28 VSAM

January 1998

update

In this issue

VSAM Update

Published by
Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38342
From USA: 01144 1635 38342
E-mail: xephon@compuserve.com

North American office
Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75067
USA
Telephone: 940 455 7050

Australian office
Xephon/RSM
GPO Box 6258
Halifax Street
Adelaide, SA 5000
Australia
Telephone: 08 223 1391

Contributions
Articles published in *VSAM Update* are paid for at the rate of £170 ($250) per 1000 words for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*.

Editor
Fiona Hewitt

Disclaimer
Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, EXECs, and other contents of this journal before making any use of it.

Subscriptions and back-issues
A year's subscription to *VSAM Update*, comprising four quarterly issues, costs £115.00 in the UK; $170.00 in the USA and Canada; £121.00 in Europe; £128.00 in Australasia and Japan; and £125.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the April 1991 issue, are available separately to subscribers for £29.00 ($43.00) each including postage.

VSAM Update on-line
Code from *VSAM Update* can be downloaded from our Web site at http://www.xephon.com; you will need the user-id shown on your address label.

Printed in England.

Updating data in an on-line VSAM file – continued

Here, we present the final part of the article begun in the July issue on updating data in an on-line VSAM file. This on-line utility program allows you to update any part of the VSAM record except the KSDS key.

