73

DB2

November 1998

In this issue

update

DB2 Update

Published by
Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38030
From USA: 01144 1635 38030
E-mail: xephon@compuserve.com

North American office
Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75077-2150
USA
Telephone: 940 455 7050

Contributions
Articles published in *DB2 Update* are paid for at the rate of £170 ($250) per 1000 words and £90 ($140) per 100 lines of code for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*.

DB2 Update on-line
Code from *DB2 Update* can be downloaded from our Web site at http://www.xephon.com; you will need the user-id shown on your address label.

Editor
Robert Burgess

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, and other contents of this journal before making any use of it.

Subscriptions and back-issues
A year's subscription to *DB2 Update*, comprising twelve monthly issues, costs £245.00 in the UK; $365.00 in the USA and Canada; £251.00 in Europe; £257.00 in Australasia and Japan; and £255.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1994 issue, are available separately to subscribers for £21.00 ($31.00) each including postage.

Printed in England.

Analysing IFCID 6 records

The buffer pool hit ratio cannot be 100%, because the initial load of data pages to the buffer takes some synchronous I/Os. My client's shop has a daily production batch job that updates the DB2 database with a buffer pool hit ratio of over 90%, but the fact is that the 10% synchronous I/O actually accounts for 90% of the run-time of the job! The job will go down the tube, with much longer elapsed time, if there is contention on a DASD with other jobs. (The job shares a DASD pool with another application and dedicating volumes to this job and its own application may cause even worse contention with itself.) Average access time to a 3390 DASD, normally 15 msec, could easily double per I/O.

Since the run-time of the job is critical for the on-time start-up of the on-line application, the DBA has to come up with an idea to reduce and stabilize the elapsed time of the job.

There are different areas that can be looked at, but buffer pool tuning is a must and cannot be ignored. To find whether the current buffer pool hit ratio is optimal for the job, the DBA needs to know whether there are minimal or balanced re-read I/Os for the same pages of each tablespace or index.

IFCID 6 records show the read activity to the buffer pool in detail by DBID, OBID, and page number. By sorting the records by DBID, OBID, and page number, and analysing the re-read activity of pages that belong to the same tablespace or index, the DBA will be able to determine which object of the database needs more buffer space – even if the buffer pool size for the database as a whole looks enough.

Here are the steps to follow for analysing the 'end read I/O trace':

1 Run the DB2 performance trace for the duration of the batch job:

```
START TRACE(PERFM) DEST(SMF) PLAN(the plan name) CLASS(30) IFCID(6)
```

Stop the trace as soon as the job ends. The trace will create SMF record type 102 with the plan name of the job.

2 Extract the IFCID 6 records from the SMF dump dataset. In most

shops the SMF dump dataset is created daily, which means the IFCID 6 records may not be available until the following day.

JCL001 AND JCL002

JCL001, with a source code in SYSIN, will assemble and link-edit a program called SMFREAD. It extracts IFCID type 6 records from the SMF dataset by DBID, OBID, and page number. The DB2 subsystem-id and the SMF record type are hard-coded in the program.

Next, JCL002 will run the program SMFREAD with the local SMF dump dataset as input. The output is written to a new dataset. The second step of the job will sort the output data in ascending order of DBID, OBID, and page number.

JCL001

```
//JCLØØ1 JOB (acct_info),'pgmer_name',NOTIFY=, <=== your job card info
// MSGCLASS=?,GROUP=,USER=
/*JOBPARM . . . . .
//*****************************************************************
//* ASSEMBLE AND LINKEDIT
//*****************************************************************
//ASMH EXEC ASMHCL
//C.SYSLIB DD DSN=SYS1.MACLIB,DISP=SHR
// DD DSN=hlq.DB2.SDSNMACS,DISP=SHR <=== your high-level qualifier
//L.SYSIN DD *
SMFREAD CSECT
*********************************************************************
* THIS PROGRAM EXTRACTS IFCID 6 RECORD FROM THE DAILY SMF DUMP.      *
* SMF record type, IFCID ID, DB2 subsystem ID are hard-coded.        *
*********************************************************************
R1 EQU 1
R2 EQU 2
R3 EQU 3
R4 EQU 4
R5 EQU 5
R6 EQU 6
R7 EQU 7
R8 EQU 8
R9 EQU 9
R1Ø EQU 1Ø
R11 EQU 11
R12 EQU 12
R13 EQU 13
```

```
R14 EQU 14
R15 EQU 15
*
BEGIN    SAVE  (14,12)
         BALR  R3,Ø
         USING *,R3
         USING SM1Ø2,R11
         USING QWØØØ6,R5 DATA SECTION
         ST    R13,SAVEAREA+4
         LA    R13,SAVEAREA
         OPEN  (SMFREC,INPUT,IFCID6,OUTPUT)
         GETMAIN RU,LV=32722
         LR    R11,R1
READREC  GET   SMFREC,Ø(R11)
         CLI   SM1Ø2RTY,X'66' RECORD TYPE OF PERF, AUDIT OR MONITOR ?
         BNE   READREC
         CLC   SM1Ø2SSI,=C'DBxx' DB2 ssid
         BNE   READREC
         L     R4,SM1Ø2END
         LA    R1Ø,Ø(R11,R4)
         CLC   4(2,R1Ø),=X'ØØØ6' IFCID 6 ?
         BNE   READREC
         LA    R5,SM1Ø2END+16
         MVC   DBID,QWØØØ6DB
         MVC   OBID,QWØØØ6OB
         MVC   BPID,QWØØØ6BP
         MVC   PAGEID,QWØØØ6PN
         MVC   READTYPE,QWØØØ6F
         MVC   RQSTOR,QWØØØ6AC
         PUT   IFCID6,WORKAREA
         B     READREC
EOFILE   CLOSE (SMFREC,,IFCID6)
         L     R13,SAVEAREA+4
         RETURN (14,12)
SMFREC   DCB   DSORG=PS,RECFM=VBS,MACRF=GM,BLKSIZE=27998,              X
               LRECL=32767Ø,DDNAME=SMFIN,EODAD=EOFILE
IFCID6   DCB   DSORG=PS,RECFM=FB,MACRF=PM,DDNAME=OUTREC,LRECL=8Ø,      X
               BLKSIZE=312Ø
*
         DS    ØF
SAVEAREA DS    18F
WORKAREA DS    ØCL8Ø
DBID     DS    XL2 DBID
OBID     DS    XL2 PAGESET OBID
BPID     DS    F BUFFER POOL INTERNAL ID
PAGEID   DS    XL3 1ST PAGE NUMBER TO BE READ
READTYPE DS    C
RQSTOR   DS    F
         DC    64C' '
         DS    ØF
```

```
         DSNDQWSP ,
         DSNDQWHS ,
         DSNDQW00
         END   SMFREAD
/*
//L.SYSLMOD DD DSN=your.linklib(SMFREAD),DISP=SHR <==== your link
library name
```

JCL002

```
//JCL002 JOB (acct-info),'addr,pgmr-name',NOTIFY=userid, <=== your job
card
// MSGCLASS=?,GROUP=group-id,USER=userid
/*JOBPARM ROOM=room-numbr
//*************************************************************
//* EXTRACT IFCID 6 RECORDS FROM THE DAILY SMF DUMP FILE
//*************************************************************
//STEP1 EXEC PGM=SMFREAD,REGION=0K
//STEPLIB DD DSN=your.linklib,DISP=SHR <=== your link library name
//SYSPRINT DD SYSOUT=*
//SMFIN DD DSN=smf.dataset,DISP=SHR, <=== your SMF dump dsn
// DCB=(RECFM=VBS,LRECL=32760,BLKSIZE=27998,EROPT=SKP)
//OUTREC DD DSN=extract.output.dataset,DISP=(NEW,CATLG), <=== your dsn
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120),
// UNIT=DASD,LABEL=RETPD=30,SPACE=(TRK,(100,500),RLSE)
//SYSUDUMP DD SYSOUT=*
//*
//*************************************************************
//* SORT BY DBID, OBID, PGID
//*************************************************************
//SORT EXEC PGM=SORT,REGION=0K
//SORTIN DD DSN=*.STEP1.OUTREC,DISP=SHR
//SORTOUT DD DSN=*.SORT.SORTIN,DISP=SHR
//SORTWK01 DD DISP=(NEW,DELETE),UNIT=SYSOUT3,SPACE=(TRK,(200,200))
//SORTWK02 DD DISP=(NEW,DELETE),UNIT=SYSOUT3,SPACE=(TRK,(200,200))
//SORTWK03 DD DISP=(NEW,DELETE),UNIT=SYSOUT3,SPACE=(TRK,(200,200))
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD *
 SORT FIELDS=(1,2,A,3,2,A,9,3,A),FORMAT=CH,EQUALS
/*
```

