January 1995

THE Quick Answer

The independent monthly guide to Q&A expertise

Automate Your Check Writing and Printing

Does your company write 100 or more checks each month? It only takes a few hours to create a database that can revolutionize the way you pay bills.

By Tom Marcellus

MORE companies are turning to the computers and printers they already own to automate their check writing and printing. And why not? Cutting checks from the keyboard instead of scratching them out by hand has advantages, especially when you write lots of checks:

- You spend less time preparing and printing checks because the process is automated. You can prepare and print 20 checks in the time it takes to write five checks the old way.
- Tracking checks individually and by groups is a snap. You can search them by payee, date range, account number, check number, or by any other criteria you build into the database.
- You can produce informative, labor-saving reports that show disbursement breakdowns, compile payroll data, help you with bank reconciliations, and so forth.

It's not difficult to design a Q&A database that will write, print, and track checks. The only tough part is converting the dollar amounts to their longhand text equivalents so that when you enter $236.44 in the check amount field, for example, Q&A prints this on the Dollars line:

```
TWO HUNDRED THIRTY SIX AND 44/100 DOLLARS********
```

Q&A doesn't have a built-in function that converts numbers or money values to text equivalents—you have to write a programming routine to do it. I'll show you such a routine and how to add it to the database. All you have to do is follow my form guidelines (see Figures 1 and 2), enter a few programming statements into the Program Spec, and type 27 entries into a Lookup table (see Table 1). Once these elements are in place, your check writing database is ready for action and will convert your check dollar amounts—as high as $999,999.99—to their text equivalents automatically.

Continues on page 3

CONTENTS

Automate Your Check Writing and Printing, *Tom Marcellus* 1

README.1ST, *Tom Marcellus* 2

Calculate Future Dates, *Deborah Burkholder* 7

@HELP, *Dave Reid* 10

How to Restrict a Field Entry's Length, *Gordon Meigs* 12

Readers Write 15

*Quick*Tips 9, 13, 14, 16

PINNACLE

The Quick Answer

Editor in Chief .... Dian Schaffhauser
Managing Editor .... Roland Winkler
Editor .... Tom Marcellus
Production Editor .... Paul Gould
Copy Editor .... Laurie Moloney
Publisher .... Susan Jameson Harker
Circulation Director .... Sharon Whiting

The Quick Answer (ISSN1052-3820) is published monthly (12 times per year) by Pinnacle Publishing, Inc., 18000 72nd Avenue South, Kent, WA 98032. Cost of domestic subscriptions: 12 issues, $79.00; 24 issues, $142.00. Outside the U.S.: 12 issues, $99.00; 24 issues, $172.00. Single copy price: $10.00; 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; multiple-copy discounts are available.

For European newsletter orders, contact: Tomalin Associates, The Old Bakery, High Street, Great Bardfield, Essex CM7 4RF, United Kingdom
Phone: 371 811299; fax: 371 811283
12 issues: £69; 24 issues: £109

Second-class postage paid at Kent, WA and additional entry points. USPS#006941. POSTMASTER: Send address changes to *The Quick Answer*, PO Box 888, Kent, WA 98035-0888.

Address editorial correspondence, @HELP questions, or requests for special permission to: The Editor, *The Quick Answer*, PO Box 888, Kent, WA 98035-0888. Or leave CompuServe messages at 72600,140. We will publish only those questions we feel are of general reader interest; individual responses are not possible. For subscription information, back-issue requests, and bulk orders, call our Customer Service department at 800-788-1900 or 206-251-1900.

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

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. Pinnacle Publishing, 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 Pinnacle Publishing, Inc.

PINNACLE

Volume 6, Issue 1

README.1ST

It’s a new year and an appropriate time to say thank you for making *The Quick Answer* a part of your computing life. I recall (perhaps you do, too) that day in June 1990 when the first issue arrived. Since then, we’ve worked to bring you a constant supply of more and better ways to use Q&A. I think we’ve succeeded, and I hope you’ll continue to look to *The Quick Answer* for the solutions you need.

I mentioned last month that there are exciting things going on in Europe, where the installed Q&A base is huge and new German language releases of both Q&A/DOS and Q&A/Windows are in the works. In the months to come, the optimist in me hopes to be able to report that English language upgrades will be available as well.

Automating your check writing and printing is a smart way to keep on top of where your money is going. A check writing database doesn’t just simplify a tedious task, it helps you track payments by payee, date range, or general ledger account number, and makes informative disbursement, payroll, and bank reconciliation reports just a few keystrokes away. I’ll show you how to create a check writer and add the custom features you need to keep on top of your payables.

Calculating a future date by adding the number of days from a starting date is child’s play, but what if you need to add months? Months have different numbers of days, and though you can get close, the exact date will prove elusive. **Deborah Burkholder** shows you how to design a future date calculator into any database so you can add months and get the exact future date.

Gordon Meigs brings you one of those nifty “who-would-have-thought-of-it?” solutions ideal for situations that demand precision merge printing. Instead of struggling with “data fit” in a subscription renewal form, Gordon controls the data when it’s typed into the database. This way, he achieves a precision fit without the risk of truncating any information.

Several readers sent improvements to techniques from recent issues. On page 15, you’ll find out how to conditionally join a subdocument to a master document and create a credit card validation routine that requires only four fields.

This month’s tips show you how to match unmatching key values in two related databases, add a handy comments field to your form, perform XLookups without a dedicated key field, search on values that violate the field format, handle QABACKUP restore failures, and why you need to exercise caution when using the @Add context function.

Dave Reid answers reader questions on how to calculate discounts, work around field naming restrictions, make custom menus more reliable, use tabs to align merge document columns, and print a birthday card for everyone born in a particular month, regardless of the year.

Tom Marcellus
Editor

Automate

Continued from page 1

A chip off the old database

The personal finance database (PERSFIN.DTF) that comes bundled with Q&A/DOS 4.0 contains a check writing template (page 2 of the PERSFIN.DTF form) along with programming and Lookup table data to convert dollar amounts to their text equivalents.

However, PERSFIN.DTF is unwieldy. It consists of nine screens to handle a host of transactions in addition to check writing. It contains fields and programming for checking and savings account deposits and withdrawals, as well as credit card charges and payments. Though you might find PERSFIN.DTF useful to help manage your personal finances, it isn't a practical application for a company, nor is it appropriate as a check writing and tracking database.

I used the check writing portion of PERSFIN.DTF as a guide in creating CHECKS.DTF, a database designed exclusively to automate the writing, printing, and tracking of checks. CHECKS.DTF follows PERSFIN.DTF's programming logic to convert dollar amounts to their longhand equivalents, but I've made a number of programming changes and rearranged the Program Spec field IDs to make CHECKS.DTF easier for you to create and use.

