# 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


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

The Quick Answer™
The Independent Guide to Q&A Expertise

June 1997
Volume 8, Number 6

1 **Database Corruption Revisited**
*John Dow*

2 ***Editorial:*** **Keyboardless in Costa Mesa**

2 ***Tip:*** **Conditional Merging**
*Jeff Nitka*

4 ***Tip:*** **Auto-Capitalize Last Names**
*Roger Skidmore*

5 **One-Click Database Spell-Checking**
*Erika Yoxall*

6 **Add Snap, Crackle 'n Pop to Your System**
*Gordon Meigs*

8 **@Help**
*Edited by T.J. Shuflin*

9 **Forms Within Forms, Part Deux**
*Tom Marcellus*

11 ***The Program Spec:*** **Typecast Functions**
*Jeff Nitka*



# Database Corruption Revisited

JOHN DOW

Is your database having a bad hair day? The Master of Database Disaster brings some insight on what might be wrong, and what you can do to improve its mood.

DOS

**IN** my article, "Using QEXTRACT on Corrupted Databases" in the January 1997 issue, I discussed how to extract data records from a damaged database. I gave the pros and cons of using Q&A itself, my DTFDATA utility, and Symantec's QEXTRACT, concluding that the best way to extract both data and specs was QEXTRACT.

Since then, people have sent me many databases with damage of one sort or another, and I decided to try to write a program that would do a better job of extracting than QEXTRACT. In the course of analyzing these damaged databases and developing a new utility program, I've learned some things about Q&A databases.

This article is an attempt to pass on some crucial knowledge to those with a database that has "gone south" or who are concerned that it might happen and want to take appropriate preventive measures. My discussion assumes you'd be without a useable backup of the database, and forced to extract what you could from a damaged one.

## Your database is a zoo

A zoo is a good analogy because it has *animals* and *enclosures*, a *perimeter* beyond which the animals can't go, and an *office* where administrative details are maintained.

The *perimeter* means that everything is kept inside the .DTF file. The *office* means that there are some bits and pieces of information inside a .DTF that Q&A needs to manage the zoo.

To understand what I mean by *enclosures* and *animals*, consider all the things in a .DTF file that pertain to the fields defined on a form. There's the data, of course. But there's also information on initial and restricted values, programming, up to 200 reports, field level password protection, mass update specs, and on and on. These, in our analogy, are the animals. For Q&A to take care of all of these animals and to find a zebra when it's needed, the .DTF file has an internal structure—*enclosures*—to contain them.

## What a Q&A Recovery does

Using this zoo analogy, you can say what Q&A's Recover Database feature does and doesn't do. First, it removes the unused space, thereby reducing the .DTF's file size. This affects the zoo's perimeter. Second, it makes sure the administrative information is correct. For example, the database contains a count of the records in it. (Let's call them rabbits in the petting zoo.) If you use File / Search and display a record, *Total Forms* appears instantly because the rabbit count is maintained in the zoo office. This count, though, can get out of sync if everyone's dealing with an emergency and doesn't notice that a couple more rabbits were born. (In an actual database, not closing it properly is a likely cause of an incorrect count.) When you recover the database, every rabbit is carefully counted to make sure the office tally is up-to-date.

Third, a database recovery rebuilds the enclosures. Perhaps you've received an error message that a database wasn't properly closed. Q&A prevents you from using it, and advises you to recover it. The recovery ensures the proper enclosures exist.

However—and this is important—recovery does nothing for the animals in the zoo. The zoo's enclosures are repaired by carpenters and bricklayers, not veterinarians. If an animal is sick before the database is recovered, it will still be sick afterward. More on what to do with sick animals later.

*Continues on page 3*

# Keyboardless in Costa Mesa

**TOM MARCELLUS**

MY keyboard of choice has always been the Northgate Omnikey. I like its solid construction (metal base, not plastic), click-tactile key action, and oversized Backspace key. But mostly I like having all the F-keys on the left where God intended them and where they facilitate one-hand use with Shift, Ctrl, and Alt in F-key-intensive programs like Q&A.

When my Omnikey of many years recently died, I found that Northgate had, too, and that *nobody* made keyboards anymore with the F-keys on the left! Then I stumbled on a reference to the company in Alan Freedman's *Computer Desktop Encyclopedia* (ISBN 0-8144-0010-8)—a 1,000-page illustrated compendium of anything you'd ever want to know about PCs. I emailed the author about it, and he replied with the name of a company (System Design Advantage, 612-703-3500) that still supplies the Omnikey. My new Omnikey includes the same bank of F-keys on the left (and another bank across the top), along with a new Windows 95 key you press to minimize Q&A—or any program—and switch to the desktop with the Start menu displayed. I highly recommend both the keyboard and Freedman's book.

Your Q&A database contains a lot more than just records. Find out why **John Dow** calls it a zoo, and what you can do if its cages begin to collapse or its animals get sick.

How would you react to a supplier's quotation that was carelessly riddled with misspelled words? If you use a database to create quotations, invoices, or the like—whether or not you merge print them—let **Erica Yoxall** show you how to spellcheck them on-the-fly.

If you share databases in a multiuser environment, **Gordon Meigs** shares some tricks and insights that can help you avoid network bottlenecks and boost Q&A's performance.

Last month, I began a two-parter on adding subforms to a database. The programming here in Part 2 should keep you off the streets for a while.


The Quick Answer™

*The Independent Guide to Q&A Expertise*

**Editor** Tom Marcellus
**Publisher** Michael Bell

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.

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





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.

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.




## Quick Tip — Conditional Merge Document Insertions

DOS

Suppose you're designing a merge document that has to include the customer's fax number—but with a catch. If the Fax Number field is empty in the merged record (the customer doesn't have a fax machine), the printed document has to contain the line, *Fax Number: Not Applicable.* Here's a conditional expression you can use in the document to do the trick:

```
Fax Number: *pg {@Txt(Fax Number <> "", Fax Number) +
              @Txt(Fax Number = "", "Not Applicable")}*
```

Now, suppose you need to include the fax number in a merge document that *doesn't* use the customer database? Here's a *non*-conditional Xlookup expression you can use:

```
Fax Number: *pg {@Xlu("CustInfo",AcctNo,"AcctNo",
                    "Fax Number")}*
```

To make it conditional like the earlier example, try this:

```
*pg {@Txt(@Xlu("CustInfo",AcctNo,"AcctNo","Fax
Number")<>"",@Xlu("CustInfo",AcctNo,"AcctNo","Fax Number"))
+ @Txt(@Xlu("CustInfo",AcctNo,"AcctNo","Fax
Number")="","Not Applicable")}*
```

Another example of a conditional insertion. I merge-print invoices from an Orders database. The invoice can include up to 10 lineitems, each of which is inserted with an Xlookup. If the lookup field is empty, I insert a line from a file named Order.fll:

```
Item No     Description      Qty   Price  Amount

*pg {@Xlu(@Fn, Line1 + 1, "Line1", "Print") +
   @Txt(@Error, @Insert("Order.Fll"))}*

*pg {@Xlu(@Fn, Line1 + 2, "Line1", "Print") +
   @Txt(@Error, @Insert("Order.Fll"))}*
```

**Jeff Nitka, 105020.2215@compuserve.com**

# Database Corruption . . .

*Continued from page 3*

### When to recover a database

Sometimes Q&A will advise you to perform a recovery. That's a good time to do one. But when else? If you get one of the system errors, such as C-2 04CC, it's probably a good time to do one.