TO RUN A REPORT JOB

The following REXX program will take the output dataset from step 2 as an input argument. It processes the IFCID 6 records to create a report that shows the buffer REREAD RATIO for each tablespace and index.

```
/* rexx                                                              */
/*********************************************************************/
/* This program reports the following information using the input */
/* data from ifcid 6 records.                                        */
/*      dbid, obid, buffer pool,                                     */
/*      total number of reads, number of pages read once,            */
/*      number of reads of same pages,                               */
/*      number of pages read multiple times                          */
/*      number of rereads, number of reads per page,                 */
/*      reread ratio                                                 */
/* As summary,                                                       */
/*      total number of read I/O, rereads,                           */
/*      number of pages per page, reread ratio                       */
/*********************************************************************/
ARG indsn
address tso
"alloc ddname(ifcid6) dsname('"indsn"') shr reuse"
"EXECIO * diskr ifcid6 (finis"
NUMERIC DIGITS 7
if queued() <> 0 then do
 say ' '
 say ' '
 say ' Bufferpool reread ratio'
 say ' as of '
 say ' ' DATE()
 say ' '
 say ' ' ' ' ' ' ' ' TOTAL # ' '# PAGES ',
 '-- REREAD -- ' ' ' ' ' 'REREAD'
 say 'DBID ' 'OBID ' 'BP ' ' OF READ ' 'READ ONCE ',
 ' #READ #PAGES ' ' #REREAD ' 'READ/PG ' 'RATIO '
 say '-- ' '-- ' '- ' '----- ' '----- ',
 '---------- ' '----- ' '--- ' '--'
 end
else EXIT

k=1                /* used for index of the bp */
                   /* initialize output variable */
dbid. = ''
obid. = ''
bpid. = ''         /* contains the buffer pool number */
pageid. = ''
readtyp. = ''
reread. = 0
n1read = 0         /* total 1 reads of an obid */
nn1read = 0        /* total non-one reads of an obid */
totread = 0        /* total reads of the obid */
totrread = 0       /* total rereads of the obid */
nn1pages = 0       /* total pages that had >1 reads */
bptread. = 0       /* total reads by bp pool and the grand total */
bptrread.= 0       /* total rereads by bp pool and the grand total */
```

```
/* process the first record to init the parms */
PARSE PULL record1
do while SUBSTR(record1,12,1) <> 'R'
   PARSE PULL record1
end
curdbid = SUBSTR(record1,1,2)
curobid = SUBSTR(record1,3,2)
curbpid = SUBSTR(record1,5,4)
curpgid = SUBSTR(record1,9,3)
reread = 1
bpid.1 = curbpid

do until queued() = 0

   PARSE PULL record1
   if SUBSTR(record1,12,1) = 'R' then do
      hexdbid = SUBSTR(record1,1,2)
      if hexdbid = curdbid then do
         hexobid = SUBSTR(record1,3,2)
         if hexobid = curobid then do
            hexpgid = SUBSTR(record1,9,3)
            if hexpgid = curpgid then
               reread = reread + 1
            else do
               if reread <> 1 then do
                  nn1read = nn1read + reread
                  nn1pages = nn1pages + 1
               end
               else n1read = n1read + 1
               reread = 1
               curpgid = hexpgid
            end
         end
         else do
            call sayrread
            curobid = hexobid
            curbpid = SUBSTR(record1,5,4)
            curpgid = SUBSTR(record1,9,3)
            reread = 1
         end
      end
      else do
         call sayrread
         curdbid = hexdbid
         hexobid = SUBSTR(record1,3,2)
         curobid = hexobid
         curbpid = SUBSTR(record1,5,4)
         hexpgid = SUBSTR(record1,9,3)
         curpgid = hexpgid
```

```
         reread = 1
      end
   end
end

/* write an output line */
call sayrread

/* write the summary line */
say ' '
say ' '
say ' '
say ' '
say ' '
say ' BP ' 'TOTAL READS ' 'TOTAL REREADS ' 'READ/PG ' 'RATIO'
say ' — ' '_____ ' '_______ ' '___ ' '__'
grtread = 0
grtrread = 0
do i = 1 to k
   bpidnum = RIGHT(C2D(bpid.i),2)
   bptrnum = RIGHT(bptread.i,10)
   bptrrnm = RIGHT(bptrread.i,9)
   bprppg = (bptread.i + bptrread.i) / (bptread.i - bptrread.i)
   bprppg = FORMAT(bprppg,6,1)
   bpratio = bptrread.i / bptread.i
   bpratio = FORMAT(bpratio,3,2)
   say ' ' bpidnum ' ' bptrnum ' ' bptrrnm bprppg bpratio
   grtread = grtread + bptread.i
   grtrread = grtrread + bptrread.i
end
grtrnum = RIGHT(grtread,11)
grtrrnm = RIGHT(grtrread,10)
grppg = (grtread + grtrread) / (grtread - grtrread)
grppg = FORMAT(grppg,6,1)
gratio = grtrread / grtread
gratio = FORMAT(gratio,3,2)
say ' TOTAL ' grtrnum ' ' grtrrnm grppg gratio

address tso
"FREE fi(ifcid6)"

EXIT

sayrread :
 if reread <> 1 then do
    nn1read = nn1read + reread
    nn1pages = nn1pages + 1
 end
 else n1read = n1read + 1
 dbidnum = RIGHT(C2D(curdbid),4)
```

```
obidnum = RIGHT(C2D(curobid),4)
bpidnum = RIGHT(C2D(curbpid),2)
totread = n1read + nn1read
totrread = nn1read - nn1pages
totrnum = RIGHT(totread,8)
totrrnum = RIGHT(totrread,7)
n1rnum = RIGHT(n1read,8)
nn1rnum = RIGHT(nn1read,8)
nn1pgnum = RIGHT(nn1pages,6)
readppg = (n1read + nn1read) / (nn1pages + n1read)
rdppgnum = FORMAT(readppg,6,Ø)
ratio = totrread / totread
rationum = FORMAT(ratio,1,2)
say dbidnum ' ' obidnum ' ' bpidnum ' ' totrnum ' ' n1rnum ' ',
nn1rnum ' ' nn1pgnum ' ' totrrnum ' ' rdppgnum ' ' rationum

/* accumulate the read stats for each bp */
do i = 1 to k
   if bpid.i = curbpid then do
      bptread.i = bptread.i + totread
      bptrread.i = bptrread.i + totrread
      leave
      end
end

if i > k then do /* new bpid */
   k = k + 1
   bpid.k = curbpid
   bptread.k = totread
   bptrread.k = totrread
   end

n1read = Ø
nn1pages = Ø
nn1read = Ø
RETURN
```

ANALYSING THE RE-READ REPORT

Figure 1 shows the actual report indicating all the re-read information of the batch job. Figure 2 shows the summary by buffer pool.

The headings for the report shown in Figure 1 are as follows:

- BP – the buffer pool-id assigned to the DBID and OBID.
- Total # of read – the total number of read I/Os to the buffer for the DBID and OBID.

DBID	OBID	BP	TOTAL # OF READ	# PAGES READ ONCE	— REREAD — #READ	#PAGES	#REREAD	READ/PG	REREAD RATIO
1	95	0	74	68	6	3	3	1	0.04
1	100	0	20	0	20	3	17	7	0.85
6	9	0	38	38	0	0	0	1	0.00
6	10	0	1	1	0	0	0	1	0.00
6	93	0	2	2	0	0	0	1	0.00
6	98	0	3	3	0	0	0	1	0.00
6	101	0	13	13	0	0	0	1	0.00
7	2	0	1	1	0	0	0	1	0.00
7	6	0	2	2	0	0	0	1	0.00
280	2	1	20078	15947	4131	1952	2179	1	0.11
280	4	1	21639	16293	5346	2439	2907	1	0.13
280	6	1	1877	1633	244	120	124	1	0.07
280	10	1	34	19	15	7	8	1	0.24
280	12	1	4365	3511	854	410	444	1	0.10
280	14	2	473	469	4	2	2	1	0.00
280	18	2	13459	5151	8308	3102	5206	2	0.39
280	24	1	241	95	146	60	86	2	0.36
280	26	2	2	2	0	0	0	1	0.00
280	28	1	5	5	0	0	0	1	0.00
280	30	2	4122	3144	978	446	532	1	0.13
280	38	1	4105	3546	559	276	283	1	0.07
280	42	1	37	11	26	10	16	2	0.43
280	44	1	39	27	12	5	7	1	0.18
280	50	2	10	10	0	0	0	1	0.00
280	52	2	43	11	32	10	22	2	0.51
280	54	2	13	7	6	2	4	1	0.31
280	56	2	2	2	0	0	0	1	0.00
280	78	2	17	17	0	0	0	1	0.00
280	80	2	2	2	0	0	0	1	0.00
280	122	3	19740	6724	13016	5494	7522	2	0.38
280	124	3	40	10	30	10	20	2	0.50
280	126	3	2262	1385	877	394	483	1	0.21
280	128	3	12	0	12	5	7	2	0.58
280	132	3	33	8	25	6	19	2	0.58
280	134	3	6	6	0	0	0	1	0.00
280	136	3	31	15	16	5	11	2	0.35
280	138	3	4075	2044	2031	914	1117	1	0.27
280	142	3	4656	1969	2687	1165	1522	1	0.33
280	148	3	16828	381	16447	4338	12109	4	0.72
280	150	3	4	4	0	0	0	1	0.00
280	152	3	4	4	0	0	0	1	0.00
280	156	3	2	2	0	0	0	1	0.00
280	164	3	19	10	9	2	7	2	0.37
280	166	3	71	67	4	2	2	1	0.03
280	174	3	7	7	0	0	0	1	0.00
280	178	3	4	0	4	2	2	2	0.50
280	180	3	464	378	86	35	51	1	0.11
280	184	3	4290	2900	1390	654	736	1	0.17
280	190	3	20107	11844	8263	3779	4484	1	0.22
280	192	3	44	6	38	17	21	2	0.48
280	194	3	21	5	16	8	8	2	0.38
280	198	3	2	2	0	0	0	1	0.00
280	202	3	344	342	2	1	1	1	0.00
280	206	3	3586	859	2727	975	1752	2	0.49
280	208	3	9	9	0	0	0	1	0.00
280	212	3	8	8	0	0	0	1	0.00
280	214	3	2800	1011	1789	696	1093	2	0.39
280	216	3	35	13	22	8	14	2	0.40
280	218	3	93	93	0	0	0	1	0.00
280	220	3	1	1	0	0	0	1	0.00

Figure 1: Buffer pool re-read ratio report

BP	TOTAL READS	TOTAL REREADS	READ/PG	RATIO
Ø	154	2Ø	1.3	Ø.13
1	52420	6Ø54	1.3	Ø.12
2	18143	5766	1.9	Ø.32
3	79598	3Ø981	2.3	Ø.39
TOTAL	15Ø315	42821	1.8	Ø.28

Figure 2: Summary report

- # Pages read once – the number of pages of the DBID and OBID that didn't get read to the buffer more than once. After the initial load the pages were read from the buffer when the program requested GETPAGES.
- Reread #read – the number of read I/Os for a page of the DBID and OBID.
- Reread #pages – the number of pages that were read to the buffer more than once.
- #reread – the number of read I/Os for a page that had been read to the buffer.
- Read/page – the average number of read I/Os per page (total # of read) / (# pages read once + reread #pages).
- Reread ratio – #reread / total # of read.

While the buffer pool hit ratio of BP1, BP2, and BP3 are all over 90% for this job according to the DB2PM report, this report discloses a possible problem with the performance of BP3.

