terry Smith" replaced with "John
Jones"
```

Now, here's an example of the second technique—the one that requires the two dashes. Suppose the following is the first record displayed during a Search/Update:

```
  Name: Jeff Nitka
Weight: 170 lbs
Height: 5' 11.5"

Number: 455
  Data:
   Ptr: 1
```

You press F10 to display the next record, and Jeff Nitka's Number is increased to the highest Number in the database. Here's the next record that comes up after Jeff's:

```
  Name: Greg Casey
Weight: 185 lbs
Height: 5' 10"

Number: 419
  Data:
   Ptr: 1
```

You move to the Height field—the third field and the one you want to ditto. The program sets Ptr to "3," and you type two dashes, like this:

```
  Name: Greg Casey
Weight: 185 lbs
Height: — 10"

Number: 419
  Data:
   Ptr: 3
```

You press Tab or Enter to exit the field (or press F8) and the ditto is executed. The "5′ 11.5″ from the Height field of the Jeff Nitka record replaces the "5′ 10″ in the Greg Casey record, and the confirming message appears. The record now looks like this:

```
  Name: Greg Casey
Weight: 185 lbs
Height: 5' 11.5"

Number: 419
  Data: 5' 11.5"
   Ptr: 3

Ditto performed: "5' 10"" replaced with "5' 11.5""
```

The F5 Ditto Key Works in Q&A for Windows, Too

You don't have to follow the instructions in the *Q&A for Windows Users' Manual and Reference* to ditto fields when adding new records. Just press F5 in each field you want to ditto. The manual tells you to select *From Last Record* from the Paste Special submenu, or press Ctrl-Shift-L to ditto a field. But if you're going to press Ctrl-Shift-L, why not simply press F5?

Like Q&A for DOS, field dittoing in Q&A for Windows doesn't work when updating records, but the programming techniques in this article should work equally well in Q&A for Windows.—*T.M.*

Conclusion

If it occurred to you that this technique ought to work when you move backward in a stack of records (by pressing F9) as well as forward (when you press F10), you're right. That's a ditto capability not available in Add Data. Moreover, using @Msg messages ensures that any inadvertent dittos will be difficult to miss.

Because the second approach requires the two dashes, it's the safest way to ditto. The first approach streamlines the process a bit, but it's more restrictive since it co-opts the F8 key.

There's a caveat to either approach: Q&A doesn't execute on-field-entry programming when you back into a field (when you press Left arrow, Up arrow, or Shift-Tab to move to a previous field). In such a case, the Ptr field will be left unchanged even though the cursor has moved to another field. The solution is to back up to a field that precedes the one you want to update, then move to the target field. To ensure the Ptr field will be properly updated if you should you move to the first field on the form, you can place a Navigation Spec statement such as > *Goto field ID* or > *Chome* in the last accessible field on the form.

Jeff Nitka recently graduated from Rutgers University with a Bachelor of Science degree in mathematics and computer science. He develops Q&A applications part-time for Epoch Software, 908-874-3989. Jeff is the author of the Q&A Program Evaluator, a program debugging utility available from Marble Publications Inc., publisher of *The Quick Answer.*

How to Lookup and Display Information from Multiple Records

Using XLookup to retrieve data is a no-brainer, as long as everything you need is in one record. But how do you retrieve *and* display data from a *range* of records?

By Tom Marcellus

DOS WIN

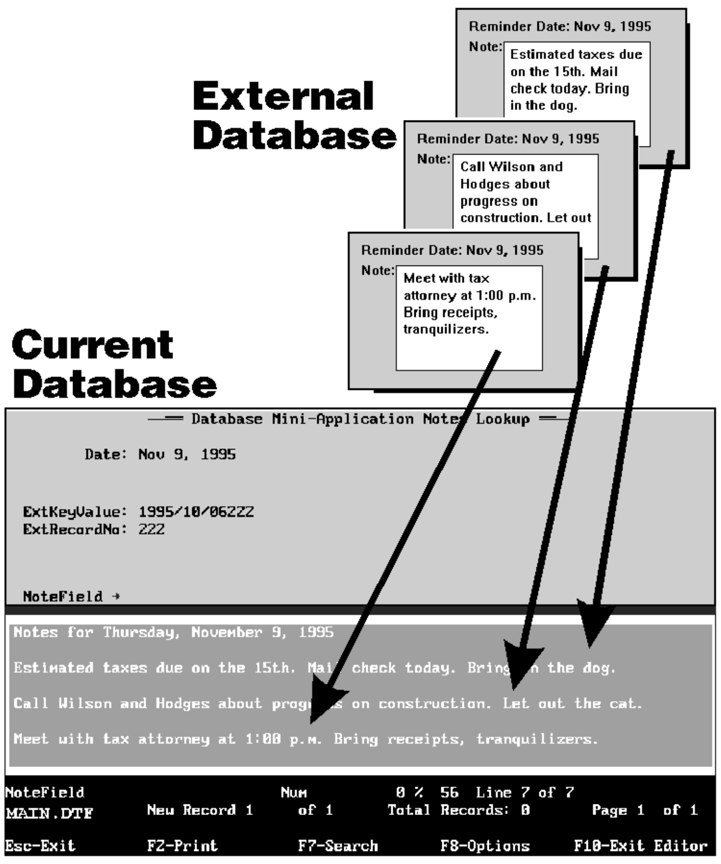
WITH a single XLookup command you can retrieve information from a dozen or more fields in the same external record. But what if you need to retrieve and display information from a *range* of external records? Assuming the records are incrementally numbered, you can use the XLookupR range command to find the record with the highest record number. But how do you find the next highest record after that, then the next one, and so on until the data from all the records in the range have been retrieved? There could be just one qualifying record in the external database, or there could be 2, 7, or 20 or more. You don't know until the information has been retrieved. And as if this weren't enough, how do you *display* all that information so it's readable?

I'll show you how to accomplish the task with two examples: retrieving and displaying date-sensitive reminder notes (see Figure 1), and retrieving and displaying invoice history information. You can adapt the techniques to meet other, similar requirements, and to make any database retrieve information from multiple records in most any external database. Though my examples are for Q&A for DOS 4.0 and 5.0, the approach works equally well in Q&A for Windows.

In the first scenario, the databases are NOTES.DTF and MAIN.DTF. NOTES, the external database, contains reminder notes, things to do, appointments to keep, and so forth. MAIN.DTF can be any database from which you might want to lookup and display those notes.

In this scenario, there could be none or any number of reminder notes in the NOTES database for a particular date. You might have 11 things to do on November 9, 1995, and you've added 11 records to NOTES, all with a reminder date of November 9, 1995.

Without leaving the MAIN database, you'll be able to display your reminder notes for any date you choose, simply by typing "11 9" (for November 9) or "3 5" (for March 5) in the Date field. If you leave the Date field blank, Q&A will display any notes for the current date.


External Database
Reminder Date: Nov 9, 1995
Note:
Estimated taxes due on the 15th. Mail check today. Bring in the dog.
Reminder Date: Nov 9, 1995
Note:
Call Wilson and Hodges about progress on construction. Let out
Reminder Date: Nov 9, 1995
Note:
Meet with tax attorney at 1:00 p.m. Bring receipts, tranquilizers.
Current Database

