February 1998

In this issue

3 Which address spaces are using all the storage?
10 Preload REXX EXECs and save I/Os
16 How to clone datasets
39 Increasing file space allocation
46 Year 2000 aid: change JCL dates – part 2
70 Converting Unix applications to MVS
72 MVS news

© Xepho plc 1998
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: 088 223 1391

Contributions
If you have anything original to say about MVS, or any interesting experience to recount, why not spend an hour or two putting it on paper? The article need not be very long – two or three paragraphs could be sufficient. Not only will you be actively helping the free exchange of information, which benefits all MVS users, but you will also gain professional recognition for your expertise, and the expertise of your colleagues, as well as some material reward in the form of a publication fee – we pay at the rate of £170 ($250) per 1000 words for all original material published in MVS Update. If you would like to know a bit more before starting on an article, write to us at one of the above addresses, and we’ll send you full details, without any obligation on your part.

Editor
Dr Jaime Kaminski

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 MVS Update, comprising twelve monthly issues, costs £310.00 in the UK; $465.00 in the USA and Canada; £316.00 in Europe; £322.00 in Australasia and Japan; and £320.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1992 issue, are available separately to subscribers for £27.00 ($39.00) each including postage.

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

© Xephon plc 1998. All rights reserved. None of the text in this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior permission of the copyright owner. Subscribers are free to copy any code reproduced in this publication for use in their own installations, but may not sell such code or incorporate it in any commercial product. No part of this publication may be used for any form of advertising, sales promotion, or publicity without the written permission of the publisher. Copying permits are available from Xephon in the form of pressure-sensitive labels, for application to individual copies. A pack of 240 labels costs $36 (£24), giving a cost per copy of 15 cents (10 pence). To order, contact Xephon at any of the addresses above.

Printed in England.
Which address spaces are using all the storage?

During a recent exercise to evaluate the effectiveness of the installed Central Storage (CSTOR) and Expanded Storage (ESTOR) on their system, a customer asked me to provide them with some tools to quickly and easily identify exactly which address spaces were consuming these valuable and costly resources. While this information could have been sifted out of various commercially available resource monitoring products, a simple REXX routine running in an ISPF session could give exactly the information required, on-line, real-time, in a neatly formatted panel and at no cost.

I have included two routines here, both use the same general method of chasing through the Address Space Vector Table (ASVT) to pick up Address Space Control Block (ASCB) entries. These point (+X’16C”) to the Real Storage Manager Address Space Block Extension (RAX), which has the required frame and page counts. Hard-coding offset values always mean that a routine might fail with a new version of the operating system, but a quick browse through the new data areas manuals will solve this problem.

The first routine is called STORUSE and is hardware oriented, showing the CSTOR and ESTOR used by the most storage-hungry address spaces in frames and megabytes. To avoid showing all the address spaces in the system, a threshold value is used – any address space using fewer frames than the threshold value is not displayed. The displayed address spaces can be sorted by jobname, or in descending order by the number of frames held in CSTOR or ESTOR or the total of CSTOR and ESTOR.

The results are displayed on the ISPF panel STORUSEP, which must be in the ISPF session panel library concatenation. Additionally, total lines are displayed for the address spaces on the panel, for all address spaces in the system – for frames allocated to common storage areas, and for the frames on-line to MVS. A help panel, STORUSEH, is also included, invoked by PF1 as usual, where any amount of detailed explanation can be included.
STORUSE REXX

/*------------------------------ REXX -------------------------------*/
/* Function   : List ASs using greater than threshold CSTOR/ESTOR.   */
/*-------------------------------------------------------------------*/
numeric digits 21
cth = 2000; eth = 4000; sort = 'T'; sortseq = 'jobn'
do forever
  cfrt = 0; efrt = 0
  cfdr = 0; efrd = 0
  address ispexec, "tbcreate cestab names(jobn cfr cmb efr emb tfr tmb),
    nowrite replace"
  cvt  = storage(d2x(16),4)
  rce  = storage(d2x(c2d(cvt)+c2d(x2c(Ø49Ø))),4)
  asvt = storage(d2x(c2d(cvt)+c2d(x2c(Ø22c))),4)
  asvu = storage(d2x(c2d(asvt)+c2d(x2c(Ø2Ø4))),4)
  maxu = c2d(asvu)
  addr = d2x(c2d(asvt)+c2d(x2c(Ø21Ø)))
  asve = storage(addr,4)
  do i = 1 to maxu
    unus = bitor(substr(asve,1,1),'7f'x)
    if unus = 'ff'x then
      nop
    else
      do
        jbn = d2x(c2d(storage(d2x(c2d(asve)+c2d(x2c(ØØac))),4))))
        if jbn = 0 then
          do
            jbn = d2x(c2d(storage(d2x(c2d(asve)+c2d(x2c(ØØbØ))),4))))
          end
        end
        jobn = storage(jbn,8)
        asn  = storage(d2x(c2d(asve)+c2d(x2c(ØØ24))),2)
        rax  = storage(d2x(c2d(asve)+c2d(x2c(Ø016c))),4)
        fmct = storage(d2x(c2d(rax)+c2d(x2c(Ø02c))),4)
        esct = storage(d2x(c2d(rax)+c2d(x2c(Ø008))),4)
        cfr = c2d(fmct)
        cmb = format(cfr/256,4,0)
        efr = c2d(esct)
        emb = format(efr/256,4,0)
        tfr = cfr + efr
        tmb = format(tfr/256,4,0)
        cfrt = cfrt + cfr
        efrt = efrt + efr
        if cfr > cth then; do
          address ispexec "tbadd cestab"
          cfdr = cfdr + cfr
          efrd = efrd + efr
        end
      else if efr > eth then; do
        address ispexec "tbadd cestab"
        cfdr = cfdr + cfr
      end
  end
  address ispexec, "tbcreate cestab names(jobn cfr cmb efr emb tfr tmb),
    nowrite replace"
  cvt  = storage(d2x(16),4)
  rce  = storage(d2x(c2d(cvt)+c2d(x2c(Ø49Ø))),4)
  asvt = storage(d2x(c2d(cvt)+c2d(x2c(Ø22c))),4)
  asvu = storage(d2x(c2d(asvt)+c2d(x2c(Ø2Ø4))),4)
  maxu = c2d(asvu)
  addr = d2x(c2d(asvt)+c2d(x2c(Ø21Ø)))
  asve = storage(addr,4)
  do i = 1 to maxu
    unus = bitor(substr(asve,1,1),'7f'x)
    if unus = 'ff'x then
      nop
    else
      do
        jbn = d2x(c2d(storage(d2x(c2d(asve)+c2d(x2c(ØØac))),4))))
        if jbn = 0 then
          do
            jbn = d2x(c2d(storage(d2x(c2d(asve)+c2d(x2c(ØØbØ))),4))))
          end
        end
        jobn = storage(jbn,8)
        asn  = storage(d2x(c2d(asve)+c2d(x2c(ØØ24))),2)
        rax  = storage(d2x(c2d(asve)+c2d(x2c(Ø016c))),4)
        fmct = storage(d2x(c2d(rax)+c2d(x2c(Ø02c))),4)
        esct = storage(d2x(c2d(rax)+c2d(x2c(Ø008))),4)
        cfr = c2d(fmct)
        cmb = format(cfr/256,4,0)
        efr = c2d(esct)
        emb = format(efr/256,4,0)
        tfr = cfr + efr
        tmb = format(tfr/256,4,0)
        cfrt = cfrt + cfr
        efrt = efrt + efr
        if cfr > cth then; do
          address ispexec "tbadd cestab"
          cfdr = cfdr + cfr
          efrd = efrd + efr
        end
      else if efr > eth then; do
        address ispexec "tbadd cestab"
        cfdr = cfdr + cfr
      end
efrd = efrd + efr
end
addr = d2x(x2d(addr)+4)
asve = storage(addr,4)
end

select
when sort = 'A' then
  sortseq = 'jobn'
when sort = 'C' then
  sortseq = 'cfr,N,D'
when sort = 'E' then
  sortseq = 'efr,N,D'
when sort = 'T' then
  sortseq = 'tfr,N,D'
otherwise
  sortseq = 'jobn'
end

address ispexec "tbtop cestab"
address ispexec "tbsort cestab fields("sortseq")"
address ispexec "tbbottom cestab"
jobn = ' ' ; cfr = ' ' ; cmb = ' ' ; efr = ' ' ; emb = ' ' ;
tfr = ' ' ; tmb = ' ' ;
address ispexec "tbadd cestab"
jobn = 'Displ AS'
cfr = cfrd; cmb = format(cfr/256,4,O)
efr = efrd; emb = format(efr/256,4,O)
tfr = cfr + efr; tmb = cmb + emb
address ispexec "tbadd cestab"
jobn = 'Total AS'
cfr = cfrt; cmb = format(cfr/256,4,O)
efr = efrt; emb = format(efr/256,4,O)
tfr = cfr + efr; tmb = cmb + emb
address ispexec "tbadd cestab"
jobn = 'Common'
rax = storage(d2x(c2d(rce)+c2d(x2c(ØØ8Ø))),4)
cfr = c2d(storage(d2x(c2d(rax)+c2d(x2c(ØØ2c))),4))
efr = c2d(storage(d2x(c2d(rax)+c2d(x2c(ØØØ8))),4))
cmb = format(cfr/256,4,O)
emb = format(efr/256,4,O)
tfr = cfr + efr; tmb = cmb + emb
address ispexec "tbadd cestab"
jobn = 'Pool'
cfr = c2d(storage(d2x(c2d(rce)+c2d(x2c(ØØØ4))),4))
efr = c2d(storage(d2x(c2d(rce)+c2d(x2c(ØØaØ))),4))
cmb = format(cfr/256,4,O)
emb = format(efr/256,4,O)
tfr = cfr + efr; tmb = cmb + emb
address ispexec "tbadd cestab"
address ispexec "tbtop cestab"
address ispexec "tbdisp1 cestab panel(STORUSEP)"
if rc ^= Ø then
  do
    address ispexec "tbclose cestab"
    exit
  end
address ispexec "vget (sort cth eth)"
address ispexec "tbclose cestab"
end
exit

STORUSEP PANEL

)attr
! type(output) color(green) just(left)
# type(output) color(yellow) just(right)
$ type(output) intens(high)
" type(text) color(turq)
    type(text) skip(on) intens(low)
)body expand(@@)
%@-@ Allocated Cstor and Estor @@
%COMMAND =====>_ZCMD
      %SCROLL
====>_AMT +
%Sort  =====>_Z"(A/C/E/T) %Cstor fr th =====>_Z  + %Estor fr th =====>_Z
+
"        ------------]-------]-------]-------]-------]-------]-------]-------]-------
)model
"  !Z  "]#Z  "]#Z  "]#Z  "]#Z  "]#Z  "]#Z  "
)init
.help = STORUSEH
.zvars = '(sort cth eth jobn cfr cmb efr emb tfr tmb)'
&zcmd = &z
&ztmark = ' '
if (&sort = ' ')
  &sort = 'T'
if (&cth = ' ')
  &cth = '2000'
if (&eth = ' ')
  &eth = '4000'
)proc
vput (sort cth eth)
)end

STORUSEH SOURCE

)attr
$ type(output) intens(high) just(right)
Allocated Cstor and Estor @@

%COMMAND ===> ZCMD

```

Total lines are:

Displ AS - total for the displayed address spaces
Total AS - total for all active address spaces
Common  - data from the common RAX
Pool     - online frames from the RCE

init
proc
&zcont = STORUSEH
end

DATAHYP REXX

The second routine is called DATAHYP and is software oriented. It lists the address spaces which have allocated data spaces or hyper spaces, and also includes CSTOR and ESTOR frame counts for those address spaces. Again the findings are displayed on a panel, DATAHYP, and can be sorted as required.

```
/*------------------------------ REXX -------------------------------*/
/* Function   : List ASs using Data/Hyper spaces.                   */
/*-------------------------------------------------------------------*/
numeric digits 21
sort = 'A'; sortseq = 'jobn'
do forever
  address ispexec,
    "tbcreate esftab names(jobn cfr efr msw mdr dsp hsp),
     nowrite replace"
cvt  = storage(d2x(16),4)
rce  = storage(d2x(c2d(cvt)+c2d(x2c(Ø49Ø))),4)
asvt = storage(d2x(c2d(cvt)+c2d(x2c(Ø22c))),4)
asvu = storage(d2x(c2d(asvt)+c2d(x2c(Ø2Ø4))),4)
maxu = c2d(asvu)
addr = d2x(c2d(asvt)+c2d(x2c(Ø21Ø)))
asve = storage(addr,4)
do i = 1 to maxu
  unus = bitor(substr(asve,1,1),'7f'x)
  if unus = 'ff'x then
    nop
  else
    do
```
jbn = d2x(c2d(storage(d2x(c2d(asve)+c2d(x2c(ØØac)))),4)))
if jbn = Ø then
do
  jbn = d2x(c2d(storage(d2x(c2d(asve)+c2d(x2c(ØØbØ)))),4)))
end
jobn = storage(jbn,8)
asn = storage(d2x(c2d(asve)+c2d(x2c(ØØ24))),2)
rax = storage(d2x(c2d(asve)+c2d(x2c(Ø16c))),4)
cfr = c2d(storage(d2x(c2d(rax)+c2d(x2c(ØØ2c)))),4))
efr = c2d(storage(d2x(c2d(rax)+c2d(x2c(ØØØ8)))),4))
msw = c2d(storage(d2x(c2d(rax)+c2d(x2c(ØØ1Ø))),4))
mdr = c2d(storage(d2x(c2d(rax)+c2d(x2c(ØØ14))),4))
dsp = c2d(storage(d2x(c2d(rax)+c2d(x2c(ØØ1c))),4))
hsp = c2d(storage(d2x(c2d(rax)+c2d(x2c(ØØ34))),4))
if dsp > Ø then; do
  address ispexec "tbadd esftab"
end
else if hsp > Ø then; do
  address ispexec "tbadd esftab"
end
addr = d2x(x2d(addr)+4)
asve = storage(addr,4)
end
select
  when sort = 'A' then
    sortseq = 'jobn'
  when sort = 'C' then
    sortseq = 'cfr,N,D'
  when sort = 'E' then
    sortseq = 'efr,N,D'
  otherwise
    sortseq = 'jobn'
end
address ispexec "tbtop esftab"
address ispexec "tsort esftab fields('sortseq')"
address ispexec "tbtop esftab"
address ispexec "tbdispl esftab panel(DATAHYPP)"
if rc ¬= Ø then
  do
    address ispexec "tbclose esftab"
    exit
  end
address ispexec "vget (sort)"
address ispexec "tbclose esftab"
end
exit

DATAHYPP PANEL

)attr
Address spaces with Dataspaces or Hyperspaces

<table>
<thead>
<tr>
<th>Address</th>
<th>Cstor</th>
<th>Estor</th>
<th>DataSp</th>
<th>HyperSp</th>
<th>Mig SWS</th>
<th>Mig Dref</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fr</td>
<td>Fr</td>
<td>Pg</td>
<td>Pg</td>
<td>Pg</td>
<td>Pg</td>
<td>Pg</td>
</tr>
</tbody>
</table>

!)body expand(@@)

%SCROLL

@-_AMT +

%COMMAND  ===> _ZCMD

%Sort ===> Z"(A/C/E) %

"Model"

| IZ | ]#Z | ]#Z | ]#Z | ]#Z | ]#Z |

)help = DATAHYPH

.proc &zcmd = &z

if (&sort = ' ')

&sort = 'T'

)end

DATAHYPH PANEL SOURCE

)attr

patrick mullen
systems programmer (canada)
Preload REXX EXECs and save I/Os

It is common practice to write separate modules for separate functions. This can be done in REXX. The major drawback is that it produces a multitude of small REXX routines that need to be loaded from disk each time they are used. This can have serious performance implications for specialized routines that are called hundreds or thousands of times.

REXX provides a facility (IRXLOAD) to preload frequently-used REXX EXECs. The following program is used to preload REXX EXECs from SYSEXEC DDNAME. An example of its use is presented after the Assembler source.

SOURCE CODE

REXXLOAD CSECT
REXXLOAD AMODE 31
REXXLOAD RMODE ANY
*
STM R14,R12,12(R13)   SAVE CALLER'S REGISTERS
LR R11,R15           ESTABLISH ADDRESSABILITY
USING REXXLOAD,R11
LR R2,R1             SAVE THE POINTER TO THE PARAM LIST
GETMAIN RU,LV=LSAVE   OBTAIN A DYNAMIC WORK AREA
USING SAVEAREA,R1
ST R1,8(R13)         PUT THE ADDRESS OF PROCESSES SAVE
*                       AREA INTO THE CALLER'S SAVE AREA
ST R13,4(R1)         PUT THE ADDRESS OF PROCESSES SAVE
*                       AREA INTO ITS OWN SAVE AREA
LR R13,R1            LOAD GETMAIN ED AREA ADDRESS
DROP R1               DON'T USE R1 ANY MORE
USING SAVE_AREA,R13   POINT TO THE DYNAMIC AREA
GETMAIN RU,LV=L_WORK_AREA OBTAIN A DYNAMIC WORK AREA
USING WORKA,R1
STM RØ,R1,WORK_AREA_GM_LENGTH SAVE LENGTH AND ADDR OF
*                       DYNAMIC AREA
LR R1Ø,R1
DROP R1
USING WORKA,R1Ø
ST R2,CPPL_PTR        SAVE THE POINTER TO THE PARAM LIST
XC RETCODE,RETCODE    RETCODE=Ø
L R3,0(R2)           LOAD ADDRESS OF PARAMETER LIST
LA R3,2(R3)          POINT TO ROUTINE NAME
*
* CHECK IF THE PROCEDURE IS ALREADY LOADED ?
* TO CALL IRXLOAD, YOU MUST PREPARE:
* 1) EXECBLK
* 2) PARMLIST (FUNCTION, ADD EXECBLK, ADD INSTBLK)
*
MVC RETCODE, ERROR4
*
* BUILD EXECBLK FOR IRXLOAD

LA  R4, EXECBLK
ST  R4, ADD_EXECBLK
MVC ADD_INSTBLK, ZERO
MVC ACRYN, IRXEXECB COPY IRXEXECB TO ACRYN
LA  R4, L_EXECBLK
ST  R4, EXECBLK_LENGTH COPY EXECBLK LENGTH
MVC MEMBER, Ø(R3) COPY ROUTINE NAME
MVC DDNAME, SPACE USE SYSEXEC
MVC SUBCOM, SPACE
MVC DSNPTR, ZERO
MVC DSNLEN, ZERO
*
* BUILD PARMLIST (STATUS, EXECBLK, INSTBLK)

LA  R1, PARMLIST
XR  R0, R0
MVC FONC, STATUS REQUEST STATUS
LA  R4, FONC
ST  R4, Ø(R1)
LA  R4, ADD_EXECBLK POINTER TO EXECBLK
ST  R4, 4(R1)
LA  R4, ADD_INSTBLK POINTER TO INSTBLK
ST  R4, 8(R1)
OI 8(R1), X'80' LAST PARAMETER...
CALL IRXLOAD
LTR R15, R15
BZ  GOBACK RC = Ø THE ROUTINE IS ALREADY LOADED
*
* LOAD THE PROCEDURE

XC  RETCODE, RETCODE RETCODE=Ø
MVC FONC, LOAD
CALL IRXLOAD
LTR R15, R15
BNZ ERROR
B  GOBACK OK, ROUTINE IS LOADED
*
ERROR
MVC RETCODE(4), ERROR12 SET RC = 12
C    R15, =F'-3'
BNE R154
TPUT ERRM3, 4Ø
B  GOBACK
R154  DS  ØH
C    R15, =F'4'
BNE R152Ø
TPUT ERR4, 4Ø
B  GOBACK
R152Ø  DS  ØH
C    R15, =F'20'

BNE R1528
TPUT ERR2Ø,4Ø
B GOBACK
R1528 DS ØH
C R15,=F'28'
BNE R15OTHER
TPUT ERR28,4Ø
B GOBACK
R15OTHER DS ØH
TPUT OTHERERR,4Ø
B GOBACK
GOBACK DS ØH
L R5,RETCODE SAVE RETURN CODE
L R1,WORK_AREA_GM_PTR POINT TO MODULE WORK AREA
FREEMAIN RU,LV=L_WORK_AREA,A=(1)
* FREE THE MODULE WORKAREA
LR R1,R13 LOAD PROCESSES SAVE AREA ADDRESS
L R13,4(R13) CHAIN TO PREVIOUS SAVE AREA
DROP R13
FREEMAIN RU,LV=LSAVE,A=(1)
* FREE THE MODULE SAVEAREA
L R14,12(R13) RETURN ADDRESS
LR R15,R5 RETURN CODE
LM RØ,R12,2Ø(R13) RESTORE REGS 0-12
BSM Ø,R14 RETURN TO THE TMP
ZERO DC F'Ø'
ERROR4 DC F'4'
ERROR12 DC F'12'
LOAD DC CL8'LOAD'
STATUS DC CL8'STATUS'
IRXEXECB DC CL8'IRXEXECB'
SPACE DC CL8''
* ERROR MESSAGES
ERRM3 DC CL4Ø'PROCEDURE NOT FOUND'
ERR4 DC CL4Ø'PROCEDURE NOT LOADED'
ERR2Ø DC CL4Ø'FATAL ERROR'
ERR28 DC CL4Ø'PROCESSOR ENVIRONMENT NOT LOCATED'
OTHERERR DC CL4Ø'UNKNOWN ERROR'
LTORG
WORK_AREA DSECT
WORKA DS ØF
WORK_AREA_GM_LENGTH DS F LENGTH OF WORKAREA
WORK_AREA_GM_PTR DS F ADDRESS OF WORKAREA
CPPL_PTR DS F ADDRESS OF PARAMETER LIST
RETCODE DS F RETURN CODE
IRXLOAD DS ØD
FONC DS D
ADD_EXECBLK DS F
ADD_INSTBLK DS F
EXECBLK DS ØD EXECBLK
ACRYN DS CL8
SAMPLE USE OF REXXLOAD

Below is an example that highlights the benefits of the REXXLOAD EXEC. The first step in the example without preload initiates 30,017 I/Os. The second step with preload produces only 38 I/Os. This results in a saving of 99.87%. Example output can be seen in Figure 1.

RLOADR0 REXX ROUTINE

/* REXX */

DO 10000
   'RLOADR1'
END

This first REXX routine calls REXX routine RLOADR1 10,000 times a second.
JOB EXECUTION

JES2 JOB LOG -- SYSTEM PROD -- NODE JES EXP

JOB07718  TSS7001I 1990557 Last-Used 09 Sep 97 17:58 System=PROD Facility=BATCH
JOB07718  TSS7001I Count=1060 Mode=Fail Locktime=None Name=RENAUD PATRICK
JOB07718  $HASP373 I990557A STARTED - INIT 44 - CLASS 4 - SYS PROD
JOB07718  IEF403I I990557A - STARTED - TIME=17.59.59

JOB07718  - - - - - - - - - - - - - - - - TIMINGS (MINS.)- - - - - - - - - - - - - - - - PAING COUNTS- - - - TAPE
JOB07718 -JOBNAME STEPNAME RC EXCP CPU SRB CLOCK SERV PG PAGES SWAPS VIO SW S/SNS
JOB07718 -I990557A STEP1 00 3017 .57 .02 3.4 K 0
JOB07718 -I990557A STEP2 00 38 .45 .00 .8 K 0
JOB07718 IEF404I I990557A - ENDED - TIME=18.04.17
JOB07718 -I990557A ENDED. NAME-RENAUD TOTAL CPUSIME= 0 TOTAL ELAPSED TIME= 4.3
JOB07718 $HASp395 I990557A ENDED

------ JES2 JOB STATISTICS ------
09 SEP 1997 JOB EXECUTION DATE
41 CARDS READ
103 SYSOUT PRINT RECORDS
0 SYSOUT PUNCH RECORDS
6 SYSOUT SPool KBYTES
4.30 MINUTES EXECUTION TIME

Figure 1: Sample output of REXXLOAD
RLOADR1 ROUTINE

/* REXX */
DO 1ØØ
   A=2Ø
END
RETURN

SAMPLE JOB

//*
//*==================================
//* FISRT STEP WITHOUT PRELOAD =
//*==================================
//*
//*STEP1 EXEC PGM=IKJEFT01,DYNAMNBR=6Ø
//*SYSTSPRT DD SYSOUT=* 
//*SYSEXEC DD DISP=SHR,DSN=I99Ø557.ASM.SOURCE
//*SYSTSIN DD *

/* CALL FISRT PROCEDURE */
RLOADRØ

/*
лом
/*==================================
//* SECOND STEP WITH PRELOAD =
/*==================================
//*
//*STEP2 EXEC PGM=IKJEFT01,DYNAMNBR=6Ø
//*SYSTSPRT DD SYSOUT=* 
//*SYSEXEC DD DISP=SHR,DSN=I99Ø557.ASM.SOURCE
//*SYSTSIN DD *

/* PRELOAD REXX ROUTINES */
CALL 'I99Ø557.ASM.LOAD.TEST(REXXLOAD)' 'RLOADRØ'
CALL 'I99Ø557.ASM.LOAD.TEST(REXXLOAD)' 'RLOADR1'

/* CALL FISRT PROCEDURE */
RLOADRØ

/*

Patrick Renard
(France) © Xephon 1998

How to clone datasets

THE CLONEDS1 EXEC
Have you ever wanted to take copies of numerous datasets? If you have, then you will know that it is time consuming to create an empty dataset and then IEBCOPY (or whatever) the existing dataset into the new one. To speed up the process, the following EXEC clones the attributes and contents of an existing dataset in one easy command, and because it is an EXEC, you can stack numerous invocations in a single EXEC. The EXEC is invoked as:

`%CLONEDS1 morn iputdsn oputdsn autsub dorn kjob nwait`

Where:

- **Morn** – should prompts be displayed on screen; 1 Yes, 0 No.
- **Iputdsn** – the name of the dataset to be cloned.
- **Oputdsn** – the name of the new dataset to be created.
- **Autsub** – E means force to the EXEC (rather then IEBCOPY), J means force to a JOB (ie use IEBCOPY), L means let the EXEC decide. To retain member statistics, you must use a job for copying (E).
- **Dorn** – automatically reply YES (Y) or no (N) to the delete prompt.
- **Kjob** – delete the generated job and output (Y) or (N).
- **Nwait** – number of passes before being prompted to see if you want to continue processing.

Below are some examples of how to specify different names for the output dataset:

```bash
%CLONEDS1 1 MY.DATASET.ELIB *OLD
```

This will clone the dataset MY.DATASET.ELIB to MY.DATASET.ELIB.OLD. Messages will be written to the screen, and you will be prompted for a delete confirmation if
MY.DATASET.ELIB.OLD already exists. The EXEC will decide whether to use a job or not.

%CLONEDS1 1 MY.DATASET.ELIB OLDER* E

This will clone the dataset MY.DATASET.ELIB to OLDER.MY.DATASET.ELIB. Messages will be written to the screen, and you will be prompted for a delete confirmation if OLDER.MY.DATASET.ELIB already exists. The EXEC will always use a job to do the copying.

To replace the first high-level qualifier with a two-part high-level qualifier, use $n$, where $s$ is required, and $n$ is the number of parts of the first high-level qualifier to be replaced. The parameter $n$ can take any numeric value. To add a qualifier at the end of the output name where you have replaced the high-level qualifier, prefix the end qualifier with a '+'.

%CLONEDS1 1 MY.DATASET.ELIB $1YOUR E

This will clone the dataset MY.DATASET.ELIB to YOUR.DATASET.ELIB (ie the $1$ means replace the first part of the dataset name with whatever follows the $1$ – no spaces after the $1$). Messages will be written to the screen, and you will be prompted for a delete confirmation if YOUR.DATASET.ELIB already exists. The E means force the EXEC to submit an IEBCOPY/IEBGENER job to do the copying.

%CLONEDS1 1 MY.DATASET.ELIB $2YOUR.DATASET.JILL

This will clone the dataset MY.DATASET.ELIB to YOUR.DATASET.JILL.ELIB. Messages will be written to the screen, and you will be prompted for a delete confirmation if YOUR.DATASET.JILL.ELIB already exists.

%CLONEDS1 1 MY.DATASET.ELIB $2YOUR.DATASET.JILL+HARRY

This will clone the dataset MY.DATASET.ELIB to YOUR.DATASET.JILL.ELIB.HARRY. Messages will be written to the screen, and you will be prompted for a delete confirmation if YOUR.DATASET.JILL.ELIB.HARRY already exists.

If you want to rename lots of datasets in one go, create a member (CLON1) in an ELIB dataset, containing the following:
This will create YOUR.DATASET.CLIST.NEW.
If you are running the above, say over lunch or from a batch job, then I would use:

`%CLONEDS1 MY.DATASET.CLIST $1YOUR+NEW * * * Ø`

The 0 will suppress the prompt about continuing to process if the embedded copy job is still running. If you are sitting in front of the terminal while the EXEC is running, then I would accept the default value for nwait, just in case there is a problem with the initiators, etc.