ZZVU01

```
*************************************************************************
* THIS ROUTINE HANDLES DISPLAY AND REENTERING OF ERRORS
*************************************************************************
ERRORCHK EQU   *
         ST    R1Ø,HOLD1ØE
ERRRETRY BAL   R1Ø,EDITINP         EDIT THE DATA
         MVC   SPFKEYSO,MESSPFO
         CLC   ERRSWS,BLANKS
         BE    ERREXIT
ERSEND   EQU   *
         SPACE 2
* DON'T ALLOW CHANGE TO KEY AT THIS POINT (SET TO AUTOSKIP, ETC.)
         MVI   SKEYA,DFHBMASK       AUTOSKIP
         MVI   SKEYC,DFHBLUE        BLUE
         MVI   SKEYH,DFHDFHI        DEFAULT HILIGHT
         MVC   SPFKEYSO,MESSPFU
         BAL   R1Ø,MAPBUILD
         BAL   R1Ø,LOCKCHEK       IS RECORD OR FILE LOCKED?
         EXEC  CICS SEND MAP('MAPØ1') MAPSET('ZZVUSØ1') FREEKB CURSOR
         MVC   SMESSAGO,BLANKS
         EXEC  CICS HANDLE AID PF1(HELP) PF2(ERSCROLL) PF4             *
               PF5(EREXNOUP) PF6 PF7 PF8 PF9 PF1Ø PF11 PF12
         EXEC  CICS RECEIVE MAP('MAPØ1') MAPSET('ZZVUSØ1')
         BAL   R1Ø,STOREINP
         B     ERRRETRY            USER WANTS TO RE-EDIT DATA
         SPACE 2
* UPDATE ABORTED
EREXNOUP EQU   *
         MVC   SPFKEYSO,MESSPFO
         MVC   SMESSAGO,BLANKS
         MVI   USREQST,UNLOCK     CHECK IF RECORD TO BE 'UNLOCKED'
         BAL   R1Ø,PROCESS        GO PROCESS FILE REQUEST
         MVC   MESSAGS(L'MESSØ8),MESSØ8
         B     PUTUPSCR
         SPACE 2
* SCROLL ERRORS
ERSCROLL EQU   *
```

```
         MVC   ERMESSAG,BLANKS
LOOP     EQU   *
         AP    ERLAST,ONE
         CP    ERLAST,NUMMSGS   DID I DISPLAY THE HIGHEST MESSAGE?
         BH    RESTART          YES, THEN START OVER
         LA    R8,ERØ1          POINT TO FIRST SWITCH
         ZAP   DOUBLE,ERLAST
         SP    DOUBLE,ONE
         CVB   R7,DOUBLE        DISPLACEMENT
         AR    R8,R7            POINT TO NEXT SWITCH
         CLI   Ø(R8),ON
         BNE   LOOP
         SPACE 2
* I FOUND A SWITCH SET
CHKERØ1  EQU   *
         CP    ERLAST,=P'1'
         BNE   CHKERØ2
         MVC   ERMESSAG(L'ERMESSØ1),ERMESSØ1
         MVC   SDELOPTL,CURSOR
         B     EXSCROLL
CHKERØ2  EQU   *
         CP    ERLAST,=P'2'
         BNE   CHKERØ3
         MVC   ERMESSAG(L'ERMESSØ2),ERMESSØ2
         MVC   SSRCHLNL,CURSOR
         B     EXSCROLL
CHKERØ3  EQU   *
         CP    ERLAST,=P'3'
         BNE   CHKERØ4
         MVC   ERMESSAG(L'ERMESSØ3),ERMESSØ3
         MVC   SSRCHFML,CURSOR
         B     EXSCROLL
CHKERØ4  EQU   *
         CP    ERLAST,=P'4'
         BNE   CHKERØ5
         MVC   ERMESSAG(L'ERMESSØ4),ERMESSØ4
         MVC   SSRCHCNL,CURSOR
         B     EXSCROLL
CHKERØ5  EQU   *
         CP    ERLAST,=P'5'
         BNE   CHKERØ6
         MVC   ERMESSAG(L'ERMESSØ5),ERMESSØ5
         MVC   SSRCHDTL,CURSOR
         B     EXSCROLL
CHKERØ6  EQU   *
         CP    ERLAST,=P'6'
         BNE   CHKERØ7
         MVC   ERMESSAG(L'ERMESSØ6),ERMESSØ6
         MVC   SOPT1L,CURSOR
         B     EXSCROLL
```

```
CHKERØ7  EQU   *
         CP    ERLAST,=P'7'
         BNE   CHKERØ8
         MVC   ERMESSAG(L'ERMESSØ7),ERMESSØ7
         MVC   SXDISP1L,CURSOR
         B     EXSCROLL
CHKERØ8  EQU   *
         CP    ERLAST,=P'8'
         BNE   CHKERØ9
         MVC   ERMESSAG(L'ERMESSØ6),ERMESSØ6
         MVC   SOPT2L,CURSOR
         B     EXSCROLL
CHKERØ9  EQU   *
         CP    ERLAST,=P'9'
         BNE   CHKER1Ø
         MVC   ERMESSAG(L'ERMESSØ7),ERMESSØ7
         MVC   SXDISP2L,CURSOR
         B     EXSCROLL
CHKER1Ø  EQU   *
         CP    ERLAST,=P'1Ø'
         BNE   CHKER11
         MVC   ERMESSAG(L'ERMESSØ6),ERMESSØ6
         MVC   SOPT3L,CURSOR
         B     EXSCROLL
CHKER11  EQU   *
         CP    ERLAST,=P'11'
         BNE   RESTART
         MVC   ERMESSAG(L'ERMESSØ7),ERMESSØ7
         MVC   SXDISP3L,CURSOR
         B     EXSCROLL
RESTART  ZAP   ERLAST,ZERO
         B     LOOP
EXSCROLL EQU   *
         MVC   SMESSAGO(L'ERMESSAG),ERMESSAG
         MVC   SPFKEYSO,MESSPFU
         B     ERSEND
         SPACE 2
* EXIT ROUTINE
ERREXIT  EQU   *
         MVC   SPFKEYSO,MESSPFO
         L     R1Ø,HOLD1ØE
         BR    R1Ø
*
         EJECT
*************************************************************************
* THIS ROUTINE TRANSLATES 3Ø CHARACTERS TO 6Ø HEX OR
*                         6Ø HEX TO 3Ø CHARACTERS
*
* 'SWITCH'  INPUT  (X=CHAR TO HEX, C=HEX TO CHAR)
*           EXIT   (X'ØØ' = OKAY, X'FF' = PROBLEM)
*
```

```
* R7  = CHARACTER DATA
* R8  = HEX DATA
* R1Ø = WORK REGISTER
*
*********************************************************************
HEXCONV  EQU   *
         ST    R1Ø,HOLD1ØF
         STM   R7,R8,HOLDR7R8
         CLI   SWITCH,HEX
         BE    CHATOHEX
         CLI   SWITCH,CHAR
         BE    HEXTOCHA
         MVI   SWITCH,HIVALUE     INVALID REQUEST
         B     HEXCNVEX
         SPACE 2
*********************************************************************
* CONVERT 12 HEX TO 6 CHARS
*********************************************************************
HEXTOCHA EQU   *
         L     R1Ø,=A(5)
HTODLOOP EQU   *
         MVC   HEXDATA,Ø(R8)
         TRT   HEXDATA,VALIDHEX  ARE ALL CHARACTERS VALID?
         BZ    *+12               YES
         MVI   SWITCH,HIVALUE     NO - POST ERROR
         B     HEXCNVEX            ... AND EXIT
         TR    HEXDATA,TRDSPHEX  TRANSLATE 'DISPLAY HEX'
         PACK  CHARDATA(7),HEXDATA(13)
         MVC   Ø(6,R7),CHARDATA
         LA    R7,6(R7)
         LA    R8,12(R8)
         BCT   R1Ø,HTODLOOP
         MVI   SWITCH,LOVALUE
         B     HEXCNVEX
*
*********************************************************************
* CONVERT 6 CHARS TO 12 HEX
*********************************************************************
CHATOHEX EQU   *
         L     R1Ø,=A(5)
CTOHLOOP EQU   *
         MVC   CHARDATA,Ø(R7)
         UNPK  HEXDATA(13),CHARDATA(7)
         TR    HEXDATA,TRHEXDSP
         MVC   Ø(12,R8),HEXDATA
         LA    R7,6(R7)
         LA    R8,12(R8)
         BCT   R1Ø,CTOHLOOP
         MVI   SWITCH,LOVALUE     OKAY
         B     HEXCNVEX
*
```

```
*********************************************************************
* DATA CONVERTED - EXIT ROUTINE
*********************************************************************
HEXCNVEX EQU   *
         LM    R7,R8,HOLDR7R8
         L     R1Ø,HOLD1ØF
         BR    R1Ø
*
         EJECT
*********************************************************************
* CONVERT DATA IN DISPLAY FORMAT TO RBA
*   R7      - REG POINTING TO 16 BYTES OF INPUT TO CONVERT
*   R8      - REG POINTING TO 16 BYTES FOR STORING RBA IN DISPLAY FMT
*   INRBA   - INPUT FIELD CONTAINING UP TO 16 CHARS OF RBA
*   OUTRBA  - OUTPUT FIELD CONTAINING RBA NUMBER (DISPLAY FORMAT)
*   FILERBA - OUTPUT FIELD CONTAINING RBA NUMBER (FULLWORD)
*********************************************************************
DSPTORBA EQU   *
         ST    R1Ø,HOLD1ØK
         MVC   INRBA,Ø(R7)         INPUT
         ST    R13,DFHEIR13        SAVE R13
         LA    R13,DFHEISA         ADDRESS OF SAVE AREA
         CALL  UNEDIT,(INRBA,INLTH,OUTRBA,OUTLTH)
         L     R13,DFHEIR13        RESTORE R13
         MVC   Ø(16,R7),OUTRBA     OUTPUT RESULTS
         PACK  DOUBLE,OUTRBA
         CP    DOUBLE,ZERO
         BE    RBASET
         CLC   HIFILE,ESDSFILE
         BE    *+1Ø
         ZAP   OUTRBA,DOUBLE       ADJUST FOR CISZ
         DP    OUTRBA,ESDSCISZ
         MP    OUTRBA(13),ESDSCISZ
         ZAP   INRBA,DOUBLE        ADJUST FOR RECORD LENGTH
         SP    INRBA,OUTRBA(13)    PARTIAL CI
         DP    INRBA,PRECLEN
         MP    INRBA(13),PRECLEN
         AP    OUTRBA(13),INRBA(13) COMBINE BOTH ADJUSTMENTS
         ZAP   DOUBLE,OUTRBA(13)
RBASET   CVB   R2,DOUBLE
         ST    R2,FILERBA
         L     R1Ø,HOLD1ØK
         BR    R1Ø
*
*********************************************************************
* CONVERT RBA TO DISPLAY FORMAT
*   R8      - REG POINTING TO 16 BYTES FOR STORING RBA IN DISPLAY FMT
*********************************************************************
RBATODSP EQU   *
         ST    R1Ø,HOLD1ØL
```

```
         L     R2,FILERBA
         CVD   R2,DOUBLE
         UNPK  OUTRBA,DOUBLE
         MVZ   OUTRBA+15(1),SIGN
         MVC   Ø(16,R8),OUTRBA    OUTPUT RESULTS
         L     R1Ø,HOLD1ØL
         BR    R1Ø
*
         EJECT
*********************************************************************
* SET TIME AND POST TO MAP
*********************************************************************
SETTIME  EQU   *
         ST    R1Ø,HOLD1ØA
         EXEC  CICS ASKTIME ABSTIME(TIME)
         EXEC  CICS FORMATTIME ABSTIME(TIME)                          *
               MMDDYY(CURDATE) YEAR(YEAR) DATESEP('/')                *
               TIME(CURTIME) TIMESEP(':')
         L     R1Ø,YEAR
         CVD   R1Ø,DOUBLE
         UNPK  CURDATE+6(4),DOUBLE
         MVZ   CURDATE+9(1),SIGN
         MVC   SM1DATEO,CURDATE
         MVC   SM1TIMEO,CURTIME
         L     R1Ø,HOLD1ØA
         BR    R1Ø
         SPACE 2
*********************************************************************
* CLEAR MAP
*********************************************************************
CLEARMAP EQU   *
         ST    R1Ø,HOLD1ØC
         NULLS MAPØ1O,ZZVUSØ1T,R1Ø
         MVC   SXDISP1O(2),=C'4Ø'
         MVC   SXDISP1O+2(58),SXDISP1O
         MVC   SXDISP2O,SXDISP1O
         MVC   SXDISP3O,SXDISP1O
         L     R1Ø,HOLD1ØC
         BR    R1Ø
         SPACE 2
*********************************************************************
* SET UP DISPLAY
*********************************************************************
MAPBUILD EQU   *
         ST    R1Ø,HOLD1Ø
         BAL   R1Ø,SETTIME        RESET AND POST DATE/TIME
         UNPK  SBRWCNTO,SRCHMAX
         MVZ   SBRWCNTO+4(1),SIGN
         MVC   SFILEO,HIFILE
         MVC   SFSTCOLO,HIFSTCOL
```

```
         MVC   SDELOPTO,HIDELOPT
         MVC   SSRCHCLO,HISRCHCL
         MVC   SSRCHLNO,HISRCHLN
         MVC   SSRCHFMO,HISRCHFM
         MVC   SSRCHCNO,HISRCHCN
         MVC   SSRCHDTO,HISRCHDT
         CLI   FILETYPE,ESDS
         BE    MBESDS             IT'S ESDS
         B     MBKSDS             IT'S KSDS
         SPACE 2
* IT'S A KSDS FILE
MBKSDS   EQU   *
         MVC   SFILETPO,=C'KSDS'
* RECORD HELD, DON'T ALLOW CHANGE TO RECORD KEY
         CLI   RECLOCK,LOCK
         BNE   *+18
         LM    R7,R8,KEYLOC       LOADS LOC & LENGTH OF RECORD KEY
         STC   R8,*+5             POST LENGTH TO NEXT INSTRUCTION
         MVC   Ø(Ø,R7),FILEKEY    PREVENTS CHANGING THE RECORD KEY
* PUT CURRENT RECORD KEY TO DISPLAY
         MVC   HIKEY,BLANKS
         L     R8,KEYLEN
         STC   R8,*+5             POST LENGTH TO NEXT INSTRUCTION
         MVC   HIKEY(Ø),FILEKEY
         MVC   SKEYO,HIKEY
         TR    SKEYO,DISPLHEX
         MVC   SKEYLOCO,FKEYLOC
         MVC   SKEYLENO,FKEYLEN
         B     MBRECLEN
         SPACE 2
* IT'S AN ESDS FILE
MBESDS   EQU   *
         MVC   SFILETPO,=C'ESDS'
         MVC   SKEYO,HIKEY
         MVC   SKEYLOCO,=C' N/A '
         MVC   SKEYLENO,=C' N/A'
         B     MBRECLEN
         SPACE 2
MBRECLEN EQU   *
         SR    R2,R2
         ICM   R2,B'ØØ11',RECLENTH
* 1ØØ NULLS AFTER VALID DATA IN RECORD (SO GARBAGE WON'T DISPLAY)
         LA    R7,RECORD
         AR    R7,R2
         MVI   Ø(R7),NULL
         MVC   1(99,R7),Ø(R7)
*
         CVD   R2,DOUBLE
         UNPK  SRECLENO,DOUBLE
         MVZ   SRECLENO+4(1),SIGN
```

```
         MVC   SOPT1O,HIOPT1
         MVC   SOPT2O,HIOPT2
         MVC   SOPT3O,HIOPT3
         PACK  DOUBLE,HIFSTCOL
         SP    DOUBLE,ONE
         MVN   DOUBLE+7(1),SIGN
         CVB   R2,DOUBLE
         LA    R7,RECORD
         AR    R7,R2               POINT TO PART OF RECORD USER WANTS
         MVC   HICDISP1,Ø(R7)      1ST 3Ø CHARS TO DISPLAY
         MVC   HICDISP2,3Ø(R7)     2ND 3Ø CHARS TO DISPLAY
         MVC   HICDISP3,6Ø(R7)     3RD 3Ø CHARS TO DISPLAY
         SPACE 2
* CONVERT CHARACTER DISPLAY TO HEX
         LA    R7,HICDISP1
         LA    R8,HIXDISP1
         MVI   SWITCH,HEX          CONVERT TO HEX
         BAL   R1Ø,HEXCONV
         LA    R7,HICDISP2
         LA    R8,HIXDISP2
         MVI   SWITCH,HEX          CONVERT TO HEX
         BAL   R1Ø,HEXCONV
         LA    R7,HICDISP3
         LA    R8,HIXDISP3
         MVI   SWITCH,HEX          CONVERT TO HEX
         BAL   R1Ø,HEXCONV
         MVC   SCDISP1O,HICDISP1
         TR    SCDISP1O,DISPLHEX
         MVI   OPT1EXHX,BLANK
         TRT   SCDISP1O,HEXCHECK
         BZ    *+8
         MVI   OPT1EXHX,HEX
         MVC   SCDISP2O,HICDISP2
         TR    SCDISP2O,DISPLHEX
         MVI   OPT2EXHX,BLANK
         TRT   SCDISP2O,HEXCHECK
         BZ    *+8
         MVI   OPT2EXHX,HEX
         MVC   SCDISP3O,HICDISP3
         TR    SCDISP3O,DISPLHEX
         MVI   OPT3EXHX,BLANK
         TRT   SCDISP3O,HEXCHECK
         BZ    *+8
         MVI   OPT3EXHX,HEX
         MVC   SXDISP1O,HIXDISP1
         MVC   SXDISP2O,HIXDISP2
         MVC   SXDISP3O,HIXDISP3
         SPACE 2
* GET RECORD COLUMN HEADINGS
         MVI   COLFMAT,CHAR
```

```
         MVI   COLHEAD,C'1'
         BAL   R1Ø,COLHEADS
         MVC   SCHEAD1O,HEADING
         MVI   COLHEAD,C'2'
         BAL   R1Ø,COLHEADS
         MVC   SCHEAD2O,HEADING
         MVI   COLHEAD,C'3'
         BAL   R1Ø,COLHEADS
         MVC   SCHEAD3O,HEADING
         MVI   COLFMAT,HEX
         MVI   COLHEAD,C'1'
         BAL   R1Ø,COLHEADS
         MVC   SXHEAD1O,HEADING
         MVI   COLHEAD,C'2'
         BAL   R1Ø,COLHEADS
         MVC   SXHEAD2O,HEADING
         MVI   COLHEAD,C'3'
         BAL   R1Ø,COLHEADS
         MVC   SXHEAD3O,HEADING
         L     R1Ø,HOLD1Ø
         BR    R1Ø
*
*************************************************************************
* DETERMINE COLUMN HEADINGS
*************************************************************************
COLHEADS EQU   *
         ST    R1Ø,HOLD1ØH
         L     R7,=A(HEADING+4)
         MVC   HEADING,BLANKS
         CLI   COLFMAT,HEX
         BE    COLHEX
* CHARACTER HEADINGS
         MVC   HEADING(3),=C'C/X'
         PACK  COLUMN,HIFSTCOL
         CLI   COLHEAD,C'2'
         BNE   *+1Ø
         AP    COLUMN,=P'3Ø'
         CLI   COLHEAD,C'3'
         BNE   *+1Ø
         AP    COLUMN,=P'6Ø'
* CLEAR (CHAR)
         MVI   Ø(R7),C'.'
         MVC   1(29,R7),Ø(R7)
* 4 POSITION (CHAR)
         UNPK  Ø(4,R7),COLUMN
         MVZ   3(1,R7),SIGN
         CLI   Ø(R7),CHARØ
         BNE   *+2Ø
         MVI   Ø(R7),BLANK
         CLI   1(R7),CHARØ
```

```
         BNE   *+8
         MVI   1(R7),BLANK
* 2 POSITION (CHAR)
         LA    R7,6(R7)
         L     R8,=A(6)
         SP    COLUMN,ONE
COLHCLP  AP    COLUMN,=P'5'
         UNPK  Ø(2,R7),COLUMN
         MVZ   1(1,R7),SIGN
         LA    R7,5(R7)
         BCT   R8,COLHCLP
         B     COLHEDEX
*
         SPACE 2
* HEX HEADINGS
COLHEX   PACK  COLUMN,HIFSTCOL
         CLI   COLHEAD,C'2'
         BNE   *+1Ø
         AP    COLUMN,=P'3Ø'
         CLI   COLHEAD,C'3'
         BNE   *+1Ø
         AP    COLUMN,=P'6Ø'
* CLEAR (HEX)
         MVC   Ø(2,R7),=C'. '