You can always recover a database just for good measure. Provided you backup a damaged database before you attempt to recover it, it can't hurt to try. (You *must* make a copy before attempting to recover. In some cases, though rare, an attempt to recover can make things much, much worse.) Understand, though, that while there are times when a recovery is exactly what's needed—and all that's needed—it won't heal any sick animals.

### When Q&A won't recover a database

Sometimes Q&A reports that it can't recover a database. Or, it simply "blows up" while making a valiant effort. In such a case, the usual technique is to try to extract the specs and data and recreate the database. Another option is to send it to Symantec's repair center. (Last I heard, Symantec's database repair and password removal line was 541-984-7785.)

Now there's another option. In some cases, my new program, DTFDOCTR, can rebuild a database enough so that you can then use Q&A to recover it. In the zoo analogy, if carpenters and bricklayers recover the database, then DTFDOCTR is a HAZMAT team that first removes any hazardous materials so the carpenters and bricklayers can safely do their jobs.

The current version of DTFDOCTR doesn't always make an otherwise unrecoverable database sufficiently healthy for Q&A to recover it, but it's usually successful enough that you can use DTFDOCTR or DTFSPECS to transplant reports and to view and extract all of the other information they customarily access in a database.

In the event that nothing works to recover the database, you at least want to get the data out of it. (Often, people have a backup that contains up-to-date specs, but is missing the most current data.) QEXTRACT was written in 1993. When you first load it, you get this friendly greeting: *This is beta software! Please do not distribute, use at your own risk*. DTFDOCTR, on the other hand, was written in 1997, and benefits from the prior experience of QEXTRACT. As a result, it doesn't have several data extraction deficiencies QEXTRACT has.

In my January article, I described two QEXTRACT deficiencies. Since then I've found a third. First, it can scramble the field order. Second, it often fails to extract some records. Third, if there's double quote mark (") in a field, QEXTRACT will extract it as a single quote (').

Also, I recently noticed that if you use QEXTRACT to extract programming that includes field names, you'll get crazy ASCII characters where the names ought to be.

Among other things, I created DTFDOCTR to fill in the gaps in QEXTRACT's performance. For example, it doesn't scramble fields. In fact, it never generates multiple output files that require you to create merge specs to import the data. Instead, as it extracts the records it makes sure that each one is current. (See the section on database generations below). Second, it doesn't miss records the way QEXTRACT does. Once in a while, it actually extracts a few extra records that appear to be duplicates, or records that were being changed at the moment the damage occurred. Third, it extracts a double quote mark properly.

Although DTFDOCTR doesn't extract programming, DTFSPECS does—and it extracts field names and each field's program to a separate file, making it easier to import them into a new database.

To summarize, at this point DTFDOCTR is a better tool for extracting records from a damaged database than QEXTRACT, and DTFDOCTR together with DTFSPECS are better at extracting programming specs.

### Caring for the animals

In our analogy, the animals in the zoo are all of the things stored in the database that relate to the fields. Some animals are more prone to illness than others. The following seem to get sick most often: Program spec, Initial values spec, Restrict spec, Report retrieve specs, Print retrieve specs.

But how can you tell a critter is sick? Unfortunately, Q&A isn't a veterinarian, so frequently what you see when looking at a spec isn't what you get when using or copying the spec.

Sometimes, when you look at a sick animal (perhaps a report retrieve spec) in Q&A, it appears perfectly healthy. But you might not notice it's missing something. Or, it might not work properly. Furthermore, even though it looks okay, if you copy the design of the database, it might cause Q&A to hang or even abort.

On the other hand, sometimes a spec *looks* sick. You might find lots of asterisks (*) in it, or "lucky charms" (odd ASCII characters). You might even see text from another spec.

DTFDOCTR can tell you which specs are sick. Some it will advise are too far gone and should be deleted; others it will offer to try to fix.

Can you edit (try to heal) a sick spec with Q&A? You can try. Sometimes it'll work. Other times it will seem to have worked (your PC isn't smoking) but it might have chopped off part of the spec in the process.

If you copy the database's design, will that cure a sick spec? Absolutely not! Copying the design usually makes sick animals much worse (Q&A 5.0 is especially likely to do this).

If you know exactly what a sick spec ought to be, the safest thing is to delete it and recreate it.

The next best thing is to use DTFDOCTR to try to fix it, then look at it yourself with Q&A. DTFDOCTR might

not always restore it to perfect health, but it will ensure that what you see is what you get. (Also, you're less apt to lose part of the spec.) However, there's a good chance that you'll have to manually touch up some of the specs (so you can save them by pressing F10) or to make them correct from a programming standpoint.

### Why databases develop problems

A Q&A database is a sprawling complex. Enclosures can get damaged. Animals can get sick. This happens most often when a PC crashes due to a Windows problem, or when there's a power loss or a network problem that prevents a database from being properly closed. Outdated system software can cause problems as well.

There's also a known problem with Q&A 5.0's Copy Design Only feature. If certain specs aren't the current generation (see below), copying the design is guaranteed to corrupt them. DTFDOCTR will tell you which specs aren't the current generation, so if you plan to copy the design, you can make everything current beforehand.

Gordon Meigs' article in the October 1996 issue discusses how report specs can go bad as a result of copying the database design, and how to repair them. He mentions an earlier program of mine, DTFCHECK, that reports on bad report specs. DTFDOCTR is a successor to DTFCHECK, and much more advanced. For example, DTFCHECK sometimes reports problems when there are none, and it doesn't check everything that could have this type of a problem. DTFDOCTR does these things, and has the ability (also in DTFSPECS) to copy report specs to macro files (so you don't have to recreate them) or to another database.

