66

DB2

April 1998

In this issue

update

DB2 Update

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

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

Australian office
Xephon/RSM
GPO Box 6258
Halifax Street
Adelaide, SA 5000
Australia
Telephone: 088 223 1391

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

Editor
Trevor Eddolls

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.

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.

Printed in England.

Buffer pool residency

There have been articles in several publications recently about tuning buffer pools and calculating buffer pool hit rates. I have always found it difficult to relate to buffer pool hit ratios so I decided to look at buffer pool residency, ie how long a page remains in the pool since it was last referenced. Research by others indicates that five minutes' residency gives the best trade off.

The DISPLAY BUFFERPOOL command with the LSTATS option provides details of how many pages are in the virtual and hiper pools. For example Figure 1 shows a display showing DSNSPT01 with no pages in the virtual pool and 212 in the hiper pool.

```
-DISPLAY BUFFERPOOL(BP0) LIST(ACTIVE) LSTATS
:
:
DSNB451I -DB2P INDEXSPACE = DSNDB01.DSNSPT01, USE COUNT = 1, GBP-DEP= N
DSNB452I -DB2P  STATISTICS FOR DATASET 1 -
DSNB453I -DB2P   VP CACHED PAGES -
CURRENT       =     0   MAX             =   216
CHANGED       =     0   MAX             =     0
DSNB454I -DB2P   HP CACHED PAGES -
CURRENT       =   212   MAX             =   313
DSNB455I -DB2P   SYNCHRONOUS I/O DELAYS -
AVERAGE DELAY =    19   MAXIMUM DELAY   =    63
TOTAL PAGES   =   517
DSNB456I -DB2P   ASYNCHRONOUS I/O DELAYS -
AVERAGE DELAY =     4   MAXIMUM DELAY   =     4
TOTAL PAGES   =    31   TOTAL I/O COUNT =     1
```

Figure 1: DSNSPT01

If you read a page from a tablespace which no-one else is accessing and then issue repeated display commands, it is possible to see the page move from the virtual pool to the hiper pool and then be discarded. The pagesets are displayed in most recently opened order, so stopping and starting the tablespace before reading it puts it at the top of the queue.

BPCK ROUTINE

I have written an ISPF application to automate the above checking process. The application does the following:

- Checks that a dummy table and tablespace have been defined for the buffer pool. If they do not exist, they are created.
- The tablespace is stopped and started to put it at the top of the display list.
- A SELECT * from the table is run.
- A DISPLAY BUFFERPOOL is processed every ‘n’ seconds.
- The statistics are written out to a history dataset.

SAMPLE SCREENS

The command in progress looks like:

```
+------------- BUFFER POOL RESIDENCY --------------------------+
|                                                              |
|       Subsystem Name  :      DB2P                            |
|       Sample Interval :      20      Secs                    |
|       Bufferpool Name :      BP1                             |
|       Batch Cycles    :      0        (Zero to run on-line)  |
|                                                              |
|       In Virtual Pool :         0       -        23      Secs|
|       In Hiper Pool   :         0       -                Secs|
|                   +-----------------------------------------+
|       Edit History fi |      45 SECS ELAPSED                 |
|                           +--------------------------------+
+--------------------------------------------------------------+
```

The history dataset looks like:

```
EDIT ---- DMG.BPCK.HISTORY --------------------------- Columns 001  072
Command ===>
Scroll ===>      CSR

         DB2    Buffer Virt. Pool Hiper Pool Min. Time
        System   Pool    Size       Size      VP   HP  Timestamp
       -------- ------ ---------- ---------- ---- ---- -----------------
--
000014    DB2P   BP1        5000    10000    0  173 97/12/18-10:40:58
000015    DB2P   BP1        5000    10000   44  486 97/09/19-15:01:56
000016    DB2P   BP1        5000    10000  178  488 97/09/19-14:41:51
```

```
000017    DB2P   BP1        5000    10000  178  488 97/09/19-14:41:51
000018    DB2P   BP1        5000    10000   23  278 97/09/19-10:26:51
000019    DB2P   BP1        5000    10000   45  158 97/09/19-10:19:44
000020    DB2P   BP1        5000    10000   88  796 97/09/19-10:15:51
000021    DB2P   BP1        5000    10000   88  796 97/09/19-10:15:51
000022    DB2P   BP1        5000    10000  153  485 97/09/19-09:49:47
```

NOTES

This routine has only been tested with DB2 Version 4.1.

The accuracy of the results is limited by the delay interval and the time it takes to process the data. I normally use a delay of 20 seconds. It also takes approximately three seconds to issue the DISPLAY command and process the results.

The delay is generated by the WAIT command, which is a feature of REXXTOOLS. If you do not use REXXTOOLS, then you will have to substitute your own version of WAIT – most sites seem to have one but, if you don't, look on the Xephon Web pages for *MVS Update*. There are a couple of routines that you can download free that offer a delay timer facility (one is called WAIT).

This was written originally as an ISPF application, but after running it a few times I got bored with watching the screen, so I added an option to run it in batch with an iteration and to browse the results without running the check.

The results dataset can be edited under normal SPF edit, but when invoked from BPCK there is a modified edit panel with column headings and an initial macro which positions you at the first line for the specified subsystem and bufferpool.

INSTALLATION

Copy the install job and make the following modifications:

- Change USERID to your user-id.
- Change YOUR DB2 LOADLIB to the DB2 load library.
- Change the line containing DB2A,DB2B,DB2C to list valid subsystems for your installation.

Modify the jobcard and run the job in the following section. BPCK.DATA is the history dataset and BPCK.SOURCE is the installation library. It is possible to run from the installation library by issuing the following two commands:

```
EXEC 'USERID.BPCK.SOURCE(BPCKLBDF)'
EXEC 'USERID.BPCK.SOURCE(BPCK)'
```