CHECK.DTF form layout

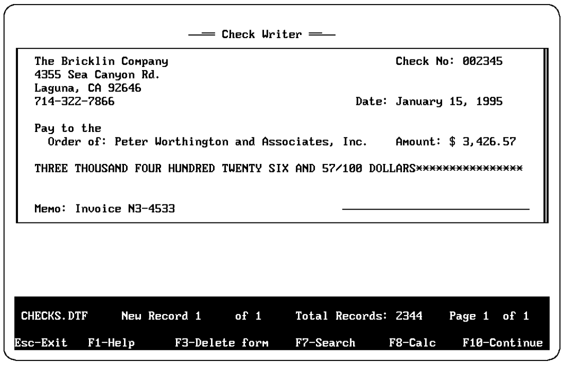
Figure 1 shows a completed check. The form was created in Q&A/DOS, but you can create an even more attractive form in Q&A/Windows. It includes text and fields for the information you'd expect to find on a check: check number, date, amount, memo, and a screen-wide field for the longhand equivalent of the check amount. You can add other fields as well. For example, you might want to include an account number field in the database, so you can run reports showing totals by general ledger account number.

I didn't include a check stub because they vary widely in form and functionality, but you can design a stub into CHECKS.DTF, along with fields to store an explanation or disbursement or deduction information.

CHECKS.DTF also requires a series of fields on the line below the check template. (See Figure 3.) These 11 read-only fields, which are only a few characters in length and don't include labels, temporarily store the results of the intermediate

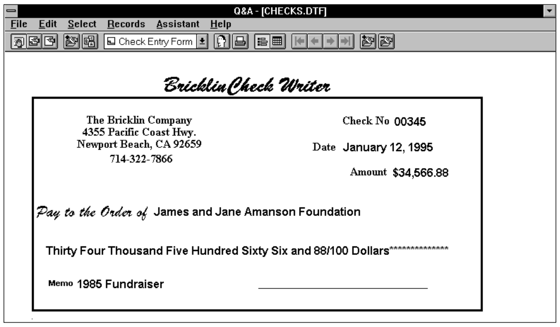
How to Obtain Computer Checks

Office supplies stores and stationery and forms suppliers such as Nebs, The Drawing Board, and Quill offer computer-compatible check styles to suit a variety of preferences and printers. You can get continuous multipart checks for printing on dot matrix printers, or checks designed for sheet-fed ink jet and laser printers. You can order your checks preprinted with your company's name, checking account number, and with a variety of stub schemes. These suppliers normally offer free samples of check styles so you can test compatibility with your form design and printer. Check with your bank to find out its requirements for computer-generated checks.



```
                      ══ Check Writer ══
 The Bricklin Company                              Check No: 002345
 4355 Sea Canyon Rd.
 Laguna, CA 92646
 714-322-7866                                Date: January 15, 1995

 Pay to the
   Order of: Peter Worthington and Associates, Inc.    Amount: $ 3,426.57

 THREE THOUSAND FOUR HUNDRED TWENTY SIX AND 57/100 DOLLARS****************


 Memo: Invoice N3-4533                         ____________________

CHECKS.DTF     New Record 1     of 1    Total Records: 2344    Page 1  of 1
Esc-Exit   F1-Help     F3-Delete form    F7-Search     F8-Calc    F10-Continue
```

Figure 1. *CHECKS.DTF form design in Q&A/DOS. Use Line Draw, fixed text, and field placement to make the form look realistic.*



Bricklin Check Writer

The Bricklin Company
4355 Pacific Coast Hwy.
Newport Beach, CA 92659
714-322-7866

Check No 00345

Date January 12, 1995

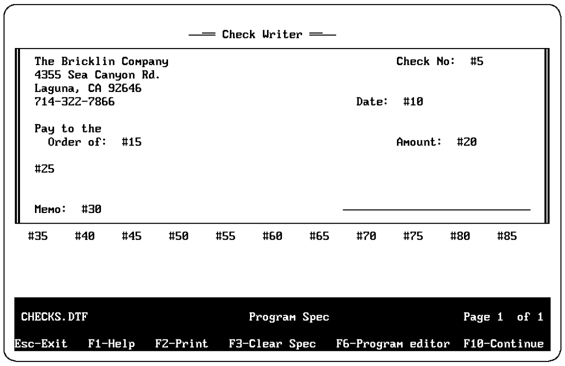
Amount $34,566.88

Pay to the Order of James and Jane Amanson Foundation

Thirty Four Thousand Five Hundred Sixty Six and 88/100 Dollars***************

Memo 1985 Fundraiser

Figure 2. *CHECKS.DTF form design in Q&A/Windows. Use fonts and other Q&A/Windows form layout tools to make your check entry form look just like the real thing.*



```
                     —═ Check Writer ═—
 The Bricklin Company                          Check No:  #5
 4355 Sea Canyon Rd.
 Laguna, CA 92646
 714-322-7866                               Date:  #10

 Pay to the
   Order of:  #15                              Amount:  #20

 #25


 Memo:  #30                             ____________________
 #35   #40   #45   #50   #55   #60   #65   #70   #75   #80   #85


CHECKS.DTF                  Program Spec               Page 1  of 1
Esc-Exit  F1-Help  F2-Print  F3-Clear Spec  F6-Program editor  F10-Continue
```

Figure 3. *CHECKS.DTF at the Program Spec with the field ID numbers in place.*

calculations required to convert the check dollar amount to its text equivalent. When placing these 11 fields on the form, use the left angle bracket (<) instead of the colon to delineate the beginning of the information blank portion of the field. This way, these fields won't show on the form during data entry.

After you've created and saved your form layout, use the Format Spec to enter the field formatting codes. Table 2's last column shows the format specs for CHECKS.DTF's fields.

Fill in the Lookup table

With the form design finished, it's time to fill in the Lookup table with the longhand equivalents of the Amount field. Table 1 shows the numbers to enter in the table's Key column, along with the corresponding text equivalents for the adjacent text column.

Table 1. *Lookup table text equivalencies. Type the numbers in the Key column and their corresponding text equivalents in the adjacent column.*

Key	Column 1	Key	Column 1	Key	Column 1
1	ONE	10	TEN	19	NINETEEN
2	TWO	11	ELEVEN	20	TWENTY
3	THREE	12	TWELVE	30	THIRTY
4	FOUR	13	THIRTEEN	40	FORTY
5	FIVE	14	FOURTEEN	50	FIFTY
6	SIX	15	FIFTEEN	60	SIXTY
7	SEVEN	16	SIXTEEN	70	SEVENTY
8	EIGHT	17	SEVENTEEN	80	EIGHTY
9	NINE	18	EIGHTEEN	90	NINETY

Programming CHECKS.DTF

If you're using the PERSFIN.DTF database supplied with Q&A 4.0, I've included Table 2 to show the relationship between the Program Spec field ID numbers in CHECKS.DTF and the corresponding fields in the check writing part of PERSFIN.DTF. The Program Spec field ID numbers for this part of PERSFIN.DTF were out of order and difficult to follow. I assigned new field ID numbers to CHECKS.DTF (in increments of five) to make programming them easier.

