Side-by-Side subfiles

Side-by-Side Subfiles

Display two subfiles side by side on a green screen

by Paul Morris

FOR A PROGRAM I RECENTLY WROTE, I needed to be able to show two subfiles side by side. As this can't be done conventionally with DDS, I looked at my books and the available literature to see what techniques are available.

The first of the two techniques I came across was to simulate subfiles by using a conventional, non-subfile display format with repeating lines and all the data held in a multiple-occurrence data structure. Although this approach could look good from the user perspective, the programming is not elegant.

The second technique involved having each subfile in a separate window and using a function key to toggle between the windows. This approach is easier on the programming but harder on the users.

Neither of these approaches suited my need, so I developed the technique that I explain in this article. I believe this method has uses beyond the example I explore here.

The Problem

To understand why I wanted something different, let me first explain the problem. Then, I'll explain how my solution appears from the users' perspective.

As part of a small package I was developing, I had a batch process that identified potential duplicate customer records (or multiple duplicates) and wrote details of them to a file. I needed a screen program to review this file of duplicated records. On the left side, I wanted to see the list of "original" customers; on the right side, for a selected "original" customer, I wanted to see the one or more duplicate records identified by the batch process. (In this article, I show a simplified version of the program so the source isn't cluttered up by irrelevant coding.)

The screen is split into three windows (Figure 1). At the top is a short window across the whole width with just the titles. The left side of the screen has a second window showing the "original" customer records. The third window on the right is similar to the second and shows the list of duplicate records for the record selected on the left-side window. This will become clearer as I explain what the users see and what actions they can take.

When the program first loads, the window on the left side of the screen shows a list, starting from the beginning, of the "original" customer records identified. The first record in the subfile is highlighted, and the top section of the window shows the full name and address details of the customer record highlighted.

The window on the right may have one or more records showing in the subfile; these are the duplicates identified for the highlighted record on the left. The first record on the right is also highlighted, and the full name and address of this record appear in the top part of that window. So what the users see are the original and the first duplicate of the first customer record identified.

If you page down, the left-side subfile loads and shows the next set of records, and the highlighted record now appears on the top record of the second page. At the top of the screen appear the full address details of this newly highlighted record. The window on the right also changes to match this record. This window now shows the duplicate(s) of the record on the left, with the top record highlighted and its full details in the top portion of the window.

If you now use F13, the highlighter moves down a record. The right side now changes to match this record. Note that the right window always reflects the record selected by the highlight on the left side of the window.

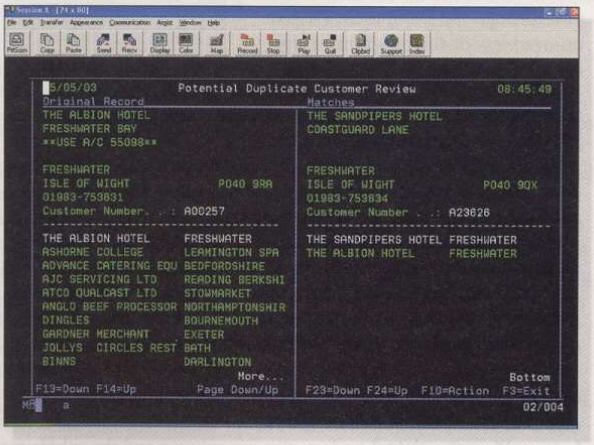

FIGURE 1
Green screen with side-by-side subfiles

If you continue to use F13, you eventually get to the bottom of the page, after which a new page is loaded with the highlight on the first record of this third page.

F14 works in a similar manner but moves the highlight back through the list one record at a time, moving over a page when required until the start of the list is reached.

F23 and F24 work one record at a time in a similar manner to F13 and F14, but they're used for controlling the highlighted record on the right side of the screen. Hopefully, there would only be one or two records here, but the program will work with more than one page full should the need arise.

F10 is for when you want to process a highlighted pair of records. Here, another program is called with the customer numbers of both the original and duplicate records passed to it. This program displays another window, which is placed under the full name and address and gives the user the opportunity to decide on a course of action.