### Generations

I and others have used the term *generations* with regard to Q&A databases. From conversations I've had with people over the past few months, the term needs to be defined:

> *A database generation is an ordered list of the fields existing in the database at a point in time.*

Imagine a database with three screens and 22 fields, and with field labels and other explanatory and cosmetic text on the screens. Suppose you removed everything but the fields, and scrunched them at the top of the first screen, like this:

```
<AA><AB><AC><AD><AE><AF><AG><AH><AI><AJ><AK><AL><AM><AN><AO><AP>
<AQ><AR><AS><AT><AU><AV>
```

If you redesigned the database this way, to Q&A it would still be the same generation, because it has the same fields in the same order. You could change numeric fields to date or text fields, change initial values and restrict values, change programming, make fields speedy, and change field security—and none of these would change the database's generation.

But, if you change the field order at all—you swap <AE> and <AS>, for example—you've created another generation. If you remove a field or add one—again, you've created another generation.

### Conclusion

Q&A is a powerful database with a complex file structure that makes it susceptible to certain kinds of trauma from internal or external sources. Q&A itself isn't very adept at revealing certain types of problems (sick animals), and its recover feature only corrects some problems (enclosures) but not others (sick animals). QEXTRACT does a pretty good job, but it has a couple of flaws. DTFDOCTR can warn you of impending problems if you're about to copy the design of a Q&A 5.0 database. It can sometimes rebuild a database that Q&A can't. It can usually fix a spec (sick animal) so that it's healthy enough for you to examine it and complete the job using Q&A. And it extracts data better than QEXTRACT does.

## Resource

DTFDOCTR is available from the author for $100 postpaid. (MasterCard and Visa accepted.) Screen shots of the utility and a demo can be downloaded from www.pgh.net/~jtd. For more information, contact the author at jtd@pgh.net, on CompuServe at *JohnDow*, or at 412-521-1577.

Quick Tip

# Auto-Capitalize Last Names

DOS WIN

Q&A's initial caps (TI) field format capitalizes the first letter of every word, turning a double-barrel last name like McKay-Smith into *Mckay-smith*, which isn't what you want. For some fields, particularly a LastName field, you might be better off having Q&A auto-capitalize just the first letter in the field. This way, with most entries you'll get the result you want (without having to press the Shift key), and you can manually format the occasional odd last name.

Here's how you do it. Format the LastName field with *T* for ordinary text. Then, add a field named Var1 (for variable #1), and format it *TU* for uppercase text. Finally, type a program like this in the LastName field:

```
> Var1 = @Left(LastName, 1);
LastName = @Replfir(LastName, Var1, Var1);
Var1 = ""
```

You can use Var1 and the same program in any other field whose first letter you want Q&A to auto-capitalize.

**Roger Skidmore, Isle of Wight, rogerskid@compuserve.com**

# One-Click Spell-Checking for Multiple Fields

ERICA YOXALL

Q&A can spellcheck an expanded database field, which is fine if you have just one or two fields to spellcheck. But what about a twenty-five field quotation?

A client of mine at a precision molding company had designed a spiffy Q&A application to automate the company's quotation process. One catch, though, was that the quote department insisted that each of twenty-five possible description fields be spellchecked before printing the quote. Obviously, no one was interested in spellchecking that many fields one at a time.

What I developed to solve the problem can easily be added to any database. It can handle any number of fields as long as the total information in all of them doesn't exceed Q&A's 32K per field limit.

## The fields to add

First, you'll need a **SpellcheckButton** field. Make it a text-formatted "button" field you click on to run the spellcheck. Place it where it's easily accessible, and use the Palette Spec to color it to distinguish it from regular data fields. Use the Initial Values Spec or an on-record-entry program to set it's text to *Spellcheck*, and justify it center (T,JC) at the Format Spec. Use the Navigation Spec to make sure it's accessible only by clicking in it.

Next, you'll need a **Spellchecker** field, a one-character text field where all the text to be spellchecked will be collected. You can hide this field and make it inaccessible except by the Spellcheck routine. Use the Navigation Spec to go to the **CorrectValues** field from it:

```
> Spellchecker: Goto CorrectValues
```

Make **CorrectValues** a one-character hidden text field, since it's used only to hold a program. Make it inaccessible except by the Spellcheck routine.

## Programming

The following programming assumes that five description fields named **Desc1** through **Desc5** can be spellchecked.

```
SpellcheckButton:
<
Spellchecker =
"~0" + " " + Desc1 + "
 ~1" + " " + Desc2 + "
 ~2" + " " + Desc3 + "
 ~3" + " " + Desc4 + "
 ~4" + " " + Desc5 + "
 ~5";
Goto Spellchecker
```

This program copies the text from the description fields to **Spellchecker**. Each value is enclosed by numbered markers so the spellchecked values can be returned to their fields. I use the tilde (~) for my marker. You can use any character as long as it won't appear in the descriptions themselves. The hard carriage return at the end of each line places each value on a separate line.

```
Spellchecker:
< @Macro("Spellcheck")
```

On entering the Spellchecker field, the following *Spellcheck* macro expands the field and presses Shift-F1 to spellcheck the collected values:

```
<begdef><nokey><name>"Spellcheck"<vidoff><f6><capsf1><enddef>

CorrectValues:
<
Desc1 = @Mid(Spellchecker,@Instr(Spellchecker,"~0")+3,
(@Instr(Spellchecker,"~1")-1)-(@Instr(Spellchecker,"~0")+3));

Desc2 = @Mid(Spellchecker,@Instr(Spellchecker,"~1")+3,
(@Instr(Spellchecker,"~2")-1)-(@Instr(Spellchecker,"~1")+3));

Desc3 = @Mid(Spellchecker,@Instr(Spellchecker,"~2")+3,
(@Instr(Spellchecker,"~3")-1)-(@Instr(Spellchecker,"~2")+3));

Desc4 = @Mid(Spellchecker,@Instr(Spellchecker,"~3")+3,
(@Instr(Spellchecker,"~4")-1)-(@Instr(Spellchecker,"~3")+3));

Desc5 = @Mid(Spellchecker,@Instr(Spellchecker,"~4")+3,
(@Instr(Spellchecker,"~5")-1)-(@Instr(Spellchecker,"~4")+3));

Clear(Spellchecker); Goto Desc1
```

This program uses @Instr to determine the position of each individual value and return it, corrected, to the original description field. It then clears the Spellchecker field and goes to Desc1.

## Using the Spellchecker

After you've filled in the form, click on the Spellcheck button, and the program will spellcheck the specified fields and let you make corrections. When the spellcheck is complete, press F6 to close the field, then press Enter to update the corrected values and return to your form.

Typing over the numbered markers while in the Spellcheck field will interfere with the program's ability to return the corrected values to the original fields. To discourage this, I place instructions on the form where they can be seen while the Spellchecker field is open.