THE INSTALL JOB

```
//YOUR JOBCARD,'BPCHK',                  <==== CHANGE THIS
// CLASS=A,MSGCLASS=X
//*
//STEP1  EXEC PGM=IEFBR14
//DD1  DD DSN=USERID.BPCK.DATA,          <==== CHANGE THIS
//        DISP=(,CATLG),
//        LRECL=8Ø,
//        BLKSIZE=616Ø,RECFM=FB,
//        UNIT=SYSALLDA,
//        SPACE=(TRK,1)
//*
//STEP2  EXEC PGM=IEBUPDTE,PARM=NEW
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD DSN=USERID.BPCK.SOURCE,    <==== CHANGE THIS
//            DISP=(,CATLG),
//            SPACE=(TRK,(1,1,9)),
//            LRECL=8Ø,
//            BLKSIZE=616Ø,
//            RECFM=FB,
//            UNIT=SYSALLDA
//SYSIN    DD DATA,DLM=##
./  ADD NAME=BPCHECK
/******************** REXX ***************************************/
/* FUNCTION:  CHECK BUFFER POOL RESIDENCY                        */
/*                                                               */
/* USES:                                                         */
/*   CLIB: BPCKMAC1 BPCKMAC2 BPCKMSUB                            */
/*   SLIB: BPCKBAT  BPCKCR8  BPCKSQL                             */
/*   PLIB: BPCK     BPCKED   BPCKEDH   BPCKJCD  BPCKHLP          */
/*                                                               */
/* DESCRIPTION:                                                  */
/*   THIS UTILITY READS A DUMMY TABLE TO LOAD ITS PAGES INTO THE  */
/*   BUFFER POOL. IT THEN ISSUES THE DISPLAY BUFFER POOL COMMAND  */
/*   WITH THE LSTATS OPTION AND ANALYSES THE OUTPUT TO CHECK THAT */
/*   THE PAGES ARE STILL IN THE POOL. THIS IS REPEATED UNTIL THE  */
/*   PAGES ARE FLUSHED FROM THE POOL. THE TIME THE PAGE REMAINED IN */
/*   THE  VIRTUAL POOL AND HIPER POOL ARE WRITTEN TO A HISTORY FILE. */
/*                                                               */
/*****************************************************************/
```

```
TRACE O

ZERRLM = ''
"ISPEXEC VGET ZENVIR SHARED"
PARSE VAR ZENVIR . 17 MODE 24 .
IF MODE = 'BATCH'
THEN DO
       ARG BPCKSYS BPCKSAMP BPCKPOOL CYCLE
       "ISPEXEC VPUT (BPCKSYS BPCKSAMP BPCKPOOL) PROFILE"
     END
LOOP = Ø
DO FOREVER
   IF MODE = 'BATCH'
   THEN DO
          IF LOOP = CYCLE
          THEN EXIT
          LOOP = LOOP + 1
        END
   RESTART = 'N'
   CALL DISPLAY
   IF RC > Ø
   THEN EXIT
   LOADLIB = 'YOUR DB2 LOADLIB' /* <==== CHANGE THIS     */
   IF BPCKED = 'Y'
   THEN DO
           "ISPEXEC REMPOP"
           "ISPEXEC EDIT DATASET('"USERID.BPCK.DATA"')",
                         "PANEL(BPCKEDH) MACRO(BPCKMAC1)"
           RESTART = 'Y'
        END
   ELSE  CALL STARTTS
   IF RESTART = 'N'
          THEN IF BPCKCYC > Ø
               THEN CALL BATCH
          ELSE DO
                 CALL READTB
                 CALL TIME('R')
                 NEWTIME = Ø
                 CALL DISBP
                 ZERRSM = ''
                 ZERRWN = 'NORESP'
                 IF RESULT = Ø
                 THEN DO
                        ZERRLM = 'TO SAVE TO HISTORY FILE',
                                 'AND EDIT PRESS ENTER, OR PF3 TO QUIT'
                        ZERRTP = 'WARNING'
                        "ISPEXEC DISPLAY PANEL(BPCK)",
                                        "MSG(ISRZØØ3) MSGLOC("LOC")"
                        IF RC = Ø
                        THEN DO
```

```
                                "ISPEXEC REMPOP"
                                ROW = LEFT(BPCKSYS,8)
                                ROW = ROW LEFT(BPCKPOOL,6)
                                ROW = ROW RIGHT(STRIP(BPCKVPSZ),1Ø)
                                ROW = ROW RIGHT(STRIP(BPCKHPSZ),1Ø)
                                ROW = ROW RIGHT(STRIP(BPCKVP1),4)
                                ROW = ROW RIGHT(STRIP(BPCKHP1),4)
                                ROW = ROW DATE('O')'-'TIME()
                                "ISPEXEC VPUT (ROW MODE) SHARED"
                                "ISPEXEC EDIT",
                                        "DATASET('"USERID.BPCK.DATA"')",
                                        "PANEL(BPCKEDH) MACRO(BPCKMAC1)"
                             END
                        END
                   ELSE DO
                          ZERRLM = 'TABLE NOT FOUND IN BUFFER POOL',
                                   'PRESS ENTER OR PF3 TO QUIT'
                          ZERRTP = 'WARNING'
                          "ISPEXEC DISPLAY PANEL(BPCK)",
                                          "MSG(ISRZØØ3)",
                                          "MSGLOC("LOC")"
                        END
                     ZERRLM = ''
                  END
END
EXIT
DISPLAY:
"ISPEXEC ADDPOP ROW(3) COLUMN(12)"
BPCKHP1 = Ø
BPCKHP2 = ''
BPCKVP1 = Ø
BPCKVP2 = ''
ZWINTTL = 'BUFFER POOL RESIDENCY'
IF ZERRLM = ''
THEN ZERRLM = 'SUPPLY VALUES FOR DB2 SUBSYSTEM, SAMPLE DELAY,',
              'BUFFERPOOL AND PRESS ENTER TO RUN.'
ZERRWN = 'NORESP'
ZERRTP = 'NOTIFY'
ZERRALRM = 'NO'
ZERRHM =  '*'
LOC = 'BPCKVP1'
"ISPEXEC DISPLAY PANEL(BPCK) CURSOR(BPCKSYS) MSG(ISRZØØ3)"
RETURN
BATCH:
 "ISPEXEC VGET (BPCKJOB1 BPCKJOB2 BPCKJOB3 BPCKJOB4) PROFILE"
 "ISPEXEC ADDPOP"
 ZERRLM = 'SUPPLY A VALID JOBCARD FOR YOUR INSTALLATION',
              'AND PRESS PF3 TO CONTINUE.'
 ZWINTTL = 'ENTER / MODIFY JOBCARD INFORMATION'
 DO UNTIL RC > Ø
```

```
    "ISPEXEC DISPLAY PANEL(BPCKJCD) MSG(ISRZØØ3)"
 END
 BPCKPARM = BPCKSYS BPCKSAMP BPCKPOOL BPCKCYC
 "ISPEXEC FTOPEN TEMP"
 "ISPEXEC FTINCL BPCKBAT"
 "ISPEXEC FTCLOSE"
 "ISPEXEC REMPOP"
 "ISPEXEC REMPOP"
 "ISPEXEC VGET (ZTEMPF)"
 "ISPEXEC EDIT DATASET('"ZTEMPF"') MACRO(BPCKMSUB)"
 "FREE DA('"ZTEMPF"')"
 RESTART = 'Y'
RETURN
DISBP:
ZERRSM = '** CHECK IN PROGRESS **'
ZERRWN = 'NORESP'
ZERRTP = 'NOTIFY'
LOC = 'BPCKVP1'
"ISPEXEC CONTROL DISPLAY LOCK"
"ISPEXEC DISPLAY PANEL(BPCK) MSG(ISRZØØ3) MSGLOC("LOC")"
FIRSTIME = 'Y'
DO UNTIL BPCKHP2 ¬= ""
   CALL WAIT 'SEC',BPCKSAMP
   X = OUTTRAP('L.','*','NOCONCAT')                    /* DISPLAY BP */
   QUEUE '-DIS BUFFERPOOL ('BPCKPOOL') DETAIL(INTERVAL) LIST(*) LSTATS'
   QUEUE 'END'
   "DSN SYSTEM("BPCKSYS")"
   X = OUTTRAP('OFF')
   IF FIRSTIME = 'Y'
   /* PICK UP POOL SIZES */
   THEN DO I = 1 TO L.Ø
          PARSE VAR L.I MSG .
          SELECT
            WHEN MSG = 'DSNB4Ø2I'
              THEN PARSE VAR L.I . '=' BPCKVPSZ .
            WHEN MSG = 'DSNB4Ø3I'
              THEN PARSE VAR L.I . '=' BPCKHPSZ .
            WHEN (MSG = 'DSNB45ØI' | MSG = 'DSNB451I')
              THEN DO
                     PARSE VAR L.I . '.' TSNAME ',' . '= ' USE ',' .
                     IF LEFT(TSNAME,8) = "TS"BPCKPOOL
                     THEN DO
                            CALL STATS
                            LEAVE
                          END
                   END
            OTHERWISE NOP
          END /* OF SELECT */
        END
   ELSE DO I = 1 TO L.Ø
```

```
          PARSE VAR L.I MSG .
          IF (MSG = 'DSNB45ØI' | MSG = 'DSNB451I')
          THEN DO
                 PARSE VAR L.I . '.' TSNAME ',' . '= ' USE ',' .
                 IF LEFT(TSNAME,8) = "TS"BPCKPOOL
                 THEN DO
                        CALL STATS
                        LEAVE
                      END
               END
        END
   FIRSTIME = 'N'
   IF I = L.Ø
   THEN RETURN 8
END
RETURN Ø
STATS:
HPPAGES = Ø
VPPAGES = Ø
J = I+2
PARSE VAR L.J MSG .
IF MSG = 'DSNB453I'   /* VP */
THEN DO
       J = J+1
       PARSE VAR L.J . . VPPAGES .
       J = J+2
       PARSE VAR L.J MSG .
     END
IF MSG = 'DSNB454I'   /* HP */
THEN DO
       J = J+1
       PARSE VAR L.J . . HPPAGES .
     END
OLDTIME = NEWTIME
NEWTIME = FORMAT(TIME('E'),4,Ø)
IF VPPAGES < OLDVPP
THEN DO
           /* FLUSHED FROM VIRTUAL POOL */
       BPCKVP1 = OLDTIME
       BPCKVP2 = NEWTIME
       ZERRSM = NEWTIME 'SECS ELAPSED'
       OLDVPP = VPPAGES
       OLDHPP = 1
       LOC = 'BPCKHP1'
     END
ELSE IF HPPAGES < OLDHPP
     THEN DO
             /* FLUSHED FROM HIPER POOL */
          OLDHPP = HPPAGES
          BPCKHP1 = OLDTIME
```

```
            BPCKHP2 = NEWTIME
            ZERRSM = NEWTIME 'SECS ELAPSED'
          END
     ELSE ZERRSM = NEWTIME 'SECS ELAPSED'
"ISPEXEC CONTROL DISPLAY LOCK"
"ISPEXEC DISPLAY PANEL(BPCK) MSG(ISRZØØ3) MSGLOC("LOC")"
RETURN
READTB:
"ISPEXEC FTOPEN TEMP"
"ISPEXEC FTINCL BPCKSQL"
"ISPEXEC FTCLOSE"
"ISPEXEC VGET (ZTEMPF)"
"ALLOC F(SYSIN) DA('"ZTEMPF"') SHR REUSE"
"FREE F(SYSPRINT)"
"ALLOC F(SYSPRINT) UNIT(VIO) NEW DELETE",
       "CYLINDER SPACE(1,9) LRECL(1Ø24) BLKSIZE(2Ø48Ø)",
       "RECFM(F,B)"
QUEUE "RUN PROGRAM(DSNTEP2) PLAN(DSNTEP41)",
          "LIBRARY('"LOADLIB"')"
QUEUE "END"
QUEUE ""
"DSN SYSTEM("BPCKSYS")"
      /* INITIALIZE TO 1 PAGE IN VIRTUAL POOL, NONE IN HIPER */
OLDVPP = 1
OLDHPP = Ø
"FREE F(SYSPRINT)"
"ALLOC F(SYSPRINT) DSNAME(*)"
"FREE F(SYSIN)"
RETURN
STARTTS:
/* STOP/START TS'S */
X = OUTTRAP('L.','*','NOCONCAT')
QUEUE '-STOP DB(DSNDBØ4) SPACENAM(TS'BPCKPOOL')'
QUEUE '-START DB(DSNDB04) SPACENAM(TS'BPCKPOOL')'
QUEUE 'END'
"DSN SYSTEM("BPCKSYS")"
X = OUTTRAP('OFF')
PARSE VAR L.1 MSGID .
SELECT
WHEN MSGID = 'DSNT3Ø2I'
  THEN DO
         "ISPEXEC FTOPEN TEMP"
         "ISPEXEC FTINCL BPCKCR8"
         "ISPEXEC FTCLOSE"
         "ISPEXEC REMPOP"
         "ISPEXEC VGET (ZTEMPF)"
         "ISPEXEC EDIT DATASET('"ZTEMPF"')",
                      "PANEL(BPCKED) MACRO(BPCKMAC2)"
         "ALLOC F(SYSIN) DA('"ZTEMPF"') SHR REUSE"
         "FREE F(SYSPRINT)"
```

```
         "ALLOC F(SYSPRINT) UNIT(VIO) NEW DELETE",
                "CYLINDER SPACE(1,9) LRECL(1Ø24) BLKSIZE(2Ø48Ø)",
                "RECFM(F,B,A)"
         QUEUE "RUN PROGRAM(DSNTEP2) PLAN(DSNTEP41)",
                  "LIBRARY('"LOADLIB"')"
         QUEUE "END"
         QUEUE ""
         "DSN SYSTEM("BPCKSYS")"
         "ISPEXEC LMINIT DATAID(BPCKSQL) DDNAME(SYSPRINT)"
         "ISPEXEC BROWSE DATAID("BPCKSQL") PANEL(BPCKBR)"
         "FREE F(SYSPRINT)"
         "ALLOC F(SYSPRINT) DSNAME(*)"
         "FREE F(SYSIN)"
         RESTART = 'Y'
       END
WHEN MSGID = 'DSNE11ØE'
  THEN DO
         ZERRLM = 'SUBSYSTEM' BPCKSYS 'NOT FOUND, TRY AGAIN'
         "ISPEXEC REMPOP"
         RESTART = 'Y'
         DELSTACK
       END
OTHERWISE NOP
END
RETURN
./  ADD NAME=BPCKMAC1
/****************** REXX ***************************************/
/*                                                               */
/* FUNCTION: BUFFER POOL RESIDENCY CHECK                         */
/*                                                               */
/* DESCRIPTION:                                                  */
/* THIS MACRO INSERTS THE LATEST RESULT INTO THE HISTORY DATASET */
/* AND SORTS THE DATA INTO SUBSYSTEM, BUFFERPOOL, TIMESTAMP      */
/* (DESCENDING) ORDER. IT THEN POSITIONS THE LATEST DATA AT THE  */
/* TOP OF THE SCREEN.  THE EDIT PANEL USED IS CUSTOMIZED TO      */
/* DISPLAY HEADINGS FOR THE DATA.                                */
/*                                                               */
/* IF THE ROUTINE HAS BEEN RUN IN BATCH, WRITE THE OUTPUT TO     */
/* SYSOUT AND SAVE IN THE HISTORY DATASET.                       */
/*                                                               */
/*****************************************************************/
"ISREDIT MACRO"
TRACE O
"ISPEXEC VGET (ROW MODE BPCKPOOL BPCKSYS) ASIS"
   /* IF NOT EDIT ONLY, INSERT A ROW */
IF ROW ¬= ' '
THEN "ISREDIT LINE_AFTER Ø = '"ROW"'"
"ISREDIT SORT 1 8 A 1Ø 15 A 49 65 D"
"ISREDIT FIND" BPCKSYS "FIRST"
"ISREDIT FIND" BPCKPOOL
```

```
"ISREDIT LOCATE" .ZCSR
IF MODE = 'BATCH'
THEN DO
       SAY ROW
       "ISREDIT END"
     END
EXIT
./  ADD NAME=BPCKMAC2
/********************** REXX ***************************************/
/*                                                                   */
/* FUNCTION: BUFFER POOL RESIDENCY CHECK                             */
/*                                                                   */
/* DESCRIPTION:                                                      */
/* ADD A HEADER OF NOTE LINES TO INFORM THE USER ABOUT CREATING      */
/* A NEW TABLESPACE AND TABLE.                                       */
/*                                                                   */
/*********************************************************************/
"ISREDIT MACRO"
TRACE O
LINE1 = "'"CENTRE(" CREATE DDL",72,"*")"'"
LINE2 = "'*"CENTRE("A NEW TABLESPACE AND TABLE ARE NEEDED",7Ø)"*'"
LINE3 = "'*"CENTRE("MODIFY THE DDL IF REQUIRED AND PF3 TO",7Ø)"*'"
LINE4 = "'*"CENTRE("RUN IT, THEN TRY AGAIN.",7Ø)"*'"
LINE5 = COPIES("*",72)
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE5
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE4
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE3
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE2
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE1
RETURN
./  ADD NAME=BPCKMSUB
/********************** REXX ***************************************/
/*                                                                   */
/* FUNCTION: BUFFER POOL RESIDENCY CHECK                             */
/*                                                                   */
/* DESCRIPTION:                                                      */
/*                                                                   */
/* ADD SOME NOTE LINES AT THE TOP OF THE GENERATED JCL TO GIVE       */
/* THE USER SOME INSTRUCTIONS.                                       */
/*                                                                   */
/*********************************************************************/
"ISREDIT MACRO"
TRACE O
LINE1 = "'"CENTRE(" JOB SUBMISSION ",72,"*")"'"
LINE2 = "'*"CENTRE("USE THE SUBMIT COMMAND TO RUN THE JOB,",7Ø)"*'"
LINE3 = "'*"CENTRE("CREATE COMMAND TO SAVE TO A PDS MEMBER,",7Ø)"*'"
LINE4 = "'*"CENTRE("OR PF3 TO QUIT",7Ø)"*'"
LINE5 = COPIES("*",72)
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE5
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE4
```

```
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE3
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE2
"ISREDIT LINE_AFTER Ø = MSGLINE" LINE1
RETURN
./  ADD NAME=BPCKBAT
)CM
)CM BUFFER POOL RESIDENCY CHECK
)CM     USED BY REXX:    BPCK
)CM                                C.ALLEN
)CM
&BPCKJOB1
&BPCKJOB2
&BPCKJOB3
&BPCKJOB4
//*
//STEP1    EXEC PGM=IKJEFTØ1,REGION=ØM
//SYSEXEC  DD  DSN=USERID.CLIB,DISP=SHR
//SYSTSPRT DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//ISPLOG   DD  DUMMY
//SYSUDUMP DD  DUMMY
//*******************************************************************/
//*    POSITIONAL PARAMETERS:                                       */
//*                                                                 */
//*               SUBSYSTEM NAME                                    */
//*               CHECK INTERVAL                                    */
//*               BUFFERPOOL                                        */
//*               NUMBER OF CYCLES                                  */
//*                                                                 */
//*******************************************************************/
//SYSTSIN  DD  *
  PROFILE PREFIX(DMG)
  ISPSTART CMD(%BPCK &BPCKPARM) BREDIMAX(9) BDISPMAX(5ØØ)
//ISPPROF  DD DSN=DMG.ISPF.PROFILE,DISP=(MOD,DELETE,DELETE),
//            SPACE=(TRK,(1,1,5)),
//            UNIT=SYSDA,
//            LRECL=8Ø,BLKSIZE=312Ø,RECFM=FB
//ISPPLIB  DD DSN=USERID.PLIB,DISP=SHR
//         DD DSN=SYS3.MODIFIED.PLIB,DISP=SHR
//         DD DSN=ISF.V1R6MØ.SISFPLIB,DISP=SHR
//         DD DSN=ISP.SISPPENU,DISP=SHR
//ISPMLIB  DD DSN=SYS3.MODIFIED.MLIB,DISP=SHR
//         DD DSN=ISF.V1R6MØ.SISFMLIB,DISP=SHR
//         DD DSN=ISP.SISPMENU,DISP=SHR
//ISPSLIB  DD DSN=USERID.SLIB,DISP=SHR
//         DD DSN=SYS3.MODIFIED.SLIB,DISP=SHR
//         DD DSN=ISP.SISPSENU,DISP=SHR
//         DD DSN=ISP.SISPSLIB,DISP=SHR
//ISPTLIB  DD DSN=ISF.V1R6MØ.SISFTLIB,DISP=SHR
//         DD DSN=ISP.SISPTENU,DISP=SHR
```

```
./  ADD NAME=BPCKCR8
SET CURRENT SQLID = 'DB2ISADM'
;
CREATE TABLESPACE TS&BPCKPOOL
       IN DSNDBØ4
       BUFFERPOOL &BPCKPOOL
       CLOSE YES
;
CREATE TABLE BPCK_&BPCKPOOL
       ( DUMMY CHAR)
       IN DSNDBØ4.TS&BPCKPOOL
;
./  ADD NAME=BPCKSQL
SELECT * FROM DB2ISADM.BPCK_&BPCKPOOL
;
./  ADD NAME=BPCK
)ATTR DEFAULT(%+_)
       %   TYPE(TEXT) INTENS(HIGH)
       +   TYPE(TEXT) INTENS(LOW)
       #   TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(lefT)
       _   TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(RIGHT)
       !   TYPE(TEXT) INTENS(LOW) SKIP(ON)
)BODY EXPAND(||) WINDOW(5Ø,11) cmd()
+
+ Subsystem Name  :#z   !
+ Sample Interval :#z !Secs
+ Bufferpool Name :#z     !
+ Batch Cycles    :#z + (Zero to run Online)
+
+ In Virtual Pool :_z   % -_z   %Secs
+ In Hiper Pool   :_z   % -_z   %Secs
+
+ Edit History file without running:_z! (Y/blank)
)INIT
  .ZVARS = '(BPCKSYS BPCKSAMP BPCKPOOL BPCKCYC +
             BPCKVP1 BPCKVP2 BPCKHP1 BPCKHP2 BPCKED)'
  .HELP = BPCKHLP
    VGET (BPCKSYS BPCKSAMP BPCKPOOL) ASIS
    &BPCKCYC = Ø
    &BPCKED  = ' '
  .CURSOR = BPCKED
)PROC
    VER (&BPCKSYS,NB,LIST,DB2A,DB2B,DB2C,DB2P)
    VER (&BPCKSAMP,NB,NUM)
    VER (&BPCKPOOL,NB,LIST,BPØ,BP1,BP2,BP3,BP4,BP5,BP6,BP7,BP8,BP9, +
                 BP1Ø,BP11,BP12,BP13,BP14,BP15,BP16,BP17,BP18,BP19, +
                 BP2Ø,BP21,BP22,BP23,BP24,BP25,BP26,BP27,BP28,BP29, +
                 BP3Ø,BP31,BP32,BP33,BP34,BP35,BP36,BP37,BP38,BP39, +
                 BP4Ø,BP41,BP42,BP43,BP44,BP45,BP46,BP47,BP48,BP49, +
                 BP32K,BP32K1,BP32K2,BP32K3,BP32K4,BP32K5, +
```

```
                BP32K6,BP32K7,BP32K8,BP32K9)
    VER (&BPCKCYC,NUM)
    VER (&BPCKED,LIST,Y)
    VPUT (BPCKSYS BPCKSAMP BPCKPOOL) PROFILE
)END
./  ADD NAME=BPCKEDH
)PANEL KEYLIST(ISRSPEC ISR)
)ATTR
   _ TYPE(INPUT) CAPS(OFF) INTENS(HIGH) FORMAT(&MIXED)
   # TYPE(FP)
)BODY  WIDTH(8Ø) EXPAND(//) WINDOW(76,2Ø)
%EDIT ----------------------/-/------------------+
#Command ===>_ZCMD           / / #Scroll ===>_Z   %

%         DB2    Buffer Virt. Pool Hiper Pool Min. Time
%        System   Pool    Size       Size      VP   HP  Timestamp
%       -------- ------ ---------- ---------- ---- ---- ----------------
)INIT
  .HELP = ISR2ØØØ
  .ZVARS = 'ZSCED'
  &MIXED = MIX             /* SET FORMAT MIX            */
  IF (&ZPDMIX = N)         /* IF EBCDIC MODE REQUESTED  */
    &MIXED = EBCDIC        /*  SET FORMAT EBCDIC        */
  VGET (ZSCED) PROFILE     /* Fill Scroll Vars if       */
  IF (&ZSCED = ' ')        /* Blank with page.          */
   &ZSCED = 'PAGE'         /*                           */
)proc
  VPUT (ZSCED) PROFILE
)END
./  ADD NAME=BPCKHLP
)ATTR
 ` TYPE(PT)                        /* panel title line            */
 ? TYPE(PIN)                       /* panel instruction line      */
 # TYPE(NT)                        /* normal text attribute       */
 } TYPE(ET)                        /* emphasized text attribute   */
 ! TYPE(DT)                        /* description text            */
 ¨ TYPE(NEF) PADC(_)           /* normal entry field padded with '_' */
 _ AREA(SCRL)                      /* scrollable area attribute   */
)BODY expand(//) cmd() window(7Ø,2Ø)
`------/-/----- BPCK Help ---/-/-----
#
_pnarea                                                          _
_                                                                _
_                                                                _
_                                                                -
_                                                                -
_                                                                -
_                                                                -
_                                                                -
_                                                                -
```

```
_                                                                      -
_                                                                      -
_                                                                      -
_                                                                      -
_                                                                      -
_                                                                      -
_                                                                      -
)AREA pnarea
}                    ------------------------------------
                     |   Buffer Pool Residency Check    |
                     ------------------------------------
#
 This utility checks the time a page stays in the DB2 buffer
 pool. It does this by reading a dummy table from a tablespace
 defined in the desired pool and then issuing a display
 bufferpool command with the LSTATS option at regular intervals.
 The output from the command is scanned to see if the page is
 still in the virtual or hiper pool.

 The utility maintains a history file for comparison purposes
 and presents it sorted in subsystem, buffer pool, and
 descending timestamp order. The file is presented in a normal
 edit session but has a non-standard panel to enable headings to
 be displayed and is pre-sorted by an initial macro.

 This routine can be run on-line or in batch.
)PROC
)END
./  ADD NAME=BPCKJCD
)ATTR DEFAULT(%+_)
       %   TYPE(TEXT) INTENS(HIGH)
       +   TYPE(TEXT) INTENS(LOW)
       #   TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(LEFT)
       _   TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(RIGHT)
       !   TYPE(TEXT) INTENS(LOW) SKIP(ON)
)BODY EXPAND(||) WINDOW(76,6)
+
#BPCKjob1
#BPCKjob2
#BPCKjob3
#BPCKjob4
)INIT
    VGET (BPCKJOB1 BPCKJOB2 BPCKJOB3 BPCKJOB4) ASIS
    IF (&BPCKJOB1 = '')
       &BPCKJOB1 = '//JOBCARD INFORMATION'
       &BPCKJOB2 = '//*'
       &BPCKJOB3 = '//*'
       &BPCKJOB4 = '//*'
)PROC
    VPUT (BPCKJOB1 BPCKJOB2 BPCKJOB3 BPCKJOB4) PROFILE
```

```
)END
./  ADD NAME=BPCKED
)ATTR
   _ TYPE(INPUT) CAPS(OFF) INTENS(HIGH) FORMAT(&MIXED)
   # TYPE(FP)
)BODY  WIDTH(8Ø) EXPAND(//)
%EDIT ----------------------/-/-----------------+
#Command ===>_ZCMD           / / #Scroll ===>_Z   %
)INIT
  .HELP = ISR2ØØØØ
  .ZVARS = 'ZSCED'
  &MIXED = MIX             /* SET FORMAT MIX            */
  IF (&ZPDMIX = N)         /* IF EBCDIC MODE REQUESTED  */
    &MIXED = EBCDIC        /*  SET FORMAT EBCDIC        */
  VGET (ZSCED) PROFILE     /* Fill Scroll Vars if       */
  IF (&ZSCED = ' ')        /* Blank with page.          */
   &ZSCED = 'PAGE'         /*                           */
)PROC
  VPUT (ZSCED) PROFILE
)END
./  ADD NAME=BPCKBR
)ATTR
   _ TYPE(INPUT) CAPS(OFF) INTENS(HIGH) FORMAT(&MIXED)
   + TYPE(TEXT) INTENS(LOW)
)BODY  WIDTH(8Ø) EXPAND(//) WINDOW(76,2Ø)
%BROWSE ---------------------------------/-/-+
%COMMAND ===>_ZCMD                        / / %SCROLL ===>_Z   +
+
)INIT
  .ZVARS = 'ZSCBR'
  &MIXED = MIX             /* SET FORMAT MIX            */
  IF (&ZPDMIX = N)         /* IF EBCDIC MODE REQUESTED  */
    &MIXED = EBCDIC        /*  SET FORMAT EBCDIC        */
  VGET (ZSCBR) PROFILE     /* FILL SCROLL VARS IF       */
  IF (&ZSCBR = ' ')        /* BLANK WITH PAGE.          */
   &ZSCBR = 'PAGE'         /*                           */
)PROC
  VPUT (ZSCBR) PROFILE
)END
./  ADD NAME=BPCKLBDF
 /* REXX */
"ISPEXEC LIBDEF ISPPLIB DATASET ID('USERID.BPCK.SOURCE') COND"
"ISPEXEC LIBDEF ISPSLIB DATASET ID('USERID.BPCK.SOURCE') COND"
"ALTLIB ACTIVATE APPLICATION(CLIST) DATASET('USERID.BPCK.SOURCE')"
RETURN
./  ENDUP
##
```