If several runs of this job keep showing high values for the 'reread ratio' of OBID 148 and 206 for DBID 280, it could be a good time to put more effort into the buffer pool tuning.

Samuel Park
DBA (USA)

Re-linking DB2 modules using SMP/E

With the delivery of DB2 Version 4.1, there may be instances where the re-linking of DB2 supplied modules is necessary. The first instance is prompted by IBM DB2 development, as specified in the *DATABASE 2 for MVS/ESA Version 4 Installation Guide* (Document Number SC26-3456-00):

2.7.1.11 DSNALI Linkage Attributes

DSNALI is now shipped with the attributes AMODE(31) and RMODE(ANY). You must re-link DSNALI if your applications need to load it below the 16MB line.

We encountered the second scenario when we migrated our production DB2 Version 3.1 subsystem to Version 4.1 and received the following error message when DSNECP10 was called by a COBOL applications program:

```
    +  IKF995I  ILBONTR AMODE ERROR. ENTERED IN AMODE 31.
```

In the case of my current shop, both DSNALI and DSNECP10 had to be re-linked with AMODE(24) and RMODE(24). There are two ways of doing this, either manually or using SMP/E.

Whether you are using the manual method or the SMP/E method, the first step is the same – an SMP/E unload of the LMOD entry that we wish to re-link. In this case, when we issued the command 'UNLOAD LMOD(DSNALI,DSNECP10)' in our DB2 target zone we received:

```
REP       LMOD            ( DSNALI   )
          LASTUPD         ( HDB4410 )
          LASTUPDTYPE     ( ADD )
          SYSLIB          ( SDSNLOAD )
          /* LEPARM    */   STD
++LMODIN
 ORDER DSNAA
 ENTRY DSNALI
 ALIAS DSNHLI2
 ALIAS DSNWLI2
 MODE AMODE(24),RMODE(24)
++ENDLMODIN
                          .
REP       LMOD            ( DSNECP10 )
```

```
          LASTUPD         ( UN9Ø165 )
          LASTUPDTYPE     ( UPD )
          SYSLIB          ( SDSNLOAD )
          /* LEPARM    */   STD
++LMODIN
 ORDER DSNAA                          /* added for KFF0365. RDD    */
 ENTRY DSNECP1Ø
 MODE AMODE(24),RMODE(24)
++ENDLMODIN
                         .
ENDUCL.
```

Using the linkage editor statements supplied in the ++LMODIN section, it was very easy to re-link the two modules, and with that the manual re-link process is complete.

The problem here is that, if we ever apply maintenance to any object module referenced by the two modules, they are re-linked automatically by SMP/E with AMODE(31) RMODE(ANY) – the linkage editor attributes listed in SMP/E for those two load modules. To ensure that this modification is carried across maintenance levels, we must make SMP/E aware of the changes that we've made. To do that, we use SMP/E usermods.

In addition to the LMOD information that we've acquired above, we also issue the following command in our DB2 target zone:

```
    UNLOAD MOD(DSNALI, DSNECP1Ø)
```

From which we receive the following output:

```
UCLIN .
REP       MOD             ( DSNALI   )
          LASTUPD         ( HDB441Ø )
          LASTUPDTYPE     ( ADD )
          DISTLIB         ( ADSNLOAD )
          FMID            ( HDB441Ø )
          RMID            ( CSHMOD2 )
          LMOD            ( DSNALI    DSNX9SER  DSNX9STP )
                          .
REP       MOD             ( DSNECP1Ø )
          LASTUPD         ( HDB441Ø )
          LASTUPDTYPE     ( ADD )
          DISTLIB         ( ADSNLOAD )
          FMID            ( HDB441Ø )
          RMID            ( CSHMOD1 )
          LMOD            ( DSNECP1Ø )
                          .
```

```
ENDUCL.
UCLIN .
```

CSHMOD1

The information from the MOD and LMOD entries is used to build our USERMODs. For DSNALI, we created CSHMOD1:

```
++USERMOD(CSHMOD2) REWORK(1998001) /* 24/24 INPLACE OF 31/ANY */ .
++VER(P115) FMID(HDB4410) .
++JCLIN .
//LKED EXEC PGM=IEWL
//SYSLMOD   DD DISP=SHR,DSN=DSN410.SDSNLOAD
//SYSIN     DD *
 ORDER DSNAA
 ENTRY DSNALI
 ALIAS DSNHLI2
 ALIAS DSNWLI2
 MODE AMODE(24),RMODE(24)
 NAME DSNALI(R)
++MOD(DSNALI)
          DISTLIB         ( ADSNLOAD )
          LMOD            ( DSNALI    DSNX9SER  DSNX9STP )
          TXLIB           ( ADSNLOAD )
                          .
```

CSHMOD2

For DSNECP2, we created CSHMOD2:

```
++USERMOD(CSHMOD1) REWORK(1998001) /* 24/24 IN PLACE OF 31/ANY */ .
++VER(P115) FMID(HDB4410) PRE(UN90165) .
++JCLIN .
//LKED EXEC PGM=IEWL
//SYSLMOD   DD DISP=SHR,DSN=DSN410.SDSNLOAD
//SYSIN     DD *
 ORDER DSNAA                      /* added for KFF0365. RDD    */
 ENTRY DSNECP10
 MODE AMODE(24),RMODE(24)
 NAME DSNECP10(R)
++MOD(DSNECP10)
          DISTLIB         ( ADSNLOAD )
          LMOD            ( DSNECP10 )
          TXLIB           ( ADSNLOAD )
                          .
```

The ++JCLIN information comes directly from the ++LMODIN section of the LMOD entries. As for the ++MOD information, the

FMID and PRE come from the MOD entries, as does the DISTLIB, LMOD, and TXTLIB information.

Once the SMP/E usermods are built, they can be received into the GLOBAL zone, where they can be applied to the DB2 target zones. You will notice that the linkage editor attribute changes to AMODE(24) and RMODE(24) when they are applied – but only for the LMODs specified in the USERMODs. Although modules DSNX9SER and DSNX9STP are specified in the LMOD list of object module DSNALI, they are still linked with AMODE(31) and RMODE(ANY).

When the time comes to receive and apply maintenance, we have to use SMP/E RESTORE to back out these two USERMODs before any new maintenance can be applied to the modules DSNALI and DSNECP10.

After we have applied our maintenance, we have to retrofit the USERMODs and let SMP/E know that the new maintenance level is recognized. We accomplish this by unloading the MOD entries for DSNALI and DSNECP10 once again, and then applying the LASTUPD information into the PRE information of the USERMODs.

One item of note – the first time maintenance is applied, the new object modules for DSNALI and DSNECP10 must be put into an object module library known to SMP/E, and the TXLIB operand changed, then the USERMODs re-received. This is because we are currently loading the object code for DSNALI and DSNECP10 out of the ADSNLOAD library, since all the maintenance has been accepted.

After we apply maintenance, the modules in ADSNLOAD will become back level until the new maintenance is accepted.

Chorng S (Jack) Hwang
Principal
HSA Systems (USA)

DB2 Update is looking for REXX EXECs, macros, program code, etc, that experienced DB2 users have written to make their life easier. Articles can be sent or e-mailed to Robert Burgess at any of the addresses shown on page 2.

Simulating a production environment – part 3

This month we conclude the article on simulating a production environment in DB2.

```
/*******  DISPLAYS SPECIFIC FILE RECORD *********/

   LIST.SCR=LEFT(TBNAME,8)' 'LEFT(COLNAME,18)' 'LEFT(COLVALUE,4Ø),
   ' 'RIGHT(FREQUENC,4,Ø)
   SAY 'TABLE    COLUMN             COLVALUE
FREQUENCE'
   SAY LIST.SCR
   PARSE PULL . +9 .+19  COLVALUE +41 FREQUENC +5 .

/*******  PREPARES CHANGED FIELDS TO REWRITE FILE RECORD ******/
   IF FREQUENC¬='' THEN
      DO
        TBNAME=LEFT(TBNAME,18)
        COLNAME=LEFT(COLNAME,18)
        COLVALUE=LEFT(COLVALUE,254)
        IF FREQUENC¬='FFFF'  &,
           DATATYPE(FREQUENC)=NUM  THEN
           FREQUENC=D2C(FREQUENC,2)
        ELSE
           IF FREQUENC='FFFF' THEN
              FREQUENC=COPIES(X2C('FF'),2)
           ELSE
              DO
                A=A-1
                SAY '***ERROR: FREQUENC IS INVALID'
                ITERATE
              END

  /*******  PROCESSES NEXT RECORD ON INPUT FILE  ******/
     DO
          NM=NM+1
          TBLO.NM=TRANSLATE(TBNAME),
                 ||TRANSLATE(COLNAME),
                 ||TRANSLATE(COLVALUE),
                 ||LEFT(FREQUENC,4,Ø)
        END
     END
   ELSE
       NOP
   IF VAR1='YES' THEN
      NOP
```

```
   ELSE
      DO
       A=A+1
       PARSE VALUE TBL.A WITH 1  TBNAME 7 .
       IF VAR2 = TBNAME THEN
         DO
           A=A-1
           ITERATE
         END
       ELSE
         DO
           SAY 'WHAT TABLE DO YOU WANT TO CHANGE?'
           PULL VAR2
           A=1
           ITERATE
         END
      END
END

"ALLOC F("DD2") SHR REUSE DA("DSN2")"
"EXECIO "NM" DISKW "DD2" (FINIS STEM TBLO.)"
"FREE F("DD2")"
RETURN

/* REXX */
```