Erika Yoxall owns Hammer Data Systems in Garrettsville, Ohio, specializing in Q&A and Microsoft Access. Phone/Fax 330-527-4018, emy103@worldnet.att.net.

# Add Snap, Crackle 'n Pop to Your System

GORDON MEIGS

Want more *Snap* (speed) in your apps, particularly on a network? How about some *Crackle* (simplified data entry) and *Pop* (pop-up lists). Try some of these clever techniques in your own applications.

**MANY** new features in Q&A 5.0 are ideal for multiuser, multi-database applications. I'll cover some tricks we developed mainly to boost Q&A's mult-database performance on networks, though they can improve performance on standalone systems as well. To illustrate, I'll focus on a system that includes a database of companies (COMPANY.DTF), and one that tracks potential applicants (APPLICNT.DTF) for jobs at those companies. The techniques focus mainly on adding or viewing records in the busy Applicant database.

## Pop-up lists

When starting new Applicant record, a pop-up selection list prompts the user to place his or her initials in the first field. (This is a non-password protected system. Otherwise, you could use a statement such as *< #7: If @Add then #7 = @Userid*.) The program that displays the user initials list look like this:

```
< #7: If @Add then
  { Userselect(@Insert("Userlist"), #7) }; Cnext
```

Here, @Insert returns the ASCII (or Q&A Write) disk file named Userlist to the Userselect command, which in turn displays it as a selection list. @Insert looks for the file in the default Q&A documents directory. If it isn't there, you have to supply the full path. This @Insert/Userselect combination offers several benefits:

- You don't need to press Alt-F7 for list of restricted values (a Q&A 4.0 technique). The list is automatically displayed when you enter the field.
- The Userlist file can be edited anytime in Write; whereas to update Restricted Values requires exclusive use of the database (no one else can be using it). Likewise, if you were to use something like *{Userselect("JF;LF;EF;WH;GM", #7)}*, you'd need exclusive access to the database to update the program.
- The Userlist file can be used by any program in any database. You maintain one file in one place, and no matter where it's inserted it will aways contain the same list entries.

## Shell out to add a record to another database

In the system I'm discussing, every Applicant record is linked to the Company database using a unique company ID field named **CompKey**. Since it's hard to remember a company's ID, the Applicant database displays a 78-character-wide pick-list of full company names and cities. (See the sidebar, "Creating Super-Wide Picklists.")

But what if you have a new company? How do you add it to the list without abandoning the incomplete applicant record? Take a look at this CompKey program:

```
> #80: #80 = @Lt(#80,5);
If @XLu("Company", CompKey,"ID", "Company") <> #85 then
   { #85=@XLu("Company", CompKey, "ID", "Company") };
If #65="" then { #65 = @XLu("Company", CompKey, "ID",
"Phone") };

If #80= "" then { If @Askuser("Do you want to add","a new
company?","") then {#85= "Press Tab to lookup newly added
Company"; @Macro("Add Company") }}
```

When the cursor enters the CompKey field, a Navigation Spec statement displays the @Msg message, *Press Alt-C for Company List menu; Leave BLANK to add a new company*. The program's first section executes if you've entered a company ID (or selected it from the picklist). On the other hand, if you leave the CompKey field empty, the program displays an *@Askuser* prompt to add a new company. If you answer No, you move on. If you answer Yes, then the company name field (#85) is filled with a message, and the *Add Company* macro is executed. The macro presses Alt-F9 (to display the External Programs menu), and selects *5*. The menu option for that External Program is *5 Add Company* (Utilities / External Programs). It launches a second copy of Q&A with this command line:

```
QA -ALMacros2.asc -m8
```

The *-AL* autoload switch loads the macro file named *Macros2.asc* and runs its Alt-8 (*-m8*) macro, which selects File / Add Data / Company.dtf, putting you in position to add a new company record. A program in Company.dtf saves the new company ID to a disk file named *Compid*. When the new company has been added, you exit the second copy of Q&A, returning to the applicant's record to find the cursor in the Company Name field with the message, *Press Tab to lookup newly added Company*. Tab out of this field, and the program inserts the CompID in the CompKey field:

```
#80 = @Insert("Compid")
```

Using the new CompKey value, the information from the newly added company record is then looked up and placed in the corresponding Applicant record's fields.

## Display a some information from another file

The Applicant database has a Keyword field that contains one or more applicant skill codes. Since you might not

remember that a "GIS" skill code stands for *Geographical Information Systems*, you'd have to exit the Applicant database and run a manual search on SKILLS.DTF.

In this case, though, since SKILLS.DTF contains only two fields—the abbreviated keyword code and its full description—you can "Query a Code" using a **Query a Code?** field. This field displays a custom help screen to remind you how to perform the query, then displays the ASCII text file SKILLS.CDS in a Userselect list. Once a selection is made, you press Tab, the program looks up the code in SKILLS.DTF, and returns its full description in an @Msg message at the bottom of the screen, like this: *The full name is Geographical Information Systems.*

Here's the programming that does it:

```
> #153: If #153 = "Y" then {
Usl(@Insert("skills.cds"),#153) };
If #153 <> "" then { @Msg("The full name is "
+ @XLu("Skills", #153, "Code", "Full name"));
  #153 = "" }; Goto #160
```

You could use similar programming to look up a customer's mailing address, phone number, or whatever.

### Another handy pop-up list technique

This one is similar to the skills code lookup, but here we wanted a "live" look at a list from another database. (Skills inserts a predefined ASCII file of skill codes).

Each applicant has a job function such as Project Manager or Marketing Manager. Originally, we had these in a one-field database that allowed the client to print lists, add and delete descriptions, prevent multiple entries for essentially the same job function, and so forth, and we used an XUserselect statement to display the list in the Applicants database. This was excruciatingly slow, particularly across a network. We found a supplementary Function database much faster.

Function.dtf contains one record with three fields. The first is a read-only, Speedy, labelless field with *1* in it. (It's the one record always looked up from Applicants.)

The second field (#10) is used to add new job functions. Since the list can be lengthy, an on-field-exit program checks to see if the job function just entered is already in the third field, **All Functions**. If it's there, nothing is added, field #10 is blanked, and the cursor returns to field #10 for another try. If field #10's entry isn't in the All Functions field, the program adds it to the end of the field, with a space/semicolon separator.