C S Allen
DBA (UK)

Updating statistics from production to test

INTRODUCTION

If you want to see what access paths your production queries will use, you should consider updating the catalog statistics on your test system to be the same as your production system. To do this, run RUNSTATS on your production tables to get the current access method statistics. Then select the statistics from the production system catalog and use them to build the SQL UPDATE statements to update the catalog of the test system.

THE STAT SERVICE

At our installation, I have implemented a service, called STAT, which enables our users easily to generate JCL to update production statistics on our test system.

The panel STATM requires the following fields to be entered:

- Creator – the owner of table (optional)
- Name – the name of table (optional)
- Tsname – the tablespace name (optional)
- Dbname – the database name (optional)
- DB2 – the production subsystem or the test subsystem.

To query the catalog, the REXX EXEC uses two programs that were previously published in *DB2 Update*:

- SQLISPF from *An SQL interface to ISPF Dialog Manager*. This is an Assembler program to pass the query as a REXX variable and return the result in an ISPF table (*DB2 Update*, pages 14-26, Issue 5, May 1992). At our installation, SQLISPF has been renamed REXSQLP.
- DB2CAF from *DB2CAF program*. The program uses two modules: DB2CAFI to initiate the Call Attachment Facility, and

DB2CAFT to terminate the Call Attachment Facility (*DB2 Update*, pages 3-8, Issue 2, August 1991).

It is also necessary to have **select** privilege on the production system tables SYSIBM.SYSTABLES, SYSIBM.SYSTABLESPACE, SYSIBM.SYSINDEXES, SYSIBM.SYSCOLUMNS, and SYSIBM.SYSCOLDIST and **update** or **insert** privilege on all the above tables on the test system.

SQLQUERY in STAT can be redefined to meet your own requirements and the generated file can be edited before it is submitted.

When you bind applications on the test system with production statistics, access paths should be similar to what you see when the same query is bound on your production system. If the access paths from test to production are different, there are three possible causes:

- The processor models are different.
- The buffer poll sizes are different.
- There is mismatching data in SYSCOLDIST.

STAT consists of:

- STAT – REXX procedure (SYSPROC)
- STATM – main panel (ISPPLIB)
- MESSMAP – message display panel (ISPPLIB)
- STATS – ISPF skeleton for JCL (ISPSLIB).

REFERENCES

- *DB2 V3 Administration Guide* Volumes III SC26-4888.