DB2CATCL

DB2CATCL displays values from the SYSCOLUMNS output file produced by the catalog extraction program PCATV3EX.

```
/* REXX */

DD1="I"TIME("S")
DD2="O"TIME("S")
DSN1="'XXXXXXXX.SYSCOLMS'"
DSN2="'XXXXXXXX.SYSCOLOS'"

"ALLOC F("DD1") SHR REUSE DA("DSN1")"
"EXECIO * DISKR "DD1" (FINIS STEM TBL.)"
"FREE F("DD1")"

/*****  READS THRU INPUT FILE UNTIL SPECIFIED RECORD IS REACHED *****/
NM=Ø
SAY 'DO YOU WANT THE WHOLE DATABASE?'
PULL VAR1
IF VAR1='YES' THEN
   NOP
ELSE
```

```
  DO
   SAY 'WHAT TABLE DO YOU WANT TO CHANGE?'
   PULL VAR2
  END
DO A=1 TO TBL.Ø
   PARSE VALUE TBL.A WITH 1  TBNAME 7 .,
                          19 NAME 37 . ,
                          37 COLCARD  41 . ,
                          46 LOW2KEY  54 . ,
                          54 HIGH2KEY 62 . ,
                          62 COLTYPE  7Ø .
   DO
     IF VAR1='YES' THEN
        NOP
     ELSE
        IF VAR2 = TBNAME THEN
           NOP
        ELSE
           ITERATE
   END

/*******  PROCESSES COLCARD VALUES ******/
   DO
     COLCARD=C2X(COLCARD)
       IF COLCARD='FFFFFFFF'    THEN
         DO
           A=A+1
           PARSE VALUE TBL.A WITH 1  TBNAME 7 .
           IF VAR2 = TBNAME THEN
             DO
               A=A-1
               ITERATE
             END
           ELSE
             DO
               SAY 'WHAT TABLE DO YOU WANT TO CHANGE?'
               PULL VAR2
               A=1
               ITERATE
             END
         END
       ELSE
           COLCARD=X2D(COLCARD,8)
   END

/*******  PROCESSES ALL TYPES OF LOW2KEY VALUES ******/
   DO
     IF COLTYPE='DATE'  THEN
       DO
         LOW2KEY=LEFT(C2X(LOW2KEY),8)
```

```
         LOW2KEY=INSERT(LOW2KEY,'        ')
       END
     ELSE
       DO
        IF COLTYPE='TIMESTMP'  THEN
           LOW2KEY=LEFT(C2X(LOW2KEY),16)
        ELSE
          DO
            IF COLTYPE='TIME'  THEN
              DO
               LOW2KEY=LEFT(C2X(LOW2KEY),6)
               LOW2KEY=INSERT(LOW2KEY,'  ')
              END
            ELSE
              IF COLTYPE = 'CHAR' |,
                 COLTYPE = 'VARCHAR' THEN
                DO
                 LOW2KEY=LEFT(TRANSLATE(LOW2KEY,".",'ØØ'x),8)
                 LOW2KEY=INSERT(LOW2KEY,'        ')
                END
              ELSE
                DO
                 IF COLTYPE = 'DECIMAL' THEN
                   DO
                     LOW2KEY=LEFT(SPACE(LOW2KEY,1,'x'),17,'x')
                     LOW2KEY=STRIP(LOW2KEY,T,'x')
                     LOW2KEY=LEFT(C2X(LOW2KEY),17)
                     LOW2KEY=STRIP(LOW2KEY,L,'F')
                   END
                 ELSE
                    IF COLTYPE = 'INTEGER' THEN
                      DO
                        LOW2KEY=LEFT(SPACE(LOW2KEY,1,'x'),8,'x')
                        LOW2KEY=STRIP(LOW2KEY,T,'x')
                        LOW2KEY=LEFT(C2X(LOW2KEY),16)
                      END
                    ELSE
                      DO
                        LOW2KEY=LEFT(SPACE(LOW2KEY,1,'x'),8,'x')
                        LOW2KEY=STRIP(LOW2KEY,T,'x')
                        LOW2KEY=LEFT(C2X(LOW2KEY),16)
                      END
                END
          END
       END
   END

/*******  PROCESSES ALL TYPES OF HIGH2KEY VALUES ******/
   DO
     IF COLTYPE='DATE'  THEN
       DO
```

```
          HIGH2KEY=LEFT(C2X(HIGH2KEY),8)
          HIGH2KEY=INSERT(HIGH2KEY,'        ')
        END
      ELSE
        DO
         IF COLTYPE='TIMESTMP'  THEN
            HIGH2KEY=LEFT(C2X(HIGH2KEY),16)
         ELSE
           DO
            IF COLTYPE='TIME'  THEN
               DO
                HIGH2KEY=LEFT(C2X(HIGH2KEY),6)
                HIGH2KEY=INSERT(HIGH2KEY,'  ')
               END
            ELSE
              DO
               IF COLTYPE = 'CHAR' |,
                  COLTYPE = 'VARCHAR' THEN
                  HIGH2KEY=LEFT(TRANSLATE(HIGH2KEY,".",'ØØ'x),16)
               ELSE
                 DO
                  IF COLTYPE = 'DECIMAL' THEN
                    DO
                      HIGH2KEY=LEFT(SPACE(HIGH2KEY,1,'x'),17,'x')
                      HIGH2KEY=STRIP(HIGH2KEY,T,'x')
                      HIGH2KEY=LEFT(C2X(HIGH2KEY),17)
                      HIGH2KEY=STRIP(HIGH2KEY,L,'F')
                    END
                  ELSE
                     IF COLTYPE = 'INTEGER' THEN
                       DO
                         HIGH2KEY=LEFT(SPACE(HIGH2KEY,1,'x'),8,'x')
                         HIGH2KEY=STRIP(HIGH2KEY,T,'x')
                         HIGH2KEY=LEFT(C2X(HIGH2KEY),16)
                       END
                     ELSE
                       DO
                         HIGH2KEY=LEFT(SPACE(HIGH2KEY,1,'x'),8,'x')
                         HIGH2KEY=STRIP(HIGH2KEY,T,'x')
                         HIGH2KEY=LEFT(C2X(HIGH2KEY),16)
                       END
                 END
              END
           END
        END
   END

/*******  DISPLAYS SPECIFIC FILE RECORD *********/
   DO
     LIST.SCR=LEFT(TBNAME,8)'  'LEFT(NAME,18)'  'RIGHT(COLCARD,8,Ø),
```

```
     '    'LOW2KEY' 'HIGH2KEY
     SAY 'TABLE     COLUMN              COLCARD      LOW2KEY
HIGH2KEY'
     SAY LIST.SCR
     PARSE PULL  +1Ø. +2Ø. COLCARD +8. +5 LOW2KEY +16. +1 HIGH2KEY +16.
COLTYPE.
   END

/*******  PREPARES CHANGED FIELDS TO REWRITE FILE RECORD ******/
    IF COLCARD¬='' THEN
      DO
        TBNAME=LEFT(TBNAME,18)
        NAME=LEFT(NAME,18)
        IF COLCARD¬='FFFFFFFF'   &,
           DATATYPE(COLCARD)=NUM   THEN
           COLCARD=D2C(COLCARD,4)
        ELSE
           IF COLCARD='FFFFFFFF' THEN
              COLCARD=COPIES(X2C('FF'),4)
           ELSE
              DO
                A=A-1
                SAY '***ERROR: COLCARD IS INVALID'
                ITERATE
              END

/*******  PREPARES LOW2KEY VALUES THAT WERE CHANGED  ******/
        DO
          LOW2KEY=STRIP(LOW2KEY,T,' ')
          IF COLTYPE = 'DATE'  THEN
             DO
              LOW2KEY=STRIP(LOW2KEY,T,'x')
              LOW2KEY=LEFT(X2C(LOW2KEY),4)
             END
          ELSE
             IF COLTYPE= 'TIMESTMP'  THEN
                LOW2KEY=LEFT(X2C(LOW2KEY),8)
             ELSE
                DO
                 IF COLTYPE='TIME'  THEN
                    LOW2KEY=LEFT(X2C(LOW2KEY),3)
                 ELSE
                    DO
                     IF COLTYPE='DECIMAL' THEN
                       DO
                        LOW2KEY=INSERT('F',LOW2KEY)
                        LOW2KEY=LEFT(X2C(LOW2KEY),16)
                       END
                     ELSE
                       IF COLTYPE='INTEGER'  THEN
```

```
                        LOW2KEY=LEFT(X2C(LOW2KEY),8)
                      ELSE
                        IF COLTYPE='SMALLINT' THEN
                          LOW2KEY=LEFT(X2C(LOW2KEY),4)
                        ELSE
   IF POS('.',LOW2KEY)= Ø then
                            LOW2KEY=LEFT(TRANSLATE(LOW2KEY," ","x"),8)
                          ELSE
                           LOW2KEY=LEFT(TRANSLATE(LOW2KEY,'ØØ'x,"."),8)
                    END
                END
         END

/*******  PREPARES HIGH2KEY VALUES THAT WERE CHANGED  ******/
   DO
          HIGH2KEY=STRIP(HIGH2KEY,T,' ')
    IF COLTYPE = 'DATE'  THEN
              DO
               HIGH2KEY=STRIP(HIGH2KEY,T,'x')
               HIGH2KEY=LEFT(X2C(HIGH2KEY),4)
              END
          ELSE
              IF COLTYPE= 'TIMESTMP'  THEN
                 HIGH2KEY=LEFT(X2C(HIGH2KEY),8)
              ELSE
                 DO
                  IF COLTYPE='TIME'  THEN
                     HIGH2KEY=LEFT(X2C(HIGH2KEY),3)
                  ELSE
                    DO
                     IF COLTYPE='DECIMAL' THEN
                       DO
                        HIGH2KEY=INSERT('F',HIGH2KEY)
                        HIGH2KEY=LEFT(X2C(HIGH2KEY),16)
                       END
                     ELSE
                       IF COLTYPE='INTEGER'  THEN
                         HIGH2KEY=LEFT(X2C(HIGH2KEY),8)
                       ELSE
                         IF COLTYPE='SMALLINT' THEN
                            HIGH2KEY=LEFT(X2C(HIGH2KEY),4)
                         ELSE
                           IF POS('.',HIGH2KEY)= Ø then
                             HIGH2KEY=LEFT(TRANSLATE(HIGH2KEY," ,"x"),8)
                            ELSE
                             HIG2KEY=LEFT(TRANSLATE(HIGH2KEY,'ØØ'x,"."),8)
                    END
                END
         END
```

```
/*******  PROCESSES NEXT RECORD ON INPUT FILE  ******/
        NM=NM+1
        TBLO.NM=TRANSLATE(TBNAME),
                ||TRANSLATE(NAME),
                ||LEFT(COLCARD,9,Ø),
                ||LEFT(LOW2KEY,8,' ')||LEFT(HIGH2KEY,8,' ')
      END
   ELSE
      NOP
   IF VAR1='YES' THEN
      NOP
   ELSE
      DO
       A=A+1
       PARSE VALUE TBL.A WITH 1  TBNAME 7 .
       IF VAR2 = TBNAME THEN
         DO
           A=A-1
           ITERATE
         END
       ELSE
         DO
           SAY 'WHAT TABLE DO YOU WANT TO CHANGE?'
           PULL VAR2
           A=1
           ITERATE
         END
      END
END

"ALLOC F("DD2") SHR REUSE DA("DSN2")"
"EXECIO "NM" DISKW "DD2" (FINIS STEM TBLO.)"
"FREE F("DD2")"
RETURN

/* REXX */
```

LIVE EXPERIENCE

In December 1997, my company implemented a totally new business, supported by a large on-line insurance application package. Because this was a new business there was hardly any data converted and the DB2 tables had very little data.