Note that there is no "X" in the box selection; therefore, there is no input into either of the subfiles. All records are selected by using the function keys to move a pointer (in this case, a highlight) to the desired records.

How is this achieved? First I'll walk you through the DDS; then, I'll give an overview of the program before examining the code in more depth.

FIGURE 2
Sample application DDS

```
A*====================================================================
A* Display File  - WAJI26D     Version   1.0      Date  21-02-2003
A* Based on     - WAJ026D     Version   1.0      As at 03-08-2002
A* Author       - Paul Morris, Welwyn Support Services Ltd.
A* Copyright (C) - 2002-2003 Welwyn Support Services Ltd.
A* Project ID    - ISN01
A* Function      - Duplicate customer review
A*                iSeries NEWS version
A*
A*--------------------------------------------------------------------
A* Warnings      -
A* Dependencies  -
A* Associated programs
A*              -
A*====================================================================
A* Modifications
A*--------------------------------------------------------------------
A* Change ID.    - XX999             Date   ../../..
A* Name          -
A* Description   -
A*
A*====================================================================
A                                      DSPSIZ(24 80 *DS3)
A                                      PRINT
A                                      CA03(03 'Exit')
A          R #1WND
A                                      WINDOW(1 2 1 75 *NOMSGLIN)
A                                      PAGEUP(26 'prior Page')
A                                      PAGEDOWN(27 'next page')
A                                      CA13(13 'next original')
A                                      CA14(14 'previous original')
A                                      CA23(23 'next match')
A                                      CA24(24 'previous match')
A                                      CA10(10 'Action')
A                                      WDWBORDER((*CHAR '___|||_|'))
A                                      USRRSTDSP
A                                      WDWTITLE((*TEXT 'Original Record___-
A                                      _____Matches_____-
A                                      _____') (*DSPATR U-
A                                      L HI) *BOTTOM)
A                                    1  1DATE
A                                      EDTCDE(Y)
A                                    1 21'Potential Duplicate Customer Revie-
A                                      w'
A                                      DSPATR(HI)
A                                    1 68TIME
A          R #2SFL                     SFL
A            CUSN#2    R        H      REFFLD(SLR05/CUSN05 *LIBL/CUSNAMES)
A            CNAM#2    R   20   0 10  1REFFLD(SLR05/CNAM05 *LIBL/CUSNAMES)
A  34                                  DSPATR(HI)
A            CADD#2    R   15   0 10 22REFFLD(SLR05/CAD105 *LIBL/CUSNAMES)
A  34                                  DSPATR(HI)
A          R #3CTL                     SFLCTL(#2SFL)
A                                      SFLSIZ(0144)
A                                      SFLPAG(0010)
A                                      WINDOW(3 2 20 36 *NOMSGLIN)
A                                      OVERLAY
A  31                                  SFLDSP
A  32                                  SFLDSPCTL
A N31                                  SFLCLR
A  33                                  SFLEND(*MORE)
A                                      WDWBORDER((*CHAR '   ||| |'))
```

The DDS

I'll start the examination of the DDS (Figure 2) from the bottom as this approach matches more closely the order in which the records are written.

Format #7WND is a failsafe window for when there is no data in the file.

#5CTL is a subfile control record for the subfile on the right side of the screen; this record also defines the window that it sits in. WDWTITLE is used to give some function key assignments at the bottom of the window, but you'll notice that there are no function keys defined to this format, nor are there any input-capable fields.

#4SFL is the associated subfile; this has a hidden field holding the customer number. Note that this format also has no input fields, but it does have an indicator for controlling the HIGHLIGHT keyword. The lack of input fields and function keys is because the window is output only with a WRITE. There is no associated EXFMT; the program, and not the user, controls what this window does.

#3CTL and #2SFL are the subfile control and the subfile for the window on the left side of the screen. These are very similar to #5CTL and #4SFL in that there are no input fields or function keys assigned. Again, this is controlled by the program, not the user.