Table 2. *PERSFIN.DTF and CHECKS.DTF Program Spec field ID comparison chart with field format specs. Fields #50 and #60 are formatted for numbers. Figure 2 shows the placement of these intermediate calculation fields on the form.*

	Program Spec	Field ID	
Field Label	**PERSFIN.DTF**	**CHECKS.DTF**	**Format**
Check No	#1	#5	T
Date	#2	#10	D
Pay to Order of	#3	#15	T
Amount	#4	#20	MC,JL
No Label (text equivalent of Amount field)	#103	#25	T
Memo	#5	#30	T
Intermediate calculation fields			
No label	#104	#35	T
No label	#105	#40	T
No label	#106	#45	T
No label	#107	#50	N
No label	#108	#55	T
No label	#109	#60	N
No label	#110	#65	T
No label	#120	#70	T
No label	#124	#75	T
No label	#125	#80	T
No label	#126	#85	T

Several fields in CHECKS require programming: #5 (Check No), #20 (Amount), #25 (the field that displays the text equivalent of Amount), and intermediate calculation fields #75, #80, and #85.

The program for field #5 (Check No) assigns the next six-digit check number in sequence (with leading zeros), plugs in the current date, and moves the cursor to the Pay to the Order of field. You can modify the following programming to suit your needs. To add it to your database, go to the Program Spec (File / Design a file / Program a file / Program form), and enter the programming statements into the pertinent fields as shown here, using Figure 3 as a guide to the field ID numbers:

```
<#5:
If #5="" then {#5= @Right("000000"+@Str
   (@Number),6);#10=@Date; goto #15}

>#20:
#35=""; #40=""; #45=""; #25="";
If @Instr(#20,".")=0
Then {#35=#20; #65=@Num(#35); #35=#65;
   #20=#20+".00"}
Else {#35=@Left(#20,@Instr(#20,".")-1);
```

```
    #65=@Num(#35); #35=#65;
    #40=@Mid(#20,@Instr(#20,".")+1,2)+"0"};
    #55=""; #70=""; #50=0; goto #25

<#25:
If #65<>0
Then {#50=#50+1; #45=@Right(#35,3);
    #35=@Left(#65,@Len(#65)-3); #65=#35;
If @Mid(#45,2,1) =""
Then #45="00"+#45;
If @Mid(#45,3,1)=""
Then #45="0" +#45; #70="";
Lookup(@Left(#45,1),1,#70); goto #75}
Else goto #85

>#30: Goto #15

<#75:
If #70 <>""
Then #70=#70+" HUNDRED";
    #45=@Del(#45,1,1); #55=""; Lookup(#45,1,#55);
If #55=""
Then {#60=@Left(#45,1); #60=#60*10;
   Lookup(#60,1,#55);
If #55<>""
Then #70=#70+" "+#55; #55="";
Lookup(@Right(#45,1),1,#55)}; cnext

<#80:
#70=#70+" "+#55;
If #50=2
Then #25=#70+" THOUSAND "+#25
Else #25=#70+" "+#25; goto #25

<#85:
If #40=""
Then {#25=#25+" AND 00/100 DOLLARS"+@Text(60,"*");
  #25=@Del(#25,73,500)}
Else {#25=#25+" AND "+@Left(#40,2)+"/100 DOLLARS"+
    @Text(60,"*"); #25=@Del(#25,73,500)};
If @Left(#25,3)="AND"
Then #25=@Del(#25,1,3);
Clear(#35..#85); goto Memo
```

The @Text functions in the program for field #85 add 60 asterisks to the longhand check amount in field #25. In this case, because field #25 is only 73 characters wide, high check amounts (resulting in lengthy text equivalents) will force the asterisks into the expanded field. To eliminate these superfluous asterisks, I added #25=@Del(#25,73,500) statements to truncate field #25's value beginning at the 73rd character position.

Debugging CHECKS.DTF

CHECKS.DTF is now ready for testing. Run the database through its paces by entering check amounts for the smallest check you might conceivably write, up through the largest, with a random selection of check amounts in between. Watch for any discrepancies between the check amounts and their longhand text equivalents. When you enter a check amount, you don't need to type the $ sign or commas—your field format takes care of these.

With your form design, Lookup table, and programming debugged, you're ready to begin entering checks. See the sidebars for details on rounding out your new check writing application.

Optional account balance information

To help avoid overdrafts, you can program Q&A to display an @Msg message showing the account balance before and after each check is written.

Redesign the form by adding a one-character read-only field below the check template, just to the right of field #85. Format it for money values and assign field #90 to it at the Program Spec. One way to obtain the account balance during check entry is to subtract the amount of the previous check from

Automating Check Writing and Printing

You can add features and bells and whistles to CHECKS.DTF to boost your productivity even more. For example, you can place the names of all your company's regular payees (suppliers, utility companies, employees) in the Restrict Spec. Then, when writing checks, pop up the list of payees and select the one you want. For payees not on the list, simply type in the name and press Enter twice to override the restriction.

You can add address fields to the check form, and program them so you can enter the payee ID and have Q&A retrieve the name and address from either the Lookup table or another database. This way, you can print your pre-addressed checks ready to slip into window envelopes for mailing or distribution.

You can design a check stub into the database with fields for a description, explanation, an account breakdown, or employee deduction information. You can program these fields to return Lookup table data based on the account number or deduction codes you enter.

For fast, trouble-free check printing, you can create a Q&A/DOS macro or a Q&A/Windows script to automate batch printing checks in check number order. Depending on your check writing routine, you can have your macro or script select and print just the checks with today's date on them, and optionally have that same macro or script mark those records as printed (via a Mass Update) to avoid printing the same checks twice.

You'll probably want to add security features to the database. At a minimum, you should password protect the file so that no one except authorized users can access it.

the previous record’s balance and bring that result forward into the new check record. Try it. Add the new field to the form, and then add the following statement to the beginning of field #5’s program:

```
If @Add then
{#90=@XLookupr(@Fn,9999999,"Check No","x#90")
    -@XLookupr(@Fn,9999999,"Check No","Amount");
     @Msg("The account balance is $ "+@Str(#90))};
```

Be sure the Check No field is coded SU (Speedy/Unique) at the Speed-up Spec.

To display the remaining balance after entering the amount of the current check, add the following statement to the beginning of field #20’s program:

```
@Msg("The account balance is now $ "+@Str(#90-#20));
```

Don’t forget to enter the opening balance in the first record. To do this, enter and save the first check, then Mass Update the file (one record) using the following update statement in field #90:

```
#1=45233.45
```

Replace 45233.45 with your current checking account balance.

Your accounting practices will dictate any additional application elements needed to keep the account up to date. You might want Q&A to handle deposits, voided checks, interest earned on the account, bank charges, and so forth. See the sidebar, “An Adjustment Database for Complete Account Control.”

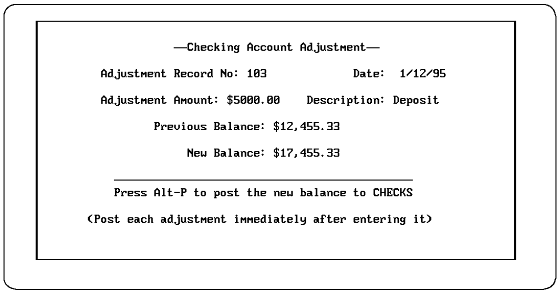
Printing Checks

If you’re printing continuous feed computer checks on a dot matrix printer, or printing one-per-page checks on an ink jet or laser printer, set your Define Page options so you can print each new check right from the screen. If you do this, be sure to set Print Field Labels and Print Expanded Fields to No at the Print Options screen.