         MVC   2(58,R7),Ø(R7)
* 4 POSITION (HEX)
         UNPK  1(4,R7),COLUMN
         MVZ   4(1,R7),SIGN
         CLI   1(R7),CHARØ
         BNE   *+2Ø
         MVI   1(R7),BLANK
         CLI   2(R7),CHARØ
         BNE   *+8
         MVI   2(R7),BLANK
* 2 POSITION (HEX)
         LA    R7,11(R7)
         L     R8,=A(6)
         SP    COLUMN,ONE
COLHXLP  AP    COLUMN,=P'5'
         UNPK  Ø(2,R7),COLUMN
         MVZ   1(1,R7),SIGN
         LA    R7,1Ø(R7)
         BCT   R8,COLHXLP
         B     COLHEDEX
*
COLHEDEX L     R1Ø,HOLD1ØH
         BR    R1Ø
*
         EJECT
*
```

```
* ********************************************************************
* ONLINE HELP
*  NOTE: THIS OPTION HAS BEEN NOP'D BECAUSE THE CODING FOR THE ON-
*  LINE HELP PROGRAMS HAS NOT BEEN INCLUDED WITH THIS ARTICLE. I
*  LEFT THE ROUTINE IN BECAUSE I MAY SUBMIT THAT ARTICLE IN
*  FUTURE.
********************************************************************
HELP     EQU   *
*      EXEC  CICS LINK PROGRAM('HELPØ1') COMMAREA(PASSINFO)          *
*            LENGTH(PSCOMLEN)
         B     RETURN
         SPACE 2
********************************************************************
* RETURN TO CICS
********************************************************************
* EXPECT TO COME BACK
RETURN   EQU   *
         MVC   MESSAGS,BLANKS
         CLI   RECLOCK,LOCK      IF FILE RECORD HELD FOR UPDATE
         BE    REENTER           KEEP PROGRAM CONVERSATIONAL
         MVC   COMLEN,=AL2(COMAREAE-COMAREAS) LENGTH OF COMMAREA
         EXEC  CICS RETURN TRANSID('ZZVU') COMMAREA(COMAREAS)          *
               LENGTH(COMLEN)
         SPACE 3
* EXIT TRANSACTION
EXIT     EQU   *
         EXEC  CICS XCTL PROGRAM('SIGNOFF')
         SPACE 3
*
         EJECT
********************************************************************
* ERROR HANDLING ROUTINES
********************************************************************
         SPACE 2
* NO RECORD FOUND
ERNOTFND EQU   *
         MVC   MESSAGS(L'MESSØ1),MESSØ1
         B     DISPERR
         SPACE 2
* FILE NOT OPEN
ERNOTOPN EQU   *
         MVC   MESSAGS(L'MESSØ2),MESSØ2
         B     DISPERR
         SPACE 2
* MAP FAIL
ERMAPFL  EQU   *
         MVC   MESSAGS(L'MESSØ3),MESSØ3
         MVC   SMESSAGO,MESSAGS
         BAL   R1Ø,SETTIME
         BAL   R1Ø,LOCKCHEK       IS RECORD OR FILE LOCKED?
```

```
         EXEC  CICS SEND MAP('MAPØ1') MAPSET('ZZVUSØ1') FREEKB FRSET
         MVC   SMESSAGO,BLANKS
         B     RETURN             COME BACK TO INITIAL SCREEN
         SPACE 2
* MISCELLANEOUS ERROR
ERMISC   EQU   *
         EXEC  CICS HANDLE CONDITION ERROR
         MVC   MESSAGS(L'MESSØ4),MESSØ4
         B     DISPERR
         SPACE 2
* FILE LENGTH ERROR
ERLENTH  EQU   *
         MVC   MESSAGS(L'MESSØ5),MESSØ5
         B     DISPERR
         SPACE 2
* BAD RBA (ESDS ONLY)
ERBADRBA EQU   *
         BAL   R1Ø,NOSRCHIT        DISPLAY 'NO RECORD FOUND'
         MVC   MESSAGS(L'MESS15),MESS15
         B     DISPERR
         SPACE 2
* AN ERROR HAS OCCURRED. DISPLAY APPROPRIATE MESSAGE.
DISPERR  EQU   *
         MVC   SMESSAGO,MESSAGS    MOVE MESSAGE TO OUTPUT
         BAL   R1Ø,MAPBUILD
         BAL   R1Ø,LOCKCHEK       IS RECORD OR FILE LOCKED?
         EXEC  CICS SEND MAP('MAPØ1') MAPSET('ZZVUSØ1') FREEKB
         MVC   SMESSAGO,BLANKS
         B     RETURN
         SPACE 2
* DISPLAY NO RECORD FOUND
NOSRCHIT EQU   *
         ST    R1Ø,HOLD1ØJ
         MVC   SCDISP1O,BLANKS
         MVC   SCDISP2O,BLANKS
         MVC   SCDISP3O,BLANKS
         MVC   SXDISP1O(2),=C'4Ø'
         MVC   SXDISP1O+2(58),SXDISP1O
         MVC   SXDISP2O,SXDISP1O
         MVC   SXDISP3O,SXDISP1O
         MVC   SCDISP1O(L'NORECFND),NORECFND
         MVC   SCDISP2O(L'NORECFND),NORECFND
         MVC   SCDISP3O(L'NORECFND),NORECFND
         L     R1Ø,HOLD1ØJ
         BR    R1Ø
*
         SPACE 2
* ********************************************************************
* CICS PRECOMPILER WILL NOT ALLOW CERTAIN CICS COMMANDS TO BE USED IN
* ASSEMBLER MACROS. THE FOLLOWING CODE WILL GET AROUND THAT.
```

```
* BASICALLY THE 'FILEKSDS' & 'FILEESDS' MACROS STORE THE ADDRESSES
* WITHIN THE MACROS WHERE THE ACTION SHOULD OCCUR. THE FOLLOWING CODE
* SIMPLY LOADS THE ADDRESSES AND GOES TO THE CODE WITHIN THE MACROS.
*********************************************************************
ENDNXT   L     R2,ADDRENXT
         BR    R2
ENDPRV   L     R2,ADDREPRV
         BR    R2
NENDBR   L     R2,ADDRNEBR
         BR    R2
PENDBR   L     R2,ADDRPEBR
         BR    R2
ENDSCH   L     R2,ADDRESCR
         BR    R2
RNDSCH   L     R2,ADDRRESC
         BR    R2
*
         EJECT
* *******************************************************************
* THE FOLLOWING TABLE IS FOR ACCESSING THE FILE PROCESSING
* SUBROUTINES
*********************************************************************
*
*                               |--CONSTANT 'FILE' FOLLOWED BY
*                               |  ID FROM FILEKSDS OR FILEESDS MACRO.
*                               |          |
*                               |          |---P (FILE UPDATE ON PROD)
*                               |          |
*                               |          ||--T (FILE UPDATE ON TEST)
*                               |          ||
*                               |          |||-I (FILE UPDATE ON ICCF)
*                               |          |||
*                   FILENAME    V          VVV
         DS    ØF
FILETAB  EQU   *
         DC    CL8'ARMSTK  ',A(FILEAA),C' TI ' PROBLEM MGT LOG FILE
         DC    CL8'ARHISE  ',A(FILEAB),C'PT  ' CONTRACT HISTORY
         DC    CL8'APMSTK  ',A(FILEAC),C'PT  ' A/R CUST NAME/ADDR
                                                 FILE
         DC    CL8'APHISE  ',A(FILEAD),C'PT  ' A/R WORKSTATION DETAIL
         DC    X'FF'      END OF TABLE
         SPACE 2
*********************************************************************
         EJECT
*********************************************************************
* CONSTANTS AND WORK AREAS
*********************************************************************
         LTORG
LOGLEN   DC    H'8'
PROGID   DC    CL8'ZZVUØ1  '
```

```
* THE FOLLOWING FIELDS ARE FOR ACCESSING ONLINE HELP
PASSINFO EQU   *
PSSCRID  DC    CL7'ZZVUSØ1'       SCREEN ID (MAPSET)
PSMAPID  DC    CL7'MAPØ1'         MAP ID
PSCOMLEN DC    H'14'              LENGTH OF ABOVE 2 FIELDS
*
* KEEP FOLLOWING 2 FIELDS TOGETHER
HEXDATA  DS    CL12
         DS    CL1
* KEEP FOLLOWING 2 FIELDS TOGETHER
CHARDATA DS    CL6
         DS    CL1
*
PROD     DC    C'PROD'
TEST     DC    C'TEST'
ICCF     DC    C'ICCF'
SYSID    DS    CL4                SYSTEM ID (PROD,TEST, OR ICCF)
ZEROES   DC    5C'Ø'
YES      DC    C'YES'
NO       DC    C'NO '
SWITCH   DS    CL1
WORK     DS    CL5
WORKI    DS    CL5
WORK6Ø   DS    CL6Ø
DOUBLE   DS    D
LENGTH   DS    D
HALFØ    DC    H'Ø'
CURSOR   DC    H'-1'
SIGN     DC    X'FC'
ZERO     DC    P'Ø'
ONE      DC    P'1'
NINETY   DC    P'9Ø'
COLHEAD  DS    CL1   1,2, OR 3
COLFMAT  DS    CL1   HEX OR CHAR
COLUMN   DS    PL3
HEADING  DS    CL78
BLANKS   DC    CL8Ø' '
EDIT     DC    X'4020202020202020202021'
NORECFND DC    C'*** NO MATCH FOUND ***'
FLDLTH   DS    H
INLTH    DC    H'16'
OUTLTH   DC    H'16'
INRBA    DS    CL16
OUTRBA   DS    CL16
LOVALUES DC    4X'ØØ'
KSDS     EQU   C'K'
ESDS     EQU   C'E'
CHARØ    EQU   C'Ø'
BLANK    EQU   C' '
LOVALUE  EQU   X'ØØ'
```

```
HIVALUE  EQU   X'FF'
ON       EQU   X'FØ'
NULL     EQU   X'ØØ'
DELETE   EQU   C'D'
UPDATE   EQU   C'U'
NEXT     EQU   C'N'
PREVIOUS EQU   C'P'
READ     EQU   C'R'
UNLOCK   EQU   C'K'
LOCK     EQU   C'L'
SHFTLEFT EQU   C'<'
SHFTRGHT EQU   C'>'
GT       EQU   C'>'
LT       EQU   C'<'
EQ       EQU   C'='
CHAR     EQU   C'C'
HEX      EQU   C'X'
Y        EQU   C'Y'
N        EQU   C'N '
*
* ERROR MESSAGES (INDIVIDUAL FIELDS MAX SIZE=5Ø)
NUMMSGS  DC    P'11'   NUMBER OF ERRORS (ERØ1-ER11)
ERMESSØ1 DC    C'DELETE OPTION IS NOT YES OR NO.'
ERMESSØ2 DC    C'LENGTH OF SEARCH DATA NOT VALID.'
ERMESSØ3 DC    C'SEARCH DATA FORMAT NOT C OR X.'
ERMESSØ4 DC    C'SEARCH CONDITION NOT >, <, OR =.'
ERMESSØ5 DC    C'SEARCH DATA SHOULD BE BUT ISNT HEX.'
ERMESSØ6 DC    C'CONTAINS HEX DATA. UPDATE IN HEX MODE.'
ERMESSØ7 DC    C'HEX DISPLAY/INPUT NOT HEX.'
* ERROR MESSAGES
MESSØ1   DC    C'NO RECORD FOUND FOR THIS KEY'
MESSØ2   DC    C'FILE NOT OPEN'
MESSØ3   DC    C'MAP FAIL, PRESS CLEAR THEN REENTER DATA'
MESSØ4   DC    C'MISCELLANEOUS ERROR'
MESSØ5   DC    C'FILE LENGTH ERROR'
MESSØ6   DC    C'RECORD HAS BEEN UPDATED'
MESSØ7   DC    C'RECORD HAS BEEN DELETED'
MESSØ8   DC    C'REQUEST ABORTED'
MESSØ9   DC    C'FILE NOT DEFINED FOR PROCESSING.'
MESS1Ø   DC    C'READING > LAST RECORD, PRESS PF7 TO GET LAST'
MESS11   DC    C'READING < FIRST RECORD, PRESS PF8 TO GET FIRST'
MESS12   DC    C' XXXXXXXXX RECORDS SEARCHED. NO HIT. PRESS ENTER TO
                   CONTINUE.'
MESS13   DC    C'END OF FILE. PRESS ANY ACTION KEY TO RESET.'
MESS14   DC    C'VSAM DOES NOT SUPPORT ESDS RECORD DELETION.'
MESS15   DC    C'INVALID RBA (PROBABLY PAST EOF). TRY ANOTHER.'
MESS16   DC    C'DELETE NOT ALLOWED UNLESS DELETE RECORD = YES'
MESSAGS  DS    CL78
MESSPFO  DC    CL78'PF1=HELP 4=EXIT 5=RESET 6=DELETE 7=PREV 8=NEXT 9=UP*
               DATE 1Ø=<< 11=>> 12=RELEASE'
```

```
MESSPFU  DC    CL78'ENTER=RE-EDIT   PF1=HELP   PF2=SCROLL-ERRORS   PF5=*
               EXIT-NOUPDATE'
*
         LTORG
         EJECT
         SPACE 3
* ********************************************************************
* THE FOLLOWING PAGES CONTAIN THE FILE HANDLING SUBROUTINES SET UP
* VIA MACRO CALL.
********************************************************************
FILETAB2 EQU   *
         FILEKSDS ID=AA,FILE=ARMSTK,KLOC=1,KLEN=12,MRECL=125Ø
         FILEESDS ID=AB,FILE=ARHISE,MRECL=24Ø,CISIZE=Ø6144
         FILEKSDS ID=AC,FILE=APMSTK,KLOC=1,KLEN=Ø8,MRECL=33Ø
         FILEESDS ID=AD,FILE=APHISE,MRECL=275,CISIZE=Ø6144
         EJECT
ZZVUØ1E  EQU   *
         LTORG
* CONVERT DISPLAY HEX TO '.'
DISPLHEX EQU   *  Ø 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B'  Ø
         DC    X'4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B'  1
         DC    X'4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B'  2
         DC    X'4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B'  3
         DC    X'4Ø4B4B4B4B4B4B4B4B4B4A4B4C4D4E4F'  4
         DC    X'5Ø4B4B4B4B4B4B4B4B4B5A5B5C5D5E5F'  5
         DC    X'6Ø614B4B4B4B4B4B4B4B6A6B6C6D6E6F'  6
         DC    X'4B4B4B4B4B4B4B4B4B797A7B7C7D7E7F'  7
         DC    X'4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B'  8
         DC    X'4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B'  9
         DC    X'4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B'  A
         DC    X'4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B4B'  B
         DC    X'C0C1C2C3C4C5C6C7C8C94B4B4B4B4B4B'  C
         DC    X'D0D1D2D3D4D5D6D7D8D94B4B4B4B4B4B'  D
         DC    X'E04BE2E3E4E5E6E7E8E94B4B4B4B4B4B'  E
         DC    X'F0F1F2F3F4F5F6F7F8F94B4B4B4B4B4B'  F
* CHECK FOR HEX CHARS IN DISPLAY DATA
HEXCHECK EQU   *  Ø 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  Ø
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  1
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  2
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  3
         DC    X'ØØØØØØØØØØØØØØØØØØØØ4BØØØØØØØØ'  4
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  5
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  6
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  7
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  8
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  9
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  A
         DC    X'ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ'  B
```

```
         DC    X'00000000000000000000000000000000'  C
         DC    X'00000000000000000000000000000000'  D
         DC    X'00000000000000000000000000000000'  E
         DC    X'00000000000000000000000000000000'  F
* VALID HEX CHARACTERS
VALIDHEX EQU   *  Ø 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  Ø
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  1
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  2
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  3
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  4
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  5
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  6
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  7
         DC    X'FF000000000000FFFFFFFFFFFFFFFFFF'  8
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  9
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  A
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  B
         DC    X'FF000000000000FFFFFFFFFFFFFFFFFF'  C
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  D
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'  E
         DC    X'00000000000000000000FFFFFFFFFFFF'  F
* TRANSLATE DISPLAY HEX TO FORMAT REQUIRED TO CONVERT TO ACTUAL HEX
TRDSPHEX EQU   *  Ø 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'00000000000000000000000000000000'  Ø
         DC    X'00000000000000000000000000000000'  1
         DC    X'00000000000000000000000000000000'  2
         DC    X'00000000000000000000000000000000'  3
         DC    X'00000000000000000000000000000000'  4
         DC    X'00000000000000000000000000000000'  5
         DC    X'00000000000000000000000000000000'  6
         DC    X'00000000000000000000000000000000'  7
         DC    X'000A0B0C0D0E0F00000000000000000'  8
         DC    X'00000000000000000000000000000000'  9
         DC    X'00000000000000000000000000000000'  A
         DC    X'00000000000000000000000000000000'  B
         DC    X'000A0B0C0D0E0F00000000000000000'  C
         DC    X'00000000000000000000000000000000'  D
         DC    X'00000000000000000000000000000000'  E
         DC    X'000102030405060708090000000000000'  F
* TRANSLATE ACTUAL HEX TO FORMAT REQUIRED TO CONVERT TO DISPLAY HEX
TRHEXDSP EQU   *  Ø 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'00000000000000000000000000000000'  Ø
         DC    X'00000000000000000000000000000000'  1
         DC    X'00000000000000000000000000000000'  2
         DC    X'00000000000000000000000000000000'  3
         DC    X'00000000000000000000000000000000'  4
         DC    X'00000000000000000000000000000000'  5
         DC    X'00000000000000000000000000000000'  6
         DC    X'00000000000000000000000000000000'  7
```

```
         DC    X'00000000000000000000000000000000'  8
         DC    X'00000000000000000000000000000000'  9
         DC    X'00000000000000000000000000000000'  A
         DC    X'00000000000000000000000000000000'  B
         DC    X'00000000000000000000000000000000'  C
         DC    X'00000000000000000000000000000000'  D
         DC    X'00000000000000000000000000000000'  E
         DC    X'F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6'  F
*
         EJECT
         COPY  DFHBMSCA
         COPY  DFHAID
         END   ZZVU01
```