If you look at the window parameters for these two windows, you'll notice they slightly overlap each other. This is to maximize the width of the windows for the data (by having only one line down the center of the screen).

The first format, #1WND, is the title format. It has the function key assignments that control the operation of the program. Notice that the page keys are defined and used just as any other function keys are; they don't control the subfiles directly. (I could

FIGURE 2 *continued*

```
A                                       WDWTITLE((*TEXT 'F13=Down F14=Up   -
A                                            Page Down/Up') *BOTTOM)
A                                       USRRSTDSP
A            RCDNBR#3       4S 0H       SFLRCDNBR(CURSOR)
A            CNAM#3    R       0  1  1REFFLD(SLR05/CNAM05 *LIBL/CUSNAMES)
A            CAD1#3    R       0  2  1REFFLD(SLR05/CAD105 *LIBL/CUSNAMES)
A            CAD2#3    R       0  3  1REFFLD(SLR05/CAD205 *LIBL/CUSNAMES)
A            CAD3#3    R       0  4  1REFFLD(SLR05/CAD305 *LIBL/CUSNAMES)
A            CAD4#3    R       0  5  1REFFLD(SLR05/CAD405 *LIBL/CUSNAMES)
A            CAD5#3    R       0  6  1REFFLD(SLR05/CAD505 *LIBL/CUSNAMES)
A            PSCD#3         10A  0  6 27
A            PHON#3    R       0  7  1REFFLD(SLR05/PHON05 *LIBL/CUSNAMES)
A                                  8  1'Customer Number. . :'
A            CUSN#3    R       0  8 22REFFLD(SLR05/CUSN05 *LIBL/CUSNAMES)
A                                       DSPATR(HI)
A                                  9  1'------------------------------------
A                                     -'
A                                       DSPATR(HI)
A          R #4SFL                      SFL
A            CUSN#4    R       H       REFFLD(SLR05/CUSN05 *LIBL/CUSNAMES)
A            MTCS#4    R       H       REFFLD(SLR05/CUSN05 *LIBL/CUSNAMES)
A            CNAM#4    R  20   0 10  1REFFLD(SLR05/CNAM05 *LIBL/CUSNAMES)
A  44                                   DSPATR(HI)
A            CADD#4    R  15   0 10 22REFFLD(SLR05/CAD105 *LIBL/CUSNAMES)
A  44                                   DSPATR(HI)
A          R #5CTL                      SFLCTL(#4SFL)
A                                       SFLSIZ(0144)
A                                       SFLPAG(0010)
A                                       WINDOW(3 41 20 36 *NOMSGLIN)
A                                       OVERLAY
A  41                                   SFLDSP
A  42                                   SFLDSPCTL
A N41                                   SFLCLR
A  43                                   SFLEND(*MORE)
A                                       WDWBORDER((*CHAR '    |  |'))
A                                       WDWTITLE((*TEXT 'F23=Down F24=Up  F-
A                                       10=Action  F3=Exit') *BOTTOM)
A                                       USRRSTDSP
A            RCDNBR#5       4S 0H       SFLRCDNBR(CURSOR)
A            CNAM#5    R       0  1  1REFFLD(SLR05/CNAM05 *LIBL/CUSNAMES)
A            CAD1#5    R       0  2  1REFFLD(SLR05/CAD105 *LIBL/CUSNAMES)
A            CAD2#5    R       0  3  1REFFLD(SLR05/CAD205 *LIBL/CUSNAMES)
A            CAD3#5    R       0  4  1REFFLD(SLR05/CAD305 *LIBL/CUSNAMES)
A            CAD4#5    R       0  5  1REFFLD(SLR05/CAD405 *LIBL/CUSNAMES)
A            CAD5#5    R       0  6  1REFFLD(SLR05/CAD505 *LIBL/CUSNAMES)
A            PSCD#5         10A  0  6 27
A            PHON#5    R       0  7  1REFFLD(SLR05/PHON05 *LIBL/CUSNAMES)
A                                  8  1'Customer Number . .:'
A            CUSN#5    R       0  8 22REFFLD(SLR05/CUSN05 *LIBL/CUSNAMES)
A                                       DSPATR(HI)
A                                  9  1'------------------------------------
A                                     --'
A                                       DSPATR(HI)
A          R #7WND
A                                       WINDOW(8 20 2 40)
A                                  1  4'No records to show for this compan-
A                                     y'
A                                       DSPATR(HI)
```

ONLINE CODE

You can download *iSeries NEWS* code from *www.iSeriesNetwork.com/code* as an iSeries save file, as separate PC files, or as a PC source code "bundle" file.

have chosen another pair of function keys, but that would have made the user interface rather clumsy.)