```
        ══ Database Mini-Application Notes Lookup ══

      Date: Nov 9, 1995


 ExtKeyValue: 1995/10/06222
 ExtRecordNo: 222



 NoteField →
Notes for Thursday, November 9, 1995

Estimated taxes due on the 15th. Mail check today. Bring in the dog.

Call Wilson and Hodges about progress on construction. Let out the cat.

Meet with tax attorney at 1:00 p.m. Bring receipts, tranquilizers.
NoteField                  Num        0 %  56  Line 7 of 7
MAIN.DTF     New Record 1   of 1     Total Records: 0      Page 1  of 1

Esc-Exit     F2-Print       F7-Search      F8-Options    F10-Exit Editor
```

Figure 1. A typical MAIN.DTF record with the NoteField open and the retrieved notes displayed. You can build this multi-record retrieval capability into any database and enjoy convenient access to as much data as you need to see.

Create the sample databases

The best way to learn the technique is to create a couple of simple databases, plug in the programming, and watch what happens. You'll then be equipped not only to incorporate the feature into your "real" databases but to adapt it to meet other kinds of multi-record lookup requirements.

First, create a database named NOTES.DTF. Include in it the following fields, formatted as shown.

```
Field Name      Format
Reminder Date   Date
Note            Text
RecordNo        Text, read-only
```

Reminder Date contains the date you want to be reminded, Note contains the text of the note, and RecordNo contains a unique sequential record number that Q&A will fill in when you save a record.

After you've added and formatted your fields and saved the design, go to the Speed-Up Spec and make RecordNo speedy and unique (SU). Then, go to the Program Spec for NOTES, and type the following program in the RecordNo field (you can assign a field ID number other than 50):

```
#50: If @Todate(@Left(RecordNo,10)) <> Reminder Date
     Then RecordNo = @Str(Reminder Date) + @Str(@Number)
```

When you've entered the program, press F8, and enter "50" (without the quote marks) on the On-Exit field ID line, then press F10 to save the Spec. The program assigns record numbers in this format:

YYYY/MM/DD*nnnn*

The YYYY/MM/DD portion of the record number is Q&A's internal date format. In other words, if the Reminder Date is November 9, 1995, Q&A stores it as 1995/11/09. If you tell Q&A to set another field equal to Reminder Date, it will set it to the internal date (in this case, 1995/11/09) not to the displayed date. What's cool about the way Q&A stores dates is that every date is in numerical as well as chronological sequence. The four-digit year comes first, followed by the two-digit month, then the two-digit day. This way, when Q&A compares dates, the greater of the two "numbers" will always be closer to the present day.

Be Careful When Programming XLookups on Lengthy Key Values

When you use the Speed-up Spec to index your fields, Q&A indexes them only through their 16th character. So don't expect your Xlookups to be reliable past the 16th character of the key value in the external file. For the purpose of this application, it means that you must limit your RecordNo field's record number to the 10-character internal date string, plus no more than a six-digit @Number, so that no RecordNo value exceeds 16 characters. The limit won't present a problem until the @Number in the NOTES database trips 1,000,000 (one million).

As for the rest of the record number, if the next @Number in sequence is 2344, the RecordNo field will get 1995/11/092344 because the program is adding the string value (@Str) of the @Number to the string value of the Reminder Date. The If condition prevents the program from assigning a new record number to a record that already has one, unless you change the note's Reminder Date.

When you've complete NOTES.DTF, add a dozen or so records to it with different dates and reminder notes.

Next, create the MAIN.DTF database with the following fields and formatting:

```
Field Name      Format
Date            Date
ExtKeyValue     Text, read-only
ExtRecordNo     Number, read-only
NoteField       Text
```

The Date field will contain the date that matches the Reminder Date value—the first 10 characters of the RecordNo value. ExtKeyValue and ExtRecordNo are for programming and temporary storage, so you can make them read-only and eventually place them out of the way. (If you're using Q&A 5.0 or Q&A for Windows, you can hide them with color). You'll view your retrieved notes in NoteField. If you're using Q&A for DOS, it doesn't matter how wide you make NoteField because you'll be using the field editor to view your notes. If you're using Q&A for Windows, you can make the field three or four inches wide—Q&A will automatically expand its length when you click on it to view your notes.

Figure 1 shows what a MAIN record might look like after you've completed the sample application. In this example, the ExtKeyValue field contains the last RecordNo checked in NOTES. It didn't qualify, so the program ended by displaying the notes that did.

MAIN.DTF programming

Three of MAIN.DTF's fields require programming. Add the following program to the Date field:

```
> If Date = "" then Date = @Date; goto ExtKeyValue
```

This program assigns the current date to the Date field if you've left it blank. It then moves the cursor to the ExtKeyValue field.

ExtKeyValue's program is shown in Listing 1, and ExtRecordNo's program is shown in Listing 2. Add these two programs at the Program Spec, and you'll be finished.

Listing 1. Add this program to MAIN's ExtKeyValue field.

```
< ExtKeyValue = @XlookupR("NOTES", @Str(Date) +
"999999", "RecordNo", RecordNo");

If @Todate(@Left(ExtKeyValue, 10)) = Date Then

{

NoteField = "Notes for " + @Dow$(Date) + ", " +
             @Month$(Date) + " " +
             @Str(@Dom(Date)) + ", " +
             @Str(@Year(Date)) + "
" + "
" + @Xlookup("NOTES", ExtKeyValue, "RecordNo", "Note");

ExtRecordNo = @Tonumber(@Mid(ExtKeyValue, 11, 6)) -1;
ExtKeyValue = @Left(ExtKeyValue, 10) +
@Str(ExtRecordNo);

Goto ExtRecordNo

}

Else {@Msg("No notes for " + @Month$(Date) + " "
     + @Str(@Dom(Date)) + ", " + @Str(@Year(Date)));
     CHome
```

Listing 2. Add this program to MAIN's ExtRecordNo field.