To batch print your checks, you can create a Print Spec to tell Q&A which checks to print and where to print the data on each check form.

An adjustment database for complete account control

You can create a supplemental database to record and post adjustments to your checking account. ADJUST.DTF would include, at a minimum, fields for the adjustment record number (auto-incremented), the date (auto-filled), the amount of the adjustment (precede deductions with a minus sign), an explanation, and fields for the current balance and the new balance after the adjustment. (See Figure 4.)



```
                —Checking Account Adjustment—

     Adjustment Record No: 103            Date:  1/12/95

     Adjustment Amount: $5000.00    Description: Deposit

             Previous Balance: $12,455.33

                  New Balance: $17,455.33

       ____________________________________________
       Press Alt-P to post the new balance to CHECKS

   (Post each adjustment immediately after entering it)
```

Figure 4. *An adjustment database lets you record, track, and post deposits, bank charges, and other adjustments to CHECKS.DTF’s balance.*

You can retrieve the current balance during data entry by programming ADJUST to look up the highest CHECKS record and subtracting that record’s check amount from its balance. (The sidebar, “Optional Account Balance Information” shows the programming logic). Your program would then add (or subtract, in the case of a deduction) the current adjustment to arrive at a new balance, which you can post back to the highest CHECKS record. If you’re using Q&A/Windows, which doesn’t have a posting feature, you can Mass Update the highest CHECKS record to look up the adjusted balance from the highest ADJUST record. These adjustment posting procedures can be automated with a macro or script.

[In addition to PERSFIN.DTF, the personal finance database that comes bundled with Q&A/DOS 4.0, see Phoebe Spinrad’s database in “Home Finances Made Easy” (February and March 1993), and Jeff Nitka’s database in “How to Track Multiple Accounts” (September 1993). See also Jeff’s update to his article (“Faster Account Tracking with @Xlookupr,” in the December 1993 issue).—Ed.]

Tom Marcellus is editor of The Quick Answer, *and the author of* PC World Q&A Bible, *published by IDG Books, and* The Quick Answer Focus *special report on monthly billing statements, published by Pinnacle Publishing.*

Calculate Future Dates

Need to know the date nine months from today—30 months from your next birthday? Here's a handy routine that does the work for you.

By Deborah Burkholder

I was asked by a client to create a program to add a variable number of months to an original date in order to arrive at a future due date. It wasn't a simple matter of adding months to a date because months have different numbers of days. After I had determined the mathematical formulas, I built them into a Q&A program. My approach involves dealing with the month, day, and year as separate components. By following my example, you can add a months-based future date calculation to any database.

Month and year calculation

The initial program calculation takes the month of the original date and adds the user entered number of months to it to arrive at an interim due date. This result is then adjusted for the particular month and year. Because the Due Date could be in the same year as the original date or in another year, the program needs to divide the number of months by 12 to determine the number of years in the future. The programming logic is this: When you divide a number of months by 12, its integer (@Int) can tell you how many years have passed, and the remainder (@Mod) can tell you the number of the month in that year. January is month 1, February is month 2, and so forth.

For example, 8/30/94 plus 22 months yields 30 months. The integer portion (@Int) of 30 divided by 12 is 2 (years), and the remainder (@Mod) is 6 (months), so the due date is 6/30/1996.

If the number of years that have passed is even (no remainder), then the year calculation has to be adjusted so that the current year isn't counted twice.

For example, 1/5/94 plus 11 months yields 12 months. The integer portion (@Int) of 12 divided by 12 equals 1 (year) and the remainder (@Mod) is 0 months, so the calculated due date is 0/5/1995. However, the result is wrong on two counts. The resulting month 0 should be month 12 (December), and the year should be 1994, not 1995. My solution was to use a Lookup table, which I'll discuss later, to convert month 0 to month 12 and adjust the year.

Day calculation

The routine must also take into account the day number of the original month and the number of days in the due date month. If the day of the original date is the same or less than the number of days in the due date month, then the same day holds for the future due date month. On the other hand, if the original day is greater than the number of days in the due date month, then the calculation needs to roll back the day to the number of days in the due date month. My program handles this by testing its result against the Lookup table.

For example, 3/31/1994 plus 3 months yields 6/31/1994, which isn't a valid date. But if the program

Add Days to a Date

Adding a number of days to a date to arrive at a future date is much simpler than working with months. You'll need the following fields:

- Date (user entered field formatted for dates)
- Days Added (user entered field formatted for numbers)
- Future Date (calculated field formatted for dates)

Go to the Program Spec, and type the following statement in the Days Added field:

```
> Future Date = Date + Days Added
```

This statement tells Q&A to add the number of days in the Days Added field to the date in the Date field and place the result in the Future Date field. The program is set to execute when the cursor leaves the Days Added field.

If you want Q&A to add the number of days you specify to the current date, you can use a program such as this in the Days Added field:

```
> Future Date = @Date + Days Added
```

—Tom Marcellus

were to roll over to 7/1/1994, the date would be in the fourth month, and that's not what my client wanted. So the date has to be rolled back to the number of days in the due date month, which in this case should be 6/30/1994.

Database design

To illustrate how the program works, I'll use a sample database that contains the following three fields: Test Date, Recall, and Due Date. (You can add these fields, along with the Lookup table and program, which I'll describe later, to any database.) Test Date contains the original or starting date. It can be a user-entered date, or a date auto-typed as an initial value. Be sure to format Test Date for date values.

The Recall field contains the number of months in the future. It's a user-entered field, formatted for numbers with no decimal places (N0). Due Date, also formatted for dates, is a calculated field that contains the future date. The user enters the appropriate date in the Test Date field if necessary, enters the number of months in the Recall field, and the program calculates the result for the Due Date field.

In the Lookup table (see Table 1), I use key column values that match the remainder (@Mod) of the months in the future divided by 12. Column one contains the month to use based on the remainder (@Mod). I use this to handle cases when the remainder is zero (the table translates 0 to month 12, December). Column two contains the number of days in each of the 12 months.

Table 1. *The Lookup table translates the month and compares the days.*

Key	1	2
1	1	31
2	2	28
3	3	31
4	4	30
5	5	31
6	6	30
7	7	31
8	8	31
9	9	30
10	10	31
11	11	30
12	12	31
0	12	31

The program

The program must handle the following five cases:

- The @Mod remainder is greater than zero, and the day is less than or equal to the number days in the due date month.
- The @Mod remainder is greater than zero, and the day is greater than the number of days in the due date month.
- The @Mod remainder is zero, and the day is less than or equal to the days in the due date month.
- The @Mod remainder is zero, and the day is greater than the number of days in the due date month.
- The due date year falls in a leap year.