Finally, a word about these windows: They each have USRRSTDSP defined. This is to stop the system from saving the windows; I take direct control of what happens in the program.

The Program

The program starts off by loading the first 10 records in the subfile on the left. It sets a "current record" pointer to 1 to indicate this is the highlighted record. It then chains back to the "current record" in the subfile for three reasons:

1. To highlight the record.
2. To get the customer number, which is then used to get the full details from the customer record and display these details in the control format.
3. To use this customer number to get the duplicate records, which are then loaded into the subfile on the right.

The loading of the subfile on the right sets a "right side current record" pointer to 1, and in a similar manner on the left side, it highlights the current record and gets the address details.

The windows are written in the following order: first #5CTL, then #3CTL, and then finally the #1WND is output with an EXFMT.

The next action is dependent on the function key taken. If F23 or F24 is used, the "right side current record" pointer is incremented or decremented, the highlight is removed from the old record and added to the new record, and the full address details in the top part of the window are updated.

Similarly, if F13 or F14 is taken, the pointer for the left side is incremented or decremented, the left highlight is removed and placed on the new record, and the full address details are placed in the top part. In addition, the subfile on the right is refreshed to show the duplicates of this newly highlighted record.

Page down and page up work in a similar manner to F13 and F14, but the pointer changes by 10 records instead of 1. The program checks for the end of the subfile (this is done for F13 and F14 as well), and more records are loaded if this is required.

I don't put out a message for trying to roll beyond either end of the subfile. This is for simplicity and because you don't get a message in Windows programs if you try to page beyond the end of a document.

Before I go into the details, I'll explain how I construct the read loops for page-at-a-time subfiles. I always read ahead so that the last record read is not output to the screen at that time; rather, it sets a condition on the SFLEND indicator. The next group of reads for a subfile starts by outputting this last record before reading from the database.

Code Details

Now let's look at the code in Figure 3. For both subfiles, we need to track where the highlight (i.e., the current record) is and the next record we wish to highlight. The program does this by tracking the RRN number of the

appropriate records in the subfile. For the left-side subfile, DSPRRN holds the record number of the record we are moving to, and DSPRRNPRV holds the record we are moving from. Similarly, DSPRRNR and DSPRRNR PRV hold the records for the right-side subfile.

The #INIT subroutine, in a call to STARTALL, loads up and writes both subfiles with the first set of records and outputs the header format. This subroutine also allows for no data being present.

#MAIN is the processing loop that processes the function keys. It sets the record number to display and then calls other subroutines to perform the action.

STARTALL is the routine called from the #INIT that initializes and loads the subfiles. It is written as a separate subroutine to allow for additional code to be added that enables the starting point to be selected. (I've removed this code for this simplified version.) Notice that STARTALL sets DSPRRN to 1 and DSPRRNPRV to 0 because there is no previous record to "unhighlight."

NEXTGRP is a read routine that gets the next 10 records and writes them to the subfile format for the left-side subfile. As a customer may have one or more records in the input file, this subroutine selects the first record for each customer. A flag, LEFTEOF, is used to mark that EOF has been reached. Notice that this subroutine is reading ahead and always has one record left in the buffer that has not been moved to the display (unless we are at EOF).

READLEFT reads the records from the database. Because this function is kept in a separate subroutine, it's easier to add selection rules.