After the physical implementation of the DB2 tables, the application had very poor response times and a lot of contention between the transactions.

After some monitoring, I concluded that most of the transactions were doing sequential, list, or dynamic prefetch – not making use of the indexes available per table.

I decided to implement these programs and REXXs in the following manner (I had already executed the RUNSTATS utility on all the tablespaces, tables, and indexes before executing the BIND of the whole application):

- Execute DB2CATEX to extract the values produced by the RUNSTATS on the various tables of the catalog.
- Execute REXX DB2 DB2CATTB and change only the CARD value, table by table, according to the total number of records expected at the end of 1998, instead of the current number of records (which was very few or none at all).
- Execute REXX DB2CATIX and change the FULLKEYCARD values equal to the CARD values entered in the previous REXX, doing this index by index.
- Execute DB2CATUP to update these respective values in the SYSTABLES and SYSINDEXES catalog tables of the production environment. By setting the other files' DD statements to dummy, the program only executes the updates for the files specified.
- Execute the BIND of the whole application with the option EXPLAIN(YES) and verify that the access paths changed from sequential and list prefetch to index accesses.

CONCLUSION

After this implementation, the application's response times improved considerably, most of the sequential accesses changed to index accesses, and all contention between CICS transactions disappeared.

By forcing DB2 to assume these values as the real ones during the BIND process, although physically it contains other values, the DB2 Optimizer decides to use index access paths.

Iolanda Lopes
Database Administator
Companhia Seguros Mundial Confiança (Portugal)

PLAN and PACKAGE management

INTRODUCTION

As an easy way to create the necessary statements for BIND and REBIND PLAN/PACKAGE, I have created two batch REXX EXECs. Starting from a DB2 catalog query, these procedures build libraries and jobs to REBIND a full project, or BIND a new project starting from an older one.

BIND PROCEDURE DESCRIPTION

With proper parameter customization, the tool builds libraries that contain the following type of BIND statement:

- I – BIND PACKAGE + BIND PLAN (including PKLIST).
- II – BIND PACKAGE + BIND PLAN (including PKLIST + MEMBER).
- III – BIND PLAN (including MEMBER).

The following parameters descibe how you can customize the REXX EXECs:

- SUBSYS – the DB2 subsystem name (no default).
- OPER – the BIND extract type. When the value is 'YYN', a type I extraction BIND is performed. When the value is 'YYY', a type II extraction BIND is performed, and when the value is 'NNY', a type III extraction BIND is performed (no default).
- CREATOR – the creator/owner of PLAN and PACKAGE (no default).
- HIWORK – the high-level name for work areas (default= '*'; when the value is '*', the high-level name is equal to the TSO user-id).
- ESOUNIT – the esoteric name for dataset allocation (default= '*'; when the value is '*', the ESOUNIT is equal to WORKA).

- ACCOUNT – The account job name (no default).

Depending upon the selection made for the OPER parameter, you can have the following output datasets:

- YYY
  - hiwork.subsys.creator.PLANPACK – library with the members of BIND PLAN, including PKLIST, and, if there is mix plan, BIND PLAN including PKLIST + MEMBER.
  - hiwork.subsys.creator.PACKAGE – library with the members of BIND PACKAGE.
  - hiwork.subsys.creator.PLAN – library with the members of BIND PLAN, including MEMBER.
  - hiwork.subsys.creator.UTILITY – library which contains the utility and BIND submission jobs.
- YYN
  - hiwork.subsys.creator.PLANPACK – library with the members of BIND PLAN, including PKLIST.
  - hiwork.subsys.creator.PACKAGE – library with the members of BIND PACKAGE.
  - hiwork.subsys.creator.UTILITY – library which contains the utility and bind submission jobs.
- NNY
  - hiwork.subsys.creator.PLAN – library with the members of BIND PLAN including MEMBER.
  - hiwork.subsys.creator.UTILITY – library which contains the utility and bind submission jobs.

The procedure may be used in several ways. For example, it might be useful during the migration of an application from the development stage to the test stage, creating all you need for the BIND process. Alternatively, it may be used during a back-up to extract all you need for the BIND process.

REBIND PROCEDURE DESCRIPTION

With proper parameter customization, the tool can build jobs for REBIND PLAN and PACKAGE. The following parameters descibe how you can customize the REXX EXEC:

- SUBSYS – the DB2 subsystem name (no default).
- CREATOR – the creator/owner of PLAN and PACKAGE (no default).
- AUTOSUB – allows you to automatically submit the jobs of Rebind PLAN/PACKAGE (default = '*'; when the value is '*', AUTOSUB is equal to NO).
- JOBNAME – the job name for REBIND procedures (default = '*'; when the value is '*', the jobname will be created automatically).
- HIWORK – the high-level name for work areas (default = '*'; when the value is '*', the high-level name is equal to TSO user-id).
- ESOUNIT – the esoteric name for dataset allocation (default = '*'; when the value is '*', the ESOUNIT is equal to WORKA).
- ACCOUNT – the account job name (no default).

The procedure has been tested in an MVS Version 4.3.0 and DB2 Version 3.1 environment.