Here's field #10's program:

```
> #10: If #10 <> "" then { If @Instr(#20, #10 +" ") > 0
Then { #10 = ""; Goto #10 }
Else { #20 = #20 +" ;" + #10; #10 = "";
If @Lt(#20, 1) = ";" Then
#20 = @Replfir(#20, ";", ""); Goto #10 } }
```

The **All Functions** field contains an on-record-entry program that displays an @Msgbox message that says, *You can't add records to this file*. Then exits to the File Menu. Here's the program:

```
#20: If @Add Then
{ @Msgbox("You can't add","records to this file","");
@Exit }
```

Remember, you can't add this program until *after* you've added the one record to the database.

Because a Q&A database field can store up to 32K, the All Functions field can contain lots and lots of entries. There's a simple Keyword report in the Function database that prints the contents of the All Function field to screen, providing an alphabetical list of all job functions in the database.

Now, back to the applicant database. The program that displays the job function selection list doesn't make use of the earlier slow XUserselect command. Instead, it performs a fast XLookup to the *one* record in the Function database, and returns the contents of the *one* field containing the entire list of functions.

The list is displayed courtesy of a Userselect command, with the list's first entry reading *--Function --* to let the user know which field they're in and what they're expected to select. Here's the complete on-field-entry program:

```
< #145: If #145=""  Then {
Usl(" — Functions —;" + (@XLu("Function", 1,
    "Record", "All Functions"), #145) };
If #145 = "— Functions —" then { #145 = ""; Goto #145 };
Goto #147
```

These techniques cover a lot of ground. Choose one that looks like it would improve your system, get it working, then try the others. One of Q&A's great strengths is that you can easily add new capabilities as your needs and expertise expand.

Gordon Meigs is vice president and general manager of Professional Computer Technology Associates of Newtown, Pennsylvania. He teaches courses and does corporate training on Q&A, and has been designing and installing advanced Q&A business applications for more than nine years. 215-598-8440, 71023.356@compuserve.com.

## Creating Super-Wide Picklists

Creating 78-character-wide selection lists of company names and addresses can be done in Q&A 4.0 and 5.0. The techniques have been the subject of past *Quick Answer* articles. (See, for example, September 1991—"Other Windows," October 1993—"Create and Use Pop-Up Lists.") Here's how it works in this application.

In the CompKey field, you press Alt-C to to display a series of custom menus to select the first letter of the company name. The list is then inserted in the expanded field, where you can press F7 to search for a particular company, or scroll the list to find the one you want.

With the cursor on the correct company line, you execute another macro that grabs the CompKey, moves it to the beginning of the still-expanded field, deletes everything after it, closes the field editor, then presses Tab to leave the field. The lists of company names, addresses, and CompKeys are created in "batch" mode using a *Create Company Lists* macro you run routinely to print the alphabetical lists of companies to a series of disk files.

# @Help

EDITED BY T.J. SHUFLIN

## Adding an @XLookup Value to a Date

**I need to lookup a *number-of-days* value from an external database, then add it to a date in the current record. I've tried the following program, but can't get it to work:**

```
Original Date: #10   Allow: #20
Extended Date: <#40: #40 = #10 +
   @Num(@XLu("Datesret", #20, "Allow", "Value"))
```

*Ben Jones, via the Internet*

XLookups always return a *text value*, which @Num won't convert to a number. (All @Num does is strip out non-numeric characters.) Use @Tonumber instead, like this:

```
Extended Date: <#40: #40 = #10 +
  @Tonumber(@XLu("Datesret", #20, "Allow", "Value"))
```

## Check for Duplicate Values

**I have a database with two text fields, and a third field that combines the first two. This third field is Speedy/Unique (SU), so I expect Q&A to warn me if another record contains the same value—but I receive no such warning.**

*Rick Shaefer, Little Rock, Arkansas*

Q&A can't use the Speedy/Unique method to check for duplicate values because you're generating your combination value with a program, like this:

```
F1: #100    F2: #110
F3: < #120: #120 = #100 + #110
```

You'll need to use an @XLookup program like this one:

```
<#140: If @Add Then { If @XLu(@Fn,#100+#110,"F4","F4") = ""
 Then { @Msg(" "); #140 = #100 + #110 } Else { #140 = "";
 @Msg("Not a unique value"); goto #100 } }
```

When adding a new record, this program checks the current database (@Fn) for a duplicate combination value. If none, the first @Msg blocks Q&A's message (*The lookup file does not have the key value...*) and the combination value is placed in field #140. If a duplicate value *is* found, a message informs you of it, field #140 is blanked, and the cursor returns to the F1 field for another try.

## Corrupted QA.CFG File

**Since I installed Q&A 5.0 for DOS on my Pentium 133, I can't design a new database then design a report for it. When I save the report Specs, I get the message, *Problem! System Error, Please Try Again. Ref# DADA 1E09*, and a "smiley face" appears in the Print Offset Field. This *doesn't* occur if I design the database and reports in Q&A 4.0, then upgrade the database. But it always occurs if I try to change global options for crosstab reports. Any idea what's causing it?**

*Gary M. Gayda, Pittsburgh, Pennsylvania*

Tom Marcellus told me that something similar happened to him once and how he fixed it. By odd coincidence, the same thing recently happened to me, and his fix worked.

The problem is most likely a corrupted QA.CFG file, which you'll have to delete. But before you do, make a note of your installed printers, and any custom global settings you've made, because you'll have to reenter them. I used my QAINDEX database to find four back issues that mentioned the QA.CFG file and what Q&A stores in it: May 1992, page 9; October 1992, page 15; March 1994, page 8; and June 1996, page 4. They'll help you determine what you'll have to redo.

When you delete your QA.CFG file (or move it to another location or rename it), then start Q&A, it won't find the file and will automatically create a new one. You'll see the prompt, *Select Utilities to customize Q&A for your Printer*. Reenter all your custom global settings, and reinstall your printer selections. Be sure to do all of this before you run any macros. You should now be able to design and save reports in Q&A 5.0 the usual way.

You can copy a healthy QA.CFG file to QACFG.BAK so you'll have a good backup.

## Getting Totals from a Line-Item Database

**We use a Q&A 4.0 DOS database to track expense payments to vendors, and a crosstab report to compile the expenses by accounting codes. The database has five lines, and each line has a field for an account code and expense amount. The problem is that the same account code can occur on any of the five lines in any record. To find total expenses by account code we have to run five crosstabs, then manually add all the amounts to get totals. Is there an easier way?**

*Dale Brooks, Lincoln, Illinois*

*Continues on page 12*

**Stumped?** Send your Q&A questions 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, phone number, and 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 aren't possible.

# Forms Within Forms, Part Deux

TOM MARCELLUS

IN last month's Part 1, we prepped two databases—Invoice and Notes—and designed subforms for them in a Customer database. I'll assume you've laid those foundations. I also described how the subforms work once programmed, and how you can obtain the subform application ready-to-use from Marble Publications.