LEFTHDR is the main workhorse of the program. It first checks to see whether the record number requested is displayable. If we try and move before the start of the subfile, its value is reset to one. If we try to move beyond the end of the subfile, it will either read another group of 10 records or, if we are already at EOF,

FIGURE 3
Sample application source excerpt

```
*=====================================================================
 * Module Name   - WAJI26RB    Version   1.0      Date  21.02.2003
 * Based on      - WAJ026RM    Version            As at 05.08.2002
 * Author        - Paul Morris, Welwyn Support Services Ltd.
 * Project ID    - WA001
 * Function      - Duplicate customer review
 *                 iSeries NEWS version
 *
 *=====================================================================
H Copyright('Copyright (C) 2002-2003 Welwyn Support Services Ltd.')
H Debug Datedit(*DMY) Datfmt(*ISO)
 *=====================================================================
 * Warnings      -
 * Dependencies  -
 * Associated programs
 *               -
 * Service Programs
 *               -
 *=====================================================================
 * Modifications
 *---------------------------------------------------------------------
 * Change ID.    - XX999              Date   ../../..
 * Name          -
 * Description   -
 *
 *=====================================================================
 * display file
Fwaji26d   cf   E             workstn infsr(*PSSR)
F                                     sfile(#2sfl:#2rrn)
F                                     sfile(#4sfl:#4rrn)
 *---------------------------------------------------------------------
 * main customer file
Fcusnames  if   e           k disk    infsr(*PSSR)
 *---------------------------------------------------------------------
 * Matching records by original customer
Fagmtrcf2  if   e           k disk    infsr(*PSSR)
 *---------------------------------------------------------------------
 * Matching records by original and matched customer
Fagmtrcla  if   e           k disk    infsr(*PSSR) prefix(m_)
F                                     rename(agmtrcr1:agmtrcra)
 *=====================================================================
 * Parameter structures
d @action          s                 3
 *=====================================================================
d #2rrn            s                 4  0
d #2rrnsav         s                 4  0
d #4rrn            s                 4  0
d #4rrnsav         s                 4  0
d #2loopcount      s                 4  0
d dsprrn           s                 4  0
d dsprrnprv        s                 4  0
d dsprrnr          s                 4  0
d dsprrnrprv       s                 4  0
d lefteof          s                 1
d cusnsv           s                 8
 *=====================================================================
 * Key lists
 *---------------------------------------------------------------------
 * customer master
c     sl05k1       Klist
c                  Kfld                    cusn@5
 *---------------------
 * matched records
c     mtrck1       Klist
c                  Kfld                    m_cusnag
 *=====================================================================
c                  Exsr      #init

c                  Exsr      #main

c                  Exsr      #exit
 *=====================================================================
 * Program  initialization
 *---------------------------------------------------------------------
c     #init        Begsr

 * get the first batch of records (left side and right) and display

c                  Exsr      startall
 * If no records, exit from here
c                  If        #2rrnsav = 0
c                  Exfmt     #7wnd
c                  Exsr      #exit
c                  Endif

c                  Endsr
 *=====================================================================
 * Main Processing
 * Process the display, actioning the function keys
 *---------------------------------------------------------------------
```