ZZVUS01

```
*************************************************************************
* MAPSET=ZZVUS01  MAP=MAP01    NAME=ONLINE VSAM FILE UPDATE
*************************************************************************
ZZVUS01  DFHMSD TYPE=&SYSPARM,MODE=INOUT,CTRL=(FREEKB,FRSET),          *
               TIOAPFX=YES,STORAGE=AUTO,LANG=ASM
MAP01  DFHMDI SIZE=(24,80),LINE=1,COLUMN=1,MAPATTS=(COLOR,HILIGHT),    *
               DSATTS=(COLOR,HILIGHT),EXTATT=YES
SPROGID  DFHMDF LENGTH=006,POS=(01,01),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(FSET,ASKIP),                                     *
               INITIAL='ZZVU01'
         DFHMDF LENGTH=001,POS=(01,17),                                *
               COLOR=NEUTRAL,HILIGHT=REVERSE,                          *
               ATTRB=(BRT,ASKIP)
               INITIAL='COMPANY NAME     '
         DFHMDF LENGTH=001,POS=(01,59),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
         DFHMDF LENGTH=005,POS=(01,64),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP),                                          *
               INITIAL='DATE:'
SM1DATE  DFHMDF LENGTH=010,POS=(01,70),                                *
               COLOR=NEUTRAL,HILIGHT=REVERSE,                          *
               ATTRB=(BRT,ASKIP),                                      *
               INITIAL='MM/DD/YYYY'
         DFHMDF LENGTH=011,POS=(02,01),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(FSET,ASKIP),                                     *
               INITIAL='ZZVUS01(01)'
         DFHMDF LENGTH=034,POS=(02,20),                                *
               COLOR=YELLOW,HILIGHT=REVERSE,                           *
               ATTRB=(ASKIP),                                          *
```

```
              INITIAL=' V S A M    F I L E    U P D A T E'
         DFHMDF LENGTH=001,POS=(02,55),                                *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP)
         DFHMDF LENGTH=005,POS=(02,64),                                *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP),                                           *
              INITIAL='TIME:'
SM1TIME  DFHMDF LENGTH=008,POS=(02,70),                                *
              COLOR=NEUTRAL,HILIGHT=REVERSE,                           *
              ATTRB=(BRT,ASKIP),                                       *
              INITIAL='HH:MM:SS'
         DFHMDF LENGTH=001,POS=(02,79),                                *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP)
         DFHMDF LENGTH=010,POS=(04,01),                                *
              COLOR=BLUE,HILIGHT=REVERSE,                              *
              ATTRB=(ASKIP),                                           *
              INITIAL='File-name:'
SFILE    DFHMDF LENGTH=007,POS=(04,13),                                *
              COLOR=GREEN,HILIGHT=UNDERLINE,                           *
              ATTRB=(FSET,IC)
         DFHMDF LENGTH=001,POS=(04,21),                                *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP)
         DFHMDF LENGTH=026,POS=(04,25),                                *
              COLOR=BLUE,HILIGHT=REVERSE,                              *
              ATTRB=(ASKIP),                                           *
              INITIAL='Display Start Position is:'
SFSTCOL  DFHMDF LENGTH=004,POS=(04,52),                                *
              COLOR=GREEN,HILIGHT=UNDERLINE,                           *
              ATTRB=(FSET,NUM)
         DFHMDF LENGTH=001,POS=(04,57),                                *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP)
         DFHMDF LENGTH=014,POS=(04,60),                                *
              COLOR=BLUE,HILIGHT=REVERSE,                              *
              ATTRB=(ASKIP),                                           *
              INITIAL='Delete Record?'
SDELOPT  DFHMDF LENGTH=003,POS=(04,76),                                *
              COLOR=GREEN,HILIGHT=UNDERLINE,                           *
              ATTRB=(FSET)
         DFHMDF LENGTH=001,POS=(04,80),                                *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP)
         DFHMDF LENGTH=017,POS=(05,01),                                *
              COLOR=BLUE,HILIGHT=REVERSE,                              *
              ATTRB=(ASKIP),                                           *
              INITIAL='KEY-KSDS/RBA-ESDS'
SKEY     DFHMDF LENGTH=060,POS=(05,19),                                *
```

```
              COLOR=GREEN,HILIGHT=UNDERLINE,                           *
              ATTRB=(FSET),                                            *
              INITIAL='                                                *
                              '
          DFHMDF LENGTH=001,POS=(05,80),                               *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP)
          DFHMDF LENGTH=009,POS=(06,01),                               *
              COLOR=BLUE,HILIGHT=BLINK,                                *
              ATTRB=(ASKIP),                                           *
              INITIAL='If SEARCH'
          DFHMDF LENGTH=013,POS=(06,11),                               *
              COLOR=BLUE,HILIGHT=REVERSE,                              *
              ATTRB=(ASKIP),                                           *
              INITIAL='Enter column:'
SSRCHCL   DFHMDF LENGTH=004,POS=(06,25),                               *
              COLOR=GREEN,HILIGHT=UNDERLINE,                           *
              ATTRB=(NUM)
          DFHMDF LENGTH=001,POS=(06,30),                               *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP)
          DFHMDF LENGTH=007,POS=(06,33),                               *
              COLOR=BLUE,HILIGHT=REVERSE,                              *
              ATTRB=(ASKIP),                                           *
              INITIAL='Length:'
SSRCHLN   DFHMDF LENGTH=002,POS=(06,41),                               *
              COLOR=GREEN,HILIGHT=UNDERLINE,                           *
              ATTRB=(NUM)
          DFHMDF LENGTH=001,POS=(06,44),                               *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP)
          DFHMDF LENGTH=007,POS=(06,47),                               *
              COLOR=BLUE,HILIGHT=REVERSE,                              *
              ATTRB=(ASKIP),                                           *
              INITIAL='Format:'
SSRCHFM   DFHMDF LENGTH=001,POS=(06,55),                               *
              COLOR=GREEN,HILIGHT=UNDERLINE,                           *
              ATTRB=(**-ATTRIB-40-???-**)
          DFHMDF LENGTH=001,POS=(06,57),                               *
              COLOR=BLUE,HILIGHT=OFF,                                  *
              ATTRB=(ASKIP)
          DFHMDF LENGTH=010,POS=(06,60),                               *
              COLOR=BLUE,HILIGHT=REVERSE,                              *
              ATTRB=(ASKIP),                                           *
              INITIAL='Condition:'
SSRCHCN   DFHMDF LENGTH=001,POS=(06,71),                               *
              COLOR=GREEN,HILIGHT=UNDERLINE,                           *
              ATTRB=(**-ATTRIB-40-???-**),                             *
              INITIAL=' '
          DFHMDF LENGTH=001,POS=(06,73),                               *
```

```
                COLOR=BLUE,HILIGHT=OFF,                                *
                ATTRB=(ASKIP)
          DFHMDF LENGTH=Ø12,POS=(Ø7,Ø6),                               *
                COLOR=BLUE,HILIGHT=REVERSE,                            *
                ATTRB=(ASKIP),                                         *
                INITIAL='Search Data:'
SSRCHDT   DFHMDF LENGTH=Ø6Ø,POS=(Ø7,19),                               *
                COLOR=GREEN,HILIGHT=UNDERLINE,                         *
                ATTRB=(**-ATTRIB-4Ø-???-**),                           *
                INITIAL='                                              *
                         '
          DFHMDF LENGTH=ØØ1,POS=(Ø7,8Ø),                               *
                COLOR=BLUE,HILIGHT=OFF,                                *
                ATTRB=(ASKIP)
SLOCKMS   DFHMDF LENGTH=Ø16,POS=(Ø8,Ø1),                               *
                COLOR=RED,HILIGHT=BLINK,                               *
                ATTRB=(ASKIP)
          DFHMDF LENGTH=ØØ1,POS=(Ø8,18),                               *
                COLOR=BLUE,HILIGHT=OFF,                                *
                ATTRB=(ASKIP)
          DFHMDF LENGTH=Ø28,POS=(Ø8,45),                               *
                COLOR=BLUE,HILIGHT=OFF,                                *
                ATTRB=(ASKIP),                                         *
                INITIAL='Maximum records to search =>'
SBRWCNT   DFHMDF LENGTH=ØØ5,POS=(Ø8,74),                               *
                COLOR=GREEN,HILIGHT=UNDERLINE,                         *
                ATTRB=(NUM)
          DFHMDF LENGTH=ØØ1,POS=(Ø8,8Ø),                               *
                COLOR=BLUE,HILIGHT=OFF,                                *
                ATTRB=(ASKIP)
          DFHMDF LENGTH=ØØ9,POS=(Ø9,Ø1),                               *
                COLOR=BLUE,HILIGHT=OFF,                                *
                ATTRB=(ASKIP),                                         *
                INITIAL='File-type'
SFILETP   DFHMDF LENGTH=ØØ4,POS=(Ø9,12),                               *
                COLOR=NEUTRAL,HILIGHT=OFF,                             *
                ATTRB=(BRT,ASKIP)
          DFHMDF LENGTH=ØØ1,POS=(Ø9,17),                               *
                COLOR=BLUE,HILIGHT=OFF,                                *
                ATTRB=(ASKIP)
          DFHMDF LENGTH=Ø13,POS=(Ø9,2Ø),                               *
                COLOR=BLUE,HILIGHT=OFF,                                *
                ATTRB=(ASKIP),                                         *
                INITIAL='Record Length'
SRECLEN   DFHMDF LENGTH=ØØ5,POS=(Ø9,35),                               *
                COLOR=NEUTRAL,HILIGHT=OFF,                             *
                ATTRB=(BRT,ASKIP),                                     *
                INITIAL='     '
          DFHMDF LENGTH=ØØ1,POS=(Ø9,41),                               *
                COLOR=BLUE,HILIGHT=OFF,                                *
```

```
               ATTRB=(ASKIP)
          DFHMDF LENGTH=Ø12,POS=(Ø9,42),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP),                                          *
               INITIAL='Key location'
SKEYLOC   DFHMDF LENGTH=Ø05,POS=(Ø9,56),                               *
               COLOR=NEUTRAL,HILIGHT=OFF,                              *
               ATTRB=(BRT,ASKIP),                                      *
               INITIAL='     '
          DFHMDF LENGTH=Ø01,POS=(Ø9,62),                               *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
          DFHMDF LENGTH=Ø1Ø,POS=(Ø9,63),                               *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP),                                          *
               INITIAL='Key length'
SKEYLEN   DFHMDF LENGTH=Ø04,POS=(Ø9,75),                               *
               COLOR=NEUTRAL,HILIGHT=OFF,                              *
               ATTRB=(BRT,ASKIP),                                      *
               INITIAL='    '
          DFHMDF LENGTH=Ø01,POS=(Ø9,8Ø),                               *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SCHEAD1   DFHMDF LENGTH=Ø78,POS=(1Ø,Ø1),                               *
               COLOR=NEUTRAL,HILIGHT=REVERSE,                          *
               ATTRB=(ASKIP),                                          *
               INITIAL='C/X    1...5...1Ø...15...2Ø...25...3Ø          *
                                        '
          DFHMDF LENGTH=Ø01,POS=(1Ø,8Ø),                               *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SOPT1     DFHMDF LENGTH=Ø01,POS=(11,Ø2),                               *
               COLOR=GREEN,HILIGHT=UNDERLINE,                          *
               ATTRB=(**-ATTRIB-4Ø-???-**),                            *
               INITIAL=' '
          DFHMDF LENGTH=Ø01,POS=(11,Ø4),                               *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SCDISP1   DFHMDF LENGTH=Ø3Ø,POS=(11,Ø8),                               *
               COLOR=GREEN,HILIGHT=REVERSE,                            *
               ATTRB=(FSET),                                           *
               INITIAL='                                '
          DFHMDF LENGTH=Ø01,POS=(11,39),                               *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SXDISP1   DFHMDF LENGTH=Ø6Ø,POS=(12,Ø8),                               *
               COLOR=GREEN,HILIGHT=REVERSE,                            *
               ATTRB=(FSET),                                           *
               INITIAL='                                               *
                            '
```

```
         DFHMDF LENGTH=ØØ1,POS=(12,69),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SXHEAD1  DFHMDF LENGTH=Ø78,POS=(13,Ø1),                                *
               COLOR=TURQUOISE,HILIGHT=REVERSE,                        *
               ATTRB=(ASKIP),                                          *
               INITIAL='    Ø1 . . .Ø5 . . . .1Ø . . . .15 . . . .2Ø   *
                . . . .25 . . . .3Ø          '
         DFHMDF LENGTH=ØØ1,POS=(13,8Ø),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SCHEAD2  DFHMDF LENGTH=Ø78,POS=(14,Ø1),                                *
               COLOR=NEUTRAL,HILIGHT=REVERSE,                          *
               ATTRB=(ASKIP),                                          *
               INITIAL='C/X   31..35...4Ø...45...5Ø...55...6Ø          *
                                          '
         DFHMDF LENGTH=ØØ1,POS=(14,8Ø),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SOPT2    DFHMDF LENGTH=ØØ1,POS=(15,Ø2),                                *
               COLOR=GREEN,HILIGHT=UNDERLINE,                          *
               ATTRB=(**-ATTRIB-4Ø-???-**),                            *
               INITIAL=' '
         DFHMDF LENGTH=ØØ1,POS=(15,Ø4),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SCDISP2  DFHMDF LENGTH=Ø3Ø,POS=(15,Ø8),                                *
               COLOR=GREEN,HILIGHT=REVERSE,                            *
               ATTRB=(FSET)
         DFHMDF LENGTH=ØØ1,POS=(15,39),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SXDISP2  DFHMDF LENGTH=Ø6Ø,POS=(16,Ø8),                                *
               COLOR=GREEN,HILIGHT=REVERSE,                            *
               ATTRB=(FSET)
         DFHMDF LENGTH=ØØ1,POS=(16,69),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SXHEAD2  DFHMDF LENGTH=Ø78,POS=(17,Ø1),                                *
               COLOR=TURQUOISE,HILIGHT=REVERSE,                        *
               ATTRB=(ASKIP),                                          *
               INITIAL='    31 . . .35 . . . .4Ø . . . .45 . . . .5Ø   *
                . . . .55 . . . .6Ø          '
         DFHMDF LENGTH=ØØ1,POS=(17,8Ø),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SCHEAD3  DFHMDF LENGTH=Ø78,POS=(18,Ø1),                                *
               COLOR=NEUTRAL,HILIGHT=REVERSE,                          *
               ATTRB=(ASKIP),                                          *
               INITIAL='C/X   61..65...7Ø...75...8Ø...85...9Ø          *
                                          '
```

```
         DFHMDF LENGTH=001,POS=(18,80),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SOPT3    DFHMDF LENGTH=001,POS=(19,02),                                *
               COLOR=GREEN,HILIGHT=UNDERLINE,                          *
               ATTRB=(**-ATTRIB-40-???-**),                            *
               INITIAL=' '
         DFHMDF LENGTH=001,POS=(19,04),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SCDISP3  DFHMDF LENGTH=030,POS=(19,08),                                *
               COLOR=GREEN,HILIGHT=REVERSE,                            *
               ATTRB=(FSET)
         DFHMDF LENGTH=001,POS=(19,39),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SXDISP3  DFHMDF LENGTH=060,POS=(20,08),                                *
               COLOR=GREEN,HILIGHT=REVERSE,                            *
               ATTRB=(FSET)
         DFHMDF LENGTH=001,POS=(20,69),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SXHEAD3  DFHMDF LENGTH=078,POS=(21,01),                                *
               COLOR=TURQUOISE,HILIGHT=REVERSE,                        *
               ATTRB=(ASKIP),                                          *
               INITIAL='    61 . . .65 . . . .70 . . . .75 . . . .80   *
                . . . .85 . . . .90          '
         DFHMDF LENGTH=001,POS=(21,80),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
         DFHMDF LENGTH=001,POS=(22,80),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SMESSAG  DFHMDF LENGTH=078,POS=(23,01),                                *
               COLOR=RED,HILIGHT=BLINK,                                *
               ATTRB=(ASKIP)
         DFHMDF LENGTH=001,POS=(23,80),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
SPFKEYS  DFHMDF LENGTH=078,POS=(24,01),                                *
               COLOR=PINK,HILIGHT=REVERSE,                             *
               ATTRB=(BRT,ASKIP),                                      *
               INITIAL='PF1=Help 4=Exit 5=Reset 6=Delete 7=Prev        *
               8=Next 9=Update 10=<< 11=>> 12=Release'                 *
         DFHMDF LENGTH=001,POS=(24,80),                                *
               COLOR=BLUE,HILIGHT=OFF,                                 *
               ATTRB=(ASKIP)
         DFHMDI TYPE=FINAL
         END
```