```
< ExtKeyValue = @XlookupR("NOTES", ExtKeyValue,
                         "RecordNo", "RecordNo");

If @Todate(@Left(ExtKeyValue, 10)) = Date Then

{

NoteField = NoteField + "
" + "
" + @Xlookup("NOTES", ExtKeyValue, "RecordNo", "Note");

ExtRecordNo = @Tonumber(@Mid(ExtKeyValue, 11, 6)) -1;
ExtKeyValue = @Left(ExtKeyValue, 10) +
@Str(ExtRecordNo);

Goto ExtRecordNo

}

Else {@Fedit; Goto NoteField}
```

The only command in the Listing 2 program that won't work in Q&A for DOS 4.0 and Q&A for Windows is the @Fedit that opens NoteField's field editor. If you're using Q&A for DOS 4.0, omit the @Fedit and press F6 to open NoteField to view your notes. If you're using Q&A for Windows, the NoteField will automatically expand when you click on it.

Unless you want to save with the record what you've retrieved into it, you'll need to clear NoteField. Here's a sophisticated program that does it:

```
> Clear(NoteField)
```

What the ExtKeyValue and ExtRecordNo programs do

The ExtKeyValue program begins by searching NOTES for the highest record number whose date matches the date in the current record, and retrieves its RecordNo into ExtKeyField. It then compares the date portion of ExtKeyField (its first 10 characters) with the date in the Date field. If the two match—bingo!—you've got at least one valid NOTES record, and the fun begins.

If no matching record is found during this step, the program skips to the Else portion, displaying a message like this:

```
No notes for November 9, 1995
```

ExtKeyValue's program places "Notes for November 9, 1995" (or whatever the date is) in NoteField, follows it with two carriage returns to create a blank line, then retrieves the text of the first note (from the record with the one with the highest record number). It assigns to ExtRecordNo the last six digits of ExtKeyValue (the @Number portion of the retrieved record's RecordNo) less 1 (one). It then assigns to ExtKeyValue the date portion of the RecordNo string, to which it adds the revised ExtRecordNo value. You've now got an external key value that's one less than that of the record just retrieved, and program control is transferred to the ExtRecordNo field to search out additional notes for the same date.

ExtRecordNo's program performs a kind of Do-While loop. As long as the date portion of the note's RecordNo matches the Date field, it keeps looping and retrieving until it finds no more matching NOTES records. (It inserts a blank line between the retrieved notes to make them easier to read.) Finally, it passes control to the NotesField, and opens the field editor so you can view your notes.

Other application ideas

You might be interested in displaying something like invoice history information instead of reminder notes. From MAIN.DTF, perhaps a customer database in this case, you might want to have the option to display a chronological list of any customer's invoices, along with each invoice's date, amount, and date paid, like this:

```
Invoices for customer 3244

92134 1995/11/22, $32.99, Paid 1995/12/10
91077 1995/11/08, $48.95, Paid 1995/11/22
90897 1995/10/06, $88.55, Paid 1995/11/03
```

Let's assume the INVOICE database contains the following fields, in addition to others:

```
CustomerNo
InvoiceNo
InvoiceDate
Amount
DatePaid
RecordNo
```

For your INVOICE RecordNo field, you can concatenate (string together) the CustomerNo and InvoiceNo fields, but you'll need to separate the two values with a marker because customer numbers, unlike date values, might not all be the same length. Here's an INVOICE on-record-exit program that assigns a unique record number:

```
#50: RecordNo = @Str(CustomerNo) + "/" + @Str(InvoiceNo)
```

If the CustomerNo were 3244, and the InvoiceNo were 92133, Q&A would assigned RecordNo 3244/92133 to the invoice.

You'd replace MAIN.DTF's Date field with a CustomerNo field, and program it like this:

```
> If CustomerNo = "" then
  {@Msg("Please enter the customer number"); Goto
CustomerNo}
  Else Goto ExtKeyValue
```

Listing 3 shows the modified program for MAIN's ExtKeyValue field, and Listing 4 shows the modified program for the ExtRecordNo field. These programs are more complex because of the @Instr functions required to parse (separate) the record number into the customer and invoice numbers. The programs are also lengthier because they're retrieving not just one value (the text of a note), but the invoice number, date, amount, and date paid.

Listing 3. MAIN's ExtKeyValue field program, modified to work with record numbers comprised of the customer number and invoice number.

```
< ExtKeyValue = @XlookupR("INVOICE", @Str(CustomerNo) +
                "/999999", "RecordNo", "RecordNo");

If @Left(ExtKeyValue, @Instr(ExtKeyValue, "/")-1)
         = CustomerNo Then
{
NoteField = "Invoices for customer " + CustomerNo + "
" + "
" + @Xlookup("INVOICE", ExtKeyValue,
             "RecordNo", "InvoiceNo") + " " +
    @Xlookup("INVOICE", ExtKeyValue,
             "RecordNo", "InvoiceDate") + ", $" +
    @Xlookup("INVOICE", ExtKeyValue,
             "RecordNo", "Amount") + ", Paid " +
    @Xlookup("INVOICE", ExtKeyValue,
             "RecordNo", "DatePaid");

ExtRecordNo = @Tonumber(@Mid(ExtKeyValue,
              @Instr(ExtKeyValue, "/") + 1, 6)) -1;

ExtKeyValue = @Left(ExtKeyValue, @Instr(ExtKeyValue,
              "/") -1) + "/" + @Str(ExtRecordNo);

Goto ExtRecordNo
}
Else {@Msg("No Invoices for customer " + CustomerNo);
      CHome}
```

Listing 4. The modified program for MAIN's ExtRecordNo field. The optional Counter routine—you'll need a Counter field for it to work—displays a message-line progress meter.