STAT – REXX PROCEDURE

```
/* REXX - Statements to generate update statistics on a test system*/
/* trace r */
X=MSG("OFF")
ADDRESS TSO "DELETE '"SYSVAR(SYSUID)".DB2.STAT'"
"ALLOC DD(DD1) DSN('"SYSVAR(SYSUID)".DB2.STAT') SPACE(1 1) ,
 TRACK NEW UNIT(339Ø) RECFM(F,B) LRECL(8Ø) F(FI1) REUSE"
ADDRESS ISPEXEC "ADDPOP ROW(1) COLUMN(1Ø)"
```

```
ind=Ø
fun = '____Update statistics on test system____'
msg = 'Enter data'
cur='tab'
top:
ZPFCTL = 'OFF'
ADDRESS ISPEXEC 'VPUT (ZPFCTL) PROFILE'
ADDRESS ISPEXEC "DISPLAY PANEL(STATM) CURSOR("cur")"
ind=1
msg = ''
IF PF3 = EXIT THEN DO
   ADDRESS ISPEXEC REMPOP ALL
   EXIT
END
IF cre=' ' & tab=' ' & tsn=' ' & dbn=' ' then do
   msg='Enter one Catalog search field.'
   cur='tab'
   signal top
end
IF db2=' ' then do
   msg='Enter a production subsystem-id.'
   cur='db2'
   signal top
end
IF db2t=' ' then do
   msg='Enter a test subsystem-id.'
   cur='db2t'
   signal top
end
call CREATE_messg
MESSG = 'Select SYSTABLES     Information'
MESSG = time() || " " || MESSG
call Send_messg
SQLQUERY="SELECT DISTINCT CARD,NPAGES,NAME,CREATOR",
         "FROM SYSIBM.SYSTABLES",
         "WHERE NAME    LIKE '"tab"%'",
         "  AND TSNAME  LIKE '"tsn"%'",
         "  AND DBNAME  LIKE '"dbn"%'",
         "  AND CREATOR LIKE '"cre"%'",
         "  AND CARD    >= Ø",
         "ORDER BY 4, 3";
/* invoke REXXSQL */
ADDRESS LINK "REXSQLP"
/* test return condition */
IF _nrows = Ø then do
   msg='Data not found.'
   cur='tab'
   signal top
end
IF _nrows = Ø | RC <> Ø
then exit 8;
```

```
ADDRESS ISPEXEC 'TBCREATE "LTABLE",
        NAMES(card,npages,name,creator) WRITE REPLACE'
/* display row_data */
DO I=1 TO _NROWS
   INTERPRET "card    =" _VN.1"."I
   INTERPRET "npages  =" _VN.2"."I
   INTERPRET "name    =" _VN.3"."I
   INTERPRET "creator =" _VN.4"."I
   ADDRESS ISPEXEC 'TBADD "LTABLE"'
END;
ADDRESS ISPEXEC 'TBTOP "LTABLE"'
MESSG = 'Select SYSTABLESPACE Information'
MESSG = time() || " " || MESSG
call Send_messg
SQLQUERY="SELECT DISTINCT NACTIVE,S.NAME,S.CREATOR",
         "FROM SYSIBM.SYSTABLES     T,",
         "     SYSIBM.SYSTABLESPACE S",
         "WHERE TSNAME    = S.NAME",
         "  AND T.NAME    LIKE '"tab"%'",
         "  AND   TSNAME  LIKE '"tsn"%'",
         "  AND T.DBNAME  LIKE '"dbn"%'",
         "  AND T.CREATOR LIKE '"cre"%'",
         "  AND T.CARD    >= Ø",
         "ORDER BY 3,2";
/* invoke REXXSQL */
ADDRESS LINK "REXSQLP"
ADDRESS ISPEXEC 'TBCREATE "LSPACE",
        NAMES(nactive,name,creator) WRITE REPLACE'
DO I=1 TO _NROWS
   INTERPRET "nactive=" _VN.1"."I
   INTERPRET "name="    _VN.2"."I
   INTERPRET "creator=" _VN.3"."I
   ADDRESS ISPEXEC 'TBADD "LSPACE"'
END;
ADDRESS ISPEXEC 'TBTOP "LSPACE"'
MESSG = 'Select SYSINDEXES    Information'
MESSG = time() || " " || MESSG
call Send_messg
SQLQUERY="SELECT FIRSTKEYCARD,FULLKEYCARD,NLEAF,NLEVELS,",
         "       CLUSTERRATIO,I.NAME,I.CREATOR",
         "FROM SYSIBM.SYSTABLES  T,",
         "     SYSIBM.SYSINDEXES I",
         "WHERE  T.NAME     = I.TBNAME",
         "  AND   T.CREATOR = I.TBCREATOR",
         "  AND T.NAME      LIKE '"tab"%'",
         "  AND   TSNAME    LIKE '"tsn"%'",
         "  AND T.DBNAME    LIKE '"dbn"%'",
         "  AND T.CREATOR   LIKE '"cre"%'",
         "  AND   FULLKEYCARD >=Ø";
/* invoke REXXSQL */
ADDRESS LINK "REXSQLP"
```

```
ADDRESS ISPEXEC 'TBCREATE "LINDEX",
        NAMES(firstkey,fullkey,nleaf,nlevels,
              ratio,name,creator) WRITE REPLACE'
DO I=1 TO _NROWS
   INTERPRET "firstkey=" _VN.1"."I
   INTERPRET "fullkey="  _VN.2"."I
   INTERPRET "nleaf="    _VN.3"."I
   INTERPRET "nlevels="  _VN.4"."I
   INTERPRET "ratio="    _VN.5"."I
   INTERPRET "name="     _VN.6"."I
   INTERPRET "creator="  _VN.7"."I
   ADDRESS ISPEXEC 'TBADD "LINDEX"'
END;
ADDRESS ISPEXEC 'TBTOP "LINDEX"'
MESSG = 'Select SYSCOLUMNS    Information'
MESSG = time() || " " || MESSG
call Send_messg
SQLQUERY="SELECT COLCARD,HIGH2KEY,LOW2KEY,",
         "       C.TBNAME,COLNO,C.TBCREATOR",
         "FROM SYSIBM.SYSTABLES  T,",
         "     SYSIBM.SYSCOLUMNS C",
         "WHERE T.NAME    = C.TBNAME",
         "  AND T.CREATOR = C.TBCREATOR",
         "  AND T.NAME    LIKE '"tab"%'",
         "  AND   TSNAME  LIKE '"tsn"%'",
         "  AND T.DBNAME  LIKE '"dbn"%'",
         "  AND T.CREATOR LIKE '"cre"%'",
         "  AND   COLCARD >= Ø",
         "ORDER BY 6,4,5";
/* invoke REXXSQL */
ADDRESS LINK "REXSQLP"
ADDRESS ISPEXEC 'TBCREATE "LCOLUM",
        NAMES(colcard,high2key,low2key,name,colno,creator),
         WRITE REPLACE'
DO I=1 TO _NROWS
   INTERPRET "colcard="  _VN.1"."I
   INTERPRET "high2key=" _VN.2"."I
   INTERPRET "low2key="  _VN.3"."I
   INTERPRET "name="     _VN.4"."I
   INTERPRET "colno="    _VN.5"."I
   INTERPRET "creator="  _VN.6"."I
   ADDRESS ISPEXEC 'TBADD "LCOLUM"'
END;
ADDRESS ISPEXEC 'TBTOP "LCOLUM"'
MESSG = 'Select SYSCOLDIST    Information'
MESSG = time() || " " || MESSG
call Send_messg
SQLQUERY="SELECT FREQUENCY, IBMREQD, TBOWNER,",
         "       TBNAME, NAME, COLVALUE",
         "FROM SYSIBM.SYSCOLDIST",
```

```
         "WHERE TBNAME   LIKE '"tab"%'",
         "  AND TBOWNER  LIKE '"cre"%'",
         "ORDER BY 4, 5";
/* invoke REXXSQL */
ADDRESS LINK "REXSQLP"
ADDRESS ISPEXEC 'TBCREATE "LCDIST",
        NAMES(freq,ibmreqd,tbowner,tbname,name,colvalue),
         WRITE REPLACE'
DO I=1 TO _NROWS
   INTERPRET "freq="      _VN.1"."I
   INTERPRET "ibmreqd="   _VN.2"."I
   INTERPRET "tbowner="   _VN.3"."I
   INTERPRET "tbname="    _VN.4"."I
   INTERPRET "name="      _VN.5"."I
   INTERPRET "colvalue="  _VN.6"."I
   ADDRESS ISPEXEC 'TBADD "LCDIST"'
END;
ADDRESS ISPEXEC 'TBTOP "LCDIST"'
MESSG = 'Building   JCL/SQL Stream'
MESSG = time() || " " || MESSG
call Send_messg
ADDRESS ISPEXEC REMPOP ALL
Call Generate_jcl
ADDRESS ISPEXEC 'TBEND "LTABLE"'
ADDRESS ISPEXEC 'TBEND "LSPACE"'
ADDRESS ISPEXEC 'TBEND "LINDEX"'
ADDRESS ISPEXEC 'TBEND "LCOLUM"'
ADDRESS ISPEXEC 'TBEND "LCDIST"'
ADDRESS ISPEXEC  "TBCLOSE "MESSTB""
EXIT
Generate_jcl:
   X=MSG("OFF")
   date=DATE()
   time=TIME(C)
   user=userid()
   tempfile=userid()||'.DB2.STAT'
   ADDRESS TSO
   "DELETE '"tempfile"'"
   "FREE DSNAME('"tempfile"')"
   "FREE DDNAME(ISPFILE)"
   "FREE ATTRLIST(FORMFILE)"
   "ATTRIB FORMFILE BLKSIZE(800) LRECL(80) RECFM(F B) DSORG(PS)"
   "ALLOC DDNAME(ISPFILE) DSNAME('"tempfile"')",
          "NEW USING (FORMFILE) UNIT(3390) SPACE(1 1) CYLINDERS"
   ADDRESS ISPEXEC
   "FTOPEN"
   "FTINCL STATS"
   "FTCLOSE"
   ZEDSMSG = "JCL shown"
   ZEDLMSG = "JCL for Update Statistics on Test System"
```

```
   "SETMSG MSG(ISRZØØ1)"
   "EDIT DATASET('"tempfile"') MACRO(RESET)"
   X=MSG("ON")
Return
Create_messg:
  MESSG = "S"||USERID()
  ADDRESS ISPEXEC "TBCREATE "MESSTB" NAMES(MESSG) WRITE REPLACE"
  Return
Send_messg:
  ADDRESS ISPEXEC "TBADD " MESSTB
  ADDRESS ISPEXEC "CONTROL DISPLAY LOCK "
  ADDRESS ISPEXEC "ADDPOP ROW(5) COLUMN(4)"
  ADDRESS ISPEXEC "TBDISPL "MESSTB" PANEL(MESSMAP)"
  ADDRESS ISPEXEC REMPOP
  Return
```

STATS – ISPF SKELETON FOR JCL

```
)TBA 72
)CM ---------------------------------------------------------------------
)CM Skeleton to update the catalog statistics on test system         --
)CM ---------------------------------------------------------------------
//&user.X JOB (ACCT#),'UPDATE DB2 TEST',
//             NOTIFY=&user,REGION=4M,
//             CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//* ********************************************************************
//*
//* STAT      : Update statistics on test system       Date:&date
//* User      : &user
//* Production                                         Time:&time
//* Db2       : &db2
//* Creator   : &cre
//* Table     : &tab
//* Tsname    : &tsn
//* Dbname    : &dbn
//* Test Db2  : &db2t
//*
//* ********************************************************************
//JOBLIB  DD  DISP=SHR,
//            DSN=DSN31Ø.SDSNLOAD
//STEPØ1  EXEC PGM=IKJEFTØ1,DYNAMNBR=2Ø
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  *
 DSN SYSTEM(&db2t)
 RUN PROGRAM(DSNTEP2) PLAN(DSNTEP31) -
     LIB('DSN31Ø.RUNLIB.LOAD')
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSIN    DD  *
```

```
-- Update on SYSIBM.SYSTABLESPACE
)DOT "LSPACE"
 UPDATE SYSIBM.SYSTABLESPACE
        SET NACTIVE=&nactive
 WHERE NAME='&name' AND CREATOR='&creator';
)ENDDOT
 COMMIT;
-- Update on SYSIBM.SYSTABLES
)DOT "LTABLE"
 UPDATE SYSIBM.SYSTABLES
        SET CARD=&card, NPAGES=&npages
 WHERE NAME='&name' AND CREATOR='&creator';
)ENDDOT
 COMMIT;
-- Update on SYSIBM.SYSINDEXES
)DOT "LINDEX"
 UPDATE SYSIBM.SYSINDEXES
        SET FIRSTKEYCARD=&firstkey, FULLKEYCARD=&fullkey,
            NLEAF=&nleaf, NLEVELS=&nlevels, CLUSTERRATIO=&ratio
 WHERE NAME='&name' AND CREATOR='&creator';
)ENDDOT
 COMMIT;
-- Update on SYSIBM.SYSCOLUMNS
)DOT "LCOLUM"
 UPDATE SYSIBM.SYSCOLUMNS
        SET COLCARD=&colcard, HIGH2KEY='&high2key', LOW2KEY='&low2key'
 WHERE TBNAME='&name'
   AND COLNO=&colno AND TBCREATOR='&creator';
)ENDDOT
 COMMIT;
-- Delete from SYSIBM.SYSCOLDIST
)DOT "LTABLE"
 DELETE FROM SYSIBM.SYSCOLDIST
 WHERE TBNAME='&name' AND TBOWNER='&creator';
)ENDDOT
 COMMIT;
-- Insert into SYSIBM.SYSCOLDIST
)DOT "LCDIST"
 INSERT INTO SYSIBM.SYSCOLDIST
 VALUES ( &freq, CURRENT TIMESTAMP,
 '&ibmreqd', '&tbowner', '&tbname', '&name',
 '&colvalue');
)ENDDOT
 COMMIT;
```

STATM – MAIN PANEL

```
)ATTR DEFAULT(%+_)
   } TYPE(OUTPUT) COLOR(WHITE)   CAPS (OFF) HILITE(REVERSE) INTENS(HIGH)
```

```
   | TYPE(TEXT)   COLOR(TURQUOISE) HILITE(REVERSE) INTENS(HIGH)
   ( TYPE(TEXT)   COLOR(GREEN)     HILITE(REVERSE) INTENS(HIGH)
   ) TYPE(TEXT)   COLOR(GREEN)                     INTENS(HIGH)
   # TYPE(OUTPUT) COLOR(WHITE)     CAPS(OFF)       INTENS(HIGH)
   _ TYPE(INPUT)  COLOR(RED)       HILITE(USCORE)  INTENS(HIGH)
   ! TYPE(TEXT)   COLOR(WHITE)     CAPS(OFF)       INTENS(HIGH)
   @ TYPE(TEXT)   COLOR(RED)       HILITE(REVERSE) INTENS(HIGH)
)BODY WINDOW(42,18)
}FUN
+
|
| +                                    |
| )     Creator:_CRE     +             |
| )     Name   :_TAB               +   |
| )     Tsname :_TSN     +             |
| )     Dbname :_DBN     +             |
| )     Db2    :_DB2 +   Production    |
| )            :_DB2T+   Test          |
| +                                    |
|
+
(
( #MSG                                 (
( +                                    (
( !Enter:Continue               PF3:End(
(
)INIT
  IF (&ind = Ø)
       .ATTR (msg)  = 'COLOR (WHITE)'
  IF (&cre = ' ' & &tab = ' ' & &tsn = ' ' & &dbn = ' ' & &ind = 1)
       .ATTR (msg)  = 'COLOR (RED)'
  IF (&db2  = ' ')
       .ATTR (db2)  = 'HILITE(REVERSE)'
       .ATTR (msg)  = 'COLOR (RED)'
  ELSE .ATTR (db2)  = 'HILITE(USCORE)'
  IF (&db2t = ' ')
       .ATTR (db2t) = 'HILITE(REVERSE) COLOR(YELLOW)'
       .ATTR (msg)  = 'COLOR (YELLOW)'
  ELSE .ATTR (db2t) = 'HILITE(USCORE)'
       .ATTR (db2t) = 'COLOR (RED)'
)PROC
  IF (.PFKEY = PFØ3) &PF3 = EXIT
  VPUT (TAB CRE DB2 DB2T TSN DBN) PROFILE
)END
```

MESSMAP – MESSAGE DISPLAY PANEL

```
)ATTR DEFAULT(%+_)
| TYPE (TEXT)   INTENS(LOW)  COLOR(WHITE)
```

```
@ TYPE (TEXT)   INTENS(HIGH) COLOR(RED)   CAPS(OFF) HILITE(REVERSE)
| TYPE (INPUT)  INTENS(NON)  COLOR(GREEN) CAPS(ON)  JUST(LEFT)
# TYPE (OUTPUT) INTENS(LOW)  COLOR(GREEN) CAPS(OFF)
)BODY DEFAULT(%~\) WINDOW(47,1Ø)
|ZCMD +     @ Message display |AMT  |
|-------------------------------------------
)MODEL CLEAR(MESSG)
#Z                                      +
)INIT
     .ZVARS = '(MESSG)'
)REINIT
)PROC
  IF (.PFKEY = PFØ3) &PF3 = EXIT
  IF (&ZCMD=END)
     &COMMAND = CANCEL
)END
```

Bernard Zver
Database Administrator
Informatika Maribor (Slovenia)

Automated JCL generation – revisited

In the article entitled *Automatically generating JCL for DB2 utilities*, published in *DB2 Update*, Issues 60-63, October 1997 – January 1998, I forgot to add the macros needed to successfully assemble and linkedit the Assembler program SQLSPDB used in my REXX to generate DB2 utilities.