## Programming the subforms

Following are the programs for the Notes and Invoice Activity subform fields, along with brief explanations. (The programming for both subforms amounts to just over 7K.) Refer to Figure 3 and Table 1 in Part 1. The programs assume a **Cust ID** field for the customer ID number, a **Company** field for the company name, as well as the Key fields in the Notes and Invoice databases I described last month. The Customer, Notes, and Invoice databases are named **Custs2**, **Notes2**, and **Inv2**:

**2. NoteUp**—Click on this button field to scroll up the notes (from older to more recent notes) until the most recent note is displayed.

```
< @Play("Sound", "100, 5");
  If Note = "" Then Chome;
  If @Len(NoteViewed) = 16 then
  { @Msg("This is the most recent Note for " + Company);
  @Play("Click"); Goto Hold };
  NoteViewed = @Replace(NoteViewed,
  @Right(NoteViewed, 16), "");
  NoteCode = @Right(NoteViewed, 16);
  XLookup("Notes2", NoteCode, "NoteCode", "Note", Note);
  NoteDate = @Left(Note, @Instr(Note, " "));
  Note = @Del(Note, 1, @Instr(Note, " "));
  @Msg(Note);  Goto Hold
```

**3. NoteBar**—This click-on button field activates and deactivates the Notes subform. If the subform is active, it clears the subform fields and returns to the main part of the form. If the subform is inactive, it activates the subform and displays the most recent Notes record for the customer.

```
< If Cust ID = "" Then
  { @Msg("Customer ID missing!"); Chome };

  If NoteUp <> "" Then {
  Clear(Note, NoteDate, NoteDel, NoteUp, NoteDn, NoteAdd,
  NoteSearch, NoteCode, NoteTemp, NoteViewed, Hold);
  @Color(NoteDel,7,7); @Color(NoteAdd,7,7);
  @Color(NoteSearch,7,7); Chome };
  NoteUp = "▲▲"; @Color(NoteUp,0,7);
  NoteDn = "▼▼"; @Color(NoteDn,0,7);
  @Play("Sound", "100, 5");
  NoteAdd = "Add Note"; @Color(NoteAdd,15,4);
  NoteDel = "Delete"; @Color(NoteDel,15,4);
  NoteSearch = "Search"; @Color(NoteSearch,15,4);
  NoteCode =
  @XLookupR("Notes2", Cust ID
    + @Text(10 - @Len(Cust ID), "0")
    + @Str(999999), "NoteCode", "NoteCode");
  If @Left(NoteCode, @Len(Cust ID)) <> Cust ID
  Then
  { @Msg("No Notes for this customer"); @Play("Click");
    Goto NoteBar }
  Else
  { XLookup("Notes2", NoteCode, "NoteCode", "Note",
    Note);
  NoteDate = @Left(Note, @Instr(Note, " "));
  Note = @Del(Note, 1, @Instr(Note, " "));
  @Msg(Note);
  NoteViewed = NoteCode };

  If NoteTemp = "Search" then
  { Hold = "+"; Goto NoteTemp } Else Goto Hold
```

**4. NoteDn**—Click on this button field to scroll down the notes (from more recent to older notes) until the oldest note appears.

```
< @Play("Sound", "100, 5");
  If Note = "" Then Chome;
  NoteCode = @Left(NoteCode, 10) +
  @Right("000000" + @Str(@Right(NoteCode, 6) -1), 6);
  NoteCode =
  @XLookupR("Notes2", NoteCode, "NoteCode", "NoteCode");
  If @Left(NoteCode, @Len(Cust ID)) = Cust ID
  Then { Xlookup("Notes2", NoteCode, "NoteCode",
    "Note", Note);
  NoteDate = @Left(Note, @Instr(Note, " "));
  Note = @Del(Note, 1, @Instr(Note, " "));
  @Msg(Note);
  NoteViewed = NoteViewed + NoteCode; }
  Else { NoteCode = @Right(NoteViewed, 16);
  @Msg("This is the oldest Note for " + Company);
  If NoteTemp = "Search" and NoteString <> "" Then
  @Msg("Didn't find another note with `" +
    NoteString + "'");
  @Play("Click"); Clear(Hold, NoteString, NoteTemp) };

  If NoteTemp = "Search" Then Goto NoteTemp Else Goto Hold
```

**5. NoteDate**

```
< If NoteUp = "" Then CHome
```

**6. NoteString**—When you click on the Search button, the program prompts you to type the search string (a word or phrase) in this field to find the most recent note that contains the string. (Navigation Spec: *< If NoteDn = "" Then Chome.*)

```
> If NoteTemp = "Search" and NoteString <> ""
     and Hold <> "+"
  Then { Clear(NoteUp); Goto NoteBar };

  If NoteString <> "" and Hold = "+"
  Then Goto NoteDn
  Else { Clear(NoteTemp, Hold); Goto Hold }
```

**7. Note**—This field displays the first 34 characters of each note as you scroll through them. (The message line displays the first 80 characters.) If you click on the Add Note button, the program prompts you type the new note here, and you can press F6 for more room. The NoteAdd program then creates a new Notes record via @Shell, then posts the new note and date to it. (Navigation Spec: *< If NoteDn = "" Then Chome.*)

```
> If Note = "" Then Goto Hold;
  If NoteTemp = "Add" Then Goto NoteAdd;
  If NoteString <> "" Then
  If @Askuser("Find next note with",
  "`" + NoteString + "'?","") Then
  Goto NoteDn Else { Clear(NoteString, NoteTemp, Hold);
  Goto Hold } Else Goto Hold
```

**8. NoteTemp**—Used to store temporary values and when performing note searches.

```
< If Note = "" Then Chome;
  If NoteString = "" Then Goto Hold;

  If @Instr(Note, NoteString) = 0 Then Goto NoteDn
  Else
  { @Play("Sound","494,100"); @Play("Sound","415,100");
  @Msg("FOUND! Press Enter to find next.");
  Goto Note }
```

**9. NoteSearch**—Click on this button field to search for a note containing a particular word or phrase.

```
< If NoteUp = "" then CHome;
  Clear(NoteString);
  NoteTemp = "Search";
  @Msg("Type search string then press Enter.");
  Goto NoteString
```