The following Program Spec routine in the Due Date field handles these five cases by translating the variables and calculating the desired results. The fields are referenced in the program by the following Program Spec field ID numbers:

- Test Date: #2
- Recall   : #4
- Due Date : #6

```
If @Lu(@Mod((@Month(#2)+#4),12),1)>0 and
    @Dom(#2)<=@Lu((@Month(#2)+#4)-
    (12*@Int((@Month(#2)+#4)/12)),2)
then #6=@Lu(@Mod((@Month(#2)+#4),12),1)+"/"+
    @Num(@Dom(#2))+"/"+@Num(@Year(#2)+@Int
    ((@Month(#2)+#4)/12));

If @Lu(@Mod((@Month(#2)+#4),12),1)>0 and
    @Dom(#2)>@Lu((@Month(#2)+#4)-(12*@Int
    ((@Month(#2)+#4)/12)),2)
then #6=@Lu(@Mod((@Month(#2)+#4),12),1)+"/"+
    @Num(@Lu((@Month(#2)+#4)-(12*@Int
    ((@Month(#2)+#4)/12)),2))+"/"+
    @Num(@Year(#2)+@Int((@Month(#2)+#4)/12));

If @Mod((@Month(#2)+#4),12)=0 and @Dom(#2)
    <=@Lu((@Month(#2)+#4)-(12*@Int((@Month(#2)+#4)/
    12)),2)
then #6=@Lu(@Mod((@Month(#2)+#4),12),1)+"/"+
    @Num(@Dom(#2))+"/"+@Num((@Year(#2)+@Int
    ((@Month(#2)+#4)/12))-1);

If @Mod((@Month(#2)+#4),12)=0 and @Dom(#2)
    >@Lu((@Month(#2)+#4)-(12*@Int((@Month(#2)+#4)/
    12)),2)
then #6=@Lu(@Mod((@Month(#2)+#4),12),1)+"/"+
    @Num(@Lu((@Month(#2)+#4)-(12*@Int
    ((@Month(#2)+#4)/12)),2))+"/"+
    @Num((@Year(#2)+@Int((@Month(#2)+#4)/12))-1);

If @Month(#6)=2 and @Mod((@Year(#6)-1992),4)=0
    and @Dom(#2)>=29
then #6=@Lu(@Mod
    ((@Month(#2)+#4),12),1)+"/"+
    @Num(29)+"/"+@Num((@Year(#2)+@Int
    ((@Month(#2)+#4)/12)))
```

Use this routine in conjunction with the Lookup table shown earlier to determine the date that's *x* number of months out from a starting date.

Deborah Burkholder is founder of the Boston Computer Society's special interest group for Q&A and a business consultant specializing in redesigning work flows to accommodate new technology. Q&A is her favorite tool for clients who need both power and ease of use. 617-935-4337.

QUICKTIP

Matching Values Between Databases with @Num

A client had two databases containing employee information. One was an in-house EMPLOYEE database; the other was a PENSION database from the pension administration company. Both databases contained a social security number (SSN) field. The client needed to perform lookups between the two files but was having trouble. The reason was that the SSN field in the EMPLOYEE database was templated in a ###-##-#### format, while the SSN field in the PENSION database was untemplated and contained dashes in the social security numbers. In a templated SSN field, even though the number displays with the dashes, the dashes aren't part of the value. When Q&A performed the external lookup to the PENSION database, it couldn't find a matching social security number.

I added another SSN field to the PENSION database, made it Speedy, and Mass Updated the database using the following Update Spec:

1. In the SSN field containing the numbers with dashes, I typed #1.

2. In the new SSN field, I typed #2=@Num(#1).

The @Num(*x*) function returns the value from field *x* with any non-digit characters stripped out. In this case, in the PENSION records, it returned the SSN without the dashes. This format matched the format of the templated SSN in the MASTER file, and lookups from EMPLOYEE to PENSION could then be performed.

Gordon Meigs, Newtown, Pennsylvania

QUICKTIP

Any Comments?

In some databases it's useful to have a field to store freeform comments or notes about the subject of the record.

For example, if a VENDOR record occupies page 1, you can add a Comments field to the top of page 2 without upsetting the form design. You can make the field up to 20 lines long, or you can make it small and expand it with F6 to add or view comments. This way, when you press PgDn from anywhere on page 1, Q&A will take you to the Comments field on page 2.

When you move from record to record, though, Q&A displays only page 1. It might be helpful to know if there are any comments on page 2 without having to move there to find out. You can have Q&A do this by adding a Navigation Spec on-field-entry statement to the first field on the form, like this:

```
< If @Len(Comments) > 0
  Then @Msg("PgDn to see comments")
```

When there's text in the Comments field, Q&A will display "PgDn to see Comments" on the message line. If the Comments field begins at the top of page 3, just change your @Msg message to remind yourself to press PgDn twice. Now, as you display each record, you'll know immediately if it contains any comments.

Keep in mind that you can easily search on a Comments field. At the Retrieve Spec, move to the Comments field and type in your search restriction surrounded by two periods, like this:

```
..search text..
```

Replace "search text" with the text you're searching for. Press F10, and Q&A will retrieve all the records that contain the text anywhere in the Comments field.

Tom Marcellus

@HELP

Edited by Dave Reid

Calculate a Discount

DOS WIN

Q I use a database to track invoices for my company. Each invoice stores customer details, along with labor, part number, and price information. I want Q&A to calculate a 10 percent discount and reflect it in the Balance Due field. How do I do it?

A You can use the following program to calculate a 10 percent discount:

```
Labor    :
Parts    :
SalesTax: <SalesTax = Parts * .07
Total    : <Total = Labor + Parts + SalesTax
Discount: <Discount = Total * .10
Bal Due : <Bal Due = Total - Discount
```

I'm assuming your sales tax is 7 percent; if it's different, change the .07 in the SalesTax field. I'm also assuming that you don't charge sales tax on Labor. If you do, change the SalesTax program like this:

```
SalesTax: <SalesTax = (Labor + Parts) * .07
```

The Total field program adds the Labor, Parts, and SalesTax fields. The Discount program multiplies the Total by .10 to get 10 percent of the Total. The Bal Due field program then subtracts the Discount from the Total.

Work Around Field Naming Restrictions

DOS WIN

Q Thank you for your tip on aligning merge document columns in the August 1994 issue of *The Quick Answer*. I had never considered using programming in mail merge, but your answer got me thinking, and I began using it productively. I recently ran up against a problem I can't solve. I want to use a mail merge program with a field named Dept # but Q&A won't accept the statement. I tried Dept_#, Dept#, and other variations, but nothing worked until I changed the name of the field to Dept No. Why can't I use the original field name?

A The problem with the original field name is that it uses a non-alphanumeric character (#). The following kinds of field names can't be used directly in mail merge programming:

- Field names that begin with a number, such as 1st Baseman, 3rd Call Date, and 1994 Filed.
- Field names that contain non-alphanumeric characters, such as Dept #, Num. Attending, and Done (Y/N).
- Field names that are the same as terms reserved by Q&A, such as Stop, Goto, and Return.