```rexx
/* REXX */
trace n
/* morn iputdsn oputdsn autsub dorn kjob nwait */
parse upper arg biglin
call hedlin1_text(' MSG (020) ')
say biglin
say copies('-',79)
/******************************************************************************/
testflag = ' '
parsed var biglin bigres '<' testflag
jk = 0
Do while testflag ^= ' '
jk = jk + 1
   parse var testflag testf.jk testflag
End /* Do until testflag = ' ' */
/* trlvl = Ø - do not write out any intermediate stats. */
/* trlvl = 1 - write out high-level intermediate stats. */
trlvl = Ø
jk1 = Ø
Do while jk > jk1
   jk1 = jk1 + 1
   If(testf.jk1 = 'TRACE') then Do
      trace r
   End /* If(testf.jk1 = 'TRACE') */
   If(testf.jk1 = 'LVL1') then Do
      trlvl = 1
   End /* If(testf.jk1 = 'LVL1') */
End /* Do jk = 1 to jk */
parsed var bigres morn iputdsn oputdsn autsub dorn kjob .
If(pos('.',morn) =Ø) then Do
   Nop
```
End /* If(pos(morn,'.') =Ø) then Do */
Else Do
    morn = 1
    iputdsn = ' '
    oputdsn = ' '
    autsub = ' '
    dorn = ' '
    kjob = ' '
    nwait = ' '
    parse var bigres iputdsn oputdsn autsub dorn kjob nwait .
End /* If(pos(morn,'.') =Ø) then Do */
If(pos(morn,'HELP') ≠ Ø 3 morn = '?' 3 morn = ' ') then Do
    xx => To clone a dataset EXEC <'
    say center(xx,79,'=')
    say ' This EXEC clones the attributes and contents of an existing' ,
       'dataset.'
    say ' ';
    say 'The EXEC is invoked as ===>' ,
       '%CLONEDS1 morn iputdsn oputdsn autsub dorn kjob nwait' 
    say ' ';
    say 'morn    - should prompts be displayed on the screen 1',
       'Yes. Ø No. Put 1.'
    say 'iputdsn - is the dataset to be cloned.'
    say 'oputdsn - is the new dataset to be created.'
    linp.1 = 'To specify a new name of old name suffixed with OLD'
    linp.2 = 'simply put *OLD'
    linp.Ø = 2
    indent = 1Ø
    call format_text
    linp.1 = 'To specify a new name of old name prefixed with OLD'
    linp.2 = 'simply put OLD*' 
    linp.Ø = 2
    indent = 1Ø
    call format_text
    linp.1  = 'To replace the first hlq with a two part hlq, then '
    linp.2  = 'use >n,' 
    linp.3  = 'where > is required, and n is the number of parts of'
    linp.4  = 'first'
    linp.5  = 'hlq to be replaced. The parameter n can take any '
    linp.6  = 'value.'
    linp.7  = 'To add a qualifier at the end of the output name where'
    linp.8  = 'you have replaced the hlq, then prefix the end qualifier'
    linp.9  = 'with a +.'
    linp.1Ø = 'See below for examples.'
    linp.Ø = 1Ø
    indent = 1Ø
    call format_text
    say 'autsub  - E means force to EXEC, J means force to a JOB'
    say ' L means let the EXEC decide.'
    say ' To retain member statistics - you MUST use a job',
       'for copying (E).'
say 'dorn    - automatically reply YES to the delete '
    'prompt (Y) or not (N).'
say 'kjob    - delete the generated job and output (Y) or (N).'
say 'nwait   - is the number of passes before the EXEC prompts'
linp.1 = 'you if you want to continue processing. If it tells'
linp.2 = 'you that it is EXECUTING, then reply Y. Otherwise'
linp.3 = 'it might be that an initiator is not available. In this'
linp.4 = 'case reply N, and check the initiators. If you are'
linp.5 = 'running this in batch or over lunch, then put nwait to Ø'
linp.6 = 'to bypass any checking.'
linp.Ø = 6
indent = Ø
call format_text
say ''
say ''
say copies('-',78)
say 'Examples:'
say copies('-',78)
say '%CLONEDS1 1 MY.ELIB  *OLD '
linp.1 = 'This will clone dataset MY.ELIB to MY.ELIB.OLD'
linp.2 = 'Messages will be written to the screen, and you will be'
linp.3 = 'prompted for a delete confirmation if MY.ELIB.OLD'
linp.4 = 'already exists.'
linp.5 = 'The EXEC will decide whether to use a job or not.'
linp.Ø = 4
indent = Ø
call format_text
say copies('-',78)
say '%CLONEDS1 1 MY.ELIB OLDER*  L'
linp.1 = 'This will clone dataset MY.ELIB to'
linp.2 = 'OLDER.MY.ELIB'
linp.3 = 'Messages will be written to the screen, and you will be'
linp.4 = 'prompted for a delete confirmation if OLDER.MY.ELIB'
linp.5 = 'already exists.'
linp.6 = 'The EXEC will decide whether to use a job or not.'
linp.Ø = 6
indent = Ø
call format_text
say copies('-',78)
say '%CLONEDS1 1 MY.ELIB  ›1YOUR.FRED  E'
linp.1 = 'This will clone dataset MY.ELIB to YOUR.FRED.ELIB'
linp.2 = 'Messages will be written to the screen, and you will be'
linp.3 = 'prompted for a delete confirmation if YOUR.FRED.ELIB'
linp.4 = 'already exists.'
linp.5 = 'Force the EXEC to submit an IECOPY/IEBGENER job to'
linp.6 = 'do the copying.'
linp.Ø = 6
indent = Ø
call format_text
say copies('-',78)
say '%CLONEDS1  1  MY.FRED.ELIB  ›2YOUR.JILL'
linp.1 = 'This will clone dataset MY.FRED.ELIB to'
linp.2 = 'YOUR.JILL.ELIB'
linp.3 = 'Messages will be written to the screen, and you will be'
linp.4 = 'prompted for a delete confirmation if YOUR.JILL.ELIB'
linp.5 = 'already exists.'
linp.Ø = 5
indent = Ø

call format_text
say copies('-',78)
say '%CLONEDS1  1  MY.FRED.ELIB  ›2YOUR.JILL+HARRY'
linp.1 = 'This will clone dataset MY.FRED.ELIB to'
linp.2 = 'YOUR.JILL.ELIB.HARRY.'
linp.3 = 'Messages will be written to the screen, and you will be'
linp.4 = 'prompted for a delete confirmation if'
linp.5 = 'YOUR.JILL.ELIB.HARRY'
linp.6 = 'already exists.'
linp.Ø = 6
indent = Ø

call format_text
say copies('›',78)
say 'What happens if you have PREFIX PROFILE set to a userid?
  FRED.ELIB ›1YOUR                                      
  iputdsn ===> MY.FRED.ELIB                             
  oputdsn ===> MY.YOUR.ELIB                             
--------------------------------------------------------
  FRED.ELIB *OLD                                        
  iputdsn ===> MY.FRED.ELIB                             
  oputdsn ===> MY.FRED.ELIB.OLD                         
--------------------------------------------------------
  FRED.ELIB YOUR*                                       
  iputdsn ===> MY.FRED.ELIB                             
  oputdsn ===> MY.YOUR.FRED.ELIB                        
--------------------------------------------------------

say 'I had TSO PROFILE PREFIX(MY) set.                     */
/* FRED.ELIB ›1YOUR                                     */
/* iputdsn ===> MY.FRED.ELIB                             */
/* oputdsn ===> MY.YOUR.ELIB                             */
/* -------------------------------------------------------- */
/* FRED.ELIB *OLD                                        */
/* iputdsn ===> MY.FRED.ELIB                             */
/* oputdsn ===> MY.FRED.ELIB.OLD                         */
/* -------------------------------------------------------- */
/* FRED.ELIB YOUR*                                       */
/* iputdsn ===> MY.FRED.ELIB                             */
/* oputdsn ===> MY.YOUR.FRED.ELIB                        */
/* */
/* Some statistics. */
/* */
/* 3Nos mems3 */
/* 3--------3 */
/* 3 53 5393 */
/* 3---3----3 */
/* EXEC 3 23 2323 */
/* job 3 53 183 */
/* 3--------3 */
/* */
/* Using 5 as the cut-off mark could be a little low. The more */
/* members you are processing, the greater the advantage of using */
/* a job in preference to letting the EXEC do the work. */
/* */
/* */
/* Check that all parameters have been entered. */
/* */
If(dorn = ' ' 3 dorn = '*') then Do
dorn = 'N'
End /* If(dorn = ' ') then Do */
If( dorn = 'Y' 3 dorn = 'N') then Do
  If(dorn = 'Y') then Do
dorn = 'D'
  ans  = 'D'
  sayexp.1 = ' - The EXEC will automatically reply YES to the ' 33 ,
             'del prompt.'
  End /* If(dorn = 'Y') then Do */
  Else Do
    sayexp.1 = ' - The EXEC will NOT automatically reply YES to '33,
              'the del prompt.'
    End /* If(dorn = 'Y') then Do */
End /* If( dorn = 'Y' 3 dorn = 'N') then Do */
Else Do
  say 'Invalid DORN specified of===>' dorn'.  DORN must be Y or N.'
call helps1
  exit
End /* If( dorn = 'Y' 3 dorn = 'N') then Do */

If(morn = Ø 3 morn = 1 3 morn = 2) then Do
  Nop
End /* If(morn = Ø 3 morn = 1) then Do */
Else Do
  say 'The first parameter morn must be Ø or 1 (usually 1) and not:'
say morn
  say 'Am exiting the EXEC.'
call helps1
exit
End /* If(morn = Ø 3 morn = 1) then Do */
If(morn = Ø) then Do
   If(dorn = 'Y' 3 dorn = 'N' 3 dorn = 'D') then Do
      Nop
   End /* If(dorn = 'Y' 3 dorn = 'N') then Do */
   Else Do
      say 'If you specify MORN as ' morn ' then DORN must be Y or N'
      iflag1 = Ø
      Do while iflag1 = Ø
         say 'Please enter a value for DORN (Y or N or Q to quit) ===>
         pull ans
         upper ans
         If(ans = 'Q') then Do
            say 'Exiting the EXEC'
            exit
         End /* If(ans = 'Q') then Do */
         If( ans = 'Y' 3 ans = 'N') then Do
            iflag1 = 1
            If(ans = 'Y') then Do
               dorn = 'D'
            End /* If(ans = 'Y') then Do */
            Else Do
               dorn = ans
            End /* If(ans = 'Y' 3 ans = 'N') then Do */
         End /* If(ans = 'Y' 3 ans = 'N') then Do */
         Else Do
            call hedlin1_text(' MSG (Ø1Ø) ')
            say 'You gave an invalid response of ' ans
         End /* If(ans = 'Y' 3 ans = 'N') then Do */
      End /* Do while iflag1 = Ø */
   End /* If(morn = Ø) then Do */
Else Do
   If(kjob = 'Y' 3 kjob = '*') then Do
      kjob = 'Y'
      sayexp.2 = ' - The processing jobs will be kept'
   End /* If(kjob = 'Y') then Do */
   Else Do
      kjob = 'N'
      sayexp.2 = ' - The processing jobs will not be kept'
   End /* If(kjob = 'Y') then Do */
End /* If(morn = Ø 3 morn = 1) then Do */
call hedlin1_text(' MSG (Ø16) ')
say 'You put nwait = ' owait 'The EXEC is making it 'nwait
End /* If(nwait < 10) then Do */
End /* If(datatype(nwait,N) = 1) then Do */
Else Do
    call hedlin1_text(' MSG (Ø15) ')
say 'You gave an invalid response for nwait of ' nwait
say 'Am exiting the EXEC'
exit
End /* If(datatype(nwait,N) = 1) then Do */
End /* If(nwait = Ø 3 nwait = '*') then Do */
sayexp.4 = ,
' - You will be prompted to continue after this many passes'
****************************************************************************
/* Check if absolute dataset names specified.                             */
****************************************************************************
If(substr(iputdsn,1,1) = '') then Do
    iasi = 1
    iputdsn = strip(iputdsn,B,'''')
End /* If(substr(iputdsn,1,1) = '') then Do */
Else Do
    iasi = Ø
End /* If(substr(iputdsn,1,1) = '') then Do */
End /* If(substr(iputdsn,1,1) = '') then Do */
If(substr(oputdsn,1,1) = '') then Do
    iaso = 1
    oputdsn = strip(oputdsn,B,'''')
End /* If(substr(oputdsn,1,1) = '') then Do */
Else Do
    iaso = Ø
End /* If(substr(oputdsn,1,1) = '') then Do */
****************************************************************************
/* Check if a PREFIX is being used.                                     */
****************************************************************************
yy = SYSVAR(SYSPREF)
If(yy = ' ') then Do
    Nop
End /* If(yy = ' ') then Do */
Else Do
    If(iasi = Ø) then Do
        iputdsn = yy 33 '.' 33 iputdsn
    End /* If(iasi = Ø) then Do */
    Else Do
        iputdsn = iputdsn
    End /* If(iasi = Ø) then Do */
    If(iaso = Ø) then Do
        oputdsn = yy 33 '.' 33 oputdsn
    End /* If(iaso = Ø) then Do */
    Else Do
        oputdsn = oputdsn
    End /* If(iaso = Ø) then Do */
End /* If(yy = ' ') then Do */
FLICTHATHE INPUT/OUTPUTDATASETNAMESSAVE/*
If(iputdsn = ' ' 3 iputdsn = '*') then Do
   iputdsn = 'MY.FRED.TEST'
   oputdsn = 'MY.FRED.TEST.OLD'
End /* If(iputdsn = ' ') then Do */
If(iputdsn = oputdsn) then Do
   call hedlin1_text(' MSG (Ø3Ø) ')
   say 'The input ' iputdsn ' and '
   say ' output ' oputdsn
   say 'datasets are identetical.'
   say 'This is not allowed.'
   say 'Re-enter the command with different'.
   'datasets specified as the input and output.'
   exit
End /* If(iputdsn = oputdsn) then Do */
parse var oputdsn oputdsn '+' endbit
Select
   When (substr(oputdsn,1,1) = '>') then Do
      t1 = translate(iputdsn,' ','.'
      nn = words(t1)
      kn = substr(oputdsn,2,1)
      kn = kn + 1
      newn = ''
      Do jk = kn to nn
         newn = newn 33 '.' 33 subword(t1,jk,1)
      End /* Do jk = kn to nn */
      newn = substr(oputdsn,3) 33 newn
      oputdsn = newn
   If(endbit = ' ') then Do
      Nop
   End /* If(endbit = ' ') then Do */
   Else Do
      oputdsn = oputdsn 33 '.' 33 endbit
   End /* If(endbit = ' ') then Do */
   End /* When (substr(oputdsn,1,1) = '>') then Do */
   Otherwise Do
      loput = length(oputdsn)
      Select
         When (substr(oputdsn,1,1) = '*') then Do
            If(substr(oputdsn,2,1) = '.') then Do
               oputdsn = iputdsn 33 '.' 33 substr(oputdsn,3)
            End /* If(substr(oputdsn,2,1) = '.') then Do */
            Else Do
               oputdsn = iputdsn 33 '.' 33 substr(oputdsn,2)
            End /* If(substr(oputdsn,2,1) = '.') then Do */
         End /* When (substr(oputdsn,1,1) = '*') then Do */
         When (substr(oputdsn,1,1) = '*') then Do
            yy = loput - 1
         End /* When (substr(oputdsn,1,1) = '*') then Do */
oputdsn = substr(oputdsn,1,yy) 33 '.' 33 iputdsn
End /* When (substr(oputdsn,loput,1) == '*') then Do */
When (loput > 8) then Do
  Nop
End /* When (loput > 8) then Do */
Otherwise Do
  call hedlin1_text(' MSG (Ø4Ø) ')
  linp.1 = 'The output dataset name was incorrectly specified.'
  linp.2 = 'You specified a prefix or suffix of ' oputdsn
  linp.3 = 'but did not say whether it was a prefix or suffix.'
  linp.4 = 'If you want a prefix specify '*33 oputdsn 'or if you'
  linp.5 = 'want a suffix specify ' oputdsn 33 '*'
  linp.Ø = 5
  indent = Ø
  call format_text
  exit
End /* Otherwise Do */
End /* Select */
End /* Otherwise Do */
End /* Select */
/**************************************************************************/ /* Check if the input dataset exists - If No, then exit. */
/***************************************************************************/
xx = outtrap('gvar."
Address 'TSO'
"LISTDS '" 33 iputdsn 33 '""
rc = rc
"FREE DATASET("' 33 iputdsn 33 "")"
xx = outtrap(OFF)
If(rcc = Ø) then Do
  y1 = subword(gvar.3,1,1) /* RECFM */
  y2 = subword(gvar.3,2,1) /* LRECL */
  y3 = subword(gvar.3,3,1) /* BLKSIZE */
  y4 = subword(gvar.3,4,1) /* PO */
  If(trlvl = 1) then Do
    say 'y1 is ' y1
    say 'y2 is ' y2
    say 'y3 is ' y3
    say 'y4 is ' y4
  End /* If(trlvl = 1) then Do */
  If (autsub = '' 3 autsub = '*') then Do
    autsub = 'L'
    sayexp.3 = ' - means let the EXEC decide between a job and ' 33.
    'the EXEC.'
  End /* If (autsub = ' ') then Do */
  iaut = Ø
  If(pos('F',y1) = Ø) then Do
    autsub = 'J'
    iaut = 1
  End /* If(pos('F',y1) = Ø) then Do */
  If(y2 > 256) then Do
autsub = 'J'
iaut = 2