**10. NoteAdd**—Click on this button field to compose and add a new note to the Notes database. (Change the drive and path in the @Shell command if different.) Requires the following Alt-1 autostart macro to add the record then return to the subform:

```
<begdef><alt1><name>"<caps,>alt1<caps.>"<vidoff>
fanotes2<enter><capsf10><esc>x<enddef>
```

```
< If NoteUp = "" then CHome;
  If NoteTemp = "Add" and Note = "" Then {
  Clear(NoteTemp); Goto NoteAdd };
  If NoteTemp = "Add" and Note <> "" Then {
  If @Askuser("Add this note",
    "for " + Company,
    "to Notes database?")
  Then {
  @Play("sound","196,75"); @Play("sound","256,75");
  @Play("sound","329,75"); @Play("sound","392,150");
  Hold = @Shell("D:\QA\QA1.EXE -m1");
  NoteCode =
  @XLookupR("Notes2", 999999, "RecordNumber",
    "RecordNumber");

  XPost("Notes2", NoteCode, "RecordNumber",
  Cust ID + @Text(10 - @Len(Cust ID), "0") +
  @Right("000000" + @Str(NoteCode), 6), "NoteCode") ;

  XPost("Notes2", NoteCode, "RecordNumber",
  @Str(@Month(NoteDate)) + "-"
  + @Str(@Dom(NoteDate))
  + "-" + @Right(@Year(NoteDate), 2)
  + " " + Note, "Note");

  If Hold = "0" Then
  { @Msg("Note added for " + Company);
  @Play("Sound","494,100"); @Play("Sound","415,100") ;
  Goto NoteBar } } };

  Clear(Note, NoteCode, NoteViewed);
  NoteDate = @Date; NoteTemp = "Add";
  @Msg("Type note, then press Enter to save it.");
  Goto Note
```

**11. NoteDel**—Clicking on this button field marks the displayed note *DELETED* in the Notes database. It doesn't delete the note, but prevents it from being displayed in the subform.

```
< If NoteUp = "" Then Chome;
  If Note = "" then Goto NoteBar;
  If @Askuser("Delete this note?","","")
  Then {
  XPost("Notes2", NoteCode, "NoteCode", "DELETED "
    + NoteCode, "NoteCode");
  @Msg("Note DELETED"); Goto NoteBar }
  Else Goto Hold
```

**15. InvUp**—This click-on button field scrolls the invoices from older to more recent.

```
< If InvUp = "" Then Chome;
  @Play("Sound", "100, 5");
  If @Len(InvViewed) = 16 then
  { @Msg("This is the latest Invoice for " + Company);
  @Play("Click"); Goto Hold };

  InvViewed = @Replace(InvViewed,
  @Right(InvViewed, 16), "");
  InvCode = @Right(InvViewed, 16);

  InvInfo = @XLookup("Inv2", InvCode, "TransCode",
    "TransData");
  InvPaid = @XLookup("Inv2", InvCode, "TransCode",
    "Paid"); Goto Hold
```

**16. InvBar—**This click-on button field activates and deactivates the Invoices subform. If the subform isn't active, the program initializes it and displays the most recent invoice. If the subform is already active, the program clears the subform fields and returns to the main part of the form.

```
< If Cust ID = "" Then
  { @Msg("Customer ID is missing!"); Chome };
  If InvCols = "" Then {
  InvCols = "Inv #    Date      Amt      Paid";
  @Color(InvCols,15,7);
  InvUp = "▲▲"; @Color(InvUp,0,7);
  InvDn = "▼▼"; @Color(InvDn,0,7);
  @Play("Sound", "100, 5");
  InvCode =
  @XLookupR("Inv2", Cust ID + @Text(10 - @Len(Cust ID),
    "0") + @Str(999999), "TransCode", "TransCode");

  If @Left(InvCode, @Len(Cust ID)) <> Cust ID
  Then{ @Msg("No Invoices for " + Company);
  @Play("Click"); Clear(InvCode, InvCols,
    InvDn, InvUp) }
  Else
  { InvInfo = @XLookup("Inv2", InvCode, "TransCode",
    "TransData");
  InvPaid = @XLookup("Inv2", InvCode, "TransCode",
    "Paid"); InvViewed = InvCode } ; Goto Hold }

  Else Clear(InvCols, InvUp, InvDn, InvInfo, InvPaid,
    InvCode, InvViewed); Chome
```

**17. InvDn**—This click-on button field scrolls the invoices down, from the more recent to the oldest.

```
< If InvUp = "" Then Chome;
  @Play("Sound", "100, 5");
  InvCode = @Left(InvCode, 10) +
  @Right("000000" + @Str(@Right(InvCode, 6) -1), 6);

  InvCode =
  @XLookupR("Inv2", InvCode, "TransCode", "TransCode");

  If @Left(InvCode, @Len(Cust ID)) = Cust ID
  Then {InvInfo = @XLookup("Inv2", InvCode, "TransCode",
    "TransData");
  InvPaid = @XLookup("Inv2", InvCode, "TransCode",
    "Paid");
  InvViewed = InvViewed + InvCode }
  Else { InvCode = @Right(InvViewed, 16);
    @Msg("This is the oldest Invoice for " + Company);
    @Play("Click") }; Goto Hold
```

**18. InvCols**

```
< If InvUp = "" then CHome
```

**19. InvInfo**

```
< If InvUp = "" Then CHome
```

**20. InvPaid**—If this field is blank (doesn't contain a paid date), clicking on it lets you mark the external invoice record as *Paid* on the current date.

```
< If InvUp = "" Then Chome;
  If InvPaid <> "" Then Goto Hold;
  If InvPaid = "" Then {
  If @Askuser("Want to mark this invoice",
    "`PAID'?","")
  Then { InvPaid = @Date;
  XPost("Inv2", InvCode, "TransCode", @Date, "Paid");
  Goto Hold } Else Goto Hold }
```

*Tom Marcellus is editor of* The Quick Answer *and author of* PC World Q&A Bible *(IDG Books). Many of his databases featured in* The Quick Answer *are available from Marble Publications.*

The Quick Answer

*The Program Spec*

# Using Typecast Functions

JEFF NITKA

The Q&A *Application Programming Tools Manual* calls them *Typecast functions*. You use them to convert one data type to another. This month, Jeff shows a few examples where he uses them, and explains why.

Q&A comes with six typecast functions—@Tonumber, @Str, @Todate, @Totime, @Toyesno, and @Tomoney. They take one data type and convert it, where possible, to a number, text, date, time, yes/no, or money data type. Most of us use @Str often enough, but not the others.