```
< Counter = Counter + 1;
  @Msg("Retrieving records... " + @Text(Counter, "■"));

ExtKeyValue = @XlookupR("INVOICE", ExtKeyValue,
                        "RecordNo", "RecordNo");

If @Left(ExtKeyValue, @Instr(ExtKeyValue, "/")-1)
         = CustomerNo Then
{
NoteField = NoteField + "
" + @Xlookup("INVOICE", ExtKeyValue,
             "RecordNo", "InvoiceNo") + " " +
    @Xlookup("INVOICE", ExtKeyValue,
             "RecordNo", "InvoiceDate") + ", $" +
    @Xlookup("INVOICE", ExtKeyValue,
             "RecordNo", "Amount") + ", Paid " +
    @Xlookup("INVOICE", ExtKeyValue,
             "RecordNo", "DatePaid");

ExtRecordNo = @Tonumber(@Mid(ExtKeyValue,
              @Instr(ExtKeyValue, "/") + 1, 6)) -1;

ExtKeyValue = @Left(ExtKeyValue, @Instr(ExtKeyValue,
              "/") -1) + "/" + @Str(ExtRecordNo);

Goto ExtRecordNo
}
Else {Clear(Counter); @Fedit; Goto NoteField}
```

Conclusion

This record retrieval and display technique is adaptable to a variety of situations. You could use it to retrieve and display stock numbers and descriptions. Or, you might use it to display purchase order information for a selected vendor, notes on clients, or even a range of document filenames (stored in a database) that pertain to a selected client. The key is in the record numbering and lookup program. You need a record numbering scheme that accommodates @XLookupR's range logic, and programming to parse those numbers or strings into meaningful components.

You can embellish these techniques in other ways, as well. For the reminder notes, you can have Q&A format them differently, place an ASCII bullet character at the beginning of each note, include the record number with the note, or even display the notes in such a way that you can tag them for deletion.

For example, you could program Q&A to place a "[ ]" check box at the beginning of each displayed note, and its record number enclosed in French braces at the end. This way, you could type an "x" in the check boxes for the notes you want to delete and have your NoteField on-exit program strip out everything except the record numbers for the tagged notes, then place semicolons between the remaining numbers, like this:

```
1995/11/092344;1995/11/092346;1995/11/092349
```

With Q&A 5.0, you could have a macro (invoked by an @Macro command) copy this value to the Clipboard, then paste it into the Retrieve Spec of a NOTES.DTF Remove Selected Records operation. If you're using Q&A 4.0, you could program the NOTES Retrieve Spec to lookup the value.

Tom Marcellus is editor of *The Quick Answer* and author of *PC World Q&A Bible*, published by IDG Books. His QuickClick Calendar Plus—a time- and activity-tracking database for Q&A 5.0—is available from Marble Publications, publisher of *The Quick Answer.*

Letters

Easier Multiple Date Searches

In the @Help column of the January 1995 issue ("Who Has a Birthday This Month?") Dave Reid shows how to find a particular month regardless of the year or day. While Dave's solution is correct, using Q&A's right bracket ("]") Retrieve Spec parameter is too complicated for some. I use a technique that also works for any date field. Whether you're using the Retrieve Spec to search a database or print a report, such as a birthday list, you can enter a command like this in any field at the Retrieve Spec:

```
={@Month$(Birth Date) = "January"}
```

You can save this Spec as *01-January*, then clone it for each subsequent month of the year. I've found this to be a more effective way to retrieve records by month because I need to search multiple date fields in my ROLODEX database. My *Card Shop* report searches date fields named Male BDate, Female BDate, and Anna Date (anniversary), along with multiple Child BDate fields and an extra date field for events such as Grandparents' Day. On the 23rd of each month, I print the *Card Shop* report for the subsequent month, which my wife takes to the card and gift shop. For example, on December 23, I print the report using the following Retrieve Spec. If any of the date fields in the database are January dates, the report includes those records:

```
 ={(@Month$(Male Bdate) = "January")
Or (@Month$(Female BDate) = "January")
Or (@Month$(Anna Date) = "January")
Or (@Month$(Child1 BDate) = "January")
Or (@Month$(Child2 BDate) = "January")
Or . . . }
```

and so forth, for the rest of the date fields. I cloned 11 of these Retrieve Specs, one for each month of the year, and saved them as *02-February*, *03-March*, *04-April*, and so forth. I've developed the habit of assigning numeric prefixes to my saved Specs, so I can display them in chronological or numerical order.

Gary V. Chappelear, Birmingham, Alabama

Division Error Unearthed in Q&A 4.0

I discovered a strange mathematical error involving posting in Q&A 4.0. Apparently at random, in about 10 percent of the cases, Q&A performs division on posted field values as though the value of the denominator were one less than it actually is. The following are typical of the results I receive in the error cases, yet Q&A performs the calculation correctly in other records containing the same numerator and denominator:

Numerator	Denominator	Result	Should be
1	3	.500	.333
3	16	.200	.188
9	28	.333	.321

In my application, File A, which contains multiple records for each company, posts (Adds) to the Group Accumulator and Case Accumulator fields in File B, which contains one record for each company. File A also posts a "Yes" to the Programming Close field in File B, replacing the "No." The record-exit program in File B then calculates a value for the Percentage field by dividing Group Accumulator by Case Accumulator. After that, it changes the "Yes" in Programming Close back to "No."

I've made Q&A Technical Support aware of the apparent bug. They told me there's no fix and suggested I perform the calculation in a different way.

Gary Markman, Westport, Connecticut

QUICKTIP

(Too) Many Unhappy RETURNS

DOS WIN

If you receive a "Too many returns" error message during data entry, it means the cursor has entered a field containing a RETURN command with no pending GOSUB to return to.

To prevent this, you can make the RETURN field read-only so you can't inadvertently move the cursor into it. If you can't make the field read-only, you can position it on the form in such a way that the cursor isn't likely to enter it except when sent there by a GOSUB. During program execution, every GOSUB that's invoked must have a RETURN, and vice versa. If you don't need to return to a GOSUB, use a simple Goto command instead.

Tom Marcellus

The Program Spec

By Jeff Nitka

Taking Yes and No Beyond "Yes" or "No"

ONE of the things I like about Q&A is its ability to handle Yes/No values like the popular high-level programming languages C, C++, and Pascal. Because they store "No" as "0" and "Yes" as "1" internally, you can use "Yes" and "No" in ways that might not be obvious.

Consider a field named Error that references a help screen based on the value in a field named Data. Here's a typical approach:

```
< If Data = "Data 1" then Error = 1
  else if Data = "data 2" then Error = 2
  else if Data = "data 3" then Error = 3
  else if Data = "data 4" then Error = 4
  else Error = 0;
  If Error > 0 then @Help( @( Error ) )
```

Q&A stores each condition as a numeric 0 or 1 after evaluating it, which enables you to write the code more compactly:

```
< Error = 0 + 1 * (Data = "data 1")
  + 2*(Data = "data 2")
  + 3*(Data = "data 3")
  + 4*(Data = "data 4");
  If Error > 0 then @Help( @( Error ) )
```

If Data contains "data 2," the program in the first example sets Error to "2." In the second example, all the conditions store "0" except *Data = "data 2,"* which stores "1." When you work out the math, you'll see that Error gets the value "2."

Pitfalls

Ask yourself two questions when adopting an approach like this. First, which method will generally be more efficient? The If-Then-Else statements will work faster because once the condition is verified as True, no other condition is tested. In the second example, every condition must be evaluated. Secondly, is rewriting the code worth the effort? Here's an example where the rewrite is more elaborate than the original:

```
If Sales >= 100 then if Sales < 200 then Discount = 2.5
else if Sales < 400 then Discount = 5.0
else if Sales < 600 then Discount = 7.5
else Discount = 10 else Discount = 0;
```

Using the earlier approach, the rewrite would look like this:

```
Discount = ( Sales >= 100 ) * ( 2.5 * (Sales < 200)
+ 5.0 * (Sales >= 200 and Sales < 400) + 7.5
* (Sales >= 400 and Sales < 600) + 10 * (Sales >= 600) )
```

What can't be matched in the rewrite is the way the If-Then-Else statement is structured. The rewrite is obviously more complex and will run slower—clearly a case where changing the code is more trouble than it's worth.

"If" alternatives

Recall the previous statement, *If Error > 0 then @Help(@(Error))*. In this case, the "> 0" isn't required. Q&A, like C, interprets any non-zero value used in an If statement as True, and any zero value as False. (This applies only to functions that return numeric values or fields that store numeric values.) Therefore, *If @Instr() > 0 then* could be written as *If @Instr() then*; *If #1 <> "" then* could be written as *If @Len(#1) then*; and *If Error > 0 then* could be written as *If Error then*.

Substitutes for multiconditional statements

Following are two examples that demonstrate a method for making multiconditional statements more compact:

```
If Value=10 or Value=21 or Value=35 or Value=36
or Value=12 or Value=17 then Statement
```

One way around this convoluted structure is to use the @Instr function, like this:

```
If @Instr("10,21,35,36,12,17", @Str( Value ) ) > 0
then Statement
```

The second example produces the same result as the first, but there's a caveat. What if Value equals "3"? The first example wouldn't execute the code following *then* because "3" doesn't satisfy any of its conditions. With @Instr, however, the code following *then* would execute because "3" exists in the string "10,21,35,36,12,17." There's a way around this, but just be careful when using this approach.

Suppose the first example had this form:

```
If Value <> 10 and Value <> 21 and Value <> 35
and Value <> 36 and Value <> 12 and Value <> 17
then Statement
```

The @Instr function could be used in this case as well (with the same caveat as in the first example):

Continues on page 16

@HELP

Edited by Dave Reid

When Speedy-Unique Fields Get In the Way

DOS WIN

I use a Q&A 4.0 database to store names and addresses. The Last Name field is Speedy and Unique (there's an "SU" in the Last Name field at the Speed-up Spec). When I'm adding new records, if Q&A tells me the Last Name is a duplicate, I have to switch to Search/Update, and type the last name in the Retrieve Spec to check the address of the existing record to see if the new name is actually a duplicate. If it's not, I then return to Add Data and complete the new record. This takes too many steps. How can I simplify the process?

Albert Weiner, Pittsburgh, Pennsylvania

Although you can certainly create a macro to automate part of the process, let me suggest a quick change that should help. Q&A's Speedy-Unique (SU) feature is designed to warn of duplicate entries in a field. It's great when entering data that isn't usually duplicated, but address databases typically contain many duplicate last names, especially common names such as Jones and Smith. You can change the "SU" to "S" at the Speed-up Spec. Then, if you need to, you can check for duplicates by first using Search/Update, then pressing Ctrl-F6 to switch to Add Data. This will save you from having to switch from Add Data to Search/Update, then back to Add Data again.

Auto-Jump to a Document

DOS 5.0

I manage a database of documents for my company. The documents, which often require editing, contain a lot of enhancements and special formatting. Each record in the database describes a single document. I know how to find the record that describes the document I want to open, but I'd like to have Q&A automatically switch to the Write module and display it.

Ron Stevens, Willowdale, Ontario, Canada

If you have Q&A 5.0, or will be upgrading soon, you can add a field to your database and program it to do exactly what you need. You can name the field something like Edit this Document, and program it along these lines:

```
<#10 = @Shell("QA " + Document Name);
       #10 = "";
       Cnext
```

The program assumes that the field containing the document filename is called Document Name. The @Shell command starts a second session of Q&A and feeds it a command line parameter containing the name of the document. This way, Q&A automatically opens the specified document in the Write module. When you're finished editing the document, exit that session and Q&A will return to the first session at the spot where you left it. The @Shell command will place a needless "0" in this field. The rest of the program deletes it, then moves the cursor to the next field.

How to Print 1099 and W2 Forms

DOS

We use Q&A 4.0 for DOS at our company. I'd like your advice on how to fill out federal W2 and 1099 forms. We have all the pertinent information in databases, but I don't know how to make it print in the right locations on the official forms.

Ben Blume, Monroe, Indiana

You can use File / Print's Coordinate printing method to specify the precise location on the page to print the corresponding data. At the Fields Spec, enter the printing coordinates in a row number, column number format. For example, Enter "2,12" (without the quotes) in the First Name field to tell Q&A to print the First Name starting on the second row at the 12th position (12 characters from the left margin). You can press F1 while in the Fields Spec for online help. Some trial and error might be necessary to get the exact locations, but once your coordinates are defined accurately, you'll get excellent output. The only other concern is that the blank pre-designed form must be positioned in the printer the same way each time you print.

Dave Reid is a Symantec senior support analyst providing second-level assistance to the technical support representatives. He's also the coauthor of *The Q&A 4.0 Wiley Command Reference*, published by John Wiley and Sons and works as an independent Q&A consultant. PO Box 12083, Eugene, OR 97440.

?

Have a nagging question? Send it to **@Help, *The Quick Answer*, Marble Publications, Inc., PO Box 9034, Gaithersburg, MD 20898-9034** or fax to 301-424-1658. Please include your name, address, and phone number, along with your Q&A version number (and whether DOS or Windows) and a detailed description of the problem. We'll publish those questions we feel are of general reader interest. Individual responses are not possible.

Create Versatile Pick-Lists That Respond to User Preferences

Using a Lookup Table or external database in novel ways, you can control what appears on your data entry pick-lists—and let users set their preferences—without having to reprogram the database.

DOS 5.0

By Tom Marcellus

Q&A 5.0's Userselect and XUserselect pick-list commands offer exciting new ways to boost data entry productivity. You're no longer limited to the Alt-F7 list of restricted values, and you don't have to resort to displaying your lengthier "pick-lists" as documents retrieved into the Field Editor—a workaround I've always considered less than elegant. I'll briefly review what these new list commands do and how you incorporate them into your database. I'll then show you some useful tricks you can perform with them, including modifying them without returning to the Program Spec.

Adding a Userselect list to your program

The Userselect command displays the list of items you program into the command. You can choose from two variations—a statement or function. Userselect's straightforward syntax looks like this:

```
Userselect("Item 1,Item 2,Item 3", Target field)
@Userselect("Item 1,Item 2,Item 3,")
```

In both variations, you can include as many list items as you want. In the statement, you must also tell Q&A the name (or field ID) of the *Target field* that receives the selected item.

You can include a Userselect statement like the following in either the Program or Navigation Spec:

```
< Userselect("Red,Blue,Green,Black", Color); Cnext
```

When the cursor enters the field, Q&A displays an alphabetical selection list containing the four items:

```
Black
Blue
Green
Red
```

You simply highlight the item you want and press Enter (or click on the item with your mouse), and Q&A places it in the Color field and moves the cursor to the next field.

You'll get the same result using the @Userselect function this way:

```
< Color = @Userselect("Red,Blue,Green,Black"); Cnext
```

Even though both commands include a field assignment (a field to contain the selected item), Q&A will accept the statement and the function in the Program Spec, but only the statement in the Navigation Spec.

Like @Lookup and @XLookup, the @Userselect function doesn't include a *target* or *destination field* parameter, so you can do something with the selected item other than placing it in a field. I'll show you later how you can take advantage of this to update your selection list without having to change your programming.

Adding an XUserselect list to your program

The XUserselect commands display a list of selections from the current database (using @Fn as the *Database* parameter) or an external database. An XUserselect list, unlike a static Userselect list, is dynamic because it includes whatever information is in the external database at the time Q&A displays the list.

This brings us to an important distinction between Userselect and XUserselect. To add, delete, or update the items on a Userselect list, you have to *reprogram* the list. To add, delete, or update the items on an XUserselect list, you update the lookup database by adding, deleting, or editing its records. The XUserselect commands include the following:

```
XUserselect("Database", "Ext key field", Target field)
@XUserselect("Database", "Ext key field")
```

Like @Userselect, you can do something with @XUserselect's value other than placing it in a field.

XUserselect and @XUserselect include range variations. Here's the XUserselectR range command's syntax:

```
XUserselectR("Database", "Ext key field", "Ext data
field",
          "Start range", "End range", Target field)
```

The @XUserselectR command's syntax is the same, but without the *Target field* parameter. See the

© Marble Publications, Inc.

sidebar, "What the Range Parameters Do."

Now that you're familiar with how these new list commands work, let me show you some power techniques you can use to get more from them than you might expect.

Power pick-lists

The first technique allows you to create a Userselect list you can modify without having to open the Program or Navigation Spec. In other words, you program Userselect at the outset, and if you want to change the items on the list, you simply change them in your Lookup Table.

The second technique lets you program an @XUserselectR command with variable parameters you can store in a Lookup Table. This way, if you want to change those parameters, you can change them in the table instead of in the programming.

Both techniques have infinite variations and can be adapted to suit all kinds of needs. I'll illustrate them within the framework of Q&A 5.0's new @Insert and @Fedit commands. @Insert inserts a document in a field, and @Fedit opens the Field Editor so you can view the inserted document or any lengthy value.

@Userselect versatility

Consider the following typical @Userselect command in an @Insert command in the Get Doc field:

```
< Get Doc = @Insert(@Userselect(

   "HOLIDAYS.DOC,VACATION.95,VACATION.96,BIRTHDYS.DOC,
    CONTRACT.DOC,REMINDER.DOC,HOLIDAYS.DOC,PHONES.DOC"));

   @Fedit
```

The action here is straightforward. When you move the cursor to the Get Doc field, Q&A displays a list of the document filenames presumably stored in your default documents directory (see Figure 1). You choose a filename, and the program inserts the document in the Get Doc field and opens the Field Editor so you can view it.

Now, suppose you want to change any of the document filenames on your pick-list. You'll have to return to the Program Spec and modify @Userselect accordingly. Or will you?

If you place the filenames in the database's Lookup Table, you can program @Userselect to retrieve them from there and avoid having to hard code them into your program. This way, should you want to substitute other documents, you can do so quickly. What's more, if someone else uses the database, you can lock the Program Spec and still give them the option to amend their list of document filenames by giving them access to the Lookup Table.

Listing 1 shows the same @Userselect command shown previously. In this case, though, instead of specifying the document filenames, you have Q&A get them from the Lookup Table.

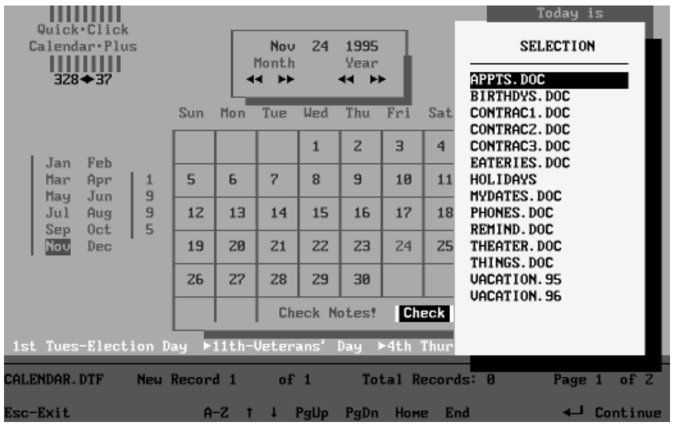


Figure 1. An @Userselect pick-list containing document filenames.

What the Range Parameters Do

Don't make the mistake of thinking that the "R" in XUserselectR makes the command work like an XLookupR range command. Their logic is altogether different. XLookupR finds the record with the next lowest *External key field* value if an exact match can't be found. XUserselectR, on the other hand, includes *Start range* and *End range* parameters for the *External key field*. When executed, it includes on the pick-list all the records that fall inclusively within the two range parameters you specify.

There's another distinctive aspect to the XUserselectR command. Although its *External data field* parameter is similar to XLookup's *Lookup field* parameter, the action is different. XUserselectR's resulting list displays the unique *External key field* values within the specified range, but your selection retrieves the record's corresponding *External data field* value. Accordingly, if your *External key field* is Company Name and your *External data field* is Acct No, Q&A will display the list of company names and return the account number of the one you select.

For XUserselectR programming examples, see my review of Q&A 5.0 in the August 1995 issue.

Listing 1. Displaying a list of documents filenames from the database's Lookup Table.