FIGURE 3 *continued*

```
C     #main         Begsr

C                   Dow       *in03 = *off
C                   Select
 *                              ------------------------
 *                              Left side forward 1 line
C                   When      *in13
C                   Eval      dsprrn = dsprrn+1
C                   Exsr      lefthdr
 *                              ------------------------
 *                              Left side back 1 line
C                   When      *in14
C                   Eval      dsprrn = dsprrn-1
C                   Exsr      lefthdr
 *                              ------------------------
 *                              Right side forward 1 line
C                   When      *in23
C                   Eval      dsprrnr = dsprrnr+1
C                   Exsr      ritehdr
 *                              ------------------------
 *                              Right side back 1 line
C                   When      *in24
C                   Eval      dsprrnr = dsprrnr-1
C                   Exsr      ritehdr
 *                              Left side back 1 page
 *                              ------------------------
C                   When      *in26
C                   Eval      dsprrn = dsprrn-10
C                   Exsr      lefthdr
 *                              ------------------------
 *                              Left side forward 1 page
C                   When      *in27
C                   Eval      dsprrn = dsprrn+10
C                   Exsr      lefthdr
 *                              ------------------------
 *                              Action a highlighted pair
C                   When      *in10
C                   Exsr      prcmatch
C                   Endsl

 * now display what we have got
C                   Eval      rcdnbr#3 = dsprrn
C                   Write     #3ctl
C                   Exfmt     #1wnd
C                   Enddo
*                        of in03 loop

C                   Endsr
 *====================================================================
 * Initial load of subfiles for left and right sides
 *--------------------------------------------------------------------
C     startall      Begsr

 * initialise left subfile
C                   Eval      *in31 = *off
C                   Eval      *in32 = *off
C                   Eval      *in33 = *on
C                   Eval      #2rrnsav = 0
C                   Eval      lefteof = 'Y'
C                   Write     #3ctl
 * get the first record
C                   Exsr      readleft
C                   If        not %eof(agmtrcf2)
C                   Eval      lefteof = 'N'
C                   Eval      dsprrn = 1
C                   Eval      dsprrnprv = 0
C                   Eval      rcdnbr#3 = 1
C                   Eval      *in31 = *on
C                   Eval      *in32 = *on
C                   Eval      *in33 = *off
 * now load up the next group of records and load the headers
C                   Exsr      nextgrp
C                   Exsr      lefthdr
C                   Write     #3ctl
C                   Exfmt     #1wnd
C                   Endif
 *                       of eof check

C                   Endsr
 *====================================================================
 * Read next group of records
 * i.e., it provides the first record for a customer where 1 or more
 * duplicates exist for that customer
 *--------------------------------------------------------------------
C     nextgrp       Begsr

 * First read is already done
 * This also provides the read-ahead for the next time through

C                   Eval      #2loopcount = 0
C                   Eval      #2rrn = #2rrnsav
C                   Eval      *in33 = *off
```

continued

keep the pointer at the last record in the subfile.

When all this is done, LEFTHDR chains to the subfile using the previous record number in DSPRRNPRV and removes the highlight. Then, we use the new record number in DSPRRN to chain to the subfile to get the record to highlight. This also retrieves the customer number in a hidden field. The next job is to get the full customer details of this record and place the details in the control format. Finally, for this customer number, the program sets up the right-side window by calling two more subroutines.

LOADRITE uses the customer number of the selected record on the left and performs a "load all" of the subfile on the right. This reads, via a logical file, the duplicate records for the customer selected on the left. As with the left-side subfile, it keeps track of the highlights by the record numbers in fields DSPRRNR and DSPRRNPRVR. These fields are set to 1 and 0 each time the subfile is refreshed.

READRITE, like READLEFT, is a simple read of the duplicate records in a subroutine that allows additional selection logic to be inserted.

RITEHDR, like LEFTHDR, sets up the full address details and controls the highlight using the fields DSPRRNR and DSPRRNPRVR. The routine is simpler than LEFTHDR in that there is no need to check for additional records to read.

PRCMATCH is executed to process the selected pair of records. This calls another program, which is passed the customer numbers of both highlighted records. I won't discuss this program here as its actions don't affect this example program.

To view the DDS of the file for the duplicates and the customer master record, as well as the sources of the called program, retrieve all the code online at *www.iSeriesNetwork.com*.

Multiple Uses

This "point and shoot" method of controlling two subfiles in two windows without having to activate them is a powerful technique. By varying the

method of loading them, we can have possible uses that include

- ***Cash matching in Accounts Receivable.*** You could load up the left window with invoices and the right window with credit notes and cash. Here, you'd want to be able to highlight more than one record on either side.
- ***Service management maintenance contracts.*** Start off with a list of a customer's equipment on the left and an empty subfile on the right. Then, select items to place on contract and enter them to the subfile on the right.