VSAM key sequenced dataset errors

A number of errors can occur with VSAM key sequenced datasets (KSDSs), and it is generally important to handle these quickly. This article considers how the errors can be recognized, and examines what action should be taken to retrieve the situation.

Information to help you determine which datasets are corrupted can be obtained from the current PSW for load module IDA019L1. If a dump has occurred from an S0C4 abend in the IDA019L1 load module, the offending cluster can be found by obtaining the address in register three which points to the AMB control block (which has an identifier of X'40'). The offending dataset's component name can be found at offset hexadecimal 88 in the AMB.

If the problem manifests itself as a WAIT or a LOOP but the current registers for IDA019L1 cannot be obtained, you can still locate the AMB from the RPL at offset hexadecimal 18. This points to the ACB, which, at offset hexadecimal 4, will have a pointer to the AMBL. This then has an address pointing to the AMB at offset hexadecimal 34. Once there, the dataset name is again at offset hexadecimal 88.

You can then analyse the damaged dataset. For non-spanned KSDSs, you can execute the EXAMINE INDEXTEST and EXAMINE DATATEST IDCAMS commands. You can also run LISTCAT. The JCL shown below can be used to perform this analysis.

```
//STS01A    JOB  (SDTS),'J.BRADLEY',MSGCLASS=Q,MSGLEVEL=(1,1)
//STEP1     EXEC PGM=IDCAMS
//SYSPRINT  DD   SYSOUT=*
//SYSIN     DD   *
    VERIFY DATASET(CORRUPT.CLUSTER)
    LISTC ENT(CORRUPT.CLUSTER) ALL
    EXAMINE NAME(CORRUPT.CLUSTER) ITEST NODTEST
    EXAMINE NAME(CORRUPT.CLUSTER) DTEST NOITEST
    PRINT INDATASET(CORRUPT.CLUSTER) DUMP
/*
```

If you need to recover a damaged cluster, try issuing the IDCAMS REPRO command for the data component of the cluster only and transferring it to a sequential dataset. The command and JCL to do this are shown below.

```
//STS01A    JOB   (SDTS),'J.BRADLEY',MSGCLASS=Q,MSGLEVEL=(1,1)
//STEP1     EXEC  PGM=IDCAMS
//DD1       DD    DSN=CORRUPT.CLUSTER.DATA.DUMP,DISP=SHR
//SYSPRINT  DD    SYSOUT=*
//SYSIN     DD    *
    REPRO INDATASET(CORRUPT.CLUSTER.DATA) OFILE(DD1)
/*
```

Use a sort program to sort the sequential dataset from the REPRO by key sequence, and delete any duplicates using the program's SUM FIELDS=NONE command. You can then delete and redefine the original cluster using IDCAMS commands. Once this is done, REPRO the sequential output dataset back, and the index is automatically rebuilt.

This procedure will work if the data component is not corrupt to start with, which might be the case if the INDEX is broken – the EXAMINE commands from the first set of sample code above would have reported this information. If the data component is broken, some form of forward recovery using DUMP programs and back-up media will be required. Alternatively, you could drop corrupt records and rebuild them manually. Remember that this procedure will not work for SPANNED datasets, or for datasets that are compressed in any way.

IDCAMS, as always, can report broken datasets in a number of ways (see Figures 1 and 2).

MSGID	**Message text**
IDC3302I	ACTION ERROR
IDC3308I	DUPLICATE RECORD
IDC3314I	OUT OF SEQUENCE RECORD, OUTOFSEQ, MISSING RECORDS, MISSREC, DUPLICATE RECORDS, DUPREC
IDC3351I	VSAM IO RETURN CODE = 24, 32, 156 SENSE = 0009 IMPRECISE END SENSE = 0004 FILE PROTECT SENSE = 0008 NO RECORD FOUND SENSE = 0040 TRACK FORMAT I/O ERRORS 86-OP READ AND WRITE FAILURES
IDC3350I	NO RECORD FOUND, NRF, INCORRECT LENGTH
IEC070I	RC32, RC202, RC8, RC18, RC24, RC104, RC203
IEA000I	IOS000I CMD REJ, COMMAND REJECT
ADR970E	HSM MISSING CONTROL INTERVAL WITHIN SEQUENCE SET TRACK=0 EXTENTS

You can also get RPL feedback codes of 000C000C

Figure 1: IDCAMS error reporting

If you run EXAMINE the following messages can be reported:

MSGID	Message text
IDC01714I	ERROR LOCATED AT OFFSET NNNN
IDC01720I	INDEX CONTROL INTERVAL DISPLAY AT RBA NNNN FOLLOWS
IDC11703I	DUPLICATE KEYS IN INDEX
IDC11704I	INDEX KEYS ARE NOT IN SEQUENCE
IDC11705I	INDEX RECORD CONTAINS DUPLICATE INDEX POINTERS
IDC11707I	DUPLICATE INDEX POINTERS FOUND IN SEQUENCE SET
IDC11711I	INDEX CONTROL INTERVAL COUNT ERROR
IDC11715I	INDEX HIGH-USED RBA IS NOT A MULTIPLE OF THE CONTROL INTERVAL SIZE
IDC11724I	DATA COMPONENT CONTROL AREA NOT KNOWN TO SEQUENCE SET
IDC11725I	SEQUENCE SET RBA INCONSISTENT WITH VSAM MAINTAINED RBA
IDC11727I	INDEX HIGH USED RBA GREATER THAN HIGH ALLOCATED
IDC11728I	DATA FOUND IN EMPTY CONTROL INTERVAL
IDC11733I	DATA COMPONENT KEY SEQUENCE ERROR
IDC11758I	SOFTWARE EOF FOUND IN INDEX CONTROL INTERVAL
IDC11763I	RBA OF INDEX CI GREATER THAN HIGH USED RBA
IDC11771I	INVALID RBA GENERATED
IDC11772I	HORIZONTAL POINTER CHAIN LOOP

In the event of an S0C4 in any of the following modules:
IDA019RC
IDA019RE
IDA019RG
IDA019RH
IDA019RF
IDA019RI
IDA019RJ
IDA019RN
IDA019RW
IDA019R4
IDA019SD

Loops in SVC 121 (x'79').

Looping in the following VSAM modules:
IDA019RB
IDA019RC
IDA019RE
IDA019RH
IDA019RZ
IDA019RI
IDA019RJ
IDA019RN
IDA019RW
IDA121A4
IDA019R2
IDA019R3
IDAM19R3
IDA019RA

Figure 2: IDCAMS error messages (continued)

You can also experience loops in STARTIO when HSM migrate or DB2 recovery is performed.

The DIAL-IBM service can be used to identify fixes associated with VSAM broken datasets by searching for DSBREAKER. As most modules impacted reside in the Link Pack Area, a system cold start is required to implement the fixes in question.

Datasets may be broken for reasons other than bad programming. For example, they may be shared between application regions without using the correct share option settings. If GRS is not used for cross-system dataset access, or if it is and SYSVSAM is not propagated around the GRS ring, the result is normally duplicate index pointers for datasets accessed by more than one system. If you share datasets and run an explicit verify on one system and leave the dataset open on the other system, output to the dataset can result in index damage. If a dataset is defined using MODEL, and has SPEED defined on the INDEX component, damage can also occur. This can also result in NO RECORD FOUND conditions and UNIT CHECKS.

For cross-system access issues, you may need to run SMF extraction routines to try to alleviate the problem.

Common user errors include running DFHSM utilities to back up IMS and DB2 datasets defined to use CIMODE processing. Although this can result in MSGIDC3351I errors, you need only specify CIMODE as part of the HSM command.

One message that is caused by programming error is IDC11724I, which can be issued for VSAM users who use CICS with the NO LSR WAIT function for DFSMS systems. If CICS has a high subtasking limit, a VSAM control area split can be in progress and can also be interrupted by another user request. This will give an exclusive control error for the PUT doing the split, and VSAM will back out from this condition, aborting the control area split. However, the copy of the control area split is left on the DASD volume. This is not flagged as an error, and the dataset remains accessible. If, when the dataset comes to be dumped, the product being used for the dump is DFDSS and it has VALIDATE coded, error messages are issued. You can then

run the IDCAMS EXAMINE with the INDEXTEST operand and receive minor error indications.

Another side effect of this is the premature filling of the VSAM dataset in question when a large number of phantom control areas exist. To overcome this, reworks for module IDA019RF have been implemented and these should be applied. APAR OW18171 describes the errors in detail.

John Bradley
Systems Programmer (UK)

Updating a VSAM cluster from REXX

The code reproduced below can be used to update a VSAM KSDS with data from a REXX program.

The REXX uses the following routines to do this:

- STORAGE – GETMAINS/FREEMAINS working storage used to hold the VSAM ACB.
- LSGGENT0 – Opens/closes VSAM ACB.
- LSGGENAD – Adds/updates VSAM KSDS.

HOW IT WORKS

The operation is as follows:

- First, the REXX code ALLOCs the required VSAM KSDS.
- STORAGE then obtains working storage in which to store an ACB.
- LSGGENT0 opens the ddname passed to it and stores the ACB in the working storage area. This means that only one open/close is required in this routine, greatly improving performance.
- LSGGENAD can then be used to update the file. The program handles records with an eight-byte key followed by four bytes of

data – ie a record length of twelve. When it finds a duplicate key, it replaces the record with the new value passed by REXX. If a unique key is found, a normal add is invoked. Any VSAM errors are passed back to the REXX through GPR15.

- As part of the clean-up process, LSGGENT0 is called to close the ACB, and the REXX then releases the file using ‘FREE’.

REXX

```
/* rexx
trace i */

initialize:

  signal on Error name error_handler
  signal on Failure name error_handler


  dbddname = audit
  auaddr = copies('00'x,4)
  length = 1024


allocate_file:
  action = allocfile
  "alloc f("dbddname") da('sg.test.audit.cluster') shr reuse"

get_storage_dbase:
  action = getstor
  call get_stor get auaddr length
  auaddr = addr

open_dbase:
  action = openfile
  call dbhand open auaddr dbddname

add_records:
  action = addrecord
  /* record = 'testkey00000' */
  /*           <-key->data  */

  record = 'key000010000'
  call write_record auaddr record

  record = 'key000020000'
  call write_record auaddr record
```

```
  record = 'key000030000'
  call write_record auaddr record

close_routine:

  close_dbase:
    action = closefile
    call dbhand close auaddr dbddname
  free_storage_audit:
    action = freestor
    call get_stor free auaddr length

free_file:
  action = freefile
  "free f("dbddname")"

cleanup:

  signal off Error
  signal off Failure

exit_point:
  exit Ø



error_handler:

  say 'Error in' action 'rc=' rc
  say 'Line' sigl 'follows....'
  say ' '
  say sourceline(sigl)
  say ' '

error_handler_end:
  signal cleanup

dbhand:
  parse arg action addr ddname
  address linkmvs "lsggentØ action addr ddname"
  return rc

get_stor:
  parse arg action addr length
  address linkmvs "storage action addr length"
  return rc

write_record:
  parse arg addr record
```

```
  address linkmvs "lsggenad addr record"
  return rc
```

STORAGE

```
STORAGE RMODE ANY
STORAGE AMODE 31
STORAGE CSECT
         DS    ØH
         B     BEGIN-STORAGE(,15)
         DC    C'STORAGE: '
         DC    C'&SYSDATE &SYSTIME '
         DS    ØH
BEGIN    EQU   *
         BAKR  14,Ø
         LR    12,15
         LR    1Ø,1
         USING STORAGE,12
         USING WORKAREA,11
*
START    EQU   *
         L     2,=A(WORK_AREA_LENGTH)
         STORAGE OBTAIN,LENGTH=(2),LOC=BELOW GET WORKING STORAGE
         LR    11,1
         ST    11,GETMAIN_ADDRESS
         LA    13,SAVEAREA
         MVC   SAVEAREA+4(4),=C'F1SA'
         MVC   DSS_EYE,=CL8'STORAGE'
*
SORT_PARMS EQU *
         LM    1,3,Ø(1Ø)
         STM   1,3,USER_PARMS
         L     2,ADDR_POINTER
         MVC   ADDR_ADDRESS,2(2)
         XC    WORK_PGN_1,WORK_PGN_1
         XC    WORK_PGN_2,WORK_PGN_2
         XC    PACK_D,PACK_D
         L     3,LENGTH_POINTER
         LH    9,Ø(3)
         BCTR  9,Ø
         LA    4,WORK_PGN_1
         EX    9,MOVE
         MVZ   WORK_PGN_1,=F'Ø'
         PACK  PACK_D,WORK_PGN_1(5)
         MVC   WORK_PGN_2+2(4),PACK_2+1
         L     3,WORK_PGN_2
         B     CHECK_INSTRUCTION
*
```

```
MOVE     MVC   Ø(Ø,4),2(3)
*
CHECK_INSTRUCTION EQU *
         L     1,INSTR_POINTER
         CLC   =C'F',2(1)
         BE    FREE_STORAGE
*
GET_STORAGE EQU *
         STORAGE OBTAIN,LENGTH=(3)
         ST    1,2(2)
         LR    5,15
         B     ENDIT
*
FREE_STORAGE EQU *
         L     2,ADDR_ADDRESS
         STORAGE RELEASE,LENGTH=(3),ADDR=(2)
         LR    5,15
         B     ENDIT
*
ENDIT    EQU   *
         L     2,=A(WORK_AREA_LENGTH)
         L     3,GETMAIN_ADDRESS
         STORAGE RELEASE,LENGTH=(2),ADDR=(3)
         LR    15,5
         PR    ,
*
         LTORG ,
*
WORKAREA DSECT
SAVEAREA DS    18F
GETMAIN_ADDRESS DS F
DSS_EYE  DS    CL8
USER_PARMS DS ØF
INSTR_POINTER DS F
ADDR_POINTER DS F
LENGTH_POINTER DS F
ADDR_ADDRESS DS F
*
PACK_D   DS    ØD
PACK_1   DS    F
PACK_2   DS    F
WORKS    DS    CL4,C
WORK_VAR DS    CL8,C
WORK_PGN_1 DS F,C
WORK_PGN_2 DS F,C
WORK_AREA_LENGTH EQU *-WORKAREA
         END
```

LSGGENTO

```
LSGGENTØ AMODE 31
LSGGENTØ RMODE ANY
LSGGENTØ CSECT
         DS    ØH
         B     BEGIN-LSGGENTØ(,15)
         DC    C'LSGGENTØ: '
         DC    C'&SYSDATE &SYSTIME '
         DS    ØH
BEGIN    EQU   *
         BAKR  14,Ø
         LR    12,15
         LR    1Ø,1
         SR    5,5
         USING LSGGENTØ,12
         USING WORKAREA,11
         USING USER_WORKAREA,3
*
START    EQU   *
         L     2,=A(WORK_AREA_LENGTH)
         STORAGE OBTAIN,LENGTH=(2)
         LR    11,1
         ST    11,GETMAIN_ADDRESS
         LA    13,SAVEAREA
         MVC   SAVEAREA+4(4),=C'F1SA'
         MVC   DSS_EYE,=CL8'LSGGENTØ'
*
GET_PARMS EQU  *
         LM    2,4,Ø(1Ø)
         LA    3,2(,3)
         ICM   3,15,Ø(3)
         LA    8,DYNAMIC_ACB
*
CHECK_ACTION EQU *
         CLC   =C'OPEN',2(2)
         BE    INIT_AND_OPEN
         CLC   =C'CLOSE',2(2)
         BE    CLOSE_AND_CLEANUP
         B     ABENDØC1
*
INIT_AND_OPEN EQU *
         MVC   DYNAMIC_ACB,STATIC_ACB
         LH    9,Ø(,4)
         BCTR  9,Ø
         EX    9,MOVE_DDNAME
         OPEN  ((8),OUTPUT),MODE=31
         B     ENDIT
*
```

```
CLOSE_AND_CLEANUP EQU *
         CLOSE ((8)),MODE=31
*
ENDIT    EQU   *
         L     2,=A(WORK_AREA_LENGTH)
         L     3,GETMAIN_ADDRESS
         STORAGE RELEASE,LENGTH=(2),ADDR=(3)
         LR    15,5
         PR    ,
*
*
MOVE_DDNAME MVC 4Ø(Ø,8),2(4)
*
ABENDØC1 DC    F'Ø'
*
*
STATIC_ACB ACB AM=VSAM,DDNAME=DUMMY,MACRF=(KEY,SEQ,OUT),RMODE31=ALL
STATIC_ACB_LENGTH EQU *-STATIC_ACB
*
*
         LTORG ,
*
WORKAREA DSECT
SAVEAREA DS    18F
GETMAIN_ADDRESS DS F
DSS_EYE  DS    CL8
*
*
WORK_AREA_LENGTH EQU *-WORKAREA
*
USER_WORKAREA DSECT
DYNAMIC_ACB DS CL(STATIC_ACB_LENGTH)
*
         END
```