```
< Get Doc = @Insert(@Userselect(

  @Lookup("A-Docs", 1) + "," + @Lookup("A-Docs", 2) + ","
+ @Lookup("A-Docs", 3) + "," + @Lookup("A-Docs", 4) + ","
+ @Lookup("B-Docs", 1) + "," + @Lookup("B-Docs", 2) + ","
+ @Lookup("B-Docs", 3) + "," + @Lookup("B-Docs", 4) + ","
+ @Lookup("C-Docs", 1) + "," + @Lookup("C-Docs", 2) + ","
+ @Lookup("C-Docs", 3) + "," + @Lookup("C-Docs", 4) + ","
+ "Unlisted Document")); @Fedit
```

In this example, the Lookup Table contains three Key column values: "A-Docs," "B-Docs," and "C-Docs" (see Figure 3). The corresponding lookup values in columns 1 through 4 are the document filenames. One additional item is included on the list, "Unlisted Document," in case you want to open the Field Editor to retrieve a different document.

When the program executes on entering the Get Doc field, Q&A builds the list of document filenames using the Lookup Table, adding a comma between each one to conform to @Userselect's syntax. (See the sidebar, "How to Keep Your List Clean.") The list appears for you to make your selection, and from then on it works the same as the earlier example with the hard-coded filenames.

Your Lookup Table could conceivably include additional key values for "D-Docs," "E-Docs," and so on. By assigning up to four document filenames to each Key value, you could create a list of documents as long as you need.

@XUserselectR versatility

Q&A compiles an XUserselect pick-list by retrieving the information from another (or the current) database. Now I'll show you a different document retrieval technique that's really two techniques rolled into one.

With the first technique, you store your document paths, filenames, and descriptions—but not the documents themselves—in a simple documents database and have Q&A build your list of document selections from its records. This way, you can add, delete, or change your document selections simply by updating the database. What's more, your pick-list will display document *descriptions* rather than filenames. You can choose documents by their more meaningful descriptions, but Q&A will still pass their filenames for the @Insert command.

The second technique uses the Lookup Table, but in a different way than you saw earlier. In this case, the table supplies the @XUserselectR command with its *Database*, *External key field*, and *External data field* parameters. This way, you can avoid hard coding these parameters into the program. You simply add or update them in the table, and Q&A builds whatever pick-list your table dictates. Again, this technique gives you more flexibility because of the following:

- The name of the documents database—and the names of its *External key field* and *External data field*—can be anything.

- By having the filename parameter point to the local drive, each user on a network can maintain and access his or her own documents database.

- No programming changes are

How to Keep Your List Clean

When retrieving document filenames or other @Userselect values from the Lookup Table, you'll run into a glitch if any of the pertinent cells are blank. You're building the @Userselect list by adding a comma between each item, so the program will add two commas for each blank cell, and you'll wind up with blank entries at the top of your displayed list.

You can prevent this by building the list in the Get Doc field (rather than in memory), where you can use Q&A's string manipulation functions to replace any superfluous commas. Here's an example of how to do this.

```
< Get Doc =

  @Lookup("A-Docs", 1) + "," + @Lookup("A-Docs", 2) + ","
+ @Lookup("A-Docs", 3) + "," + @Lookup("A-Docs", 4) + ","
+ @Lookup("B-Docs", 1) + "," + @Lookup("B-Docs", 2) + ","
+ @Lookup("B-Docs", 3) + "," + @Lookup("B-Docs", 4) + ","
+ @Lookup("C-Docs", 1) + "," + @Lookup("C-Docs", 2) + ","
+ @Lookup("C-Docs", 3) + "," + @Lookup("C-Docs", 4);

Get Doc = @Replace(Get Doc, ",,,,,", ",");
Get Doc = @Replace(Get Doc, ",,,,", ",");
Get Doc = @Replace(Get Doc, ",,,", ",");
Get Doc = @Replace(Get Doc, ",,", ",");

If @Left(Get Doc, 1) = "," Then
Get Doc = @Replfir(Get Doc, ",", "");

If @Right(Get Doc, 1) = "," Then
Get Doc = @Repllas(Get Doc, ",", "");

Get Doc = @Insert(@Userselect(Get Doc)); @Fedit
```

The @Replace commands delete incrementally smaller blocks of commas until only one comma remains between each item. @Replfir (replace first) and @Repllas (replace last) then delete any commas at the beginning and end of the list. The scrubbed list is now ready for display.

required to specify a different database name or different field names. Without having to touch the programming, a sales order database that includes links to inventory and customer databases lets the user plug in the database name along with the *External key field* and *External data field*, making the application more generic and distributable.

To take advantage of this technique, simply add a "Get Docs" or similar Key column entry to your Lookup Table. Leaving the corresponding column 1 blank for the time being, enter the name of the documents database, such as C:\QA\DATA\DOCS in column 2. In column 3, enter the name of the *External key field*, which might be named Description, since it's the field that contains the document description. (This field must be Speedy and Unique, and should be no longer than 19 characters, so the entire description will fit on the pick-list.) Then, in column 4, you enter the name of the *External data field*—the field that contains the document filename.

Now, let's return to column 1 on the "Get Docs" row. Since you don't need this column for anything else, you can optionally have it contain a conditional switch and program Q&A to display the pick-list of documents only if it contains "Yes." With that in mind, Listing 2 shows what your revised get Get Doc program might look like.

Listing 2. Using @XUserselectR to display a list of document descriptions rather than filenames.