End /* If(y2 > 256) then Do */
If(y4 = 'PS' 3 y4 = 'PO') then Do
    Nop
End /* If(y4 = 'PS' 3 y4 = 'PO') then Do */
Else Do
    call hedlin1_text(' MSG (Ø5Ø) ')
say 'The type of dataset ' y4 ' is invalid. (Should be PS/PO)
    exit
End /* If(y4 = 'PS' 3 y4 = 'PO') then Do */
End /* If(rcc = Ø) then Do */
Else Do
    call hedlin1_text(' MSG (Ø6Ø) ')
say 'The input dataset ' iputdsn ' does not exist'
    exit
End /* If(rcc = Ø) then Do */
If(autsub = 'J') then Do
    sayexp.3 = '- means force to a JOB.'
End /* If(autsub = 'J') then Do */
If(autsub = 'E') then Do
    sayexp.3 = '- means force to EXEC.'
End /* If(autsub = 'E') then Do */
If(trlvl = 1) then Do
    say 'iaut is (Ø=none, 1=y1, 2=y2) ' iaut
    say 'autsub is ' autsub
End /* If(trlvl = 1) then Do */
/****************************
/* Write out what parameters the EXEC thinks it is using.          */
/****************************
If(morn = 1 3 trlvl= 1) then Do
    say 'morn    ===>' morn
    say 'iputdsn ===>' iputdsn
    say 'oputdsn ===>' oputdsn
    say 'autsub  ===>' autsub sayexp.3
    say 'dorn    ===>' dorn sayexp.1
    say 'kjob    ===>' kjob sayexp.2
    say 'nwait   ===>' nwait sayexp.4
    say copies('*',79)
End /* If(morn = 1 3 trlvl= 1) then Do */
/****************************
/* Check that the output dataset name is not too long.            */
/****************************
lopds = length(oputdsn)
If(lopds > 44) then Do
    call hedlin1_text(' MSG (Ø65) ')
say 'The output dataset name is ' lopds ' characters long.'
say 'The maximum length is 44.'
    exit
End /* If(lopds > 44) then Do */
/****************************
/* Check if the output dataset exists - If YES, then prompt.      */
/****************************
xx = outtrap('gvar.(')
Address 'TSO'
"LISTDS "' 33 oputdsn 33 ""
rcc = rc
"FREE DATASET("' 33 oputdsn 33 ")"
x = outtrap(OFF)
If(rcc = Ø) then Do
iflag1 = Ø
If(morn >= 1) then Do
If (ans = 'D') then Do
iflag1 = Ø
If(morn = 1) then Do
ans = dorn
If(ans = 'Q' 3 ans = 'N') then Do
If(ans = 'Q') then Do
If(ans = 'D') then Do
If(ans = 'Q') then Do
End /* If(ans = 'D') then Do */
End /* If(ans = 'D') then Do */
If(ans = 'Q') then Do
If(ans = 'D') then Do
End /* If(ans = 'D') then Do */
End /* If(morn = 1) then Do */
linp.5 = 'The default value is N'.
linp.6 = 'This means do not delete/define.'
linp.7 = 'You either specified N or * in the command.'
linp.Ø = 7
indent = Ø
call format_text
exit
End /* If(ans = 'Q') then Do */
If(ans = 'D') then Do
xx = outtrap('gvar.')
Address 'TSO'
"DELETE '" 33 oputdsn 33 '""
xx = outtrap(OFF)
End /* If(ans = 'D') then Do */
End /* If(rcc = Ø) then Do */
/*************************************************************************/
/* Allocate the output dataset.                                        */
/*************************************************************************/
Address TSO "ALLOC DATASET('"33 oputdsn 33"')",
"LIKE('" 33 iputdsn 33 '"')"
xx = outtrap('gvar.')
"FREE DATASET('" 33 iputdsn 33 '"')"
"FREE DATASET('" 33 oputdsn 33 '"')"
xx = outtrap(OFF)
If(morn = 2) then Do
call hedlin1_text(' MSG (110) ')
say 'Exiting after creating the o/p d/s oputdsn
say 'Nothing has been copied to it.'
exit
End /* If(morn = 2) then Do */
If(y4 = 'PO') then Do
If(autsub = 'J') Then Do
xm = 9999
End /* If(autsub = 'J') Then Do */
Else Do
/*************************************************************************/
/* Get a list of member names from the PDS                           */
/*************************************************************************/
xx = outtrap('gvar.')
Address 'TSO'
"LISTDS '" 33 iputdsn 33 '"' Members"
xx = outtrap(OFF)
iflag1 = Ø
Do jk = 1 to gvar.Ø while iflag1 = Ø
  If(subword(gvar.jk,1,1) = '--MEMBERS--') then Do
    iflag1 = 1
  End /* If(subword(gvar.jk,1,1) = '--MEMBERS--') then Do */
End /* Do jk = 1 to gvar.Ø while iflag1 = Ø */
If(iflag1 = 1) then Do
xx = Ø
Do jkk = jk to gvar.Ø
xm = xm + 1
    mem.xm = subword(gvar.jkk,1,1)
End /* Do jkk = jk to gvar.Ø */
End /* If(iflag1 = 1) then Do */
say 'Dataset ' iputdsn ' contains ' xm ' members.'
End /* If(autsub = 'J') Then Do */
/* iflag3 = Ø - undecided. */
/* iflag3 = 1 - means use a job */
/* iflag3 = 2 - means use the EXEC */
iflag3 = Ø
/* If more than 5 members use a job, irrespective of autsub. */
/* If autsub = J, then use a job */
/* If autsub = E, then force to use the EXEC, irrespective of mems*/
If( xm > 5) then Do
    iflag3 = 1
End /* If( xm > 5) then Do */
If( (5 >= xm) & (autsub = 'L') ) then Do
    iflag3 = 2
End /* If( (5 >= xm) & (autsub = 'L') ) then Do */
If(autsub = 'E') then Do
    iflag3 = 2
End /* If(autsub = 'E') then Do */
If( (xm > 5) & (autsub = 'E') ) then Do
    st.1 = 'The i/p d/s ' iputdsn ' contains more than 5 members'
    st.2 = 'and you are forcing it to use the EXEC.'
    iflag4 = 1
End /* If( (xm > 5) & (autsub = 'E') then Do */
End /* If(y4 = 'PO') then Do */
Else Do
    Address "TSO"
    interpret "xx = LISTDSI(" 33 iputdsn 33 ")"
    n1 = SYSUSED
    n2 = SYSUNITS
    dd = sysreason
    If(trlvl = 1) then Do
        say 'n1  is ' n1
        say 'n2  is ' n2
        say 'dd  is ' dd
    End /* If(trlvl = 1) then Do */
    iflag3 = 1
    If(dd > Ø) then Do
        call hedlin1_text(' MSG (12Ø) ')
        say 'A reason code of ' dd ' was retunred from the ' 'LISTDSI ' iputdsn ' command.'
        linp.1 = 'The EXEC will try and continue - but please check the'
        linp.2 = 'output dataset carefully:'
        linp.Ø = 2
        indent = Ø
        call format_text
        say oputdsn
    End /* If(dd > Ø) then Do */
If (n2 = 'BLOCK' & n1 < 5) then Do
  iflag3 = 2
End /* If (n2 = 'BLOCK' & n1 < 5) then Do */
If (autsub = 'E') then Do
  iflag3 = 2
End /* If(autsub = 'E') then Do */
If (n2 = 'TRACK' & n2 = 'CYLINDER') & (autsub = 'E') then Do
  st.1 = 'The sequential dataset was defined in ',
  'cylinders/tracks'
  st.2 = 'and you are forcing it to use the EXEC.'
  iflag4 = 1
End /* If (n2 = 'TRACK' & n2 = 'CYLINDER') & autsub = 'E' */
End /* If(y4 = 'PO') then Do */
If (iflag4 = 1) then Do
  iflag1 = Ø
  Do while iflag1 = Ø
    call hedlin1_text(' MSG (13Ø) ')
    say st.1
    say st.2
    say 'Do you want to use the EXEC? (Y/N)'
pull ans
  upper ans
  If (ans = 'Y' & ans = 'N') then Do
    iflag1 = 1
  If (ans = 'N') then Do
    iflag3 = 1
    call hedlin1_text(' MSG (14Ø) ')
    say 'Am switching from the EXEC to a job'
  End /* If(ans = 'N') then Do */
  Else Do
    If (ans = 'Q') then Do
      say 'Am exiting the EXEC'
      exit
    End /* If (ans = 'Q') then Do */
  End /* Else do */
  Else Do
    call hedlin1_text(' MSG (15Ø) ')
    say 'Will continue to use the EXEC.'
  End /* If(ans = 'Q') then Do */
End /* If(ans = 'N') then Do */
Else Do
  call hedlin1_text(' MSG (16Ø) ')
  say 'You gave an invalid response of ' ans
End /* If (ans = 'D' & ans = 'Q') then Do */
End /* Do while iflag1 = Ø */
End /* If(iflag4 = 1) then Do */
If (trlvl = 1) then Do
  say 'iflag3 is ' iflag3
End /* If(trlvl = 1) then Do */
/*****************************/
/* Start the clock!! */
/**************************************************************/
xxtim = TIME('R')
/**************************************************************/
/* Use a job rather than the EXEC.                          */
/**************************************************************/
If(iflag3 = 1) then Do
  /**************************************************************/
  /* Build the IEBCOPY or IEBGENER JCL.                     */
  /**************************************************************/
  jcl = userid() 33 "C"
  jclf.1 = "//" 33 jcl 33 " JOB ".
            "5555,'CLONEDS1',CLASS=F,MSGCLASS=E"
  If(y4 = 'PO') then Do
    /**************************************************************/
    /* Build the IEBCOPY deck.                               */
    /**************************************************************/
    jclf.Ø = 11
    jclf.2 = "//*
    jclf.3 = "//COPYEM EXEC PGM=IEBCOPY"
    jclf.4 = "//SYSUT1 DD UNIT=SYSDA,SPACE=(CYL,(5,1))"
    jclf.5 = "//IN DD DISP=SHR,DSN=" 33 iputdsn
    jclf.6 = "//OUT DD DISP=SHR,DSN=" 33 oputdsn
    jclf.7 = "//SYSPRINT DD SYSOUT="
    jclf.8 = "//SYSIN DD *"
    jclf.9 = " COPY OUTDD=OUT,INDD=IN"
    jclf.10 = "*/"
    jclf.11 = "//**
  End /* If(y4 = 'PO') then Do */
Else Do
  /**************************************************************/
  /* Build the IEBGENER deck.                              */
  /**************************************************************/
  jclf.Ø = 7
  jclf.2 = "//STEP1 EXEC PGM=IEBGENER"
  jclf.3 = "//SYSPRINT DD SYSPRT=E 
  jclf.4 = "//SYSUT1 DD DISP=SHR,DSN=" 33 iputdsn
  jclf.5 = "//SYSUT2 DD DISP=SHR,DSN=" 33 oputdsn
  jclf.6 = "//SYSIN DD dummy "
  jclf.7 = "//**
  End /* If(y4 = 'PO') then Do */
  /**************************************************************/
  /* Build the timestamp part of the dataset names.         */
  /**************************************************************/
  xx = TIME(L)
  yy = translate(xx,' ',':',')
  zz = ''
  Do jk = 1 to words(yy)
    zz = zz 33 subword(yy,jk,1)
  End /* Do jk = 1 to words(yy) */
  zz = "D" 33 substr(zz,1,7)
  /**************************************************************/
  /* Build the input job dataset dsn2.                      */
  /**************************************************************/
dsn2 = userid() 33 ".CLONEDS1.JOB." 33 zz
xx = outtrap('gvar.')
"Delete " 33dsn2 33"
xx = outtrap(off)
"Attr out lrecl(80) bksize(6400) recfm(f b) dsorg(ps)"
"Alloc fi(writeo) da('"33dsn233"') ",
"unit(sysda) using(out) space(3,1) tracks",
"catalog"
"Free attrlist(out)"
"Free attrlist(writeo)"
"Alloc fi(outp) da('"dsn2"') shr"
"EXECIO" jcf.Ø "DISKW OUTP (FINIS STEM JCLF.)"
"Free fi(outp)"
/*--------------------------------------------------------------*/
/* Submit the copy job, and wait for it to complete. */
/*--------------------------------------------------------------*/
xx = outtrap('gvar.')
"SUBMIT "33 dsn2 33"
xx = outtrap(off)
iflag2 = Ø
Do kl = 1 to gvar.Ø
  If(subword(gvar.kl,3,1) = 'SUBMITTED' 3 ,
    subword(gvar.kl,4,1) = 'SUBMITTED') then do
    iflag2 = 1
    nkl = kl
  End /* if(subword(gvar.1,4,1) = 'submitted') then do */
End /* do kl = 1 to gvar.Ø */
If(iflag2 = Ø) then do
  dattim = right(date('n'),11,'Ø') time('n')
  wex =dattim 'job not submitted' dsn2
  say wex
  "se "wex " u("33userid()33") logon"
  exit
End /* if(iflag2 = Ø) then do */
Else do
  If(subword(gvar.nkl,3,1) = 'SUBMITTED' ) then do
    jobnm = subword(gvar.nkl,2,1)
  End /* if(subword(gvar.nkl,3,1) = 'submitted' ) then do */
  If(subword(gvar.nkl,4,1) = 'SUBMITTED' ) then do
    jobnm = subword(gvar.nkl,3,1)
  End /* if(subword(gvar.nkl,4,1) = 'submitted' ) then do */
  yy = pos('(',jobnm,1)
  yy = yy - 1
  /*----------------------------------------*/
  /* Wait until the job completes. */
  /*----------------------------------------*/
  /* Every iwait1 of nwait put out a msg giving the status of */
  /* the job. */
iwait1 = Ø
iflag1 = Ø
Do until iflag1 = 1
    gvar. = ' '
    xx = outtrap('gvar. ')
    'status ' jobnm
    xx = outtrap('off')
    If(subword(gvar.1,3,3) = 'ON OUTPUT QUEUE' 3 ,
       subword(gvar.1,4,3) = 'ON OUTPUT QUEUE') then do
        iflag1 = 1
    End /* if(subword(gvar.1,4,3) = 'on output queue') then do */
    Else Do
        If(subword(gvar.1,3,1) = 'EXECUTING') then Do
            Nop
        End /* If(subword(gvar.1,3,1) = 'EXECUTING') then Do */
        Else Do
            If(nwait = Ø) then Do
                Nop
            End /* If(nwait = Ø) then Do */
            Else Do
                If(morn = 1 3 trlvl= 1) then Do
                    iwait1 = iwait1 + 1
                End /* If(morn = 1 3 trlvl= 1) then Do */
                If(iwait1 > nwait) then Do
                    iwait1 = Ø
                    iflag4 = Ø
                    Do while iflag4 = Ø
                        call hedlin1_text(' MSG (17Ø) ')
                        say 'The status of job:' gvar.1
                        say 'Do you wish to continue processing?(Y/N)'
                        pull ans
                        upper ans
                        If( ans = 'Y' 3 ans = 'N') then Do
                            iflag4 = 1
                        If(ans = 'N') then Do
                            iflag3 = 1
                            call hedlin1_text(' MSG (18Ø) ')
                            say 'Am exiting the EXEC.'
                            say 'You will need to cancel:' jobnm
                            exit
                        End /* If(ans = 'N') then Do */
                        Else Do
                            If (ans = 'Q') then Do
                                say 'Am continuing processing'
                            End /* If (ans = 'Q') then Do */
                            End /* If(ans = 'N') then Do */
                            End /* If( ans = 'Y' 3 ans = 'N') then Do */
                        Else Do
                            call hedlin1_text(' MSG (19Ø) ')
                            say 'You gave an invalid response of ' ans
                        End /* If(ans = 'Q') then Do */
                        End /* Do while iflag4 = Ø */
                End /* If(iwait1 > nwait) then Do */
End /* If(morn = 1 3 trvl= 1) then Do */
End /* If(nwait = Ø) then Do */
End /* If(subword(gvar.1,3,1) = 'EXECUTING') then Do */
End /* if(subword(gvar.1,4,3) = 'on output queue') then do */
End /* do until iflag1 = 1 */
End /* if(iflag2 = Ø) then do */
/*--------------------------------------------------------------*/
/* Define the output dataset dsn3. */
/*--------------------------------------------------------------*/
dsn3 = userid() 33 ".CLONEDS1.OUT." 33 zz
gvar. = ' '
x = outtrap('gvar.')
"Delete "33 dsn3 33 ""
"Free fi(inp)"
"Attr out lrecl(133) blksize(1330) recfm(f b) dsorg(ps)"
"Alloc fi(inp) da('"33dsn333"') " ,
"unit(sysda) using(out) space(3,1) CYLINDERS" ,
"catalog"
"Free attrlist(out)"
"Free fi(inp)"
Address 'TSO'
"OUT " jobnm "PRINT('"33 dsn3 33"')"
x = outtrap(off)
"ALLOC FI(PIN2) DA('"33dsn333"') SHR"
"EXECIO * DISKR PIN2 (FINIS STEM GEND."
"free fi(pin2)"
/*--------------------------------------------------------------*/
/* Interrogate the output to check for condition codes. */
/*--------------------------------------------------------------*/
iflag2 = Ø
idel1 = Ø
Do jk = 1 to gend.Ø
  xx = substr(gend.jk,2)
  yy = subword(xx,1,1)
  If(subword(xx,1,1) = 'IEF142I') then Do
    parse var xx '- COND CODE' retcod .
    If(retcod = 0000) then Do
      If(y4 = 'PS') then Do
        idel1 = 1
      End /* If(Y4 = 'PS') then Do */
    End /* If(retcod = 0000) then Do */
  Else Do
    call hedlin1_text( ' MSG (2ØØ) ' )