DB2BIND REXX EXEC

```
/* REXX */
trace ?o
   /* ------------------------------------------------------------------*/
   /*      +-   Bind Catalog Extractor for DB2 v3.1   -+                -*/
   /*-                  ///////// * //////////                          -*/
   /*-           P A R A M E T E R S                  DEFAULT           -*/
   /*  ------------------------------+------------------------------- -*/
   /*-   - Subsys : DB2 subsystem name           | no default        -*/
   /*-   - Oper   : Bind extract type            | no default        -*/
   /*-   - Creator: Creator/Owner                | no default        -*/
   /*-   - Hiwork : HI-level work areas          | * = USERID        -*/
   /*-   - Esounit: Esoteric name for allocation | * = worka         -*/
   /*-   - Account: Name of job ACCOUNT          | no default        -*/
```

```
   /*- ------------------------------+------------------------------------ -*/
   /*-                    ///////// * //////////                          -*/
   /*- ------------------------------------------------------------------- */
$i     = 0
ct     = 0
ct1    = 0
ctrpk  = 0
ctrppk = 0
ctrppm = 0
blk    =
wrk    =
swsyspr= off
yesppk = off
jna    = userid()
user   = userid()
notif  = userid()
arg parmin
parm     = translate(parmin,' ',',')
nparm    = words(parm)
subsys   = word(parm,1)
oper     = word(parm,2)
creator  = word(parm,3)
hiwork   = word(parm,4)
esounit  = word(parm,5)
account  = word(parm,6)
   /*-------------------------------------------------*/
   /*-    Test input parameters                 ---*/
   /*-------------------------------------------------*/
 if nparm < 6 then do
    say '        '
    say '        '
    say '>>>>>>>>'
    say '>>>>>>>>  Parameter string is incomplete !!!!'
    say '>>>>>>>>           'parmin
    say '>>>>>>>>'
    say '        '
    say '        '
    exit
    end
 loper = length(oper)
 if loper  ¬= 3 then do
    say '     '
    say '     '
    say '>>>>>>>>'
    say '>>>>>>>> The variable OPER is wrong !!!!'
    say '>>>>>>>> it must be 3 bytes long.          '
    say '>>>>>>>>'
    say '     '
    say '     '
    exit
```

```
   end
if oper  ¬= 'YYN' &,
   oper  ¬= 'YYY' &,
   oper  ¬= 'NNY' then do
   say '     '
   say '     '
   say '>>>>>>>>'
   say '>>>>>>>>  The variable OPER is wrong !!!!!                    '
   say '>>>>>>>>         Specify :                                   '
   say '>>>>>>>>'
   say '>>>>>>>>     1)  YYN  Package + Plan with ( Pklist )         '
   say '>>>>>>>>     2)  YYY  Package + Plan with ( Pklist/Member )  '
   say '>>>>>>>>     3)  NNY  Plan with ( Member )                   '
   say '>>>>>>>>'
   say '     '
   say '     '
   exit
   end
if hiwork = '*' then
   hiwork = userid()
if esounit = '*' then
   esounit = worka
  /*-------------------------------------------------------*/
  /*-      File ipoupdte allocation          ---*/
  /*-------------------------------------------------------*/
outdsupd= hiwork'.'subsys'.'creator'.UTILITY'
xx=outtrap(trp01.)
   address tso "delete '"outdsupd"'"
   "alloc da('"outdsupd"') dir(40) space(15,45) dsorg(ps)" ,
   "recfm(f,b) lrecl(80) blksize(27920) tracks ",
   "unit("esounit") new catalog f(fiupd) "
xx=outtrap(off)
if rc > 0 then do
   do #a = 1 to trp01.0
      say trp01.#a
   end
   say '        '
   say '        '
   say '>>>>>>>>'
   say '>>>>>>>> "'outdsupd'"  Allocation OK'
   say '>>>>>>>>  RC='rc'. Verify.          '
   say '>>>>>>>>'
   say '        '
   say '        '
   exit
   end
else
   say '>>>>>>>>  'outdsupd '         Allocation OK  '
  /*-------------------------------------------------------*/
  /*-      File systsinp allocation          ---*/
```

```
   /*-------------------------------------------------------*/
 outds1= hiwork'.'subsys'.'creator'.SYSTSINP'
 xx=outtrap(trp02.)
    address tso "delete '"outds1"'"
    "alloc da('"outds1"') dir(0) space(5,1) dsorg(ps)" ,
    "recfm(f,b) lrecl(80) blksize(27920) tracks ",
    "unit("esounit") release new catalog f(systsinp) "
 xx=outtrap(off)
 if rc > 0 then do
    do #a = 1 to trp02.0
       say trp02.#a
    end
    say '        '
    say '        '
    say '>>>>>>>>'
    say '>>>>>>>> "'outds1'"  Allocation OK'
    say '>>>>>>>>  RC='rc'. Verify.          '
    say '>>>>>>>>'
    say '        '
    say '        '
    exit
    end
 else
    say '>>>>>>>>  'outds1 '        Allocation OK  '
sk.1='DSN SYSTEM('subsys')                                          '
sk.2='RUN PROGRAM(DSNTEP2) PLAN(DSNTEP31) LIB(''DSN310.RUNLIB.LOAD'')'
sk.3='END                                                           '
 sk.0=3;
 "execio * diskw systsinp (stem sk. finis"
 call Pulisci
   /*-------------------------------------------------------*/
   /*-     Number of select to do             ---*/
   /*-------------------------------------------------------*/
 wrk = translate(oper,'0','N')
 wrk = translate(wrk,'1','Y')
 do #y = 1 to 3
    ct = substr(wrk,#y,1)
    if ct = 1 then
       ct1 = ct1 + 1
    end
 select
 when ct1 = 1 then do
    call Allppm
   /*-------------------------------------------------------*/
   /*-    Bind Plan with member               ---*/
   /*-------------------------------------------------------*/
    tipsel = BPMEMB
    call Allsysp
    call Allsysi
    jobw = sysin
```

```
   "alloc da('"outdsØ"') f("jobw") shr reuse"
             sk.1=' SELECT * FROM VBPMEMB ORDER BY 1,2 ;        '
   sk.Ø=1
   call Writeout
   #i = 1
   call Rundb2
   macnr = @mdb2Ø23
   call Exmacro
   call Testout
   if build = yes then do
      call Wrbpmemb
      say '         '
      say '>>>>>>>>'
      say '>>>>>>>>  +-------------------------------------+'
      say '>>>>>>>>    Total Bind Plan "DBRM" no' ctrppm
      say '>>>>>>>>  +-------------------------------------+'
      say '>>>>>>>>'
      say '         '
      address tso "delete '"outdsØ"'"
      address tso "delete '"outds2"'"
      say '         '
      say '         '
      say '         '
      end
   else do
      say '         '
      say '>>>>>>>>'
      say '>>>>>>>> The select BPMEMB has Ø rows !!! '
      say '>>>>>>>> The 'outdsppm' will be empty.    '
      say '>>>>>>>>'
      say '         '
      address tso "delete '"outdsØ"'"
      address tso "delete '"outds2"'"
      say '         '
      end
   end
when ct1 = 2 then do
   call Allppk
   call Allpk
   do #i = 1 to 2
      if #i = 1 then do
  /*-------------------------------------------------------*/
  /*-      Bind Plan with Package                  ---*/
  /*-------------------------------------------------------*/
         tipsel = BPPACK
         call Allsysp
         call Allsysi
         jobw = sysin
         "alloc da('"outdsØ"') f("jobw") shr reuse"
              sk.1=' SELECT * FROM VBPPACK ORDER BY 2,1 ;    '
```

```
        sk.Ø=1
        Call Writeout
        Call Rundb2
        macnr = @mdb2Ø18
        call Exmacro
        call Testout
        if build = yes then do
           call Wrbppack
           say '         '
           say '>>>>>>>>'
           say '>>>>>>>>  +-----------------------------------------+'
           say '>>>>>>>>    Total Bind Plan "Pklist" no' ctrppk
           say '>>>>>>>>  +-----------------------------------------+'
           say '>>>>>>>>'
           say '         '
           address tso "delete '"outdsØ"'"
           address tso "delete '"outds2"'"
           say '         '
           say '         '
           say '         '
           end
        else do
           say '         '
           say '>>>>>>>>'
           say '>>>>>>>> The select BPPACK has Ø rows !!! '
           say '>>>>>>>> The 'outdsppk' will be empty,    '
           say '>>>>>>>>'
           say '         '
           address tso "delete '"outdsØ"'"
           address tso "delete '"outds2"'"
           say '         '
           end
        end
     if #i = 2 then do
/*-------------------------------------------------------*/
/*-      Bind Package                                 ---*/
/*-------------------------------------------------------*/
        tipsel = BPACK
        call Allsysp
        call Allsysi
        jobw = sysin
        "alloc da('"outdsØ"') f("jobw") shr reuse"
             sk.1=' SELECT * FROM VBPACK  ORDER BY 2,1 ;     '
        sk.Ø=1
        call Writeout
        call Rundb2
        macnr = @mdb2Ø16
        call Exmacro
        call Testout
        if build = yes then do
```

```
            call Wrbpack
            if #pk > Ø then do
              say '        '
              say '>>>>>>>>'
              say '>>>>>>>>  Warning there are no '#pk' packages
                               with member name PK??????'
              say '>>>>>>>>  because the name of Package is
                               not unique !!!!'
              say '>>>>>>>>'
              end
            say '        '
            say '>>>>>>>>'
            say '>>>>>>>>  +-----------------------------------+'
            say '>>>>>>>>    Total Bind Package no' ctrpk
            say '>>>>>>>>  +-----------------------------------+'
            say '>>>>>>>>'
            say '        '
            address tso "delete '"outdsØ"'"
            address tso "delete '"outds2"'"
            say '        '
            end
          else do
            say '        '
            say '>>>>>>>>'
            say '>>>>>>>> The select BPACK has Ø rows !!! '
            say '>>>>>>>> The 'outdspk' will be empty.    '
            say '>>>>>>>>'
            say '        '
            address tso "delete '"outdsØ"'"
            address tso "delete '"outds2"'"
            say '        '
            end
          end
       end
    end
 when ct1 = 3 then do
    call Allppk
    call Allpk
    call Allppm
    do #i = 1 to 3
       if #i = 1 then do
   /*-------------------------------------------------------*/
   /*-      Bind Plan with Package                  ---*/
   /*-------------------------------------------------------*/
          tipsel = BPPACK
          call Allsysp
          call Allsysi
          jobw = sysin
          "alloc da('"outdsØ"') f("jobw") shr reuse"
              sk.1=' SELECT * FROM VBPPACK ORDER BY 2,1 ;     '
```

```
         sk.Ø=1
         call Writeout
         call Rundb2
         macnr = @mdb2Ø18
         call Exmacro
         call Testout
         if build = yes then do
            call Wrbppack
            say '         '
            say '>>>>>>>>'
            say '>>>>>>>>  +-----------------------------------------+'
            say '>>>>>>>>    Total Bind Plan "Pklist" no' ctrppk
            say '>>>>>>>>  +-----------------------------------------+'
            say '>>>>>>>>'
            say '         '
            address tso "delete '"outdsØ"'"
            address tso "delete '"outds2"'"
            say '         '
            end
         else do
            say '         '
            say '>>>>>>>>'
            say '>>>>>>>> The select BPPACK has Ø rows !!! '
            say '>>>>>>>> The 'outdsppk' will be empty.    '
            say '>>>>>>>>'
            say '         '
            address tso "delete '"outdsØ"'"
            address tso "delete '"outds2"'"
            say '         '
            end
         end
      if #i = 2 then do
/*-------------------------------------------------------*/
/*-      Bind Package                                   -*/
/*-------------------------------------------------------*/
         tipsel = BPACK
         call Allsysp
         call Allsysi
         jobw = sysin
         "alloc da('"outdsØ"') f("jobw") shr reuse"
               sk.1=' SELECT * FROM VBPACK  ORDER BY 2,1 ;     '
         sk.Ø=1
         call Writeout
         call Rundb2
         macnr = @mdb2Ø16
         call Exmacro
         call Testout
         if build = yes then do
            call Wrbpack
            if #pk > Ø then do
```

```
            say '        '
            say '>>>>>>>>'
            say '>>>>>>>>  Warning there are no '#pk' packages
                                with member name PK??????'
            say '>>>>>>>>  because the name of Package is
                                not unique !!!!'
            say '>>>>>>>>'
            end
         say '        '
         say '>>>>>>>>'
         say '>>>>>>>>  +-----------------------------------+'
         say '>>>>>>>>    Total Bind Package no' ctrpk
         say '>>>>>>>>  +-----------------------------------+'
         say '>>>>>>>>'
         say '        '
         address tso "delete '"outdsØ"'"
         address tso "delete '"outds2"'"
         say '        '
         end
      else do
         say '        '
         say '>>>>>>>>'
         say '>>>>>>>> The select BPACK has Ø rows !!!'
         say '>>>>>>>> The 'outdspk' will be empty.    '
         say '>>>>>>>>'
         say '        '
         address tso "delete '"outdsØ"'"
         address tso "delete '"outds2"'"
         say '        '
         end
      end
   if #i = 3 then do
/*-------------------------------------------------------*/
/*-      Bind Plan with member (Mix Plan)  ---*/
/*-------------------------------------------------------*/
      tipsel = BPMEMB
      call Allsysp
      call Allsysi
      jobw = sysin
      "alloc da('"outdsØ"') f("jobw") shr reuse"
            sk.1=' SELECT * FROM VBPMEMB ORDER BY 1,2 ;   '
      sk.Ø=1
      call Writeout
      call Rundb2
      macnr = @mdb2Ø23
      call Exmacro
      call Testout
      if build = yes then do
         call Wrbpmemb
         say '        '
```

```
          say '>>>>>>>>'
          say '>>>>>>>>  +-------------------------------------+'
          say '>>>>>>>>    Total Bind Plan "DBRM" no' ctrppm
          say '>>>>>>>>  +-------------------------------------+'
          say '>>>>>>>>'
          say '        '
          address tso "delete '"outdsØ"'"
          address tso "delete '"outds2"'"
          say '        '
/*-------------------------------------------------------*/
/*-      File out compare allocation             ---*/
/*-------------------------------------------------------*/
          if yesppk = on then do
             call Alldelt
/*-------------------------------------------------------*/
/*-      List member .PLANPACK                   ---*/
/*-------------------------------------------------------*/
             xx=outtrap(trpdl.)
                address tso "listds '"outdsppk"' members "
             xx=outtrap(off)
             jobw = newdd
             "alloc da('"outdsupd"($PACK)') f("jobw") shr reuse"
             call wrcomp
/*-------------------------------------------------------*/
/*-      List member .PLAN                       ---*/
/*-------------------------------------------------------*/
             xx=outtrap(trpdl.)
                address tso "listds '"outdsppm"' members "
             xx=outtrap(off)
             jobw = olddd
             "alloc da('"outdsupd"($PLAN)') f("jobw") shr reuse"
             call wrcomp
/*-------------------------------------------------------*/
/*-      Compare member                          ---*/
/*-------------------------------------------------------*/
             "ispexec select pgm(isrsupc) parm(CHNGL,LINECMP)"
             if rc > 4 then do
                say '        '
                say '        '
                say '>>>>>>>>'
                say '>>>>>>>>  Compare dataset error '
                say '>>>>>>>>  RC = 'rc'. Verify.     '
                say '>>>>>>>>'
                say '        '
                say '        '
                call free
                exit
                end
/*-------------------------------------------------------*/
/*-    Macro to process out compare            ---*/
```

```
/*-------------------------------------------------------*/
            xx=OUTTRAP(trp03.)
               "ispexec edit dataset('"outdsdlt"') macro(@MDB2027)"
            xx=OUTTRAP(OFF)
            if rc > 0 then do
               do #a = 1 to trp03.0
                  say trp03.#a
                  end
               exit
               end
/*-------------------------------------------------------*/
/*-      Extract member name of mix plans  ---*/
/*-------------------------------------------------------*/