To reference any of these "illegal" field names in mail merge programming (as well as in database programming), type them using a special "pound-quote" syntax: enclose the field name in quotes, and precede it with a pound sign. For example, your original field name can be referenced like this:

```
#"Dept #"
```

Make Custom Menus More Reliable

DOS

Q I can't seem to get my custom menus to work reliably when I use multiple menu layers. Though my menu structure includes only two layers, whenever I go from the top menu to the second menu and run a macro from it, Q&A messes up. A sample portion of my menu structure looks like this:

```
Top Menu          Menu macros
--------
Data Entry    DE calls the Data Entry menu
Reports       RP calls the Reporting menu

Data Entry
----------
Clients       ClientDE goes to Add Data for CLIENT.DTF.
```

I have no trouble going from Top Menu to Data Entry, but when I try to run the Client macro (or any others), I get unexpected results. Sometimes Q&A winds up at the Report menu, other times at Mass Update for a different database. What can I do to make Q&A recognize my multilevel menu structure?

A The problem appears to be the way you're having Q&A move from one menu to another: you're using macros whose only function is to display another menu. I like to call these *null macros* because they do nothing but display menus. You should never use null macros to move between custom menus. Q&A allows you to specify the Menu Name of the menu you want to display. I suggest you do this, after which your problems should go away. Your updated Top Menu should look like this:

```
Top Menu        Custom Menu Names
----------
Data Entry      Data Entry Menu
Reports         Reporting Menu
```

You don't need to make any changes to your second layer menus except to ensure that every menu in your entire structure has Menu Returns set to Yes at the Application Menu Options screen.

Use Real Tabs to Align Merge Document Columns

DOS

Q I'm trying to use the answer to an @Help question that appeared in the August 1994 issue, "Align Merge Document Columns." I've set up my document as follows:

```
*pg {First1 + " " + Last1} (L) *        *SS1*
*pg {First2 + " " + Last2} (L) *        *SS2*
*pg {First3 + " " + Last3} (L) *        *SS3*
```

This works fine for the first 16 entries, as you said it would. But I have 20 name and Social Security number fields in the database. Is there any other way I can keep the right-hand column left-aligned?

A Yes, you've come up against the limit of 16 mail merge programming statements per document in Q&A/DOS. You can, however, solve the problem by using Real Tabs. First, verify that Real Tabs are turned on. From the Write menu, select Utilities / Set Global Options / Set Editing Options. Set the Use Real Tabs option to Yes. Press F10, and then Get your merge document. Remove all the mail merge programming so that only the merge fields remain, like this:

```
*First1* *Last1* *SS1*
*First2* *Last2* *SS2*
*First3* *Last3* *SS3*
```

Clear all the tabs from the document except for one at approximately the five inch mark on the ruler line. This tab will mark the beginning of the SS column—you can place it exactly where you want it. Finally, place your cursor on the asterisk that begins each SS field, be sure you're in Insert mode, press the tab key, and the SS field should jump to the Real Tab position. When you print the document, the names will be neatly joined, and all 20 SS fields will be left-aligned.

Who Has a Birthday this Month?

DOS WIN

Q My database contains a Date of Birth field. I'd like to mail birthday cards to everyone born in a certain month regardless of the year. How can I select the records I need when printing mailing labels?

A Here's a trick you can use at any Retrieve Spec, whether you're printing a report, merge letters, or mailing labels. In the Date of Birth field, type the following characters, substituting the two-digit number of the appropriate month for the 01 (January):

```
]????/01/??
```

This is a wildcard date search. It begins with the right square bracket, followed by a date in the YYYY/MM/DD format. The question marks are wildcards—they tell Q&A that you don't care about that part. The previous example says, "I don't care which year; the month must be January; and I don't care which day."

[See the "Search for Irregular Values" QuickTip in this issue for more on using the square bracket search—Ed.]

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*, PO Box 888, Kent, WA 98035-0888** or fax to 206-251-5057. When writing, 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 will publish those questions we feel are of general reader interest; individual responses are not possible.

How to Restrict a Field Entry's Length

When you merge data with a Write document or mailing label whose print format must be controlled, the length of the merged data can be critical. Restricting it in the database, rather than in the merge document, is the best solution.

By Gordon Meigs

DOS WIN

Q&A 4.0 and Q&A/Windows allow you to expand a text field during data entry to enter as much text as you need, up to about 16 pages. (In Q&A/Windows, the field expands automatically as you type. In Q&A/DOS, you have to press F6 to expand it.) The field width can even be as small as one character and still be expanded. What's more, an expanded field's contents can be searched for any word or phrase, such as ..green.. or ..City of Brotherly Love...

Sometimes, though, you need to *prevent* entry of a text value greater than the width of the displayed field—for example, when the data in the field is to be merged with a predesigned form—a merge document or mailing label—that includes a restricted area for the information. In such a case, if "excess" data were allowed during data entry, that data, when merged during printing, would extend beyond the restricted area and mess up the printed form.

Control a field value's length

One way to limit the length of a field entry is to program the database to count the number of characters in the field and have Q&A display a warning message and return to the field if the maximum number of characters has been exceeded. There's an easier method, though, that doesn't require programming and in most cases will work just as effectively: a field template. You might think that controlling the length of a field's entry would be a Restrict Values function, but we're not interested in restricting what can be entered into the field, just the number of characters the field can contain. So a field template is the answer.

To access the Field Template Spec in Q&A/DOS, from the Main menu select File / Design a File / Customize a file / Field template. (For Q&A/Windows, see the sidebar, "Creating a Field Template in Q&A/Windows".) At the Field Template Spec, you can press F1 for Q&A's Help screen. (See Figure 1).

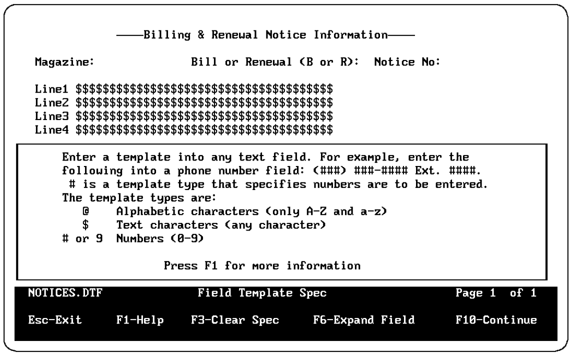
The example in Figure 1 shows a message area for reminders (fields named Line1 through Line4) to be merged with a Write document and mailed to subscribers whose magazine subscriptions are expiring. In the merge document that prints the notices, these four message lines are merged at a precisely defined location and their lengths must be controlled to avoid upsetting the surrounding text.

Creating a Field Template in Q&A/Windows

In Q&A/Windows, a field template is called a field *mask*. To mask a text field, open the database, click on the Select menu, and choose Database Structure. Move to the Masking column in the structure table (it's the last column) and double-click on any cell to display the Edit Mask dialog box. (You can also get to the Edit Mask dialog box by selecting Edit field attributes from the Database menu, and choosing Masking from the submenu.) Select the pertinent field, and choose one of the ready-made masks or select Custom mask. For a Custom mask, type in the characters that represent the mask restriction ($, @, #, or 9—see Figure 1) along with the characters (such as dashes or parenthesis) you want fixed into the mask. Click OK when you're finished.