```
<
If @Lookup("Get Doc", 1) = "Yes" Then
{
Get Doc = @Insert(@XuserselectR(
                  @Lookup("Get Doc", 2),
                  @Lookup("Get Doc", 3),
                  @Lookup("Get Doc", 4),"",""));
If Get Doc <> "" Then @Fedit
}
Else
{
@Msg("Document display switch off or document
not found");
Cnext
}
```

The program first checks the Lookup Table's "Get Doc" switch. If it's set to "Yes," Q&A creates the @XUserselectR command using the table's parameters. The command executes, opening the external database. Because the *Start range* and *End range* parameters have been left blank (the two sets of double quote marks), the resulting pick-list will include the descriptions of all the documents in the database (see Figure 2). You make your selection, and Q&A returns it's filename to the @Insert command and opens the Field Editor with the document displayed.

If Q&A can't find the document, or you escaped from the list, Get Doc will be empty and the Else condition will execute, displaying a message and moving the cursor, in this case, to the next field.

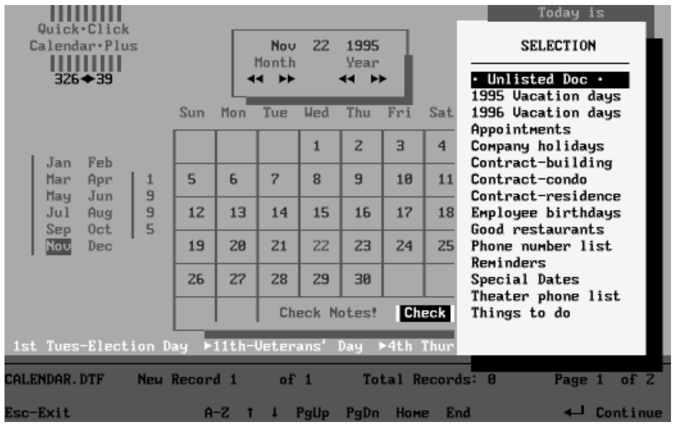
To return to the record without saving the document in the field, simply press Esc, then confirm that you want to abandon your "changes."

Using Lookup Tables to store switches and user preferences

As you've seen, you can get a great deal more from your Lookup Table than merely static data. In addition to using it in the traditional way to store information, such as tax tables, shipping zones, state names and abbreviations, and so forth, you can use it to specify and store configuration and program execution preferences as well.

The sample Lookup Table shown in Figure 3 (see page 16) includes some of the options we've looked at. The table's first row, instead of containing the usual Key value and column entries, contains column headings that describe the "CheckNotes," "Get Doc," and "NotesMacro" switches along with their corresponding parameters. The second and third rows contain configurable parameters for an @XLookup and @XUserselect command, respectively. The fourth row contains a switch that determines whether or not an @Macro command—and which macro—is executed. Three rows in the Key column contain dashed lines as separators, and three rows contain document filenames for the @Userselect list I discussed earlier.

This is contained on page 1 of the table. You can


Quick•Click
Calendar•Plus
326◆39
Today is
Nov 22 1995
Month Year
Sun Mon Tue Wed Thu Fri Sat
Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1995
Check Notes!
Check
1st Tues-Election Day ▸11th-Veterans' Day ▸4th Thur

```
SELECTION
• Unlisted Doc •
1995 Vacation days
1996 Vacation days
Appointments
Company holidays
Contract-building
Contract-condo
Contract-residence
Employee birthdays
Good restaurants
Phone number list
Reminders
Special Dates
Theater phone list
Things to do

CALENDAR.DTF   New Record 1   of 1   Total Records: 0   Page 1  of 2
Esc-Exit       A-Z  ↑  ↓  PgUp  PgDn  Home  End     ↵ Continue
```

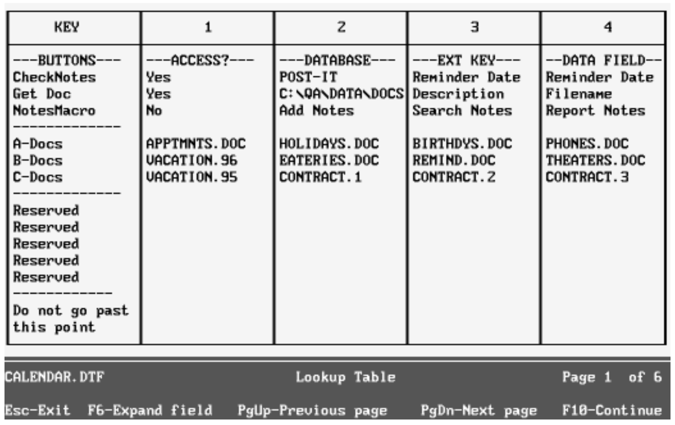
Figure 2. An @XUserselectR pick-list containing document descriptions. When you make a selection, Q&A returns the corresponding filename to the @Insert command and opens the Field Editor.

always use subsequent pages to store your usual lookup information.

Although you can create and use an external "preferences" *database* to store and access the same options and parameters as a Lookup Table, the table gives you several advantages:

- Because Q&A stores the Lookup Table with the database, the table is automatically available when you use the database.
- Because the table is in memory, your programs access it at RAM speed.
- You have the option of locking the database design and programming, while leaving a way for users to configure the database to better serve their needs.

These techniques aren't limited to displaying a filename list and inserting documents. You can adapt them to most any application requiring pick-lists that can be customized without reprogramming.



KEY	1	2	3	4
---BUTTONS---	---ACCESS?---	---DATABASE---	---EXT KEY---	--DATA FIELD--
CheckNotes	Yes	POST-IT	Reminder Date	Reminder Date
Get Doc	Yes	C:\QA\DATA\DOCS	Description	Filename
NotesMacro	No	Add Notes	Search Notes	Report Notes

A-Docs	APPTMNTS.DOC	HOLIDAYS.DOC	BIRTHDYS.DOC	PHONES.DOC
B-Docs	VACATION.96	EATERIES.DOC	REMIND.DOC	THEATERS.DOC
C-Docs	VACATION.95	CONTRACT.1	CONTRACT.2	CONTRACT.3

Reserved				
Reserved				
Reserved				
Reserved				
Reserved				

Do not go past				
this point				

```
CALENDAR.DTF                    Lookup Table                    Page 1  of 6
Esc-Exit  F6-Expand field   PgUp-Previous page    PgDn-Next page    F10-Continue
```

Figure 3. Not your everyday, run-of-the-mill Lookup Table. This one does a lot more than store the usual data values—it stores preferences and switches, too.

Tom Marcellus is editor of *The Quick Answer* and author of *PC World Q&A Bible*, published by IDG Books. His QuickClick Calendar Plus—a time- and activity-tracking database for Q&A 5.0—is available from Marble Publications, publisher of *The Quick Answer.*

The Program Spec

Continued from page 10

```
If @Instr("10,21,35,36,12,17", @Str( Value ) ) = 0
then Statement
```

In Q&A, as in any high-level programming language, these techniques won't normally improve the program's efficiency. (In some cases they might actually hinder it.) But they give you options for making your programs more compact, more readable, or more in line with your personal preferences. Whatever your programming style, it can be useful to know that Q&A allows you to manipulate Yes/No values in different ways.

Jeff Nitka recently graduated from Rutgers University with a Bachelor of Science degree in mathematics and computer science. He develops Q&A applications part-time for Epoch Software, 908-874-3989. Jeff is the author of the Q&A Program Evaluator, a program debugging utility available from Marble Publications, Inc., publisher of *The Quick Answer.*

MARBLE
PUBLICATIONS
PO Box 9034
Gaithersburg, MD 20898-9034

Second-Class
Postage Pending at
Rockville, MD