    say 'The job for copying between datasets'
    say iputdsn
    say oputdsn
    say 'has returned a condition code of ===>' retcod
    say 'The job is in dataset:' dsn2
    say 'The output from the job is in dataset:' dsn3
    say '(Ø1Ø) Please investigate.'
  idel1 = Ø
End /* If(retcod = Ø) then Do */
iflag2 = 1
End /* If(subword(xx,1,1) = 'IEF142I') then Do */
If(y4 = 'PO') then Do
If(subword(xx,1,1) = 'IEB159I') then Do
If(subword(xx,2,3) = 'NO MEMBERS COPIED') then Do
  call hedlin1_text(' MSG (210) ')
  say 'No members were copied from/to:
  say iputdsn
  say oputdsn
  say 'The job is in dataset:' dsn2
  say 'The output from the job is in dataset:' dsn3
  say 'Please investigate.'
  idel1 = 0
End /* If(subword(xx,2,3) = 'NO MEMBERS COPIED') then Do */
End /* If(subword(xx,1,1) = 'IEB159I') then Do */
If(subword(xx,1,1) = 'IEB1098I') then Do
  msg1 = subword(xx,2,5)
  call hedlin1_text(' MSG (220) ')
  say msg1
  say iputdsn
  say oputdsn
  nosi = subword(xx,2,1)
  noso = subword(xx,4,1)
  If(nosi = noso) then Do
    idel1 = 1
  End /* If(nosi = noso) then Do */
Else Do
  call hedlin1_text(' MSG (230) ')
  say 'Not all members were copied.'
  say 'The job is in dataset:' dsn2
  say 'The output from the job is in dataset:' dsn3
  say 'Please review the dataset.'
  idel1 = 0
End /* If(nosi = noso) then Do */
End /* If(subword(xx,1,1) = 'IEB1098I') then Do */
End /* If(y4 = 'PO') then Do */
End /* Do jk = 1 to gend.Ø while iflag2 = Ø */
If(iflag2 = Ø) then Do
  call hedlin1_text(' MSG (240) ')
  say 'The EXEC could not find reference to IEF142I in the copy job output.'
  say 'The job is in dataset:' dsn2
  say 'The output from the job is in dataset:' dsn3
  say 'Please review the dataset.'
  idel1 = 0
End /* If(iflag2 = Ø) then Do */
/*--------------------------------------------------------------*/
/* If all copied cleanly, then delete the job and output dsets. */
/*--------------------------------------------------------------*/
If(kjob = 'Y') then Do

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
If(morn = 1 and trlvl= 1) then Do
    call hedlin1_text(' MSG (250) ')
    say 'The datasets used to perform the copy will not be deleted:'
    say dsn2
    say dsn3
    say copies('**',79)
End /* If(morn = 1 and trlvl= 1) then Do */
End /* If(kjob = 'Y') then Do */
Else Do
    If (idel1 = 1) then Do
        xx = outtrap('gvar. ')
        "Delete " 33dsn2 33"
        "Delete " 33dsn3 33"
        xx = outtrap(off)
    End /* If (idel1 = 1) then Do */
End /* If(kjob = 'Y') then Do */
End /* If(iflag3 = 1) then Do */
If(iflag3 = 2) then Do
    /***************************************************************/
    /* Use the EXEC to read in each mem of the i/p and wrt to o/p. */
    /***************************************************************
    If(y4 = 'PO') then Do
        Do jk = 1 to xm
            dsni = iputdsn 33 "(" 33 mem.jk 33 ")"
            Address TSO "ALLOC FI(DD1) DA('dsni') SHR"
            "EXECIO * DISKR DD1 (STEM MIDS. FINIS"
            Address TSO "FREE F(DD1)"
            dsno = oputdsn 33 "(" 33 mem.jk 33 ")"
            Address TSO "ALLOC FI(DD2) DA('dsno') SHR"
            "EXECIO " mids.Ø "DISKW DD2 (STEM MIDS. FINIS"
            Address TSO "FREE F(DD2)"
        End /* Do jk = 1 to xm */
    End /* If(y4 = 'PO') then Do */
Else Do
    dsni = iputdsn
    Address TSO "ALLOC FI(DD1) DA('dsni') SHR"
    "EXECIO * DISKR DD1 (STEM MIDS. FINIS"
    Address TSO "FREE F(DD1)"
    dsno = oputdsn
    Address TSO "ALLOC FI(DD2) DA('dsno') SHR"
    "EXECIO " mids.Ø "DISKW DD2 (STEM MIDS. FINIS"
    Address TSO "FREE F(DD2)"
End /* If(y4 = 'PO') then Do */
End /* If(iflag3 = 2) then Do */
If(iflag3 = 1 3 iflag3 = 2) then Do  
If(morn = 1) then Do  
  des1 = 'not sure'  
  If(iflag3 = 1) then des1 = 'a Job'  
  If(iflag3 = 2) then des1 = 'the EXEC'  
  call hedlin1_text(' MSG (27Ø) ')  
  say 'The d/s' iputdsn  
  say 'has been cloned using ' des1 ' to:'  
  say oputdsn  
End /* If(morn = 1) then Do */  
End /* If(iflag3 = 1 3 iflag3 = 2) then Do */

/************************************************************/  
/* Say how long the process has taken. */  
/************************************************************/  
xxtim = TIME('E')  
If(morn = 1) then Do  
  call hedlin1_text(' MSG (28Ø) ')  
  say 'Time taken to process is:' xxtim 'seconds.'  
End /* If(morn = 1) then Do */
exit

/*****************************************
*/
helps1:  
ms1 = 'Press PF1'  
ms2 = , 'CLONEDS1 morn(1) i/p o/p dorn(N) kjob nwait '  
ZEDSMGS = ms1  
ZEDLMSG = ms2  
"ISPEXEC SETMSG MSG(ISRZØØØ)"  
return

/*****************************************/
format_text: procedure  expose indent linp.  
mb = Ø  
Do jk = 1 to linp.Ø  
  Do until linp.jk = " "  
    mb = mb + 1  
    parse var linp.jk wd.mb linp.jk  
    wd.mb = strip(wd.mb,B)  
  End /* Do until linp.jk = " " */  
End /* Do jk = 1 to linp.Ø */

lenl = 78 - indent  
xp = 1  
x1 = Ø  
jkl = Ø  
wrt.l = ''  
Do until jkl >= mb  
  jkl = jkl + 1  
  x1 = x1 + length(wd.jkl) + 1
Increasing file space allocation

INTRODUCTION
There are many occasions when I need to increase the size of a file that I am editing. When for example:

• The directory of a PDS is full, and using COMPRESS will not create more space.
• I need to copy the members from one PDS to another, and the receiving PDS has insufficient space.

The INCREASE CLIST is written in REXX and runs under TSO ISPF. It will increase the space allocated to a PDS. The CLIST INCRSQS will do the same for sequential (QSAM) files.
INCREASE CLIST

/* REXX */

ADDRESS TSO "FREE FILE(TEMPCOP)"
DSNAMEOFILE = ''
IF ARG() = Ø THEN CALL MAP
ELSE DO
    PARSE ARG DSNAMEOFILE
    IF DSNAMEOFILE = '' THEN EXIT 4
    ADDRESS TSO "LISTCAT ENTRY("DSNAMEOFILE") ALL "
    IF RC = Ø THEN CALL PROC
    ELSE CALL NOTFOUND
END

PROC:
ADDRESS TSO "FREE FILE(TEMPCOP)"
ADDRESS ISPEXEC
   "LMFREE DATAID("SYSUT1")"
   "LMFREE DATAID("SYSUT2")"
SAY DSNAMEOFILE 'FILE NAME OF DATASET TO BE INCREASED'
X = LISTDSI(DSNAMEOFILE "DIRECTORY" "RECALL")
DO 3; SAY ''; END
SAY 'NOM : ' LEFT(SYSDSNAME,15)
SAY ''
SAY 'LIFE-CYCLE DETAILS:'
SAY ' CREATION  : ' LEFT(SYSCREATE,15) 'PASSWORD     : ' SYSPASSWORD
SAY ' EXPIRE    : ' LEFT(SYSEXDATE,15)  'RACF         : ' SYSRACFA
SAY ' LAST  REF : ' LEFT(SYSREFDATE,15) 'INDIC MAJ    : ' SYSUPDATED
SAY ''
SAY 'ATTRIBUTS:'
SAY ' DSORG     : ' LEFT(SYSDSORG,15)
SAY ' RECFM     : ' LEFT(SYSRECFM,15)   'BLKSIZE      : ' SYSBLKSIZE
SAY ' LRECL     : ' LEFT(SYSLRECL,15)   'KEYLEN       : ' SYSKEYLEN
SAY ''
SAY 'ALLOCATION:'
SAY ' TYPE      : ' LEFT(SYSUNITS,15)
SAY ' PRIMARY   : ' LEFT(SYSPRIMARY,15) 'USED         : ' SYSUSED
SAY ' SECONDARY : ' LEFT(SYSSECONDS,15) 'EXTENTION    : ' SYSEXTEXTENTS
SAY ''
SAY 'LOCATION:'
SAY ' VOLUME    : ' LEFT(SYSVOLUME,15) 'TRACKS/ CYLS : ' SYSTRKSCYL
SAY ' TYPE      : ' LEFT(SYSUNIT,15) 'BLOCK / TRACK  ' SYSBLKSTRK
SAY ''
V1 = SYSDSNAME
V2 = SYSUNIT
V3 = SYSRECFM
V4 = SYSLRECL
V5 = SYSBLKSIZE
V6 = SYSALLOC
V7 = SYSEXDATE
V8 = SYSDSORG
V9 = SYSMEMBERS
V10 = SYSUSED
V11 = SYSUDIRBLK
V12 = SYSEXTENTS
V13 = SYSPRIMARY
V14 = SYSSECONDS
V15 = SYSUNITS
V16 = SYSDSNAME

/*******************************************/
/* COMPUTE FOR THE NEW ALLOCATION */
/*******************************************/
NV6 = (2 * V6)
NV11 = (2 * V11)
NV13 = (2 * V13)
NV14 = (2 * V14)

/*******************************************/
/* FOR THE RECFM VALUE */
/*******************************************/
VAR3A = F
VAR3B = B
IF SYSRECFM = FB THEN DO
VAR3A = F
VAR3B = B
END
ELSE NOP

/*******************************************/
/* IF THE ALLOCATION USED BLOCK COMPUTE THE VALUE*/
/* OF THE BLOCK */
/*******************************************/
IF SYSUNITS = BLOCK THEN DO
NV15 = "BLOCK'!!'('NV13!!')'
SAY NV15
END
ELSE NV15 = V15

SAY NV15
IF SYSRECFM = T THEN VAR3A = T ELSE NOP
IF SYSRECFM = V THEN VAR3B = V ELSE NOP
IF SYSRECFM = U THEN VAR3A = U ELSE NOP
IF SYSRECFM = D THEN VAR3A = D ELSE NOP
IF SYSRECFM = F THEN VAR3A = F ELSE NOP
IF SYSRECFM = M THEN VAR3A = M ELSE NOP
IF SYSRECFM = S THEN VAR3A = S ELSE NOP
SAY VAR3A VAR3B NV15
DATASET = DSNAMEOFFILE /* JUST FOR THE LENGTH OF THE VARIABLE */
USE OF TEMPORARY FILE

ADDRESS TSO "DELETE 'USERID().TEMP."V16"'
    IF RC > 8 THEN EXIT RC
    ELSE NOP

ALLOCATION AFTER DELETE OF THE TEMPORARY DATASET

ADDRESS TSO "ALLOCATE DATASET('USERID().TEMP."V16") FILE(TEMPCOP)"
    DSORG('V8') SPACE('NV14','NV14') "NV15" RELEASE DIR('NV11')
    LRECL('V4') BLKSIZE('V5') RECFM('VAR3A','VAR3B')
    NEW CATALOG"

ADDRESS TSO "LISTCAT ENTRY('USERID().TEMP."V16") ALL "
    IF RC > Ø THEN EXIT RC
    ELSE NOP

COPY PDS OR PDS INTO PDS OR PDSE FOR INCREASE

ADDRESS ISPEXEC
    "LMINIT DATAID(SYSUT1) DATASET("DSNAMEOFFILE") ENQ(SHR)"
    "LMINIT DATAID(SYSUT2) DATASET('USERID().TEMP."V16") ENQ(SHR)"
    "LMCOPY FROMID('SYSUT1') FROMMEM(*)"
    "TODATAID('SYSUT2') REPLACE"

SELECT
    WHEN RC=Ø THEN DO
        ZEDLMSG="THE MEMBER"MEMBER" HAS BEEN COPIED"
        ZEDSMSG="OK"
        "SETMSG MSG(ISRZØØ1)"
    END

    WHEN RC=4 THEN DO
        ZEDLMSG="THE MEMBER "MEMBER" DOES NOT EXIST",
            " VIDE ; RC="RC
        ZEDSMSG="EMPTY FILE !! "
        "LMFREE DATAID('SYSUT1')"
        "LMFREE DATAID('SYSUT2')"
        "SETMSG MSG(ISRZØØ1)"
        EXIT RC
    END

    WHEN RC=8 THEN DO
        ZEDLMSG="THE MEMBER "MEMBER" DOES NOT EXIST;RC="RC
        ZEDSMSG="MEMBER NOT FOUND !! "
        "SETMSG MSG(ISRZØØ1)"
        "LMFREE DATAID('SYSUT1')"
        "LMFREE DATAID('SYSUT2')"
        EXIT RC
    END
OTHERWISE
   DO
      ZEDLMSG="ERROR LCOPIE ; RC="RC
      ZEDMSMG="ERROR"
      "SETMSG MSG(ISRZØØ1)"
      "LMFREE DATAID("SYSUT1")"
      "LMFREE DATAID("SYSUT2")"
      EXIT RC
   END
END
"LMFREE DATAID("SYSUT1")"
"LMFREE DATAID("SYSUT2")"
ADDRESS TSO "RENAME "DATASET" '"V16".OLD.INCREASE.A.DELETE'"
      IF RC > Ø THEN EXIT RC
      ELSE NOP
ADDRESS TSO "RENAME '"USERID()'.TEMP."V16"' "DATASET"
      IF RC > Ø THEN EXIT RC
      ELSE NOP
ADDRESS TSO "FREE FILE(TEMPCOP)"
      IF RC > Ø THEN EXIT RC
      ELSE NOP
END
EXIT RC
NOTFOUND:
   SAY 'FILE NOT FOUND REENTER THE NAME OF THE FILE '
   CALL MAP
RETURN RC
MAP:
   SAY "ENTER THE NAME OF THE FILE WITH QUOTES"
   PARSE EXTERNAL DSNAMEOFILE
   SAY DSNAMEOFILE 'FILE NAME OF DATASET TO BE INCREASED'
   ADDRESS TSO "LISTCAT ENTRY("DSNAMEOFILE") ALL "
      IF RC = Ø
         THEN CALL PROC
      ELSE CALL NOTFOUND
RETURN RC

INCRSQ CLIST
/* REXX */
FICHIER = ''
IF ARG() = Ø   THEN CALL MAP
   ELSE DO
      PARSE ARG FILE
      IF FILE = '' THEN EXIT 4
      ADDRESS TSO "LISTCAT ENTRY("FILE") ALL "
      IF RC = Ø
         THEN CALL PROC
      ELSE CALL NOTFOUND
   END
PROC:
X = LISTDSI(FILE DIRECTORY NORECALL)
DO 3; SAY ''; END
SAY 'NAME : ' LEFT(SYSDSNAME,15)
SAY ''
SAY 'LIFE-CYCLE DETAILS:'
SAY ' CREATION : ' LEFT(SYSCREATE,15) 'PASSWORD : ' SYSPASSWORD
SAY ' EXPIRE : ' LEFT(SYSEXDATE,15) 'RACF : ' SYSRACFA
SAY ' LAST REF : ' LEFT(SYSREFDATE,15) 'INDIC MAJ : ' SYSUPDATED
SAY ''
SAY 'ATTRIBUTES:'
SAY ' DSORG : ' LEFT(SYSDSORG,15)
SAY ' RECFM : ' LEFT(SYSSRECFM,15) 'BLKSIZE : ' SYSBLKSIZE
SAY ' LRECL : ' LEFT(SYSLRECL,15) 'KEYLEN : ' SYSKEYLEN
SAY ''
SAY 'ALLOCATION:'
SAY ' TYPE : ' LEFT(SYSUNITS,15)
SAY ' PRIMARY : ' LEFT(SYSPRIMARY,15) 'USED : ' SYSUSED
SAY ' SECONDARY : ' LEFT(SYSSECONDS,15) 'EXTENTION : ' SYSEXTENTS
SAY ''
SAY 'LOCATION :'
SAY ' VOLUME : ' LEFT(SYSVOLUME,15) 'TRACKS/ CYLS : ' SYSTRKSCYL
SAY ' TYPE : ' LEFT(SYSUNIT,15) 'BLOCK / TRACK : ' SYSBLKSTRK
SAY ''
SAY 'DIRECTORY:'
SAY 'NB BLOCKS ALLOC : ' SYSADIRBLK
SAY 'NB BLOCKS USED : ' SYSUDIRBLK
SAY 'NB THE MEMBER : ' SYSMEMBERS
SAY ''
SAY 'REASONS CODE : ' SYSREASONS
SAY 'ERROR MESSAGE : ' SYMSMSGLVL1
SAY ''
SAY "THE DATASET NAME IS" SYSDSNAME
SAY "THE VOLUME IS" SYSVOLUME
SAY "THE DEVICE UNIT IS" SYSUNIT
SAY "THE RECORD FORMAT IS" SYSSRECFM
SAY "THE LOGICAL RECORD LENGTH IS" SYSLRECL
SAY "THE BLOCKSIZE IS" SYSBLKSIZE
SAY "THE ALLOCATION IN SPACE UNIT IS" SYSALLOC
SAY "THE TYPE OF RACF PROTECTION IS" SYSRACFA
SAY "THE EXPIRY DATE IS" SYSEXDATE
SAY "THE ORGANIZATION IS" SYSDSORG
SAY "THE KEYLENGTH IS" SYSKEYLENGTH
SAY "THE NUMBER OF MEMBERS IS" SYSMEMBERS
SAY "THE K% USED IS" SYSUSED
SAY "THE USED DIRECTORY BLOCK IS" SYSUDIRBLK
SAY "THE CHANGED INDICATOR IS" SYSUPDATED
SAY "LATS REF DATE" SYSREFDATE
SAY "NUMBERS OF EXTENTS" SYSEXTENTS
SAY "NUMBERS OF PRIMARY ALLOCATION" SYSPRIMARY

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
SAY " NUMBERS OF SECONDARY ALLOCATION " SYSSECONDS
SAY " SPACE UNITS IS " SYSUNITS
V1 = SYSDSNAME
V2 = SYSUNIT
V3 = SYSRECFM
V4 = SYSLRECL
V5 = SYSBLKSIZE
V6 = SYSALLOC
V7 = SYSEXDATE
V8 = SYSDSORG
V9 = SYSMEMBERS
V10 = SYSUSED
V11 = SYSUDIRBLK
V12 = SYSEXIENTS
V13 = SYSPRIMARY
V14 = SYSSECONDS
V15 = SYSUNITS
V16 = SYSDSNAME
NV6 = (2 * V6 )
NV13 = (2 * V13 )
NV14 = (2 * V14 )
VAR3A = F
VAR3B = B
IF SYSRECFM = FB THEN DO
  VAR3A = F
  VAR3B = B
END
ELSE NOP
IF SYSRECFM = T THEN VAR3A = T ELSE NOP
IF SYSRECFM = U THEN VAR3B = U ELSE NOP
IF SYSRECFM = D THEN VAR3A = D ELSE NOP
IF SYSRECFM = F THEN VAR3A = F ELSE NOP
IF SYSRECFM = M THEN VAR3A = M ELSE NOP
IF SYSRECFM = S THEN VAR3A = S ELSE NOP
IF V5 > 65535 THEN V5 = 65535 ELSE NOP
SAY VAR3A VAR3B
ADDRESS TSO "DELETE '"USERID()'.TEMP.V16'"
ADDRESS TSO "ALLOCATE DATASET('"USERID()'.TEMP."V16") FILE(TEMPCOP)
  DSORG('V8') SPACE('NV14','NV14') "V15" RELEASE
  LRECL('V4') BLKSIZE('V5') RECFM('VAR3A','VAR3B')
  NEW CATALOG"
ADDRESS TSO "LISTCAT ENTRY('"USERID()'.TEMP."V16") ALL "
ADDRESS TSO "FREE FILE(SYSIN)"
ADDRESS TSO "FREE FILE(SYSPRINT)"
ADDRESS TSO "ALLOCATE FILE(SYSPRINT) DSN(*) REUSE "

Year 2000 aid: change JCL dates – part 2

This month we complete our look at the source code for the YEAR2KC EXEC. This reads a PDS, identifies EXEC statements, and determines if these statements contain ‘DATE=’ fields within a ‘PARM=’ operand. When such fields are found they are modified to a specified date.

Claude Dunand  
(France)  
© Xephon 1998
MVI GU02LL+1,X'50'   SET ENTRY LENGTH TO 80
MVC GU02NAM, MEMBER  MOVE MEMBER NAME INTO BLDL AREA
LA R1,PDS           R1 POINTS TO OPEN DCB
LA R0,BLDLNLTRY     R0 POINTS TO BLDL ENTRY AREA
BLDL (R1),(R0)      EXECUTE BLDL
LTR R15,R15        TEST RETURN CODE
*          Ø0 - FOUND
*          Ø4 - NOT FOUND
*          Ø8 - I/O ERROR OR VS SHORTAGE
BNZ GSRETURN  EXIT IF NOT NORMAL RETURN
TM GU02C,X'80'    IF AN ALIAS
BNO GSRETURN     THEN
LA R15,12        TURN ON ALIAS FLAG
GSRETURN L       RESTORE LINKAGE REGISTER
BR RBAL         RETURN

******************************************************************************
***  WRITE TSO STATISTICS  ***
******************************************************************************
PUTSTATS ST RBAL,SAVPSBAL  SAVE LINKAGE REGISTER