LSGGENAD

```
LSGGENAD AMODE 31
LSGGENAD RMODE ANY
LSGGENAD CSECT
         DS    ØH
         B     BEGIN-LSGGENAD(,15)
         DC    C'LSGGENAD: '
         DC    C'&SYSDATE &SYSTIME '
         DS    ØH
BEGIN    EQU   *
         BAKR  14,Ø
         LR    12,15
```

```
         LR    10,1
         SR    5,5
         USING LSGGENAD,12
         USING WORKAREA,11
*
START    EQU   *
         L     2,=A(WORK_AREA_LENGTH)
         STORAGE OBTAIN,LENGTH=(2)
         LR    11,1
         ST    11,GETMAIN_ADDRESS
         LA    13,SAVEAREA
         MVC   SAVEAREA+4(4),=C'F1SA'
         MVC   DSS_EYE,=CL8'LSGGENAD'
*
GET_PARMS EQU  *
         LM    1,2,0(10)
         LH    8,0(,2)
         LA    2,2(,2)
         MVC   KEYAREA,0(2)
         L     6,2(,1)
*
BUILD_RPL EQU  *
         LA    4,MESSAGE_AREA
         LA    10,WORK
         L     9,=A(L'KEYAREA)
         GENCB BLK=RPL,AM=VSAM,LOC=ANY,                                X
               ACB=(6),OPTCD=(KEY,DIR,SYN,UPD,KEQ,FKS,MVE),            X
               AREA=(10),AREALEN=VSAM_REC_LENGTH,                      X
               ARG=(2),KEYLEN=(9),                                     X
               MSGAREA=(4),MSGLEN=128
         LR    3,1
*
EDIT_FILE EQU  *
         GET   RPL=(3)
         CHECK RPL=(3)
         LTR   15,15
         BZ    UPDATE_ENTRY
*
ADD_ENTRY EQU  *
         MODCB RPL=(3),RECLEN=(8),AREA=(2),OPTCD=(KEY,SEQ,SYN,NUP,MVE)
         MODCB ACB=(6),MACRF=(KEY,SEQ,OUT)
         PUT   RPL=(3)
         CHECK RPL=(3)
         LTR   15,15
         BNZ   ERROR_ROUTINE
         ENDREQ RPL=(3)
         B     END_OF_INPUT
*
```

```
UPDATE_ENTRY EQU *
         MODCB RPL=(3),RECLEN=(8),AREA=(2)
         PUT   RPL=(3)
         CHECK RPL=(3)
         LTR   15,15
         BNZ   ERROR_ROUTINE
         ENDREQ RPL=(3)
         B     END_OF_INPUT
*
ERROR_ROUTINE EQU *
         LA    4,FEED_BACK_AREA
         SHOWCB RPL=(3),FIELDS=FDBK,LENGTH=4,AREA=(4)
         L     5,FEED_BACK_AREA
*
END_OF_INPUT EQU *
*
ENDIT    EQU   *
         L     2,=A(WORK_AREA_LENGTH)
         L     3,GETMAIN_ADDRESS
         STORAGE RELEASE,LENGTH=(2),ADDR=(3)
         LR    15,5
         PR    ,
*
*
         LTORG ,
*
VSAM_REC_LENGTH EQU L'KEYAREA+1Ø
*
WORKAREA DSECT
SAVEAREA DS    18F
GETMAIN_ADDRESS DS F
DSS_EYE  DS    CL8
SAVE_REGS23 DS 2F
*
ACB_ADDR DS    F
KEYAREA  DS    CL8
WORK     DS    CL(VSAM_REC_LENGTH)
VSAM_BUFFER_ADDRESS DS F
FEED_BACK_AREA DS F
ACB_DATA_ADDRESS DS F
RECORD_LENGTH DS F
MESSAGE_AREA DS CL128
*
WORK_AREA_LENGTH EQU *-WORKAREA
*
         END
```

Calum Reid
Systems Technician (UK)

ISPF and VSAM

While ISPF revolutionized on-line programming, and VSAM revolutionized file access methods, the two have traditionally not worked together very well. Over the years, however, newer versions of ISPF have made it easier to work with VSAM ... if you know how.

VSAM UTILITIES

ISPF Option 3.2, typically used to create a new sequential or partitioned dataset (PDS), has a new menu item: V for VSAM Utilities. Selecting it initiates a window where you can choose to create (Define), delete, or obtain information on a VSAM dataset. In the same window, you can also choose what type of VSAM component you want to work with; this choice is mandatory only for Define.

Hitting Enter displays a panel where additional information can be entered. Assuming the "Edit IDCAMS command" field has a slash ('/') in it, the generated IDCAMS command is displayed for any necessary modification. When it's correct, EXEC can be typed on the Command line to actually perform the command.

OPTION 3.4

The dataset listing (DSLIST) displayed by ISPF Option 3.4 provides a generous set of line commands, but it also allows CLISTs, REXX EXECs, and TSO commands to be executed against any of the datasets listed. TSO also recognizes IDCAMS commands as TSO commands, although purists might note that the allowable abbreviations are slightly different. The result is significant VSAM functionality from DSLIST, although it does take a little typing.

Until recently, the I line command beside a VSAM component gave an error message, as it still does for tape and GDG base datasets. The I line command will also try to recall a migrated dataset. An effective alternative is typing LISTCAT ENTRIES(/) ALL where you would normally type the I line command. The slash requests that the dataset

name, with single quotes around it, be substituted into the command at that point. If a slash is not specified, the dataset name is appended to the end of the command after a trailing blank.

For non-touch-typists, the shortest available abbreviation is LISTC ENT(/) ALL. However, some organizations have added LC as a line command that translates into LISTCAT ENTRIES(/) ALL. Unfortunately, unlike the VSAM Utilities available from Option 3.2, the LISTCAT output is not scrollable.

D LINE COMMAND

Major improvements have also been made to the D line command. It no longer forces the recall of a migrated dataset, but intelligently chooses to do an HDELETE. Likewise, for a VSAM cluster, a D line command initiates the same panel as would a Delete from the Option 3.2 VSAM Utilities. Typing DEL, the previous solution, still initiates the DELETE TSO command, an implementation of the IDCAMS DELETE command.

Basically, all that can be done with IDCAMS in batch can be done from Option 3.4's dataset listing. Although PRINT also comes in handy sometimes, REPRO is probably the most useful, especially since Option 3.3 does not (yet?) support copying and moving VSAM datasets.

BROWSE AND VIEW

There are non-IBM products that will allow Browse and Edit of VSAM datasets from within ISPF. But if you do not have access to one of these products, or do not wish to learn how to use one, REPRO presents some alternatives.

Create a PDS called VSAMCOPY.DATA under your TSO ID, and you are set for life. The record format should be VB. Pick an optimal block size for the type of DASD being used – for 3390 and most RAMAC, 27998 is the best choice. The record size should be four bytes less (eg 27994 for 3390).

To browse a VSAM dataset, type

```
REPRO IDS(/) ODS(VSAMCOPY.DATA(name))
```

as a line command beside the cluster name of the VSAM dataset, where ‘name’ is an appropriate name to help you remember the VSAM dataset from which it came. Then type B as a line command beside VSAMCOPY.DATA, hit Enter, and put an S beside the ‘name’ you selected on the member list that is displayed. Alternatively, V will allow you to see the dataset in View format.

EDIT

Assuming the dataset is small enough, you can also use E to edit the dataset. Once the changes are complete, it must be REPROed back to actually make the changes in the VSAM dataset:

```
REPRO IDS(VSAMCOPY.DATA(name)) ODS(/) REUSE
```

This requires that REUSE be set for the dataset. If not, use:

```
ALTER / REUSE
```

If an Alternate Index has been defined for the VSAM dataset, REUSE cannot be set. That leaves you with three alternatives:

- Delete all the records in the VSAM dataset and then REPRO without the REUSE parameter. Unfortunately, IDCAMS does not have any way (that I know of) to delete records.
- Delete the Alternate Index(es), ALTER the Base Cluster to REUSE, REPRO with REUSE, then redefine and rebuild the alternate index(es) and redefine the path(s).
- Delete and redefine the Base Cluster, REPRO without REUSE, then redefine and rebuild the alternate index(es) and redefine the path(s).

EDITING LARGE DATASETS

Datasets too large to edit with the ISPF Editor require significantly more effort. Unlike Browse, which can handle datasets as large as 100M records, the maximum dataset size for Edit depends on the

amount of memory available in the user's TSO region. If you suspect that the dataset is just a little too big, be sure that the Size parameter displayed in TSO full screen logon has been set as large as permitted by the MAXSIZE parameter set for your User ID by the TSO Administrator.

Although there are a number of ways to accomplish the task, the basic approach is to extract the records that need to be modified, change them with the ISPF Editor, and put them back in the VSAM dataset. This approach may not be practical if there are changes to all or many of the records of a large dataset, although it is still theoretically possible to carve up a large dataset into many smaller datasets and edit each individually. It is more likely, however, that massive edits of large datasets are better done with other tools. For example, a sort package can be used to overwrite columns 9-10 of every record with the characters AB. For 80-byte fixed length records, this would be:

```
INREC FIELDS=(1,8,C'AB',11,70)
SORT FIELDS=COPY
```

Alternatively, select all records from a key sequenced VSAM dataset with the characters QW in columns 9-10 and change them to AB:

```
INCLUDE COND=(9,2,EQ,C'QW'),FORMAT=CH
SORT FIELDS=COPY
OUTREC FIELDS=(1,8,C'AB',11,70)
```

then REPRO with REPLACE. This assumes that column 9-10 is not part of the primary key.

Recent versions of sort packages can even find a character string anywhere in a field or even the entire record:

```
INCLUDE COND=(1,80,EQ,C'QW'),FORMAT=CH
```

If you have a smaller number of records to change, there is a general approach to changing one or more consecutive records, even if they reside in non-VSAM datasets that are too large for the ISPF Editor to handle:

1 Extract the records that need to be changed. If the VSAM dataset has already been copied to VSAMCOPY.DATA (or if you are working with a large non-VSAM dataset), the COPY command

in the ISPF Editor can be used once you know the record numbers of the first and last record you want. Both ISPF Browse and View indicate the record number of the first line on the screen. Hitting Enter is sometimes required if the record number is overlaid with another message, as would be the case after a FIND command. If the record in question is not at the top of the screen, hit PF8 with the cursor on the record whose number you want to know and CSR in the Scroll field.

Once you know the record number of the first and last records that you want to include, create a new member in VSAMCOPY.DATA with the ISPF Editor, then type COPY on the command line. Enter the dataset name on the panel, then fill in the three Line Numbers fields. The First Line and Last Line are the first and last record numbers (they can be the same if you only want one record), and R is the Number Type to indicate record numbers.

Alternatively, if you want to create the records for editing directly from the VSAM dataset, REPRO has quite a few useful parameters, including FROMNUMBER and TONUMBER for Relative Record (RRDS), and FROMKEY and TOKEY for Key Sequenced (KSDS). This allows selection by the Primary Key or any Alternate Keys for which a Path has been defined for an Alternate Index. Generic (partial) keys can also be used. If the FROMKEY does not exist, the next highest key value is selected.

FROMADDRESS and TOADDRESS select records by Relative Byte Address (RBA). The RBAs must be accurate since the REPRO will fail if the addresses do not correspond to the beginning of records. For KSDS, the records are retrieved in physical, not logical, order.

SKIP and COUNT work sequentially on any supported type of VSAM or non-VSAM dataset. To select records 500, 501, and 502, specify:

```
SKIP(499) COUNT(3).
```

Use DISP=MOD or create multiple members of VSAMCOPY.DATA if you need to change several sets of records in a large dataset.

2 Make the changes with the ISPF Editor.

3 Put the changed records back in the dataset. How this can be done depends on how the dataset is organized. RRDS and KSDS can be done with REPRO and the REPLACE parameter. The only exception would be where the change was made to the primary key field of a KSDS dataset. In this case, and for all other dataset types, the easiest method is to rebuild the dataset.

Conceptually, the idea is to create a new copy of the dataset by sequentially copying:

- All the records before the records that were changed.
- The records that were changed.
- All the records after those that were changed.

REPRO can be used, but it loses its ease-of-use edge over other methods because OUTDATA SET opens datasets with DISP=OLD. Rather than using OUTFILE and an ALLOCATE or DD statement, it is almost as easy – and certainly quicker – to use a sort package.

```
//        EXEC
//SORTIN  DD DSN=dataset being modified
//SORTOUT DD ...,DISP=(NEW,CATLG)
//SYSIN   DD *
  OPTION STOPAFT=499
  SORT FIELDS=COPY
//        EXEC
//SORTIN  DD DSN=the modifications
//SORTOUT DD ...,DISP=MOD
//SYSIN   DD *
  SORT FIELDS=COPY
//        EXEC
//SORTIN  DD DSN=dataset being modified
//SORTOUT DD ...,DISP=MOD
//SYSIN   DD *
  OPTION SKIPREC=502
  SORT FIELDS=COPY
```

AUTOMATING THE PROCESS

At least for Browse or View, the process can be automated by writing a CLIST or REXX EXEC. Although certainly not as bullet-proof, the

following REXX code provides an alternative to the B line command in DSLIST, extending its functionality to include browsing of VSAM datasets:

```
/*     REXX       */
arg dsn
"FREE DDNAME(BRDD)"
listdsirc = LISTDSI(dsn)
if listdsirc <= 4
then do
   if SYSDSORG = "VS"
   then do
      "ALLOC DDNAME(BRDD) NEW DELETE SPACE(1,100) CYLINDERS",
         "BLKSIZE(27998) LRECL(27994) RECFM(V, B) DSORG(PS)"
      "REPRO IDS("dsn") OUTFILE(BRDD)"
      end
   else
      "ALLOC DDNAME(BRDD) DSNAME("dsn") SHR"
   end
"ISPEXEC LMINIT DATAID(brdid) DDNAME(BRDD)"
"ISPEXEC BROWSE DATAID("brdid")"
"FREE DDNAME(BRDD)"
exit
```

The REXX EXEC eliminates the need for the VSAMCOPY.DATA PDS by creating a temporary dataset with a system-supplied name. The LMINIT service must be added because BROWSE does not support DDnames.