I hope you agree that this method provides a programming tool that can make us more efficient in our work. ■

Paul Morris *works independently in the U.K. through his own company to provide programming and systems support for the iSeries. He can be contacted via e-mail at Paul@wssltd.demon.co.uk.*

FIGURE 3 *continued*

```
c                   Dow         not %eof(agmtrcf2)
c                               and #2loopcount < 10
 * skip subsequent records for this customer
c                   If          cusnsv <> cusnag
c                   Eval        cusn05 = cusnag
c                   Eval        cusnsv = cusnag
c                   Eval        cusn#2 = cusn05
c     sl05k1        Chain       cusnames
c                   If          not %found(cusnames)
c                   Eval        cnam05 = 'Missing Record!!!'
c                   Eval        cad405 = '???'
c                   Endif
c                   Eval        cnam#2 = cnam05
c                   Eval        cadd#2 = cad405

c                   Eval        #2rrn = #2rrn + 1
c                   Eval        #2loopcount = #2loopcount +1
c                   Eval        *in34 = *off
c                   Write       #2sfl
c                   Endif
 *                        of customer check
c                   Exsr        readleft

c                   Enddo

c                   If          %eof(agmtrcf2)
c                   Eval        lefteof = 'Y'
c                   Eval        *in33 = *on
c                   Endif
c
c                   Eval        #2rrnsav = #2rrn
c
c                   Endsr

 *====================================================================
 * Read a record for the left hand side
 * In the original version, there was a lot of selection processing,
 * hence the use of a subroutine
 *--------------------------------------------------------------------
c     readleft      Begsr

c                   read        agmtrcf2
c
c                   Endsr
 *====================================================================
 * prepare the detail in the left hand control record  (header)
 *--------------------------------------------------------------------
c     lefthdr       Begsr

 * check that the requested record is displayable, and adjust or get it
c                   Select
c                   When        dsprrn < 1
c                   Eval        dsprrn = 1
c                   When        dsprrn > #2rrnsav and lefteof = 'N'
c                   Exsr        nextgrp
c                   When        dsprrn > #2rrnsav and lefteof = 'Y'
c                   Eval        dsprrn = #2rrnsav
c                   Endsl

 * remove the previous highlight if the requested record has changed
c                   If          dsprrnprv > 0
c                               and dsprrnprv <> dsprrn
c     dsprrnprv     Chain       #2sfl
c                   Eval        *in34 = *off
c                   Update      #2sfl
c                   Endif

 * set the highlight and load the header details if the
 * requested record has changed
c                   If          dsprrnprv <> dsprrn
c     dsprrn        Chain       #2sfl
c                   Eval        *in34 = *on
c                   Update      #2sfl
c                   Eval        cusn05 = cusn#2
c     sl05k1        Chain       cusnames
c                   If          not %found
c                   Eval        cnam05 = 'Missing Record!!!'
c                   Eval        cad405 = '???'
c                   Endif
c                   Eval        cusn#3 = cusn#2
c                   Eval        cnam#3 = cnam05
c                   Eval        cad1#3 = cad105
c                   Eval        cad2#3 = cad205
c                   Eval        cad3#3 = cad305
c                   Eval        cad4#3 = cad405
c                   Eval        cad5#3 = cad505
c                   Eval        pscd#3 = pcd105 + pcd205
c                   Eval        phon#3 = phon05
 * now set up the right hand side for the requested record
c                   Exsr        loadrite
```