OC GU02DATC, GU02DATC  CREATION DATE BINARY ZEROS?
BZ RDNOSTAT     YES
MVC LINE+1(11),=C'ISPF STATS: '
UNPK LINE+13(6), GU02TTR(L'GU02TTR+1) UNPACK TTR NYBLS
NC LINE+13(5),=B'F'   MASK OUT ZONES
TR LINE+13(5),=C'0123456789ABCDEF' CONVERT TO DIXPLAY
XR R1,R1       CLEAR REGISTER
IC R1, GU02MOD  GET MODIFICATION
ST R1, DOUBLE  SAVE
IC R1, GU02VER  GET VERSION
MH R1, =H'100'   MOVE 2 DECIMAL DIGITS LEFT
A R1, DOUBLE    ADD MODIFICATION
CVD R1, DOUBLE  CONVERT TO DECIMAL
MVC LINE+18(7),=X'402021204B2020' SET EDIT PATTERN
ED LINE+18(7), DOUBLE+5 FORMAT VV.MM
ICM R1, B'1111', GU02DATC GET CREATION DATE
ST R1, JGYYDDD  SAVE FOR CONVERSIONT
BAL RBAL, JULGREG CONVERT TO MM/DD/YY
MVC LINE+26(8), JGMMDDYY MOVE TO LINE
ICM R1, B'1111', GU02DATM GET CREATION DATE
ST R1, JGYYDDD  SAVE FOR CONVERSIONT
BAL RBAL, JULGREG CONVERT TO MM/DD/YY
MVC LINE+35(8), JGMMDDYY MOVE TO LINE
UNPK LINE+46(5), GU02TIMM(3) UNPACK MODIFIED TIME
MVC LINE+45(2), LINE+46 MOVE HH LEFT
MVI LINE+47, C':' SEPARATE HH:MM
LH R1, GU02SIZE  LOAD SIZE FROM DIRECTORY
CVD R1, DOUBLE   CONVERT TO DECIMAL
MVC LINE+50(7), EDITPAT SET EDIT PATTERN
ED LINE+50(7), DOUBLE+5 FORMAT SIZE
LH R1, GU02INIT  LOAD INITIAL SIZE FROM DIRECTORY
CVD R1,DOUBLE CONVERT TO DECIMAL
MVC LINE+57(7),EDITPAT SET EDIT PATTERN
ED LINE+57(7),DOUBLE+5 FORMAT SIZE
ICM R1,'B'ØØ11',GUØ2MOD LOAD COUNT OF MOD LINES
CVD R1,DOUBLE CONVERT TO DECIMAL
MVC LINE+64(7),EDITPAT SET EDIT PATTERN
ED LINE+64(7),DOUBLE+5 FORMAT SIZE
MVC LINE+71(7),GUØ2ID MOVE USER ID TO LINE
BAL RBAL,PRINT PRINT STATISTICS
L RBAL,SAVPSBAL RESTORE LINKAGE REGISTER
BR RBAL RETURN
EJECT

***********************************************************************
*** REWRITE ANY CHANGED RECORDS                                      ***
***********************************************************************
WRITEREC ST RBAL,SAWWRBAL SAVE LINKAGE REGISTER
LA 2,DECBA POINT TO DECB
WRITE (2),SF,PDS,MF=E READ BLOCK FROM MEMBER
CHECK (2) AWAIT ECB POSTING
TM SWITCHES,X'FF'-UPDATBIT RESET UPDATE BIT
WRRETURN L RBAL,SAWWRBAL RESTORE LINKAGE REGISTER
BR RBAL RETURN
EJECT

***********************************************************************
*** WRITE ERROR LINES                                                ***
***********************************************************************
PUTERR ST RBAL,SAVPEBAL SAVE LINKAGE REGISTER
AP ERRORTOT,'=P'1' COUNT ERROR
MVC INAREA+L'INAREA(9),='<=BEFORE' SET IMAGE
PUT ERRORS,OUTAREA WRITE BEFORE IMAGE
MVC INAREA+L'INAREA(9),='<=AFTER ' SET IMAGE
L R1,INRECLOC POINT TO MODIFIED RECORD
MVC INAREA,Ø(R1) MOVE AFTER IMAGE
PUT ERRORS,OUTAREA WRITE AFTER IMAGE
L RBAL,SAVPEBAL RESTORE LINKAGE REGISTER
BR RBAL RETURN
EJECT

***********************************************************************
*** SCAN RECORD TO SEE IF DATE PARM APPEARS ON //XXX EXEC JCL         ***
*** STATEMENT. REPLACE ANY DATES FOUND WITH SPECIFIED DATE.           ***
***********************************************************************
SCANREC ST RBAL,SAVSRBAL SAVE LINKAGE REGISTER
MVC MEMBNAME,Ø(R6) MOVE MEMBER NAME
MVC MEMBERNO,EDITPAT MOVE EDIT PATTERN
ED MEMBERNO,MEMBERS+1 FORMAT MEMBER NUMBER
MVC INAREA,Ø(R1) MOVE RECORD
AP CARDS,'=P'1' COUNT CARD IMAGE
MVC CARNO,EDITPAT MOVE EDIT PATTERN
ED CARNO,RECORDS+1 FORMAT CARD NUMBER
TM OPTIONS,LISTBIT+DIAGBIT LIST OR DIAGNOSE?
BZ NOLIST NO
BAL  RBAL,TESTX  PRINT DIAGNOSTIC LINE
*  B  Nolist  BYPASS VANILLA LISTING
*  BAL  RBAL,PRINT  PRINT CARD IMAGE

Nolist  STM  R5,R9,SAVE5TO9  SAVE REGISTERS
CLC  =C'//',Ø(R1)  JCL CARD?
BNE  NOTJCL  NO
CLI  2(R1),C'*'  COMMENTS CARD?
BE  NOTJCL  YES
TM  SWITCHES,CONTBIT  CONTINUATION CARD EXPECTED?
BO  CONTINUUD  YES
NI  SWITCHES,UPDATABIT  TURN OFF ALL EXCEPT UPDATE SWITCH

CONTINUUD  LR  R6,R1  POINT TO BEGINNING OF CARD IMAGE
BCTR  R6,Ø  POINT TO PREVIOUS BYTE
LA  R4,NOTJCL  SET NULL RETURN
XR  R7,R7  CLEAR FIELD LENGTH
LA  R8,72  SET CARD IMAGE FIELD LENGTH
BAL  RBAL,TEST  FOR TESTING
BAL  R14,KHNSCAN  SKIP PAST '//NAME'
LA  R1,1(R6,R7)  POINT PAST FIRST BLANK AT STMNT END
ST  R1,AVSP1  SAVE FOR POSSIBLE SHIFTING OF STMNT
XC  AVSP2,AVSP2  INITIALIZE SECOND POSSIBLE SPACE
BAL  RBAL,TEST  FOR TESTING
TM  SWITCHES,CONTBIT  IS THIS A CONTINUATION CARD?
BO  SKIPTYPE  YES
BAL  R14,KHNSCAN  SEARCH FOR JCL TYPE OPERATOR
LA  R1,1(R6,R7)  POINT PAST FIRST BLANK AFTER OP CODE
ST  R1,AVSP2  SAVE FOR POSSIBLE SHIFTING OF STMNT
BAL  RBAL,TEST  FOR TESTING
CLC  =C'EXEC',Ø(R6)  EXECUTE CARD?
BNE  SKIPTYPE  NO
MVI  SWITCHES,Ø  TURN OFF ALL OPTIONS
ZAP  NESTS,=P'Ø'  INITIALIZE (...) NESTING LEVEL
BAL  R14,KHNSCAN  SCAN FOR POTENTIAL PROGRAM NAME
BAL  RBAL,TEST  FOR TESTING
NI  SWITCHES,X'FF'-CONTBIT  TURN OFF CONTINUATION BIT

*  CLC  =C'PGM=DUO',Ø(R6)  DUO?
    CLC  =C'PGM=',Ø(R6)  ANY PROGRAM?
    BNE  NEXTSTEP  NO
    OI  SWITCHES,DUOBIT  TURN ON DUO SWITCH
    B  NEXTSTEP  GO CHECK FOR END OF STATEMENT

SKIPTYPE  NI  SWITCHES,X'FF'-CONTBIT  TURN OFF CONTINUATION BIT
CONTINUE  BAL  R14,KHNSCAN  SCAN FOR JCL GROUP
TM  SWITCHES,DATEBIT  MM/DD/YY EXPECTED?
BO  CHEKDATE  YES
BAL  RBAL,TEST  FOR TESTING
TM  SWITCHES,DUOBIT  DUO PROGRAM?
BZ  NEXTSTEP  NO
CLC  =C'PARM=',Ø(R6)  'PARM' GROUP?
BNE  NOTPARM  NO
OI  SWITCHES,PARMBIT  TURN ON PARM BIT
B  NEXTSTEP  GO GET PARAMETER FIELDS
NOTPARM TM SWITCHES,PARMBIT IN 'PARM' GROUP?
BZ NEXTSTEP NO
TM SWITCHES,QUOTEBIT STILL IN PARM='...'?
BO PARMYET YES
CP NESTS,=P'0' STILL IN PARM=(...)?
BNZ PARMYET YES
NI SWITCHES,X'FF'-PARMBIT TURN OFF PARM BIT

NEXTSTEP LA R1,(R6,R7) POINT PAST END OF FIELD
TM SWITCHES,QUOTEBIT WITHIN QUOTATION?
BO CONTINUE YES
CLI 0(R1),C' '
BE NOTJCL YES
B CONTINUE GO GET NEXT PARAMETER

PARMYET CLC =C'DATE=',Ø(R6) DATE SUB PARAMETER?
BNE NEXTSTEP NO
OI SWITCHES,DATEBIT TURN ON DATE BIT
BAL R14,KHNSCAN GET DATE
TM OPTIONS,DIAGBIT DIAGNOSE?
BZ CHEKDATE NO
BAL RBAL,TEST PRINT DIAGNOSTIC

CHEKDATE NI SWITCHES,X'FF'-DATEBIT TURN OFF DATE BIT
CH R7,=H'7' IS IT OF THE FORM 'MM/DD/YY'?
BE DATEIS8 NO
CH 7,=H'8' LENGTH 9?
BNE MAYBE1O NO
CLI 8(R6),C'/ ' XXXXXXX/?
BNE NEXTSTEP NO

DATEIS8 CLI 2(R6),C'/ ' FIRST AND SECOND FIELDS SEPARATED?
BNE NEXTSTEP NO

MAYBMDCY CLI 5(R6),C'/ ' SECOND AND THIRD FIELDS SEPARATED?
BNE NEXTSTEP NO
B LISTIT GO CHECK LIST OPTION

MAYBE1O CH 7,=H'9' LENGTH 10?
BE DATEIS1O NO
CH 7,=H'10' LENGTH 11?
BNE NEXTSTEP NO
CLI 1O(R6),C'/ ' XXXXXXXXXX/?
BNE NEXTSTEP NO

DATEIS1O CLI 2(R6),C'/ ' MM/DD/CCYY?
BE MAYBMDCY NO
CLI 4(R6),C'/ ' CCYY/MM/DD?
BNE NEXTSTEP NO
CLI 7(R6),C'/ ' BNE NEXTSTEP NO

LISTIT TM OPTIONS,LISTBIT LIST OPTION?
BO NOTBFORE YES (IE, DONE)
TM OPTIONS,DIAGBIT+BFOREBIT DIAGNOSE OR BEFORE OPTIONS?
BNM NOTBFORE BOTH (IE, DONE) OR NEITHER
TM OPTIONS,BFOREBIT BEFORE OPTION?
BZ NOTBFORE NO
L R1,SAVE5TO9+8 LOAD ADDRESS OF CARD IMAGE
MVC INAREA,0(R1) MOVE TO PRINT LINE  
BAL RBAL,PRINT PRINT BEFORE REPLACING DATE  
NOTBFOR L R1,INRECLOC LOAD ADDRESS OF CARD IMAGE  
CLM R7,1,DLENGTH IS NEW DATE SAME SIZE?  
BE SAMESIZE YES  
BH NEWISLT NO (NEW < OLD)  

*************************************************************  
IN THE CASE OF A 10 CHARACTER DATE FIELD REPLACING A 8 CHARACTER  
DATE FIELD, AN ATTEMPT IS MADE TO FIND THE EXTRA 2 CHARACTERS ON  
The STATEMENT. LOGIC IS AS FOLLOWS:  
*************************************************************  

<table>
<thead>
<tr>
<th>CASE</th>
<th>SPACES AVAILABLE</th>
<th>ACTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BETWEEN</td>
<td>BETWEEN</td>
</tr>
<tr>
<td></td>
<td>LABEL &amp;</td>
<td>OP &amp;</td>
</tr>
<tr>
<td></td>
<td>OP</td>
<td>OPERAND</td>
</tr>
</tbody>
</table>

CASE 1A  
2 | GETWEEN | BETWEEN | AT END |  
LABEL & | OP | STATEMENT |  
SHIFT LEFT 2SP BETWEEN LABEL*  
AND DATE=  
CASE 1B  
<2 | <2 | BETWEEN | BETWEEN |  
LABEL & | OP | STATEMENT |  
SHIFT LEFT 2SP BETWEEN OP  
AND DATE=  
CASE 2  
<2 | <2 | <2 | 2 |  
SHIFT RIGHT 2SP AFTER  
MM/DD/YY AND END OF STMNT  
CASE 3  
1 | 1 |  
SHIFT LEFT 1SP AFTER LABEL *  
& 1SP AFTER OP  
CASE 4  
1 | Ø | 1 |  
SHIFT LEFT 1 SP AFTER LABEL  
& RIGHT 1 SP AFTER MM/DD/YY  
CASE 5  
Ø | 1 | 1 |  
SHIFT LEFT 1 SP AFTER OP  
& RIGHT 1 SP AFTER MM/DD/YY  
CASE 6  
1 | Ø | Ø |  
SAME AS CASE 4  
CASE 7  
Ø | 1 | Ø |  
SAME AS CASE 5  
CASE 8  
Ø | Ø | Ø OR 1 |  
SAME AS CASE 2  

*************************************************************  

L R2,AVSPI LOAD ADDRESS OF POSSIBLE AVLBL SPACE  
CLC Ø(3,R2),=8C' ' TWO CHARACTERS AVAILABLE?  
BE CASE1 YES  

LR   R3,R2   SAVE FIRST POSSIBILITY
ICM  R2,15,AVSP2  IS THERE A SECOND?
BZ   NOSPACE2  NO
CLC  Ø(3,R2),=8C' '  TWO CHARACTERS AVAILABLE?
BE   CASE1  YES
CLC  Ѝ(2,R1),=8C' '  SPACE AVAILABLE AT END OF RECORD?
BE   CASE2  YES
CLC  Ø(2,R3),=8C' '  ONE CHARACTER IN SPACE1?
BNE  NOSPACE1  NO
CLC  Ø(2,R2),=8C' '  ONE CHARACTER IN SPACE2?
BE   CASE3  YES
SPACE1 CLI  71(R1),C' '  ONE SPACE AT END?
BE   CASE4  YES
B   CASE6  GO OVERLAY END OF RECORD
NOSPACE1 CLC  Ø(2,R2),=8C' '  ONE CHARACTER IN SPACE 2?
BNE  CASE8  NO
CLI  71(R2),C' '  ONE CHARACTER AT END?
BE   CASE5  YES
B   CASE7  NO
NOSPACE2 CLC  Ø(2,R3),=8C' '  ONE CHARACTER IN SPACE 1?
BNE  CASE8  NO
B   SPACE1  GO CHECK IS SPACE AT END
CASE1  LA    R4,2  SHIFT OFFSET
BAL  RBAL,MOVELEFT  GO MOVE IMAGE LEFT 2 BYTES
B   SAMESIZE  GO CONTINUE
CASE8  OI    SWITCHES,ERRORBIT  FLAG THAT DATA LOST
CASE2  LA    R4,2  SHIFT OFFSET
BAL  RBAL,MOVERGHT  GO MOVE IMAGE RIGHT 2 BYTES
B   SAMESIZE  GO CONTINUE
CASE3  LA    R4,1  SHIFT OFFSET
LR   R0,R2  SAVE ADDRESS OF SPACE 2
LR   R2,R3  POINT TO SPACE 1
BAL  RBAL,MOVELEFT  MOVE IMAGE LEFT 1 BYTE
LR   R2,R0  POINT TO SECOND SPACE
BAL  RBAL,MOVELEFT  MOVE IMAGE LEFT 1 BYTE
B   SAMESIZE  GO CONTINUE
CASE6  OI    SWITCHES,ERRORBIT  FLAG THAT DATA LOST
CASE4  LA    R4,1  SHIFT OFFSET
LR   R2,R3  POINT TO SPACE 1
BAL  RBAL,MOVELEFT  MOVE IMAGE LEFT 1 BYTE (TO DATE)
BAL  RBAL,MOVERGHT  MOVE IMAGE RIGHT 2 BYTES (FROM DATE)
B   SAMESIZE  GO CONTINUE
CASE7  OI    SWITCHES,ERRORBIT  FLAG THAT DATA LOST
CASE5  LA    R4,1  SHIFT OFFSET
BAL  RBAL,MOVELEFT  MOVE IMAGE LEFT 1 BYTE (TO DATE)
BAL  RBAL,MOVERGHT  MOVE IMAGE RIGHT 2 BYTES (FROM DATE)
B   SAMESIZE  GO CONTINUE
NEWISLT LR   R2,R6  POINT TO CURRENT LOCATION
L   R1,INRECLOC  POINT TO BEGINNING OF RECORD
LA   R6,71(R1)  POINT TO END OF RECORD
LA   R4,2  NUMBER OF BYTES TO SHIFT
**BAL** **RBAL**, **MOVELEFT**  
**LR** **R6,R2**  
**SR** **R7,R4**  
**SR** **R8,R4**  
**SAMESIZE IC** **R2,DLENGTH**  
**EX** **R2,SEMTNEWDT**  
**TM** **SWITCHES,ERRORBIT**  
**BZ** **DATEFIT**  
**BZ** **DATEFIT**  
**EJECT**  

*** CONVERT JULIAN DATE TO GREGORIAN DATE ***

**JULGREG**  
**ST** **RBAL,SAVJGBAL**  
**CLI** **JGYDDD,1**  
**BH** **JGACTUAL**  
**TR** **JGYDDD(1),=X'1920'**  
**JGACTUAL**  
**ZAP** **JGDAYS,JGYDDD+2(2)**  
**ZAP** **JGMONTHS,=P'1'**  
**LA** **R15,JANUARY**  
**LA** **R8,L'JANUARY**  
**LA** **R1,DECEMBER**  
**ZAP** **FEBRUARY,=P'28'**  
**CLC** **=X'2000',JGYDDD**  
**BE** **JGYR2000**  
**JG20THCN**  
**TM** **JGYDDD+1,1**  
**BO** **JGLOOP**  
**TM** **JGYDDD+1,X'12'**  
**BM** **JGLOOP**  
**JGLOOP**  
**AP** **JGDAYS,0(L'JANUARY,R15)**  
**BNH** **JGFOUND**  
**AP** **JGMONTHS,=P'1'**  
**SP** **JGDAYS,0(L'JANUARY,R15)**  
**BXLE** **R15,R8,JGLOOP**  
**JGFOUND**  
**UNPK** **JGMDDYY(2),JGMONTHS**  
**UNPK** **JGMDDYY+3(2),JGDAYS**
UNPK JGMMDDYY+6(3),JGYYDDD+1(2) UNPACK YEAR
MVI JGMMDDYY+2,C'/' SEPARATE MONTH AND DAY
MVI JGMMDDYY+5,C'/' SEPARATE DAY AND YEAR
OI JGMMDDYY+1,C'Ø' FORCE MONTH NUMERIC
OI JGMMDDYY+4,C'Ø' FORCE DAY NUMERIC
OI JGMMDDYY+7,C'Ø' FORCE YEAR NUMERIC
UNPK JGMDCY+6(5),JGYYDDD(3) UNPACK CCYY
MVC JGMDCY(6),JGMMDDYY SET MM/DD/
JGRETURN L RBAL,SAVJGBAL LOAD LINKAGE REGISTER
BR RBAL RETURN
EJECT

***********************************************************************
***      PRINT ROUTINE                                              ***
***********************************************************************
PRINT PUT PRINTER,LINE PRINT LINE
MVI LINE,C' ' SET SEED
MVC LINE+1(L'LINE),LINE CLEAR LINE
DOUBLES P BCTR R9, RBAL RETURN IF PAGE NOT FULL
HEADPAGE MVC PAGENO,=X'40202120' SET EDIT PATTERN
ED PAGENO,PAGES FORMAT PAGE NUMBER
AP PAGES,=P'1' INCREMENT PAGE COUNT
PUT PRINTER,HEADER PRINT PAGE HEADING
LA R9,56 SET LINES/PAGE
MVI LINE,C'Ø' SET TO DOUBLE SPACE AFTER HEADER
BR RBAL RETURN
EJECT

***********************************************************************
*** THIS IS AN INTERNAL SUBROUTINE TO SCAN CHARACTER STRINGS FOR    ***
*** 'WORDS' (IE, ALPHAMERIC SUBSTRINGS). RETURNED FIELDS ARE        ***
*** NON-BLANK CHARACTER STRINGS THAT ARE CONCATENATED BY AT         ***
*** LEAST ONE BLANK OR NON-ALPHAMERIC CHARACTER.                    ***
*---------------------------------------------------------------------*
*** TO REDUCE INSTRUCTION PATH LENGTH IT NEITHER SAVES              ***
*** REGISTERS NOR USES CONVENTIONAL CALLING SEQUENCE.               ***
*---------------------------------------------------------------------*
*** USAGE:                                                          ***
***                                                                 ***
*** 1)  TO SCAN FOR FIELD SEPARATED BY ' ', ',', '', '(', ')'       ***
***                                                                 ***
***      LA R4,NULL LOAD ADDRESS OF EOB RETURN ***
***      BAL R14,KHNSCAN SCAN FOR NEXT INPUT FIELD ***
***                                                                 ***
*** 2)  TO VALIDATE NUMERIC FIELDS:                                ***
***                                                                 ***
***      LA R4,ERROR LOAD ADDRESS OF NON-NUMERIC RETURN *
***      BAL R14,NUMTEST CHECK FIELD FOR NUMERIC DATA ***
*---------------------------------------------------------------------*
*  REGISTER USAGE:                                                   *
* * 1) FOR KHNSCAN, CONTENTS OF REGISTER 1 IS USED AS *
* A WORK REGISTER AND IS NOT RESTORED.  *
2) On entry to KHN1scan and NUMTEST, the following assumptions are made:
   Register 6 contains the address of the current field; Register 7, the length - 1 of that field; Register 8, the remaining length of the TIOA.

3) On return, KHN1scan and KHN1scan, Registers 6-8 are set to those values defined in "2)".

4) For numeric fields, NUMTEST packs the field into 'PACKWORK'.
   Else, this field is initialized to zero.

--------------------
KHN1scan

MVC TRTAB,TRTAB-1   set table to non-zero
MVI TRTAB+C' ',Ø   clear blank position
XR R1,R1           clear register (high order byte)
LA R6,1(R6,R7)     point past last field

PRESCAN

CLI Ø(R6),C'='     equal sign?
BE SPECIAL       yes
CLI Ø(R6),C'+'+    plus sign?
BNE NOTPLUS       no
MVI SIGN,X'C'     set sign
B SPECIAL         go adjust position and length