            xx=outtrap(trp04.)
               "execio * diskr outdd (stem outdd. finis"
               "free fi(outdd)"
            xx=outtrap(off)
            if rc > 0 then do
               do #a = 1 to trp04.0
                  say trp04.#a
                  end
               say '        '
               say '        '
               say '>>>>>>>>'
               say '>>>>>>>>  Error reading file "'outdsdlt'"'
               say '>>>>>>>>  RC='rc'. Verify.         '
               say '>>>>>>>>'
               say '        '
               say '        '
               exit
               end
            if outdd.0 = 0 then do
               say '        '
               say '        '
               say '>>>>>>>>'
               say '>>>>>>>>  The file "'outdsdlt'" is empty.    '
               say '>>>>>>>>  Probably error during compare !!!!'
               say '>>>>>>>>'
               say '        '
               say '        '
               exit
               end
/*-------------------------------------------------------*/
/*-      Member name library list           ---*/
/*-------------------------------------------------------*/
            #o = 0
            #k = 0
            sw = off
            do while mem_com ¬= Z9999999
               #k = #k + 1
```

```
                mem_com = word(outdd.#k,1)
                tst_com = substr(outdd.#k,2,4)
                if sw = on then do
                   if tst_com = blk then do
                      #o = #o + 1
                      arr.#o = mem_com
                      end
                   end
                if mem_com = $$$COIBM then
                   sw = on
                end
/*———————————————————————————————————————————*/
/*—      If there are no mix plans         ———*/
/*———————————————————————————————————————————*/
         select
            when #o = Ø | #o = 1 then do
            say '         '
            say '         '
            say '>>>>>>>>'
            say '>>>>>>>>  There are no mix plan (pklist + member). '
            say '>>>>>>>>'
            say '         '
            say '         '
            address tso "delete '"outdsupd"($PLAN)'"
            address tso "delete '"outdsupd"($PACK)'"
            address tso "delete '"outdsppk"(Z9999999)'"
            address tso "delete '"outdsppm"(Z9999999)'"
            end
           when #o > 1 then do
            say '>>>>>>>>'
            say '>>>>>>>>  Building mix plans (Member + pklist).'
            say '>>>>>>>>'
            say '         '
            do #z = 1 to #o - 1
               xx=OUTTRAP(trpØ5.)
                  "ispexec edit dataset('"outdsppm"("arr.#z")')
                  macro(@mdb2Ø21)"
               xx=OUTTRAP(OFF)
               if rc > Ø then do
                  do #a = 1 to trpØ5.Ø
                     say trpØ5.#a
                     end
                  exit
                  end
               "ispexec lminit dataid("inp1")
               dataset('"outdsppm"') enq(shr)"
               "ispexec lminit dataid("out1")
               dataset('"outdsppk"') enq(shr)"
               lwrk = length(arr.#z)
               movmemb = # || substr(arr.#z,2,lwrk)
```

```
               "ispexec lmmove fromid("inp1") frommem("arr.#z")
                             todataid("out1") tomem("movmemb") "
               xx=OUTTRAP(trpØ6.)
                  address ispexec 'vput (movmemb) profile'
                  "ispexec edit dataset('"outdsppk"("arr.#z")')
                  macro(@mdb2Ø24)"
               xx=OUTTRAP(OFF)
               if rc > Ø then do
                  do #a = 1 to trpØ6.Ø
                     say trpØ6.#a
                     end
                  exit
                  end
               end
            say '>>>>>>>>'
            say '>>>>>>>>  There are '#o - 1' mix plans.'
            say '>>>>>>>>'
            say '        '
            say '        '
/*———————————————————————————————————————————*/
/*–      List member .PLAN                  ———*/
/*———————————————————————————————————————————*/
             address tso "delete '"outdsupd"($PLAN)'"
             address tso "delete '"outdsupd"($PACK)'"
             address tso "delete '"outdsppk"(Z9999999)'"
             address tso "delete '"outdsppm"(Z9999999)'"
             xx=outtrap(trpdl.)
                address tso "listds '"outdsppm"' members "
             xx=outtrap(off)
             jobw = olddd
             "alloc da('"outdsupd"($PLAN)') f("jobw") shr reuse"
             #x  = 1
             #x1 = Ø
             #r  = Ø
             call hdrbnd
             sw= off
             do #h = 1 to trpdl.Ø
                if sw = on then do
                   wrk = word(trpdl.#h,1)
                   #r = #r + 1
                   if #r > 1 then do
                     if swsyspr = on then do sb.#x='//SYSTSIN
                          DD  DSN='outdsppm'('wrk'),DISP=SHR  '
                        #x = #x + 1
                        swsyspr = off
                        end
                      else do sb.#x='// DD  DSN='outdsppm
                                '('wrk'),DISP=SHR    '
                        #x = #x + 1
                        end
```

```
                        if #x1 > 8Ø then dosb.#x='
                              //* ————————————————————————— * '
                           #x = #x + 1
                           call hdrbnd
                           #x1 = Ø
                           end
                        #x1 = #x1 + 1
                        end
                     end
                  if trpdl.#h = '—MEMBERS—' then
                     sw = on
                  end sb.#x='//* ————————————————————————— *    '
               sb.Ø = #x
               jobw = fiupd
               "alloc da('"outdsupd"(bindppm)') f("jobw") shr reuse"
               call Wrindx
               if #r = 1 then do
                  address tso "delete '"outdsppm"'"
                  address tso "delete '"outdsupd"(bindppm)'"
                  address tso "delete '"outdsupd"(ipoupppm)'"
                  end
               end
            otherwise
               say '        '
               say '        '
               say '>>>>>>>>'
         say '>>>>>>>>  Unpredictable error. Stop elaboration !!!! '
               say '>>>>>>>>'
               say '        '
               say '        '
               call free
               exit
               end
            address tso "delete '"outdsdlt"'"
            end
         end
      else do
         say '        '
         say '>>>>>>>>'
         say '>>>>>>>> The select BPMEMB has Ø rows !!! '
         say '>>>>>>>> The 'outdsppm' will be empty.    '
         say '>>>>>>>>'
         say '        '
         address tso "delete '"outdsØ"'"
         address tso "delete '"outds2"'"
         say '        '
         end
      end
   end
end
```

```
otherwise
   say '        '
   say '        '
   say '>>>>>>>>'
   say '>>>>>>>>  Unpredictable OPER value. Stop elaboration !!!!! '
   say '>>>>>>>>'
   say '        '
   say '        '
end
call Free
exit
  /*-------------------------------------------------------*/
  /*-     File Plan allocation (whit packlist) -*/
  /*-------------------------------------------------------*/
Allppk:
   outdsppk= hiwork'.'subsys'.'creator'.PLANPACK'
   xx=outtrap(trp07.)
      address tso "delete '"outdsppk"'"
      "alloc da('"outdsppk"') dir(900) space(150,450) dsorg(ps)" ,
      "recfm(f,b) lrecl(80) blksize(27920) tracks ",
      "unit("esounit") release new catalog f(fippk) "
   xx=outtrap(off)
   if rc > 0 then do
      do #a = 1 to trp07.0
         say trp07.#a
         end
      say '        '
      say '        '
      say '>>>>>>>>'
      say '>>>>>>>> "'outdsppk'"  Allocation OK'
      say '>>>>>>>>  RC='rc'. Verify.          '
      say '>>>>>>>>'
      say '        '
      say '        '
      exit
      end
   else
      say '>>>>>>>>  'outdsppk '        Allocation OK  '
   return
  /*---------.-----------------------------------------*/
  /*-       File Package allocation          ---*/
  /*-------------------------------------------------------*/
Allpk:
   outdspk= hiwork'.'subsys'.'creator'.PACKAGE'
   xx=outtrap(trp08.)
      address tso "delete '"outdspk"'"
      "alloc da('"outdspk"') dir(2000) space(150,900) dsorg(ps)" ,
      "recfm(f,b) lrecl(80) blksize(27920) tracks ",
      "unit("esounit") release new catalog f(fipk) "
   xx=outtrap(off)
```

```
   if rc > Ø then do
      do #a = 1 to trpØ8.Ø
         say trpØ8.#a
         end
      say '         '
      say '         '
      say '>>>>>>>>'
      say '>>>>>>>> "'outdspk'"  Allocation OK'
      say '>>>>>>>>  RC='rc'. Verify.          '
      say '>>>>>>>>'
      say '         '
      say '         '
      exit
      end
   else
      say '>>>>>>>>  'outdspk '         Allocation OK  '
   return
  /*-------------------------------------------------------*/
  /*-  File Plan allocation (with member)     --*/
  /*-------------------------------------------------------*/
Allppm:
   outdsppm= hiwork'.'subsys'.'creator'.PLAN'
   xx=outtrap(trpØ9.)
      address tso "delete '"outdsppm"'"
      "alloc da('"outdsppm"') dir(9ØØ) space(15Ø,45Ø) dsorg(ps)" ,
      "recfm(f,b) lrecl(8Ø) blksize(2792Ø) tracks ",
      "unit("esounit") release new catalog f(fippm) "
   xx=outtrap(off)
   if rc > Ø then do
      do #a = 1 to trpØ9.Ø
         say trpØ9.#a
         end
      say '         '
      say '         '
      say '>>>>>>>>'
      say '>>>>>>>> "'outdsppm'"  Allocation OK'
      say '>>>>>>>>  RC='rc'. Verify.          '
      say '>>>>>>>>'
      say '         '
      say '         '
      exit
      end
   else
      say '>>>>>>>>  'outdsppm '            Allocation OK  '
   return
  /*-------------------------------------------------------*/
  /*-      File sysin allocation             ---*/
  /*-------------------------------------------------------*/
Allsysi:
   outdsØ= hiwork'.'subsys'.'creator'.'tipsel'.SYSIN'
```

```
   xx=outtrap(trp1Ø.)
      address tso "delete '"outdsØ"'"
      "alloc da('"outdsØ"') dir(Ø) space(5,1) dsorg(ps)" ,
      "recfm(f,b) lrecl(8Ø) blksize(2792Ø) tracks ",
      "unit("esounit") release new catalog f(sysin)"
   xx=outtrap(off)
   if rc > Ø then do
      do #a = 1 to trp1Ø.Ø
         say trp1Ø.#a
         end
      say '        '
      say '>>>>>>>>'
      say '>>>>>>>> "'outdsØ'"  Allocation OK'
      say '>>>>>>>>  RC='rc'. Verify.          '
      say '>>>>>>>>'
      say '        '
      say '        '
      exit
      end
   else do
      say '>>>>>>>>  'outdsØ '    Allocation OK  '
      say '          '
      end
   return
  /*———————————————————————————————————————————*/
  /*—    File sysprint allocation            ———*/
  /*———————————————————————————————————————————*/
Allsysp:
   outds2= hiwork'.'subsys'.'creator'.'tipsel'.SYSPRINT'
   xx=outtrap(trp11.)
      address tso "delete '"outds2"'"
      "alloc da('"outds2"') dir(Ø) space(35Ø,35Ø) dsorg(ps)" ,
      "recfm(f,b,a) lrecl(121) blksize(121Ø)  tracks ",
      "unit("esounit") release new catalog f(sysprint) "
   xx=outtrap(off)
   if rc > Ø then do
      do #a = 1 to trp11.Ø
         say trp11.#a
         end
      say '        '
      say '        '
      say '>>>>>>>>'
      say '>>>>>>>> "'outds2'"  Allocation OK'
      say '>>>>>>>>  RC='rc'. Verify.          '
      say '>>>>>>>>'
      say '        '
      say '        '
      exit
      end
   else
```

```
      say '>>>>>>>>  'outds2 ' Allocation OK  '
   return
  /*-------------------------------------------------------*/
  /*-      Run DB2                                  ---*/
  /*-------------------------------------------------------*/
Rundb2 :
   wrk = center(tipsel,8)
   say ' ********************************************************** '
   say '*  ' #i '                                        'time()'     *'
   say '*      The query 'wrk    'is running                          *'
   say '*                                                            *'
   say '*               Please Stand by ......                       *'
   say '*                                                            *'
   say ' ********************************************************** '
   say '                                                               '
   xx=outtrap(trp12.)
     address tso "ex '"outds1"'"
   xx=outtrap(off)
   if rc > 0 then do
      analisi = substr(trp12.1,1,8)
      if analisi = 'DSNE100I' then do
         subs = center(subsys,10)
         say '      '
         say '      '
         say '>>>>>>'
         say '>>>>>>'
         say '>>>>>> The DB2 subsystem ->'subs'<- is not active'
         say '>>>>>>'
         say '>>>>>>'
         say '      '
         say '      '
         end
      else do
         do #a = 1 to trp12.0
            say trp12.#a
            end
         end
      exit
      end
      say '>>>>>>>>  DB2 query terminated. '
      say '       '
  /*-------------------------------------------------------*/
  /*-      Print output DB2 query           ---*/
  /*-------------------------------------------------------*/
   address tso "pr dataset('"outds2"') class(R) writer(n0z2) fcb(6) "
   return
  /*-------------------------------------------------------*/
  /*-    Macro to process output of sysprint  --*/
  /*-------------------------------------------------------*/
Exmacro:
```

```
   xx=OUTTRAP(trp13.)
      "ispexec edit dataset('"outds2"') macro("macnr")"
   xx=OUTTRAP(OFF)
   if rc > 0 then do
      do #a = 1 to trp13.0
         say trp13.#a
         end
      exit
      end
   return
  /*-------------------------------------------------------*/
  /*-      Test output DB2 query                  ---*/
  /*-------------------------------------------------------*/
Testout:
   build = yes
   xx=outtrap(trp14.)
      "execio * diskr sysprint (stem sysprint. finis"
      "free fi(sysprint)"
   xx=outtrap(off)
   if rc > 0 then do
      do #a = 1 to trp14.0
         say trp14.#a
         end
      say '        '
      say '        '
      say '>>>>>>>>'
      say '>>>>>>>>  Error reading file "'outds2'"'
      say '>>>>>>>>  RC='rc'. Verify.             '
      say '>>>>>>>>'
      say '        '
      say '        '
      exit
      end
   if sysprint.0 = 0 then do
      say '        '
      say '        '
      say '>>>>>>>>'
      say '>>>>>>>>  The file "'outds2'" is empty.    '
      say '>>>>>>>>  Probably error accessing DB2 !!!!'
      say '>>>>>>>>'
      say '        '
      say '        '
      exit
      end
   if substr(sysprint.1,33,8) = ' 0 ROW(S)' then
      build = no
   return
  /*-------------------------------------------------------*/
  /*-      Build members Bind Package         ---*/
  /*-------------------------------------------------------*/
```

```
 Wrbpack:
    say '>>>>>>>>  Building library 'outdspk
    jobw = fipk
    "alloc da('"outdspk"($$$coibm)') f("jobw") shr reuse"
sk.1='     ////////////////////////////////////////////////////// '
sk.2='     ****** Do not erase this member !!!!     Thanks.  ****** '
sk.3='     ////////////////////////////////////////////////////// '
    sk.Ø = 3
    call Writeout
    #pk = Ø
    #x  = 1
    #x1 = Ø
    Call Hdrbnd
    DO #b = 1 to sysprint.Ø
       pak_pk = word(sysprint.#b,1)
       col_pk = word(sysprint.#b,2)
       own_pk = word(sysprint.#b,3)
       qua_pk = word(sysprint.#b,4)
       val_pk = word(sysprint.#b,5)
       exp_pk = word(sysprint.#b,6)
       pds_pk = word(sysprint.#b,7)
       iso_pk = word(sysprint.#b,8)
       rel_pk = word(sysprint.#b,9)
       if substr(sysprint.#b,122,1) = blk then
          iso_pk = S
       if substr(sysprint.#b,127,1) = blk then
          rel_pk = C
   /*———————————————————————————————————————*/
   /*–      VALIDATE  "Bind/Run"             ——*/
   /*———————————————————————————————————————*/
       if val_pk = B then
          valid = BIND
       else
          valid = RUN
   /*———————————————————————————————————————*/
   /*–  ISOLATION "Rep.Read/Cursor Stability   –*/
   /*———————————————————————————————————————*/
       if iso_pk = R then
          isol  = RR
       else
          isol  = CS
```