There are three macros – the BEGIN and EINDE macros are used for standard linkage conventions, and the REGEQ macro is used to prefix all register names with an R.

BEGIN

```
*
*     REQUIRED KEYWORDS:  SAVE AND BASE
*     DEPENDENCIES : GLOBAL PARAMETERS ALSO USED IN MACRO "DATE"
*
         MACRO
&NAME    BEGIN &SAVE=,&BASE=,&PARM=,&DATE=,&TIME=,&PRINT=ON
```

```
         SPACE 2
*************************************************************************
*                                                                       *
*                                                                       *
*************************************************************************
         SPACE 2
         AIF   ('&PRINT' NE 'NOGEN').A1
         PRINT &PRINT
.A1      ANOP
         GBLC  &SYDNBØ1,&SYDNBØ2,&SYDNBØ3,&SYDNBØ4
         GBLC  &SYDNBØ5,&SYDNBØ6,&SYDNBØ7
         LCLC  &B,&C,&D,&E,&F,&N,&O,&P
&B       SETC  'B'.'&SYSNDX'.'AA'
&C       SETC  'B'.'&SYSNDX'.'HW'
&D       SETC  'B'.'&SYSNDX'.'AC'
&E       SETC  'B'.'&SYSNDX'.'AB'
&F       SETC  'B'.'&SYSNDX'.'AD'
&N       SETC  'B'.'&SYSNDX'.'AE'
&O       SETC  'B'.'&SYSNDX'.'AF'
&P       SETC  'B'.'&SYSNDX'.'ED'
         AIF   ('&SAVE' EQ '').FT1
         AIF   ('&BASE' EQ '').FT2
         AIF   ('&PRINT' EQ 'NOGEN').AØ
         AIF   ('&PRINT' EQ 'ON').AØ
         PRINT &PRINT
.AØ      ANOP
&NAME    DS    ØH
         B     6Ø(15)            BRANCH AROUND CONSTANTS
MY_CSECT DC    CL1Ø'&SYSECT'     CSECT NAME
         DC    C'&SYSDATE'       ASSEMBLY DATE
         DC    C' '
         DC    C'&SYSTIME'       ASSEMBLY TIME
         DC    C' '
         DC    C' ∏ DE NEDERLANDSCHE BANK N.V.  '
         STM   14,12,12(13)
         CNOP  2,4
         BALR  &BASE(1),Ø
         USING *,&BASE(1)        1ST BASE
.*
         AIF   (N'&BASE LT 5).A
         MNOTE 4,'*** MAXIMUM NUMBER OF BASE-REGISTERS IS FOUR. ***'
.*
.A       ANOP
         AIF   ('&BASE(2)' EQ '').B
&B       EQU   *
         L     &BASE(2),*+8
         USING &B+4Ø96,&BASE(2)    2ND BASE
         B     *+8
         DC    A(&B+4Ø96)
*
```

```
         AIF   ('&BASE(3)' EQ '').B
         L     &BASE(3),*+8
         USING &B+8192,&BASE(3)    3RD BASE
         B     *+8
         DC    A(&B+8192)
*
         AIF   ('&BASE(4)' EQ '').B
         L     &BASE(4),*+8
         USING &B+12288,&BASE(4)    4TH BASE
         B     *+8
         DC    A(&B+12288)
*
.B       ANOP
         LA    14,&SAVE
         ST    14,8(13)
         ST    13,&SAVE+4
         LA    13,&SAVE
*
         AIF   ('&PARM' EQ '').C
         L     14,Ø(1)
         LH    15,Ø(14)
         CH    15,&C
         BE    &D
         BCTR  15,Ø
         EX    15,&E
         B     &D
*
&E       MVC   &PARM.(1),2(14)
&C       DC    H'Ø'
*
&D       EQU   *
.C       ANOP
         AIF   ('&DATE' EQ '' AND '&TIME' EQ '').H
.*       AIF   ('&SYDNBØ6' NE '').CNT1Ø
&SYDNBØ6 SETC  'B'.'&SYSNDX'.'FW'
.CNT1Ø   ANOP
         ST    1,&SYDNBØ6                      SAVE REG 1
         LA    1,2(Ø,Ø)                        SPECIFY MAGNITUDE
         SVC   11                              GET TIMER
         B     &F                              BRANCH AROUND CONSTANTS
*
.*       AIF   ('&SYDNBØ1' NE '').CNT2Ø
&SYDNBØ1 SETC  'B'.'&SYSNDX'.'DB'
&SYDNBØ1 DC    D'Ø'
.CNT2Ø   ANOP
.*       AIF   ('&SYDNBØ6' NE 'B&SYSNDX.FW').CNT3Ø
&SYDNBØ6 DS    F
.CNT3Ø   AIF   ('&DATE' EQ '').D
.*       AIF   ('&SYDNBØ2' NE '').CNT4Ø
&SYDNBØ2 SETC  'B'.'&SYSNDX'.'P1'
&SYDNBØ2 DC    PL2'1'
```

```
.CNT4Ø   ANOP
.*       AIF   ('&SYDNBØ3' NE '').CNT5Ø
&SYDNBØ3 SETC  'B'.'&SYSNDX'.'PØ'
&SYDNBØ3 DC    PL2'Ø'
.CNT5Ø   ANOP
.*       AIF   ('&SYDNBØ4' NE '').D
&SYDNBØ4 SETC  'B'.'&SYSNDX'.'C19'
&SYDNBØ4 DC    C'19'
.*
.D       ANOP
         AIF   ('&TIME' EQ '').E
.*       AIF   ('&SYDNBØ7' NE '').E
&SYDNBØ7 SETC  'B'.'&SYSNDX'.'ED'
&SYDNBØ7 DC    CL14' '
.*
.E       ANOP
         AIF   ('&DATE' EQ '').F
.*       AIF   ('&SYDNBØ5' NE '').CNT6Ø
&SYDNBØ5 SETC  'B'.'&SYSNDX'.'TAB'
&SYDNBØ5 DC    PL2'31',C'JAN.'
         DC    PL2'28',C'FEB.'
         DC    PL2'31',C'MRT.'
         DC    PL2'3Ø',C'APR.'
         DC    PL2'31',C'MEI '
         DC    PL2'3Ø',C'JUNI'
         DC    PL2'31',C'JULI'
         DC    PL2'31',C'AUG.'
         DC    PL2'3Ø',C'SEPT'
         DC    PL2'31',C'OKT.'
         DC    PL2'3Ø',C'NOV.'
         DC    PL2'31',C'DEC.'
*
.CNT6Ø   ANOP
&F       EQU   *                    SUPPLY DATE
         ST    1,&SYDNBØ1+4
         MVO   &SYDNBØ1,&SYDNBØ1.(6)
         MVC   &DATE+8(2),&SYDNBØ4
         UNPK  &DATE+1Ø(2),&SYDNBØ1
         OI    &DATE+11,X'FØ'
         CVB   14,&SYDNBØ1
         ST    14,&SYDNBØ1
         TM    &SYDNBØ1+3,B'ØØØØØØ11'
         BNZ   *+1Ø
         AP    &SYDNBØ5+6(2),&SYDNBØ2
         ST    1,&SYDNBØ1
         XC    &SYDNBØ1.(2),&SYDNBØ1
         ZAP   &SYDNBØ1+4(4),&SYDNBØ3
         LA    1,&SYDNBØ5
         LA    14,12
*
&N       EQU   *
```

```
         AP    &SYDNBØ1+4(4),Ø(2,1)
         CP    &SYDNBØ1.(4),&SYDNBØ1+4(4)
         BNH   &O
         LA    1,6(1)
         BCT   14,&N
         DC    X'FFFF'
*
&O       EQU   *
         SP    &SYDNBØ1+4(4),Ø(2,1)
         SP    &SYDNBØ1.(4),&SYDNBØ1+4(4)
         UNPK  &DATE.(2),&SYDNBØ1+2(2)
         OI    &DATE+1,X'FØ'
         MVC   &DATE+3(4),2(1)
*
.F       ANOP
         AIF   ('&DATE' NE '').G
&F       EQU   *                 SUPPLY TIME
.G       ANOP
         AIF   ('&TIME' EQ '').I
         XC    &SYDNBØ1,&SYDNBØ1
         ST    Ø,&SYDNBØ1+4
         MVC   &SYDNBØ1+3(4),&SYDNBØ1+4
         MVI   &SYDNBØ1+7,X'ØC'
         MVO   &SYDNBØ1.(8),&SYDNBØ1.(7)
         MVC   &SYDNBØ7.(13),=X'4Ø2Ø21214B21214B21216B2121'
         ED    &SYDNBØ7.(13),&SYDNBØ1+3
         MVC   &TIME.(11),&SYDNBØ7+2
.I       ANOP
         L     1,&SYDNBØ6
.H       ANOP
         AIF   ('&PRINT' EQ 'NOGEN').K
         AIF   ('&PRINT' EQ 'ON').K
         PRINT ON
.K       ANOP
*                                    END "BEGIN"
         MEXIT
.FT1     MNOTE 8,'*** KEYWORD SAVE MISSING. ***'
         AGO   .H
.FT2     MNOTE 8,'*** KEYWORD BASE MISSING. ***'
         AGO   .H
         MEND
```

EINDE

```
MACRO
&NAME    EINDE &SAVE=,&RC=,&RCR=,&PARMLST=
         AIF   ('&SAVE' EQ '').FT
         AIF   ('&RCR' NE '').B
&NAME    L     13,&SAVE+4
         LM    14,12,12(13)
```

```
         AIF   ('&PARMLST' EQ '').AØ
         LA    1,&PARMLST
.AØ      AIF   ('&RC' EQ '').A1
         LA    15,&RC.(Ø,Ø)
.A1      ANOP
         BR    14
*
         MEXIT
.B       ANOP
&NAME    DS    ØH
         LR    15,&RCR                              LOAD RETURNCODE
         L     13,&SAVE+4
         L     14,12(13)
         LM    Ø,12,2Ø(13)
         AIF   ('&PARMLST' EQ '').BØ
         LA    1,&PARMLST
.BØ      ANOP
         BR    14
*
         MEXIT
.FT      MNOTE 8,'KEYWORD SAVE MISSING'
         MEND
```

REGEQ

```
MACRO
         REGEQ
RØ       EQU   Ø
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
         MEND
```

Hans Valentijn
DB2 Systems Programmer
De Nederlandsche Bank NV (The Netherlands)