NOTPLUS

CLI Ø(R6),C'-'-    minus sign?
BNE NOTMINUS      no
MVI SIGN,X'D'     set sign
B SPECIAL         go adjust position and length

NOTMINUS

CLI Ø(R6),C'('     open parentheses?
BNE NOTLEFT       no
AP NESTS,=P'1'    increment nesting count
B SPECIAL         go adjust position and length

NOTLEFT

CLI Ø(R6),C')'     right parenthesis?
BNE NOTRIGHT      no
SP NESTS,=P'1'    decrement nesting count
B SPECIAL         go adjust position and length

NOTRIGHT

CLI Ø(R6),C'''     was field followed by a quote?
BNE NOTQUOTE      no
XI SWITCHES,QUOTEBIT flip quote bit
B SPECIAL         go adjust position and length

NOTQUOTE

CLI Ø(R6),C','     is current position a comma?
BNE NONSPCL       no
CLI 1(R6),C'       designates continuation?
BNE SPECIAL       no
TM SWITCHES,QUOTEBIT inside a quotation?
B SPECIAL         yes
OI SWITCHES,CONTBIT no, ie, a continuation indication
BR 4              give null return (bypass any comments)

SPECIAL

LA R6,1(R6)       skip past special character
BCTR R8,Ø         decrement length
LTR R8,R8         end of card?
BMR R4            yes
B PRESCAN         go process next character

NONSPCL

EX R8,TRT       search for first non-blank
BZR R4          exit if not found
LR R7,R1       address of first non-blank
SR    R7,R6                   DEDUCT ADDRESS OF LAST POSITION
SR    R8,R7                   SUBTRACT LENGTH FROM TOTAL LENGTH
BMR   R4                       EXIT IF NEGATIVE
LR    R6,R1                   POINT TO FIRST NON BLANK
CLI   Ø(R6),C'****'           QUOTATION AT BEGINNING?
BE    PRESCAN                 YES, RECYCULATE
CLI   Ø(6),C'('               OPEN PAREN AT BEGINNING?
BE    PRESCAN                 YES, RECYCULATE
CLI   Ø(6),C','                NULL FIELD AT BEGINNING?
BE    PRESCAN                 YES, RECYCULATE
CLI   Ø(6),C'++'              UNARY PLUS SIGN AT BEGINNING?
BE    PRESCAN                 YES, RECYCULATE
CLI   Ø(6),C'-'                UNARY MINUS SIGN AT BEGINNING?
BE    PRESCAN                 YES, RECYCULATE
Xc    TRTAB,TRTAB              SET TABLE TO ZEROES
MVI   TRTAB+C' ',C' '         TURN ON BLANK POSITION
MVI   TRTAB+C',',C','         TURN ON COMMA POSITION
MVI   TRTAB+C''''',C'''''      TURN ON C'''' POSITION
MVI   TRTAB+C'(','C'('        TURN ON C(' POSITION
MVI   TRTAB+C')',C')'        TURN ON C')' POSITION
MVI   TRTAB+C'=',C'='         TURN ON C=' POSITION
MVI   TRTAB+C'+',C'+'        TURN ON C+' POSITION
MVI   TRTAB+C'-'',C'-'       TURN ON C'-' POSITION
LR    R15,R8                  SAVE CURRENT LENGTH
LR    RØ,R6                   SAVE CURRENT LOCATION
LASTSCAN EX    R8,TRT          SEARCH FOR FIRST BLANK
BZ    NOHITS                  IF NO OBJECTS FOUND
TM    SWITCHES,QUOTEBIT       WITHIN QUOTATION?
BZ    SCANHIT                 NO
CLC   =C''''''',Ø(1)          IMBEDDED QUOTES?
BNE   SCANHIT                 NO
LA    R1,2(R1)                STEP OVER IMBEDDED QUOTES
AR    R8,R6                   ADJUSTED LENGTH=PREVIOUS LENGTH
SR    R8,R1                   +(PREVIOUS-CURRENT)LOCATION
LR    R6,R1                   RESET CURRENT POSITION
BP    LASTSCAN                IF POSITIVE LENGTH, CONTINUE SCAN
LR    R6,RØ                   RESTORE ORIGINAL LOCATION
LR    R8,R15                  RESTORE ORIGINAL LENGTH
B    SCANHIT                  GO PROCESS
NOHITS LR    R6,RØ             RESTORE ORIGINAL LOCATION
LR    R8,R15                  RESTORE ORIGINAL LENGTH
LA    R1,Ø(R6,R8)             POINT TO END OF INPUT
SCANHIT LR    R7,R1            LOAD ADDRESS OF BLANK
SR    R7,R6                   SUBTRACT ADDRESS OF FIRST NON-BLANK
BCR   13,R4                   NULL IF NOT POSITIVE
SR    R8,R7                   DEDUCT FROM TOTAL LENGTH
BCTR  R7,R14                  RETURN
BR    R14                     RETURN
TRT    TRT Ø(*-*,R6),TRTAB
TESTNUM TRT Ø(*-*,R6),TRTAB+16
NUMTEST MVC TRTAB,TRTAB-1     FILL WITH NON ZEROES
**EX** R7,TESTNUM IS FIELD NUMERIC?
**BCR** 7, R4 NO
**EX** R7,PACK PACK FIELD
**NI** PACKWORK+L'PACKWORK-1,X'F0' MASK SIGN BITS OFF
**OC** PACKWORK+L'PACKWORK-1(1),SIGN TURN SIGN BITS ON
**BR** R14 RETURN
PACK PACK PACKWORK,Ø(*-*),6)
EJECT

******************************************************
*** THIS IS A ROUTINE THAT PRINTS DIAGNOSTIC DATA IF 'DIAG' ***
*** OPTION IS SPECIFIED. IT IS USED PRIMARILY IN TESTING ***
*** CHANGES TO THE PROGRAM OR IN DIAGNOSING ANY PROBLEMS WITH ***
*** SPECIFIC DATA. ***
******************************************************

**TEST** TM OPTIONS,DIAGBIT DIAGNOSE BIT ON?
**BR** RBAL NO
**TESTX** ST RBAL,SAVTSBAL SAVE LINKAGE REGISTER
**UNPK** TESTOPTS(3),OPTIONS(2) UNPACK NYBLS OF BIT SWTCHS
**UNPK** TESTSWTS(3),SWITCHES(2) UNPACK NYBLS OF BIT SWTCHS
**ST** R7,DOUBLE SAVE LENGTH
**UNPK** TESTLEN(5),DOUBLE(5) UNPACK NYBLS OF LENGTH
**UNPK** TESTLOC(5),DOUBLE(5) UNPACK NYBLS OF ADDRESS
**NC** TESTOPTS(15),=15X'F' TURN OFF ZONE BITS
**TR** TESTOPTS(15),=C'Ø123456789ABCDEF' CONVERT TO HEX DSPLY
**MVI** TESTSWTS+2,C' ' SET SEPARATOR
**MVI** TESTOPTS+2,C' ' SET SEPARATOR
**MVI** TESTLEN+4,C' ' "
**MVI** TESTLOC+4,C' ' "
**BAL** RBAL,PRINT GO PRINT DIAGNOSTIC LINE
**TSRETURN** L RBAL,SAVTSBAL RESTORE LINKAGE REGISTER
**BR** RBAL RETURN
**TSMOVE** MVC INAREA(*-*),Ø(6)
EJECT

******************************************************
*** MOVE DATA LEFT BY AN OFFSET SPECIFIED IN REGISTER R4. R6 ***
*** POINTS TO CURRENT POSITION. R2 SPECIFIES THE DESTINATION ***
*** OF THE MOVE. ***
******************************************************

**MOVELEFT** ST RBAL,SAVMLBAL SAVE LINKAGE REGISTER
**LA** R14,Ø(R2,R4) DESTINATION + OFFSET
**LR** R15,R6 POINT TO CURRENT POSITION
**SR** R15,R14 LENGTH OF MOVE
**BCTR** R15,Ø LENGTH-1
**EX** R15,MLMOVE
**SR** R6,R4 ADJUST CURRENT POSITION
**AR** R7,R4 ADJUST SCAN LENGTH
**L** RBAL,SAVMLBAL RESTORE LINKAGE REGISTER
**BR** RBAL RETURN
**MLMOVE** MVC Ø(*-*),R(R14)
EJECT

******************************************************
*** MOVE DATA RIGHT ***

MoVerght St RBal,SAvMRBal Save linkage register
   La R0,.71(R1) Point to end of data
   Lr R15,R0 Save
   Sr R0,R6 Subtract current location
   Sr R0,R7 Less current length
   BnP MrReturn Exit if not positive byte count
   Lr R14,R15 Point at end of data
   Sr R14,R4 Point to byte to be moved to end
MrLoop Mvc Ø(1,R15),Ø(R14) Move date right
   BcTr R14,Ø Decrement source register
   BcTr R15,Ø Decrement target register
   BcT R0,Mrloop Continue
   Ar R7,R4 Increase field length by offset
MrReturn L RBal,SAvMRBal Restore linkage register
   Br RBal Return
Ltorg

OcuRus DC C'CONTAINS'
OcuR1 DC X'4020420204204202120'
   DC C' RECORDS OF WHICH'
OcuR2 DC X'4020420204204202120'
   DC C' CONTAIN DATE= PARM OCCURRENCES'
Loocurs EQU *-OcuRus
OcuRpat DC X'402020202120'
Editpat EQU ocuRpat
Eject

***********************************************************************
*** SPECIAL INITIALIZING ROUTINE TO CONSERVE BASE REGISTER ***
*** ADDRESSING PAGE ***
***********************************************************************

IniTial St RBal,SAvILBal Save linkage register
   La R11,2048(RBase) Load Rbase + half page
   La R11,2048(R11) Load Rbase + full page
   UsIng &myName,Rbase,R11 Addressability
   Mvi Format,3 Assume MM/DD/CCYY
   Mvi Dlength,9 Assume length of 10
   ZAP Nests,=P'Ø' Initialize '(',')' nesting count
   Mvi Trtab-1,X'FF' Initialize non-zero prefix
   Mvc Trtab+L'Trtab(10),=10X'FF' Set non-blank for num test
   Mvc Thruname,=19x'FF' Set initial 'thru' member name
   xc Fromname,Fromdate "'from' member name
   Mvi Dflag,Ø Initialize flag
   ZAP Finds,=P'Ø' Initialize string found count
   ZAP Members,=P'Ø' Initialize members in pds
   ZAP Modified,=P'Ø' Initialize modified members
   ZAP Excluded,=P'Ø' Initialize excluded members
   ZAP Records,=P'Ø' Initialize records in 1st member
   ZAP TreCs,=P'Ø' Initialize records in all member
   ZAP Tfinds,=P'Ø' Initialize modifies in all members
   ZAP Errortot,=P'Ø' Initialize modifies in all members
MVC JGMOTBL(13*L'JGMOTBL),JGMOTBLD COPY JULGREG DAYS/MONTH
BAL RBAL,GETPARMS GET PARMS

* BEGIN DCB INITIALIZATION

MVC PRINTER(PRINTERL),PRINTERD INITIALIZE DCB
MVC PDSDIR(PDSDIRL),PDSDIRD INITIALIZE PDSDIR DCB
MVC PDS(PDSL),PDSD INITIALIZE PDS DCB
MVC CARDS(CARDSL),CARDSD INITIALIZE CARDS DCB
MVC ERRORS(ERRORSL),ERRORSD INITIALIZE ERRORS DCB

* END DCB INITIALIZATION

* BEGIN DCB OPENS

MVC PROPENL(PROPENLN),OPEND INITIALIZE SET PRINTER OPEN LIST
OPEN (PRINTER,(OUTPUT)),MF=(E,PROPENL) OPEN PRINTER
MVC DROPNL(DROPNLN),OPEND SET PDSDIR OPEN LIST
OPEN (PDSDIR,(INPUT)),MF=(E,DROPNL) OPEN PDSDIR
MVC PDOPENL(PDOPENLN),OPEND SET PDS OPEN LIST
OPEN (PDS,(UPDAT)),MF=(E,PDOPENL) OPEN PDS
MVC EROPNL(EROPNLN),OPEND SET ERRORS OPEN LIST
OPEN (ERRORS,(OUTPUT)),MF=(E,EROPENL) OPEN ERRORS
MVC PRCLOS(L(PRCLOSLN),CLOSED INITIALIZE CLOSE LIST
MVC DRCLOS(L(DRCLOSLN),CLOSED SET PDSDIR CLOSE LIST
MVC PDCLOS(L(PDCLOSLN),CLOSED SET PDS CLOSE LIST
MVC ERCLOS(L(ERCLOSLN),CLOSED SET ERRORS CLOSE LIST

LA R3,PDS GET ADDRESS OF PDS DCB
 USING IHADCB,R3 ESTABLISH ADDRESSABILITY
LH R5,DCBLRECL LOAD RECORD LENGTH
STH R5,INLRECL SAVE
LH R3,DCBLKSIZ LOAD MAXIMUM BLOCK SIZE
STH R3,INBLKSIZ SAVE
LA R3,100(R3) ADD PAD
DROP R3 DROP ADDRESSABILITY
GETMAIN R,LV=(R3) GET WORK AREA FOR INPUT BLOCKS
ST R1,BLOCKLOC SAVE ADDRESS
MVC CDOPENL(CDOPENLN),OPEND SET CARDS OPEN LIST
OPEN (CARDS,(INPUT)),MF=(E,CDOPENL) OPEN CARDS

* END DCB OPENS

TIME
ST R1,JGYYDDD SAVE JULIAN DATE
ST R1,TODAY SAVE FORM PARM DATA
BAL RBAL,JULGREG CONVERT TO MM/YY/DD
MVC HEADER(L'HEAD),HEAD INITIALIZE HEADER
MVC HEADER=L'HEADER+L'HEAD),HEADER=L'HEAD-1 CLEAR
MVC HEADER=L'HEADER-8(4),=C'PAGE' SET PAGE NUMBER ID
ZAP PAGES,=P'1' INITIALIZE PAGE COUNT
MVC DDNAME,PSSDDN MOVE SELECTION FILE NAMES
BAL RBAL,GETNAMES PUT JOB/DSN NAMES IN HEADER
MVC HEADDATE,JGMDCY MOVE MM/DD/CCYY TO HEADING
BAL RBAL,HEADPAGE PRINT PAGE HEADER
MVC DECBALN),DECB INITIALIZE DECB
LA R3,PDS GET ADDRESS OF PDS DCB
 USING IHADCB,R3 ESTABLISH ADDRESSABILITY
LH R5,DCBLRECL  LOAD RECORD LENGTH
STH R5,INLRECL  SAVE
LH R3,DCBBLSZ   LOAD MAXIMUM BLOCK SIZE
STH R3,INBLKSZ  SAVE
LA R3,100(R3)   ADD PAD
DROP R3         DROP ADDRESSABILITY
GETMAIN R,LV=(R3) GET WORK AREA FOR INPUT BLOCKS
ST R1,BLOCKLOC SAVE ADDRESS
LA R3,EXCLUDES  POINT TO FIRST ELEMENT
LA R4,EXCLUDEX-EXCLUDES(R3) POINT TO LAST EXCLUDE
ST R3,EXCLUDE1  SAVE BEGINNING ADDRESS
MVC LINE(27),=C'ØMANUALLY EXCLUDED MEMBERS:'
BAL RBAL,DOUBLESP ALLOW FOR DOUBLE SPACE
BAL RBAL,PRINT PRINT EXCLUSION SUBHEADER
MVI LINE,C'Ø'   SET TO DOUBLE SPACE
BAL RBAL,DOUBLESP ALLOW FOR DOUBLE SPACE
ILCDLOOP GET CARDS,CARDAREA READ EXCLUSION CARD
MVC Ø(L'EXCLUDES,R3),CARDAREA MOVE MEMBER NAME TO EXCL TABLE
LA R3,L'EXCLUDES(R3) POINT TO NEXT ENTRY
CR R3,R4        PAST END OF SAVE AREA?
BL ILCDLOOP      NO
CARDEOF MVC CDCLOSL(CDCLOSLN),CLOSED SET CARDS CLOSE LIST
CLOSE (CARDS),MF=(E,CDCLOSL) CLOSE CARDS
MVC Ø(L'EXCLUDES,R3),=8X'FF' SET HIGH VALUES
ST R3,EXCLUDE2  SAVE LAST CARD IMAGE
C R3,EXCLUDE1   ANY EXCLUSIONS?
BNE ILSORT     NO
MVC LINE+5(L),=C'* NONE *' INDICATE NO EXCLUSIONS
BAL RBAL,PRINT PRINT INDICATION
B ILEXIT       GO EXIT
ILSORT L R3,EXCLUDE1 LOAD START OF LIST
ILSORTL2 LA R4,L'EXCLUDES(R3) POINT TO NEXT ELEMENT OF VECTOR
C R4,EXCLUDE2   AT END OF VECTOR?
BE ILSORTX2    YES (BUT PRINT LAST ENTRY)
BH ILEXIT       YES
ILSORTL1 CLC Ø(L'EXCLUDES,R4),Ø(R3) CURRENT ENTRY LOWER?
BH ILSORTX1    NO
XC Ø(L'EXCLUDES,R3),Ø(R4) SWAP
XC Ø(L'EXCLUDES,R4),Ø(R3) . VECTOR
XC Ø(L'EXCLUDES,R3),Ø(R4) . ELEMENTS
ILSORTX1 LA R4,L'EXCLUDES(R4) POINT TO NEXT ENTRY
C R4,EXCLUDE2   AT END OF LIST?
BL ILSORTL1    NO
ILSORTX2 MVC LINE+5(L'EXCLUDES),Ø(R3) MOVED SORTED ENTRY
BAL RBAL,PRINT PRINT ENTRY
LA R3,L'EXCLUDES(R3) POINT TO NEXT ENTRY
B ILSORTL2     CONTINUE
ILEXIT MVI LINE,C'Ø'   SET TO DOUBLE SPACE
BAL RBAL,DOUBLESP ALLOW FOR DOUBLE SPACE
L RBAL,SAVILBAL RESTORE LINKAGE REGISTER
BR RBAL        RETURN
GETNAMES ST RBAL,SAVGNBAL SAVE LINKAGE REGISTER
XR R15,R15 ADDRESS OF PSA
USING PSA,R15 ESTABLISH ADDRESSABILITY
L R14,FLCCVT ADDRESS OF CVT
DROP R15 DROP ADDRESSABILITY TO PSA
USING CVTMAP,R14 ESTABLISH ADDRESSABILITY TO CVT
L R15,CVTTCBP ADDRESS OF CURRENT TCB
DROP R14 DROP ADDRESSABILITY TO CVT
USING TCB,R15 ESTABLISH ADDRESSABILITY CURRENT TCB
L R14,TCBTIO ADDRESS OF TIOT
USING TIOT,R14 ESTABLISH ADDRESSABILITY TO TIOT
MVC HEADJOBN,TIOCNJOB MOVE JOB NAME TO HEADER
MVC HEADJOBN-4(4),=C'JOB=' SET JOBNAME ID
DROP R15 DROP ADDRESSABILITY TO TCB
LA R15,TIOELNGH ADDRESS OF FIRST TIOT ENTRY
USING TIOENTRY,R15 ESTABLISH ADDRESSABILITY TO TIOT
GNTIOTLP CLI TIOELNGH,X'ØØ' END OF TIOT CHAIN?
BE GNRETURN YES (SHOULDN'T HAPPEN)
CLC TIOEDDNM(8),DDNAME PDS NAME FOUND?
BE GNDSN YES
XR RØ,RØ CLEAR REGISTER
IC RØ,TIOELNGH INSERT ENTRY LENGTH
AR R15,RØ POINT TO NEXT ENTRY
B GNTIOTLP CONTINUE
GNDSN XR R1,R1 CLEAR REGISTER
ICM R1,7,TIOEJFCB ADDRESS OF JFCB
USING JFCB,R1 ESTABLISH ADDRESSABILITY TO JFCB
MVC HEADDRN,JFCBDSNM MOVE DSNAME TO HEADER
MVC HEADDRN-4(4),=C'DSN=' SET DSN ID IN HEADER
DROP R1,R15 DROP ADDRESSING TO JFCB,TIOT,ENTRY
GNRETURN L RBAL,SAVGNBAL RESTORE LINKAGE REGISTER
BR RBAL RETURN
EJECT

GETPARMS ST RBAL,SAVGPBAL SAVE LINKAGE REGISTER
L R6,R1SAVE GET ADDRESS OF AREA
L R6,Ø(R6) GET ADDRESS OF PARM= DATA
LH R8,Ø(R6) LOAD LENGTH OF PARM
LA R6,1(R6) POINT TO BYTE PRECEEDING INFO FIELD
XR R7,R7 CLEAR INITIAL LENGTH
PARMLOOP LA R4,PARMEND  POINT TO NULL RETURN
BAL R14,KHNSCAN   GET PARAMETER
BAL RBAL,TEST    FOR TESTING
NOTSUBPM LA R4,PARMERR  POINT TO NULL RETURN
CLC =C'DATE=',Ø(R6) 'DATE' OPTION?
BE SETDATE      YES
CLC =C'FMT=',Ø(R6) 'DATE' OPTION?
BE SETFMT       YES
CLC =C'PRNT=',Ø(R6) 'PRINT' OPTION?
BE SETPRINT     YES
CLC =C'FROM=',Ø(R6) 'FROM' OPTION?
BE SETFROM      YES
CLC =C'THRU=',Ø(R6) 'THRU' OPTION?
BE SETTHRU      YES
CLC =C'CTRL=',Ø(R6) 'CONTROL' OPTION?
BNE PARMERR     NO
MVI SIGN,X'F' INITIALIZE SIGN (NOT REALLY NECESRY)
BAL R14,KHNSCAN GO GET CONTROL VALUE
CLI Ø(R6),C'X' OVERRIDE?
BE OVERRIDE     YES
BAL RBAL,TEST   FOR TESTING
BAL R14,NUMTEST VERIFY THAT IT'S NUMERIC
ZAP CONTROL,PACKWORK SET VALUE
B PARMLOOP CONTINUE PARAMETER SCAN
OVERRIDE MVI CONTROL,X'FF' SET OVERRIDE
B PARMLOOP CONTINUE PARAMETER SCAN
SETDATE BAL R14,KHNSCAN SCAN FOR MM/DD/YY,MM/DD/CCYY,ETC.
BAL RBAL,TEST FOR TESTING
CLC =C'TODAY',Ø(R6) CURRENT SYSTEM DATE?
BE SETTODAY     YES
CH R7,=H'7' IS FIELD 8 BYTES LONG?
BNE PARMD8OK    NO
CH R7,=H'9' IS FIELD 10 BYTES LONG?
BNE PARMERR     NO
*PARMD8OK ZAP TODAY,SAVEDAYS INITIALIZE DATE
PARMD8OK DS ØH
EX R7,CLCDATE SAME AS TODAY?
BE PARMLOOP YES
CLI FORMAT,FMTCYMD OF THE FORM CCYY/MM/DD?
BNL PARMCYMD YES
CLI 2(R6),C'/' SEPARATOR BETWEEN MONTH AND DAY?
BNE PARMERR NO
CLI 5(R6),C'/' SEPARATOR DAY AND YEAR OR CCYY?
BNE PARMERR NO
MVC MMDDYY,=X'0001030400000607' ASSUME MM/DD/YY