Editor's note: this article will be continued next month.

Giuseppe Rendano
DB2 Systems Programmer (Italy)

IBM has announced Version 1.5 of DFSMS/MVS, promising less overhead and redundancies than in earlier versions of HFS. In this new release, OAM will exploit sysplex architecture and DB2 data sharing, and will remove the 100 storage group limitation in a configuration. Users can now specify their own high-level qualifier for the object groups and their associated DB2 tables.

OAM will also exploit sysplex architecture to enhance availability and allow instances of OAM which belong to an XCF group and are connected to DB2 subsystems, using DB2 data sharing to have full access to OAM objects.

For further information contact your local IBM representative.

* * *

ETI has announced Data System Library (DSL) for COBOL/DB2 Release 1.2.1. Developed jointly with IBM, DSL provides support for IBM DB2 Universal Database (including the Extended Enterprise Edition with parallel loading), DB2 for OS/400, DB2 for MVS and OS/390, with support for parallel sysplex. It offers the ability to load data into any DB2-based application or data warehouse from any number of different sources. DSL can be used in conjunction with The MetaTransport Utility for Release 1.0.0 which integrates the ETIïExtract Tool Suite with IBM's Visual Warehouse datamart and data warehouse software.

For further information contact:
ETI, 66 Bovet Road, Suite 320, San Mateo, CA 94402, USA.
Tel: (650) 345 9100.
URL: http://www.etiusa.com.

* * *

IBM is to include Platinum's ERwin 3.5 DB2 data modelling tool, for the design, generation, and maintenance of database applications, as part of its VisualAge family. The companies plan to sell an enhanced version of ERwin, by the end of the year, that will integrate additional functions and features of VisualAge DataAtlas.

Further collaboration is to focus on an open, industry standard XML interface, which will integrate ERwin into VisualAge through TeamConnection, IBM's workgroup software. This will give development teams the ability to model databases with ERwin and store and maintain them in the VisualAge TeamConnection repository where other application code resides.

For further information contact your local IBM representative.

* * *