The FREE at the beginning prevents problems if the EXEC was previously interrupted. A return code of zero or four for LISTDSI is acceptable since a successful call with a VSAM dataset name gives four, indicating that not all the kinds of information that are associated with non-VSAM datasets are available.

The BRIF service could be used in place of BROWSE. It could significantly improve initial performance for large VSAM datasets, avoiding the REPRO step. Unfortunately, BRIF requires that the program – in this case, a REXX EXEC – read the dataset, and REXX's only I/O mechanism, EXECIO, does not support VSAM. Neither does CLIST, so that leaves writing a program in one of the many traditional programming languages that supports calls to ISPF services.

USING IDCAMS COMMANDS IN ISPF

Probably the biggest problem associated with using IDCAMS commands in ISPF Option 3.4 is remembering the keyword parameter used to specify the dataset, and its shortest possible abbreviation. Parenthesis can also be problematic, too. Here is a quick summary of the most common commands, including some of the non-obvious uses of the slash, such as DEFINE CLUSTER, which could be used right after a delete to redefine the same cluster name.

```
ALLOC DA(/)
ALTER / or NEWNM(/)
BIX IDS(/) or ODS(/)
DEF ALIAS (NAME(/) or REL(/))
DEF AIX (NAME(/) or REL(/))
DEF CL (NAME(/))
DEF GDG (NAME(/))
DEF NVSAM (NAME(/))
DEF PATH (NAME(/))
DEL (/)
EXAMINE NAME(/)
EXP / or ODS(/)
IMP IDS(/) ODS(/)
LISTC ENT(/) or LVL(/)
PRINT IDS(/)
REPRO IDS(/) or ODS(/) or ENT(/) or LVL(/)
VFY DS(/)
```

An ‘or’ indicates multiple parameters where the slash could be used.

NOT DISCUSSED

The ISPF Workplace, Option 11 on the Main Menu of ISPF Version 4, has been conspiciously absent from this discussion. This is because the installation of OS/390, which included ISPF Version 4.2, has been a relatively recent event and not enough hands-on experience has been gained to determine the best way to access VSAM from the Workplace.

Jon E Pearkins
(Canada)

Space information for VSAM datasets

The REXX EXEC presented here gives space information for VSAM datasets. It can be executed with the dataset name supplied as a parameter, but is best used as a line command in the TSO Dataset List Utility.

The following information is given:

- The Allocation and Used Space in tracks and cylinders.
- The total Free Space available on the dataset (including Free Space in CIs, etc).

This alleviates the need to calculate Free Space and Used Space manually from the RBA information.

Note that the allocation figures are rounded UP to the nearest track, and that cylinder and free space are rounded DOWN.

VSAMSTAT REXX SOURCE

```
/* REXX EXEC to Read LISTCAT output and display space allocated
   in tracks and cylinders for VSAM datasets

   Author: Martin Symmers

   To execute: a) TSO EXEC VSAMSTAT datasetname
               b) As line command 'VSAMSTAT' from option 3.4

*/
arg dsn                                      /* get dataset name.... */
dsn=strip(dsn,b,"'")       /* ...and strip off surrounding quotes' */
dsn=strip(dsn,b,'"')

MAINLINE:
call INIT
call GET_DATA
call READ_CATALOG_LISTING
call CLEAN_UP_INPUT_DATA
call DO_THE_SUMS
call OUTPUT
EXIT
```

```
INIT:
freespace=''                                    /* initialize variables */
hi_alloc_rba=''
hi_used_rba=''
phyrec_size=''
phyrecs_trk=''
tracks_ca=''
RETURN

GET_DATA:
x=outtrap(catalog_listing.)                      /* get LISTCAT output */
address tso "listc ent('"dsn"') all"
x=outtrap("off")
RETURN

READ_CATALOG_LISTING:                                 /* Parse it... */
do loop = 1 to catalog_listing.Ø
  nonvsam = wordpos('NONVSAM',catalog_listing.loop)
  if nonvsam <> Ø then do
     say 'Error - Non-VSAM dataset'            /* skip non-vsam files */
     EXIT
  end
  gdgbase = wordpos('GDG BASE',catalog_listing.loop)
  if gdgbase <> Ø then do
     say 'Error - Non-VSAM dataset (GDG)'           /* skip GDG files */
     EXIT
  end

  parse var catalog_listing.loop . 'FREESPC' fspc .     /  * get what */
  if fspc <> '' & freespace = '' then freespace = fspc    /* we need  */
  parse var catalog_listing.loop . 'HI-U-RBA' hur .       /* to work  */
  if hur <> '' & hi_used_rba = '' then hi_used_rba = hur  /* out      */
  parse var catalog_listing.loop . 'PHYREC-SIZE' ps .     /* allocs & */
  if ps <> '' & phyrec_size = '' then phyrec_size = ps    /* freespace*/
  parse var catalog_listing.loop . 'PHYRECS/TRK' pt .
  if pt <> '' & phyrecs_trk = '' then phyrecs_trk = pt
  parse var catalog_listing.loop . 'TRACKS/CA' tc .
  if tc <> '' & tracks_ca = '' then tracks_ca = tc
  parse var catalog_listing.loop . 'HI-A-RBA' har .
  if har  > hi_alloc_rba then hi_alloc_rba = har
end
if tracks_ca = '' then do                  /* not found what we need? */
   say 'Error - VSAM Stats unobtainable'   /* then bail out....       */
   EXIT
end
RETURN

CLEAN_UP_INPUT_DATA:
freespace = strip(freespace,l,'-')         /* tidy up data we have    */
```

```
hi_alloc_rba = strip(hi_alloc_rba,l,'-')   /* obtained                */
hi_used_rba = strip(hi_used_rba,l,'-')
phyrec_size = strip(phyrec_size,l,'-')
phyrecs_trk = strip(phyrecs_trk,l,'-')
tracks_ca = strip(tracks_ca,l,'-')
RETURN

DO_THE_SUMS:
fspc_trks = (freespace % phyrec_size) % phyrecs_trk       /* work out */
fspc_cyls = fspc_trks % tracks_ca                         /* allocs & */
                                                          /* freespace*/
alloc_trks = (hi_alloc_rba / phyrec_size) / phyrecs_trk
parse var alloc_trks before_point '.' after_point
if after_point <> '' then do
   alloc_trks = before_point + 1          /* round up for allocations */
end
alloc_cyls = alloc_trks / tracks_ca
parse var alloc_cyls before_point '.' after_point
if after_point <> '' then do
   alloc_cyls = before_point + 1
end
used_trks = ((hi_used_rba / phyrec_size) / phyrecs_trk)
parse var used_trks before_point '.' after_point
if after_point <> '' then do
   used_trks = before_point + 1
end
used_cyls = used_trks / tracks_ca
parse var used_cyls before_point '.' after_point
if after_point <> '' then do
   used_cyls = before_point + 1
end
RETURN

OUTPUT:                                          /* output results */
say dsn': '
say 'Allocated Space  : 'alloc_trks' tracks  'alloc_cyls' Cylinders'
say 'Used Space       : 'used_trks' tracks  'used_cyls' Cylinders  '
say 'Total Space Free : 'fspc_trks' tracks  'fspc_cyls' Cylinders  '
RETURN
```

USING VSAMSTAT

The following is an example of how to use VSAMSTAT as a line command from TSO option 3.4 (Dataset List Utility), and of the output returned.

```
DSLIST - Data Sets Matching MY.VSAM.DATASET                    Row 1 of 2
Command ===>                                                 Scroll ===> PAGE

Command - Enter "/" to select action            Message           Volume
-------------------------------------------------------------------------
vsamstat MY.VSAM.DATASET                                           *VSAM*
         MY.VSAM.DATASET.DATA                                     VOL01Ø+
************************* End of Data Set list ************************

 MY.VSAM.DATASET:
 Allocated Space  : 32700 tracks  2180 Cylinders
 Used Space       : 32488 tracks  2166 Cylinders
 Total Space Free :   212 tracks    14 Cylinders
 ***
```

USAGE NOTES

- In order to use it as a line command, the EXEC will have to be concatenated on the SYSEXEC or SYSCLIST DD in your logon procedure.

  If this is not the case, you can still use this EXEC, but you will have to execute it explicitly. To do so, enter the following from the TSO command line or the ISPF command shell:

  ```
  TSO EXEC 'my.rexx.library(VSAMSTAT)'    'my.vsam.dataset'
  ```

  where ‘my.rexx.library’ is the PDS that contains the VSAMSTAT EXEC and ‘my.vsam.dataset’ is the target VSAM dataset. The latter can be either the index or data portion or the cluster name.

- VSAMSTAT fails for migrated datasets and does not force an HSM recall.

- This EXEC has been designed to run under DFSMS 1.3.

Martin Symmers
DBA (UK)

File compression with DFSMS/MVS

DFSMS/MVS Version 1.2.0 on MVS/ESA 5.2.0 offers the ability to compress the data component of Extended Format KSDSs, using the hardware compression facility present in IBM's 711-, 511-, and 211-based ES/9000 processors.

The program presented here is designed to show which files can be compressed.

The program consists of the following:

- FILECOMP, which reads an SAS database created daily by the SASJFIC JOB, which uses IDCAMS DCOLLECT to collect information on files. FILECOMP produces a report on each file eligible for compression, showing which should be saved, deleted, defined, or restored to be compressed.
- A SYSIN file RJVOL, which gives the Free Space by volume (whether under SMS or not).
- A SYSIN file HIER, which gives yesterday's date.

We have defined two DATACLASses for compressed files in DFSMS/MVS:

- VSMCOMPD for VSAM files.
- SEQCOMPD for PS files (sequential).

For these DATACLASses, the keyword COMPACTION is YES, and the keyword dataset name type is EXTENDED. These DATACLASses are used when we define the file with the IDCAMS DEFINE utility.

FILECOMP

```
//FILECOMP JOB EXP,'FILECOMP',CLASS=W,MSGCLASS=Ø,MSGLEVEL=(1,1),
//        NOTIFY=DUNAND,USER=SYSOP8,PASSWORD=MANXX
//DELCOMP EXEC PGM=IDCAMS,REGION=2Ø48K
//SYSPRINT DD SYSOUT=Ø
//SYSIN  DD *
   DELETE 'EXPL69.FIC.A.COMPRESS'
```

```
/*
//FILECOMP EXEC SAS,REGION=8M,
//       WORK='200,50',
//         OPTIONS='MEMSIZE=16M DMSBATCH BATCH TERMINAL'
//SOURCLIB DD DSN=SAS.BERCY.REPORTS,DISP=SHR
//LIBRARY  DD DSN=SAS.MXG.FORMATS,DISP=SHR
//SASLIST DD DSN=EXPL69.FIC.A.COMPRESS,DISP=(NEW,CATLG,DELETE),
//        UNIT=SYSDA,SPACE=(TRK,(5,9),RLSE),
//        DCB=(RECFM=F,LRECL=133,BLKSIZE=0),MGMTCLAS=DEL32
//SOURCLIB DD DSN=SAS.BERCY.REPORTS,DISP=SHR
//LIBRARY  DD DSN=SAS.MXG.FORMATS,DISP=SHR
//SASLIST DD SYSOUT=*
//SYSIN   DD *

  OPTIONS PAGESIZE=60 LINESIZE=132 ;

  %LET RETCODE=.;

LIBNAME MONTH 'SAS.BERCY.FICPDB.MONTH' DISP=SHR;
LIBNAME DETAIL'SAS.BERCY.FICPDB.DETAIL' DISP=SHR;
   %INCLUDE SOURCLIB(HIER);
TITLE "FICHIERS QUI DEVRAIENT ETRE COMPRES DATE      &HIER";
FOOTNOTE "FICHIERS DE LYON  A COMPRESSER      ";
OPTIONS LINESIZE=133  PAGESIZE=68;
OPTIONS NOCENTER;
PROC SORT DATA=DETAIL.DCOLDSE OUT=COMPR NODUPKEY;
               BY DCDSNAM;
DATA COMPR;
SET  COMPR;
KEEP DCDSORG DCVOLSR DCMGTCL DCALLSP DCUSESP LSTPDATE DCSTGRP DCDATCL
                     DCDSNAM;
PROC PRINT DATA=COMPR (WHERE=((DCDATCL ¬='SEQCOMPD'
                    AND DCALLSP > 05000000)
                                      OR
         (     DCDATCL ¬= 'VSMCOMPD'   AND DCALLSP > 05000000) ));
  ID  DCDSNAM;
VAR DCDSORG DCVOLSR DCMGTCL DCALLSP DCUSESP LSTPDATE DCSTGRP ;
  LABEL DCDATCL = 'DATACLASS'
  LABEL DCDSORG = 'ORGANISATION'
  LABEL DCVOLSR = 'VOLSER'
  LABEL DCMGTCL = 'MGMTCLAS'
  LABEL DCALLSP = 'DCALLSPCATED'
  LABEL DCUSESP = 'USED'
  LABEL LSTPDATE = 'LAST RED DATE'
  LABEL DCSTGRP = 'STORGROUP';
RUN;
```

SASJFIC

```
//SASJFIC  JOB COM,'SASFIC',CLASS=W,MSGCLASS=Ø,
//       COND=(4,LT)
//DELETE   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 DELETE SAS.DCOLLECT
 IF MAXCC <= 8 THEN SET MAXCC=Ø
/*
//*
//* DCOLLECT
//*
//DCOLLECT    EXEC PGM=IDCAMS
//SYSPRINT    DD SYSOUT=*
//MCDS        DD DSN=HSM.MCDS,DISP=SHR
//BCDS        DD DSN=HSM.BCDS,DISP=SHR
//OUTDS       DD DSN=SAS.DCOLLECT,DISP=(,CATLG,DELETE),
//*           UNIT=SYSDA,DCB=(RECFM=VB,LRECL=264),
//            UNIT=338Ø,DCB=(RECFM=VB,LRECL=644,BLKSIZE=Ø),
//            SPACE=(TRK,(1Ø5,15),RLSE)
//SYSIN       DD *
    DCOLLECT OFILE(OUTDS)    -
             VOLUMES(*)
/*
//DCOL   EXEC SAS,REGION=8M,
//       WORK='15Ø,2Ø',
//         OPTIONS='MEMSIZE=16M DMSBATCH BATCH TERMINAL'
//DCOLLECT DD DSN=SAS.DCOLLECT,DISP=SHR
//REPORT   DD DSN=SAS.BERCY.REPORTS,DISP=SHR
//* TLMS     DD DSN=EXPL69.TLMS.VMF,DISP=SHR
//SASLIST  DD SYSOUT=Ø
//SYSIN    DD *

  OPTIONS PAGESIZE=6Ø LINESIZE=132 ;

  %CPSTART(MODE=BATCH,
           SYSTEM=MVS,
            ROOT=SAS.SAS6Ø9.TS45Ø.CPE.,
            PDB=SAS.BERCY.FICPDB.,
            DISP=OLD,
            ROOTSERV=,
            SHARE=N/A,
            MXGSRC=('SAS.BERCY.SOURCLIB' 'SAS.MXG.V1313.SOURCLIB'),),
            MXGLIB=SAS.MXG.V1313.FORMATS
            ) ;


  %INCLUDE SOURCLIB(TYPEDCOL);
  RUN;
```

```
  %CMPROCES(,
            COLLECTR=GENERIC,
            TOOLNM=SASDS,
            UNIT=DISK,
            GENLIB=WORK
           );

  %CPREDUCE();

                  /****  DAILY REPORTS  *****/

   %INCLUDE REPORT(OPTIONS);
   %INCLUDE REPORT(HIER);
   %INCLUDE REPORT(RJVOL);
/*
//* DELETE FROM FILE AFTER EXECUTION
//*
//DELETE   EXEC PGM=IDCAMS,COND=(Ø,NE,DCOL.SAS)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 DELETE SAS.DCOLLECT
/*
```