CLI FORMAT,FMTYMD IS ASSUMPTION CORRECT
BL PARMFOK YES
BE PARMYMD NO, YY/MM/DD
MVC MMDDYY,=X'0001030406070809' MUST BE MM/DD/CCYY
B PARMFOK GO CHECK VALIDITY
PARMYMD MVC MMDDYY,=X'0304060700000001' MUST BE YY/MM/DD
B PARMFOK Go check validity

PARMCYMD MVC MMDYY=X'05060809010203' Must be CCYY/MM/DD
CLI 4(R6),C'/' Separator between CCYY and month?
BNE PARMERR NO
CLI 7(R6),C'/' Separator month and day?
BNE PARMERR NO
B PARMFOK Go check validity

CLCDATE CLC DATE(*-*),Ø(6)
MVCDATE MVC NEWDATE(*-*),Ø(6)
TRTDATE TRT MMDYY(*-*),Ø(6)

PARMFOK TR MMDYY,Ø(R6) Gather MMDCCYY

MVC TRTAB,TRTAB-1 Make table non zero
XC TRTAB+C'Ø'(10),TRTAB+C'Ø' Turn off numeric portion
TRT MMDYY,TRTAB Is MMDYY numeric?
BNZ PARMERR NO
CLC =C'ØØ',MMDYY Is month okay?
BNL PARMERR NO
CLC =C'12',MMDYY NO
BL PARMERR NO
CLC =C'Ø0',MMDYY+2 Is day okay?
BNL PARMERR NO
PACK MONTHS,MMDYY+2 Pack month
PACK DAYS,MMDYY+2(2) Day

CLI FORMAT,FMTYMD Is century present?
BH PARMCOK YES

PARMCOK PACK YEARS,MMDYY+4(4) Year

ZAP DOUBLE,YEARS Move year to double word
CVB R0.DOUBLE Load into register
STC R0,LEAPFLAG Save binary low order byte
ZAP FEBRUARY,=P'28' Assemble not loop year
TM LEAPFLAG,3 Leap year?
BNZ NOTLEAP NO
ZAP FEBRUARY,=P'29' Set for leap year

NOTLEAP ZAP DOUBLE,MONTHS Move month to double word

CVB R1.DOUBLE Load into register
LR R15,R1 Save for below
MH R1,=AL2(L'JANUARY) * Table width
LA R1,JANUARY-L'JANUARY(R1) Index table
CP Ø(L'JANUARY,1),DAYS IS DATE TOO LARGE?
BL PARMERR YES

MVC NEWDATE,Ø(R6) Save date for overlay (see CKEKDATE)
LR R1,R0 Save year
BCTR R1,0 Last year
M R0,=F'365.25' 100*YEARS*DAYS/YEAR
D R0,100 DAYS FROM 1/0/00 TO 12/31/(YR-1)
CVD R1,DOUBLE Convert to decimal
ZAP TODAY,DOUBLE Save days

D A P V O D A Y S Add days from entry
LA R1,JANUARY-L'JANUARY Point to zero days

DATELOOP AP TODAY,Ø(L'JANUARY,R1) Add days in month
LA R1,'JANUARY(R1)  POINT TO NEXT MONTH
BCT R15,DATELOOP  ACCUMULATE DAYS IN PREVIOUS MONTHS
B PARMLOOP  GO GET NEXT PARAMETER
SETTODAY MVI CONTROL,X'FF'  FORCE NON-CHECK OF CONTROL
CLI 5(R6),C'+' DAYS AFTER TODAYS DATE?
BE TODAYP  YES
CLI 5(R6),C'-' DAYS BEFORE TODAYS DATE?
BE TODAYM  YES
MVC NEWDATE,DATE GET CURRENT DATE
B PARMLOOP  GO GET NEXT PARAMETER
TODAYP BAL R14,KHNSCAN GET NUMBER OF DAYS
BAL R14,NUMTEST CHECK IF NUMERIC
CP PACKWORK,=P'365' TOO MANY DAYS?
BH PARMERR  YES
AP TODAY,PACKWORK ADJUST JULIAN DATE
ZAP DAYS,=P'365' DAYS PER YEAR
TM LEAPFLAG,3 IS THIS A LEAP YEAR?
BNZ TODAYPNL  NO
AP DAYS,=P'1' ADJUST DAYS FOR LEAP YEAR
TODAYPNL CP TODAY+L'TODAY-2(2),DAYS BEFORE END OF YEAR?
BNH TODAYPTY  YES
SP TODAY,DAYS SUBTRACT DAYS IN CURRENT YEAR
AP TODAY,=P'1988' ADJUST TO NEXT YEAR
TODAYPTY BAL R1,JULGREG CONVERT TO MM/DD/YY
MVC NEWDATE,DATE SAVE SYSTEM DATE +/- N
B PARMLOOP  GO GET NEXT PARAMETER
TODAYM BAL R14,KHNSCAN GET NUMBER OF DAYS
BAL R14,NUMTEST CHECK IF NUMERIC
OI PACKWORK+L'PACKWORK-1,X'F' DISREGARD SIGN
CP PACKWORK,=P'365' TOO MANY DAYS?
BH PARMERR  YES
CP TODAY+L'TODAY-2(2),PACKWORK DATE IN THIS YEAR?
BH TODAYMTY  YES
ZAP DAYS,=P'365' DAYS PER YEAR
TM LEAPFLAG,3 WAS LAST YEAR A LEAP YEAR?
BNO TODAYMNL  NO
AP DAYS,=P'1' ADJUST DAYS FOR LEAP YEAR
TODAYMNL AP TODAY,DAYS JULIAN DATE FROM BEGINNING OF LST YR
SP TODAY,=P'1988' ADJUST TO NEXT YEAR
TODAYMTY SP TODAY,PACKWORK SUBTRACT NUMBER OF DAYS
B TODAYPTY  GO GET SYSTEM DATE - N
SETPRINT BAL R14,KHNSCAN GET PRINT OPTION
BAL RBAL,TEST FOR TESTING
CLC =C'DIAG',Ø(R6) DIAGNOSE OPTION?
BE DIAGNOSE  YES
CLC =C'BEFORE',Ø(R6) BEFORE OPTION?
BE BEFORE  YES
CLC =C'AFTER',Ø(R6) AFTER OPTION?
BE AFTER  YES
CLC =C'LIST',Ø(R6)
BNE NOTSUBPM  NO
OI OPTIONS, LISTBIT TURN ON OPTION
PRINT4 CLI 4(R6), C', ', ADDITIONAL PRINT OPTION?
BE SETPRINT YES
B PARMLOOP NO
DIAGNOSE OI OPTIONS, DIAGBIT TURN ON OPTION
B PRINT4 GO CHECK FOR ADDITIONAL PRNT OPTS
BEFORE OI OPTIONS, BFOREBIT TURN ON OPTION
CLI 6(R6), C', ', ADDITIONAL PRINT OPTION?
BE SETPRINT YES
B PARMLOOP NO
AFTER OI OPTIONS, AFTERBIT TURN ON OPTION
CLI 5(R6), C', ', ADDITIONAL PRINT OPTION?
BE SETPRINT YES
B PARMLOOP NO
SETFROM BAL R14, KHNSCAN GET PRINT OPTION
BAL RBAL, TEST FOR TESTING
MVC MEMBER, =8C " INITIALIZ NAME PADDING
EX R7, MOVENAME MOVE MEMBER NAME
OI OPTIONS, MEMBBIT SET OPTION BIT
MVC FROMNAME, MEMBER MOVE TO BEGINNING NAME
B PARMLOOP NO
SETTHRU BAL R14, KHNSCAN GET PRINT OPTION
BAL RBAL, TEST FOR TESTING
MVC MEMBER, =8C " INITIALIZ NAME PADDING
EX R7, MOVENAME MOVE MEMBER NAME
OI OPTIONS, MEMBBIT SET OPTION BIT
MVC THRNAME, MEMBER MOVE TO ENDING NAME
B PARMLOOP NO
MOVENAME MVC MEMBER(*-*) , Ø(6)
SETFMT BAL R14, KHNSCAN GET FORMAT
LA R1, FORMATS POINT TO VALID FORMATS
LA RØ, #FORMATS
FMTLOOP CLC Ø(L'FORMATS, R1), Ø(6) MATCH FOUND?
BE FORMATOK YES
LA R1, L'FORMATS+1(R1) POINT TO NEXT FORMAT
BCT RØ, FMTLOOP CONTINUE SEARCH
B PARMERR NO MATCH
FORMATOK STC RØ, FORMAT SET FORMAT
MVC DLENGTH, L'FORMATS(R1) SET LENGTH-1
B PARMLOOP CONTINUE
PARMEND CLI CONTROL, X'FF' OVERRODE?
PARMERR DS ØH FOR NOW
L RBAL, SAVGPBAL RESTORE LINKAGE REGISTER
BR RBAL RETURN
* END STUB DEFINE
EJECT
***********************************************************************
***      FIXED DATA AREA                                           ***
***********************************************************************
HEAD DC C'1YEAR2KSD -- SET 10 CHARACTER DATE '
* CHECK REFERENCES TO THE FOLLOWINT EQUATES IF CHANGES ARE MADE TO
* TABLE.
FMTCYMD EQU 4 FORMATS WITH CCYY AT BEGINNING >=
FMTYMD EQU 2 LOGIC USES THIS IN PARM/DATA ANALYSIS
FORMATS DC CL8'CCYY/MM/DD',X'9' 4
DC CL8'MM/DD/CCYY',X'9' 3
DC CL8'YY/MM/DD',X'7' 2
DC CL8'MM/DD/YY',X'7' 1
#FORMATS EQU (*-FORMATS)/(L'FORMATS+1)
OPEND OPEN (,),MF=L
CLOSED CLOSE (,),MF=L
READ DECB, SF, MF=L
* BEGIN DCB CONSTANTS
PRINTERD DCB DDNAME=PRINTER, DEVD=DA, DSORG=PS, LRECL=133, -
BLKSIZE=133, MACRF=(PM), RECFM=FBA
PDSDIRD DCB DDNAME=PDS, DSORG=PS, MACRF=GM, BLKSIZE=256, LRECL=256, -
EODAD=GDEND, RECFM=F
PDSD DCB DDNAME=PDS, DSORG=PO, MACRF=R, EODAD=GREOF
PDSDDN EQU PDSDIRD+DCBDDNAM-DCBRELAD
CARDSD DCB DDNAME=CARDS, DSORG=PS, MACRF=GM, EODAD=CARDEOF, -
RECFM=FB, LRECL=80
ERRORSD DCB DDNAME=ERRORS, DEVD=DA, DSORG=PS, LRECL=133, -
BLKSIZE=133, MACRF=(PM), RECFM=FBA
* END DCB CONSTANTS
JGmotbld DC PL2'0,31,28,31,30,31,30,31,31,30,31,30,31',
* END CONSTANTS
LTORG
EJECT
***********************************************************************
***      DSECT FOR MY SAVE AREA AND VARIABLES.                      ***
***********************************************************************
WORKD DSECT
MYSAVE DS 18F MY REGISTER SAVE AREA
COMPCODE DS F PROGRAM COMPLETION CODE
RETCDE DS F INTERNAL RETURN CODE
RISAVE DS F INITIAL VALUE IN R1
BLOCKLOC DS F
BLOCKEND DS F
INLRECL DS H
INBLKSSIZ DS H
INRECLOC DS F
TTRN DS F
PAGES DS PL2
HIT DS C
DFLAG DS C
NESTS DS PL2
ERRORTOT DS PL3
MEMBERS DS PL3
MODIFIED DS PL3
EXCLUDED DS PL3
RECORDS DS PL4
FINDS DS CL4
TRECS DS PL4
TFINDS DS PL4
MEMBER DS CL8
FROMNAME DS XL8'0'
THRUNAME DS XL8'FFFFFFFFFFFFFFFF'
NEWDATE DS CL10
OPTIONS DC X'0'
CHNGBIT EQU X'40'
DIAGBIT EQU X'20'
LISTBIT EQU X'10'
BFOREBIT EQU X'0B'
AFTERBIT EQU X'04'
MEMBRBIT EQU X'02'
SWITCHES DC X'0'
QUOTEBIT EQU X'80'
COMMABIT EQU X'40'
CONTBIT EQU X'20'
PARMBIT EQU X'10'
UPDATBIT EQU X'0B'
DUOBIT EQU X'04'
ERRORBIT EQU X'02'
DATEBIT EQU X'01'
DLENGTH DS X
FORMAT DS X'0'
* X'01' = MM/DD/YY
* X'02' = YY/MM/DD
* X'03' = MM/DD/CCYY
* X'04' = CCYY/MM/DD
CONTROL DC PL2'0'
LEAPFLAG DC X'0'
SIGN DC X'C'
DAYS DS PL2
MONTHS DS PL2
YEARS DS D
SAVEDAYS DS D
MMDDYY DS CL8
DATE DS C'MM/DD/CCYY'
TODAY DS F
AVSP1 DS A
AVSP2 DS A
SAVE5TO9 DS 5F
FROMDATE DS CL8
THRUDATE DS CL8
DDNAME DS CL8
DOUBLE DS D
PACKWORK DS PL16

* BEGIN STUB LINK SAVE
SAVGBAL DS A              BAL REGISTER SAVE AREA FOR GETDIR
SAVGNBAL DS A            BAL REGISTER SAVE AREA FOR GETNAMES
SAVGPBAL DS A           BAL REGISTER SAVE AREA FOR GETPARMS

SAVGRBAL DS A                  BAL REGISTER SAVE AREA FOR GETREC
SAVGSBAL DS A                  BAL REGISTER SAVE AREA FOR GETSTATS
SAVJGBAL DS A                  BAL REGISTER SAVE AREA FOR JUGREG
SAVILBAL DS A                  BAL REGISTER SAVE AREA FOR INITIAL
SAVMLBAL DS A                  BAL REGISTER SAVE AREA FOR MOVELEFT
SAVMRBAL DS A                  BAL REGISTER SAVE AREA FOR MOVELEFT
SAVPEBAL DS A                  BAL REGISTER SAVE AREA FOR PUTERR
SAVPBSBAL DS A                 BAL REGISTER SAVE AREA FOR PUTSTATS
SAVRDBAL DS A                  BAL REGISTER SAVE AREA FOR READDIR
SAVRSRBAL DS A                 BAL REGISTER SAVE AREA FOR SCANREC
SAVTSBAL DS A                  BAL REGISTER SAVE AREA FOR TEST
SAVWRRAL BAL A                 BAL REGISTER SAVE AREA FOR WRITEREC
* END STUB LINK SAVE
*
* BEGIN OPEN/CLOSE LIST
*  
DS ØD
PROPENL OPEN (,),MF=L
PROPENLN EQU *-PROPENL
PRCLOSL CLOSE (,),MF=L
PRCLOSLN EQU *-PRCLOSL
DROPENL OPEN (,),MF=L
DROPENLN EQU *-DROPENL
DRCLOSL CLOSE (,),MF=L
DRCLOSLN EQU *-DRCLOSL
PDOPENL OPEN (,),MF=L
PDOPENLN EQU *-PDOPENL
PDCLOSL CLOSE (,),MF=L
PDCLOSLN EQU *-PDCLOSL
CDOPENL OPEN (,),MF=L
CDOPENLN EQU *-CDOPENL
EOPENL OPEN (,),MF=L
EOPENLN EQU *-EOPENL
ERCLOSL CLOSE (,),MF=L
ERCLOSLN EQU *-ERCLOSL
* END OPEN/CLOSE LIST
BBLNTRY SMUMØØ2 DSECT=NO               BLDL FORMAT ENTRY
BLDLEN EQU *-BLDLNTRY                 LENGTH OF BLDL ENTRY
READ DECB,SF,MF=L                  DECB FOR PDS
DECBALN EQU *-DECB
* BEGIN DCB DSECTS
PRINTER DCB DDNAME=PRINTER,DEVD=DA,DSORG=PS,LRECL=133, -
           BLKSIZE=133,MACRF=(PM),RECFM=FBA
PRINTERL EQU *-PRINTER
PDSDIR DCB DDNAME=PDS,DSORG=PS,MACRF=GM,BLKSIZE=256,LRECL=256, -
            EODAD=GDEND,RECFM=F
PDSDIRL EQU *-PDSDIR
PDS DCB DDNAME=PDS,DSORG=PO,MACRF=R,EODAD=GREOF
PDSL EQU *-PDS
CARDS DCB DDNAME=CARDS,DSORG=PS,MACRF=GM,EODAD=CARDEOF,
RECFM=FB, LRECL=80

CARDSL EQU *-CARDS
ERRORS DCB DDNAME=ERRORS, DEVD=DA, DSORG=PS, LRECL=133, BLKSIZE=133, MACRF=(PM), RECFM=FBA

ERRORSL EQU *-ERRORS
* END DCB DSECTS
JGMOTBL DS PL2'0'
JANUARY DS P'31'
* M A M J J A S O N
FEBRUARY DS P'28, 31, 30, 31, 30, 31, 30, 31, 30'
DECEMBER DS P'31'
JGDAYS DS PL2
JGMONTHS DS PL2
JGMMDDYY DS C'MM/DD/YY'
JGMDCY DS C'MM/DD/CCYY', C
JGYYDDD DS F
* END DSECT INSERT
HEADER DS CL133
ORG HEADER+L'HEAD+10
HEADJOBN DS CL8, C' DSN='
HEADDSN DS CL44, 5C
HEADDRESS DS CL10
ORG HEADDRESS+L'HEAD-5
PAGENO DS CL4
ORG
LINE DS CL133
OUTAREA DC CL133'0'
ORG OUTAREA+2
MEMBERNO DS CL4, C
MEMBNAME DS CL8
CARDNO DS CL6, C
INAREA DS CL80, C
TESTOPTS DS CL2, C
TESTSWTS DS CL2, C
TESTLEN DS CL4, C
TESTLOC DS CL4, C
ORG
DIRENTRY DS F
DIRSPACE DS H
DIRBLOCK DS CL256
EXCLUDE1 DS F
EXCLUDE2 DS F
CARDAREA DS CL80
DS C'FF'
TRTAB DS CL256
DS 10X'0'
EXCLUDES DS 300CL8
EXCLUDEX DS CL8
DS 0D
WORKDLEN EQU *-WORKD
Converting Unix applications to MVS

INTRODUCTION
In the past few years, there has been an explosion in the number of Web servers and sites implemented on the Internet. These Web servers are used to present information to the public in words and pictures, sound and animation, and to communicate information throughout an individual organization over a proprietary intranet.

A Web server contains software running on a host operating system – such as Unix or Windows NT – that enables an organization to publish information on the Internet or in a private intranet. A Web server can be an IBM host computer running OS/390 with TCP/IP and OpenEdition. In some cases it can be useful to convert applications from Unix to OpenEdition.

UNIX TO OPENEDITION CONVERSION STEPS
Unix source code provided on a Unix tape.

1 Archive the source file to tape:

    tar -cvf /dev/rmt0
2 Load Unix system at porting centre:
   ```
   tar -xvf mvsport.tar /unix/source
   ```

3 Transfer to MVS, using the following steps:
   - ftp hostname
   - user-id
   - password
   - binary
   - put `/unix/source/mvsport.tar 'mvs.dataset'
   - close
   - quit

4 Transfer to hfs:
   ```
   touch /mvs/hfs
   chmod +e/mvs/hfs
   oput 'mvs.dataset''mvs/hfs' binary
   ```

5 Unpack:
   ```
   pax -o from=iso8859-1,to=ibm-1047 -rf/mvs/hfs/mvsport.tar
   ```

   You now have the option to convert from ASCII to EDCDIC.

   Set environment variable in .PROFILE for pax:
   ```
   export a2e='-o from=iso8859-1,to=ibm-1047'
   ```

   Then the command becomes:
   ```
   pax $a2e -rf /mvs/hfs/mvsport.tar
   ```
Serena Software has released StarWarp, a tool for automating the Year 2000 testing phase. Users can simulate future dates without writing specialized batch programs for each test scenario. StarWarp provides tools for assessing the scope of Year 2000 projects. When used in conjunction with DBMS import/export facilities, StarWarp can create aged test data for IMS, IDMS, ADABAS, DB2. It supports VSAM, PDS, PDSE, sequential, and direct access files in OS/390 and MVS.

For further information contact:
Serena Software, 500 Airport Blvd, 2nd Floor, Burlingame, CA 94010-1904, USA.
Tel: (415) 696 1800.
Fax: (415) 696 1776.

Compuware has announced Release 3.1 of its CICS Abend-Aid/FX fault management tool, geared towards resolving transaction and region problems. It provides programmers with on-line access to information about faults, identifying problems, capturing key fault information, listing all concurrent problems, and analysis and diagnosis of captured information to pinpoint the cause of the problem. Among the new facilities are full transaction abend and region dump processing support for CICS Transaction Server for OS/390 Version 1.2 and compatibility support for DB2, IMS, and other IBM products. There are specific diagnostics for CICS Abend-Aid/FX in the sysplex environment, and custom support for Language Environment for MVS.

For further information contact:
Compuware, 31440 Northwestern Highway, PO Box 908, Farmington Hills, MI 48334-2564, USA.
Tel: (810) 737 7300.
Fax: (810) 737 7199.
Compuware, 163 Bath Road, Slough, Berkshire, SL1 4AA, UK.
Tel: (01753) 774000.
Fax: (01753) 774200.

IBM has announced EDMSuite OnDemand for OS/390 Version 1 Release 1. EDMSuite is a solution for storing, retrieving, and presenting computer-generated reports and other business-related documents. Once the documents are indexed and stored, the OnDemand 3270, client/server, or Internet interface allows users to search, retrieve, view, print, and fax documents. OnDemand also provides Computer Output to Laser Disk (COLD) functions.

IBM has also announced the release of ImagePlus Version 3 for OS/390. This extends the reach of ImagePlus into the client/server environment and increases the capabilities of ImagePlus Object Distribution Manager MVS/ESA. ImagePlus captures large volumes of document-based information and stores them as electronic images. ImagePlus Object Distribution Manager MVS/ESA Version 3.1 includes support for TCP/IP.

Contact your local IBM representative for further information.