When you enter data into a masked field, Q&A won't auto-expand the field as usual but will restrict you to the number of characters allowed by the mask, just as it does in Q&A/DOS.—*T.M.*

Write normally allocates as much room as needed for merged information, adjusting the spacing and wrapping the lines as necessary. But in this case, without restricting the number of characters in these lines, the printed notices could wind up looking unsightly.



```
        ———Billing & Renewal Notice Information———

Magazine:               Bill or Renewal (B or R):  Notice No:

Line1 $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
Line2 $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
Line3 $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
Line4 $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    Enter a template into any text field. For example, enter the
    following into a phone number field: (###) ###-#### Ext. ####.
     # is a template type that specifies numbers are to be entered.
    The template types are:
       @    Alphabetic characters (only A-Z and a-z)
       $    Text characters (any character)
    # or 9  Numbers (0-9)

                    Press F1 for more information
 NOTICES.DTF              Field Template Spec                Page 1  of 1

 Esc-Exit    F1-Help    F3-Clear Spec     F6-Expand Field     F10-Continue
```

***Figure 1**. At the Field Template Spec the number of characters allowed in four fields is restricted by templates. The $ code enables the field to contain anything, but in this case only up to 38 characters. Here, Q&A's built-in Help screen is displayed.*

Why Use a Field Template?

Normally, you place a template on a text field to speed data entry and improve the format. For example, you can create a ###-##-#### template for a Social Security number field, a (###) ###-#### template for a phone number field, or an ABX-#### template for a part number field where there are always four-digits with an ABX- prefix. In these cases, you don't have to type the dashes, the parentheses, or the part number prefix during data entry because Q&A "fixes" them in the field.

In the case at hand, however, we're using the template to control the number of characters in the field, and to prevent the field from being expanded during data entry.—*Tom Marcellus*

QUICKTIP

Dedicated Key Field Not Needed for XLookups

You might want to program a transaction database to perform lookups, but because of space, aesthetic, or security reasons, you might not want to include a field for the ID or account number in the database. You can still perform your lookups.

Here's an example. Suppose your customer database includes an Account ID field, and your invoice database is programmed to lookup the customer's name and address during order entry based on that account ID.

You don't need an Account ID field in the invoice database because you can use the Name field to temporarily hold the key value. Suppose your invoice database includes the following fields:

```
   Name:
Address:
   City:              State:    Zip:
```

Simply write your XLookup statement to retrieve the values in reverse order, that is, the Zip, State, City, Address, and Name fields, respectively. This way, during invoice entry, you can type the customer's account ID in the Name field, press Enter, and Q&A will fill in all the fields, including the Name field. Assuming the fields in the customer database have the same names as the corresponding fields in the invoice database (though they don't have to), here's a sample XLookup statement for the Name field that will do the trick:

```
> XLookup("Customer", Name, "Account ID",
          "Zip", Zip,
          "State", State,
          "Address", Address,
          "Name", Name)
```

The field that holds the temporary key value (in this case, Name) must be the final parameter in the XLookup statement because it overwrites the temporary key value, and no further lookups on the Account ID are possible. However, you can always return to the Name field, press Shift-F4 to clear it, type another account ID, and press Enter to execute another series of lookups on the new account number.

Tom Marcellus

How a template controls an entry's length

Q&A's help screens and manuals don't mention everything a $$$$$ field template can do, so it isn't easy to find out without trying it. About all you're told is that using the $ code will allow any text character in that position in the field. For our needs, however, here's the important thing. When you template a field with $s as shown in Figure 1, *nothing happens when you try to expand the field during data entry.* This simple fact prevents the person editing the field from typing too many characters and allows you to control the length of the data in your merge documents.

Whenever the output format of your data is critical, such as in mail merge, a mailing label, or even a report, you can control the length of the entry in any text field with a template, and thereby prevent users from entering values that exceed the predetermined number of characters.

Gordon Meigs is vice president and general manager of Professional Computer Technology Associates of Newtown, Pennsylvania, and author of The Quick Answer Focus *special report on reports. He teaches courses and does corporate training on Q&A and has been designing and installing advanced Q&A applications for more than eight years. 215-968-4185, CompuServe 71023,356.*

Using Merge-Field to Control the Number of Characters

Why not control the field's length in the merge document instead of in the database? You can, but you'll pay a price. You can set up your fields in the merge document to force Q&A not to exceed a specific character allotment. Such controlled merge fields might look like these:

```
*Line1                   (L) *
*Line2                   (L) *
*Line3                   (L) *
*Line4                   (L) *
```

The problem with restricting the number of characters this way is that if the database field contains more characters than the merge field permits, Write will truncate any that don't fit. In such cases, the remaining characters (part of a message, the end of an address line, even the last part of the person's name) will be chopped off.

On the other hand, the field template method prevents anyone from *entering* a value that exceeds the predetermined character allotment, and you won't have to worry about merge documents with truncated information being printed and mailed.

QUICKTIP

Search for Irregular Values

The *Q&A 4.0 User Manual* covers how to use the right bracket, "]," to search date fields for non-date values, but the feature is more powerful than described there. You can retrieve records dated May of any year with a]????/05/?? parameter; records dated July 4 of any year with a]????/07/04 parameter; or records dated December in the 1980s with a]198?/12/?? parameter. The question marks are wildcards. Q&A uses the YYYY/MM/DD format to store dates internally—not to be confused with the date display format you select at the Global Format Options screen.

The right bracket search has uses beyond date values. For example, to indicate "held," you might want to mark a record with an H in a Yes/No formatted field. In another situation, if you didn't have the correct date, you might want to type "unknown" in a date field. Or, you might want to place 12/7/93-2 in a date field to designate the second batch on 12/7/93.

During data entry, when you attempt to enter a value that doesn't fit the field format, Q&A displays a warning message. But you can press Enter or Tab to force Q&A to accept the unusual data. However, it will then be difficult to retrieve records on those irregular values.

In the earlier examples, if you entered "H, unknown," or "12/7/93-2" as the retrieval parameter, Q&A would tell you it's not a valid Retrieve Spec. All that's necessary to retrieve those records, though, is to precede the irregular value with the right bracket:]H,]unknown, and]12/7/93-2, respectively. The square bracket says, "Find what I'm telling you to find, exactly as typed." It's a useful tool for retrieving records on fields containing irregular values.

Gordon Meigs, Newtown, Pennsylvania

Conditional JOIN

Tom Marcellus' article in the November issue ("*Really* Customize Your Letters," page 14) states that you can't make a *JOIN* command conditional when merging a paragraph into a document. This isn't true. The following command joins the TAKE.DOC document with the merge document if the number in the Visits field is greater than zero:

```
*JOIN *Pg {@Text(Visits > 0,"TAKE.DOC")}**
```

I use several such merge paragraphs in a variety of Write documents. The nice thing about this is that no additional programming or fields are needed.

Michael Kelly, M.D., San Diego, California

[The doctor is right! If the document you're joining isn't in your default documents directory, be sure to specify its complete path.—Ed.]