REXX extensions for DB2 – part 3

This month we continue the set of functions and subroutines that extend IBM REXX. These functions interface with DB2. Requests to DB2 are made under TSO using standard SQL language through the ADDRESS DB2 statement.

```
DECL772  EQU   *
         CH    R15,=H'8'       DOUBLE PRECISION FLOATING?
         BNE   DECL774         NO, B
         MVC   Ø(2,R1),=CL2'53' YES. STORE PRECISION
         LA    R1,2(,R1)       UPDATE OUTPUT POINTER
         B     DECL779
DECL774  EQU   *               UNKNOWN FLOATING. (EXTENDED PREC?)
         MVI   Ø(R1),C'?'      STORE UNKNOWN LNG
         LA    R1,1(,R1)       UPDATE OUTPUT POINTER
         B     DECL79
DECL775  EQU   *
         CLC   $IRH1,T484      IS IT PACKED DECIMAL ?
         BNE   DECL778         NO, B
         XR    R15,R15         YES, ZERO R15
         IC    R15,SQLLEN      R15:=PRECISION
         CVD   R15,$IRPACK1    CONV INTO DECIMAL
         UNPK  Ø(2,R1),$IRPACK1 CONV INTO EXTENDED
         OI    1(R1),X'FØ'     FORCE SIGN
         MVI   2(R1),C','      DELIMITER FOR PRECISION, SCALE
         MVI   3(R1),C' '      BLANK DELIMITER
         IC    R15,SQLLEN+1    R15:=SCALE
         CVD   R15,$IRPACK1    CONV INTO DECIMAL
         UNPK  4(2,R1),$IRPACK1 CONV INTO EXTENDED
         OI    5(R1),X'FØ'     FORCE SIGN
         LA    R1,6(,R1)       UPDATE OUTPUT POINTER
         B     DECL779
DECL778  EQU   *               STORE THE LENGTH WITHOUT ANY CHANGE
         CVD   R15,$IRPACK1    CONV INTO DECIMAL
         UNPK  Ø(5,R1),$IRPACK1 CONV INTO EXTENDED, 5 BYTES
         OI    4(R1),X'FØ'     FORCE SIGN
         LA    R1,5(,R1)       UPDATE OUTPUT POINTER
DECL779  EQU   *
         MVI   Ø(R1),C')'      CLOSING PARENT.
         LA    R1,1(,R1)       UPDATE OUTPUT POINTER
DECL78   EQU   *
         TM    SQLTYPE+1,X'Ø1' MAY THIS COL BE NULL ?
         BO    DECL781         YES, B
         MVC   Ø(L'NOTNULL,R1),NOTNULL NO, STORE 'NOT NULL'
         LA    R1,L'NOTNULL(,R1) UPDATE OUTPUT POINTER
```

```
DECL781  EQU   *
         LA    R15,DB2SQLS     R15->BEGINNING OF AREA
         ST    R15,$IRVALA     STORE VALUES ADDR
         SR    R1,R15          R1:=LNG OF AREA
         ST    R1,$IRVALL      STORE LNG OF VALUE TO BE DISPLAYED
* CREATE NAME OF REXX VAR TO ASSIGN: ($COLNT.)NUM_OF_COL
         LH    R1,$IRHØ        R1:=CURRENT COLUMN NUMBER
         CVD   R1,$IRPACK1     CONV INTO DECIMAL
         UNPK  $IRPACK2,$IRPACK1 CONV INTO EXTENDED
         OI    $IRPACK2+L'$IRPACK2-1,X'FØ' FORCE SIGN
         LA    R15,$IRPACK2+L'$IRPACK2-4 R15->1ST POSS. ¬Ø CHAR
         LA    R1,4            MAX NUMBER OF LOOPS SEARCHING FOR NON-Ø
DECL782  EQU   *
         CLI   Ø(R15),C'Ø'     Ø ?
         BNE   DECL785         NO, B
         LA    R15,1(,R15)     YES, NEXT BYTE
         BCT   R1,DECL782      LOOP
         LA    R1,1            BIZARRE... FORCE LNG 1 MINI
         BCTR  R15,Ø           AND COME AGAIN ON LAST Ø
DECL785  EQU   *               R15->1ST BYTE NON Ø. R1:=LNG NUMBER
         BCTR  R1,Ø            LNG -1 FOR EX
         LA    R2,$IRCOLNV     R2->...
         AH    R2,$IRCOLNT     1ST BYTE AFTER NAME OF VAR
         EX    R1,DECLMVC4     STORE SUFFIX OF REXX VAR NAME
         LH    R15,$IRCOLNT    R15:=...
         LA    R15,1(R1,R15)   LNG OF NAME
         ST    R15,$IRNAML     STORE ADDR NAME OF VAR
         LA    R15,$IRCOLNV    R15->NAME OF VAR
         ST    R15,$IRNAMA     STORE ADDR NAME OF VAR
         BAS   R14,EXCOM       ASSIGNMENT TO REXX VARIABLE
DECL79   EQU   *
         LA    R3,SQLSIZV(,R3) NEXT SQLVAR
         BCT   RØ,DECL75       LOOP FOR ALL SQLVAR
DECL8Ø   EQU   *               END OF LOOP FOR SQLVAR
* ASSIGN THE NUMBER OF COLUMNS TO REXX VAR ($COLNT).Ø
         LH    R15,SQLD        R15:=NUMBER OF COLUMNS
         CVD   R15,$IRPACK1    CONV INTO DECIMAL
         UNPK  $IRPACK2,$IRPACK1 CONV INTO EXTENDED
         OI    $IRPACK2+L'$IRPACK2-1,X'FØ' FORCE SIGN
         LA    R15,$IRPACK2+L'$IRPACK2-4 R15->VALUE
         ST    R15,$IRVALA     STORE ADDR OF VALUE
         LA    R15,4           R15:=LNG VALUE
         ST    R15,$IRVALL     STORE LNG VALUE
         LH    R15,$IRCOLNT    R15:=LNG PREFIX FOR VAR
         LA    R15,1(,R15)     +1 BECAUSE OF THE "Ø"
         ST    R15,$IRNAML     STORE LNG NAME OF VAR
         LA    RØ,$IRCOLNT+1   RØ->NAME OF VAR -1
         AR    R15,RØ          R15->1ST BYTE AFTER PREFIX
         MVI   Ø(R15),C'Ø'     NAME OF REXX VAR=($COLNT.)Ø
         LA    R15,$IRCOLNV    R15->NAME OF VAR
```

```
         ST    R15,$IRNAMA     STORE ADDR NAME OF VAR
         BAS   R14,EXCOM       ASSIGN TO VARIABLE
DECL9Ø   EQU   *
         B     FINEXEC
DECLMVC  MVC   Ø(R2-R2,R1),Ø(R15)
DECLMVC2 MVC   DB2SQLS(R15-R15),Ø(R14)
DECLMVC4 MVC   Ø(R1-R1,R2),Ø(R15)
         DROP  R4
*********************************************************************
OPEN     EQU   *               OPEN CX/HX ...
         LR    R15,R6          R15:=...
         SR    R15,R7          LNG OF PARAMETER
         CH    R15,=H'2'       LONG ENOUGH FOR A CURSOR NAME ?
         BL    ERR45           NO, B ERROR
         CLI   1(R7),C'1'      CURSOR NUMBER IS NUMERIC ?
         BL    ERR45           NO, B SYNTAX ERROR
         CLI   Ø(R7),C'C'      OPEN CX ?
         BE    OPEN5           YES, B
         CLI   Ø(R7),C'H'      OPEN HX ?
         BNE   ERR45           NO, B SYNTAX ERROR
         CLI   1(R7),NBCUHMAX  CURSOR NUMBER TOO HIGH ?
         BH    ERR45           YES, B SYNTAX ERROR
         LA    RØ,NBCURMAB     NUMBER TO BE ADDED TO CURSOR NUMBER
         B     OPEN6
OPEN5    EQU   *
         CLI   1(R7),NBCURMAX  CURSOR NUMBER TOO HIGH?
         BH    ERR45           YES, B SYNTAX ERROR
         XR    RØ,RØ           NUMBER TO BE ADDED TO CURSOR NUMBER
OPEN6    EQU   *
         IC    R2,1(,R7)       R2:=CURSOR NUMBER, EXTENDED
         N     R2,FØ           R2:=CURSOR NUMBER, BINARY
         BCTR  R2,Ø            CURSOR NUMBER / Ø
         AR    R2,RØ           ...INCLUDING "C" CURSORS
         SLL   R2,2            * 4
         B     *+4(R2)         B ACCORDING TO CURSOR NUMBER
         B     OPENC1          B IF CURSOR C1
         B     OPENC2          B IF CURSOR C2
         B     OPENH1          B IF CURSOR H1
         B     OPENH2          B IF CURSOR H2
OPENC1   EQU   *
         EXEC SQL OPEN C1
         B     OPEN8Ø
OPENC2   EQU   *
         EXEC SQL OPEN C2
         B     OPEN8Ø
OPENH1   EQU   *
         EXEC SQL OPEN H1
         B     OPEN8Ø
OPENH2   EQU   *
         EXEC SQL OPEN H2
```

```
*         B     OPEN8Ø
OPEN8Ø    EQU   *
          LTR   R15,R15          RETURN CODE OK ?
          BNZ   ERR49            NO, B ERROR
          B     FINEXEC
*************************************************************************
CLOSE     EQU   *                CLOSE CX/HX
          LR    R15,R6           R15:=...
          SR    R15,R7           LNG OF PARAM
          CH    R15,=H'2'        LONG ENOUGH FOR A CURSOR NAME ?
          BL    ERR45            NO, B ERROR
          CLI   1(R7),C'1'       CURSOR NUMBER NUMERIC ?
          BL    ERR45            NO, B SYNTAX ERROR
          CLI   Ø(R7),C'C'       CLOSE CX ?
          BE    CLOSE5           YES, B
          CLI   Ø(R7),C'H'       CLOSE HX ?
          BNE   ERR45            NO, B SYNTAX ERROR
          CLI   1(R7),NBCUHMAX   CURSOR NUMBER TOO HIGH?
          BH    ERR45            YES, B SYNTAX ERROR
          LA    RØ,NBCURMAB      NUMBER TO BE ADDED TO CURSOR NUMBER
          B     CLOSE6
CLOSE5    EQU   *
          CLI   1(R7),NBCURMAX   CURSOR NUMBER TOO HIGH ?
          BH    ERR45            YES, B SYNTAX ERROR
          XR    RØ,RØ            NUMBER TO BE ADDED TO CURSOR NUMBER
CLOSE6    EQU   *
          IC    R2,1(,R7)        R2:=CURSOR NUMBER, EXTENDED
          N     R2,FØ            R2:=CURSOR NUMBER, BINARY
          BCTR  R2,Ø             CURSOR NUMBER / Ø
          AR    R2,RØ            ... INCLUDING THE "C" CURSORS
          SLL   R2,2             * 4
          B     *+4(R2)          B ACCORDING TO CURSOR
          B     CLOSEC1          B IF CURSOR C1
          B     CLOSEC2          B IF CURSOR C2
          B     CLOSEH1          B IF CURSOR H1
          B     CLOSEH2          B IF CURSOR H2
CLOSEC1   EQU   *
          EXEC SQL CLOSE C1
          B     CLOSE8Ø
CLOSEC2   EQU   *
          EXEC SQL CLOSE C2
          B     CLOSE8Ø
CLOSEH1   EQU   *
          EXEC SQL CLOSE H1
          B     CLOSE8Ø
CLOSEH2   EQU   *
          EXEC SQL CLOSE H2
          B     CLOSE8Ø
CLOSE8Ø   EQU   *
          LTR   R15,R15          RETURN CODE OK ?
```

```
         BNZ   ERR49           NO, B ERROR
         B     FINEXEC
*********************************************************************
FETCH    EQU   *               FETCH CX/HX
         LR    R15,R6          R15:=...
         SR    R15,R7          LNG OF PARAMETER
         CH    R15,=H'2'       LONG ENOUGH FOR NAME OF CURSOR ?
         BL    ERR45           NO, B ERROR
         CLI   1(R7),C'1'      CURSOR NUMBER IS NUMERIC ?
         BL    ERR45           NO, B SYNTAX ERROR
         CLI   Ø(R7),C'C'      FETCH CX ?
         BE    FETCH5          YES, B
         CLI   Ø(R7),C'H'      FETCH HX ?
         BNE   ERR45           NO, B SYNTAX ERROR
         CLI   1(R7),NBCUHMAX  NUMBER FOR "H" CURSOR TOO HIGH ?
         BH    ERR45           YES? B SYNTAX ERROR
         LA    RØ,NBCURMAB     NUMBER TO BE ADDED TO CURSOR NUMBER
         B     FETCH8
FETCH5   EQU   *
         CLI   1(R7),NBCURMAX  NUMBER FOR "C" CURSOR TOO HIGH?
         BH    ERR45           YES, B SYNTAX ERROR
         XR    RØ,RØ           NUMBER TO BE ADDED TO CURSOR NUMBER
FETCH8   EQU   *
         IC    R4,1(,R7)       R4:=CURSOR NUMBER, EXTENDED
         N     R4,FØ           R4:=CURSOR NUMBER IN BINARY
         BCTR  R4,Ø            CURSOR NUMBER / Ø
         AR    R4,RØ           CURSOR NUMBER / Ø / ALL CURSORS
         STH   R4,$IRH1        SAVE CURSOR NUMBER
         MH    R4,=AL2($IRSQLDL) * NB BYTES IN 1 ENTRY ADDR/LNG
         LA    R15,$IRSQLP1    R15->1ST ENTRY
         AR    R4,R15          R2->OUR ENTRY
         USING $IRSQLAD,R4
         L     R5,$IRSQLAØ     R5->SQLDA ADDR FOR THIS CURSOR
         DROP  R4
         LTR   R5,R5           SQLDA ALREADY GETMAINED?
         BZ    ERR47           NO, B FETCH WITHOUT DECLARE
         LH    R2,$IRH1        R2:=CURSOR NUMBER, BINARY / Ø
         SLL   R2,2            CURSOR NUMBER * 4
         B     *+4(R2)         B DEPENDING ON CURSOR NUMBER
         B     FETCHC1         B IF CURSOR IS C1
         B     FETCHC2         B IF CURSOR IS C2
         B     FETCHH1         B IF CURSOR IS H1
         B     FETCHH2         B IF CURSOR IS H2
FETCHC1  EQU   *
         EXEC SQL FETCH C1 USING DESCRIPTOR :SQLDA
         B     FETCH2Ø
FETCHC2  EQU   *
         EXEC SQL FETCH C2 USING DESCRIPTOR :SQLDA
         B     FETCH2Ø
FETCHH1  EQU   *
```

```
         EXEC SQL FETCH H1 USING DESCRIPTOR :SQLDA
         B     FETCH2Ø
FETCHH2  EQU   *
         EXEC SQL FETCH H2 USING DESCRIPTOR :SQLDA
*        B     FETCH2Ø
FETCH2Ø  EQU   *
         LTR   R15,R15         RETURN CODE IS OK?
         BNZ   ERR49           NO, B ERROR
         L     R15,SQLCODE     R15:=SQLCODE
         LTR   R15,R15         SQLCODE OK ?
         BNZ   FINEXEC         NO, B
*                              ASSIGNMENT TO REXX VARIABLES
         LH    R4,SQLD         R4:=NUMBER OF COLUMNS
         LTR   R4,R4           Ø VAR TO BE ASSIGNED?
         BNP   FETCH8Ø         YES, NOTHING TO DO
         LA    R3,SQLVAR       R3->1ST SQLVAR
         XR    R15,R15         ZERO R15 FOR ...
         STH   R15,$IRHØ       ZERO CURRENT COL NUMBER
FETCH3Ø  EQU   *
         LH    R15,$IRHØ       R15:=CURRENT COL NUMBER
         LA    R15,1(,R15)     +1
         STH   R15,$IRHØ       STORE IT
         MVC   $IRH1,SQLTYPE   STORE ENTITY TYPE
         NI    $IRH1+1,X'FE'   RESET BIT "MAY BE NULL"
         TM    $IRDB2DR,$IRDB2FO MUST WE FORCE THE NAME OR REXX VAR?
         BO    FETCH31         YES, B
         LH    R15,SQLNAME     NO, R15:=LNG OF NAME
         LTR   R15,R15         DOES THIS COL HAVE A NAME ?
         BNP   FETCH31         NO, B
         ST    R15,$IRNAML     YES, STORE LNG OF NAME
         LA    R15,SQLNAME+2   R15->NAME OF VARIABLE
         ST    R15,$IRNAMA     STORE ADDR OF NAME FOR EXCOM
         B     FETCH32
FETCH31  EQU   *               COLUMN W/O NAME OR WE MUST FORCE IT
         LH    R15,$IRCOL      R15:=LNG OF NAME TO BE GIVEN
         LTR   R15,R15         A NAME WAS PROVIDED?
         BNP   FETCH78         NO, NOTHING TO DO WITH THIS COL
         LH    R15,$IRHØ       R15:=CURRENT COL NUMBER
         CVD   R15,$IRPACK1    CONV TO DECIMAL
         UNPK  $IRPACK2,$IRPACK1 CONV TO EXTENDED
         OI    $IRPACK2+L'$IRPACK2-1,X'FØ' FORCE SIGN
         LA    R15,$IRPACK2+L'$IRPACK2-4 R15->1ST POSSIBLE NON Ø
         LA    R1,4            MAX NB OF LOOPS SEARCHING FOR NON Ø
FETCH315 EQU   *
         CLI   Ø(R15),C'Ø'     Ø ?
         BNE   FETCH316        NO, NON-Ø FOUND
         LA    R15,1(,R15)     YES, NEXT BYTE
         BCT   R1,FETCH315     LOOP
         LA    R1,1            BIZARRE... WE FORCE LNG=1
         BCTR  R15,Ø           AND WE COME BACK ON LAST Ø
```

```
FETCH316 EQU   *
         BCTR  R1,Ø            LNG -1 FOR EX
         LA    R2,$IRCOLV      R2->...
         AH    R2,$IRCOL       1ST BYTE AFTER NAME
         EX    R1,FETCHMV4     STORE SUFFIX INTO $IRCOL
         LH    R15,$IRCOL      R15:=...
         LA    R15,1(R1,R15)   LENGTH OF WHOLE NAME
         ST    R15,$IRNAML     STORE LNG OF NAME
         LA    R15,$IRCOLV     R15->NAME OF REXX VARIABLE
         ST    R15,$IRNAMA     STORE ADDR OF NAME FOR EXCOM
FETCH32  EQU   *               NULL VALUE (X'FFFF') OR NOT (X'ØØØØ') ?
         TM    SQLTYPE+1,X'Ø1' MAY THIS COL BE NULL?
         BNO   FETCH35         NO, B
         L     R15,SQLIND      YES. R15->VALUE (NULL OR NOT)
         LH    R15,Ø(,R15)     R15:=CODE FOR NULL(-1) OR NOT(Ø)
         LTR   R15,R15         NULL ?
         BZ    FETCH35         NO, B
         XR    R15,R15         YES. ZERO LNG
         B     FETCH45
FETCH35  EQU   *
         LH    R15,SQLLEN      R15:=LNG OF VALUE
         CLC   $IRH1,T484      PACKED DECIMAL ?
         BNE   FETCH36         NO, B
         XR    R15,R15         YES. R15:=...
         IC    R15,SQLLEN      PRECISION
         LA    R15,2(,R15)     +2 FOR SIGN
         SRA   R15,1           /2 : R15:=LNG IN BYTES
         B     FETCH39
FETCH36  EQU   *
         CLC   $IRH1,T448      TYPE VARYING ?
         BE    FETCH38         YES, B
         CLC   $IRH1,T456      TYPE VARYING ?
         BE    FETCH38         YES, B
         CLC   $IRH1,T464      TYPE VARYING ?
         BE    FETCH38         YES, B
         CLC   $IRH1,T472      TYPE VARYING ?
         BNE   FETCH39         NO, B
FETCH38  EQU   *               TYPE VARYING: LNG IS BEHIND VALUE
*                              (SQLLEN IS ONLY MAX LNG)
         L     R15,SQLDATA     R15->LNG + VALUE
         LH    R14,Ø(,R15)     R14:=LNG
         ST    R14,$IRVALL     STORE LNG OF VALUE
         LA    R15,2(,R15)     R15->VALUE
         ST    R15,$IRVALA     STORE ADDR OF VALUE
         B     FETCH5Ø