## Perform an action from a selection list

Suppose you have a database with a **Choice** field. When you move to Choice, it's program prompts you to select one of four mathematical list actions (@Sum, @Avg, @Var, or @Std) to perform on 10 fields (Field1..Field10). A program for such an operation might look like this:

```
< Choice =
@Userselect("1-Totals,2-Averages,3-Std Dev,4-Variances");
if   Choice = ""
then {@Msgbox("Make a Choice!","","",""); Goto Choice} else
if   @Lt(Choice,1) = "1"
then Result = @Sum( Field1..Field10 ) else
if   @Lt(Choice,1) = "2"
then Result = @Ave( Field1..Field10 ) else
if   @Lt(Choice,1) = "3"
then Result = @Std( Field1..Field10 ) else
     Result = @Var( Field1..Field10 )
```

You could reduce this program to one line with the help of Gosub, the @Field field reference function, and the @Tonumber typecast function:

```
Gosub @Field( @ToNumber( @Left(
@Userselect("1-Total,2-Average,3-Std Dev,4-Variance"),1)))
```

Here, @Tonumber converts the leftmost character of your selection to a number, to which the Gosub transfers control. (For this to work, you'd need four fields to store the four list function equations.) Although this example might not require the optional shorter method, it might come in handy for large selection lists involving more complex programs.

## Reformatting dates and times

I have an application that stamps the date and time on an order when it's invoiced. Because I access these values in an external database (using XLookup), I needed to place both values in the same text field. Initially, I used the following statement:

```
Stamp = "Invoiced: " + @Str(@Date)+" at "+@Str(@Time)+".";
```

But this doesn't format the date and time in a familiar way. For example, on May 5, 1997 at 2:30 pm, Stamp would contain "Invoiced: 1997/05/05 at 14:30."

This was no surprise because using @Str for the conversion returns the date and time as Q&A stores them internally. To make Stamps more readable, I changed the program as follows:

```
Stamp = "Invoiced: " + @Str(@Month(@Date)) + "/" +
                       @Str(@DOM(@Date))   + "/" +
                       @Str(@Year(@Date)) + " at";
if   @ToNumber(@Time) = 0
then Stamp = Stamp + " 12:00 M." else
if   @ToNumber(@Time) = 720
then Stamp = Stamp + " 12:00 N." else
if   @ToNumber(@Time) < 720
then Stamp = Stamp + " " + @Str(@Time) + " AM."
else Stamp = Stamp + " " +
     @Str(@ToTime(@ToNumber(@Time)-720)) + " PM."
```

Reformatting the date makes use of the Q&A's date functions @Month(D), @DOM(D), and @Year(D). Each function extracts the numeric month, day, and year, respectively, from the date value D.

Q&A doesn't provide time extracting functions, so I had to do this another way. The key to formatting time values is to convert the time to the current number of minutes in the day using @Tonumber(@Time).

The special cases of 12 midnight and 12 noon are considered in the first two If-Then-Else statements. (12 midnight is the beginning of a new day, thus 0 minutes have elapsed. At 12 noon, 12 hours have elapsed in the day, which is 12 hours * 60 minutes/hour = 720 minutes.)

If the current time isn't 12 noon or 12 midnight, the program must determine whether it's before or after 12 noon. If it's before 12 noon, then all that's needed is to attach "am" to the current time. However, if the time is after 12 noon, the program must subtract 720 minutes (12 hours) from the current time. Using @Totime, the result is reconverted to a time value in the familiar format. All that's needed finally is the "pm" to denote the correct part of the day.

Using the program in the previous example, Stamp would now contain *Invoiced: 5/5/1997 at 2:30 pm*.

*Jeff Nitka works for a chemical manufacturer and develops Q&A applications part-time for Epoch Software, 908-874-3989. Jeff is the author of the Program Evaluator (a Q&A program debugging utility) and FaxMan (a Q&A faxing database), both of which are available from Marble Publications.*

## @Help Continued from page 8

Creating reports from line-item databases like yours is always tough. Tom Marcellus' article in the August 1990 *Quick Answer* ("Generating Invoice Line Item Totals") describes a technique that might get the result you're after. Or, you could redesign the database to have a single record for each Account Code and Amount. See Jeff Nitka's article in the May 1995 issue ("Create and Print Invoices with Unlimited Line Items.") The report samples you sent show that you're only interested in getting account code totals (not sorted by vendor or anything like that), and the following should help you do this:

1. Backup your database. Create a new database named Coderpt.dtf with just two fields—Code and Amount. Make Code a text field, and Amount a money field.
2. From the Main menu, choose File / Copy / Copy Selected Records. You'll be copying records from your original database to the new Coderpt database.
3. At the Retrieve Spec, select the date range (if pertinent) of the records to copy. Then, type /= (not empty) in the Code field on the *first* line, and press F10.
4. At the Merge Spec, type *1* in the *first line's* Code field, and *2* in the corresponding Amount field. Press F10, and Q&A will copy the selected records to Coderpt.
5. Repeat steps 2 through 4 four more times. Each time, at the Retrieve Spec, type the /= in the next Code field down. And at the Merge Spec, type the *1* and *2* in the Code and Amount fields on *that same* line.

When you're done, you'll have a record in Coderpt for each Code and Amount line from each record of your original database, and you can create a columnar report to get the information you need. Here's a Column/Sort Spec that will give you subcounts and subtotals by account code, as well as the grand count and grand total:

```
Code: 1, AS
Amount: 2, SC, C, ST, T
```

To suppress printing the individual values, set Print Totals Only to *yes* at the Report Print Options screen.

Before you perform this procedure again (each month, for example) be sure to remove all the records from Coderpt. Also, be sure your Retrieve Spec selects just the records for the pertinent time period, unless you want to generate a report on the entire database.

Q&A "power users" would create a macro to automate a task like this. If you do, you should first create, name, and save a separate Retrieve and Merge Spec for each of the five "passes." (Macros that call Saved Specs rather than "tabbing" to fields and "typing" codes are more reliable.)

Providing a way for the macro to select a *date range* of records to include in the five copy "passes" is trickier. One way is to program each Retrieve Spec to lookup the starting and ending dates from *another* supplementary database with a Speedy Key field and date fields for the starting and ending dates. That database (Dates.dtf in this example) would contain one record where you specify the date range like this:

```
  Key: 1          (Speedy field)
Start: 6-1-97     (Date field)
  End: 6-30-97    (Date field)
```

This way, you can use a Retrieve Spec expression like the following in the original database's Date field:

```
>={@XLu("Dates", "1", "Key", "Start")}..
<={@XLu("Dates", "1", "Key", "End")}
```

In this case, the expression would retrieve just the records dated June 1, 1997 through June 30, 1997.

T.J. Shuflin is a forensic chemist and crime lab director in Alexandria, Louisiana. He served as the National Q&A User Group's first president in 1991-92, and remains active by maintaining the group's Web site at http://www.qaug.com.

MARBLE PUBLICATIONS

**PO Box 9034**
**Gaithersburg, MD 20898-9034**

Periodicals
Postage
**PAID**
at Rockville, MD