Merge Printing Success

Your article on merge paragraphs ("Really Customize Your letters with a Database of Merge Paragraphs," November 1994) couldn't have been more timely. Because I have different kinds of potential customers, I've had to use several different form letters and haven't been able to print them all at the same time. However, by setting up a paragraph database as you describe in the article, I'm now able to print all the letters at the same time. Thanks a bunch for the timesaver.

Jerry Altshuler, Century 21 Investors, Chicago, Illinois

Every Inch Counts

I liked Dave Jampole's routine to check credit card numbers as they're entered ("Validate Credit Card Numbers," October 1994). However, I couldn't afford all those additional fields on my mail order form. Even hidden, they'd take up too much space. And because we print the record after each order, that many additional fields would cause Q&A to print an extra page.

My validation routine requires only four fields: Card Number (#1), Type (#2), Verification (#5), and Expiration Date (#9), making it easy to incorporate into any database. It also inserts dashes in the number, so it looks like the number on the card. The Type field contains most of the calculation programming. Upon entry of a valid number, it also returns the card type. We use single letters to identify the card type, though you can display the full card names (Visa, Mastercard, American Express, Discover) if desired.

The Verification field program makes sure the card number contains a valid sequence of digits and stores the bank's authorization number for a permanent record.

Finally, the Expiration Date field is programmed to accept dates as they're printed on the card. For example, when you enter 794 or 0794, you get 07/94. Here are the programs for the four fields:

```
>#1:If @Left(@Num(#1),1) <3 or @Left(@Num(#1),1)
     >6 Then {@Help(#1); @Msg("#1: Wrong Starting
     Number"); @Help(#1); Goto #1};
   If @Len(@Num(#1)) <>13 and @Len(@Num(#1)) <>
     16 and @Len(@Num(#1)) <>15
   Then {@Help(#1); @Msg("#2: Incorrect Length");
     @Help(#1); goto #1};
   If @Len(@Num(#1)) = 13 Then #1="000"+@Num(#1);
   If @Len(@Num(#1)) = 15 Then #1= "0" +@Num(#1)
   Else #1=@Num(#1); goto #2

<#2:#2=0; #5=0;
    #2=@Mid(#1,15,1)*2; #5=#5+@Mid(#2,2,1)+
       @Mid(#2,1,1); #5=#5+@Mid(#1,14,1);
    #2=@Mid(#1,13,1)*2; #5=#5+@Mid(#2,2,1)+
       @Mid(#2,1,1); #5=#5+ @Mid(#1,12,1);
    #2=@Mid(#1,11,1)*2; #5=#5+@Mid(#2,2,1)+
      @Mid(#2,1,1); #5=#5+@Mid(#1,10,1);
    #2=@Mid(#1,9,1)*2; #5=#5+@Mid(#2,2,1)+
       @Mid(#2,1,1); #5=#5+@Mid(#1,8,1);
    #2=@Mid(#1,7,1)*2; #5=#5+@Mid(#2,2,1)+
      @Mid(#2,1,1); #5=#5+@Mid(#1,6,1);
    #2=@Mid(#1,5,1)*2; #5=#5+@Mid(#2,2,1)+
       @Mid(#2,1,1);#5=#5+@Mid(#1,4,1);
    #2=@Mid(#1,3,1)*2; #5=#5+@Mid(#2,2,1)+
       @Mid(#2,1,1); #5=#5+@Mid(#1,2,1);
    #2=@Mid(#1,1,1)*2; #5=#5+@Mid(#2,2,1)+
       @Mid(#2,1,1); goto #5

<#5:If @Rt(#5,1) =0 Then #2 =0;
    If @Rt(#5,1) >0 and @Rt(#5,1) <=9 then #2 =10
       -@Rt(#5,1);
    If #2 <> @Rt(#1,1) Then {#5="BAD"; Clear(#2);
    If @Left(#1,3)=000 Then #1=@Del (#1,1,3);
    If @Left(#1,1)=0   Then #1=@Del (#1,1,1);
       @Help(#1); @Msg("#3: Invalid Account
       Number"); @Help(#1); Goto #1};
    If #2= @Rt(#1,1) then #5="OK"; Clear(#2);
    If @Left(#1,3)=000 then #1=@Del(#1,1,3);
    If @Left(#1,1)=0 then #1=@Del (#1,1,1);
    If @Len(@Num(#1))=16 then #1=@Left
       @Num(#1),4) +"-"+ @Mid(@Num(#1),5,4) +"-"+
       @Mid(@Num(#1),9,4) +"-"+
       @Right(@Num(#1),4);
    If @Len(@Num(#1))=13 then #1=@Left
       @Num(#1),4) +"-"+ @Mid(@Num(#1),5,3)
       +"-"+ @Mid(@Num(#1),8,3) +"-"+
       @Mid(@Num(#1),11,3);
    If @Len(@Num(#1))=15 Then #1=@Left
       @Num(#1),4)+"-"+@Mid(@Num(#1),5,6)+"-"+
       @Right(@Num(#1),5);
    If @Left(#1,1)=3 then #2="A";
    If @Left(#1,1)=4 then #2="V";
    If @Left(#1,1)=5 then #2="M";
    If @Left(#1,1)=6 then #2="D"; goto #9

>#9: If @Len(@Num(#9))=4
    Then #9=@Left(@Num(#9),2)+"/"+@Right
       (@Num(#9),2);
    If @Len(@Num(#9))=3
    Then #9="0"+@Left(@Num(#9),1)+"/"+@Right
       (@Num(#9),2); goto #10
```

Joe D. Bryant, Moore, Oklahoma

QUICKTIP

Use Caution with the @Add Context Function

Be careful when using If @Add in a programming statement. It executes in the current record and when you press F9 to view or edit records entered during the same Add Data session. For example, the following statement will execute in Add Data mode on any displayed record added during the current session:

```
<#1: If @Add then #1=@Number
```

If you press F9, this statement will assign the next @Number to the previous record, overwriting its original record number. To prevent this from happening, add a second condition to the statement, like this:

```
<#1: If @Add and #1="" then #1=@Number
```

This statement executes in Add Data mode only when field #1 is blank.

Robert Laudensack, Leiden, The Netherlands.

QUICKTIP

Handle QABACKUP Restore Failures

If QABACKUP *[see "Q&A Utilities to the Rescue!" October 1994]* backs up your database but won't restore it, the database could be corrupted. First, make a copy of the database (don't overwrite an earlier good backup) and run Recover on the copy. Then, use QABACKUP to backup and restore the copy. If the restore operation proceeds smoothly, you can delete the original corrupted database, and rename the copy with the original database's file name.

Inexplicable problems and error messages can occur when a Q&A database has been even slightly damaged (a small power surge can do it). Running Recover usually handles them. Never run Recover on your original database, though, because in rare cases Recover can actually damage it further.

Tom Marcellus

PINNACLE

Pinnacle Publishing, Inc.
PO Box 888
Kent, WA 98035-0888
206-251-1900
206-251-5057 (fax)

Printed in the U.S.A. on recycled paper.