FETCH39  EQU   *               NOT VARYING
FETCH45  EQU   *
         ST    R15,$IRVALL     STORE LNG OF VALUE
         L     R15,SQLDATA     R15->VALUE
         ST    R15,$IRVALA     STORE ADDR OF VALUE
```

```
FETCH5Ø  EQU   *
         L     R15,$IRVALL     R15:=LNG OF DATA
         LTR   R15,R15         LNG=Ø ?
         BNP   FETCH68         YES, DON'T TRANSLATE, EVEN IF REQUESTED
         TM    $IRDB2DR,$IRDB2NC RAW DATA REQUESTED?
         BO    FETCH68         YES, DON'T TRANSLATE
         CLC   $IRH1,T5ØØ      WE MUST TRANSLATE. SMALLINT ?
         BNE   FETCH54         NO, B
*                              TRANSLATION OF A SMALLINT
         L     R15,SQLDATA     R15->BINARY VALUE
         LH    R15,Ø(,R15)     R15:=BINARY VALUE
         CVD   R15,$IRPACK1    CONV TO DECIMAL
         UNPK  $IRPACK2,$IRPACK1 CONV TO EXTENDED
         LA    R2,$IRPACK2     R2->NUMBER TO BE NORMALIZED
         LA    R1,L'$IRPACK2   R1:=LNG OF NUMBER OT BE NORMALIZED
         B     FETCH55
FETCH54  EQU   *
         CLC   $IRH1,T496      INTEGER ?
         BNE   FETCH56         NO, B
*                              TRANSLATION OF AN INTEGER
         L     R15,SQLDATA     R15->BINARY VALUE
         L     R15,Ø(,R15)     R15:=BINARY VALUE
         CVD   R15,$IRPACK3    CONV TO DECIMAL
*        UNPK DOES NOT WORK FOR A NUMBER CONTAINING MORE THAN 8 DIGITS:
*        UNPK  $IRPACK2,$IRPACK3 THIS INSTRUCTION REPLACED BY:
         LA    R2,$IRPACK3     R2->DECIMAL NUMBER
         LA    R1,L'$IRPACK3   R1:=LNG OF NUMBER
         L     R15,AP2D        R15->PGM FOR DECIMAL TO EXTENDED CONV.
         BASR  R14,R15         CONVERT THE NUMBER INTO EXTENDED
FETCH55  EQU   *               HERE, R2->EXTENDED NUMBER, R1=LNG
* NORMALIZATION OF AN EXTENDED NUMBER W/ SIGN OVERPUNCHED ON LAST BYTE
         L     R15,ANORM       R15->PGM FOR NORMALIZATION
         BASR  R14,R15         NORMALIZATION. R2->RESULT, R1=LNG
         LR    R15,R2          R15->RESULT
         B     FETCH6Ø
FETCH56  EQU   *
         CLC   $IRH1,T484      PACKED DECIMAL ?
         BNE   FETCH58         NO, B
*                              CONVERSION OF A PACKED DECIMAL
         XR    R1,R1           R1:=...
         IC    R1,SQLLEN       LNG OF DECIMAL NUMBER
         LA    R1,2(,R1)       +2 FOR SIGN
         SRA   R1,1            /2: LNG IN BYTES
         L     R2,SQLDATA      R2->DECIMAL NUMBER
         L     R15,AP2D        R15->PGM FOR CONV TO EXTENDED
         BASR  R14,R15         CONV TO EXTENDED (INTO $IRET31 NOT NORM)
         LA    R15,L'$IRET31   R15:=LNG OF NON-NORMALIZED NUMBER
         XR    R1,R1           R1:=...
         IC    R1,SQLLEN+1     NUMBER OF DIGITS AFTER DECIMAL POINT
         SR    R15,R1          R15:=NB OF DIGIT BEFORE DECIMAL POINT
```

```
         BCTR  R15,Ø           -1 FOR EX
         EX    R15,FETCHM1Ø    STORE INTO $IRZ32
         LA    RØ,1            INDICATE "NO DECIMAL POINT"
         LTR   R1,R1           DIGIT AFTER THE POINT ?
         BZ    FETCH564        NO, DONE. LET'S NORMALIZE
         LA    R2,$IRZ32+1     R2->...
         AR    R2,R15          1ST BYTE AFTER INTEGER PART
         MVI   Ø(R2),C'.'      WHERE WE STORE THE DECIMAL POINT
         LA    RØ,$IRET31+1    RØ->AREA TO BE NORM., +1
         AR    R15,RØ          R15->DECIMAL PART (LNG¬=Ø)
         BCTR  R1,Ø            LNG DECIMAL PART -1 FOR EX
         EX    R1,FETCHM12     STORE DECIMAL PART
         XR    RØ,RØ           INDICATE "WE HAVE A DECIMAL POINT"
FETCH564 EQU   *
         LA    R2,$IRZ32       R2->NUMBER TO BE NORMALIZED
         LA    R1,L'$IRZ32     R1:=LNG OF NUMBER
         SR    R1,RØ           MINUS 1 IF NOT DECIMAL POINT
         L     R15,ANORM       R15->PGM FOR NORMALIZATION
         BASR  R14,R15         NORMALIZE THE NUMBER
         LR    R15,R2          R15->NORMALIZED NUMBER. R1=LNG
         B     FETCH6Ø
FETCH58  EQU   *
         CLC   $IRH1,T48Ø      TYPE FLOATING ?
         BNE   FETCH68         NO, B
*                              CONVERSION OF A FLOATING POINT NUMBER
         LH    R15,SQLLEN      R15:=LNG OF FLOAT.
         CH    R15,=H'4'       SINGLE PREC FLOAT.?
         BE    FETCH581        YES, B
         CH    R15,=H'8'       DOUBLE PREC FLOAT.?
         BE    FETCH582        YES,B
         B     ERRØØ           NO. NOT FORESEEN(EXTENDED FLOAT.?)
FETCH581 EQU   *               SINGLE PREC FLOAT.
         LH    R15,CODECF4E    R15:=CODE FOR FLOAT->EXTENDED
         B     FETCH583
FETCH582 EQU   *               DOUBLE PREC FLOAT.
         LH    R15,CODECF8E    R15:=CODE FOR DOUBLE FLOAT->EXTENDED
FETCH583 EQU   *
         STH   R15,$IRMFE+12   STORE CODE FOR CONVERSION
         LA    R15,$IRMFE+12   R15->1ST ARG (CODE FOR CONVERSION)
         ST    R15,$IRMFE      STORE ADDR 1ST PARAM
         L     R14,SQLDATA     R14->VALUE
         LH    R15,SQLLEN      R15:=LNG OF FLOAT. NUMBER
         BCTR  R15,Ø           -1 FOR EX
         EX    R15,FETCHM2Ø    STORE IN WORK AREA UNDER THE LINE
         LA    R15,$IRPACK1    R15->2NE ARG (FLOAT. NUMBER)
         ST    R15,$IRMFE+4    STORE ADDR 2NDPARAM
         LA    R15,$IRZ32      R15->AREA THAT WILL RECEIVE EXTENDED NB
         ST    R15,$IRMFE+8    STORE ADDR 3RDPARAM
         OI    $IRMFE+8,X'8Ø'  INDICATE LAST PARAM
         L     R15,$IRAMCF     R15:=ADDR MODULE FOR FLOAT. CONV.
```

```
         LTR   R15,R15         MODULE ALREADY LOADED?
         BNZ   FETCH584        YES, B
         LOAD  EP=$IRXCNV2     NO. LOAD THE MODULE
         ST    RØ,$IRAMCF      STORE ADDR OF MODULE
         LR    R15,RØ          R15->MODULE
FETCH584 EQU   *
         A24   ,               SET 24 BITS ADDR MODE, FOR COBOL PGM
         LA    R1,$IRMFE       R1->PARAMETERS
         BASR  R14,R15         CALL THE MODULE FOR CONVERSION
         A31   ,               BACK TO 31 BITS ADDRESSING MODE
         LA    R15,$IRZ32      R15->EXTENDED NUMBER (22 BYTES)
         LH    RØ,SQLLEN       RØ:=LNG OF NUMBER IN DB2
         CH    RØ,=H'8'        LNG = 8 ?
         BE    FETCH588        YES, B
*                               NO. LNG=4: SIMPLE PREC FLOAT.
         MVC   $IRZ15E1,$IRZ22E WE SHIFT THE "E+99" A BIT ON THE LEFT
         LA    R1,$IRZ15L      R1:=LNG OF EXTENDED NUMBER
         B     FETCH6Ø
FETCH588 EQU   *
         LA    R1,$IRZ22L      R1:=LNG OF DOUBLE PREC FLOAT NUMBER
         B     FETCH6Ø
FETCH6Ø  EQU   *               R15->RESULT, R1=LNG OF RESULT
         ST    R15,$IRVALA     STORE ADDR OF VALUE
         ST    R1,$IRVALL      STORE LNG OF VALUE
FETCH68  EQU   *               ------END OF CONVERSIONS---------------
         BAS   R14,EXCOM       ASSIGN TO THIS VARIABLE
         LTR   R15,R15         OK ?
         BNZ   ERR48           NO, B ERROR
*        ASSIGN TO THE VARIABLE NAMED NULL_<NAME_OF_COLUMN>
         MVC   $IRMFE(L'LNULL),LNULL STORE PREFIX
         L     R15,$IRNAML     R15:=LNG OF NAME W/O PREFIX
         L     R14,$IRNAMA     R14->NAME WITHOUT PREFIX
         BCTR  R15,Ø           -1 FOR EX
         EX    R15,FETCHM3Ø    STORE NAME OF VAR AFTER NULL_
         LA    R15,L'LNULL+1(,R15) R15:=LNG OF NAME WITH PREFIX
         ST    R15,$IRNAML     STORE LNG OF NAME
         LA    R15,$IRMFE      R15->NAME
         ST    R15,$IRNAMA     STORE ADDR OF NAME
         MVI   $IRXØ,C'Ø'      FOR INIT VAR TO Ø (NOT NULL)
         TM    SQLTYPE+1,X'Ø1' MAY THIS COL BE NULL?
         BNO   FETCH75         NO, B
         L     R15,SQLIND      R15->CODE NULL(-1) OR NOT NULL(Ø)
         LH    R15,Ø(,R15)     R15:=CODE
         LTR   R15,R15         NULL ?
         BZ    FETCH75         NO, B
         MVI   $IRXØ,C'1'      YES. THIS FOR INIT VAR TO 1
FETCH75  EQU   *
         LA    R15,$IRXØ       R15->VALUE
         ST    R15,$IRVALA     STORE ADDR OF VALUE
         LA    R15,1           R15:=LNG OF VALUE
```

```
         ST    R15,$IRVALL     STORE LNG OF VALUE
         BAS   R14,EXCOM       ASSIGN TO THE VARIABLE (COL)_NULL
         LTR   R15,R15         OK?
         BNE   ERR48           NO, B
FETCH78  EQU   *
         LA    R3,SQLSIZV(,R3) NEXT SQLVAR
         BCT   R4,FETCH3Ø      LOOP FOR ALL SQLVAR
FETCH8Ø  EQU   *
         B     FINEXEC
FETCHMVC MVC   $IRMFE(R15-R15),Ø(R14)
FETCHMV4 MVC   Ø(R1-R1,R2),Ø(R15)
FETCHM1Ø MVC   $IRZ32(R15-R15),$IRET31
FETCHM12 MVC   1(R1-R1,R2),Ø(R15)
FETCHM2Ø MVC   $IRPACK1(R15-R15),Ø(R14)
FETCHM3Ø MVC   $IRMFE+L'LNULL(R15-R15),Ø(R14)
*********************************************************************
SET      EQU   *
DELETE   EQU   *
UPDATE   EQU   *
INSERT   EQU   *
ROLLBACK EQU   *
GRANT    EQU   *
REVOKE   EQU   *
ALTER    EQU   *
CREATE   EQU   *
COMMIT   EQU   *
DROP     EQU   *
LOCK     EQU   *
         BAS   R14,STOSQL      STORE SQL INTO DB2SQL
         EXEC SQL EXECUTE IMMEDIATE :DB2SQL
         LTR   R15,R15         RETURN CODE OK ?
         BNZ   ERR49           NO, B ERROR
FINEXEC  EQU   *
         XR    R15,R15         ZERO RETURN CODE
         ST    R15,$IRDB2RT    STORE RETURN CODE
         B     STOC
*********************************************************************
*STORE STRING IN WORK AREA "DB2SQL". $IRF1->BEGIN.OF STRING, R6->END+1
STOSQL   EQU   *
         STM   R14,R12,12(R13) SAVE REGISTERS
         LA    R2,DB2SQLS      R2->RECEIVING AREA
         LA    R3,L'DB2SQLS    R3:=MAX LNG
         L     R14,$IRF1       R14->SENDING AREA
         LR    R15,R6          R15:=...
         SR    R15,R14         LNG OF SENDED DATA
         STH   R15,DB2SQLL     STORE LNG
         CR    R15,R3          SENDING AREA TOO LONG?
         BH    ERR46           YES, B ERROR
         MVCL  R2,R14          MOVE IT
         LM    R14,R12,12(R13) RESTORE REGISTERS
```

```
         BR    R14             RETURN
*************************************************************************
RECHNB   EQU   *               SEARCH FOR 1ST NON BLANK
*        OUTPUT:R7->1ST NON BLANK, OR R7=Ø IF ONLY BLANKS
RECHNB1Ø EQU   *
         CR    R7,R6           BEYOND END?
         BNL   RECHNB2Ø        YES, B
         CLI   Ø(R7),C' '      BLANK ?
         BNER  R14             NO, DONE
         LA    R7,1(,R7)       YES. NEXT BYTE
         B     RECHNB1Ø        LOOP
RECHNB2Ø EQU   *
         XR    R7,R7           NOTHING FOUND: ZERO R7
         BR    R14
*************************************************************************
ERRØØ    LA    R2,ERØØØ        THIS FUNCTION NOT IMPLEMENTED YET
         B     ERR
ERR45    LA    R2,ERØØ1        SYNTAX ERROR
         B     ERR
ERR46    LA    R2,ERØØ2        STRING TOO LONG FOR ME
         B     ERR
ERR47    LA    R2,ERØØ3        FETCH WITHOUT PREVIOUS DECLARE
         B     ERR
ERR48    LA    R2,ERØØ4        ERROR ASSIGNING TO A VARIABLE
         B     ERR
ERR49    LA    R2,ERØØ5        BAD RETURN CODE FROM EXEC SQL
         B     ERR
ERR54    LA    R2,ERØØ6        $DB2INST FUNCTION NOT PREVIOUSLY CALLED
         B     ERR
ERR      EQU   *
         LINK  EP=$IRXMSG      DISPLAY THE ERROR MESSAGE
         LA    R15,16          RETURN CODE:=16
         ST    R15,$IRDB2RT    STORE INTO WORK AREA
         B     FIN
*************************************************************************
STOCREAS EQU   * $IRDB2RT=RET CODE TO BE STUFFED INTO 4TH PARM AND R15
*CONVERT $IRDB2RE INTO $IRPACK1, TO "EXTENDED HEXADECIMAL"
         UNPK  $IRPACK1(9),$IRDB2RE(5) UNPK 4 USEFULL CHARACTERS
         TR    $IRPACK1(8),TABHEXE-C'Ø' CONVERSION TO "HEXADECIMAL"
*        STORE $IRDB2RE INTO REXX VARIABLE "REASON"
         LA    R15,$IRPACK1    R15->VALUE
         ST    R15,$IRVALA     STORE ADDR OF VALUE
         LA    R15,8           R15:=LNG OF VALUE (8 BYTES)
         ST    R15,$IRVALL     STORE LNG
         LA    R15,NREASON     R15->NAME OF VARIABLE
         ST    R15,$IRNAMA     STORE IT
         LA    R15,L'NREASON   R15:=LNG OF NAME
         ST    R15,$IRNAML     STORE IT
         BAS   R14,EXCOM       ASSIGN TO THE VARIABLE
         LTR   R15,R15         OK?
```

```
         BNZ   ERR48           NO, B
         B     FIN
*********************************************************************
STOC     EQU   * $IRDB2RT=RET CODE TO BE STUFFED INTO 4TH PARM AND R15
*        CONVERT SQLCODE TO EXTENDED
         L     R15,SQLCODE     R15:=BINARY SQLCODE
         CVD   R15,$IRPACK1    CONV TO DECIMAL
         UNPK  $IRPACK2,$IRPACK1 CONV TO EXTENDED
         LA    R2,$IRPACK2     R2->NUMBER TO BE NORMALIZED
         LA    R1,L'$IRPACK2   R1:=LNG OF NUMBER
         L     R15,ANORM       R15->PGM FOR NORMALIZATION
         BASR  R14,R15         NORMALIZATION
*        ASSIGN SQLCODE TO THE VARIABLE "SQLCODE"
         ST    R2,$IRVALA      STORE ADDR OF NORM. EXTENDED NUMBER
         ST    R1,$IRVALL      STORE LNG
         LA    R15,NSQLCODE    R15->NAME OF VARIABLE
         ST    R15,$IRNAMA     STORE IT
         LA    R15,L'NSQLCODE  R15:=LNG OF NAME
         ST    R15,$IRNAML     STORE IT
         BAS   R14,EXCOM       ASSIGN TO THE VARIABLE
         LTR   R15,R15         OK?
         BNZ   ERR48           NO, B
*        ASSIGN SQLSTATE TO THE VARIABLE "SQLSTATE"
         LA    R15,SQLSTATE    R15->VALUE OF SQLSTATE
         ST    R15,$IRVALA     STORE ADDRESS
         LA    R15,5           LNG OF VALUE=5
         ST    R15,$IRVALL     STORE LNG
         LA    R15,NSQLSTAT    R15->NAME OF VARIABLE
         ST    R15,$IRNAMA     STORE IT
         LA    R15,L'NSQLSTAT  R15:=LNG OF NAME
         ST    R15,$IRNAML     STORE IT
         BAS   R14,EXCOM       ASSIGN TO THE VARIABLE
         LTR   R15,R15         OK?
         BNZ   ERR48           NO, B
FIN      EQU   *               $IRDB2RT MUST CONTAIN RETURN CODE
         L     R15,$IRDB2RT    R15:=RETURN CODE
         L     R2,4(,R13)      R2->PREVIOUS SAVE AREA
*        STORE RETURN CODE IN PARAMETERS
         L     R1,24(,R2)      R1->LIST AT ENTRY
         L     R1,16(,R1)      R1->4TH PARAM (RETURN CODE)
         ST    R15,Ø(,R1)      STORE RETURN CODE
         LR    R3,R15          SAVE RETURN CODE
         FREEMAIN RU,LV=SAUVL,A=(R13) FREE OUR WORK AREA
         LR    R13,R2
         LR    R15,R3          RESTORE RETURN CODE
         L     R14,12(,R13)    RESTORE R14
         LM    RØ,R12,2Ø(R13)  RESTORE REGS BUT R15
         BR    R14             RETURN
*********************************************************************
* EXCOM: CALL IBM IRXEXCOM FOR ASSIGNMENT
```

```
* AT ENTRY, PARAMETERS FOR IRXEXCOM MUST HAVE BEEN SET UP:
*             $IRNAMA->NAME OF VARIABLE, $IRNAML=LNG OF NAME
*             $IRVALA->VALUE,            $IRVALL=LNG OF VALUE
EXCOM    EQU   *
         STM   R14,R12,12(R13)
         LR    R2,R13
         LA    R13,72(,R13)    NEXT SAVE AREA
         ST    R2,4(,R13)
         ST    R13,8(,R2)
         L     R15,$IRXEXCA    R15->IRXEXCOM
         LTR   R15,R15         MODULE ALREADY LOADED?
         BNZ   EXCOM1Ø         YES, B
         LOAD  EP=IRXEXCOM     NO, LOAD THE MODULE
         ST    RØ,$IRXEXCA     AND STORE ITS ADDRESS
EXCOM1Ø  EQU   *
         LA    R1,$IREXCOM     R1->PARAMETERS
         LA    R2,$IRCOMNM     R2->NAME
         L     R15,$IRXEXCA    R15->IRXEXCOM
         L     RØ,$IRXENVB     RØ->ENV BLOCK
         BASR  R14,R15         CALL IRXEXCOM
         L     R13,4(,R13)
         L     R14,12(,R13)
         LM    RØ,R12,2Ø(R13)
         BR    R14
```