RJVOL

```
/*********************************************************************/
/* Disk space analysis                                               */
/*                                                                   */
/* 1) Creation of a way to count megabytes and gigabytes             */
/*    in 1Ø-6 and 1Ø-9 base, and of a way to separate SMS and        */
/*    non-SMS volumes                                                */
/*                                                                   */
/* 2) TABULATE procedure to present the total SMS and non-SMS        */
/*    volumes                                                        */
/*                                                                   */
/*********************************************************************/
proc format;
 picture cpbytes (min=2 default=1Ø)
          1ØØØ-9999999='ØØØØØØ1K' (MULT=.ØØ1)
      1ØØØØØØØ-9999999999='ØØØØØØ1M' (MULT=.ØØØØØ1)
   1ØØØØØØØØØØ-9999999999999='ØØØØØØ1G' (MULT=.ØØØØØØØØ1)
 ;
 value $SMS 'X'='SMS            '
            'B'='S/Total non SMS'
            'Y'='S/Total SMS    '
            'Z'='Total          '
          other='non SMS        ' ;
 value $VOL 'ZZZZZZ'='Total ';
run;
```

```
proc summary data=detail.dcolvol missing ;
     where jour = "&hier" ;
     class dcmangd dcdvtyp dcvolsr ;
     var dcalloc dcfresp dcvlcap ;
     output out=total sum=;
run;
data total ;
     set total;
     where _type_ = Ø or _type_=4 or _type_ = 6 or _type_ = 7 ;
     if dcmangd = 'Y' then dcmangd ='X' ;
     if _type_ = 4 and dcmangd='X' then dcmangd='Y';
     if _type_ = 4 and dcmangd=' ' then dcmangd='B';
     if _type_=Ø then dcmangd = 'Z';
     else if dcvolsr = ' ' then dcvolsr='ZZZZZZ' ;
     dcperct = dcfresp / dcvlcap * 1ØØ ;
run;

proc tabulate data=total missing format=f1Ø.;
     class dcvolsr dcmangd dcdvtyp ;
     var dcalloc dcfresp dcperct dcvlcap ;
     format dcmangd $SMS. dcvolsr $VOL. ;
     keylabel sum=' ';
     table dcmangd=' '
            * dcdvtyp=' '
              * (dcvolsr=' ')
           , dcalloc='MGBytes alloues'*f=cpbytes.
             dcfresp='MGBytes libres'*f=cpbytes.
             dcperct='% Espace libre'
             dcvlcap='Capacite'*f=cpbytes.
             / box="Suivi disques du &HIER"
             rts=5Ø
        ;
run;
```

HIER

```
/* Calculation of previous day's date */

data _null_ ;
     cejour = input("&sysdate", date7. );
     if "&sysday" ne 'Monday' then
     call symput ('HIER', put (cejour-1 , ddmmyy8.));
        else
     call symput ('HIER', put (cejour-3 , ddmmyy8.));
run;
```

Claude Dunand
(France)

An alternative to DEFINE CLUSTER

If you have not looked too closely at a JCL manual since MVS/XA or earlier, some of the less obvious improvements may have escaped you. Additions to the DD statement now make it possible to allocate a new VSAM dataset without resorting to a DEFINE CLUSTER using IDCAMS.

THE JES3 DILEMMA

This is especially welcome news to JES3 users, who have continually battled with the problem of defining a new VSAM dataset and using it in the same job:

```
//DEF EXEC PGM=IDCAMS
//SYSIN DD *
  DEFINE CLUSTER (NAME(A.B.C)...
//PR  EXEC PGM=PROG
//ABC   DD DSN=A.B.C,DISP=OLD
```

JES3 cancels the job with the messages:

```
IAT44Ø4 DATASET NOT FOUND ON MAIN PROCESSOR SYS1
IAT48Ø1 JOB ... EXPRESS CANCELLED BY INTERPRETER DSP
```

None of the four solutions below, suggested by various sources, solves the problem:

- AMP=AMORG on the MAST DD statement.
- //*MAIN SETUP=DHWS.
- //*MAIN SETUP=/PR.MAST.
- Two //*PROCESS statements, one with MAIN and the other with OUTSERV (because JES3 inserts its own CI).

One solution is to split the job into two and to use JES3's NET facility to ensure that the jobs are run in the correct sequence. But if there are any references between job steps in the JCL, this can be more difficult than it first appears.

An overly tricky solution, observed in a current production job, inserts another job step before the IDCAMS:

```
//DEL EXEC PGM=IEFBR14
//DEL1  DD DSN=MAST,DISP=(MOD,DELETE),SPACE=(TRK,(1,1))
```

DISP=MOD uses an existing dataset if it already exists, or creates a new one if not – in this case, a single track sequential dataset. The second parameter, DELETE, deletes the dataset in both cases.

But the best solution is to create the VSAM dataset using a DD statement instead of IDCAMS. This combines the creation and use of the VSAM dataset into a single job step and satisfies JES3.

NOT JUST FOR JES3

Obviously, the creation of VSAM datasets through DD statements deserves a careful look whenever new JCL is being written. It provides a viable alternative to IDCAMS and the DEFINE CLUSTER statement.

Admittedly, there are a significant number of DEFINE CLUSTER parameters that have no equivalent in DD parameters. Some are virtually obsolete. Others, including the once popular CONTROLINTERVALSIZE, are performance-related, and VSAM's default is normally good enough.

But there are notable absences. For example, LRECL allows only one number to be specified, used as both average and maximum record size for the internal equivalent of the DEFINE CLUSTER's RECORDSIZE parameter. The result is fixed length records, with no way on the DD statement to request variable length records.

One way to overcome this problem is to use the DD parameters DATACLAS, LIKE, or REFDD to specify a data class or existing VSAM dataset that has the required values of attributes that cannot be specified. The IDCAMS ALTER command is not usually an option, since it does not allow many of these attributes to be changed. You could, of course, use DELETE and DEFINE CLUSTER to redefine the dataset with the missing attributes, but that is really just a convoluted form of the DISP=MOD solution described above.

Figure 1 shows the DEFINE CLUSTER parameters and their DD statement equivalents. Although not exhaustive, it includes the ones most likely to be used, especially in an SMS environment:

DEFINE CLUSTER	//dd1 DD DISP=(NEW,CATLG,CATLG),
(NAME(dsn)	DSN=dsn,
CYLINDERS(primary secondary)	SPACE=(CYL,(primary,secondary)),
KILOBYTES(primary secondary)	SPACE=(1024,(primary,secondary)),
MEGABYTES(primary secondary)	SPACE=(1048576,(primary,secondary)),
RECORDS(primary secondary)	SPACE=(recsize,(primary,secondary)), AVGREC=U
TRACKS(primary secondary)	SPACE=(TRK,(primary,secondary)),
VOLUMES(vol1 vol2)	VOL=SER=(vol1,vol2),
BUFFERSPACE(bytes)	AMP=('BUFSP=bytes'),
DATACLASS(class)	DATACLAS=class,
FILE(dd2)	include the DD parms from dd2 and delete the DD statement for dd2
NUMBERED	RECORG=RR,
NONINDEXED	RECORG=ES,
LINEAR	RECORG=LS,
INDEXED	RECORG=KS,
KEYS(length offset)	KEYLEN=length,KEYOFF=offset,
MANAGEMENTCLASS(class)	MGMTCLAS=class,
MODEL(dsn)	LIKE=dsn,
RECORDSIZE(recsize recsize)	LRECL=recsize,
STORAGECLASS(class)	STORCLAS=class,
FOR(days)	RETPD=days,
TO(yyyyddd)	EXPDT=yyyy/ddd,

Figure 1: DEFINE CLUSTER parameters and their DD statement equivalents

Note that there are overrides in many installations that force Expiry Date to be 0000/000, making FOR, TO, RETPD, and EXPDT of no practical purpose. Likewise, in an SMS environment, VOL and UNIT are honoured only in exceptional circumstances.

The DD AMP parameter also includes three subparameters with no exact equivalents. BUFND= specifies the number of I/O buffers used for data records, BUFNI= for index records, and STRNO= is the maximum number of RPLs to be processed concurrently.

Although there is no way to specify the dataset names for the data and index components, they default to the cluster name with .DATA and .INDEX appended. IDCAMS now uses these same defaults, having previously generated unintelligible names completely unrelated to the cluster name supplied.

DEFINE CLUSTER is, of course, not the only IDCAMS command used to create fully functional VSAM datasets. DEFINE

ALTERNATEINDEX, BLDINDEX, and DEFINE PATH are needed to define an alternate key that can be used by programs as if it were a primary key – by specifying the path name as a DSNAME on a DD statement. JCL offers no equivalent of these three IDCAMS commands.

CONCLUSION

To build a batch job that will consistently create a VSAM dataset no matter what, even in a JES3 environment, use the following:

```
//DELETE EXEC PGM=IDCAMS
//SYSIN    DD *
  DELETE (A.B.C)
  IF LASTCC = 8 -
  THEN -
     SET MAXCC = Ø
//PR  EXEC PGM=PROG
//ABC   DD DSN=A.B.C,DISP=(MOD,CATLG,CATLG),...
```

If the VSAM dataset is a work file used only in the single job step PR, the two CATLG dispositions should be changed to DELETE. As well as keeping the dataset out of the catalogue, it also prevents potential delays when the job runs that would occur if the dataset gets migrated to tape by DFSMShsm.

The DELETE step ensures that the job will not fail if the dataset already exists, as might occur in a job rerun situation. This is even required if the dispositions are set to DELETE, because there is no guarantee that a system crash might not occur before the job step completes and the dataset is deleted.

The MAXCC/LASTCC logic is required only if non-zero job step condition codes trigger alerts to job scheduling software. Even though you are assured that the dataset will not exist when step PR begins, the MOD disposition is still required in a JES3 environment because JES3 looks for the dataset before the job begins, and it may or may not exist at that point.

HISTORICAL PERSPECTIVE

One argument against the use of DD statement parameters to replace DEFINE CLUSTER is coding complexity. When VSAM was first

introduced in the early 1970s, IDCAMS command syntax was hailed as a major improvement over IBM utilities and JCL.

The command processor for the latter two had been derived from the macro processor in the System/360 Assembler – commas between parameters, no spaces allowed around commas or equals signs, apostrophes as quotation marks around anything unusual, and eight-character naming restrictions. IDCAMS offered longer names, parentheses around values, and spaces as delimiters.

But, 25 years later, it is almost as easy to forget where to put hyphens as continuation marks in IDCAMS as it is to make a JCL error. Besides, the simplification of the batch job by combining the VSAM dataset creation and use into a single job step more than makes up for the differences in syntax.

Jon E Pearkins
(Canada)

VSAM Update contributions

Tell us what you have done to make working with VSAM easier or quicker at your site. We welcome contributions from VSAM novices as well as from more experienced users, and are keen to receive anything from very short 'hints and tips' type articles to longer discussion articles and example code.

Articles for *VSAM Update* can be sent to the editor, Fiona Hewitt, at any of the addresses shown on page 2. Alternatively, articles can be sent using the Internet to 100336.1412@compuserve.com.

Remember that we pay $250 (£170) per 1,000 words and $140 (£90) per 100 lines of code published (if you give us copyright).

April 1991 – January 1998 index

Items below are references to articles that have appeared in *VSAM Update* since April 1991. References show the issue number followed by the page number(s). Individual copies of all issues from that date are available.

Compute (Bridgend) has announced Release 9.8 of CBLVCAT, its VSAM catalog tuning display utility, which, in addition to being Year 2000 compliant, has in-storage catalog processing for improved performance. VRDS and local timestamp reporting are also included.

For more information, contact:
Compute (Bridgend), 8 Merthyr Mawr Road, Bridgend, Wales, CF31 3NH, UK.
Telephone: (1656) 652222.

* * *

CorVu Corporation, the Minneapolis-based provider of business intelligence solutions, has announced an alliance with Massachusetts-based Liant Software, providing Liant's Relational DataBridge customers with CorVu's Integrated Business Intelligence suite.

Relational DataBridge gives customers access to their VSAM and IMS data in mainframe applications. CorVu will enable such customers to use its business intelligence applications to create *ad hoc* queries and reports against that data. Liant customers will also be able to use CorVu's OLAP analysis and business performance management (balanced scorecard) applications.

For more information, contact CorVu on (800) 610-0769.

* * *

IBM has announced the delivery of SnapShot 1.2, with support for VSAM datasets. The new version supports very large datasets that span multiple volumes, and includes enhancements that extend the power of SnapShot to virtually all mainframe data types in the enterprise today.

For more information, contact your local IBM representative.

* * *

Sterling Software has released Version 2.2 of Vision:Journey, its host access PC tool for incorporating mainframe data into spreadsheets and other PC applications. The new version has 32-bit emulation support for Windows 95 and NT, and users can now launch the program from the PC desktop without needing to use JCL.

For more information, contact:
Sterling Software, 1800 Alexander Bell Drive, Reston, VA 22091, USA.
Telephone: (703) 264 8000.
Sterling Software, 75 London Road, Reading, Berkshire, RG1 5BS, UK.
Telephone: (1734) 391139.

* * *

Software AG has announced that its data warehouse manager system, SourcePoint, will support SAP's Business Application Program Interface (BAPI) once it arrives. This will allow users to integrate operational data contained in SAP's Business Information Warehouse with data extracted from non-SAP systems automatically.

For more information, contact:
Software AG, 11190 Sunrise Valley Drive, Reston VA 22091, USA.
Telephone: (703) 860 5050.
Software AG, Charter Court, 74-78 Victoria St, St Albans, AL1 3XH, UK.
Telephone: (01727) 844455.