FIGURE 3 *continued*

```
c                   Exsr      ritehdr
c                   Endif
 *                        of dsprrn check

c                   Eval      dsprrnprv = dsprrn

c                   Endsr

*---------------------------------------------------------------
 * load right hand side
 * This can load 1 or more records for the record highlighted on
 * the left
 *---------------------------------------------------------------
c     loadrite      Begsr

 * initialize the subfile
c                   Eval      #4rrn = 0
c                   Eval      *in43 = *on
c                   Eval      *in41 = *off
c                   Eval      *in42 = *off
c                   Eval      *in43 = *off
c                   Write     #5ctl

 * using the customer number selected on the left get the
 * matching records
c                   Eval      m_cusnag = cusn#2
c     mtrck1        setll     agmtrcla
c                   Exsr      readrite

c                   Dow       not %eof(agmtrcla)
c                   Eval      cusn05 = m_mtcsag
c                   Eval      cusn#4 = cusn05
c     sl05k1        Chain     cusnames
c                   If        not %found(cusnames)
c                   Eval      cnam05 = 'Missing Record!!!'
c                   Eval      cad405 = '???'
c                   endif
c                   Eval      cnam#4 = cnam05
c                   Eval      cadd#4 = cad405

c                   Eval      cusn#4 = m_mtcsag
c                   Eval      #4rrn = #4rrn + 1
c                   Eval      *in44 = *off
c                   Write     #4sfl
c                   Exsr      readrite

c                   Enddo

c                   If        %eof(agmtrcla)
c                   Eval      *in43 = *on
c                   Endif
c
c                   Eval      #4rrnsav = #4rrn
c                   Eval      dsprrnr = 1
c                   Eval      dsprrnrprv = 0

c                   Endsr

*---------------------------------------------------------------
 * read a matching record that satisfies the parameters
 * In the original version, there was a lot of selection processing,
 * hence the use of a subroutine
 *---------------------------------------------------------------
c     readrite      Begsr

c     mtrck1        reade     agmtrcla

c                   Endsr

*---------------------------------------------------------------
 * prepare the detail in the right hand control record (header)
 *---------------------------------------------------------------
c     ritehdr       Begsr

 * check that the requested record is displayable, and adjust
c                   Select
c                   When      dsprrnr < 1
c                   Eval      dsprrnr = 1
c                   When      dsprrnr > #4rrnsav
c                   Eval      dsprrnr = dsprrnr - 1
c                   Endsl

 * remove the previous highlight if the requested record has changed
c                   If        dsprrnrprv > 0
c                             and dsprrnrprv <> dsprrnr
c     dsprrnrprv    Chain     #4sfl
c                   Eval      *in44 = *off
c                   Update    #4sfl
c                   Endif

 * set the highlight and load the header details if the
 * requested record has changed
c                   If        dsprrnrprv <> dsprrnr
c     dsprrnr       Chain     #4sfl
c                   Eval      *in44 = *on
c                   Update    #4sfl
c                   Eval      cusn05 = cusn#4
c     sl05k1        Chain     cusnames
c                   If        not %found(cusnames)
c                   Eval      cnam05 = 'Missing Record!!!'
c                   Eval      cad405 = '???'
c                   Endif
c                   Eval      cusn#5 = cusn#4
c                   Eval      cnam#5 = cnam05
c                   Eval      cad1#5 = cad105
c                   Eval      cad2#5 = cad205
c                   Eval      cad3#5 = cad305
c                   Eval      cad4#5 = cad405
c                   Eval      cad5#5 = cad505
c                   Eval      pscd#5 = pcd105 + pcd205
c                   Eval      phon#5 = phon05
c                   Endif

c                   Eval      dsprrnrprv = dsprrnr

c                   Eval      *in41 = *on
c                   Eval      *in42 = *on
c                   Eval      *in43 = *on
c                   Eval      rcdnbr#5 = dsprrnr
c                   Write     #5ctl

c                   Endsr

*---------------------------------------------------------------
 * Process a matched record
 *---------------------------------------------------------------
c     prcmatch      Begsr

c                   Eval      @action = ' '
c                   Call      'WAJI28RB'
c                   Parm                    @action
c                   Parm                    cusn#3
c                   Parm                    cusn#5
c                   Write     #5ctl

c                   Endsr

*---------------------------------------------------------------
 * Final Exit
 *---------------------------------------------------------------
c     #Exit         Begsr

 * closedown called program
c                   Eval      @action = 'LR'
c                   Call      'WAJI28RB'
c                   Parm                    @action
c                   Parm                    cusn#3
c                   Parm                    cusn#5

c                   Eval      *inlr = *on
c                   Return

c                   Endsr

*---------------------------------------------------------------
c     *PSSR         Begsr
 *---------------------------------------------------------------

c                   Dump

c                   Endsr     '*CANCL'

*---------------------------------------------------------------
```