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

Patrick Leloup
System Engineer
Credit Agricole de Loire Atlantique (France)

Accessing directory information – help wanted

Issue 13 of *DB2 Update*, November 1993, contained an article by Alberto Grassi called *Accessing directory information.* J Naumovic has written to us asking why the technique, which worked perfectly with DB2 Version 4, does not work with DB2 Version 5. Does anyone know the answer? Is there a fix for the problem? Please write to Trevor Eddolls, the editor of *DB2 Update*, at any of the addresses shown on page 2 if you can help.

DB2 news

IBM has announced new licensing for DB2 which removes the requirement for DB2 user entitlement tracking for Lotus Domino and Notes applications accessing local data on certain DB2 servers.

Users of DB2 Universal Database Enterprise Edition and Extended Edition servers will no longer need to count DB2 users for these types of application. This, claims IBM, should result in cost saving on DB2 purchases for Domino and Notes environments.

For DB2 Workgroup Edition users, the company is supplying a single-server authorization for unlimited access to local DB2 data from Domino server and Lotus Notes applications.

Meanwhile, the company said that it intends to provide a DB2 LotusScript Extension, or LSX, that supports Domino 4.6 and Notes 4.5.3 clients on NT and Windows 95, but no dates were given. It already has a DB2 LSX plug-in in the Domino.Connect product for Domino servers and Notes clients.

For further information contact your local IBM representative.

* * *

Haht Software and Neon Systems plan to integrate Hahtsite e-business tools and Neon's Shadow Direct, which accesses mainframe data and business logic.

Hahtsite's Application Server can be used now to integrate mainframe data so that, for example, SAP R/3, client/server databases, and Lotus Notes can be combined with data coming from DB2 (as well as IMS, CICS, VSAM, and MQSeries) to create new applications.

Shadow Direct integrates with the Hahtsite Integrated Development Environment, so developers can use Hahtsite's wizards and data-aware components to develop Web applications. Neon's products work with Hahtsite's form wizards, report wizards, and master/detail wizards.

For further information contact:
Haht Software, 4200 Six Forks Road, Raleigh, NC 27609, USA.
Tel: (919) 786 5100.

Neon Systems, 14141 Southwest Freeway, Suite 6200, Sugar Land, TX 77478, USA.
Tel: (281) 491 4200.

* * *

Xephon is holding its *DB2 Update '98* conference at the Mountbatten Hotel in London on 11-12 June 1998. *DB2 Update '98* is an update and analysis of key developments in the DB2 environment, including Universal Database. Delegates attending *DB2 Update '98* will gain a clearer understanding of the evolution of DB2 both within the mainframe environment and across multiple platforms, and be better able to evaluate and exploit future DB2 developments. Sessions at the conference cover both DB2-specific issues, and issues relating to the exploitation of database technologies in the enterprise environment.

The preferential attendance fee for subscribers is £540.00 plus £63.25 VAT. For further information about *DB2 Update '98* contact Xephon on 01635 33823.